FUJITSU

FUJITSU Software

富士通

SSL II 使用手引書

（科学用サブルーチンライブラリ）

J2UL-1903-01Z0(00)
2014年10月

まえがき

本マニュアルは，科学用サブルーチンライブラリ SSL II (Scientific Subroutine Library II) の機能と使用方法について記述しています．

SSL II は，パーソナルコンピュータからベクトル計算機にわたる種々のシステムで利用できます．利用者プログラムと SSL II のインタフェースは，システムが異なっても全て同一になっています．したがって，本マニュアルは，SSL II が提供される各種システムで共通に利用されることを意図して記述しています．

本マニュアルは，いくつかの章から成り，SSL II を初めてお使いになる方は，あらかじめ“本マニュアルの読み方”をお読みになってからお使い下さい．

なお，SSL II の開発にあたっては，下記に示した関係各所の深大な御協力，御指導をいただきました．ここに深く感謝致します．

北海道大学大型計算機センタ殿
名古屋大学大型計算機センタ殿
京都大学大型計算機センタ殿
九州大学大型計算機センタ殿
日本原子力研究所核設計研究室殿
（順不同）

SSL II は，最新の技術を維持するために，改良並びに新規追加がなされることがあります．また，改良若しくは新規追加したサブルーチンが，既存サブルーチンの機能を包含し，性能的にまさる場合には，一定の猶予期間をおいて，既存サブルーチンを削除することがありますので，あらかじめご承知おきください．

（ご注意）

特定のシステムでは，ハードウェア上の制約のために SSL II の一部の機能が制限される場合があります．本マニュアルでは，該当機能を “SSL II 一覧表”に明記しています．

輸出管理規制について

本ドキュメントを輸出または第三者へ提供する場合は，お客様が居住する国および米国輸出管理関連法規等の規制をご確認のうえ，必要な手続きをおとりください．

出版年月および版数

版数	マニュアルコード
2014 年 10 月 第 2 版	J2UL-1903-01Z0(00)
2011 年 3 月 初版 改訂 3	—
2008 年 2 月 初版 改訂 2	—
2007 年 5 月 初版 改訂	—
1987 年 12 月 初版	—

変更履歴

変更内容	変更箇所	版数
体裁の見直し	表紙, まえがき	第 2 版

SSL II の増補拡充に当たって

SSL II は，機能・性能・使い易さ・信頼性・移行性などの点で，SSL を改善拡充することを目標に開発を進めてきましたが，幸い各方面から，高い評価を得るようになりました．これはひとえに，関係各位の暖かい御理解と御支援の賜物と深く感謝いたします．

SSL II のような科学用サブルーチンライブラリは，専門家にとっては，新しい研究成果を正確に広く伝達する強力な手段であり，利用者にとっては，専門家の深い知識と経験に基づく成果を詳細に知らなくても，簡単に利用できる利点をもっています．そのため，欧米では，専門家の成果を集大成して，流通の労をとる組織が発達しています．一方，我が国では，大学・研究所などで開発収集が進められていますが，その成果の流通は，今一つ円滑さを欠いているように思われます．わが国でも早急に欧米のような方式を取り入れ，専門家の成果を各位の利用の便に供する必要があると考えます．

そこで SSL II は，我が国の実情に合うハンドブックとして，著作を広く全国の専門家にお願いし，富士通は自らも著作を行いつつ，編集出版の役割を果すのが良いと考え，これを識者の方々に披露したところ，多くの御賛同を得ました．

ここに今日までに著作頂いた各位の御氏名を記し，感謝の意を表明します．また，新しい試みに参画されたことに対し，深く敬意を表明します．利用者各位におかれましても，著作者の苦心を評価して頂き，論文・報告書などの執筆に当たっては，可能な限り，SSL II 利用の旨を記載頂ければ幸いです．

富士通は，今後も広く関係各位の御尽力を仰ぎ，ソフトウェアの流通が円滑に行われ，計算機科学が一層発展することを期待して，課せられた任務を積極的に果たす所存です．関係各位の御協力を賜わりますようお願い致します．

1980 年 12 月
富士通株式会社

著作者氏名一覧表

氏　名	所　属
田中　正次	山梨大学工学部
二宮　市三	中部大学経営情報学部
鳥居　達生	名古屋大学工学部
長谷川　武光	福井大学工学部
秦野　和郎	愛知工業大学工学部
秦野　甯世	中京大学教養部
吉田　年雄	中部大学経営情報学部
刀根　薫	埼玉大学大学院政策科学研究所
古林　隆	埼玉大学大学院政策科学研究所
細野　敏夫	日本大学理工学部

(順不同，敬称略，1987 年 11 月現在)

目　次

第 II 部 サブルーチン使用方法

SSL II 一覧表

以下に SSL II の各機能項目を示す．各機能に対応して原則的に単精度演算用と倍精度演算用の 2 種類のルーチンを利用できる．“サブルーチン名”には単精度演算用のルーチン名を示している．倍精度演算用のルーチン名は，先頭に“D”を付加すればよい．

ハードウェア上の制約等のために機能が制限される場合は，“備考”欄にその旨を示している．“備考”欄の記号の意味は以下の通りである．

*： 通常，単精度演算用と倍精度演算用の両ルーチンを利用できるが，以下の様な条件のシステムでは，単精度演算用ルーチンしか利用できない．
条件：FORTRAN システムが 4 倍精度演算機能をサポートしていない場合．
例 ： SX/G100 シリーズ SSL II
FM シリーズ SSL II

#： 全てのシステムで単精度演算用ルーチンだけ利用できる．倍精度演算用ルーチンはないことを意味する．

A. 線型計算

行列格納モードの変換

サブルーチン名	項　　目	ページ	備　考
CGSM	行列格納モードの変換（一般モード→対称行列用圧縮モード）	256	
CSGM	行列格納モードの変換（対称行列用圧縮モード→一般モード）	279	
CGSBM	行列格納モードの変換（一般モード→対称バンド行列用圧縮モード）	255	
CSBGM	行列格納モードの変換（対称バンド行列用圧縮モード→一般モード）	277	
CSSBM	行列格納モードの変換（対称行列用圧縮モード→対称バンド行列用圧縮モード）	280	
CSBSM	行列格納モードの変換（対称バンド行列用圧縮モード→対称行列用圧縮モード）	278	

行列操作

サブルーチン名	項　　目	ページ	備　考
AGGM	行列の和（実行列）	75	
SGGM	行列の差（実行列）	552	
MGGM	行列の積（実行列）	452	
MGSM	行列の積（実行列・実対称行列）	453	
ASSM	行列の和（実対称行列）	120	
SSSM	行列の差（実対称行列）	571	
MSSM	行列の積（実対称行列）	465	
MSGM	行列の積（実対称行列・実行列）	464	
MAV	実行列と実ベクトルの積	444	
MCV	複素行列と複素ベクトルの積	448	
MSV	実対称行列と実ベクトルの積	466	
MSBV	実対称バンド行列と実ベクトルの積	462	
MBV	実バンド行列と実ベクトルの積	446	

連立 1 次方程式

サブルーチン名	項　　目	ページ	備　考
LAX	実行列の連立 1 次方程式（クラウト法）	374	
LCX	複素行列の連立 1 次方程式（クラウト法）	393	
LSX	正値対称行列の連立 1 次方程式（変形コレスキー法）	431	
LSIX	実対称行列の連立 1 次方程式（ブロック対角ピボッティング手法）	424	
LSBX	正値対称バンド行列の連立 1 次方程式（変形コレスキー法）	420	
LSBIX	実対称バンド行列の連立 1 次方程式（ブロック対角ピボッティング手法）	418	
LBX1	実バンド行列の連立 1 次方程式（ガウス消去法）	388	
LSTX	正値対称 3 項行列の連立 1 次方程式（変形コレスキー法）	428	
LTX	実 3 項行列の連立 1 次方程式（ガウス消去法）	436	
LAXR	実行列の連立 1 次方程式の解の反復改良	385	*
LCXR	複素行列の連立 1 次方程式の解の反復改良	395	*
LSXR	正値対称行列の連立 1 次方程式の解の反復改良	433	*
LSIXR	実対称行列の連立 1 次方程式の解の反復改良	426	*
LSBXR	正値対称バンド行列の連立 1 次方程式の解の反復改良	422	*
LBX1R	実バンド行列の連立 1 次方程式の解の反復改良	390	*

逆行列

サブルーチン名	項　　目	ページ	備　考
LUIV	LU 分解された実行列の逆行列	439	
CLUIV	LU 分解された複素行列の逆行列	270	
LDIV	LDL^T 分解された正値対称行列の逆行列	398	

行列の三角分解

サブルーチン名	項　　目	ページ	備　考
ALU	実行列の LU 分解（クラウト法）	88	
CLU	複素行列の LU 分解（クラウト法）	268	
SLDL	正値対称行列の LDL^T 分解（変形コレスキー法）	559	
SMDM	実対称行列の MDM^T 分解（ブロック対角ピボッティング手法）	561	
SBDL	正値対称バンド行列の LDL^T 分解（変形コレスキー法）	542	
SBMDM	実対称バンド行列の MDM^T 分解（ブロック対角ピボッティング手法）	544	
BLU1	実バンド行列の LU 分解（ガウス消去法）	178	

三角分解された連立 1 次方程式

サブルーチン名	項　　目	ページ	備　考
LUX	LU 分解された実行列の連立 1 次方程式	442	
CLUX	LU 分解された複素行列の連立 1 次方程式	272	
LDLX	LDL^T 分解された正値対称行列の連立 1 次方程式	400	
MDMX	MDM^T 分解された実対称行列の連立 1 次方程式	450	
BDLX	LDL^T 分解された正値対称バンド行列の連立 1 次方程式	125	
BMDMX	MDM^T 分解された実対称バンド行列の連立 1 次方程式	181	
BLUX1	LU 分解された実バンド行列の連立 1 次方程式	175	

最小二乗解

サブルーチン名	項　　目	ページ	備　考
LAXL	実行列の最小二乗解（ハウスホルダー変換）	376	
LAXLR	実行列の最小二乗解の反復改良	383	*
LAXLM	実行列の最小二乗最小ノルム解（特異値分解法）	379	
GINV	実行列の一般逆行列（特異値分解法）	329	
ASVD1	実行列の特異値分解（ハウスホルダー法，QR 法）	121	

B.　固有値固有ベクトル

固有値固有ベクトル

サブルーチン名	項　　目	ページ	備　考
EIG1	実行列の固有値及び固有ベクトル（2 段 QR 法）	288	
CEIG2	複素行列の固有値及び固有ベクトル（QR 法）	233	
SEIG1	実対称行列の固有値及び固有ベクトル（QL 法）	547	
SEIG2	実対称行列の固有値及び固有ベクトル（バイセクション法，逆反復法）	549	
HEIG2	エルミート行列の固有値及び固有ベクトル（バイセクション法，逆反復法）	343	
BSEG	実対称バンド行列の固有値及び固有ベクトル （ルティスハウザー・シュワルツ法，バイセクション法，逆反復法）	195	
BSEGJ	実対称バンド行列の固有値及び固有ベクトル（ジェニングス法）	197	
TEIG1	実対称 3 重対角行列の固有値及び固有ベクトル（QL 法）	572	
TEIG2	実対称 3 重対角行列の固有値及び固有ベクトル （バイセクション法，逆反復法）	574	
GSEG2	実対称行列の一般固有値及び固有ベクトル（バイセクション法，逆反復法）	335	
GBSEG	実対称バンド行列の一般固有値及び固有ベクトル（ジェニングス法）	324	

固有値

サブルーチン名	項　　目	ページ	備　考
HSQR	実ヘッセンベルグ行列の固有値（2 段 QR 法）	348	
CHSQR	複素ヘッセンベルグ行列の固有値（QR 法）	261	
TRQL	実対称 3 重対角行列の固有値（QL 法）	587	
BSCT1	実対称 3 重対角行列の固有値（バイセクション法）	187	

固有ベクトル

サブルーチン名	項　　目	ページ	備　考
HVEC	実ヘッセンベルグ行列の固有ベクトル（逆反復法）	350	
CHVEC	複素ヘッセンベルグ行列の固有ベクトル（逆反復法）	263	
BSVEC	実対称バンド行列の固有ベクトル（逆反復法）	207	

その他

サブルーチン名	項　　目	ページ	備　考
BLNC	実行列の平衡化	173	
CBLNC	複素行列の平衡化	230	
HES1	実行列の実ヘッセンベルグ行列への変換（ハウスホルダー法）	345	
CHES2	複素行列の複素ヘッセンベルグ行列への変換（安定化基本相似変換）	259	
TRID1	実対称行列の実対称 3 重対角行列への変換（ハウスホルダー法）	585	
TRIDH	エルミート行列の実対称 3 重対角行列への変換（ハウスホルダー法）	582	
BTRID	実対称バンド行列の実対称三重対角行列への変換 （ルティスハウザー・シュワルツ法）	210	
HBK1	実行列の固有ベクトルへの逆変換と正規化	341	
CHBK2	複素行列の固有ベクトルへの逆変換	257	
TRBK	実対称行列の固有ベクトルへの逆変換	578	
TRBKH	エルミート行列の固有ベクトルへの逆変換	580	
NRML	実行列の固有ベクトルの正規化	487	
CNRML	複素行列の固有ベクトルの正規化	274	
GSCHL	一般形から標準形への変換（実対称行列の一般固有値問題）	333	
GSBK	一般形の固有ベクトルへの逆変換（実対称行列の一般固有値問題）	331	

C. 非線型計算

サブルーチン名	項　　目	ページ	備　考
RQDR	実係数 2 次方程式	541	
CQDR	複素係数 2 次方程式	276	
LOWP	実係数低次代数方程式（5 次以下）	410	
RJETR	実係数高次代数方程式（ジェンキンス・トラウブの方法）	535	
CJART	複素係数高次代数方程式（ヤラット法）	266	
TSD1	実超越方程式 $f(x) = 0$（ブレント法）	593	
TSDM	実超越方程式 $f(x) = 0$（マラー法）	590	
CTSDM	複素超越方程式 $f(z) = 0$（マラー法）	281	
NOLBR	連立非線型方程式（ブレント法）	475	

D. 極値問題

サブルーチン名	項　　目	ページ	備　考
LMINF	1 変換関数の極小化（微係数不要，2 次補間法）	405	
LMING	1 変換関数の極小化（微係数要，3 次補間法）	407	
MINF1	多変数関数の極小化（微係数不要，改訂準ニュートン法）	454	
MING1	多変数関数の極小化（微係数要，準ニュートン法）	458	
NOLF1	関数二乗和の極小化（微係数不要，改訂マルカート法）	479	
NOLG1	関数二乗和の極小化（微係数要，改訂マルカート法）	483	
LPRS1	線形計画問題（改訂シンプレックス法）	412	
NLPG1	非線形計画問題（微係数要，パウエル法）	470	

E. 補間・近似

補間

サブルーチン名	項　　目	ページ	備　考
AKLAG	エイトケン・ラグランジュ補間	79	
AKHER	エイトケン・エルミート補間	76	
SPLV	3 次 spline 補間式による補間	568	
BIF1	B-spline 補間式 (I) による補間	145	
BIF2	B-spline 補間式 (II) による補間	147	
BIF3	B-spline 補間式 (III) による補間	149	
BIF4	B-spline 補間式 (IV) による補間	151	
BIFD1	B-spline 2 次元補間式 (I-I) による補間	140	
BIFD3	B-spline 2 次元補間式 (III-III) による補間	143	
AKMID	2 次元準エルミート補間式による補間	81	
INSPL	3 次 spline 補間式	363	
AKMIN	準エルミート補間式	85	
BIC1	B-spline 補間式 (I)	132	
BIC2	B-spline 補間式 (II)	134	
BIC3	B-spline 補間式 (III)	136	
BIC4	B-spline 補間式 (IV)	138	
BICD1	B-spline 2 次元補間式 (I-I)	127	
BICD3	B-spline 2 次元補間式 (III-III)	129	

近似

サブルーチン名	項　　目	ページ	備　考
LESQ1	最小二乗近似多項式	402	

平滑化

サブルーチン名	項　　目	ページ	備　考
SMLE1	最小二乗近似多項式による平滑化（等間隔離散点）	564	
SMLE2	最小二乗近似多項式による平滑化（不等間隔離散点）	566	
BSF1	B-spline 平滑化式による平滑化	205	
BSC1	B-spline 平滑化式（固定節点）	190	
BSC2	B-spline 平滑化式（節点追加方式）	192	
BSFD1	B-spline 2 次元平滑化式による平滑化	203	
BSCD2	B-spline 2 次元平滑化式（節点追加方式）	183	

級数

サブルーチン名	項　　目	ページ	備　考
FCOSF	偶関数の cosine 級数展開（関数入力，高速 cosine 変換）	302	
ECOSP	cosine 級数の求和	286	
FSINF	奇関数の sine 級数展開（関数入力，高速 sine 変換）	314	
ESINP	sine 級数の求和	292	
FCHEB	実関数のチェビシェフ級数展開（関数入力，高速 cosine 変換）	296	
ECHEB	チェビシェフ級数の求和	284	
GCHEB	チェビシェフ級数の導関数	327	
ICHEB	チェビシェフ級数の不定積分	354	

F.　変換

サブルーチン名	項　　目	ページ	備　考
FCOST	離散型 cosine 変換（台形公式，2 基底 FFT）	311	
FCOSM	離散型 cosine 変換（中点公式，2 基底 FFT）	308	
FSINT	離散型 sine 変換（台形公式，2 基底 FFT ）	322	
FSINM	離散型 sine 変換（中点公式，2 基底 FFT）	319	
RFT	離散型実フーリエ変換	532	
CFTM	多次元離散型複素フーリエ変換（混合基底 FFT）	242	
CFT	多次元離散型複素フーリエ換（8, 2 基底 FFT）	239	
CFTN	離散型複素フーリエ変換（8, 2 基底 FFT，逆順出力）	246	
CFTR	離散型複素フーリエ変換（8, 2 基底 FFT，逆順入力）	251	
PNR	ビット逆転によるデータの置換	512	
LAPS1	ラプラス変換（複素右半平面で正則な有理関数）	365	
LAPS2	ラプラス変換（一般の有理関数）	367	
LAPS3	ラプラス変換（一般関数）	369	
HRWIZ	Hurwitz 多項式の判定	347	

G. 数値微積分

数値微分

サブルーチン名	項　　目	ページ	備　考
SPLV	数値微分（不等間隔離散点入力，3 次 spline 補間式）	568	
BIF1 BIF2 BIF3 BIF4	数値微分（不等間隔離散点入力，B-spline 補間式）	145 147 149 151	
BSF1	数値微分（不等間隔離散点入力，B-spline 平滑化式）	205	
BIFD1 BIFD3	2 次元数値微分（不等間隔格子点入力，B-spline 2 次元補間式）	140 143	
BSFD1	2 次元数値微分（不等間隔格子点入力，B-spline 2 次元平滑化式）	203	
GCHEB	チェビシェフ級数の導関数	327	

数値積分

サブルーチン名	項　　目	ページ	備　考
SIMP1	1 次元有限区間積分（等間隔離散点入力，シンプソン則）	553	
TRAP	1 次元有限区間積分（不等間隔離散点入力，台形則）	577	
BIF1 BIF2 BIF3 BIF4	1 次元有限区間積分（不等間隔離散点入力，B-spline 補間式）	145 147 149 151	
BSF1	1 次元有限区間積分（不等間隔離散点入力，B-spline 平滑化式）	205	
BIFD1 BIFD3	2 次元有限領域積分（不等間隔格子点入力，B-spline 2 次元補間式）	140 143	
BSFD1	2 次元有限領域積分（不等間隔格子点入力，B-spline 2 次元平滑化式）	203	
SIMP2	1 次元有限区間積分（関数入力，適応型シンプソン則）	554	
AQN9	1 次元有限区間積分（関数入力，適応型ニュートン・コーツ 9 点則）	115	
AQC8	1 次元有限区間積分（関数入力，クレンショー・カーチス型積分法）	90	
AQE	1 次元有限区間積分（関数入力，二重指数関数型積分公式）	96	
AQEH	1 次元半無限区間積分（関数入力，二重指数関数型積分公式）	100	
AQEI	1 次元全無限区間積分（関数入力，二重指数関数型積分公式）	102	
AQMC8	多次元有限領域積分（関数入力，クレンショー・カーチス型積分法）	104	
AQME	多次元積分（関数入力，二重指数関数型積分公式）	110	

H. 微分方程式

サブルーチン名	項　　目	ページ	備　考
RKG	連立 1 階常微分方程式（ルンゲ・クッタ・ギル法）	539	
HAMNG	連立 1 階常微分方程式（ハミング法）	337	
ODRK1	連立 1 階常微分方程式（ルンゲ・クッタ・ヴァーナー法）	508	
ODAM	連立 1 階常微分方程式（アダムス法）	489	
ODGE	スティフ連立 1 階常微分方程式（ギア法）	498	

I. 特殊関数

サブルーチン名	項　　目	ページ	備　考
CELI1	第 1 種完全楕円積分 $K(x)$	236	
CELI2	第 2 種完全楕円積分 $E(x)$	237	
EXPI	指数積分 $E_i(x), \overline{E}_i(x)$	294	
SINI	正弦積分 $S_i(x)$	558	
COSI	余弦積分 $C_i(x)$	275	
SFRI	正弦フレネル積分 $S(x)$	551	
CFRI	余弦フレネル積分 $C(x)$	238	
IGAM1	第 1 種不完全ガンマ関数 $\gamma(\nu,x)$	359	
IGAM2	第 2 種不完全ガンマ関数 $\Gamma(\nu,x)$	360	
IERF	逆誤差関数 $\mathrm{erf}^{-1}(x)$	356	
IERFC	逆余誤差関数 $\mathrm{erfc}^{-1}(x)$	357	
BJ0	第 1 種 0 次ベッセル関数 $J_0(x)$	163	
BJ1	第 1 種 1 次ベッセル関数 $J_1(x)$	164	
BY0	第 2 種 0 次ベッセル関数 $Y_0(x)$	217	
BY1	第 2 種 1 次ベッセル関数 $Y_1(x)$	219	
BI0	第 1 種 0 次変形ベッセル関数 $I_0(x)$	157	
BI1	第 1 種 1 次変形ベッセル関数 $I_1(x)$	158	
BK0	第 2 種 0 次変形ベッセル関数 $K_0(x)$	171	
BK1	第 2 種 1 次変形ベッセル関数 $K_1(x)$	172	
BJN	第 1 種整数次ベッセル関数 $J_n(x)$	159	
BYN	第 2 種整数次ベッセル関数 $Y_n(x)$	212	
BIN	第 1 種整数次変形ベッセル関数 $I_n(x)$	153	
BKN	第 2 種整数次変形ベッセル関数 $K_n(x)$	165	
CBIN	複素変数第 1 種整数次変形ベッセル関数 $I_n(z)$	221	
CBKN	複素変数第 2 種整数次変形ベッセル関数 $K_n(z)$	227	
CBJN	複素変数第 1 種整数次ベッセル関数 $J_n(z)$	223	
CBYN	複素変数第 2 種整数次ベッセル関数 $Y_n(z)$	232	
BJR	第 1 種実数次ベッセル関数 $J_\nu(x)$	161	
BYR	第 2 種実数次ベッセル関数 $Y_\nu(x)$	213	
BIR	第 1 種実数次変形ベッセル関数 $I_\nu(x)$	155	
BKR	第 2 種実数次変形ベッセル関数 $K_\nu(x)$	166	
CBJR	複素変数第 1 種実数次ベッセル関数 $J_\nu(z)$	225	
NDF	正規分布関数 $\phi(x)$	468	
NDFC	余正規分布関数 $\psi(x)$	469	
INDF	逆正規分布関数 $\phi^{-1}(x)$	361	
INDFC	逆余正規分布関数 $\psi^{1}(x)$	362	

J. 擬似乱数

サブルーチン名	項　　目	ページ	備　考
RANU2	一様乱数 (0, 1) の生成	523	#
RANU3	一様乱数 (0, 1) の生成（シャフル型）	525	#
RANN1	正規乱数の生成（高速型）	518	#
RANN2	正規乱数の生成	520	#
RANE2	指数乱数の生成	517	#
RANP2	ポアソン乱数の生成	521	#
RANB2	二項乱数の生成	515	#
RATF1	一様乱数 (0, 1) の頻度テスト	527	#
RATR1	一様乱数 (0, 1) の上昇・下降連テスト	529	#

本マニュアルの読み方

本マニュアルを利用するにあたり，利用者が必要な情報を迅速に，正確に，かつ誤ることなく把握できるように，ここでは本マニュアルの論理的構造を示すとともに，必要な情報を得るための手順を述べる．

本マニュアルは，大きく 2 部から構成されている．

第 I 部には SSL II の概説，第 II 部には SSL II の各サブルーチンの使用方法を述べている．

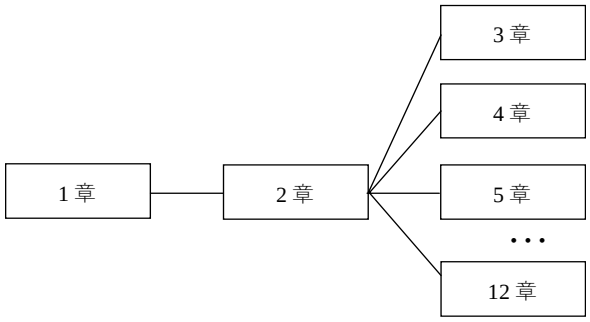
第 I 部は，12 章から構成されている．

第 1 章では，SSL II の開発背景，開発方針，あるいは全体としての特徴を述べてある．

第 2 章では，SSL II のどのサブルーチンを使う場合にも必要となる SSL II の一般規約を述べてある．したがって，SSL II の利用にあたり，この章は必ず目を通していただきたい．

第 3 章から第 12 章までは，目次に見られるとおり数値計算の各分野ごとに章を設け，必要な章だけ拾って読むことが容易にできるように，できるだけ互いに独立させてある．そして，その章に属するサブルーチンの使分けなり，目的なりを述べたので使用を誤らないためにも必ず一読していただきたい．

以上のように各章は非常に簡単な論理的関係しかなく，それを図示すれば次のようになる．



この図から分かるように，例えば行列の固有値を求めたい利用者なら第 2 章を読んだあと第 4 章へ進んでどのサブルーチンを使えばよいかをつかむことができる．

第 II 部では，SSL II の個々のサブルーチンの使用方法の説明が，サブルーチン名のアルファベット順に編集されている．

個々のサブルーチンの説明においては，

(1) 機能
(2) パラメタ
(3) 使用上の注意
(4) 手法概要

の四つの部分に分けて順に述べてある．これらの各々において意図したことは次のとおりである．

(1) 機能

そのサブルーチンがどのような数学的な量を求めるか，あるいはどのような処理を行うかを述べる．

(2) パラメタ

そのサブルーチンが必要とする入力情報あるいは演算後の出力情報のための変数，配列について説明する．

パラメタ名についてはなるべく慣用に従い，しかも統一をとりながら名前をつけている．

(3) 使用上の注意

ここでは，更に次の三つの部分に分けている．

a． 使用する副プログラム

そのサブルーチンが内部で SSL II の他のサブルーチンを使っている場合は「SSL II」という項でそれを挙げる．また FORTRAN の組込み関数，基本外部関数を使っている場合は，「FORTRAN 基本関数」という項でそれを挙げる．

b． 注意

そのサブルーチンを使うにあたっての注意事項を挙げる．

c． 使用例

そのサブルーチンの使い方について一つの例を挙げる．ここでは分かりやすい例を挙げることを考えたため，工学，物理学などへの応用にはいたっていない．

数学的な量がそのサブルーチンだけでは得られず，他のサブルーチンを併用することにより得られる場合は，その組合せを重視しなければならないので，それを明らかにするような例にしている（特に線型計算，固有値固有ベクトルの分野においてはこのような例が多い）．

なお，使用例を挙げるにあたり入力データの大きさ，その他の条件についての仮定は，使用例の先頭で説明している．

(4) 手法概要

そのサブルーチンが使っている手法を概略的に述べる．本マニュアルは使用手引書であるから，その手法の考え方，計算手順だけに目標をしぼって述べている．

なおそのサブルーチンの作成にあたり直接に引用した文献，あるいは理論において重要な文献は「付録 4. 参考文献一覧表」に載せたので，「手法概要」の説明でいたらないところは，それらを参照していただきたい．

本マニュアルは以上 2 部のほかに，「SSL II 一覧表」及び「付録 1. 補助サブルーチン」，「付録 2.

SSL II サブルーチン名一覧表」，「付録 3. 分類コードとサブルーチン名対応表」及び「付録 4. 参考文献一覧表」を添えている．

「SSL II 一覧表」は，用意されているサブルーチンを分野別に，しかも機能別に載せたものである．この表はサブルーチンの索引などに使える．

「付録 1. 補助サブルーチン」は，補助サブルーチンの機能について述べてある．

「付録 2. SSL II サブルーチン名一覧表」は，

1. 一般サブルーチン
2. スレーブサブルーチン
3. 補助サブルーチン

の三つの表から成る．1. は，SSL II 一覧表のサブルーチン名を全分野をとおしてアルファベット順に並べ，各サブルーチンが内部でどのようなサブルーチンを使っているかを示したものである．更に 2. は一つ一つのスレーブサブルーチン（この定義については 2.1 節で述べる）がどのサブルーチンから使われるかを示したものであり，3.は補助サブルーチンの一覧表である．

「付録 3. 分類コードとサブルーチン名対応表」は，サブルーチン名を分類コード順に載せたものである．この表は，分類コードからサブルーチン名を索引するときに有効である．

「付録 4. 参考文献一覧表」は，SSL II の開発にあたり直接に引用した文献，あるいは理論において重要な文献を載せたものである．

本マニュアルを利用するにあたっての予備知識としては FORTRAN 言語の文法のほか何ら想定はしないが，利用者を，ある程度，数値計算に興味を持っていることを予期している．

なお，下記に本マニュアル中で使用している数学記号を示してある．

数学記号表

記　号	凡　例	意　味	備　考
T	$\boldsymbol{A}^{\mathrm{T}}$	（行列 $\boldsymbol{A}$ の）転置行列.	
	$\boldsymbol{x}^{\mathrm{T}}$	（縦ベクトル $\boldsymbol{x}$ の）転置ベクトル（横ベクトル）.	
	$\boldsymbol{x} = (x_1,\ldots,x_n)^{\mathrm{T}}$	縦ベクトル.	記号 ( ) を見よ.
-1	$\boldsymbol{A}^{-1}$	（行列 $\boldsymbol{A}$ の）逆行列.	
*	$\boldsymbol{A}^{*}$	（行列 $\boldsymbol{A}$ の）共役転置行列.	
	$\boldsymbol{x}^{*}$	（縦ベクトル $\boldsymbol{x}$ の）共役転置ベクトル.	
	z^{*}	（複素数 z の）共役複素数.	$z = a + ib$ のとき，$z^{*} = a - ib$
−	$\bar{z}$		$\bar{z} = z^{*}$
[ ]	$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & & \ldots \\ \ldots & & \ldots & \ldots \\ a_{n1} & \ldots & \ldots & a_{nn} \end{bmatrix}$	（$\boldsymbol{A}$ は要素 a_{ij} を持つ $n \times n$ の）行列.	
( )	$\boldsymbol{x} = (x_1,\ldots, x_n)^{\mathrm{T}}$	（$\boldsymbol{x}$ は要素 x_i を持つ n 次元の）縦ベクトル.	
	$\boldsymbol{A} = (a_{ij})$	行列 $\boldsymbol{A}$ の要素（を a_{ij} で表す）.	
	$\boldsymbol{x} = (x_i)$	縦ベクトル $\boldsymbol{x}$ の要素（を x_i で表す）.	
diag	$\boldsymbol{A} = \mathrm{diag}\,(a_{ii})$	（行列 $\boldsymbol{A}$ は要素 a_{ii} を持つ）対角行列.	
$\boldsymbol{I}$		単位行列.	
det	$\det(\boldsymbol{A})$	（行列 $\boldsymbol{A}$ の）行列式	
rank	$\mathrm{rank}\,(\boldsymbol{A})$	（行列 $\boldsymbol{A}$ の）階数	
‖ ‖	$\|\boldsymbol{x}\|$	（ベクトル $\boldsymbol{x}$ の）ノルム n 次元の $\boldsymbol{x} = (x_i)$ なるベクトルのとき， $\| \boldsymbol{x} \|_1 = \sum_{i=1}^{n} \| x_i \|$ ：一様ノルム $\| \boldsymbol{x} \|_2 = \sqrt{\sum_{i=1}^{n} \| x_i \|^2}$ ：ユークリッドノルム $\| \boldsymbol{x} \|_\infty = \max_i \| x_i \|$ ：インフィニティノルム 記号 Σ , | |, max を見よ.	
	$\|\boldsymbol{A}\|$	（行列 $\boldsymbol{A}$ の）ノルム $n \times n$ の $\boldsymbol{A} = (a_{ij})$ なる行列のとき， $\|\boldsymbol{A}\|_\infty = \max_i \left(\sum_{j=1}^{n} \|a_{ij}\| \right)$:インフィニティノルム.	
(,)	$(\boldsymbol{x}, \boldsymbol{y})$	（ベクトル $\boldsymbol{x}$ と $\boldsymbol{y}$ の）内積.	$\boldsymbol{x}$, $\boldsymbol{y}$ が複素ベクトルのとき， $(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^{\mathrm{T}} \bar{\boldsymbol{y}}$
	(a, b)	開区間.	

[,]	$[a, b]$	閉区間.	

記　号	凡　例	意　味	備　考
>>	$a >> b$	（a は b より）非常に大きい.	
≠	$a \neq b$	（a は b に）等しくない.	
≈	$f(x) \approx P(x)$	近似記号（$f(x)$を $P(x)$で近似する）.	
≡	$F(x) \equiv f'(x) / f(x)$	定義記号（$F(x)$を $f'(x) / f(x)$で定義する）	
{ }	$\{x_i\}$	数列	
Σ	$\sum_{i=m}^{n} x_i$	（$x_m, ..., x_n$ の）総和.	$n<m$ の場合は，無視する. $\sum_{\substack{i=m \\ i\neq l}}^{n} x_i$ の場合は，x_l を除いた総和.
	$\sum_i x_{i+2j}$	（i に関する）総和.	
	y',　$f'(x)$	$y' = \frac{dy}{dx}$,　$f'(x) = \frac{df(x)}{dx}$	n 階の導関数は, $f^{(n)}(x) = \frac{d^n f(x)}{dx^n}$
| |	$\|z\|$	（z の）絶対値.	$z = a + ib$ の場合, $\|z\| = \sqrt{a^2 + b^2}$
max	$\max(x_1,..., x_n)$ $\max_i x_i$	$(x_1,..., x_n)$の最大値.	
min	$\min(x_1,..., x_n)$ $\min_i x_i$	$(x_1,..., x_n)$の最小値.	
sign	sign (x)	x の符号.	x が正のとき 1. x が負のとき -1.
log	log x	x の自然対数	
Re	Re (z)	（複素数 z の）実部.	
Im	Im (z)	（複素数 z の）虚部.	
arg	arg z	（複素数 z の）偏角.	
δ_{ij}		クロネッカーのデルタ.	
γ		オイラー定数.	
π		円周率.	
i	$z = a + ib$	虚数単位.	$i = \sqrt{-1}$
P.V.	P.V. $\int_{-\infty}^{x} \frac{e^t}{t} dt$	（積分の）主値.	
$\frac{\|}{\|}$	$b_0 + \frac{a_1\|}{\|b_1} + \frac{a_2\|}{\|b_2} + \cdots$	連分数.	
∈	$x \in X$	（元 x は集合 X に）属する.	
{| }	$\{x \mid x = \varphi(x)\}$	（$x = \varphi(x)$ を満たす x 全体の）集合.	
C^k	$f(x) \in C^k[a,b]$	($f(x)$は$[a, b]$で) k 階導関数までが連続な関数.	

［**注意**］ 各記号の意味を違えて用いる場合は，その箇所で再定義をしている.
また，慣用に従い明確な記号については，本表から省略した.

第Ⅰ部　概　　説

第 1 章
SSL II の概要

1.1 SSL II の開発背景

数値計算ライブラリとしての SSL (Scientific Subroutine Library) が開発されてから，既にかなりの歳月が流れている．その間，数値計算技術の進歩や計算機の能力向上に伴い，SSLも何回かにわたって，いくつかの改善や機能追加がなされてきた．しかしながら，次のような要望に，SSLの改善や機能追加の範囲で対処することは困難になってきた．

① ライブラリ全体として，個々のサブルーチンの機能と仕様のバランスをよくすること．
② 新手法の追加によって，体系がくずれることのないようにすること．
③ マニュアルにおいて，各手法や使分けに関する説明を充実すること．

そこで，以上のような背景を踏まえて，新たに開発したものが SSL II である．

1.2 SSL II の開発方針

(1) 体系化

利用者が自分の要求に合うサブルーチンを的確に速く見い出せることは，ライブラリとして最も大切な要素である．SSL II では次の2点に重点をおいて体系化を図っている．

① 数値計算の分野を，次のように区分している．
   A 線型計算
   B 固有値固有ベクトル
   C 非線形計算
   D 極値問題
   E 補間・近似
   F 変換
   G 数値微積分
   H 微分方程式
   I 特殊関数
   J 擬似乱数

また，それぞれの分野に応じて，更に細区分し，ライブラリ全体が階層構造をなすようになっている．これによって，個々のサブルーチンの位置づけが，明確になっている．

② 分野によっては機能の細分化を図っている．したがって，一つのまとまった機能と細分化された機能との2種類のサブルーチンを用意しており，よりきめの細かい使用も可能となっている．

(2) 性能の向上

アルゴリズムとプログラミングの面から，精度と速度の向上を図っている．

① アルゴリズムの面では，精度的に安定な手法のうち新しいものをベースとしている．したがって，従来標準的なものとされてきた手法でも，SSL II では除かれているものがある．
② プログラミングの面では特に速度向上に重点がおかれている．システム間の互換性及び信頼性を高めるなどの理由から，使用言語はアセンブラを使わずに，FORTRANを用いて，コンパイラの最適化機能の恩恵をできるだけ受けるようにコーディングされている．
   なお，VS方式に対するプログラミング上の配慮としては，VS方式でない計算機に対して逆効果とならない程度に，プログラムの局所性を高める工夫がなされている．

(3) 信頼性の向上

一部の分野を除いては，単精度，倍精度用ルーチンが同一のソースプログラムから作り出されるようになっている．

(4) 互換性の維持

開発されたソフトウェアは，複数の異種システム間で持ち運びされることが日常化して来ており，これに対応するためにライブラリは，システムに依存しないように作られるべきであるといわれている．

SSL II の構成は，システムに依存する箇所を少数の補助サブルーチンにして，異種システム間での互換性を維持できるようにしている．

1.3　SSL II　の特徴

以下にSSL II の特徴を列挙する.

- SSL II　は FORTRAN で作成されたサブルーチンの集合体である.
  したがって，利用者は，自分の要求に合ったサブルーチンを利用者プログラムから CALL 文によって利用できる.
- すべてのサブルーチンは入力文を使わずに作られている．したがって，利用者の与えるデータは，すべて主記憶上にあるものとして扱われる.
- データの大きさはサブルーチンのパラメタを通して与えるようになっており，その大きさはサブルーチンの内部では制限されない.
- 行列を扱う線型計算や固有値固有ベクトルの分野においては，記憶領域節約のため，対称行列，バンド行列は，圧縮モード（「2.8 データの格納方法」参照）で処理される.
- すべてのサブルーチンは，実行後の状態を示す出力パラメタを持っている．このパラメタの内容は，処理された状態に応じて，大きくいくつかの段階にわかれている（「2.6 サブルーチン実行後の状態」参照）．サブルーチンはどのような場合においても必ず利用者のプログラムに戻るので，利用者は，サブルーチン実行後このパラメタの内容を調べることによって適切な処置をとることができる.
  更に，利用者の指示に従って，このサブルーチン実行後の状態を出力することもできる.
  （「2.6 サブルーチン実行後の状態」参照）.

1.4　SSL II　を利用できるシステム

利用者システムにおいて，FORTRAN コンパイラが使用できるなら，SSL II　はシステム構成の大きさにかかわらず，原則として使用できる．ただし主記憶の大きさは，使用されるSSL II　サブルーチンの数とその大きさ，利用者プログラムの大きさ及びデータの大きさによって左右される.

なおSSL II　は，上述のように FORTRAN のサブルーチンとなるよう記述されているが，そのシステムで許容されるならば，ALGOL, PL/I, COBOL，その他の言語でも使うことができる.

その利用方法は，そのシステムの FORTRAN 使用手引書及びその言語の使用手引書の異種言語間結合の項を参照されたい.

第 2 章
SSL II の一般規約

2.1 サブルーチンの種類

SSL II のサブルーチンには，利用形態に応じて，表 2.1 のように，三つの種類のルーチンがある．

表 2.1 サブルーチンの種類

サブルーチンの種類	副プログラムの区分	利用形態
一般サブルーチン	サブルーチン副プログラム	利用者の使用できるルーチンである．
スレーブサブルーチン	サブルーチン副プログラム又は,関数副プログラム	一般サブルーチンが固有に呼び出すルーチンである． 利用者が直接このルーチンを扱うことはできない．
補助サブルーチン		一般サブルーチン，スレーブサブルーチンを全般にわたり補助するルーチンである．

更に，表 2.1 のうち一般サブルーチンは，機能の程度に応じて，表 2.2 のように，二つのレベルのルーチンにわけられる．この区分は，ある一つの機能を果たすためのルーチンが，内部的には更に幾つかの機能単位に分かれる場合に生じる．

表 2.2 サブルーチンのレベル

種 類	機能の程度
標準ルーチン (standard routine)	一つのまとまった機能を果たす．例えば，連立 1 次方程式を解く，など．
コンポーネントルーチン (component routine)	一つのまとまった機能を果たすための機能単位． 例えば，係数行列の三角分解など． いくつかのコンポーネントルーチンを組み合せて，標準ルーチンとなる．

2.2 分類コード

SSL II のサブルーチンには，次ページ図 2.1 のような一般規則に従って，11 桁の分類コード が割り振られている．

2.3 サブルーチン名

SSL II の各種サブルーチンには，以下に述べるような一般規則に従って，サブルーチン名がつけられている．

(1) 一般サブルーチン

図 2.2 に従っている．先頭の文字が S 又は D により，演算精度を区別している．

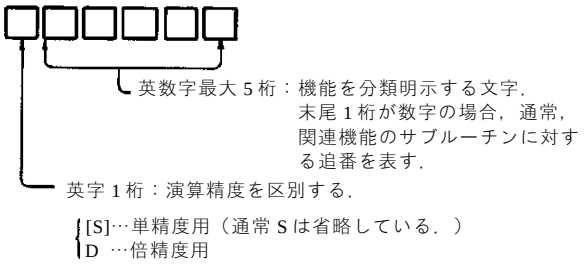


図 2.2 一般サブルーチン名

(2) スレーブサブルーチン

a. サブルーチン副プログラム又は実数型関数副プログラムの場合

図 2.3 に従っている．演算精度の区別は，(1) と同様であるが，更に固定的に“U”が続いている．

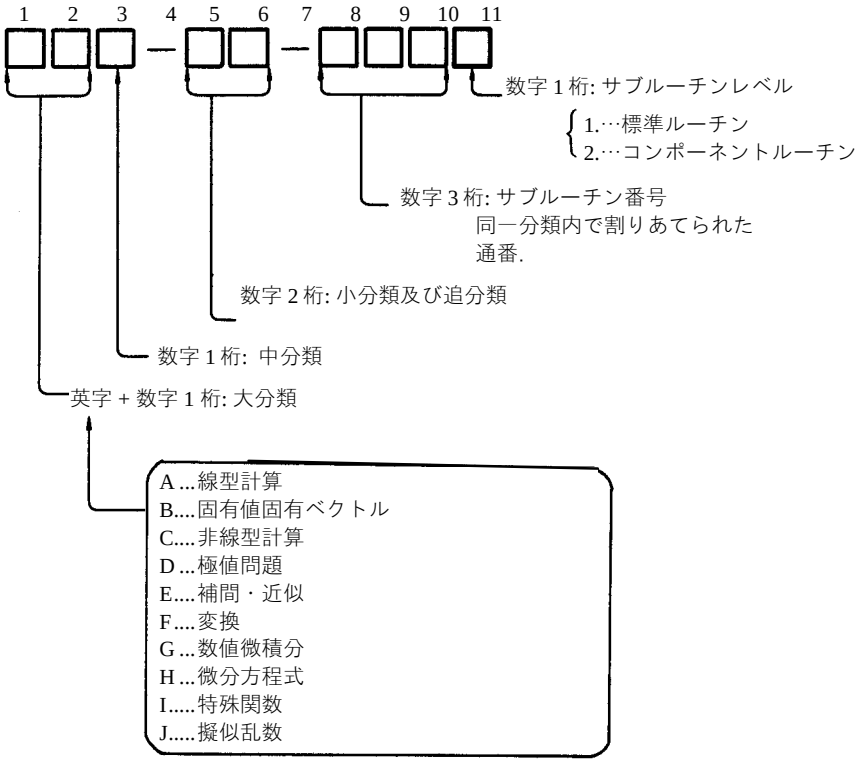


図 2.1　分類コードの割振り

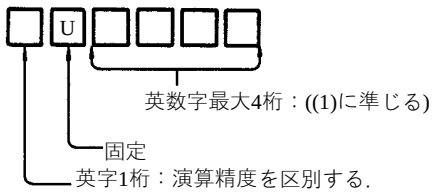


図 2.3　スレーブサブルーチンのサブルーチン名 (I)

b.　複素数型関数副プログラムの場合
図 2.4 に従っている．先頭は固定的に“Z”が付けられており，他は a.に準じている．

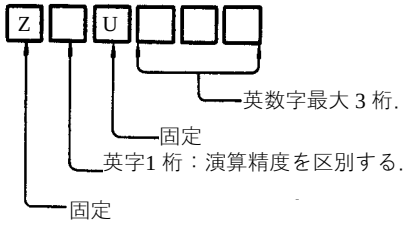


図 2.4　スレーブサブルーチンのサブルーチン名 (II)

(3)　補助サブルーチン

補助サブルーチンについては，それぞれの機能に応じた適切なサブルーチン名が付けられている．詳細は，付録 1 を参照すること．

2.4　パラメタ

SSL II のサブルーチンと利用者のプログラムとのデータの受け渡しは，すべてパラメタ渡しにより行う．

ここでは，SSL II におけるパラメタの種類とその並び順，及び諸注意について述べる．

(1)　パラメタの種類

利用者側からみたときの入出力関係に応じて，次の種類を定義する．

a.　入出力パラメタ…パラメタに値を入力する．また演算後，結果が同一領域上に出力される．

b.　入力パラメタ…パラメタに値を入力する．演算後内容は保存される．
なお，保存されない場合には，“演算後，内容は保存されない”と各サブルーチンの説明の項で明記される．

c.　出力パラメタ…パラメタ値が出力される．

d.　作業領域パラメタ…作業領域として使用される．
演算後，内容は原則として意味を持たない．

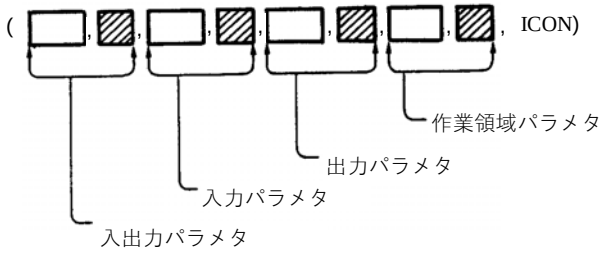
更に，パラメタの内容に応じて，次の種類を定義する．

a.　主部パラメタ…数値計算の実質的な対象となるデータを格納する（例えば，行列の要素など）．

b.　制御部パラメタ…数値計算の実質的な対象とならないデータを格納する．（例えば，行列の次数，判定値など）

(2)　パラメタの並び順

パラメタの並び順は，原則的には，パラメタの種類に応じて図 2.5 のような順に従っている．



□部は主部パラメタ，▨部は制御部パラメタである．
ICON は実行後の状態を示すパラメタである．

図 2.5　パラメタの並び順

なお，制御部パラメタのうち，図 2.5 の原則に従わないものもあるので（例えば，行列の整合寸法など），実際の並びは，各サブルーチンの説明の項を参照すること．

(3)　パラメタの扱いに関する諸注意

a.　パラメタの型
パラメタの型は，FORTRAN が定める暗黙の型宣言に従っている．
例えば，A は 4 バイトの実数型（倍精度用サブルーチンでは，8 バイトの実数型），IA は標準バイト長の整数型である．
更に，複素データを複素変数として扱う場合には，パラメタの先頭が“Z”で始まっている．例えば，ZA は 8 バイトの複素数型（倍精度用サブルーチンでは 16 バイトの複素数型）である．

b.　パラメタの外部手続き名
パラメタとして，外部手続き名を指定する場合には，SSL II のサブルーチンを呼び出す利用者のプログラムで，この外部手続き名を EXTERNAL 文で宣言する必要がある．

c.　実行後の状態
SSL II のサブルーチンには，実行後の状態を示すパラメタ ICON が用意されている．
詳細は，2.6 節を参照すること．

2.5　各種定義

(1)　行列の区分

SSL II で扱う行列は，表 2.3 に従って区分する．

表 2.3　行列の区分

区分の要因	区分内容
構造 (structure)	密行列　(dense matrix) バンド行列　(band matrix)
形式 (form)	対称　(symmetric matrix) 非対称　(unsymmetric matrix)
型 (type)	実行列　(real matrix) 複素行列　(complex matrix)
性質 (character)	正値　(positive definite) 正則　(non singular) 非正則　(singular)

(2)　行列の各部分の呼称

SSL II で扱う行列において，その各部分を次ページ図 2.6 のように呼称する．ここで，$n \times n$ の行列 $\boldsymbol{A}$ の要素を a_{ij} と表す．

(3)　行列の定義と呼称

SSL II で扱う行列のうち，その構造に従って固有の呼称を持つ行列について定義する．

a.　上三角行列
上三角行列とは，(2.1) により特徴づけられる行列である．

$$a_{ij} = 0, \quad j < i \tag{2.1}$$

すなわち，行列の下三角部分はすべて零要素である．

b.　単位上三角行列
単位上三角行列とは，(2.2) により特徴づけられる行列である．

$$a_{ij} = \begin{cases} 1, & j = i \\ 0, & j < i \end{cases} \tag{2.2}$$

すなわち，すべての対角要素が 1 である上三角行列である．

c.　下三角行列
下三角行列とは，(2.3) により特徴づけられる行列である．

$$a_{ij} = 0, \quad j > i \tag{2.3}$$

すなわち，行列の上三角部分はすべて零要素である．

d.　単位下三角行列
単位下三角行列とは，(2.4) により特徴づけられる行列である．

$$a_{ij} = \begin{cases} 1, & j = i \\ 0, & j > i \end{cases} \tag{2.4}$$

すなわち，すべての対角要素が 1 である下三角行列である．

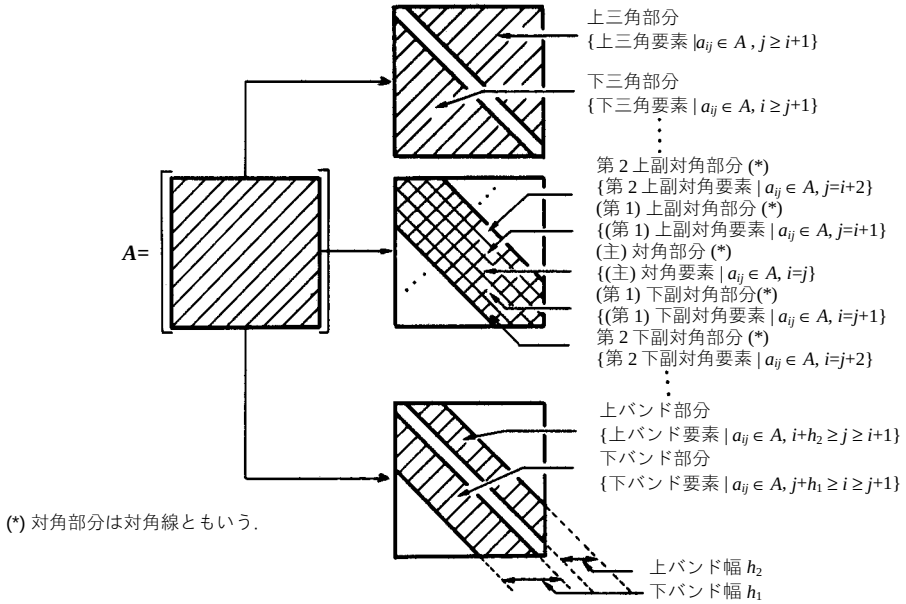


図 2.6　行列の各部分の呼称

e. 対角行列
対角行列とは，(2.5) により特徴づけられる行列である.

$$a_{ij} = 0, \quad j \neq i \tag{2.5}$$

すなわち，行列の上，下三角部分はすべて零要素である.

f. 三重対角行列（3 項行列）
三重対角行列（3 項行列）とは，(2.6) に特徴づけられる行列である.

$$a_{ij} = \begin{cases} 0, & j < i-1 \\ 0, & j > i+1 \end{cases} \tag{2.6}$$

すなわち，行列の上，下副対角部分及び主対角部分を除きすべて零要素である.

g. ブロック対角行列
$n \times n$ 行列 $\boldsymbol{A}$ において $n_i \times n_j$ 行列 $\boldsymbol{A}_{ij}$ (ブロック) を考える．ここで，$n = \sum n_i = \sum n_j$.
ブロック対角行列とは，(2.7) で特徴づけられる行列である.

$$\boldsymbol{A}_{ij} = 0, \quad i \neq j \tag{2.7}$$

すなわち，ブロックが対角線上にあり，それらブロックの直和で表せる行列である.

h. ヘッセンベルグ行列
ヘッセンベルグ行列とは，(2.8) により特徴づけられる行列である.

$$a_{ij} = 0, \quad j < i-1 \tag{2.8}$$

すなわち，行列の上三角部分，主体角部分及び下副対角部分を除き，すべて零要素である.

i. 対称バンド行列
対称バンド行列とは，上，下バンド幅 h として，(2.9) により特徴づけられる行列である.

$$a_{ij} = \begin{cases} 0, & |i-j| > h \\ a_{ji}, & |i-j| \leq h \end{cases} \tag{2.9}$$

すなわち，行列の対角部分及び上，下バンド部分を除きすべて零要素である.

j. バンド行列
バンド行列とは，下バンド幅 h_1, 上バンド幅 h_2 として，(2.10) により特徴づけられる行列である.

$$a_{ij} = \begin{cases} 0, & j > i+h_2 \\ 0, & i > j+h_1 \end{cases} \tag{2.10}$$

すなわち，対角部分及び上，下バンド部分を除き，すべて零要素である.

k. 上バンド行列
上バンド行列とは，上バンド幅 h として，(2.11) により特徴づけられる行列である.

$$a_{ij} = \begin{cases} 0, & j > i+h \\ 0, & j < i \end{cases} \tag{2.11}$$

すなわち，対角部分及び上バンド部分を除き，すべて零要素である.

l.　単位上バンド行列
単位上バンド行列とは，上バンド幅 *h,* として，(2.12) により特徴づけられる行列である．

$$a_{ij} = \begin{cases} 1, & j = i \\ 0, & j > i + h \\ 0, & j < i \end{cases} \qquad (2.12)$$

すなわち，すべての対角要素が 1 である上バンド行列である．

m.　下バンド行列
下バンド行列とは，下バンド幅 *h* として，(2.13)により特徴づけられる行列である．

$$a_{ij} = \begin{cases} 0, & j < i - h \\ 0, & j > i \end{cases} \qquad (2.13)$$

すなわち，対角部分及び下バンド部分を除き，すべて零要素である．

n.　単位下バンド行列
単位下バンド行列とは，下バンド幅 *h* として，(2.14) により特徴づけられる行列である．

$$a_{ij} = \begin{cases} 1, & j = i \\ 0, & j < i - h \\ 0, & j > i \end{cases} \qquad (2.14)$$

すなわちすべて対角要素が 1 である下バンド行列である．

o.　エルミート行列
エルミート行列とは，(2.15) により特徴づけられる行列である．

$$a_{ji} = a_{ij}^{*} \qquad (2.15)$$

すなわち，共役転置行列ともとの行列が一致する行列である．

2.6　サブルーチン実行後の状態

SSL II のサブルーチンには，実行後の状態を示すパラメタ ICON が用意されている．ICON には，実行後のコンディションコードがセットされているので，その値を判定した後に結果を用いる必要がある．ここでは，コンディションコードの内容と，関連した諸注意について述べる．

(1)　コンディションコード

コードは 0 から 30000 の値をとるが，結果が保障されるか否かに応じて，表 2.4 のように区分されている．

表 2.4　コンディションコードの区分

コード	意　　味	結果の保障	区分
0	正常に処理が終了している．	結果は保障される．	正常
1 ~ 9999	正常に処理が終了しているが，なんらかの補助的な情報を含んでいる．		
10000 ~ 19999	処理の過程で，内部的な制限を加えることにより，一応の処理は終了している．	制限付きで結果は保障される．	警告
20000 ~ 29999	処理の過程で，異常が生じ，処理が打ち切られた．	結果は保障されない．	異常
30000	入力パラメタにエラーがあったため，処理が打ち切られた．		

実際のコード細分は，サブルーチンにより異なるので，各サブルーチンの説明の項のコンディションコード表を参照すること．

(2)　コンディションコードに関する諸注意

a.　コード判定による処理の制御
SSL II のサブルーチンを呼び出す文の直後で，コンディションコードの値を判定し，結果が保障されるか否かにより以降の処理を制御する必要がある．

```
   …
CALL LSX (A, N, B, EPSZ, ISW, ICON)
IF (ICON. GE. 20000) STOP
   …
```

b.　コンディションメッセージの出力
SSL II のサブルーチンには，コンディションメッセージを出力する機能がある．
通常はメッセージ出力をしないが，利用者があらかじめ，メッセージ出力制御用ルーチン MGSET (SSL II の補助サブルーチン)を呼び出すことにより，自動的にメッセージ出力を行うことができる．

2.7　SSL II における配列

SSL II では，ベクトル及び行列などの演算において，配列上でそのデータを処理することが多い．

ここでは，パラメタに現れる配列と，その扱いについて使用上の注意を述べる．

SSL II で用いるところの配列は，その大きさが利用者プログラム内の配列宣言により決定される配列を用いている．したがって，利用者プログラムで処理するデータの大きさに見合った配列を用意すればよい．

(1)　1 次元配列

大きさ N の 1 次元配列 B なるパラメタに，n (= **N**) 次元ベクトル $\boldsymbol{b}(=(b_1, \ldots, b_n)^T)$ を格納する場合は，以下のように幾つかの使用方法が可能である．（以下の例では，連立 1 次方程式を解くサブルーチン LAX のパラメタ B に注目している． $n \leq 10$ の場合である．）

a.　一般的な使用例

大きさ 10 の 1 次元配列 B に，定数ベクトル $\boldsymbol{b}$ を B(1) = b_1, B(2) = b_2, ...のように格納する場合．

```
DIMENSION B(10)
   …
CALL LAX (…, N, B, …)
   …
```

b.　応用的な使用例

C (10, 10) なる 2 次元配列の第 I 列に，定数ベクトル $\boldsymbol{b}$ を，C (1, I) = b_1, C (2, I) = b_2, ...のように格納する場合．

```
DIMENSION C (10, 10)
   …
CALL LAX (…, N, C (1, I), …)
   …
```

上の例から分かるように，パラメタ B には，データが主記憶上で**連続的に**格納される，大きさ N 以上の領域の先頭要素（先頭番地）を指定すれば“大きさ N の 1 次元配列”に制約されない．

ベクトル $\boldsymbol{b}$ を，C(I, 1) = b_1 ,C (I, 2) = b_2,...のように第 I 行に格納して，

```
   …
CALL LAX (…, N, C (I, 1), …)
   …
```

なる使用をすることは不可能である．

(2)　2 次元配列

A (K, N) なる 2 次元配列パラメタに，$n \times n$ の実行列　$A\,(=(a_{ij}))$ を格納する場合を考える．

ところで SSL II のサブルーチンの 2 次元配列を扱う場合には，パラメタとして配列 A，次数 N だけでなく，原則的には更に整合寸法 K を入力するようになっている．

SSL II　で用いるところの整合寸法とは，利用者プログラムで用意した 2 次元配列 A の行数 K のことを意味する．そこで，この整合寸法を用いた 2 次元配列のパラメタを持つ使用方法を示す．（以下の例では，サブルーチン LAX のパラメタ A に注目している．$n \leq 10$ の場合である．）

A (10, 10) なる 2 次元配列に，係数を図 2.7 のように A (1, 1) = a_{11}, A (2, 1) = a_{21}, ..., A (1, 2) = a_{12} ....と格納する場合．

```
DIMENSION A (10, 10)
   …
K = 10
CALL LAX (A, K, N, …)
   …
```

ここで N の値のいかんによらず，整合寸法として K = 10 と与えなければならない．

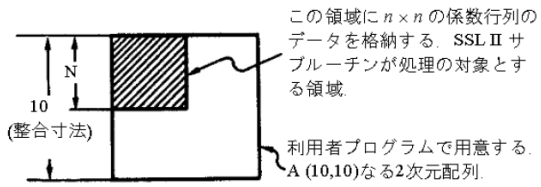


図 2.7　2 次元の整合配列例

次数の異なる幾つかの方程式を解く場合には，そのうち最も大きな次数を NMAX とすれば，利用者プログラムでは，A (NMAX, NMAX) なる 2 次元配列を一つ用意しておけば，その領域を逐次利用することができる．このときは，整合寸法として常に，NMAX の値を与えなければならない．

2.8　データの格納方法

ここでは，データとして行列及びベクトルを扱う場合のデータの格納方法について述べる．

(1)　行列

行列の格納方法は，その構造と形式に応じて異なる．非対称密行列ー通常，一般行列と称すーは全要素を 2 次元配列に格納するが，それ以外は，原則として必要とする要素だけ 1 次元配列に格納する方法を採用している．前者の格納形態を“一般モード”といい，後者の格納形態を“圧縮モード”という．

以下に，各種の行列とその格納方法を示す．

①　一般モード

図 2.8 のように格納することを一般モードという．

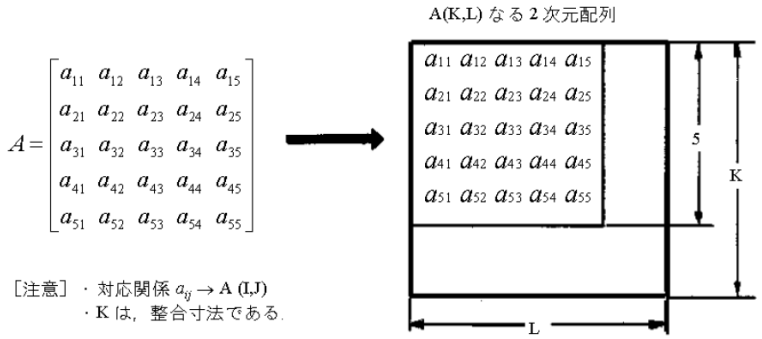


図 2.8　非対称密行列の格納方法

② 対称行列用圧縮モード
図 2.9 のように，対称密行列 ***A*** の対角部分と下三角部分の要素を，1 次元配列 A 上に行ごとに格納することを対称行列用圧縮モードという．

③ エルミート行列用圧縮モード
図 2.10 のようにエルミート行列 ***A*** の対角部分と下三角部分の要素を 2 次元配列 A 上に格納することをエルミート行列用圧縮モードという．

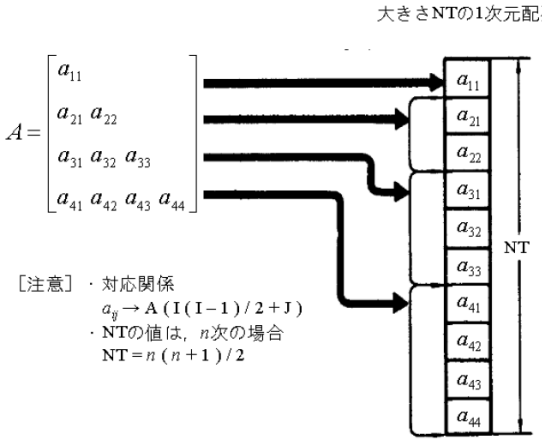


図 2.9　対称密行列の格納方法

④ 対称バンド行列用圧縮モード
図 2.11 のように対称バンド行列 ***A*** の対角部分と下バンド部分の要素を 1 次元配列 A 上に行ごとに格納することをバンド行列用圧縮モードという．

⑤ バンド行列用圧縮モード
図 2.12 のように非対称バンド行列 ***A*** の対角部分と上,下バンド部分の要素を 1 次元配列 A 上に行ごとに格納することをバンド行列用圧縮モードという．

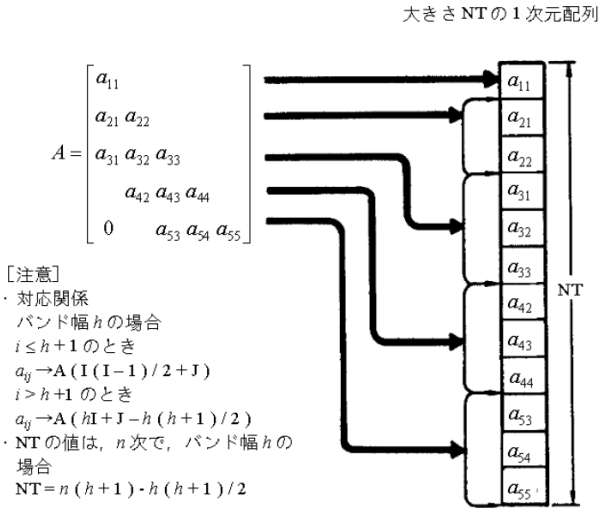


図 2.11　対称バンド行列の格納方法

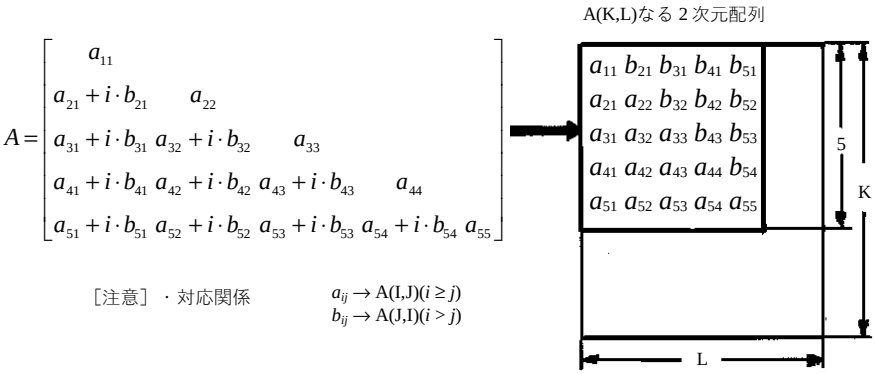


図 2.10　エルミート行列の格納方法

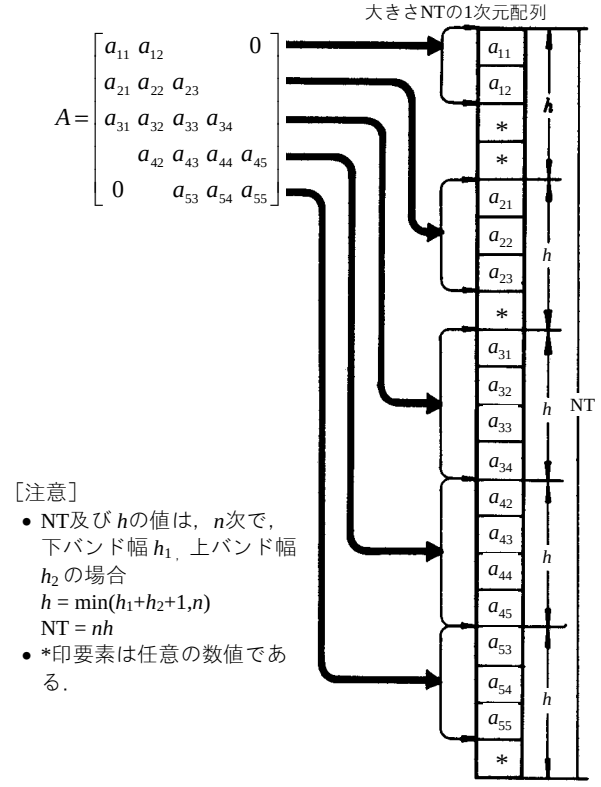


図 2.12　非対称バンド行列の格納方法

(2)　ベクトル

図 2.13 のように格納する.

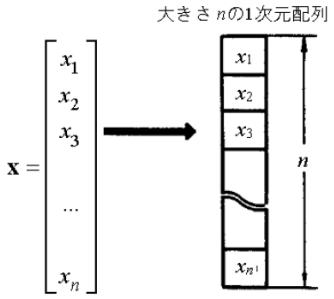


図 2.13　ベクトルの格納方法

(3)　代数方程式の係数

代数方程式の一般式を，(2.16) のように表す.

$$a_0x^n + a_1x^{n-1} + \ldots + a_{n-1}x + a_n = 0 \qquad (2.16)$$

この係数をベクトル要素として，図 2.14 のように格納する.

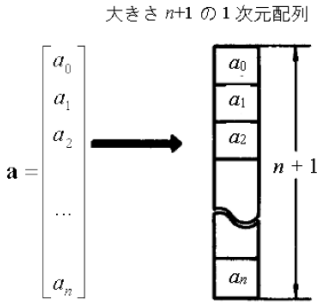


図 2.14　代数方程式の係数の格納方法

(4)　近似多項式の係数

近似式としての多項式の一般式を，(2.17)のように表す.

$$P_n(x) = c_0 + c_1x + c_2x^2 + \ldots + c_nx^n \qquad (2.17)$$

この係数をベクトル要素として，図 2.15 のように格納する.

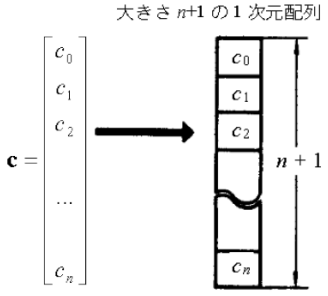


図 2.15 近似多項式の係数の格納方法

2.9　丸め誤差の単位

SSL II においては，丸め誤差の単位 (unit round off)を各所で用いている.

丸め誤差の単位は，浮動小数点演算の誤差解析における基本的な概念である.

丸め誤差の単位 u は，次のように定義されている.

定義

正規化された (normalized) M 進法 L 桁の浮動小数点演算における，丸め誤差の単位 u は,

$u = M^{1-L} / 2,$　　（四捨五入演算）

$u = M^{1-L},$　　（切り捨て演算）

である.

SSL II においては，この丸め誤差の単位を収束判定，零判定などに用いている.

（「付録 1　補助サブルーチン」AMACH 参照）

浮動小数点演算の誤差解析については，次の文献を参照されたい.

① Yamashita, S.
On the Error Estimation in Floating-point Arithmetic.
Information Processing in Japan Vol. 15, pp.935-939, 1974

② Wilkinson, J.H.
Rounding Errors in Algebraic Process,
Her Britannic Majesty's Stationery Office, London 1963

2.10　累和計算

数値計算の多くの計算においては，累和計算がしばしば現れる．例えば，連立 1 次方程式における積和計算や,各種ベクトル演算における内積計算などに現れる.

浮動小数点演算においては，その誤算解析の理論より，演算過程での有効数字を保持するためには，この累和計算を正確に行うことが必要である.

SSL II においては，原則的に随所に現れる累和計算を精度を上げて計算することにより，丸め誤差の影響を少なくしている.

（「付録 1　補助サブルーチン」ASUM 参照）.

2.11　計算機定数

本マニュアルでは，ハードウェア上の制約による計算機定数については，記号を用いて表現している．以下に，その記号を定義する．

- 浮動小数点数の正の最大値　(fl_{max})
  （「付録 1　補助サブルーチン」AFMAX 参照）
- 浮動小数点数の正の最小値　(fl_{min})
  （「付録 1　補助サブルーチン」AFMIN 参照）
- 三角関数（sin,cos）の引数の上限値　(t_{max})

引数の上限値		適 用 例
単精度	8.23×10^{5}	FACOM M シリーズ FACOM S シリーズ
倍精度	3.53×10^{15}	SX/G 100 , 200 シリーズ FM シリーズ

第 3 章
線型計算

3.1 概要

線型計算では，行列の構造と問題などに従って，表 3.1 のように体系化されている．

表 3.1 線型計算の体系

構 造	問 題	説明箇所
行列格納モードの変換		3.2
密行列	行列操作	3.3
	連立 1 次方程式・逆行列	3.4
	最小二乗解	3.5
バンド行列	行列操作	3.3
	連立 1 次方程式	3.4

この体系の背景には，密行列は比較的低次の行列，バンド行列は比較的高次の行列を扱うことを想定している．

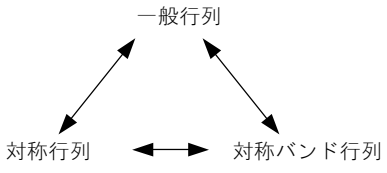
すなわち，一般には次数が高い行列はバンド行列となる傾向があり,その場合に適切な処理（不要な領域，演算を削除する等）を行えるようになっている．

現実的には，行列の格納方法（格納モード）が異なるため，いずれの行列として扱うかをみきわめてから使用していただきたい．

詳細は，各説明箇所で述べる．

3.2 行列格納モードの変換

モード変換では，次のとおり行列格納モードを変換する．



行列の格納方法は，その構造と形式に応じて異なる．

例えば，実対称行列では下三角部分並びに対角部分だけ格納するようになっている．（詳細は，2.8 節参照．）

したがって，連立 1 次方程式を解く場合でも，固有の格納方法に対応して，サブルーチンが用意されている．

そこで，ある格納方法で行列が与えられているとき，異なった格納方法に対応したサブルーチンを用いる場合は，格納方法をあらかじめ変換する必要がある．

SSL II では，このために表 3.2 に示す変換用サブルーチンが用意されている．

表 3.2 モード変換のサブルーチン

変換前＼変換後	一般モード	対称行列用圧縮モード	対称バンド行列用圧縮モード
一般モード		CGSM (A11-10-0101)	CGSBM (A11-40-0101)
対称行列用圧縮モード	CSGM (A11-10-0201)		CSSBM (A11-50-0101)
対称バンド行列用圧縮モード	CSBGM (A11-40-0201)	CSBSM (A11-50-0201)	

3.3 行列操作

行列操作では，次の基本演算を扱う．

- 行列と行列の和・差　　$\boldsymbol{A} \pm \boldsymbol{B}$
- 行列とベクトルの積　　$\boldsymbol{Ax}$
- 行列と行列の積 $\boldsymbol{AB}$

SSL II では，行列操作のために表 3.3 に示すサブルーチンが用意されている．

(1) 使用上の注意

a. 行列とベクトルの積

$$\boldsymbol{y} = \boldsymbol{Ax}$$

表 3.3　行列操作のサブルーチン

A ＼ *B* 又は *x*		実行例	対称行列	ベクトル
実行例	和	AGGM (A21-11-0101)		
	差	SGGM (A21-11-0201)		
	積	MGGM (A21-11-0301)	MGSM (A21-11-0401)	MAV (A21-13-0101)
複素行列	積			MCV (A21-15-0101)
対称行列	和		ASSM (A21-12-0101)	
	差		SSSM (A21-12-0201)	
	積	MSGM (A21-12-0401)	MSSM (A21-12-0301)	MSV (A21-14-0101)
バンド行列	積			MBV (A51-11-0101)
対称バンド行列	積			MSBV (A51-14-0101)

を行うサブルーチンは連立 1 次方程式において，近似解に対する残差ベクトルを求める場合を意図して用意されている．すなわち，近似解の反復改良や，方程式の解を反復解法により求める場合に利用することができる．

3.4　連立 1 次方程式・逆行列

ここでは，次の問題を扱う．

- 連立 1 次方程式

  $\boldsymbol{Ax} = \boldsymbol{b}$

  を解く．$\boldsymbol{A}$ は $n \times n$ の行列．$\boldsymbol{x}$ ，$\boldsymbol{b}$ は n 次元ベクトル．
- 行列 $\boldsymbol{A}$ の逆行列 $\boldsymbol{A}^{-1}$ を求める．
- 行列 $\boldsymbol{A}$ の行列式 det ($\boldsymbol{A}$) を求める．

SSL II には，これらの問題を解くために，次の四つの基本機能を持つサブルーチンが，行列の区分に応じて用意されている（コンポーネントルーチン）．

a. 係数行列の分解 (numeric decomposition).
b. 分解された要素に基づく求解 (solve).
c. 解の反復改良 (accuracy solve).
d. 分解された要素に基づく行列の逆転.

これらを組み合わせて用いれば，連立 1 次方程式を解くこと，逆行列を求めること及び，行列式を求めることができる．

連立 1 次方程式

方程式の解は，次のように順次コンポーネントルーチンを呼び出せば求めることができる．

```
   …
CALL a．のコンポーネントルーチン
CALL b．のコンポーネントルーチン
   …
```

逆行列

逆行列は，次のように順次コンポーネントルーチンを呼び出せば求めることができる．

```
   …
CALL a．のコンポーネントルーチン
CALL d．のコンポーネントルーチン
   …
```

ただし，バンド行列の逆行列は一般には密行列となり，逆行列を求めることは計算上得策ではない．したがって，SSL II には，バンド行列の逆行列を求めるコンポーネントルーチンは用意されていない．

行列式

行列式の値を求めるコンポーネントルーチンは用意されていないが，a. の コンポーネントルーチンにより分解された結果を用いて求めることができる．

このように，問題に応じてコンポーネントルーチンを組合わせればよいが，SSL II ではそれを標準的に使う場合も考慮して結合したルーチン（標準ルーチン）も用意されている．

通常は，この標準ルーチンを用いればよい．

表 3.4 に，行列の種類に応じたサブルーチンを示す．

(1)　使用上の注意

a. 解の反復改良について
方程式の解を標準ルーチンにより求めた後，その解を初期近似解と見なし反復改良して精度を上げる場合には，続けて c.のコンポーネントルーチンを呼び出せばよい．
また，この c.のコンポーネントルーチンにより，初期近似解の推定精度を調べることもできる．

b. 逆行列について
連立 1 次方程式を解くために，逆行列を求めて計算することは，一般的には得策ではない．すなわち,連立 1 次方程式 (3.1) において，

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (3.1)$$

まず，逆行列 $\boldsymbol{A}^{-1}$ を求め，次に左側から $\boldsymbol{b}$ に掛ける (3.2) なる操作により解を求めることは避けるべきである．

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} \qquad (3.2)$$

通常は，$\boldsymbol{A}$ を LU 分解し，(3.3) なる操作（前進代入と後退代入）により解を求める．

$$\left.\begin{aligned} \boldsymbol{Ly} &= \boldsymbol{b} \\ \boldsymbol{Ux} &= \boldsymbol{y} \end{aligned}\right\} \quad (3.3)$$

(3.3)　による方が，演算速度・精度の点ですぐれている．ちなみに (3.2) と(3.3) の各々の場合における，計算で必要となる乗算回数の概算を比較すると(3.4) となる．

$$\left.\begin{aligned} &\text{(3.2) の場合} \quad n^3 + n^2 \\ &\text{(3.3) の場合} \quad n^3/3 \end{aligned}\right\} \quad (3.4)$$

したがって，逆行列は本質的にそれを必要とするときだけ求めること．

c. 同一の係数行列を持つ方程式について
係数行列 $\boldsymbol{A}$ が同一で，定数ベクトル $\boldsymbol{b}$ だけ異なる複数組の連立 1 次方程式 (3.5) を解く場合，

$$\left.\begin{aligned} \mathbf{Ax}_1 &= \mathbf{b}_1 \\ \mathbf{Ax}_2 &= \mathbf{b}_2 \\ &\cdots \\ \mathbf{Ax}_m &= \mathbf{b}_m \end{aligned}\right\} \quad (3.5)$$

各々の方程式で行列 $\boldsymbol{A}$ を分解することは得策ではない．この場合は，最初の方程式を解く時にだけ $\boldsymbol{A}$ を分解し，それ以降の方程式を解く時には，　(3.3) に準じて前進代入，後退代入だけを行えばよい．
標準ルーチンにおいては，$\boldsymbol{A}$ の分解から行うか，$\boldsymbol{A}$ の分解は省略するかを制御するパラメタ ISW が用意されている．

(2)　内部処理上の留意点

ここでは，内部処理の立場から，使用上の留意点について述べる．

a.　クラウト法，変形コレスキー法
SSL II において，正則な非対称行列の分解の方法として，クラウト法を採用している．
正則な非対称行列の分解 (3.6) の方法として，通常ガウス消去法とクラウト法が知られている．

$$\boldsymbol{A} = \boldsymbol{LU} \quad (3.6)$$

ここで，　$\boldsymbol{L}$ は下三角行列，
$\boldsymbol{U}$ は上三角行列である．

両者は，その計算順序が異なる．すなわち，今 $\boldsymbol{L}$ 又は $\boldsymbol{U}$ の一つの要素に注目した場合，前者は中間的な値をもち基本的には分解がすべて終了した時点でその要素が得られるのに対して，後者は分解の過程でその要素が得られる．数値的には後者は積和演算により分解要素が求められるので，そこでの丸め誤差の影響が少なくなるように考慮すれば，一般には前者より精度よく分解できる．更に後者の方が前者より，分解過程でのデータの参照順序の局所性が高い．したがって，　SSL II では後者クラウト法を採用し，積和部分の精度を上げて行うことにより，丸めの誤差の影響を少なくしている．
一方，正値対称行列に対しては，変形コレスキー法を採用している．すなわち，(3.7)なる分解を行う．

表 3.4　連立 1 次方程式・逆行列のサブルーチン

問題，基本機能 / 行列の種類	標準ルーチン名	コンポーネントルーチン名			
	連立 1 次方程式	a.	b.	c.	d.
実行例	LAX (A22-11-0101)	ALU (A22-11-0202)	LUX (A22-11-0302)	LAXR (A32-11-0401)	LUIV (A22-11-0602)
複素行列	LCX (A22-15-0101)	CLU (A22-15-0202)	CLUX (A22-15-0302)	LCXR (A22-15-0401)	CLUIV (A22-15-0602)
対称行列	LSIX (A22-21-0101)	SMDM (A22-21-0202)	MDMX (A22-21-0302)	LSIXR (A22-21-0401)	
正値対称行列	LSX (A22-51-0101)	SLDL (A22-51-0202)	LDLX (A22-51-0302)	LSXR (A22-51-0401)	LDIV (A22-51-0702)
バンド行列［3 項行列］	LBX1 (A52-11-0101) [LTX (A52-11-0501)]	BLU1 (A52-11-0202)	BLUX1 (A52-11-0302)	LBX1R (A52-11-0401)	
実対称バンド行列	LSBIX (A52-21-0101)	SBMDM (A52-21-0202)	BMDMX (A52-21-0302)		
正値対称バンド行列［正値対称 3 項行列］	LSBX (A52-31-0101) [LSTX (A52-31-0501)]	SBDL (A52-31-0202)	BDLX (A52-31-0302)	LSBXR (A52-31-0401)	

表 3.5　分解された行列

行列の種類	分解された行列の内容
実行例	$\boldsymbol{PA} = \boldsymbol{LU}$ $\boldsymbol{L}$: 下三角行列 $\boldsymbol{U}$: 単位上三角行列 なお，$\boldsymbol{P}$ は置換行列
正値対称行列	$\boldsymbol{A} = \boldsymbol{LDL}^T$ $\boldsymbol{L}$: 単位下三角行列 $\boldsymbol{D}$: 対角行列 なお，対角行列は計算の手間を少なくするために $\boldsymbol{D}^{-1}$ として出力される．

$$\boldsymbol{A} = \boldsymbol{LDL}^T \tag{3.7}$$

ここで，　$\boldsymbol{L}$ は下三角行列，
$\boldsymbol{D}$ は対角行列である．

なお，分解された行列は，表 3.5 のように与えられるので注意されたい．

b.　ピボッティングとスケーリング

正則な実行例の分解 (3.6) を数値的に行うことを考える．例えば (3.8) の行列の分解を考えると，

$$\mathbf{A} = \begin{bmatrix} 0.0 & 1.0 \\ 2.0 & 0.0 \end{bmatrix} \tag{3.8}$$

このままの状態では，LU 分解は不可能である．また，(3.9) の場合には，

$$\mathbf{A} = \begin{bmatrix} 0.0001 & 1.0 \\ 1.0 & 1.0 \end{bmatrix} \tag{3.9}$$

3 桁程度の浮動小数点演算では，まったく不安定である．このような事態は，行列の条件が悪い場合にはしばしば発生することであるが，これは，絶対値最大要素をピボットとして選択するピボッティングを行うことにより，回避できることが知られている．

(3.9) の場合は，1 行目と 2 行目の各要素を交換することにより回避できる．

さて，このピボッティングを行うためは，絶対値最大のものを選択する方法に一意性が必要である．ところで，ある行のすべての要素に大きな定数を掛ければ，どの要素でも（それが零でないなら）他の行のどの要素よりも絶対値を大きくすることができる．

したがって，ピボッティングで，絶対値最大の大小でピボットを決めることが有効に働くように，行及び列の均衡化 (equilibration) を図る必要がある．

SSL II では，“行の均衡化を考慮した部分ピボッティング”を採用している．なお，行の均衡化は，分解する行列の各行において，絶対値最大が 1 となるようなスケーリングにより行うが，実際には各要素の値を変更することなく，ピボット選択時にスケーリングファクターを考慮することで対処している．

また，ピボッティングによる行の入換えを行うので，その履歴をトランスポジションベクトルとして保存する．

この部分ピボッティングを伴う行列の分解は，(3.10) により行列表現される．

$$\boldsymbol{PA} = \boldsymbol{LU} \tag{3.10}$$

ここで $\boldsymbol{P}$ は，行の入換えを行う置換行列である．

c. トランスポジションベクトル

b. で述べたように，ピボッティングに伴う行の入換えは，(3.10) の置換行列 $\boldsymbol{P}$ で示される．SSL II では，この置換行列 $\boldsymbol{P}$ を直接保存することなく，トランスポジションベクトルとしてあつかう．

すなわち，分解の j 段階 (j = 1,.., n) において，第 i 行 ($i \geq j$) をピボット行として選択した場合には，分解過程の行列の i 行と j 行とを入れ換え，トランスポジションベクトルの第 j 要素を i とする．

d. ピボット要素の相対零判定

分解の過程で，ピボット要素が零となった場合，若しくは零に近い値となった場合は，行列が非正則 (singular) である可能性が大きい．この場合，有限の桁数の演算で零に近いピボットを用いて計算を続行することは，解が不正解になることが予想される．

SSL II では，このような状態になった場合の処理の続行又は，打ち切りを制御するために，パラメタ EPSZ を用意している．

すなわち， EPSZ = 10^{-s} を設定すると，ピボットとなるべき要素の計算で，10 進 s 桁以上の桁落ちを起こした場合に，そのピボットを相対的に零と見なす．

e. 解の反復改良

連立 1 次方程式 (3.11) を数値的に解くことは，方程式の解でなく，近似値を求めることである．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.11}$$

SSL II では，求まった近似値をできるだけ解に近づけられるように，近似解を改良できるサブルーチンが用意されている．

SSL II では，方程式の近似解 $\boldsymbol{x}^{(1)}$ を反復改良するために

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{3.12}$$
$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{3.13}$$
$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{3.14}$$

ここで，s = 1, 2, …

なる計算を一定の収束条件を満たすまで繰り返す．

ここで，(3.12), (3.13) , (3.14)の計算を正確に行うなら，$\boldsymbol{x}^{(2)}$ が (3.11) の正確な解であることは次のとおり導かれる．

$$\boldsymbol{A}\boldsymbol{x}^{(2)} = \boldsymbol{A}\left(\boldsymbol{x}^{(1)} + \boldsymbol{d}^{(1)}\right) = \boldsymbol{A}\boldsymbol{x}^{(1)} + \boldsymbol{r}^{(1)} = \boldsymbol{b}$$

実際は，(3.12), (3.13) , (3.14)において丸め誤差が発生する．そこで，SSL II では，(3.13)から修正量 $\boldsymbol{d}^{(s)}$ 計算するときだけ丸め誤差が発生すると仮定すれば，その $\boldsymbol{d}^{(s)}$ は次の方程式の正確な解と考えられる．ただし，$\boldsymbol{E}$ は誤差行列である．

$$(\boldsymbol{A} + \boldsymbol{E})\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{3.15}$$

(3.12), (3.14) , (3.15) から，次の関係が得られる．

$$\boldsymbol{x}^{(s+1)} - \boldsymbol{x} = \left[\boldsymbol{I} - (\boldsymbol{A} + \boldsymbol{E})^{-1}\boldsymbol{A}\right]^{s}\left(\boldsymbol{x}^{(1)} - \boldsymbol{x}\right) \tag{3.16}$$
$$\boldsymbol{r}^{(s+1)} = \left[\boldsymbol{I} - \boldsymbol{A}(\boldsymbol{A} + \boldsymbol{E})^{-1}\right]^{s}\boldsymbol{r}^{(1)} \tag{3.17}$$

(3.16), (3.17) の式から分かるように，以下の条件を満たせば，s→∞ のとき $\boldsymbol{x}^{(s+1)}$ と $\boldsymbol{r}^{(s+1)}$ は，それぞれ解 $\boldsymbol{x}$ と $\boldsymbol{0}$ に収束する．

$$\left\|\mathbf{I} - (\mathbf{A} + \mathbf{E})^{-1}\mathbf{A}\right\|_{\infty}, \left\|\mathbf{I} - \mathbf{A}(\mathbf{A} + \mathbf{E})^{-1}\right\|_{\infty} < 1$$

すなわち，次の関係を満足すれば，改良解が得られる．

$$\|\boldsymbol{E}\|_{\infty} \cdot \left\|\boldsymbol{A}^{-1}\right\|_{\infty} < 1/2 \tag{3.18}$$

以上が反復改良の原理のあらましである．

SSL II では，1 回の反復あたり最低 2 進 1 桁程度以下しか精度が改善されなくなるまで反復を繰り返す．なお，原理から，行列 $\boldsymbol{A}$ の条件が非常に悪い場合には，$\left\|\boldsymbol{A}^{-1}\right\|_{\infty}$ が大きくなって改良解が得られないことが分かる．

f. 近似解の精度推定

連立 1 次方程式の近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)}$ ($= \boldsymbol{x}^{(1)} - \boldsymbol{x}$) とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_{\infty} / \left\|\boldsymbol{x}^{(1)}\right\|_{\infty}$ である．

そこで $\boldsymbol{d}^{(1)}$ と $\boldsymbol{e}^{(1)}$ の関係を調べてみると (3.12), (3.15) より，(3.19) のとおりになる．ただし， $\boldsymbol{E}$ は e. で述べた誤差行列である．

$$\boldsymbol{d}^{(1)} = (\boldsymbol{A} + \boldsymbol{E})^{-1}\left(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(1)}\right) = \left(\boldsymbol{I} + \boldsymbol{A}^{-1}\boldsymbol{E}\right)^{-1}\boldsymbol{e}^{(1)} \tag{3.19}$$

いま，反復法が (3.18) なる条件を十分満足し収束するならば， $\left\|\boldsymbol{d}^{(1)}\right\|_{\infty}$ と $\left\|\boldsymbol{e}^{(1)}\right\|_{\infty}$ は，ほぼ等しいと考えられる．

よって，近似解の相対誤差を $\left\|\boldsymbol{d}^{(1)}\right\|_{\infty} / \left\|\boldsymbol{x}^{(1)}\right\|_{\infty}$ によって推定できる．

3.5　最小二乗解

ここでは，$m \times n$ 行列を扱う次の問題をとりあげる．

- 最小二乗解
- 最小二乗最小ノルム解
- 一般逆行列
- 特異値分解

SSL II では，このために，表 3.6 に示されるサブルーチンが用意されている．

表 3.6　$m \times n$ 行列を扱うサブルーチン

機能＼行列	実行列
最小二乗解	LAXL (A25-11-0101)
最小二乗解の反復改良	LAXLR (A25-11-0401)
最小二乗最小ノルム解	LAXLM (A25-21-0101)
一般逆行列	GINV (A25-31-0101)
特異値分解	ASVD1 (A25-31-0201)

最小二乗解

最小二乗解は，連立 1 次方程式

$\boldsymbol{Ax} = \boldsymbol{b}$

$\boldsymbol{A}$ は $m \times n$ 行列 $(m \geq n, \text{rank}(\boldsymbol{A}) = n)$，$x$ は n 次元ベクトル，$\boldsymbol{b}$ は m 次元ベクトル．

において，$\|\boldsymbol{Ax} - \boldsymbol{b}\|_2$ が最小となる解 $\tilde{\boldsymbol{x}}$ である．

SSL II では，この問題のために，機能を次の二つに区分し，それぞれのサブルーチンを用意している．

a.　最小二乗解を求める．
b.　最小二乗解を反復改良する．

最小二乗最小ノルム解

最小二乗最小ノルム解は，連立 1 次方程式

$\boldsymbol{Ax} = \boldsymbol{b}$

$\boldsymbol{A}$ は $m \times n$ 行列，$\boldsymbol{x}$ は n 次元ベクトル，$\boldsymbol{b}$ は m 次元ベクトル．

において，$\|\boldsymbol{Ax} - \boldsymbol{b}\|_2$ が最小となるベクトル $\boldsymbol{x}$ の中で，$\|x\|_2$ が最小となる解 $\boldsymbol{x}^+$ である．

一般逆行列

$m \times n$ 行列 $\boldsymbol{A}$ 対して，$n \times m$ 行列 $\boldsymbol{X}$ が (3.20) を満たすとき，これを $\boldsymbol{A}$ のムーアペンローズ (Moore-Penrose) の一般逆行列という．

$$\left.\begin{aligned} &\boldsymbol{AXA} = \boldsymbol{A} \\ &\boldsymbol{XAX} = \boldsymbol{X} \\ &(\boldsymbol{AX})^{\mathrm{T}} = \boldsymbol{AX} \\ &(\boldsymbol{XA})^{\mathrm{T}} = \boldsymbol{XA} \end{aligned}\right\} \tag{3.20}$$

このような一般逆行列は常に存在し一意的である．なお，行列 $\boldsymbol{A}$ の一般逆行列 $\boldsymbol{X}$ を $\boldsymbol{A}^+$ と表す．

SSL II では，以上のような一般逆行列を計算する．

SSL II で扱う行列は，縦長行列 $(m > n)$，正方行列 $(m = n)$ あるいは横長行列 $(m < n)$ のどれであってもよい．

特異値分解

$m \times n$ の実行例 $\boldsymbol{A}$ が与えられたとき，$\boldsymbol{A}$ の (3.21) なる分解を特異値分解という．

$$\boldsymbol{A} = \boldsymbol{U}_0 \boldsymbol{\Sigma}_0 \boldsymbol{V}^{\mathrm{T}} \tag{3.21}$$

ここで $\boldsymbol{U}_0$ 及び $\boldsymbol{V}$ は，$m \times m$ 及び $n \times n$ の直交行列，$\boldsymbol{\Sigma}_0$ は $\boldsymbol{\Sigma}_0 = \text{diag}(\sigma_i)$，$\sigma_i \geq 0$ なる $m \times n$ の対角行列であり，σ_i を $\boldsymbol{A}$ の特異値という．

いま，$m \geq n$ と仮定する．

$\boldsymbol{\Sigma}_0$ が $m \times n$ の対角行列であることから，(3.21) の積 $\boldsymbol{U}_0\ \boldsymbol{\Sigma}_0\ \boldsymbol{V}^{\mathrm{T}}$ の計算値に対して，$\boldsymbol{U}_0$ の先頭 n 列だけが寄与していることが分かる．つまり，$\boldsymbol{U}_0$ は本質的に $m \times n$ の行列でよいわけである．このような行列を $\boldsymbol{U}$ とする．合わせて，$\boldsymbol{\Sigma}_0$ の下側の $(m-n) \times n$ の零行列を除いた $n \times n$ の行列を $\boldsymbol{\Sigma}$ とする．行列 $\boldsymbol{U}$ と $\boldsymbol{\Sigma}$ を利用するならば，m が n よりかなり大きいと，それなりに記憶領域の節約ができる．その上，実用上，$\boldsymbol{U}_0$ 及び $\boldsymbol{\Sigma}_0$ より，$\boldsymbol{U}$ 及び $\boldsymbol{\Sigma}$ の方が便利である．

$m \times n$ の場合の以上の事柄が，$m < n$ の場合にも準じて言える．

SSL II では，以上の事柄を考慮して，(3.22) なる特異値分解を行っている．

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}} \tag{3.22}$$

ここで，$l = \min(m, n)$ として，$\boldsymbol{U}$ は $m \times l$ の行列，$\boldsymbol{\Sigma}$ は $\boldsymbol{\Sigma} = \text{diag}(\sigma_i)$，$\sigma_i \geq 0$ なる l 次の対角行列，$\boldsymbol{V}$ は $n \times l$ の行列である．

かつ，

$l = n$ $(m \geq n)$ のとき，

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{V}^{\mathrm{T}}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^{\mathrm{T}} = \boldsymbol{I}_n$$

$l = m$ $(m < n)$ のとき，

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^{\mathrm{T}} = \boldsymbol{V}^{\mathrm{T}}\boldsymbol{V} = \boldsymbol{I}_m$$

である．

行列 $\boldsymbol{A}$ を特異値分解することにより計算される行列 $\boldsymbol{U}$, $\boldsymbol{V}$ 及び $\boldsymbol{\Sigma}$ は，次のような意味と用途を持つ．（詳細は，参考文献 [86] pp.212-264 参照）．

- 特異値 σ_i, $i = 1, 2, ..., l$ は，行列 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$，及び $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ の固有値の，大きい方から数えて l 番目までのも

のの正平方根である．また，$\boldsymbol{V}$ の第 i 列は，固有値σ_i^2 に対する $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ の固有ベクトルであり，$\boldsymbol{U}$ の第 i 列は，固有値 σ_i^2 に対する $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ の固有ベクトルである．これらを確かめるためには，$\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\mathrm{T}}$ を，(3.22) の左及び右から乗じ，$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{V}^{\mathrm{T}}\boldsymbol{V} = \boldsymbol{I}_l$ を利用して，

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{\Sigma}^{2} \tag{3.23}$$
$$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Sigma}^{2} \tag{3.24}$$

となることを見ればよい．

- $\boldsymbol{A}$ の条件数
  すべての i について，$\sigma_i > 0$, i=1, 2, ..., l であるとき，$\boldsymbol{A}$ の条件数は次の式で与えられる．

$$\mathrm{cond}(\boldsymbol{A}) = \sigma_1 / \sigma_l \tag{3.25}$$

- $\boldsymbol{A}$ の階数
  $\sigma_r > 0$, $\sigma_{r+1} = \cdots = \sigma_l = 0$, であるとき，$\boldsymbol{A}$ の階数は，

$$\mathrm{rank}\,(\boldsymbol{A}) = r \tag{3.26}$$

で与えられる．

- 同次方程式 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$, $\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{y} = \boldsymbol{0}$ の基本解
  同次方程式 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$ 及び $\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{y} = \boldsymbol{0}$ の $\boldsymbol{0}$ でない一次独立な解の組は，$\sigma_i = 0$ なる特異値に対応する $\boldsymbol{V}$ の列の組及び $\boldsymbol{U}$ の列の組によって，それぞれ与えられる．このことは，$\boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{\Sigma}$ 及び $\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{U} = \boldsymbol{V}\boldsymbol{\Sigma}$ から 容易にわかる．
- $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ の最小二乗最小ノルム解
  上記の解 $\boldsymbol{x}$ は，$\boldsymbol{A}$ の特異値分解を用いて

$$\boldsymbol{x} = \boldsymbol{V}\boldsymbol{\Sigma}^{+}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{b} \tag{3.27}$$

と表される．ただし，対角行列 $\boldsymbol{\Sigma}^{+}$ は

$$\boldsymbol{\Sigma}^{+} = \mathrm{diag}(\sigma_1^{+}, \sigma_2^{+}, ..., \sigma_l^{+}) \tag{3.28}$$
$$\sigma_i^{+} = \begin{cases} 1/\sigma_i, & \sigma_i > 0 \\ 0, & \sigma_i = 0 \end{cases} \tag{3.29}$$

で定義される．LAXLM の手法概要を参照のこと．

- 一般逆行列
  $\boldsymbol{A}$ の一般逆行列 $\boldsymbol{A}^{+}$ は，特異値分解により，

$$\boldsymbol{A}^{+} = \boldsymbol{V}\boldsymbol{\Sigma}^{+}\boldsymbol{U}^{\mathrm{T}} \tag{3.30}$$

で与えられる．

(1)　使用上の注意

a.　連立 1 次方程式と係数行列の階数
$m \times n$ 行列を係数に持つ連立 1 次方程式 $(\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b})$ を解く場合，係数行列 $\boldsymbol{A}$ の行数と列数の大小関係や階数に制約されないで，最小二乗最小ノルム解は求められる．いわば最小二乗最小ノルム解はいかなる方程式にも適用可能な解である．しかし，この解は求める過程の計算量が多いという欠点を持つので，連立 1 次方程式の係数行列が縦長行列でかつ最大階数 (full rank, rank ($\boldsymbol{A}$) = n) であることが明らかである場合は，最小二乗解のサブルーチンを使用した方が計算量は少なく合理的である．ちなみに，このような場合，最小二乗解が理論上最小二乗最小ノルム解である．

b.　最小二乗最小ノルム解と一般逆行列
$m \times n$ 行列 $\boldsymbol{A}$ ($m \geq n$ 又は $m < n$, rank($\boldsymbol{A}$) $\neq 0$) の連立 1 次方程式 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ は，通常解が一意に定まらないが，最小二乗最小ノルム解は常に存在し一意に定まる．この解は係数行列 $\boldsymbol{A}$ の一般逆行列 $\boldsymbol{A}^{+}$ を求めるなら，$\boldsymbol{x} = \boldsymbol{A}^{+}\boldsymbol{b}$ で計算できる．しかし，この方法では，計算量が多く得策でない．この場合，最小二乗最小ノルム解を求める専用サブルーチンを用いるなら処理が速い．特に，このサブルーチンには，以降に説明するとおり，同一の係数行列を持つ方程式を出来るだけ少ない計算量で解いたり，単一の方程式を効率よく解くことを利用者が指示できるパラメタ ISW を用意しており，幅広い用途がある．

c.　同一の係数行列を持つ方程式について
連立 1 次方程式の最小二乗解や最小二乗最小ノルム解は，通常次の手順により求められる．

- 係数行列の分解
  最小二乗解のときは，三角行列化を，最小二乗最小ノルム解のときは，特異値分解を行う．
- 求解
  最小二乗解のときは，後退代入を行い，最小二乗最小ノルム解のときは，行列やベクトルの乗算を行う．

ところが，同一の係数行列を持つ複数組の方程式の最小二乗解若しくは最小二乗最小ノルム解を求める場合．

$$\begin{aligned} &\boldsymbol{A}\boldsymbol{x}_1 = \boldsymbol{b}_1 \\ &\boldsymbol{A}\boldsymbol{x}_2 = \boldsymbol{b}_2 \\ &\quad\vdots \\ &\boldsymbol{A}\boldsymbol{x}_m = \boldsymbol{b}_m \end{aligned}$$

各方程式で上記手順を繰り返し計算することは得策でない．この場合，最初の方程式を解くときにだけ，$\boldsymbol{A}$ を分解し，それ以降は求解だけの手順を繰り返し適用すれば，計算量が節約できる．
SSL II には，$\boldsymbol{A}$ の分解から行うか，分解を省略するかどうかを制御するパラメタ ISW が用意されている．

d.　特異値の求め方
特異値は，(3.31) のように特異値分解すれば得られる．

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}} \tag{3.31}$$

しかし，(3.31) に現れるとおり，数学的には，特異値からなる行列 $\boldsymbol{\Sigma}$ の他に行列 $\boldsymbol{U}$ 及び $\boldsymbol{V}$ が存在する．ところで，特異値分解のための計算量は多いので，利用者にとって行列 $\boldsymbol{U}$ 及び $\boldsymbol{V}$ が不要なときは計算されないことが望ましい．そこで，SSL II には，行列 $\boldsymbol{U}$ 又は $\boldsymbol{V}$ も求めるかどうかを制御するパラメタ ISW が用意されている．また SSL II で扱う行列は，縦長行列，正方行列あるいは，横長行列のどれであっても良く，制約がない．

第 4 章
固有値固有ベクトル

4.1 概要

固有値問題は，問題の種類 **($Ax=\lambda x$, $Ax=\lambda Bx$)** ，行列の構造（密，バンド），行列の型（実，複素）及び行列の形式（対称，非対称）により，表 4.1 のように体系化される．したがって，行列 A の種類に応じ，それぞれの該当箇所を参照していただきたい．

表 4.1 固有値問題の体系一覧

行列の構造	問題の種類	行列の型・形式	説明箇所
密行列	$Ax=\lambda x$	実行列 複素行列 実対称行列 エルミート行列	4.2 4.3 4.4 4.5
	$Ax=\lambda Bx$	実対称行列	4.7
バンド行列	$Ax=\lambda x$	実対称バンド行列	4.6
	$Ax=\lambda Bx$	実対称バンド行列	4.8

［注意］ 実対称 3 重対角行列については，実対称行列の項を参照すること．

4.2 実行列の固有値固有ベクトル

実行列の固有値固有ベクトルを求めるとき，SSL II の各サブルーチンをどのように用いるか標準的な例をあげて簡単に説明する．

SSL II には，実行列の固有値固有ベクトルを 1 度に求めることができる標準ルーチンと，細分化された各機能を果たすコンポーネントルーチンが用意されている．（表 4.2 参照）．

ここでは具体的に，利用者の問題を

□ すべての固有値を求めるとき．

□ すべての固有値固有ベクトル（あるいはすべての固有値と一部の固有ベクトル）を求めるとき．

に分け，続く(1)と (2)でコンポーネントルーチンと標準ルーチンの使い方を，それぞれの手順に従って，個々に説明する．

そして(3)では処理に関する若干の補足説明をする．

表 4.2 実行例の標準固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	実行列の固有値固有ベクトル	EIG1 (B21-11-0101)
コンポーネントルーチン	実行例の平衡化	BLNC (B21-11-0202)
	実行列の 実ヘッセンベルグ行列への変換	HES1 (B21-11-0302)
	実ヘッセンベルグ行列の固有値	HSQR (B21-11-0402)
	実ヘッセンベルグ行列の 固有ベクトル	HVEC (B21-11-0502)
	実行列の固有ベクトルへの 逆変換と正規化	HBK1 (B21-11-0602)
	実行列の固有ベクトルの正規化	NRML (B21-11-0702)

実行列の固有値固有ベクトルを求めるとき，コンポーネントルーチンを順次呼出すか，あるいは標準ルーチンを使うかは，利用者に任されている．通常は，使いやすさの面で後者を利用することを勧めたい．

ただし，指定された一部の固有値に対応する固有ベクトルを求めることは，前者によってだけ可能である．

(1) すべての固有値を求めるとき

次のパス①～③により，実行列 A のすべての固有値を求めることができる．

```
   …
CALL BLNC (A, K, N, DV, ICON)      ①
IF (ICON .EQ. 30000) GO TO 1000
CALL HES1 (A, K, N, PV, ICON)      ②
CALL HSQR (A, K, N, ER, EI, M,ICON)
                                   ③
IF (ICON .EQ. 20000) GO TO 2000
   …
```

① A の平衡化を行う．平衡化が不要なら直接②を呼ぶ．

② ハウスホルダー法により A をヘッセンベルグ行列に変換する．

③ 2 段 QR 法によりヘッセンベルグ行列の固有値，すなわち実行列 A の固有値を求める．

(2)　すべての固有値固有ベクトル（あるいはすべての固有値と一部の固有ベクトル）を求めるとき

次のパス①～⑤又は⑥により実行列 ***A*** のすべての固有値固有ベクトルを求めることができる．

```
      …
   CALL BLNC (A, K, N, DV, ICON)        ①
   IF (ICON .EQ. 30000) GO TO 1000
   CALL HES1 (A, K, N, PV, ICON)        ②
   DO 10 I = 1, N
   DO 10 J = 1, N
   AW (J, I) = A (J, I)
10 CONTINUE
   CALL HSQR (A, K, N, ER, EI, M,ICON)
                                        ③
   IF (ICON .GE. 20000) GO TO 2000
   DO 20 I = 1, M
   IND (I) = 1
20 CONTINUE
   CALL HVEC (AW, K, N, ER, EI, IND,
  *M, EV, MK, VW, ICON)                 ④
   IF (ICON .GE. 20000) GO TO 3000
   CALL HBK1 (EV, K, N, IND, M, A, PV,
  *DV, ICON)                            ⑤
      …
```

又は標準ルーチン

```
      …
   CALL EIG1 (A, K, N, MODE, ER, EI, EV, VW, ICON)
                                        ⑥
   IF(ICON .GE. 20000) GO TO 1000
      …
```

①②③は「(1)すべての固有値を求めるとき」と同じである．ただし，②実行後のパラメタ A を③に用いると A の内容が変わるので，④のために A を配列 AW に移しておく．

④　固有値に対応する固有ベクトルを逆反復法により求める．パラメタ IND はどの固有値に対応する固有ベクトルを求めるかの指標ベクトルであり，これによって利用者は，部分固有ベクトルを求めるための制御を行うことができる．

⑤　④で求めた固有ベクトルの逆変換を行う．あわせて正規化も行う．
①②の処理のため，④の固有ベクトルは，実行列 ***A*** のそれとはなっていない．したがって①②の逆変換を行って実行列 ***A*** の固有ベクトルを求める．
なお，正規化はパラメタ EV の各列（すなわち各固有ベクトル）のユークリッドノルムが 1 となるように行う．

⑥　**全固有値固有ベクトルを求めるとき，①～⑤までのパスをまとめて行うことができる．**ただし，固有ベクトルは逆反復法を用いずに，固有値が求まる迄の変換行列を順次掛け合わせることによって，すべての固有ベクトルすべてを 1 度に求めている．したがって，固有値が一つでも求まらない場合は，固有値ベクトルの計算は不可能となるが，一方固有値が近接根や重根となる場合においては，逆反復法にくらべ，より正しく固有ベクトルを求めることができる．
なお，⑥の標準ルーチンは，常に①の処理を行うわけではない．①の処理を行うかどうかは，パラメタ MODE によって指定することができる．

(3)　行列の平衡化について

固有値固有ベクトルの誤差を減少させるためには，実行列 ***A*** のノルムを小さくすればよい．パスに含まれる①の処理はこのためのもので，対角行列による相似変換により，***A*** のたがいに対応する行と列の絶対和を等しく（平衡化）しようとするものである．（したがって，実対称行列やエルミート行列は，この意味で既に平衡化されている．）

A に大きい要素や小さい要素が混在しているとき有効なことがあるので，その場合はできるだけ平衡化を行うことを勧めたい．平衡化の，パス全体の処理時間に占める割合は，特別な場合（***A*** の次数が小さい）を除いて，多くても 10%程度である．

4.3　複素行列の固有値固有ベクトル

複素行列の固有値固有ベクトルを求めるとき，SSL II の各サブルーチンをどのように用いるか標準的な例をあげて簡単に説明する．

SSL II には，複素行列に固有値固有ベクトルを一度に求めることができる標準ルーチンと，細分化された各機能を果すコンポーネントルーチンが用意されている（表 4.3 参照）．

表 4.3　複素行列の標準固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	複素行列の固有値固有ベクトル	CEIG2 (B21-15-0101)
コンポーネントルーチン	複素行列の平衡化	CBLNC (B21-15-0202)
	複素行列の複素ヘッセンベルグ行列への変換	CHES2 (B21-15-0302)
	複素ヘッセンベルグ行列の固有値	CHSQR (B21-15-0402)
	複素ヘッセンベルグ行列の固有ベクトル	CHVEC (B21-15-0502)
	複素行列の固有ベクトルへの逆変換	CHBK2 (B21-15-0602)
	複素行列の固有ベクトルの正規化	CNRML (B21-15-0702)

ここでは具体的に，利用者の問題を

□ すべての固有値を求めるとき.

□ すべての固有値固有ベクトルを求める（あるいは，すべての固有値と一部の固有ベクトルを求める）とき.

に分け，続く(1)と(2)でコンポーネントルーチンと標準ルーチンの使い方を，それぞれの手順に従って個々に説明する.

すべての固有値固有ベクトルを求めるとき，標準ルーチンだけを使用するか，又は各コンポーネントルーチンを組み合わせて使用するかは利用者に任されている．ただし，使い易さの点で，通常は前者の使用を勧める.

(1)　すべての固有値を求めるとき

次のパス①～③により，複素行列 ***A*** のすべての固有値を求めることができる.

```
CALL CBLNC (ZA, K, N, DV, ICON)     ①
IF (ICON. EQ. 30000) GO TO 1000
CALL CHES2 (ZA, K, N, IP, ICON)     ②
CALL CHSQR (ZA, K, N, ZE, M, ICON) ③
IF (ICON. GE. 15000) GO TO 2000
```

① ***A*** の平衡化を行う．平衡化が不要なら直接②を呼ぶ.

② 安定化基本相似変換により ***A*** を複素ヘッセンベルグ行列に変換する.

③ 複素 QR 法により複素ヘッセンベルグ行列の固有値，すなわち ***A*** の固有値を求める.

(2)　すべての固有値固有ベクトルを求めるとき（あるいは，すべての固有値と一部の固有ベクトルを求めるとき）

次のパス①～⑥又は⑦により複素行例 ***A*** のすべての固有値固有ベクトルを求めることができる.

```
   …
CALL CBLNC (ZA, K, N, DV, ICON)     ①
IF (ICON. EQ. 30000) GO TO 1000
CALL CHES2 (ZA, K, N, IP, ICON)     ②
DO 10 J = 1, N
DO 10 I = 1, N
ZAW (I, J) = ZA (I, J)
10 CONTINUE
   CALL CHSQR (ZA, K, N, ZE, M, ICON) ③
   IF (ICON. GE. 15000) GO TO 2000
   DO 20 I = 1, M
   IND (I) = 1
20 CONTINUE
   CALL CHVEC (ZAW, K, N, ZE, IND,
  *          M, ZEV, ZW, ICON)        ④
   IF (ICON. GE. 15000) GO TO 3000
   CALL CHBK2 (ZEV, K, N, IND, M, ZP,
  *          IP, DV, ICON)            ⑤
   CALL CNRML (ZEV, K, N, IND, M, 1,
  *            ICON)                  ⑥
            …
```

又は標準ルーチン

```
      …
   CALL CEIG2 (ZA, K, N, MODE, ZE,
  *            ZEV, VW, IVW, ICON)    ⑦
   IF (ICON. GE. 20000) GO TO 1000
      …
```

①,②,③「すべての固有値を求めるとき」と同じである．ただし，②を実行した後のパラメタ ZA をそのまま③に用いると ZA の内容が変わるので，④以後のパスのために ZA を配列 ZAW に移しておく.

④ 固有値に対応する複素ヘッセンベルグ行列の固有ベクトルを逆反復法により求める．パラメタ IND はどの固有値に対応する固有ベクトルかを求めるかの指標ベクトルであり，これによって利用者は，一部の固有ベクトルを求めるための制御を行うことができる.

⑤ ④で求めた固有ベクトルの逆変換を行って ***A*** の固有ベクトルを求める.

⑥ ⑤で求めた固有ベクトルの正規化を行う．正規化（ノルムを 1 とする）の際に，ユークリッドノルムを用いるか又はインフィニティノルムを用いるかの選択が行える．ここでは前者の意味で正規化を行う.

⑦ すべての固有値固有ベクトルを求めるとき，①～⑥までのパスをまとめて行うことができる．ただし，固有ベクトルは逆反復法を用いずに，固有値が求まるまでの変換行列を掛け合わせることによって，固有ベクトルを一度に求めている．したがって，求まらない固有値がある場合は，もはや固有ベクトルの計算は不可能となる．なお，行列の平衡化を行うかどうかは，利用者によるパラメタ MODE の指示による.

4.4　実対称行列の固有値固有ベクトル

実対称行列の固有値固有ベクトルを求めるとき，SSL II の各サブルーチンをどのように用いるか標準的な例をあげて簡単に説明する.

SSL II には，実対称行列の固有値固有ベクトルを一度に求めることができる標準ルーチンと，細分化

された各機能を果すコンポーネントルーチンが用意されている．（表 4.4 参照）．

表 4.4　実対称行列の標準固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	実対称行列の固有値固有ベクトル	SEIG1 (B21-21-0101)
		SEIG2 (B21-21-0201)
コンポーネントルーチン	実対称行列の実対称 3 重対角行列への変換	TRID1 (B21-21-0302)
	実対称 3 重対角行列の固有値	TRQL (B21-21-0402)
		BSCT1 (B21-21-0502)
	実対称 3 重対角行列の固有値固有ベクトル	TEIG1 (B21-21-0602)
		TEIG2 (B21-21-0702)
	実対称行列の固有ベクトルへの逆変換	TRBK (B21-21-0802)

ここでは具体的に，利用者の持つ問題を，
□ すべての固有値を求めるとき
□ 一部の固有値を求めるとき
□ すべての固有値固有ベクトルを求めるとき
□ 一部の固有値固有ベクトルを求めるとき
に分け，続く(1)～(4)でコンポーネントルーチンと標準ルーチンの使い方を，それぞれの手順に従って個々に説明する．

そして(5)(6)では処理に関する若干の補足説明を行う．

全固有値固有ベクトルあるいは部分固有値固有ベクトルを求める場合，コンポーネントルーチンを順次呼び出すことによって求めるか，それとも標準ルーチンによって求めるかは利用者に任されている．後者は前者を組み合わせたものであるので，使いやすさという理由から，通常は後者を利用することを勧めたい．

ただし，実対称行列が最初から 3 重対角行列である場合は，当然のことながらコンポーネントルーチン TEIG1 あるいは TEIG2 を使用することができる．

なお，SSL II で扱う実対称行列は，対称行列用圧縮モードで与えることが必要である（「2.8　データの格納方法」参照）．

(1)　すべての固有値を求めるとき

次のパス①～②により実対称行列 ***A*** のすべての固有値を求めることができる．

```
      …
CALL TRID1 (A, N, D, SD, ICON)      ①
IF (ICON .EQ. 30000) GO TO 1000
CALL TRQL (D, SD, N, E, M, ICON)    ②
      …
```

① ハウスホルダー法により ***A*** を 3 重対角行列に変換する．***A*** が既に 3 重対角行列である場合は，この処理は不要である．
② QL 法により 3 重対角行列の固有値すなわち ***A*** の固有値を求める．

(2)　一部の固有値を求めるとき

次のパス①～②により実対称行列 ***A*** の一部の固有値を求めることができる．

```
      …
CALL TRID1 (A, N, D, SD, ICON)      ①
IF (ICON.EQ. 30000) GO TO 1000
CALL BSCT1 (D, SD, N, M, EPST, E,
*VW, ICON)                          ②
      …
```

① ハウスホルダー法により ***A*** を 3 重対角行列に変換する．
② バイセクション法により 3 重対角行列の固有値を求める．利用者はパラメタ M を用いて固有値の大きい方からいくつ求めるか，また小さい方からいくつ求めるかを指定する．
ただし，***A*** の次数 n に対して $n/4$ 以上の固有値を求めたいときは，(1)①②で対処した方が一般的には処理速度は速い．

(3)　すべての固有値固有ベクトルを求めるとき

次のパス①～③又は④により実対称行列 ***A*** のすべての固有値固有ベクトルを求めることができる．

```
      …
CALL TRID1(A,N,D,SD,ICON)           ①
IF(ICON .EQ. 30000)GO TO 1000
CALL TEIG1(D,SD,N,E,EV,K,M,ICON)    ②
IF(ICON .GE. 20000)GO TO 2000
CALL TRBK(EV,K,N,M,A,ICON)          ③
      …
```

又は，標準ルーチン

```
      …
CALL SEIG1(A,N,E,EV,K,M,VW,ICON)    ④
IF(ICON .GE. 20000)GO TO 1000
      …
```

① ハウスホルダー法により ***A*** を 3 重対角行列に変換する．
② QL 法と，QL 法による変換行列を順次掛けあわせることにより，3 重対角行列のすべての固有値固有ベクトルを求める．なお，固有ベクトルはユークリッドノルムが 1 となるように正規化されている．
③ ①の処理のために②の固有ベクトルは実対称行列 ***A*** のそれとはなっていない．したがって，ここでは①の逆変換を行って，実対称行列 ***A*** の固有ベクトルを求める．
④ ①～③の処理をまとめて行うことができる．

(4)　一部の固有値固有ベクトルを求めるとき

次のパス①～③又は④により，実対称行列 ***A*** の一部の固有値固有ベクトルを求めることができる．

```
   …
CALL TRID1 (A,N,D,SD,ICON)                ①
IF(ICON .EQ. 30000)GO TO 1000
CALL TEIG2(D,SD,N,M,E,EV,K,VW,ICON)
                                          ②
IF(ICON .GE. 20000)GO TO 2000
CALL TRBK(EV,K,N,M,A,ICON)                ③
   …
```

又は，標準ルーチン

```
   …
CALL SEIG2(A,N,M,E,EV,K,VW,ICON)           ④
IF(ICON .GE. 20000)GO TO 1000
   …
```

① ハウスホルダー法により ***A*** を 3 重対角行列へ変換する．

② バイセクション法と逆反復法により 3 重対角行列の一部の固有値固有ベクトルを求める．逆反復法により求めた固有ベクトルは，固有値が接近している場合，必ずしも互いに直交するとは限らない．したがって②では，接近した固有値に対する固有ベクトルをすでに求まった分に対して直交するよう修正することをあわせて行っている．また，求められた固有ベクトルは，ユークリッドノルムが 1 となるように正規化されている．

③ ①の処理のために②の固有ベクトルは実対称行列 ***A*** のそれとはなっていない．したがって，ここでは①の逆変換を行って，実対称行列 ***A*** の固有ベクトルを求める．

④ ①～③の処理をまとめて行うことができる．

(5)　QL 法について

① QR 法との比較

(1)②や(3)②で採用した QL 法は，実行例の固有値を求めるために採用している QR 法と基本的には同じである．ただ，QR 法では固有値を行列の右下すみから求めるのに対して， QL 法では，左上すみから求める．これらのことは行列のデータの並び具合に関連している．すなわち，前者は行列の各要素の絶対値が左上から右下に進むに従って小さくなっている場合に最適であり，後者は逆に左上から右下に進むに従って大きくなっている場合に最適である．(1)①や(3)①の TRID1 によって出力される 3 重対角行列は，一般には右下方の要素の絶対値が左上方に比べて大きくなりやすい．続く(1)②や(3)②で QL 法を採用しているのはこのためである．

② 原点移動を暗黙に行う QL 法

QL 法には，更に明示型 (explicit) QL 法と黙示型(implicit) QL 法とがある．両者の違いは，固有値を求める収束速度を速くするための手段：原点移動（サブルーチン TRQL 参照）を陽に行うか，暗に行うかの違いである．前者は各要素の絶対値が左上から右下へ進むにつれて大きくなっている場合はよいが，その逆の場合には小さな固有値の精度はかなり悪くなる．これをある程度カバーするものとして，(1)②や(3)②では黙示型 QL 法を用いている．

(6)　行列の直和分解について

固有値固有ベクトルを求めるとき，できるだけ行列を小行列に分解（直和分解）して，各小行列ごとに固有値固有ベクトルを求めた方が精度や処理速度の点で望ましい．(1)②，(2) ②，(3) ②及び(4) ②の処理では，3 重対角行列を次の条件を調べることにより小行列に分解し，固有値固有ベクトルを求めている．

$$|c_i| \le u\left(|b_{i-1}| + |b_i|\right) \qquad (i = 2,3,\ldots,n) \tag{4.1}$$

ここで u は丸め誤差の単位であり， c_i, b_i は図 4.1 に示すとおりである．

4.5　エルミート行列の固有値固有ベクトル

エルミート行列の固有値固有ベクトルを求めるとき，SSL II の各サブルーチンをどのように用いるか標準的な例をあげて簡単に説明する．

エルミート行列の固有値固有ベクトルを求める手順は，

① エルミート行列を実対称 3 重対角行列に変換する．

② 実対称 3 重対角行列の固有値を求める．

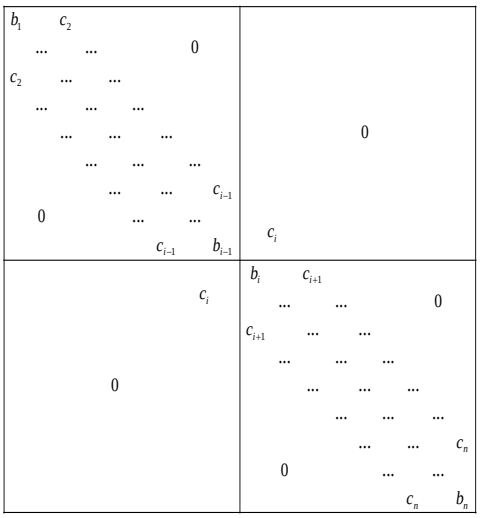


図 4.1　3 重対角行列が二つの小行列に直和分解された例

③　実対称 3 重対角行列の固有ベクトルを求める.
④　実対称 3 重対角行列の固有ベクトルを，エルミート行列の固有ベクトルへ逆変換する.

のように 4 段階に分けられる．SSL II では，これらの手順に対応して，それぞれのコンポーネントルーチンと，全手順を一度に行えるようにした標準ルーチンが用意されている．（表 4.5 参照）

表 4.5　エルミート行列の標準固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	エルミート行列の固有値固有ベクトル	HEIG2 (B21-25-0201)
コンポーネントルーチン	エルミート行列の実対称 3 重対角行列への変換	TRIDH (B21-25-0302)
	実対称 3 重対角行列の固有値	TRQL (B21-21-0402)
		BSCT1 (B21-21-0502)
	実対称 3 重対角行列の固有値固有ベクトル	TEIG1 (B21-21-0602)
		TEIG2 (B21-21-0702)
	エルミート行列の固有ベクトルへの逆変換	TRBKH (B21-25-0402)

ここでは具体的に，エルミート行列についての利用者の問題を目的別に,
□ すべての固有値を求めるとき
□ 一部の固有値を求めるとき
□ すべての固有値固有ベクトルを求めるとき
□ 一部の固有値固有ベクトルを求めるとき
のように分ける，以下の(1)～(4)ではそれぞれの目的別に分類し，コンポーネントルーチンと標準ルーチンの使い方を説明する.

固有値固有ベクトルのすべて又は，一部を求める場合において，標準ルーチンだけを使用するか，あるいは各コンポーネントルーチンを使用するかは利用者に任されている．ただ，使いやすさという点で，通常は前者を使うことを勧める.

なお，SSL II で扱うエルミート行列は，エルミート行列用圧縮モードで与えることが必要である（「2.8 データの格納方法」参照）.

(1)　すべての固有値を求めるとき

次のパス①～②により，エルミート行列 ***A*** のすべての固有値を求めることができる.

```
      ･･･
CALL TRIDH(A,K,N,D,SD,V,ICON)        ①
IF(ICON .EQ. 30000)GO TO 1000
CALL TRQL(D,SD,N,E,M,ICON)           ②
      ･･･
```

① ハウスホルダー法によりエルミート行列 ***A*** を実対称 3 重対角行列に変換する.
② QL 法により実対称 3 重対角行列の全固有値を求める.

(2)　一部の固有値を求めるとき

次のパス①～②により，***A*** の固有値のうち大きい方，又は小さい方から *m* 個求めることができる.

```
      ･･･
CALL TRIDH(A,K,N,D,SD,V,ICON)        ①
IF(ICON .EQ. 30000)GO TO 1000
CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)
                                     ②
      ･･･
```

① ハウスホルダー法によりエルミート行列 ***A*** を実対称 3 重対角行列へ変換する.
② バイセクション法により実対称 3 重対角行列の固有値を大きい方（又は小さい方）から *m* 個求める.

ただし，***A*** の次数 *n* に対して *n*/4 以上の固有値を求めるときは，TRIDH とすべての固有値を求めるサブルーチン TRQL を使う方が一般に処理は速い.

(3)　すべての固有値固有ベクトルを求めるとき

次の①～③，又は④により，すべての固有値固有ベクトルを求めることができる.

```
      ･･･
CALL TRIDH(A,K,N,D,SD,V,ICON)        ①
IF(ICON .EQ. 30000)GO TO 1000
CALL TEIG1(D,SD,N,E,EV,K,M,ICON)     ②
IF(ICON .GE. 20000)GO TO 2000
CALL TRBKH(EV,EVI,K,N,M,P,V,ICON)    ③
IF(ICON .EQ. 30000)GO TO 3000
      ･･･
```

又は，標準ルーチン,

```
      ･･･
 CALL HEIG2(A,K,N,N,E,EVR,EVI,VW,
*ICON)                               ④
 IF(ICON .GE. 20000)GO TO 1000
      ･･･
```

① エルミート行列 ***A*** を実対称 3 重対角行列に変換する.
② QL 法により実対称 3 重対角行列の固有値（すなわち ***A*** の固有値）固有ベクトルを求める.
③ ②で求めた固有ベクトルを ***A*** の固有ベクトルへ逆変換する.
④ 標準ルーチン HEIG2 を使って，①～③の処理を一度にまとめて行うことができる.
このとき HEIG2 の第 4 パラメタを N としているのは固有値を大きいほうから *n* 個求めることを指定した場合である.

(4)　一部の固有値固有ベクトルを求めるとき

次のパス①～③，又は④により，エルミート行列の m 個の固有値固有ベクトルを求めることができる．

```
      ･･･
CALL TRIDH(A,K,N,D,SD,V,ICON)          ①
IF(ICON .EQ. 30000)GO TO 1000
CALL TEIG2(D,SD,N,M,E,EV,K,VW,ICON)
                                       ②
IF(ICON .GE. 20000)GO TO 2000
CALL TRBKH(EV,EVI,K,N,M,P,V,ICON)      ③
IF(ICON .EQ. 30000)GO TO 3000
      ･･･
```

又は，標準ルーチン

```
      ･･･
CALL HEIG2(A,K,N,M,E,EVR,EVI,VW,ICON)  ④
IF(ICON .GE. 20000)GO TO 1000
      ･･･
```

① エルミート行列 $\boldsymbol{A}$ を実対称 3 重対角行列に変換する．

② バイセクション法と逆反復法により，実対称 3 重対角行列の固有値を大きい方（又は小さい方）から m 個求め，それらに対応する固有ベクトルを求める．

③ ②で求めた固有ベクトルの逆変換を行う．

④ 標準ルーチン HEIG2 を使って，①～③の処理を，一度にまとめて行うことができる．

4.6　実対称バンド行列の固有値固有ベクトル

実対称バンド行列の固有値固有ベクトルを求めるためのサブルーチンとして，BSEG, BTRID, BSVEC, BSEGJ などが用意されている．

これらは一般に，行列の次数 n が 100 以上の大型行列で，かつ行列のバンド幅 h との間で $h/n < 1/6$ である場合に適している．またジェニングス法による BSEGJ は，$n/10$ 個以内の固有値を求める場合に有効である．実対称バンド行列の場合，全固有値固有ベクトルを要求するケースは一般に少なく，したがって，標準ルーチンもその趣旨に沿って用意されている．BSEG と BSEGJ は標準ルーチンであり，BTRID と BSVEC は BSEG を構成するコンポーネントルーチンである．（表 4.6 参照）．今，これらの各サブルーチンを利用者の問題に応じて，どのように使うことができるか例をあげて説明する．なお，SSL II で扱う実対称バンド行列は，対称バンド行列用圧縮モードで与える必要がある（「2.8　データの格納方法」参照）．

表 4.6　実対称バンド行列の標準固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	実対称バンド行列の固有値固有ベクトル	BSEG (B51-21-0201)
		BSEGJ (B51-21-1001)
コンポーネントルーチン	実対称バンド行列の実対称 3 重対角行列への変換	BTRID (B51-21-0302)
	実対称 3 重対角行列の固有値	TRQL (B21-21-0402)
		BSCT1 (B21-21-0502)
	実対称バンド行列の固有ベクトル	BSVEC (B51-21-0402)

(1)　一部の固有値を求めるとき

(a)　標準ルーチンを使う場合

```
      ･･･
 CALL BSEG(A,N,NH,M,0,EPST,E,EV,K,
*VW,ICON)                              ①
 IF(ICON.GE.20000)GO TO 1000
      ･･･
```

① 次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の固有値を，大きい方（又は小さい方）から m 個求める．第 5 パラメタが 0 であるのは固有ベクトルが不要であることを意味する．

(b)　コンポーネントルーチンを使う場合

```
      ･･･
CALL BTRID(A,N,NH,D,SD,ICON)①
IF(ICON.EQ.30000) GO TO 1000
CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)    ②
IF(ICON.EQ.30000) GO TO 2000
      ･･･
```

① 次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ をルティスハウザー・シュワルツ法により，実対称 3 重対角行列 $\boldsymbol{T}$ に変換する．

② $\boldsymbol{T}$ の固有値のうち，大きい方（又は小さい方）から m 個求める．

(2)　全固有値を求めるとき

(a)　標準ルーチンを使う場合

一部の固有値を求めるときの BSEG の例において，第 4 パラメタを N とすることにより全固有値を求めることができるが，通常は次のコンポーネントルーチンによるパスを利用することを勧める．

(b)　コンポーネントルーチンを使う場合

```
      ･･･
CALL BTRID(A,N,NH,D,SD,ICON)           ①
IF(ICON.EQ.30000) GO TO 1000
CALL TRQL(D,SD,N,E,M,ICON)             ②
      ･･･
```

① 次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ をルティスハウザー・シュワルツ法により，実対称 3 重対角行列 $\boldsymbol{T}$ へ変換する．

②　$\boldsymbol{T}$ の全固有値を QL 法により求める.

(3)　一部の固有値固有ベクトルを求めるとき

(a)　標準ルーチンを使う場合
標準ルーチンとして次の二つがある

```
      ･･･
   CALL BSEG(A,N,NH,M,NV,EPST,E,EV,K,
  *VW,ICON)                                  ①
   IF(ICON.GE.20000)GO TO 1000
      ･･･
   CALL BSEGJ(A,N,NH,M,EPST,LM,E,EV,K,
  *IT,VW,ICON)                               ①'
   IF(ICON.GE.20000)GO TO 1000
      ･･･
```

①はルティスハウザー・シュワルツ法，バイセクション法，逆反復法を順次用いて，まず固有値を求め，それから固有ベクトルを求めるのに対し，①′は同時反復によるジェニングス法を用いて，固有値固有ベクトルを同時に求める．また，①は固有値の大きい方（又は小さい方）から求めるのに対し，①′は絶対値の大きい順（又は小さい順）に求まる．①′は手法の性格上，行列の次数 n に較べて極めて少ない（n/10 以下）固有値固有ベクトルを求める場合に，その利用が勧められる．

①　次数 n, バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の固有値を m 個，固有ベクトルを n_v 個求める．

①′与えられた m 個の初期固有ベクトルをもとに，上記同行列 $\boldsymbol{A}$ の固有ベクトルを求める．同時に対応する固有値も求める．固有ベクトルの初期値及び反復回数の上限を EV と LM に与える点で若干の考慮が必要である．

(b)　コンポーネントルーチンを使う場合
以下のように BSEG を構成するサブルーチンを順次呼び出せばよい．

```
      ･･･
   NN = (NH + 1)*(N + N -NH)/2
   DO 10 I = 1, NN
10 AW (I) = A (I)
   CALL BTRID(A,N,NH,D,SD,ICON)①
   IF(ICON.EQ.30000) GOTO 1000
   CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)
                                              ②
   IF(ICON.EQ.30000) GO TO 2000
   CALL BSVEC(AW,N,NH,NV,E,EV,K,VW,
  *ICON)                                     ③
   IF(ICON.GE.20000) GO TO 3000
      ･･･
```

①②一部の固有値を求めるときのコンポーネントルーチンによるパスと同じである．ただし，BTRID によりパラメタ A の内容は壊されるので，③のためにあらかじめ A の内容を配列 AW に移しておく．

③　②より求めた m 個の固有値のうち，最初の n_v 個に対応する固有ベクトルを逆反復法により求める．

(4)　全固有値固有ベクトルを求めるとき

(a)　標準ルーチンを使う場合
一部の固有値固有ベクトルを求めるときの BSEG の例で，第 4 及び第 5 パラメタを N とすることにより全固有値固有ベクトルを求めることができる．もし，速度を重要視するなら次のコンポーネントルーチンによるパスを利用することを勧める．

(b)　コンポーネントルーチンを使う場合

```
      ･･･
   NN=(N+1)*(N+N-NH)/2
   DO 10 I=1,NN
10 AW(I)=A(I)
   N1 = NN+1
   N2 = N1 + N
   CALL BTRID(A,N,NH,VW(N1),VW(N2),ICON)①
   IF(ICON.EQ.30000) GO TO 1000
   CALL TRQL(VW(N1),VW(N2),N,E,M,ICON)
                                              ②
   IF(ICON.GE.15000) GO TO 2000
   CALL BSVEC(AW,N,NH,N,E,EV,K,VW,
   *ICON)                                     ③
   IF(ICON.GE.20000) GO TO 3000
      ･･･
```

①②全固有値を求めるときのコンポーネントルーチンによるパスと同じである．ただし BTRID によりパラメタ A の内容は壊されるので，③のためにあらかじめ A の内容を配列 AW に移しておく．

③　②で求めた全固有値に対応する固有ベクトルを逆反復法により求める．

4.7　実対称行列の一般固有値固有ベクトル

$\boldsymbol{Ax}=\lambda\boldsymbol{Bx}$ （$\boldsymbol{A}$: 対称行列，$\boldsymbol{B}$: 正値対称行列）型の固有値固有ベクトルを求めるとき，SSL II の各サブルーチンをどのように用いるか標準的な例をあげて簡単に説明する．

実対称行列の一般固有値固有ベクトルを求める手順は，

①　一般固有値問題 $(\boldsymbol{Ax}=\lambda\boldsymbol{Bx})$ を実対称行列の標準固有値問題 $(\boldsymbol{Sy}=\lambda\boldsymbol{y})$ に変換する．

②　実対称行列 $\boldsymbol{S}$ を実対称 3 重対角行列 $\boldsymbol{T}$ に変換する$(\boldsymbol{Sy}=\lambda\boldsymbol{y}\rightarrow\boldsymbol{Ty}'=\lambda\boldsymbol{y}'$への変換$)$.

③　実対称 3 重対角行列 $\boldsymbol{T}$ の固有値λを求める．

④　実対称 3 重対角行列 $\boldsymbol{T}$ の固有ベクトル $\boldsymbol{y}'$ を求める．

⑤　実対称 3 重対角行列 $\boldsymbol{T}$ の固有ベクトル $\boldsymbol{y}'$を実対称行列 $\boldsymbol{S}$ の固有ベクトル $\boldsymbol{y}$ に逆変換する．

⑥　実対称行列 $\boldsymbol{S}$ の固有ベクトル $\boldsymbol{y}$ を一般固有値問題の固有ベクトル $\boldsymbol{x}$ に逆変換する.

のように 6 段階に分けられる．SSL II では，これらの手順に対応してそれぞれのコンポーネントと，全手順を一度に行うことができる標準ルーチンが用意されている（表）4.7 参照）.

表 4.7　実対称行列の一般固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	実対称行列の一般固有値固有ベクトル	GSEG2 (B22-21-0201)
コンポーネントルーチン	一般形から標準形への変換	GSCHL (B22-21-0302)
	実対称行列の実対称 3 重対角行列への変換	TRID1 (B21-21-0302)
	実対称 3 重対角行列の固有値	TRQL (B21-21-0402)
		BSCT1 (B21-21-0502)
	実対称 3 重対角行列の固有値固有ベクトル	TEIG1 (B21-21-0602)
		TEIG2 (B21-21-0702)
	実対称行列の固有ベクトルへの逆変換	TRBK (B21-21-0802)
	一般形の固有ベクトルへの逆変換	GSBK (B22-21-0402)

ここでは具体的に，実対称行列の一般固有値固有ベクトルについての利用者の問題を目的別に,
□ すべての固有値を求めるとき
□ 一部の固有値を求めるとき
□ すべての固有値固有ベクトルを求めるとき
□ 一部の固有値固有ベクトルを求めるとき
のように分け，以下の(1)～(4)でそれぞれについて，コンポーネントルーチンと標準ルーチンの使い方を説明する.

固有値固有ベクトルのすべて又は，一部を求めるとき，コンポーネントルーチンを順次呼出すか，あるいは標準ルーチンを使用するかは利用者に任されている．ただ，使いやすさという点で，通常は後者を使うことを勧める.

なお，SSL II で扱う実対称行列は，対称行列用圧縮モードで与えることが必要である（「2.8　データの格納方法」参照）.

(1)　すべての固有値を求めるとき

次のパス①～③により，すべての固有値を求めることができる.

```
      ･･･
CALL GSCHL(A,B,N,EPSZ,ICON)          ①
IF(ICON.GE.20000) GO TO 1000
CALL TRID1(A,N,D,SD,ICON)            ②
CALL TRQL(D,SD,N,E,M,ICON)           ③
      ･･･
```

① 実対称行列の一般固有値問題 $(\boldsymbol{Ax} = \lambda\boldsymbol{Bx})$ を標準固有値問題 $(\boldsymbol{Sy} = \lambda\boldsymbol{y})$ に変換する.
② ハウスホルダー法により実対称行列 $\boldsymbol{S}$ を実対称 3 重対角行列に変換する.
③ QL 法により実対称 3 重対角行列のすべての固有値を求める.

(2)　一部の固有値を求めるとき

次のパス①～③により，固有値を大きい方（又は小さい方）から m 個求めることができる.

```
      ･･･
CALL GSCHL(A,B,N,EPSZ,ICON)          ①
IF(ICON.GE.20000) GO TO 1000
CALL TRID1(A,N,D,SD,ICON)            ②
CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)
                                     ③
      ･･･
```

① 実対称行列の一般固有値問題 $(\boldsymbol{Ax} = \lambda\boldsymbol{Bx})$ を標準固有値問題 $(\boldsymbol{Sy} = \lambda\boldsymbol{y})$ に変換する.
② ハウスホルダー法により実対称行列 $\boldsymbol{S}$ を実対称 3 重対角行列に変換する.
③ バイセクション法により実対称 3 重対角行列の固有値を大きい方（又は小さい方）から m 個求める.

ここで，$\boldsymbol{A}$ の次数 n に対して $n/4$ 以上の固有値を求めたいときは，すべての固有値を求めるサブルーチン TRQL を使う方が一般に処理は速い.

(3)　すべての固有値固有ベクトルを求めるとき

次のパス①～⑤，又は⑥により，すべての固有値固有ベクトルを求めることができる.

```
      ･･･
CALL GSCHL(A,B,N,EPSZ,ICON)          ①
IF(ICON.GE.20000) GO TO 1000
CALL TRID1(A,N,D,SD,ICON)            ②
CALL TEIG1(D,SD,N,E,EV,K,M,ICON)     ③
IF(ICON.GE.20000)GO TO 3000
CALL TRBK(EV,K,N,M,A,ICON)           ④
CALL GSBK(EV,K,N,M,B,ICON)           ⑤
      ･･･
```

又は標準ルーチン,

```
      ･･･
CALL GSEG2(A,B,N,N,EPSZ,EPST,E,EV,K,
*VW,ICON)                            ⑥
IF(ICON.GE.20000) GO TO 1000
      ･･･
```

① 実対称行列の一般固有値問題 $(\boldsymbol{Ax} = \lambda\boldsymbol{Bx})$ を標準固有値問題 $(\boldsymbol{Sy} = \lambda\boldsymbol{y})$ に変換する.
② ハウスホウルダー法により実対称行列 $\boldsymbol{S}$ を実対称 3 重対角行列に変換する.
③ QL 法により実対称 3 重対角行列のすべての固有値固有ベクトルを求める.
④ 実対称 3 重対角行列の固有ベクトルを実対称行列の固有ベクトルに逆変換する.

⑤　実対称行列の固有ベクトルを一般固有値問題の固有ベクトルに逆変換する.
⑥　標準ルーチン GSEG2 を使って，①～⑤の処理を一度にまとめて行うことができる.
このとき，GSEG2 の第 4 パラメタを N としているのは，固有値を大きい方からすべて求めることを指定した場合である.

(4)　一部の固有値固有ベクトルを求めるとき

次のパス①～⑤，又は⑥により，*m* 個の固有値固有ベクトルを求めることができる.

```
      ･･･
  CALL GSCHL(A,B,N,EPSZ,ICON)              ①
  IF(ICON.GE.20000) GO TO 1000
  CALL TRID1(A,N,D,SD,ICON)                ②
  IF(ICON.EQ.3000) GO TO 2000
  CALL TEIG2(D,SD,N,M,E,EV,K,VW,
 *ICON)                                    ③
  IF(ICON.GE.20000) GO TO 3000
  CALL TRBK(EV,K,N,M,A,ICON)               ④
  CALL GSBK(EV,K,N,M,B,ICON)               ⑤
      ･･･
```

又は標準ルーチン,

```
      ･･･
  CALL GSEG2(A,B,N,M,EPSZ,EPST,E,EV,K,
 * VW,ICON)                                ⑥
  IF(ICON.GE.20000) GO TO 1000
      ･･･
```

①　実対称行列の一般固有値問題 (***Ax*** = *λ**Bx***) を標準固有値問題 (***Sy*** = *λ**y***) に変換する.
②　ハウスホルダー法により実対称行列 ***S*** を実対称 3 重対角行列に変換する.
③　バイセクション法，逆反復法により，実対称 3 重対角行列の固有値の大きい方（又は小さい方）から *m* 個求め，それらに対応する固有ベクトルを求める.
④　実対称 3 重対角行列の固有ベクトルを実対称行列の固有ベクトルに逆変換する.
⑤　実対称行列の固有ベクトルを一般固有値問題の固有ベクトルに逆変換する.
⑥　標準ルーチン GSEG2 を使って，①～⑤の処理を一度にまとめて行うことができる.

4.8　実対称バンド行列の一般固有値固有ベクトル

Ax=λBx (***A***: 対称バンド行列，***B***: 正値対称バンド行列) 型の固有値固有ベクトルを求めるために SSL II では，表 4.8 に示すようなサブルーチンを用意している.

これらは一般には大型行列で，かつ行列の次数 *n* , バンド幅 *h* との間に，*h*/*n* < 1/6 である場合に適している.

また，ジェニングス法によるサブルーチン GBSEG は，*n* /10 個以内の固有値固有ベクトルを求める場合時間的に有効である．サブルーチン GBSEG では指定された *m* 個の固有値と固有ベクトルを同時に求めるので，正常終了しないときは，固有値と固有ベクトルが一つも求まっていないので注意すること.

今，利用者の問題に応じて，サブルーチンをどのように使うことができるか例をあげて説明する．なお，SSL II で扱う実対称バンド行列は，対称バンド行列用圧縮モードで与える必要がある（「2.8　データの格納方法」参照）.

表 4.8　実対称バンド行列の一般固有値問題のサブルーチン

レベル	機能	サブルーチン名
標準ルーチン	実対称バンド行列の一般固有値固有ベクトル	GBSEG (B52-11-0101)

(1)　一部の固有値固有ベクトルを求めるとき

```
      ･･･
  CALL GBSEG(A,B,N,NH,M,EPSZ,EPST,
 *           LM,E,EV,K,IT,VW,ICON)    ①
  IF(ICON.GE.20000) GO TO 1000
      ･･･
```

①　同時反復法によるジェニングス法を用いて，固有値固有ベクトルを求める．パラメタ M の与え方により，固有値の絶対値の大きい方又は小さい方から *m* 個の固有値固有ベクトルが求められる.

第 5 章
非線型計算

5.1 概要

この章では以下の問題を取り上げる.
方程式の根：
代数方程式，超越方程式，連立非線型方程式の根を求める.

5.2 代数方程式

代数方程式を (5.1) で表す.

$$a_0 x^n + a_1 x^{n-1} + \ldots + a_n = 0, \qquad |a_0| \neq 0 \qquad (5.1)$$

ここで a_i (i = 0, 1 ... n) は実数又は複素数である.
a_i が実数の場合を実係数代数方程式と言い，a_i が複素数の場合を複素係数代数方程式と言う．後者の場合は，(5.1) において特に x の代りに z を用いることにする.
代数方程式を解くサブルーチンは断らないかぎりすべての根を求めることを目的としている.
代数方程式のサブルーチンを表 5.1 に示す.

表 5.1 代数方程式のサブルーチン

目的	サブルーチン名	手法	備考
実係数 2 次方程式	RQDR (C21-11-0101)	根の公式	
複素係数 2 次方程式	CQDR (C21-15-0101)	根の公式	
実係数低次 代数方程式	LOWP (C21-41-0101)	代数的手法と反復的手法を併用	次数は 5 次以下
実係数高次 代数方程式	RJETR (C22-11-0111)	ジェンキンス・トラウブの方法	
複素係数高次 代数方程式	CJART (C22-15-0101)	ヤラット法	

5 次以下の実係数代数方程式を解くときは LOWP を使えばよいが，特に 2 次方程式だけを解く場合は RQDR を使うこと.
以下，手法とそれにかかわる事柄について述べる.
代数方程式を解く手法には，代数的手法と反復的手法とがある．代数的手法とは，4 次以下の方程式に対して代数学で言うところの“根の公式”を用いて解く方法であり，一方反復的手法とは，任意の次数の方程式に対して根の大ざっぱな近似値を設定しそれを反復的に改良してゆき最後に良い近似値を得る，という手法である．反復的手法は多くの場合，根を一つずつ順番に求めていく方法をとり，ある根が求まったあとその根を除いた，より低次の代数方程式を作り出してまた別の根を求めるという方法である.
代数的手法と反復的手法とを優劣の点で比較するのは難しいように思われる．というのは各々独自の難点を持っているからである．それを以下に挙げる.

a. 代数的手法の難点
(a) (5.1) の係数 a_i に，大きさにおいて激しいへだたりがあるものが混在している場合は計算過程でオーバフローあるいはアンダフローを起こす可能性がある.

b. 反復的手法の難点
(a) 初期の近似値を設定することが難しい．それを誤ると反復しても近似値が収束しない（逆に近似値が収束しなかったときは初期近似の設定に誤りがあると判断できる．また，ある根は求まっても別な根は求まらないということもある．）
(b) 反復的に求まってくる近似値が収束したか否かの判定，いわゆる収束判定という手間の問題がある.

SSL II においては，特に a.の (a) をさけることに重点をおき，2 次方程式の場合を除いて，もっぱら反復的手法に頼っている.
ここで収束判定法について，SSL II のサブルーチンがとっている方法を述べておく.
代数方程式 $f(x) \equiv \sum_{k=0}^{n} a_k x^{n-k} = 0$ を反復的に求めるときは $f(x)$ の計算値が計算誤差の範囲内の大きさに

なってしまうと，もはやそれ以上 $f(x)$ を小さくしても意味がない．ところで $f(x)$ を計算するときの計算誤差の上限を $\varepsilon(x)$ とすれば，誤差理論より

$$|\varepsilon(x)| = u\sum_{k=0}^{n}\left|a_k x^{n-k}\right| \tag{5.2}$$

が成り立つ．ここでは u は丸め誤差の単位である．したがって

$$|f(x)| \le |\varepsilon(x)| \tag{5.3}$$

を満たす x の領域において，x が真の根であるかどうかを確認する方法はない．ゆえに，

$$\left|\sum_{k=0}^{n} a_k x^{n-k}\right| \le u\sum_{k=0}^{n}\left|a_k x^{n-k}\right| \tag{5.4}$$

を満たす x が求まったとき収束したと判断し，それを一つの根として採用する．（なお，(5.2) について詳しくは文献［23］を参照すること．）

最後に根の精度について注意を述べる．代数的手法であれ反復的手法であれ，一定の桁数で計算する限り，ある根は精度良く求まり，ある根はそれに反して精度良く求まらないことが起こる．一般に，重根あるいは近接根は，そうでない根の精度ほどにはよく求**まらない**．利用者はこのことに注意され，もし求まった根のなかに近接しているものがあれば，そうでない根ほどには精度は良くないと考えてもらたい．

5.3 超越方程式

超越方程式を (5.5) で表す．

$$f(x) = 0 \tag{5.5}$$

$f(x)$ が実関数の場合を実超越方程式と言い，複素関数の場合を複素超越方程式と言う．後者の場合は (5.5) において特に x の代りに z を用いることにする．超越方程式を解くサブルーチンは，独立変数の指定された範囲内の，あるいは指定された点の近傍の，ただ一つの根を求めることを目的としている．

超越方程式のサブルーチンを表 5.2 に示す．

超越方程式を解く手法はもっぱら反復的手法である．反復的手法における収束の速さはまず第 1 に指定された範囲がいかに狭いか，あるいは指定された点がいかに根に近いかに依存している．収束判定法についてはサブルーチンごとに異なるので個々の説明のところで述べてある．

表 5.2 超越方程式のサブルーチン

目的	サブルーチン名	手法	備考
実超越方程式	TSD1 (C23-11-0101)	2 分法，線型補間法，逆 2 次補間法を併用	微係数不要 区間指定
	TSDM (C23-11-0111)	マラー法	微係数不要 初期値指定
複素超越方程式	CTSDM (C23-15-0101)	マラー法	微係数不要 初期値指定

5.4 連立非線型方程式

方程式を (5.6) で表す．

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \tag{5.6}$$

ここで，$\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}),..., f_n(\boldsymbol{x}))^{\mathrm{T}}$，$\boldsymbol{x}=(x_1,x_2\ldots,x_n)^{\mathrm{T}}$ 及び $\boldsymbol{0}$ は n 次元ベクトルである．

連立非線型方程式の解法は，反復的手法である．すなわち，利用者が与える初期ベクトル $\boldsymbol{x}_0$ から出発して，逐次改良していくことにより，(5.6) の解ベクトルを，必要な精度まで求めようとするものである．

連立非線型方程式のサブルーチンを表 5.3 に示す．

表 5.3 連立非線型方程式のサブルーチン

目的	サブルーチン名	手法	備考
連立非線型方程式	NOLBR (C24-11-0101)	ブレント法	微係数不要

反復的手法の中で最もよく知られているのはニュートン法であり，それは (5.7) で表せる．

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \boldsymbol{J}_i^{-1}\boldsymbol{f}(\boldsymbol{x}_i), \quad i = 0, 1, ... \tag{5.7}$$

ここで $\boldsymbol{J}_i$ は，$\boldsymbol{x} = \boldsymbol{x}_i$, における $\boldsymbol{f}(\boldsymbol{x})$ のヤコビアン行列であり，(5.8) を意味する．

$$\boldsymbol{J}_i = \left[\begin{array}{cccc} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \cdots & \cdots & & \cdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{array}\right]_{\boldsymbol{x}=\boldsymbol{x}i} \tag{5.8}$$

このニュートン法は，理論的には良い方法である．すなわち，収束の速さは 2 次であり，計算式が単純であるという利点がある．しかし，複雑な（又は大規模な）方程式になると，計算上のいくつかの問題点が出てくる．主な原因は次の 3 つである．

a. (5.8) の係数 $\partial f_i / \partial x_j$ の計算式を作るのが困難である（式が複雑で事実上偏微分できないことがある）．

b. (5.8) の全要素の計算量が多すぎる.
c. 反復 1 回ごとに，$\boldsymbol{J}_i$ を係数行列とする連立 1 次方程式を解くので時間がかかる.

これらの問題点を解消し，しかも収束の速さを 2 次に保つことができれば，利用者の使いやすさ及び計算時間の短縮が期待できる．上記の問題点を解消する具体的な方法としては以下の方法が考えられる.

a. に対しては，$\partial f_i / \partial x_j$ を差分で近似する．すなわち，適当な h (>0) を選ぶことにより，

$$\frac{\partial f_i}{\partial x_j} \approx \frac{f_i(x_1,\ldots,x_j+h,\ldots,x_n)-f_i(x_1,\ldots,x_n)}{h} \quad (5.9)$$

とする.

b. 及び c.に対しては，ヤコビアン行列を直接求めないで，計算量が少なくて済むような，擬似的（全要素を求めないという意味で）なヤコビアン行列を使って連立 1 次方程式を解く.

SSL II ではこれらを実現するような手法を採用している.

次に，連立非線型方程式のサブルーチンを使用する上での注意を述べる.

利用者は，方程式を定義する関数系を計算する関数副プログラムを用意する必要がある．この関数副プログラムは，サブルーチンを効率良く利用し，指定した精度の解を得るためにも，次の点に十分注意して用意しなければならない.

(a) 関数の計算は，桁落ちを避けるように工夫すること．これは，この関数値が，サブルーチン内で微係数を評価するために使われることからも特に重要である.
(b) 変数ベクトル $\boldsymbol{x}$ や関数ベクトル $\boldsymbol{f}(\boldsymbol{x})$ などの各要素の大きさをそろえることが望ましい．なぜならば，計算の過程で，絶対値が極めて大きな要素のために，他の要素が無視されてしまうようなことが起きるからである（特に本サブルーチンでは，収束判定において，最大値要素の変化を見て収束したか否かの判断をしているため）.

また，解ベクトルの精度は，利用者の与える収束判定値に依存する．一般に，収束判定値を小さくすれば，解ベクトルの精度の向上が期待できる．しかしながら，丸め誤差の影響のため精度の限界があるので，むやみに収束判定値を小さく与えても，それだけの精度が得られないことがあるので注意を要する.

次に，初期ベクトル $\boldsymbol{x_0}$ の選び方であるが，これは方程式の物理的情報によってなされるべきであり，利用者に任された問題である．もし，そのような物理的情報が無い場合は，初期ベクトルをいろいろ変えて，何回か試行するのもよい.

最後に，方程式の性質によっては，単精度サブルーチンでは解けないが，倍精度サブルーチンでは解けることがあるので，通常は倍精度サブルーチンを利用することを勧める.

第 6 章
極値問題

6.1　概要

この章では，以下の問題を取り上げる．

- 1 変数関数の極小化
- 制約なし多変数関数の極小化
- 制約なし関数二乗和の極小化
  （非線型最小二乗解）
- 線形計画問題
- 非線形計画問題
  （制約付き多変数関数の極小化）

6.2　1 変数関数の極小化

この問題では，1 変数の実関数 $f(x)$ が与えられたとき，区間 $[a, b]$ で $f(x)$ の極小点 x^* とその関数値 $f(x^*)$ を求める．

(1)　サブーチンの種類

利用者が関数 $f(x)$ 以外に，導関数 $g(x)$ を解析的に定義できるか否かの条件に応じて，表 6.1 に示すサブルーチンが用意されている．

表 6.1 1 変数関数の極小化のサブルーチン

解析的定義	サブルーチン名	備考
$f(x)$	LMINF (D11-30-0101)	2 次補間法
$f(x), g(x)$	LMING (D11-40-0101)	3 次補間法

(2)　使用上の注意

a.　区間 $[a, b]$ の与え方

SSL II では，区間 $[a, b]$で $f(x)$ が単峰であることを仮定し，$f(x)$ の唯一の極小点を許容誤差の範囲内で求める．区間 $[a, b]$ に多くの極小点を持つ場合には，いずれの極小点に収束するかは保証されない．

したがって，求める極小点 x^* を含む区間の端点 a 及び b の値は，可能なかぎり x^* の近くにとることが望ましい．

6.3　制約なし多変数関数の極小化

この問題では，n 変数の実関数 $f(\boldsymbol{x})$ と初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，$f(\boldsymbol{x})$ の極小値をとるベクトル（極小点） $\boldsymbol{x}^*$ と，その関数値 $f(\boldsymbol{x}^*)$ を求める．ここで $\boldsymbol{x}$ は，$\boldsymbol{x} = (x_1, x_2, ..., x_n)^{\mathrm{T}}$ なるベクトルである．

関数の極小化とは，一般に任意の初期ベクトル $\boldsymbol{x}_0$ から出発し，

$$f(\boldsymbol{x}_{k+1}) < f(\boldsymbol{x}_k),\ k = 0, 1, ... \qquad (6.1)$$

$\boldsymbol{x}_k$: 反復ベクトル

なる関係を満たすように反復し，極小点 $\boldsymbol{x}^*$ を求めることである．ここで反復ベクトルは，関数 $f(\boldsymbol{x})$ が $\boldsymbol{x}_k$ において局所的に減少する方向に基づき修正されるが，そのため通常，$f(\boldsymbol{x})$ の値だけでなく，傾斜ベクトル $\boldsymbol{g}$ とヘシアン行列 $\boldsymbol{B}$ とが利用される．

$$\boldsymbol{g} = \left(\frac{\partial f}{\partial x_1},\ \frac{\partial f}{\partial x_2},\ \cdots,\ \frac{\partial f}{\partial x_n} \right)^{\mathrm{T}} \qquad (6.2)$$

$$\boldsymbol{B} = \left(b_{ij}\right),\quad b_{ij} = \partial^2 f / \partial x_i \partial x_j$$

ニュートン法を基本とした公式

関数 $f(\boldsymbol{x})$ が 2 次関数で，凹関数である場合，ニュートン法による反復公式により，理論的には高々 n 回の反復で最小点 $\boldsymbol{x}^*$ を求めることができる．

ところで，一般の関数の場合は，極小点 $\boldsymbol{x}^*$ の近傍では近似的に 2 次関数となる．すなわち，

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^*) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^*)^T \boldsymbol{B} (\boldsymbol{x} - \boldsymbol{x}^*) \qquad (6.3)$$

と表せる．したがってヘシアン行列 $\boldsymbol{B}$ が正定値行列であるなら，2 次関数の場合に適用されるニュートン法を基本とした反復公式は，一般の関数に対しても (6.3) に基づき良い反復公式となることが期待できる．そこで極小点 $\boldsymbol{x}^*$ の近傍の任意な点 $\boldsymbol{x}_k$ における傾斜ベクトルを $\boldsymbol{g}_k$ とすると，(6.3) よりニュートン法の基本反復公式を得る．

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{B}^{-1}\boldsymbol{g}_k \qquad (6.4)$$

SSL II では，(6.4) に基づく，次の二つの反復公式を導入している．

改訂準ニュートン法
反復公式

$$\begin{aligned} &\boldsymbol{B}_k \boldsymbol{p}_k = -\boldsymbol{g}_k \\ &\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k \\ &\boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \boldsymbol{E}_k \end{aligned} \qquad (6.5)$$

ここで，$\boldsymbol{B}_k$ はヘシアン行列の近似行列であり，反復の過程で階数 2 の行列 $\boldsymbol{E}_k$ により改良される．$\boldsymbol{p}_k$ は，関数が局所的に減少する方向を定める探索ベクトルである．α_k は，$f(\boldsymbol{x}_{k+1})$ が局所的最小となるように定められる（直線探索）定数である．

準ニュートン法
反復公式

$$\begin{aligned} &\boldsymbol{p}_k = -\boldsymbol{H}_k \boldsymbol{g}_k \\ &\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k \\ &\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \boldsymbol{F}_k \end{aligned}$$

ここで，$\boldsymbol{H}_k$ はヘシアン行列の逆行列 $\boldsymbol{B}^{-1}$ の近似行列であり，反復の過程で階数 2 の行列 $\boldsymbol{F}_k$ により改良される．$\boldsymbol{p}_k$ は，関数が局所的に減少する方向を定める探索ベクトルである．α_k は，$f(\boldsymbol{x}_{k+1})$ が局所的最小となるように定められる（直線探索）定数である．

(1) サブルーチンの種類

利用者が関数 $f(\boldsymbol{x})$ 以外に傾斜ベクトル $\boldsymbol{g}$ を解析的に定義できるか否かの条件に応じて，表 6.2 に示すサブルーチンが用意されている．

表 6.2 制約なし多変数関数の極小化のサブルーチン

解析的定義	サブルーチン名	備考
f(x)	MINF1 (D11-10-0101)	改訂準ニュートン法
f(x), g(x)	MING1 (D11-20-0101)	準ニュートン法

(2) 使用上の注意

a. 初期ベクトル $\boldsymbol{x}_0$ の与え方
初期ベクトル $\boldsymbol{x}_0$ は，可能なかぎり期待する極小点 $\boldsymbol{x}^*$ の近傍に取ることが望ましい．関数 $f(\boldsymbol{x})$ がいくつかの極小点を持つ場合，初期ベクトルが適切でないと，期待する極小点 $\boldsymbol{x}^*$に収束しない可能性がある．通常，$\boldsymbol{x}_0$ は関数 $f(\boldsymbol{x})$ の持つ物理的情報によって設定されるべきであろう．

b. 関数計算プログラム
利用者は，関数 $f(\boldsymbol{x})$，傾斜ベクトル $\boldsymbol{g}$ の値を計算する副プログラムを用意する場合，効率の良いコーディングをすることが望ましい．
SSL II のサブルーチンによる各関数の評価回数は，手法，初期ベクトルなどに依存するが，極値問題において関数計算の占める割合は大きく，全体の処理速度はコーディング方法に影響される．
更に，関数 $f(\boldsymbol{x})$ だけを与えるサブルーチンの場合は，通常傾斜ベクトル $\boldsymbol{g}$ を差分により近似するため，桁落ちなどを避ける工夫も必要である．
また，関数 $f(\boldsymbol{x})$ の定義において，可能なかぎり変数ベクトル $\boldsymbol{x}$ の大きさが同程度となるように平衡化することが望ましい

c. 収束判定値と極小値 $f(\boldsymbol{x}^*)$ の精度
極小化のアルゴリズムでは，極小点 $\boldsymbol{x}^*$ において関数 $f(\boldsymbol{x})$ の傾斜ベクトル $g(\boldsymbol{x}^*)$ は，

$$\boldsymbol{g}(\boldsymbol{x}^*) = \boldsymbol{0} \qquad (6.6)$$

となることを前提としている．すなわち反復公式は，関数 $f(\boldsymbol{x})$ を極小点 $\boldsymbol{x}^*$ の近傍では，2 次関数

$$f(\boldsymbol{x}^* + \delta\boldsymbol{x}) \approx f(\boldsymbol{x}^*) + \frac{1}{2}\delta\boldsymbol{x}^{\mathrm{T}}\boldsymbol{B}\delta\boldsymbol{x} \qquad (6.7)$$

として近似している．
(6.7) は $f(\boldsymbol{x})$ が適度に平衡化されているとき，$\boldsymbol{x}$ に対して ε 程度の摂動を与えた場合に，関数 $f(\boldsymbol{x})$ は ε^2 程度の変化率を示すことを意味している．
ところで，SSL II での収束判定条件は，反復ベクトルを $\boldsymbol{x}_k$ と表すと，

$$\left\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right\|_\infty \le \max\left(1.0\,,\ \left\|\boldsymbol{x}_k\right\|_\infty\right)\cdot\varepsilon \qquad (6.8)$$

ここで ε は収束判定値

を満たした場合に $\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$ とみなす．
したがって，極小値 $f(\boldsymbol{x}^*)$ を丸め誤差程度まで正しく求めるためには，収束判定値 ε を

$$\varepsilon \approx \sqrt{u}\ ,\ u \text{ は丸め誤差の単位}$$

程度に与えるのが適当である．
SSL II では，$2\cdot\sqrt{u}$ を収束判定の標準値としている．

6.4 制約なし関数二乗和の極小化（非線型最小二乗解）

この問題では，n 変数の m 個の実関数 $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ と初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，

$$F(\boldsymbol{x}) = \sum_{i=1}^{m} \{f_i(\boldsymbol{x})\}^2 \tag{6.9}$$

が極小値をとるベクトル（極小点）$\boldsymbol{x}^*$ と，その関数値 $F(\boldsymbol{x}^*)$ を求める．ここで $\boldsymbol{x}$ は，$\boldsymbol{x} = (x_1, x_2, \ldots x_n)^T$ なるベクトルである．ただし，$m \geq n$ である．

すべての関数 $f_i(\boldsymbol{x})$ が $\boldsymbol{x}$ の 1 次式の場合は，線型最小二乗解の問題であり，そのための解法は「3.5 最小二乗解」で解説されている．（例えば，サブルーチン LAXL）．

一方，$f_i(\boldsymbol{x})$ のいずれかが $\boldsymbol{x}$ の非線型関数である場合に，この節で述べるサブルーチンを利用する．

さて，$\boldsymbol{x}^*$ の近似ベクトル $\boldsymbol{x}_k$ を $\Delta\boldsymbol{x}$ だけ変化させるとき，$F(\boldsymbol{x}_k + \Delta\boldsymbol{x})$ は (6.10) により近似される．

$$\begin{aligned} F(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k) &= \boldsymbol{f}^{\mathrm{T}}(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k)\boldsymbol{f}(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k) \\ &\approx \boldsymbol{f}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{f}(\boldsymbol{x}_k) + 2\boldsymbol{f}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{J}_k\Delta\boldsymbol{x}_k \\ &\quad + \Delta\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{J}_k\Delta\boldsymbol{x}_k \end{aligned} \tag{6.10}$$

ただし $|F(\boldsymbol{x}_k)|$ は十分小であると仮定する．

ここで，$\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^{\mathrm{T}}$ であり，$\boldsymbol{J}_k$ は $\boldsymbol{x}_k$ における $\boldsymbol{f}(\boldsymbol{x})$ のヤコビアン行列である．

この $F(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k)$ を最小化する $\Delta\boldsymbol{x}_k$ は (6.10) の右辺を微分して得られる連立 1 次方程式 (6.11) の解として与えられる．

$$\boldsymbol{J}_k^{\mathrm{T}}\ \boldsymbol{J}_k \Delta\boldsymbol{x}_k = -\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{x}_k) \tag{6.11}$$

(6.11) 式は正規方程式と呼ばれ，この $\Delta\boldsymbol{x}_k$ に基づく反復法が Newton-Gauss 法である．この方法は，$\Delta\boldsymbol{x}_k$ の方向としては $F(\boldsymbol{x})$ の降下方向ではあるが，$\Delta\boldsymbol{x}_k$ 自体は発散する可能性がある．

一方，$F(\boldsymbol{x})$ の $\boldsymbol{x}_k$ における傾斜ベクトル $\nabla F(\boldsymbol{x}_k)$ は

$$\nabla F(\boldsymbol{x}_k) = 2\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{x}_k) \tag{6.12}$$

で与えられる．$-\nabla F(\boldsymbol{x}_k)$ は $F(\boldsymbol{x})$ の $\boldsymbol{x}_k$ における最急降下方向であり，

$$\Delta\boldsymbol{x}_k = -\nabla F(\boldsymbol{x}_k) \tag{6.13}$$

とするのが，最急降下法である．(6.13) の $\Delta\boldsymbol{x}_k$ は，$F(\boldsymbol{x})$ の減少を最も確実に保証するが，反復を繰り返すとジグザグ運動を始めるという欠点がある．

Levenberg-Marquardt 法を基本とした公式

Newton-Gauss 法と最急降下法の折衷案として，Leverberg, Marquardt 及び Morrison は，次の方程式より $\Delta\boldsymbol{x}_k$ を決定することを提案した．

$$\left\{\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{J}_k + \nu_k^2\boldsymbol{I}\right\}\Delta\boldsymbol{x}_k = -\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{x}_k) \tag{6.14}$$

ここで ν_k は正数 (Marquardt 数と呼ぶ)である．

(6.14) により決定される $\Delta\boldsymbol{x}_k$ は，明らかに ν_k の値に依存する．すなわち，$\Delta\boldsymbol{x}_k$ の方向は，$\nu_k \to 0$ とすれば，Newton-Gauss 法の方向となり，$\nu_k \to \infty$ とすれば最急降下法の方向となる．

SSL II では，(6.14) に基づく反復公式を採用している．ただし，(6.14) を直接解くのではなく，数値的な安定性を保つために，(6.14) と同値な系

$$\begin{bmatrix} \boldsymbol{J}_k \\ \cdots\cdots \\ \nu_k\boldsymbol{I} \end{bmatrix} \Delta\boldsymbol{x}_k = -\begin{bmatrix} \mathbf{f}(\boldsymbol{x}_k) \\ \cdots\cdots\cdots \\ \boldsymbol{0} \end{bmatrix} \tag{6.15}$$

に対する最小二乗法（ハウスホルダー法）により求めている．

また，ν_k の値は反復の状況を観察しながら，適応的に決定する方法を採用している．

(1) サブルーチンの種類

利用者が関数 $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ 以外にヤコビアン行列 $\boldsymbol{J}$ を解析的に定義できるか否かの条件に応じて，表 6.3 に示すサブルーチンが用意されている．

表 6.3 制約なし関数二乗和の極小化のサブルーチン

解析的定義	サブルーチン名	備考
$f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$	NOLF1 (D15-10-0101)	改訂マルカート法
$f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}), \boldsymbol{J}$	NOLG1 (D15-20-0101)	改訂マルカート法

(2) 使用上の注意

a. 初期ベクトル $\boldsymbol{x}_0$ の与え方
初期ベクトル $\boldsymbol{x}_0$ は，可能なかぎり期待する極小点 $\boldsymbol{x}^*$ の近傍に取ることが望ましい．
関数 $F(\boldsymbol{x})$ がいくつかの極小点を持つ場合，初期ベクトルが適切でないと，期待する極小点 $\boldsymbol{x}^*$ に収束しない可能性がある．
通常，$\boldsymbol{x}_0$ は本サブルーチンを適用しようとする問題の性質をよく考察した上で設定されるべきであろう．

b. 関数計算プログラム
利用者は，関数 $\{f_i(\boldsymbol{x})\}$，又はヤコビアン行列 $\boldsymbol{J}$ の値を計算する副プログラムを用意する場合，効率の良いコーディングをすることが望ましい．SSL II のサブルーチンによる各関数の評価回数は，手法，初期ベクトルなどに依存す

るが，極値問題において関数計算の占める割合は大きく，全体の処理速度はコーディング方法に影響される．
更に，関数 $\{f_i(\boldsymbol{x})\}$ だけを与えるサブルーチンの場合は，通常ヤコビアン行列 $\boldsymbol{J}$ を差分により近似するため，桁落ちなどを避ける工夫も必要である．
また，関数 $f_i(\boldsymbol{x})$ の定義において，可能なかぎり変数ベクトル $\boldsymbol{x}$ の大きさが同程度となるように平衡化することが望ましい．
また，$f_i(\boldsymbol{x})$ 自体の値もできるだけ同程度になるように平衡化することが望ましい．

c. 収束判定値と極小値 $F(\boldsymbol{x}^*)$ の精度
極小化のアルゴリズムでは，極小点 $\boldsymbol{x}^*$ において，

$$\nabla F(\boldsymbol{x}^*) = 2\boldsymbol{J}^T \boldsymbol{f}(\boldsymbol{x}^*) = 0 \qquad (6.16)$$

となることを前提としている．すなわち反復公式は，関数 $F(\boldsymbol{x})$ を極小点 $\boldsymbol{x}^*$ の近傍では，2次関数

$$F(\boldsymbol{x}^* + \delta\boldsymbol{x}) \approx F(\boldsymbol{x}^*) + \delta\boldsymbol{x}^T\boldsymbol{J}^T\boldsymbol{J}\delta\boldsymbol{x} \qquad (6.17)$$

として近似している．
(6.17) は，$F(\boldsymbol{x})$ が適度に平衡化されているとき，$\boldsymbol{x}$ に対して ε 程度の摂動を与えた場合に，関数 $F(\boldsymbol{x})$ は ε^2 程度の変化率を示すことを意味している．
ところで，SSL II での収束判定条件は，反復ベクトルを $\boldsymbol{x}_k$ と表すと，

$$F(\boldsymbol{x}_{k+1}) < F(\boldsymbol{x}_k) \text{かつ} \quad \left\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right\|_2 \le \max(1.0, \left\|\boldsymbol{x}_k\right\|_2)\cdot\varepsilon \qquad (6.18)$$

ここで ε は収束判定値

を満たした場合に $\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$ とみなす．
したがって，極小値 $F(\boldsymbol{x}^*)$ を丸め誤差程度まで正しく求めるためには，収束判定値 ε を

$\varepsilon \approx \sqrt{u}$, u は丸め誤差の単位

程度に与えるのが適当である．
SSL II での標準収束判定値は，$2\cdot\sqrt{u}$ を採用している．

6.5 線形計画問題

いくつかの変数に関する一次不等式と一次等式の組合せによって表される制約条件のもとで，与えられた一次関数を最小（又は最大）にする変数の値及びその関数の最小値（又は最大値）を求める問題を，線形計画問題という．
次の形の線形計画問題を標準形という．
“条件 $\boldsymbol{Ax} = \boldsymbol{d}$ (6.19)
$\boldsymbol{x} \ge \boldsymbol{0}$ (6.20)

のもとで，線形の目的関数

$$z = \boldsymbol{c}^T\boldsymbol{x} + c_0$$

を最小にせよ”
ここで，$\boldsymbol{A}$ は $m \times n$ なる係数行列で，

$\text{rank}\,\boldsymbol{A} = m \le n$

とする．また，$\boldsymbol{x} = (x_1, x_2, \ldots x_n)^{\mathrm{T}}$ は変数ベクトル，$\boldsymbol{d} = (d_1, d_2, \ldots d_m)^T$ は定数ベクトル，$\boldsymbol{c} = (c_1, c_2, \ldots c_n)^{\mathrm{T}}$ は係数ベクトル，c_0 は定数項である．
いま，$\boldsymbol{A}$ の第 j 列（x_j の係数ベクトル）を $\boldsymbol{a}_j$ で表すことにする．$\boldsymbol{A}$ の m 個の列

$$\boldsymbol{a}_{k_1}, \boldsymbol{a}_{k_2}, \ldots, \boldsymbol{a}_{k_m}$$

が 1 次独立であるとき，対応する変数の組 $(x_{k_1}, x_{k_2}, \ldots x_{k_m})$ を基底といい，x_{k_i} を（i 番目の）基底変数という．非基底変数（基底変数でない変数）の値をすべて 0 とおいて得られる (6.19) の解を基底解といい，その中で，(6.20) も満たすものを可能基底解という．(6.19) (6.20) を満たす解が存在するならば，可能基底解が存在し，目的関数の値を最小にする最適解が存在するならば，可能基底解の中に最適解が存在することが示されている．（線形計画法の基本定理）．一つの可能基底解から出発して，基底変数を一つずつ交換することにより，z がより小さくなるような可能基底解を順次求めていって，最適解を求める方法をシンプレックス法という．シンプレックス法の反復計算において，交換すべき基底変数の決定に必要な係数や定数項だけを，基底行列

$$\boldsymbol{B} = [\boldsymbol{a}_{k_1}, \boldsymbol{a}_{k_2}, \ldots, \boldsymbol{a}_{k_m}]$$

の逆行列と，はじめの係数 $\boldsymbol{A}$，$\boldsymbol{c}$ 及び定数項 $\boldsymbol{d}$ から計算する方法を改訂シンプレックス法という．
SSL II では，この改訂シンプレックス法を採用したサブルーチン LPRS1 を用意している．
制約条件の中に不等式条件が含まれているときは，変数を付加して等式条件に直す．例えば，

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le d_1$$

は，

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + x_{n+1} = d_1,\ x_{n+1} \ge 0$$

とする．また，

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \geq d_2$$

は,

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n - x_{n+2} = d_2,\ x_{n+2} \geq 0$$

とする．x_{n+1} や x_{n+2} のように不等式を等式に直すために付加する非負変数をスラック変数という．
最大化問題は，目的関数に -1 をかけることにより最小化問題に変えられる．
LPRS1 では，以上の処理を行うので，与える問題は，不等式条件を含んでいてもよいし，最大化問題であってもよい．
可能基底解が得られていないときは，解法を 2 段階に分けて，第 1 段階で可能基底解を求め，可能基底解が得られたら，第 2 段階で最適解を求める．このような方法を 2 段階法という．第 1 段階では，次の問題の最適解を求める．

“条件

$$\boldsymbol{A}\boldsymbol{x} + \boldsymbol{A}^{(a)}\boldsymbol{x}^{(a)} = \boldsymbol{d},$$
$$\boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{x}^{(a)} \geq \boldsymbol{0}$$

のもとで,

$$z_1 = \sum_{i=1}^{m} x_i^{(a)}$$

を最小にせよ”

ここで，$\boldsymbol{x}^{(a)}$ は $\boldsymbol{x}^{(a)} = (x_1^{(a)}, x_2^{(a)}, \ldots, x_m^{(a)})^T$ なる m 次元ベクトル，$\boldsymbol{A}^{(a)}$ は $\boldsymbol{A}^{(a)} = (a_{ij}^{(a)})$ なる m 次の対角行列で,

$$a_{ii}^{(a)} = \begin{cases} 1 & d_i \geq 0 \quad \text{のとき,} \\ -1 & d_i < 0 \quad \text{のとき,} \end{cases}$$

である．$x_i^{(a)}$ を人為変数という．
この最適解が得られたとき，$z_1 > 0$ の場合は，(6.19)，(6.20) を満たす $\boldsymbol{x}$ は存在しない．
一方，$z_1 = 0$ すなわち $\boldsymbol{x}^{(a)} = \boldsymbol{0}$ であれば，与えられた問題の可能基底解が得られているので，第 2 段階に進む．
ただし,

$$\text{rank}\,\boldsymbol{A} < m$$

であって，(6.19) を満たす $\boldsymbol{x}$ が存在するときは,

$$r = \text{rank}\,\boldsymbol{A}$$

とおくと，$(m\text{-}r)$ 個の条件式はむだである．(r 個の条件式が成り立てば，他は必ず成り立つ．) 第 1 段階の最適解は，むだな条件式を除いたあとの可能基底解になっている．この場合基底変数の中に $(m\text{-}r)$ 個，人為変数がのこっているが，それらに対応する条件式 （i 番目の基底変数は i 番目の条件式に対応している）がむだな条件式である．

(1) 使用上の注意

a. 係数の相対零判定

LPRS1 では，反復過程での要素の絶対値がある程度小さくなった場合に，それを零とみなす相対零判定を行っており，そのための判定値をパラメタ EPSZ に入力するようになっている．
いま，係数行列 $\boldsymbol{A}$, 定数ベクトル $\boldsymbol{d}$, 係数ベクトル $\boldsymbol{c}$ 及び定数項 c_0 とからなる，次のような拡張された係数行列を考える．

$$\left[\begin{array}{c:c} \boldsymbol{A} & \boldsymbol{d} \\ \hdashline \boldsymbol{c} & c_0 \end{array}\right]$$

この行列の絶対値最大要素を a_{max} とする．
そこで，反復計算で求められる係数ベクトル，定数項等の絶対値が $a_{max} \cdot \text{EPSZ}$ より小さい場合，零とみなす．
EPSZ の値が適切でないと，可能解が存在するのに，第 1 段階の最適解が求められたときに，$z_1 > 0$ になることがある．更に，第 1 段階で最適解が得られる前に，反復計算が行えなくなることがあるので注意すること．
なお，精度を維持するためには，拡張された係数行列の要素の絶対値最大と最小の比ができるだけ小さくなるように，行または列毎に適当な定数を乗じておくとよい．

b. 反復回数

LPRS1 では，一つの可能基底解が 2 度以上現れないように基底変数の交換を行っているので，有限回の反復で最適解が得られるか又は存在しないことが分かる．しかし，適当な反復回数で処理を打ち切ることもできるように，その上限を指定するパラメタ IMAX が用意されている．
もし，反復回数が上限に到達して処理が打ち切られても，可能基底解が得られている（第 2 段階に入っている）ならば，計算を続行させることもできる．

6.6 非線形計画問題（制約付き多変数関数の極小化）

この問題では，n 変数の実関数 $f(\boldsymbol{x})$ と初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，制約条件

$$c_i(\boldsymbol{x}) = 0,\ i = 1, 2, ..., m_1 \qquad (6.21)$$
$$c_i(\boldsymbol{x}) \geq 0,\ i = m_1 + 1, ..., m_1 + m_2 \qquad (6.22)$$

のもとで，$f(\boldsymbol{x})$ の極小値を与える $\boldsymbol{x}^*$ とその関数値 $f(\boldsymbol{x}^*)$ を求める．ここでは $\boldsymbol{x}$ は，$\boldsymbol{x} = (x_1, x_2, ..., x_n)^T$ なるベクトル，m_1 及び m_2 はそれぞれ等式制約，不等式制約の個数である．

この制約付き極小化の解法は，基本的には 6.3 節で述べた制約なし極小化の解法に対して，(6.21), (6.22) の条件に基づく制約を付加したものである．すなわち，極小点の近似点 $\boldsymbol{x}_k$ における $f(\boldsymbol{x})$ の 2 次近似

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_k) + \boldsymbol{y}^{\mathrm{T}}\boldsymbol{g}_k + \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{y} \qquad (6.23)$$

ここで $\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{x}_k$, $\boldsymbol{B}$ はヘシアン行列

に基づく極小化において，(6.21), (6.22) の 1 次近似よりなる制約

$$c_i(\mathbf{x}_\mathbf{k}) + \mathbf{y}^T \nabla c_i(\mathbf{x}_\mathbf{k}) = 0, \quad i = 1,2,\ldots,m_1 \qquad (6.24)$$

$$c_i(\mathbf{x}_\mathbf{k}) + \mathbf{y}^T \nabla c_i(\mathbf{x}_\mathbf{k}) \geq 0, \quad i = m_1 + 1,\ldots,m_1 + m_2 \qquad (6.25)$$

∇c_i は c_i の勾配ベクトル

を満たす解を求めるわけである．これは，$\boldsymbol{y}$ に関する 2 次計画問題である．

SSL II では，反復の過程で逐次 2 次計画問題を解く方法を採用した NLPG1 を用意している．

第 7 章
補間・近似

7.1 概要

この章では，以下の問題を取り上げる．

補間：　離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して関数値 $y_i = f(x_i)$, $i = 1, ..., n$ が与えられたとき（場合によっては $y'_i = f'(x_i)$ も与えられたとき)，それらの点を通るような， $f(x)$ の近似式（以後，補間式と言う）を求める．あるいはその補間式を使って，与えられた離散点以外の点 $x = v$ における $f(x)$ の近似値（以後，補間値という）を求める．

最小二乗近似：離散点 $x_1, x_2 ... x_n$ $(x_1 < x_2 < ... < x_n)$ に対して観測値，y_i, $i = 1, ..., n$ が与えられたとき，

$$\sum_{i=1}^{n} w(x_i)\{y_i - \bar{y}_m(x_i)\}^2, w(x_i) \geq 0$$

を最小にする近似式 $\bar{y}_m(x)$ を求める．ここで $w(x)$ は重み関数であり，$\bar{y}_m(x)$ は m 次の多項式である．

この問題は，y_i が実験データであってかつ y_i の観測誤差の大きさが個々にまちまちである場合に適用される．

平滑化：　離散点 $x_1, x_2 ... x_n$ $(x_1 < x_2 < ... < x_n)$ に対して観測値 y_i, $i = 1, ...n$ が与えられたとき$\{y_i\}$に含まれる観測誤差を消して，真の関数をより正しく表すような新しい点列 $\{\bar{y}_i\}$ を求める．

以後，この操作を平滑化と言い，$\bar{y}_i$ (あるいは $\{\bar{y}_i\}$) を y_i (あるいは $\{y_i\}$) に対する平滑値，$|y_i - \bar{y}_i|$ を y_i に対する平滑化の程度，平滑化に用いる近似式を平滑化式と言う．

級数：　任意の有限区間上で滑らかな関数 $f(x)$ の関数値計算が複雑で手間がかかったり，$f(x)$ の微分積分が解析的にできない場合，$f(x)$ を一旦チェビシェフ級数に展開すればよい．

チェビシェフ級数展開の特徴は，次のとおりである．

- 級数の収束が速い．
- 項別の微分積分が容易である．
- 高速フーリェ変換の手法を用いるので能率がよく，数値的に安定である．

そこで，要求精度に応じて項数 n と，n 個のチェビシェフ展開係数を求める．更に，得られた級数の項別微分，積分によって $f(x)$ の導関数，不定積分を級数の形で求める．また，これらの級数を求和することにより，関数値，微係数，定積分を求める．

関数 $f(x)$ が任意の周期を持つ滑らかな周期関数である場合は，三角級数に展開できる．ここでは，偶関数，奇関数を要求精度に応じて，それぞれ cosine 級数，sine 級数に展開する．

この章の補間及び平滑化の分野をはじめ，第 9 章の数値微積分においても，“スプライン関数”と呼ばれる関数が有力な道具として使われる．それで以下に，スプライン関数の定義及びその表現などについて述べる．

スプライン (Spline) 関数について

(1) Spline 関数の定義

ここで述べる Spline 関数とは次のような関数である．

すなわち，

$$a = x_0 < x_1 < ... < x_n = b \qquad (7.1)$$

を実軸上の区間 $[a, b]$ の分割とし，$D \equiv d/dx$ として，

a. 各部分区間 (x_i, x_{i+1}) で $D^k S(x) = 0$

b. $S(x) \in C^{k-2}[a, b]$ (7.2)

を満たすような関数 $S(x)$ を k-1 次の Spline 関数という．また，点列 $\{x_i\}$ を Spline 関数の節点という．

(7.2) から分かるように，$S(x)$は各部分区間 $(x_i,\ x_{i+1})$ において別々に定義される高々 k-1 次の多項式であって，しかも全区間 $[a,\ b]$ においては k-2 次までの導関数が連続であるような関数である．

(2)　Spline 関数の表現 – その 1

任意の $S(x)$ は次の式で表現される．すなわち，

$a_j, j = 0, 1, ...,\ k-1$ 及び $b_i, i = 1, 2, ...,\ n-1$ を任意の定数として，

$$\begin{cases} S(x) = p(x) + \sum_{i=1}^{n-1} b_i (x - x_i)_+^{k-1} \\ \text{ここで} \\ \qquad p(x) = \sum_{j=0}^{k-1} a_j (x - x_0)^j \end{cases} \tag{7.3}$$

である．ただし記号 $(x - x_i)_+^{k-1}$ は (7.4) を意味する．

$$(x - x_i)_+^{k-1} = \begin{cases} (x - x_i)^{k-1}, & x \ge x_i \\ 0, & x < x_i \end{cases} \tag{7.4}$$

式 (7.3) が (7.2) を満たすことを示そう．

式 (7.3) において，x を x_0 から右へ動かしてみる．

$x_0 \le x < x_1$ では，

$$S(x) = p(x)$$

であるから $S(x)$ は k-1 次の多項式である．

$x_1 \le x < x_2$ では，

$$S(x) = p(x) + b_1(x - x_1)^{k-1}$$

であるから $S(x)$ はやはり k-1 次の多項式である．

一般に，$x_i \le x < x_{i+1}$ では，

$$S(x) = p(x) + \sum_{r=1}^{i} b_r (x - x_r)^{k-1}$$

であるから $S(x)$ は各区間で別々に定義される k-1 次の多項式であることは容易に分かる．

式 (7.3) から，

$$\begin{aligned} &\frac{d^l}{dx^l} S(x) = S^{(l)}(x) \\ &= \sum_{j=1}^{k-1} j(j-1)\cdots(j-l+1) a_j (x - x_0)^{j-l} \\ &+ \sum_{i=1}^{n-1} (k-1)(k-2)\cdots(k-l) b_i (x - x_i)_+^{k-1-l} \end{aligned}$$

である．したがって，$S(x)$ の x_i における左側 l 階微係数，右側 l 階微係数は各々，

$$\begin{aligned} \lim_{\varepsilon \to 0} S^{(l)}(x_i - \varepsilon) &= \sum_{j=1}^{k-1} j(j-1)\cdots(j-l+1) a_j (x_i - x_0)^{j-l} \\ &+ \sum_{r=1}^{i-1} (k-1)(k-2)\cdots(k-l) b_r (x_i - x_r)^{k-1-l} \\ \lim_{\varepsilon \to 0} S^{(l)}(x_i + \varepsilon) &= \sum_{j=1}^{k-1} j(j-1)\cdots(j-l+1) a_j (x_i - x_0)^{j-l} \\ &+ \sum_{r=1}^{i-1} (k-1)(k-2)\cdots(k-l) b_r (x_i - x_r)^{k-1-l} \\ &+ \lim_{\varepsilon \to 0} (k-1)(k-2)\cdots(k-l) b_i (x_i + \varepsilon - x_i)^{k-1-l} \end{aligned}$$

となる．このことから，

$$\begin{aligned} &\lim_{\varepsilon \to 0} S^{(l)}(x_i + \varepsilon) - \lim_{\varepsilon \to 0} S^{(l)}(x_i - \varepsilon) \\ &\qquad = \lim_{\varepsilon \to 0} (k-1)(k-2)\cdots(k-l) b_i \varepsilon^{k-1-l} \end{aligned} \tag{7.5}$$

である．

$l = 0, 1, ...,\ k-2$ に対しては，(7.5) の右辺は 0 であるから，

$$\lim_{\varepsilon \to 0} S^{(l)}(x_i + \varepsilon) = \lim_{\varepsilon \to 0} S^{(l)}(x_i - \varepsilon) \tag{7.6}$$

であり，$S^{(l)}(x)$ は $x = x_i$ で連続である．

一方，$l = k-1$ に対しては，(7.5) の右辺は $(k-1)\ (k-2)\ \cdots\ 1 \cdot b_i$ であり，一般に $b_i \ne 0$ であるから，

$$\lim_{\varepsilon \to 0} S^{(k-1)}(x_i + \varepsilon) \ne \lim_{\varepsilon \to 0} S^{(k-1)}(x_i - \varepsilon) \tag{7.7}$$

となり $S(x)$ の k-1 次導関数は $x = x_i$ で不連続となる．もしすべての $b_i, i = 1, 2, ..., n-1$ が 0 のときは，$S(x)$ は全区間 $[a,\ b]$ で k-1 次導関数までが連続になり，この場合 (7.3) から明らかなように全区間 $[a,\ b]$ で $S(x) = p(x)$ である．これは $[a,\ b]$ で定義される k-1 次多項式の $x = x_0$ を中心とするべき級数展開にほかならない．このことから，$[a,\ b]$ で定義される任意の多項式は，Spline 関数の特殊形であることが分かる．

式 (7.3) Spline 関数の打切りベキ関数 (truncated power function) による表現と呼ぶ．この表現は Spline 関数の性質を直観的に表す式ではあるが，$(x - x_i)_+^{k-1}$ などは絶対値が大きくなりがちであるから，この表現は一般に数値的に不安定である．

(3)　Spline 関数の表現 – その 2 (B-spline の導入)

(7.3) に対して，以下に定義される B-spline による表現には数値的な不都合が少ない．今，点列$\{t_r\}$を，

$$t_{-k+1} \le t_{-k+2} \le \ldots \le t_{-1} \le t_0 = x_0 < t_1 = x_1 < \ldots < t_n = x_n \le t_{n+1} \le t_{n+2} \le \ldots \le t_{n+k-1} \tag{7.8}$$

のようにとる（図 7.1 参照）．また，$g_k\ (t;\ x)$ を x をパラメタとする t の関数として，

$$g_k(t;x) = (t-x)_+^{k-1} = \begin{cases} (t-x)^{k-1}, & t \ge x \\ 0, & t < x \end{cases} \tag{7.9}$$

とする（図 7.2 参照）．

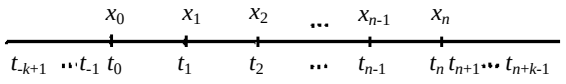


図 7.1　点列 $\{t_r\}$ の採り方

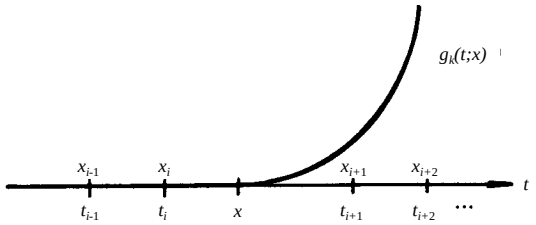


図　7.2　$g_k\ (t;\ x)$

このとき (7.9) の $t = t_j,\ t_{j+1},\ \ldots,\ t_{j+k}$ における k 階差分商[(1)] の定数倍，

$$N_{j,k}(x) = (t_{j+k} - t_j)\, g_k\left[t_j, t_{j+1}, \ldots, t_{j+k}; x\right] \tag{7.10}$$

を，$k-1$ 次の正規化された B-spline（又は単に B-spline）と呼ぶ．

ここで B-spline $N_{j,k}\ (x)$ の性質について述べる．今，$t_j,\ t_{j+1},\ \ldots,\ t_{j+k}$ を固定して x を動かして見る．$x \le t_j$ のとき $N_{j,k}(x)$ は t についての $k-1$ 次多項式の k 階差分商の定数倍であるから 0 である．また，$t_{j+k} \le x$ のとき，$N_{j,k}(x)$ は恒等的に 0 であるような関数の k 階差分商の定数倍であるから 0 である．

$t_j < x < t_{j+k}$ のときは $N_{j,k}(x)$ は 0 ではない．これをまとめると，

$$N_{j,k}(x) \begin{cases} \ne 0 & ,t_j < x < t_{j+k} \\ = 0 & ,x \le t_j \text{及び} t_{j+k} \le x \end{cases} \tag{7.11}$$

である（実際には $t_j < x < t_{j+k}$ において $0 < N_{j,k}(x) \le 1$ である）．次に x を固定して j を変えてみる．今，$t_i = x_i < x < x_{i+1} = t_{i+1}$ とする．このとき上と同じような考えで，

$$N_{j,k}(x) \begin{cases} \ne 0 & ,i-k+1 \le j \le i \\ = 0 & ,j \le i-k \text{ 及び } i+1 \le j \end{cases} \tag{7.12}$$

がいえる．

以上の (7.11)，(7.12) を B-spline の局所性と呼ぶことにする．

ところで，この B-spline $N_{j,k}(x)$ は (7.10) より，

$$N_{j,k}(x) = (t_{j+k} - t_j) \sum_{r=j}^{j+k} \frac{(t_r - x)_+^{k-1}}{(t_r - t_j)\ldots(t_r - t_{r-1})(t_r - t_{r+1})\ldots(t_r - t_{j+k})} \tag{7.13}$$

となる．したがって，$N_{j,k}(x)$ は各区間 $(x_i,\ x_{i+1})$ で別々に定義される k-1 次の多項式であり，k-2 次導関数までが連続である．$N_{j,k}(x)$ のこの性質に基づいて，(7.2) を満たす任意の spline 関数 $S(x)$ は，$c_j,\ j = -k+1,\ -k+2,\ \ldots.,\ n-1$ を定数として，

$$S(x) = \sum_{j=-k+1}^{n-1} c_j N_{j,k}(x) \tag{7.14}$$

と表現できることが証明される．

---

(1) 差分商 (divided difference) の定義

関数 $f(t)$ の $t = t_j,\ t_{j+1},\ldots,\ t_{j+k}\ (t_j \le t_{j+1} \le \ldots \le t_{j+k})$ における k 階差分商 $f\,[t_j,\ t_{j+1},\ldots,\ t_{j+k}]$ とは次のように再帰的に定義される量である．

$$f[t_j, t_{j+1}, \ldots, t_{j+k}] = \begin{cases} \dfrac{f[t_{j+1}, t_{j+2}, \ldots, t_{j+k}] - f[t_j, t_{j+1}, \ldots, t_{j+k-1}]}{t_{j+k} - t_j} & : t_{j+k} \ne t_j \\ \dfrac{1}{k!} \dfrac{d^k f(t_j)}{dt^k} & : t_{j+k} = t_j \end{cases}$$

ただし，$f[t_j] = f(t_j)$

また，$t_r \ne t_s : r \ne s$ のとき，以下の 2 式が成り立つ．

$$f[t_j, t_{j+1}, \ldots, t_{j+k}] = \sum_{r=j}^{j+k} \frac{f(t_r)}{(t_r - t_j)\ldots(t_r - t_{r-1})\ (t_r - t_{r+1})\ldots(t_r - t_{j+k})}$$

$$f[t_j, t_{j+1}, \ldots, \overbrace{t_i, t_i, \ldots, t_i}^{r+1}, t_{i+1}, \ldots, t_{j+k}] = \frac{1}{r!} \frac{\partial^r}{\partial t_i^r} f[t_j, t_{j+1}, \ldots, t_{i-1}, t_i, t_{i+1}, \ldots, t_{j+k}]$$

なお，差分商についての詳細は例えば[51]pp.51-57 を参照すること．

(4) Spline 関数の計算法

$k-1$ 次の spline 関数

$$S(x)=\sum_{j=-k+1}^{n-1}c_j N_{j,k}(x) \tag{7.15}$$

が与えられたとき $x \in [x_i, x_{i+1})$ における関数値，微係数，あるいは積分

$$\int_{x_0}^{x} S(y)dy$$

などの計算のしかたについて述べる．

a. 関数値の計算

$x \in [x_i, x_{i+1})$ における $S(x)$ の値は，$N_{j,k}(x)$ を計算することにより計算される．ところが，$N_{j,k}(x)$ の局所性 (7.12) により非零のものだけを計算すればよい．

$N_{j,k}(x)$ の計算は，次の漸化式に基づく．

$$N_{r,s}(x)=\frac{x-t_r}{t_{r+s-1}-t_r}N_{r,s-1}(x)+\frac{t_{r+s}-x}{t_{r+s}-t_{r+1}}N_{r+1,s-1}(x) \tag{7.16}$$

ここで，

$$\begin{aligned}N_{r,1}(x)&=(t_{r+1}-t_r)g_1[t_r,t_{r+1};x]\\&=(t_{r+1}-t_r)\frac{g_1(t_{r+1};x)-g_1(t_r;x)}{t_{r+1}-t_r}\\&=(t_{r+1}-x)_+^0-(t_r-x)_+^0\\&=\begin{cases}1, & r=i\\0, & r\neq i\end{cases}\end{aligned} \tag{7.17}$$

そこで (7.16)，(7.17) を使って

$s = 2, 3, ..., k, r = i-s+1, i-s+2, ..., i$

に対して適用すれば，図 7.3 に挙げたすべての $N_{r,s}(x)$ を計算することができ，最右列のものが $S(x)$ の計算に使われる．

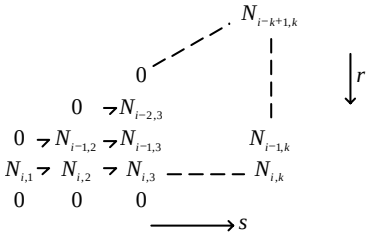


図 7.3　$N_{r,s}(x)$ の計算，ただし $x \in [x_i, x_{i+1})$

b. 微係数，積分の計算

$$\frac{d^l}{dx^l}S(x)=S^{(l)}(x)=\sum_{j=-k+1}^{n-1}c_j N_{j,k}^{(l)}(x) \tag{7.18}$$

より，$S^{(l)}(x)$ は $N_{j,k}^{(l)}(x)$ を計算することにより求まる．(7.9) から

$$\frac{\partial^l}{\partial x^l}g_k(t;x)=\frac{(-1)^l(k-1)!}{(k-1-l)!}(t-x)_+^{k-1-l} \tag{7.19}$$

となるが，$N_{j,k}^{(l)}(x)$ は (7.19) の $t = t_j, t_{j+1}, ..., t_{j+k}$ における k 階差分商である．今，

$$d_k(t;x)=(t-x)_+^{k-1-l}=\begin{cases}(t-x)^{k-1-l}, & t\geq x\\0, & t<x\end{cases}$$

とし，これの，$t = t_j, t_{j+1},\cdots, t_{j+k,}$における k 階差分商を $D_{j,k}(x)$ とする．すなわち，

$$D_{j,k}(x)=d_k\left[t_j,t_{j+1},...,t_{j+k};x\right] \tag{7.20}$$

この $D_{j,k}(x)$ は次の漸化式より計算できる．

$x \in [x_i, x_{i+1})$ として，

$$\begin{aligned}D_{r,1}(x)&=\begin{cases}1/(x_{i+1}-x_i), & r=i\\0, & r\neq i\end{cases}\\D_{r,s}(x)&=\frac{D_{r+1,s-1}(x)-D_{r,s-1}(x)}{t_{r+s}-t_s}, \quad 2\leq s\leq l+1\\D_{r,s}(x)&=\frac{(x-t_r)D_{r,s-1}(x)+(t_{r+s}-x)D_{r+1,s-1}(x)}{t_{r+s}-t_r}\\&, \quad l+2\leq s\leq k\end{aligned} \tag{7.21}$$

より，$s = 2, 3, ..., k, r = i-s+1, i-s+2, ..., i$ に対して適用すれば，$D_{j,k}$, $i-k+1 \leq j \leq i$ を得ることができる．目的の $N_{j,k}^{(l)}(x)$ は，

$$N_{j,k}^{(l)}(x)=(t_{j+k}-t_j)\frac{(-1)^l(k-1)!}{(k-1-l)!}D_{j,k}(x)$$

として求まり，これを使って $S^{(l)}(x)$ は計算できる．

次に積分は，

$$I=\int_{x_0}^{x}S(y)dy=\sum_{j=-k+1}^{n-1}c_j\int_{x_0}^{x}N_{j,k}(y)dy \tag{7.22}$$

であるから，$\int_{x_0}^{x}N_{j,k}(y)dy$ を計算することにより求まる．この $N_{j,k}(x)$ を積分するには，$N_{j,k}(x)$ が含む差分商の演算と積分の演算の順序を交換することによりなされる．

まず，(7.9) より $g_k(t\ ;\ x)$ の不定積分は，積分定数を省略して

$$\int g_k(t;x)dx = -\frac{1}{k}(t-x)_+^k$$

である．$e_k(t;x) = (t-x)^k_+$ として，これの $t = t_j, t_{j+1}, ..., t_{j+k}$ における k 階差分商を

$$I_{j,k}(x) = e_k[t_j, t_{j+1}, ..., t_{j+k}\,; x] \qquad (7.23)$$

とすれば，$I_{j,k}(x)$ は次の漸化式を満たす．$x \in [x_i, x_{i+1})$ として

$$I_{r,1}(x) = \begin{cases} 0\,, & r \le i-1 \\ (x_{i+1}-x)/(x_{i+1}-x_i), & r = i \\ 1\,, & r \ge i+1 \end{cases}$$

$$I_{r,s}(x) = \frac{(x-t_r)I_{r,s-1}(x) + (t_{r+s}-x)I_{r+1,s-1}(x)}{t_{r+s}-t_r} \qquad (7.24)$$

この (7.24) を $s = 2, 3, ..., k$, $r = i-s+1, i-s+2, ..., i$ について適用すれば $I_{j,k}(x)$ は図 7.4 に挙げるように最右列として求まる．

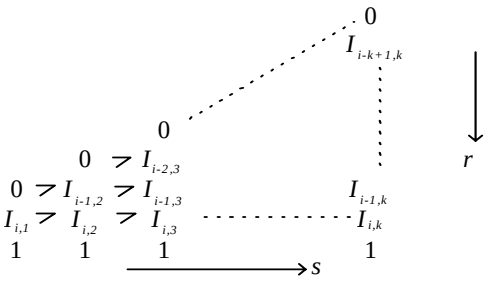


図 7.4　$I_{r,s}(x)$ の計算，ただし $x \in [x_i, x_{i+1})$

$$\int_{x_0}^{x} N_{j,k}(y)dy = -\frac{(t_{i+k}-t_j)}{k}\left[I_{j,k}(x) - I_{j,k}(x_0)\right] = \frac{(t_{j+k}-t_j)}{k}\left[I_{j,k}(x_0) - I_{j,k}(x)\right]$$

よって (7.22) より

$$I = \int_{x_0}^{x} S(y)dy = \frac{1}{k}\sum_{j=-k+1}^{n-1} c_j\left(t_{j+k}-t_j\right)\left[I_{j,k}(x_0) - I_{j,k}(x)\right]$$
$$= \frac{1}{k}\left\{ \sum_{j=-k+1}^{0} c_j\left(t_{j+k}-t_j\right) I_{j,k}(x_0) - \sum_{j=i-k+1}^{i} c_j\left(t_{j+k}-t_j\right) I_{j,k}(x) + \sum_{j=1}^{i} c_j\left(t_{j+k}-t_j\right) \right\} \qquad (7.25)$$

として得られる．

以上，Spline 関数 $S(x)$ の関数値，微係数及び積分の計算法を述べたが，今まで，(7.15) の係数 c_j は既知と仮定してきた．c_j の決定は $S(x)$ が補間式ならば補間条件から，あるいは，$S(x)$ が平滑化式ならば最小二乗近似的条件から決められる．例えば，(7.15) では，$n+k-1$ 個の係数 c_j $(-k+1 \le j \le n-1)$ を含むので (7.15) が補間式として用いられるなら，$n+k-1$ 個の補間条件を与えることにより決定される．仮に図 7.1 において，$x = x_0, x_1, ..., x_n$ という $n+1$ 個の点で関数値が与えられているとしても，補間条件より $n+k-1$ 個の c_j を決めようとする場合，あと $(n+k-1)-(n+1) = k-2$ 個の点で関数が与えられるか，それとも $k-2$ 個の別な条件（例えば $S(x)$ の微係数に対する条件）が付加されなくてはならない．これについてはあとの「7.2 補間」で述べる．

SSL II では，(7.15) で表現される Spline 関数を，補間，数値微分，数値積分，及び最小二乗近似による平滑化に応用する．

(5)　2 変数 Spline 関数の定義，表現及び計算法

2 変数 Spline 関数は，先に述べた 1 変数の場合の拡張として定義される．

まず，$x\,y$ 平面上の閉領域 $R = \{(x,y) \mid a \le x \le b,\ c \le y \le d\}$ を考え，この上に分割 (7.26) に従って，格子点 (x_i, y_j)　$0 \le i \le m$ ，$0 \le j \le n$ を作る．

$$a = x_0 < x_1 < \cdots\cdots < x_m = b$$
$$c = y_0 < y_1 < \cdots\cdots < y_n = d \qquad (7.26)$$

このとき，$D_x = \partial/\partial x$ ，$D_y = \partial/\partial y$ として

a.　各部分開領域

$$R_{i,j} = \left\{(x,y) \middle| x_i < x < x_{i+1}, y_j < y < y_{j+1}\right\} \qquad (7.27)$$

上で $D_x^{\,k}\,S(x,y) = D_y^{\,k}\,S(x,y) = 0$

b.　$S(x,y) \in C^{k-2,k-2}[R]$

を満たすような関数 $S(x,y)$ を双 $k-1$ 次の 2 変数 Spline 関数という．(7.27) の a.は，$S\,(x,y)$ が各 $R_{i,j}$ 上で x ，y についての多項式であり，x についても y についても高々 $k-1$ 次であることを表し，更に b. は，$S\,(x,y)$ が R 全体では，$\lambda = 0, 1, .., k-2$ ，$\mu = 0, 1, ..., k-2$ とするとき，

$$\frac{\partial^{\lambda+\mu}}{\partial x^{\lambda}\partial y^{\mu}} S(x,y)$$

が存在してしかも連続であることを表す．

今，点列 $\{s_i\},\{t_j\}$ を次のようにとる．

$$s_{-k+1} \le s_{-k+2} \le \cdots \le s_{-1} \le s_0 = x_0 < s_1 = x_1 < \cdots < s_m = x_m \le s_{m+1} \le \cdots \le s_{m+k-1}$$

$$t_{-k+1} \le t_{-k+2} \le \cdots \le t_{-1} \le t_0 = y_0 < t_1 = y_1 < \cdots < t_n = y_n \le t_{n+1} \le \cdots \le t_{n+k-1}$$

これに基づいて，x 方向，y 方向の $k-1$ 次の B-spline を，1 変数の場合と同様に

$$N_{\alpha,k}(x)=(s_{\alpha+k}-s_{\alpha})\, g_k[s_{\alpha}, s_{\alpha+1}, \cdots\cdots, s_{\alpha+k}; x]$$
$$N_{\beta,k}(y)=(t_{\beta+k}-t_{\beta})\, g_k[t_{\beta}, t_{\beta+1}, \cdots\cdots, t_{\beta+k}; y]$$

と定義する．このとき上で定義した双 $k-1$ 次の 2 変数 Spline 関数は，$c_{\alpha,\beta}$ を任意定数として

$$S(x,y)=\sum_{\beta=-k+1}^{n-1}\sum_{\alpha=-k+1}^{m-1} c_{\alpha,\beta}N_{\alpha,k}(x)N_{\beta,k}(y) \quad (7.28)$$

という形で書き表すことができる．

$S(x,y)$ の関数値，偏微分あるいは不定積分の計算は，(7.28) の表現を使えば，1 変数の場合の計算を単に応用するだけでできる．

まず $\lambda \geq 0$ ，$\mu \geq 0$ として

$$S^{(\lambda,\mu)}(x,y)=\frac{\partial^{\lambda+\mu}}{\partial x^{\lambda}\partial y^{\mu}}S(x,y)$$
$$=\sum_{\beta=-k+1}^{n-1}\sum_{\alpha=-k+1}^{m-1} c_{\alpha,\beta}N_{\alpha,k}^{(\lambda)}(x)N_{\beta,k}^{(\mu)}(y) \quad (7.29)$$

と書けるから，関数値，偏微分の計算は $N_{\alpha,k}^{(\lambda)}(x)$，$N_{\beta,k}^{(\mu)}(y)$ を別々に計算することによりなされ，これは，先に述べた 1 変数の場合の手法を適用するだけでよい．

次に，$S(x,y)$ を y については μ 階偏微分し，x については積分した量

$$S^{(-1,\mu)}(x,y)=\int_{x_0}^{x}\frac{\partial^{\mu}S(x,y)}{\partial y^{\mu}}dx \quad (7.30)$$

を考える．ここで，偏微分と積分の順序を逆にしても値は同じであることに注意する．

(7.28) を使って (7.30) の右辺を書き直せば

$$\sum_{\alpha=-k+1}^{m-1}\left\{\sum_{\beta=-k+1}^{n-1} c_{\alpha,\beta}N_{\beta,k}^{(\mu)}(y)\right\}\cdot\int_{x_0}^{x}N_{\alpha,k}(x)dx$$
$$=\sum_{\alpha=-k+1}^{m-1}c_{\alpha}\int_{x_0}^{x}N_{\alpha,k}(x)dx$$
$$\text{，ただし}\quad c_{\alpha}=\sum_{\beta=-k+1}^{n-1}c_{\alpha,\beta}N_{\beta,k}^{(\mu)}(y) \quad (7.31)$$

となるが，これは，先に挙げた (7.23) に類似する．したがって (7.31) を計算するには，まず c_{α} を計算し，続いて，1 変数の積分の手法を使って (7.31) を計算すればよい．

$S^{(-1,\mu)}(x,y)$ のほか，

$$S^{(\lambda,-1)}(x,y)=\int_{y_0}^{y}\frac{\partial^{\lambda}S(x,y)}{\partial\ x^{\lambda}}dy$$
$$S^{(-1,-1)}(x,y)=\int_{y_0}^{y}dy\int_{x_0}^{x}S(x,y)dx$$

の計算も，1 変数の場合の微分，積分の手法を，x, y 別々に適用することにより処理できる．

7.2　補間

補間を行う場合の一般的手順は，与えられた標本点，(x_i,y_i) を通るある近似関数－多項式，区分的多項式，その他－を作り，それを計算することから成る．

近似関数が多項式の場合，それはラグランジュ補間多項式，あるいはエルミート補間多項式（関数値のほかに微係数の情報も使う）と呼ばれている．SSL II のエイトケン・ラグランジュ補間及びエイトケン・エルミート補間は，これに類する手法であるが，特徴として反復的に補間多項式の次数を上げていくことにより，最も適当と思われる補間値を見出そうとするものである．

一方，近似関数が区分的多項式の場合は，単なる多項式を用いたのでは補間がうまくいかないときに用いることができる．SSL II はこれに属するものとして，準エルミート補間，spline 関数による補間を用意している．

spline 関数による補間式は，7.1 の「スプライン関数について」で述べた性質に，補間の条件－与えられた標本点を通ること－を加えた spline 関数のことであり，実はそれは，何らかの付加的条件を加えることによりいろいろな補間式を作り出すことができる．SSL II では四つのタイプの spline 補間式を考え各々サブルーチンを用意している．

spline 関数の表現形式は 7.1 の「スプライン関数について」で述べた B-spline の安定性を重要視し，もっぱらそれによる表現を用いている．

B-spline による補間

B-spline を用いる補間のサブルーチンは目的により，次の二つに分けられる．

(a)補間値（あるいは微分値，積分値）を求めるサブルーチン

(b)補間式を求めるサブルーチン

(a) はすべて，(b) で求まった補間式を利用するので，利用者は手順として，(b) を先に呼び出し，続いて (a) を呼び出すことになる．

B-spline を用いる補間式として，SSL II ではいくつか用意する．すなわち，離散点を x_i, i = 1, 2, ..., n として，m (=2l−1, l≥2) 次の B-spline 補間式 $S(x)$ を求める場合，$S(x)$の境界条件の有無，あるいは境界

条件の内容によって，次に示す四つの“タイプ”に分類し，各々サブルーチンを用意する．

タイプ I .....$S^{(j)}(x_1)$, $S^{(j)}(x_n)$, j = 1, 2, ⋯, l – 1 を（利用者の）指定した値にとる．

タイプ II .....$S^{(j)}(x_1)$, $S^{(j)}(x_n)$, $j = l, l+1, \cdots, 2l-2$ を指定した値にとる．

タイプ III .....境界条件なし．

タイプ IV .....$S^{(j)}(x_1) = S^{(j)}(x_n)$, j = 0, 1,⋯, 2l–2 を満たす．このタイプは周期関数の補間に適している．

これら四つのタイプの使分けは，利用者の持っている元の関数についての情報量による．

通常は，タイプ III のサブルーチンを利用すればよい．

2 次元補間の場合は，(7.28) で表現される 2 変数 spline 関数 $\boldsymbol{S}(x,y)$ を補間式として用いる．このとき，x 方向，y 方向に対して独立にタイプを考えることもできるが，SSL II では両方向に対してタイプ I 又はタイプ III を使った補間式を用意する．

B-spline 補間式により補間値を求める際，その補間式の次数 m の選び方が問題となる．次数は 3 または 5 程度がよい．倍精度では，元の関数がおとなしく，関数値が高精度で与えられているならばもう少し次数を高くしてもよい．しかし 15 を超えるとよくないといわれている．

表 7.1 に補間のサブルーチンを示す．

準エルミート補間

これは，spline 補間と同じように，区分的な多項式による補間である．ただ，準エルミート補間式は，高階導関数の連続性を spline 補間式ほどには強く要求しない点が異なっている．

準エルミート補間式の唯一の特徴は，離散点がどのような分布で与えられても離散点の間に不自然な屈曲点が現れないということであり，このため「曲線又は曲面のあてはめ」に適している．曲線（又は曲面）のあてはめでは，どんな離散点に対しても，あたかも熟練した製図工が手で描くような曲線を作り出すことに重点が置かれる．準エルミート補間式は，まさにこの目的のための補間式である．

しかし，高精度な補間値，又は微分値，積分値を得るためには，むしろ，B-spline による補間を用いた方がよい．

表 7.1　補間のサブルーチン

目的	サブルーチン名	手法	備考
補間値	AKLAG (E11-11-0101)	エイトケン・ラグランジュ補間	微係数不要
	AKHER (E11-11-0201)	エイトケン・エルミート補間	微係数要
	SPLV (E11-21-0101)	3 次 spline 補間	
	BIF1 (E11-31-0101)	B-spline 補間式 (I) の利用	タイプ I
	BIF2 (E11-31-0201	B-spline 補間式 (II) の利用	タイプ II
	BIF3 (E11-31-0301	B-spline 補間式 (III) の利用	タイプ III
	BIF4 (E11-31-0401	B-spline 補間式 (IV) の利用	タイプ IV
	BIFD1 (E11-32-1101)	B-spline 2 次元補間式 (I-I) の利用	タイプ I-I
	BIFD3 (E11-32-3301)	B-spline 2 次元補間式 (III-III) の利用	タイプ III-III
	AKMID (E11-42-0101)	2 次元準エルミート補間	
補間式	INSPL (E12-21-0101)	3 次 spline 補間式	両端 (x_1, x_n) における，2 階微係数要
	AKMIN (E12-21-0201)	準エルミート補間式	
	BIC1 (E12-31-0102)	B-spline 補間式 (I)	タイプ I
	BIC2 (E12-31-0202)	B-spline 補間式 (II)	タイプ II
	BIC3 (E12-31-0302)	B-spline 補間式 (III)	タイプ III
	BIC4 (E12-31-0402)	B-spline 補間式 (IV)	タイプ IV
	BICD1 (E11-32-1102)	B-spline 2 次元補間式 (I-I)	タイプ I-I
	BICD3 (E12-32-3302)	B-spline 2 次元補間式 (III-III)	タイプ III-III

7.3　近似

ここでは，最小二乗近似多項式を扱う（表 7.2）．B-spline による最小二乗近似は，「平滑化」の分野で扱われている．

7.4　平滑化

平滑化に関して用意されているサブルーチンを表 7.3 に示す．SMLE1 ，SMLE2 は平滑化を求めるのに，観測値の全体にわたるただ一つの最小二乗近似多項式をあてはめて行う代わり，局部的に，それぞれ異なる最小二乗近似多項式をあてはめることにより行うものである．

表 7.2　近似のサブルーチン

目的	サブルーチン名	手法	備考
最小二乗近似多項式	LESQ1 (E21-20-0101)	離散点に関して直交する多項式を導入する方法	近似式の次数はサブルーチン内で決定する.

表 7.3　平滑化のサブルーチン

目的	サブルーチン名	手法	備考
平滑値	SMLE1 (E31-11-0101)	局部的最小二乗近似多項式の適用	等間隔離散点
	SMLE2 (E31-21-0101)	局部的最小二乗近似多項式の適用	不等間隔離散点
	BSF1 (E31-31-0101)	B-spline 平滑化式の利用	不等間隔離散点
	BSFD1 (E31-32-0101)	B-spline 2 次元平滑化式の利用	不等間隔格子点
平滑化式	BSC1 (E32-31-0102)	B-spline 平滑化式 (固定節点)	不等間隔離散点
	BSC2 (E32-31-0202)	B-spline 平滑化式(節点追加方式)	
	BSCD2 (E32-32-0202)	B-spline 2 次元平滑化式 (節点追加方式)	不等間隔格子点

しかし，一般目的のためには，B-spline によるサブルーチンを利用したほうがよい．B-spline による平滑化は，1 次元の場合は，(7.14)，2 次元の場合は (7.28) で表現される spline 関数を平滑化式として用いる．その際，係数 c_j 若しくは $c_{\alpha,\beta}$ は最小二乗法により決定される．平滑値は，求まった平滑化式を評価することにより得られ，そのためのサブルーチンも用意されている．

ところで，B-spline 平滑化式を求めるサブルーチンには，節点列の決め方の違いにより二つのタイプがある．一つは，節点列を利用者が与えるタイプ（固定節点）であり，他のひとつは，節点列をサブルーチンが適応的に決定するタイプ（節点追加方式）である．前者のサブルーチンは，節点列の与え方についての経験を要する．したがって，通常は，後者のサブルーチンを使えばよい．

7.5　級数

関数のチェビシェフ級数展開，級数の求和，導関数，不定積分のために，表 7.4 に示すサブルーチンが用意されている．

表 7.4　チェビシェフ級数のサブルーチン

目的	サブルーチン名	手法	備考
級数展開	FCHEB (E51-30-0101)	高速 cosine 変換	項数は (2 のべき) + 1
級数の求和	ECHEB (E51-30-0201)	後退漸化式	
級数の導関数	GCHEB (E51-30-0301)	チェビシェフ多項式の微分公式	
級数の不定積分	ICHEB (E51-30-0401)	チェビシェフ多項式の積分公式	

一方，周期関数に対しては，cosine 級数展開，sine 級数展開と，それぞれの級数の求和を求めるために，表 7.5 に示すサブルーチンが用意されている．

表 7.5　cosine , sine 級数のサブルーチン

目的	サブルーチン名	手法	備考
Cosine 級数展開	FCOSF (E51-10-0101)	高速 cosine 変換	偶関数
Cosine 級数の求和	ECOSP (E51-10-0201)	後退漸化式	偶関数
sine 級数展開	FSINF (E51-20-0101)	高速 sine 変換	奇関数
sine 級数の求和	ESINP (E51-20-0201)	後退漸化式	奇関数

第 8 章
変　　換

8.1　概要

この章では，離散型フーリエ変換及びラプラス変換を取り上げる．

離散型フーリエ変換には，変換の対象とするデータの性質に応じてサブルーチンが用意されている．

データの性質とは，次の内容を指す．

- 実データか複素データか．
- 実データでも偶関数か奇関数か．

8.2　離散型実フーリエ変換

データが実データである場合に，(8.1) なる変換又は，(8.2) なる逆変換に相当した変換を行うサブルーチンが用意されている．

$$\left.\begin{aligned} a_k &= \frac{2}{n}\sum_{j=0}^{n-1} x_j \cos\frac{2\pi kj}{n}, \quad k = 0,1,\ldots,\frac{n}{2} \\ b_k &= \frac{2}{n}\sum_{j=0}^{n-1} x_j \sin\frac{2\pi kj}{n}, \quad k = 1,2,\ldots,\frac{n}{2}-1 \end{aligned}\right\} \tag{8.1}$$

$$\begin{aligned} x_j = {} & \frac{1}{2}a_0 + \sum_{k=1}^{n/2-1}\left(a_k \cos\frac{2\pi kj}{n} + b_k \sin\frac{2\pi kj}{n}\right) \\ & + \frac{1}{2}a_{n/2}\cos\pi\, j, \quad j = 0,1,\ldots,n-1 \end{aligned} \tag{8.2}$$

a_k , b_k を離散型フーリエ係数という．

周期 2πの実数値関数 $x(t)$ のフーリエ係数を定義する積分

$$\left.\begin{aligned} &\frac{1}{\pi}\int_0^{2\pi} x(t)\cos kt\,dt \\ &\frac{1}{\pi}\int_0^{2\pi} x(t)\sin kt\,dt \end{aligned}\right\} \tag{8.3}$$

に対し，閉区間 $[0, 2\pi]$ を n 等分し $x(t)$ を $x_j = x\left(\frac{2\pi}{n}j\right), j = 0,1,\ldots,n-1$ と標本化し台形公式を適用したものが変換 (8.1) である．特に $x(t)$ が $n/2 - 1$ 次の三角多項式ならば (8.1) は，積分 (8.3) の正確な数値積分公式となる．すなわち，離散型フーリエ係数と，本来のフリーエ係数とは一致する．

更に $x(t)$ が偶関数あるいは奇関数ならば，それらの性質を利用した離散型 cosine 変換，sine 変換が用意されている．

8.3　離散型 cosine 変換

偶関数 $x(t)$ に対して 2 種類の変換を行うサブルーチンが用意されている．一方は，閉区間 $[0,\pi]$ の端点を標本点に含む方法であり，他方は含まない．

① 離散型 cosine 変換 （台形公式）

偶関数 $x(t)$ を閉区間 $[0,\pi]$ 上で，$x_j = \left(\frac{\pi}{n}j\right)$, j=0, 1, ..., n と標本化し，(8.4) なる変換又は，(8.5) なる逆変換に相当した変換を行う．

$$a_k = \frac{2}{n}\sum_{k=0}^{n}{}'' x_j \cos\frac{\pi}{n}kj, \quad k = 0,1,\ldots,n \tag{8.4}$$

$$x_j = \sum_{k=0}^{n-1}{}'' a_k \cos\frac{\pi}{n}kj, \quad j = 0,1,\ldots,n \tag{8.5}$$

ここで Σ″ は，初項と末項を 1/2 倍して和をとることを意味する．

偶関数 $x(t)$ のフーリエ係数を定義する積分

$$\frac{2}{\pi}\int_0^{\pi} x(t)\cos\, kt\, dt \tag{8.6}$$

に対し，閉区間 [0,π] を n 等分して $x(t)$ を $x_j = x\left(\frac{\pi}{n} j\right), j = 0,1,\ldots,n$ と標本化し，台形公式を適用したものが変換 (8.4) である．

② 離散型 cosine 変換 （中点公式）
偶関数 $x(t)$ を開区間 $(0,\pi)$ 上で $x_{j+1/2} = x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right), \quad j = 0,1,\ldots,n-1$ と標本化し，(8.7) なる変換又は，(8.8) なる逆変換に相当した変換を行う．

$$a_k = \frac{2}{n}\sum_{j=0}^{n-1} x_{j+\frac{1}{2}} \cos\frac{\pi}{n}k\left(j+\frac{1}{2}\right), \quad k = 0,1,\ldots,n-1 \tag{8.7}$$

$$x_{j+\frac{1}{2}} = \sum_{k=0}^{n-1}{}' a_k \cos\frac{\pi}{n}k\left(j+\frac{1}{2}\right), \quad j = 0,1,\ldots,n-1 \tag{8.8}$$

ここで Σ' は，初項だけを 1/2 倍して和をとることを意味する．
積分 (8.6) に n 項の中点公式を適用したものが，変換 (8.7) である．

8.4　離散型 sine 変換

関数 $x(t)$ が奇関数の場合，2 種類の変換を行うサブルーチンが用意されている．離散型 cosine 変換の場合と同様に，一方は台形公式に基づく変換であり，他方は中点公式による方法である．

① 離散型 sine 変換 （台形公式）
奇関数 $x(t)$ を閉区間 [0,π] で $x_j = x\left(\frac{\pi}{n} j\right)$, j=1, 2, ..., n-1 と標本化し (8.9) なる変換又は，(8.10) なる逆変換に相当した変換を行う．

$$b_k = \frac{2}{n}\sum_{j=1}^{n-1} x_j \sin\frac{\pi}{n}k\, j, \quad k = 1,2,\ldots,n\text{-}1 \tag{8.9}$$

$$x_j = \sum_{k=1}^{n-1} b_k \sin\frac{\pi}{n}k\, j, \quad j = 1,2,\ldots,n-1 \tag{8.10}$$

奇関数 $x(t)$ のフーリエ係数を定義する積分

$$\frac{2}{\pi}\int_0^{\pi} x(t)\sin kt\,dt \tag{8.11}$$

に対し，閉区間 [0,π] を n 等分して $x(t)$ を $x_j = x\left(\frac{\pi}{n} j\right)$, j=1, 2, ..., n-1 と標本化し，台形公式を適用したものが変換 (8.9) である．

② 離散型 sine 変換（中点公式）
奇関数 $x(t)$ を開区間 $(0,\pi)$ 上で $x_{j+1/2} = x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right), \quad j = 0,1,\ldots,n-1$ と標本化し，(8.12) なる変換又は，(8.13) なる逆変換に相当した変換を行う．

$$b_k = \frac{2}{n}\sum_{j=0}^{n-1} x_{j+\frac{1}{2}} \sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right), \quad k = 1,2,\ldots,n \tag{8.12}$$

$$x_{j+\frac{1}{2}} = \sum_{k=1}^{n-1} b_k \sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right) + \frac{1}{2}b_n \sin\pi\left(j+\frac{1}{2}\right) \quad j = 0,1,\ldots,n-1 \tag{8.13}$$

積分 (8.11) に n 項の中点公式を適用したものが，変換 (8.12) である．

8.5　離散型複素フーリエ変換

データが複素データある場合には，(8.14) なる変換又は (8.15) なる逆変換に相当した変換を行うサブルーチンが用意されている．

$$\alpha_k = \frac{1}{n}\sum_{j=0}^{n-1} x_j \exp\left(-2\pi i\frac{jk}{n}\right), \quad k = \ 0,1,\ldots,n-1 \tag{8.14}$$

$$x_j = \sum_{k=0}^{n-1} \alpha_k \exp\left(2\pi i\frac{jk}{n}\right), \quad j = 0,1,\ldots,n-1 \tag{8.15}$$

周期 2π の複素数値関数 $x(t)$ のフーリエ係数を定義する積分

$$\frac{1}{2\pi}\int_0^{2\pi} x(t)\exp(-ikt)\,dt \tag{8.16}$$

に対し，閉区間 [0,2π] を n 等分し $x(t)$ を $x_j = x\left(\frac{2\pi}{n} j\right), j = 0,1,\ldots,n,$ と標本化し，$x_0 = x_n$ を考慮して台形公式を適用したものが，変換 (8.14) である．

これらの離散型フーリエ変換は，高速変換手法 (FFT) により行うので処理速度が速い．
ところで，高速変換手法で変換を行う場合，内部での主な処理は次の二つに分けられる．
(a)データが与えられた領域上で，部分的な小さな項数の変換を繰り返すことにより変換を達成する．
(b)データを正しい順に並べかえる．

この各々の処理を行うサブルーチン（コンポーネントルーチン）が用意されている．

表 8.1　離散型フーリエ変換のサブルーチン名

<table>
<tr><th colspan="2" rowspan="2">変換の種類</th><th rowspan="2">データ数</th><th colspan="3">サブルーチン名</th></tr>
<tr><th>標準ルーチン</th><th>コンポーネントルーチン
(a)</th><th>(b)</th></tr>
<tr><td rowspan="2">cosine 変換</td><td>台形公式</td><td>2 の巾 + 1</td><td>FCOST
(F11-11-0101)</td><td></td><td></td></tr>
<tr><td>中点公式</td><td rowspan="5">2 の巾</td><td>FCOSM
(F11-11-0201)</td><td></td><td></td></tr>
<tr><td rowspan="2">sine 変換</td><td>台形公式</td><td>FSINT
(F11-21-0101)</td><td></td><td></td></tr>
<tr><td>中点公式</td><td>FSINM
(F11-21-0201)</td><td></td><td></td></tr>
<tr><td colspan="2">実変換</td><td>RFT
(F11-31-0101)</td><td></td><td></td></tr>
<tr><td colspan="2" rowspan="2">複素変換</td><td rowspan="2">CFT
(F12-15-0101)</td><td>CFTN
(F12-15-0202)</td><td rowspan="2">PNR
(F12-15-0402)</td></tr>
<tr><td>CFTR
(F12-15-0302)</td></tr>
<tr><td colspan="2"></td><td>素数の巾の積</td><td>CFTM
(F12-11-0101)</td><td></td><td></td></tr>
</table>

[注意]
(a) と (b) は，8.5 節 特徴で説明した内容である．

フーリエ変換は，(a) 及び (b) のサブルーチンを組み合せて用いれば行えるが，それを標準的に使う場合を考慮して両者を結合したサブルーチン（標準ルーチン）も用意されている．一般には後者を用いればよい．

データ数は，処理速度を考慮して 2 の巾で表せる数を原則としている．しかし，複素フーリエ変換については更に，次の点が考慮されている．

・データ数は，2 の巾又は素数の巾の積のいずれも可能である．
・多次元変換も可能である．

表 8.1 に，データの性質に応じたサブルーチン名を示す．

使用上の注意

a. 標本数（項数）
変換の対象となるデータの標本数 n は，関数 $x(t)$ の性質（データの性質）に応じて，その定義される意味が異なるので注意すること．
すなわち，n は，
- cosine , sine 変換の場合，半周期分($[0,\pi]$ 又は $(0,\pi)$) の標本に相当する．
- 実変換，複素変換の場合，1 周期分 ($[0,2\pi]$) の標本に相当する．

b. 実変換と cosine , sine 変換との使い分け
関数 $x(t)$ が一般の実数関数の場合は，実変換のサブルーチンが利用できるが，$x(t)$ に対してあらかじめ偶関数か奇関数かの判別がつく場合には，cosine , sine 変換のサブルーチンを利用することが望ましい．（処理速度は，実変換による場合の約半分である．）
なお，$x(t)$ が実数値関数でも，$x(t) + x(-t)$ 及び $x(t) - x(-t)$ は，それぞれ偶関数，奇関数となるので，cosine 変換，sine 変換に分解して計算することができる．

c. 実変換と複素変換によるフーリエ係数の関連
実変換 (cosine, sine 変換を含む) によるフーリエ係数 $\{a_k\}$, $\{b_k\}$ と複素変換によるフーリエ係数，$\{\alpha_k\}$ との間には，次の関係式が成り立つ．

$$\left.\begin{aligned} a_0 &= 2\alpha_0 \quad , \quad a_{n/2} = 2\alpha_{n/2} \\ a_k &= (\alpha_k + \alpha_{n-k}) \quad , \quad k = 1,2,\ldots,n/2-1 \\ b_k &= i(\alpha_k - \alpha_{n-k}) \quad , \quad k = 1,2,\ldots n/2-1 \end{aligned}\right\} \quad (8.17)$$

ここで n は，1 周期 $[0,2\pi]$ の等分数を表す．

この関係に基づき，利用者は必要に応じて，実変換と複素変換のサブルーチンを併用することができる，ただし，正規化及びデータの並びについては注意すること．

d.　三角関数表について
cosine, sine 変換の場合，処理効率を高めるために，変換で必要な三角関数表をサブルーチン内部で作成している．関数表はパラメタ TAB に出力され，繰り返し変換を行う場合の効率を高めるためにも，有効に利用できる．各々の変換に対して，台形公式と中点公式に基づく二つのサブルーチンが用意されているが，三角関数表の大きさは，前者の方が小さいので若干能率が良い．

e.　正規化について
変換を行うすべてのサブルーチンは，変換結果に対する正規化は行っていない．
利用者は，必要に応じて任意に正規化することができる．

8.6　ラプラス変換

関数 $f(t)$ のラプラス変換と逆変換は次式で定義される．

$$F(s)=\int_0^{\infty} f(t)e^{-st}dt \tag{8.18}$$

$$f(t)=\frac{1}{2\pi i}\int_{\gamma-i\infty}^{\gamma+i\infty} F(s)e^{st}ds \tag{8.19}$$

ただし $\gamma>\gamma_0, \gamma_0$：収束座標，$i=\sqrt{-1}$．

$f(t)$ をラプラス変換における原関数，$F(s)$ を像関数という．像関数 $F(s)$ に次の条件を仮定する．

$$\left.\begin{array}{l}\text{(i)}\ \mathrm{Re}(s)>\gamma_0 \text{ で} F(s) \text{は正則．}\\ \text{(ii)}\ \mathrm{Re}(s)>\gamma_0 \text{ で} \lim_{|s|\to\infty} F(s)=0\\ \text{(iii)}\ \mathrm{Re}(s)>\gamma_0 \text{ で} F^*(s)=F(s^*)\end{array}\right\} \tag{8.20}$$

Re [•] は [•] の実数部，$F^*(s)$ は $F(s)$ の共役複素数を表す．条件 (i) は常に成り立ち，条件 (ii) は $f(t)$ が超関数の場合を除き満足される．条件 (iii) は $f(t)$ が実関数ということである．用意されているサブルーチンは式 (8.19) の逆変換を数値的に行うものである．以下，その手法の概要を示す．

①　変換の計算式
まず簡単のため，$\gamma_0 \le 0$ の場合を考える．すなわち，$F(s)$ は $\mathrm{Re}(s)>0$ で正則でありかつ (8.19) の積分は，$\gamma>0$ なる任意の実数 γ に対して存在するものとする．

$$e^s=\lim_{\sigma_0\to\infty} e^{\sigma_0}/\left[2\cosh(\sigma_0-s)\right]$$

の関係に着目して (8.19) の e^{st} を適当な σ_0 を選ぶことにより，

$$E_{ec}(st,\sigma_0)\equiv e^{\sigma_0}/\left[2\cosh(\sigma_0-st)\right]$$

で近似する．関数 $E_{ec}(st,\sigma_0)$ の特徴は，$\mathrm{Re}(s)=\sigma_0/t$ の直線上に無数の極を持つことで，極の配置を図 8.1 に×印で示す．また，このことを式で陽に書くと

$$E_{ec}(st,\sigma_0)=\frac{e^{\sigma_0}}{2t}\sum_{n=-\infty}^{\infty}\frac{(-1)^n i}{s-\left[\sigma_0+i(n-0.5)\pi\right]/t}$$

となる．このとき，原関数 $f(t)$ の σ_0 近似 $f(t,\sigma_0)$ は

$$f(t,\sigma_0)\equiv\frac{1}{2\pi i}\int_{r-i\infty}^{r+i\infty}F(s)E_{ec}(st,\sigma_0)ds \tag{8.21}$$

となる．ただし γ は $\gamma_0<\gamma<\sigma_0/t$ を満たすものとする．右辺の積分は $E_{ec}(st,\sigma_0)$ の極のまわりの積分に変換できることが証明できる．

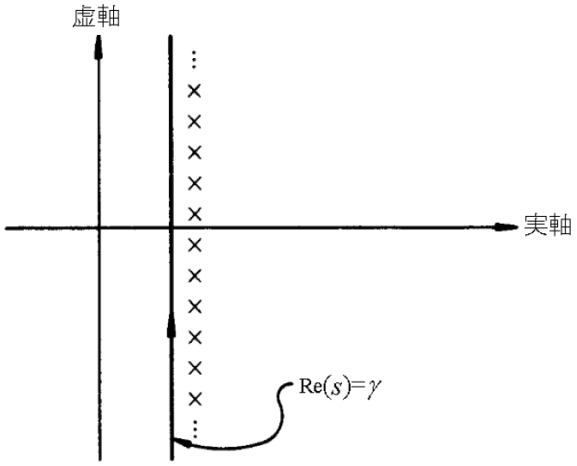


図 8.1　$E_{ec}(st,\sigma_0)$ の極

$F(s)$ は $\mathrm{Re}(s)>0$ で正則であるからコーシーの積分公式より

$$f(t,\sigma_0)=\frac{e^{\sigma_0}}{2t}\sum_{n=-\infty}^{n}(-1)^{n+1}iF\left(\frac{\sigma_0+i(n-0.5)\pi}{t}\right)$$
$$=\frac{e^{\sigma_0}}{t}\sum_{n=1}^{\infty}(-1)^n\,\mathrm{Im}\left[F\left(\frac{\sigma_0+i(n-0.5)\pi}{t}\right)\right] \tag{8.22}$$

となる．Im [•] は [•] の虚数部を表す．ただし (8.21) の条件 $\gamma_0<\gamma<\sigma_0/t$ において，もし $\gamma_0>0$ であると，ある $t(0<t<\infty)$ でこの条件が満足できなくなる．したがって (8.22) を $0<t<\infty$ に対して適用するときは $\gamma_0\le 0$ の条件が必要である．
$f(t,\sigma_0)$ は $f(t)$ の近似値を与え，文献 [98] の誤差解析より

$$f(t,\sigma_0)=f(t)-e^{-2\sigma_0}\cdot f(3t)+e^{-4\sigma_0}\cdot f(5t)-\cdots\cdots \tag{8.23}$$

となる．したがって，$\sigma_0 \gg 1$ のとき $f(t,\sigma_0)$ は $f(t)$ の良い近似であることを示し，また，(8.23) は近似誤差の評価式として使える．
数値計算を行う場合，原理的には (8.22) を適当な項数 N で打ち切ればよいが，単純な打切りでは実用的でない．したがって，個々の場合に応じて実用的な計算式を工夫する必要がある．用意されているサブルーチンでは，かなり一般性のある Euler 変換を利用している．

$$F_n \equiv (-1)^n \mathrm{Im}\left[F\left(\frac{\sigma_0+i(n-0.5)\pi}{t}\right)\right] \tag{8.24}$$

と定義すると，Euler 変換は次の条件が成り立つ場合に有効である（文献 [100]）．

(i)　適当な整数 $k\geq 1$ が存在し，$n\geq k$ のとき F_n の符号が交代する．
(ii)　$n \geq k$ のとき $1/2 \leq |F_{n+1}/F_n| < 1$ (8.25)

F_n がこの条件を満足しているとき，(8.22) の級数を次ぎのように書き換える．

$$\sum_{n=1}^{\infty} F_n = \sum_{n=1}^{k-1} F_n + \sum_{q=0}^{p} \frac{1}{2^{q+1}} D^q F_k + R_{p+1}(k) \tag{8.26}$$

ここで $R_p(k)$ は
$R_p(k) \equiv 2^{-P}(D^p F_k + D^p F_{k+1} + D^p F_{k+2} + \cdots)$
で与えられる．$D^p F_k$ は次式で定義される第 p 階差である．

$$D^0 F_k = F_k, \quad D^{p+1} F_k = D^p F_k + D^p F_{k+1} \tag{8.27}$$

サブルーチンでは

$$f_N(t,\sigma_0)=\frac{e^{\sigma_0}}{t}\sum_{n=1}^{N} F_n = \frac{e^{\sigma_0}}{t}\left\{\sum_{n=1}^{k-1} F_n + \sum_{q=0}^{p}\frac{D^q F_k}{2^{q+1}}\right\} \tag{8.28}$$

を用いている．$N = k + p$,

$$\left.\begin{aligned} &\sum_{q=0}^{p}\frac{D^q F_k}{2^{q+1}} = \frac{1}{2^{p+1}}\sum_{r=0}^{p} A_{p,r} F_{k+r} \\ &A_{p,p}=1, A_{p,r-1} = A_{p,r} + \binom{p+1}{r} \end{aligned}\right\} \tag{8.29}$$

σ_0, k, p の決め方は各サブルーチンの箇所で述べる．

$f_N(t,\sigma_0)$ の打切り誤差については，次のことが証明されている．$\phi(n) \equiv F_n$ とすると，もし，$\phi(x)$の p 階微分係数 $\phi^{(p)}(x)$ が x の正の値に対して定符号であり，かつ x の増加と共に単調に減少するならば，（例えば $F(s)$ が有理関数の場合），

$$\left.\begin{aligned} &\left|f(t,\sigma_0)-f_N(t,\sigma_0)\right| = \frac{e^{\sigma_0}}{t}\left|R_{p+1}(k)\right| \\ &\leq \left|f_{N+1}(t,\sigma_0)-f_N(t,\sigma_0)\right| = \frac{e^{\sigma_0}}{t}\left|\frac{1}{2^{p+1}} D^{p+1} F_k\right| \end{aligned}\right. \tag{8.30}$$

が成り立つ．ここで $f_{N+1}(t,\sigma_0)$ は，(8.28) において k を 1 つ増やしたものである．上式において $D^{p+1}F_k$ を計算するには，$f_N(t,\sigma_0)$ の計算に使われる列 $\{F_n;\ n = k, k+1, \ldots, k+p\}$ に加えて F_{k+p+1} を必要とし，関数計算を 1 回余分に行うことになる．これを避けるため，サブルーチンでは，$f_N(t,\sigma_0)$ の打切り誤差を

$$\left|f_N(t,\sigma_0)-f_{N-1}(t,\sigma_0)\right| = \frac{e^{\sigma_0}}{t}\left|\frac{1}{2^{p+1}} D^{p+1} F_{k-1}\right|$$

で代用している．また，サブルーチンでは，打切り誤差を，相対誤差の形，

$$\left|\frac{f_N(t,\sigma_0)-f_{N-1}(t,\sigma_0)}{f_N(t,\sigma_0)}\right| = \left|\frac{\frac{1}{2^{p+1}} D^{p+1} F_{k-1}}{\sum_{n=1}^{k-1} F_n + \frac{1}{2^{p+1}}\sum_{r=0}^{p} A_{p,r} F_{k+r}}\right|$$

で出力する．
$D^{p+1}F_{k-1}$ は F_{k-1}, F_k, ..., F_{k+p} の 1 次結合となるが，その係数は 2 項展開係数に等しく，また，
$A_{p,\ r}$ は (8.29) より分かるように 2 項展開係数の累和として計算できる．したがって，これらの係数は，いずれもパスカルの三角形から容易に求めることができる．例として，p = 4 の場合を図 8.2 に示す．

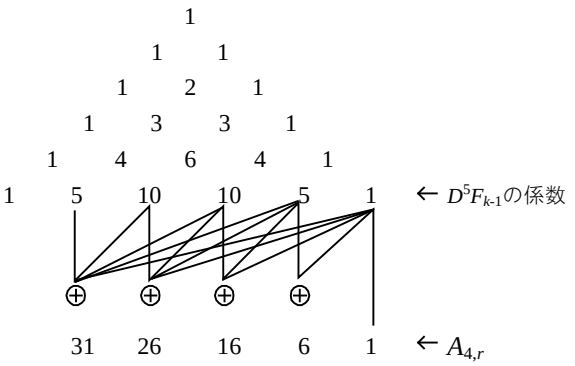


図　8.2　p=4 の場合のパスカルの三角形

次に $\gamma_0>0$ の場合を考える．このとき $F(s)$ は $\mathrm{Re}(s) > 0$ で正則とはならないので $F(s)$ に対して上記の手法を直接適用することはできない．しかし (8.19) の積分は，

$$
\begin{aligned}
f(t) &= \frac{1}{2\pi i}\int_{r-i\infty}^{r+i\infty} F(s+\gamma_0)e^{(s+\gamma_0)t}ds \\
&= \frac{e^{r_0 t}}{2\pi i}\int_{r-i\infty}^{r+i\infty} G(s)\,e^{st}ds \\
&= e^{r_0 t}g(t)
\end{aligned}
$$

ここで $r>0,\ G(s)=F(s+r_0)$

$$
g(t) = \frac{1}{2\pi i}\int_{r-i\infty}^{r+i\infty} G(s)e^{st}ds \qquad (8.31)
$$

と書けることに注目する． $G(s)$ は $\mathrm{Re}(s) > 0$ で正則となるので，$g(t)$ を上記の手法で計算することにすれば $f(t)$ は $g(t)$ に $e^{\gamma_0 t}$ を乗ずることにより求めることができる．

② 有理関数の変換

$F(s)$ が有理関数の場合，実係数多項式 $Q(s)$ ，$P(s)$ を用いて

$$
F(s) = Q(s) / P(s) \qquad (8.32)
$$

と表せる．収束座標 γ_0 が $\gamma_0\leq 0$ であるか $\gamma_0>0$ であるかの判定は，$P(s)$ が Hurwitz 多項式（すべての零点が複素平面の左半面，すなわち $\mathrm{Re}(s)<0$ の領域に存在する多項式）かどうかを判定すればよい．サブルーチンで使用した判定法を以下に示す（文献 [94] ）．

n 次の実係数多項式 $P(s)$ を

$$
\begin{aligned}
P(s) &= a_1 s^n + a_2 s^{n-1} + \ldots + a_n s + a_{n+1} \\
&= m(s) + n(s) \\
&\text{ただし} m(s) = a_1 s^n + a_3 s^{n-2} + \ldots, a_1 \neq 0 \\
&\qquad n(s) = a_2 s^{n-1} + a_4 s^{n-3} + \ldots
\end{aligned}
$$

と分解する．$m(s)$と $n(s)$ の比を

$$
W(s) \equiv m(s)/n(s)
$$

と定義し，$W(s)$ を連分数展開する．

$$
W(s) = h_1 s + \cfrac{1}{\cfrac{1}{h_2 s} + \cfrac{1}{\cfrac{1}{h_3 s} + \cfrac{1}{\cfrac{1}{h_4 s} + \ldots}}} \qquad (8.33)
$$

$h_1, h_2, \cdots$がすべて正であれば $P(s)$ は Hurwitz 多項式である．

$F(s)$ が $\mathrm{Re}(s) >0$ に特異点を持つ場合，上記の判定法を繰り返し使うことにより $G(s)=F(s+\alpha)$ が $\mathrm{Re}(s) >0$ で正則となるように$\alpha(> 0)$ を増加させる．この $G(s)$ に対して $g_N(t,\sigma_0)$ を求め，それに $e^{\alpha t}$ を乗ずれば $f_N(t,\sigma_0)$ が得られる．

$F(s)$ が無理関数や超関数の場合，$\mathrm{Re}(s) > 0$ で $F(s)$ が正則かどうかを判定する有効な方法はない．したがって $F(s)$ が一般関数の場合は，収束座標 γ_0 の決定は利用者に任ねられる．

③ サブルーチンの使分け

ラプラス逆変換のためのサブルーチンを表 8.2 に示す．LAPS1 は $\gamma_0 \leq 0$ であることが既知な場合に，LAPS2 は $\gamma_0 > 0$ であることが既知かそれとも γ_0 については不明な場合に，それぞれ有理関数の変換を行うものである．一方，HRWIZ は$\gamma_0 \leq 0$ であるか否かの判定，すなわち，(8.32) の $P(s)$ が Hurwitz 多項式であるか否かの判定を行い，かつ $\gamma_0 > 0$ であったときは γ_0 の近似値を計算する．

$\gamma_0 > 0$ であることは，原関数 $f(t)$ が $t\to\infty$ のとき指数関数的に増大することを意味する．そのような振舞を調べる目的でも HRWIZ を使用することができる．

以上の使分けを流れ図に示すと図 8.3 のようになる．

表 8.2　ラプラス変換のサブルーチン

関数の型	サブルーチン名	備考
有理関数	LAPS1 (F20-01-0101)	複素右半平面で正則な有理関数
	LAPS2 (F20-02-0101)	一般の有理関数
	HAWIZ (F20-02-0201)	Hurwitz 多項式の判定
一般関数	LAPS3 (F20-03-0101)	収束座標 γ_0 の入力が必要.

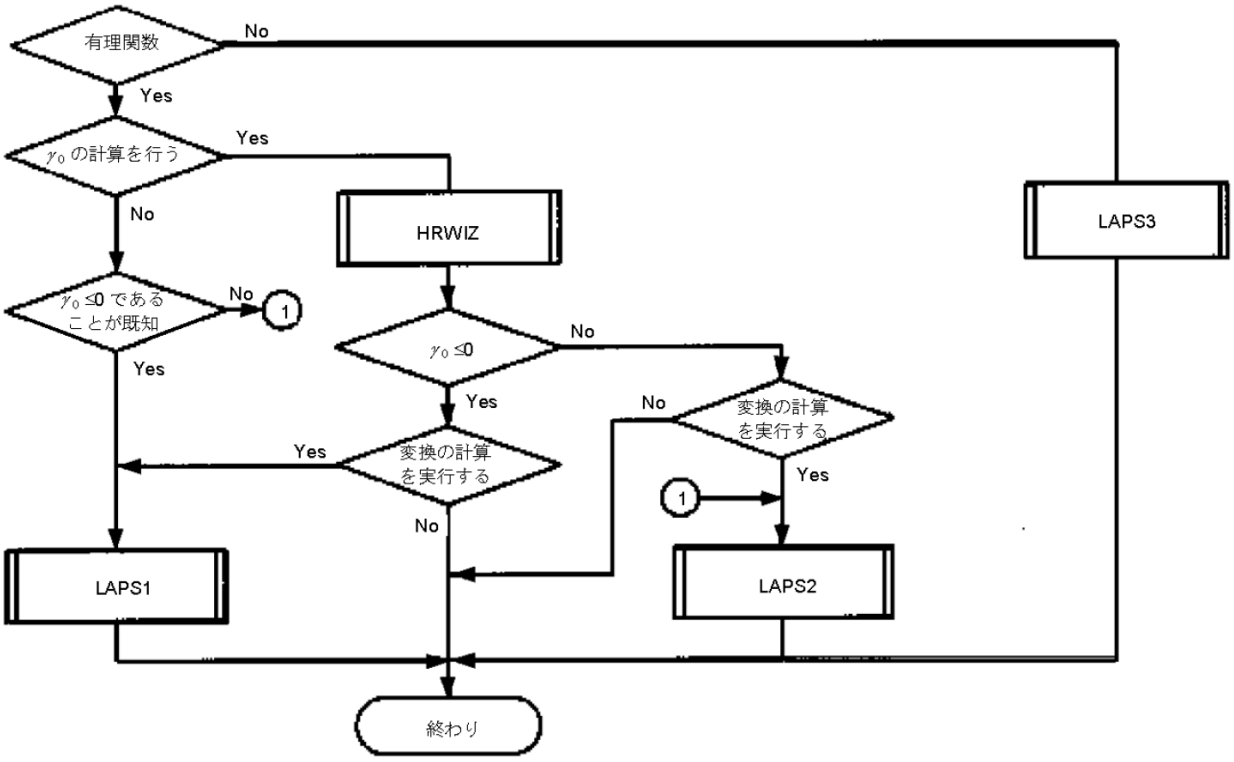


図 8.3　ラプラス変換サブルーチンの使分け

第 9 章
数値微積分

9.1 概要

この章では以下の問題を取り上げる．

数値微分：離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して関数値 $y_i = f(x_i)$, $i = 1, ... n$ が与えられたとき，区間 $[x_1, x_n]$ 内の $x = v$ における $f(x)$ の微分値

$f^{(l)}(v)$, $l \geq 1$

を近似する．さらに 2 次元の微分も扱う．
また，関数 $f(x)$ が与えられたとき，
その導関数

$$f^{(l)}(x) = d^l f(x)/dx^l, \quad l \geq 1$$

をチェビシェフ級数で展開する．

数値積分：離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して関数値 $y_i = f(x_i)$, $i = 1, ..., n$ が与えられたとき，区間 $[x_1, x_n]$ に渡る $f(x)$ の積分値を求める．あるいは関数 $f(x)$ が与えられたとき，積分値

$$S = \int_a^b f(x)\, dx$$

を所要の精度まで求める．さらに多重積分も扱う．

9.2 数値微分

数値微分では，問題を次の二つに分けてある．

a. 離散点入力
微分値を求めるには，標本点 (x_i,y_i) , $i = 1, 2, ..., n,$ に対して適当な補間式をあてはめ，それを微分することにより行う．その際，補間式の種類はたくさん考えられるが，SSL II としては，もっぱら spline 関数を利用した補間式によっている．しかも，spline 補間式の表現は数値的安定性を重視して，B-spline を使った表現を採っている．spline 関数及び B-spline については「第 7 章　補間・近似」を参照されたい．

b. 関数入力
関数 $f(x)$ と定義域 $[a, b]$ が与えられたとき，まず，$f(x)$ を所要の精度でチェビシェフ級数展開する．
すなわち，

$$f(x) \approx \sum_{k=0}^{n-1} c_k T_k\left(\frac{2x-(b+a)}{b-a}\right)$$

と近似する．次に，これを項別微分し，

$$f^{(l)}(x) \approx \sum_{k=0}^{n-l-1} c_k^l T_k\left(\frac{2x-(b+a)}{b-a}\right)$$

として導関数をチェビシェフ級数で展開する．
微分値は，関数が定義される区間の任意の点 $x = v$ において $f^{(l)}(v)$ を評価すること，すなわちチェビシェフ級数を求和することにより求める．
表 9.1 に数値微分のサブルーチンを示す．

9.3 数値積分

数値積分では問題を次の二つに分けてある．

a. 離散点入力
離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して関数値 $y_i = f(x_i)$, $i = 1, ..., n$ が与えられたとき，積分値

$$S = \int_{x_1}^{x_n} f(x)dx \qquad (9.1)$$

を与えられた関数値 y_i だけを用いて近似する．この場合は，得られた近似値がどれだけの誤差をもつかを計算できない．また離散点が等間隔か否かにより，サブルーチンを使い分けなければならない．

b. 関数入力
関数 $f(x)$ $[a, b]$ と積分区間 $[a, b]$ が与えられたとき，積分値

$$S = \int_a^b f(x)dx \qquad (9.2)$$

表 9.1　数値微分のサブルーチン

目的	サブルーチン名	手法	備考
微分値	SPLV (E11-21-0101)	3 次 spline 補間 1	離散点入力
	BIF1 (E11-31-0101)	B-spline 補間式 (I) の利用	
	BIF2 (E11-31-0201)	B-spline 補間式 (II) の利用	
	BIF3 (E11-31-0301)	B-spline 補間式 (III) の利用	
	BIF4 (E11-31-0401)	B-spline 補間式 (IV) の利用	
	BSF1 (E31-31-0101)	B-spline 平滑化式の利用	
	BIFD1 (E11-32-1101)	B-spline 2 次元補間式 (I-I) の利用	離散点入力 2 次元
	BIFD3 (E11-32-3301)	B-spline 2 次元補間式 (III-III) の利用	
	BSFD1 (E31-32-0101)	B-spline 2 次元平滑化式の利用	
導関数と微分値	FCHEB (E51-30-0101)	高速 cosine 変換	関数入力 チェビシェフ級数展開
	GCHEB (E51-30-0301)	後退漸化式	チェビシェフ級数の導関数
	ECHEB (E51-30-0201)	後退漸化式	チェビシェフ級数の求和

を所要の精度まで求める．用意されたサブルーチンは，積分区間と $f(x)$ の形あるいは性質とにより使い分けるようになっている．
また，(9.2) のほかに，

$$\int_0^\infty f(x)dx,$$

$$\int_{-\infty}^\infty f(x)dx,$$

$$\int_a^b dx \int_c^d f(x,y)dy$$

の型の積分も考える．
表 9.2 に数値積分に関して用意されているサブルーチンを示す．

表 9.2　数値積分のサブルーチン

目的	サブルーチン名	手法	備考
1 次元有限区間（等間隔）	SIMP1 (G21-11-0101)	シンプソン則	離散点入力
1 次元有限区間（不等間隔）	TRAP (G21-21-0101)	台形則	
	BIF1 (E11-31-0101)	B-spline 補間式 (I) の利用	
	BIF2 (E11-31-0201)	B-spline 補間式 (II) の利用	
	BIF3 (E11-31-0301)	B-spline 補間式 (III) の利用	
	BIF4 (E11-31-0401)	B-spline 補間式 (IV) の利用	
	BSF1 (E31-31-0101)	B-spline 平滑化式の利用	
2 次元有限区間	BIFD1 (E11-32-1101)	B-spline 2 次元補間式 (I-I) の利用	離散点入力 2 次元
	BIFD3 (E11-32-3301)	B-spline 2 次元補間式 (III-III) の利用	
	BSFD1 (E31-32-0101)	B-spline 2 次元平滑化式の利用	
1 次元有限区間	SIMP2 (G23-11-0101)	適応型シンプソン則	関数入力
	AQN9 (G23-11-0201)	適応型ニュートン・コーツ 9 点則	
	AQC8 (G23-11-0301)	クレンショー・カーチス型積分法	
	AQE (G23-11-0401)	二重指数関数型積分公式	
1 次元半無限区間	AQEH (G23-21-0101)	二重指数関数型積分公式	
1 次元全無限区間	AQEI (G23-31-0101)	二重指数関数型積分公式	
多次元有限領域	AQMC8 (G24-13-0101)	クレンショー・カーチス型積分法	多変数関数入力
多次元領域	AQME (G24-13-0201)	二重指数関数型積分公式	

(1)　数値積分の一般的規約と注意

数値積分に関するサブルーチンを区分するのに，主に次の点に注意した．

積分変数の次元数 ... 1 次元か 2 次元か．
積分区間 ..... 有限区間か全無限区間かそれとも半無限区間か．

各サブルーチンはこの区分に従って名前が決められている．例えば

1 次元有限区間積分
1 次元全無限区間積分

という具合である．この区分のほかに，細かい条件，あるいは手法により特徴づけられる場合は（ ）の中に示した．例えば

1 次元有限区間積分（不等間隔離散点入力，台形則）
1 次元有限区間積分（関数入力，適応型シンプソン則）

という具合である．

数値積分の手法については離散点入力の場合と関数入力の場合とで決定的な違いがある．離散点入力の場合は，与えられた関数値 $y_i = f(x_i), i = 1, ...n$ だけを使って積分値を求めるという制約があるので精度の良い近似値を計算するのは難しい．一方，関数入力の場合は，一般に積分区間のいたる点（特異な場合を除く）における関数値を計算できるので，関数値を多く計算することによって，望む精度まで積分値を計算できるし，また誤差がいくら位かを，計算によって予測することができる．

(2) 関数入力 1 次元有限区間積分

積分 $\int_a^b f(x)dx$ のためのサブルーチンを使用する際の注意事項を述べる．

a. 自動積分ルーチン

積分 $\int_a^b f(x)dx$ のためのサブルーチンとしては，表 9.2 に見られるように， SIMP2, AQN9, AQC8, 及び AQE の四つがある．これらのサブルーチンはすべて自動積分ルーチンである．自動積分ルーチンとは，被積分関数 $f(x)$ ，積分区間 $[a, b]$, 及び積分値に対する要求精度が与えられたとき，その要求精度を満たす積分値を求める積分ルーチンのことを意味し，その目的のために考案されたアルゴリズムを自動積分法という．

自動積分ルーチンでは一般に，最初，少ない分点（関数を計算する点）から積分値の計算を始めて，そのあと徐々に分点を増やしながら，要求精度を満たすまで積分値を改良していき，そして要求精度が満たされた時点で反復を止め，そのときの積分値を出力して計算を終える．

自動積分ルーチンは，興味深いルーチンが世界中で数多く作られ，これに伴って，各ルーチンの信頼性（計算された積分値が要求精度を満たしていること）と経済性（計算量が少ないこと）に関する比較テストも，多くの研究者によって繰り返されてきた．SSL II のサブルーチンは，これらの成果を十分に反映して作られたものである．

b. 適応型手法

自動積分法の代表として，現在最も広く用いられている積分の計算法である．これは，特定の積分公式（例えばシンプソン則，ニュートン・コーツ 9 点則，あるいはガウス側など）を言うのではなく，一般に，被積分関数の変化の急なところでは分点を密にとり，反対に緩やかなところでは疎にとるようにして，なるべく関数の挙動に応じて分点の位置と個数を自動的に調整する機能を具えた方法である．サブルーチン SIMP2 及び AQN9 がこの方法を用いている．

c. サブルーチンの使分け

サブルーチンの使分けを述べる前準備として，実用面で現れる被積分関数の型を，ここで，表 9.3 のように分類しておく．

さて，サブルーチンの使分けで大切なことは，被積分関数がどのサブルーチンに適しているかを知ることである．そこで，各サブルーチンが得意，不得意とする関数を，表 9.3 の分類に従って述べる．

表 9.3　関数の分類

型	意味	例
滑らか型	収束性のよいべき級数展開を持つ解析関数	$\int_0^{\pi} \sin x\,dx$, $\int_0^1 e^{-x}dx$
ピーク型	急激な山や谷が積分区間の中に存在する関数	$\int_{-1}^{1} dx/(x^2+10^{-6})$
振動型	波長の短い，激しい振動を持つ関数	$\int_0^1 \sin 100\pi\, x dx$
特異型	代数特異点 $(x^{\alpha}, -1 < \alpha)$, 対数特異点（$\log x$）を持つ関数	$\int_0^1 dx/\sqrt{x}$, $\int_0^1 \log x dx$
不連続型	関数値や微係数の不連続点を持つ関数	$\int_0^1 [2x]dx$, []はガウス記号, $\int_0^{\pi} \lvert\cos x\rvert\, dx$

SIMP2 .... シンプソン則に基づく適応型手法を用いている．これは，適応型手法として SSL II に最初に組み込まれたものであり，適応型手法の歴史においても，最も古いものである．今やこれに勝る適応型手法が多く現れ，その利用価値も少なくなってきた．すなわち，SIMP2 は同じく適応型手法による AQN9 に多くの面で劣る．したがって，一般目的のためには，SIMP2 よりも AQN9 の方を勧める．

AQN9 ....ニュートン・コーツ 9 点則に基づく適応型手法である．現在のところ，適応型手法の中では信頼性，経済性のどちらの面でも最も優れたものといってよい．このサブルーチンは，関数の局所的挙動を巧みにとらえるため，比較的広範な関数に対して使用することができる．まず，積分区間内に代数特異点，対数特異点あるいは不連続点などの異常点を持つ関数に対して，さらにはピーク型の関数に対しても効果的に処理する．

AQC8　... 関数のチェビシェフ級数展開に基づく手法であるために，この級数展開の収束性の良い関数ほど効果的に処理される．例えば滑らかな関数，振動型の関数がそれである．特異点を持つ関数，ピーク型の関数に対しては弱い．

AQE　.....積分区間 [*a, b*] を変数変換により区間 $(-\infty,\infty)$，へ拡張し，そのあと台形則を適用する手法である．その際，変換後の被積分関数が，$x\to\pm\infty$ のとき二重指数関数的 ($a>0$ として，$\exp(-a\cdot\exp|x|)$) に減衰するような変換が選ばれる．このため，元の区間[*a, b*] の端点付近で変化の激しい関数でも効果的に処理される．特に，端点だけに代数特異点あるいは対数特異点を持つ関数に対しては，他のサブルーチンに比べ最も効率良く積分される．区間 [*a, b*] の中に特異点を持つ関数に対しては弱い．

以上をまとめて，表 9.4 に，関数の型ごとに最優先的に使用すべきサブルーチンに〇印，使用を避けるべきものに×印をつけ，使分けを明らかにした．空欄の部分は，その使用は最良ではないにしても実用に十分耐えられるか，使用してみる価値があることを意味する．また，滑らか型の関数に対しては，これらのどのサブルーチンを使用しても要求精度の積分値が得られるので，ことさらサブルーチンの選択に気を使う必要はないが，関数の計算回数という経済性からいえば，AQC8 が最も良い．

表 9.4　サブルーチンの使分け

型 / ルーチン名	滑らか型	ピーク型	振動型	特異点		不連続型	* 不明
				端点	端点以外		
AQN9		O			O	O	O
AQC8	O	×	O	×	×	×	
AQE				O	×	×	

* 性質がよくわからない関数

(3)　関数入力多次元積分

3 次元以内の多次元積分として，サブルーチン AQMC8 及び AQME が用意されている．いずれも自動積分ルーチンである．以下，特徴を述べる．

AQMC8　.... クレンショー・カーチス型積分法を各次元に適用する．このため，滑らかな関数は勿論のこと，振動型の関数に対して強く，反対に，特異点を持つ関数，ピーク型の関数に対して弱い．

AQME　..... 二重指数関数型積分公式を各次元に適用する．本サブルーチンは，1 次元のサブルーチン AQE, AQEH 及び AQEI で使ったすべての公式を備えているので，積分領域は，有限，半無限，全無限のいずれであってもよく，更に，これらの組合せでできる領域であってもよい．
領域の境界で特異点を持つ関数でも効率良く積分する．領域の内部に特異点を持つ関数に対して弱い．

第 10 章
微分方程式

10.1 概要

この章では以下の問題を取り上げる.
常微分方程式（初期値問題）：
連立 1 階常微分方程式の初期値問題.

$$\left.\begin{aligned} y_1' &= f_1(x, y_1, \ldots\ldots, y_N), \quad y_{10} = y_1(x_0) \\ y_2' &= f_2(x, y_1, \ldots\ldots, y_N), \quad y_{20} = y_2(x_0) \\ &\cdots \qquad\qquad\qquad\quad \cdots \\ y_N' &= f_N(x, y_1, \ldots\ldots, y_N), \quad y_{N0} = y_N(x_0) \end{aligned}\right\} \qquad (10.1)$$

を解く.
高階常微分方程式の初期値問題も (10.1) の形に変形して解くことができる．すなわち,

$$y^{(k)} = f\left(x, y, y', y'', \ldots\ldots, y^{(k-1)}\right),$$
$$y_{10} = y(x_0), y_{20} = y'(x_0), \ldots\ldots, y_{k0} = y^{(k-1)}(x_0)$$

は,

$$y_1 = y(x), y_2 = y'(x), \ldots\ldots, y_k = y^{(k-1)}(x)$$

とおくことにより次の (10.2) のような連立 1 階常微分方程式に置き換えることができる.

$$\left.\begin{aligned} y_1' &= y_2 && y_{10} = y_1(x_0) \\ y_2' &= y_3 && y_{20} = y_2(x_0) \\ &\cdots && \cdots \\ y_{k-1}' &= y_k && y_{k-10} = y_{k-1}(x_0) \\ y_k' &= f(x, y_1, y_2, \ldots, y_k) && y_{k0} = y_k(x_0) \end{aligned}\right\} \qquad (10.2)$$

10.2 常微分方程式

初期値問題 $y' = f(x,y)$, $y(x_0) = y_0$ を区間 $[x_0, x_e]$ の上で数値的に解くということは，図 10.1 に示すように，離散点,

$x_0 < x_1 < x_2 < \ldots < x_e$

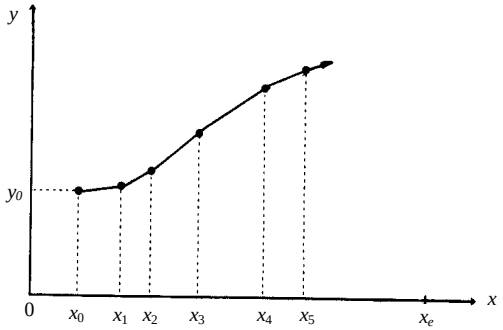


図 10.1 $y' = f(x, y), y(x_0) = y_0$ の近似解

における近似解を順次求めていくことを意味する.

a. 解の出力

図 10.1 において，解の出力点 x_1, x_2, x_3, ... は，利用者が指定した点でなくてはならないこともあれば，微分方程式サブルーチンがキザミ制御 (step size control) の結果として選んだ点であってよいこともある.
ところで，利用者の立場から見ると，微分方程式を数値的に解く目的としては，以下のいくつかが挙げられる.

(a) x_e における解 $y(x_e)$ だけを求める.
(b) サブルーチンがキザミ制御の結果として選んだ点における解を求める．この場合は，解の挙動を知るのが目的であり，解の出力点に関して何ら制限を与えず，ただ解の挙動を知るのに十分な分布を成す離散点上で解が求まればよい.
(c) 利用者の意図した点列 $\{\xi_j\}$ の上で解を求める．あるいは，一定間隔にとられた点列上で解を求める.

SSL II の微分方程式サブルーチンは，解の出力方法（サブルーチンから利用者プログラムに戻るタイミング）として，上記の目的に合わせて，次の 2 つを設定している.

- 最終値出力
  解 $y(x_e)$ が得られたとき利用者プログラムに戻る．上記 (c) の目的のためには，x_e を順番に ξ_i , $i = 1, 2, ...,$ にとりながらサブルーチンを繰り返し呼び出す．
- ステップ出力
  キザミ制御のもとで，1 ステップ積分するごとに利用者プログラムに戻る．利用者は，繰り返しサブルーチンを呼び出すことにより，上記 (b) の目的を果すことができる．

SSL II のサブルーチン ODRK1, ODAM 及び ODGE は，これら 2 つの出力方法を備えている．利用者はパラメタの指定により出力方法を選ぶことができる．

b.　スティフな微分方程式

ここでは，多くの応用分野で現れるスティフな微分方程式について述べ，その定義を明らかにし，簡単な例を挙げる．

(10.1) をベクトルの形で表すと，

$$\boldsymbol{y}' = \boldsymbol{f}(x, \boldsymbol{y}), \quad \boldsymbol{y}(x_0) = \boldsymbol{y}_0 \tag{10.3}$$

となる．ここで，

$\boldsymbol{y} = (y_1, y_2, ..., y_N)^{\mathrm{T}}$,
$\boldsymbol{f}(x, \boldsymbol{y}) = (f_1(x, \boldsymbol{y}), f_2(x, \boldsymbol{y}), ..., f_N(x, \boldsymbol{y}))^{\mathrm{T}}$,
$f_i(x, \mathbf{y}) = f_i(x, y_1, y_2, ..., y_N)$ である．

手始めに，$\boldsymbol{f}(x,\boldsymbol{y})$ が線形，すなわち，

$$\boldsymbol{f}(x, \boldsymbol{y}) = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{\Phi}(x) \tag{10.4}$$

である場合を考える．ここでは $\boldsymbol{A}$ は定数の係数行列，$\boldsymbol{\Phi}(x)$ は，ある関数ベクトルである．このとき (10.3) の解は，$\boldsymbol{A}$ の固有値 λ_i と対応する固有ベクトル $\boldsymbol{u}_i$ を用いて，

$$\boldsymbol{y}(x) = \sum_{i=1}^{N} k_i e^{\lambda_i x} \boldsymbol{u}_i + \boldsymbol{\Psi}(x) \tag{10.5}$$

k_i : 定数

と表せる．いま，(10.5) の λ_i , $\boldsymbol{\Psi}(x)$ に関して，次のように仮定する．

(a)　$\mathrm{Re}(\lambda_i)<0$, $i=1, 2, ..., N$

(b)　$\boldsymbol{\Psi}(x)$ は，どの $e^{\lambda_i x}$ よりも滑らか（すなわち，収束性の良いべき級数展開を持つ）関数である．

このような条件下では，$x \to \infty$ のとき，

$$\sum_{i=1}^{N} k_i e^{\lambda_i x} \boldsymbol{u}_i \to 0 \tag{10.6}$$

となり，解 $\boldsymbol{y}(x)$ は，$\boldsymbol{\Psi}(x)$ に近づく．このように，$\boldsymbol{\Psi}(x)$ が支配的になったあとは，$\boldsymbol{\Psi}(x)$ に対する近似解を求めれば十分であり，したがって，キザミ幅も比較的大きく採れることができる．

しかし，オイラー法，ルンゲ・クッタ法などの公式では，キザミ幅をある値 (λ_i に依存する) よりも大きく採ると，あるステップで発生した誤差が，後続のステップで増幅されるという現象が起こる．したがって，これらの公式を適用する限り，キザミ幅の大きさは制約され，実際，$\max(|\mathrm{Re}(\lambda_i)|)$ が大きければ大きいほど，それに逆比例してキザミ幅は一層小さく採らざるを得ない．

このように，解 $\boldsymbol{y}(x)$ が実質的に滑らかな関数 $\boldsymbol{\Psi}(x)$ で近似できる状況下にあるにもかかわらず，小さなキザミ幅で積分せざるを得ないという困難さは，結局，解を近似するのに十分なキザミ幅と，誤差の増幅を防ぐのに必要なキザミ幅との間の不均衡に起因する．

(10.3) において $\boldsymbol{\Phi}(x)=\boldsymbol{0}$, すなわち $\boldsymbol{\Psi}(x)=\boldsymbol{0}$ の場合は，解 $\boldsymbol{y}(x)$ は減衰し，したがって，実質的に，小さい $|\mathrm{Re}(\lambda_i)|$ に対応する項 $k_i e^{\lambda_i x}\boldsymbol{u}_i$ で近似される．

この場合でも，$\max|\mathrm{Re}(\lambda_i)|$ が大きいと上記の困難さが生ずる．

さて，スティフな微分方程式を次のように定義する．

定義 1：線形微分方程式

$$\boldsymbol{y}' = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{\Phi}(x) \tag{10.7}$$

が，以下の (10.8) , (10.9) 満たすとき，その方程式はスティフであると言う．

$$\mathrm{Re}(\lambda_i)<0, \quad i=1, 2, \cdots, N \tag{10.8}$$

$$\frac{\max\left(\left|\mathrm{Re}(\lambda_i)\right|\right)}{\min\left(\left|\mathrm{Re}(\lambda_i)\right|\right)} \gg 1 \tag{10.9}$$

(10.9) の左辺は，スティフ率 (stiff ratio) と呼ばれ，この値が大きいか小さいかにより，(10.7) は強スティフ又は弱スティフであると言う．実際の応用分野では，スティフ率が 10^6 にも及ぶ強スティフな方程式もしばしば現れる．

以下の (10.10) は，スティフな線形微分方程式の例であり，解は (10.11) で表せる(図 10.2 参照)．

$$\boldsymbol{y}' = \begin{pmatrix} 998 & 1998 \\ -999 & -1999 \end{pmatrix}\boldsymbol{y}, \quad \boldsymbol{y}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{10.10}$$

$$\boldsymbol{y} = e^{-x}\begin{pmatrix} 2 \\ -1 \end{pmatrix} + e^{-1000x}\begin{pmatrix} -1 \\ 1 \end{pmatrix} \tag{10.11}$$

明らかに $x \to \infty$ のとき $y_1 \to 2e^{-x}$，$y_2 \to -e^{-x}$ である．

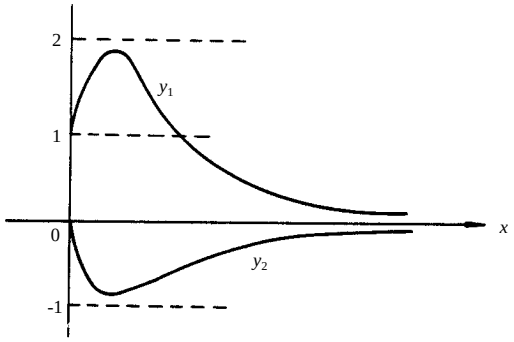


図 10.2　解 (10.11) のグラフ

次に $\boldsymbol{f}(x, \boldsymbol{y})$ が非線形である場合を考える．このときは，ヤコビアン行列

$$\boldsymbol{J} = \frac{\partial \boldsymbol{f}(x, \boldsymbol{y})}{\partial \boldsymbol{y}}$$

の固有値がスティフ性を決定するが，固有値は x と共に変化する．そこで，$\boldsymbol{f}(x, \boldsymbol{y})$ が非線形のときは，定義 1 を次のように拡張する．

定義 2：非線形微分方程式

$$\boldsymbol{y}' = \boldsymbol{f}(x, \boldsymbol{y}) \qquad (10.12)$$

が，ある区間 I で，以下の (10.13)，(10.14) を満たすとき，その方程式は区間 I でスティフであると言う．

$$\mathrm{Re}(\lambda_i(x)) < 0, \quad i = 1,2,..,N \quad (10.13)$$
$$x \in I$$

$$\frac{\max\left(\left|\mathrm{Re}(\lambda_i(x))\right|\right)}{\min\left(\left|\mathrm{Re}(\lambda_i(x))\right|\right)} \gg 1, \quad x \in I \qquad (10.14)$$

ここで $\lambda_i(x)$ は $\boldsymbol{J}$ の固有値である．

解こうとしている方程式がスティフであるか否かを見分ける方法については，一般に次のことが言える．
方程式が (10.7) のように線形ならば，$\boldsymbol{A}$ の固有値を計算することにより直接，スティフ性を調べることができる．一方，方程式が非線型であるときは，スティフ性検出の能力を持つサブルーチン ODAM を使えばよい．ODAM は，非スティフな方程式を解く手法のひとつであるアダムス法を適用するが，解法の過程で，方程式がアダムス法にとって苦手な問題であると判断したとき，その旨をパラメタ ICON に出力する．そのときは方程式がスティフであると思ってよい．

SSL II では，スティフな方程式を解くサブルーチンとして ODGE を用意している．

c.　サブルーチンの使分け

微分方程式のサブルーチンを表 10.1 に示す．その使分けは以下のように行う．

(a) 方程式がスティフなら ODGE を使うこと．
(b) 方程式が非スティフなら ODRK1，ODAM のいずれかを使うこと．ODRK1 は特に，以下に示す条件の場合に効果的に利用できる．
- 微係数 $\boldsymbol{f}(x, \boldsymbol{y})$ を評価するための計算量が少ない．
- 解に対する要求精度が高くない
- 独立変数 x の特定の点列の上で解の出力を望むとき，その点列の間隔が十分に広い．

これらの条件のいずれでもないときは ODAM を使うこと．
(c) 方程式がスティフであるか否か不明なときは，まず，ODAM を使うこと．これにより，もしスティフ性が検出されたときは，ODGE を使うようにすればよい．

表 10.1　常微分方程式のサブルーチン

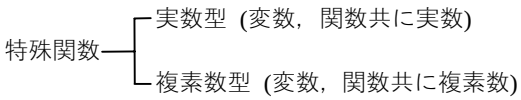
目的	サブルーチン名	手法	備考
初期値問題	RKG (H11-20-0111)	ルンゲ・クッタ・ギル法	キザミ固定
	HAMNG (H11-20-0121)	ハミング法	キザミ可変
	ODRK1 (H11-20-0131)	ルンゲ・クッタ・ヴァーナー法	キザミ可変
	ODAM (H11-20-0141)	アダムス法	キザミ可変 次数可変
	ODGE (H11-20-0151)	ギア法	キザミ可変 次数可変 （スティフ方程式）

第 11 章
特殊関数

11.1 概要

SSL II の特殊関数とは，FORTRAN の基本関数に含まれていない関数のことであり，これらの関数値を求めるものである．このような特殊関数は，変数と関数が実数型の場合と複素数型の場合に大別され，多くの種類がある．

すなわち，



と分けられる．以下に，幾つかの特殊関数サブルーチンに共通な事項を述べる．

(1) 精度について

関数で重要なことは，精度と速度のバランスであるが計算式の選定にあたっては，この点に留意し，理論精度（近似式の精度）を単精度で 8 桁程度，倍精度で 18 桁程度となるように選定した．そして，一部の単精度の関数は，精度を保証するために，内部計算を倍精度で行っている．しかし，計算された関数値の精度は，計算機の計算桁数に依存するので，理論精度がいつでも保証されるわけではない．

単精度サブルーチンの精度の検定は，倍精度サブルーチンの結果を原器と見なし，倍精度サブルーチンの精度の検定は，4 倍精度で計算するサブルーチン（倍精度サブルーチンよりも十分精度が良いと確認されるサブルーチン）の結果を原器と見なして比較を行い，十分実用的であることを確認した．

(2) 速度について

特殊関数は，どちらかというと精度に重点を置き，その上で，速度を上げる配慮をした．実数型は，複素数型の一部であるが，速度を上げるために，別々に用意してある．また，同じ理由で，特殊関数の相互間に関係のある関数も別々に用意してある．特に，単精度サブルーチンと倍精度サブルーチンは別々に用意した．また頻繁に使われるサブルーチンは，一般的なサブルーチンの他に，専用のサブルーチンを用意してある．

(3) ICON について

特殊関数は，その計算のために，FORTRAN 基本関数の指数関数，三角関数などを利用しているが，それらの中でオーバフロー，アンダフローなど，エラーが発生すると，トラブルの真の原因の発見が遅れる．そこで，特殊関数の側で，できるだけ早めに，トラブルを検出するように配慮し，検出されたトラブルは，パラメタ ICON に出力するようにしてある．

(4) 呼び出し方法について

特殊関数の計算では，いろいろな状況が起きるので，その状況を知らせるために，パラメタ ICON を設定するようにしてある．このため，サブルーチンは CALL 文で呼び出すサブルーチン副プログラムで作成されている．

FORTRAN 基本関数は関数副プログラムであるので，特殊関数も関数副プログラムとした方がよいという考え方があるが，これを採用しなかったのは，上記の理由による．

11.2 楕円積分

楕円積分は，表 11.1 のような種類に分かれる．

表 11.1 楕円積分のサブルーチン

項目		数学記号	サブルーチン名
完全	第 1 種完全楕円積分	$K(k)$	CELI1 (I11-11-0101)
	第 2 種完全楕円積分	$E(k)$	CELI2 (I11-11-0201)

完全楕円積分の計算法には，2 次の反復法があるが，この方法は変数の大きさによって，速度が変

る欠点がある．そこで，本サブルーチンは，変数の大きさに関係なく速度を一定に保つように，近似式を用いて計算する．

11.3　指数積分

指数積分は，表 11.2 のようになっている．

表 11.2　指数積分のサブルーチン

項目	数学記号	サブルーチン名
指数積分	$E_i(-x), \quad x>0$	EXPI (I11-31-0101)
	$\overline{E}_i(-x), \quad x>0$	

指数積分は，計算の困難な関数で，計算式はいくつかの区間に分けられて，いろいろな近似式が使われている．

11.4　正弦・余弦積分

正弦・余弦積分は，表 11.3 のようになっている．

表 11.3　正弦・余弦積分のサブルーチン

項目	数学記号	サブルーチン名
正弦積分	$S_i(x)$	SINI (I11-41-0101)
余弦積分	$C_i(x)$	COSI (I11-41-0201)

11.5　フレネル積分

フレネル積分は表 11.4 のようになっている．

表 11.4　フレネル積分のサブルーチン

項目	数学記号	サブルーチン名
正弦フレネル積分	$S(x)$	SFRI (I11-51-0101)
余弦フレネル積分	$C(x)$	CFRI (I11-51-0201)

11.6　ガンマ関数

ガンマ関数には，表 11.5 のものが用意されている．

表 11.5　ガンマ関数のサブルーチン

項目	数学記号	サブルーチン名
第 1 種不完全ガンマ関数	$\gamma(\nu,x)$	IGAM1 (I11-61-0101)
第 2 種不完全ガンマ関数	$\Gamma(\nu,x)$	IGAM2 (I11-61-0201)

（完全）ガンマ関数 $\Gamma(\nu)$ と第 1 種及び第 2 種不完全ガンマ関数の間には，

$$\Gamma(\nu)=\gamma(\nu,x)+\Gamma(\nu,x)$$

の関係がある．

なお $\Gamma(\nu)$ については FORTRAN 基本外部関数を利用のこと．

11.7　誤差関数

誤差関数には表 11.6 のものが用意されている．

表 11.6　誤差関数のサブルーチン

項目	数学記号	サブルーチン名
逆誤差関数	$\mathrm{erf}^{-1}(x)$	IERF (I11-71-0301)
逆余誤差関数	$\mathrm{erfc}^{-1}(x)$	IERFC (I11-71-0401)

逆誤差関数と逆余誤差関数の間には

$$\mathrm{erf}^{-1}(x)=\mathrm{erfc}^{-1}(1-x)$$

の関係があり，それぞれ x の範囲に応じて適切な方で計算している．

なお，誤差関数 $\mathrm{erf}(x)$ については FORTRAN 基本外部関数を利用のこと．

11.8　ベッセル関数

実変数のベッセル関数は，表 11.7 のような種類に分けられ，非常に多くの種類がある．また利用頻度も高い．特に，0 次，1 次は利用頻度が高いので，専用サブルーチンを用意した．0 次，1 次を計算するときには，専用サブルーチンを使用する方が速度的に有利である．

複素変数のベッセル関数は，表 11.8 のようになっている．

表 11.7　実変数のベッセル関数のサブルーチン

項目		数学記号	サブルーチン名
第一種	0 次ベッセル関数	$J_0(x)$	BJ0 (I11-81-0201)
	1 次ベッセル関数	$J_1(x)$	BJ1 (I11-81-0301)
	整数次ベッセル関数	$J_n(x)$	BJN (I11-81-1001)
	実数次ベッセル関数	$J_\nu(x)$ $(\nu \geq 0.0)$	BJR (I11-83-0101)
	0 次変形ベッセル関数	$I_0(x)$	BI0 (I11-81-0601)
	1 次変形ベッセル関数	$I_1(x)$	BI1 (I11-81-0701)
	整数次変形ベッセル関数	$I_n(x)$	BIN (I11-81-0701)
	実数次変形ベッセル関数	$I_\nu(x)$ $(\nu \geq 0.0)$	BIR (I11-83-0301)
第二種	0 次ベッセル関数	$Y_0(x)$	BY0 (I11-81-0401)
	1 次ベッセル関数	$Y_1(x)$	BY1 (I11-81-0501)
	整数次ベッセル関数	$Y_n(x)$	BYN (I11-81-1101)
	実数次ベッセル関数	$Y_\nu(x)$	BYR (I11-83-0201)
	0 次変形ベッセル関数	$K_0(x)$	BK0 (I11-81-0801)
	1 次変形ベッセル関数	$K_1(x)$	BK1 (I11-81-0901)
	整数次変形ベッセル関数	$K_n(x)$	BKN (I11-81-1301)
	実数次変形ベッセル関数	$K_\nu(x)$	BKR (I11-83-0401)

表 11.8　複素変数のベッセル関数のサブルーチン

項目		数学記号	サブルーチン名
第一種	整数次ベッセル関数	$J_n(z)$	CBJN (I11-82-1301)
	実数次ベッセル関数	$J_\nu(z)$ $(\nu \geq 0.0)$	CBJR (I11-84-0101)
	整数次変形ベッセル関数	$I_n(z)$	CBIN (I11-82-1101)
第二種	整数次ベッセル関数	$Y_n(z)$	CBYN (I11-82-1401)
	整数次変形ベッセル関数	$K_n(z)$	CBKN (I11-82-1201)

11.9　正規分布関数

正規分布関数には，表 11.9 のものが用意されている．

表 11.9　正規分布関数のサブルーチン

項目	数学記号	サブルーチン名
正規分布関数	$\phi(x)$	NDF (I11-91-0101)
余正規分布関数	$\psi(x)$	NDFC (I11-91-0201)
逆正規分布関数	$\phi^{-1}(x)$	INDF (I11-91-0301)
逆余正規分布関数	$\psi^{-1}(x)$	INDFC (I11-91-0401)

第 12 章
擬似乱数

12.1 概要

本章では種々の分布に従う実数又は整数の擬似乱数の生成と検定を扱う．

12.2 擬似乱数の生成

種々の確率分布に従う擬似乱数はすべて，区間(0, 1) での一様分布から得られた擬似乱数に適当な変換を施すことにより得られる．すなわち，ある分布の擬似乱数を生成するとき，その分布の確率密度関数を $g(x)$ とするならば，擬似乱数 y はその分布の累積分布関数 $F(y)=\int_0^y g(x)dx$ の逆関数 (12.1) で示される．

$$y = F^{-1}(u) \qquad (12.1)$$

ここに，u は生成された一様乱数である．

ただし，離散型分布の擬似乱数生成は中間の計算で少し複雑になる．例えばサブルーチン RANP2 は，まず，ポアソン分布の累積分布表を作成する．次に累積分布表を参照する索引表を作成し，それから累積分布表で与えられた分布に従う擬似乱数を生成するといった具合である．

SSL II で準備する擬似乱数生成サブルーチンの種類を表 12.1 に示す．なお，これらのサブルーチンには初期値を与えるパラメタが設けられており，これによってある初期値のもとでの擬似乱数列の生成を続けるかどうかのコントロールを行なうことができる．（通常は，ただ一回だけの初期値設定を行ったかたちで使用される）．

表 12.1 擬似乱数生成サブルーチンの種類

種類	サブルーチン名
一様乱数 (0, 1)	RANU2 (J11-10-0101)
一様乱数 (0, 1)（シャフル型）	RANU3 (J11-10-0201)
指数乱数	RANE2 (J11-30-0101)
正規乱数（高速型）	RANN1 (J11-20-0301)
正規乱数	RANN2 (J11-20-0101)
ポアソン乱数	RANP2 (J12-10-0101)
二項乱数	RANB2 (J12-20-0101)

12.3 擬似乱数の検定

擬似乱数を利用するためには，その乱数の性質を充分に認識する必要がある．すなわち，計算機により算術的に生成された擬似乱数列が“特定の確率分布に従う独立な確率変数の系列の実現値”とみなせる否かを検定しておくことが必要である．

SSL II では種々の確率分布に従う擬似乱数を，区間 (0, 1) の擬似一様乱数に適当な変換を施すことにより生成している．そこでは，幾つかの異なった乱数列を生成することを容易にするために，擬似一様乱数生成の初期値を与えるパラメタ IX を用意している．

通常， IX = 0 として生成すればよく，その場合の擬似一様乱数列に対する検定結果はサブルーチン RANU2 の「使用上の注意」に示されている．

一般には IX の値に応じて，擬似乱数列の性質が異なる．そこで，SSL II では，生成した擬似乱数列の検定をするために，表 12.2 のような検定用ルーチンを用意している．

表 12.2　擬似乱数検定サブルーチン

検定項目	サブルーチン名	備考
頻度テスト	RATF1 (J21-10-0101)	等出現性の検定
上昇・下降連テスト	RATR1 (J21-10-0201)	無規則性の検定

以下に，統計的仮説検定の概念と，SLL II で採用しているカイ二乗 (χ^2) 検定法について概説し，幾つかの注意事項を述べる．

統計的仮説検定

無作為標本から得られる統計量の実現値によって，ある仮説を棄てたり，受け入れたりする方法を統計的仮説検定という．

例えば，サイコロを何回か投げて，どのように目が出るかを調べることによって，“サイコロが正常である”か否かを検定することを考える．

いま，5 回続けて投げて，全部等しい目（例えば 1）が出たとする．もし，“サイコロが正常である”すなわち“どの目の出る確率も等しい”とするなら，続けて全部等しい目が出る確率は $(1/6)^5 = 1/7776$ であって，同様な実験を 7776 回も繰り返し行ったときに平均的に 1 回起こることが期待される．したがって，1 度の実験で全部等しい目が出たことは，“サイコロが正常である”という仮説に疑いがあることになる．

このように，ある仮説のもとで実験を行って，確率 α%（習慣的に，5%又は 1%）以下で起こる事象が，ただ 1 回の実験で起きたとすると，その仮説は疑わしいとして棄てる．そうでない場合は仮説を受け入れるわけである．このときの仮説は，棄てるか棄てられないかとい立場から検定されるもので，これを帰無仮説といい，確率α%以下の領域を棄却域という．また，棄てる基準のことを有意水準という．

このような検定方法を採用すると，実は仮説が真であるにもかかわらず，これを棄てるという誤りをおかすことがあり，α%を危険率ともいう．

カイ二乗 (χ^2) 検定法

いま，有意水準をα%とする．また，母集団が l 個の排反な階級 $c_1, c_2, ..., c_l$ に分類されており，n 個の抽出において，現実にこれらに属する度数（実現度数）が $f_1, f_2, ..., f_l$ であり，帰無仮説に基づくそれぞれの期待度数が $F_1, F_2, ..., F_l$ であるとする．

階級	$c_1, c_2, ..., c_l$	計
実現度数	$f_1, f_2, ..., f_l$	n
期待度数	$F_1, F_2, ..., F_l$	n

このとき，実現度数の期待度数からの隔りの度合として，

$$\chi_0^2 = \sum_{i=1}^{l} \frac{(f_i - F_i)^2}{F_i} \tag{12.2}$$

なる値を考える．期待度数からの隔りが大きいと χ_0^2 の値も大きくなり，その大きさの程度によって仮説を棄却することになる．

ところで，大きさ n の標本に対して，標本ごとに変動する度数を $\tilde{f}_1, \tilde{f}_2, ..., \tilde{f}_l$ と表すと，これは確率変数であり

$$\chi^2 = \sum_{i=1}^{l} \frac{(\tilde{f}_i - F_i)^2}{F_i} \tag{12.3}$$

なる値は統計量である．ここで，期待度数 F_i が十分に大きいとき，χ^2 は近似値に自由度 l-1 のカイ二乗分布に従って分布する．

そこで，自由度 l-1 のカイ二乗分布において，有意水準α%に相当する点 χ_α^2 を求め，次のような検定を行う．

$\chi_\alpha^2 < \chi_0^2$ なら仮説は棄却する．

$\chi_\alpha^2 \geq \chi_0^2$ なら仮説は棄却しない．

このような検定法を，カイ二乗検定法という．

実現度数と期待度数は，検定の内容（頻度テスト，連テスト等）に依存する値である．

使用上の注意

(1)　標本の大きさ n について

標本の大きさ n は，十分に大きくとることが必要である．すなわち，(12.3) の統計量は，十分大きな n に対して自由度 l-1 のカイ二乗分布に近似されるからである．したがって，n が小さいとカイ二乗分布による近似が悪く，検定の結果に対する信頼性が低い．

一般には，期待度数 F_i が

$$F_i > 10 \quad , \quad i=1, 2, \cdots, l \tag{12.4}$$

とすることが望ましい．(12.4) を満たさない場合は，いくつかの階級をまとめて自由度を低くする必要がある．

第 II 部 サブルーチン使用方法

A21-11-0101 AGGM, DAGGM

行列の和（実行列）
CALL AGGM (A, KA, B, KB, C, KC, M, N, ICON)

(1) 機能

$m \times n$ の実行列 $\boldsymbol{A}$ と実行列 $\boldsymbol{B}$ の和 $\boldsymbol{C}$ を求める.

$$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B}$$

ここで $\boldsymbol{C}$ は，$m \times n$ の実行列である.
$m, n \geq 1$ であること.

(2) パラメタ

A ····· **入力**. 行列 $\boldsymbol{A}$.
A (KA, N) なる 2 次元配列.
KA ····· **入力**. 配列 A の整合寸法 (≥ M).
B ····· **入力**. 行列 $\boldsymbol{B}$.
B (KB, N) なる 2 次元配列.
KB ····· **入力**. 配列 B の整合寸法 (≥M).
C ····· **出力**. 行列 $\boldsymbol{C}$.
C (KC, N) なる 2 次元配列.
(使用上の注意①参照)
KC ····· **入力**. 配列 C の整合寸法 (≥ M).
M ····· **入力**. 行列 $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ の行数 m.
N ····· **入力**. 行列 $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ の列数 n.
ICON ····· **出力**. コンディションコード.
表 AGGM-1 参照.

表 AGGM-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	M<1, N<1, KA<M, KB<M 又は, KC<M であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .....MGSSL
② FORTRAN 基本関数...なし,

b. 注意
① 領域の節約について
配列 A あるいは，B 上の内容を保存する必要のない場合は，次のように呼び出すことにより領域が節約できる.

- 配列 A の内容が不要の場合.

  CALL AGGM (A, KA, B, KB, A, KA, M, N, ICON)

- 配列 B の内容が不要の場合.

  CALL AGGM (A, KA, B, KB, B, KB, M, N, ICON)

この場合，行列 $\boldsymbol{C}$ は配列 A あるいは，B 上に得られる.

c. 使用例
実行列 $\boldsymbol{A}$，$\boldsymbol{B}$ の和を求める．$m, n \leq 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(50,50),B(60,60),
     *          C(100,100)
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
      DATA KA/50/,KB/60/,KC/100/
   10 READ(5,100) M,N
      IF(M.EQ.0) STOP
      WRITE(6,150)
      READ(5,200) ((A(I,J),I=1,M),J=1,N)
      READ(5,200) ((B(I,J),I=1,M),J=1,N)
      CALL AGGM(A,KA,B,KB,C,KC,M,N,ICON)
      IF(ICON.NE.0) GOTO 10
      CALL PGM(IA,1,A,KA,M,N)
      CALL PGM(IB,1,B,KB,M,N)
      CALL PGM(IC,1,C,KC,M,N)
      GOTO 10
  100 FORMAT(2I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** MATRIX ADDITION **')
      END
```

本使用例中のサブルーチン PGM は，実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている.

E11-11-0201 AKHER, DAKHER

エイトケン・エルミート補間
CALL AKHER (X, Y, DY, N, V, M, EPS, F, VW, ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して，関数値 $y_i = f(x_i)$，1 階微係数 $y'_i = f'(x_i)$, $i = 1, ..., n$ が与えられたとき，$x = v$ における補間値をエイトケン・エルミート補間法により求める．$n \geq 1$ であること．

(2) パラメタ

X ……**入力**．離散点 x_i.
大きさ n の 1 次元配列.

Y ……**入力**．関数値 y_i.
大きさ n の 1 次元配列.

DY ……**入力**．1 階微係数 y'_i.
大きさ n の 1 次元配列.

N ……**入力**．離散点の個数 n.

V ……**入力**．補間したい点の x 座標 v.

M ……**入力**．補間に使う離散点の個数の上限 ($\leq n$).
出力．実際に使用した離散点の個数．(使用上の注意参照).

EPS ……**入力**．しきい値.
出力．補間値の絶対誤差.
(使用上の注意参照).

F ……**出力**．補間値.

VW ……**作業領域**．大きさ $5n$ の 1 次元配列.

ICON ……**出力**．コンディションコード.
表 AKHER-1 参照.

表 AKHER-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	υ が離散点 xi の一つであった.	F = yi とする.
30000	n < 1, M = 0, xi ≥xi+1 のいずれかであった.	F = 0.0 として処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AFMAX, MGSSL

② FORTRAN 基本関数 ... ABS, IABS

b. 注意

① 収束判定条件について
まず補間多項式の次数と，数値的なふるまいを考える．ここで $x=v$ の近傍での j 個の離散点を用いて補間した補間値を Z_j で表す．(離散点のとり方は，$x=v$ に近い順とする．)

次に間差 D_j を次のように定義する.

$$D_j \equiv Z_j - Z_{j-1} \quad j = 2,\ldots,m$$

ここで m は補間に使う離散点の個数の上限である．一般に，補間多項式の次数を上げていくと，$|D_j|$は図 AKHER-1 のようなふるまいを示す.

図 AKHER-1 で l は近似多項式の打切り誤差と計算誤差が互いに同程度になる事を示しており，一般に Z_l が数値的には最適の補間値とみなすことができる.

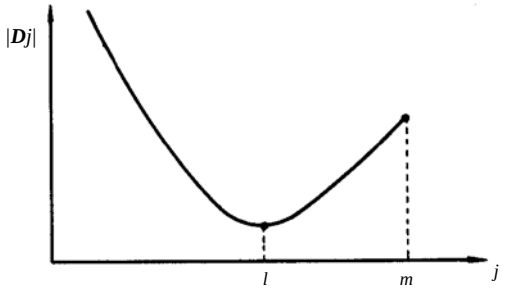


図 AKHER-1

② EPS の与え方
次の条件を考慮する.
収束判定は①に従って行われるが，D_j は近似関数により種々のふるまいを示す．図 AKHER-2 で示されるように振動する場合もある.

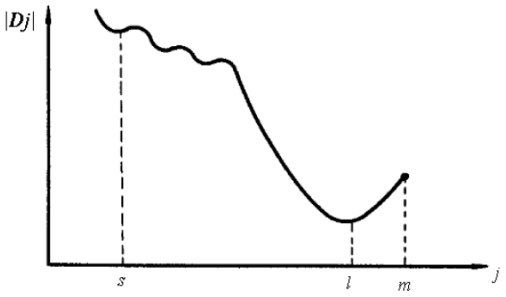


図 AKHER-2

この場合，補間値として Z_s を採用するのは好ましくなく Z_l を採用すべきである．この考えに基いて出力すべき補間値を次のように決定する.
$D_2, D_3, \cdots, D_m$ と計算していく過程で,

(a) $|D_j|>|EPS|$, j=2, 3, ⋯, m であれば,

$$|D_l| = \min_j\left(|D_j|\right) \tag{3.1}$$

となる l を求め Z_l, l, $|D_l|$ をそれぞれパラメタ F，M，EPS に出力し,

(b) $|D_j| \leq |EPS|$となり始めたならば，その時点からは,

$$|D_l| \leq |D_{l+1}| \tag{3.2}$$

となる l を求め Z_l，l，$|D_l|$を出力する．
もし (3.2) がいつまでも成立たなければ $l = m$ として $Z_m, m, |D_m|$ を出力する．
又 EPS = 0.0 と入力すると $|D_j|$ が最小となる時点の Z_j を補間値として出力することになる．

③ M の与え方

(a) もし，真の関数が，$x=v$ の近傍で高々$2k$-1 次の多項式で近似できることが分かっていれば，多項式補間法においても，高々$2k$-1 次の多項式で行うのが自然である．もし，このような状況にあるならパラメタ M に k を入れる．

(b) 上記(a)の事実が分からないときは，パラメタ M に，パラメタ N の値と等しい値を入れる．

(c) もし，ある定まった個数の離散点を，収束判定することなく無条件に使って，補間値を求めたいときにはパラメタ M に，その個数に-1 を乗じた値を入れる．

c. 使用例

入力パラメタの値を読込み，補間値 F を求める．
$n \le 30$ の場合．

```
C     **EXAMPLE**
      DIMENSION X(30),Y(30),DY(30),VW(150)
      READ(5,500) N,(X(I),Y(I),DY(I),
     *            I=1,N)
      WRITE(6,600)(I,X(I),Y(I),DY(I),
     *            I=1,N)
   10 READ(5,510) M,EPS,V
      IF(M.GT.30) STOP
      CALL AKHER(X,Y,DY,N,V,M,EPS,F,VW,
     *           ICON)
      WRITE(6,610) ICON,M,V,F
      IF(ICON.EQ.30000) STOP
      GO TO 10
  500 FORMAT(I2/(3F10.0))
  510 FORMAT(I2,2F10.0)
  600 FORMAT(15X,I2,5X,3E20.8)
  610 FORMAT(20X,'ICON =',I5,10X,'M =',
     * I2/20X,'V          =',E20.8/
     * 20X,'COMPUTED VALUES =',E20.8)
      END
```

(4) 手法概要

離散点 x_i を v に近い順に並べかえて $v_1, v_2, ..., v_n$ とし，それに対応して $y_i = f(v_i)$，$y'_i = f'(v_i)$とする．

点 v_i において関数値 y_i を持つ条件を記号（i,0）で，点 v_i において 1 階の微分係数 y'_i を持つ条件を記号（i,1）で表すことにするとき，$2m$ 個の条件(1,0), (1,1), (2,0), (2,1),...,(m,0), (m,1) を満足する高々(2m-1)次の補間多項式による，補間点 v における補間値（今後，単に条件(i,0),(i,1), i=1,....,m を満たす補間値と呼ぶ）を求める．

一般の場合を述べる前に，m=2 の場合を例にとり (1,0), (1,1), (2,0), (2,1) の 4 つの条件を満たす補間値を求める手順を説明する．

手順 1. 条件(1,0),(1,1)を満たす補間値
$P_{A1}(v) \equiv y_{1,1'} \equiv y_1 + y'_1(v - v_1)$を求める．
条件(2,0),(2,1)を満たす補間値
$P_{A3}(v) \equiv y_{2,2'} \equiv y_2 + y'_2(v - v_2)$を求める．

手順 2. 条件(1,0),(2,0)を満たす補間値
$P_{A2}(v) \equiv y_{1,2} \equiv y_1 + \dfrac{y_1 - y_2}{v_1 - v_2}(v - v_1)$を求める．

手順 3. 条件(1,0),(1,1),(2,0)を満たす補間値
$P_{A4}(v) \equiv y_{1,1',2} \equiv P_{A1}(v) + \dfrac{P_{A1}(v) - P_{A2}(v)}{v_1 - v_2}(v - v_1)$を求める．

条件(1,0),(2,0),(2,1)を満たす補間値
$P_{A5}(v) \equiv y_{1,2,2'} \equiv P_{A2}(v) + \dfrac{P_{A2}(v) - P_{A3}(v)}{v_1 - v_2}(v - v_1)$を求める．

手順 4. 条件(1,0),(1,1),(2,0),(2,1)を満たす補間値
$P_{A6}(v) \equiv y_{1,1',2,2'} \equiv P_{A4}(v) + \dfrac{P_{A4}(v) - P_{A5}(v)}{v_1 - v_2}(v - v_1)$を求める．

$P_{A6}(v)$ が目的とする補間値となる．
一般の場合は

$$y_{i,i'} \equiv y_i + y'_i(v - v_i) \quad (i = 1,\ldots,m) \tag{5.1}$$

$$y_{i,i+1} \equiv y_i + \frac{y_i - y_{i+1}}{v_i - v_{i+1}}(v - v_i) \quad (i = 1,\ldots,m) \tag{5.2}$$

の式をもとにして，m=2 の場合と同じ要領で計算する．それを図 AKHER-3 に示す．

すなわち左の列から右の列へ，下の行から上の行へ，$y_{1.1}'$，$y_{1.2}$，$y_{1.1}'_{.2}$，$y_{2.2}'$，$y_{1.2.2}'$，$y_{1.1}'_{.2.2}'$，$y_{2.3}$，$y_{2.2}'_{.3}$，$y_{1.2.2}'_{.3}$，……の順にエイトケンの反復補間法の基本定理に基づいて計算していき補間値 $y_{1.1}'$，...，$_{m.m}'$ を求める．

$$
\begin{array}{ccccccc}
y_{1.1}' & y_{1.1}'_{.2} & y_{1.1}'_{.2.2}' & y_{1.1}'_{.2\,2}'_{.3} & y_{1.1}'_{.2.2}'_{.3.3}' & \cdot & y_{1.1}'_{.2.2}'_{....m.m}' \\
y_{1.2} & y_{1.2.2}' & y_{1.2.2}'_{.3} & y_{1.2.2}'_{.3.3}' & \cdot & \cdot & \\
y_{2.2}' & y_{2.2}'_{.3} & y_{2.2}'_{.3.3}' & \cdot & \cdot & & \\
y_{2.3} & y_{2.3.3}' & \cdot & \cdot & & & \\
y_{3.3}' & \cdot & \cdot & & & & \\
\cdot & \cdot & & & & & \\
y_{m.m}' & & & & & &
\end{array}
$$

図 AKHER-3 (一般の場合)

なお，詳細については，参考文献 [47] を参照すること．

E11-11-0101 AKLAG, DAKLAG

エイトケン･ラグランジュ補間
CALL AKLAG (X,Y, N, V, M, EPS, F, VW, ICON)

(1) 機能

離散点 x_1 , x_2 , ... , x_n ($x_1 < x_2 < ... < x_n$) に対して関数値 $y_i = f(x_i)$, i = 1, ..., n が与えられたとき，$x = v$ における補間値をエイトケン･ラグランジュ補間法により求める．

$n \geq 1$ であること．

(2) パラメタ

X……**入力**．離散点 x_i.
大きさ n の 1 次元配列.

Y……**入力**．関数値 y_i.
大きさ n の 1 次元配列.

N……**入力**．離散点の個数 n.

V……**入力**．補間したい点の x 座標 v.

M……**入力**．補間に使う離散点の個数の上限 ($\leq n$).
出力．実際に使用した離散点の個数.
(使用上の注意参照).

EPS……**入力**．しきい値.
出力．補間値の絶対誤差.
(使用上の注意参照).

F……**出力**．補間値

VW……**作業領域**．大きさ 4n の 1 次元配列.

ICON **出力**．コンディションコード.
表 AKLAG-1 参照.

表 AKLAG-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	v が離散点 x_i と一致した.	F = y_i とする.
30000	n < 1, M = 0, $x_i \geq x_{i+1}$ のいずれかであった.	F = 0.0 として処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AFMAX, MGSSL

② FORTRAN 基本関数 ... ABS,IABS

b. 注意

① 収束判定条件について
まず補間多項式の次数と，数値的なふるまいを考える．ここで x=v の近傍での j 個の離散点を用いて補間した補間値を Z_j で表す．(離散点のとり方は，x=v に近い順とする．)

次に間差 D_j を次のように定義する．

$$D_j \equiv Z_j - Z_{j-1} \quad j = 2,\ldots,m$$

ここで m は補間に使う離散点の個数の上限である．一般に，補間多項式の次数を上げていくと，$|D_j|$は図 AKLAG-1 のようなふるまいを示す．

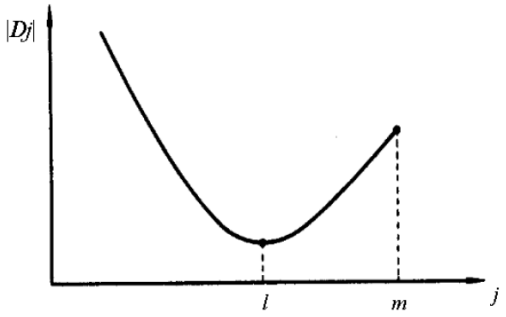


図 AKLAG-1

図 AKLAG-1 で l は近似多項式の打切り誤差と計算誤差が互いに同程度になる事を示しており，一般に Z_l が数値的には最適の補間値とみなすことができる．

② EPS の与え方
次の条件を考慮する．
収束判定は①に従って行われるが，D_j は近似関数により種々のふるまいを示す．図 AKLAG-2 で示されるように振動する場合もある．

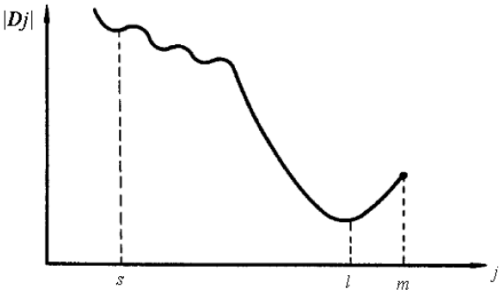


図 AKLAG-2

この場合，補間値として Z_s を採用するのは好ましくなく Z_l を採用すべきである．この考えに基いて出力すべき補間値を次のように決定する．
$D_2, D_3, \cdots, D_m$ と計算していく過程で，
(a) $|D_j|>|EPS|, j=2, 3, \cdots, m$ であれば，

$$|D_l| = \min_j(|D_j|) \qquad (3.1)$$

となる l を求め Z_l, l, $|D_l|$ をそれぞれパラメタ F，M，EPS に出力し，

(b) $|D_j| \le |\mathrm{EPS}|$となり始めたならば，その時点からは，

$$|D_l| \le |D_{l+1}| \qquad (3.2)$$

となる l を求め Z_l，l，$|D_l|$を出力する．
もし (3.2) がいつまでも成立たなければ $l = m$ として Z_m, m, $|D_m|$ を出力する．
又 EPS = 0.0 と入力すると $|D_j|$ が最小となる時点の Z_j を補間値として出力することになる．

③ M の与え方

(a) もし，真の関数が，$x=v$ の近傍で高々k 次の多項式で近似できることが分かっていれば，多項式補間法においても，高々k 次の多項式で行うのが自然である．もし，このような状況にあるなら，パラメタ M に(k+1)を入れる．

(b) 上記(a)の事実がわからないときは，パラメタ M に，パラメタ N の値と等しい値を入れる．

(c) もし，ある定まった個数の離散点を，収束判定することなく無条件に使って補間値を求めたいときには，パラメタ M に，その個数に-1 を乗じた値を入れる．

c. 使用例

入力パラメタの値を読込み，補間値 F を求める．
$n \le 30$ の場合．

```
C     **EXAMPLE**
      DIMENSION X(30),Y(30),VW(120)
      READ(5,500) N,(X(I),Y(I),I=1,N)
      WRITE(6,600) (I,X(I),Y(I),I=1,N)
   10 READ(5,510) M,EPS,V
      IF(M.GT.30) STOP
      CALL AKLAG(X,Y,N,V,M,EPS,F,VW,ICON)
      WRITE(6,610) ICON,M,V,F
      IF(ICON.EQ.30000) STOP
      GO TO 10
  500 FORMAT(I2/(2F10.0))
  510 FORMAT(I2,2F10.0)
  600 FORMAT(23X,'ARGUMENT VALUES',15X,
     * 'FUNCTION VALUES'/(15X,I2,5X,
     * E15.7,15X,E15.7))
  610 FORMAT(20X,'ICON =',I5,10X,
     * 'M =',I2/20X,'V         =',E15.7/
     * 20X,'COMPUTED VALUES =',E15.7)
      END
```

(4) 手法概要

離散点 x_i を v に近い順に並べかえて $v_1, v_2, ..., v_n$ とし，それに対応して $y_i = f(v_i)$とする．

一般に部分標本点 $(v_i, y_i = f(v_i);\ i = 1, ..., m)$ を補間に使う場合を説明する．離散点(v_1, y_1), (v_2, y_2), ..., (v_i, y_i), (v_j, y_j) を通る i 次のラグランジュ補間多項式による補間値を単に $y_{1,2...,i,j}$ で表すとする．

(ここで $j > i$).

エイトケン･ラグランジュ補間法は

$$y_{1,2,\ldots,i,j} = \frac{1}{v_j - v_i}\begin{vmatrix} y_{1,2,\ldots,i} & v_i - v \\ y_{1,2,\ldots,i-1,j} & v_j - v \end{vmatrix}$$

$$= y_{1,2,\ldots,i} + \frac{y_{1,2,\ldots,i} - y_{1,2,\ldots,i-1,j}}{v_j - v_i}(v_i - v)$$

に基づき，図 AKLAG-3 において上の行から下の行へ，左の列から右の列へ，すなわち $y_{1.2}$, $y_{1.3}$, $y_{1.2.3}$, $y_{1.4}$, $y_{1.2.4}$, ...の順に計算していく方法である．

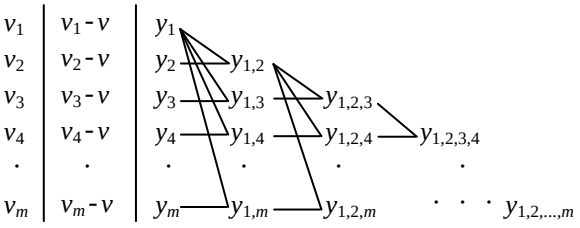


図 AKLAG-3

対角線上に表れた $y_{1,\,2,\,.....,\,k}$ は離散点 $(v_i, y_i, i = 1, ..., k)$ を通るラグランジュ補間式による補間値にほかならなく，$y_{1,\,2,\,.....,\,m}$ が目的とする補間値である．

なお，詳細については，参考文献 [46] pp.57-59 を参照すること．

E11-42-0101 AKMID, DAKMID

2 次元準エルミート補間式による補間
CALL AKMID (X, NX, Y, NY, FXY, K, ISW, VX, IX, VY, IY, F, VW, ICON)

(1) 機能

格子点(x_i, y_j), $i = 1, 2, ..., n_x$, $j = 1, 2, ..., n_y$($x_1 < x_2 < ... < x_{nx}$, $y_1 < y_2 < ... < y_{ny}$)に対して，関数値 $f_{ij} = f(x_i, y_j)$ が与えられたとき，点 $P(v_x, v_y)$ における補間値を，双 3 次の区分的 2 次元準エルミート補間式により求める．(図 AKMID-1 参照).

$x_1 \le v_x \le x_{nx}$, $y_1 \le v_y \le y_{ny}$ 及び $n_x \ge 3$, $n_y \ge 3$ であること.

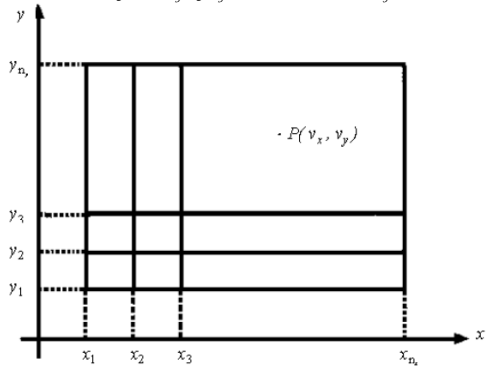


図 AKMID-1 領域 $R=\{(x,y) \mid x_1 \le x_{nx}, y_1 \le y \le y_{ny}\}$内の P

(2) パラメタ

X…… **入力**．x 方向の離散点 x_i.
大きさ n_x の 1 次元配列.

NX…… **入力**． x_i の個数 n_x .

Y…… **入力**．y 方向の離散点 y_j.
大きさ n_y の 1 次元配列.

NY…… **入力**．y_j の個数 n_y.

FXY…… **入力**．関数値 f_{ij}.
大きさ FXY (K, NY)なる 2 次元配列． FXY (I, J) には f_{ij} を入力する.

K…… **入力**．2 次元配列 FXY の整合寸法 (K ≥ NX).

ISW …… **入力**．与えられた入力データ (x_i, y_j, f_{ij}) に対して初めて本サブルーチンを呼び出すときに,

ISW = 0

と入力する (INTEGER *4 であること)．同一の入力データに対して，本サブルーチンを繰り返し呼び出すことにより，一連の補間値を求めるときは，2 回目以降，ISW を変更してはならない.

出力． $x_i \le v_x < x_{i+1}$, $y_j \le v_y < y_{j+1}$ を満たす (i,j) に関する情報.

利用者は，新たに与えられたデータ (x_i, y_j, f_{ij})に対して補間を始めるときは, 再び ISW =0 とする必要がある.

VX…… **入力**． 点 $P(v_x, v_y)$ の x 座標.

IX…… **入力**． $x_i \le v_x < x_{i+1}$ を満たす i.
$v_x = x_{nx}$ のときは IX = $n_x - 1$ とすること.
出力． $x_i \le v_x < x_{i+1}$ を満たす i.
(使用上の注意 3 参照)

VY…… **入力**． 点 $P(v_x, v_y)$ の y 座標.

IY…… **入力**． $y_j \le v_y < y_{j+1}$ を満たす j.
$v_y = y_{ny}$ のときは IY = $n_y - 1$ とすること.
出力． $y_j \le v_y < y_{j+1}$ を満たす j.
(使用上の注意 3 参照)

F…… **出力**．補間値.

VW…… **作業領域**．大きさ 50 の 1 次元配列.
同一の入力データ (x_i, y_j, f_{ij}) に対して繰り返し本サブルーチンを呼び出すときには，その間，VW の内容を変更してはならない.

ICON…… **出力**．コンディションコード.
表 AKMID-1 参照.

表 AKMID-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	X(IX)≤VX<X(IX+1) 又は Y(IY)≤VY<Y(IY+1) が満たされていない.	サブルーチン内で左記の IX，又は IY をさがして処理を続ける.
30000	次のいずれかであった. 1 X(I) ≥ X (I+1) なる X (I) がある. 2 Y(J) ≥ Y (J+1) なる Y(J) がある. 3 NX<3 又は NY < 3 4 K< NX 5 VX < X(1) 又は VX > X(NX) 6 VY < Y(1) 又は VY > Y(NY) 7 ISW の指定に誤りがある.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... IABS, ABS, MOD

b. 注意

① 本サブルーチンで用いる補間式は，領域 $R=\{(x,y) \mid x_1 \le x \le x_{nx}, y_1 \le y \le y_{ny}\}$で，$x$ 方向，y 方向各々独立に 1 階までの導関数は連続であるが，2 階以上の導関数は一般に不連続となる．一方で，この補間式は，不自然な屈曲点や面が現れないのが特徴であり，「曲面のあてはめ」を効率良く実行するものである.
2 変数関数の補間値，又は微分値，積分値をより高精度で得るためには，Spline 関数による補間サブルーチン（BIFD3 又は BIFD1）を用いた方がよい.

② 本サブルーチンにより，同一の入力データ (x_i, y_j, f_{ij}) のもとで，多数の補間値を求めたいときは，同一の格子内に属する点が連続するようにして本サブルーチンを呼び出すと効率がよい(使用例参照)．なお，このときパラメタ ISW，VW の内容を変更をしてはならない．

③ パラメタ IX，IY は X(IX) ≤ VX < X(IX+1)，Y(IY) ≤ VY < Y(IY+1) を満足していることが好ましい．もし満足していないときは，このような条件を満足するような IX，IY を探索して処理を続ける．

④ 本サブルーチンが ICON = 30000 としてパラメタエラーとする条件は，表 AKMID-1 に示したとおりであるが，その中で条件の① ～ ④については，ISW = 0 と指定されたとき（すなわち，与えられた入力データ (x_i, y_j, f_{ij}) に対して，初めて呼び出されたとき）だけチェックし，2 回目以降の呼び出し時には，チェックしていない．

c. 使用例

格子点 (x_i, y_j)，関数値 f_{ij}: i = 1, 2, ..., n_x, j = 1, 2, ..., n_y を入力し，次のような点 (v_{il}, v_{jk}) における補間値を求める．$n_x \le 121$, $n_y \le 101$ の場合．

$v_{il} = x_i + (x_{i+1} - x_i) \times (l/4)$
$i = 1, 2, ..., n_x - 1,\ l = 0, 1, 2, 3$
$v_{jk} = y_j + (y_{j+1} - y_j) \times (k/2)$
$j = 1, 2, ..., n_y - 1,\ k = 0, 1$

```
C      **EXAMPLE**
       DIMENSION  X(121),Y(101),FXY(121,101),
      *          VW(50),XV(4),YV(2),FV(4,2)
       READ(5,500) NX,NY
       READ(5,510) (X(I),I=1,NX)
       READ(5,510) (Y(J),J=1,NY)
       READ(5,510) ((FXY(I,J),I=1,NX),
      *             J=1,NY)
       WRITE(6,600) NX,NY
       WRITE(6,610) (I,X(I),I=1,NX)
       WRITE(6,620) (J,Y(J),J=1,NY)
       WRITE(6,630) ((I,J,FXY(I,J),
      *              I=1,NX),J=1,NY)
       ISW=0
       NX1=NX-1
       NY1=NY-1
       DO 40 I=1,NX1
       HX=(X(I+1)-X(I))*0.25
       DO 10 IV=1,4
    10 XV(IV)=X(I)+HX*FLOAT(IV-1)
       DO 40 J=1,NY1
       HY=(Y(J+1)-Y(J))*0.5
       DO 20 JV=1,2
    20 YV(JV)=Y(J)+HY*FLOAT(JV-1)
       DO 30 IV=1,4
       DO 30 JV=1,2
    30 CALL AKMID(X,NX,Y,NY,FXY,121,ISW,
      *           XV(IV),I,YV(JV),J,
      *           FV(IV,JV),VW,ICON)
    40 WRITE(6,640) I,J,((IV,JV,FV(IV,JV),
      *             IV=1,4),JV=1,2)
       STOP
   500 FORMAT(2I6)
   510 FORMAT(6F12.0)
   600 FORMAT('1'//10X,'INPUT DATA',3X,
      * 'NX=',I3,3X,'NY=',I3/)
   610 FORMAT('0','X'/(6X,6(I6,E15.7)))
   620 FORMAT('0','Y'/(6X,6(I6,E15.7)))
   630 FORMAT('0','FXY'/(6X,5('(',2I4,
      * E15.7,')')))
   640 FORMAT('0','APP.VALUE',2I5/(6X,
      * 5('(',2I4,E15.7,')')))
       END
```

(4) 手法概要

本サブルーチンは，以下に述べる双 3 次の 2 次元準エルミート補間式に基づいて，補間値を求めるものである．ここでいう 2 次元準エルミート補間式は，サブルーチン AKMIN が求める 1 次元の準エルミート補間式をそのまま 2 次元へ拡張したものである．

a. 双 3 次の 2 次元準エルミート補間式

ここで述べる補間式 $S(x, y)$ は領域 $R = \{(x,y) | x_1 \le x \le x_{nx},\ y_1 \le y \le y_{ny}\}$ 上で定義され，次の条件を満たす関数である．

(a) $S(x, y)$ は各部分領域

$$R_{i,j} = \{(x,y) | x_i \le x < x_{i+1},\ y_j \le y < y_{j+1}\}$$

で高々，双 3 次の多項式である．

(b) $S(x,y) \in C^{1,1}[R]$，すなわち，R 上で

$$S^{(\alpha,\beta)}(x,y) = \frac{\partial^{\alpha+\beta}}{\partial x^{\alpha} \partial y^{\beta}} S(x,y) \quad ,$$
$$\alpha = 0,1, \quad \beta = 0,1$$

が存在して，しかも連続である．

(c) $S^{(\alpha,\beta)}(x_i, y_j) = f^{(\alpha,\beta)}(x_i, y_j)$,
$\alpha = 0,1,\ \beta = 0,1,\ i = 1,2,\ldots, n_x\ ,\ j = 1,2,\ldots, n_y$

このような関数 $S(x, y)$ は一意的に存在し，部分領域 $R_{i,j}$ 上では (4.1) のように表現できることが証明されている．

$$\begin{aligned} S(x,y) &= S_{i,j}(s,t) \\ &= \sum_{\alpha=0}^{1}\sum_{\beta=0}^{1} a_i^{\alpha} b_j^{\beta} \{ f_{ij}^{(\alpha,\beta)} p_{\alpha}(s) q_{\beta}(t) \\ &\quad + f_{i+1,j}^{(\alpha,\beta)} q_{\alpha}(s) p_{\beta}(t) + f_{i,j+1}^{(\alpha,\beta)} p_{\alpha}(s) q_{\beta}(t) \\ &\quad + f_{i+1,j+1}^{(\alpha,\beta)} q_{\alpha}(s) q_{\beta}(t) \} \end{aligned} \qquad (4.1)$$

ここで

$$a_i = x_{i+1} - x_i \quad , \quad b_j = y_{j+1} - y_j$$

$$s = \frac{x - x_i}{a_i} \quad , \quad t = \frac{y - y_j}{b_j} \quad , \quad 0 \le s \quad , \quad t < 1$$

$$p_0(t) = 1 - 3t^2 + 2t^3$$
$$q_0(t) = 3t^2 - 2t^3$$
$$p_1(t) = t(t-1)^2$$
$$q_1(t) = t^2(t-1)$$
$$f_{ij}^{(\alpha,\beta)} = f^{(\alpha,\beta)}(x_i, y_j) = \frac{\partial^{\alpha+\beta}}{\partial x^\alpha \partial y^\beta} f(x_i, y_j)$$

(4.1) では，四つの格子点 (x_i, y_j), (x_{i+1}, y_j), (x_i, y_{j+1}), (x_{i+1}, y_{j+1}) における関数値と微係数を必要とする．したがって微係数を何らかの方法で求める（近似する）ことができれば，(4.1) により $R_{i,j}$ 上の補間値を求めることができる．このように近似された微係数を使って得られる (4.1) の $S(x, y)$ を，双 3 次の区分的 2 次元準エルミート補間式という．

b. 格子点における微係数 $f_{ij}^{(1,0)}$, $f_{ij}^{(0,1)}$, $f_{ij}^{(1,1)}$ の決定

本サブルーチンでは，この微係数を求める方法として秋間により提唱された幾何学的方法を採用している．
これは，1 次元の準エルミート補間式 (サブルーチン AKMIN) で用いた方法の 2 次元への応用である．以下その概要を示す．
前準備として，次の量を定義する．

$$a_i = x_{i+1} - x_i \ , \ b_j = y_{j+1} - y_j$$
$$c_{ij} = (f_{i+1,j} - f_{ij}) / a_i$$
$$d_{ij} = (f_{i,j+1} - f_{ij}) / b_j$$
$$e_{ij} = (c_{i,j+1} - c_{ij}) / b_j = (d_{i+1,j} - d_{ij}) / a_i \qquad (4.2)$$

いま，簡単のため，x 方向に続く 5 点を x_1, x_2, x_3, x_4, x_5 とし，y 方向に続く 5 点を y_1, y_2, y_3, y_4, y_5 とする．このとき点 (x_3, y_3) における偏微係数を得ることを考える．図 AKMID-2 に，必要な c, d, e を示す．

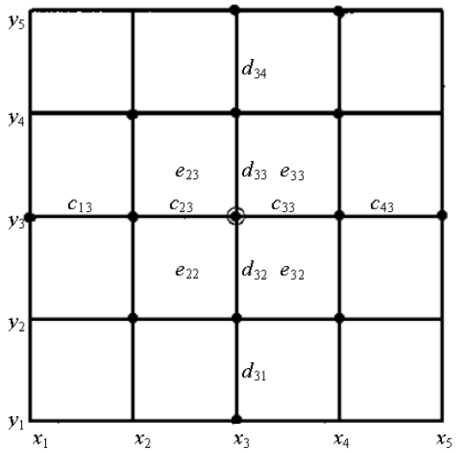


図 AKMID-2 点 (x_3, y_3) における微係数の決定

1 次元準エルミート補間式と同様な仮定をおいて，x 方向，y 方向の 1 階偏微係数を次式によって求める．

$$f_{33}^{(1,0)} = \left(w_{x2}c_{23} + w_{x3}c_{33}\right) \Big/ \left(w_{x2} + w_{x3}\right)$$
$$f_{33}^{(0,1)} = \left(w_{y2}d_{32} + w_{y3}d_{33}\right) \Big/ \left(w_{y2} + w_{y3}\right) \qquad (4.3)$$

ここで

$$w_{x2} = |c_{43} - c_{33}|, \quad w_{x3} = |c_{23} - c_{13}|$$
$$w_{y2} = |d_{34} - d_{33}|, \quad w_{y3} = |d_{32} - d_{31}|$$

すなわち，$f_{33}^{(1,0)}$ は c についての，$f_{33}^{(0,1)}$ は d についての重みつき平均として表される．$f_{33}^{(1,1)}$ については，これと同様な仮定をおき，さらに x 方向，y 方向について対称となるような e についての重みつき平均をとって，次のように決定する．

$$f_{33}^{(1,1)} = \left\{w_{x2}\left(w_{y2}e_{22} + w_{y3}e_{23}\right) + w_{x3}\left(w_{y2}e_{32} + w_{y2}e_{33}\right)\right\} \Big/ \left\{\left(w_{x2} + w_{x3}\right)\left(w_{y2} + w_{y3}\right)\right\} \qquad (4.4)$$

c. 境界における微係数の決定

境界における偏微係数 $f_{ij}^{(0,1)}$, $f_{ij}^{(1,0)}$, $f_{ij}^{(1.1)}$; i = 1, 2, $n_x - 1$, n_x, j = 1, 2, $n_y - 1$, n_y については，1 次元の準エルミート補間式の場合と同様な仮定をおいて，領域外の c_{ij}, d_{ij}, e_{ij} を求め，その後 (4.3) , (4.4) を適用して求める．ここでは $i = j = 1$ の場合を例にとって説明する．図 AKMID-3 にその様子を示す．図中の〇印の点は，1 次元の準エルミート補間式の場合と同様なやり方で仮想した点である．

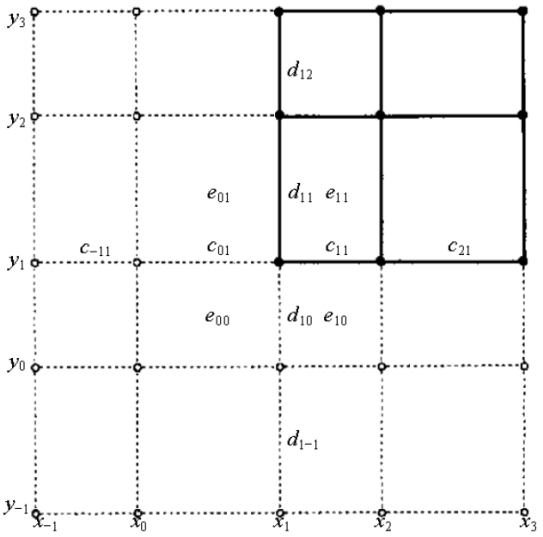


図 AKMID-3 境界における微係数の決定

このとき領域外の c_{ij}, d_{ij}, e_{ij} は次のようにして求まる.

$$
\begin{aligned}
c_{-11} &= 3c_{11} - 2c_{21} \\
c_{01} &= 2c_{11} - c_{21} \\
d_{1-1} &= 3d_{11} - 2d_{12} \\
d_{10} &= 2d_{11} - d_{12} \\
e_{01} &= 2e_{11} - e_{21} \\
e_{00} &= 2e_{01} - e_{02} \quad (e_{02} = 2e_{12} - e_{22}) \\
e_{10} &= 2e_{11} - e_{12}
\end{aligned}
\tag{4.5}
$$

以上のようにして，必要な c, d, e を求めたあと (4.3)，(4.4) を適用すれば，点 (x_1, y_1) における偏微係数が得られる.

d. 補間値の計算

点 (v_x, v_y) における補間値は，その点が属する格子領域の番号 (i_x, i_y) により，(4.1) の $S_{ix,iy}(s, t)$ を形成し，それを評価することにより計算される. このとき本サブルーチンは，必要な微係数の中で，前回までに呼び出されたときに，既に計算されているものがあればそれを利用し，そうでないものについてだけ新たに計算する (これらは次回以降の呼出しに備えて，作業領域に蓄えられる).

なお，詳細については，参考文献 [54] を参照すること.

E12-21-0201 AKMIN, DAKMIN

準エルミート補間式
CALL AKMIN (X, Y, N, C, D, E, ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$に対して，関数値 $y_i = f(x)$, $i = 1, ..., n$ が与えられたとき，準エルミート補間法を用いて，$f(x)$ に対する (1.1) で表される 3 次の区分的な補間式 $S(x)$を求める．ただし，$n \geq 3$ であること．

$$S(x) = y_i + c_i(x - x_i) + d_i(x - x_i)^2 + e_i(x - x_i)^3 \quad (1.1)$$
$$x_i \leq x \leq x_{i+1}, \quad i = 1,2,\ldots,n-1$$

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.

Y …… **入力**．関数値 y_i.
大きさ n の 1 次元配列.

N …… **入力**．離散点の個数 n.

C …… **出力**．(1.1) 式における係数 c_i.
大きさ $n-1$ の 1 次元配列.

D …… **出力**．(1.1) 式における係数 d_i.
大きさ $n-1$ の 1 次元配列.

E …… **出力**．(1.1) 式における係数 e_i.
大きさ $n-1$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 AKMIN-1 参照.

表 AKMIN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	$n < 3$, $x_i \geq x_{i+1}$ のいずれかであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II .... MGSSL

② FORTRAN 基本関数 ... ABS

b. 注意

① 本サブルーチンにより得られる補間式は，不自然な屈曲点 (wiggles) が現れず，人間が手で描いた曲線に最も近いという特徴を持っている．しかし，この補間式は区間 (x_1, x_n) で 1 階までの導関数は連続であるが，2 階以上は一般に不連続である.

② もし，$f(x)$ が 2 次元多項式であり，かつ，x_i, $i = 1, ..., n$ が等間隔で与えられたときは，得られる補間式は，計算誤差がないものとすると，$f(x)$ そのものである.

③ もし，区間外 $(x < x_1$ 又は $x > x_n)$ に対して利用したいときは，精度は良くないが (1.1) における $i = 1$ 又は $i = n - 1$ の式を用いればよい.

c. 使用例

離散点の個数 n，離散点 x_i，関数値 y_i, $i = 1, ..., n$ を入力し，準エルミート補間式を求め，それによりある区間 $[x_k, x_{k+1}]$ 内のある点 $x = v$ における補間値を求める．$n \leq 10$ の場合.

```
C      **EXAMPLE**
       DIMENSION X(10),Y(10),C(9),D(9),E(9)
       READ(5,500) N
       READ(5,510) (X(I),Y(I),I=1,N)
       CALL AKMIN(X,Y,N,C,D,E,ICON)
       WRITE(6,600) ICON
       IF(ICON.NE.0) STOP
       READ(5,500) K,V
       XX=V-X(K)
       YY=Y(K)+(C(K)+(D(K)+E(K)*XX)*XX)*XX
       N1=N-1
       WRITE(6,610)
       WRITE(6,620) (I,C(I),I,D(I),I,E(I)
      *              ,I=1,N1)
       WRITE(6,630) K,V,YY
       STOP
  500 FORMAT(I5,F10.0)
  510 FORMAT(2F10.0)
  600 FORMAT('0',10X,
     * 'RESULTANT CONDITION CODE=',I5//)
  610 FORMAT('0',10X,
     * 'RESULTANT COEFFICIENTS'//)
  620 FORMAT(' ',15X,'C(',I2,')=',E15.7,
     * 5X,'D(',I2,')=',E15.7,
     * 5X,'E(',I2,')=',E15.7)
  630 FORMAT('0',10X,2('*'),'RANGE',
     * 2('*'),5X,2('*'),'DESIRED POINT',
     * 2('*'),5X,2('*'),'INTERPOLATED '
     * 'VALUE',2('*')//13X,I2,2X,
     * 2(8X,E15.7))
       END
```

(4) 手法概要

離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ....< x_n)$ に対して，関数値 $y_i = f(x_i)$, $i = 1, ..., n$ が与えられたとき，(1.1) で表される補間式を求めることを考える．(1.1) は区間 $[x_i, x_{i+1}]$ ごとに異なった高々3 次の多項式を表している.

ここで，(1.1) で表した $S(x)$ を区分的に (4.1) で表す.

$$\left.\begin{aligned} S(x) &= S_i(x) \\ &= y_i + c_i(x - x_i) + d_i(x - x_i)^2 + e_i(x - x_i)^3 \\ x_i \leq x \leq x_{i+1}, &\quad i = 1,\ldots,n-1 \end{aligned}\right\} \quad (4.1)$$

各々の $S_i(x)$ は次の手順で定める.

① 2 点 x_i , x_{i+1} における 1 階微係数を近似する (それらを t_i , t_{i+1} とする).

② 次の四つの条件により $S_i(x)$ を決定する.

$$\left.\begin{aligned} S_i'(x_i) &= t_i \\ S_i'(x_{i+1}) &= t_{i+1} \\ S_i(x_i) &= y_i \\ S_i(x_{i+1}) &= y_{i+1} \end{aligned}\right\} \quad (4.2)$$

このような手順で求まる補間式を準エルミート補間式という.
本サブルーチンは，①の 1 階微係数の近似法に特徴があり，それは，幾何学的に定める方法である．その意味で，以下補間式を「曲線」と呼び，1 階微係数を「曲線の傾き」と呼ぶことにする.

a. 各離散点における曲線の傾きの決定
各々の離散点における曲線の傾きは，その点と両側 2 点ずつの計 5 点の座標により局所的に決定される.
今，続く 5 点を 1，2，3，4，5 として，点 3 における曲線の傾きを得ることを考える.
(図 AKMIN-1 参照).

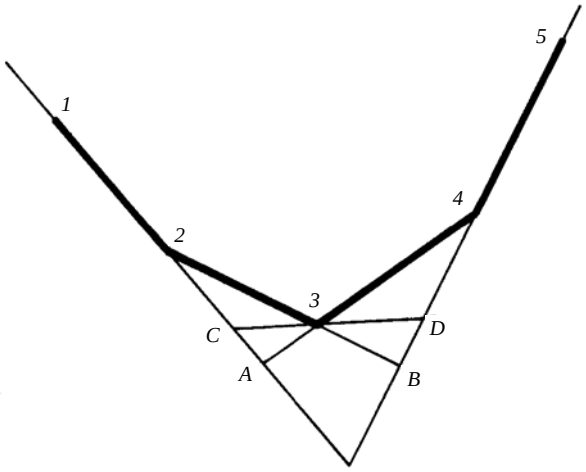


図 AKMIN-1

図 AKMIN-1 において線分 $\overline{1\ 2}$ と，線分 $\overline{3\ 4}$ の延長線の交点を A とし，線分 $\overline{2\ 3}$，線分 $\overline{4\ 5}$ の延長線の交点を B とする．また，点 3 における求めようとしている曲線の接線と線分 $\overline{2\ \boldsymbol{A}}$ と線分 $\overline{4\ \boldsymbol{B}}$ の交点をそれぞれ C，D とする.
ここで，線分 $\overline{1\ 2}$，$\overline{2\ 3}$，$\overline{3\ 4}$，$\overline{4\ 5}$ の傾きをそれぞれ m_1，m_2，m_3，m_4 とする．点 3 における曲線の傾きを t とすると，t は m_1 が m_2 に近づくとき m_2 に近づくように，あるいは，m_4 が m_3 に近づけば m_3 に近づくように定める．この条件を満足するための一つの十分条件は(4.3)である.

$$\left|\frac{\overline{2C}}{\overline{CA}}\right| = \left|\frac{\overline{4D}}{\overline{DB}}\right| \tag{4.3}$$

(4.3) の関係より (4.4) が導かれる.

$$\left.\begin{aligned} t &= \frac{w_2 m_2 + w_3 m_3}{w_2 + w_3} \\ w_2 &= \left|(m_3 - m_1)(m_4 - m_3)\right|^{\frac{1}{2}} \\ w_3 &= \left|(m_2 - m_1)(m_4 - m_2)\right|^{\frac{1}{2}} \end{aligned}\right\} \tag{4.4}$$

(4.4) で得られる t は，以下の好ましい性質を持つ.

$$\left.\begin{aligned} &\text{(a) } m_1 = m_2, m_3 \neq m_4, m_2 \neq m_3 \text{ のとき } t = m_1 = m_2 \\ &\text{(b) } m_3 = m_4, m_1 \neq m_2, m_4 \neq m_2 \text{ のとき } t = m_3 = m_4 \\ &\text{(c) } m_2 = m_3, m_1 \neq m_4, m_3 \neq m_4 \text{ のとき } t = m_2 = m_3 \end{aligned}\right\} \tag{4.5}$$

しかし，一方，(4.4) は以下の (d)，(e) の欠点を持つ.

$$\left.\begin{aligned} &\text{(d) } w_2 = w_3 = 0 \text{ のとき } t \text{ は不定} \\ &\text{(e) } m_2 = m_4, m_3 \neq m_1, m_4 \neq m_3 \text{ のとき } t = m_2 \\ &\quad \text{あるいは} \\ &\quad m_1 = m_3, m_2 \neq m_4, m_3 \neq m_2 \text{ のとき } t = m_3 \end{aligned}\right\} \tag{4.6}$$

これらの欠点を回避するため，本サブルーチンは，(4.4) の代わりに (4.7) により t を決定している.
$m_1 \neq m_2$ 又は，$m_3 \neq m_4$ のとき,

$$\left.\begin{aligned} t &= \frac{\left|m_4 - m_3\right| m_2 + \left|m_2 - m_1\right| m_3}{\left|m_4 - m_3\right| + \left|m_2 - m_1\right|} \\ &m_1 = m_2 \text{かつ}，\ m_3 = m_4 \text{のとき}, \\ &t = 1/2\ (m_2 + m_3) \end{aligned}\right\} \tag{4.7}$$

ちなみに，この (4.7) は (4.5) の性質を満たしている.

b. 両端点における曲線の傾きの決定.
両端に位置する点 (x_1, y_1), (x_2, y_2), (x_{n-1}, y_{n-1}), (x_n, y_n) においては，以下に述べるように仮想的な離散点を採ることにより，その傾きを決定する.
いま，左端を例にとって述べる.
図 AKMIN-2 に示すような 5 点を設定し，(x_1, y_1) における曲線の傾き t_1 を得ることを考える.

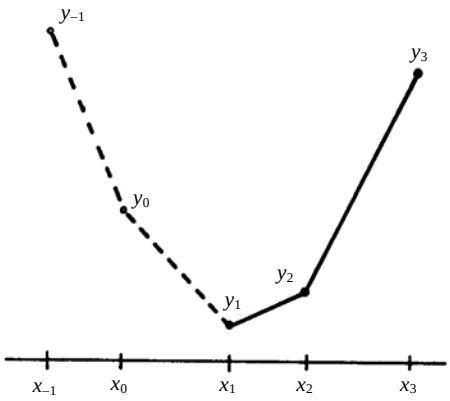


図 AKMIN-2

2 点 (x_{-1}, y_{-1}) , (x_0, y_0) は仮想した点であり，その x_{-1} , x_0 は (4.8) により決める．

$$x_3 - x_1 = x_2 - x_0 = x_1 - x_{-1} \tag{4.8}$$

y_{-1} , y_0 は 3 点 (x_1, x_1), (x_2, y_2), (x_3, y_3) を通る 2 次の多項式を x_{-1} , x_0 において評価して得られる値とする．このとき，5 点は (4.9) を満たす．

$$\left.\begin{aligned} \frac{y_3 - y_2}{x_3 - x_2} - \frac{y_2 - y_1}{x_2 - x_1} &= \frac{y_2 - y_1}{x_2 - x_1} - \frac{y_1 - y_0}{x_1 - x_0} \\ &= \frac{y_1 - y_0}{x_1 - x_0} - \frac{y_0 - y_{-1}}{x_0 - x_{-1}} \end{aligned}\right\} \tag{4.9}$$

ここで線分 $\overline{(x_{-1}, y_{-1}), (x_0, y_0)}$ の傾きを m_1 とし，同様に，右に続く線分の傾きをそれぞれ m_2, m_3, m_4, とすると (4.9) より，

$$m_2 = 2\,m_3 - m_4, \quad m_1 = 3\,m_3 - 2\,m_4 \tag{4.10}$$

となる．
(x_1, y_1) における傾き t_1 は，これらの m_1, m_2, m_3, m_4 を a. の方法に適用することにより求める．なお，点 (x_2, y_2) における t_2 を求める場合，
(x_0, y_0), (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, x_4) の 5 点を求める．
右端の (x_{n-1}, y_{n-1}), (x_n, y_n) における傾き t_{n-1}, t_n も上に述べた方法と同様に，(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}) を仮想して求める．

c. 曲線の決定
(4.1) に示した $S_i(x)$ の係数は，(4.2) の条件より決定すれば，(4.11) のようになる．

$$\left.\begin{aligned} c_i &= t_i \\ d_i &= \{3(y_{i+1} - y_i)/(x_{i+1} - x_i) - 2t_i - t_{i+1}\} \\ &\quad /(x_{i+1} - x_i) \\ e_i &= \{t_i + t_{i+1} - 2(y_{i+1} - y_i)/(x_{i+1} - x_i)\} \\ &\quad /(x_{i+1} - x_i)^2 \end{aligned}\right\} \tag{4.11}$$

なお，詳細については参考文献 [52] を参照すること．

A22-11-0202 ALU, DALU

実行列の LU 分解（クラウト法）
CALL ALU (A, K, N, EPSZ, IP, IS, VW, ICON)

(1) 機能

$n \times n$ の正則な実行列 $\boldsymbol{A}$ をクラウト法により LU 分解する.

$$\boldsymbol{PA} = \boldsymbol{LU} \qquad (1.1)$$

ただし，$\boldsymbol{P}$ は部分ピボッティングによる行の入換えを行う置換行列，L は下三角行列，U は単位上三角行列である.

$n \geq 1$ であること.

(2) パラメタ

A …… **入力**. 行列 ***A***.
出力. 行列 ***L*** と行列 ***U***.
図 ALU-1 参照.
A (K, N) なる 2 次元配列.

K …… **入力**. 配列 A の整合寸法 (≥N).

N …… **入力**. 行列 ***A*** の次数 *n*.

EPSZ …… **入力**. ピボットの相対零判定値 (≥ 0.0)
0.0 のときは標準値が採用される.
(使用上の注意①参照)

IP …… **出力**. 部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル. 大きさ *n* の 1 次元配列.
(使用上の注意②参照)

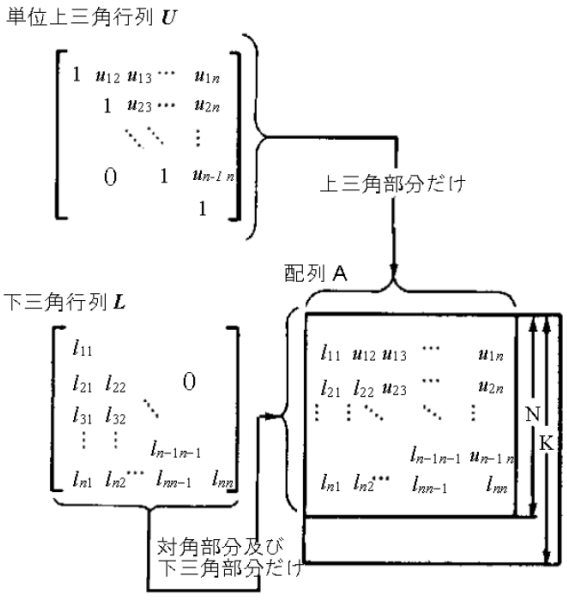


図 ALU-1 演算後の配列 A における *L* 及び *U* の各要素の格納方法

IS …… **出力**. 行列 ***A*** の行列式を求めるための情報.
演算後の配列 A の *n* 個の対角要素と IS の値を掛け合わせると行列式が得られる.

VW …… **作業領域**. 大きさ *n* の 1 次元配列.

ICON …… **出力**. コンディションコード.
表 ALU-1.

表 ALU-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	行列 A のある行の要素がすべて零であったか，又はピボットが相対的に零となった. 行列 A は非正則の可能性が強い.	処理を打ち切る.
30000	K<N, N<1 又は EPSZ<0.0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .... AMACH, MGSSL
② FORTRAN 基本関数 ... ABS

b. 注意
① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている. すなわち，クラウト法による LU 分解の過程でピボットの値が 10 進 *s* 桁以上の桁落ちを生じた場合にそのピボットを相対的に零とみなし，ICON = 20000 として処理を打ち切る. EPSZ の標準値は丸め誤差の単位を *u* としたとき，EPSZ = $16 \cdot u$ である.
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない.
② トランスポジションベクトルとは，部分ピボッティングを行う LU 分解

$$\boldsymbol{PA} = \boldsymbol{LU}$$

における置換行列 *P* に相当する.
本サブルーチンでは，部分ピボッティングに伴い，配列 A の内容を実際に交換している. すなわち，分解の J 段階目 ($J = 1, ..., n$) において第 I 行 ($I \geq J$) がピボット行として選択された場合には，配列 A の第 I 行と第 J 行の内容が交換される. そして，その履歴を示すために
IP (J) に I が格納される.
③ 本サブルーチンに続けて，サブルーチン LUX を呼び出すことにより，連立 1 次方程式を解くことができる. しかし，通常は，サブルーチン LAX を呼び出せば、一度に解が求められる.

c. 使用例

$n \times n$ の実行例を入力して，LU 分解する．

$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(100,100),VW(100),IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,A(I,J),J=1,N),
     *             I=1,N)
      CALL ALU(A,100,N,0.0,IP,IS,VW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) GOTO 10
      DET=IS
      DO 20 I=1,N
      DET=DET*A(I,I)
   20 CONTINUE
      WRITE(6,620) (I,IP(I),I=1,N)
      WRITE(6,630) ((I,J,A(I,J),J=1,N),
     *             I=1,N)
      WRITE(6,640) DET
      GOTO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT(///10X,'** INPUT MATRIX **'/
     * 12X,'ORDER=',I5//(10X,4('(',I3,',',
     * I3,')',E16.8)))
  610 FORMAT('0',10X,'CONDITION CODE =',
     * I5)
  620 FORMAT('0',10X,
     * 'TRANSPOSITION VECTOR'/
     * (10X,10('(',I3,')',I5)))
  630 FORMAT('0',10X,'OUTPUT MATRICES'/
     * (10X,4('(',I3,',',I3,')',E16.8)))
  640 FORMAT('0',10X,'DETERMINANT OF THE ',
     * 'MATRIX =',E16.8)
      END
```

(4) 手法概要

a. クラウト法

$n \times n$ の正則な実行列 $\boldsymbol{A}$ は通常，部分ピボッティングによる行の入換えを行って，下三角行列 $\boldsymbol{L}$ と単位上三角行列 $\boldsymbol{U}$ の積に分解することができる．

$$\boldsymbol{PA} = \boldsymbol{LU} \tag{4.1}$$

ただし，$\boldsymbol{P}$ は部分ピボッティングによる行の入換えを行う置換行列である．この $\boldsymbol{L}$ と $\boldsymbol{U}$ の要素を逐次求める方法の一つがクラウト法である．本サブルーチンでは，次の等式により，$\boldsymbol{L}$ の j 番目の列，$\boldsymbol{U}$ の j 番目の (j = 1, ..., n) の順に値を求める．

$$u_{ij} = \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \right) \Big/ l_{ii}, \quad i = 1,..., j-1 \tag{4.2}$$

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj}, \quad i = j,...,n \tag{4.3}$$

ただし，$\boldsymbol{A} = (a_{ij})$, $\boldsymbol{L} = (l_{ij})$ and $\boldsymbol{U} = (u_{ij})$ である．実際には部分ピボッティングによる行の入換えを行う．

クラウト法はガウス消去法の変形であって，両者は同じ計算を行うので計算量は等しいが，計算順序は異なっている．クラウト法では，$\boldsymbol{L}$ と $\boldsymbol{U}$ の要素を求めるのに，(4.2), (4.3) なる式で一度に計算するが，このときに積和計算を精度を上げて計算することにより，丸め誤差の影響を少なくしている．

b. 部分ピボッティング

例えば，行列 $\boldsymbol{A}$ が

$$\boldsymbol{A} = \begin{bmatrix} 0.0 & 1.0 \\ 1.0 & 0.0 \end{bmatrix}$$

で与えられたとき，これは数値的に極めて安定であるにもかかわらず，このままではもはや LU 分解は不可能である．また行列が数値的に安定であっても，そのまま単純に LU 分解したのでは大きな誤差が生じる場合もある．本サブルーチンではこのような事態を回避するために，行を均衡化 (low equilibrated) した部分ピボッティングを行っている．

なお，詳細については，参考文献 **[1], [3]** 及び **[4]** を参照すること．

G23-11-0301 AQC8, DAQC8

1 次元有限区間積分 (関数入力，クレンショー･カーチス型積分法)
CALL AQC8 (A, B, FUN, EPSA, EPSR, NMIN, NMAX, S, ERR, N, ICON)

(1) 機能

関数 $f(x)$ 及び a, b, ε_a, ε_r が与えられたとき，

$$\left|S-\int_a^b f(x)dx\right| \le \max\left(\varepsilon_a, \varepsilon_r \cdot \left|\int_a^b f(x)dx\right|\right) \qquad (1.1)$$

を満たす積分の近似値 S を，等差数列的に分点を増すクレンショー・カーチス型積分法により求める．

(2) パラメタ

A …… **入力**．積分区間の下限 a.
B …… **入力**．積分区間の上限 b ($b \le a$ でも何).
FUN …… **入力**．被積分関数 $f(x)$ を計算する関数副プログラム名 (使用例参照).
EPSA …… **入力**．積分の近似値 S に対する絶対誤差の上限 ε_a (≥ 0.0).
EPSR …… **入力**．積分の近似値 S に対する相対誤差の上限 ε_r (≥ 0.0).
NMIN …… **入力**．被積分関数の計算回数の下限 (≥ 0)，標準値としては 15 が適当.
NMAX …… **入力**．被積分関数の計算回数の上限，(NMAX ≥ NMIN).
標準値としては 511 が適当．(これよりも大きく指定した場合は 511 と見なす).
(使用上の注意④参照)
S …… **出力**．積分の近似値 (使用上の注意⑤参照).
ERR …… **出力**．積分の近似値 S の絶対誤差の推定値.
N …… **出力**．被積分関数の計算回数.
ICON …… **出力**．コンディションコード.
表 AQC8-1 参照.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, MGSSL
② FORTRAN 基本関数 ... ABS, AMAX1, FLOAT, MAX0, MIN0, SQRT

b. 注意
① パラメタ FUN に対応する関数副プログラムは，積分変数だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムで，その関数名を EXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図る．(使用例参照)

表 AQC8-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	丸め誤差のために要求精度が得られなかった.	S には，得られた近似値を出力する．精度は達成可能な限度まで達している.
20000	被積分関数の計算回数がその上限に達しても要求精度が得られなかった.	処理を打ち切る．S には，そのときまでに得られている近似値を出力するが精度は保証できない.
30000	次のいずれかであった. ① EPSA < 0.0 ② EPSR < 0.0 ③ NMIN < 0 ④ NMAX < NMIN	処理を打ち切る.

② 本サブルーチンを多数回呼ぶとき，最初だけ 511 個の定数（積分公式の分点と重みの表）を計算し，2 回目以降は省略する．したがって，2 回目以降は，計算量がその分だけ少ない.

③ 本サブルーチンは，被積分関数 $f(x)$ が振動型の関数の場合に特に有効にはたらく．また，滑らかな関数の場合は，サブルーチン AQN9 及び AQE に比べ $f(x)$ の計算回数が少なく，効率が最もよい.
したがって，滑らかな関数及び振動型の関数に対しては本サブルーチンの使用を第一に勧める．特異点を持つ関数に対しては，その特異点が積分区間 [a,b] の端点だけに位置する場合はサブルーチン AQE を，また，特異点が区間内に位置する場合及びピーク型の関数に対してはサブルーチン AQN9 を用いた方がよい.

④ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数 $f(x)$ の計算関数を

NMIN ≤ 計算回数 ≤ NMAX

の範囲に限定している．すなわち収束判定にかかわりなく，$f(x)$ は少なくとも NMIN 回計算され，しかも NMAX 回を超えて計算されることはない．もし NMAX 回以内に (1.1) を満たす S が求まらなかった場合には ICON を 20000 として処理を打ち切る.
なお NMAX が 15 未満の値であったときは，15 が与えられたものと見なす.

⑤ 積分の近似値 S の精度

本サブルーチンは，与えられたε_a, ε_r に対して (1.1) を満たす S を求める．したがって，ε_r=0 とおけば絶対誤差 ε_a 以内の，またε_a=0 とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質とε_a, ε_r の値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べて，ε_a, ε_r が小さすぎる場合には，関数の計算回数がその上限に達しない場合でも，丸め誤差の影響のほうが大きくなり，これ以上計算を続行しても無意味である．このような状態を見出したときは ICON を 10000 にして処理を打ち切る．このときの S の精度は，使用計算機で達成可能な限度まで達している．また関数の計算回数が NMAX 回以内では収束しないこともある．このときの S の値は，それまでに得られている近似値であって，その精度は保証できない．この状態は ICON が 20000 の値をとることから知ることができる．

なお，本サブルーチンは，いずれの場合にも，積分の近似値 S のほかに，その絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

c. 使用例

助変数 p を 1 から 1 刻みに 10 まで変えて，積分，

$$\int_{-1}^{1}\cos(px)\,dx$$

を計算する．

```
C     **EXAMPLE**
      COMMON P
      EXTERNAL FUN
      A=-1.0
      B=1.0
      EPSA=1.0E-5
      EPSR=1.0E-5
      NMIN=15
      NMAX=511
      DO 10 I=1,10
      P=FLOAT(I)
      CALL AQC8(A,B,FUN,EPSA,EPSR,NMIN,
     *     NMAX,S,ERR,N,ICON)
   10 WRITE(6,600) P,ICON,S,ERR,N
      STOP
  600 FORMAT(' ',30X,'P=',F6.1,5X,
     * 'ICON=',I5,5X,'S=',E15.7,5X,
     * 'ERR=',E15.7,5X,'N=',I5)
      END

      FUNCTION FUN(X)
      COMMON P
      FUN=COS(P*X)
      RETURN
      END
```

(4) 手法概要

本サブルーチンは, 等差数列的 (公差 8) に分点を追加する拡張された意味のクレンショー・カーチス型積分法を用いている．元来のクレンショー・カーチス則は分点を倍々と追加しているが，倍までしなくともあとわずか追加すれば要求精度が得られることがあるので，元来の方法は分点を無駄にすることがある．本サブルーチンではこの無駄をできるだけ少なくするために，分点を 8 点ずつ等差数列的に追加するようになっている．また，高速フーリエ変換 (FFT) を用いて演算回数が減るように工夫してある．

a. 等差数列的に分点を増すクレンショー・カーチス型積分法

与えられた積分 $\int_a^b f(x)\,dx$ は，1 次変換

$$x=\frac{b-a}{2}t+\frac{a+b}{2}$$

により，

$$\frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}t+\frac{a+b}{2}\right)dt$$

と変換できるので，以下説明を簡単にするため，有限区間 [-1,1] における積分 (4.1) を考える．

$$I=\int_{-1}^{1} f(x)\,dx \qquad (4.1)$$

ここで元来のクレンショー・カーチス則について簡単に説明する．単位半円周上の等分点列を [-1,1] 上に射影した点列 (図 AQC8-1) を補間点とする補間多項式(すなわちチェビシェフ補間式)で $f(x)$ を近似し，これを項別積分して積分の近似値とする．そして要求精度が得られるまで分点を倍々と追加して積分の近似値の精度を上げるのが元来のクレンショー．カーチス則である．この方法では分点を無駄にすることがある．次に、等差数列的に分点を追加する方法を説明する。まず (0,1) 上に一様分布する Van der Corput 列を基にして点列$\{\alpha_j\}$(j=1,2,3,...)を

$$\alpha_1=1/4\quad,\alpha_{2j}=\alpha_j/2,\quad \alpha_{2j+1}=\alpha_{2j}+1/2$$
$$(j=1,2,3,.....)$$

なる漸化式から作る．単位円周上の点列 $\{\exp(2\pi i\alpha_j)\}$は原点に関して対称，実軸に関して非対称である(図 AQC8-2)．これを実軸に射影した点列$\{x_j=\cos 2\pi\alpha_j\}$ (j=1,2,3,...) は (-1,1)上にチェビシェフ分布するので，これを分点として用いる (図 AQC8-2).

図 AQC8-2 に示されるように，初め 7 点を分点として用い, 次々に 8 点ずつ分点を追加してゆく．

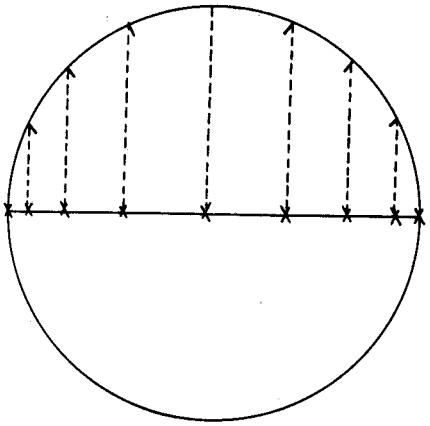


図 AQC8-1 元来のクレンショー・カーチス則で用いられる分点

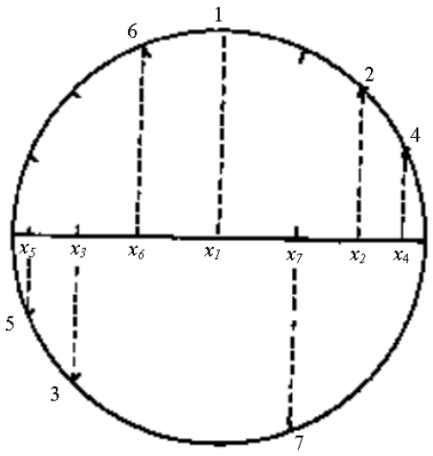


図 AQC8-2 本サブルーチンで用いる分点の列 {x_j}

分点の総数が，2^n-1 で表される数になったとき，全体の分点の配置は，単位半円周上を 2^n 等分割した点列を開区間 (-1,1) 上に射影した点列の配置と一致し，したがって開区間でのクレンショー・カーチス則のそれと同じになる．
次に，$f(x)$に対する補間多項式の作り方について説明する．α_i の性質から，$\{x_j\}$(j=1,2,...,7) は $\{\cos \pi_j/8\ \}$に一致し，また N 次のチェビシェフ多項式を $T_N(x)$ で表すと，x_{8k+j}(j=0, 1, ..., 7) は $T_8(x)-x_k=0$ (k=1,2,…) の 8 個の零点に一致する．
ここで，この性質を利用して初め 7 点，以後 8 点ずつ追加して次のようにして，補間多項式 $P_l(x)$の列を作る．以下，$P_0(x)$ は初めの 7 点を用いたときの補間式を表し，$P_l(x)$は、8 点ずつの追加を l 回行って合計 $8l$+7 個の点を用いたときの補間式を表す．
x=cos θ と変換すると

$$I=\int_{-1}^{1} f(x)dx=\int_{0}^{\pi} f(\cos\theta)\sin\theta\ \ d\theta$$

この $f(\cos\theta)$ を $P_l(\cos\theta)$ で近似する．

$$f(\cos\theta)\approx P_l(\cos\theta)$$

ここで，

$$P_0(\cos\theta)=\sum_{k=1}^{7} A_{-1,k}\sin k\theta/\sin\theta$$

$$P_{l+1}(\cos\theta)=P_l(\cos\theta)+\frac{\sin 8\theta}{\sin\theta}\omega_l(\cos 8\theta)\sum_{k=0}^{7}{}' A_{l,k}\cos k\theta$$

(l=0, 1, 2,⋯⋯) (4.2)

ただし，

$$\omega_0(\cos 8\theta)=1$$

$$\omega_l(\cos 8\theta)=2^l\prod_{k=1}^{l}(\cos 8\theta-\cos 2\pi\alpha_k)$$

$$=2^l\prod_{k=1}^{l}(T_8(x)-x_k),(l\ge 1)$$

(4.2) の右辺 Σ'は初項だけ 1/2 倍して和をとることを意味する．補間多項式 $P_l(x)$ の係数 $A_{-1,k}$ 及び $A_{i,k}$ は補間条件より決定する．
まず，$A_{-1,k}$(k=1,2,...,7) は初めの 7 個の分点 {cos π_j /8 }(j=1,2,...,7)を用いて，

$$A_{-1,k}=\frac{2}{8}\sum_{j=1}^{7}\left\{f\left(\cos\frac{\pi}{8}j\right)\sin\frac{\pi}{8}j\right\}\sin\frac{\pi}{8}kj,$$

(k=1, 2, ⋯, 7)

で与えられる．次に，$A_{i,k}$ (-1 ≤ i ≤ l-1) を既知として　$P_{l+1}(\cos\theta)$ に現れる $A_{l,k}$ (l ≥ 0) を求める．
この段階で追加される分点は $T_8(x)$-x_{l+1}=0 の零点，すなわち cos θ_j^{l+1} ,(θ_j^{l+1} =2π/8·(j+α_{l+1}), (j=0,1,2,...,7) であるから，この段階での補間条件，

$$f(\cos\theta_j^{l+1})\sin\theta_j^{l+1}=\sum_{k=1}^{7}A_{-1,k}\sin k\theta_j^{l+1}+\sin 2\pi\alpha_{l+1}\sum_{i=0}^{l}\omega_i(\cos 2\pi\alpha_{l+1})\sum_{k=0}^{7}{}' A_{l,k}\cos k\theta_j^{l+1}$$

$$(0\le j\le 7)$$

(4.3)

から $A_{l,k}$ を決定する．(4.3) の左辺に対して次のようなパラメタα_{l+1} を含む cosine 変換を施す．

$$f\left(\cos\theta_j^{l+1}\right)\sin\theta_j^{l+1}=\sum_{k=0}^{7}{}'a_k\cos k\theta_j^{l+1}\ ,\ (0\le j\le 7)$$

これを a_k に関する連立方程式と見なして解けば

$$a_k=\frac{2}{8}\sum_{j=0}^{7}f\left(\cos\theta_j^{l+1}\right)\sin\theta_j^{l+1}\cos k\theta_j^{l+1}$$

となる．実際の計算は実数型 FFT を用いて計算される．すると (4.3) から

$$a_k=\frac{1}{\sin 2\pi\ \alpha_{l+1}}\left(A_{-1,8-k}-\cos 2\pi\ \alpha_{l+1}\cdot A_{-1,k}\right)$$
$$+\sin 2\pi\ \alpha_{l+1}\sum_{i=0}^{l}\omega_i\left(\cos 2\pi\ \alpha_{l+1}\right)A_{i,k},$$
$$(0\le k\le 7)$$

が得られる．ただし，$A_{-1,0}=A_{-1,8}=0$ とする．
これを $A_{l,k}$ に関して解く．$l=0$ のとき
$\omega_0(\cos 2\pi\alpha_1)=1$ だから $A_{0,k}$ は容易に求められる．
$l\ge 1$ の場合，

$$l=m+2^n\qquad (0\le m<2^n)$$

と置き，関係式

$$\sin 2\pi\alpha_{l+1}\cdot\omega_{2^n-1}(\cos 2\pi\alpha_{l+1})=\sin(2^{n+1}\pi\alpha_{l+1})=1$$

を用いると，$A_{l,k}$ は次のようにして求められる．
初めに

$$B_{m+1}=a_k-\left(A_{-1,8-k}-\cos 2\pi\alpha_{l+1}\cdot A_{-1,k}\right)/\sin 2\pi\alpha_{l+1}$$
$$-\sin 2\pi\alpha_{l+1}\sum_{i=0}^{2^n-2}\omega_i\left(\cos 2\pi\alpha_{l+1}\right)A_{i,k}$$
$$(4.4)$$

と置いて，

$$B_{m-i}=(B_{m+1-i}-A_{2^n+i-1})\ /\ (\cos 2\pi\alpha_{l+1}-\cos 2\pi\alpha_{2^n+i})$$
$$i=0,1,2,...,m\qquad (4.5)$$

を順々に求めると，

$$A_{l,k}=B_0$$

によって $A_{l,k}$ が得られる．通常のニュートンの差分商の公式による場合と比較して，(4.4)，(4.5) による計算の安定性は，割算の回数が $l+2$ から $m+2$ に減るために少し良くなる．以上によって得られる補間式 $P_{l+1}(\cos\theta)$ を使って，積分の近似値は項別積分により，

$$I_{l+1}=\int_0^{\pi}P_{l+1}(\cos\theta)\sin\theta\,d\theta$$
$$=\sum_{k=1}^{7}A_{-1,k}\frac{2}{k}+\sum_{i=1}^{l}\sum_{k=0}^{7}{}'A_{i,k}W_{i,k}$$

(ただし，k による和は奇数番目だけ)

から計算される．ここで，重み係数 $W_{i,k}$ は，

$$W_{i,k}=\int_0^{\pi}\omega_i\left(\cos 8\theta\right)\sin 8\theta\cos k\theta d\theta\qquad (4.6)$$

で定義され，次のようにして計算される．重み $W_{2^n-1,k}$ は，

$$W_{2^n-1,k}=\int_0^{\pi}\sin 2^{n+3}\theta\cos k\theta\ \ d\theta=\frac{2^{n+4}}{4^{n+3}-k^2}$$

で求められる．この $W_{2^n-1,k}$ を初期値として，次の漸化式によって必要な $N(=2^m\times 4)$ 個の $W_{i,k}(0\le i\le 2^m-1, k=1,3,5,7)$ が計算される．

$$W_{2^n-1+2^{n-j+1}\cdot s+2^{n-j},k}=W_{2^n-1+2^{n-j+1}\cdot s,\,2^{n-j}\cdot 8+k}$$
$$+W_{2^n-1+2^{n-j+1}\cdot s,2^{n-j}\cdot 8-k}$$
$$-2\cos 2\pi\alpha_{2^j+2s}\cdot W_{2^n-1+2^{n-j+1}\cdot s,k}$$
$$,\ n=1,2,...,m-1\ ,\ 1\le j\le n,\ 0\le s<2^{j-1}-1,$$
$$0\le k\le 2^{n-j}\cdot 8-1(k\text{ は奇数}).\qquad (4.7)$$

この重み係数のための乗除算は，$N/2(\log_2 N-2)+4$ 回である．

b. 計算の手順

手順 1…初期設定

積分区間 $[a,b]$ を $[-1,1]$ へ変換するための定数，$m=(a+b)/2$, $r=(b-a)/2$, を定める．その他必要なあらゆる初期設定をする．

手順 2…分点と重み係数の決定

分点の個数の上限を 511 $(=2^9-1)$ 個にしたとき，分点 $\{x_j\}=\{\cos 2\pi\alpha_j\}$ $(j=1,2,...,511)$ は常に原点に関して対称に存在するので，256$(=2^8)$ 個の $\{\cos\pi\alpha_j\}(j=1,2,...,256)$ のテーブルがあればよい．このテーブルを漸化式によって作る．更に重み係数 $\{W_{i,2k+1}\}(0\le i\le 63,\ 0\le k\le 3)$ を漸化式 (4.7) に従って作り，1 次元配列に入れる．手順 2 は本サブルーチンが初めて呼ばれたときだけ実行し，2 回目以降呼ばれるときは省略する．

手順 3…初めの 7 分点による積分の近似値の計算

分点 $\cos\pi j/8$ $(j=1,2,...,7)$ 上で 7 項の実数型 FFT を用いて $A_{-1,2k+1}(0\le k\le 3)$ を計算し，これと手順 2 で求めた重み係数を乗じて，積分の近似値 I_0 を得る．

手順 4…三角関数の計算

分点用のテーブル $\{\cos\pi\alpha_j\}$ から，手順 5 で必要となる三角関数 $\cos 2\pi\alpha_{l+1}$, $\sin 2\pi\alpha_{l+1}$, $\sin\pi\alpha_{l+1}$ などの値を計算する．

手順 5…a_k と $A_{l,k}$ の計算

追加する 8 個の分点上で関数値を計算し，8 項実数型 FFT を用いて $a_{2k+1}(0\le k\le 3)$ を求める．次に (4.4) ～ (4.6) に基づいて $A_{l,2k+1}(0\le k\le 3)$ を計算する．

手順 6…積分値の計算と収束判定

$\sum_{k=0}^{3} A_{l,2k+1} W_{l,2k+1}$ をこれ以前の積分の近似値 $I_l (l \geq 0)$ に加えて新しい近似値 I_{l+1} を求める．次に I_{l+1} が持つ打切り誤差の推定値 e_{l+1} を求めこれに対して収束判定を行う．同時に e_l に対しても判定を行い両方共，判定に満足したら合格として処理を終える．もし不合格なら l を 1 つ増加して手順 4 へ戻る．

c. 誤差評価

$f(x)$ を a.で述べた補間多項式 $P_l(x)$ で近似したときの誤差項を $R_l(x)$ で表すと

$$\begin{aligned} f(x) &= P_l(x) + R_l(x) \\ &= \sum_{k=1}^{7} A_{-1,k} U_{k-1}(x) + U_7(x) \sum_{i=0}^{l-1} \omega_i (T_8(x)) \\ &\quad \times \sum_{k=0}^{7}{}' A_{i,k} T_k(x) + U_7(x) \omega_l (T_8(x)) \\ &\quad \times 2^{-(8l+7)} f[x, x_1, x_2, \ldots, x_{8l+7}] \end{aligned}$$

と表される．係数 $2^{-(8l+7)}$ は，差分商による通常の誤差項表示に合わせるために現れたものである．

$U_k(x)$ は次のような第 2 種 k 次チェビシェフ多項式である．

$U_k(x)=\sin(k+1)\theta / \sin\theta$ ， $x=\cos\theta$

積分の近似値 I_l に対する打切り誤差 E_l は，

$$E_i = \int_{-1}^{1} R_l(x) dx = \int_{-1}^{1} U_7(x) \omega_l (T_8(x)) 2^{-(8l+7)} f[x, x_1, \ldots, x_{8l+7}] dx$$

で表される．差分商を積分表示し，さらにチェビシェフ展開する．

$$\begin{aligned} 2^{-(8l+7)} f[x, x_1, \ldots, x_{8l+7}] &= \frac{1}{2\pi i} \oint_C \frac{f(z) dz}{(z-x) U_7(z) \omega_l (T_8(z))} \\ &= \sum_{k=0}^{\infty}{}' C_{l,k} T_k(x) \end{aligned}$$

ここで，積分路 c は，図 AQC8-3 に示す単一閉曲線である．

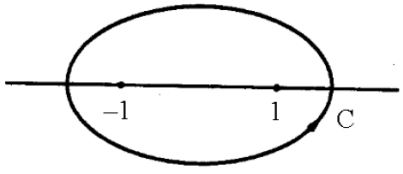


図 AQC8-3

また，係数 $C_{l,k}$ は，

$$C_{l,k} = \frac{1}{2\pi i} \frac{2}{\pi} \oint_C \frac{U_k^*(z) f(z) dz}{U_7(z) \omega_l (T_8(z))}$$

である．$U_k^*(z)$ は，第 2 種チェビシェフ関数で，

$$U_k^*(z) = \int_{-1}^{1} \frac{T_k(x) dx}{(z-x)\sqrt{1-x^2}} = \frac{\pi}{\left(z + \sqrt{z^2-1}\right)^k \sqrt{z^2-1}}$$

で定義される．したがって，

$$E_l = \sum_{k=0}^{\infty}{}' C_{l,k} W_{l,k} \qquad (k = \text{奇数})$$

となる．もし $f(z)$ が点 z_m $(m=1,2,...M)$ に極を持つ有理型関数であれば，

$$C_{l,k} = -\frac{2}{\pi} \sum_{m=1}^{M} \frac{U_k^*(z_m) \mathrm{Res}\, f(z_m)}{U_7(z_m) \omega_l (T_8(z_m))}$$

となる．ここで Res $f(z_m)$ は点 z_m での留数である．さて，

$$U_k^*(z_m) \propto r_m^{-k}, \quad r_m = \left| z_m + \sqrt{z_m^2 - 1} \right| > 1$$

であり，z_m が実軸上の区間 [– 1,1] に非常に近くない限り，

$$\left|C_{l,1}\right| > \left|C_{l,k}\right| \quad , (k \geq 3)$$

が成立つ．したがって，

$$|E_l| \approx |C_{l,1}| \cdot |W_{l,1}| \leq \left(\left|A_{l-1,7}\right| + \left|A_{l-1,5}\right| \right) \cdot |W_{l,1}| \equiv e_l$$

より打切り誤差を評価できる．実際に値が評価できない $\left|C_{l,1}\right|$ の代わりに $\left|A_{l-1,7}\right|$ と $\left|A_{l-1,5}\right|$ を用いた．

z_m が [-1,1] に非常に近いか，$f(x)$ の p 階微分 $f^{(p)}(x)$ $(p \geq 1)$ が [– 1,1] で不連続の場合には以上の誤差評価が成り立たない．この対策として

次のようにする．$A_{i,k}$ が速く減少すれば，
$\left|I_{2^n-1}-I_{2^{n-1}-1}\right|$ は I_{2^n-1} に対するよい誤差評価となる．
それゆえ，もし
$e_{2^n-1}>\left|I_{2^n-1}-I_{2^{n-1}-1}\right|$

が成立すれば，$2^n-1 \le l \le 2^{n+1}-1$ において e_l を誤差評価として用い，そうでなければ，
$e'_l \equiv e_l \left|I_{2^n-1}-I_{2^{n-1}-1}\right| \Big/ e_{2^{n-1}-1}$

を誤差評価とする．

d. 収束判定基準
積分の近似値 I_{l+1} は，打切り誤差のほかに計算誤差を持つが，本サブルーチンでは計算誤差の上限 ρ を
$\rho = u(l+1)\|f\|_\infty$
と推定している．ただし u は丸め誤差の単位であり，$\|f\|_\infty = \max_j \left|f\left(x_j\right)\right|$ である．分点がチェビシェフ分布し，FFT を用いているから，この仮定は実用上妥当である．
さて収束判定定数を

$\tau = \max\left(\varepsilon_a, \varepsilon_r \left|\int_{-1}^{1} f(x)dx\right|, \rho\right)$

とおいて

e_{l+1} (又は e'_{l+1}) $< \tau$

なら I_{l+1} を積分値としてパラメタ ***S*** へ出力する．
パラメタ ERR には，$e_{l+1} \ge \rho$ のときは
e_{l+1} を (ICON = 0)，$e_{l+1} < \rho$ のときは ρ を (ICON=10000) 出力する．
τ の値を定めるときの $\left|\int_{-1}^{1} f(x)dx\right|$ としては，$|I_l|$ を用いる．
なお，詳細については，参考文献 [65] , [66] を参照すること．

G23-11-0401 AQE,DAQE

1 次元有限区間積分 (関数入力,二重指数関数型積分公式)
CALL AQE (A,B,FUN,EPSA,EPSR,NMIN,NMAX,S, ERR,N, ICON)

(1) 機能

関数 $f(x)$ 及び $a,b,\varepsilon_a,\varepsilon_r$ が与えられたとき

$$\left|S-\int_a^b f(x)dx\right| \le \max\left(\varepsilon_a,\varepsilon_r\cdot\left|\int_a^b f(x)dx\right|\right) \qquad (1.1)$$

を満たす積分の近似値 S を，高橋・森の二重指数関数型積分公式により求める．

(2) パラメタ

A …… **入力**．積分区間の下限 *a*.
B …… **入力**．積分区間の上限 *b*. (*b*≤*a* でも可)
FUN …… **入力**．被積分関数 *f*(*x*) を計算する関数副プログラム名 (使用例参照).
EPSA …… **入力**．積分の近似値 *S* に対する絶対誤差の上限 $\varepsilon_a(\ge 0.0)$.
EPSR …… **入力**．積分の近似値 *S* に対する相対誤差の上限 $\varepsilon_r(\ge 0.0)$.
NMIN …… **入力**．被積分関数の計算回数の下限 (≥0)
標準値としては 20 が適当.
NMAX …. **入力**．被積分関数の計算回数の上限.
(NMAX≥NMIN)
標準値としては 641 が適当 (これよりも大きく指定した場合は 641 と見なす).
(使用上の注意⑤参照).
S …… **出力**．積分の近似値 (使用上の注意⑥参照).
ERR …… **出力**．積分の近似値 *S* の絶対誤差の推定値.
N …… **出力**．被積分関数の計算回数.
ICON …… **出力**．コンディションコード.
表 AQE-1 参照.

(3) 使用上の注意

a. 使用する副プログラム.
① SSL II... AMACH, MGSSL, AFMIN
② FORTRAN 基本関数... MAX0, AMAX1,AMIN1, ABS, FLOAT, EXP, COSH, SINH

b. 注意
① パラメタ FUN に対応する関数副プログラムは，積分変数だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムで，その関数名を EXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図る．

表 AQE-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	丸め誤差のために要求精度が得られなかった.	S には，得られた近似値を出力する．精度は達成可能な限度まで達している.
11000 12000 13000	1000 の位が 1,2,3 の順に区間の上限,下限,上･下限付近で関数値が急激に増大していることを表す.	要求精度をゆるめて処理を続ける．S には得られた近似値を出力する．精度は達成可能な限度まで達している.
20000	被積分関数の計算回数がその上限に達しても要求精度が得られなかった.	処理を打ち切る. S には，そのときまでに得られている近似値を出力するが精度は保証できない.
21000 22000 23000	コード 11000 - 13000 の事象が発生した後，被積分関数の計算回数がその上限に達した.	
25000	分点のための表（作業領域）が足りなくなった.	処理を打ち切る. S には，本サブルーチンで可能な最小の刻み幅による近似値を出力する.
30000	次のいづれかであった. ① EPSA < 0.0 ② EPSR < 0.0 ③ NMIN < 0 ④ NMAX < NMIN	処理を打ち切る.

② 本サブルーチンを多数回呼ぶとき，最初だけ 641 個の定数（積分公式の分点と重みの表）を計算し，2 回目以降は省略する．したがって，2 回目以降は，計算量がその分だけ少ない.
③ 本サブルーチンは，被積分関数 *f* (*x*) が，積分区間 [a,b] の端付近で変化が激しい場合に，特に有効にはたらき，代数特異点及び対数特異点が端点だけ（片方でも両方でもよい）に位置する場合は，本サブルーチンの使用を第一に勧める. 積分区間内に特異点を持つ場合は，そのような点で積分領域を分割して本サブルーチンを使用するか，あるいは，サブルーチン AQN9 を用いた方がよい.
ピーク型の関数に対してはサブルーチン AQN9 を，滑らかな関数及び振動型の関数に対してはサブルーチン AQC8 を用いた方がよい.
④ 本サブルーチンは積分区間の両端では関数を計算しない．したがって両端で関数値が無限大 ($f(x)\to\pm\infty$) となってもかまわない．しかし区間内に無限大となる点があってはならない.
⑤ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数 *f*(*x*) の計算回数を

NMIN ≤ 計算回数 ≤ NMAX

の範囲に限定している．すなわち収束判定にかかわりなく，$f(x)$ は少なくとも NMIN 回計算され，しかも NMAX 回を超えて計算されることはない．もし NMAX 以内に (1.1) を満たす $\boldsymbol{S}$ が求まらなかった場合は，ICON を 20000～23000 として処理を終える．
なお NMAX が極端に小さく，例えば NMAX=2 と与えられたときは，$f(x)$ の挙動に応じて決まる一定の値まで自動的に引き上げている．

⑥ 積分の近似値 $\boldsymbol{S}$ の精度.
本サブルーチンは，与えられた ε_a , ε_r に対して (1.1) を満たす $\boldsymbol{S}$ を求める．したがって，$\varepsilon_r = 0$ とおけば絶対誤差ε_a以内の，また $\varepsilon_a=0$ とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質と ε_a , ε_r の値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べて，ε_a , ε_r が小さすぎる場合には，関数の計算回数がその上限に達しない場合でも，丸め誤差の影響の方が大きくなり，これ以上計算を続行しても無意味である．このような状態を見出したときは ICON を 10000 にして処理を打ち切る．このときの $\boldsymbol{S}$ の精度は使用計算機で達成可能な限度まで達している．また関数の計算回数が NMAX 回以内では収束しないこともある．このときの $\boldsymbol{S}$ の値は，それまでに得られている近似値であって，その精度は保証できない．この状態は ICON が 20000～23000 の範囲の値をとることから知ることができる．また，サブルーチン内で定める最小の刻み幅で積分しても収束しないこともあり，このとき ICON は 25000 の値をとる.
なお，本サブルーチンは，いずれの場合にも，積分の近似値 $\boldsymbol{S}$ のほかに，その絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる.

⑦ 桁落ちを防ぐための関数副プログラムの記述
例えば，積分

$$I=\int_1^3 \frac{dx}{x(3-x)^{1/4}(x-1)^{3/4}}$$

における被積分関数は区間の端点 x = 1,3 で特異点を持ち，しかもその点で関数値は発散するので積分値への寄与の度合は大きい．この積分を精度よく求めるには，端点近傍の関数値を精度よく計算することが重要である．しかし，端点近傍では

(3.0 - X) 及び (X-1.0).

の計算で桁落ちを起すため関数値を精度よく計算できない.
本サブルーチンでは，この桁落ちを防ぐため，被積分関数を座標変換した形で記述できるようになっている．すなわち，関数副プログラムのパラメタは，以下に示す仕様を持っている.
FUNCTION FUN(X)
ここで
X...入力．大きさ 2 の 1 次元配列.
X(1) は積分変数 x に対応し，X(2) は x の値に依存して以下のように設定されている．AA = min(a,b), BB = max(a,b) として
· AA ≤ x <(AA+BB) / 2 のとき
X(2)=AA – x
· (AA+BB) / 2 ≤ x ≤ BB のとき
X(2)=BB – x
この X(2) は端点からの正確な距離である．利用者は，この X(2) の値を使って関数を記述するには，次のようにすればよい.

$$f(x)=\begin{cases} \mathrm{X(2)}<0.0 \text{ のとき } f(\mathrm{AA}-\mathrm{X(2)}) \\ \mathrm{X(2)}\geq 0.0 \text{ のとき } f(\mathrm{BB}-\mathrm{X(2)}) \end{cases}$$

X(1) , X(2) のどちらを使うかは利用者の自由である (使用例参照).

c. 使用例
2 つの積分

$$I_1=\int_0^1 \frac{dx}{\sqrt{x}},\quad I_2=\int_1^3 \frac{dx}{x(3-x)^{1/4}(x-1)^{3/4}}$$

を計算する．I_1 の被積分関数は関数副プログラム FUN1 で，I_2 のそれは FUN2 で定義するものとする．特に FUN2 では，注意⑦で述べた記述法を使っている.

```
C     **EXAMPLE**
      EXTERNAL FUN1,FUN2
      A=0.0
      B=1.0
      EPSA=1.0E-5
      EPSR=0.0
      NMIN=20
      NMAX=641
      CALL AQE(A,B,FUN1,EPSA,EPSR,NMIN,
     *     NMAX,S1,ERR1,N1,ICON1)
      A=1.0
      B=3.0
      CALL AQE(A,B,FUN2,EPSA,EPSR,NMIN,
     *     NMAX,S2,ERR2,N2,ICON2)
      WRITE(6,600) ICON1,S1,ERR1,N1,
     *             ICON2,S2,ERR2,N2
      STOP
  600 FORMAT(' ',30X,'ICON1=',I5,5X,
     *'S1=',E15.7,5X,'ERR1=',E15.7,
     *5X,'N1=',I5//
     *      ' ',30X,'ICON2=',I5,5X,
     *'S2=',E15.7,5X,'ERR2=',E15.7,
     *5X,'N2=',I5)
      END

      FUNCTION FUN1(X)
      FUN1=0.0
      IF(X.GT.0.0) FUN1=1.0/SQRT(X)
      RETURN
      END
```

```
      FUNCTION FUN2(X)
      DIMENSION X(2)
      T=X(2)
      IF(T.GE.0.0) GO TO 10
      P=(1.0-T)*(2.0+T)**0.25*(-T)**0.75
      GO TO 20
   10 P=(3.0-T)*T**0.25*(2.0-T)**0.75
   20 FUN2=0.0
      IF(P.GT.0.0) FUN2=1.0/P
      RETURN
      END
```

(4) 手法概要

本サブルーチンは，高橋・森の二重指数関数型積分公式に基づく自動積分法を用いている．最初にこの方法の原理を述べ，次に本サブルーチンの算法を述べる．

a. 二重指数関数型積分公式

与えられた積分 $\int_a^b f(x)dx$ は，1 次変換，

$$x=\frac{b-a}{2}t+\frac{a+b}{2}$$

により

$$\frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}t+\frac{a+b}{2}\right)dt$$

と変換できるので，以下説明を簡単にするため，有限区間 [-1,1] における (4.1) を考える．

$$I=\int_{-1}^{1} f(x)dx \qquad (4.1)$$

ここで，$f(x)$ は開区間 (-1,1) で解析的であれば，$x=\pm1$ において

$(1-x)^{\alpha}(1+x)^{\beta}$ ， $-1<\alpha, \beta$

程度の特異性を持っていてもさしつかえない．今，(4.1) に変数変換，

$$x=\phi(t) \qquad (4.2)$$

を行って，区間 [-1,1] を $(-\infty, \infty)$ に変換すると，積分 (4.1) は

$$I=\int_{-\infty}^{\infty} f(\phi(t))\phi'(t)dt \qquad (4.3)$$

となる．無限区間における台形則の最良性に注目して，これに，刻み幅 h の台形則を適用すれば，

$$I_h=h\sum_{n=-\infty}^{\infty} f(\phi(nh))\phi'(nh) \qquad (4.4)$$

なる一つの積分公式が得られる．ここで，高橋・森は (4.4) の無限和を有限和で近似するときに生ずる打切り誤差を考慮し，比較解析した結果，$|t|$ の増加に伴い，(4.3) の被積分関数が

$$\left|f(\phi(t))\phi'(t)\right| \approx \exp(-a\exp|t|), a>0 \qquad (4.5)$$

のように減衰して 0 に近づくような変換$\phi(t)$ が最適であることを示した (図 AQE-1 参照).

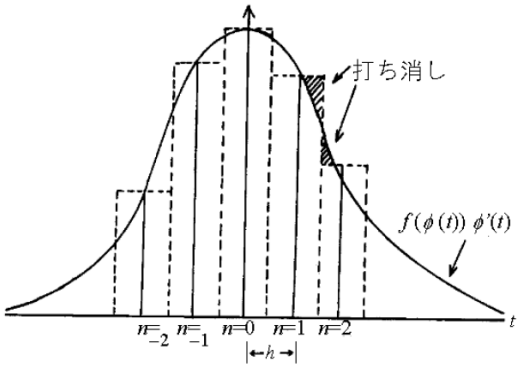


図 AQE-1 $f(\phi(t))\,\phi'(t)$ に対する台形則

(4.5) のような二重指数関数型減衰をするような変数変換として，本サブルーチンでは

$$x=\phi(t)=\tanh\left(\frac{3}{2}\sinh(t)\right)$$
$$\phi'(t)=\frac{3}{2}\cosh(t)\Big/\cosh^2\left(\frac{3}{2}\sinh(t)\right) \qquad (4.6)$$

を採用している．

b. 計算の手順

手順 1…初期限定

有限区間 $[a, b]$ の [-1,1] への変換のための定数，

$r=(b-a)/2$

を定める．その他必要なあらゆる初期設定を行う．

手順 2…無限和 (4.4)を有限和で近似するときの上限下限を設定する．同時に積分の非常に大ざっぱな近似値 S'も求める．

手順 3…刻み幅を 2 等分しながら，有限個数の積和計算により，$\boldsymbol{I}_h$, $\boldsymbol{I}_{h/2}$, $\boldsymbol{I}_{h/4}$, ...に対する近似値 $S(h)$, $S(h/2)$, $S(h/4)$...を収束するまで求めていく．

手順 4…S，ERR，ICON の設定

c. 収束判定基準
等間隔刻み台形則における刻み幅を h とすると，h が十分小さければ，誤差を$\Delta \boldsymbol{I}_h = \boldsymbol{I} - \boldsymbol{I}_h$ と置いて

$$\left|\Delta I_{h/2}\right| \approx \left|\Delta I_h\right|^2$$

と表されることが解析的に示されている．これから，要求精度をεとして，$\left|\Delta I_{h/2}\right| \le \varepsilon$ としようとすれば，$\left|\Delta I_h\right| \le \varepsilon^{1/2}$ であればよい．また数値的に$|\Delta \boldsymbol{I}_{h/2}| \ll |\Delta \boldsymbol{I}_h|$であることを考慮すれば

$$\left|S(h) - S(h/2)\right| \approx \left|I_h - I_{h/2}\right| = \left|\Delta I_h - \Delta I_{h/2}\right| \approx \left|\Delta I_h\right|$$

と書ける．したがって，収束条件として

$$\left|S(h) - S(h/2)\right| \le \eta = \varepsilon^{\alpha}$$

とおけば，理論的には，α=1/2 を採用できる．本サブルーチンでは，安全性を見込んで経験的に，αとして次の値を用いている．
ε' = max(ε_a / |S'|, ε_r) (ただし S' = 0.0 のときは $\varepsilon' = \varepsilon$)とおいて

$10^{-4} \le \varepsilon'$ のときα=1.0
$10^{-5} \le \varepsilon' < 10^{-4}$ のときα=0.9
$10^{-10} \le \varepsilon' < 10^{-5}$ のときα=0.8
$\varepsilon' < 10^{-10}$ のときα=0.75

を採用し，要求精度 εとしては max (ε_a, $\varepsilon_r \cdot |S(h/2)|$,) を採用している．

d. 刻み幅の初期値，しきい値の設定
刻み幅の初期値を 0.5 と置く．
$F = \left| f\left(\phi(nh)\right)\phi'(nh)\right| \le \eta'$ なる条件が二度続けて成立する n (n= ±1, ..., ±10) をさがす．ここで，$\eta' = \min\left(\max\left(\varepsilon_a, \varepsilon_r \|F\|_\infty\right), 10^{-4}\|F\|_\infty\right)$とする．もし，$n$ = ±10 においても，$F > \eta'$であるときには，収束判定基準 ε'を F に置き換えて手順 3 を実行する．

e. 丸め誤差の影響の検出
$e_h = |S(h) - S(h/2)|$ とすると，$\varepsilon_h \le \eta$ となれば，要求精度の積分の近似値として $S(h/2)$ をパラメタ S に，また $e_h^{1/\alpha}$ をパラメタ ERR に出力するが，$e_h \le \eta$ であっても次の事象が発生すれば，丸め誤差の影響が打切り誤差を超えたと判断し，計算をやめ，ICON=10000 とする．

事象 1　$e_{h/2} \ge e_h \ge e_{2h}$ が成り立つ場合．
事象 2　$e_h \le u^{\alpha}\,|S(h/2)|$．　u は丸め誤差の単位．

このとき，パラメタ ERR として事象 1 の場合は e_h を，事象 2 の場合は $u|S(h/2)|$ を出力する．

なお，詳細については，参考文献 [67] , [68] を参照すること．

G23-21-0101 AQEH, DAQEH

1 次元半無限区間積分 (間数入力，二重指数関数型積分公式)
CALL AQEH (FUN,EPSA,EPSR,NMIN,NMAX,S, ERR,N,ICON)

(1) 機能

関数 $f(x)$ 及び ε_a ε_r が与えられたとき，

$$\left|S-\int_0^\infty f(x)\,dx\right| \le \max\left(\varepsilon_a, \varepsilon_r\cdot\left|\int_0^\infty f(x)\,dx\right|\right) \qquad (1.1)$$

を満たす積分の近似値 S を，高橋・森の二重指数関数型積分公式により求める．

(2) パラメタ

FUN……**入力**．被積分関数 $f(x)$ を計算する関数副プログラム名 (使用例参照)．

EPSA……**入力**．積分の近似値 $\boldsymbol{S}$ に対する絶対誤差の上限 ε_a(≥0.0)．

EPSR……**入力**．積分の近似値 $\boldsymbol{S}$ に対する相対誤差の上限 ε_r(≥0.0)．

NMIN……**入力**．被積分関数の計算回数の下限 (≥0)
標準値としては 20 が適当．

NMAX…**入力**．被積分関数の計算回数の上限 (NMAX≥NMIN)．
標準値としては 689 が適当 (これよりも大きく指定した場合は 689 と見なす)．
(使用上の注意⑤参照)．

S……**出力**．積分の近似値 (使用上の注意⑥参照)

ERR…**出力**．積分の近似値 S の絶対誤差の推定値．

N……**出力**．被積分関数の計算回数．

ICON……**出力**．コンディションコード．
表 AQEH-1 参照．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AFMAX, AMACH, MGSSL

② FORTRAN 基本関数 ... AMIN1, ABS, AMAX1, FLOAT, SINH, COSH, EXP

b. 注意

① パラメタ FUN に対応する関数副プログラムは，積分変数だけを引数にもつ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムで，その関数名を EXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することより，呼び出す側のプログラムとの連絡を図る．

② 本サブルーチンを多数回呼ぶとき，最初だけ 689 個の定数 (積分公式の分点と重みの表) を計算し，2 回目以降は省略する．したがって，2 回目以降は計算量がその分だけ少ない．

表 AQEH-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	丸め誤差のために要求精度が得られなかった．	S には，得られた近似値を出力する．精度は達成可能な限度まで達している．
11000 12000 13000	1000 の位が 1, 2, 3 の順に ① x=0 の付近で関数値が急激に増大している． ② X→∞のとき関数値の 0 への収束が遅い ③ ①，②が同時に起きたことを表わす．	要求精度をゆるめて処理を続ける．S には得られた近似値を出力する．精度は達成可能な限度まで達している．
20000	被積分関数の計算回数がその上限に達しても要求精度が得られなかった．	処理を打ち切る． S には，そのときまでに得られている近似値を出力するが精度は保証できない．
21000 22000 23000	コード 11000 - 13000 の事象が発生した後，被積分関数の計算回数がその上限に達した．	
25000	分点のための表（作業領域）が足りなくなった．	処理を打ち切る． S には，本サブルーチンで可能な最小の刻み幅による近似値を出力する．
30000	次のいづれかであった． ① EPSA < 0.0 ② EPSR < 0.0 ③ NMIN < 0 ④ NMAX < NMIN	処理を打ち切る．

③ 本サブルーチンは，被積分関数 $f(x)$ が $x\to+\infty$ において 0 への収束が比較的遅いものや，ガウス・ラゲールの公式が適用できないものに対しても有効に働く．
$f(x)$ が激しく振動する場合には，高精度な積分値を得ることができない場合がある．

④ 本サブルーチンは，積分区間の下限（原点）では関数を計算しない．したがって下限で関数値が無限大 ($f(x)\to+\infty$) となってもかまわない．積分区間の上端のほうでは，x の大きな点での関数値を必要とするので，殊に高い要求精度を与えた場合には，関数副プログラム FUN の中では，オーバフロー，アンダフローに対する処置が必要である．

⑤ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数 $f(x)$ の計算回数を

NMIN≤計算回数≤NMAX

の範囲に限定している．すなわち収束判定にかかわりなく，$f(x)$ はすくなくとも NMIN 回計算され，しかも NMAX 回を超えて計算されるこ

とはない．もし NMAX 回以内に (1.1) を満たす S が求まらなかった場合は，ICON を 20000～23000 として処理を終える．なお，NMAX が極端に小さく，例えば NMAX=2 と与えられたときは，$f(x)$ の挙動に応じて決まる一定の値まで自動的に引き上げている．

⑥ 積分の近似値の精度

本サブルーチンは，与えられた ε_a , ε_r に対して (1.1)を満たす S を求める．したがって $\varepsilon_r = 0$ とおけば絶対誤差 ε_a 以内の，また $\varepsilon_a = 0$ とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質と ε_a , ε_r の値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べて，ε_a , ε_r が小さすぎる場合には，関数の計算回数がその上限に達しない場合でも，丸め誤差の影響の方が大きくなり，これ以上計算を続行しても無意味である．このような状態を見出したときは ICON を 10000 にして処理を打ち切る．このときの $\boldsymbol{S}$ の精度は使用計算機で達成可能な限度まで達している．また関数の計算回数が NMAX 回以内では収束しないことがある．このときの $\boldsymbol{S}$ の値は，それまでに得られている近似値であって，その精度は保証できない．この状態は ICON が 20000 ～ 23000 の範囲の値をとることから知ることができる．また，サブルーチン内で定める最小の刻み幅で積分しても収束しないこともあり，このとき ICON は 25000 の値をとる．

なお，本サブルーチンは，いずれの場合にも，積分の近似値 $\boldsymbol{S}$ のほかに，その絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

c. 使用例

積分，

$$\int_0^{\infty} e^{-x}\sin x\,dx$$

を計算する．

```
C     **EXAMPLE**
      EXTERNAL FUN
      EPSA=1.0E-5
      EPSR=0.0
      NMIN=20
      NMAX=689
      CALL AQEH(FUN,EPSA,EPSR,NMIN,NMAX,
     *S,ERR,N,ICON)
      WRITE(6,600) ICON,S,ERR,N
      STOP
  600 FORMAT(' ',30X,'ICON=',I5,5X,
     *'S=',E15.7,5X,'ERR=',E15.7,
     *5X,'N=',I5)
      END
      FUNCTION FUN(X)
      IF(X.GT.176.0)GO TO 10
      FUN=EXP(-X)*SIN(X)
      RETURN
   10 FUN=0.0
      RETURN
      END
```

(4) 手法概要

本サブルーチンは高橋・森の二重指数関数型積分公式に基づく自動積分法を用いている．

この方法の原理及び，本サブルーチンの手順についてはサブルーチン AQE の手法概要を参照されたい．ただし, ここでは, 積分変数 x に対して, 変換,

$$x = \phi(t) = \exp\left(\frac{3}{2}\sinh t\right)$$

を,したがって,重み関数 $\phi'(t)$ に対して

$$\phi'(t) = \frac{3}{2}\cosh t \cdot \exp\left(\frac{3}{2}\sinh t\right)$$

を採用している．

G23-31-0101 AQEI, DAQEI

1 次元全無限区間積分 (関数入力，二重指数関数型積分公式)
CALL AQEI(FUN,EPSA,EPSR,NMIN,NMAX,S,ERR, N,ICON)

(1) 機能

関数 $f(x)$ 及び ε_a, ε_r が与えられたとき

$$\left|S-\int_{-\infty}^{\infty} f(x)dx\right| \le \max\left(\varepsilon_a, \varepsilon_r \cdot \left|\int_{-\infty}^{\infty} f(x)dx\right|\right) \qquad (1.1)$$

を満たす積分の近似値 S を，高橋・森の二重指数関数型積分公式により求める．

(2) パラメタ

FUN……**入力**．被積分関数 $f(x)$ を計算する関数副プログラム名 (使用例参照)．

EPSA……**入力**．積分の近似値 S に対する絶対誤差の上限 ε_a(≥0.0)．

EPSR……**入力**．積分の近似値 S に対する相対誤差の上限 ε_r(≥0.0)．

NMIN……**入力**．被積分関数の計算回数の下限 (≥0)
標準値としては 20 が適当．

NMAX……**入力**．被積分関数の計算回数の上限 (NMAX≥NMIN)．
標準値としては 645 が適当 (これよりも大きく指定した場合は 645 と見なす)．
(使用上の注意⑤参照)．

S……**出力**．積分の近似値 (使用上の注意⑥参照)

ERR……**出力**．積分の近似値 S の絶対誤差の推定値．

N……**出力**．被積分関数の計算回数．

ICON……**出力**．コンディションコード．
表 AQEH-1 参照．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AFMAX, AMACH, MGSSL

② FORTRAN 基本関数 ... AMIN1, ABS, AMAX1, FLOAT, SINH, COSH, EXP

b. 注意

① パラメタ FUN に対応する関数副プログラムは，積分変数だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムで，その関数名を EXTERNAL 文で宣言しなければならない．この関数が助変数を含むときは，これらは COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図る．

表 AQEI-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	丸め誤差のために要求精度が得られなかった．	S には，得られた近似値を出力する．精度は達成可能な限度まで達している．
11000 12000 13000	1000 の位が 1,2,3 の順に $x\to-\infty$, $x\to\infty$, $x\to\pm\infty$ のとき関数値の 0 への収束が遅いことを表す．	要求精度をゆるめて処理を続ける．S には得られた近似値を出力する．精度は達成可能な限度まで達している．
20000	被積分関数の計算回数がその上限に達しても要求精度が得られなかった．	処理を打ち切る．S には，そのときまでに得られている近似値を出力するが精度は保証できない．
21000 22000 23000	コード 11000 ～ 13000 の事象が発生した後，被積分関数の計算回数がその上限に達した．	
25000	分点のための表（作業領域）が足りなくなった．	処理を打ち切る．S には，本サブルーチンで可能な最小の刻み幅による近似値を出力する．
30000	次のいづれかであった． ① EPSA < 0.0 ② EPSR < 0.0 ③ NMIN < 0 ④ NMAX < NMIN	処理を打ち切る．

② 本サブルーチンを多数回呼ぶとき，最初だけ 645 個の定数 (積分公式の分点と重みの表) を計算し，2 回目以降は省略する．したがって，2 回目以降は計算量がその分だけ少ない．

③ 本サブルーチンは，被積分関数 $f(x)$ が $x\to\pm\infty$ において 0 への収束が比較的遅いものや，ガウス・エルミートの公式が適用できないものに対しても有効に働く．
$f(x)$ が原点付近で高いピークを持つもの，激しく振動するものに対しては，高精度の積分値を得ることができない場合がある．

④ 本サブルーチンは，積分変数 x の絶対値の大きな点での関数値を必要とするので，殊に高い要求精度を与えた場合には，関数プログラム FUN の中では，このような場合のオーバフロー，アンダフローに対する処理が必要となる．

⑤ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数 $f(x)$ の計算回数を

NMIN≤計算回数≤NMAX

の範囲に限定している．すなわち，収束判定にかかわりなく，$f(x)$ は少なくとも NMIN 回計算され，しかも NMAX 回を超えて計算されることはない．もし NMAX 回内に (1.1) を満たす S

が求まらなかった場合は，ICON を 20000～23000 として処理を終える．
なお，NMAX が極端に小さく，例えば NMAX=2 と与えられたときは，$f(x)$ の挙動に応じて決まる一定の値まで自動的に引き上げている．

⑥ 積分の近似値の精度
本サブルーチンは，与えられた ε_a , ε_r に対して(1.1)を満たす S を求める．したがって $\varepsilon_r = 0$ とおけば絶対誤差 ε_a 以内の，また $\varepsilon_a = 0$ とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質と ε_a , ε_r の値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べて，ε_a , ε_r が小さすぎる場合には，関数の計算回数がその上限に達しない場合でも，丸め誤差の影響の方が大きくなり，これ以上計算を続行しても無意味である．このような状態を見出したときは ICON を 10000 にして処理を打ち切る．このときの ***S*** の精度は使用計算機で達成可能な限度まで達している．また関数の計算回数が NMAX 回以内では収束しないことがある．このときの ***S*** の値は，それまでに得られている近似値であって，その精度は保証できない．この状態は ICON が 20000 - 23000 の範囲の値をとることから知ることができる．また，サブルーチン内で定める最小の刻み幅で積分しても収束しないこともあり，このとき ICON は 25000 の値をとる．
なお，本サブルーチンは，いずれの場合にも，積分の近似値 ***S*** のほかに，その絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

c. 使用例
積分

$$\int_{-\infty}^{\infty} \frac{1}{10^{-2} + x^2} dx$$

を計算する．

```
C     **EXAMPLE**
      EXTERNAL FUN
      EPSA=1.0E-5
      EPSR=0.0
      NMIN=20
      NMAX=645
      CALL AQEI(FUN,EPSA,EPSR,NMIN,NMAX,
     *S,ERR,N,ICON)
      WRITE(6,600) ICON,S,ERR,N
      STOP
  600 FORMAT(' ',30X,'ICON=',I5,5X,
     *'S=',E15.7,5X,'ERR=',E15.7,
     *5X,'N=',I5)
      END
      FUNCTION FUN(X)
      IF(ABS(X).GT.1.0E+35) GO TO 10
      IF(ABS(X).LT.1.0E-35) GO TO 20
      FUN=1.0/(1.0E-2+X*X)
      RETURN
   10 FUN=0.0
      RETURN
   20 FUN=100.0
      RETURN
      END
```

(4) 手法概要

本サブルーチンは高橋・森の二重指数関数型積分公式に基づく自動積分法を用いている．

この方法の原理及び，本サブルーチンの手順についてはサブルーチン AQE の手法概要を参照されたい．ただし, ここでは, 積分変数 x に対して, 変換,

$$x = \phi(t) = \sinh\left(\frac{3}{2}\sinh t\right)$$

を,したがって,重み関数 $\phi(t)$ に対して

$$\phi'(t) = \frac{3}{2}\cosh t \cdot \cosh\left(\frac{3}{2}\sinh t\right)$$

を採用している．

G24-13-0101 AQMC8, DAQMC8

多次元有限領域積分 (関数入力，クレンショー・カーチス型積分法)
CALL AQMC8(M,LSUB,FUN,EPSA,EPSR,NMIN, NMAX, S,ERR,N,ICON)

(1) 機能

m 重積分($1 \le m \le 3$) を次で定義する.

$$I = \int_{\varphi_1}^{\psi_1} dx_1 \int_{\varphi_2}^{\psi_2} dx_2 \cdots \int_{\varphi_m}^{\psi_m} dx_m \, f(x_1, x_2, \ldots, x_m) \tag{1.1}$$

ただし，積分の下限，上限は

$\varphi_1 = a$ (定数) ，$\psi_1 = b$ (定数)

$\varphi_2 = \varphi_2(x_1)$ ，$\psi_2 = \psi_2(x_1)$

…　　…

$\varphi_m = \varphi_m(x_1, x_2, \ldots, x_{m-1})$，$\psi_m = \psi_m(x_1, x_2, \ldots, x_{m-1})$

で与えられたものとする．このとき，与えられた ε_a, ε_r に対して

$$|S - I| \le \max(\varepsilon_a, \varepsilon_r |I|) \tag{1.2}$$

を満たす積分の近似値 S を，各次元に等差数列的に分点を増すクレンショー・カーチス型積分法を繰り返して適用することにより求める.

(2) パラメタ

M ……**入力**．積分の次数 m．$1 \le m \le 3$.

LSUB……**入力**．各積分変数の下限 φ_k 及び上限 ψ_k を計算するサブルーチン副プログラム名.
[副プログラムの用意のし方]
SUBROUTINE LSUB(K,X,A,B)
ここで,
K …… **入力**．座標軸の番号 k. $1 \le k \le m$.
X …… **入力**．X(1)=x_1, X(2)=x_2,..., X(M-1)=x_{m-1} なる対応を持つ大きさ (M-1) の 1 次元配列.
A …… **出力**．下限 $\varphi_k(x_1, x_2, \ldots, x_{k-1})$ の値.
B …… **出力**．上限 $\psi_k(x_1, x_2, \ldots, x_{k-1})$ の値.

FUN……**入力**．被積分関数 $f(x_1, x_2, \ldots, x_m)$ を計算する関数副プログラム名.
[副プログラムの用意のし方]
FUNCTION FUN(X)
ここでパラメタ X は，X(1)=x_1, X(2)=x_2, ..., X(M)=x_m なる対応を持つ大きさ M の 1 次元配列である.

EPSA ……**入力**．積分の近似値 S に対する絶対誤差の上限 ε_a (≥0.0).

EPSR ……**入力**．積分の近似値 S に対する相対誤差の上限 ε_r(≥ 0.0).

NMIN ……**入力**．各積分変数ごとに積分する際の被積分関数の計算回数の下限 (≥0)．標準値としては 7 が適当.

NMAX……**入力**．各積分変数ごとに積分する際の被積分関数の計算回数の上限 (NMAX ≥ NMIN).
標準値としては 511 が適当 (これよりも大きく指定した場合は 511 と見なす)．(使用上の注意④参照).

S ……**出力**．積分の近似値 (使用上の注意⑤参照).

ERR……**出力**．積分の近似値 S の絶対誤差の推定値.

N……**出力**．被積分関数の全体としての計算回数.
4 バイトの整数型変数であること.

ICON…… **出力**．コンディションコード.
表 AQMC8-1 参照.

表 AQMC8-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
100 1000 1100 10000 10100 11000 11100	ある座標軸方向で積分しているとき，丸め誤差のため，その軸方向の要求精度が得られなかった. 100 の位は x_3 軸方向の積分において，その現象が (x_1,x_2) の色々な組に対して起きたことを表す. 同様に 1000 の位は，x_2 軸方向の積分において，その現象が x_1 の色々な値に対して起きたことを表す. 10000 の位は，x_1 軸方向でその現象が起きたことを表す.	S には得られた近似値を出力する．100～1100 の場合は，精度は要求精度を満足しているか，達成可能な限度に達しているかいずれかである. 10000～11100 の場合は，精度は達成可能な限度に達している. いずれの場合にも，誤差は ERR に出力される.
200 2000 2200 20000 20200 22000 22200	ある座標軸方向で積分しているとき被積分関数の計算回数がその上限に達しても，その軸方向の要求精度が得られなかった. それぞれの位の意味は，上記と同じくどの軸方向で起きたかを表す.	S には得られた近似値を出力する．200～2200 の場合は要求精度を満足している場合もしていない場合もある．20000～22200 の場合は，要求精度を満足していない．いずれの場合にも，誤差は ERR に出力される. NMAX の値を大きく与えれば精度の改善の余地がある (NMAX=511 が限度).
300 〜 23300	コード 100～11100 と 200～22200 の現象が同時に起きたことを表す．それぞれの位の意味は，上記と同じくどの軸方向で起きたかを表す.	S には得られた近似値を出力する．精度は要求精度を満足している場合もしていない場合もある. いずれの場合にも，誤差は ERR に出力される.
30000	次のいずれかであった. ① EPSA < 0.0 ② EPSR < 0.0 ③ NMIN < 0 ④ NMAX < NMIN ⑤ M ≤ 0，又は M ≥ 4	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数 ... ABS, AMAX1, FLOAT, MAX0, MIN0, SQRT

b. 注意

① パラメタ FUN に対応する関数副プログラムは，積分変数ベクトル X だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムでその関数名を EXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図ること (使用例参照)．同様にパラメタ LSUB に対応する副プログラム名も EXTERNAL 文で宣言すること．

② 本サブルーチンを多数回呼ぶとき，最初だけ 511 個の定数 (積分公式の分点と重みの表) を計算し，2 回目以降は省略する．したがって，2 回目以降は計算量がその分だけ少ない．

③ 本サブルーチンは，被積分関数 $f(x_1,x_2,...,x_m)$ が滑らかな場合はもちろん，振動型の関数の場合にも有効に働く．

④ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数の各座標軸方向での計算回数 n_i (i=1,2,...,m) を

NMIN≤n_i≤NMAX

の範囲に限定している．すなわち収束判定にかかわりなく $f(x_1,x_2,...,x_m)$ は各座標軸方向に少なくとも NMIN 回計算され，しかも NMAX 回を超えて計算されることはない．もし，ある座標軸方向で NMAX 回以内に積分が収束しなかった場合は，ICON を 200～22200 の値とする．ここで，数字の 2 が現れる位置は，座標軸の区別を意味し，100, 1000, 及び 10000 の位の順に，x_3 軸，x_2 軸，及び x_1 軸と対応する．なお，NMAX が 7 未満の値であったときは，7 が与えられたものと見なす．

⑤ 積分の近似値 $\boldsymbol{S}$ の精度
本サブルーチンは，与えられたε_a , ε_rに対して，(1.2) を満たす S を求める．したがって，ε_r=0 とおけば，絶対誤差 ε_a以内の，またε_a=0 とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質とε_a , ε_rの値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べてε_a , ε_rが小さすぎる場合には，関数のある軸方向の計算回数がその上限 (NMAX) に達しない場合でも丸め誤差の影響の方が大きくなり，これ以上計算を続行しても無意味な状態を見出すことがある．このとき ICON を 100～11100 とする．ここで数字の 1 が現れる位置は，座標軸の区別を意味し，100, 1000, 及び 10000 の位の順に，x_3 軸，x_2 軸，及び x_1 軸と対応する．ICON が 100～1100 のとき，すなわち，x_3 軸，x_2 軸だけで丸め誤差の影響が大きくなったときでも，全体の積分値 S の精度にそれほど影響を及ぼさないことがあり，得られた積分値が要求精度を満足していることもある．しかし，この確認は，誤差の推定値 ERR に基づいて行われるべきである．
また，注意④で述べたように，ある座標軸方向の積分が NMAX 回以内の関数計算では収束しないこともある．このとき ICON は 200～22200 の値をとり，それぞれの 2 の位置は，座標軸の区別を意味する．ICON が 200～2200 のときは，得られた積分値が要求精度を満足していることもある．
更に，コード 100～11100 とコード 200～22200 の事象が同時に起きたときは，ICON を 300～23300 とする．
なお，本サブルーチンは，いずれの場合にも，積分の近似値 $\boldsymbol{S}$ のほかに，絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

c. 使用例
助変数 p を 1.0 から 3.0 まで，1.0 刻みに変えて，積分，

$$I=\int_{-1}^{1}dx\int_{-2}^{2}dy\int_{-3}^{3}dz\frac{1}{\cos(px)\cos(py)\cos(pz)+2}$$

を計算する．

```
C     **EXAMPLE**
      INTEGER*4 N
      COMMON P
      EXTERNAL FUN,LSUB
      EPSA=1.0E-5
      EPSR=1.0E-5
      NMIN=7
      NMAX=511
      M=3
      DO 10 I=1,3
      P=FLOAT(I)
      CALL AQMC8(M,LSUB,FUN,EPSA,EPSR,
     *NMIN,NMAX,S,ERR,N,ICON)
   10 WRITE(6,600) P,ICON,S,ERR,N
      STOP
  600 FORMAT(' ',30X,'P=',F6.1,5X,'ICON=',
     *I5,5X,'S=',E15.7,5X,'ERR=',E15.7,5X,
     *'N=',I5)
      END
      SUBROUTINE LSUB(K,X,A,B)
      DIMENSION X(2)
      GO TO (10,20,30),K
   10 A=-1.0
      B=1.0
      RETURN
   20 A=-2.0
      B=2.0
      RETURN
   30 A=-3.0
      B=3.0
      RETURN
      END

      FUNCTION FUN(X)
      DIMENSION X(3)
```

```
      COMMON P
      FUN=1.0/(COS(P*X(1))*COS(P*X(2))
     **COS(P*X(3))+2.0)
      RETURN
      END
```

(4) 手法概要

本サブルーチンは，等差数列的 (公差 8) に分点を追加する拡張された意味のクレンショー・カーチス型積分法 (サブルーチン AQC8 参照) を，それぞれの次元方向に繰り返す直積型多重積分法を用いている．AQC8 が滑らかな関数ばかりでなく，振動型の関数にも有効であるので，本サブルーチンも同様な性質を持つ．

クレンショー・カーチス型積分法については，AQC8 の「手法概要」を参照されたい．ここでは，多次元への応用についてだけ述べる．

a. 直積型多重積分法

3 重積分,

$$I = \int_{\varphi_1}^{\psi_1} dx_1 \int_{\varphi_2}^{\psi_2} dx_2 \int_{\varphi_3}^{\psi_3} dx_3 \, f(x_1, x_2, x_3) \tag{4.1}$$

について考える．ここで, 積分の下限, 上限は,

$\varphi_1=a$ ， $\psi_1=b$

$\varphi_2=\varphi_2(x_1)$ ， $\psi_2=\psi_2(x_1)$

$\varphi_3=\varphi_3(x_1,x_2)$ ， $\psi_3=\psi_3(x_1,x_2)$

で与えられたものとする．(4.1) を段階的に書くと次のようになる．

$$I_2(x_1, x_2) = \int_{\varphi_3}^{\psi_3} f(x_1, x_2, x_3)^{dx_3} \tag{4.2}$$

$$I_1(x_1) = \int_{\varphi_2}^{\psi_2} I_2(x_1, x_2)^{dx_2} \tag{4.3}$$

$$I = \int_{\varphi_1}^{\psi_1} I_1(x_1)^{dx_1} \tag{4.4}$$

したがって，I は関数 $I_1(x_1)$ を x_1 について積分したものであり，その場合，関数 $I_1(x_1)$ の値は，x_1 を固定して，関数 $I_2(x_1,x_2)$ を x_2 について $[\varphi_2(x_1),\psi_2(x_1)]$ 上で積分することにより得られる．更に関数 $I_2(x_1,x_2)$ は x_1 及び x_2 を固定して，$f(x_1,x_2,x_3)$ を x_3 について$[\varphi_3(x_1,x_2),\psi_3(x_1,x_2))$ 上で積分することにより得られる．

本サブルーチンは，x_1, x_2, 及び x_3 の各座標軸に沿って積分するのに，クレンショー・カーチス型積分法を用いるものである．

b. 計算の手順

手順 1…初期設定

諸変数の初期値といろいろな判定用の定数を定める．

手順 2…分点と重み係数の決定

クレンショー・カーチス型積分法で用いる分点と重みを，それぞれ漸化式によって作り，作業用 1 次元配列に入れる．分点の個数の上限が $511(=2^9-1)$ であるから，分点と重みのために必要な 1 次元配列の大きさはそれぞれ $256(=2^8)$ である．手順 2 は，本サブルーチンが初めて呼ばれたときだけ実行され，2 回目以降呼ばれるときは省略される．

以下の手順では 3 重積分の場合について述べる．

手順 3… x_1 軸方向の積分のための初期化

変数 x_1 の 積分区間 $[\varphi_1,\psi_1]$ 上に 7 点を定め，それらを x_1^i (i=1,2,3,...7) とする．これらは手順 2 で求めた分点の列から初めの 7 点を選び 1 次変換により区間 $[\varphi_1,\psi_1]$ に移したものである．

手順 4… x_2 軸方向の積分のための初期化

$x_1^i(i=1,2,...,7)$ の一つを，あるいは手順 8 からきた場合は x_1 軸上に追加された 8 個の分点の一つを X_1 とし，この X_1 に対して変数 x_2 の積分区間 $[\varphi_2(X_1),\psi_2(X_1)]$ 上に 7 点を定め，それらを x_2^i (i = 1, 2, ..., 7) とする．これらは手順 3 と同様に，手順 2 で求めた分点列の初めの 7 点に 1 次変換を施したものである．

手順 5… x_3 軸方向の積分のための初期化

$x_2^i(i=1,2,...,7)$ の一つ，あるいは手順 7 からきた場合は x_2 軸上に追加された 8 個の分点の一つを X_2 とし，(X_1,X_2) に対して変数 x_3 の積分区間 $[\varphi_3(X_1,X_2),\psi_3(X_1,X_2)]$ 上に 7 点を定めそれらを $x_3^i(i=1,2,...,7)$ とする．これらは手順 3 と同様に，手順 2 で求めた分点列の初めの 7 点に 1 次変換を施したものである．

手順 6… x_3 軸方向の積分

分点 (X_1,X_2,x_3^i) (i=1,2,...,7) 上で，関数 f の値を計算し，x_3 軸方向の積分の近似値 $SI_7(X_1,X_2)$ を求める．これは，(x_1,x_2) を固定したときの (4.2) に対する初期近似である．更に，手順 2 で求めた分点列を基に x_3^i (i=8,9,...,15) を定めて，(X_1,X_3,x_3^i) (i=8,9,...15) 上で関数 f の値を計算し，15 点による (4.2) の近似値 $SI_{15}(X_1,X_2)$ を求める．以下同様に 8 点ずつ追加して収束するまで繰り返す．もし NMAX 個の分点を用いても収束しないか，又は誤差が丸め誤差程度になってそれ以上改良できなくなった場合はそれに応じて ICON の値を設定しておく．

手順 7… x_2 軸方向の積分

手順 4 で定めた x_2^i (i=1,2,...,7) のすべてについて手順 5, 6 を実行し，x_2 軸方向の積分の近似値 $SI_7(X_1)$ を求める．これは x_1 を固定したときの (4.3) に対する初期近似である．更に手順 2 で求めた分点列を基に x_2^i (i=8,9,...,15) を定め，手順 5, 6 を実行し，$SI_{15}(X_1)$ を求める．以下同様に 8 点

ずつ追加して収束するまで繰り返す．収束しない場合，ICON の設定を行う．

手順 8… x_1 軸方向の積分

手順 3 で定めた $x_1^{\,i}$ (i=1,2,...,7) のすべてについて手順 4, 5, 6, 7 を実行し，x_1 軸方向の積分の近似値 SI_7 を求める．これは (4.4)，すなわち，目的の 3 重積分に対する初期近似である．更に手順 2 で求めた分点列を基に $x_1^{\,i}$ (i=8,9,...,15) を定め，手順 4, 5, 6, 7 を実行し，SI_{15} を求める.以下同様に 8 点ずつ追加して収束するまで繰り返す．収束しない場合，ICON の設定を行い処理を打ち切る．

c. 誤差評価

3 重積分 (4.1) の場合について述べる．x_1 についての積分区間 $[\varphi_1, \psi_1]$ を [-1,1] に移すため，変数変換

$$x_1 = \frac{\psi_1 - \varphi_1}{2} x_1' + \frac{\psi_1 + \varphi_1}{2} \equiv \alpha_1 x_1' + \beta_1$$

を行う．同様に x_2 , x_3 について積分区間 $[\varphi_2(x_1), \psi_2(x_1)]$, $[\varphi_3(x_1,x_2), \psi_3(x_1,x_2)]$ を [-1,1] へ移すため，それぞれ変数変換

$$\begin{aligned} x_2 &= \frac{\psi_2(x_1) - \varphi_2(x_1)}{2} x_2' + \frac{\psi_2(x_1) + \varphi_2(x_1)}{2} \\ &\equiv \alpha_2(x_1')x_2' + \beta_2(x_1') \\ x_3 &= \frac{\psi_3(x_1,x_2) - \varphi_3(x_1,x_2)}{2} x_3' + \frac{\psi_3(x_1,x_2) + \varphi_3(x_1,x_2)}{2} \\ &\equiv \alpha_3(x_1',x_2')x_3' + \beta_3(x_1',x_2') \end{aligned}$$

を行う．すると積分 (4.1) は次のようになる．

$$I = \alpha_1 \int_{-1}^{1} dx_1' h(x_1') \tag{4.5}$$

$$h(x_1') = \alpha_2(x_1') \int_{-1}^{1} dx_2' g(x_1', x_2') \tag{4.6}$$

$$\begin{aligned} g(x_1', x_2') = \alpha_3(x_1', x_2') \int_{-1}^{1} dx_3' f(&\alpha_1 x_1' + \beta_1, \alpha_2(x_1')x_2' \\ &+ \beta_2(x_1'), \alpha_3(x_1', x_2')x_3' + \beta_3(x_1', x_2')) \end{aligned} \tag{4.7}$$

(4.7) の右辺の [-1.1] 上の積分に対してクレンショー・カーチス則を適用し，そのときの打ち切り誤差を $p(x_1', x_2')$ とすれば，

$$g(x_1', x_2') = \tilde{g}(x_1', x_2') + \alpha_3(x_1', x_2') p(x_1', x_2') \tag{4.8}$$

となる．ここで，$\tilde{g}(x_1', x_2')$ は，

$$\tilde{g}(x_1', x_2') = \alpha_3(x_1', x_2') \left[\sum_{k=0}^{7} A_{-1,k} W_{-1,k} + \sum_{i=0}^{P} \sum_{k=0}^{7} A_{i,k} W_{i,k} \right] \tag{4.9}$$

であり，(8P+7) 個の分点に基づく近似値であって，$A_{i,k}$ は (x_1', x_2') に依存する値，$W_{i,k}$ は重み係数である．

$p(x_1', x_2')$ は

$$\left| p(x_1', x_2') \right| = \left(\left| A_{P,5} \right| + \left| A_{P,7} \right| \right) \cdot \left| W_{P+1,1} \right| \tag{4.10}$$

で評価する．

次に，(4.8) を (4.6) に代入して，

$$\begin{aligned} h(x_1') = \alpha_2(x_1') &\int_{-1}^{1} \tilde{g}(x_1', x_2') dx_2' \\ &+ \alpha_2(x_1') \int_{-1}^{1} \alpha_3(x_1', x_2') p(x_1', x_2') dx_2' \end{aligned} \tag{4.11}$$

を得る．右辺第 1 項の [-1,1] 上の積分にクレンショー・カーチス則を適用し，そのときの打ち切り誤差を $q(x_1')$ とすれば，第 1 項は

$$\left.\begin{aligned} &\alpha_2(x_1') \int_{-1}^{1} \tilde{g}(x_1', x_2') dx_2' = \tilde{h}(x_1') + \alpha_2(x_1') q(x_1') \\ &\text{ここで} \\ &\tilde{h}(x_1') = \alpha_2(x_1') \left[\sum_{k=0}^{7} B_{-1,k} W_{-1,k} + \sum_{i=0}^{Q} \sum_{k=0}^{7} B_{i,k} W_{i,k} \right] \end{aligned}\right\} \tag{4.12}$$

となる．

$q(x_1')$ は，(4.10) と同様に，

$$\left| q(x_1') \right| = \left(\left| B_{Q,5} \right| + \left| B_{Q,7} \right| \right) \cdot \left| W_{Q+1,1} \right| \tag{4.13}$$

で評価する．

(4.12) の第 1 式を (4.11) に代入し，得られた式を (4.5) に代入すれば，

$$\begin{aligned} I = \alpha_1 &\int_{-1}^{1} \tilde{h}(x_1') dx_1' + \alpha_1 \int_{-1}^{1} \alpha_2(x_1') q(x_1') dx_1' \\ &+ \alpha_1 \int_{-1}^{1} \alpha_2(x_1') \int_{-1}^{1} \alpha_3(x_1', x_2') p(x_1', x_2') dx_2' dx_1' \end{aligned} \tag{4.14}$$

を得る．(4.14) の右辺第 1 項の [-1,1] 上の積分にクレンショー・カーチス則を適用し，その時の打ち切り誤差 r とすれば，

$$\left.\begin{aligned}&\alpha_1\int_{-1}^{1}\tilde{h}(x_1')dx_1' = \tilde{I} + \alpha_1 r\\ &\text{ここで}\\ &\tilde{I} = \alpha_1\left[\sum_{k=0}^{7} C_{-1,k}W_{-1,k} + \sum_{i=0}^{R}\sum_{k=0}^{7} C_{i,k}W_{i,k}\right]\end{aligned}\right\} \qquad (4.15)$$

となる．r は

$$|r| = \left(\left|C_{R,5}\right| + \left|C_{R,7}\right|\right)\cdot\left|W_{R+1,1}\right| \qquad (4.16)$$

で評価する．したがって，(4.15) の $\tilde{I}$ が 3 重積分の近似値であり，誤差 $\left|I-\tilde{I}\right|$ は，(4.15) の第 1 式を (4.14) に代入して，

$$\begin{aligned}\left|I-\tilde{I}\right| &\le \left|\alpha_1 r\right| + \left|\alpha_1\int_{-1}^{1}\alpha_2(x_1')q(x_1')dx_1'\right|\\ &+ \left|\alpha_1\int_{-1}^{1}\alpha_2(x_1')\int_{-1}^{1}\alpha_3(x_1',x_2')\,p(x_1',x_2')dx_2'dx_1'\right|\end{aligned} \qquad (4.17)$$

となる．これは，打ち切り誤差 $p(x'_1, x'_2)$, $q(x'_1)$, 及び r が，全体の誤差にいかに影響を与えるかを示すものである．さて，$p(x'_1, x'_2)$, $q(x'_1)$ 及び r の推定値は (4.10), (4.13) 及び (4.16) であるが，これは各座標軸方向での積分の収束が速い場合に成立する．積分の収束が遅い場合の対策として，もう一つ別の推定法を考える．$2^n - 1$ 個の分点を用いてクレンショー・カーチス則を適用したときの積分の近似値を I_{2^n-1} とおくと，$|I_{2^n-1} - I_{2^{n-1}-1}|$ は $2^{n-1} - 1$ 個の分点を用いたクレンショー・カーチス則による積分の近似値に対する誤差評価になりうる．(4.10), (4.13) 及び (4.16) のような展開係数と重み係数を用いた，$2^{n-1} - 1$ 個の分点によるクレンショー・カーチス則の誤差評価式を $E(2^{n-1} - 1)$ とおくと，これは積分の収束が遅いとき過小評価となるので，

$$\left|I_{2^n-1} - I_{2^{n-1}-1}\right| > E\left(2^{n-1}-1\right) \qquad (4.18)$$

が成立する．そこで (4.18) が成立した場合，$2^n - 1 \le 8P+7 < 2^{n+1} - 1$ の P に対して $E(8P+7)$ の代りに，

$$E(8P+7)\cdot\left|I_{2^n-1} - I_{2^{n-1}-1}\right| \Big/ E\left(2^{n-1}-1\right) \qquad (4.19)$$

を誤差評価式として用いる．ここで，I_{2^n-1}, $I_{2^{n-1}-1}$ は，それぞれ 2^n-1 , $2^{n-1} - 1$ 個の分点を用いたときの $\tilde{g}(x'_1, x'_2)$, $\tilde{h}(x'_1)$ 及び $\tilde{I}$ に対応する．

d. 収束判定基準

(4.17) において，$\left|I-\tilde{I}\right| \le \varepsilon_a$ (絶対誤差) とするためには，(4.17) の右辺の各項が，

$$\left|\alpha_1 r\right| \le \varepsilon_a/3$$
$$\left|\alpha_1\int_{-1}^{1}\alpha_2(x_1')q(x_1')dx_1'\right| \le \varepsilon_a/3$$
$$\left|\alpha_1\int_{-1}^{1}\alpha_2(x_1')\int_{-1}^{1}\alpha_3(x_1',x_2')\,p(x_1',x_2')dx_2'dx_1'\right| \le \varepsilon_a/3$$

であればよい．このためには，

$$\left.\begin{aligned}&|r| \le \frac{\varepsilon_a}{3\alpha_1} \equiv \varepsilon_a^{(1)}\\ &\left|q(x_1')\right| \le \frac{\varepsilon_a}{3\times 2\alpha_1\alpha_2(x_1')} \equiv \varepsilon_a^{(2)}\\ &\left|p(x_1',x_2')\right| \le \frac{\varepsilon_a}{3\times 2\times 2\alpha_1\alpha_2(x_1')\alpha_3(x_1',x_2')} \equiv \varepsilon_a^{(3)}\end{aligned}\right\} \qquad (4.20)$$

であれば十分である．ここで $\alpha_1>0$, $\alpha_2(x'_1)>0$, $\alpha_3(x'_1,x'_2)>0$ と仮定した．

一方，計算の各段階,すなわち $\tilde{g}(x'_1,x'_2)/\alpha_3(x'_1,x'_2)$, $\tilde{h}(x'_1)/\alpha_2(x'_1)$ 及び $\tilde{I}/\alpha_1$ を求めるときの計算誤差の上限 $\rho^{(3)}(x'_1,x'_2)$, $\rho^{(2)}(x_1)$ 及び $\rho^{(1)}$ は，丸め誤差の単位 u と P，Q 及び R に依存して，

$$\left.\begin{aligned}\rho^{(3)}(x'_1,x'_2) &= 8u(P+1)\left\|f\right\|_\infty\\ \rho^{(2)}(x'_1) &= 16u(Q+1)\left\|\tilde{g}\right\|_\infty\\ \rho^{(1)} &= 32u(R+1)\left\|\tilde{h}\right\|_\infty\end{aligned}\right\} \qquad (4.21)$$

と推定している．ここで，

$$\left\|f\right\|_\infty = \max_{x_3}\left|f(\alpha_1 x'_1+\beta_1, \alpha_2 x'_2+\beta_2, \alpha_3 x'_3+\beta_3)\right|$$
$$\left\|\tilde{g}\right\|_\infty = \max_{x_2}\left|\tilde{g}(x'_1,x'_2)\right|$$
$$\left\|\tilde{h}\right\|_\infty = \max_{x_1}\left|\tilde{h}(x'_1)\right|$$

である．以上をまとめて，各段階の収束判定定数を

$$\tau^{(3)}(x'_1,x'_2) = \max(\varepsilon_a^{(3)}, \varepsilon_r\left|\tilde{g}(x'_1,x'_2)/\alpha_3(x'_1,x'_2)\right|, \rho^{(3)}(x'_1,x'_2))$$
$$\tau^{(2)}(x'_1) = \max(\varepsilon_a^{(2)}, \varepsilon_r\left|\tilde{h}(x'_1)/\alpha_2(x'_1)\right|, \rho^{(2)}(x'_1))$$
$$\tau^{(1)} = \max(\varepsilon_a^{(1)}, \varepsilon_r\left|\tilde{I}/\alpha_1\right|, \rho^{(1)})$$

と定める．ここで，右辺の $\tilde{g}$, $\tilde{h}$, $\tilde{I}$ は，それまでに得られている近似値である．さて収束判定は次のように行う．

$$\left|p\left(x_1', x_2'\right)\right| \leq \tau^{(3)}\left(x_1', x_2'\right) \tag{4.22}$$

を満足したとき $\tilde{g}\left(x_1', x_2'\right)$ は収束したとする．次に，

$$\left|q\left(x_1'\right)\right| \leq \tau^{(2)}\left(x_1'\right) \tag{4.23}$$

を満足したとき $\tilde{h}\left(x_1'\right)$ は収束したとする．更に，

$$|r| \leq \tau^{(1)} \tag{4.24}$$

を満足したとき，そのとき得られている $\tilde{I}$ を，多重積分の近似値としてパラメタ S に出力する．

G24-13-0201 AQME, DAQME

多次元積分 (関数入力，二重指数関数型積分公式)
CALL AQME(M,INT,LSUB,FUN,EPSA,EPSR, NMIN,NMAX,S,ERR,N,ISF,ICON)

(1) 機能

m 重積分 $(1 \le m \le 3)$ を次で定義する．

$$I = \int_{\varphi_1}^{\psi_1} dx_1 \int_{\varphi_2}^{\psi_2} dx_2 \cdots \int_{\varphi_m}^{\psi_m} dx_m \, f(x_1, x_2, \cdots x_m) \qquad (1.1)$$

ただし，積分の下限，上限は，一般に

$$\varphi_1 = a(\text{定数}) \quad , \psi_1 = b(\text{定数})$$
$$\varphi_2 = \varphi_2(x_1) \quad , \psi_2 = \psi_2(x_1)$$
$$\cdots \qquad\qquad \cdots$$
$$\varphi_m = \varphi_m(x_1, x_2, \ldots, x_{m-1}), \psi_m = \psi_m(x_1, x_2, \ldots, x_{m-1})$$

で与えられるが，本サブルーチンは，このような有限領域のほかに，半無限，あるいは全無限領域も扱う．
すなわち，各積分変数 x_k についての積分区間 $[\varphi_k, \psi_k]$ は独立に
[0,∞] あるいは (–∞,∞)

であってよいものとする．

以上の条件のもとに，本サブルーチンは，与えられたε_a, ε_rに対して

$$|S - I| \le \max(\varepsilon_a, \varepsilon_r |I|) \qquad (1.2)$$

を満たす積分の近似値 S を，高橋・森の 2 重指数関数型積分公式を繰り返し適用することにより求める．

(2) パラメタ

M …… **入力**．積分の次元数 m.

INT …… **入力**．各積分変数についての積分区間の種類を表す情報．大きさ M の 1 次元配列．k 番目の要素 INT(K) は，k 番目の変数 x_k についての積分区間の種類を表すものとし，以下の約束にしたがって，1, 2, 3 のいずれかの整数を与える．
1 ... 有限区間
2 ... 半無限区間
3 ... 全無限区間
例えば，3 重積分

$$I = \int_0^{\infty} dx_1 \int_0^{\pi} dx_2 \int_0^{2\pi} dx_3 f(x_1, x_2, x_3)$$

の場合は INT(1)=2, INT(2)=1, INT(3)=1 と与える．

LSUB …… **入力**．各積分変数の下限 φ_k 及び ψ_k を計算するサブルーチン副プログラム名．
[副プログラムの用意のし方]
SUBROUTINE LSUB(K,X,A,B)
ここで，
K……**入力**．座標軸の番号 k. $1 \le k \le m$.
X……**入力**．X(1)=x_1,X(2)=x_2,....., X(M-1) =x_{m-1} なる対応を持つ大きさ (M-1) の 1 次元配列.
A……**出力**．下限 $\varphi_k(x_1,x_2,...,x_{k-1})$ の値
B……**出力**．上限 $\psi_k(x_1,x_2,...,x_{k-1})$ の値
ただし，区間 $[\varphi_k,\psi_k]$ が [0,∞) あるいは (–∞,∞) の場合は，その k については，A, B に何の値も設定する必要はない．

FUN…… **入力**．被積分関数 $f(x_1,x_2,...,x_m)$ を計算する関数副プログラム名．
[副プログラムの用意のし方]
FUNCTION FUN(X)
ここで，パラメタ X は，X(1)=x_1, X(2)=x_2,...,X(M)=x_m なる対応を持つ大きさ M の 1 次元配列.

EPSA…… **入力**．積分の近似値 S に対する絶対誤差の上限 ε_a (≥0.0).

EPSR …… **入力**．積分の近似値 S に対する相対誤差の上限 ε_r (≥0.0).

NMIN …… **入力**．各積分変数ごとに積分する際の被積分関数の計算回数の下限 (≥0).
標準値としては 20 が適当.

NMAX … **入力**．各積分変数ごとに積分する際の被積分関数の計算回数の上限(NMAX≥NMIN)
標準値としては 705 が適当．(これよりも大きく指定した場合は 705 と見なす).
(使用上の注意⑤を参照).

S …… **出力**．積分の近似値 (使用上の注意⑥を参照)

ERR…..**出力**．積分の近似値 S の絶対誤差の推定値.

N……**出力**．被積分関数の全体としての計算回数.
4 バイトの整数型変数であること.

ISF…… **出力**．ICON の出力時の値が 25000 台であったとき，被積分関数の悪条件な振舞についての情報.
ISF は 10 進 3 桁の正整数であり，いま，

$$ISF = 100j_1 + 10j_2 + j_3$$

と置くと，j_3, j_2, j_1 はそれぞれ x_3 軸，x_2 軸，及び，x_1 軸方向における被積分関数の振舞を表す．各 j_i は独立に，1, 2, 3 又は 0 のいずれかをとり，その意味は次のとおりである．

1 ····· 積分区間の下限の付近で関数値が急激に増大しているか，あるいは，区間が無限区間であるならば，$x_i \to -\infty$のとき関数値の 0 への収束が遅い.

2 ····· 積分区間の下限の付近で関数値が急激に増大しているか，あるいは，区間が半無限もしくは無限区間であるならば，$x_i \to +\infty$のとき関数値の 0 への収束が遅い.

3 ····· 上記 1, 2 の事象が共に生じた.

0 ····· 上記の事象は生じなかった.

ICON·····**出力**．コンディションコード.
表 AQME-1 参照.

表 AQME-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10001 | 10077	x_3軸方向，又は x_2軸方向の積分において以下に示す事象が起きた. 1 の位は x_3軸方向，10 の位は x_2軸方向の事象を表し，各々独立に 0～7 の値をとる．(ただし，共に 0 をとることはない). 1. 丸め誤差のため，その軸方向の要求精度が得られなかった. 2. その軸方向での被積分関数の計算回数が上限 (NMAX) に達しても要求精度が得られなかった. 4. サブルーチン内で定める最小の刻み幅で積分しても，その軸方向の要求精度が得られなかった. 3. 上記 1, 2 の事象が共に起きた. 5. 上記 1, 4 の事象が共に起きた. 6. 上記 2, 4 の事象が共に起きた. 7. 上記 1, 2, 4 の事象が共に起きた. 0. 上記のどの事象も起きなかった.	S には得られた近似値を出力する．精度は要求精度を満たしていることもある.
10100 | 10177	x_1軸方向の積分において，丸め誤差のため要求精度が得られなかった. 下位 2 桁の意味は上記と同じである.	S には得られた近似値を出力する．精度は達成可能な限度に達している.
20200 | 20277	x_1軸方向の積分において，被積分関数の計算回数が上限(NMAX) に達しても要求精度が得られなかった. 下位 2 桁の意味は上記と同じである.	S には得られた近似値を出力するが，精度は保証できない. NMAX の値を大きく与えば，精度の改善の余地がある. (NMAX=705 が限度).
20400 | 20477	x_1軸方向の積分において，サブルーチン内で定める最小の刻み幅で積分しても要求精度が得られなかった. 下位 2 桁の意味は上記と同じである.	S には得られた近似値を出力する.
25000 | 25477	ある座標軸方向で積分しているとき，積分区間の下限若しくは上限の付近で関数値が急激に増大している. あるいは，積分区間が半無限若しくは無限区間の場合，積分変数が無限遠点へ進むときの被積分関数の 0 への収束が遅い. 下位 3 桁の意味は上記と同じである.	要求精度をゆるめて処理を続ける．S には得られた近似値を出力する．与えられた積分が，理論的な意味で積分値が存在しない (積分可能でない) 場合も，この範囲のコードが立つので注意すること. なお，被積分関数のこのような振舞についての詳細はパラメタ ISF に出力するので必要ならば参照すること.
30000	次のいずれかであった. ① EPSA<0.0 ② EPSR<0.0 ③ NMIN<0 ④ NMAX<NMIN ⑤ M≤0 or M≥4 ⑥ INT のある要素に，1, 2, 3 以外の値が入力された.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH,AFMAX,AFMIN, MGSSL UAQE1,UAQE2,UAQE3,UFN10,UFN20,UFN30, UFN40

② FORTRAN 基本関数 ... ABS,ALOG,SQRT, EXP,FLOAT,COSH,SINH,MAX0,AMAX1, AMIN1,MOD

b. 注意

① パラメタ FUN に対応する関数副プログラムは，積分変数ベクトル X だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムでその関数名を ETERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより呼び出す側のプログラムとの連絡を図ること.

② 本サブルーチンを多数回呼ぶとき，最初だけ定数表(積分公式の分点と重みの表)を計算し，2 回目以降は省略する．したがって，2 回目以降は計算量がその分だけ少ない.

③ 本サブルーチンは，積分領域の境界付近で変化が激しい場合に，特に有効に働き，代数特異点及び対数特異点が境界面上に位置する場合は，本サブルーチンの使用を第一に勧める．積分領域が有限で，被積分関数が滑らかな関数及び振動型の関数の場合にはサブルーチン AQMC8 を用いた方がよい．
積分領域が無限の場合で，被積分関数 $f(x_1,x_2,...,x_m)$ が $x_i \to \pm\infty$ において，0 への収束が比較的遅いものに対しても有効に働く．ただし，領域内部で激しく振動する場合には，高精度な積分値を得ることができない場合がある．

④ 本サブルーチンは，積分領域の境界面上では関数を計算しない．したがって，その面上で関数値が無限大となってもかまわない．ただし，領域内に無限大となる点があってはならない．
ある座標軸方向 (i 番目の座標軸とする) の積分区間が無限である場合，$|x_i|$ の大きな点での関数値 $f(x_1,x_2,...,x_m)$ を必要とするので，殊に高い要求精度を与えた場合には，関数副プログラム FUN の中では，オーバフロー，アンダフローに対する処置が必要である．

⑤ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数の各座標軸方向での計算回数 $n_i (i = 1, 2, ..., m)$ を
$\mathrm{NMIN} \le n_i \le \mathrm{NMAX}$

の範囲に限定している．すなわち収束判定にかかわりなく $f(x_1,x_2,...,x_m)$ は各座標軸方向に少なくとも NMIN 回計算され，しかも NMAX 回を超えて計算されることはない．もし，ある座標軸方向で NMAX 回以内に積分が収束しなかった場合は，その軸が x_1 軸，x_2 軸，x_3 軸のいずれであるかに応じて ICON の 100, 10, 1 の位にその情報を出力する．
なお，NMAX が極端に小さく，例えば NMAX=2 と与えられたときは，被積分関数の挙動に応じて決まる一定の値まで自動的に引き上げている．

⑥ 積分の近似値 S の精度
本サブルーチンは，与えられた ε_a , ε_r に対して，(1.2) を満たす S を求める．したがって ε_r=0 とおけば，絶対誤差 ε_a 以内の，また ε_a=0 とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質と ε_a , ε_r の値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べて ε_a , ε_r が小さすぎる場合には，関数のある軸方向の計算回数がその上限 (NMAX) に達しない場合でも，丸め誤差の影響の方が大きくなり，それ以上計算を実行しても無意味な状態を見出すことがある．
このとき軸の種類に応じて ICON の 100, 10, 1 の位にその情報を出力する．一般に，x_3 軸，x_2 軸だけで丸め誤差の影響が大きくなったときでも，全体の積分値 S の精度にそれほど影響を及ぼさないことがあり，得られた積分値が要求精度を満足していることもある．しかし，この確認は，誤差の推定値 ERR に基づいて行われるべきである．
また，注意⑤でも述べたように，ある座標軸方向の積分が NMAX 回以内の関数計算では収束しないこともある．このときも，ICON に情報を出力するが，x_3 軸，x_2 軸だけでこの事象が起きたときでも，得られた積分値が要求精度を満足していることもある．
また，サブルーチン内で定める最小の刻み幅で積分しても収束しないことがある．これについても，ICON に出力するが，以前と同様に x_3 軸，x_2 軸だけでこの事象が起きても，要求精度を満足していることもある．
本サブルーチンは，いずれの場合も，積分の近似値 S のほかに，絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

c. 使用例
積分

$$I = \int_0^{\infty} dx_1 \int_0^{x_1} dx_2 \int_0^{1-x_2} \frac{e^{-x_1}}{x_1 \sqrt{x_2 + x_3}} dx_3$$

を計算する．

```
C       **EXAMPLE**
        INTEGER*4 N
        EXTERNAL FUN,LSUB
        DIMENSION INT(3)
        INT(1)=2
        INT(2)=1
        INT(3)=1
        EPSA=1.0E-3
        EPSR=1.0E-3
        NMIN=20
        NMAX=705
        M=3
        CALL AQME(M,INT,LSUB,FUN,EPSA,EPSR,
       *      NMIN,NMAX,S,ERR,N,ISF,ICON)
        WRITE(6,600) ICON,S,ERR,N
        IF(ICON.GE.25000) WRITE(6,610) ISF
        STOP
  600 FORMAT(' ',10X,'ICON=',I6,5X,'S=',
     *E15.7,5X,'ERR=',E15.7,5X,'N=',I6)
  610 FORMAT(' ',20X,'ISF=',I3)
        END
        SUBROUTINE LSUB(K,X,A,B)
        DIMENSION X(2)
        A=0.0
        GO TO (10,20,30),K
   10 RETURN
   20 B=X(1)
        RETURN
   30 B=1.0-X(2)
        RETURN
        END
        FUNCTION FUN(X)
        DIMENSION X(3)
        Y=X(2)+X(3)
        IF(Y.LT.1.0E-30) GO TO 20
        IF(X(1).GT.80.0) GO TO 20
```

```
      Y=X(1)*SQRT(Y)
      IF(Y.LT.1.0E-30) GO TO 20
      FUN=EXP(-X(1))/Y
      RETURN
   20 FUN=0.0
      RETURN
      END
```

(4) 手法概要

本サブルーチンは，高橋，森の二重指数関数型積分公式に基づく自動積分法を，それぞれの軸方向に繰り返す直積型多重積分法を用いている．積分変数 x_k についての積分区間が有限の場合は，サブルーチン AQE，半無限の場合は，AQEH，全無限の場合は，AQEI に用いた手法をそのまま採用しているので，各々のサブルーチンの「手法概要」を参照されたい．

以下では，二重指数関数型積分公式の多重積分への適用法と計算手順について述べる．

a. 二重指数関数型積分公式の多重積分への適用

式 (1.1) で定義される m 重積分において，m=3 の場合を考える．積分

$$I=\int_{\varphi_1}^{\psi_1}dx_1\int_{\varphi_2}^{\psi_2}dx_2\int_{\varphi_3}^{\psi_3}dx_3\,f(x_1,x_2,x_3) \tag{4.1}$$

において，各積分変数に，変数変換

$$x_k=\phi_k(t_k)\ ,\ k=1,2,3 \tag{4.2}$$

を行って，区間 $[\varphi_k,\psi_k]$ を $(-\infty,\infty)$ に変換すると，(4.1) の積分は，

$$I=\int_{-\infty}^{\infty}dt_1\int_{-\infty}^{\infty}dt_2\int_{-\infty}^{\infty}dt_3\,f(\phi_1(t_1),\phi_2(t_2),\phi_3(t_3))\times\phi_1'(t_1)\phi_2'(t_2)\phi_3'(t_3) \tag{4.3}$$

となる．(4.3) を段階的に書くと次のようになる．

$$I_2(\phi_1(t_1),\phi_2(t_2))=\int_{-\infty}^{\infty}f(\phi_1(t_1),\phi_2(t_2),\phi_3(t_3))\phi_3'(t_3)dt \tag{4.4}$$

$$I_1(\phi_1(t_1))=\int_{-\infty}^{\infty}I_2(\phi_1(t_1),\phi_2(t_2))\phi_2'(t_2)dt_2 \tag{4.5}$$

$$I_0=\int_{-\infty}^{\infty}I_1(\phi_1(t_1))\phi_1'(t_1)dt_1\equiv I \tag{4.6}$$

したがって，I は関数 $I_1(\phi_1(t_1))\phi_1'(t_1)$ を t_1 について無限区間で積分したものであり，その場合 $I_1(\phi_1(t_1))$ の値は，t_1 を固定して，関数 $I_2(\phi_1(t_1),\phi_2(t_2))\phi_2'(t_2)$ を t_2 について積分することにより得られる．更に，関数 $I_2(\phi_1(t_1),\phi_2(t_2))$ は t_1, t_2 を固定して $f(\phi_1(t_1),\phi_2(t_2),\phi_3(t_3))\times\phi_3'(t_3)$ を t_3 について積分することにより得られる．

ここで，式 (4.4), (4.5), (4.6) の無限積分 $I_{k-1}(k=1,2,3)$ を求めるのに台形則を適用すると，

$$I_{k-1}=h_k\sum_{n_k=-\infty}^{\infty}I_k(\phi_1(n_1h_1),\ldots,\phi_k(n_kh_k))\phi_k'(n_kh_k)\ ,\quad k=1,2,3 \tag{4.7}$$

なる積分公式が得られる．ただし，h_k は t_k についての等間隔刻み幅である．

なお，本サブルーチンでは，式 (4.2) の変数変換として，

① φ_k=A, ψ_k=B のとき (ただし，A, B は有限値)

$$\phi_k(t_k)=\frac{B-A}{2}\tanh\left(\frac{3}{2}\sinh t_k\right)+\frac{A+B}{2} \tag{4.8}$$

② φ_k=0, $\psi_k\to\infty$のとき，

$$\phi_k(t_k)=\exp\left(\frac{3}{2}\sinh t_k\right) \tag{4.9}$$

③ $\varphi_k\to-\infty$, $\psi_k\to\infty$のとき，

$$\phi_k(t_k)=\sinh\left(\frac{3}{2}\sinh t_k\right) \tag{4.10}$$

を採用する．

b. 多重積分の収束判定

(4.4), (4.5), (4.6) の各 I_k に対して，

$$|S_2-I_2|\le\max(\varepsilon_a,\varepsilon_r|I_2|) \tag{4.11}$$

$$|S_1-I_1|\le\max(\varepsilon_a,\varepsilon_r|I_1|) \tag{4.12}$$

$$|S_0-I_0|\le\max(\varepsilon_a,\varepsilon_r|I_0|) \tag{4.13}$$

を満たす S_k を段階的に求めて，最後の S_0 を式 (1.2)を満たす S としている．

等間隔刻み台形則における刻み幅の設定，収束判定基準，無限和 (4.7) を有限和で近似するときのしきい値の設定，丸め誤差の影響の検出法については，サブルーチン AQE の「手法概要」を参照されたい．

c. 計算の手順

手順 1…初期設定

諸変数の初期値といろいろな判定用の定数を定める．

手順 2…分点と重み係数の計算

INT の値に従って，積分変数の変換を行い，分点 ($\phi_k(t_k)$) と重み ($\phi_k'(t_k)$) の値を計算する．

分点と重みの個数は 2×352×3 である．

この計算は，本サブルーチンが初めて呼ばれたときだけ実行され，2 回目以降呼ばれるときは省略される．

手順 3$\cdots x_1$ 軸方向の積分 ― I_0 の計算

(4.6) の I_0 に対する近似値 S_0 を計算する．収束判定は (4.13) で行う．I_0 の計算において必要となる $I_1(\phi_1(t_1))$ の値は手順 4 で計算される．求った S_0 と，その誤差の推定値は，パラメタ S，及び ERR に出力される．

手順 4$\cdots x_2$ 軸方向の積分 ― $I_1(\phi_1(t_1))$ の計算

(4.5) の I_1 $(\phi_1(t_1))$ に対する近似値 S_1 を計算する．収束判定は (4.12) で行う．$I_1(\phi_1(t_1))$ の値は手順 5 で計算される．

手順 5$\cdots x_3$ 軸方向の積分 ―$I_2(\phi_1(t_1),\phi_2(t_2))$ の計算

(4.4) の I_2 $(\phi_1(t_1), \phi_2(t_2))$ に対する近似値 S_2 を計算する．収束判定は (4.11) で行う．

G23-11-0201 AQN9, DAQN9

1 次元有限区間積分 (関数入力，適応型ニュートン・コーツ 9 点則)
CALL AQN9(A,B,FUN,EPSA,EPSR,NMIN,NMAX, S,ERR,N, ICON)

(1) 機能

関数 $f(x)$ 及び $a, b, \varepsilon_a, \varepsilon_r$ が与えられたとき，

$$\left|S-\int_a^b f(x)dx\right| \le \max\left(\varepsilon_a, \varepsilon_r \cdot \left|\int_a^b f(x)dx\right|\right) \qquad (1.1)$$

を満たす積分の近似値 S を，ニュートン・コーツ 9 点則に基づく適応型手法により求める．

(2) パラメタ

A…… **入力**．積分区間の下限 a．
B…… **入力**．積分区間の上限 b．(b≤a でも可)．
FUN…… **入力**．被積分関数 $f(x)$ を計算する関数副プログラム名 (使用例参照)．
EPSA…… **入力**．積分の近似値 S に対する絶対誤差の上限 $\varepsilon_a(\ge 0.0)$．
EPSR …… **入力**．積分の近似値 S に対する相対誤差の上限 $\varepsilon_r(\ge 0.0)$．
NMIN …… **入力**．被積分関数の計算回数の下限．
標準値としては 21 が適当．0≤NMIN<150．
NMAX…. **入力**．被積分関数の計算回数の上限．
NMAX>NMIN+8．
標準値としては 2000 程度が適当．
(使用上の注意④参照)．
S …… **出力**．積分の近似値 (使用上の注意⑤参照)．
ERR…. **出力**．積分の近似値 Sの絶対誤差の推定値．
N…… **出力**．被積分関数の計算回数
ICON…… **出力**．コンディションコード．
表 AQN9-1 参照．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, MGSSL
② FORTRAN 基本関数 ... ABS, MAX0, AMAX1, AMIN1, ALOG, DLOG, MOD, FLOAT

b. 注意
① パラメタ FUN に対応する関数副プログラムは，積分変数だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムでその関数名を ETERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより呼び出す側のプログラムとの連絡を図る (使用例参照)．

表 AQN9-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000 ｜ 13111	特異点など異常点を検出した． { 1000 の位が 1 のとき代数特異点 2 のときコーシー特異点 3 のときその両方 100 の位は対数特異点 10 の位は不連続点 1 の位はその他の異常点 } があったことを表す．	処理を続ける．代数特異点，対数特異点，不連続点だけの場合は，S の値は要求精度を満たしている場合が多い．しかしその他の異常点の場合は要求精度は得られていない．
20000 ｜ 23111	被積分関数の計算回数がその上限に達しても要求精度が得られなかった．1000 位以下の意味は，コード 10000~13111 の場合と同じ．	処理を打ち切る．S にはそのときまでに得られている近似値を出力するが，精度は保証できない．
30000	次のいずれかであった． ① EPSA<0.0 ② EPSR<0.0 ③ NMIN<0, NMIN≥150 ④ NMAX≤NMIN+8	処理を打ち切る．

② 本サブルーチンは，広範な関数に対して使用することができ，特に被積分関数 $f(x)$ がピーク型の関数である場合にもっとも有効である．更に $f(x)$ が代数特異点，対数特異点，不連続点などの異常点を積分区間 $[a,b]$ の端，中点，4 等分点など，2 等分割を繰り返すことにより早期に到達できる点に持つ場合でも有効にはたらく．したがって，性質のよく分からない関数をはじめ，ピーク型の関数及び積分区間内に特異点を持つ関数に対しては本サブルーチンの使用を第一に勧める．ただし，精度を上げるためには，これらの特異点を積分区間の端で，しかも原点に位置するように，必要ならば適当に変数変換することが望ましい．
滑らかな関数及び振動型の関数の場合はサブルーチン AQC8 を，特異点が積分区間の端点だけに位置する場合はサブルーチン AQE を使用した方が効率の面で良い．

③ 積分区間 [a,b] 内のある点で，被積分関数 $f(x)$ が無限大 ($f(x)\to\pm\infty$) となる場合は，あとの使用例で挙げるように，その点での $f(x)$ の値を適当な有限値に置き換える (例えば 0 と置く) ことが必要である．

④ パラメタ NMIN 及び NMAX
本サブルーチンは，被積分関数 $f(x)$ の計算回数を
NMIN≤計算回数≤NMAX
の範囲に限定している．すなわち収束判定に

かかわりなく，$f(x)$ は少なくとも NMIN 回計算され，しかも NMAX 回を超えて計算されることはない．もし NMAX 回以内に (1.1) を満たす S が求まらなかった場合は，ICON を 20000 ~ 21111 として処理を終える．なお NMAX が 21 未満の値であったときは，21 が与えられたものと見なす．

⑤ 積分の近似値 S の精度

本サブルーチンは，与えられた ε_a , ε_r に対して，(1.1) を満たす S を求める．したがってε_r=0 とおけば，絶対誤差 ε_a 以内の，また ε_a=0 とおけば相対誤差 ε_r 以内の精度で積分の近似値を求めることになる．しかし，関数の性質と ε_a , ε_r の値によっては，この目的が達成できないこともある．例えば，関数の計算精度に比べて ε_a , ε_r が小さすぎる場合には，このようなことが起こり，関数の計算回数が多くなって，NMAX 回以内では収束しないことがある．このときの S の値は，それまでに得られている近似値であってその精度は保証できない．この状態は，ICON が 20000 ~ 23111 の範囲の値をとることから知ることができる．なお，本サブルーチンは，いずれの場合にも，積分の近似値 S のほかに，その絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

c. 使用例

助変数 p を 0.1 から 0.1 刻みに 0.9 まで変えて，積分

$$\int_0^1 (x^{-p} + \sin(px))dx$$

を計算する．

```
C      **EXAMPLE**
       COMMON P
       EXTERNAL FUN
       A=0.0
       B=1.0
       EPSA=1.0E-4
       EPSR=1.0E-4
       NMIN=21
       NMAX=2000
       DO 10 I=1,9
       P=FLOAT(I)/10.0
       CALL AQN9(A,B,FUN,EPSA,EPSR,NMIN,
      *     NMAX,S,ERR,N,ICON)
    10 WRITE(6,600) P,ICON,S,ERR,N
       STOP
   600 FORMAT(' ',30X,'P=',F6.3,5X,
      *'ICON=',I5,5X,'S=',E15.7,5X,
      *'ERR=',E15.7,5X,'N=',I5)
       END
       FUNCTION FUN(X)
       COMMON P
       FUN=0.0
       IF(X.GT.0.0) FUN=X**(-P)+SIN(P*X)
       RETURN
       END
```

(4) 手法概要

本サブルーチンは，ニュートン・コーツ 9 点則に基づく適応型自動積分法を用いている．適応型自動積分法というのは，被積分関数 $f(x)$ の変化の急なところでは，分点を密にとり，反対に緩やかなところでは，疎にとるように自動的に調整する機能を具えた方法であって，現在のところ，一般目的のための自動積分法としては，信頼性と経済性において，最も均衡のとれた良い方法である．

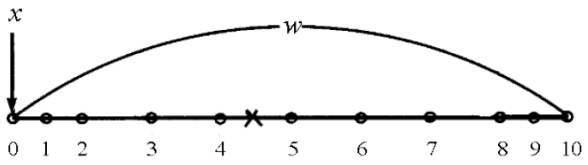
a. 計算の手順

手順 1…初期設定

積分区間の左端 x，幅 w，積分の近似値 S の初期値をそれぞれ $x=a$, $w=w_0=(b-a)$, $S=0$ と定める．その他の諸変数の初期値を，種々の判定のための定数を求める．両端を含む積分区間の 8 等分点での関数値 f_0, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_{10} と両端の 16 等分点での関数値 f_1, f_9 とを計算する (図 AQN9-1 参照)．これら 11 個の関数値を用い，修正ニュートン・コーツ 9 点則 (4.5) により，$f(x)$ の積分区間での平均値 $\overline{f}$ を計算する．

手順 2…2 等分割と情報の蓄え

現在の部分区間 (左端 x, 幅 w) を 2 等分割する．右半分の関数値 f_6, f_7, f_8, f_9, f_{10} を幅 w，打ち切り誤差の推定値 e などと共に，スタックに蓄える．左半分の右端から 2 番目の 8 等分点 (図 AQN9-1 の×印の点) での関数値を計算して f_8 とする．左半分を処理するために，手順 3 へ進む．



点 0, 2 ~ 8, 10 は 8 等分点
点 1, 9 は 16 等分点

図 AQN9-1 分点の配置

手順 3…収束判定

現在の部分区間の 8 等分点と両端の 16 等分点の関数値の中で，また計算されていない 4 個の値，f_1, f_4, f_6, f_9 を計算する．これら 11 個の関数値を用いた評価式 (4.4) によって，現在の部分区間の 9 点則による積分値の打ち切り誤差 e を推定しこれに対して収束判定を行う．判定結果が合格ならば，手順 7 へ進み，不合格ならば手順 4 へ進む．

手順 4…異常点の検出
現在の部分区間の端に何らかの異常点が存在するかどうかを調べ，その存在が認められれば手順 6 へ進む．幅 w が，$a, b, \varepsilon_a, \varepsilon_r$ などの入力変数によって定められる許容最小値よりも小さくなれば，手順 5 へ進む．上記以外の場合は手順 2 へ戻る．

手順 5…異常点の存在の再検査
緩和された検出基準を用いて異常点の存在の再検査を行い，合格すれば手順 6 へ，さもなければ手順 7 へ進む．

手順 6…異常点の処理
検出された異常点の種類と，必要なパラメタの値によって定められる解析的な計算を行って，現在の部分区間の積分値を定める．

手順 7…積分値の積算と情報の取出し
正常の場合には，現在の部分区間に対する積分値を，修正 9 点則 (4.5) で計算し，これを現在までの S の値に積算する．異常点の場合には，手順 6 で得られた値を S に加える．最後に蓄えておいた情報をスタックから取出し，新しい部分区間における関数値 f_2 を計算して，手順 3 に戻る．もしスタックが空であれば，処理を終了し S を積分値とする．

b. ニュートン・コーツ 9 点則と誤差評価
図 AQN9-1 に示すような，幅 w の区間に，ニュートン・コーツ 9 点則を適用したときの計算値 $S(x,w)$ は (4.1)，その打ち切り誤差 e は(4.2) で与えられる． ただし I は積分の真値，f_i は点 i における被積分関数の値である．

$$\begin{aligned} S(x,w) &= \frac{w}{28350}\{989(f_0+f_{10})+5888(f_2+f_8) \\ &\quad -928(f_3+f_7)+10496(f_4+f_6)-4540f_5\} \\ &= I+e \end{aligned} \tag{4.1}$$

$$e = \frac{37w^{11}f^{(10)}}{62783697715200} \tag{4.2}$$

ここに $f^{(10)}$ は区間内の適当な点での $f(x)$ の 10 階導関数の値である．従来は，この e の値を推定するために区間を 2 等分して左右の半区間にニュートン・コーツ 9 点則を適用して，$S(x,w/2)$ と $S(x+w/2,w/2)$ とを計算しこれらの和と $S(x,w)$ とから (4.3) の計算を行った．

$$e \approx \frac{1024}{1023}\left\{S(x,w)-\left(S(x,w/2)+S(x+w/2,w/2)\right)\right\} \tag{4.3}$$

このようにすると新たに 8 個の関数値を計算しなければならない．ところが，(4.2) から明らかなように，e を推定するには $f^{(10)}$を推定すればよく，それには 11 個の関数値があればよい．したがって新たに 2 個の関数値を計算するだけで十分である．本サブルーチンでは，図 AQN9-1 に示すように，区間の 8 等分点と，二つの 16 等分点 1，9 に基づく 10 次数値微分公式 (4.4) によって e を推定する．

$$\begin{aligned} e \approx \frac{2383w}{468242775}\{&3003(f_0+f_{10})-16384(f_1+f_9) \\ &+27720(f_2+f_8)-38220(f_3+f_7) \\ &+56056(f_4+f_6)-64350f_5\} \end{aligned} \tag{4.4}$$

e の値が収束判定基準 (4.7) を満足したとき，この e の値をニュートン・コーツ 9 点則の値 $S(x,w)$ から差し引いて，より良い近似値を計算する．本サブルーチンでは，この計算を (4.1) と (4.4) を経由しないで、(4.1) の右辺と (4.4) の右辺の解析的な差として得られる．修正ニュートン・コーツ 9 点則

$$\begin{aligned} \frac{w}{936485550}\{&18447429(f_0+f_{10})+77594624(f_1+f_9) \\ &+63216384(f_2+f_8)+150355296(f_3+f_7) \\ &+81233152(f_4+f_6)+154791780f_5\} \end{aligned} \tag{4.5}$$

によって直接行っている．(4.1) は関数値に乗ぜられる重み係数の符号が一定でないのに対して，(4.5) は符号がすべて正であって，数値的安定性の高い，好適な積分公式である．

c. 収束判定基準
大域的な自動積分法とは異なり，適応型の自動積分法では各々の部分区間ごとに収束判定を行わなければならない．最終的に得られる積分値 S が，(1.1)の規準を満足するようにするために，従来から行われている判定規準は

$$|e| \le \max(\varepsilon_a, \varepsilon_r|S'|)w/w_0 \tag{4.6}$$

である．ただし，S' は $\int_a^b f(x)dx$ の近似値，w_0 は全積分区間の幅 $b-a$ である．この式は許容誤差を各部分区間に，その幅に比例して分配しようという，自然な考えに基づくものである．しかしながら，本サブルーチンにおけるように，推定誤差を積分値から差し引いて誤差の主要項を消去している場合には，一般に得られた近似値は，要求精度を大幅に上廻る精度を持つものと考えられる．したがってこのような場合に，真正直に (4.6) を用いるのは，あまりにも保守的である．むしろ，全体の精度への影響の小さい，短い区間に対しては判定規準を積極的に緩和して，早めに処理を終了して無駄な関数計算の節約を図るのが得策である．本サブルーチンでは，そのような試みの一つとして，(4.7) を採用している．

$$|e| \le \max(\varepsilon_a, \varepsilon_r |S'|) w/w_0 \cdot \log_2(w_0/w) \qquad (4.7)$$

(4.7) は理論的根拠に欠けるが，実際的な見地からは十分効力のある方式であるといえる．なお (4.7) の S'はそのとき現在の S，x と，手順 1 で計算された $f(x)$ の平均値 $\overline{f}$ を用いて

$$S' = S + \overline{f}(b - x) \qquad (4.8)$$

として計算される．

d. 異常点の検出と処理

積分区間が異常点を含む場合を考えよう．ここに異常点としては，不連続点，対数特異点及び代数特異点を取り上げる．ただし代数特異点としては，通常の積分の意味では発散するがコーシーの主値積分の意味で収束するようなものも含めるものとし，これをコーシー特異点という．以上のような異常点において関数値が無限大となる場合には，これを適当な有限値，例えば 0 で置き換えるものとする．さて，部分区間が異常点を含む場合には，(4.4) の e は収束判定基準を満足しないので，区間の細分化が進行し，異常点をこの内部又は端に含む，区間の列が得られる．異常点が積分区間の 2^m 等分点の一つに位置していると仮定しよう．ただし m は正の整数である．この場合には，細分化のある段階からは，異常点は区間の端に固定され，この端点を共有する，長さが次第に半減する区間の列$\{I_i, i=1,2,...\}$がえられる．簡単のために，異常点は原点 $x=0$ であり，区間の左端に位置するものと仮定する．(図 AQN9-2 参照)

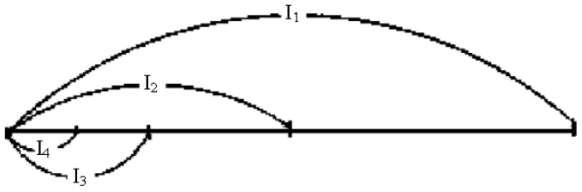


図 AQN9-2 異常点の検出

ここで打ち切り誤差 e を区間の幅 w で正規化した，正規化誤差 $\tilde{e} = e/w$ を考えよう．上の区間 I_I での $\tilde{e}$ の値を $\tilde{e}_i$ とする．$\tilde{e}_i$ はそれぞれの区間の左端の異常点に関して相似の位置にある点での関数値の，区間の幅に無関係な係数による同じ一次結合であり，その係数の和は 0 であることを考え合わせると，次のことがわかる．

- 不連続点 ⟷ $\tilde{e}_i$が一定値 $\tilde{e}$に近づく．
- 対数特異点 ⟷ $\Delta\tilde{e}_i = \tilde{e}_{i+1} - \tilde{e}_i$が一定値 dに近づく．
- 代数特異点 ⟷ $\Delta\tilde{e}_i / \Delta\tilde{e}_{i+1}$が一定値 rに近づく．

$\tilde{e}_i$ が上のどれかの挙動を示すとき，その時の部分区間の積分値は，それぞれ以下のようにして計算される．

1. 不連続点…. 不連続点での跳躍量をδとすると (4.4) から

$$\delta = \frac{468242775}{2383 \times 3003} \tilde{e} \qquad (4.8)$$

である．この δ だけのずれが f_0 に加わっているから，$f_0 - \delta$ をあらためて f_0 として，(4.5) を計算すればよい．

2. 対数特異点…特異点の近傍での関数形を

$$f(x) = \alpha \log x + \beta \qquad (4.9)$$

とする．(4.4) から

$$\alpha = \frac{468242775}{2383 \times 3003 \times \log 2} d \qquad (4.10)$$

となることがわかる．

$$I = \int_0^w f(x)dx = w(\alpha(\log w - 1) + \beta),$$
$$f_5 = f(w/2) = \log(w/2) + \beta$$

を考慮すると

$$I = w(f_5 + \alpha(\log 2 - 1)) \qquad (4.11)$$

と計算できる．

3. 代数特異点…特異点の近傍での関数形を

$$f(x)=\alpha\, x^{p}+\beta\, x^{p+1}+\gamma \qquad (4.12)$$

とする．特異点の次数 p は

$$p=\log_2 r \qquad (4.13)$$

によって得られる．α, β, γ を f_0, $f_5=f(w/2)$, $f_{10}=f(w)$, γ, $\tilde{e}$ (最後の $\tilde{e}_i$), $\tilde{e}'$ ($\tilde{e}$ のひとつ前の $\tilde{e}_i$) から計算し，これらを用いて

$$I=\int_0^w f(x)dx=w\left(\frac{\alpha\, w^{p}}{p+1}+\frac{\beta\, w^{p+1}}{p+2}+\gamma\right) \qquad (4.14)$$

を解析的に計算する．

4. コーシー特異点…ある区間の右端で代数特異点が検出され，その次数 p が $p \le -1$であるとする．同じ特異点が右に隣接する区間の左の端点として同じ次数 p の代数特異点であり，その主要部が左の区間での主要部と相打ち消し合うとき，これをコーシー特異点として取り扱う．当該部分の積分は，被積分関数から，主要部を取り除いたものの積分として計算される．

e. 種々の情報の蓄え
手順 2 では，右半分の区間の 5 個の関数値 f_6, f_7, f_8, f_9, f_{10} (後で右半分の f_3, f_5, f_7, f_8, f_{10} として，再利用される)，右半分の幅 w，$\log_2(w_0/w)$ の値，異常点の検出のために必要な 3 個の量 $\tilde{e}_i$, $\Delta\tilde{e}_i$, $\Delta\tilde{e}_i/\Delta\tilde{e}_{i+1}$, 総計 10 個の情報をスタックに蓄える．これらは手順 7 で最後に蓄えたものから逆順に取り出されて再利用される．スタックの深さはすべて，60 に取ってある．したがって，必要な器量容量は 600 である．
なお，詳細は参考文献 [96]，[97] を参照のこと．

A21-12-0101 ASSM, DASSM

行列の和 (実対称行列)
CALL ASSM(A,B,C,N,ICON)

(1) 機能

$n \times n$ の実対称行列 ***A*** と実対称行列 ***B*** の和 ***C*** を求める.

$$\boldsymbol{C}=\boldsymbol{A}+\boldsymbol{B}$$

ここで, ***C*** は $n \times n$ の実対称行列である.
$n \geq 1$ であること.

(2) パラメタ

A …… **入力**. 行列 ***A***. 対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.
B …… **入力**. 行列 ***B***. 対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.
C …… **出力**. 行列 ***C***. 対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.
(使用上の注意①参照)
N …… **入力**. 行列 ***A***, ***B*** 及び ***C*** の次数 n.
ICON …… **出力**. コンディションコード.
表 ASSM-1 参照.

表 ASSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N<1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ...なし.

b. 注意
① 配列 A あるいは, B 上の内容を保存する必要のない場合は, 次のように呼び出すことにより領域が節約できる.
・ 配列 A の内容が不要の場合.
CALL ASSM(A,B,A,N,ICON)
・ 配列 B の内容が不要の場合.
CALL ASSM(A,B,B,N,ICON)
この場合, 行列 ***C*** は配列 A あるいは, B 上に得られる.

c. 使用例
実対称行列 ***A***, ***B*** の和を求める. $n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),B(5050),C(5050)
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
   10 READ(5,100) N
      IF(N.EQ.0) STOP
      WRITE(6,150)
      NT=N*(N+1)/2
      READ(5,200) (A(I),I=1,NT)
      READ(5,200) (B(I),I=1,NT)
      CALL ASSM(A,B,C,N,ICON)
      IF(ICON.NE.0) GOTO 10
      CALL PSM(IA,1,A,N)
      CALL PSM(IB,1,B,N)
      CALL PSM(IC,1,C,N)
      GOTO 10
  100 FORMAT(I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     *'** MATRIX ADDITION **')
      END
```

本使用例中のサブルーチン PSM は, 実対称行列を印刷するものである. プログラムは, サブルーチン MGSM の使用例に記載されている.

A25-31-0201 ASVD1, DASVD1

実行例の特異値分解 (ハウスホルダー法，QR 法)
CALL ASVD1(A,KA,M,N,ISW,SIG,U,KU,V,KV,VW, ICON)

(1) 機能

$m \times n$ の実行例 $\boldsymbol{A}$ をハウスホルダー法，QR 法により特異値分解する．

$$\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}} \qquad (1.1)$$

ここで，l=min(m,n) として，$\boldsymbol{U}$ 及び $\boldsymbol{V}$ は $m \times l$ 及び $n \times l$ の行列であり，

$l=n(m\geq n)$ のとき，$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{V}=\boldsymbol{V}\boldsymbol{V}^{\mathrm{T}}=\boldsymbol{I}_n$

$l=m(m<n)$ のとき，$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}=\boldsymbol{U}\boldsymbol{U}^{\mathrm{T}}=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{V}=\boldsymbol{I}_m$

である．$\boldsymbol{\Sigma}$ は$\boldsymbol{\Sigma}$=diag(σ_i),$\sigma_i\geq 0$ なる l 次の対角行列であり，σ_i を $\boldsymbol{A}$ の特異値という．また，特異値σ_i は行列 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 固有値の正平方根であり，$\boldsymbol{V}$ の第 i 列は対応する固有ベクトルである．

$m\geq 1$, $n\geq 1$ であること．

行列 $\boldsymbol{A}$, $\boldsymbol{U}$, $\boldsymbol{\Sigma}$, $\boldsymbol{V}$ の寸法の関係については図 ASVD1-1 を参照のこと．

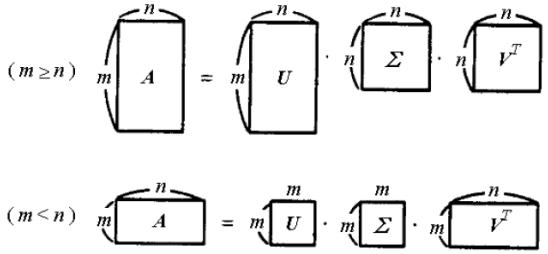


図 ASVD1-1 特異値分解の各行列の寸法の関係

(2) パラメタ

A …… **入力**. 行列 $\boldsymbol{A}$.
A(KA,N) なる 2 次元配列.
(使用上の注意参照).

KA …… **入力**. 配列 A の整合寸法 (≥M).

M …… **入力**. 行列 $\boldsymbol{A}$ と $\boldsymbol{U}$ の行数 m.

N …… **入力**. 行列 $\boldsymbol{A}$, $\boldsymbol{U}$ の列数，$\boldsymbol{V}$ の行数 n.

ISW …… **入力**. 制御情報.
ISW=10d_1+d_0. ただし，0≤d_0, d_1≤1.
d_1=0　　のとき行列 $\boldsymbol{U}$ を求めない.
d_1=1　　のとき行列 $\boldsymbol{U}$ を求める.
d_0=0　　のとき行列 $\boldsymbol{V}$ を求めない.
d_0=1　　のとき行列 $\boldsymbol{V}$ を求める.

SIG …… **出力**. 行列 $\boldsymbol{A}$ の特異値.
大きさ l の 1 次元配列.
(使用上の注意④参照).

U …… **出力**. 行列 $\boldsymbol{U}$.
U(KU,N) なる 2 次元配列.
(使用上の注意参照).

KU …… **入力**. 配列 U の整合寸法 (≥M).

V …… **出力**. 行列 $\boldsymbol{V}$.
V(KV,K) なる 2 次元配列. K= min(M+1,N).
(使用上の注意参照)

KV …… **入力**. 配列 V の整合寸法 (≥N).

VW …… **作業領域**. 大きさ n の 1 次元配列.

ICON …… **出力**. コンディションコード.
表 ASVD1-1 参照.

表 ASVD1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
15000	ある特異値が求められなかった.	処理を打ち切る.
30000	M<1,N<1,KA<M,KU<M,KV<N 又は ISW≠0,1,10,11 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数 ... MIN0, MOD, SIGN, SQRT, AMAX1, ABS

b. 注意

① 特異値分解により求められる分解要素 $\boldsymbol{U}$, $\boldsymbol{\Sigma}$及び $\boldsymbol{V}$ を用いるならば，元の行列 $\boldsymbol{A}$ の一般逆行列 $\boldsymbol{A}^{+}$ 又は，連立 1 次方程式 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ の最小二乗最小ノルム解が求められる．(説明は「3.5 最小二乗解」参照).
しかし，このような場合，そのための専用サブルーチン GINV 又は LAXLM を利用する方が効果的である.

② 特異値分解は，応用の広い有効な方法である (“3.5 最小二乗解” 参照) が，計算量が非常に多いことが欠点である．したがって $\boldsymbol{U}$，$\boldsymbol{V}$ などは，必要がなければ計算しないようにすることが望ましい．入力パラメタ ISW はそのためのものである.
$\boldsymbol{U}$，$\boldsymbol{V}$ を計算しないときは，これらを参照しないので，これらに対応する実引数は，2 次元配列である必要がなく変数でよい.

③ 本サブルーチンでは，記憶容量節約のために，$\boldsymbol{U}$ 又は $\boldsymbol{V}$ の中のどちらか一方を A の上に重ね書きできるように配慮されている．$\boldsymbol{A}$ を保存する必要がない場合に，$\boldsymbol{U}$ 又は $\boldsymbol{V}$ に対応する実引数として A を書くことにより目的を達成することができる.

④ 特異値はすべて非負で，大きさの減少する順に格納される．ただし，ICON = 15000 の場合には，非負のものだけが求められた特異値で，その他のものは -1 となっており，特異値も大小順になっていない.

⑤ 本サブルーチンは，行列 $\boldsymbol{A}$ の行数 m と列数 n の大小関係において何ら制約がない，つまり A が縦長行列，正方行列又は横長行列であっても，本サブルーチンは，特異値分解を行うことができる．

c. 使用例

$m \times n$ 行列 $\boldsymbol{A}$ の特異値分解を行い，すべての特異値と，対応する $\boldsymbol{V}$ の列ベクトルを出力する．ただし，$\boldsymbol{U}$ は計算せず，$\boldsymbol{V}$ を $\boldsymbol{A}$ の上に重ね書きするものとする．

$1 \le n \le 100$，$1 \le m \le 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(100,100),SIG(100),
     *          VW(100)
   10 READ(5,500) M,N
      IF(M.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,M),J=1,N)
      WRITE(6,600) M,N
      DO 20 I=1,M
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL ASVD1(A,100,M,N,1,SIG,A,100,
     *           A,100,VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.15000) GO TO 10
      CALL SVPRT(SIG,A,100,N,N)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',20X,'ORIGINAL MATRIX',
     *5X,'M=',I3,5X,'N=',I3/)
  610 FORMAT('0',4(5X,'A(',I3,',',I3,')=',
     *E14.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END

      SUBROUTINE SVPRT(SIG,V,K,N,M)
      DIMENSION SIG(N),V(K,M)
      WRITE(6,600)
      DO 20 INT=1,M,5
      LST=MIN0(INT+4,M)
      WRITE(6,610) (J,J=INT,LST)
      WRITE(6,620) (SIG(J),J=INT,LST)
      DO 10 I=1,N
   10 WRITE(6,630) I,(V(I,J),J=INT,LST)
   20 CONTINUE
      RETURN
  600 FORMAT('1',20X,
     *'SINGULAR VALUE AND EIGENVECTOR')
  610 FORMAT('0',10X,5I20)
  620 FORMAT('0',5X,'SV',3X,5E20.7/)
  630 FORMAT(5X,I3,3X,5E20.7)
      END
```

本使用例中のサブルーチン SVPRT は，特異値と固有ベクトルを印刷するものである．

(4) 手法概要

$m \times n$ 行列 $\boldsymbol{A}$ をハウスホルダー法，QR 法により(4.1)のとおり特異値分解を行う．

$$\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}} \tag{4.1}$$

ここで，$l=\min(m,n)$ として，$\boldsymbol{U}$ 及び $\boldsymbol{V}$ は，$m \times l$ 及び $n \times l$ の行列，$\boldsymbol{\Sigma}$ は $\boldsymbol{\Sigma}=\mathrm{diag}(\sigma_i),\sigma_i \ge 0$ なる l 次の対角行列である．かつ

$l=n(m \ge n)$ のとき $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{V}=\boldsymbol{V}\boldsymbol{V}^{\mathrm{T}}=\boldsymbol{I}_n$
$l=m(m<n)$ のとき $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}=\boldsymbol{U}\boldsymbol{U}^{\mathrm{T}}=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{V}=I_m$

である．
なお，σ_i なる量を $\boldsymbol{A}$ の特異値という．

本サブルーチンでは，m と n の大小関係については制限を設けず，どのような長方形行列をも分解することができる．しかしながら，実際に必要となるのは，$m \ge n$ の場合が多いので，以下の説明では、簡単のために $m \ge n$ と仮定する．

a. 本サブルーチンにおける計算手順

本サブルーチンでは，次の 2 段階の手順で (4.1) の特異値分解を行う．

(a) $m \times n$ 行列の上 2 重対角行列への変換 (ハウスホルダー法)

2 つのハウスホルダー変換の列 $\boldsymbol{P}_1,\ldots,\boldsymbol{P}_n$，$\boldsymbol{Q}_1,\ldots,\boldsymbol{Q}_{n-2}$ を，行列 $\boldsymbol{A}$ の左と右から交互に作用させて，

$$\boldsymbol{J}_0=\boldsymbol{P}_n\cdots\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{Q}_1\cdots\boldsymbol{Q}_{n-2} \tag{4.2}$$

を図 ASVD1-2 に示すような，上 2 重対角行列に変換する．

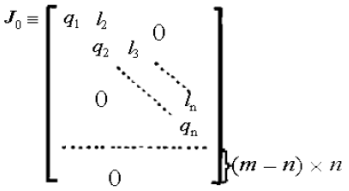


図 ASVD1-2 上 2 重対角行列の構造

$\boldsymbol{A}_1=\boldsymbol{A}$ から始めて，

$$\boldsymbol{A}_{k+1/2}=\boldsymbol{P}_k\boldsymbol{A}_k,\ k=1,\ldots,n \tag{4.3}$$

$$\boldsymbol{A}_{k+1}=\boldsymbol{A}_{k+1/2}\boldsymbol{Q}_k,\quad k=1,\ldots,n-2 \tag{4.4}$$

と定義する．$\boldsymbol{A}_k=(a_{ij}^{(k)})$ として，$\boldsymbol{P}_k$ を

$$a_{ik}^{(k+1/2)}=0\quad,\quad j=k+1,\ldots,m$$

が成り立つように，そして $\boldsymbol{Q}_k$ を

$$a_{kj}^{(k+1)}=0\quad,\quad j=k+2,\ldots,n$$

が成り立つように定めればよい．
そのためには，

$$\boldsymbol{P}_k=\boldsymbol{I}-\boldsymbol{x}_k\,\boldsymbol{x}_k^T/b_k,\quad k=1,\ldots,n \tag{4.5}$$

$$\boldsymbol{Q}_k=\boldsymbol{I}-\boldsymbol{y}_k\,\boldsymbol{y}_k^T/c_k,\quad k=1,\ldots,n-2 \tag{4.6}$$

とすればよい．ただし，

$$\left.\begin{aligned}
&\mathbf{x}_k = \left(0,\ldots,0,a_{k,k}^{(k)}+d_k,a_{k+1,k}^{(k)},\ldots,a_{n,k}^{(k)}\right)^T\\
&d_k = \left(\left(a_{k,k}^{(k)}\right)^2+\ldots+\left(a_{n,k}^{(k)}\right)^2\right)^{1/2}\operatorname{sign}\left(a_{k,k}^{(k)}\right)\\
&b_k = d_k^2 + d_k a_{k,k}^{(k)}\\
&\mathbf{y}_k = \left(0,\ldots,0,a_{k,k+1}^{(k+1/_2)}+e_k,a_{k,k+2}^{(k+1/_2)},\ldots,a_{k,n}^{(k+1/_2)}\right)^T\\
&e_k = \left(\left(a_{k,k+1}^{(k+1/_2)}\right)^2+\ldots+\left(a_{k,n}^{(k+1/_2)}\right)^2\right)^{1/2}\operatorname{sign}\left(a_{k,k+1}^{(k+1/_2)}\right)\\
&c_k = e_k^2 + e_k a_{k,k+1}^{(k+1/_2)}
\end{aligned}\right\} \quad (4.7)$$

である.

(b) 上 2 重対角行列の対角行列への変換 (QR 法)
二つの直行変換の列 $\boldsymbol{S}_i$, $\boldsymbol{T}_i$, i=0,1,... を適当に選び，$\boldsymbol{J}_0$ の左右から作用させて順次

$$\boldsymbol{J}_{i+1} = \boldsymbol{S}_i^T \boldsymbol{J}_i \boldsymbol{T}_i \quad (4.8)$$

を作り，$\boldsymbol{J}_{i+1}$ が対角行列 $\boldsymbol{\Sigma}$ に収束するようにする．すなわち，ある整数 q に対して

$$\Sigma = \boldsymbol{S}_q^{\mathrm{T}}\ldots\boldsymbol{S}_0^{\mathrm{T}}\boldsymbol{J}_0\boldsymbol{T}_0\ldots\boldsymbol{T}_q \quad (4.9)$$

となるようにする.
以上により，行列 $\boldsymbol{A}$ は対角行列化され，変換行列 $\boldsymbol{U}$ と $\boldsymbol{V}$ は，(4.1), (4.2) 及び (4.9) より

$$\mathbf{U} = \mathbf{P}_1\ldots\mathbf{P}_n\mathbf{S}_0\ldots\mathbf{S}_q \quad (4.10)$$

$$\boldsymbol{V} = \boldsymbol{Q}_1\ldots\boldsymbol{Q}_{n-2}\boldsymbol{T}_0\ldots\boldsymbol{T}_q \quad (4.11)$$

と表される．これらは，変換行列を逐次右から乗ずることにより生成することができる.
ところで，(4.8) における変換行列 $\boldsymbol{T}_i$ は，$\boldsymbol{M}_i = \boldsymbol{J}_i^T\boldsymbol{J}_i$ なる対称 3 重対角行列が，対角行列に収束するように選び，一方変換行列 $\boldsymbol{S}_i$ は，すべての $\boldsymbol{J}_i$ が上 2 重対角行列になるように選ぶこと.

b. QR 法における変換行列の選び方
(4.8) における変換行列 $\boldsymbol{T}_i$ 及び $\boldsymbol{S}_i$ の選び方について説明する.
簡単のために，次のような記号を用いる.

$$\boldsymbol{J} \equiv \boldsymbol{J}_i, \bar{\boldsymbol{J}} \equiv \boldsymbol{J}_{i+1}, \boldsymbol{S} \equiv \boldsymbol{S}_i, \boldsymbol{T} \equiv \boldsymbol{T}_i, \boldsymbol{M} \equiv \boldsymbol{J}^T\boldsymbol{J}, \overline{\boldsymbol{M}} \equiv \bar{\boldsymbol{J}}^T\bar{\boldsymbol{J}}.$$

$\boldsymbol{J}$ から $\bar{\boldsymbol{J}}$ への変換は，ギブンスの 2 次元回転変換を $\boldsymbol{J}$ の右及び左から交互に作用させることにより達成される.

$$\bar{\boldsymbol{J}} = \boldsymbol{L}_n^T\boldsymbol{L}_{n-1}^T\ldots\boldsymbol{L}_2^T\boldsymbol{J}\boldsymbol{R}_2\boldsymbol{R}_3\ldots\boldsymbol{R}_n \quad (4.12)$$

ここに,

$$L_k = \begin{bmatrix}
1 & \ddots & 0 & \vdots & 0 & \\
 & \ddots & 1 & \vdots & \vdots & \\
0 & & \cos\theta_k & -\sin\theta_k & \cdots & (k-1)\\
 & & \sin\theta_k & \cos\theta_k & \cdots & (k)\\
 & & & & 1 \quad 0 & \\
 & & 0 & & 0 \quad 1 &
\end{bmatrix}$$

であり，また $\boldsymbol{R}_k$ は θ_k を ϕ_k に代えて同様に定義される．角 θ_k, k=2,...,n と角 ϕ_k, k=3,...,n ををを適当に定めれば，任意の ϕ_2 に対して $\bar{\boldsymbol{J}}$ が上 2 重対角行列になるようにできる．すなわち，次のようにすればよい．以下，$\boldsymbol{J}$=(j_{ij}) とする.
$\boldsymbol{R}_2$ によって，非零要素 j_{21} が発生する.
$\boldsymbol{L}_2^{\mathrm{T}}$ によって，j_{21} を消去すると，非零要素 j_{13} が発生する.
$\boldsymbol{R}_3$ によって，j_{13} を消去すると，非零要素 j_{32} が発生する.
$\boldsymbol{R}_n$ によって，$j_{n\text{-}2n}$ を消去すると，非零要素 $j_{nn\text{-}1}$ が発生する.
$\boldsymbol{L}_n^T$ によって，$j_{nn\text{-}1}$ を消去する.
以上の経過を，n=5 の場合について図 ASVD1-3 に示す.

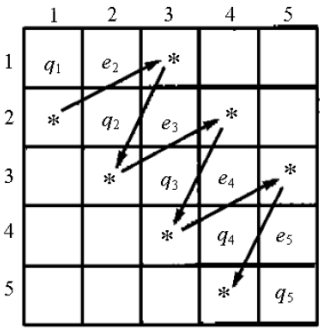


図 ASVD1-3 非零要素の推移 (n=5)

$$\boldsymbol{S} = \boldsymbol{L}_2\boldsymbol{L}_3\ldots\boldsymbol{L}_n \quad (4.13)$$

$$\boldsymbol{T} = \boldsymbol{R}_2\boldsymbol{R}_3\ldots\boldsymbol{R}_n \quad (4.14)$$

とおくと，(4.12) から

$$\boldsymbol{J} = \boldsymbol{S}^{\mathrm{T}}\boldsymbol{J}\boldsymbol{T} \quad (4.15)$$

であり，$\boldsymbol{J}$ が上 2 重対角行列であることから,

$$\overline{\boldsymbol{M}} = \boldsymbol{J}^T\bar{\boldsymbol{J}} = \boldsymbol{T}^T\boldsymbol{M}\boldsymbol{T} \quad (4.16)$$

は，$\boldsymbol{M}$ と同様に対称 3 重対角行列となる.
ところで，最初の回転角 ϕ_2，つまり最初の変換 $\boldsymbol{R}_2$ は未定であったが，これを適当に定めて (4.16) の変換が，原点移動量 s を伴う QR 変換であるようにできる.

すなわち，(4.17) のような QR 変換であるようにできる．

$$\overline{\boldsymbol{M}}_s = \boldsymbol{T}_s^{\mathrm{T}} \boldsymbol{M} \boldsymbol{T}_s \tag{4.17}$$

ここに，

$\mathbf{M} - s\mathbf{I} = \mathbf{T}_s \mathbf{R}_s$

$\mathbf{R}_s \mathbf{T}_s + s\mathbf{I} = \overline{\mathbf{M}}_s$

$\mathbf{T}_s^{\mathrm{T}} \mathbf{T}_s = \mathbf{I}$

$\mathbf{R}_s$: 上三角行列

である．
このためには，$\boldsymbol{R}_2$ の第 1 列が $\boldsymbol{M} - s\boldsymbol{I}$ の第 1 列に比例するようにすればよい．原点移動量 s は，QR 法と同様に，$\boldsymbol{M}$ の右下の 2 次の小行列の固有値の中の，絶対値が小さいものとして定められる．実際には，$\boldsymbol{T}_s$ と $\boldsymbol{T}$ は同じであることが確かめられる．

c. QR 法の収束判定及び直和分解収束の判定は次のようにする．

$$|e_n| \le \varepsilon_1 \tag{4.18}$$

となったら，$|q_n|$を特異値として採択し，行列の次数を 1 だけ下げる．ただし，

$$\varepsilon_1 = \|\mathbf{J}_0\|_\infty u \tag{4.19}$$

であり u は丸めの誤差の単位である．
もしも，$k \neq n$ なる k について

$$|e_k| \le \varepsilon_1 \tag{4.20}$$

となったら，行列を二つの小行列の直和に分解し，各々の小行列を別々に処理する．
もしも $k \neq 1$ なる k について，

$$|q_k| \le \varepsilon_1 \tag{4.21}$$

となったときには，第 k 列に関するギブンスの 2 次元回転変換を $\boldsymbol{J}$ の右から，次のように作用させる．

$e_k \equiv j_{k-1,k}$ を消去すると，$j_{k-2,k}$ と $j_{k,k-1}$ が発生する．
$j_{k-2,k}$ を消去すると，$j_{k-3,k}$ と $j_{k,k-2}$ が発生する．
.....
$j_{1,k}$ を消去すると，$j_{k,1}$ が発生する．

このようにして，行列は次の形となる．

$$\bar{\boldsymbol{J}} = \begin{bmatrix} \tilde{q}_1 & \tilde{e}_2 & & & 0 & \\ & \ddots & \ddots & & & \\ & & \tilde{e}_{k-1} & & & \\ 0 & & \tilde{q}_{k-1} & 0 & & \\ \delta_1 & \delta_2 & \cdots & \delta_{k-1} & \tilde{q}_k & e_{k+1} \\ & & & & \ddots & \ddots \\ & 0 & & & & e_n \\ & & & & & q_n \end{bmatrix} \begin{matrix} \\ \\ \\ \\ (k) \\ \\ \\ \end{matrix}$$

$(k-1)$ (k)

ところが，変換行列の直交性により，

$$\delta_1^2 + \delta_2^2 + \ldots + \delta_{k-1}^2 + q_k^{-2} = q_k^2 \le \varepsilon_1^2 \tag{4.22}$$

が成り立つ．したがって，$\delta_1, \delta_2, \ldots, \delta_{k-1}$ はすべて絶対値が ε_1 以下となり， 行列 $\bar{\boldsymbol{J}}$ は 2 つの小行列に分解することができる．
なお，一つの特異値あたり 30 回の反復で QR 法が収束しない場合には，処理を打ち切り，まだ定められていない特異値を -1 とする．
なお，詳細については，参考文献 [11] を参照すること．

A52-31-0302 BDLX, DBDLX

LDL^T分解された正値対称バンド行列の連立1次方程式
CALL BDLX(B,FA,N,NH,ICON)

(1) 機能

LDL^T 分解された正値対称バンド行列を係数に持つ連立1次方程式

$$\boldsymbol{LDL}^T\boldsymbol{x}=\boldsymbol{b} \qquad (1.1)$$

を解く．ただし $\boldsymbol{L}$，$\boldsymbol{D}$ はそれぞれ $n\times n$，下バンド幅 h の単位下バンド行列と対角行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．$n>h\geq 0$ であること．

(2) パラメタ

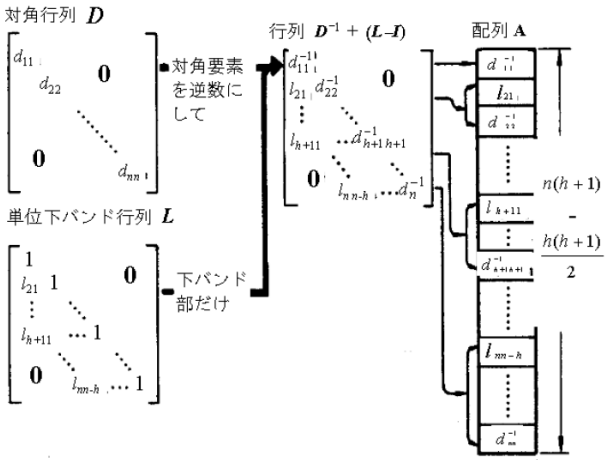
B……**入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の1次元配列.

FA……**入力**．行列 $\boldsymbol{L}$ と 行列 $\boldsymbol{D}^{-1}$.図 BDLX-1 参照.
対称バンド行列用圧縮モード.
大きさ $n(h+1)-h(h+1)/2$ の1次元配列.

N……**入力**．行列 $\boldsymbol{L}$，$\boldsymbol{D}$ の次数 n.

NH……**入力**．下バンド幅 h.

ICON……**出力**．コンディションコード.
表 BDLX－1 参照.



［注意］ 配列FAに行列 $D^{-1}+(L-I)$ の対角部分及び下バンド部分を，対称バンド行列用圧縮モードで格納する.

図 BDLX-1 行列 L 及び D^{-1} の格納方法

表 BDLX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	係数行列が正値でなかった.	処理は続行する.
30000	NH<0 又は NH≥N であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .... MGSSL
② FORTRAN 基本関数 ... なし.

b. 注意
① 本サブルーチンでは行列 $\boldsymbol{L}$ のバンド部分の外側にある要素にかかわる計算は省略しているので正値対称行列用のサブルーチン LDLX よりも処理が速い.
② FA には LDL^T 分解された結果の係数行列が対称バンド行列用圧縮モードで格納されていなければならない．更に，行列 $\boldsymbol{D}$ については $\boldsymbol{D}^{-1}$ としてその対角要素を入力する必要があるので注意すること.
本サブルーチンを，サブルーチン SBDL に続けて呼び出すことにより，連立 1 次方程式を解くことができる．しかし，通常は，サブルーチン LSBX を呼び出せば，一度に解が求められる.

c. 使用例
$n\times n$，上，下バンド幅 h の係数行列をサブルーチン SBDL で LDL^T 分解し，連立1次方程式を解く．$n\leq 100$，$h\leq 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(3825),B(100)
   10 READ(5,500)N,NH
      IF(N.EQ.0)STOP
      NH1=NH+1
      NT=N*NH1-NH*NH1/2
      READ(5,510)(A(I),I=1,NT)
      WRITE(6,640)
      L=0
      LS=1
      DO 20 I=1,N
      L=L+MIN0(I,NH1)
      JS=MAX0(1,I-NH)
      WRITE(6,600)I,JS,(A(J),J=LS,L)
   20 LS=L+1
      CALL SBDL(A,N,NH,1.0E-6,ICON)
      WRITE(6,610)ICON
      IF(ICON.GE.20000)STOP
      READ(5,510)(B(I),I=1,N)
      CALL BDLX(B,A,N,NH,ICON)
      WRITE(6,610)ICON
      DET=A(1)
      L=1
      DO 30 I=2,N
      L=L+MIN0(I,NH1)
   30 DET=DET*A(L)
      DET=1.0/DET
      WRITE(6,620)(B(I),I=1,N)
      WRITE(6,630)DET
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT(' ','(',I3,',',I3,')'
     */(10X,5E17.8))
  610 FORMAT(/10X,'ICON=',I5)
  620 FORMAT(/10X,'SOLUTION VECTOR'
     *//(10X,5E17.8))
  630 FORMAT(/10X,
     *'DETERMINANT OF COEFFICIENT MATRIX=',
     *E17.8)
```

```
640 FORMAT(/10X,'INPUT MATRIX')
    END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{LDL}^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{b} \tag{4.1}$$

を解くことは，次の二つの方程式

$$\boldsymbol{Ly}=\boldsymbol{b} \tag{4.2}$$
$$\boldsymbol{L}^{T}\boldsymbol{x}=\boldsymbol{D}^{-1}\boldsymbol{y} \tag{4.3}$$

を解くことに帰着される．

a. $\boldsymbol{Ly}=\boldsymbol{b}$ を解く(前進代入)

$$y_i = b_i - \sum_{k=\max(1,i-h)}^{i-1} l_{ik} y_k, i = 1,\ldots,n \tag{4.4}$$

なる式で逐次求める．ただし，

$\mathbf{L} = \left(l_{ij}\right), \mathbf{y}^T = \left(y_1,\ldots,y_n\right), \mathbf{b}^T = \left(b_1,\ldots,b_n\right)$

である．

b. $\boldsymbol{L}^{T}\boldsymbol{x}=\boldsymbol{D}^{-1}\boldsymbol{y}$ を解く(後退代入)

$$x_i = y_i d_i^{-1} \sum_{k=i+1}^{\min(i+h,n)} l_{ki} x_k, \quad i = n,\ldots,1 \tag{4.5}$$

なる式で逐次求める．ただし，

$\boldsymbol{D}^{-1} = \mathrm{diag}(d_i^{-1})$, $\boldsymbol{x}^T=(x_1, ..., x_n)$.

である．

なお，詳細については，参考文献 [7] を参照すること．

E12-32-1102 BICD1, DBICD1

B-spline 2 次元補間式 (I-I).
CALL BICD1(X,NX,Y,NY,FXY,K,M,C,VW,ICON)

(1) 機能

xy 平面上の格子点 (x_i, y_j), $(x_1< x_2< ...< x_{n_x}, y_1< y_2< ...< y_{n_y})$ において関数値, $f_{i,j}=f(x_i,y_j)$ が与えられ, 更に周囲において偏微係数 $f_{i,j}^{(\lambda,\mu)}$, i=1,n_x, j=1, n_y, λ=1, 2, ..., $(m-1)/2$, μ=1,2, ..., $(m-1)/2$ が与えられたとき, 双 m（奇数）次の B-spline 2 次元補間式

$$s(x,y)=\sum_{\beta=-m+1}^{n_y-1}\sum_{\alpha=-m+1}^{n_x-1}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \qquad (1.1)$$

の補間係数 $c_{\alpha,\beta}$ を求める.

m≥3, n_x≥2, n_y≥2 であること.

尚, 後節での参照のため, 関数値などの $\hat{f}_{i,j}$ を次のように定義する.（但し, l=(m − 1)/2）. また $\hat{f}_{i,j}$ を行列の要素として並べたものを図 BICD1-1 に示す.

$$\hat{f}_{i,j}=f_{11}^{(l-i+1,l-j+1)}:i=1,2,...,l+1,\quad j=1,2,...,l+1$$
$$\hat{f}_{i,j}=f_{i-l,1}^{(0,l-j+1)}:i=l+2,l+3,...,n_x+l-1,\quad j=1,2,...,l+1$$
$$\hat{f}_{i,j}=f_{n_x,1}^{(i-n_x-l,l-j+1)}:i=n_x+l,n_x+l+1,...,n_x+2l,\quad j=1,2,...,l+1$$
$$\hat{f}_{i,j}=f_{1,j-l}^{(l-i+1,0)}:i=1,2,...,l+1,\quad j=l+2,l+3,...,n_y+l-1$$
$$\hat{f}_{i,j}=f_{i-l,j-l}:i=l+2,l+3,...,n_x+l-1,\quad j=l+2,l+3,...,n_y+l-1$$
$$\hat{f}_{i,j}=f_{n_x,j-1}^{(i-n_x-l,0)}:i=n_x+l,n_x+l+1,...,n_x+2l,\quad j=l+2,l+3,...,n_y+l-1$$

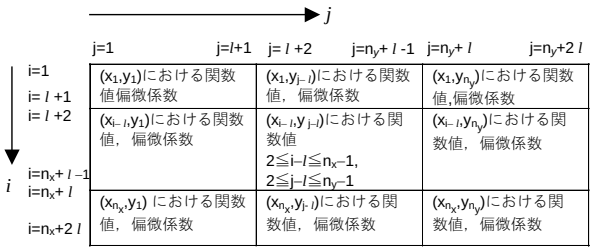


図 BICD1-1 関数値などの $\hat{f}_{i,j}$ (但し, l=(m-1)/2)

$$\hat{f}_{i,j}=f_{1,n_y}^{(l-i+1,j-n_y-l)}:i=1,2,...,l+1,\quad j=n_y+l,n_y+l+1,...,n_y+2l$$
$$\hat{f}_{i,j}=f_{i-l,n_y}^{(0,j-n_y-l)}:i=l+2,l+3,...,n_x+l-1,\quad j=n_y+l,n_y+l+1,...,n_y+2l$$
$$\hat{f}_{i,j}=f_{n_x,n_y}^{(i-n_x-l,j-n_y-l)}:i=n_x+l,n_x+l+1,...,n_x+2l,\quad j=n_y+l,n_y+l+1,...,n_y+2l$$

(2) パラメタ

X …… **入力**. x 方向の離散点 x_i.
大きさ n_x の 1 次元配列.

NX …… **入力**. x_i の個数 n_x.

Y …… **入力**. y 方向の離散点 y_j.
大きさ n_y の 1 次元配列.

NY …… **入力**. y_j の個数 n_y.

FXY …… **入力**. 関数値などの $\hat{f}_{i,j}$.
FXY (K,NY+M-1)なる 2 次元配列.
FXY (I,J) に $\hat{f}_{i,j}$ を代入する.（図 BICD1-1 参照）

K …… **入力**. 2 次元配列 FXY 及び C の整合寸法 (K≥NX+M-1).

M …… **入力**. B-spline の次数 m.
（使用上の注意②参照）.

C …… **出力**. 補間係数 $c_{\alpha,\beta}$.
C(K,NY+M-1)なる 2 次元配列.
C(I,J)には, $c_{-m+i,-m+j}$ が出力される.

VW …… **作業領域**. 次式で示す大きさの 1 次元配列.
$\{\max(n_x,n_y)+1\}(m+2)-3+(m+1)^2/2$.

ICON …… **出力**. コンディションコード
表 BICD1-1 参照.

表 BICD1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. (a) M が奇数でない. (b) x_i≥x_{i+1} なる x_i がある. (c) y_j≥y_{j+1} なる y_j がある. (d) M<3 (e) NX<2 又は NY<2	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UMIO1, UCIO1, UBAS1, ULUI1
② FORTRAN 基本関数 ... MOD, FLOAT

b. 注意

① 本サブルーチンに続けて，サブルーチン BIFD1 を呼び出すことにより，B-spline 2 次元補間式 (1.1) に基づく補間値あるいは偏微分値あるいは二重積分値を求めることができる．その時，パラメタ X, NX, Y, NY, K, M, C の値は BIFD1 の入力となる．

② 次数 *m* は 3 又は 5 程度がよい.倍精度では，元の関数がおとなしく，$\hat{f}_{i,j}$ が高精度で与えられているならば，もう少し次数を高くしてもよい．しかし 15 が限界である．

c. 使用例

サブルーチン BIFD1 の使用例参照.

(4) 手法概要

ここで求める B-spline2 次元補間式 S(*x*,y) は，サブルーチン BIC1 が求める B-spline 補間式 (I) をそのまま 2 次元へ拡張したものである．すなわち S(*x*,y) は領域 $R=\{(x,y)|x_1\le x<x_{n_x},y_x\le y\le y_{n_y}\}$ 上で定義され，次の条件を満たす関数である．

(a) (*x*,y) は各部分領域

$$R_{i,j}=\{(x,y)|x_i\le x<x_{i+1},\ y_j\le y<y_{j+1}\}$$

で，高々，双 *m* 次の多項式である．双 *m* 次とは *x* 方向， *y* 方向共に *m* 次であることをいう．

(b) $S(x,y)\in C^{m-1,m-1}$ [R].すなわち λ=0, 1, ..., *m*-1, μ=0,1,..., *m*-1 とするとき

$$\frac{\partial^{\lambda+\mu}}{\partial x^{\lambda}\partial y^{\mu}}S(x,y)$$

が存在してしかも連続である．

(c) $S(x_i,y_i)=f_{i,j},i=1,2,...,n_x,j=1,2,...,n_y$

$S^{(\lambda,\mu)}(x_i,y_j)=f_{i,j}^{(\lambda,\mu)},\quad i=1,n_x,\ j=1,n_y$

$\lambda=1,2,...,(m-1)/2,\quad \mu=1,2,...,(m-1)/2$

(a)，(b)を満たす *S*(*x*,y) は $c_{\alpha,\beta}$ を任意定数として，

$$S(x,y)=\sum_{\beta=-m+1}^{n_y-1}\sum_{\alpha=-m+1}^{n_x-1}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y)\quad (4.1)$$

と書き表すことができる．ここで $N_{\alpha,m+1}(x)$, $N_{\beta,m+1}(y)$ は共に *m* 次の B-splines で(4.2),(4.3)で表す．

$$N_{\alpha,m+1}(x)=(s_{\alpha+m+1}-s_{\alpha})g_{m+1}[s_{\alpha},s_{\alpha+1},...,s_{\alpha+m+1};x]\quad (4.2)$$

$$N_{\beta,m+1}(y)=(t_{\beta+m+1}-t_{\beta})g_{m+1}[t_{\beta},t_{\beta+1},...,t_{\beta+m+1};y]\quad (4.3)$$

ただし点列 $\{s_i\}$， $\{t_j\}$ は 1 次元の B-spline 補間式(I)の場合と同じようにとる．

ここで $\hat{N}_{\alpha,i}$, $\hat{N}_{\beta,i}$ を次のように定義する．

$$\hat{N}_{\alpha,i}=\begin{cases}N_{\alpha,m+1}^{(l-i+1)}(x_1): i=1,2,...,l+1\\ N_{\alpha,m+1}(x_{i-l}): i=l+2,l+3,...,n_x+l-1\\ N_{\alpha,m+1}^{(i-n_x-l)}(x_{n_x}): i=n_x+l,n_x+l+1,...,n_x+2l\end{cases}$$

$$\hat{N}_{\beta,j}=\begin{cases}N_{\beta,m+1}^{(l-j+1)}(y_1): j=1,2,...,l+1\\ N_{\beta,m+1}(y_{j-l}): j=l+2,l+3,...,n_y+l-1\\ N_{\beta,m+1}^{(j-n_y-l)}(y_{n_y}): j=n_y+l,n_y+l+1,...,n_y+2l\end{cases}$$

さて (4.1) の係数 $c_{\alpha,\beta}$は, 補間条件としての, (c) より一意的に決定される．(c) は (4.1)を使って,

$$\sum_{\beta=-m+1}^{n_y-1}\left\{\sum_{\alpha=-m+1}^{n_x-1}c_{\alpha,\beta}\hat{N}_{\alpha,i}\right\}\hat{N}_{\beta,j}=\hat{f}_{i,j}\quad (4.4)$$

と書けるが，これはいくつかの行列を定義することによって，もっと簡単に書き直すことができる．すなわち，行列 ***F, C, Φ, Ψ*** を以下のように定義する．

- ***F*** は $\hat{f}_{i,j}$ を要素とする(n_x+*m* − 1) 行 (n_y+*m* − 1) 列の行列
- ***C*** は $c_{\alpha,\beta}$ を要素とする(n_x+*m* − 1) 行 (n_y+*m* − 1) 列の行列
- ***Φ*** は第 *i* 行が $\hat{N}_{\alpha,i}$: α= −*m*+1, − *m*+2,..., n_x − 1 から成る(n_x+*m*−1) 行 (n_x+*m*−1) 列の行列.
- ***Ψ*** は第 *j* 行が $\hat{N}_{\beta,j}$: β = −*m*+1, −*m*+2,..., n_y-1 から成る(n_y+*m*-1) 行 (n_y+*m*−1) 列の行列.

これらを使えば，(4.4) は

$$\Phi\boldsymbol{C}\Psi^{T}=\boldsymbol{F}\quad (4.5)$$

と書ける．そこで目的の行列 ***C*** は，次のようにして解くことができる．まず

$$\Psi\boldsymbol{X}=\boldsymbol{F}^{T}\quad (4.6)$$

を (n_x+*m* − 1) 組の(n_y+*m* − 1) 元連立 1 次方程式と見なして ***X*** について解く．続いて

$$\Phi\boldsymbol{C}=\boldsymbol{X}^{T}\quad (4.7)$$

を (n_y+*m* − 1) 組の (n_x+*m* − 1) 元連立 1 次方程式と見なせば, ***C*** について解くことができる．行列 ***Φ*** 及び ***Ψ***は 1 次元の B-spline 補間式 (I).（サブルーチン BIC1 参照）を求める際の連立 1 次方程式に現れる係数行列と全く同じ形状をしている．

本サブルーチンでは(4.6),(4.7)の連立 1 次方程式をクラウト法（LU 分解法）で解いている（スレーブサブルーチン UMIO1,UCIO1,ULUI1 使用）

E12-32-3302 BICD3, DBICD3

B-spline 2 次元補間式 (III-III)
CALL BICD3(X,NX,Y,NY,FXY,K,M,C,XT,VW,ICON)

(1) 機能

xy 平面上の格子点 (x_i, y_j) $(x_1< x_2< ...< x_{n_x}, y_1< y_2< ...< y_{n_y})$ において関数値 $f_{ij}=f(x_i,y_j)$ が与えられたとき，双 *m*（奇数）次の B-spline2 次元補間式，

$$S(x,y)=\sum_{\beta=-m+1}^{n_y-m}\sum_{\alpha=-m+1}^{n_x-m}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \quad (1.1)$$

の補間係数 $c_{\alpha,\beta}$ を求める．

ここで $S(x,y)$ の節点は，*x* 方向には (1,2)，*y* 方向には (1.3) のように採る（図 BICD3-1 参照）

$$\xi_i=\begin{cases}x_1 & ,i=1\\ x_{i+(m-1)/2} & ,i=2,3,...,n_x-m\\ x_{n_x} & ,i=n_x-m+1\end{cases} \quad (1.2)$$

$$\eta_j=\begin{cases}y_1 & ,j=1\\ y_{j+(m-1)/2} & ,j=2,3,...,n_y-m\\ y_{n_y} & ,j=n_y-m+1\end{cases} \quad (1.3)$$

$m\geq 3$, $n_x\geq m+2$, $n_y\geq m+2$であること．

(2) パラメタ

X …… **入力**．*x* 方向の離散点 x_i．
大きさ n_x の 1 次元配列．

NX …… **入力**．x_i の個数 n_x．

Y …… **入力**．*y* 方向の離散点 y_j．
大きさ n_y の 1 次元配列．

NY …… **入力**．y_j の個数 n_y

FXY …… **入力**．関数値 f_{ij}．
FXY(K,NY)なる 2 次元配列．
FXY(I,J) には f_{ij} を入力する．

K …… **入力**．2 次元配列 FXY 及び C の整合寸法 (K≥NX)．

M …… **入力**．B-spline の次数 *m*．
（使用上の注意②参照）．

C …… **出力**．補間係数 $c_{\alpha,\beta}$．
C(K,NY)なる 2 次元配列．
C(I,J)には $c_{-m+i,-m+j}$ を出力する．

XT …… **出力**．節点列 $\{\xi_i\}$ 及び $\{\eta_j\}$．
大きさ$(n_x-m+1)+(n_y-m+1)$の 1 次元配列．
$\{\xi_i\}$，$\{\eta_j\}$の順に出力する．

VW …… **作業領域**．次式で示す大きさの 1 次元配列．
$\{\max(n_x,n_y)-2\}m+2(m+1)+2\max(n_x,n_y)$．

ICON …… **出力**．コンディションコード．
表 BICD3-1 参照．

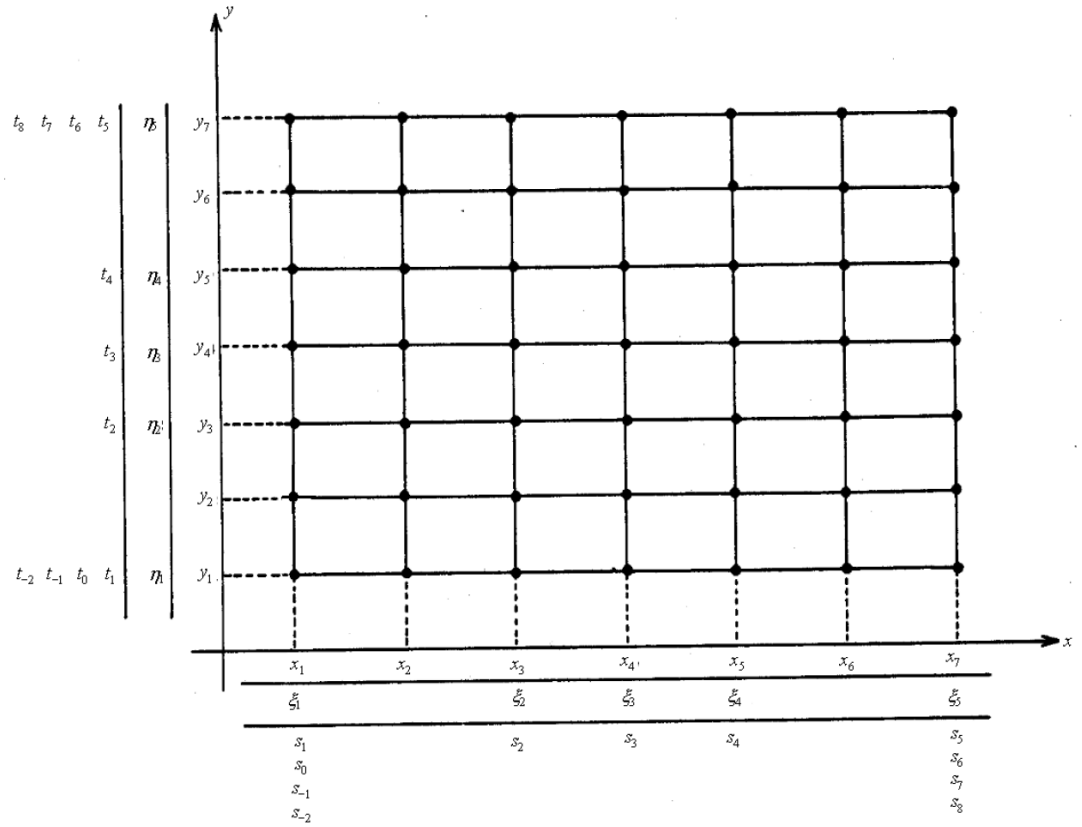


図 BICD3-1 節点列 $\{\xi_i\}\{\eta_j\}$ ($n_x=n_y=7$, $m=3$ の場合)

表 BICD3-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① M が奇数でない. ② NX<M+2 又は NY<M+2 ③ $x_i \geq x_{i+1}$ なる x_i がある. ④ $y_j \geq y_{j+1}$ なる y_j がある. ⑤ M<3	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UMIO3, UCIO3, UBAS1, ULUI3

② FORTRAN 基本関数 ... MOD, FLOAT

b. 注意

① 本サブルーチンに続けて，サブルーチン BIFD3 を呼び出すことにより，B-spline 2 次元補間式 (1.1) に基づく補間値あるいは偏微分値あるいは二重積分値を求めることができる．そのとき，パラメタ X，NX, Y, NY, K, M, C, XT の値は BICD3 の入力となる．

② 次数 m は 3 又は 5 程度がよい．倍精度では，元の関数がおとなしく，f_{ij} が高精度で与えられているならば，もう少し次数を高くしてもよい．しかし 15 を超えるとよくないといわれている．

c. 使用例

サブルーチンBIFD3の使用例参照.

(4) 手法概要

ここで求める B-spline 2 次元補間式 $S(x,y)$ は，サブルーチン BIC3 が求める B-spline 補間式 (III) をそのまま 2 次元へ拡張したものである．すなわち，x 方向，y 方向の節点列 $\{\xi_i\}$, $\{\eta_j\}$ をそれぞれ (1.2), (1.3) のようにとるとき，$S(x,y)$ は領域

$$R=\{(x,y) \mid x_1 \leq x \leq x_{n_x},\ y_1 \leq y \leq y_{n_y}\}$$

上で定義され，次の条件を満たす関数である.

(a) $S(x,y)$ は各部分領域

$$R_{i,j} = \left\{(x,y) \middle| \xi_i \leq x \leq \xi_{i+1},\ \eta_j \leq y \leq \eta_{j+1}\right\}$$

で高々，双 m 次の多項式である．双 m 次とは，x 方向，y 方向共に m 次であることをいう.

(b) $S(x,y) \in C^{m-1,m-1}$ [R]. すなわち，$\lambda = 0, 1, ..., m-1$, $\mu = 0, 1, \cdots, m-1$ とするとき，

$\dfrac{\partial^{\lambda+\mu}}{\partial x^{\lambda} \partial y^{\mu}} S(x,y)$ が存在して，しかも連続である.

(c) $S(x_i,y_j)=f_{ij}$, $i=1,2,...,n_x$, $j=1,2,...,n_y$.

(a), (b) を満たす $S(x,y)$ は $c_{\alpha,\beta}$ を任意定数として，

$$S(x,y) = \sum_{\beta=-m+1}^{n_y-m} \sum_{\alpha=-m+1}^{n_x-m} c_{\alpha,\beta} N_{\alpha,m+1}(x) N_{\beta,m+1}(y) \quad (4.1)$$

と書き表すことができる．ここで $N_{\alpha,m+1}(x)$, $N_{\beta,m+1}(y)$ は共に m 次の B-spline で (4.2), (4.3) で表す.

$$N_{\alpha,m+1}(x) = (s_{\alpha+m+1} - s_{\alpha}) g_{m+1}[s_{\alpha}, s_{\alpha+1}, ..., s_{\alpha+m+1}; x] \quad (4.2)$$

$$N_{\beta,m+1}(y) = (t_{\beta+m+1} - t_{\beta}) g_{m+1}[t_{\beta}, t_{\beta+1}, ..., t_{\beta+m+1}; y] \quad (4.3)$$

ただし点列 $\{s_i\}$, $\{t_j\}$ は，1 次元の B-spline 補間式 (III) の場合と同じようにとる．その例を図 BICD3-1 に示す.

さて，(4.1) の係数 $c_{\alpha,\beta}$ は，補間条件としての (c) より一意的に決定される. (c) は，(4.1) を使って，

$$\sum_{\beta=-m+1}^{n_y-m} \left\{ \sum_{\alpha=-m+1}^{n_x-m} c_{\alpha,\beta} N_{\alpha,m+1}(x_i) \right\} N_{\beta,m+1}(y_i) = f_{ij} \quad (4.4)$$

$$i = 1,2,...,n_x,\ j = 1,2,...,n_y$$

と書けるが，これは，いくつかの行列を定義することによって，もっと簡単に書き直すことができる．すなわち行列 $\boldsymbol{F}$, $\boldsymbol{C}$, $\boldsymbol{\Phi}$, $\boldsymbol{\Psi}$ を以下のように定義する.

- $\boldsymbol{F}$ は，f_{ij} を要素とする n_x 行 n_y 列の行列.
- $\boldsymbol{C}$ は，$c_{\alpha,\beta}$ を要素とする n_x 行 n_y 列の行列.
- $\boldsymbol{\Phi}$ は，第 i 行が $N_{\alpha,m+1}(x_i)$, $\alpha = -m+1, ..., n_x - m$ からなる n_x 行 n_x 列の行列.
- $\boldsymbol{\Psi}$ は，第 j 行が $N_{\beta,m+1}(y_j)$, $\beta = -m+1, ..., n_y - m$ からなる n_y 行 n_y 列の行列.

これらを使えば (4.4) は,

$$\boldsymbol{\Phi C \Psi}^T = \boldsymbol{F} \quad (4.5)$$

と書ける．そこで目的の行列 $\boldsymbol{C}$ は次のように解くことができる．まず,

$$\boldsymbol{\Psi X} = \boldsymbol{F}^T \quad (4.6)$$

を n_x 組の，n_y 元連立方程式と見なして $\boldsymbol{X}$ について解く．続いて,

$$\boldsymbol{\Phi C} = \boldsymbol{X}^T \quad (4.7)$$

を n_y 組の n_x 元連立方程式と見なせば $\boldsymbol{C}$ について解くことができる．行列 $\boldsymbol{\Phi}$ 及び $\boldsymbol{\Psi}$ は，1 次元の B-spline 補間式 (III)（サブルーチン BIC3 参照）．をもとめる際の連立 1 次方程式に現れる係数行列と全く同じ形状をしている．

本サブルーチンは，(4.6), (4.7) の連立 1 次方程式をクラウト法（LU 分解法）で解いている（スレーブサブルーチン UMIO3, UCIO3, ULUI3 使用）．

E12-31-0102 BIC1, DBIC1

B-spline 補間式 (I)
CALL BIC1(X,Y,DY,N,M,C,VW,ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n (x_1 < x_2 < ... < x_n)$ において関数値 $y_i = f(x_i)$, i=1, 2, ..., n が与えられ，更に両端 x_1, x_n において微係数，$y_1^{(l)} = f^{(l)}$, $y_n^{(l)} = f^{(l)}(x_n)$, l=1, 2, ..., $(m-1)/2$ が与えられたとき，m （奇数）次の正規化された B-spline の線型結合による補間式

$$S(x) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \tag{1.1}$$

の補間係数 c_j , $j = -m+1, -m+2, ..., n-1$ を求める．

ここで求める $S(x)$ は，

$$\begin{cases} S^{(l)}(x_1) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}^{(l)}(x_1) = y_1^{(l)} \\ l = 0,1,...,(m-1)/2 \\ S(x_i) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x_i) = y_i \\ i = 2,3,...,n-1 \\ S^{(l)}(x_n) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}^{(l)}(x_n) = y_n^{(l)} \\ l = (m-1)/2,(m-1)/2-1,...,0 \end{cases} \tag{1.2}$$

を満たすものである．

$m \geq 3$, $n \geq 2$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.

Y …… **入力**．関数値 y_i.
大きさ n の 1 次元配列.

DY …… **入力**．両端 x_1, x_n における微係数.
DY(2,$(m-1)/2$) なる 2 次元配列.
DY(1,l) に $y_1^{(l)}$，DY(2, l) に $y_n^{(l)}$,
l=1,2,...,$(m-1)/2$ を入力する.

N …… **入力**．離散点の個数 n.

M …… **入力**．B-spline の次数 m.
（使用上の注意②参照）

C …… **出力**．補間係数 c_j.
大きさ $n+m-1$ の 1 次元配列.

VW …… **作業領域**.
大きさ $(n-2)m+(m+1)^2/2+(m+1)$ の 1 次元配列.

ICON …… **出力**. コンディションコード.
表 BIC1-1 参照.

表 BIC1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① M が奇数でない. ② $x_i \geq x_{i+1}$ なる x_i がある. ③ M<3. ④ N<2.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UMIO1, UCIO1, UBAS1, ULUI1

② FORTRAN 基本関数 ... MOD, FLOAT

b. 使用上の注意

① 本サブルーチンに続けて，サブルーチン BIF1 を呼び出すことにより，B-spline 補間式 (1.1) に基づく補間値あるいは微分値あるいは積分値を求めることができる．その時，パラメタ X, N, M, C の値はサブルーチン BIF1 の入力となる.

② 次数 m は 3 又は 5 程度がよい．倍精度では，元の関数がおとなしく，y_i が高精度で与えられているならば，もう少し次数を高くしてもよい．しかし，15 程度が限界である.

c. 使用例
サブルーチンBIF1の使用例参照.

(4) 手法概要

ここで求める m 次の B-spline 補間式 $S(x)$ は，下記の (a) ～ (d) の条件を満たす，区間 $[x_1,x_n]$ で定義される関数である.

(a) $S(x)$ は区間 $[x_i,x_{i+1})$, i=1, 2, ..., $n-1$ で高々m 次の多項式である.

(b) $S(x) \in C^{m-1}[x_1,x_n]$. すなわち $S(x)$ は区間 $[x_1,x_n]$ でその $(m-1)$ 階導関数までが連続である.

(c) $S(x_i)=y_i$, i=2, 3, ..,n-1

(d) $S^{(l)}(x_1)=y_1^{(l)}$, $S^{(l)}(x_n)=y_n^{(l)}$
l = 0, 1, ..., (m-1)/2

節点列$\{t_j\}$, $j = -m+1, -m+2, ..., n+m$を

$$t_j = \begin{cases} x_1 : j = -m+1, -m+2, ..., 1 \\ x_j : j = 2,3,...,n-1 \\ x_n : j = n, n+1, ..., n+m \end{cases} \tag{4.1}$$

と採って定義される m 次の spline 関数は c_j を任意の定数として，

$$S(x) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \tag{4.2}$$

と表せる．ここで$N_{j,m+1}(x)$ はm次のB-splineで，

$$N_{j,m+1}(x)=(t_{j+m+1}-t_j)g_{m+1}[t_j,t_{j+1},...,t_{j+m+1};x]$$

である（詳しくは７章7.1の (3) 参照）．
$N_{j,m+1}(x)$ の局所性

$$N_{j,m+1}(x_i)\begin{cases}>0: j=i-m,i-m+1,...,i-1\\ =0:\begin{cases}j=-m+1,-m+2,...,i-m-1\\ j=i,i+1,...,n-1\end{cases}\end{cases}$$

$$N^{(l)}_{j,m+1}(x_1)\begin{cases}\neq 0: j=-m+1,-m+2,...,-m+1+l\\ =0: j=-m+2+l,-m+3+l,...,n-1\end{cases}$$

$$N^{(l)}_{j,m+1}(x_n)\begin{cases}=0: j=-m+1,-m+2,...,n-2-l\\ \neq 0: j=n-1-l,n-l,...,n-1\end{cases}$$

(4.3)

を考慮して，式 (4.2) に補間条件としての上記 (c) (d) を適用すると，$c_j : j= -m+1, -m+2,...,n-1$を未知数とする$n+m-1$元の連立方程式,

$$\begin{cases}\displaystyle\sum_{j=-m+1}^{-m+1+l}c_jN^{(l)}_{j,m+1}(x_1)=y_1^{(l)}: l=0,1,...,(m-1)/2\\ \displaystyle\sum_{j=i-m}^{i-1}c_jN_{j,m+1}(x_i)=y_i: i=2,3,...,n-1\\ \displaystyle\sum_{j=n-1-l}^{n-1}c_jN^{(l)}_{j,m+1}(x_n)=y_n^{(l)}: l=(m-1)/2,\\ \qquad\qquad (m-1)/2-1,..,0\end{cases}\tag{4.4}$$

が得られる．これを解くことにより，すべての補間係数c_jを求めることができる. 連立１次方程式 (4.4) の係数行列の形状を，一例として m=5, n=8 の場合を図 BIC1-1 に示す.

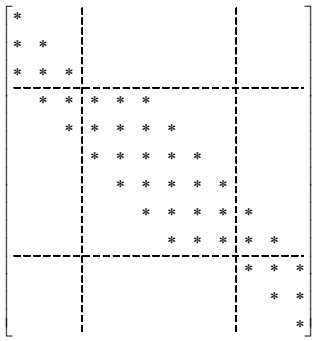


図 BIC1-1 係数行列 (m=5, n=8 の場合)

本サブルーチンではこの連立 1 次方程式をクラウト法（LU 分解法）で解いている（スレーブサブルーチン UMIO1, ULUI1, UCIO1 使用）.

E12-31-0202 BIC2, DBIC2

B-spline 補間式 (II)
CALL BIC2(X,Y,DY,N,M,C,VW,ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n (x_1 < x_2 < ... < x_n)$ において関数値 $y_i = f(x_i)$, i=1, 2, ..., n が与えられ，更に両端 x_1, x_n において微係数 $y_1^{(l)} = f^{(l)}(x_1)$, $y_n^{(l)} = f^{(l)}(x_n)$, l= (m+1)/2, (m+1)/2+1, ..., m – 1 が与えられたとき，m (奇数) 次の正規化された B-spline の線型結合による補間式,

$$S(x) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

の補間係数 c_j , j= –m+1, –m+2, ..., n – 1 を求める.
ここで求める $S(x)$ は,

$$S^{(l)}(x_1) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}^{(l)}(x_1) = y_1^{(l)}$$
$$i = (m+1)/2, (m+1)/2+1, ..., m\text{-}1$$
$$S(x_i) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x_i) = y_i \qquad (1.2)$$
$$i = 1, 2, ..., n$$
$$S^{(l)}(x_n) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}^{(l)}(x_n) = y_n^{(l)}$$
$$i = m\text{-}1, m\text{-}2, ..., (m+1)/2$$

を満たすものである.
m≥3, n≥(m+1)/2 であること.

(2) パラメタ

X …… **入力**. 離散点 x_i.
大きさ n の 1 次元配列.
Y …… **入力**. 関数値 y_i.
大きさ n の 1 次元配列.
DY …… **入力**. 両端 x_1, x_n における微係数.
DY(2,(m – 1)/2) なる 2 次元配列.
DY(1,l-(m – 1)/2) に $y_1^{(l)}$,
DY(2,l-(m – 1)/2) に $y_n^{(l)}$,
l=(m+1)/2,(m+1)/2+1,...,m – 1 を入力する.
N …… **入力**. 離散点の個数 n.
M …… **入力**. B-spline の次数 m.
（使用上の注意②参照）
C …… **出力**. 補間係数 c_j.
大きさ n+m – 1 の 1 次元配列.
VW …… **作業領域**.
大きさ m(n+m – 3)+2(m+1) の 1 次元配列.
ICON …… **出力**. コンディションコード.
表 BIC2-1 参照.

表 BIC2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① M が奇数でない. ② $x_i \geq x_{i+1}$ なる x_i がある. ③ M<3 ④ N<(M+1)/2	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UMIO2, UCIO2, UBAS1, ULUI1
② FORTRAN 基本関数 ... MOD, FLOAT

b. 使用上の注意
① 本サブルーチンに続けて，サブルーチン BIF2 を呼び出すことにより，B-spline 補間式 (1.1) に基づく補間値あるいは微分値あるいは積分値を求めることができる．その時，パラメタ X, N, M, C の値はサブルーチン BIF2 の入力となる.
② 次数 m は 3 又は 5 程度がよい．倍精度では，元の関数がおとなしく, y_i が高精度で与えられているならば，もう少し次数を高くしてもよい．しかし，15 程度が限界である.

c. 使用例
サブルーチン BIF2 の使用例参照.

(4) 手法概要

ここで求める m 次の B-spline 補間式 $S(x)$ は，下記の (a) ～ (d) の条件を満たす，区間 $[x_1,x_n]$ で定義される関数である.
(a) $S(x)$ は区間 $[x_i,x_{i+1})$, i=1, 2, ..., n – 1 で高々m 次の多項式である.
(b) $S(x) \in C^{m-1}[x_1,x_n]$. すなわち $S(x)$ は区間 $[x_1,x_n]$ でその (m – 1) 階導関数までが連続である.
(c) $S(x_i)=y_i$, i=1, 2, .., n
(d) $S^{(l)}(x_1)=y_1^{(l)}$, $S^{(l)}(x_n)=y_n^{(l)}$
l = (m+1)/2, (m+1)/2+1, ..., m-1

節点列$\{t_j\}$, j = –m+1, –m+2, ..., n+m を

$$t_j = \begin{cases} x_1 & : j = -m+1, -m+2, ..., 1 \\ x_j & : j = 2, 3, ..., n-1 \\ x_n & : j = n, n+1, ..., n+m \end{cases} \qquad (4.1)$$

と採って定義される m 次の spline 関数は c_j を任意の定数として,

$$S(x) = \sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (4.2)$$

と表せる．ここで $N_{j,m+1}(x)$ は m 次の B-spline で，

$$N_{j,m+1}(x)=(t_{j+m+1}-t_j)g_{m+1}[t_j,t_{j+1},...,t_{j+m+1};x]$$

である（詳しくは 7 章 7.1 の (3) 参照）．
$N_{j,m+1}(x)$ の局所性

$$N_{j,m+1}(x_i)\begin{cases}>0: j=i-m,i-m+1,...,i-1\\ =0:\begin{cases}j=-m+1,-m+2,...,i-m-1\\ j=i,i+1,...,n-1\end{cases}\end{cases}$$

$$N^{(l)}_{j,m+1}(x_1)\begin{cases}\neq 0: j=-m+1,-m+2,...,-m+1+l\\ =0: j=-m+2+l,-m+3+l,...,n-1\end{cases}$$

$$N^{(l)}_{j,m+1}(x_n)\begin{cases}=0: j=-m+1,-m+2,...,n-2-l\\ \neq 0: j=n-1-l,n-l,...,n-1\end{cases}$$

(4.3)

を考慮して，式 (4.2) に補間条件としての上記 (c), (d) を適用すると，$c_j : j= -m+1, -m+2,...,n-1$ を未知数とする $n+m-1$ 元の連立方程式，

$$\begin{cases}\displaystyle\sum_{j=-m+1}^{-m+1+l}c_jN^{(l)}_{j,m+1}(x_1)=y^{(l)}_1: l=(m+1)/2,(m+1)/2+1,\\ \qquad\qquad ...,m-1\\ \displaystyle\sum_{j=i-m}^{i-1}c_jN_{j,m+1}(x_i)=y_i: i=1,2,...,n\\ \displaystyle\sum_{j=n-1-l}^{n-1}c_jN^{(l)}_{j,m+1}(x_n)=y^{(l)}_n: l=m-1,m-2,...,(m+1)/2\end{cases}$$

(4.4)

が得られる．これを解くことにより，すべての補間係数 c_j を求めることができる. 連立 1 次方程式 (4.4) の係数行列の形状を，一例として m=5, n=8 の場合を図 BIC2-1 に示す.

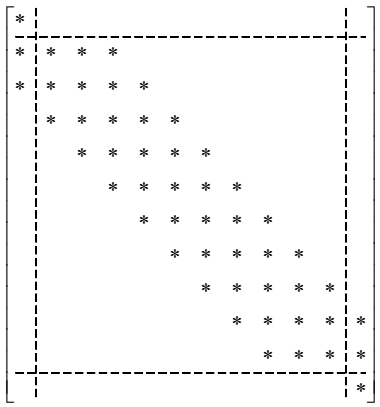


図 BIC2-1 係数行列 (m=5, n=8 の場合)

本サブルーチンではこの連立 1 次方程式をクラウト法 (LU 分解法) で解いている (スレーブサブルーチン UMIO2, ULUI2, UCIO2 使用).

E12-31-0302 BIC3, DBIC3

B-spline 補間式 (III)
CALL BIC3(X,Y,N,M,C,XT,VW,ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n (x_1<x_2<...<x_n)$ に対して，関数値 $y_i=f(x_i), i=1,2,...,n$ が与えられたとき，

$$\xi_1 = x_1$$
$$\xi_i = x_{i+(m-1)/2}, \quad i = 2,3,...,n-m$$
$$\xi_{n-m+1} = x_n$$

を節点とする m (奇数) 次の正規化された B-spline の線型結合による補間式

$$S(x) = \sum_{j=-m+1}^{n-m} c_j N_{j,m+1}(x) \qquad (1.1)$$

の補間係数$c_j, j = -m+1, -m+2, \ldots, n-m$を求める．

$m \geq 3, n \geq m+2$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
　　大きさ n の 1 次元配列.
Y …… **入力**．関数値 y_i.
　　大きさ n の 1 次元配列.
N …… **入力**．離散点の個数 n.
M …… **入力**．B-spline の次数 m.
　　（使用上の注意②参照）
C …… **出力**．補間係数 c_j.
　　大きさ n の 1 次元配列.
XT …… **出力**．節点列ξ_i.
　　大きさ $n-m+1$ の 1 次元配列.
VW …… **作業領域**．大きさ $mn+2$ の 1 次元配列.
ICON …… **出力**．コンディションコード.
　　表 BIC3-1 参照.

表 BIC3-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① M が奇数でない. ② N<M+2 ③ $x_i \geq x_{i+1}$ なる x_i がある. ④ M<3	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... MGSSL, UMIO3, UCIO3, UBAS1, ULUI3

② FORTRAN基本関数 ..... MOD

b. 注意

① 本サブルーチンに続けて，サブルーチン BIF3 を呼び出すことにより，B-spline 補間式 (1.1) に基づく補間値あるいは微分値あるいは積分値を求めることができる．そのとき，パラメタ X, N, M, C, XT の値はサブルーチン BIF3 の入力となる．

② 次数 m は 3 又は 5 程度がよい．倍精度では，元の関数がおとなしく，y_i が高精度で与えられているならば，もう少し次数を高くしてもよい．しかし，15 を超えると良くないといわれている．

c. 使用例

サブルーチンBIF3の使用例を参照.

(4) 手法概要

離散点 $x_1,x_2,...,x_n (x_1<x_2<...<x_n)$ において，関数値 $y_i=f(x_i), i=1,2,...,n$ が与えられたとき，ここで求める m 次の spline 補間式 $S(x)$ とは次の条件を満たす，区間 $[x_1,x_n]$ で定義される関数である．

(a) $S(x)$ は区間 $[\xi_i, \xi_{i+1}], i=1, 2, \cdots, n-m$ で高々m 次の多項式である．ここで，$\xi_1=x_1, \xi_i=x_{i+(m-1)/2}$ ，$i=2, 3, \cdots, n-m$ ，$\xi_{n-m+1}=x_n$ である．

(b) $S(x) \in C^{m-1}[x_1,x_n]$. すなわち $S(x)$ は区間 $[x_1,x_n]$ でその $(m-1)$ 階導関数までが連続である．

(c) $S(x_i)=y_i, i=1,2,..,n$.

(a), (b), (c) を満たす $S(x)$ は $c_j, j=-m+1, -m+2,...,n-m$ を定数として，(4.1) で与えられる．

$$S(x) = \sum_{j=-m+1}^{n-m} c_j N_{j,m+1}(x) \qquad (4.1)$$

ここで $N_{j,m+1}(x)$ は m 次の正規化された B-spline であり，

$$N_{j,m+1}(x) = (t_{j+m+1} - t_j) g_{m+1}[t_j, t_{j+1}, ..., t_{j+m+1}; x] \qquad (4.2)$$

と定義される（詳しくは 7.1 の (3) 参照）．

B-spline の節点列$\{t_r\}$は，本サブルーチンでは

$$t_r = \begin{cases} \xi_1 & , r = -m+1, -m+2, ..., 1 \\ \xi_r & , r = 2,3,..., n-m \\ \xi_{n-m+1} & , r = n-m+1, n-m+2, ..., n+1 \end{cases} \qquad (4.3)$$

としている．このように$\{t_r\}$を設定した場合の $N_{j,m+1}(x)$ の一例として図BIC3-1にグラフを示す.

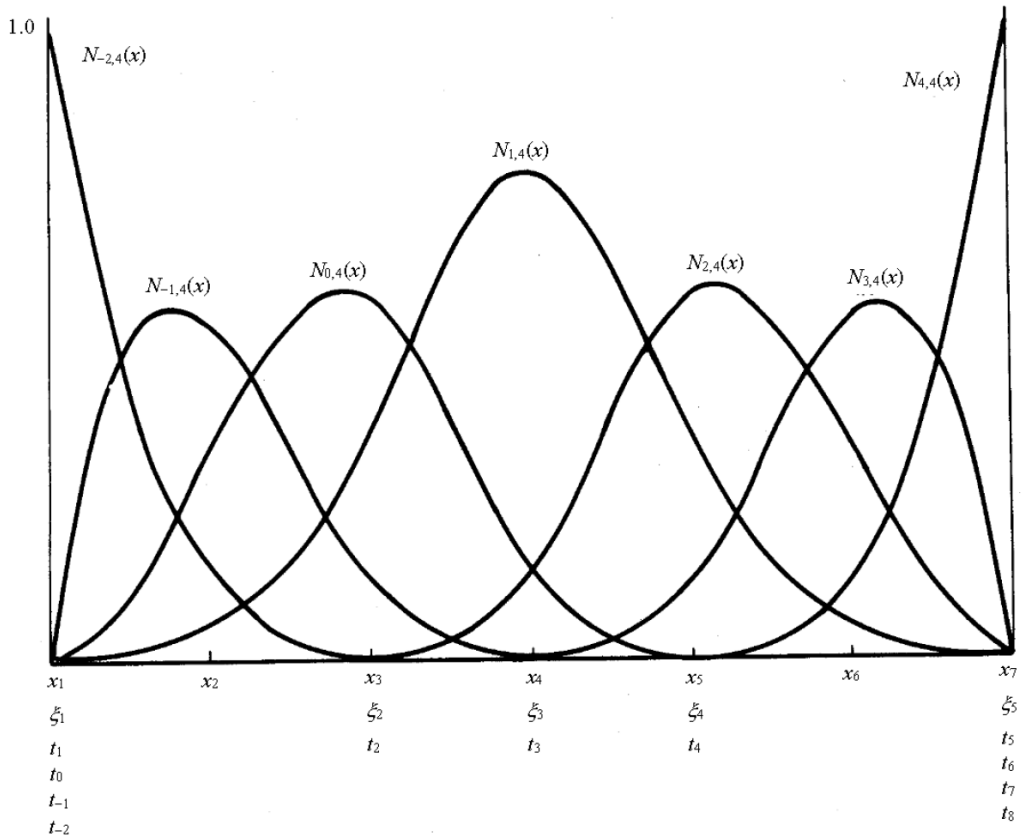


図 BIC3-1 B-spline $N_{j,m+1}(x)$ の例 : 等間隔な 7 個の離散点で m=3 の場合

補間係数 c_j は補間の条件を満たす連立 1 次方程式,

$$S(x_i)=\sum_{j=-m+1}^{n-m} c_j N_{j,m+1}(x_i)=y_i, i=1,2,...,n \qquad (4.4)$$

を解くことにより決定する．この方程式の係数行列は $N_{j,m+1}(x_i)$ から成るが，B-spline の局所性のため零要素を多く持ち，バンド行列に似た形状となる. 係数行列の一例として, 図 BIC3-2 に n=7, m=3 の場合を示す.

$$\begin{bmatrix} * & & & & & & \\ * & * & * & * & & 0 & \\ & * & * & * & & & \\ & & * & * & * & & \\ & & & * & * & * & \\ & 0 & & * & * & * & * \\ & & & & & & * \end{bmatrix}$$

図 BIC3-2 係数行列の例 (n =7, m =3 の場合)

本サブルーチンはこの連立 1 次方程式をクラウト法（LU 分解法）で解いている．（スレーブサブルーチン ULUI3 使用）.

E12-31-0402 BIC4, DBIC4

B-spline 補間式 (IV)
CALL BIC4(X,Y,N,M,C,VW,ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ ($x_1<x_2<...<x_n$) に対して，周期 (x_n-x_1) の周期関数値 $y_i=f(x_i)$, i=1, 2, ..., n (ただし $y_1=y_n$) が与えられたとき，m (奇数) 次の正規化された B-spline の線型結合による補間式,

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

の補間係数 c_j, $j=-m+1, -m+2, ..., n-1$ を求める．
ここで求める $S(x)$ は $f(x)$ と同じく周期 (x_n-x_1) の周期関数で，境界条件,

$$S^{(l)}(x_1)=S^{(l)}(x_n), \quad l=0,1,...,m-1 \qquad (1.2)$$

を満たすものである．
$m\geq 3$, $n\geq m+2$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.
Y …… **入力**．関数値 y_i.
大きさ n の 1 次元配列.
$y_1=y_n$ であること．もし $y_1\neq y_n$ であったときは y_n の方を採用する．
N …… **入力**．離散点の個数 n.
M …… **入力**．B-spline の次数 m.
（使用上の注意②参照）．
C …… **入力**．補間係数 c_j.
大きさ $n+m-1$ の 1 次元配列.
VW …… **作業領域**．.
大きさ$(n-1)(2m-1)+m+1$ の 1 次元配列.
ICON …… **出力**．コンディションコード.
表 BIC4-1 参照.

表 BIC4-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① M が奇数でない. ② N<M+2 ③ $x_i\geq x_{i+1}$ なる x_i がある. ④ M<3	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UMIO4, UCIO4, UBAS4, ULUI4, UPEP4
② FORTRAN基本関数 ... MOD, FLOAT

b. 使用上の注意
① 本サブルーチンに続けて，サブルーチン BIF4 を呼び出すことにより，B-spline 補間式 (1.1) に基づく補間値あるいは微分値あるいは積分値を求めることができる．そのとき，パラメタ X, N, M, C の値はサブルーチン BIF4 の入力となる.
② 次数 m は 3 又は 5 程度がよい．倍精度では，元の関数がおとなしく，y_i が高精度で与えられているならば，もう少し次数を高くしてもよい．しかし 15 を超えると良くないといわれている.

c. 使用例
サブルーチンBIF4の使用例参照.

(4) 手法概要

ここで求める m 次の B-spline 補間式 $S(x)$ は，下記の (a) ～ (d) の条件を満たす区間 $[x_1,x_n]$ で定義される関数である．条件 (c) は周期的条件であり，本サブルーチン固有の条件である.
(a) $S(x)$ は区間 $[x_i,x_{i+j}]$, i=1, 2, ... $n-1$ で高々 m 次の多項式である.
(b) $S(x)\in C^{m-1}[x_1,x_n]$．すなわち $S(x)$ は区間 $[x_1,x_n]$ でその $(m-1)$ 階導関数までが連続である.
(c) $S^{(l)}(x_1)=S^{(l)}(x_n)$, l=0,1,...,$m-1$.
(d) $S(x_i)=y_i$, i=2,3,..,n ($y_1=y_n$ と見なす).

まず，(c) を満たす Spline 関数について述べる.
B-spline の節点列$\{t_j\}$を次のように採る.
（図 BIC4-1）.

$$t_j=\begin{cases} x_{n-1-j}-(x_n-x_1), & -m+1\leq j\leq 0 \\ x_j, & 1\leq j\leq n \\ x_{-n+1+j}+(x_n-x_1), & n+1\leq j\leq n+m \end{cases} \qquad (4.1)$$

この節点列に基づく m 次の Spline 関数は c_j を任意の定数として,

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (4.2)$$

と表せる．ここで $N_{j,m+1}(x)$ は m 次の B-spline で,

$$N_{j,m+1}(x)=(t_{j+m+1}-t_j)g_{m+1}[t_j,t_{j+1},...,t_{j+m+1};x]$$

である（詳しくは7.1の (3) 参照）.

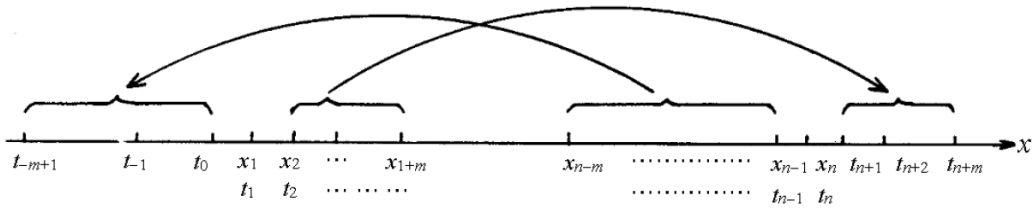


図 BIC4-1 節点列$\{t_j\}$

(4.2) に

$$c_j = c_{j+n-1} \quad ,j = -m+1,...,0 \tag{4.3}$$

という条件を加えると，周期条件としての上記 (c)が満たされる.

次に補間条件 (d) は,

$$\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x_i) = y_i \quad ,i = 2,...,n \tag{4.4}$$

と書ける．いま，$m=2l-1 (l \geq 2)$ として，(4.3) を次の (4.5) のように書き直す.

$$\begin{aligned} c_j &= c_{j+n-1} \quad ,j = -2l+2,...,-l+1 \\ c_j &= c_{j-n+1} \quad ,j = n-l+1,...,n-1 \end{aligned} \tag{4.5}$$

これを使って (4.4) を書き直すと,

$$\begin{aligned} &\sum_{j=-l+2}^{0} c_j \{N_{j,2l}(x_i) + N_{j+n-1,2l}(x_i)\} + \sum_{j=1}^{n-2l} c_j N_{j,2l}(x_i) \\ &+ \sum_{j=n-2l+1}^{n-l} c_j \{N_{j-n+1,2l}(x_i) + N_{j,2l}(x_i)\} = y_i \\ &\qquad i = 2,3,...,n \end{aligned} \tag{4.6}$$

を得る．これは，$(n-1)$ 個の c_j ($j=-l+2,-l+3,...,n-l$) を未知数とする $(n-1)$ 元連立方程式であるから，これを解き，更に (4.5) の関係を使うことにより，(4.2) で表せる Spline 補間式のすべての補間係数 c_j を求めることができる．ちなみに連立方程式 (4.6) の係数行列はバンド行列に似た形状となるが，一例として $m=5(l=3)$, $n=9$ の場合を図 BIC4-2 に示す.

$$\begin{bmatrix} * & * & * & & & & * & * \\ * & * & * & * & & & & * \\ * & * & * & * & * & & 0 & \\ & * & * & * & * & * & & \\ & & * & * & * & * & * & \\ & 0 & & * & * & * & * & * \\ * & & & & * & * & * & * \\ * & * & & & & * & * & * \end{bmatrix}$$

図 BIC4-2 係数行列 ($m=5$, $n=9$ の場合)

本サブルーチンはこの連立 1 次方程式をクラウト法（LU 分解法）で解いている．（スレーブサブルーチン UMIO4, ULUI4, UCIO4 使用）.

E11-32-1101 BIFD1, DBIFD1

B-spline 2 次元補間式 (I-I) による補間，数値微分，数値積分
CALL BIFD1(X,NX,Y,NY,M,C,K,ISWX,VX,IX,ISWY, VY,IY,F,VW,ICON)

(1) 機能

xy 平面上の格子点 (x_i, y_j), $(x_1 < x_2 < \cdots < x_{n_x}, y_1 < y_2 < \cdots < y_{n_y})$ において関数値 $f_{ij}= f(x_i, y_j)$ が与えられ，更に周囲において偏微係数

$$f_{1,j}^{(\lambda,0)} = f^{(\lambda,0)}\left(x_1, y_j\right), f_{n_x,j}^{(\lambda,0)} = f^{(\lambda,0)}\left(x_{n_x}, y_j\right)$$
$$f_{i,1}^{(0,\mu)} = f^{(0,\mu)}\left(x_i, y_1\right), f_{i,n_y}^{(0,\mu)} = f^{(0,\mu)}\left(x_i, y_{n_y}\right)$$
$$f_{11}^{(\lambda,\mu)} = f^{(\lambda,\mu)}\left(x_1, y_1\right), f_{n_x,1}^{(\lambda,\mu)} = f^{(\lambda,\mu)}\left(x_{n_x}, y_1\right)$$
$$f_{1,n_y}^{(\lambda,\mu)} = f^{(\lambda,\mu)}\left(x_1, y_{n_y}\right) f_{n_x,n_y}^{(\lambda,\mu)} = f^{(\lambda,\mu)}\left(x_{n_x}, y_{n_y}\right)$$
$$i = 1,2,...,n_x, j = 1,2,...,n_y$$
$$\lambda = 1,2,...,(m-1)/2, \mu = 1,2,...,(m-1)/2$$

が与えられたとき点 $P\ (v_x,\ v_y)$ における補間値あるいは偏微分値あるいは領域$\{(x,y) \mid x_1 \le x \le v_x\ , y_1 \le y \le v_y\}$上の二重積分値を求める（図 BIFD1-1 参照）.

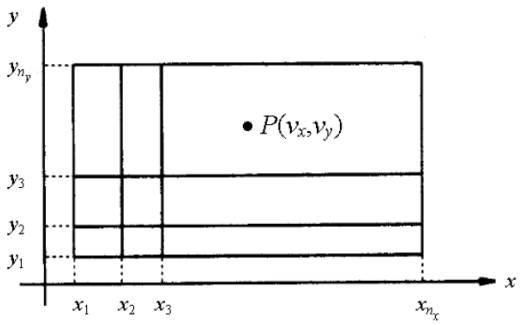


図 BIFD1-1 領域 $R=\{(x,y) \mid x_1 \le x \le x_{n_x}\ , y_1 \le y \le y_{n_y}\}$内の点 *P*

ただし，本サブルーチンを利用する前に，サブルーチン BICD1 により B-spline2 次元補間式

$$S(x,y) = \sum_{\beta=-m+1}^{n_y-1} \sum_{\alpha=-m+1}^{n_x-1} c_{\alpha,\beta} N_{\alpha,m+1}(x) N_{\beta,m+1}(y) \quad (1.1)$$

における補間係数 $c_{\alpha,\beta}$ が計算されているとする．ここで *m* は奇数で B-spline $N_{\alpha,m+1}(x)$ 及び $N_{\beta,m+1}(y)$ の次数である．$x_1 \le v_x \le x_{n_x}, y_1 \le v_y \le y_{n_y}$, $m \ge 3$ 及び $n_x \ge 2$, $n_y \ge 2$ であること．

(2) パラメタ

X …… **入力**．*x* 方向の離散点 x_i.
大きさ n_x の 1 次元配列.

NX …… **入力**．x_i の個数 n_x.

Y …… **入力**．*y* 方向の離散点 y_j.
大きさ n_y の 1 次元配列.

NY …… **入力**．y_j の個数 n_y.

M …… **入力**．B-spline の次数 *m*.
（使用上の注意,①参照）

C …… **入力**．補間係数 $c_{\alpha,\beta}$（BICD1 の出力）.
C(K,NY+M-1) なる 2 次元配列.

K …… **入力**．2 次元配列 C の整合寸法.
K≥NX+M-1 であること.

ISWX …… **入力**．*x* 方向に関する計算の種類を与える.
$-1 \le ISWX \le m$（パラメタ F の項参照）

VX …… **入力**．点 $P(v_x,v_y)$ の *x* 座標.

IX …… **入力**．$x_i \le v_x < x_{i+1}$ を満たす *i*.
$v_x = x_{n_x}$ のときは $IX = n_x - 1$ とする.
出力．$x_i \le v_x < x_{i+1}$ を満たす *i*.
（使用上の注意②参照）

ISWY …… **入力**．*y* 方向に関する計算の種類を与える.
$-1 \le ISWY \le m$（パラメタ F の項参照）

VY …… **入力**．点 $P(v_x,v_y)$ の *y* 座標.

IY …… **入力**．$y_j \le v_y < y_{j+1}$ を満たす *j*.
$v_y = y_{n_y}$ のときは $IY = n_y - 1$ とする.
出力．$y_j \le v_y < y_{j+1}$ を満たす *j*.
（使用上の注意②参照）.

F …… **出力**．補間値あるいは偏微分値あるいは積分値.
ISWX=λ, ISWY=μとすると，λ, μの組合せごとに次の値を出力する.

- $0 \le \lambda, \mu$のとき

$$F = \frac{\partial^{\lambda+\mu}}{\partial x^{\lambda} \partial y^{\mu}} S\left(v_x, v_y\right)$$

補間値はλ =μ = 0 とすることにより得られる.

- λ = –1, $0 \le \mu$のとき

$$F = \int_{x_1}^{v_x} \frac{\partial^{\mu}}{\partial y^{\mu}} S\left(x, v_y\right) dx$$

- λ ≥ 0, μ = –1のとき

$$F = \int_{y_1}^{v_y} \frac{\partial^{\lambda}}{\partial x^{\lambda}} S\left(v_x, y\right) dy$$

- λ = μ = –1のとき

$$F = \int_{y_1}^{v_y} dy \int_{x_1}^{v_x} S(x,y) dx$$

VW …… **作業領域**.
大きさ $4(m+1) + \max(n_x,n_y) + m - 1$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BIFD1-1 参照.

表 BIFD1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	X(IX)≤VX<X(IX+1) 又は Y(IY)≤VY<Y(IY+1) が満たされていない.	サブルーチン内で左記の IX 又は IY をさがして処理を続ける.
30000	次のいずれかであった. ① VX<X(1) 又は VX>X(NX) ② VY<Y(1) 又は VY>Y(NY) ③ ISWX<-1 又は ISWX>M ④ ISWY<-1 又は ISWY>M	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UCAD1, UBAS1

② FORTRAN基本関数 ... FLOAT

b. 注意

① 本サブルーチンは，サブルーチン BICD1 により求められた B-spline2 次元補間式 (1.1) に基づいて，補間値あるいは偏微分値あるいは二重積分値を求めるものである.
したがって，本サブルーチンを呼び出す前に BICD1 を呼び出して補間式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより補間値などを求めることができる．そのとき，パラメタ X, NX, Y, NY, K, M, C の値は BICD1 のそれらと同一でなければならない.

② パラメタ IX, IY はそれぞれ X(IX)≤VX<X(IX+1),Y(IY)≤VY≤Y(IY+1) を満足していることが望ましい．もし満足していないときは，これらを満足するような IX, IY を探索して処理を続ける.

c. 使用例

格子点 (x_i,y_j)，関数値 f_{ij}, i=1,2,...,n_x, j=1, 2, ..., n_y, 周囲における偏微係数 $f_{i,j}^{(\lambda,\mu)}$, i=1, n_x, j=1, n_y,λ=1, 2, ..., (m −1)/2, μ=1, 2, ..., (m −1)/2 及び次数 m を入力し，点 (v_{ir},u_{js}); $v_{ir}= x_i+(x_{i+1}- x_i)\times(r/4)$, $u_{js}=y_j+(y_{j+1}-y_j)\times(s/4)$, i=1, 2, ..., n_x−1, j=1, 2, ..., n_y−1 , r=0, 1, ..., 4, s=0, 1, …, 4 における補間値，偏微分値，あるいは領域$\{(x,y) \mid x_1 \le x \le v_{ir}\ , y_1 \le y \le u_{js}\}$上の積分値を求める． n_x≤30, n_y≤30, m≤5 の場合.
なお，FXY(I,J) に入力されるデータは，例えば m=5であれば

$$\begin{bmatrix} f_{11}^{(2,2)} & f_{11}^{(1,2)} & f_{11}^{(0,2)} & f_{21}^{(0,2)} & \cdots & f_{n_x,1}^{(0,2)} & f_{n_x,1}^{(1,2)} & f_{n_x,1}^{(2,2)} \\ f_{11}^{(2,1)} & f_{11}^{(1,1)} & f_{11}^{(0,1)} & f_{21}^{(0,1)} & \cdots & f_{n_x,1}^{(0,1)} & f_{n_x,1}^{(1,1)} & f_{n_x,1}^{(2,1)} \\ f_{11}^{(2,0)} & f_{11}^{(1,0)} & f_{11} & f_{21}, f_{31} & \cdots & f_{n_x,1} & f_{n_x,1}^{(1,0)} & f_{n_x,1}^{(2,0)} \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ f_{1,n_y}^{(2,0)} & f_{1,n_y}^{(1,0)} & f_{1,n_y} & f_{2,n_y} & \cdots & f_{n_x,n_y} & f_{n_x,n_y}^{(1,0)} & f_{n_x,n_y}^{(2,0)} \\ f_{1,n_y}^{(2,1)} & f_{1,n_y}^{(1,1)} & f_{1,n_y}^{(0,1)} & f_{2,n_y}^{(0,1)} & \cdots & f_{n_x,n_y}^{(0,1)} & f_{n_x,n_y}^{(1,1)} & f_{n_x,n_y}^{(2,1)} \\ f_{1,n_y}^{(2,2)} & f_{1,n_y}^{(1,2)} & f_{1,n_y}^{(0,2)} & f_{2,n_y}^{(0,2)} & \cdots & f_{n_x,n_y}^{(0,2)} & f_{n_x,n_y}^{(1,2)} & f_{n_x,n_y}^{(2,2)} \end{bmatrix}$$

のように与えられていること.

```
C     **EXAMPLE**
      DIMENSION X(30),Y(30),FXY(32,32),
     *C(32,32),VW(232),R(5,5)
      READ(5,500) NX,NY,M
      READ(5,510) (X(I),I=1,NX),
     *            (Y(J),J=1,NY)
      NXM=NX+M-1
      NYM=NY+M-1
      READ(5,510) ((FXY(I,J),I=1,NXM),
     *J=1,NYM)
      WRITE(6,600) M,(I,X(I),I=1,NX)
      WRITE(6,610) (J,Y(J),J=1,NY)
      WRITE(6,620) ((I,J,FXY(I,J),I=1,NXM),
     *J=1,NYM)
      K=32
      CALL BICD1(X,NX,Y,NY,FXY,K,M,C,
     *VW,ICON)
      IF(ICON.EQ.0) GO TO 20
      WRITE (6,630)
      STOP
   20 NX1=NX-1
      NY1=NY-1
      M2=M+1
      DO 80 LY2=1,M2
      ISWY=LY2-2
      DO 70 LX2=1,M2
      ISWX=LX2-2
      WRITE(6,640) ISWX,ISWY
      DO 60 IY=1,NY1
      HY=(Y(IY+1)-Y(IY))*0.25
      YJ=Y(IY)
      DO 50 IX=1,NX1
      HX=(X(IX+1)-X(IX))*0.25
      XI=X(IX)
      DO 40 J=1,5
      VY=YJ+HY*FLOAT(J-1)
      DO 30 I=1,5
      VX=XI+HX*FLOAT(I-1)
      IXX=IX
      IYY=IY
      CALL BIFD1(X,NX,Y,NY,M,C,K,ISWX,
     * VX,IXX,ISWY,VY,IYY,F,VW,ICON)
      R(I,J)=F
   30 CONTINUE
   40 CONTINUE
      WRITE(6,650) IXX,IYY,((R(I,J),I=1,5),
     *J=1,5)
   50 CONTINUE
   60 CONTINUE
   70 CONTINUE
   80 CONTINUE
      STOP
```

```
500 FORMAT(3I6)
510 FORMAT(2F12.0)
600 FORMAT('1'//10X, 'INPUT DATA',3X,'M=',
   *I2//20X,'NO.',10X,'X'/
   *(20X,I3,E18.7))
610 FORMAT(//20X,'NO.',10X,'Y'/(20X,I3,
   *E18.7))
620 FORMAT(3(10X,'FXY(',I2,',',I2,')=',
   *E15.7))
630 FORMAT('0',10X,'ERROR')
640 FORMAT(//5X,'ISWX=',I2,3X,'ISWY=',
   *I2//)
650 FORMAT(10X,'IX=',I3,2X,'IY=',I3/
   *(15X,5(5X,E15.7)))
    END
```

(4) 手法概要

サブルーチン BICD1 により双 m 次の B-spline2 次元補間式

$$S(x,y)=\sum_{\beta=-m+1}^{n_y-1}\sum_{\alpha=-m+1}^{n_x-1}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \quad (4.1)$$

における補間係数 $c_{\alpha,\beta}$ が求まっているとする.

本サブルーチンでは (4.1) の補間式に基づいて，補間値，偏微分値，あるいは積分値などを計算する. その方法は，7 章 (7.1) の「(5)2 変数 Spline 関数の定義，表現及び計算法」で述べたのでそこを参照されたい.

E11-32-3301 BIFD3, DBIFD3

B-spline2 次元補間式 (III-III)による補間，数値微分，数値積分
CALL BIFD3 (X,NX,Y,NY,M,C,K,XT,ISWX,VX, IX, ISWY,VY,IY,F,VW,ICON)

(1) 機能

xy 平面上の格子点 (x_i, y_j), $(x_1 < x_2 < \cdots < x_{n_x}, y_1 < y_2 < \cdots < y_{n_y})$ において関数値 $f_{ij}=f(x_i,y_j)$ が与えられたとき，点 $P(v_x,v_y)$ における補間値あるいは偏微分値あるいは領域 $[x_1 \le x \le v_x\ ,\ y_1 \le y \le v_y]$ 上の二重積分値を求める．（図 BIFD3-1 参照）．

ただし，本サブルーチンを利用する前に，サブルーチン BICD3 により，*x* 方向の節点列$\{\xi_j\}$及び *y* 方向の節点列$\{\eta_j\}$，更に B-spline2 次元補間式，

$$S(x,y)=\sum_{\beta=-m+1}^{n_y-m}\sum_{\alpha=-m+1}^{n_x-m}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \quad (1.1)$$

における補間係数 $c_{\alpha,\beta}$ が計算されているとする．ここで *m* は奇数で B-spline $N_{\alpha,m+1}(x)$ 及び $N_{\beta,m+1}(y)$ の次数である．$x_1 \le v_x \le x_{n_x}$, $y_1 \le v_y \le y_{n_y}$, $m \ge 3$ 及び $n_x \ge m+2$, $n_y \ge m+2$ であること．

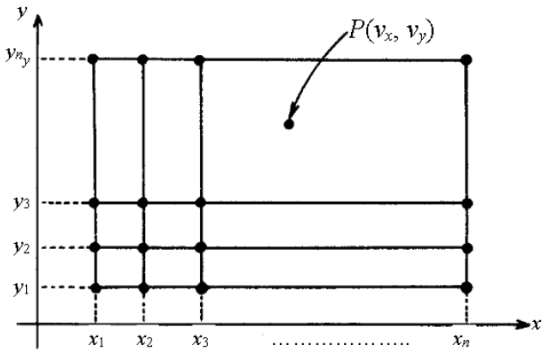


図 BIFD3-1 領域 $R=\{(x,y) \mid x_1 \le x \le x_{n_x}, y_1 \le y \le y_{n_y}\}$内の *P*

(2) パラメタ

X …… **入力**．*x* 方向の離散点 x_i.
大きさ n_x の 1 次元配列.

NX …… **入力**． x_i の個数 n_x.

Y …… **入力**．*y* 方向の離散点 y_j.
大きさ n_y の 1 次元配列.

NY …… **入力**．y_j の個数 n_y.

M …… **入力**．B-spline の次数 *m*.
（使用上の注意①参照）．

C …… **入力**．補間係数 $C_{\alpha,\beta}$ (BICD3 の出力).
C(K,NY) なる 2 次元配列.

K …… **入力**．2 次元配列 C の整合寸法.

XT …… **入力**．*x* 方向，*y* 方向の節点列 (BICD3 の出力).
大きさ$(n_x-m+1)+(n_y-m+1)$ の 1 次元配列.

ISWX …… **入力**．*x* 方向に関する計算の種類を与える．
$-1 \le ISWX \le m$ （パラメタ F の項参照）

VX …… **入力**．点 $P(v_x,v_y)$ の *x* 座標.

IX …… **入力**． $x_i \le v_x < x_{i+1}$ を満たす *i*.
$v_x = x_{n_x}$ のときは $IX=n_x-1$ とする．
出力． $x_i \le v_x < x_{i+1}$ を満たす *i*.
（使用上の注意②参照）

ISWY …… **入力**．*y* 方向に関する計算の種類を与える．
$-1 \le ISWY \le m$（パラメタ F の項参照）

VY …… **入力**．点 $P(v_x,v_y)$の *y* 座標.

IY …… **入力**．$y_j \le v_y < y_{j+1}$ を満たす *j*.
$v_y = y_{n_y}$ のときは $IY=n_y-1$ とする．
出力．$y_j \le v_y < y_{j+1}$ を満たす *j*.
（使用上の注意②参照）．

F …… **出力**．補間値あるいは偏微分値あるいは積分値.
ISWX=λ, ISWY=μとすると，λ, μの組合せごとに次の値を出力する．

- $0 \le \lambda, \mu$のとき

$$F=\frac{\partial^{\lambda+\mu}}{\partial x^{\lambda}\partial y^{\mu}}S(v_x,v_y)$$

補間値はλ =μ= 0とすることにより得られる．

- λ = –1, $0 \le \mu$のとき

$$F=\int_{x_1}^{v_x}\frac{\partial^{\mu}}{\partial y^{\mu}}S(x,v_y)dx$$

- λ ≥ 0, μ = –1のとき

$$F=\int_{y_1}^{v_y}\frac{\partial^{\lambda}}{\partial x^{\lambda}}S(v_x,y)dy$$

- λ =μ = –1のとき

$$F=\int_{y_1}^{v_y}dy\int_{x_1}^{v_x}S(x,y)dx$$

VW …… **作業領域**.
大きさ $4(m+1) + \max(n_x,n_y)$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BIFD3-1 参照.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UCAD1, UBAS1

② FORTRAN 基本関数 ... FLOAT

表 BIFD3-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	X(IX)≤VX<X(IX+1) 又は Y(IY)≤VY<Y(IY+1) が満たされていない.	サブルーチン内で左記の IX 又は IY をさがして処理を続ける.
30000	次のいずれかであった. ① VX<X(1) 又は VX>X(NX) ② VY<Y(1) 又は VY>Y(NY) ③ ISWX<-1 又は ISWX>M ④ ISWY<-1 又は ISWY>M	処理を打ち切る.

b. 注意

① 本サブルーチンは，サブルーチン BICD3 により求められた B-spline2 次元補間式 (1.1) に基づいて，補間値あるいは偏微分値あるいは二重積分値を求めるものである.
したがって，本サブルーチンを呼び出す前に，BICD3 を呼び出して補間式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより補間値などを求めることができる．そのとき,パラメタ X, NX, Y, NY, K, M, C, XT の値は BICD3 のそれらと同一でなければならない.

② パラメタ IX, IY はそれぞれ X(IX)≤VX<X(IX+1), Y(IY)≤VY<Y(IY+1)を満足していることが好ましい．もし満足されていないときは，これらを満足するような IX, IY を探索して処理を続ける.

c. 使用例

格子点 (x_i,y_j)，関数値 f_{ij}, i=1,2,...,n_x, j=1, 2, ..., n_y, 及び次数 m を入力し，点 (v_{ir},u_{js}) ; $v_{ir}=x_i+(x_{i+1}-x_i)\times(r/4)$, $u_{js}=y_j+(y_{j+1}-y_j)\times(s/4)$, i=1,2, ..., n_x-1, j=1,2, ..., n_y-1, r=0,1, ...,4, s=0,1, ...,4 における補間値，偏微分値，あるいは領域 $[x_1 \le x \le v_{ir}, y_1 \le y \le u_{js}]$ 上の積分などを求める．n_x≤30, n_y≤30, m≤5 の場合.

```
C     **EXAMPLE**
      DIMENSION X(30),Y(30),FXY(30,30),
     *C(30,30),XT(52),VW(182),R(5,5)
      READ(5,500) NX,NY,M
      READ(5,510) (X(I),I=1,NX),
     *            (Y(J),J=1,NY)
      READ(5,510) ((FXY(I,J),J=1,NY),
     *            I=1,NX)
      WRITE(6,600) M,(I,X(I),I=1,NX)
      WRITE(6,610) (J,Y(J),J=1,NY)
      WRITE(6,620)
      DO 10 I=1,NX
   10 WRITE(6,630) (I,J,FXY(I,J),J=1,NY)
      K=30
      CALL BICD3(X,NX,Y,NY,FXY,K,M,C,
     *XT,VW,ICON)
      IF(ICON.EQ.0) GO TO 20
      WRITE(6,640)
      STOP
   20 NX1=NX-1
      NY1=NY-1
      M2=M+2
      DO 80 LX2=1,M2
      ISWX=LX2-2
      DO 70 LY2=1,M2
      ISWY=LY2-2
      WRITE(6,650) ISWX,ISWY
      DO 60 IX=1,NX1
      HX=(X(IX+1)-X(IX))*0.25
      XI=X(IX)
      DO 50 IY=1,NY1
      HY=(Y(IY+1)-Y(IY))*0.25
      YJ=Y(IY)
      DO 40 I=1,5
      VX=XI+HX*FLOAT(I-1)
      DO 30 J=1,5
      VY=YJ+HY*FLOAT(J-1)
      IXX=IX
      IYY=IY
      CALL BIFD3(X,NX,Y,NY,M,C,K,XT,
     *ISWX,VX,IXX,ISWY,VY,IYY,F,VW,ICON)
      R(I,J)=F
   30 CONTINUE
   40 CONTINUE
      WRITE(6,660)IXX,IYY,((R(I,J),J=1,5),
     *I=1,5)
   50 CONTINUE
   60 CONTINUE
   70 CONTINUE
   80 CONTINUE
      STOP
  500 FORMAT(3I6)
  510 FORMAT(2F12.0)
  600 FORMAT('1'//10X,'INPUT DATA',3X,
     *'M=',I2//20X,'NO.',10X,'X'/
     *(20X,I3,E18.7))
  610 FORMAT(//20X,'NO.',10X,'Y'/
     *(20X,I3,E18.7))
  620 FORMAT(//20X,'FXY'/)
  630 FORMAT(3(10X,'FXY(',I2,',',I2,')=',
     *E15.7))
  640 FORMAT ('0',10X,'ERROR')
  650 FORMAT(//5X,'ISWX=',I2,3X,'ISWY=',
     *I2//)
  660 FORMAT(10X,'IX=',I3,2X,'IY=',I3/
     *(15X,5(5X,E15.7)))
      END
```

(4) 手法概要

サブルーチン BICD3 により双 m 次の B-spline2 次元補間式,

$$S(x,y)=\sum_{\beta=-m+1}^{n_y-m}\sum_{\alpha=-m+1}^{n_x-m}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \quad (4.1)$$

が求まっているとする.

本サブルーチンでは，(4.1) の補間式に基づいて，補間値，偏微分あるいは積分値などを計算する．その方法は 7 章 7.1 の「(5) 2 変数 Spline 関数の定義, 表現及び計算法」で述べたのでそこを参照されたい.

E11-31-0101 BIF1, DBIF1

B-spline 補間式 (I) による補間, 数値微分, 数値積分
CALL BIF1(X,N,M,C,ISW,V,I,F,VW,ICON)

(1) 機能

離散点 $x_1,x_2...,x_n(x_1<x_2<...<x_n)$ において関数値 $y_i=f(x_i)$, i=1,2,...,n が与えられ，更に両端 x_1, x_n において微係数，$y_1^{(l)} = f^{(l)}(x_1)$, $y_n^{(l)} = f^{(l)}(x_n)$, l=1, 2, ..., $(m-1)/2$ が与えられたとき $x=v\in[x_1, x_n]$ における補間値あるいは微分値あるいは x_1 から v までの積分値を求める．

ただし，本サブルーチンを利用する前に，サブルーチン BIC1 により B-spline 補間式

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

における補間係数 c_j, $j=-m+1, -m+2, ..., n-1$ が計算されているとする．ここで m は奇数で B-spline $N_{j,m+1}(x)$ の次数である．

$x_1\le v\le x_n$, $m\ge 3$, $n\ge 2$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.

N …… **入力**．離散点の個数 n.

M …… **入力**． B-spline の次数 m.
（使用上の注意①参照）.

C …… **入力**．補間係数 c_j（BIC1 の出力）.
大きさ $n+m-1$ の 1 次元配列.

ISW …… **入力**．計算の種類を与える.
ISW = 0 のとき補間値 F=$S(v)$,
ISW = l $(1\le l\le m)$ のとき l 階微分値 F=$S^{(l)}(v)$,
ISW = –1 のとき積分値 F= $\int_{x_1}^{v} S(x)dx$ を求める.

V …… **入力**．補間値などを求めたい点 v.

I …… **入力**．$x_i\le v<x_{i+1}$ を満たす i.
$v=x_n$ のときは I=$n-1$ とする.
出力．$x_i\le v<x_{i+1}$ を満たす i.
（使用上の注意②参照）.

F …… **出力**．補間値あるいは l 階微分値あるいは積分値（ISW の項参照）.

VW …… **作業領域**．大きさ $m+1$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BIF1-1 参照.

表 BIF1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	X(I)≤V<X(I+1) が満たされていない.	サブルーチン内で左記の I をさがして処理を続ける.
30000	次のいずれかであった. ① V<X(1) 又は V>X(N) ② ISW<-1 又は ISW>M	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UCAR1, UBAS1
② FORTRAN基本関数 ... FLOAT

b. 注意
① 本サブルーチンは，サブルーチン BIC1 により求められた B-spline 補間式 (1.1) に基づいて補間値あるいは微分値あるいは積分値を求めるものである.
したがって，本サブルーチンを呼び出す前に，BIC1 を呼び出して補間式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより補間値を求めることができる．そのとき，パラメタ X, N, M, C は BCI1 のそれらと同一でなければならない.
② パラメタ I は X(I)≤V<X(I+1) を満足していることが好ましい．もし満足されていないときは X(I)≤V<X(I+1) を満足するような I を探索して処理を続ける.

c. 使用例
離散点 x_i，関数値 y_i, i=1,2,..., n, 両端における微係数 $y_1^{(l)}$, l=1, 2, ..., $(m-1)/2$, $y_n^{(l)}$, l=1, 2, ..., $(m-1)/2$ 及び次数 m を入力し，$v_{ij}=x_i+(x_{i+1}-x_i)\times(j/5)$, i=1, 2, ..., $n-1$, j=0, 1, ..., 5 における，x_1 からの積分値，補間値，1 階から m 階までの微分値を求める．$n\le 101$, $m\le 5$ の場合.

```
C     **EXAMPLE**
      DIMENSION X(101),Y(101),C(105),
     *DY(2,2),VW(519),R(6)
      READ(5,500) N,M
      LM1=(M-1)/2
      READ(5,510) (X(I),Y(I),I=1,N)
     *,((DY(I,L),I=1,2),L=1,LM1)
      WRITE(6,600) N,M,(I,X(I),Y(I),I=1,N)
      CALL BIC1(X,Y,DY,N,M,C,VW,ICON)
      IF(ICON.EQ.0) GO TO 10
      WRITE(6,610)
      STOP
```

```
 10 N1=N-1
    M2=M+2
    DO 40 L2=1,M2
    ISW=L2-2
    WRITE(6,620) ISW
    DO 30 I=1,N1
    H=(X(I+1)-X(I))/5.0
    XI=X(I)
    DO 20 J=1,6
    V=XI+H*FLOAT(J-I)
    II=I
    CALL BIF1(X,N,M,C,ISW,V,II,F,VW,ICON)
    R(J)=F
 20 CONTINUE
    WRITE(6,630) II,(R(J),J=1,6)
 30 CONTINUE
 40 CONTINUE
    STOP
500 FORMAT(2I6)
510 FORMAT(2F12.0)
600 FORMAT('1'//10X,'INPUT DATA',3X,
   *'N=',I3,3X,'N=',I2//20X,'NO.',10X,
   *'X',17X,'Y'//(20X,I3,2E18.7))
610 FORMAT('0',10X,'ERROR')
620 FORMAT('1'//10X,'L=',I2/)
630 FORMAT(6X,I3,6E18.7)
    END
```

(4) 手法概要

サブルーチン BIC1 により m 次の B-spline 補間式,

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \tag{4.1}$$

が求まっているとする.

本サブルーチンでは，(4.1) の補間式に基づいて，補間値，l 階微分値あるいは積分値をそれぞれ (4.2), (4.3), (4.4) より求める.

$$S(v)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(v) \tag{4.2}$$

$$S^{(l)}(v)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}^{(l)}(v) \tag{4.3}$$

$$I=\int_{x_1}^{v} S(x)dx \tag{4.4}$$

これら三つの量の計算法については 7 章 7.1 の「(4) Spline 関数の計算法」で一般的に述べたのでそこを参照されたい.

本サブルーチンでは，$N_{j,m+1}(x)$ 及びその微分，積分の計算は，スレーブサブルーチン UBAS1 で行っている.

E11-31-0201 BIF2, DBIF2

B-spline 補間式 (II) による補間, 数値微分, 数値積分
CALL BIF2(X,N,M,C,ISW,V,I,F,VW,ICON)

(1) 機能

離散点 $x_1,x_2...,x_n(x_1<x_2<...<x_n)$ において関数値 $y_i=f(x_i)$, i=1,2,...,n が与えられ, 更に両端 x_1, x_n において微係数 $y_1^{(l)} = f^{(l)}(x_1)$, $y_n^{(l)} = f^{(l)}(x_n)$, l= (m + 1)/2, (m + 1)/2+1, …,m – 1 が与えられたとき x=v∈[x_1, x_n] における補間値あるいは微分値あるいは x_1 から v までの積分値を求める.

ただし, 本サブルーチンを利用する前に, サブルーチン BIC2 により B-spline 補間式

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

における補間係数 c_j, j=–m+1, –m+2, ..., n – 1 が計算されているとする. ここで m は奇数で B-spline $N_{j,m+1}(x)$ の次数である.

x_1≤v≤x_n, m≥3, n≥(m+1)/2 であること.

(2) パラメタ

X…… **入力**. 離散点 x_i.
大きさ n の 1 次元配列.

N…… **入力**. 離散点の個数 n.

M…… **入力**. B-spline の次数 m.
（使用上の注意①参照）.

C…… **入力**. 補間係数 c_i（BIC2 の出力）.
大きさ n+m–1 の 1 次元配列.

ISW …… **入力**. 計算の種類を与える.
ISW=0 のとき補間値 F=$S(v)$,
ISW=l ($1 \le l \le m$) のとき
l 階微分値 F=$S^{(l)}(v)$.
ISW=–1 のとき積分値 F=$\int_{x_1}^{v} S(x)dx$ を求める.

V…… **入力**. 補間値などを求めたい点 v.

I …… **入力**. x_i≤v<x_{i+1} を満たす i.
v=x_n のときは I=n–1 とする.
出力. x_i≤v<x_{i+1} を満たす i.
（使用上の注意②参照）.

F …… **出力**. 補間値あるいは l 階微分値あるいは積分値（ISW の項参照）.

VW…… **作業領域**. 大きさ m+1 の 1 次元配列.

ICON…… **出力**. コンディションコード.
表 BIF2-1 参照.

表 BIF2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	X(I)≤V<X(I+1) が満たされていない.	サブルーチン内で左記の I をさがして処理を続ける.
30000	次のいずれかであった. ① V<X(1) 又は V>X(N) ② ISW<-1 又は ISW>M	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UCAR1, UBAS1
② FORTRAN 基本関数 ... FLOAT

b. 注意
① 本サブルーチンは, サブルーチン BIC2 により求められた B-spline 補間式 (1.1) に基づいて補間値あるいは微分値あるいは積分値を求めるものである.
したがって, 本サブルーチンを呼び出す前に, BIC2 を呼び出して補間式 (1.1) を求め, 続けて本サブルーチンを呼び出すことにより補間値などを求めることができる. その時, パラメタ X, N, M, C は BIC2 のそれらと同一でなければならない.
② パラメタ I は X(I)≤V<X(I+1) を満足していることが好ましい. もし満足されていないときは X(I)≤V<X(I+1) を満足するような I を探索して処理を続ける.

c. 使用例
離散点 x_i, 関数値 y_i, i=1,2,..., n, 両端における微係数 $y_1^{(l)}$, $y_n^{(l)}$, l= (m + 1)/2, (m + 1)/2+1, ..., m – 1 及び次数 m を入力し, v_{ij}=x_i+(x_{i+1}-x_i)×(j/5), i=1, 2, ..., n – 1, j=0, 1, ..., 5 における, x_1 からの積分値, 補間値, 1 階から m 階までの微分値を求める. n≤101, m≤5 の場合.

```
C     **EXAMPLE**
      DIMENSION X(101),Y(101),C(105),
     *DY(2,2),VW(527),R(6)
      READ(5,500) N,M
      LM1=(M-1)/2
      READ(5,510) (X(I),Y(I),I=1,N)
     * ,((DY(I,L),I=1,2),L=1,LM1)
      WRITE(6,600) N,M,(I,X(I),Y(I),I=1,N)
      CALL BIC2(X,Y,DY,N,M,C,VW,ICON)
      IF(ICON.EQ.0) GO TO 10
      WRITE(6,610)
      STOP
```

```
 10 N1=N-1
    M2=M+2
    DO 40 L2=1,M2
    ISW=L2-2
    WRITE(6,620) ISW
    DO 30 I=1,N1
    H=(X(I+1)-X(I))/5.0
    XI=X(I)
    DO 20 J=1,6
    V=XI+H*FLOAT(J-I)
    II=I
    CALL BIF2(X,N,M,C,ISW,V,II,F,VW,ICON)
    R(J)=F
 20 CONTINUE
    WRITE(6,630) II,(R(J),J=1,6)
 30 CONTINUE
 40 CONTINUE
    STOP
500 FORMAT(2I6)
510 FORMAT(2F12.0)
600 FORMAT('1'//10X,'INPUT DATA',3X,
   *'N=',I3,3X,'N=',I2//20X,'NO.',10X,
   *'X',17X,'Y'//(20X,I3,2E18.7))
610 FORMAT('0',10X,'ERROR')
620 FORMAT('1'//10X,'L=',I2/)
630 FORMAT(6X,I3,6E18.7)
    END
```

(4) 手法概要

サブルーチン BIC2 により m 次の B-spline 補間式

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \tag{4.1}$$

が求まっているとする.

本サブルーチンでは，(4.1) の補間式に基づいて，補間値，l 階微分値あるいは積分値をそれぞれ (4.2), (4.3), (4.4) より求める.

$$S(v)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(v) \tag{4.2}$$

$$S^{(l)}(v)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}^{(l)}(v) \tag{4.3}$$

$$I=\int_{x_1}^{v} S(x)dx \tag{4.4}$$

これら三つの量の計算法については7章7.1の「(4)Spline関数の計算法」で一般的に述べたのでそこを参照されたい.

本サブルーチンでは，$N_{j,m+1}(x)$ 及びその微分，積分の計算は，スレーブサブルーチン UBAS1 で行っている.

E11-31-0301 BIF3, DBIF3

B-spline 補間式 (III) による補間，数値微分，数値積分
CALL BIF3(X,N,M,C,XT,ISW,V,I,F,VW,ICON)

(1) 機能

離散点 $x_1,x_2,...,x_n(x_1<x_2<...<x_n)$ に対して関数値 $y_i=f(x_i)$, i=1,2,...,n が与えられたとき，$x=v$ における補間値あるいは微分値あるいは x_1 から v までの積分値を求める．

ただし，本サブルーチンを利用する前に，サブルーチン BIC3 により節点列ξ_i, i=1, 2, ..., n–m+1 及び B-spline 補間式，

$$S(x)=\sum_{j=-m+1}^{n-m} c_j N_{j,m+1}(x) \qquad (1.1)$$

における補間係数 c_j, j=–m+1, –m+2, ..., n – m が計算されているとする．ここで m は奇数で B-spline $N_{j,m+1}(x)$ の次数である．

$x_1\le v\le x_n$, $m\ge 3$ かつ $n\ge m+2$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.

N …… **入力**．離散点の個数 n.

M …… **入力**． B-spline の次数 m.
（使用上の注意①参照）.

C …… **入力**．補間係数 c_j（BIC3 の出力）.
大きさ n の 1 次元配列.

XT …… **入力**．節点列ξ_i,（BIC3 の出力）.
大きさ n–m+1 の 1 次元配列.

ISW …… **入力**．計算の種類を与える.
ISW=0 のとき補間値 F=$S(v)$,
ISW=l (l=1, 2, …, m) のとき,
l 階微分値 F=$S^{(l)}(v)$,
ISW=–1 のとき積分値 F=$\int_{x_1}^{v} S(x)dx$ を求める.

V …… **入力**．補間値などを求めたい点 v.

I …… **入力**．$x_i\le v<x_{i+1}$ を満たす i.
$v=x_n$ のときは I=n–1 とすること.
出力．$x_i\le v<x_{i+1}$ を満たす i.
（使用上の注意②参照）.

F …… **出力**．補間値あるいは l 階微分値あるいは積分値（ISW の項参照）.

VW …… **作業領域**．大きさ 2m+2 の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BIF3-1 参照.

表 BIF3-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	X(I)≤V<X(I+1) が満たされていない.	サブルーチン内で左記の I をさがして処理を続ける.
30000	次のいずれかであった. ① V<X(1) 又は V>X(N) ② ISW<-1 又は ISW>M	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UCAR1, UBAS1
② FORTRAN 基本関数 ... FLOAT

b. 注意
① 本サブルーチンは，サブルーチン BIC3 により，求められた B-spline 補間式 (1.1) に基づいて，補間値あるいは微分値あるいは積分値を求めるものである.
したがって，本サブルーチンを呼び出す前に,サブルーチン BIC3 を呼び出して補間式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより補間値などを求めることができる．そのとき，パラメタ X, N, M, C, XT は BIC3 のそれらと同一でなければならない.
② パラメタ I は X(I)≤V<X(I+1) を満足していることが好ましい．もし満足されていないときは，X(I)≤V<X(I+1) を満足するような I を探索して処理を続ける.

c. 使用例
離散点 x_i，関数値 y_i, i=1,2,..., n 及び次数 m を入力し，$v_{ij}=x_i+(x_{i+1}-x_i)\times(j/5)$, i=1, 2, ..., n – 1, j=0, 1, ..., 5 における，x_1 からの積分値，補間値，1 階から m 階までの微分値を求める．$n\le 101$, $m\le 5$ の場合.

```
C     **EXAMPLE**
      DIMENSION X(101),Y(101),C(101),
     *XT(99),VW(507),R(6)
      READ(5,500) N,M
      READ(5,510) (X(I),Y(I),I=1,N)
      WRITE(6,600) N,M,(I,X(I),Y(I),I=1,N)
      CALL BIC3(X,Y,N,M,C,XT,VW,ICON)
      IF(ICON.EQ.0) GO TO 10
      WRITE(6,610)
      STOP
```

```
 10 N1=N-1
    M2=M+2
    DO 40 L2=1,M2
    ISW=L2-2
    WRITE(6,620) ISW
    DO 30 I=1,N1
    H=(X(I+1)-X(I))/5.0
    XI=X(I)
    DO 20 J=1,6
    V=XI+H*FLOAT(J-1)
    II=I
    CALL BIF3(X,N,M,C,XT,ISW,V,II,F,
   *          VW,ICON)
    R(J)=F
 20 CONTINUE
    WRITE(6,630) II,(R(J),J=1,6)
 30 CONTINUE
 40 CONTINUE
    STOP
500 FORMAT(2I6)
510 FORMAT(2F12.0)
600 FORMAT('1'//10X,'INPUT DATA',3X,
   *'N=',I3,3X,'N=', I2//20X, 'NO.',10X,
   *'X',17X,'Y'//(20X,I3,2E18.7))
610 FORMAT('0',10X,'ERROR')
620 FORMAT('1'//10X,'L=',I2/)
630 FORMAT(6X,I3,6E18.7)
    END
```

(4) 手法概要

サブルーチン BIC3 により, m 次の B-spline 補間式,

$$S(x)=\sum_{j=-m+1}^{n-m} c_j N_{j,m+1}(x) \tag{4.1}$$

が求まっているとする. 本サブルーチンでは, (4.1) の補間式に基づいて，補間値，l 階微分値あるいは積分値をそれぞれ (4.2), (4.3), (4.4) より求める.

$$S(v)=\sum_{j=-m+1}^{n-m} c_j N_{j,m+1}(v) \tag{4.2}$$

$$S^{(l)}(v)=\sum_{j=-m+1}^{n-m} c_j N^{(l)}_{j,m+1}(v) \tag{4.3}$$

$$I=\int_{x_1}^{v} S(x)dx \tag{4.4}$$

これら三つの量の計算法については 7 章 7.1 の「(4)Spline 関数の計算法」で一般的に述べたのでそこを参照されたい.

本サブルーチンでは，$N_{j,m+1}(x)$ 及びその微分，積分の計算はスレーブサブルーチン UBAS1 で行っている.

E11-31-0401 BIF4, DBIF4

B-spline 補間式 (IV) による補間，数値微分，数値積分
CALL BIF4(X,N,M,C,ISW,V,I,F,VW,ICON)

(1) 機能

離散点 $x_1,x_2,...,x_n(x_1<x_2<...<x_n)$ に対して，周期 (x_n-x_1) の周期関数値 $y_i=f(x_i)$, i=1,2,...,n（ただし $y_1=y_n$）が与えられたとき，$x=v\in[x_1, x_n]$ における補間値あるいは微分値あるいは x_1 から v までの積分値を求める．

ただし，本サブルーチンを利用する前に，サブルーチン BIC4 により，周期条件を満たす B-spline 補間式

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

における補間係数 c_j, $j=-m+1,-m+2,...,n-1$ が計算されているとする．ここで m は奇数で B-spline $N_{j,m+1}(x)$ の次数である．

$x_1\le v\le x_n$, $m\ge 3$, $n\ge m+2$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.

N …… **入力**．離散点の個数 n.

M …… **入力**． B-spline の次数 m.
（使用上の注意①参照）．

C …… **入力**．補間係数 c_j（BIC4 の出力）．
大きさ $n+m-1$ の 1 次元配列.

ISW …… **入力**．計算の種類を与える．
ISW=0 のとき補間値 F=$S(v)$,
ISW=l $(1\le l\le m)$ のとき
l 階微分値 F=$S^{(l)}(v)$,
ISW=-1 のとき積分値 F=$\int_{x_1}^{v} S(x)dx$ を求める．

V …… **入力**．補間値などを求めたい点 v.

I …… **入力**．$x_i\le v<x_{i+1}$ を満たす i.
$v=x_n$ のときは I=$n-1$ とする．
出力．$x_i\le v<x_{i+1}$ を満たす i.
（使用上の注意②参照）．

F …… **出力**．補間値あるいは l 階微分値あるいは積分値（ISW の項参照）．

VW …… **作業領域**．大きさ $m+1$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BIF4-1 参照.

表 BIF4-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	X(I)≤V<X(I+1) が満たされていない．	サブルーチン内で左記の I をさがして処理を続ける．
30000	次のいずれかであった． ① V<X(1) 又は V>X(N) ② ISW<-1 又は ISW>M	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UCAR4, UBAS4, UPEP4

② FORTRAN 基本関数 ... FLOAT

b. 注意

① 本サブルーチンは，サブルーチン BIC4 により求められた B-spline 補間式 (1.1) に基づいて，補間値あるいは微分値あるいは積分値を求めるものである．
したがって，本サブルーチンを呼び出す前に,サブルーチン BIC4 を呼び出して補間式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより補間値などを求めることができる．そのとき，パラメタ X, N, M, C は BIC4 のそれらと同一でなければならない．

② パラメタ I は X(I)≤V<X(I+1) を満足していることが好ましい．もし満足されていないときは，X(I)≤V<X(I+1) を満足するような I を探索して処理を続ける．

c. 使用例
離散点 x_i，関数値 y_i, i=1,2,..., n（周期 (x_n-x_1)）及び次数 m を入力し，$v_{ij}=x_i+(x_{i+1}-x_i)\times(j/5)$, i=1, 2, ..., $n-1$, j=0, 1, ..., 5 における，x_1 からの積分値，補間値，1 階から m 階までの微分値を求める．
$n\le 101$, $m\le 5$ の場合．

```
C     **EXAMPLE**
      DIMENSION X(101),Y(101),C(105),
     *VW(906),R(6)
      READ(5,500) N,M
      READ(5,510) (X(I),Y(I),I=1,N)
      WRITE(6,600) N,M,(I,X(I),Y(I),I=1,N)
      CALL BIC4(X,Y,N,M,C,VW,ICON)
      IF(ICON.EQ.0) GO TO 10
      WRITE(6,610)
      STOP
```

```
 10 N1=N-1
    M2=M+2
    DO 40 L2=1,M2
    ISW=L2-2
    WRITE(6,620) ISW
    DO 30 I=1,N1
    H=(X(I+1)-X(I))/5.0
    XI=X(I)
    DO 20 J=1,6
    V=XI+H*FLOAT(J-1)
    II=I
    CALL BIF4(X,N,M,C,ISW,V,II,F,
   *VW,ICON)
    R(J)=F
 20 CONTINUE
    WRITE(6,630) II,(R(J),J=1,6)
 30 CONTINUE
 40 CONTINUE
    STOP
500 FORMAT(2I6)
510 FORMAT(2F12.0)
600 FORMAT('1'//10X,'INPUT DATA',3X,
   *'N=',I3,3X,'N=',I2//20X,'NO.',10X,
   *'X',17X,'Y'//(20X,I3,2E18.7))
610 FORMAT('0',10X,'ERROR')
620 FORMAT('1'//10X,'L=',I2/)
630 FORMAT(6X,I3,6E18.7)
    END
```

(4) 手法概要

サブルーチン BIC4 により m 次の B-spline 補間式,

$$S(x)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(x) \tag{4.1}$$

が求まっているとする．ただし，ここでの $S(x)$ は,

$$S^{(l)}(x_1)=S^{(l)}(x_n),\quad l=0,1,\ldots,m-1 \tag{4.2}$$

を満たす，周期 (x_n-x_1) の周期関数である．

本サブルーチンでは，(4.1) の補間式に基づいて，補間値，l 階微分値あるいは積分値をそれぞれ (4.3), (4.4), (4.5) より求める．

$$S(v)=\sum_{j=-m+1}^{n-1} c_j N_{j,m+1}(v) \tag{4.3}$$

$$S^{(l)}(v)=\sum_{j=-m+1}^{n-1} c_j N^{(l)}_{j,m+1}(v) \tag{4.4}$$

$$\mathrm{I}=\int_{x_1}^{v} S(x)dx \tag{4.5}$$

これら三つの量の計算法については 7 章 7.1 の「(4)Spline 関数の計算法」で一般的に述べたのでそこを参照されたい．

本サブルーチンでは，$N_{j,m+1}(x)$，及びその微分，積分の計算はスレーブサブルーチン UBAS4 で行っている．

I11-81-1201 BIN, DBIN

第 1 種整数次変形ベッセル関数 $I_n(x)$
CALL BIN(X,N,BI,ICON)

(1) 機能

第 1 種整数次変形ベッセル関数 $I_n(x)$ の値

$$I_n(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{2k+n}}{k!(n+k)!}$$

をテイラー展開式及び漸化式を用いて計算する.

(2) パラメタ

X…… **入力**. 独立変数 *x*.
N…… **入力**. $I_n(x)$ の次数 *n*.
BI…… **出力**. 関数値 $I_n(x)$.
ICON…… **出力**. コンディションコード.
表 BIN–1 参照.
ただし N=0 又は N=1 のときはそれぞれ BI0, BI1 の ICON に準ずる.

表 BIN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X及びNの値が下記のいずれかであった. · |X| > 100 · 1/8 ≤ |X| < 1でかつ |N| ≥ 19 |X|+29 · 1 ≤ |X| < 10でかつ |N| ≥ 4.7 |X| + 43 · 10 ≤ |X| ≤ 100でかつ |N| ≥ 1.83 |X| + 71	BI = 0.0とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMAX, AFMIN AMACH, BI0, BI1, MGSSL, ULMAX
② FORTRAN 基本関数 ... ABS, IABS, FLOAT, EXP, MAX0, AND SQRT.

b. 注意
① |***X***|, |***N***| の範囲は, 図 BIN-1 の白地の部分とする. 計算途中のオーバフロー, アンダフローを前もって防ぐためである.（手法概要 (4.2) 参照）
② $I_0(x)$ 及び $I_1(x)$ を計算するには, 本サブルーチンよりもそれぞれ BI0, BI1 を直接用いた方がよい.

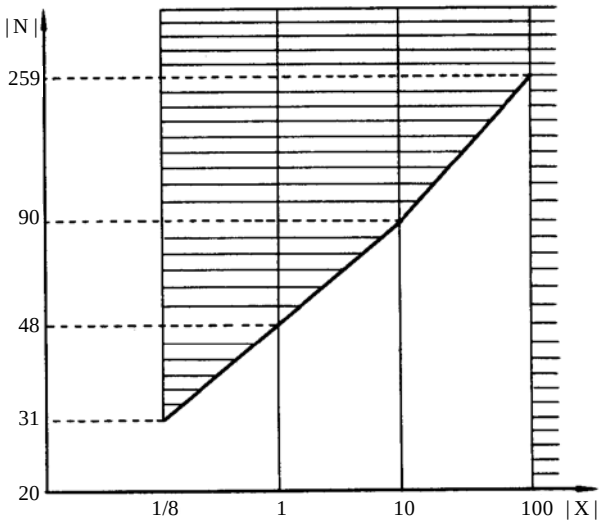


図 BIN-1 引数の計算範囲

c. 使用例
$I_n(x)$ の値を *x* が 0 から 10, N が 20 から 30 まで 1 おきに計算し数表を作る.

```
C      **EXAMPLE**
       WRITE(6,600)
       DO 20 N=20,30
       DO 10 K=1,11
       X=K-1
       CALL BIN(X,N,BI,ICON)
       IF(ICON.EQ.0) WRITE(6,610) X,N,BI
       IF(ICON.NE.0) WRITE(6,620) X,N,BI,
      *                           ICON
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',5X,'N',8X,
     *'IN(X)'/)
  610 FORMAT(' ',F8.2,I5,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'N=',I5,5X,'BI=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4) 手法概要

ベッセル関数 $I_n(x)$ の計算は, $|x|=1/8$ を境にして, 計算式の形が異なる. $I_{-n}(x)=I_n(x)$, $I_n(-x)=(-1)^nI_n(x)$ であるから, 以下では$|n|$, $|x|$をそれぞれ *n*, *x* として説明する.

a. $0\le x<1/8$ の場合

$$I_n(x)=\frac{x_n}{2^n n!}\left\{1+\frac{x^2}{2(2n+2)}+\frac{x^4}{2.4(2n+2)(2n+4)}+\ldots\right\} \quad (4.1)$$

により第 n 項目が初項に比べ丸め誤差の単位以下になるまで計算する.

b. $1/8 \le x \le 100$ の場合
n, x より十分大きな値 $\boldsymbol{M}$ を決め
$k=\boldsymbol{M}, \boldsymbol{M}-1,...,1$ に対して漸化式

$$F_{k-1}(x)=\frac{2k}{x}F_k(x)+F_{k+1}(x) \tag{4.2}$$

を反復適用する.
ただし $F_{M+1}(x)=0$, $F_M(x)=fl_{min}$
とする. このとき, $I_n(x)$ は,

$$I_n(x)=e^x F_n(x)/\left\{F_0(x)+2\sum_{i=1}^{M}F_i(x)\right\} \tag{4.3}$$

で与えられる.

[$\boldsymbol{M}$の決め方]

$$M=2\left[\left\{m_0+1+\max(n-n_0,0)\right\}/2\right] \tag{4.4}$$

ただし, [ ] はガウス記号である.

(a) $1/8 \le x < 1$ のとき
単精度：$m_0=4.8x+6.1$ (4.5)
$n_0=3.6x+2.1$ (4.6)
倍精度：$m_0=9.4x+9.5$ (4.7)
$n_0=5.0x+4.5$ (4.8)

(b) $1 \le x < 10$ のとき
単精度：$m_0=1.4x+9.6$ (4.9)
$n_0=0.9x+5.1$ (4.10)
倍精度：$m_0=2.5x+15.5$ (4.11)
$n_0=1.4x+8.6$ (4.12)

(c) $10 \le x \le 100$ のとき
単精度：$m_0=0.75x+17.5$ (4.13)
$n_0=0.28x+11.2$ (4.14)
倍精度：$m_0=0.77x+32.3$ (4.15)
$n_0=0.45x+17.5$ (4.16)

なお, 詳細については, 参考文献 [81], [82] を参照のこと.

I11-83-0301 BIR, DBIR

第 1 種実数次変形ベッセル関数 $I_v(x)$
CALL BIR(X,V,BI,ICON)

(1) 機能

第 1 種実数次変形ベッセル関数 $I_v(x)$

$$I_v(x)=\left(\frac{1}{2}x\right)^v\sum_{k=0}^{\infty}\frac{\left(\frac{1}{4}x^2\right)^k}{k!\Gamma(v+k+1)}$$

の値をべき級数展開式 (上式) 及び漸化式による方法を用いて計算する.

(2) パラメタ

X…… **入力**. 独立変数 x(x≥0).
V…… **入力**. $I_v(x)$ の次数 v (v≥0).
BI…… **出力**. 関数値 $I_v(x)$.
ICON…… **出力**. コンディションコード.
表 BIR-1 参照.

表 BIR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X>log(fl_{max}) であった.	BI = 0.0 とする.
30000	X<0 又は V<0 であった.	BI = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMAX, AMACH, AFMIN, MGSSL, ULMAX
② FORTRAN 基本関数 ... FLOAT, ABS, GAMMA, AMAX1, EXP

b. 注意
① $0 \le X \le \log(fl_{max})$ 及び $V \ge 0$ であること.
② $I_0(x)$ あるいは $I_1(x)$ を計算するときには, 本サブルーチンよりもそれぞれ BI0, BI1 を直接用いた方がよい.
③ $I_v(x)$, $I_{v+1}(x)$, $I_{v+2}(x)$, ..., $I_{v+M}(x)$ の値をいっせいに必要とするときには, 本サブルーチンにより $I_{v+M}(x)$ と $I_{v+M-1}(x)$ を求め, 漸化式を反復適用して $I_{v+M-2}(x)$, $I_{v+M-3}(x)$, ..., $I_v(x)$ と次数の低いものを求めていくとよい. 逆に, 本サブルーチンにより $I_v(x)$ と $I_{v+1}(x)$ を求め, 漸化式により $I_{v+2}(x)$, $I_{v+3}(x)$, ..., $I_{v+M}(x)$ と次数の高いものを求めていくことは不安定が生ずるので避けなければならない.
④ 倍精度のサブルーチン DBIR を使用するときは, X, V 及び BI を倍精度宣言することが必要である.

c. 使用例
$I_v(x)$ の値を, x を 0 から 10 まで 1 おきに, v を 0.4 から 0.6 まで 0.01 刻みで計算する.

```
C     **EXAMPLE**
      DO 20 K=1,11
      X=K-1
      DO 10 NV=40,60
      V=FLOAT(NV)/100.0
      CALL BIR(X,V,BI,ICON)
      IF(ICON.EQ.0) WRITE(6,600) X,V,BI
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT(' ',F8.2,F8.3,E17.7)
      END
```

(4) 手法概要

求められるべき $I_v(x)$ の値がアンダフローした値になることがあらかじめ分かった場合には, 以下の計算は行わず, 結果を 0.0 とする.

$I_v(x)$ の計算は, x により方法が異なる.

① $0 \le x \le 1$ の場合
べき級数展開

$$I_v(x)=\left(\frac{1}{2}x\right)^v\sum_{k=0}^{\infty}\frac{\left(\frac{1}{4}x^2\right)^k}{k!\Gamma(v+k+1)} \qquad (4.1)$$

により, 第k項目が初項に比べて丸め誤差の単位以下になるまで計算する.

② $1<x\le\log(fl_{max})$ の場合
漸化式による方法を用いて計算する.
m を適当に大きな整数 (x, v 及び要求精度によって決まる), δを十分に小さい定数 (使用する計算機の正の最小数) とする. n 及びαを次式のように決める.

$v=n+\alpha$(n : 整数$0\le\alpha<1$)

初期値を

$G_{\alpha+m+1}(x)=0$, $G_{\alpha+m}(x)=\delta$

として, $k=m,m-1,...,1$に対して漸化式

$$G_{\alpha+k-1}(x)=\frac{2(\alpha+k)}{x}G_{\alpha+k}(x)+G_{\alpha+k+1}(x) \qquad (4.2)$$

を反復適用する. このとき, $I_v(x)$ は

$$I_v(x)\approx\frac{1}{2}\left(\frac{x}{2}\right)^{\alpha}\frac{\Gamma(2\alpha+1)}{\Gamma(\alpha+1)}e^xG_{\alpha+n}(x)\Bigg/\left(\sum_{k=0}^{m}\frac{(\alpha+k)\Gamma(2\alpha+k)}{k!}G_{\alpha+k}(x)\right) \qquad (4.3)$$

として求められる.
なお，m の決め方及びその他詳細については,
参考文献 [81] を参照のこと.

I11-81-0601 BI0,DBI0

第 1 種 0 次変形ベッセル関数 $I_0(x)$
CALL BI0(X,BI,ICON)

(1) 機能

第 1 種 0 次変形ベッセル関数 $I_0(x)$ の値

$$I_0(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{2k}}{(k!)^2}$$

を多項式及び漸近形式の近似式を用いて計算する.

(2) パラメタ

X…… **入力**. 独立変数 x.
BI…… **出力**. 関数値 $I_0(x)$.
ICON …… **出力**. コンディションコード.
表 BI0-1 参照.

表 BI0-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$\|X\| > \log(fl_{max})$ であった.	BI = fl_{max}とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMAX, MGSSL, ULMAX
② FORTRAN 基本関数 ... ABS, EXP, SQRT

b. 注意
① 引数 X の範囲は
$|\boldsymbol{X}| \le \log(fl_{max})$
であること.
$|\boldsymbol{X}|$ が上記範囲を超えると e^x の計算でオーバフローを生じる. これを本サブルーチン内で前もって防ぐためである. （手法概要(4.4), (4.5) 参照）

c. 使用例
$I_0(x)$ の値を x が 0 から 100 まで, 1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=K-1
      CALL BI0(X,BI,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,BI
      IF(ICON.NE.0) WRITE(6,620) X,BI,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',9X,'I0(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'BI=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4) 手法概要

変形ベッセル関数 $I_0(x)$ の計算は, $|\boldsymbol{X}| = 8$ を境にして, 近似式の形が異なる. $I_0(-x)=I_0(x)$ であるから,以下では$|x|$ を x として説明する.

a. $0 \le x < 8$ の場合
$I_0(x)$ のべき級数展開

$$I_0(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{2k}}{(k!)^2} \tag{4.1}$$

をそれぞれ (4.2), (4.3) の近似式を用いて計算する.
単精度：

$$I_0(x)=\sum_{k=0}^{11}a_k x^{2k} \tag{4.2}$$

倍精度：

$$I_0(x)=\sum_{k=0}^{16}a_k x^{2k} \tag{4.3}$$

b. $8 \le x \le \log(fl_{max})$の場合
$I_0(x)$ の漸近展開

$$I_0(x)=\frac{e^x}{\sqrt{2\pi x}}\left\{1+\frac{1}{8x}+\frac{1^2\cdot 3^2}{2!}\cdot\frac{1}{(8x)^2}+\frac{1^2\cdot 3^2\cdot 5^2}{3!}\cdot\frac{1}{(8x)^3}+\ldots\right\} \tag{4.4}$$

をそれぞれ (4.5), (4.6) の近似式を用いて計算する.
単精度：

$$I_0(x)=\frac{e^x}{\sqrt{x}}\left(\sum_{k=0}^{5}a_k z^k\right),\quad z=8/x \tag{4.5}$$

倍精度：

$$I_0(x)=\frac{e^x}{\sqrt{x}}\left(\sum_{k=0}^{11}a_k z^k \Big/ \sum_{k=0}^{11}b_k z^k\right),\quad z=(x-8)/x \tag{4.6}$$

I11-81-0701 BI1,DBI1

第 1 種 1 次変形ベッセル関数 $\boldsymbol{I}_1(x)$
CALL BI1(X,BI,ICON)

(1) 機能

第 1 種 1 次変形ベッセル関数 $I_1(x)$ の値

$$I_1(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{2k+1}}{(k!)(k+1)!}$$

を多項式及び漸近形式の近似式を用いて計算する.

(2) パラメタ

X …… **入力**. 独立変数 x.
BI …… **出力**. 関数値 $I_1(x)$.
ICON …… **出力**. コンディションコード.
表 BI1-1 参照.

表 BI1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X>log(fl_{max}) であった. X<-log(fl_{max}) であった.	BI = fl_{max} とする. BI = $-fl_{max}$ とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMAX, MGSSL, ULMAX
② FORTRAN 基本関数 ... ABS, EXP, SQRT

b. 注意
引数 X の範囲は
$|\boldsymbol{X}| \le \log(fl_{max})$
であること.
$|\boldsymbol{X}|$ が上記範囲を超えると e^x の計算でオーバフローを生じる. これを本サブルーチン内で前もって防ぐためである.
（手法概要 (4.4), (4.5) 参照）

c. 使用例
$I_1(x)$ の値を x が 0 から 100 まで, 1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=K-1
      CALL BI1(X,BI,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,BI
      IF(ICON.NE.0) WRITE(6,620) X,BI,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'//6X,'X',9X,'I1(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'BI=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4) 手法概要

変形ベッセル関数 $I_1(x)$ の計算は, $|x| = 8$ を境にして, 近似式の形が異なる. $I_1(-x)=-I_1(x)$ であるから, 以下では$|x|$を x として説明する.

a. $0 \le x < 8$の場合
$I_1(x)$ のべき級数展開

$$I_1(x)=\sum_{k=0}^{\infty}\frac{(x/2)^{2k+1}}{(k!)(k+1)!} \tag{4.1}$$

をそれぞれ (4.2), (4.3) の近似式を用いて計算する.
単精度：

$$I_1(x)=\sum_{k=0}^{10}a_k x^{2k+1} \tag{4.2}$$

倍精度：

$$I_1(x)=\sum_{k=0}^{16}a_k x^{2k+1} \tag{4.3}$$

b. $8 \le x \le \log(fl_{max})$ の場合
$I_1(x)$ の漸近展開

$$I_1(x)=\frac{e^x}{\sqrt{2\pi x}}\left\{1-\frac{3}{8x}+\frac{3\cdot(-5)}{2!}\cdot\frac{1}{(8x)^2}-\frac{3\cdot(-5)\cdot(-21)}{3!}\cdot\frac{1}{(8x)^3}+\cdots\right\} \tag{4.4}$$

をそれぞれ (4.5), (4.6) の近似式を用いて計算する.
単精度：

$$I_1(x)=\frac{e^x}{\sqrt{x}}\left(\sum_{k=0}^{5}a_k z^k\right),\quad z=8/x \tag{4.5}$$

倍精度：

$$I_1(x)=\frac{e^x}{\sqrt{x}}\left(\sum_{k=0}^{11}a_k z^k \Big/ \sum_{k=0}^{11}b_k z^k\right),\quad z=(x-8)/x \tag{4.6}$$

I11-81-1001 BJN, DBJN

第 1 種整数次ベッセル関数 $J_n(x)$
CALL BJN(X,N,BJ,ICON)

(1) 機能

第 1 種整数次ベッセル関数 $J_n(x)$ の値

$$J_n(x)=\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k+n}}{(k!)(n+k)!}$$

をテーラー展開式及び漸化式を用いて計算する.

(2) パラメタ

X …… **入力**. 独立変数 x.
N …… **入力**. $J_n(x)$ の次数 n.
BJ …… **出力**. 関数値 $J_n(x)$.
ICON …… **出力**. コンディションコード.
表 BJN-1 参照.
ただし N=0 又は N=1 のときはそれぞれ BJ0, BJ1 の ICON に準ずる.

表 BJN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X 及び N の値が下記のいずれかであった. · |X| > 100 · 1/8 ≤ |X| < 1 でかつ |N| ≥ 19 |X| + 29 · 1 ≤ |X| < 10 でかつ |N| ≥ 4.7 |X| + 43 · 10 ≤ |X| ≤ 100 でかつ |N| ≥ 1.83 |X| + 71	BJ = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMIN, AMACH, BJ0, BJ1, MGSSL, UTLIM
② FORTRAN 基本関数 ... ABS, IABS, FLOAT, MAX0, DSIN, DCOS, DSQRT

b. 注意
① |***X***|, |***N***|の範囲は，図 BJN-1 の白地の部分とする. 計算途中のオーバフロー，アンダフローを前もって防ぐためである.（手法概要 (4.2) 参照）
② $J_0(x)$ 及び $J_1(x)$ を計算するには，本サブルーチンよりもそれぞれ，BJ0, BJ1 を直接用いた方がよい.

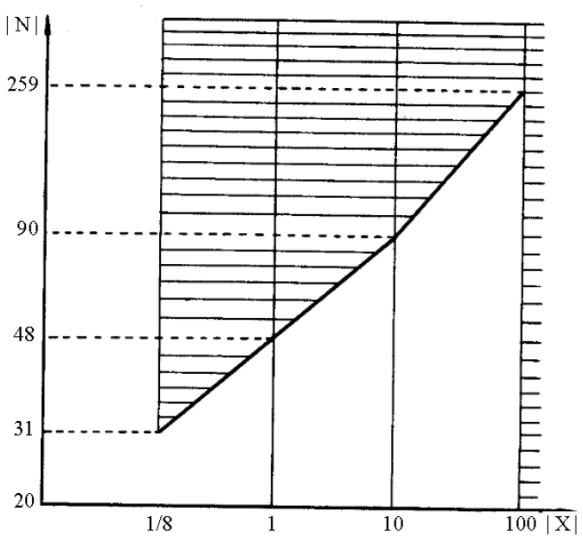


図 BJN-1 引数の計算範囲

c. 使用例
$J_n(x)$ の値を x が 0 から 10，N が 20 から 30 まで 1 おきに計算し数表を作る.

```
C       **EXAMPLE**
        WRITE(6,600)
        DO 20 N=20,30
        DO 10 K=1,11
        X=K-1
        CALL BJN(X,N,BJ,ICON)
        IF(ICON.EQ.0) WRITE(6,610) X,N,BJ
        IF(ICON.NE.0) WRITE(6,620) X,N,BJ,
       *                           ICON
   10 CONTINUE
   20 CONTINUE
        STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
       *'FUNCTION'///6X,'X',5X,'N',8X,
       *'JN(X)'/)
  610 FORMAT(' ',F8.2,I5,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
       *E17.7,5X,'N=',I5,5X,'BJ=',E17.7,5X,
       *'CONDITION=',I10)
        END
```

(4) 手法概要

ベッセル関数 $J_n(x)$ の計算は，$|x| = 1/8$ を境にして，計算式の形が異なる．$J_{-n}(x)=J_n(-x)=(-1)^nJ_n(x)$ であるから，以下では$|n|$, $|x|$それぞれ n, x として説明する.

a. $0 \le x < 1/8$ の場合

$$J_n(x)=\frac{x_n}{2^n n!}\left\{1-\frac{x^2}{2(2n+2)}+\frac{x^4}{2\cdot 4(2n+2)(2n+4)}+\cdots\right\} \quad (4.1)$$

により第 n 項目が初項に比べ丸め誤差の単位以下になるまで計算する.

b. $1/8 \le x \le 100$ の場合
n, x より十分大きな値 $\boldsymbol{M}$ を決め
$k=\boldsymbol{M},\boldsymbol{M}-1,...1$ に対して漸化式

$$F_{k-1}(x)=\frac{2k}{x}F_k(x)-F_{k+1}(x) \tag{4.2}$$

を反復適用する.
ただし $F_{M+1}(x)=0$, $F_M(x)=fl_{min}$ とする. このとき, $J_n(x)$ は,

$$J_n(x)=F_n(x)/\left\{F_0(x)+2\sum_{i=1}^{[M/2]}F_{2i}(x)\right\} \tag{4.3}$$

で与えられる.

[M の決め方]

$$M=2\left[\left\{m_0+1+\max\left(n-\frac{m_0+x}{2},0\right)\right\}/2\right] \tag{4.4}$$

ただし [ ] はガウス記号である.

(a) $1/8 \le x < 1$ のとき
単精度：$m_0=5.5x+5$ (4.5)
倍精度：$m_0=8x+10$ (4.6)

(b) $1 \le x < 10$ のとき
単精度：$m_0=1.8x+9$ (4.7)
倍精度：$m_0=2x+19$ (4.8)

(c) $10 \le x \le 100$ のとき
単精度：$m_0=1.25x+18$ (4.9)
倍精度：$m_0=1.3x+34$ (4.10)

なお, 詳細については, 参考文献 [81], [82] 参照のこと.

I11-83-0101 BJR, DBJR

第 1 種実数次ベッセル関数 $J_v(x)$
CALL BJR(X,V,BJ,ICON)

(1) 機能

第 1 種実数次ベッセル関数 $J_v(x)$

$$J_v(x)=\left(\frac{1}{2}x\right)^v\sum_{k=0}^{\infty}\frac{\left(-\frac{1}{4}x^2\right)^k}{k!\Gamma(v+k+1)}$$

の値をべき級展開式（上式），漸化式による方法及び漸近展開式を用いて計算する.

(2) パラメタ

X…… **入力**. 独立変数 x ($x\geq 0$).
V…… **入力**. $J_v(x)$ の次数 v ($v\geq 0$).
BJ …… **出力**. 関数値 $J_v(x)$.
ICON…… **出力**. コンディションコード.
表 BJR-1 参照.

表 BJR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	次のいずれかであった. · X>100 でかつ V>15 · X≥t_{max}	BJ = 0.0 とする.
30000	X<0 又は V<0 であった.	BJ = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, AFMIN, MGSSL, UTLIM
② FORTRAN 基本関数 ... FLOAT, ABS, GAMMA, AMAX1, MOD, SQRT, COS, SIN

b. 注意
① X≥0, V≥0 であること.
X, V の範囲は，図 BJR-1 の白地の部分とする. また，X<t_{max}
であることが必要であるが，これは X が大きくなると漸近展開の中の sin (X–(1/2·V+1/4)π) 及び cos (X–(1/2·V+1/4)π) の値が正確に計算できなくなるためである（手法概要の (4.4) 式参照）.

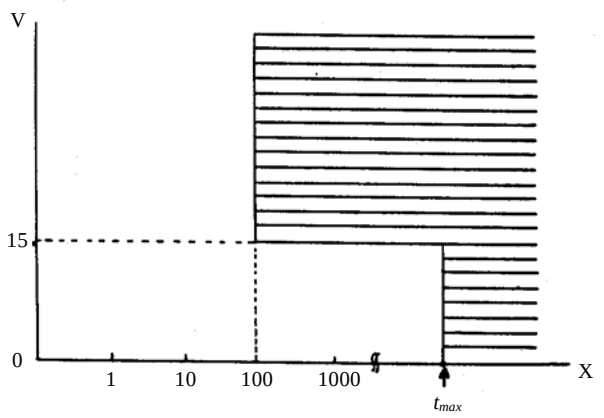


図 BJR-1 引数の範囲

② $J_0(x)$ あるいは $J_1(x)$ を計算するときには，本サブルーチンよりもそれぞれ BJ0, BJ1 を直接用いた方がよい.
③ $J_v(x)$, $J_{v+1}(x)$, $J_{v+2}(x)$, ..., $J_{v+M}(x)$ の値をいっせいに必要とするときには，本サブルーチンにより $J_{v+M}(x)$ と $J_{v+M-1}(x)$ を求め，漸化式を反復適用して $J_{v+M-2}(x)$, $J_{v+M-3}(x)$, ..., $J_v(x)$ と次数の低いものを求めていくとよい．逆に，本サブルーチンにより $J_v(x)$ と $J_{v+1}(x)$ を求め，漸化式により $J_{v+2}(x)$, $J_{v+3}(x)$, ..., $J_{v+M}(x)$ と次数の高いものを求めていくことは不安定が生ずるので避けなければならない.

c. 使用例
$J_v(x)$ の値を，x を 0 から 10 まで 1 おきに，v を 0.4 から 0.6 まで 0.01 刻みで計算する.

```
C     **EXAMPLE**
      DO 20 K=1,11
      X=K-1
      DO 10 NV=40,60
      V=FLOAT(NV)/100.0
      CALL BJR(X,V,BJ,ICON)
      IF(ICON.EQ.0) WRITE(6,600) X,V,BJ
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT(' ',F8.2,F8.3,E17.7)
      END
```

(4) 手法概要

求められるべき $J_v(x)$ の値がアンダフローする値（約 10^{-75} 以下）になることがあらかじめ分かった場合には，以下の計算は行わず，結果を 0.0 とする.
$J_v(x)$ の計算は，x 及び v により方法が異なる.

① $0 \le x < 1$ の場合
べき級数展開

$$J_v(x) = \left(\frac{1}{2}x\right)^v \sum_{k=0}^{\infty} \frac{\left(-\frac{1}{4}x^2\right)^k}{k!\Gamma(v+k+1)} \qquad (4.1)$$

により，第 k 項が初項に比べ丸め誤差の単位以下になるまで計算する．

② 単精度：$1 \le x \le 16$, 又は $16 < x \le 100$ かつ $v > 0.115x+4$ の場合
倍精度：$1 \le x < 30$, 又は $30 \le x \le 100$ かつ $v > 0.115x+4$ の場合
漸化式による方法を用いて計算する．
m を適当に大きな整数（x, v 及び要求精度によって決まる），δを十分に小さい定数（使用する計算機の正の最小数）とする．n 及び α を次式のように決める．

$v = n + \alpha$ (n：整数，$0 \le \alpha < 1$)

初期値を

$$F_{\alpha+m+1}(x) = 0, \quad F_{\alpha+m}(x) = \delta$$

として，$k = m, m-1, \ldots, 1$ に対して漸化式

$$F_{\alpha+k-1}(x) = \frac{2(a+k)}{x} F_{\alpha+k}(x) - F_{a+k+1}(x) \qquad (4.2)$$

を反復適用する．このとき，$\boldsymbol{J}_v(x)$ は

$$J_v(x) \approx \left(\frac{x}{2}\right)^a F_{a+n}(x) \Bigg/ \left(\sum_{k=0}^{[m/2]} \frac{(a+2k)\Gamma(a+k)}{k!} \times F_{a+2k}(x) \right) \qquad (4.3)$$

として求められる．
なお，m の決め方及びその他詳細については，参考文献 [81] を参照のこと．

③ 単精度：$100 < x < t_{max}$ かつ $v \le 15$，又は，$16 < x \le 100$ かつ $v \le 0.115x+4$ の場合
倍精度：$100 < x < t_{max}$ かつ $v \le 15$，又は，$30 \le x \le 100$ かつ $v \le 0.115x+4$ の場合
漸近展開式

$$J_v(x) = \sqrt{\frac{2}{\pi x}} \left\{ P(x,v) \cos\left(x - \left(\frac{1}{2}v + \frac{1}{4}\right)\pi \right) - Q(x,v) \sin\left(x - \left(\frac{1}{2}v + \frac{1}{4}\right)\pi \right) \right\} \qquad (4.4)$$

を用いて計算する．ただし

$$P(x,v) = \sum_{k=0}^{\infty} (-1)^k \frac{(v,2k)}{(2x)^{2k}} \qquad (4.5)$$

$$Q(x,v) = \sum_{k=0}^{\infty} (-1)^k \frac{(v,2k+1)}{(2x)^{2k}} \qquad (4.6)$$

$$(v,k) = \frac{\left(v^2 - \left(\frac{1}{2}\right)^2\right)\left(v^2 - \left(\frac{3}{2}\right)^2\right) \cdots\cdots \left(v^2 - \left(\frac{2k-1}{2}\right)^2\right)}{k!} \quad (k \ne 0)$$

$(v, 0) = 1$

である．$P(x,v)$ 及び $Q(x,v)$ はそれぞれ第 k 項が第 1 項に比べ
単精度：max (丸め誤差の単位，10^{-10})
倍精度：max (丸めの誤差の単位，10^{-20})
以下になるまで計算する．

I11-81-0201 BJ0,DBJ0

第 1 種 0 次ベッセル関数 $J_0(x)$
CALL BJ0(X,BJ,ICON)

(1) 機能

第 1 種 0 次ベッセル関数 $J_0(x)$ の値

$$J_0(x)=\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k}}{(k!)^2}$$

を有理式及び，漸近形式の近似式を用いて計算する．

(2) パラメタ

X…… **入力**．独立変数 x.
BJ …… **出力**．関数値 $J_0(x)$.
ICON…… **出力**．コンディションコード.
表 BJ0-1 参照.

表 BJ0-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$\|X\| \ge t_{max}$ であった.	BJ = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UTLIM
② FORTRAN 基本関数 ... ABS, DSIN, DCOS, DSQRT

b. 注意
引数 X の範囲は

$$|X| \le t_{max}$$

であること.
$|X|$ が大きくなると sin(x−π/4), cos(x−π/4) の値が不正確になるためである（手法概要 (4.4) 参照）.

c. 使用例
$J_0(x)$ の値を x が 0 から 100 まで, 1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=K-1
      CALL BJ0(X,BJ,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,BJ
      IF(ICON.NE.0) WRITE(6,620) X,BJ,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',9X,'J0(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'BJ=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4) 手法概要

ベッセル関数 $J_0(x)$ の計算は，$|x|$=8 を境にして，近似式の形が異なる．$J_0(-x)=J_0(x)$ であるから，以下では$|x|$を x として説明する.

a. $0 \le x \le 8$ の場合
$J_0(x)$ のべき級数展開

$$J_0(x)=\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k}}{(k!)^2} \tag{4.1}$$

を次の有理近似式を用いて計算する.
単精度：

$$J_0(x)=\sum_{k=0}^{5}a_k x^{2k}\Big/\sum_{k=0}^{5}b_k x^{2k} \tag{4.2}$$

理論精度 8.54桁
倍精度：

$$J_0(x)=\sum_{k=0}^{8}a_k x^{2k}\Big/\sum_{k=0}^{8}b_k x^{2k} \tag{4.3}$$

理論精度 19.22桁

b. $x>8$ の場合
$J_0(x)$ の漸近展開

$$J_0(x)=\sqrt{\frac{2}{\pi x}}\left\{P_0(x)\cos(x-\pi/4) - Q_0(x)\sin(x-\pi/4)\right\} \tag{4.4}$$

より計算する．ただし $P_0(x)$, $Q_0(x)$ は次の近似式を用いる.
単精度：

$$P_0(x)=\sum_{k=0}^{2}a_k z^{2k}\Big/\sum_{k=0}^{2}b_k z^{2k},\quad z=8/x \tag{4.5}$$

理論精度 10.66桁

$$Q_0(x)=\sum_{k=0}^{1}c_k z^{2k+1}\Big/\sum_{k=0}^{2}d_k z^{2k},\quad z=8/x \tag{4.6}$$

理論精度 9.58桁
倍精度：

$$P_0(x)=\sum_{k=0}^{5}a_k z^{2k}\Big/\sum_{k=0}^{5}b_k z^{2k}, z=8/x \tag{4.7}$$

理論精度 18.16桁

$$Q_0(x)=\sum_{k=0}^{5}c_k z^{2k+1}\Big/\sum_{k=0}^{5}d_k z^{2k},\quad z=8/x \tag{4.8}$$

理論精度 18.33桁

なお，詳細については，参考文献 [78] のpp.141~149 を参照のこと.

I11-81-0301 BJ1, DBJ1

第 1 種 1 次ベッセル関数 $\boldsymbol{J}_1(x)$
CALL BJ1(X,BJ,ICON)

(1) 機能

第 1 種 1 次ベッセル関数 $\boldsymbol{J}_1(x)$ の値

$$J_1(x)=\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k+1}}{k!(k+1)!}$$

を有理式及び，漸近形式の近似式を用いて計算する．

(2) パラメタ

X …… **入力**．独立変数 x.
BJ …… **出力**．関数値 $\boldsymbol{J}_1(x)$.
ICON …… **出力**．コンディションコード．
表 BJ1-1 参照.

表 BJ1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$\|X\| \ge t_{max}$ であった.	BJ = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UTLIM
② FORTRAN 基本関数 ... ABS, DSIN, DCOS, DSQRT

b. 注意
① 引数 X の範囲は

$$|X| \le t_{max}$$

であること．
$|X|$が大きくなると $\sin(x-3\pi/4)$, $\cos(x-3\pi/4)$ の値が不正確になるためである（手法概要 (4.4) 参照）．

c. 使用例
$\boldsymbol{J}_1(x)$ の値を x が 0 から 100 まで，1 おきに計算し数表を作る．

```
C      **EXAMPLE**
       WRITE(6,600)
       DO 10 K=1,101
       X=K-1
       CALL BJ1(X,BJ,ICON)
       IF(ICON.EQ.0) WRITE(6,610) X,BJ
       IF(ICON.NE.0) WRITE(6,620) X,BJ,ICON
    10 CONTINUE
       STOP
   600 FORMAT('1','EXAMPLE OF BESSEL ',
      *'FUNCTION'///6X,'X',9X,'J1(X)'/)
   610 FORMAT(' ',F8.2,E17.7)
   620 FORMAT(' ','** ERROR **',5X,'X=',
      *E17.7,5X,'BJ=',E17.7,5X,
      *'CONDITION=',I10)
       END
```

(4) 手法概要

ベッセル関数 $\boldsymbol{J}_1(x)$ の計算は，$|x| = 8$ を境にして，近似式の形が異なる．$\boldsymbol{J}_1(-x)=-\boldsymbol{J}_1(x)$ であるから，以下では $|x|$を x として説明する．

a. $0 \le x \le 8$ の場合
$\boldsymbol{J}_1(x)$ のべき級数展開

$$J_1(x)=\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k+1}}{k!(k+1)!} \tag{4.1}$$

を次の有理近似式を用いて計算する．
単精度：

$$J_1(x)=\sum_{k=0}^{4}a_k x^{2k+1}\Big/\sum_{k=0}^{5}b_k x^{2k} \tag{4.2}$$

理論精度 8.19桁
倍精度：

$$J_1(x)=\sum_{k=0}^{7}a_k x^{2k+1}\Big/\sum_{k=0}^{8}b_k x^{2k} \tag{4.3}$$

理論精度 18.68桁

b. $x>8$ の場合
$\boldsymbol{J}_1(x)$ の漸近展開

$$J_1(x)=\sqrt{\frac{2}{\pi x}}\left\{P_1(x)\cos\left(x-3\pi/4\right)-Q_1(x)\sin\left(x-3\pi/4\right)\right\} \tag{4.4}$$

より計算する．ただし $P_1(x)$, $Q_1(x)$ は次の近似式を用いる．
単精度：

$$P_1(x)=\sum_{k=0}^{2}a_k z^{2k}\Big/\sum_{k=0}^{2}b_k z^{2k},\quad z=8/x \tag{4.5}$$

理論精度 10.58桁

$$Q_1(x)=\sum_{k=0}^{1}c_k z^{2k+1}\Big/\sum_{k=0}^{2}d_k z^{2k},\quad z=8/x \tag{4.6}$$

理論精度 9.48桁
倍精度：

$$P_1(x)=\sum_{k=0}^{5}a_k z^{2k}\Big/\sum_{k=0}^{5}b_k z^{2k},\quad z=8/x \tag{4.7}$$

理論精度 18.11桁

$$Q_1(x)=\sum_{k=0}^{5}c_k z^{2k+1}\Big/\sum_{k=0}^{5}d_k z^{2k},\quad z=8/x \tag{4.8}$$

理論精度 18.28桁

なお,詳細については，参考文献 [78] の pp.141~149 を参照のこと．

I11-81-1301 BKN, DBKN

第 2 種整数次変形ベッセル関数 $K_n(x)$
CALL BKN(X,N,BK,ICON)

(1) 機能

第 2 種整数次変形ベッセル関数 $K_n(x)$ の値

$$K_n(x)=(-1)^{n+1}I_n(x)\{\gamma+\log(x/2)\}+\frac{1}{2}\sum_{k=0}^{n-1}\frac{(-1)^k(n-k-1)!}{k!}(x/2)^{2k-n}+\frac{(-1)^n}{2}\sum_{k=0}^{\infty}\frac{(x/2)^{n+2k}}{k!(n+k)!}\left(\sum_{m=1}^{k}1/m+\sum_{m=1}^{k+n}1/m\right)$$

を漸化式を用いて計算する．ただし $x>0$ であること．ただし，$\boldsymbol{I}_n(x)$：第 1 種整数次変形ベッセル関数，γ：オイラー定数.

(2) パラメタ

X…… **入力**．独立変数 x.
N…… **入力**．$\boldsymbol{K}_n(x)$ の次数 n.
BK …… **出力**．関数値 $\boldsymbol{K}_n(x)$.
ICON …… **出力**．コンディションコード.
表 BKN-1 参照.
ただし，N=0 又は N=1 のときはそれぞれ BK0, BK1 の ICON に準ずる.

表 BKN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	N>log(fl_{max}) であった.	BK = 0.0 とする.
30000	X≤0 であった.	BK = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... BK0, BK1, MGSSL, ULMAX
② FORTRAN 基本関数 ... IABS, FLOAT, ALOG, EXP, SQRT

b. 注意
① 引数 X の範囲は

$0<X\leq\log(fl_{max})$

であること.
X が上記範囲を超えると e^{-x} の計算においてアンダフローする．これを本サブルーチン内で前もって防ぐためである.
（BK0, BK1 手法概要 (4.4), (4.5) 参照）
② $\boldsymbol{K}_0(x)$ 及び $\boldsymbol{K}_1(x)$ を計算するには，本サブルーチンよりもそれぞれ，BK0, BK1 を直接用いた方がよい.

c. 使用例
$\boldsymbol{K}_n(x)$ の値を x が 1 から 10，N が 20 から 30 まで 1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 20 N=20,30
      DO 10 K=1,10
      X=K
      CALL BKN(X,N,BK,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,N,BK
      IF(ICON.NE.0) WRITE(6,620) X,N,BK,
     *                           ICON
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',5X,'N',8X,
     *'KN(X)'/)
  610 FORMAT(' ',F8.2,I5,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'N=',I5,5X,'BK=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4) 手法概要

$K_n(x)$ に関する次の漸化式関係を用いて計算する.

$$K_{k+1}(x)=\frac{2k}{x}K_k(x)+K_{k-1}(x),k=1,2,...,n-1 \qquad (4.1)$$

ただし $\boldsymbol{K}_0(x)$, $\boldsymbol{K}_1(x)$ は，それぞれ BK0, BK1 を利用している.

I11-83-0401 BKR, DBKR

第 2 種実数次変形ベッセル関数 $K_v(x)$
CALL BKR(X,V,BK,ICON)

(1) 機能

第 2 種実数次変形ベッセル関数 $K_v(x)$

$$K_v(x)=\frac{\pi}{2}\frac{I_{-v}(x)-I_v(x)}{\sin(v\pi)}$$

の値を，吉田・二宮による方法で計算する．
$I_v(x)$ は第 1 種変形ベッセル関数である．$x>0$ であること．

(2) パラメタ

X …… **入力**．独立変数 x.
V …… **入力**．$K_v(x)$ の次数 v.
BK …… **出力**．関数値 $K_v(x)$.
ICON …… **出力**．コンディションコード.
表 BKR-1 参照.

表 BKR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X=0.0 であった．又は，BK がオーバフローするほど大きな値となった.	BK には浮動小数点数の最大値を出力する.
30000	X<0.0 であった.	BK=0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, AFMAX, MGSSL, ULMAX
② FORTRAN 基本関数 ... FLOAT, ALOG, AMAX1, ALGAMA, GAMMA, ABS, SQRT, EXP

b. 注意
① X>0.0 であること.
② $K_0(x)$ あるいは $K_1(x)$ を計算するときには，本サブルーチンよりも，それぞれ，BK0, BK1 を直接用いた方が効率が良い.
③ $K_v(x), K_{v+1}(x), K_{v+2}(x),..., K_{v+M}(x)$ の値をいっせいに必要とするときには，本サブルーチンにより，$K_v(x)$ と $K_{v+1}(x)$ を求め，漸化式（手法概要の (4.3)）を反復適用して，$K_{v+2}(x), K_{v+3}(x),..., K_{v+M}(x)$ と次数の高いものへと順次求めていくとよい.
④ x が大きいところでは，続けて同じ v の値で，本サブルーチンが呼ばれたときには，計算の共通部分を通過し，効率良く $K_v(x)$ の値を計算できるようにしてある.

c. 使用例
$K_v(x)$ の値を，v を 0.4 から 0.6 まで 0.01 おきに，x を 1 から 10 まで 1 きざみで計算する.

```
C     **EXAMPLE**
      DO 20 NV=40,60
      V=FLOAT(NV)/100.0
      DO 10 K=1,10
      X=FLOAT(K)
      CALL BKR(X,V,BK,ICON)
      WRITE(6,600) V,X,BK,ICON
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT(' ',F8.3,F8.2,E17.7,I7)
      END
```

(4) 手法概要

v 次の第 2 種変形ベッセル関数 $K_v(x)$ は，第 1 種変形ベッセル関数 $I_v(x)$ と $I_{-v}(x)$ を用いて，

$$K_v(x)=\frac{\pi}{2}\frac{I_{-v}(x)-I_v(x)}{\sin(v\pi)} \tag{4.1}$$

として定義される．ただし，次数 v が整数 n のときには，$v\to n$ の極限で与えられる．また，

$$K_{-v}(x)=K_v(x) \tag{4.2}$$

なる関係があるので，$v\geq 0$ の場合の計算法のみ考えれば十分である.

本計算法は，$0\leq v\leq 2.5$ では，直接に $K_v(x)$ の値を計算する．$v>2.5$ では，v の小数部分を μ とすると，$\mu\leq 0.5$ ならば $K_{\mu+1}(x)$ と $K_{\mu+2}(x)$ を，$\mu>0.5$ ならば $K_\mu(x)$ と $K_{\mu+1}(x)$ を直接求め，漸化式

$$K_{v+1}(x)=\frac{2v}{x}K_v(x)+K_{v-1}(x) \tag{4.3}$$

により $K_v(x)$ の値を計算する．以下に，吉田・二宮による，$0\leq v\leq 2.5$ での $K_v(x)$ の計算法について説明する.

$K_v(x)$ の計算は，x 及び v により方法が異なる.
図 BKR-1 の領域 A では，x が小さい場合の計算法，領域 B では x が大きい場合の計算法を用いる.

a. x が小さい場合の計算法（領域 A）
本方法は，次式で与えられる $I_{-v}(x)$ 及び $I_v(x)$ の級数展開を利用する（この計算法の詳細については参考文献 [88] 参照）.

$$I_{-v}(x)=\left(\frac{x}{2}\right)^{-v}\left\{\frac{1}{\Gamma(1-v)}+\frac{\left(\frac{x}{2}\right)^2}{1!\Gamma(2-v)}+\frac{\left(\frac{x}{2}\right)^4}{2!\Gamma(3-v)}+...\right\} \tag{4.4}$$

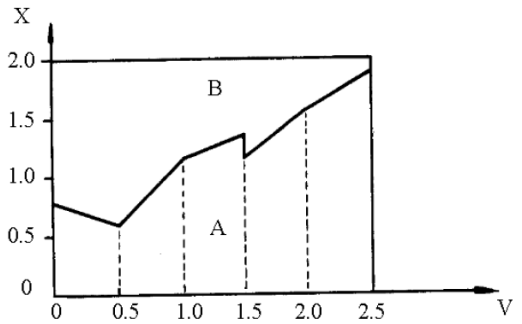


図 BKR-1 領域 A 及び領域 B

$$I_\nu(x)=\left(\frac{x}{2}\right)^\nu\left\{\frac{1}{\Gamma(1-\nu)}+\frac{\left(\frac{x}{2}\right)^2}{1!\,\Gamma(2-\nu)}+\frac{\left(\frac{x}{2}\right)^4}{2!\,\Gamma(3-\nu)}+\cdots\right\} \tag{4.5}$$

以下で用いる$\phi_1(\nu,x)$ 及び$\phi_2(\nu,x)$ を次のように定義する．

$$\left.\begin{aligned}\phi_1(\nu,x)&=\frac{\left(\frac{x}{2}\right)^{-\nu}-1}{\nu}\\ \phi_2(\nu,x)&=\frac{1-\left(\frac{x}{2}\right)^{\nu}}{\nu}\end{aligned}\right\} \tag{4.6}$$

ν の区間，$0\le\nu\le0.5$, $0.5<\nu\le1.5$, $1.5<\nu\le2.5$ によって計算式が異なる．ここでは，$0\le\nu\le0.5$ での計算法を説明する．(4.4), (4.5) 及び (4.6) から，次式が得られる．

$$I_{-\nu}(x)-I_\nu(x)=\nu\sum_{k=0}^{\infty}\{A_k(\nu,x)+B_k(\nu,x)\} \tag{4.7}$$

ただし，

$$\left.\begin{aligned}A_k(\nu,x)&=\left(\frac{x}{2}\right)^{2k}\left\{\frac{1}{k!\nu}\left(\frac{1}{\Gamma(k+1-\nu)}-\frac{1}{\Gamma(k+1+\nu)}\right)\right\}\\ B_k(\nu,x)&=\left(\frac{x}{2}\right)^{2k}\left\{\frac{1}{k!}\left(\frac{\phi_1(\nu,x)}{\Gamma(k+1-\nu)}-\frac{\phi_2(\nu,x)}{\Gamma(k+1+\nu)}\right)\right\}\end{aligned}\right\} \tag{4.8}$$

である．上式の $A_k(\nu,x)$ 及び $B_k(\nu,x)$ は，そのまま式のとおりに計算を行うと，$\nu\approx0$ で桁落ちが生ずる．桁落ちを避けるために以下で述べる工夫を行う．
まず，$B_k(\nu,x)$ において，$\phi_1(\nu,x)$ と$\phi_2(\nu,x)$ は同符号であるので，その加算では桁落ちは無い．

しかし，(4.6) で定義されている$\phi_1(\nu,x)$ 及び$\phi_2(\nu,x)$ は，その右辺のとおりに計算できるのは，$(x/2)^\nu<1/2$, あるいは, $(x/2)^\nu>2$ の場合だけである．$1/2\le(x/2)^\nu\le2$, すなわち, $-\log2\le\nu\log(x/2)\le\log2$ のときには桁落ちをする (ここでは，$a+b$ なる加算において，$\max(|a|,|b|)/|a+b|\ge2$ のとき，桁落ちが生ずるということにする．これは，2 進数で 1 桁以上の桁落ちが生じた場合に相当する)．桁落ち無しに$\phi_1(\nu,x)$ 及び $\phi_2(\nu,x)$ を計算するためには，関数

$$f(t)=\frac{e^t-1}{t}=\frac{1}{1!}+\frac{t}{2!}+\frac{t^2}{3!}+\cdots\cdots \tag{4.9}$$

について，$-\log2\le t\le\log2$ において所要の相対精度を有する最良近似式があればよい．なぜならば，その近似式により，

$$\left.\begin{aligned}\phi_1(\nu,x)&=-f\left(-\nu\log\left(\frac{x}{2}\right)\right)\log\left(\frac{x}{2}\right)\\ \phi_2(\nu,x)&=-f\left(\nu\log\left(\frac{x}{2}\right)\right)\log\left(\frac{x}{2}\right)\end{aligned}\right\} \tag{4.10}$$

を計算すればよいからである．本サブルーチンでは，f(t) に対して，次の形の最良近似式

$$f(t)\approx\frac{2\sum_{k=0}^{M}p_kt^{2k}}{\sum_{k=0}^{N}q_kt^{2k}-t\sum_{k=0}^{M}p_kt^{2k}} \tag{4.11}$$

を用いる（M, N, p_k 及び q_k の具体的な数値については [88] 参照）．
(4.8) の $A_k(\nu,x)$ においても，$(1/\Gamma(k+1-\nu)-1/\Gamma(k+1+\nu))/(k!\nu)$ の値を桁落ち無しに計算するためには，それに対して所要の精度を有する最良近似式があればよい．ただし，(4.8) の $A_k(\nu,x)$ の { } の部分を，$\tilde{A}_k(\nu)$ と表したとき，$\tilde{A}_k(\nu)$ に対して，

$$\tilde{A}_k(\nu)=\frac{1}{(k+\nu)(k-\nu)}\left[\frac{1}{k!}\left\{\frac{1}{\Gamma(k-\nu)}+\frac{1}{\Gamma(k+\nu)}\right\}+\tilde{A}_{k-1}(\nu)\right]\qquad(k\ge1) \tag{4.12}$$

なる関係が成り立つので，$\tilde{A}_0(\nu)$ の近似式があればよいはずである．しかし，$k=1$ に対しては，上式は桁落ちを伴うので，$k\ge2$ に対して，上式が適用できる．したがって，$\tilde{A}_0(\nu)$ 及び $\tilde{A}_1(\nu)$ の $0\le\nu\le0.5$ における最良近似式が必要である．本サブルーチンでは，

$$\left.\begin{aligned}\tilde{A}_0(v) &\approx \sum_{k=0}^{M} p_k v^{2k} \\ \tilde{A}_1(v) &\approx \sum_{k=0}^{N} q_k v^{2k}\end{aligned}\right\} \qquad (4.13)$$

の形の最良近似多項式によって計算を行っている（M, p_k, N, q_kについては，[88] 参照）．
このようにして，$A_k(v,x)$ 及び $B_k(v,x)$ は桁落ち無しに求めることができる．$0 \le v \le 0.5$ では $A_0(v,x) \le 0$, $A_k(v,x) \ge 0$ $(k=1,2,...)$ である．また，$x \le 2$ のとき，$B_k(v,x) \ge 0$，$x>2$ のとき，$B_k(v,x)<0$ $(k=0,1,2,...)$ である．したがって，(4.7) の加算において桁落ちの可能性がある．しかし，実際に，数値的に桁落ちの有無を調べた結果，図 BKR-1 の領域 A の $0 \le v \le 0.5$ において，桁落ちが生じない．(4.7) の和を構成している項は，k が大きくなるにつれて十分に小さくなるので，要求精度内に収束するまでに取る項は少なくて済む．
したがって，

$$g(v) = \frac{2}{\pi v}\sin(v\pi) \qquad (4.14)$$

なる関数に対する次の形の最良近似式

$$g(v) \approx \sum_{k=0}^{M} p_k v^{2k} \qquad (4.15)$$

を用いれば（M 及び p_k については [88] 参照），$\boldsymbol{K}_v(x)$ の値は，

$$K_v(x) = \frac{I_{-v}(x) - I_v(x)}{\frac{2}{\pi}\sin(v\pi)} \approx \frac{\sum_{k=0}^{\infty}\{A_k(v,x) + B_k(v,x)\}}{g(v)} \qquad (4.16)$$

により計算することができる．
$v=0$ のときには，(4.16) により $\boldsymbol{K}_0(x)$ を計算するよりは，むしろ，(4.1) の $v \to 0$ の極限として得られる

$$K_0(x) = \sum_{k=0}^{\infty} \frac{\left(\frac{x}{2}\right)^{2k}}{(k!)^2}\left\{-\left(\gamma + \log\left(\frac{x}{2}\right)\right) + \Phi_k\right\} \qquad (4.17)$$

により特別に計算すると効率がよい．ただし，γ はオイラーの定数，$\Phi = 0$, $\Phi_k = \sum_{m=0}^{k} \frac{1}{m} (k \ge 1)$ である．また，要求する相対精度を ε とすると，

$$v < 1.723 x \varepsilon^{\frac{1}{2}} \qquad (4.18)$$

のときには，その精度内で $\boldsymbol{K}_v(x)$ と $\boldsymbol{K}_0(x)$ は一致するので，そのときの $\boldsymbol{K}_v(x)$ は (4.17) により計算を行う．
以上により，$0 \le v \le 0.5$ での計算法について述べた．$0.5 < v \le 1.5$ 及び $1.5 < v \le 2.5$ の場合の計算法については省略するが，ほぼ同様な方法を考えることができ，図 BKR-1 の領域 A において適用できることが分っている．

b. x が大きい場合の計算法 (領域 B)
本計算法は，τ-method による整数次の $\boldsymbol{K}_n(x)$ の計算法（参考文献 [84] 参照）を一般化し，任意の実数次の $\boldsymbol{K}_v(x)$ の値を能率的に計算できるように工夫したものである．
この方法では，$\boldsymbol{K}_v(x)$ を

$$K_v(x) = \sqrt{\frac{\pi}{2x}} e^{-x} f_v\left(\frac{1}{x}\right) \qquad (4.19)$$

の形として，$f_v(1/x)$ についての近似式を求めている．以下，$t=1/x$ とする．$f_v(t)$ は

$$t^2 f_v''(t) + 2(t+1) f_v'(t) - \left(v^2 - \frac{1}{4}\right) f_v(t) = 0 \qquad (4.20)$$

を満足する．(4.20) の右辺に，直交区間 $[0,\eta]$ のずらし超球 (shifted Ultraspherical) 多項式を τ 倍したものを付加すると，

$$t^2 f_v''(t) + 2(t+1) f_v'(t) - \left(v^2 - \frac{1}{4}\right) f_v(t) = \tau C_m^{*(\alpha)}\left(\frac{t}{\eta}\right) \qquad (4.21)$$

となる．ここで，

$$C_m^{*(\alpha)}(t) = \sum_{i=0}^{m} C_{mi}^{*(\alpha)} t^i \qquad (4.22)$$

は，ずらし超球多項式である．(4.21) は次の形の m 次の多項式解をもつ．

$$f_{vm}(t) = \tau \sum_{k=0}^{m} \frac{C_{mk}^{*(a)} \sum_{i=0}^{k} a_i t^i}{(k+1) a_{k+1} \eta^k} \qquad (4.23)$$

ただし，

$$\left.\begin{aligned} a_0 &= 1 \\ a_k &= \frac{\left(4v^2 - 1^2\right)\left(4v^2 - 3^2\right).....\left(4v^2 - (2k-1)^2\right)}{k! 8^k} \\ &\quad (k \ge 1)\end{aligned}\right\} \qquad (4.24)$$

である．この $f_{vm}(t)$ は $f_v(t)$ に対する近似多項式と考えることができる．初期条件 $f_{vm}(0)=1$ ($t \to 0$ のとき，$f_v(t) \to 1$) から τ を決めると，

$$f_{vm}(t)=\frac{\sum_{k=0}^{m}\frac{C_{mk}^{*(a)}\sum_{i=0}^{k}a_i t^i}{(k+1)a_{k+1}\eta^k}}{\sum_{k=0}^{m}\frac{C_{mk}^{*(a)}}{(k+1)a_{k+1}\eta^k}} \tag{4.25}$$

が得られる．(4.25) は，未知数として，α 及び η を含んでいるが，α=0.5 (このとき $C_m^{*(a)}(t)$ はずらしルジャンドル多項式となる)，$\eta=t$ のとき，最も精度が高いことが既に分っている．そのように選べば，

$$f_{vm}(t)=\frac{\sum_{k=0}^{m}\frac{P_{mk}^{*}\sum_{i=0}^{k}a_i t^i}{(k+1)a_{k+1}t^k}}{\sum_{k=0}^{m}\frac{P_{mk}^{*}}{(k+1)a_{k+1}t^k}} \tag{4.26}$$

となる．ただし，$\boldsymbol{P}_{mk}^{*}$ は，ずらしルジャンドル多項式

$$P_m^{*}(t)=\sum_{i=0}^{m}P_{mi}^{*}t^i \tag{4.27}$$

の係数である．(4.26) の分母，分子に t^m を乗じ，t のべきでまとめれば，

$$f_{vm}(t)=\frac{\sum_{i=0}^{m}G_i(m,v)t^i}{\sum_{i=0}^{m}H_i(m,v)t^i} \tag{4.28}$$

となる．ただし，

$$G_i(m,v)=\sum_{k=0}^{i}\frac{P_{m,m-i+k}^{*}a_k}{(m+1-i+k)a_{m+1-i+k}} \tag{4.29}$$

$$H_i(m,v)=\frac{P_{m,m-i}^{*}}{(m-i+1)a_{m-i+1}} \tag{4.30}$$

である．
(4.29) は，v^2 のべきで次のように書き表すことができる．

$$G_i(m,v)=\frac{1}{a_{m+1}}\sum_{j=0}^{i}b_{ij}\left(v^2\right)^j \tag{4.31}$$

ただし，

$$b_{ij}=2^{2j-3i}\sum_{l=0}^{i}\sum_{k=j-l}^{i-l}\frac{P_{m,m-k}^{*}p_{i-k,l}q_{k,j-l}}{(i-k)!\prod_{n=0}^{k}(m-n+1)} \tag{4.32}$$

である．上式の $p_{k,l}$ 及び $q_{k,l}$ は，それぞれ，

$$p_{0,0}=1,\ p_{1,0}=-1,\ p_{1,1}=1 \tag{4.33}$$

$$q_{0,0}=1,\ q_{1,0}=-(2m+1)^2,\ q_{1,1}=1 \tag{4.34}$$

を初期値として，$2\le k\le i$ に対して，次の漸化式により定義されるものである．

$$\left.\begin{aligned}p_{k,0}&=-(2k-1)^2p_{k-1,0}\\ p_{k,l}&=p_{k-1,l-1}-(2k+1)^2p_{k-1,l}\quad(1\le l\le k-1)\\ p_{k,k}&=p_{k-1,k-1}\end{aligned}\right\} \tag{4.35}$$

$$\left.\begin{aligned}q_{k,0}&=-(2m-2k+3)^2q_{k-1,0}\\ q_{k,l}&=q_{k-1,l-1}-(2m-2k+3)^2q_{k-1,l}\,(1\le l\le k-1)\\ q_{k,k}&=q_{k-1,k-1}\end{aligned}\right\} \tag{4.36}$$

また，(4.30) は，次のように表すことができる．

$$H_i(m,v)=\frac{1}{a_{m+1}}c_i\Psi_i \tag{4.37}$$

ただし，

$$c_i=\frac{(m-i)!\,P_{m,m-i}^{*}}{(m+1)!(-2)^i} \tag{4.38}$$

$$\left.\begin{aligned}\Psi_0&=1\\ \Psi_i&=\prod_{l=0}^{i-1}\left\{\left(m-l+\frac{1}{2}\right)^2-v^2\right\}(i\ge1)\end{aligned}\right\} \tag{4.39}$$

である．
以上より，$\boldsymbol{K}_v(x)$ の計算式として，(4.19), (4.28), (4.31) 及び (4.37) を用いて

$$\mathrm{K}_v(x)\approx\sqrt{\frac{1}{x}}e^{-x}\frac{\sum_{i=0}^{m}\left(\frac{1}{x}\right)^i\sum_{j=0}^{i}d_{ij}\left(v^2\right)^j}{\sum_{i=0}^{m}\left(\frac{1}{x}\right)^i e_i\psi_i} \tag{4.40}$$

が得られる．ただし，

$$\left.\begin{aligned}d_{ij}&=b_{ij}/b_{1,1}\\ e_i&=\sqrt{\frac{2}{\pi}}\,c_i/b_{1,1}\end{aligned}\right\} \tag{4.41}$$

である．(4.40) は，m を固定したとき，x を大きくすると精度が高くなり，また，x を固定したとき，m を大きくすると精度が高くなる．図 BKR-1 の領域 B において (4.40) を用いる．能率の観点から，本サブルーチンでは，単精度の場合には，

$$\left.\begin{array}{ll} x<2 \text{ では,} & m=9 \\ 2\le x<10 \text{ では,} & m=6 \\ 10\le x \text{ では,} & m=4 \end{array}\right\} \tag{4.42}$$

とし，倍精度の場合には，

$$\left.\begin{array}{ll} x<2 \text{ では,} & m=28 \\ 2\le x<10 \text{ では,} & m=16 \\ 10\le x \text{ では,} & m=11 \end{array}\right\} \tag{4.43}$$

としている．本サブルーチン内では，定数 d_{ij} 及び e_i は，あらかじめ計算されているものをデータ文により表としてもっている．

I11-81-0801 BK0, DBK0

第 2 種 0 次変形ベッセル関数 $\boldsymbol{K}_0(x)$
CALL BK0(X,BK,ICON)

(1) 機能

第 2 種 0 次変形ベッセル関数 $\boldsymbol{K}_0(x)$ の値

$$K_0(x)=\sum_{k=1}^{\infty}\frac{(x/2)^{2k}}{(k!)^2}\left(\sum_{m=1}^{k}1/m\right)-I_0(x)\{\gamma+\log(x/2)\}$$

（ただし，$\boldsymbol{I}_0(x)$：第 1 種 0 次変形ベッセル関数，γ ：オイラー定数）を多項式及び漸近形式の近似式を用いて計算する．

ただし，x>0 であること．

(2) パラメタ

X …… **入力**．独立変数 x.
BK …… **出力**．関数値 $\boldsymbol{K}_0(x)$.
ICON …… **出力**．コンディションコード.
表 BK0-1 参照.

表 BK0-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X>log(fl_{max}) であった.	BK = 0.0 とする.
30000	X≤0 であった.	BJ = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, ULMAX
② FORTRAN 基本関数 ... ALOG, EXP, SQRT

b. 注意
① 引数 X の範囲は

$0<X\leq\log(fl_{max})$

であること.
X が上記範囲を超えると e^{-x} の計算でアンダフローする．これを本サブルーチン内で前もって防ぐためである．（手法概要 (4.5), (4.6) 参照）

c. 使用例
$\boldsymbol{K}_0(x)$ の値を x が 1 から 100 まで，1 おきに計算し数表を作る.

```
C      **EXAMPLE**
       WRITE(6,600)
       DO 10 K=1,100
       X=K
       CALL BK0(X,BK,ICON)
       IF(ICON.EQ.0) WRITE(6,610) X,BK
       IF(ICON.NE.0) WRITE(6,620) X,BK,ICON
    10 CONTINUE
       STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',9X,'K0(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'BK=',E17.7,5X,
     *'CONDITION=',I10)
       END
```

(4) 手法概要

変形ベッセル関数 $\boldsymbol{K}_0(x)$ の計算は x=2 を境にして，近似式の形が異なる.

a. 0<x<2 の場合
$\boldsymbol{K}_0(x)$ のべき級数展開

$$K_0(x)=\sum_{k=1}^{\infty}\frac{(x/2)^{2k}}{(k!)^2}\left(\sum_{m=1}^{k}1/m\right)-I_0(x)\{\gamma+\log(x/2)\} \tag{4.1}$$

（ただし，$\boldsymbol{I}_0(x)$：第 1 種 0 次変形ベッセル関数，γ：オイラー定数）をそれぞれ (4.2), (4.3) の近似式を用いて計算する.
単精度：

$$K_0(x)=\log\left(\frac{x}{2}\right)\sum_{k=0}^{5}a_k x^{2k}+\sum_{k=0}^{5}b_k x^{2k} \tag{4.2}$$

倍精度：

$$K_0(x)=\log\left(\frac{x}{2}\right)\sum_{k=0}^{9}a_k x^{2k}+\sum_{k=0}^{9}b_k x^{2k} \tag{4.3}$$

b. $2\leq x\leq\log(fl_{max})$ の場合
$\boldsymbol{K}_0(x)$ の漸近展開

$$K_0(x)=\sqrt{\frac{\pi}{2x}}e^{-x}\left\{1+\frac{(-1)}{8x}+\frac{(-1)(-9)}{2!}\cdot\frac{1}{(8x)^2}+\frac{(-1)(-9)(-25)}{3!}\cdot\frac{1}{(8x)^3}+\ldots\right\} \tag{4.4}$$

をそれぞれ (4.5), (4.6) の近似式を用いて計算する.
単精度：

$$K_0(x)=\frac{e^{-x}}{\sqrt{x}}\left(\sum_{k=0}^{8}a_k z^k\right),\quad z=2/x \tag{4.5}$$

倍精度：

$$K_0(x)=\frac{e^{-x}}{\sqrt{x}}\left(\sum_{k=0}^{8}a_k z^k\Big/\sum_{k=0}^{8}b_k z^k\right) \tag{4.6}$$

I11-81-0901 BK1, DBK1

第 2 種 1 次変形ベッセル関数 $\boldsymbol{K}_1(x)$
CALL BK1(X,BK,ICON)

(1) 機能

第 2 種 1 次変形ベッセル関数 $\boldsymbol{K}_1(x)$ の値

$$K_1(x) = I_1(x)\{\gamma + \log(x/2)\} + \frac{1}{x} - 1/2 \cdot \sum_{k=0}^{\infty} \frac{(x/2)^{2k+1}}{k!(k+1)!}\left(\sum_{m=1}^{k} 1/m + \sum_{m=1}^{k+1} 1/m\right)$$

（ただし，$\boldsymbol{I}_1(x)$：第 1 種 1 次変形ベッセル関数，γ：オイラー定数）を多項式及び漸近形式の近似式を用いて計算する．

ただし，x>0 であること．

(2) パラメタ

X …… **入力**．独立変数 x.
BK …… **出力**．関数値 $\boldsymbol{K}_1(x)$.
ICON …… **出力**．コンディションコード．
表 BK1-1 参照.

表 BK1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X>log(fl_{max}) であった.	BK = 0.0 とする.
30000	X≤0.	BK = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, ULMAX
② FORTRAN 基本関数 ... ALOG, EXP, SQRT

b. 注意
① 引数 X の範囲は

0<X≤log(fl_{max})

であること.
Xが上記範囲を超えるとe^{-x}の計算でアンダフローする．これを本サブルーチン内で前もって防ぐためである．（手法概要 (4.5), (4.6) 参照）

c. 使用例
$\boldsymbol{K}_1(x)$ の値を x が 1 から 100 まで，1 おきに計算し数表を作る.

```
C       **EXAMPLE**
        WRITE(6,600)
        DO 10 K=1,100
        X=K
        CALL BK1(X,BK,ICON)
        IF(ICON.EQ.0) WRITE(6,610) X,BK
        IF(ICON.NE.0) WRITE(6,620) X,BK,ICON
   10 CONTINUE
        STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',9X,'K1(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'BK=',E17.7,5X,
     *'CONDITION=',I10)
        END
```

(4) 手法概要

変形ベッセル関数 $\boldsymbol{K}_1(x)$ の計算は x=2 を境にして，近似式の形が異なる.

a. 0<x<2 の場合
$\boldsymbol{K}_1(x)$ のべき級数展開

$$K_1(x) = I_1(x)\{\gamma + \log(x/2)\} + 1/x - 1/2 \cdot \sum_{k=0}^{\infty} \frac{(x/2)^{2k+1}}{k!(k+1)!}\left(\sum_{m=1}^{k} 1/m + \sum_{m=1}^{k+1} 1/m\right) \quad (4.1)$$

（ただし，$\boldsymbol{I}_1(x)$：第 1 種 1 次変形ベッセル関数，γ：オイラー定数）をそれぞれ (4.2), (4.3) の近似式を用いて計算する.
単精度：

$$K_1(x) = \left\{\log\left(\frac{x}{2}\right)\sum_{k=0}^{5} a_k x^{2k} + \sum_{k=0}^{5} b_k x^{2k}\right\} x + \frac{1}{x} \quad (4.2)$$

倍精度：

$$K_1(x) = \left\{\log\left(\frac{x}{2}\right)\sum_{k=0}^{9} a_k x^{2k} + \sum_{k=0}^{9} b_k x^{2k}\right\} x + \frac{1}{x} \quad (4.3)$$

b. 2 ≤ x ≤ log(fl_{max}) の場合
$\boldsymbol{K}_1(x)$ の漸近展開

$$K_1(x) = \sqrt{\frac{\pi}{2x}} e^{-x}\left\{1 + \frac{3}{8x} + \frac{3\cdot(-5)}{2!}\cdot\frac{1}{(8x)^2} + \frac{3\cdot(-5)\cdot(-21)}{3!}\cdot\frac{1}{(8x)^3} + \ldots\right\} \quad (4.4)$$

をそれぞれ (4.5), (4.6) の近似式を用いて計算する.
単精度：

$$K_1(x) = \frac{e^{-x}}{\sqrt{x}}\left(\sum_{k=0}^{8} a_k z^k\right), \quad z = 2/x \quad (4.5)$$

倍精度：

$$K_1(x) = \frac{e^{-x}}{\sqrt{x}}\left(\sum_{k=0}^{8} a_k x^k \Big/ \sum_{k=0}^{8} b_k x^k\right) \quad (4.6)$$

B21-11-0202 BLNC, DBLNC

実行列の平衡化
CALL BLNC(A,K,N,DV,ICON)

(1) 機能

n 次の実行列 $\boldsymbol{A}$ を (1.1) の対角相似変換により,平衡化する.

$$\tilde{\boldsymbol{A}} = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D} \qquad (1.1)$$

ここで，平衡化するとは，変換後の実行列 $\tilde{\boldsymbol{A}}$ について第 *i* 行 (*i*=1, ..., *n*) 要素のノルムの和と，第 *i* 列要素のノルムの和をほぼ等しくすることである．また，$\boldsymbol{D}$ は対角行列である.
n≥1なること.

(2)パラメタ

A…… **入力**. 実行列 $\boldsymbol{A}$.
出力. 平衡化後の実行列 $\tilde{\boldsymbol{A}}$.
A(K,N) なる 2 次元配列.

K…… **入力**. 配列 A の整合寸法.

N…… **入力**. 実行列 $\boldsymbol{A}$ 及び $\tilde{\boldsymbol{A}}$ の次数 *n*.

DV…… **出力**. スケーリングファクタ（$\boldsymbol{D}$ の対角要素）.
大きさ *n* の 1 次元配列.

ICON…… **出力**. コンディションコード.
表 BLNC-1 参照.

表 BLNC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1 であった.	平衡化は行わない.
30000	N<1 又は K<N であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... IRDIX, MGSSL
② FORTRAN 基本関数 ... ABS

b. 注意
① 行列の各要素の大きさが不揃いな場合，この行列の固有値及び固有ベクトルの計算値の精度が悪くなりやすい．本サブルーチンは，これを防ぐためのものである.
② 行列の各要素の大きさがほぼ揃っているときには，本サブルーチンにより何等の変換も行われないので，本サブルーチンの実行を省略した方がよい.
③ 本サブルーチンは行列の平衡化にあたり，行又は列の要素が，対角要素を除いてすべて零となるような行 (又は列) がある場合, その行(又は列)と対応する列 (又は行) については平衡化を省略する.
④ 実行列 $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ を求めるときは，本サブルーチンにより平衡化した行列 $\tilde{\boldsymbol{A}}$ の固有ベクトル $\tilde{\boldsymbol{x}}$ に対して，(3.1) の逆変換を行う必要がある.

$$\boldsymbol{x} = \boldsymbol{D}\tilde{\boldsymbol{x}} \qquad (3.1)$$

(3.1) の逆変換はサブルーチン HBK1 により実行できる（HBK1 参照のこと）.

c. 使用例
n 次の実行列 $\boldsymbol{A}$ を平衡化した後，サブルーチン HES1 により実ヘッセンベルグ行列に変換し，固有値をサブルーチン HSQR で求める.
n≤100 の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),DV(100),PV(100)
     *          ,ER(100),EI(100)
   10 CONTINUE
      READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL BLNC(A,100,N,DV,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
      CALL HES1(A,100,N,PV,ICON)
      CALL HSQR(A,100,N,ER,EI,L,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,630) (ER(I),EI(I),I=1,L)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',5X,
     * 'N=',I3/)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT('0',20X,'ICON=',I5)
  630 FORMAT('0',5X,'EIGENVALUE'/'0',20X,
     * 'REAL',16X,'IMAG'/('0',15X,2(E15.7,
     * 5X)))
      END
```

(4) 手法概要

n 次の実行列 $\boldsymbol{A}$ を，(4.1) の対角相似変換を反復的に行うことにより平衡化する.

$$\boldsymbol{A}_s=\boldsymbol{D}_s^{-1}\boldsymbol{A}_{s-1}\boldsymbol{D}_s \quad s = 1,2,... \qquad (4.1)$$

ここに，$\boldsymbol{A}_0=\boldsymbol{A}$ であり，$\boldsymbol{D}_s$ は (4.2) で表される対角行列，*s* は反復回数である.

$$\boldsymbol{D}_s = \begin{bmatrix} d_1^{(s)} & & & \\ & d_2^{(s)} & & 0 \\ & & \ddots & \\ & 0 & & d_n^{(s)} \end{bmatrix} \qquad (4.2)$$

(4.1) の平衡化は，$\boldsymbol{A}_s$の第 i 行 (i = 1, 2, ..., n) の対角要素を除いた絶対和と第 i 列の絶対和がほぼ等しくなるように行う．

すなわち，$\boldsymbol{A}_s=(a_{ij}^{(s)})$ として，

$$\sum_{\substack{k=1\\k\neq i}}^{n}\left|a_{ik}^{(s)}\right| \approx \sum_{\substack{k=1\\k\neq i}}^{n}\left|a_{ik}^{(s)}\right| \tag{4.3}$$

を満足するように平衡化する．

いま，$\boldsymbol{D}_{si}$ を

$$D_{si}=\begin{bmatrix} & & i & & \\ 1 & & & & \\ & \searrow & & 0 & \\ & & 1 & & \\ & & d_i^{(s)} & & i \\ & & & 1 & \\ & 0 & & & \\ & & & & \searrow \\ & & & & 1 \end{bmatrix} \tag{4.4}$$

と定義すれば，(4.2) の $\boldsymbol{D}_s$ は，

$$\boldsymbol{D}_s=\boldsymbol{D}_{s1}\boldsymbol{D}_{s2}\cdots\boldsymbol{D}_{sn} \tag{4.5}$$

と表せる．

(4.5) より，(4.1) の変換は，(4.6) で示す変換を，i=1,2,...,n と次々に行うことにより実行できることが分かる．

$$\boldsymbol{A}_{si}=\boldsymbol{D}_{si}^{-1}\boldsymbol{A}_{s\,i-1}\boldsymbol{D}_{si} \tag{4.6}$$

ただし $\mathbf{A}_{s0}=\mathbf{A}_{s-1}$, $\mathbf{A}_s=\mathbf{A}_{sn}$ である．

ここで，$\boldsymbol{D}_{si}$ すなわち $\boldsymbol{d}_i^{(s)}$ は，変換後の i 行 i 列が，(4.3) を満足するように定め，変換前に，既に i 行 i 列が，(4.3) を満足していれば，$\boldsymbol{d}_i^{(s)}$=1 すなわち $\boldsymbol{D}_{si}=\boldsymbol{I}$ とする．

(4.1) の反復は，すべての行と列に対し，(4.3) が満足されたとき終了する．

本ルーチンでは，この操作を以下に示す手順で実行する．

① 第 i 行と第 i 列の要素の絶対値の和を，対角要素を除いて計算する．

$$C \approx \sum_{\substack{k=1\\k\neq i}}^{n}\left|a_{ki}\right| \tag{4.7}$$

$$R \approx \sum_{\substack{k=1\\k\neq i}}^{n}\left|a_{ik}\right| \tag{4.8}$$

② $\boldsymbol{d}_i^{(s)}$を次のように定める．

$$\boldsymbol{d}_i^{(s)}=\rho^k \tag{4.9}$$

ここに$\rho=\begin{cases}2, & \text{2進法の計算機のとき}\\ 16, & \text{16進法の計算機のとき}\end{cases}$

なる定数である．

k は，次の条件を満足するように定める．

$$R\cdot\rho > \mathrm{C}\cdot\rho^{2k} \geq R/\rho \tag{4.10}$$

(4.10) より k は，$\boldsymbol{C}<\boldsymbol{R}/\rho$のとき，$k>0$ であり，$\boldsymbol{C}\geq\boldsymbol{R}/\rho$のとき，$k\leq 0$ である．

③ (4.11) の条件により変換の要不要を判定する．

$$\left(C\cdot\rho^{2k}+R\right)/\rho < 0.95\left(C+R\right) \tag{4.11}$$

(4.11) が満足されれば，$d_i^{(s)}=\rho^k$ として変換を行い，満足されなければ変換を省略する．

④ すべての行と列について，全く変換が行われなくなったとき，平衡化の処理を終了する．
このとき，配列 DV には，

$$\boldsymbol{D}=\boldsymbol{D}_1\boldsymbol{D}_2\ldots\boldsymbol{D}_s \tag{4.12}$$

で与えられる D の対角要素がスケーリングファクタとして格納されている．

なお，詳細については，参考文献 [13] pp.315-326 を参照すること．

A52-11-0302 BLUX1, DBLUX1

LU 分解された実バンド行列の連立 1 次方程式
CALL BLUX1(B,FA,N,NH1,NH2,FL,IP,ICON)

(1) 機能

LU 分解されたバンド行列を係数に持つ連立 1 次方程式

$$\boldsymbol{LUx}=\boldsymbol{b} \qquad (1.1)$$

を解く.

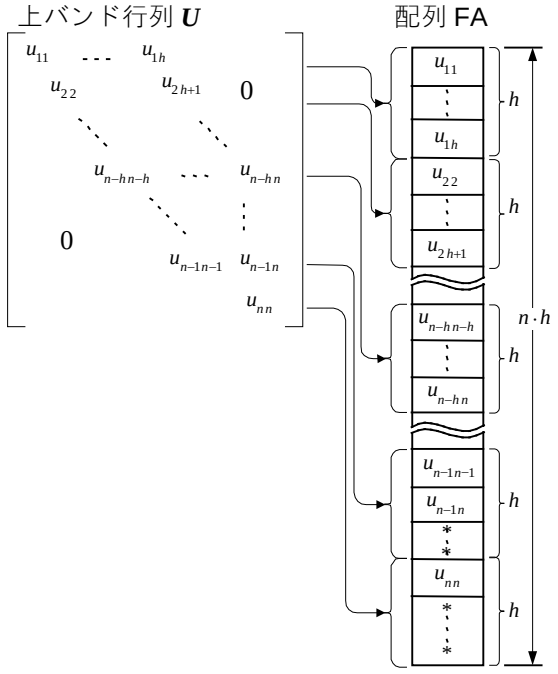
ただし, $\boldsymbol{L}$ は下バンド幅 h_1 の単位下バンド行列, $\boldsymbol{U}$ は上バンド幅 $h(=\min(h_1+h_2+1,n))$ の上バンド行列, b は n 次元の実定数ベクトルである. なお, LU 分解された元の行列の次数, 下バンド幅及び上バンド幅は n, h_1 及び h_2 である.

$n>h_1\geq 0$, $n>h_2\geq 0$ であること.

(2) パラメタ

B …… **入力**. 定数ベクトル b.
出力. 解ベクトル x.
大きさ n の 1 次元配列.

FA …… **入力**. 行列 $\boldsymbol{U}$.
図 BLUX1-1 参照.
大きさ $n\cdot h$ の 1 次元配列.



[注意] *...*印要素は任意の数値である.

図 BLUX1-1 配列 FA における $\boldsymbol{U}$ の各要素の格納方法

N …… **入力**. 行列 $\boldsymbol{L}$ 及び $\boldsymbol{U}$ の次数 n.

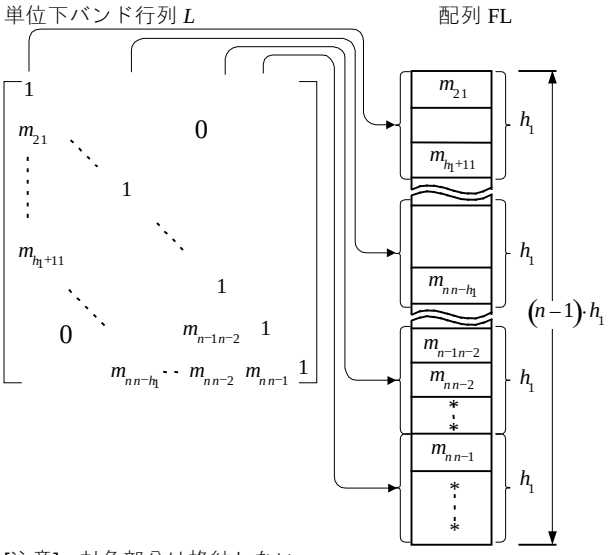
NH1 …… **入力**. LU 分解された元の行列の下バンド幅 h_1.

NH2 …… **入力**. LU 分解された元の行列の上バンド幅 h_2.

FL …… **入力**. 行列 $\boldsymbol{L}$.
大きさ $(n-1)\cdot h_1$ の 1 次元配列.
図 BLUX1-2 参照.

IP …… **入力**. 部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさ n の 1 次元配列.

ICON …… **出力**. コンディションコード.
表 BLUX1-1 参照.



[注意] 対角部分は格納しない.
*...*印要素は任意の数値である.

図 BLUX1-2 配列 FL における $\boldsymbol{L}$ の各要素の格納方法

表 BLUX1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列が非正則であった.	処理を打ち切る.
30000	N≤NH1, N≤NH2, NH1<0, NH2<0 又は IP に誤りがあった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .... MGSSL
② FORTRAN 基本関数 ... MIN0

b. 注意
① 連立 1 次方程式を解く場合, サブルーチン BLU1 を呼び出してから, 本サブルーチンを呼び出せば方程式を解くことができる. このとき, 本サブルーチンの入力パラメタ（定数ベクトルは除く）は, サブルーチン BLU1 の出力パラメタをそのまま指定すること.
しかし, 通常はサブルーチン LBX1 を呼び出せば, 一度に解が求められる.

② 本サブルーチンでは，バンド行列の特性を生かして，データ格納領域を節約出来るように，バンド行列を格納しているが，バンド幅の大きさによっては，密行列の連立 1 次方程式を解くサブルーチン LUX より (作業領域を含めて) データ格納領域を多く必要とする場合がある．その時は，サブルーチン LUX を用いた方がデータ格納領域を節約できる．
本サブルーチンの特性を生かせるのは，次数 n の係数行列の上バンド幅と下バンド幅が等しいなら, 各々のバンド幅が約 $n/3$ 以下の場合である．

c. 使用例
$n \times n$，下バンド幅 h_1，上バンド幅 h_2 の行列を入力し，L U 分解する．続けて，この行列を係数に持つ l 組の連立 1 次方程式を解く．
$n \leq 100$, $h_1 \leq 20$, $h_2 \leq 20$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(4100),B(100),FL(1980),
     *          IP(100),VW(100)
      CHARACTER*4 NT1(6),NT2(6),NT3(6)
      DATA NT1/'CO  ','EF  ','FI  ','CI  '
     *        ,'EN  ','T   '/,
     *     NT2/'CO  ','NS  ','TA  ','NT  '
     *        ,2*'    '/,
     *     NT3/'SO  ','LU  ','TI  ','ON  '
     *        ,2*'    '/
      READ(5,500) N,NH1,NH2,L
      NT=N*MIN0(NH1+NH2+1,N)
      READ(5,510) (A(I),I=1,NT)
      M=1
      WRITE(6,600) N,NH1,NH2
      CALL PBM(NT1,6,A,N,NH1,NH2)
      CALL BLU1(A,N,NH1,NH2,1.0E-6,
     *          IS,FL,IP,VW,ICON)
      WRITE(6,610) ICON
      IF(ICON.EQ.0) GO TO 10
      WRITE(6,620)
      STOP
   10 READ(5,510) (B(I),I=1,N)
      CALL PGM(NT2,6,B,N,N,1)
      CALL BLUX1(B,A,N,NH1,NH2,
     *           FL,IP,ICON)
      CALL PGM(NT3,6,B,N,N,1)
      M=M+1
      IF(L.GT.M) GO TO 10
      WRITE(6,630)
      STOP
  500 FORMAT(4I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'///5X,
     * 'LINEAR EQUATIONS  AX=B'
     * /5X,'ORDER=',I4
     * /5X,'SUB-DIAGONAL LINES=',I4
     * /5X,'SUPER-DIAGONAL LINES=',I4)
  610 FORMAT(' ',4X,'ICON=',I5)
  620 FORMAT(' '/5X,'** ABNORMAL END **')
  630 FORMAT(' '/5X,'** NORMAL END **')
      END
```

本使用例中のサブルーチン PBM 及び PGM は，バンド行列及び実行列を印刷するものである．プログラムは，サブルーチン LBX1 及び MGSM の各使用例に記載されている．

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{LUx}=\boldsymbol{b} \tag{4.1}$$

を解くことは,

$$\boldsymbol{Ly}=\boldsymbol{b} \tag{4.2}$$
$$\boldsymbol{Ux}=\boldsymbol{y} \tag{4.3}$$

を解くことに帰着される．

ここで，$\boldsymbol{L}$ は下バンド幅 h_1 の $n \times n$ 単位下バンド行列，$\boldsymbol{U}$ は上バンド幅 h (=min(h_1+h_2+1,n)) の $n \times n$ 上バンド行列である．

本サブルーチンでは，行列 $\boldsymbol{L}$ 及び $\boldsymbol{U}$ は，ガウス消去法により三角化された行列であることを前提としている．特に，$\boldsymbol{L}$ は (4.4) で表される．

$$\boldsymbol{L}=(\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\ldots\boldsymbol{M}_1\boldsymbol{P}_1)^{-1} \tag{4.4}$$

ここで，$\boldsymbol{M}_k$ はガウス消去法に置ける第 k 段階目の消去過程で, 第 k 列の対角要素以下を消去するための行列であり, $\boldsymbol{P}_k$ は第 k 列のピボット行を選択するための置換行列である (詳細は，サブルーチン BLU1 の手法概要参照）．

a. $\boldsymbol{Ly}=\boldsymbol{b}$ を解く (前進代入)
(4.4) より (4.2) は (4.5) となる．

$$\boldsymbol{y}=\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\ldots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{b} \tag{4.5}$$

(4.5) は，次のように逐次求める．

$$\boldsymbol{b}^{(1)}=\boldsymbol{b}$$
$$\boldsymbol{b}^{(2)}=\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{b}^{(1)}$$
$$\boldsymbol{b}^{(3)}=\boldsymbol{M}_2\boldsymbol{P}_2\boldsymbol{b}^{(2)}$$
$$\cdots$$
$$\boldsymbol{b}^{(n)}=\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\boldsymbol{b}^{(n-1)}$$
$$\boldsymbol{y}=\boldsymbol{b}^{(n)}$$

ここで，$\boldsymbol{M}_k$ は次のような行列である．

$$M_k = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & 0 & & \\ & & \cdots & & & & \\ & & & 1 & & & \\ & & & -m_{k+1k} & & & \\ & 0 & & -m_{k+2k} & 1 & & \\ & & & \cdots & 0 & \cdots & \\ & & & -m_{nk} & & & 1 \end{bmatrix}$$

本サブルーチンでは，次の手順により $\boldsymbol{b}^{(k+1)}$ (k=0,1,...,$n-1$) を逐次求める.

・ ガウス消去法の第 k 段階目の消去過程で行った行の入換えに対応して, 定数ベクトル $\boldsymbol{b}^{(k)}$ の各要素を入れ換える.

$$\tilde{\boldsymbol{b}}^{(k)} = \boldsymbol{P}_k \boldsymbol{b}^{(k)}$$

・ $b_i^{(k+1)} = \tilde{b}_i^{(k)} - m_{ik}\tilde{b}_k^{(k)}, i = k+1,...,\min(k+h_1, n)$
ただし $\tilde{b}^{(k)} = (\tilde{b}_1^{(k)}, \tilde{b}_2^{(k)}, ..., \tilde{b}_n^{(k)})^T$, $L = (m_{ij})$,
$y = (y_1, y_2, ..., y_n)^T$ である.

b. $\boldsymbol{U}x$=y を解く (後退代入)

$$x_k = y_k - \sum_{i=k+1}^{\min(n,k+h)} u_{ki} x_i \quad , k = n, n-1, ..., 1 \qquad (4.6)$$

なる式で逐次求める.
ただし，$\boldsymbol{U} = (u_{ij})$, $x = (x_1, x_2, \cdots, x_n)^T$ である.
なお，(4.6) の積和計算は精度を上げて行うことにより，丸め誤差の影響を少なくしている.

A52-11-0202 BLU1, DBLU1

実バンド行列の LU 分解 (ガウス消去法)
CALL BLU1(A,N,NH1,NH2,EPSZ,IS,FL,IP,VW, ICON)

(1) 機能

$n \times n$，下バンド幅 h_1，上バンド幅 h_2 のバンド行列 $\boldsymbol{A}$ をガウス消去法により LU 分解する．

$$\boldsymbol{A}=\boldsymbol{LU} \qquad (1.1)$$

ただし，$\boldsymbol{L}$ は単位下バンド行列，$\boldsymbol{U}$ は上バンド行列である．

$n>h_1 \geq 0$，$n>h_2 \geq 0$ であること．

(2) パラメタ

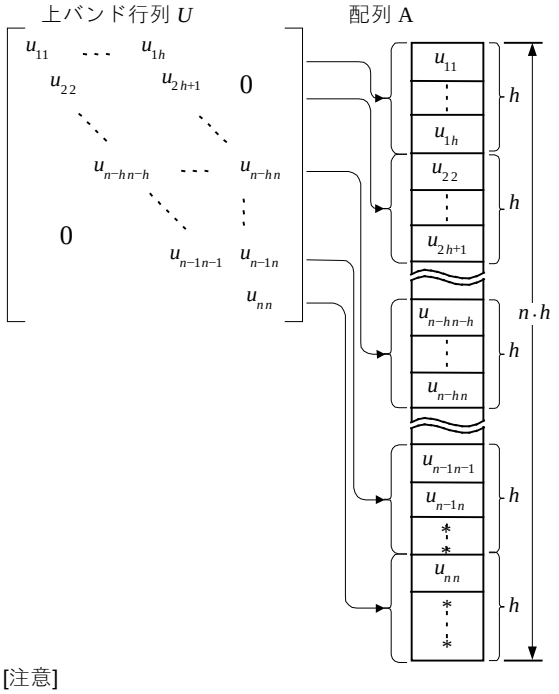
A ····· **入力**．行列 $\boldsymbol{A}$．
バンド行列用圧縮モード．
出力．行列 $\boldsymbol{U}$．
図 BLU1-1 参照．
大きさ $n \cdot \min(h_1+h_2+1, n)$ の 1 次元配列．

N ····· **入力**．行列 $\boldsymbol{A}$ の次数 n．

NH1 ····· **入力**．行列 $\boldsymbol{A}$ の下バンド幅 h_1．

NH2 ····· **入力**．行列 $\boldsymbol{A}$ の上バンド幅 h_2．

EPSZ····· **入力**．ピボットの相対零判定値 (≥0.0)．
0.0 のときは標準値が採用される．
（使用上の注意①参照）．



[注意]
*...*印要素は保障されない．
$h=\min(h_1+h_2+1, n)$.

図 BLU1-1 配列 A における $\boldsymbol{U}$ の各要素の格納方法

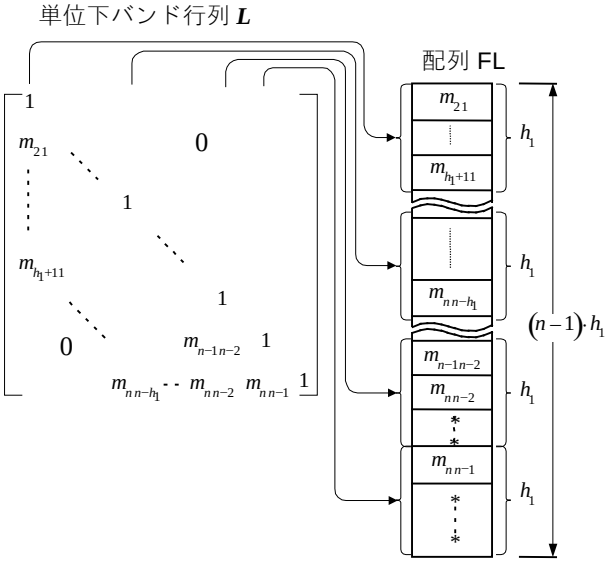
IS ····· **出力**. 行列 $\boldsymbol{A}$ の行列式を求めるための情報．
（使用上の注意②参照）．

FL ····· **出力**．行列 $\boldsymbol{L}$．
図 BLU1-2 参照．
大きさ $(n-1) \cdot h_1$ の 1 次元配列．

IP ····· **出力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．
大きさ n の 1 次元配列．
（使用上の注意③参照）．

VW ····· **作業領域**．
大きさ n の 1 次元配列．

ICON ····· **出力**．コンディションコード．
表 BLU1-1 参照．



[注意]
対角部分は格納されない．
*...*印要素は保障されない．

図 BLU1-2 配列 FL における L の各要素の格納方法

表 BLU1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	ピボットが相対的に零となった．行列は非正則の可能性が強い．	処理を打ち切る．
30000	N≤NH1, N≤NH2, NH1<0, NH2<0 又は, EPSZ<0.0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .... AMACH, MGSSL
② FORTRAN 基本関数 .... ABS, MIN0

b. 注意

① 本サブルーチンでは，ピボットの相対零判定法として、ガウス消去法による LU 分解の過程で，ピボットの値が(行列の絶対値最大要素の値) *EPSZ より小さい場合に，そのピボットを相対的に零と見なし, ICON=20000 として処理を打ち切る.
EPSZの標準値は丸め誤差の単位を u としたとき, EPSZ =16$\cdot u$ である.
なお，ピボットが小さくなっても計算を続行させる場合には, EPSZ へ極小の値 (例えば 10^{-70}) を与えればよいが，その結果は保証されない.

② 行列 $\boldsymbol{U}$ の要素は配列 A 上に，図 BLU1-1 で示すとおり格納される．よって，行列の行列式は, n 個の対角要素，すなわち，配列要素 A($i\cdot h$+1), i=0,1,...,n − 1 の積と IS の値を掛け合わせて得られる．ただし，h=min(h_1+h_2+1,n).

③ 本サブルーチンでは，部分ピボッティングに伴い，配列 A の内容を実際に交換している．すなわち，分解の J 段階目 (J=1,2,...,n − 1) において第 I 行 (I ≥ J) がピボット行として選択された場合には, 配列 $\boldsymbol{A}$ の第 I 行と第 J 行の内容が交換される．そして，その履歴を示すために IP(J) に I が格納される.

④ 本サブルーチンに続けて, サブルーチンBLUX1を呼び出すことにより連立1次方程式を解くことが出来る．しかし，通常はサブルーチンLBX1を呼び出せば，一度に解が求められる.

⑤ 本サブルーチンでは，バンド行列の特性を生かして，データ格納領域を節約出来るように，バンド行列を格納しているが，バンド幅の大きさによっては, 密行列を分解するサブルーチン ALU より (作業領域を含めて) データ格納領域を多く必要とする場合がある．その時は，サブルーチン ALU を用いた方がデータ格納領域を節約できる.
本サブルーチンの特性を生かせるのは，次数 n の行列の上バンド幅と下バンド幅が等しいなら，各々のバンド幅が約 n/3 以下の場合である.

c. 使用例
$n\times n$，下バンド幅 h_1，上バンド幅 h_2 の行列を入力し，LU 分解する.
n≤100, h_1≤20 and h_2≤20 の場合.

```
C     **EXAMPLE**
      DIMENSION A(4100),FL(1980),IP(100),
     *          VW(100)
      CHARACTER*4 NT1(6),NT2(15),NT3(10)
      DATA NT1/'MA  ','TR  ','IX  ',
     *         3*'    '/,
     *     NT2/'LU  ','-D  ','EC  ','OM  '
     *        ,'PO  ','SE  ','D   ','MA  '
     *        ,'TR  ','IX  ',5*'    '/,
     *     NT3/'TR  ','AN  ','SP  ','OS  '
     *        ,'IT  ','IO  ','N   ','VE  '
     *        ,'CT  ','OR  '/
      READ(5,500) N,NH1,NH2
      NT0=MIN0(NH1+NH2+1,N)
      NT=N*NT0
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,600) N,NH1,NH2
      CALL PBM(NT1,6,A,N,NH1,NH2)
      CALL BLU1(A,N,NH1,NH2,1.0E-6,
     *          IS,FL,IP,VW,ICON)
      WRITE(6,610) ICON
      IF(ICON.EQ.0) GO TO 10
      WRITE(6,620)
      STOP
   10 CALL PBM(NT2,15,A,N,NH1,NH2)
      CALL PGM(NT3,10,IP,N,N,1)
      DET=IS
      DO 20 I=1,NT,NT0
      DET=DET*A(I)
   20 CONTINUE
      WRITE(6,630) DET
      WRITE(6,640)
      STOP
  500 FORMAT(3I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'
     * ///5X,'DETERMINANT COMPUTATION'
     * /5X,'ORDER=',I4
     * /5X,'SUB-DIAGONAL LINES=',I4
     * /5X,'SUPER-DIAGONAL LINES=',I4)
  610 FORMAT(' ',4X,'ICON=',I5)
  620 FORMAT(' '/5X,'** ABNORMAL END **')
  630 FORMAT(' ',4X,'DET=',E16.8)
  640 FORMAT(' '/5X,'** NORMAL END **')
      END
```

本使用例中のサブルーチン PBM 及び PGM は，バンド行列及び実行列を印刷するものである.プログラムは，サブルーチン LBX1 及び MGSM の各使用例に記載されている.

(4) 手法概要

a. ガウス消去法
行列 $\boldsymbol{A}$ は，通常，部分ピボッティングを行って，単位下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ の積に分解することができる.

$$\boldsymbol{A}=\boldsymbol{LU} \tag{4.1}$$

先づ，ガウス消去法の第 k 段階目 (k=1, 2, …, n-1) の行列を $\boldsymbol{A}^{(k)}$と表す．ただし，$\boldsymbol{A}^{(1)}=\boldsymbol{A}$ とする.
消去の第 1 段階目は，第 1 列のうち，対角要素以下を消去して，$\boldsymbol{A}^{(2)}$を求めるが，この過程を行列表現すれば

$$\boldsymbol{A}^{(2)}=\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A}^{(1)} \tag{4.2}$$

となる．ここで，$\boldsymbol{P}_1$ は第 1 列のピボット行を選択するための置換行列であり，$\boldsymbol{M}_1$ は置換された行列の第 1 列の対角要素以下を消去するための行列である.
同様に，第 k 段階目の消去過程は，(4.3) で表現される.

$$\boldsymbol{A}^{(k+1)}=\boldsymbol{M}_k\boldsymbol{P}_k\boldsymbol{A}^{(k)} \tag{4.3}$$

ここで,

$$\boldsymbol{M}_k = \begin{bmatrix} 1 & & & & & \\ & 1 & & 0 & & \\ & & \ddots & & & \\ & & & 1 & & \\ & & & -m_{k+1k} & 1 & \\ & 0 & & -m_{k+2k} & & \\ & & & \vdots & 0 & \cdots \\ & & & -m_{nk} & & 1 \end{bmatrix} \tag{4.4}$$

$$m_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}, \qquad \boldsymbol{A}^{(k)} = \left(a_{ij}^{(k)}\right)$$

最終段階まで進めば,

$$\begin{aligned} \boldsymbol{A}^{(n)} &= \boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\boldsymbol{A}^{(n-1)} \\ &= \boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A}^{(1)} \end{aligned} \tag{4.5}$$

となり，行列 $\boldsymbol{A}$ ($=\boldsymbol{A}^{(1)}$) は，上三角行列 $\boldsymbol{A}^{(n)}$へ変換される．ここで,

$$\boldsymbol{U} = \boldsymbol{A}^{(n)} \tag{4.6}$$

$$\boldsymbol{L} = (\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{P}_1)^{-1} \tag{4.7}$$

とおくと，(4.5) は,

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U} \tag{4.8}$$

と表せる.
ところで,

$$\boldsymbol{M}_k^{-1} = \begin{bmatrix} 1 & & & & & \\ & 1 & & 0 & & \\ & & \cdots & & & \\ & & & 1 & & \\ & & & m_{k+1k} & 1 & \\ & 0 & & m_{k+2k} & & \\ & & & \cdots & 0 & \cdots \\ & & & m_{nk} & & 1 \end{bmatrix} \tag{4.9}$$

であるから, (4.7) で表せる行列 $\boldsymbol{L}$ は下三角行列であることが分かる.
よって，行列 $\boldsymbol{A}$ は単位下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ の積に分解される.

b. 本サブルーチンにおける手順
本サブルーチンでは, 第 k 段階目 (k=1,...,$n-1$) の消去過程において，$\boldsymbol{L}$ の k 番目の列，$\boldsymbol{U}$ の k 番目の行 (k=1, 2, ...,n) の各要素を (4.10) 及び(4.11) により求めている.

$$m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, \quad i = k+1, k+2, .., \ \min(k+h_1, n) \tag{4.10}$$

$$u_{kj} = a_{ik}^{(k)}, \quad i = k, k+1, .., \ \min(k+h_1+h_2, n) \tag{4.11}$$

一方，第 (k+1) 段階目の消去の対象となる行列 $\boldsymbol{A}^{(k+1)}$ は (4.12) のとおりになる.

$$\begin{aligned} &a_{kl}^{(k+1)} = a_{kl}^{(k)} - m_{kj}u_{jl} \\ &k = j+1, j+2, ..., \min(j+h_1, n) \\ &l = j+1, j+2, ..., \min(j+h_1+h_2, n) \end{aligned} \tag{4.12}$$

ただし， $\boldsymbol{A}^{(k)}=(a_{ij}^{(k)})$, $\boldsymbol{L}=(m_{ij})$, $\boldsymbol{U}=(u_{ij})$ である.

なお，この消去過程に先だって，計算誤差を少なくするために，(4.10) におけるピボット要素 $a_{kk}^{(k)}$ を次のように選択する．すなわち，スケーリングファクター$\boldsymbol{V}_l$を考慮した (4.13) の中より最大となるような値を選び, その時の要素$a_{lk}^{(k)}$をピボット要素とする.

$$V_l \left| a_{lk}^{(k)} \right|, l = k, k+1, ..., \min(k+h_1, n) \tag{4.13}$$

ただし，$\boldsymbol{V}_l$ は行列 $\boldsymbol{A}$ の第 l 行絶対値最大要素の逆数である.
そして，第 l 行と第 k 行を入れ換える.
選択されたピボット要素 $a_{kk}^{(k)}$ が

$$\left| a_{kk}^{(k)} \right| \le \max \left| a_{ij} \right| \cdot 16u$$

ここで，$\boldsymbol{A}=(a_{ij})$，$\boldsymbol{u}$ は丸め誤差の単位である.
であるならば，行列 $\boldsymbol{A}^{(1)}$ は数値的に非正則であると見なし, ICON=20000 として処理を打ち切る.

なお，参考文献として [1], [3], [4] 及び [8] を参照すること.

A52-21-0302 BMDMX, DBMDMX

MDM^{T}分解された実対称バンド行列の連立 1 次方程式
CALL BMDMX(B,FA,N,NH,MH,IP,IVW,ICON)

(1) 機能

MDM^{T}分解された実対称バンド行列を係数に持つ連立 1 次方程式

$$\boldsymbol{P}^{-1}\boldsymbol{M}\boldsymbol{D}\boldsymbol{M}^{T}\boldsymbol{P}^{-T}\boldsymbol{x}=\boldsymbol{b} \qquad (1.1)$$

を解く．ただし，$\boldsymbol{M}$ は $n\times n$，バンド幅 $\tilde{h}$ の単位下バンド行列, $\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列，$\boldsymbol{P}$ は置換行列 (係数行列を MDM^{T} 分解するときのピボッティングによる行の入換えを行う), $\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．ここで，$d_{k+1,k}\neq 0$ なら $m_{k+1,k}=0$ である．ただし，$\boldsymbol{M}=(m_{ij})$, $\boldsymbol{D}=(d_{ij})$.

$n>\tilde{h}\geq 0$ であること.

(2) パラメタ

B …… **入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.

FA …… **入力**．行列 $\boldsymbol{M}$ と行列 $\boldsymbol{D}$.
$\boldsymbol{M}$ をバンド幅 h_m の行列とみなし，$\boldsymbol{M}$ と $\boldsymbol{D}$ を対称バンド行列用圧縮モードで与える．図 BMDMX-1 参照.
大きさ $n(h_m+1)-h_m(h_m+1)/2$ の 1 次元配列.

N …… **入力**．行列 $\boldsymbol{M}$，行列 $\boldsymbol{D}$，定数ベクトル $\boldsymbol{b}$ 及び解ベクトル $\boldsymbol{x}$ の次数 n.

NH …… **入力**．行列 $\boldsymbol{M}$ のバンド幅 $\tilde{h}$.
（使用上の注意②参照）

MH …… **入力**．行列 $\boldsymbol{M}$ の最大許容バンド幅 h_m (N>MH≥NH)．（使用上の注意②参照）

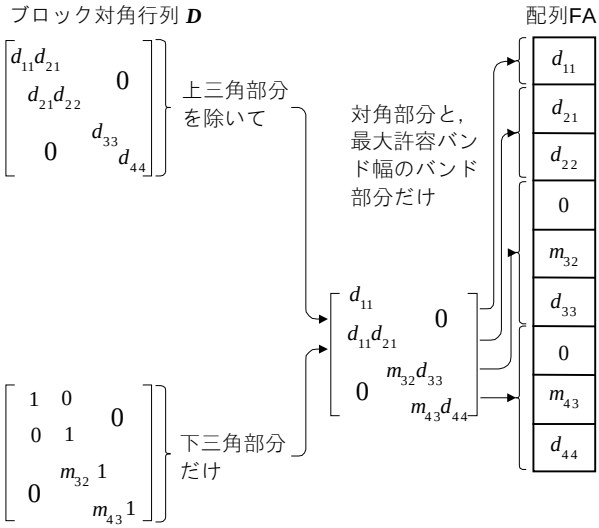
IP …… **入力**．ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．大きさ n の 1 次元配列．（使用上の注意②参照）

IVW …… **作業領域**．大きさ n の 1 次元配列.

ICON …… **出力**．コンディションコード.
（表 BMDMX-1 参照）

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... MIN0



[注意] $\boldsymbol{D}$ の各ブロック次数は 2，1 及び 1，$\boldsymbol{M}$ のバンド幅は 1，最大許容バンド幅は 2 の場合である.

図 BMDMX-1 行列 $\boldsymbol{M}$ 及び $\boldsymbol{D}$ の格納方法

表 BMDMX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列が非正則であった.	処理を打ち切る.
30000	NH<0, NH>MH, N≥MH 又は IP に誤りがあった.	処理を打ち切る.

b. 注意

① 連立 1 次方程式を解く場合，サブルーチン SBMDM を呼び出して，係数行列を MDM^{T}分解させてから，本サブルーチンを呼び出せば方程式を解くことができる. しかし, 通常はサブルーチン LSBIX を呼び出せば，一度に解が求められる.

② 本サブルーチンの入力パラメタ FA, NH, IP, MH は, サブルーチン SBMDM の出力パラメタ A, NH, IP 及び入力パラメタ MH と同じである.

c. 使用例

$n\times n$, バンド幅 h の実対称バンド行列をサブルーチン SBMDM で分解させてから，連立 1 次方程式を解く.
$n\geq 100$, $\tilde{h}\leq h_m\leq 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(3825),B(100),IP(100),
     *          IVW(100)
      READ(5,500) N,NH,MH
      WRITE(6,600) N,NH,MH
      NT=(N+N-MH)*(MH+1)/2
      READ(5,510) (A(J),J=1,NT)
      EPSZ=0.0
```

```
      CALL SBMDM(A,N,NH,MH,EPSZ,IP,IVW,
     *           ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      READ(5,510) (B(I),I=1,N)
      CALL BMDMX(B,A,N,NH,MH,IP,IVW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,630) (B(I),I=1,N)
      STOP
  500 FORMAT(3I4)
  510 FORMAT(4E15.7)
  600 FORMAT('1'/10X,'N=',I3,5X,'NH=',I3,
     * 5X,'MH=',I3)
  610 FORMAT(' ',5X,'ICON OF SBMDM=',I6)
  620 FORMAT(' ',5X,'ICON OF BMDMX=',I6)
  630 FORMAT(11X,'SOLUTION VECTOR'
     * /(15X,5E15.6))
      END
```

(4) 手法概要

MDM$^{\mathrm{T}}$ 分解された実対称バンド行列を係数に持つ連立 1 次方程式

$$\boldsymbol{P}^{-1}\boldsymbol{M}\boldsymbol{D}\boldsymbol{M}^{\mathrm{T}}(\boldsymbol{P}^{\mathrm{T}})^{-1}\boldsymbol{x}=\boldsymbol{b} \tag{4.1}$$

を解くことは，次の四つの方程式を解くことに帰着される．

$$\boldsymbol{M}\boldsymbol{x}^{(1)}=\boldsymbol{P}\boldsymbol{b} \tag{4.2}$$
$$\boldsymbol{D}\boldsymbol{x}^{(2)}=\boldsymbol{x}^{(1)} \tag{4.3}$$
$$\boldsymbol{M}^{\mathrm{T}}\boldsymbol{x}^{(3)}=\boldsymbol{x}^{(2)} \tag{4.4}$$
$$(\boldsymbol{P}^{\mathrm{T}})^{-1}\boldsymbol{x}=\boldsymbol{x}^{(3)} \tag{4.5}$$

ここで，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列，$\boldsymbol{b}$ は定数ベクトル，$\boldsymbol{x}$ は解ベクトルである．本サブルーチンは，$\boldsymbol{M}$ と $\boldsymbol{D}$ がブロック対角ピボッティング手法により分解された行列であることを前提としており，$\boldsymbol{P}$ はその際の置換行列である．（詳細はサブルーチン SBMDM の手法概要参照）．

a. $\boldsymbol{M}\boldsymbol{x}^{(1)}=\boldsymbol{P}\boldsymbol{b}$ を解く（後退代入）
1 × 1 ピボットのとき (行列 $\boldsymbol{D}$ のブロックの次数が 1 であるとき) は，(4.6) なる式で逐次求める．

$$x_i^{(1)}=b_i'-\sum_{k=1}^{i-1}m_{ik}x_k^{(1)} \quad ,i=1,...,n \tag{4.6}$$

しかし，i 番目が 2 × 2 ピボットのとき (行列 $\boldsymbol{D}$ のブロックの次数が 2 であるとき) は，$x_i^{(1)}$ に続けて $x_{i+1}^{(1)}$ を (4.7) で求め，i+2 番目へ進む．

$$x_{i+1}^{(1)}=b_{i+1}'-\sum_{k=1}^{i-1}m_{ik}x_k^{(1)} \tag{4.7}$$

ただし $M=\left(m_{ij}\right), x^{(1)^T}=\left(x_1^{(1)},...,x_n^{(1)}\right),$
$(\boldsymbol{P}\boldsymbol{b})^{\mathrm{T}}=(\boldsymbol{b}_1' ,..., \boldsymbol{b}_n')$ である．

b. $\boldsymbol{D}\boldsymbol{x}^{(2)}=\boldsymbol{x}^{(1)}$を解く
1 × 1 ピボットの時は, (4.8) なる式で逐次求める．

$$x_i^{(2)}=x_i^{(1)}/d_{ii}, \quad i=1,...,n \tag{4.8}$$

しかし，i 番目が 2 × 2 ピボットのときは，$x_i^{(2)}$ と $x_{i+1}^{(2)}$ を (4.9) なる式で求め，i+2 番目へ進む．

$$x_i^{(2)}=\left(x_i^{(1)}\cdot d_{i+1,i+1}/d_{i+1,i}-x_{i+1}^{(1)}\right)/d$$
$$x_{i+1}^{(2)}=\left(x_{i+1}^{(1)}\cdot d_{ii}/d_{i+1,i}-x_i^{(1)}\right)/d$$
ここで，$d=d_{i+1,i+1}\cdot\left(d_{ii}/d_{i+1,i}\right)-d_{i+1,i}$ (4.9)

ただし，$\boldsymbol{D}=\left(d_{ij}\right), \boldsymbol{x}^{(2)^T}=\left(\boldsymbol{x}_1^{(2)},...,\boldsymbol{x}_n^{(2)}\right)$ である．

c. $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{x}^{(3)}=\boldsymbol{x}^{(2)}$を解く（前進代入）
1 × 1 ピボットのときは，(4.10) なる式で逐次求める．

$$x_i^{(3)}=x_i^{(2)}-\sum_{k=i+1}^{n}m_{ki}x_k^{(3)} \quad , i=n,...,1 \tag{4.10}$$

しかし，i 番目が 2 × 2 ピボットのときは，$x_i^{(3)}$ に続けて $x_{i-1}^{(3)}$ を (4.11) で求め，i-2 番目へ進む．

$$x_{i-1}^{(3)}=x_{i-1}^{(2)}-\sum_{k=i+1}^{n}m_{k,i-1}x_k^{(3)} \tag{4.11}$$

ただし，$x^{(3)^T}=\left(x_1^{(3)},...,x_n^{(3)}\right)$ である．

d. $(\boldsymbol{P}^{\mathrm{T}})^{-1}\boldsymbol{x}=\boldsymbol{x}^{(3)}$を解く
ベクトル$\boldsymbol{x}^{(3)}$に対して置換行列を掛け合わせて解ベクトル $\boldsymbol{x}$ の要素 $\boldsymbol{x}_i$ を求める．しかし，実際には，トランスポジションベクトル IP の値を参照して, ベクトル$\boldsymbol{x}^{(3)}$の要素の入換えだけで求める．

上の説明は簡単のためにバンド構造を省略して行ったが，実際の処理はバンド構造を利用して能率的に進められる．

E32-32-0202 BSCD2, DBSCD2

B-spline 2 次元平滑化式（節点追加方式）
CALL BSCD2 (X,NX,Y,NY,FXY,KF,SX,SY,M,XT, NXT,YT,NYT,NXL,NYL, RNOT, C, KC,RNOR,VW,IVW,ICON)

(1) 機能

xy 平面上の格子点 (x_i,y_j); i=1,2,...,n_x, j=1,2,...,n_y において，観測値 $f_{i,j}=f(x_i,y_j)$，観測誤差$\sigma_{i,j}=\sigma_{x_i}\cdot\sigma_{y_j}$，残差二乗和の許容値 δ_t^2，及び *x* 方向，*y* 方向の初期節点列 $\xi_1,\xi_2,...,\xi_{n_s}$；$\eta_1,\eta_2,...,\eta_{l_s}$ が与えられたとき，これら *x* 方向，*y* 方向に適当な節点を段階的に追加し観測値を最小二乗の意味で近似する場合，残差二乗和が許容値以下となるような双 *m* 次の B-spline 平滑化式を求める．すなわち，*x* 方向，*y* 方向の最終的な節点の個数を n_t, l_t, そのときの残差二乗和を$\delta^2_{n_t+l_t}$とするとき，

$$\delta^2_{n_t+l_t}=\sum_{j=1}^{n_y}\sum_{i=1}^{n_x}\frac{1}{\left(\sigma_{x_i}\cdot\sigma_{y_j}\right)^2}\left\{f_{i,j}-\bar{S}\left(x_i,y_j\right)\right\}^2\le\delta_t^2 \quad (1.1)$$

が満足されるような双 *m* 次の B-spline 平滑化式,

$$\bar{S}(x,y)=\sum_{\beta=-m+1}^{l_t-1}\sum_{\alpha=-m+1}^{n_t-1}C_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \quad (1.2)$$

の平滑化係数 $C_{\alpha,\beta}$ を求める．

なお，本サブルーチンは，係数 $C_{\alpha,\beta}$ のほか，*x* 方向の節点列 $\xi_1,\xi_2,...,\xi_{n_t}$，*y* 方向の節点列 $\eta_1,\eta_2,...,\eta_{l_t}$，節点追加の各段階における残差二乗和 (1.3)，及び統計量 (1.4), (1.5) を出力する．

$$\delta^2_{n_r+l_r}=\sum_{j=1}^{n_y}\sum_{i=1}^{n_x}\frac{1}{\left(\sigma_{x_i}\cdot\sigma_{y_j}\right)^2}\left\{f_{i,j}-\bar{S}\left(x_i,y_j\right)\right\}^2 \quad (1.3)$$

（ただし，$\bar{S}(x,y)$ は $\xi_1,\xi_2,...,\xi_{n_r}$ 及び $\eta_1,\eta_2,...,\eta_{l_r}$ をそれぞれ *x*, *y* 方向の節点列とする双 *m* 次の B-spline 平滑化式）．

$$\bar{\sigma}^2_{n_r+l_r}=\delta^2_{n_r+l_r}\Big/\left\{n_x\cdot n_y-\left(n_r+m-1\right)\left(l_r+m-1\right)\right\}, \quad (1.4)$$

$$AIC_r=n_x\cdot n_y\log\delta^2_{n_r+l_r}+2\left(n_r+m-1\right)\left(l_r+m-1\right), \quad (1.5)$$

$$n_r+l_r=n_s+l_s,\ n_s+l_s+1,\ \cdots,\ n_t+l_t$$

$\sigma x_i>0$, $\sigma y_j>0$, $m\ge1$, $n_s\ge2$, $l_s\ge2$，及び，初期節点列 ξ_i, i=1,2,...,n_s; η_j, j=1,2,...,l_s に対して，

$$\min_i(\xi_i)\le\min_i(x_i),\ \max_i(\xi_i)\ge\max_i(x_i)$$
$$\min_j(\eta_j)\le\min_j(y_j),\ \max_j(\eta_j)\ge\max_j(y_j)$$

であること．

(2) パラメタ

X …… **入力**．*x* 方向の離散点 x_i.
大きさ n_x の 1 次元配列.

NX …… **入力**．x_i の個数 n_x.

Y …… **入力**．*y* 方向の離散点 y_j.
大きさ n_y の 1 次元配列.

NY …… **入力**．y_j の個数 n_y.

FXY …… **入力**．観測値 $f_{i,j}$.
FXY (KF,NY) なる 2 次元配列.

KF …… **入力**．配列 FXY の整合寸法 (≥NX).

SX …… **入力**．*x* 方向の観測誤差 σx_i.
大きさ n_x の 1 次元配列.

SY …… **入力**．*y* 方向の観測誤差 σy_j.
大きさ n_y の 1 次元配列.

M …… **入力**．B-spline の次数 *m* （使用上の注意②参照）.

XT …… **入力**．*x* 方向の初期節点 ξ_i, i=1,2,...,n_s.
（使用上の注意③参照）.
出力．求められた平滑化式の *x* 方向の節点列 ξ_i, i=1,2,...,n_t. 結果は $\xi_1<\xi_2<...<\xi_{n_t}$ の順に並べられる.
大きさ NXL の 1 次元配列.

NXT …… **入力**．*x* 方向の初期節点の個数 n_s.
出力．求められた平滑化式の，*x* 方向の節点の個数 n_t.

YT …… **入力**．*y* 方向の 初期節点 η_j, j=1,2,...,l_s.
（使用上の注意③参照）.
出力. 求められた平滑化式の *y* 方向の節点 η_j, j=1,2,...,l_t. 結果は，$\eta_1<\eta_2<...<\eta_{l_t}$ の順に並べられる.
大きさ NYL の 1 次元配列.

NYT …… **入力**．*y* 方向の初期節点の個数 l_s.
出力．求められた平滑化式の，*y* 方向の節点の個数 l_t.

NXL …… **入力**．*x* 方向の節点の個数の上限 (≥n_s).
（使用上の注意④参照）.

NYL …… **入力**．*y* 方向の節点の個数の上限 (≥l_s).
（使用上の注意④参照）.

RNOT … **入力**．残差二乗和の許容値 δ_t^2.
標準値としては，$\delta_t^2=n_x\cdot n_y$ が適当.

C …… **出力**．平滑化係数 $C_{\alpha,\beta}$, $\alpha=-m+1,-m+2,...,n_t-1$, $\beta=-m+1,-m+2,...,l_t-1$，$C_{\alpha,\beta}$ は $C(\alpha+m,\beta+m)$ に格納される.
C(KC,NYL+M–1) なる 2 次元配列.

KC …… **入力**．配列 C の整合寸法 (≥NKL+M-1).

RNOR …… **出力**. 節点追加の各段階における (1.3), (1.4), (1.5) の値.
RNOR(3,KR) なる 2 次元配列. ただし,

KR=(NXL - n_s)+(NYL - l_s)+1.

$n_r+l_r=n_s+l_s$, n_s+l_s+1, ... n_t+l_t とするとき,
$\delta^2_{n_r+l_r}$ は RNOR (1,P_r) に
$\overline{\sigma}^2_{n_r+l_r}$ は RNOR (2,P_r) に
AIC_r は RNOR (3,P_r) に格納される.
ここで $P_r=(n_r-n_s)+(l_r-l_s)+1$.
（使用上の注意⑤参照）.

VW …… **作業領域**.
大きさ max (s_1, s_2)なる 1 次元配列.
ここで,

$$s_1=(n_x+n_y+2m)(m+1)+\max\{\max(n_x+m,n_y+m),\ 2+\min(n_x+m,n_y+m)(m+1)\}$$

$$s_2=\{\min(n_x,n_y)+3\}(m+1)+n_x+n_y+\mathrm{NXL}+\mathrm{NYL}+2$$

IVW …… **作業領域**.
大きさ $n_x+n_y+\max(\mathrm{NXL},\mathrm{NYL})\cdot m$ なる 1 次元配列.

ICON …… **出力**. コンディションコード.
表 BSCD2-1 参照.

表 BSCD2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	x 方向の節点の個数が，その上限に達しても収束判定 (1.1) が満たされなかった.	最後に得られた平滑化式を出力する.
11000	y 方向の節点の個数が，その上限に達しても収束判定 (1.1) が満たされなかった.	
30000	次のいずれかであった. ① $\sigma x_i \le 0$ のものがある. ② $\sigma y_j \le 0$ のものがある. ③ M<1 ④ XT(I)=XT(k) (I≠K) 又は YT(I)=YT(k) (I≠K) ⑤ n_s<2 又は l_s<2 ⑥ NXL<n_s 又は NYL<l_s ⑦ $\min_i(\xi_i)>\min_i(x_i)$ 又は $\max_i(\xi_i)<\max_i(x_i)$ ⑧ $\min_j(\eta_i)>\min_j(y_i)$ 又は $\max_j(\eta_j)<\max_j(y_j)$	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UPOB2, UPCA2, UREO1, UBAS0, UCDB2

② FORTRAN 基本関数 ... FLOAT, IFIX, ALOG, ABS

b. 注意

① 本サブルーチンに続けて，サブルーチン BSFD1 を呼び出すことにより, B-spline 平滑化式 (1.2) に基づく平滑値，偏微分値あるいは二重積分値を求めることができる. そのとき, パラメタ M, XT, YT, NXT, NYT, C の値はサブルーチン BSFD1 の入力となる.

② 次数 m (奇数でも偶数でもよい) は 3 程度が良く，5 を超えるのは良くない．これは、平滑化係数 $C_{\alpha,\beta}$ を求める際の正規方程式が m が大きくなるにつれ，悪条件になるからである.

③ x 方向，y 方向の初期節点列 ξ_i, η_j；i=1,2,...,n_s; j=1,2,...,l_s は，通常,

$$n_s=l_s=2$$
$$\xi_1=\min_i(x_i),\quad \xi_{n_s}=\max_i(x_i)$$
$$\eta_1=\min_j(y_j),\quad \eta_{l_s}=\max_j(y_j),$$

と与えればよい.

④ x 方向，y 方向の節点の個数の上限 NXL, NYL はそれぞれ，n_x/2, n_y/2 程度に与えるのがよい（節点の個数が増えると，正規方程式が悪条件となる）．なお，本サブルーチンは，節点の個数がその上限に達しても (1.1) が満たされないときは，x 方向，y 方向に応じて ICON=10000 又は 11000 として処理を打ち切る.

⑤ RNOR に出力される情報は，節点を追加してゆく過程での各種評価基準値の履歴である．これらの履歴は，得られた平滑化式の良悪しの確認に使える． すなわち，これら評価基準値は，通常，節点の追加と共に減少するが，その変化は徐々に緩やかになる．特に $\overline{\sigma}^2_{n_r+l_r}$ 及び AIC_r が殆んど変化しなくなったときは，平滑化式が良好になったことの現れである．したがって，利用者は，RNOR の内容を印刷することにより，得られた平滑化式の良好さを確めるのがよい．もし節点の個数が，x 方向，y 方向合わせて n_t+l_t よりも少ない段階で，良好な平滑化式が得られていたことが分かれば，その段階での $\delta^2_{n_r+l_r}$ を新たに許容値 δ^2_t として，再度，本サブルーチンを呼び出せば，その段階での平滑化式を求めることができる.

c. 使用例

格子点 (x_i,y_j): i=1, 2, ...,80, j=1, 2, ...,60, 観測値 f_{ij}, 及び, x 方向, y 方向の観測誤差 σx_i, σy_j を入力し，双 3 次の B-spline 平滑化式を求める．初期節点としては，x 方向，y 方向にそれぞれ(3.1), (3.2) のように与え，また，節点の個数の上限はそれぞれ 20, 15 とする．

$$\xi_1 = x_{min} = \min_i(x_i),\ \xi_2 = x_{max} = \max_i(x_i) \qquad (3.1)$$

$$\eta_1 = y_{min} = \min_j(y_j),\ \eta_2 = y_{max} = \max_j(y_j) \qquad (3.2)$$

続いて，VXr, VYs を (3.3), (3.4) で表すとき，サブルーチン BSFD1 により，(VXr, VYs) の各点における平滑値，及び，x 方向，y 方向の 1 階偏微分値を計算する．

$$VX_r = x_{min} + (x_{max} - x_{min})r/10 \qquad r = 0,1,..,10 \qquad (3.3)$$

$$VY_s = y_{min} + (y_{max} - y_{min})s/10 \qquad s = 0,1,..,10 \qquad (3.4)$$

```
C     **EXAMPLE**
      DIMENSION X(80),Y(60),FXY(80,60),
     *          SX(80),SY(60),XT(20),
     *          YT(15),C(22,17),
     *          RNOR(3,32),VW(670),
     *          IVW(200),RR(3)
      NX=80
      NY=60
      READ(5,500)(X(I),SX(I),I=1,NX),
     *           (Y(J),SY(J),J=1,NY)
      DO 10 I=1,NX
      DO 10 J=1,NY
      READ(5,510) FXY(I,J)
   10 CONTINUE
      M=3
      NXT=2
      NYT=2
      XMAX=X(1)
      XMIN=X(1)
      YMAX=Y(1)
      YMIN=Y(1)
      DO 20 I=2,NX
      IF(X(I).GT.XMAX) XMAX=X(I)
      IF(X(I).LT.XMIN) XMIN=X(I)
   20 CONTINUE
      DO 30 J=2,NY
      IF(Y(J).GT.YMAX) YMAX=Y(J)
      IF(Y(J).LT.YMIN) YMIN=Y(J)
   30 CONTINUE
      XT(1)=XMIN
      XT(2)=XMAX
      YT(1)=YMIN
      YT(2)=YMAX
      NXL=20
      NYL=15
      RNOT=FLOAT(NX*NY)
C
      CALL BSCD2(X,NX,Y,NY,FXY,80,SX,SY,
     *     M,XT,NXT,YT,NYT,NXL,NYL,
     *     RNOT,C,22,RNOR,VW,IVW,ICON)
      WRITE(6,600) ICON
      IF(ICON.EQ.30000) STOP
      WRITE(6,610)
      NT=NXT+NYT
      DO 40 I=4,NT
      IADD=I-3
      WRITE(6,620) I,(RNOR(J,IADD),J=1,3)
   40 CONTINUE
C
      HX=(XMAX-XMIN)/10.0
      HY=(YMAX-YMIN)/10.0
      DO 70 I=1,11
      VX=XMIN+HX*FLOAT(I-1)
      WRITE(6,630) VX
      WRITE(6,640)
      DO 60 J=1,11
      VY=YMIN+HY*FLOAT(J-1)
      DO 50 K=1,3
      KM1=K-1
      ISWX=MOD(KM1,2)
      ISWY=KM1/2
      CALL BSFD1(M,XT,NXT,YT,NYT,C,22,
     *     ISWX,VX,IX,ISWY,VY,IY,RR(K),
     *     VW,ICON)
   50 CONTINUE
      WRITE(6,650) VY,(RR(K),K=1,3)
   60 CONTINUE
   70 CONTINUE
      STOP
  500 FORMAT(2F10.0)
  510 FORMAT(F10.0)
  600 FORMAT(10X,'ICON=',I5//)
  610 FORMAT(8X,'NO. OF KNOTS',9X,
     * 'RNOR(1,*)',11X,'RNOR(2,*)',11X,
     * 'RNOR(3,*)'/)
  620 FORMAT(10X,I2,8X,3(5X,E15.8))
  630 FORMAT(//5X,'VX=',E15.8)
  640 FORMAT(16X,'VY',13X,
     * 'SMOOTHED VALUE',8X,'DS(X,Y)/DX',
     * 10X,'DS(X,Y)/DY')
  650 FORMAT(5X,4(5X,E15.8))
      END
```

(4) 手法概要

本サブルーチンは，節点を適応的に追加して残差二乗和が，与えられた許容値δ_t^2 以下となるような双 m 次の，B-spline 平滑化式を求める．

いま，x 方向の節点列 $\xi_1,\xi_2,...,\xi_{n_r}$，$y$ 方向の節点列 $\eta_1,\eta_2,...,\eta_{l_r}$ が既に決定されていて，

$$\xi_1 < \xi_2 < ... < \xi_{n_r}$$

$$\eta_1 < \eta_2 < ... < \eta_{l_r}$$

を満たしているとする. ただし, 初期の段階では, n_r=n_s, l_r=l_sである．このとき，これらを x 方向，y 方向の節点とする双 m 次の spline 関数

$$\bar{S}(x,y) = \sum_{\beta=-m+1}^{l_r-1}\sum_{\alpha=-m+1}^{n_r-1} C_{\alpha,\beta} N_{\alpha,m+1}(x) N_{\beta,m+1}(y) \qquad (4.1)$$

の係数 $C_{\alpha,\beta}$を，残差二乗和

$$\sum_{j=1}^{n_y}\sum_{i=1}^{n_x} \frac{1}{\sigma x_i^2 \cdot \sigma y_j^2} \left\{ f_{i,j} - \bar{S}(x_i, y_j) \right\}^2 \qquad (4.2)$$

が最小になるように決める．そのような $C_{\alpha,\beta}$は，(4.2) を $C_{\alpha,\beta}$ で偏微分して，それを零と置いて得られる連

立 1 次方程式（正規方程式）を解いて得られる．得られた $C_{\alpha,\beta}$ に対する (4.2) の値を$\delta^2_{n_r+l_r}$とする．

もし $\delta^2_{n_r+l_r} \le \delta^2_t$ ならば (4.1) を求めるべき双 m 次の B-spline 平滑化式とする．$\delta^2_{n_r+l_r} > \delta^2_t$ ならば，以下の手順により追加すべき新たな節点をきめる．まず，各点における残差$\varepsilon_{i,j}=f_{i,j} - \bar{S}(x_i, y_j)$ i=1,2,...n_x, j=1,2,...,n_y を計算する．次にこれを使って，

$$
\begin{aligned}
u_i &= \sum_{\{r|\xi_i \le x_r < \xi_{i+1}\}} \frac{1}{\sigma x_r^2 \cdot n_y} \left\{ \sum_{s=1}^{n_y} \frac{\varepsilon_{r,s}}{\sigma y_s} \right\}^2 \\
v_j &= \sum_{\{s|\eta_j \le y_s < \eta_{j+1}\}} \frac{1}{\sigma y_s^2 \cdot n_x} \left\{ \sum_{r=1}^{n_x} \frac{\varepsilon_{r,s}}{\sigma x_r} \right\}^2
\end{aligned}
\qquad (4.4)
$$

を計算する．$\{ u_i | 1 \le i \le n_r - 1\}$, $\{ v_j | 1 \le j \le l_r - 1\}$の最大値を u_i または v_j とする．u_i が最大のときは，

$\xi=(\xi_i+\xi_{i+1})/2$ を追加すべき新たな節点とし，n_r を 1 だけ増す．反対に，v_j が最大のときは，$\eta=(\eta_j+\eta_{j+1})/2$ を追加すべき新たな節点とし，l_r を 1 だけ増す．x 方向または y 方向の更新された節点列を，小さい順に並べ替えて，上記過程を繰り返す．

このように節点列を更新しながら次第に節点の個数を増加させ，残差二乗和が許容値δ^2_t 以下になるまで反復する．

B21-21-0502 BSCT1, DBSCT1

実対称 3 重対角行列の固有値 (バイセクション法)
CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)

(1) 機能

n 次の実対称 3 重対角行列 ***T*** の固有値をバイセクション法により，大きい方から，若しくは小さい方から *m* 個求める．ただし $1 \le m \le n$ であること．

(2) パラメタ

D…… **入力**．3 重対角行列 ***T*** の主対角要素．
大きさ *n* の 1 次元配列．

SD…… **入力**．3 重対角行列 ***T*** の副対角要素．
大きさ *n* の 1 次元配列．
ただし SD(2), …, SD(N) に格納しておくこと．

N…… **入力**．3 重対角行列 ***T*** の次数 *n*．

M …… **入力**．求めたい固有値の個数 *m*．
M=+*m* のときは大きい方から求める．
M=–*m* のときは小さい方から求める．

EPST …… **入力**．固有値の収束判定に使われる絶対誤差の上限（手法概要の (4.9) 式参照）．ただし，負の値を入れると標準値が設定される．

E …… **出力**．*m* 個の固有値．
M が正の場合大きい順に，M が負の場合小さい順に並ぶ．
大きさ *m* の 1 次元配列．

VW …… **作業領域**．大きさ *n*+2*m* の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 BSCT1-1 参照．

表 BSCT1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N=1 であった．	E(1)=D(1) とする．
30000	N<|M| 又は M=0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数 ... IABS,ABS, AMAX1

b. 注意

① 通常，実対称行列の固有値を求める場合は，サブルーチン TRID1 により 3 重対角化し，本サブルーチン又はサブルーチン TRQL で固有値を求めること．

② *n*/4 個以上の固有値を求める場合は，サブルーチン TRQL を用いた方が，一般的に処理速度が速い．

③ 零を固有値として持つことが考えられる場合，EPST は，手法概要中の〔**収束判定法と EPST の与え方**〕の項を参照して与えること．

c. 使用例

n 次の実対称行列を，サブルーチン TRID1 によって 3 重対角行列へ変換した後，*m* 個の固有値を計算する．
n≤100 の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),D(100),SD(100),
     *          E(100),VW(300)
   10 READ(5,500) N,M,EPST
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N,M
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+1
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL TRID1(A,N,D,SD,ICON)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)
      WRITE(6,630)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      MM=IABS(M)
      WRITE(6,640) (I,E(I),I=1,MM)
      GO TO 10
  500 FORMAT(2I5,E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'/
     *11X,'** ORDER =',I5,10X,'** M =',I3/)
  610 FORMAT('0',7X,I3,5E20.7/(11X,5E20.7))
  620 FORMAT(/11X,'** CONDITION CODE =',I5/)
  630 FORMAT('0'/11X,'** EIGENVALUES')
  640 FORMAT(5X,'E(',I3,')=',E15.7)
      END
```

(4) 手法概要

n 次の実対称 3 重対角行列を ***T*** とし，その固有値をバイセクション法により大きい方から，若しくは小さい方から *m* 個求める．

図 BSCT1-1 に示す実対称 3 重対角行列を ***T*** は，副対角要素に零があれば，そこでより小さな小行列に直和分解できる．***T*** が直和分解できるときは，各々の小行列の固有値を求めれば ***T*** の固有値を求めたことと等しくなる．したがって，各小行列ごとにバイセクション法を適用するので，以下の説明の ***T*** は，直和分解されていない，すなわち $b_i \ne 0$ としておく．

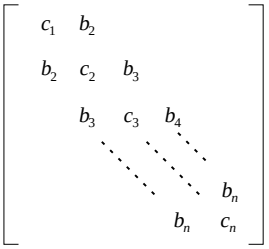


図 BSCT1-1 実対称 3 重対角行列 $\boldsymbol{T}$

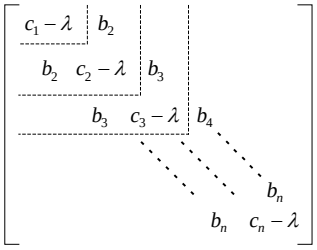


図 BSCT1-1 行列 [$\boldsymbol{T}$–λ$\boldsymbol{I}$]

図 BSCT1-2 に示す行列 ($\boldsymbol{T}$–λ$\boldsymbol{I}$) において

$$\det(\boldsymbol{T}-\lambda\boldsymbol{I})=0 \tag{4.1}$$

を満足するλが，$\boldsymbol{T}$ の固有値である．行列 ($\boldsymbol{T}$–λ$\boldsymbol{I}$) の左上からの主小行列式の値を $\boldsymbol{P}_i(\lambda)$ とすれば，(4.2) のような漸化関係が成り立つ．

$$\begin{aligned} &P_0(\lambda)=1, P_1(\lambda)=c_1-\lambda \\ &P_i(\lambda)=(c_i-\lambda)P_{i-1}(\lambda)-b_i^2P_{i-2}(\lambda), \\ &\qquad i=2,3,...,n \end{aligned} \tag{4.2}$$

(4.2) の多項式列 $P_0(\lambda), P_1(\lambda), ..., P_n(\lambda)$ はスツルム列をなす．したがって，$P_0(\lambda)$ から $P_n(\lambda)$ までの，隣り合う項の符号が一致する回数を $\boldsymbol{L}(\lambda)$ とすれば，この $\boldsymbol{L}(\lambda)$ は，λより大きい固有値の個数と一致する．

ただし，$P_i(\lambda)=0$ ならば，$P_{i-1}(\lambda)$ の符号を採用する．

実際は，(4.2) をそのまま用いずに，(4.3) に示す多項式列 $q_i(\lambda)$ に置き換える．

$$q_i(\lambda)=P_i(\lambda)/P_{i-1}(\lambda), i=1,2,...,n \tag{4.3}$$

そうすると $\boldsymbol{L}(\lambda)$ は，$q_i(\lambda)$ の多項式列が正又は零の値をとった時の回数となるので判定しやすい．

(4.2), (4.3) から，$q_i(\lambda)$ は次のようになる．

$$\left.\begin{aligned} &q_1(\lambda)=c_1-\lambda \\ &q_i(\lambda)=(c_i-\lambda)-b_i^2/q_{i-1}(\lambda), i=2,3,...,n \end{aligned}\right\} \tag{4.4}$$

もし $q_{i-1}(\lambda)=0$ となったならば,

$$q_i(\lambda)=(c_i-\lambda)-|b_i|/u \tag{4.5}$$

とする．ただし，u は丸め誤差の単位である．この方法により，オーバフロー，アンダフローの発生を防ぐことができ，$b_i=0$ であったとしても $\boldsymbol{L}(\lambda)$ は,正しく計算される．

大きい方から k 番目の固有値を求める場合を考えてみる．ただし，固有値は，(4.6) の関係を持っているものとする．

$$\lambda_1\geq\lambda_2\geq.....\geq\lambda_k\geq.....\geq\lambda_n \tag{4.6}$$

① ゲルシュゴリンの定理を用いて，n 個のすべての固有値を含む区間 $[l_0,r_0]$ を求める．

$$\begin{aligned} r_0&=\max_{1\leq i\leq n}\{c_i+(|b_i|+|b_{i+1}|)\} \\ l_0&=\min_{1\leq i\leq n}\{c_i-(|b_i|+|b_{i+1}|)\} \end{aligned} \tag{4.7}$$

ただし，$b_1=0, b_{n+1}=0$ とする．

② (4.8) の反復を行い λ_k が存在する区間 $[l_j,r_j]$ を十分縮めて，λ_k をその区間の中点で近似する．

$$\left.\begin{aligned} &j=0,1,2,..... \\ &h_j=(l_j+r_j)/2 \\ &L(h_j)<k \text{ ならば } l_{j+1}=l_j, r_{j+1}=h_j \\ &L(h_j)\geq k \text{ ならば } l_{j+1}=h_j, r_{j+1}=r_j \end{aligned}\right\} \tag{4.8}$$

(4.8) の 1 回の反復により，区間 $[l_j,r_j]$ は半分に縮小される．

本サブルーチンでは，①,②の操作によって，全体の行列の大きい方から m 番目までの固有値の存在範囲を求めておいて，その区間で小行列ごとに固有値を求め，求まった固有値の合計が m となったならば処理を終了する．

収束判定法とEPSTの与え方

本サブルーチンの収束判定は，(4.9) により行う．

$$r_j-l_j\leq 2u(|l_j|+|r_j|)+\text{EPST} \tag{4.9}$$

ただし，EPST は求める固有値の絶対誤差の上限として指定された値である．(4.9) を満足したならば，$(l_j+r_j)/2$ を固有値とする．もし EPST=0.0 として与えると，(4.9) は (4.10) となる．

$$r_j-l_j\leq 2u(|l_j|+|r_j|) \tag{4.10}$$

このとき，l_j と r_j の最下位の桁が，ほぼ一致するまで 2 分割して行くことになる．したがって時間もかかる．
このとき EPST には，必要な精度までの打ち切りを指定する値としての役割があり，また固有値が零となる場合に，(4.10) が満足されなくなるの防ぐ意味もある．
EPST が負で与えられると，小行列毎に標準値として

$$\mathrm{EPST} = u \cdot \max\left(|l_0|, |r_0|\right) \tag{4.11}$$

が採用される．ここで，l_0, r_0 は，ゲルシュゴリンの定理より求めた小行列ごとの固有値を含む範囲の下限と上限である．

なお, 詳細については, 参考文献 [12] の pp.299-302, [13] の pp.249-256 及び [15] を参照すること．

E32-31-0102　　BSC1, DBSC1

B-spline 平滑化式（固定節点）
CALL BSC1 (X, Y, W, N, M, XT, NT, C, R, RNOR, VW, IVW, ICON)

(1)　機能

離散点 $x_1, x_2, ..., x_n$ に対して $y_1, y_2, ..., y_n$，重み関数値 $w_i=w(x_i)$, i=1, 2, ..., n 及び spline 関数の節点 $\xi_1, \xi_2, ..., \xi_{n_t}$ が与えられたとき，これに対する最小二乗近似の意味の B-spline 平滑化式を求める．すなわち，求めるべき m (奇数でも偶数でもよい) 次の B-spline 平滑化式を，

$$\overline{S}(x)=\sum_{j=-m+1}^{n_t-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

とし，このとき重み付き残差の 2 乗和，

$$\delta_m^2=\sum_{i=1}^{n} w_i\left\{y_i-\overline{S}(x_i)\right\}^2 \qquad (1.2)$$

を最小にするような平滑化係数 c_j を求める．

ここで，節点列{ξ_i}の張る区間 $I_\xi=[\min_j(\xi_j),\max_j(\xi_j)]$ は，n 個の離散点のすべてを含む必要はない．例えば，図 BSC1-1 のように，I_ξ を離散点の張る区間 $I_x=[\min_i(x_i),\max_i(x_i)]$ の部分区間として指定することができる．この場合は，(1.1) の $\overline{S}(x)$ は I_ξ に含まれる離散点 (個数を n_e とする) だけを対象とする平滑化式となり，(1.2) の和をとる際も I_ξ に含まれる離散点だけが対象となる．

$w_i\geq 0$, $m\geq 1$, $n_t\geq 3$, $\xi_j\neq\xi_k$ ($j\neq k$), $n_e\geq n_t+m-1$ であること．

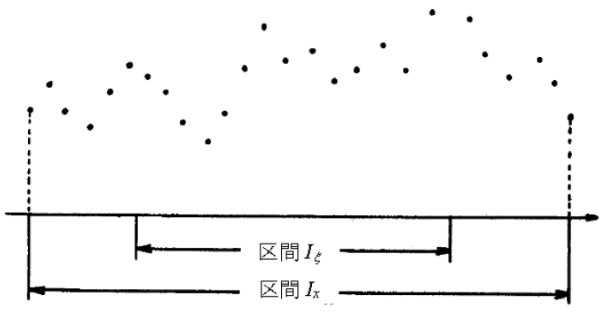


図 BSC1-1 平滑化式を求める区間 I_ξ の例

(2)　パラメタ

X ····· **入力**．離散点 x_i.
　　大きさ n の 1 次元配列.
Y ····· **入力**．観測値 y_i.
　　大きさ n の 1 次元配列.
W ····· **入力**．重み関数値.
　　大きさ n の 1 次元配列.
N ····· **入力**．離散点の個数 n.
M ····· **入力**．B-spline の次数 m.
　　（使用上の注意②参照）.
XT ····· **入力**．節点ξ_j（使用上の注意③参照）.
　　大きさ n_t の 1 次元配列.
　　出力. もし入力時に XT(1)<XT(2)<...<XT(n_t) が満たされていなかったときは，出力時にこのように並べ換える.
NT ····· **入力**．節点の個数 n_t.
C ····· **出力**．平滑化係数 c_j.
　　大きさ n_t+m-1 の 1 次元配列.
R ····· **出力**．残差 $y_i-\overline{S}(x_i)$, i=1,2,...,n.
　　大きさ n の 1 次元配列.
RNOR ····· **出力**．重み付き残差の 2 乗和δ_m^2.
VW ····· **作業領域**.
　　大きさ $(n_t+m)(m+1)$ の 1 次元配列.
IVW ····· **作業領域**.
　　大きさ n の整数型 1 次元配列.
ICON ····· **出力**．コンディションコード.
　　表 BSC1-1 参照.

表 BSC1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① w_i に負のものがある. ② M<1 ③ XT(I)=XT(K) (I≠K) ④ NT<3 ⑤ n_e<NT+M-1	処理を打ち切る.

(3)　使用上の注意

a.　使用する副プログラム
①　SSL II ... MGSSL, UREO1, UNCA1, UCDB1, UCAO1, UBAR1
②　FORTRAN 基本関数 ... DSQRT

b.　注意
①　本サブルーチンに続けて，サブルーチン BSF1 を呼び出すことにより，B-spline 平滑化式 (1.1) に基づく平滑値あるいは微分値あるいは積分値を求めることができる．そのとき，パラメタ M, XT, NT, C の値はサブルーチン BSF1 の入力となる．
②　次数 m は 3 程度が良く，5 を超えるのは良くない．これは，平滑化係数 c_j を求める際の正規方程式（手法概要参照）が，m が大きくなるにつれ，悪条件になるからである．

③ 節点 ξ_j の採り方（位置決め）は観測値の挙動に応じて決めることが大切である．一般的に言えば，観測値がピークになる点あるいは急激に変化する点には必ず節点を採るべきである.逆に滑らかに変化している領域には節点を採るべきではない（図 BSC1-2 参照）.

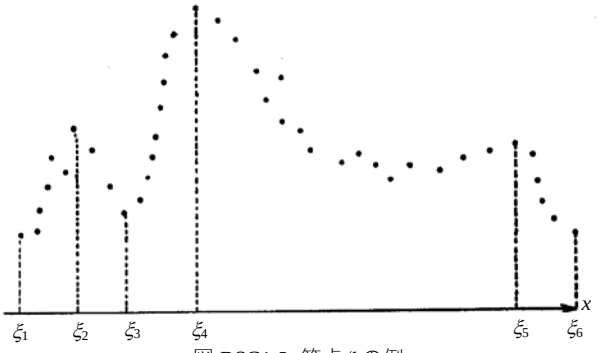


図 BSC1-2 節点ξ_iの例

c. 使用例
サブルーチン BSF1 の使用例参照.

(4) 手法概要

本サブルーチンは，m 次の B-spline 平滑化式を,

$$\bar{S}(x)=\sum_{j=-m+1}^{n_t-1} c_j N_{j,m+1}(x) \tag{4.1}$$

として，重み付き残差の 2 乗和,

$$\delta_m^2=\sum_{i=1}^{n} w_i\left\{y_i-\bar{S}(x_i)\right\}^2 \tag{4.2}$$

を最小にするような平滑化係数 c_j を求める．$\bar{S}(x)$ の定義域は，利用者の与える節点列$\{\xi_j\}$の張る区間 $I_\xi=[\min_j(\xi_j),\max_j(\xi_j)]$ である．以下，説明を簡単にするため，節点列$\{\xi_j\}$は，$\xi_j<\xi_{j+1}$ (j=1,2,...,n_t-1) を満たし，また n 個の離散点 x_i はすべて区間 [ξ_1, ξ_{n_t}] に含まれているとする.

まず，B-spline $N_{j,m+1}(x)$ の節点列$\{t_j\}$は，$\{\xi_j\}$を基にして次のように採る （図 BSC1-3 参照）.

$$t_j\begin{cases}\xi_1\ , -m+1\le j\le 0\\ \xi_j\ , \quad 1\le j\le n_t\\ \xi_{n_t}\ , \ n_t+1\le j\le n_t+m\end{cases} \tag{4.3}$$

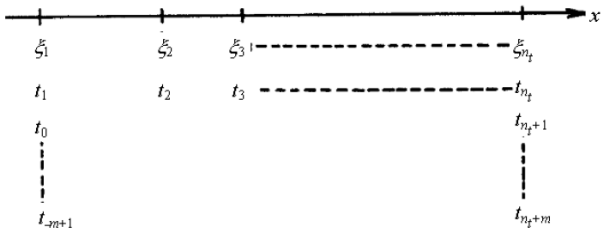


図 BSC1-3 節点列$\{t_j\}$

さて，(4.1) で表される Spline 関数の中で，(4.2) を最小にするものを求めるには，いわゆる c_j に関する正規方程式を解くことによりなされる. すなわち, (4.2) のδ_m^2 を，各 c_j について偏微分したものを 0 とおくことにより c_j に関する正規方程式 (4.4) を得る.

$$\sum_{j=-m+1}^{n_t-1}\left\{\sum_{i=1}^{n} w_i N_{k,m+1}(x_i)N_{j,m+1}(x_i)\right\}c_j=\sum_{i=1}^{n} w_i y_i N_{k,m+1}(x_i)$$
$$k=-m+1,\ -m+2,\ .....,\ n_t-1 \tag{4.4}$$

これは (n_t+m-1) 元連立 1 次方程式であり，係数行列は対数バンド行列となる．一例として m=3, n_t=5 の場合を図 BSC1-4 に示す.

$$\begin{bmatrix} * & * & * & * & & & 0\\ * & * & * & * & * & & \\ * & * & * & * & * & * & \\ * & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & & * & * & * & * & * \\ 0 & & & * & * & * & * \end{bmatrix}$$

図 BSC1-4 係数行列 (m=3, n_t=5 の場合)

したがって (4.4) を解けばc_jを求めることができる. 本サブルーチンはこの連立1次方程式をコレスキー法 (L^TL 分解法) で解いている（スレーブサブルーチン UNCA1, UCDB1 使用）.

E32-31-0202 BSC2, DBSC2

B-spline 平滑化式（節点追加方式）
CALL BSC2 (X, Y, S, N, M, XT, NT, NL, RNOT, C, RNOR, VW, IVW, ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ において観測値 $y_1, y_2, ..., y_n$, 観測誤差 $\sigma_1, \sigma_2, ..., \sigma_n$, 残差二乗和の許容値 δ_t^2, 初期節点列 $\xi_1, \xi_2, ..., \xi_{n_s}$ が与えられたとき, この節点列に適当な節点を段階的に追加して観測値を最小二乗の意味で近似する場合, 残差二乗和が許容値以下となるような m 次の B-spline 平滑化式を求める. すなわち, 最終的な節点の個数を n_t, それに対応する残差二乗和を $\delta_{n_t}^2$ とするとき,

$$\delta_{n_t}^2 = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} \left\{ y_i - \bar{S}(x_i) \right\}^2 \le \delta_t^2 \qquad (1.1)$$

が満足されるような m 次の B-spline 平滑化式

$$\bar{S}(x) = \sum_{j=-m+1}^{n_t-1} c_j N_{j,m+1}(x) \qquad (1.2)$$

の平滑化係数 c_j を求める.

なお, 本サブルーチンは, 係数 c_j のほか, 最終的な節点列 $\xi_1, \xi_2, ..., \xi_{n_t}$, 節点追加の各段階における残差二乗和 (1.3), 及び統計量 (1.4), (1.5) も出力する.

$$\delta_{n_r}^2 = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} \left\{ y_i - \bar{S}(x_i) \right\}^2 \qquad (1.3)$$

（ただし $\bar{S}(x)$ は $\xi_1, \xi_2, ..., \xi_{n_r}$ を節点列とする m 次の B-spline 平滑化式. ）

$$\bar{\sigma}_{n_r}^2 = \delta_{n_r}^2 \Big/ \left\{ n - (n_r + m - 1) \right\} \qquad (1.4)$$

$$\begin{aligned} &\mathrm{AIC}r = n \log \delta_{n_r}^2 + 2(n_r + m - 1) \\ &n_r = n_s, n_s + 1, ..., n_t \end{aligned} \qquad (1.5)$$

$\sigma_i > 0$, $m \ge 1$, $n_s \ge 2$, 及び初期節点列 ξ_j に対して, $\min_j(\xi_j) \le \min_i(x_i), \max_j(\xi_j) \ge \max_i(x_i)$ であること.

(2) パラメタ

X ····· **入力**. 離散点 x_i.
大きさ n の 1 次元配列.

Y ····· **入力**. 観測値 y_i.
大きさ n の 1 次元配列.

S ····· **入力**. 観測誤差 σ_i. （使用上の注意④参照）.
大きさ n の 1 次元配列.

N ····· **入力**. 離散点の個数 n.

M ····· **入力**. B-spline の次数 m（使用上の注意②参照）.

XT ····· **入力**. 初期節点 ξ_j, j=1,2,...,n_s.
（使用上の注意③参照）.
出力. 求められた平滑化式の節点 ξ_j, j=1, 2, ..., n_t. 結果は $\xi_1 < \xi_2 < ... < \xi_{n_t}$ の順に並べられる.
大きさ NL の 1 次元配列.

NT ····· **入力**. 初期節点の個数 n_s.
出力. 求められた平滑化式の節点の個数 n_t.

NL ····· **入力**. 節点の個数の上限.
（使用上の注意⑤参照）.

RNOT ···· **入力**. 残差二乗和の許容値 δ_t^2.
標準値としては $\delta_t^2 = n$ が適当.

C ····· **出力**. 平滑化係数 C_j, j=-m+1, -m+2, ..., n_t-1.
c_j は C (j+m) に格納される.
大きさ (NL+M-1) の 1 次元配列.

RNOR ···· **出力**. 節点追加の各段階における (1.3), (1.4), (1.5) の値.
RNOR (3,NL-n_s+1) なる 2 次元配列.
$n_r = n_s, n_s+1, ..., n_t$ とするとき,
$\delta_{n_r}^2$ は RNOR(1,n_r-n_s+1) に,
$\bar{\sigma}_{n_r}^2$ は RNOR(2,n_r-n_s+1) に,
AICr は RNOR(3,n_r-n_s+1) に格納される.
（使用上の注意⑥参照）.

VW ····· **作業領域**.
大きさ (M+1) + (M+2) (NL+M) の 1 次元配列.

IVW ····· **作業領域**.
大きさ (N+NL+M) の 1 次元配列.

ICON ····· **出力**. コンディションコード.
表 BSC2-1 参照.

表 BSC2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	節点の個数がその上限に達しても収束判定 (1.1) が満たされなかった.	最後に得られた平滑化式を出力する.
30000	次のいずれかであった. ① $\sigma_i \le 0$のものがある. ② M<1 ③ XT(I)=XT(K) I≠K ④ n_s<2 ⑤ NL<n_s ⑥ $\min_j(\xi_j) > \min_i(x_i)$ 又は, $\max_j(\xi_j) < \max_i(x_i)$	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AFMAX, MGSSL, UPOB1, UPCA1, UREO1, UBAS0, UCDB2

② FORTRAN 基本関数 ... SQRT, IFIX, ABS, FLOAT, ALOG

b. 注意

① 本サブルーチンに続けて，サブルーチン BSF1 を呼び出すことにより, B-spline 平滑化式 (1.2) に基づく平滑値，微分値あるいは積分値を求めることができる．そのとき，パラメタ M, XT, NT, C の値はサブルーチン BSF1 の入力となる．

② 次数 m (奇数でも偶数でもよい) は 3 程度が良く，5 を超えるのは良くない．これは，平滑化係数 C_j を求める際の正規方程式が，m が大きくなるにつれ，悪条件になるからである．

③ 初期節点列 ξ_j, j=1,2,...,n_s は通常，

$$n_s = 2$$
$$\xi_1 = \min_i(x_i),\ \xi_{n_s} = \max_i(x_i)$$

と与えればよい．

④ 観測誤差 σ_i は，観測値 y_i が持つ誤差の見積りである．例えば，y_i の値が，d_i 桁有効であるならば，σ_i は $10^{-d_i}|y_i|$ と与えればよい．

σ_i は，サブルーチン内で，$\bar{S}(x)$ をどの程度 y_i に近づけるかを示す量として使われる．σ_i が大きいと y_i へ近づける程度は小さい．

⑤ 節点列の個数の上限 NL は，n/2 程度に与えるのがよい (節点の個数が増えると，正規方程式が悪条件となる)．なお，本サブルーチンは，節点の個数がその上限に達しても (1.1) が満足されないときは，ICON=10000 として処理を打ち切る．

⑥ RNOR に出力される情報は，節点を追加してゆく過程での各種評価基準値の履歴である．これらの履歴は，得られた平滑化式の良悪しの確認に使える．すなわち，これら評価基準値は，通常，節点の追加と共に減少するが，その変化は徐々に緩やかになる．特に $\bar{\sigma}_{n_r}^2$ 及び AICr が殆んど変化しなくなったときは，平滑化式が良好になったことの現れである．したがって，利用者は，RNOR の内容を印刷することにより得られた平滑化式の良好さを確かめるのがよい．もし，節点の個数が n_t よりも少ない段階で，良好な平滑化式が得られていたことが分かれば，その段階での $\sigma_{n_r}^2$ を新たに許容値 δ_t^2 として，再度，本サブルーチンを呼び出せば，その段階での平滑化式を求めることができる．

c. 使用例

離散点 x_i, 観測値 y_i, 及び観測誤差 σ_i をそれぞれ 100 個入力し，これに対する 3 次の B-spline 平滑化式を求める．初期節点としては，

$$\xi_1 = x_{\min} = \min_i(x_i),\ \xi_2 = x_{\max} = \max_i(x_i)$$

の 2 点を与え，節点の個数の上限は 20 とする．続いて, サブルーチン BSF1 を用いることにより，

$$v_j = \xi_1 + (x_{\max} - x_{\min}) \cdot j/50$$
$$j = 0,1,...,50$$

の各点における平滑値，及び 1 階から 3 階までの微分値を計算する．

```
C     **EXAMPLE**
      DIMENSION X(100),Y(100),S(100),
     *          XT(20),C(25),RNOR(3,20),
     *          VW(120),IVW(125),RR(4)
      N=100
      READ(5,500) (X(I),Y(I),S(I),I=1,N)
      M=3
      NT=2
      XMAX=X(1)
      XMIN=X(1)
      DO 10 I=2,N
      IF(X(I).GT.XMAX) XMAX=X(I)
      IF(X(I).LT.XMIN) XMIN=X(I)
   10 CONTINUE
      XT(1)=XMIN
      XT(2)=XMAX
      NL=20
      RNOT=FLOAT(N)
C
      CALL BSC2(X,Y,S,N,M,XT,NT,NL,RNOT,
     *          C,RNOR,VW,IVW,ICON)
      WRITE(6,600) ICON
      IF(ICON.EQ.30000) STOP
      WRITE(6,610)
      DO 20 I=2,NT
      IADD=I-1
      WRITE(6,620) I,(RNOR(J,IADD),J=1,3)
   20 CONTINUE
C
      H=(XMAX-XMIN)/50.0
      WRITE(6,630)
      DO 40 J=1,51
      V=XT(1)+H*FLOAT(J-1)
      DO 30 L=1,4
      ISW=L-1
      CALL BSF1(M,XT,NT,C,ISW,V,I,
     *          RR(L),VW,ICON)
   30 CONTINUE
      WRITE(6,640) V,(RR(L),L=1,4)
   40 CONTINUE
      STOP
C
```

```
500 FORMAT(3F10.0)
600 FORMAT(10X,'ICON=',I5//)
610 FORMAT(8X,'NO. OF KNOTS',9X,
   *'RNOR(1,*)',11X,'RNOR(2,*)',11X,
   *'RNOR(3,*)'/)
620 FORMAT(10X,I2,8X,3(5X,E15.8))
630 FORMAT(//14X,'ARGUMENT',9X,
   *'SMOOTHED VALUE',8X,'1ST DERIV.',
   *10X,'2ND DERIV.',10X,
   *'3RD DERIV.'/)
640 FORMAT(10X,E15.8,4(5X,E15.8))
    END
```

(4) 手法概要

本サブルーチンは，節点を適応的に追加して残差二乗和が，与えられた許容値 δ_t^2 以下となるような m 次の B-spline 平滑化式を求める．

いま，節点列$\xi_1, \xi_2, ..., \xi_{n_r}$ が既に決定されていて，

$$\xi_1<\xi_2<.....<\xi_{n_r}$$

を満たしているとする．ただし，初期の段階では$n_r=n_s$である．これらを節点とする m 次の spline 関数

$$\bar{S}(x)=\sum_{j=-m+1}^{n_r-1} c_j N_{j,m+1}(x) \qquad (4.1)$$

の係数 c_j を残差二乗和，

$$\sum_{i=1}^{n}\frac{1}{\sigma_i^2}\left\{y_i-\bar{S}(x_i)\right\}^2 \qquad (4.2)$$

が最小になるようにきめる (詳細は BSC1 の手法概要参照）．求まった c_j に対する (4.2) の値を $\delta_{n_r}^2$ とする．もし，$\delta_{n_r}^2 \le \delta_t^2$ ならば，(4.1) を求めるべき m 次の B-spline 平滑化式とする．$\delta_{n_r}^2 \le \delta_t^2$ ならば，節点列が張る部分区間 $[\xi_j,\xi_{j+1}]$ における残差二乗部分和，

$$\delta^2(j,n_r)=\sum_{\xi_j\le x_i<\xi_{j+1}}\frac{1}{\sigma_i^2}\left\{y_i-\bar{S}(x_i)\right\}^2 \qquad (4.3)$$

を各 j について計算し，その中で$\delta^2(j,n_r)$ が最大となる区間の中点$\xi=(\xi_j+\xi_{j+1})/2$ を新たな節点として節点列に追加する．更新された節点列を，小さい順に並べ替え，それを$\xi_1, \xi_2, ..., \xi_{n_r+1}^2$ として，上記過程を繰り返す．

このように節点列を更新しながら次第に節点の個数を増加させ，残差二乗和が許容値 δ_t^2 以下になるまで反復する．

B51-21-0201　BSEG, DBSEG

実対称バンド行列の固有値及び固有ベクトル(ルティスハウザー・シュワルツ法，バイセクション法，逆反復法)
CALL BSEG (A, N, NH, M, NV, EPST, E, EV, K, VW, ICON)

(1)　機能

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の，大きい方から又は小さい方からの m 個の固有値を，ルティスハウザー・シュワルツ法及びバイセクション法により求め，これに対応する n_v 個の固有ベクトルを逆反復法により求める. 固有ベクトルは，$\|x\|_2 = 1$ となるように正規化する．$1 \le m \le n$, $0 \le n_v \le m$, $0 \le h << n$ であること.

(2)　パラメタ

A …… **入力**．実対称バンド行列 $\boldsymbol{A}$.
対称バンド行列用圧縮モード.
大きさ $n(h+1)-h(h+1)/2$ の 1 次元配列.
$n_v \neq 0$ のとき，演算後内容は保存される.
$n_v = 0$ のとき，演算後内容は保存されない.

N …… **入力**．行列 $\boldsymbol{A}$ の次数 n.

NH …… **入力**．バンド幅 h.

M …… **入力**．求める固有値の個数 m.
M=+m のときは大きい方から求める.
M=-m のときは小さい方から求める.

NV …… **入力**．求める固有ベクトルの個数 n_v.
NV=-n_vのときは NV=+n_v とする.
n_v=0 のときは固有ベクトルを計算しない.

EPST …… **入力**．固有値の収束判定に使われる絶対誤差の上限（サブルーチン BSCT1 の手法概要参照）．ただし，負の値を入れると標準値が設定される.

E …… **出力**．固有値．大きさ m の 1 次元配列.

EV …… **出力**．固有ベクトル.
固有ベクトルは列方向に格納される.
EV(K,NV) なる 2 次元配列.

K …… **入力**．配列 EV の整合寸法.
n_v=0 のときは任意の変数でよい.

VW …… **作業領域**．大きさ $\max(3n+2m, 2n(h+1))$ の 1 次元配列．ただし，n_v=0 のときは大きさ $3n+2m$ で済む.

ICON …… **出力**．コンディションコード.
表 BSEG-1 参照.

表 BSEG-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	NH=0 であった.	正常に実行される.
15000	固有ベクトルのすべてを求めることはできなかった.	求められなかった固有ベクトルを 0 ベクトルとする.
20000	固有ベクトルが一つも求められなかった.	すべての固有ベクトルを 0 ベクトルとする.
30000	NH<0, NH≥N, N>K, M=0, $\|M\| < \|NV\|$又は$\|M\| > N$ であった.	処理を打ち切る.

(3)　使用上の注意

a.　使用する副プログラム

① SSL II ... BTRID, BSCT1, BSVEC, AMACH, MGSSL

② FORTRAN 基本関数 ... IABS

b.　注意

① 本サブルーチンは実対称バンド行列用である.行列の次数 n とバンド幅 h の比 h/n が，1/6 以下程度の行列の固有値を，大きい方からあるいは小さい方から少数個求める場合に適する.

② 求めた固有値に対応する固有ベクトルも同時に計算できるが，本サブルーチンで用いている逆反復法は，実対称 3 重対角行列ではなく，入力されたバンド行列に対して直接処理せざるをえないために比較的非能率である．したがって，不必要な固有ベクトルはなるべく計算しない方がよい.
例えば NV=0 とすれば固有ベクトルを求めないで済む.

③ 高次数の実対称バンド行列の固有値を，絶対値の大きい方から，あるいは小さい方から極めて少数個求め，これに対する固有ベクトルをも同時に計算したい場合には，ジェニングスの同時反復法を比較検討し，より有利な方法を採用することが望ましい.

c.　使用例
次数 n．バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の大きい方から又は小さい方から m 個の固有値と，これらに対応する n_v個の固有ベクトルを求める．$n \le 100$, $h \le 10$, $m \le 10$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(1100),E(10),EV(100,10)
     *         ,VW(2100)
   10 READ(5,500) N,NH,M,NV,EPST
      IF(N.EQ.0) STOP
      NN=(NH+1)*(N+N-NH)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N,NH,M,NV
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=MIN0(NH+1,I)+NE
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL BSEG(A,N,NH,M,NV,EPST,E,EV,100,
     *          VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      MM=IABS(M)
      NNV=IABS(NV)
      IF(NNV.EQ.MM) GO TO 40
      NNV1=NNV+1
      DO 30 J=NNV1,MM
      DO 30 I=1,N
   30 EV(I,J)=0.0
   40 CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(4I5,E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'/
     *       11X,'** ORDER =',I5,10X,'NH=',
     *       I3,10X,'M=',I3,10X,'NV=',I3/)
  610 FORMAT('0',7X,I3,5E20.7/(11X,5E20.7))
  620 FORMAT(/11X,'** CONDITION CODE =',I5/)
      END
```

なお，サブルーチン SEPRT についてはサブルーチン SEIG1 の「使用例」参照のこと．

(4) 手法概要

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の固有値を大きい方から又は小さい方から m 個求め，対応する固有ベクトルを n_v個計算する．

まず $\boldsymbol{A}$ をルティスハウザー・シュワルツ法により 3 重対角行列 $\boldsymbol{T}$ に変換する．この操作を (4.1) で示す．

$$\boldsymbol{T} = \boldsymbol{Q}_s^T \boldsymbol{A} \boldsymbol{Q}_s \tag{4.1}$$

ここに $\boldsymbol{Q}_s$ は直交行列で，図 BSEG-1 に示す直交行列の積として実現される．この操作は，サブルーチン BTRID を用いて行う．

次にバイセクション法により $\boldsymbol{T}$ の m 個の固有値を求める．この操作は，サブルーチン BSCT1 を用いて行う．

次に逆反復法により，対応する $\boldsymbol{A}$ の固有ベクトル x を求める．逆反復法は $\boldsymbol{A}$ の固有値の近似解μが与えられたとき，

$$(\boldsymbol{A} - \mu \boldsymbol{I}) x_r = x_{r-1}, r = 1,2,... \tag{4.2}$$

を反復的に解いて固有ベクトルを求める方法である．ただし，μはバイセクション法により求めた固有値，x_0は適当な初期ベクトルである．この操作は，サブルーチン BSVEC を用いて行う．固有値は，$\|x\|_2 = 1$ の意味で正規化されている．

なお，詳細については，サブルーチン BTRID, BSCT1 及び BSVEC の手法概要ならびに，文献 [12], [13] を参照のこと．

$$\begin{array}{c} \\ i \\ i+1 \end{array} \begin{array}{c} \begin{array}{cccccccc} i & & & & & i+1 & & \end{array} \\ \begin{bmatrix} 1 & & & & & & & \\ & \ddots & & & & & 0 & \\ & & 1 & & & & & \\ & & & \cos\theta & \sin\theta & & & \\ & & & -\sin\theta & \cos\theta & & & \\ & & & & & 1 & & \\ & 0 & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix} \end{array}$$

図 BSEG-1 ルティスハウザー・シュワルツ法による直交相似変換行列の一般形

B51-21-1001 BSEGJ, DBSEGJ

実対称バンド行列の固有値及び固有ベクトル(ジェニングス法)
CALL BSEGJ (A, N, NH, M, EPST, LM, E, EV, K, IT, VW, ICON)

(1) 機能

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の固有値を絶対値の大きい方 (又は小さい方) から m 個，及びそれらに対応する固有ベクトルを, m 個の与えられた初期ベクトルをもとにして，ジェニングスの加速を伴うジェニングスの同時反復法により求める．ただし，絶対値の小さい方から求めるときは, $\boldsymbol{A}$ は正値行列でなければならない. なお固有ベクトルは $\|x\|_2 = 1$ のように正規化されている．$1 \le m << n$, $0 \le h << n$ であること.

(2) パラメタ

A …… **入力**．実対称バンド行列 $\boldsymbol{A}$.
絶対値の小さい方から求めるときには，演算後，その内容は保存されない.
対称バンド行列用圧縮モード.
大きさ $n(h+1)-h(h+1)/2$ の 1 次元配列.

N …… **入力**．行列 $\boldsymbol{A}$ の次数 n.

NH …… **入力**．行列 $\boldsymbol{A}$ のバンド幅 h.

M …… **入力**．求める固有値及び固有ベクトルの個数 m.
M=+m のときは絶対値の大きい方から求める.
M=-m のときは絶対値の小さい方から求める.

EPST …… **入力**．固有ベクトルの収束判定に用いられる定数 ε．零又は負のときは標準値が設定される．（使用上の注意②参照）

LM …… **入力**．反復回数の上限．反復回数がこれを超えると処理を中断する．（使用上の注意③参照）

E …… **出力**．固有値．指定された順序に格納される.
大きさ m の 1 次元配列.

EV …… **入力**．列方向に格納された m 個の初期ベクトル．（使用上の注意①参照）
出力．固有ベクトル．列方向に格納される大きさ EV (K,m+2) なる 2 次元配列.

K …… **入力**．EV の整合寸法.

IT …… **出力**．固有値固有ベクトルが求まるまでの反復回数.

VW …… **作業領域**．大きさ $\max(n,2m)+m(3m+1)/2$ 以上の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BSEGJ-1 参照.

表 BSEGJ-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	反復回数が上限 LM を超えた.	処理を終了する. E と EV にはそのときまでに得られた固有値と固有ベクトルの近似値が入っている.
28000	反復ごとの固有ベクトルの直交化処理が不可能となった.	処理を打ち切る.
29000	行列 A が正値行列でない. 又は非正則の可能性が強い.	処理を打ち切る.
30000	NH<0, NH≥N, N>K, M=0 又は |M|>N であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MSBV, SBDL, BDLX, TRID1, TEIG1, TRBK, UCHLS, USERT, MGSSL

② FORTRAN 基本関数 ... MAX0, AMAX1, ABS, IABS, FLOAT, SQRT

b. 注意

① 初期固有ベクトルは，求めようとする固有値に対応する固有ベクトルの良い近似であることが望ましい．しかしそのような近似ベクトルがわからないときには，単位行列 I の初めの m 個の列ベクトルを初期ベクトルとして採用するのが標準的である．固有値及び固有ベクトルの個数 m は，n に比べて小さく，例えば $m/n<1/10$ 程度にとるのがよい．固有値を絶対値の大きい方から, あるいは小さい方から番号をつけて, $\lambda_1, \lambda_2, \ldots, \lambda_n$ とするとき，もし可能ならば $\left|\lambda_{m+1}/\lambda_m\right| << 1$ 又は $\lambda_{m+1}/\lambda_m >> 1$ となるように m を選ぶのが，収束を速める上で望ましい.

② EPST は $\|x\|_2 = 1$ の意味で正規化された固有ベクトルの各成分の収束を判定するために用いられる．固有ベクトルが判定定数 ε に対して収束したとき，固有値は少なくとも $\|\boldsymbol{A}\| \cdot \varepsilon$ の精度で，そして多くの場合 $\|\boldsymbol{A}\| \cdot \varepsilon$ よりも高い精度で収束している．このことを考慮して幾分大きめに EPST の値を選ぶのがよい．丸め誤差の単位を u とするとき，標準値は $\varepsilon=16u$ としている．しかし，固有値が密集した問題では，収束が得られないことがあり，この場合は $\varepsilon \ge 100u$ にとるのが安全である.

③ 反復回数の上限 LM は，何らかの理由で収束が達成されないとき，強制的に反復を中断するためのものである．要求精度や固有値の密集の度合などを考慮して定められるべきものであるが, 500～1000 あたりが妥当な標準値である.

④ SSL II には，実対称バンド行列の固有値と固有ベクトルを直接法によって求めるためのサブルーチン BSEG が用意されている．同じ問題を解く場合には，概して直接法の方が速いが，固有ベクトルの計算に余分の記憶容量を必要とする難点がある．問題の規模，要求精度，記憶容量，計算機の速度などを考慮して適当なサブルーチンを採用すべきである．

c. 使用例
次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ の固有値及び固有ベクトルを求める．
n≤100, h≤10, m≤10 の場合．

```
C     **EXAMPLE**
      DIMENSION A(1100),E(10),EV(100,12)
     *         ,VW(300)
   10 READ(5,500) N,NH,M,EPST
      IF(N.EQ.0) STOP
      MM=IABS(M)
      NN=(NH+1)*(N+N-NH)/2
      READ(5,510) (A(I),I=1,NN),
     *            ((EV(I,J),I=1,N),J=1,MM)
      WRITE(6,600) N,NH,M
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=MIN0(NH+1,I)+NE
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL BSEGJ(A,N,NH,M,EPST,500,E,EV,
     *           100,IT,VW,ICON)
      WRITE(6,620) ICON,IT
      IF(ICON.GE.20000) GO TO 10
      CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(3I5,E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',20X,'ORIGINAL MATRIX',
     *       5X,'N=',I3,5X,'NH=',I3,5X,
     *       'M=',I3/)
  610 FORMAT('0',7X,I3,5E20.7/(11X,5E20.7))
  620 FORMAT('0',20X,'ICON=',I5,5X,'IT=',
     *       I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン SEIG1 の使用例を参照のこと．

(4) 手法概要

a. ジェニングスの同時反復法
同時反復法について説明する．
n 次の実対称行列 $\boldsymbol{A}$ の固有値を，絶対値の大きい方から番号をつけて，

$$\lambda_1,\lambda_2,...,\lambda_n$$

とし，これらに対応する正規直交化された固有ベクトルを，

$$v_1,v_2,...,v_n$$

とする．対角行列$\boldsymbol{\Lambda}$と，直交行列 $\boldsymbol{V}$ を

$$\boldsymbol{\Lambda}=\mathrm{diag}(\lambda_1,\lambda_2,...,\lambda_n)$$
$$\boldsymbol{V}=(v_1,v_2,...,v_n)$$

で定義すれば，

$$\boldsymbol{AV}=\boldsymbol{V\Lambda} \tag{4.1}$$

が成り立つ．

さて，同時反復法はべき乗法の拡張であり，適当な n 行 m 列行列 X_1 から出発して

$$\boldsymbol{X}_{i+1}=\boldsymbol{AX}_i \quad i=1,2,..... \tag{4.2}$$

を反復し，$\boldsymbol{X}_i$ を絶対値の最も大きい m 個の固有値に対応する固有ベクトルに，すなわち $\boldsymbol{V}$ の最初の m 列に同時に収束させようという目的を持つ．しかしながら，単に (4.2) を反復するだけでは $\boldsymbol{X}_i$ の各列はすべて V の第 1 列 v_1 に比例するベクトルに収束してしまって，目的は達成されない．そのために，同時反復法では何等かの手段により，反復行列 $\boldsymbol{X}_i$ を常に正規直交化している．これは，固有値と固有ベクトルの次のような基本的性質に基づいている．

いま，固有値$\lambda_1,...,\lambda_{k-1}$ と対応する固有ベクトル $v_1,...,v_{k-1}$ が得られているとする．
単位ベクトル v を，制約条件

$$v^T v_i=0, i=1,...,k-1$$

を満たしながら $\left|v^T Av\right|$ が $\boldsymbol{v}$=$\boldsymbol{v}_k$ のときに最大値をとるものとすると，

$$\lambda_k=v_k^T Av_k$$

は k 番目に大きい絶対値を持つ固有値であり，v_k は対応する固有ベクトルである．

ジェニングスの同時反復法の手順を示そう．簡単のために，反復行列の添字を省略する．

① $\boldsymbol{X}$ の左から $\boldsymbol{A}$ を乗じて $\boldsymbol{Y}$ とする．

$$\boldsymbol{Y}=\boldsymbol{AX} \tag{4.3}$$

② $\boldsymbol{Y}$ の左から $\boldsymbol{X}^T$ を乗じて，m 次の対称行列 $\boldsymbol{B}$ を作る．

$$\boldsymbol{B}=\boldsymbol{X}^T\boldsymbol{Y} \tag{4.4}$$

③ $\boldsymbol{B}$ の固有値μ_i, i=1,2,...,m と対応する固有ベクトル $\boldsymbol{p}_i$, i=1,2,...,m を求める.

$$\boldsymbol{B}=\boldsymbol{PMP}^T \tag{4.5}$$

ただし, $\boldsymbol{M}$ は対角行列, $\boldsymbol{P}$ は直交行列である.

$$\boldsymbol{M}=\mathrm{diag}(\mu_1,\mu_2,...,\mu_m),|\mu_1|\geq...\geq|\mu_m|$$
$$\boldsymbol{P}=(\boldsymbol{p}_1,\boldsymbol{p}_2,...,\boldsymbol{p}_m)$$

④ $\boldsymbol{Y}$ の右から $\boldsymbol{P}$ を乗じて, $\boldsymbol{Z}$ を作る.

$$\boldsymbol{Z}=\boldsymbol{YP} \tag{4.6}$$

⑤ $\boldsymbol{Z}$ の左から $\boldsymbol{Z}^T$ を乗じて, m 次の正値対称行列を作り, これをコレスキー分解 ($\boldsymbol{LL}^T$分解) する.

$$\boldsymbol{Z}^T\boldsymbol{Z}=\boldsymbol{LL}^T \tag{4.7}$$

⑥ 方程式 $\boldsymbol{X}^*\boldsymbol{L}^T=\boldsymbol{Z}$ を解いて $\boldsymbol{X}^*$ を作る.

$$\boldsymbol{X}^{*=}\boldsymbol{ZL}^{-T} \tag{4.8}$$

この $\boldsymbol{X}$* は, (4.7) により正規直交行列である.

$$\boldsymbol{X}^{*T}\boldsymbol{X}^*=\boldsymbol{I}_m \tag{4.9}$$

ただし, $\boldsymbol{I}_m$ は m 次の単位行列である.

⑦ $\boldsymbol{X}^*$に対して, 収束判定を行う. 必要に応じて, 適当に加速を施し, $\boldsymbol{X}^*$を改めて $\boldsymbol{X}$ として手順①に戻る.

以上の操作を, 反復行列 $\boldsymbol{X}$ を固有ベクトル $\boldsymbol{V}$ で展開したときの係数行列を通じて見直してみよう. $\boldsymbol{X}$ の各列を固有ベクトルで展開して

$$\boldsymbol{x}_j=\sum_{i=1}^{n}c_{ij}\boldsymbol{v}_i,\quad j=1,...,m \tag{4.10}$$

とする. (4.10) は n 行 m 列の係数行列 $\boldsymbol{C}$=(c_{ij}) によって,

$$\boldsymbol{X}=\boldsymbol{VC} \tag{4.11}$$

と書くことができる. $\boldsymbol{X}$ の正規直交性

$$\boldsymbol{X}^T\boldsymbol{X}=\boldsymbol{I}_m$$

と, $\boldsymbol{V}$ の直交性から $\boldsymbol{C}$ の直交性

$$\boldsymbol{C}^T\boldsymbol{C}=\boldsymbol{I}_m \tag{4.12}$$

がでる. $\boldsymbol{C}$ を最初の m 行と残りの n-m 行に分割して

$$\boldsymbol{C}=\begin{pmatrix}\boldsymbol{C}_1\\ \hline \boldsymbol{C}_2\end{pmatrix} \tag{4.13}$$

と表し, これに対応して$\boldsymbol{\Lambda}$をブロックに分割して

$$\boldsymbol{\Lambda}=\begin{pmatrix}\boldsymbol{\Lambda}_1 & \boldsymbol{O}\\ \hline \boldsymbol{O} & \boldsymbol{\Lambda}_2\end{pmatrix} \tag{4.14}$$

と表す. 以上の準備の下に①..., ⑦の手順を $\boldsymbol{C}$ の立場から調べてみよう.

①' (4.3) に (4.11) を代入し, (4.1) を考慮すると

$$\boldsymbol{Y}=\boldsymbol{AX}=\boldsymbol{AVC}=\boldsymbol{V\Lambda C} \tag{4.15}$$

となる. すなわち, 係数行列の第 i 行は$\boldsymbol{\lambda}_i$ 倍される. これは, (4.15) において絶対値の大きい固有値に対応する固有ベクトルの成分が強調され, 濃縮されることを示している.

②' (4.4) に (4.11) と (4.15) を代入し, $\boldsymbol{V}$ の直交性を考慮すると

$$\boldsymbol{B}=\boldsymbol{X}^T\boldsymbol{Y}=\boldsymbol{C}^T\boldsymbol{\Lambda C} \tag{4.16}$$

となる. (4.13) と (4.14) の分割を考えると

$$\boldsymbol{B}=\boldsymbol{C}_1^T\boldsymbol{\Lambda}_1\boldsymbol{C}_1+\boldsymbol{C}_1^T\boldsymbol{\Lambda}_2\boldsymbol{C}_2 \tag{4.17}$$

と書くことができる.

③' この手順については, $\boldsymbol{C}$ に何等かの仮定を設けなければ, 新しい見解は得られない.
今, (4.13) において, $\boldsymbol{C}_2$=$\boldsymbol{0}$ と仮定しよう. これは, $\boldsymbol{X}$ が $\boldsymbol{V}$ の最初のm個のベクトルで完全に表されることを意味する. このときには, (4.17) は

$$\boldsymbol{B}=\boldsymbol{C}_1^T\boldsymbol{\Lambda}_1\boldsymbol{C}_1 \tag{4.18}$$

となり, $\boldsymbol{C}_1$ は (4.12) から m 次の直交行列となる. (4.5) と (4.18) は, 同じ m 次行列 $\boldsymbol{B}$ に対する直交相似変換であるから, $\boldsymbol{V}$ を適当に取る（固有ベクトル $\boldsymbol{v}_i$の符号を変えたり, 重複固有値に対しては対応する固有ベクトルを適当に選ぶ）ことにより,

$$\boldsymbol{C}_1\boldsymbol{P}=\boldsymbol{I}_m \tag{4.19}$$
$$\boldsymbol{M}=\boldsymbol{\Lambda}_1 \tag{4.20}$$

とすることができる.

④' (4.6) と (4.15) から,

$$\boldsymbol{Z}=\boldsymbol{YP}=\boldsymbol{V\Lambda CP} \tag{4.21}$$

である. $\boldsymbol{C}_2$=$\boldsymbol{0}$ のときには,

$$\boldsymbol{Z} = \boldsymbol{V}\begin{pmatrix}\boldsymbol{\Lambda}_1 \\ \hline \boldsymbol{O}\end{pmatrix} \tag{4.22}$$

となる.

⑤' (4.21) と $\boldsymbol{V}$ の直交性から,

$$\boldsymbol{Z}^T\boldsymbol{Z} = \boldsymbol{P}^T\boldsymbol{C}^T\boldsymbol{\Lambda}^2\boldsymbol{C}\boldsymbol{P} = \boldsymbol{L}\boldsymbol{L}^T$$

となる. $\boldsymbol{C}_2=\boldsymbol{0}$ ならば,

$$\boldsymbol{Z}^T\boldsymbol{Z} = \boldsymbol{\Lambda}_1^2 = \boldsymbol{L}\boldsymbol{L}^T$$

より,

$$\boldsymbol{L} = |\boldsymbol{\Lambda}_1| = \mathrm{diag}\left(|\lambda_1|,...,|\lambda_m|\right) \tag{4.23}$$

となる.

⑥' (4.8) と (4.21) により,

$$\boldsymbol{X}^* = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{C}\boldsymbol{P}\boldsymbol{L}^{-T} \tag{4.24}$$

である. 特に $\boldsymbol{C}_2=\boldsymbol{0}$ ならば,

$$\boldsymbol{X}^* = \boldsymbol{V} \tag{4.25}$$

となる. すなわち, $\boldsymbol{C}_2=\boldsymbol{0}$ ならば 1 回の反復で, 固有値と固有ベクトルが完全に求められる. なお, $\boldsymbol{C}$ が,

$$\boldsymbol{C} = \begin{pmatrix}\boldsymbol{I}_m + \boldsymbol{E}_1 \\ \hline \boldsymbol{E}_2\end{pmatrix} \tag{4.26}$$

の形であり, $\boldsymbol{E}_1$ と $\boldsymbol{E}_2$ の成分が 1 次の微小量である場合には, 2 次の微小量を省略すれば (4.19) と, (4.20) が成り立ち,

$$\boldsymbol{X}^* = \boldsymbol{V}\cdot\begin{pmatrix}\boldsymbol{I}_m \\ \hline \boldsymbol{\Lambda}_2\boldsymbol{C}_2\boldsymbol{\Lambda}_1^{-1}\end{pmatrix} \tag{4.27}$$

となることが分かる. これは, 1 回の反復ごとに $\boldsymbol{X}$ の中の $\boldsymbol{x}_i$ $(i\leq m)$ に含まれる, 固有ベクトル $\boldsymbol{v}_j$ $(j>m)$ の成分が,

$$\boldsymbol{\lambda}_j/\boldsymbol{\lambda}_i$$

の比で, 減少して行くことを示している. このことは, $\boldsymbol{C}$ の $\boldsymbol{C}_2$ 部分が微小な値となっていくことである.

以上の観察から, 次のように言うことができる.
①の操作は, 絶対値の大きい固有値に対応する固有ベクトルの成分を濃縮する手段である.
②, ③, ④の操作は, 固有値を求めると共に, 固有ベクトルを純化する手段である.
⑤, ⑥の操作は, 固有ベクトルを正規直交化する手段である.

b. 本サブルーチンにおける計算手順
本サブルーチンでは, 次数 n, バンド幅 h の実対称行列 $\boldsymbol{A}$ の, 絶対値の大きい方 (又は小さい方) から数えて m 個の固有値と対応する固有ベクトルを, m 個の与えられた初期ベクトルをもとにして, ジェニングスの同時反復法により求めている. 手順の大要は上に示した通りであるから, 以下には具体的な詳細のみを記述する.

(a) 絶対値の小さい方から求める場合には, ①の (4.3) の代りに,

$$\boldsymbol{Y} = \boldsymbol{A}^{-1}\boldsymbol{X} \tag{4.28}$$

とする.
まず, $\boldsymbol{A}$ をサブルーチン SBDL を用いて LDL^T 分解する. このとき $\boldsymbol{A}$ が正値行列でなければ, あるいは非正則の可能性が強ければ, ICON=29000 として処理を打ち切る.

(b) ②の $\boldsymbol{B}$ を作る操作は, 領域を節約するために, 実際は次のように①の操作と組合せて行われる. すなわち, $\boldsymbol{Y}$ の列ベクトル $\boldsymbol{y}_i$ を (4.29) で計算する.

$$\begin{aligned} &v = \boldsymbol{x}_i \\ &\boldsymbol{y}_i = \boldsymbol{A}v \text{ 又は } \boldsymbol{y}_i = \boldsymbol{A}^{-1}v \\ &i = 1,...,m \end{aligned} \tag{4.29}$$

(4.29) はサブルーチン MSBV 又は BDLX により行う. $\boldsymbol{B}$ は m 次の実対称行列であるから, その対角部分と下三角部分に対して

$$\begin{aligned} &b_{ii} = v^T\boldsymbol{y}_i \\ &b_{ji} = \boldsymbol{x}_j^T\boldsymbol{y}_i, j = i+1,...,.m \end{aligned}$$

と計算する. このようにして, $\boldsymbol{X}$ の列ごとに $\boldsymbol{x}_i$ を取り出して $\boldsymbol{y}_i, b_{ji}$ を求めることにより, $\boldsymbol{Y}$ はそのまま $\boldsymbol{X}$ の領域に作られる.

(c) ③の操作を順次サブルーチン TRID1, TEIG1, TRBK によって行う. 次に固有値と対応する固有ベクトルを, 絶対値の大きい方から整列する. これをサブルーチン UESRT によって行う.

(d) ⑤の操作を, サブルーチン UCHLS により行う. もし, LL^T 分解が不可能となれば, ICON =28000 として処理を打ち切る.

(e) ⑦では $\boldsymbol{X}$ と $\boldsymbol{X}^*$の第 m 列 $\boldsymbol{x}_m$ と x_m^* について，

$$d = \left\| x_m^* - x_m \right\|_\infty \le \varepsilon \tag{4.30}$$

が成り立つかどうかを調べている．
収束判定が成立した場合は，③での $\boldsymbol{M}$ の対角要素を固有値 (絶対値の小さい方から求める場合には，対角要素の逆数を固有値) とし，$\boldsymbol{X}^*$の各列を対応する固有ベクトルとする．

c. ジェニングスの加速法
本サブルーチンでは，上に述べた本来のジェニングス法に，ベクトル列に関するジェニングスの加速法を組み入れている．

(a) 加速法の原理
ベクトル x に収束するベクトル列 $\boldsymbol{x}^{(k)}$, k=1,2,... があり，互いに直交するベクトル $\boldsymbol{v}_j$, j=1,2,...,n と $|p_j|<1$, j=1,2,...,n なる定数によって

$$x^{(k)} = x + \sum_{j=1}^{n} p_j^k v_j \tag{4.31}$$

と表されるものとする．今，三つの相連続するベクトル$x^{(k)}, x^{(k+1)}, x^{(k+2)}$ から新しいベクトル

$$\bar{x} = x^{(k+2)} + s\left(x^{(k+2)} - x^{(k+1)}\right) \tag{4.32}$$

を作る．ただし，

$$s = \frac{\left(x^{(k)} - x^{(k+1)}\right)^{\mathrm{T}}\left(x^{(k+1)} - x^{(k+2)}\right)}{\left(x^{(k)} - x^{(k+1)}\right)^{\mathrm{T}}\left(x^{(k)} - 2x^{(k+1)} + x^{(k+2)}\right)} \tag{4.33}$$

である．(4.31) を (4.33) に代入し，v_j の直交性を考慮すれば，

$$s = \frac{\sum_{j=1}^{n} p_j z_j}{\sum_{j=1}^{n}\left(1-p_j\right) z_j} \tag{4.34}$$

となる．ここに，

$$z_j = p_j^{2k}\left(1-p_j\right)^2 \left\| v_j \right\|_2^2, \quad j = 1,2,...,n \tag{4.35}$$

である．(4.32) に (4.31) と (4.34) を代入し，s の代りに

$$\bar{p} = \frac{s}{1+s} = \frac{\sum_{j=1}^{n} p_j z_j}{\sum_{j=1}^{n} z_j} \tag{4.36}$$

を用いると，

$$\bar{x} = x + \sum_{j=1}^{n} \frac{p_j - \bar{p}}{1-\bar{p}} \cdot p_j^{k+1} v_j \tag{4.37}$$

となる．(4.36) より，$\bar{p}$ は正の重み z_j に関する，p_j の平均値である．今 $|p_1| > |p_2| > ...$ とすれば，十分大きい k に対して
$|z_1| >> |z_j|$, $j > 1$
となるので，$p \approx p_1$ となり，したがって $\bar{x} \approx x$ が結論される．
以上がジェニングスの加速法の原理である．
さて，ジェニングス法の反復行列 x_k の第 i 列ベクトルを $x_i^{(k)}$ とすれば，十分大きい k に対して

$$x_i^{(k)} = v_i + \sum_{j=m+1}^{n} \left(\lambda_j / \lambda_i\right)^k c_{ji} v_j \tag{4.38}$$

と表され，$|\lambda_j / \lambda_i| < 1$ である．したがって，ジェニングスの加速法を適用できる．

(b) 加速法の適用手順
① 加速を行うかどうかの判定定数δの初期値を

$$\delta = \frac{1}{2\sqrt{n}} \tag{4.39}$$

と定める．
② 加速に関するもう一つの判定定数ηを

$$\eta = \max\left(\delta^{1.5}, \varepsilon\right) \tag{4.40}$$

と定める．加速をかけるべきベクトルの番号 j_a を m にセットし，以下の加速サイクルに入る．
③ EV の第 m+1 列と第 m+2 列に，x_{j_a} の最も新しい，一つ置きの反復ベクトルを格納する．
④ ジェニングスの同時反復を 1 回行う．
⑤ j_a=m のときだけ (4.30) の収束判定を行う．
⑥ (4.41) を満足するならば，⑩へ行く．

$$\left\| x_{j_a}^* - x_{j_a} \right\| \le \eta \tag{4.41}$$

⑦ (4.42) を満足するならば，③へ戻る．

$$\left\| x_{j_a}^{*} - x_{j_a} \right\| \le \delta \tag{4.42}$$

⑧ x_{j_a} の，三つの相連続する，一つ置きの反復ベクトルを用いて，ジェニングスの加速を行って，$\bar{x}_{j_a}$ を求める．

⑨ $\bar{x}_{j_a}$ に正規直交化を行い，

$$\begin{aligned} & x_{j_a}^{\mathrm{T}} \cdot x_i = 0, \quad j \ne j_a \\ & \left\| x_{j_a} \right\|_2 = 1 \end{aligned} \tag{4.43}$$

が成り立つようにする．

⑩ j_a を 1 だけ減じて，$j_a \ge 1$ なら③に戻る．

⑪ ⑤における収束判定が合格であったならば，反復を終了する．さもなければ，δ を

$$\delta = \max\left(\delta/1000, \eta, 10\varepsilon\right) \tag{4.44}$$

によって更新し，②へ戻る．

アルゴリズム上の留意点

ジェニングスの加速を，一つ置きに相連続するベクトルに施す理由は次のとおりである．

ジェニングスの加速の効果は，(4.37) から分るように，p_j の正の重みつき平均 $\bar{p}$ が p_1 に近い時に発揮される．すべての p_j が同符号である場合には，このようなことが起り易いが p_j の中に p_1 と異符号で，絶対値が $|p_1|$ に近いものがある時には，必ずしも加速の効果は期待できない．相連続するベクトルの代りに，一つ置きに相連続するベクトルを取れば，p_j を p_j^2 で置き換えたことになり，p_j の異符号性による困難が除去される．固有値問題に即していえば，符号の異なる，絶対値がほぼ等しい固有値がある場合でも，加速の効果が期待できる．

なお，ジェニングスの同時反復法については参考文献 [18] 及び [19] を，加速については [18] を参照すること．

E-31-32-0101　BSFD1, DBSFD1

B-Spline 2 次元平滑化式による平滑化，数値微分，数値積分
CALL BSFD1 (M, XT, NXT, YT, NYT, C, KC, ISWX, VX, IX, ISWY, VY, IY, F, VW, ICON)

(1) 機能

$x\,y$ 平面上の格子点 (x_i,y_j); i=1,2,...,n_x, j=1,2,...,n_y において，観測値 $f_{i,j}=f(x_i,y_j)$，観測誤差 $\sigma_{i,j}=\sigma x_i\cdot\sigma y_j$ が与えられたとき，双 m 次の B-Spline 平滑化式に基づいて点 $P(v_x,v_y)$ における平滑値，偏微分値，あるいは領域 $[\xi_1\le x\le v_x,\ \eta_1\le y\le v_y]$ 上の二重積分値を求める．

ただし，本サブルーチンを利用する前に，サブルーチン BSCD2 により，x 方向の節点列 $\xi_1, \xi_2,..., \xi_{n_t}$，$y$ 方向の節点列 $\eta_1, \eta_2,..., \eta_{l_t}$，更に，双 m 次の B-spline 平滑化式

$$\bar{S}(x,y)=\sum_{\beta=-m+1}^{l_t-1}\sum_{\alpha=-m+1}^{n_t-1}c_{\alpha,\beta}N_{\alpha,m-1}(x)N_{\beta,m+1}(y)\qquad(1.1)$$

における平滑化係数 $c_{\alpha,\beta}$ が計算されているとする．ここで，$m\ge1$, $\xi_1\le v_x<\xi_{n_t}$, $\eta_1\le v_y\le\eta_{l_t}$ であること．

(2) パラメタ

M ····· **入力**．B-spline の次数 m.

XT ····· **入力**．x 方向の節点 ξ_i.
大きさ n_t の 1 次元配列.

NXT ····· **入力**．x 方向の節点 ξ_i の個数 n_t.

YT ····· **入力**．y 方向の節点 η_j.
大きさ l_t の 1 次元配列.

NYT ····· **入力**．y 方向の節点 η_j の個数 l_t.

C ····· **入力**．平滑化係数 $c_{\alpha,\beta}$.
C(KC, l_t+m-1) なる 2 次元配列.

KC ····· **入力**．配列 C の整合寸法 ($\ge n_t+m-1$).

ISWX····· **入力**．x 方向に関する計算の種類を与える．
-1≤ISWX≤M （パラメタ F の項参照）．

VX ····· **入力**．点 $P(v_x,v_y)$ の x 座標.

IX ····· **入力**．$\xi_i\le v_x<\xi_{i+1}$ を満たす i.
$v_x=\xi_{n_t}$ のときは，IX=n_t-1 とする．
出力．$\xi_i\le v_x<\xi_{i+1}$ を満たす i.
（使用上の注意②参照）．

ISWY····· **入力**．y 方向に関する計算の種類を与える．
-1≤ISWY≤M （パラメタ F の項参照）．

VY ····· **入力**．点 $P(v_x,v_y)$ の y 座標.

IY ····· **入力**．$\eta_j\le v_y<\eta_{j+1}$ を満たす j.
$v_y=\eta_{lt}$ のときは，IY=l_t-1 とする．
出力．$\eta_j\le v_y<\eta_{j+1}$ を満たす j.
（使用上の注意②参照）．

F····· **出力**．平滑値あるいは偏微分値あるいは，積分値．いま，ISWX=λ, ISWY=μとすると，λ, μの組合せごとに次の値を出力する．

- $0\le\lambda,\mu$のとき，

$$F=\frac{\partial^{\lambda+\mu}}{\partial x^{\lambda}\partial y^{\mu}}\bar{S}(v_x,v_y)$$

平滑値は $\lambda=\mu=0$ とすることにより得られる．

- λ=-1, $0\le\mu$のとき，

$$F=\int_{\xi_1}^{v_x}\left\{\frac{\partial^{\mu}}{\partial y^{\mu}}\bar{S}(x,v_y)\right\}dx$$

- λ≥0, μ=-1のとき，

$$F=\int_{\eta_1}^{v_y}\left\{\frac{\partial^{\lambda}}{\partial x^{\lambda}}\bar{S}(v_x,y)\right\}dy$$

- λ=μ=-1のとき，

$$F=\int_{\eta_1}^{v_y}dy\int_{\xi_1}^{v_x}\bar{S}(x,y)dx$$

VW ····· **作業領域**.
大きさ 5(m+1)+max(n_t, l_t) の 1 次元配列.

ICON ····· **出力**．コンディションコード.
表 BSFD1-1 参照.

表 BSFD1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	XT(IX)≤VX<XT(IX+1) 又は YT(IY)≤VY<YT(IY+1) が満たされていない.	サブルーチン内で左記の IX 又は IY をさがして処理を続ける.
30000	次のいずれかであった. ① VX<XT(1)又は VX>XT(NXT) ② VY<YT(1)又は VY>YT(NYT) ③ ISWX<-1 又は ISWX>M ④ ISWY<-1 又は ISWY>M	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UCAR2, UBAS1

② FORTRAN 基本関数 ... FLOAT

b. 注意

① 本サブルーチンは，サブルーチン BSCD2 により求められた B-spline2 次元平滑化式 (1.1) に基づいて，平滑値，偏微分値，あるいは二重積分値を求めるものである．

したがって，本サブルーチンを呼び出す前に，BSCD2 を呼び出して平滑化式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより平滑値などを求めることができる．そのとき，パラメタ M, XT, NXT, YT, NYT, C, KC の値は BSCD2 のそれらと同一でなければならない．

② パラメタ IX, IY は，それぞれ XT(IX)≤ VX< XT (IX+1), YT(IY)≤VY<YT(IY+1) を満足していることが好ましい．もし満足されていないときは，これらを満足するような IX, IY を探索して処理を続ける．

c. 使用例

サブルーチン BSCD2 の使用例参照.

(4) 手法概要

サブルーチン BSCD2 により双 *m* 次の B-spline2 次元平滑化式,

$$\bar{S}(x,y)=\sum_{\beta=-m+1}^{l_c-1}\sum_{\alpha=-m+1}^{n_c-1}c_{\alpha,\beta}N_{\alpha,m+1}(x)N_{\beta,m+1}(y) \qquad (4.1)$$

が求まっているとする.

本サブルーチンでは, (4.1) の平滑化式に基づいて，平滑値，偏微値，あるいは積分値などを計算する．その手法については，第 I 部 7.1 の「(5) 2 変数 Spline 関数の定義，表現及び計算法」を参照されたい.

E31-31-0101 BSF1, DBSF1

B-spline 平滑化式による平滑化，数値微分，数値積分
CALL BSF1(M,XT,NT,C,ISW,V,I,F,VW,ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ に対して観測値 $y_1, y_2, ..., y_n$, 重み関数値 $w_i=w(x_i)$, i=1,2,...,n 及び Spline 関数の節点ξ_1, $\xi_2, ..., \xi_{n_t}$ ($\xi_1<\xi_2<...<\xi_{n_t}$) が与えられたとき，B-spline 平滑化式に基づいて $x=v\in\left\lfloor \xi_1,\xi_{n_t}\right\rfloor$における平滑値あるいは微分値あるいは$\xi_1$ から v までの積分値を求める.

ただし，本サブルーチンを利用する前に，サブルーチン BSC1 又は BSC2 により，B-spline 平滑化式,

$$\overline{S}(x)=\sum_{j=-m+1}^{n_t-1} c_j N_{j,m+1}(x) \qquad (1.1)$$

における平滑化係数 c_j, j=-m+1, -m+2, ..., n_t-1 が計算されているとする．ここで m は B-spline $N_{j,m+1}(x)$ の次数である.

$\xi_1\le v\le\xi_{n_t}$, $m\ge1$, $n_t\ge3$ であること.

(2) パラメタ

M ····· **入力**．B-spline の次数 m.
（使用上の注意①参照）.

XT ····· **入力**．節点ξ_j .
大きさ n_t の 1 次元配列.

NT ····· **入力**．節点ξ_j の個数 n_t .

C ····· **入力**．平滑化係数 c_j （BSC1 又は BSC2 の出力).
大きさ n_t+m-1 の 1 次元配列.

ISW ····· **入力**．計算の種類を与える.
ISW=0 のとき平滑値 F = $\overline{S}(v)$,
ISW= $l\left(1\le l\le m\right)$ のとき,
l 階微分値 F = $\overline{S}^{(l)}(v)$,
ISW=-1 のとき積分値 F = $\int_{\xi_1}^{v}\overline{S}(x)dx$
を求める.

V ····· **入力**．平滑値などを求めたい点 v.

I ····· **入力**．$\xi_i\le v<\xi_{i+1}$ を満たす i.
$v=\xi_{n_t}$ のときは $I=n_t$-1 とする.
出力．$\xi_i\le v<\xi_{i+1}$ を満たす i.
（使用上の注意②参照）.

F ····· **出力**．平滑値あるいは l 階微分値あるいは積分値 (ISW の項参照）.

VW ····· **作業領域**．大きさ m+1 の 1 次元配列.

ICON ····· **出力**．コンディションコード.
表 BSF1-1 参照.

表 BSF1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	XT(I) ≤V<XT(I+1) が満たされていない.	サブルーチン内で左記の I をさがして処理を続ける.
30000	次のいずれかであった. ① V<XT(1) 又は V>XT(NT) ② ISW<-1 又は ISW>M	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UCAR1, UBAS1
② FORTRAN 基本関数 ... FLOAT

b. 注意
① 本サブルーチンは, サブルーチン BSC1 又は BSC2 により求められた B-spline 平滑化式 (1.1) に基づいて平滑値あるいは微分値あるいは積分値を求めるものである．したがって，本サブルーチンを呼び出す前に，BSC1 又は BSC2 を呼び出して平滑化式 (1.1) を求め，続けて本サブルーチンを呼び出すことにより平滑値などを求めることができる. そのとき, パラメタ M, XT, NT, C は BSC1 又は BSC2 のそれらと同一でなければならない.
② パラメタ I は XT(I)≤V<XT(I+1) を満足していることが好ましい．もし満足されていないときは, XT(I)≤V<XT(I+1) を満足するような I を探索して処理を続ける.

c. 使用例
離散点 x_i, 観測値 y_i, i=1,2,...,n, 節点ξ_j, j=1,2,...,n_t 及び次数 m を入力し,

$$v_{ij}=\xi_i+\left(\xi_{i+1}-\xi_i\right)\times\left(j/5\right),i=1,2,..,n_t-1,$$
$$j=0,1,..,5$$

における，ξ_1 からの積分値，平滑値，1 階から m 階までの微分値を求める．ただし，各観測値に対する重み関数値 w_i, i=1,2,...,n はすべて 1.0 とし，更に $\min(\xi_j)\le x_i\le\max(\xi_j)$, i=1,2,...,n とする．n≤101, n_i≤10, m≤5 の場合.

```
C     **EXAMPLE**
      DIMENSION X(101),Y(101),W(101),
     *XT(10),C(14),R(101),VW(66),IVW(101),
     *RR(6)
      READ(5,500) N,M
      READ(5,510) (X(I),Y(I),I=1,N)
      READ(5,500) NT
      READ(5,520) (XT(I),I=1,NT)
      WRITE(6,600) N,M,(I,X(I),Y(I),I=1,N)
      WRITE(6,610) NT,(I,XT(I),I=1,NT)
```

```
      DO 10 I=1,N
   10 W(I)=1.0
      CALL BSC1 (X,Y,W,N,M,XT,NT,C,R,
     *RNOR,VW,IVW,ICON)
      IF(ICON.EQ.0) GO TO 20
      WRITE(6,620)
      STOP
   20 WRITE(6,630) RNOR
      N1=NT-1
      M2=M+2
      DO 50 L2=1,M2
      ISW=L2-2
      WRITE(6,640) ISW
      DO 40 I=1,N1
      H=(XT(I-1)-XT(I))/5.0
      XI=X(I)
      DO 30 J=1,6
      V=XI+H*FLOAT(J-1)
      II=I
      CALL BSF1(M,XT,NT,C,ISW,V,
     *II,F,VW,ICON)
      RR(J)=F
   30 CONTINUE
      WRITE(6,650) II,(RR(J),J=1,6)
   40 CONTINUE
   50 CONTINUE
      STOP
  500 FORMAT(2I6)
  510 FORMAT(2F12.0)
  520 FORMAT(F12.0)
  600 FORMAT('1'//10X,'INPUT DATA',3X,
     *'N=',I3,3X,'M=',I2//20X,'NO.',10X,
     *'X',17X,'Y'//(20X,I3,2E18.7))
  610 FORMAT(/10X,'INPUT KNOTS',3X,
     *'NT=',I3/20X,'NO.',10X,'XT'//
     *(20X,I3,E18.7))
  620 FORMAT('0',10X,'ERROR')
  630 FORMAT(10X,'SQUARE SUM OF',1X,
     *'RESIDUALS=',E18.7)
  640 FORMAT('1'//10X,'L=',I2/)
  650 FORMAT(6X,I3,6E18.7)
      END
```

(4) 手法概要

サブルーチン BSC1 又は BSC2 により m 次の B-spline 平滑化式,

$$\bar{S}(x) = \sum_{j=-m+1}^{n_t-1} c_j N_{j,m+1}(x) \tag{4.1}$$

が求まっているとする.

本サブルーチンは (4.1) の平滑化式に基づいて，平滑値, l 階微分値あるいは積分値をそれぞれ (4.2), (4.3), (4.4) より求める.

$$\bar{S}(v) = \sum_{j=-m+1}^{n_t-1} c_j N_{j,m+1}(v) \tag{4.2}$$

$$\bar{S}^{(l)}(v) = \sum_{j=-m+1}^{n_t-1} c_j N^{(l)}_{j,m+1}(v) \tag{4.3}$$

$$I = \int_{\xi_1}^{v} \bar{S}(x)dx \tag{4.4}$$

これら三つの計算法については，7 章 7.1 の「(4) Spline 関数の計算法」で一般的に述べたので，そこを参照されたい.

本サブルーチンでは，$N_{j,m+1}(x)$, 及びその微分，積分の計算はスレーブサブルーチン UBAS1 で行っている.

B51-21-0402　　BSVEC, DBSVEC

実対称バンド行列の固有ベクトル（逆反復法）
CALL BSVEC(A,N,NH,NV,E,EV,K,VW,ICON)

(1) 機能

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ において，n_v 個の固有値が与えられたとき，それらに対応する固有ベクトルを，逆反復法によって求める．

(2) パラメタ

A …… **入力**．実対称バンド行列 $\boldsymbol{A}$.
対称バンド行列用圧縮モード．
大きさ $n(h+1)-h(h+1)/2$ の 1 次元配列.

N …… **入力**．行列 $\boldsymbol{A}$ の次数 n.

NH …… **入力**．バンド幅 h.

NV …… **入力**．求める固有ベクトルの数 n_v.
NV=$-n_v$ のときは NV=$+n_v$ とする．

E …… **入力**．固有値.
少なくとも大きさ n_v の 1 次元配列.

EV …… **出力**．固有ベクトル.
固有ベクトルは列方向に格納される．
EV (K, n_v) なる 2 次元配列.

K …… **入力**．配列 EV の整合寸法.

VW …… **作業領域**．大きさ $2n(h+1)$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 BSVEC-1 参照.

表 BSVEC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	NH=0 であった.	正常に実行される.
15000	固有ベクトルのすべてを求めることはできなかった.	求められなかった固有ベクトルを 0 ベクトルとする.
20000	固有ベクトルが一つも求められなかった.	すべての固有ベクトルを 0 ベクトルとする.
30000	NH<0, NH≥N, N>K,NV=0 又は |NV|>N であった.	処理を打ち切る.

(3) 使用上の注意

a.　使用する副プログラム

① SSL II ...　AMACH, MGSSL

② FORTRAN 基本関数 ... MAX0, MIN0, ABS, SIGN, SQRT

b.　注意

① 本サブルーチンは実対称バンド行列用である．実対称行列の固有値及び固有ベクトルを求める場合は，サブルーチン SEIG1 又は SEIG2 を用いる．また，実対称 3 重対角行列の固有値及び固有ベクトルを求める場合は, サブルーチン TEIG1 又は TEIG2 を用いること.

② 固有値がある狭い範囲に密集しているときには, 対応する固有ベクトルを求めるための逆反復法が収束しないことがある．このような場合には ICON は 15000 又は 20000 となり, 求められなかった固有ベクトルは 0 ベクトルになっている.

c.　使用例

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ と，その n_v 個の固有値が与えられたとき，対応する固有ベクトルを逆反復法で求める. ただし $n\leq 100$, $h\leq 10$, $n_v\leq 10$ とする.

```
C     **EXAMPLE**
      DIMENSION A(1100),E(10),EV(100,10),
     *          VW(2100)
   10 READ(5,500) N,NH,NV
      IF(N.EQ.0) STOP
      NN=(NH+1)*(N+N-NH)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N,NH,NV
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=MIN0(NH+1,I)+NE
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      NNV=IABS(NV)
      READ(5,510) (E(I),I=1,NNV)
      CALL BSVEC(A,N,NH,NV,E,EV,100,VW,
     *           ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL SEPRT(E,EV,100,N,NNV)
      GO TO 10
  500 FORMAT(3I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'/
     *       11X,'** ORDER =',I5,'NH=',I3,
     *       10X,'NV=',I3/)
  610 FORMAT('0',7X,I3,5E20.7/(11X,5E20.7))
  620 FORMAT(/11X,'** CONDITION CODE =',I5/)
      END
```

なお，サブルーチン SEPRT についてはサブルーチン SEIG1 の「使用例」参照のこと.

(4) 手法概要

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ とその n_v 個の固有値が与えられているとき, 対応する固有ベクトルを逆反復法で求める.

今，真の固有値は

$$\lambda_1>\lambda_2>\ldots>\lambda_n \qquad (4.1)$$

となっているとする．λ_j の近似値として μ_j が与えられたとき，逆反復法では，$\boldsymbol{x}_0$ を適当な初期ベクトルとして

$$\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)\boldsymbol{x}_r=\boldsymbol{x}_{r-1}\quad r=1,2,... \tag{4.2}$$

を反復的に解いて行き，収束条件を満足したときの $\boldsymbol{x}_r$ を固有ベクトルとする．

まず，$\boldsymbol{A}-\mu_j\boldsymbol{I}$ を単位下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ によって

$$\boldsymbol{P}\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)=\boldsymbol{LU} \tag{4.3}$$

と分解し

$$\boldsymbol{LU}\boldsymbol{x}_r=\boldsymbol{P}\boldsymbol{x}_{r-1} \tag{4.4}$$

として解く．ただし $\boldsymbol{P}$ は部分的ピボッティングのための行交換を行う置換行列である．(4.4) を解くには，

$$\boldsymbol{L}\boldsymbol{y}_{r-1}=\boldsymbol{P}\boldsymbol{x}_{r-1}\ \text{(前進代入)} \tag{4.5}$$
$$\boldsymbol{U}\boldsymbol{x}_r=\boldsymbol{y}_{r-1}\quad\text{(後退代入)} \tag{4.6}$$

という二つの連立方程式を解けばよい．

以上が本来の逆反復法であるが，これをそのまま実対称バンド行列に適用すると，(4.3) での行交換によってバンド幅が拡大し，LU 分解された行列の記憶容量が，もとの行列 $\boldsymbol{A}$ のそれに較べ，大きく増加することになる．この増加を軽減するために本サブルーチンでは $\boldsymbol{L}$ 成分を捨て，$\boldsymbol{U}$ 成分だけを残し，

$$\boldsymbol{U}\boldsymbol{x}_r=\boldsymbol{x}_{r-1} \tag{4.7}$$

を反復的に解くことにしている．

適当な初期固有ベクトル $\boldsymbol{x}_0$ は，真の固有値に対応する真の固有ベクトル $\boldsymbol{u}_1,\boldsymbol{u}_2,...,\boldsymbol{u}_n$ によって，

$$\boldsymbol{x}_0=\sum_{i=1}^{n}\alpha_i^{(0)}\boldsymbol{u}_i \tag{4.8}$$

と表すことができるから，(4.7) より

$$\boldsymbol{x}_1=\boldsymbol{U}^{-1}\boldsymbol{x}_0 \tag{4.9}$$

として，(4.3) と (4.8) を (4.9) へ代入すると，

$$\begin{aligned}\boldsymbol{x}_1&=\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)^{-1}\boldsymbol{P}^T\boldsymbol{L}\sum_{i=1}^{n}\alpha_i^{(0)}\boldsymbol{u}_i\\&=\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)^{-1}\sum_{i=1}^{n}\alpha_i^{(0)}\boldsymbol{P}^T\boldsymbol{L}\boldsymbol{u}_i\end{aligned}$$

ここで，$\boldsymbol{P}^T\boldsymbol{L}\boldsymbol{u}_i=\sum_{k=1}^{n}\beta_{ik}\boldsymbol{u}_k$ と置くと，

$$\begin{aligned}x_1&=\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)^{-1}\sum_{i=1}^{n}\alpha_i^{(0)}\left(\sum_{k=1}^{n}\beta_{ik}\boldsymbol{u}_k\right)\\&=\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)^{-1}\sum_{k=1}^{n}\left(\sum_{i=1}^{n}\alpha_i^{(0)}\beta_{ik}\right)\boldsymbol{u}_k\end{aligned}$$

この $\sum_{i=1}^{n}\alpha_i^{(0)}\beta_{ik}$ をあらためて $\alpha_k^{(1)}$ とすると，

$$\boldsymbol{x}_1=\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)^{-1}\sum_{i=1}^{n}\alpha_i^{(1)}\boldsymbol{u}_i$$

となる．同様にして $\sum_{i=1}^{n}\alpha_i^{(r-1)}\beta_{ik}$ を $\alpha_k^{(r)}$ と置くと，

$$\boldsymbol{x}_r=\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)^{-r}\sum_{i=1}^{n}\alpha_i^{(r)}\boldsymbol{u}_i\quad r=1,2,... \tag{4.10}$$

から，$\boldsymbol{x}_r$ は

$$\boldsymbol{x}_r=\frac{1}{(\lambda_j-\mu_j)^r}\left\{\alpha_j^{(r)}\boldsymbol{u}_j+\sum_{\substack{i=1\\i\neq j}}^{n}\alpha_i^{(r)}\boldsymbol{u}_i(\lambda_j-\mu_j)^r\Big/(\lambda_i-\mu_j)^r\right\} \tag{4.11}$$

で与えられる．定数 $1/(\lambda_j-\mu_j)^r$ は，反復ごとに $\boldsymbol{x}_r$ を調整することにより省略することができる．したがって，

$$\boldsymbol{x}_r=\alpha_j^{(r)}\boldsymbol{u}_j+\sum_{\substack{i=1\\i\neq j}}^{n}\alpha_i^{(r)}\boldsymbol{u}_i\left(\lambda_j-\mu_j\right)^r/\left(\lambda_i-\mu_j\right)^r \tag{4.12}$$

となる．一般に $\left|\left(\lambda_j-\mu_j\right)/\left(\lambda_i-\mu_j\right)\right|\ll 1$ であるから，(4.12) は，$\alpha_j^{(r)}\neq 0$ であれば r が大きくなる程 $\boldsymbol{x}_r$ は固有ベクトル $\alpha_j^{(r)}\boldsymbol{u}_j$ に接近することを示している．

① 初期ベクトルと収束判定について

本サブルーチンでは，各段階で，$\boldsymbol{x}_{r-1}$ を

$$\left\|\boldsymbol{x}_{r-1}\right\|_1=n^{3/2}u\left\|\boldsymbol{A}\right\| \tag{4.13}$$

となるように調整し，$\boldsymbol{x}_r$ が

$$\left\|\boldsymbol{x}_r\right\|_1\geq 1 \tag{4.14}$$

を満たしたとき，$\boldsymbol{x}_r$ を固有ベクトルとして採用する．ただし u は丸めの誤差単位であり，$\left\|\boldsymbol{A}\right\|$ としては $\left\|\boldsymbol{A}\right\|_1$ に近く，計算が容易な

$$\left\|\boldsymbol{A}\right\|=\frac{2}{n}\sum_{i\geq j}\left|a_{ij}\right| \tag{4.15}$$

を用いている．その理由は次のとおりである．(4.3) と (4.7) から，

$$\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)\boldsymbol{x}_r/\left\|\boldsymbol{x}_r\right\|_1=\boldsymbol{P}^T\boldsymbol{L}\boldsymbol{x}_{r-1}/\left\|\boldsymbol{x}_r\right\|_1 \qquad (4.16)$$

となる．(4.16) の右辺は，$\boldsymbol{x}_r/\left\|\boldsymbol{x}_r\right\|_1$ を固有ベクトルとしたときの残差ベクトルである．$\boldsymbol{L}$ は対角要素が 1 で，非対角要素の絶対値が 1 より小さい下三角行列であり，$\boldsymbol{P}$ は置換行列であるから，1次変換 $\boldsymbol{P}^T\boldsymbol{L}$ によってベクトルのノルムは増大しないと考えられる．したがって，$\left\|\boldsymbol{x}_r\right\|_1\geq 1$ となれば, (4.16) の右辺のノルムは非常に小さいので，$\boldsymbol{x}_r$ を固有ベクトルとみなしても差支えない．
初期ベクトル $\boldsymbol{x}_0$ としては，統計的な擬似乱数ではないが、漸近的に [0,1] 区間で一様分布する乱数

$$R_i=i\phi-\left[i\phi\right]\ i=1,2,... \qquad (4.17)$$

の連続する n 個の値を成分とするベクトルを用いる．
ただし

$$\phi=\left(\sqrt{5}-1\right)/2 \qquad (4.18)$$

である．このようにすることにより，重複固有値の場合に初期ベクトルの取り方を変更したり，あるいは固有値に摂動を与えたりする必要がなくなり，算法が簡単となる．本サブルーチンでは計算値の再現性を確保するために引用の度ごとに乱数を $R_1=\phi$ に初期化している．

(4.7) の反復を 5 回行っても収束判定 (4.14) が成立しない場合は, (4.13) の $\left\|\boldsymbol{A}\right\|$ の係数 $10n^{3/2}u$ として判定基準を緩和し，更に 5 回の反復を試みる．それでも収束が達成できない場合には非収束としてみなして ICON を 15000 にセットし，対応する EV の列をすべて零とする．

② 固有ベクトルの直交化について
実対称行列の固有ベクトルはすべて直交するはずであるが，固有値が互いに接近すると，直交性がくずれる傾向がある．
そこで，本サブルーチンでは，固有ベクトルの直交性を確保するために次のように対処している．
まず，対応する固有ベクトルを求めようとする固有値 μ_i と直前の固有値 μ_{i-1} が

$$\left|\mu_i-\mu_{i-1}\right|\leq 10^{-3}\left\|\boldsymbol{A}\right\| \qquad (4.19)$$

を満足するかどうかを調べる．もしそうであれば，μ_i に対する固有ベクトル $\boldsymbol{x}_i$ を μ_{i-1} に対する固有ベクトル $\boldsymbol{x}_{i-1}$ に対して

$$\left(\boldsymbol{x}_i,\boldsymbol{x}_{i-1}\right)=0 \qquad (4.20)$$

を満足するように修正する．

同様の考え方により，連続して (4.19) の関係を満足する固有値を一つのグループとし，対応する固有ベクトルの最直交化を行っている．

B51-21-0302　BTRID, DBTRID

実対称バンド行列の実対称三重対角行列への変換(ルティスハウザー・シュワルツ法)
CALL BTRID(A,N,NH,D,SD,ICON)

(1) 機能

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ を，ルティスハウザー・シュワルツの直交相似変換により，実対称 3 重対角行列 $\boldsymbol{T}$ に変換する．$\boldsymbol{T}=\boldsymbol{Q}_s^T\boldsymbol{A}\boldsymbol{Q}_s$．ただし $\boldsymbol{Q}_s$ は直交行列である．$0\le h<<n$ であること．

(2) パラメタ

A……**入力**．実対称バンド行列 $\boldsymbol{A}$．演算後，内容は保存されない．対称バンド行列用圧縮モード．
大きさ $n(h+1)-h(h+1)/2$ の 1 次元配列．

N……**入力**．行列 $\boldsymbol{A}$ の次数 n．

NH……**入力**．バンド幅 h．

D……**出力**．3 重対角行列の主対角要素．
大きさ n の 1 次元配列．

SD……**出力**．3 重対角行列 $\boldsymbol{T}$ の副対角要素．大きさ n の 1 次元配列．ただし SD(2) ～SD(N) を使用し，SD(1) には零がセットされる．

ICON……**出力**．コンディションコード．
表 BTRID-1 参照．

表 BTRID-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	NH=0 又は NH=1 であった．	変換しない．
30000	NH<0 又は NH≥N であった．	処理を打ち切る．

(3) 使用上の注意

a.　使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数 ... MIN0, ABS, SQRT

b.　注意

① 実対称 3 重対角行列への変換法であるハウスホルダー法と比較すると，バンド幅と次数の比 $r=h/n$ が小さいときは，記憶容量の点でも計算量の点でも本サブルーチンのルティスハウザー法がまさるが，r が 1/6 程度を超えると，少なくとも計算量と精度に関してはハウスホルダー法がまさる．

c.　使用例

次数 n，バンド幅 h の実対称バンド行列を 3 重対角行列に変換した後，サブルーチン BSCT1 により，固有値を計算する．ただし $n\le100$, $h\le10$ とする．

```
C     **EXAMPLE**
      DIMENSION A(1100),D(100),SD(100),
     *          E(100),VW(300)
   10 READ(5,500) N,NH,M,EPST
      IF(N.EQ.0) STOP
      NN=(NH+1)*(N+N-NH)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N,NH,IABS(M)
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=MIN0(NH+1,I)+NE
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL BTRID(A,N,NH,D,SD,ICON)
      WRITE(6,620)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,640) (I,D(I),SD(I),I=1,N)
      CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)
      WRITE(6,650)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      MM=IABS(M)
      WRITE(6,660) (I,E(I),I=1,MM)
      GO TO 10
  500 FORMAT(3I5,E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'/
     *       11X,'** ORDER =',I5,10X,'NH=',
     *       I3,'M=',I3/)
  610 FORMAT('0',7X,I3,5E20.7/(11X,5E20.7))
  620 FORMAT('0'/11X,'** TRIDIAGONAL ',
     *       'MATRIX')
  630 FORMAT(/11X,'** CONDITION CODE =',I5/)
  640 FORMAT(5X,I5,2E16.7)
  650 FORMAT('0'/11X,'** EIGENVALUES')
  660 FORMAT(5X,'E(',I3,')=',E15.7)
      END
```

(4) 手法概要

実対称行列の 3 重対角行列への変換法としては，ハウスホルダー法 (サブルーチン TRID1 の「手法概要」参照） が有名であるが，これをバンド行列に適用すると，変換の途中では非零要素が行列全体に広がってしまうので，バンド部分の枠内で処理することができない．これに対して，ルティスハウザー・シュワルツ法では，バンド行列 $\boldsymbol{A}$ (次数 n, バンド幅 h, 要素 a_{ij}) のおおかたバンド部分だけの処理によって，副対角線を外側から 1 本ずつ消去して行くことができる．

今，対称性を利用して $\boldsymbol{A}$ の対角線を含む下バンド部分だけを考えよう．

まず $a_{h+1,1}$ を消去するために，$\boldsymbol{A}$ の第 h 行と第 $h+1$ 行に関する直交行列

$$R_1=\begin{array}{c} \\ \\ \\ \\ h \\ h+1 \\ \\ \\ \end{array}\begin{array}{c} h\quad h+1 \\ \left[\begin{array}{ccccccc} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & 0 & & \\ & & & \cos\theta_1 & \sin\theta_1 & & \\ & & & -\sin\theta_1 & \cos\theta_1 & & \\ & & 0 & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{array}\right] \end{array}$$

により, $\boldsymbol{A}_1 = \boldsymbol{R}_1\boldsymbol{A}\boldsymbol{R}_1^T$ とする．ただし $\tan\theta_1 = a_{h+1,1}/a_{h,1}$ である．この操作により $a_{h+1,1}$ は 0 となるが，新しい非零要素 $a_{2h+1,h}$ が発生する.これを消去するために第 $2h$ 行と第 $2h$+1 行に関する直交行列

$$R_2 = \begin{array}{c} \begin{matrix} & & & 2h & 2h+1 & & \end{matrix} \\ \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & 0 & & \\ & & & \cos\theta_2 & \sin\theta_2 & & \\ & & & -\sin\theta_2 & \cos\theta_2 & & \\ & & 0 & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{bmatrix} \end{array} \begin{matrix} \\ \\ \\ 2h \\ 2h+1 \\ \\ \\ \end{matrix}$$

により $\boldsymbol{A}_1$=$\boldsymbol{R}_2\boldsymbol{A}_1\boldsymbol{R}_2^T$ とする．ただし $\tan\theta_2 = a_{2h+1,h}/a_{2h,h}$ である．この操作により $a_{2h+1,h}$ は 0 となるが，新しい非零要素 $a_{3h+1,2h}$ が発生する．このようにして新しく発生する非零要素を消去するために，適当な直交行列による相似変換 (直交相似変換) を施して行くことにより，$a_{h+1,1}$ だけを完全に消去することができる．次に，$a_{h+2,2}$ を消去するために適当な直交相似変換を行い，付随的に発生する非零要素を，また直交相似変換で消去する．これを繰り返せば，$a_{h+2,2}$ を完全に消去することができる．以上の操作を繰り返せば，ついに最も外側の副対角線全体を消去することができてバンド幅は h-1 となる．

この新しい行列に対して初めの行列と同様の処理を行えば, 更にバンド幅を 1 だけ小さくすることができる．このような操作を繰り返せば，ついにバンド幅は 1 にまで減少し，3 重対角行列に変換することができる．図 BTRID-1 に，n=10, h=2 のときの $a_{h+1,1}$, すなわち a_{31}（図中の*印) を消去するときに，順次発生する非零要素（図中×印) 及びこれらを消去するための直交相似変換の影響線を示す．

バンド幅 h の行列の最外部の副対角線を消去するのに必要な乗算数はおおよそ $4n^2$ であり，したがって 3 重対角化に必要な乗算数はほぼ $4hn^2$ である．(参考文献 [12] pp.567-569 参照）．これに対して n 次の実対称行列をハウスホルダー法で 3 重対角化するのに必要な乗算数は約 $2n^3/3$ である．したがって，r=h/n<1/6 の場合，乗算数に関してルティスハウザー法が有利になる．

もとのバンド行列 $\boldsymbol{A}$ を 3 重対角行列 $\boldsymbol{T}$ に変換するための直交行列 $\boldsymbol{Q}_s$ ($\boldsymbol{Q}_s\boldsymbol{A}\boldsymbol{Q}_s^T$=$\boldsymbol{T}$) は，$\boldsymbol{R}_1$, $\boldsymbol{R}_2$ のような直交行列の積として $\boldsymbol{Q}_s$=$\boldsymbol{R}_s$...$\boldsymbol{R}_2\boldsymbol{R}_1$ として表されるが，この $\boldsymbol{Q}_s$ を $n\times n$ 行列として作るのは，記憶容量と計算量の点から見て得策ではないので，本サブルーチンでは行わない．

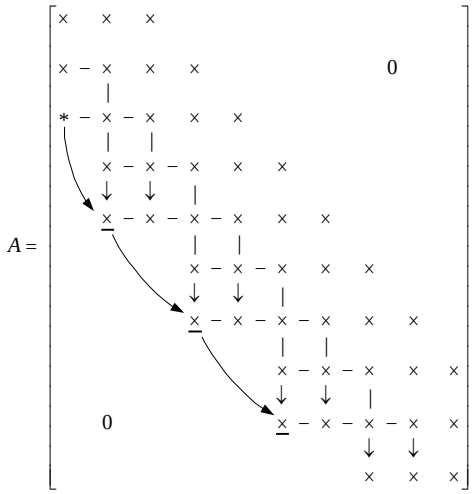


図 BTRID-1 ルティスハウザー・シュワルツ法による要素の消去

I11-81-1101　　BYN, DBYN

第 2 種整数次ベッセル関数 $Y_n(x)$
CALL BYN(X,N,BY,ICON)

(1)　機能

第 2 種整数次ベッセル関数 $Y_n(x)$ の値

$$Y_n(x)=\frac{2}{\pi}\left[J_n(x)\{\log(x/2)+\gamma\}\right]-\frac{1}{\pi}\sum_{k=0}^{n-1}\frac{(n-k-1)!}{k!}(x/2)^{2k-n}-\frac{1}{\pi}\sum_{k=0}^{\infty}\frac{(-1)^k}{k!(n+k)!}\cdot\left\{(x/2)^{2k+n}\left(\sum_{m=1}^{k}1/m+\sum_{m=1}^{n+k}1/m\right)\right\}$$

を漸化式を用いて計算する．ただし $x>0$ であること．

ただし, $J_n(x)$: 第 1 種整数次ベッセル関数, $\sum_{m=1}^{0}1/m=0$,

γ: オイラー定数.

(2)　パラメタ

X …… **入力**．独立変数 x.

N …… **入力**．$Y_n(x)$ の次数 n.

BY …… **出力**．関数値 $Y_n(x)$.

ICON …… **出力**．コンディションコード.

表 BYN-1 参照.

ただし，N=0 又は N=1 のときはそれぞれ BY0, BY1 の ICON に準ずる.

表 BYN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X≥t_{max}であった.	BY = 0.0とする.
30000	X≤0 であった.	BY = 0.0とする.

(3)　使用上の注意

a.　使用する副プログラム

①　SSL II ... BY0, BY1, UBJ0, UBJ1, MGSSL, UTLIM

②　FORTRAN 基本関数 ... IABS, FLOAT, DSIN, DCOS, DLOG, DSQRT

b.　注意

①　引数 X の範囲は

$$0<X<t_{max}$$

であること.

X が大きくなると，初期値 $Y_0(x)$, $Y_1(x)$ の近似式における $\sin(x-\pi/4)$, $\cos(x-\pi/4)$ の値が正確に計算できなくなる.

これを本サブルーチン内で前もって防ぐためである．（BY0, BY1 手法概要 (4.4) 参照）

②　$Y_o(x)$及び$Y_1(x)$ を計算するには，本サブルーチンよりもそれぞれ，BY0, BY1 を直接用いた方がよい.

c.　使用例

$Y_n(x)$ の値を x が 1 から 10，N が 20 から 30 まで 1 おきに計算し数表を作る.

```
C       **EXAMPLE**
        WRITE(6,600)
        DO 20 N=20,30
        DO 10 K=1,10
        X=K
        CALL BYN(X,N,BY,ICON)
        IF(ICON.EQ.0) WRITE(6,610) X,N,BY
        IF(ICON.NE.0) WRITE(6,620) X,N,BY,
       *                           ICON
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',5X,'N',8X,
     *'YN(X)'/)
  610 FORMAT(' ',F8.2,I5,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'N=',I5,5X,'BY=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4)　手法概要

ベッセル関数 $Y_n(x)$ を次の漸化式関係を用いて計算する.

$$Y_{k+1}(x)=\frac{2k}{x}Y_k(x)-Y_{k-1}(x), k=1,2,...,n-1 \qquad (4.1)$$

ただし，$Y_0(x)$, $Y_1(x)$ は，それぞれ BY0, BY1 を利用している.

I11-83-0201　　BYR, DBYR

第 2 種実数次ベッセル関数 $Y_\nu(x)$
CALL BYR(X,V,BY,ICON)

(1)　機能

x>0 かつ ν≥0 に対して，第 2 種実数次ベッセル関数 $Y_\nu(x)$

$$Y_\nu(x)=\frac{J_\nu(x)\cos(\nu\pi)-J_{-\nu}(x)}{\sin(\nu\pi)}$$

の値を，級数展開の変形による方法及び τ 法を用いて計算する．上式において，$J_\nu(x)$ は第 1 種ベッセル関数である．

(2)　パラメタ

X…… **入力**．独立変数 x.
V…… **入力**．$Y_\nu(x)$ の次数 ν.
BY …… **出力**．関数値 $Y_\nu(x)$.
ICON …… **出力**．コンディションコード.
　　表 BYR-1 参照.

表 BYR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	□ X=0.0であった．又は，BY がオーバフローするほど大きな値となった. □ X≥t_{max}であった.	BY は浮動小数点数の負の最大値を出力する. BY = 0.0 とする.
30000	X<0.0 又は V<0.0 であった.	BY = 0.0 とする.

(3)　使用上の注意

a.　使用する副プログラム
① SSL II ... AMACH, AFMAX, MGSSL, ULMAX, UTLIM
② FORTRAN 基本関数 ... FLOAT, ALOG, AMAX1, ALGAMA, ABS, GAMMA, SQRT, SIN, COS

b.　注意
① X>0.0 及び ***V***≥0.0 であること.
② $Y_0(\boldsymbol{x})$ あるいは $Y_1(\boldsymbol{x})$ を計算するときは，本サブルーチンよりも，それぞれ，BY0, BY1 を直接用いた方が効率が良い.
③ $Y_\nu(\boldsymbol{x}), Y_{\nu+1}(\boldsymbol{x}), Y_{\nu+2}(\boldsymbol{x}), \ldots Y_{\nu+M}(\boldsymbol{x})$ の値をいっせいに必要とするときには，本サブルーチンにより，$Y_\nu(\boldsymbol{x})$ と $Y_{\nu+1}(\boldsymbol{x})$ を求め，漸化式（手法概要の (4.2)）を反復適用して，$Y_{\nu+2}(\boldsymbol{x}), Y_{\nu+3}(\boldsymbol{x}), \ldots, Y_{\nu+M}(\boldsymbol{x})$ と次数の高いものへと順次求めていくとよい.
④ x が大きいところでは，続けて同じ ν の値で，本サブルーチンが呼ばれたときには，計算の共通部分を通過し，能率良く $Y_\nu(\boldsymbol{x})$ の値を計算できるようにしてある.

c.　使用例
$Y_\nu(\boldsymbol{x})$ の値を，ν を 0.5 から 0.8 まで 0.01 おきに，$\boldsymbol{x}$ を 1 から 10 まで 1 きざみで計算する.

```
C     **EXAMPLE**
      DO 20 NV=50,80
      V=FLOAT(NV)/100.0
      DO 10 K=1,10
      X=FLOAT(K)
      CALL BYR(X,V,BY,ICON)
      WRITE(6,600) V,X,BY,ICON
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT(' ',F8.3,F8.2,E17.7,I7)
      END
```

(4)　手法概要

ν 次の第 2 種ベッセル関数 $Y_\nu(x)$ は，第 1 種ベッセル関数 $J_\nu(x)$ と $J_{-\nu}(x)$ を用いて，

$$Y_\nu(x)=\frac{J_\nu(x)\cos(\nu\pi)-J_{-\nu}(x)}{\sin(\nu\pi)} \tag{4.1}$$

として定義される．ただし，次数 ν が整数 n のときには，$\nu\to n$ の極限として与えられる.

本計算法は，0≤ν≤2.5 では，直接に $Y_\nu(x)$ の値を計算する．ν>2.5 では，ν の小数部分を μ とすると，μ≤0.5 ならば $Y_{\mu+1}(x)$ と $Y_{\mu+2}(x)$ を，μ>0.5 ならば $Y_\mu(x)$ と $Y_{\mu+1}(x)$ を直接求め，漸化式

$$Y_{\nu+1}(x)=\frac{2\nu}{x}Y_\nu(x)-Y_{\nu-1}(x) \tag{4.2}$$

により $Y_\nu(x)$ の値を計算する.

以下に，0≤ν≤2.5 での $Y_\nu(x)$ の計算法について説明する．この $Y_\nu(x)$ の計算は x により，方法が異なる.

x≤3.66　　(4.3)

では，$\boldsymbol{x}$ が小さい場合の計算法，

x>3.66　　(4.4)

では，x が大きい場合の計算法を用いる.

a.　x が小さい場合の計算法
本方法は，次式で与えられる $J_\nu(x)$ 及び $J_{-\nu}(x)$ の級数展開を利用する.

$$J_\nu(x)=\left(\frac{x}{2}\right)^\nu\left\{\frac{1}{\Gamma(1+\nu)}+\frac{-\frac{1}{4}x^2}{1!\,\Gamma(2+\nu)}+\frac{\left(-\frac{1}{4}x^2\right)^2}{2!\,\Gamma(3+\nu)}+\cdots\cdots\right\} \tag{4.5}$$

$$J_{-\nu}(x)=\left(\frac{x}{2}\right)^{-\nu}\left\{\frac{1}{\Gamma(1-\nu)}+\frac{-\frac{1}{4}x^2}{1!\Gamma(2-\nu)}\right.$$
$$\left.+\frac{\left(-\frac{1}{4}x^2\right)^2}{2!\Gamma(3-\nu)}+.....\right\} \tag{4.6}$$

以下で用いる$\phi_1(\nu,x)$ 及び$\phi_2(\nu,x)$ を次のように定義する.

$$\left.\begin{aligned}\phi_1(\nu,x)&=\frac{\left(\frac{x}{2}\right)^{-\nu}-1}{\nu}\\ \phi_2(\nu,x)&=\frac{1-\left(\frac{x}{2}\right)^{\nu}}{\nu}\end{aligned}\right\} \tag{4.7}$$

ν の区間, $0\le\nu\le0.5$, $0.5<\nu\le1.5$, $1.5<\nu\le2.5$ によって計算式が異なる. ここでは, $0\le\nu\le0.5$ での計算法を説明する.
(4.5), (4.6) 及び (4.7) から, 次式が得られる.

$$J_\nu(x)\cos(\nu\pi)-J_{-\nu}(x)=\nu\sum_{k=0}^{\infty}\{A_k(\nu,x)$$
$$+B_k(\nu,x)\} \tag{4.8}$$

ただし,

$$\left.\begin{aligned}A_k(\nu,x)&=-\left(-\frac{1}{4}x^2\right)^k\\ &\cdot\left\{\frac{1}{k!\,\nu}\left(\frac{1}{\Gamma(k+1-\nu)}-\frac{\cos(\nu\pi)}{\Gamma(k+1-\nu)}\right)\right\}\\ B_k(\nu,x)&=-\left(-\frac{1}{4}x^2\right)^k\\ &\cdot\left\{\frac{1}{k!}\left(\frac{\phi_1(\nu,x)}{\Gamma(k+1-\nu)}+\frac{\phi_2(\nu,x)\cos(\nu\pi)}{\Gamma(k+1+\nu)}\right)\right\}\end{aligned}\right\} \tag{4.9}$$

である. 上式の $A_k(\nu,x)$ 及び $B_k(\nu,x)$ は, そのまま, 式のとおりに計算を行うと, $\nu\approx0$ で桁落ちが生ずる. 桁落ちを避けるために, 以下で述べる工夫を行う.

まず, $B_k(\nu,x)$ において, $\phi_1(\nu,x)$ と$\phi_2(\nu,x)$ は同符号であるので, その加算では桁落ちは無い. しかし, (4.7) で定義されている$\phi_1(\nu,x)$ 及び$\phi_2(\nu,x)$ は, $(x/2)^\nu$が 1 に近いとき, その右辺のとおりに計算すると桁落ちを生ずる. 2 進丸め誤差程度で $\phi_1(\nu,x)$ 及び$\phi_2(\nu,x)$ を求めるためには, 関数

$$f(t)=\frac{e^t-1}{t}=\frac{1}{1!}+\frac{t}{2!}+\frac{t^2}{3!}+..... \tag{4.10}$$

について, $-\log2\le t\le\log2$ において所要の相対精度を有する最良近似式があればよい. なぜならば, その最良近似式により, $\phi_1(\nu,x)$ 及び $\phi_2(\nu,x)$ を

$$\left.\begin{aligned}\phi_1(\nu,x)&=-f\left(-\nu\log\left(\frac{x}{2}\right)\right)\log\left(\frac{x}{2}\right)\\ \phi_2(\nu,x)&=-f\left(\nu\log\left(\frac{x}{2}\right)\right)\log\left(\frac{x}{2}\right)\end{aligned}\right\} \tag{4.11}$$

として精度良く計算できるからである. 本サブルーチンでは, $f(t)$ に対して, 次の形の最良近似式を用いてる.

$$f(t)\approx\frac{2\sum_{k=0}^{M}p_kt^{2k}}{\sum_{k=0}^{N}q_kt^{2k}-t\sum_{k=0}^{M}p_kt^{2k}} \tag{4.12}$$

(4.9) の $A_k(\nu,x)$ においても, $(1/\Gamma(k+1-\nu)-\cos(\nu\pi)/\Gamma(k+1+\nu))/(k!\,\nu)$ の値を桁落ち無しに計算するためには, それに対して所要の精度を有する最良近似式があればよい. ただし, (4.9) の $A_k(\nu,x)$ の { }の部分を $\tilde{A}_k(\nu)$と表わしたとき, $\tilde{A}_k(\nu)$に対して

$$\tilde{A}_k(\nu)=\frac{1}{(k+\nu)(k-\nu)}\left[\tilde{A}_{k-1}(\nu)+\frac{1}{k!}\left\{\frac{1}{\Gamma(k-\nu)}+\frac{\cos(\nu\pi)}{\Gamma(k+\nu)}\right\}\right]$$
$$(k\ge1) \tag{4.13}$$

なる関係が成り立つので, $0\le\nu\le0.5$ における $\tilde{A}_0(\nu)$ の近似式があればよい. 本サブルーチンでは,

$$\tilde{A}_0(\nu)\approx(\nu-\nu_0)\sum_{k=0}^{M}p_k\nu^k \tag{4.14}$$

の形の最良近似式によって計算を行っている.

このようにして, $A_k(\nu,x)$ 及び $B_k(\nu,x)$ は桁落ち無しに求めることができるが, これらのものは, すべては, 同符号ではない. したがって, (4.8) の加算において, 桁落ちの可能性がある. しかし, いま, 求めようとしている $Y_\nu(x)$ は, ν を固定したとき, x に対して振動的な関数であり, かつ, 零点を有する. したがって, このような関数に対しては, 相対精度でその間数値を求めることは, 原理的に不可能である. 絶対精度でもって議論しなければならない (x が小さく, $Y_\nu(x)$ の値が負の大きな場合を除いて). 原理的に可能な絶対精度で $Y_\nu(x)$ を計算できる x の範囲を調べた結果として,

$x \le 3.66$ (4.15)

なる範囲で，本方法を用いることができることがわかった．(4.8) の和を構成している項は，k が大きくなるにつれて十分に小さくなるので，所要精度内に収束するまでに取る項は少なくて済む．
したがって，

$$g(\nu) = \frac{\sin(\nu\pi)}{\nu} \tag{4.16}$$

なる関数に対する次の形の最良近似式

$$g(\nu) \approx \sum_{k=0}^{M} P_k \nu^{2k} \tag{4.17}$$

を用いれば，$Y_\nu(x)$ の値は

$$Y_\nu(x) = \frac{J_\nu(x)\cos(\nu\pi) - J_{-\nu}(x)}{\sin(\nu\pi)} = \frac{\sum_{k=0}^{\infty}\{A_k(\nu,x) + B_k(\nu,x)\}}{g(\nu)} \tag{4.18}$$

により計算することができる．
$\nu = 0$ のときには，(4.18) により $Y(x)$ を計算するよりは，むしろ，(4.1) の $\nu \to 0$ の極限として得られる．

$$Y_0(x) = \frac{2}{\pi}\sum_{k=0}^{\infty}\frac{\left(-\frac{1}{4}x^2\right)^k}{(k!)^2}\left\{\gamma + \log\left(\frac{x}{2}\right) - \Phi_k\right\} \tag{4.19}$$

により特別に計算すると能率がよい．ただし，γ はオイラーの定数，$\Phi_0 = 1$, $\Phi_k = \sum_{m=0}^{k}\frac{1}{m}(k \ge 1)$ である．
以上において，$0 \le \nu \le 0.5$ での計算法を述べた．$0.5 < \nu \le 1.5$ 及び $1.5 < \nu \le 2.5$ の場合の計算については省略するが，同様な方法を考えることができる．

b. x が大きい場合の計算法
第 2 種ベッセル関数 $Y_\nu(x)$ は，第 1 種ハンケル関数 $H_\nu^{(1)}(x)$

$$H_\nu^{(1)}(x) = J_\nu(x) + iY_\nu(x) \tag{4.20}$$

の虚部として与えられる．ただし，i は虚数の単位 $\left(i = \sqrt{-1}\right)$, $J_\nu(x)$ は第 1 種ベッセル関数である．ここで，$f_\nu\left(\frac{1}{x}\right)$ を次式にて定義する．

$$f_\nu\left(\frac{1}{x}\right) = \left(\frac{\pi x}{2}\right)^{\frac{1}{2}} e^{-i\left(x - \frac{1}{2}\nu\pi - \frac{1}{4}\pi\right)} H_\nu^{(1)}(x) \tag{4.21}$$

$f_\nu\left(\frac{1}{x}\right)$ は複素関数であり，その実部を $P\left(\nu, \frac{1}{x}\right)$, 虚部を $Q\left(\nu, \frac{1}{x}\right)$ とすると $f_\nu\left(\frac{1}{x}\right)$ は，

$$f_\nu\left(\frac{1}{x}\right) = P\left(\nu, \frac{1}{x}\right) + iQ\left(\nu, \frac{1}{x}\right) \tag{4.22}$$

と表される．(4.20), (4.21) 及び (4.22) を用いて，$Y_\nu(x)$ は

$$Y_\nu(x) = \left(\frac{2}{\pi x}\right)^{\frac{1}{2}}\left\{P\left(\nu, \frac{1}{x}\right)\sin\left(x - \frac{1}{2}\nu\pi - \frac{1}{4}\pi\right) + Q\left(\nu, \frac{1}{x}\right)\cos\left(x - \frac{1}{2}\nu\pi - \frac{1}{4}\pi\right)\right\} \tag{4.23}$$

と表すことができる．
以下では，(4.21) の $f_\nu\left(\frac{1}{x}\right)$ の近似式を求めることにする．この近次式が得られれば，(4.22) により，その実部，虚部として，それぞれ，$P\left(\nu, \frac{1}{x}\right)$ 及び $Q\left(\nu, \frac{1}{x}\right)$ の値が求められ，(4.23) により，$Y_\nu(x)$ の値が求められる．
第 1 種ハンケル関数 $H_\nu^{(1)}(x)$ は，次の微分方程式

$$x^2\frac{d^2w}{dx^2} + x\frac{dw}{dx} + \left(x^2 - \nu^2\right)w = 0 \tag{4.24}$$

を満足する．以下，$t = \frac{1}{x}$ とする．(4.21) の $H_\nu^{(1)}(x)$ を (4.24) に代入することにより，$f_\nu\left(\frac{1}{x}\right)$ すなわち $f_\nu(t)$ は次の微分方程式を満足することが分かる．

$$t^2 f_\nu''(t) + 2(t - i) f(t) + \left(\frac{1}{4} - \nu^2\right) f_\nu(t) = 0 \tag{4.25}$$

上の微分方程式に τ 法を適用して，t が小さいときの，$f_\nu(t)$ の近似式を求めることにする．そこで，上式の右辺に，直交区間 $[0, \eta]$ のずらし超球多項式 (shifted Ultraspherical Polynomial) を τ 倍したものを付加した次式の形の微分方程式を考える．

$$t^2 f_{\nu m}''(t) + 2(t - i) f_{\nu m}'(t) + \left(\frac{1}{4} - \nu^2\right) f_{\nu m}(t) = \tau C_m^{*(\alpha)}\left(\frac{t}{\eta}\right) \tag{4.26}$$

上式は，次の形の m 次の多項式を特解としてもつ．

$$f_{\nu m}(t) = -\tau\sum_{k=0}^{m}\frac{C_{mk}^{*(a)}\sum_{l=0}^{k}a_l t^l}{2(k+1)a_{k+1}\eta^k} \tag{4.27}$$

ただし，

$$\left.\begin{aligned} a_0 &= 1 \\ a_l &= i^l \frac{\left(4\nu^2-1^2\right)\left(4\nu^2-3^2\right).....\left(4\nu^2-(2l-1)^2\right)}{l!8^l} \end{aligned}\right\} \tag{4.28}$$

であり，$C_{mk}^{*(a)}$は，$C_m^{*(a)}(t)$のk次の係数である．(4.26)の右辺のτが十分に小さいならば，この$f_{\nu m}(t)$は，(4.25)の$f_\nu(t)$に対する近似多項式と考えることができるであろう．初期条件$f_{\nu m}(0)=1(t\to 0$のとき，$f_\nu(t)\to 1)$から，τを決めると（τはmを増すと小さくなる），$0\le t\le\eta$における$f_\nu(t)$に対する近似多項式$f_{\nu m}(t)$として，

$$f_{\nu m}(t)=\frac{\sum_{k=0}^{m}\frac{C_{mk}^{*(\alpha)}\sum_{l=0}^{k}a_l t^l}{(k+1)a_{k+1}\eta^k}}{\sum_{k=0}^{m}\frac{C_{mk}^{*(\alpha)}}{(k+1)a_{k+1}\eta^k}} \tag{4.29}$$

が得られる．
上式の$f_{\nu m}(t)$の精度に関する数値実験の結果から，a=1と置くことにする．
$P\left(\nu,\frac{1}{x}\right)$及び$Q\left(\nu,\frac{1}{x}\right)$は，それぞれ，$f_\nu\left(\frac{1}{x}\right)$の実部及び虚部であるので，(4.29)を用いると，次式のように表される．

$$\begin{aligned} \begin{Bmatrix} P\left(\nu,\frac{1}{x}\right) \\ Q\left(\nu,\frac{1}{x}\right) \end{Bmatrix} &\approx \begin{Bmatrix} \mathrm{Re} \\ \mathrm{Im} \end{Bmatrix} \left(\frac{\sum_{k=0}^{m}\frac{C_{mk}^{*(1)}\sum_{l=0}^{k}a_l\left(\frac{1}{x}\right)^l}{(k+1)a_{k+1}\eta^k}}{\sum_{k=0}^{m}\frac{C_{mk}^{*(1)}}{(k+1)a_{k+1}\eta^k}} \right) \\ &= \begin{Bmatrix} \mathrm{Re} \\ \mathrm{Im} \end{Bmatrix} \left(\frac{\sum_{l=0}^{m}a_l\left(\frac{1}{x}\right)^l\sum_{k=l}^{m}\frac{C_{mk}^{*(1)}}{(k+1)a_{k+1}\eta^k}}{\sum_{k=0}^{m}\frac{C_{mk}^{*(1)}}{(k+1)a_{k+1}\eta^k}} \right) \end{aligned} \tag{4.30}$$

上式を変形すれば，$P\left(\nu,\frac{1}{x}\right)$及び$Q\left(\nu,\frac{1}{x}\right)$に対する能率的な計算として，

$$\begin{Bmatrix} P\left(\nu,\frac{1}{x}\right) \\ Q\left(\nu,\frac{1}{x}\right) \end{Bmatrix} \approx \begin{Bmatrix} \mathrm{Re} \\ \mathrm{Im} \end{Bmatrix} \left(\frac{\sqrt{\frac{\pi}{2}}\sum_{l=0}^{m}\left(\frac{1}{x}\right)^l(-i)^l\frac{d_l}{W_l}\sum_{k=l}^{m}i^k W_{k+1}}{\sum_{k=0}^{m}i^k W_{k+1}} \right) \tag{4.31}$$

が得られる．ただし，

$$\left.\begin{aligned} W_0 &= 1 \\ W_k &= \prod_{n=0}^{k-1}\frac{e_n}{\left(n+\frac{1}{2}\right)^2-\nu^2} \qquad (k\ge 1) \end{aligned}\right\} \tag{4.32}$$

であり，d_l及び上式のe_nは，

$$\left.\begin{aligned} d_0 &= \sqrt{\frac{2}{\pi}} \\ d_l &= \sqrt{\frac{2}{\pi}}\,\frac{C_{m,l-1}^{*(1)}}{\eta^{l-1}l} \qquad (l\ge 1) \end{aligned}\right\} \tag{4.33}$$

$$\left.\begin{aligned} e_0 &= 2C_{m0}^{*(1)} \\ e_n &= \frac{2n\,C_{mn}^{*(1)}}{\eta\,C_{m,n-1}^{*(1)}} \qquad (n\ge 1) \end{aligned}\right\} \tag{4.34}$$

である．
最初のところで述べたように，本方法は，x>3.66で用いるので，η=1/3.66と置き，mを所要の精度を得るための最小の値に選べば，計算式を決定できる．ただし，ηが小さければ小さいほど，所要の精度を得るための最小のmの値は小さくなるので，xが大きいときの効率の改善のために，次のように，xの区間によって，η及びmを変えるとよい．結果として，単精度の場合には，

$$\left.\begin{aligned} 3.66<x<10\text{では，}\ \eta=\frac{1}{3.66},m=8 \\ x\ge 10\text{では，}\ \eta=\frac{1}{10},m=5 \end{aligned}\right\} \tag{4.35}$$

とし，倍精度の場合には，

$$\left.\begin{aligned} 3.66<x<10\text{では，}\ \eta=\frac{1}{3.66},m=24 \\ x\ge 10\text{では，}\ \eta=\frac{1}{10},m=15 \end{aligned}\right\} \tag{4.36}$$

とすると，$P\left(\nu,\frac{1}{x}\right)$及び$Q\left(\nu,\frac{1}{x}\right)$を能率的に計算できることが分かっている．本サブルーチン内では，定数d_l及びe_nは，あらかじめ計算されているものをデータ文により表としてもっている．

I11-81-0401　　BY0, DBY0

第 2 種 0 次ベッセル関数 $Y_0(x)$
CALL BY0(X,BY,ICON)

(1)　機能

第 2 種 0 次ベッセル関数 $Y_0(x)$ の値

$$Y_0(x)=\frac{2}{\pi}\left[J_0(x)\left\{\log(x/2)+\gamma\right\}-\sum_{k=1}^{\infty}\frac{(-1)^k(x/2)^{2k}}{(k!)^2}\cdot\sum_{m=1}^{k}1/m\right]$$

（ただし，$J_0(x)$: 第 1 種 0 次ベッセル関数，γ: オイラー定数）を有理式及び，漸近形式の近似式を用いて計算する．ただし，x>0 であること．

(2)　　パラメタ

X…… **入力**．独立変数 x.
BY …… **出力**．関数値 $Y_0(x)$.
ICON…… **出力**．コンディションコード．
表 BY0-1 参照.

表 BY0-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X≥t_{max} であった.	BY = 0.0 とする.
30000	X≤0 であった.	BY = 0.0 とする.

(3)　使用上の注意

a.　使用する副プログラム
① SSL II...UBJ0,MGSSL, UTLIM
② FORTRAN 基本関数...DSIN, DCOS, DLOG, DSQRT

b.　注意
① 引数 X の範囲は
0<X<t_{max}
であること.
X が大きくなると sin(x-π/4), cos(x-π/4) の値が不正確になるためである（手法概要 (4.4) 参照）.

c.　使用例
$Y_0(x)$ の値を x が 1 から 100 まで，1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,100
      X=K
      CALL BY0(X,BY,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,BY
      IF(ICON.NE.0) WRITE(6,620) X,BY,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF BESSEL ',
     *'FUNCTION'///6X,'X',9X,'Y0(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'BY=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4)　手法概要

ベッセル関数 $Y_0(x)$ の計算は，x=8 を境にして，近似式の形が異なる.

a.　0<x≤8 の場合
$Y_0(x)$ のべき級数展開

$$Y_0(x)=\frac{2}{\pi}\left[J_0(x)\left\{\log(x/2)+\gamma\right\}-\sum_{k=1}^{\infty}\frac{(-1)^k(x/2)^{2k}}{(k!)^2}\sum_{m=1}^{k}1/m\right] \tag{4.1}$$

（ただし，$J_0(x)$: 第 1 種 0 次ベッセル関数，γ: オイラー定数）を次の有理近似式を用いて計算する.

単精度 :

$$Y_0(x)=\sum_{k=0}^{5}a_kx^{2k}\Big/\sum_{k=0}^{5}b_kx^{2k}+\frac{2}{\pi}J_0(x)\log(x) \tag{4.2}$$

理論精度 8.14 桁

倍精度 :

$$Y_0(x)=\sum_{k=0}^{8}a_kx^{2k}\Big/\sum_{k=0}^{8}b_kx^{2k}+\frac{2}{\pi}J_0(x)\log(x) \tag{4.3}$$

理論精度 18.78 桁

b.　x>8 の場合 $Y_0(x)$ の漸近展開

$$Y_0(x)=\sqrt{\frac{2}{\pi x}}\left\{P_0(x)\sin(x-\pi/4)+Q_0(x)\cos(x-\pi/4)\right\} \tag{4.4}$$

より計算する．ただし，$P_0(x)$, $Q_0(x)$ は次の近似式を用いる.

単精度 :

$$P_0(x)=\sum_{k=0}^{2}a_kz^{2k}\Big/\sum_{k=0}^{2}b_kz^{2k},\ z=8/x \tag{4.5}$$

理論精度 10.66 桁

$$Q_0(x)=\sum_{k=0}^{1}c_kz^{2k+1}\Big/\sum_{k=0}^{2}d_kz^{2k},z=8/x \tag{4.6}$$

理論精度 9.58 桁

倍精度 :

$$P_0(x)=\sum_{k=0}^{5}a_k z^{2k}\Big/\sum_{k=0}^{5}b_k z^{2k},z=8/x \qquad (4.7)$$

理論精度 18.16 桁

$$Q_0(x)=\sum_{k=0}^{5}c_k z^{2k+1}\Big/\sum_{k=0}^{5}d_k z^{2k},z=8/x \qquad (4.8)$$

理論精度 18.33 桁

なお，詳細は参考文献 [78] の pp.141 – 149 を参照のこと.

I11-81-0501　BY1, DBY1

第 2 種 1 次ベッセル関数 $Y_1(x)$
CALL BY1(X,BY,ICON)

(1)　機能

第 2 種 1 次ベッセル関数 $Y_1(x)$ の値

$$Y_1(x)=2/\pi\{J_1(x)(\log(x/2)+\gamma)-1/x\}$$
$$-1/\pi\left\{\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k+1}}{k!(k+1)!}\left(\sum_{m=1}^{k}1/m+\sum_{m=1}^{k+1}1/m\right)\right\}$$

（ただし $J_1(x)$: 第1種1次ベッセル関数, γ: オイラー定数）を有理式，及び漸近形式の近似式を用いて計算する.

ただし $x>0$ であること.

(2)　パラメタ

X…… **入力**. 独立変数 x.
BY …… **出力**. 関数値 $Y_1(x)$.
ICON…… **出力**. コンディションコード.
表 BY1-1 参照.

表 BY1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$X\geq t_{max}$ であった.	BY=0.0 とする.
30000	X≤0 であった.	BY=0.0 とする.

(3)　使用上の注意

a.　使用する副プログラム
① SSL II ... UBJ1, MGSSL, UTLIM
② FORTRAN 基本関数 ... DSIN, DCOS, DLOG, DSQRT

b.　注意
① 引数 X の範囲は
$0<X<t_{max}$
であること.
X が大きくなると sin(x-3π/4), cos(x-3π/4) の値が不正確になるためである（手法概要 (4.4) 参照）.

c.　使用例
$Y_1(x)$ の値を x が 1 から 100 まで, 1 おきに計算し数表を作る.

```
C       **EXAMPLE**
        WRITE(6,600)
        DO 10 K=1,100
        X=K
        CALL BY1(X,BY,ICON)
        IF(ICON.EQ.0) WRITE(6,610) X,BY
        IF(ICON.NE.0) WRITE(6,620) X,BY,ICON
     10 CONTINUE
        STOP
    600 FORMAT('1','EXAMPLE OF BESSEL ',
       *'FUNCTION'///6X,'X',9X,'Y1(X)'/)
    610 FORMAT(' ',F8.2, E17.7)
    620 FORMAT(' ','** ERROR **',5X,'X=',
       *E17.7,5X,'BY=',E17.7,5X,
       *'CONDITION=',I10)
        END
```

(4)　手法概要

ベッセル関数 $Y_1(x)$ の計算は，x=8 を境にして，近似式の形が異なる.

a.　$0<x\leq 8$ の場合
$Y_1(x)$ のべき級数展開

$$Y_1(x)=2/\pi\{J_1(x)(\log(x/2)+\gamma)-1/x\}$$
$$-1/\pi\left\{\sum_{k=0}^{\infty}\frac{(-1)^k(x/2)^{2k+1}}{k!(k+1)!}\left(\sum_{m=1}^{k}1/m+\sum_{m=1}^{k+1}1/m\right)\right\} \quad (4.1)$$

（ただし $J_1(x)$: 第 1 種 1 次ベッセル関数, γ: オイラー定数）を次の有理近似式を用いて計算する.

単精度 :

$$Y_1(x)=\sum_{k=0}^{7}a_k x^{2k+1}/\sum_{k=0}^{2}b_k x^{2k}+\frac{2}{\pi}\{J_1(x)\log(x)-1/x\} \quad (4.2)$$

理論精度 8.96 桁

倍精度 :

$$Y_1(x)=\sum_{k=0}^{7}a_k x^{2k+1}/\sum_{k=0}^{8}b_k x^{2k}+\frac{2}{\pi}\{J_1(x)\log(x)-1/x\} \quad (4.3)$$

理論精度 18.24 桁

b.　$x>8$ の場合
$Y_1(x)$ の漸近展開

$$Y_1(x)=\sqrt{\frac{2}{\pi x}}\{P_1(x)\sin(x-3\pi/4)+Q_1(x)\cos(x-3\pi/4)\} \quad (4.4)$$

より計算する．ただし，$P_1(x)$, $Q_1(x)$ は次の近似式を用いる.

単精度 :

$$P_1(x)=\sum_{k=0}^{2}a_k z^{2k}/\sum_{k=0}^{2}b_k z^{2k},z=8/x \quad (4.5)$$

理論精度 10.58 桁

$$Q_1(x)=\sum_{k=0}^{1}c_k z^{2k+1}\Big/\sum_{k=0}^{2}d_k z^{2k},z=8/x \quad (4.6)$$

理論精度 9.48 桁

倍精度 :

$$P_1(x)=\sum_{k=0}^{5}a_k z^{2k}\Big/\sum_{k=0}^{5}b_k z^{2k},z=8/x \qquad (4.7)$$

理論精度 18.11 桁

$$Q_1(x)=\sum_{k=0}^{5}c_k z^{2k+1}\Big/\sum_{k=0}^{5}d_k z^{2k},z=8/x \qquad (4.8)$$

理論精度 18.28 桁

なお，詳細は参考文献 [78] の pp.141 – 149 を参照のこと.

I11-82-1101　　CBIN, DCBIN

複素変数第1種整数次変形ベッセル関数$I_n(z)$
CALL CBIN(Z,N,ZBI,ICON)

(1)　機能

複素変数 z の第1種整数次変形ベッセル関数$I_n(z)$

$$I_n(z)=\left(\frac{1}{2}z\right)^n\sum_{k=0}^{\infty}\frac{\left(\frac{1}{4}z^2\right)^k}{k!(n+k)!}$$

の値をべき級数展開式（上式）及び漸化式による方法を用いて計算する.

(2)　パラメタ

Z …… **入力**. 独立変数z. 複素数型変数.
N …… **入力**. $I_n(z)$ の次数n .
ZBI …… **出力**. 関数値$I_n(z)$. 複素数型変数.
ICON …… **出力**. コンディションコード.
表CBIN-1参照.

表CBIN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	|Re(z)| > log(fl_{max}) 又は, |Im(z)| > log(fl_{max}) であった.	ZBI=(0.0,0.0)とする.

(3)　使用上の注意

a.　使用する副プログラム
① SSL II ... AMACH, MASSL, ULMAX
② FORTRAN 基本関数 ... REAL, AIMAG, ABS, IABS, FLOAT, CEXP, MAX0

b.　注意
① $|\mathrm{Re}(z)| \le \log(fl_{max})$ 及び $|\mathrm{Im}(z)| \le \log(fl_{max})$ であること.
② $I_n(z)$, $I_{n+1}(z)$, $I_{n+2}(z)$, ..., $I_{n+M}(z)$ の値をいっせいに必要とするときには, 本サブルーチンにより $I_{n+M}(z)$ と $I_{n+M-1}(z)$ を求め, 漸化式を反復適用して, $I_{n+M-2}(z)$, $I_{n+M-3}(z)$, ..., $I_n(z)$ と次数の低いものを求めていくとよい. 逆に, 本サブルーチンにより, $I_n(z)$ と $I_{n+1}(z)$ を求め, 漸化式により $I_{n+2}(z)$, $I_{n+3}(z)$, ..., $I_{n+M}(z)$ と次数の高いものを求めていくことは安定でないので避けなければならない.

c.　使用例
$I_n(z)$ の値を, z=10+5iについて, n を 1 と 2 について計算する.

```
C     **EXAMPLE**
      COMPLEX Z,ZBI
      Z=(10.0,5.0)
      DO 10 N=1,2
      CALL CBIN(Z,N,ZBI,ICON)
      WRITE(6,600) Z,N,ZBI,ICON
   10 CONTINUE
      STOP
  600 FORMAT(' ',2F8.2,I6,5X,2E17.7,I7)
      END
```

(4)　手法概要

求められるべき$I_n(z)$ について, その絶対値がアンダフローする値になることがあらかじめ分かった場合には, 以下の計算は行わず, 結果を (0.0,0.0) とする.

$I_n(z)$ の計算は, zにより方法が異なる.

① $|\mathrm{Re}(z)|+|\mathrm{Im}(z)|\le 1$ の場合,
べき級数展開

$$I_n(z)=\left(\frac{1}{2}z\right)^n\sum_{k=0}^{\infty}\frac{\left(\frac{1}{4}z^2\right)^k}{k!(n+k)!} \qquad (4.1)$$

により, 第 k 項目が初項に比べ丸め誤差の単位以下になるまで計算する.

② $|\mathrm{Re}(z)|+|\mathrm{Im}(z)|>1$ かつ $|\mathrm{Re}(z)|\le\log(fl_{max})$ かつ $|\mathrm{Im}(z)|\le\log(fl_{max})$ の場合, 漸化式による方法を用いて計算する.
mを適当に大きな整数 (z , n及び要求精度によって決まる), δ を適当に小さな定数(10^{-38}) とする.
初期値を,

$$G_{m+1}(z)=0,\ G_m(z)=\delta$$

として, 漸化式,

$$G_{k-1}(z)=\frac{2k}{z}G_k(z)+G_{k+1}(z) \qquad (4.2)$$

を, $k=m,m-1,...,1$ に対して反復適用する.
このとき, $I_n(z)$ は

$$I_n(z)\approx e^z G_n(z)\Big/\left(\sum_{k=0}^{m}\varepsilon_k G_k(z)\right) \qquad (4.3)$$

として求められる. ここで,

$$\varepsilon_k = \begin{cases} 1 & (k=0) \\ 2 & (k \geq 1) \end{cases}$$

である．上式は，Re(z)≧0 で用いることができる．Re(z)<0 では，上式は桁落ちを生ずるので，関係式 $I_n(-z)=(-1)^n I_n(z)$ を用いて Re(z)≧0 の場合に帰着させる．

なお，mの決め方及びその他詳細については，参考文献 [81]，[83]を参照のこと．

I11-82-1301　　CBJN, DCBJN

複素変数第1種整数次ベッセル関数$J_n(z)$
CALL CBJN(Z,N,ZBJ,ICON)

(1)　機能

複素変数 z の第1種整数次ベッセル関数$J_n(z)$

$$J_n(z)=\left(\frac{1}{2}z\right)^n\sum_{k=0}^{\infty}\frac{\left(-\frac{1}{4}z^2\right)^k}{k!(n+k)!}$$

の値をべき級数展開式（上式）及び漸化式による方法を用いて計算する.

(2)　パラメタ

Z …… **入力**. 独立変数z. 複素数型変数.
N …… **入力**. $J_n(z)$ の次数n.
ZBJ …… **出力**. 関数値$J_n(z)$. 複素数型変数.
ICON …… **出力**. コンディションコード.
表CBJN-1参照.

表CBJN-1　コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$\|Re(z)\| > \log(fl_{max})$ 又は, $\|Im(z)\| > \log(fl_{max})$ であった.	ZBJ=(0.0,0.0)とする.

(3)　使用上の注意

a.　使用する副プログラム

① SSL II ... AMACH, MGSSL, ULMAX

② FORTRAN 基本関数 ... REAL, AIMAG, ABS, IABS, MOD, FLOAT, CEXP, CONJG, MAX0, CMPLX

b.　注意

① $|Re(z)|\le\log(fl_{max})$, 及び $|Im(z)|\le\log(fl_{max})$ であること.

② $J_n(z)$, $J_{n+1}(z)$, $J_{n+2}(z)$, ..., $J_{n+M}(z)$ の値をいっせいに必要とするときには, 本サブルーチンにより $J_{n+M}(z)$ と $J_{n+M-1}(z)$ を求め, 漸化式を反復適用して $J_{n+M-2}(z)$, $J_{n+M-3}(z)$, ..., $J_n(z)$ と次数の低いものを求めていくとよい. 逆に, 本サブルーチンにより, $J_n(z)$ と $J_{n+1}(z)$ を求め, 漸化式により, $J_{n+2}(z)$, $J_{n+3}(z)$, ..., $J_{n+M}(z)$ と次数の高いものを求めていくことは不安定が生ずるので避けなければならない.

c.　使用例

$J_n(z)$ の値を, z=10+5iについて, nを1と2について計算する.

```
C     **EXAMPLE**
      COMPLEX Z,ZBJ
      Z=(10.0,5.0)
      DO 10 N=1,2
      CALL CBJN(Z,N,ZBJ,ICON)
      WRITE(6,600) Z,N,ZBJ,ICON
   10 CONTINUE
      STOP
  600 FORMAT(' ',2F8.2,I6,5X,2E17.7,I7)
      END
```

(4)　手法概要

求められるべき$J_n(z)$ について, その絶対値がアンダフローした値になることがあらかじめ分った場合には, 以下の計算は行わず, 結果を (0.0,0.0) とする.

$J_n(z)$ の計算は, z により方法が異なる.

① $|Re(z)|+|Im(z)|\le 1$ の場合

べき級数展開

$$J_n(z)=\left(\frac{1}{2}z\right)^n\sum_{k=0}^{\infty}\frac{\left(-\frac{1}{4}z^2\right)^k}{k!(n+k)!} \qquad (4.1)$$

により, 第 k 項目が初項に比べ丸め誤差の単位以下になるまで計算する.

② $|Re(z)|+|Im(z)|>1$ かつ $|Re(z)|\le\log(fl_{max})$ かつ $|Im(z)|\le\log(fl_{max})$ の場合

漸化式による方法を用いて計算する.

m を適当に大きな整数（z, n 及び要求精度によって決まる）, δを適当に小さな定数（10^{-38}）とする.

初期値を

$$F_{m+1}(z)=0,\ F_m(z)=\delta$$

として, 漸化式

$$F_{k-1}(z)=\frac{2k}{z}F_k(z)-F_{k+1}(z) \qquad (4.2)$$

を, k=m, m-1,...,1 に対して反復適用する. このとき, $J_n(z)$ は

$$J_n(z)\approx e^{-iz}F_n(z)\Big/\left(\sum_{k=0}^{m}\varepsilon_k i^k F_k(z)\right) \qquad (4.3)$$

として求められる. ここで,

$$\varepsilon_k = \begin{cases} 1 & (k=0) \\ 2 & (k \geq 1) \end{cases}$$

である．上式は 0≤arg z≤πで用いることができる． -π<arg z<0 では，上式は桁落ちを生ずるので，関係式 $J_n(\bar{z}) = \overline{J_n(z)}$ を用いて 0≤arg z≤π の場合に帰着させる．

なお，m の決め方及びその他詳細については，参考文献［81］，［83］を参照のこと．

I11-84-0101　　CBJR, DCBJR

複素変数第1種実数次ベッセル関数J_ν(z)
CALL CBJR(Z,V,ZBJ,ICON)

(1)　機能

複素変数zの第1種実数次ベッセル関数$J_\nu(z)$

$$J_\nu(z)=\left(\frac{1}{2}z\right)^\nu\sum_{k=0}^{\infty}\frac{\left(-\frac{1}{4}z^2\right)^k}{k!\,\Gamma(\nu+k+1)}$$

の値をべき級数展開式（上式）及び漸化式による方法を用いて計算する.

上式において，$(1/2\cdot z)^\nu$ の値は，その主値を採用する．$(1/2\cdot z)^\nu\equiv e^{\nu\log(z/2)}$ の主値は，log(z/2) の主値の選び方に依存するが，本サブルーチンではFORTRAN基本関数CLOGによって求められるものとなる．通常は，log(z/2)の主値$=\log|z/2|+i\arg z$, $-\pi<\arg z\le\pi$ である.

(2)　パラメタ

Z …… **入力**．独立変数z（複素変数）.
V …… **入力**．$J_\nu(z)$ の次数$\nu(\nu\ge0)$.
ZBJ …… **出力**．関数値$J_\nu(z)$（複素変数）.
ICON …… **出力**．コンディションコード.
表CBJR-1 参照.

表CBJR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$\|\mathrm{Re}(z)\|>\log(fl_{max})$ 又は，$\|\mathrm{Im}(z)\|>\log(fl_{max})$ であった.	ZBJ=(0.0,0.0)とする.
30000	V<0であった.	ZBJ=(0.0,0.0)とする.

(3)　使用上の注意

a.　使用する副プログラム
① SSL II ... MGSSL, AMACH, ULMAX
② FORTRAN 基本関数 ... REAL, AIMAG, ABS, FLOAT, CEXP, CLOG, GAMMA, CONJG, AMAX1, CMPLX

b.　注意
① $|\mathrm{Re}(z)|\le\log(fl_{max})$, $|\mathrm{Im}(z)|\le\log(fl_{max})$ および V≥0 であること.
② $J_\nu(z)$, $J_{\nu+1}(x)$, $J_{\nu+2}(z)$, ..., $J_{\nu+\mathrm{M}}(z)$ の値をいっせいに必要とするときには，本サブルーチンにより $J_{\nu+\mathrm{M}}(z)$ と $J_{n+\mathrm{M}-1}(z)$を求め，漸化式を反復適用して $J_{\nu+M-2}(z)$, $J_{\nu+M-3}(z)$, ..., $J_\nu(z)$ と次数の低いものを求めていくとよい．逆に，本サブルーチンにより $J_\nu(z)$ と $J_{\nu+1}(z)$を求め，漸化式により $J_{\nu+2}(z)$, $J_{\nu+3}(z)$, ..., $J_{\nu+M}(z)$と次数の高いものを求めていくことは不安定が生ずるので避けなければならない.

c.　使用例
$J_\nu(z)$ の値を，z=10+5iについて，ν を 0.1 から 10 まで 0.1 刻みで計算する.

```
C     **EXAMPLE**
      COMPLEX Z,ZBJ
      Z=(10.0,5.0)
      DO 10 NV=1,100
      V=FLOAT(NV)/10.0
      CALL CBJR(Z,V,ZBJ,ICON)
      IF(ICON.EQ.0) WRITE(6,600) Z,V,ZBJ
   10 CONTINUE
      STOP
  600 FORMAT(' ',2F8.2,F10.3,5X,2E17.7)
      END
```

(4)　手法概要

求められるべき $J_\upsilon(z)$について，その絶対値がアンダフローする値になることがあらかじめ分かった場合には，以下の計算は行わず，結果を（0.0, 0.0）とする.

$J_\nu(z)$ の計算は，zにより方法が異なる.

① $|\mathrm{Re}(z)|+|\mathrm{Im}(z)|\le1$の場合
べき級数展開

$$J_\nu(z)=\left(\frac{1}{2}z\right)^\nu\sum_{k=0}^{\infty}\frac{\left(-\frac{1}{4}z^2\right)^k}{k!\,\Gamma(\nu+k+1)}\qquad(4.1)$$

により，第 k 項目が初項に比べ丸め誤差の単位以下になるまで計算する.

② $|\mathrm{Re}(z)|+|\mathrm{Im}(z)|>1$ かつ $|\mathrm{Re}(z)|\le\log(fl_{\max})$ かつ $|\mathrm{Im}(z)|\le\log(fl_{\max})$ の場合
漸化式による方法を用いて計算する.
m を適当に大きな整数（z, ν 及び要求精度によって決まる），δ を適当に小さな定数(10^{-38})とする.
n 及びα を次式のように決める.

$v = n + \alpha (n: 整数, 0 \le \alpha < 1)$

初期値を

$F_{\alpha+m+1}(z) = 0,\ F_{\alpha+m}(z) = \delta$

として，$k=m,m\text{-}1,\ldots,1$ に対して漸化式

$$F_{\alpha+k-1}(z) = \frac{2(\alpha+k)}{z} F_{\alpha+k}(z) - F_{\alpha+k+1}(z) \qquad (4.2)$$

を反復適用する．このとき，$J_\nu(z)$ は

$$J_\nu(z) \approx \frac{1}{2}\left(\frac{z}{2}\right)^\alpha \frac{\Gamma(2\alpha+1)}{\Gamma(\alpha+1)} e^{-iz} F_{\alpha+n}(z) \Bigg/ \left(\sum_{k=0}^{m} \frac{(\alpha+k)\Gamma(2\alpha+k) i^k}{k!} F_{\alpha+k}(z) \right) \qquad (4.3)$$

として求められる．ただし，上式は $0 \le arg\ z \le \pi$ で用いることができる．$-\pi \le arg\ z < 0$ では，上式は桁落ちを生ずるので，関係式 $J_\nu(\bar{z}) = \overline{J_\nu(z)}$ を用いて $0 \le arg\ z \le \pi$ の場合に帰着させる．

なお，mの決め方及びその他詳細については，参考文献 [81] , [83] を参照のこと．

I11-82-1201　　CBKN, DCBKN

複素変数第2種整数次変形ベッセル関数$K_n(z)$
CALL CBKN(Z,N,ZBK,ICON)

(1) 機能

複素変数zの第2種整数次変形ベッセル関数$K_n(z)$

$$K_n(z) = K_{-n}(z)$$
$$= (-1)^{n+1}\left\{\gamma + \log\left(\frac{z}{2}\right)\right\} I_n(z)$$
$$+ \frac{(-1)^n}{2}\left(\frac{z}{2}\right)^n \sum_{k=0}^{\infty} \frac{\left(\frac{z^2}{4}\right)^k}{k!(n+k)!}\left(\Phi_k + \Phi_{k+n}\right)$$
$$+ \frac{1}{2}\left(\frac{z}{2}\right)^{-n} \sum_{k=0}^{n-1} \frac{(n-k-1)!}{k!}\left(-\frac{z^2}{4}\right)^k$$

の値を漸化式による方法及びτ-methodを用いて計算する．ただし,上式において，n=0のとき，最後の項は0であり，γはオイラーの定数，$I_n(z)$ は第1種変形ベッセル関数，また,

$$\Phi_0 = 0$$
$$\Phi_k = \sum_{m=1}^{k} \frac{1}{m} \quad (k \geq 1)$$

である.

(2) パラメタ

Z …… **入力**．独立変数z．複素数型変数.
N …… **入力**．$K_n(z)$の次数n .
ZBK …… **出力**．関数値$K_n(z)$．複素数型変数.
ICON …… **出力**．コンディションコード.
表CBKN-1参照.

表CBKN-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	次のいずれかであった. ・ $\|\mathrm{Re}(z)\| > \log(fl_{max})$ ・ $\mathrm{Re}(z) < 0$ かつ $\|\mathrm{Im}(z)\| \geq \log(fl_{max})$ ・ $\mathrm{Re}(z) \geq 0$ かつ $\|\mathrm{Im}(z)\| \geq t_{max}$	ZBK=(0.0,0.0)とする.
30000	$\|z\| = 0.0$ であった.	ZBK=(0.0,0.0)とする.

(3) 使用上の注意

a.　使用する副プログラム

① SSL II ....AMACH, CBIN, MGSSL, ULMAX, UTLIM

② FORTRAN 基本関数 ... REAL, AIMAG, ABS, IABS, MOD, CSQRT, CEXP, CLOG, FLOAT, CMPLX

b.　注意

① $|z| \neq 0$ であること.

② $|\mathrm{Re}(z)| \leq \log(fl_{\max})$ であること.

③ $\mathrm{Re}(z) \geq 0$ の場合には，$|\mathrm{Im}(z)| < t_{\max}$ であることが必要であるが，これは，計算式中の exp(-z) の値が正確に計算できなくなるためである.
$\mathrm{Re}(z)<0$ の場合には、$|Im(z)| \leq \log(fl_{max})$ であることが必要である.

④ $\mathrm{Re}(z)\geq 0$ の場合において，$K_n(z)$, $K_{n+1}(z)$, $K_{n+2}(z)$, ..., $K_{n+M}(z)$ の値をいっせいに必要とするときには，本サブルーチンにより，$K_n(z)$ と $K_{n+1}(z)$ を求め，漸化式（手法概要の (4.1)）を反復適用して，$K_{n+2}(z)$, $K_{n+3}(z)$, ..., $K_{n+M}(z)$ と次数の高いものへと順次求めていくとよい．$\mathrm{Re}(z)<0$ の場合には，上記のことは安定ではないので，必要とする次数について本サブルーチンを呼ばなければならない.

c.　使用例

$K_1(1+2i)$ の値を計算する.

```
C     **EXAMPLE**
      COMPLEX Z,ZBK
      Z=(1.0,2.0)
      N=1
      CALL CBKN(Z,N,ZBK,ICON)
      WRITE(6,600) Z,N,ZBK,ICON
      STOP
600   FORMAT(' ',2F8.2,I6,5X,2E17.7,I7)
      END
```

(4) 手法概要

関係式

$$K_{-n}(z) = K_n(z)$$

が成り立つので，n<0の場合は，n≥0の場合に帰着させる.

$K_n(z)$ の計算は，$\mathrm{Re}(z)\geq 0$あるいは$\mathrm{Re}(z)<0$により方法が異なる.

a. $|\mathrm{Re}(z)| \geq 0$ の場合

この場合には，$K_n(z)$ は，$K_0(z)$ 及び $K_1(z)$ から，漸化式

$$K_{n+1}(z) = \frac{2k}{z} K_k(z) + K_{k-1}(z) \qquad (4.1)$$
$$k = 1,2,...,n-1$$

により計算できる．$K_0(z)$ 及び $K_1(z)$ の計算は，領域によって計算法が異なる．

① 単精度：$|\mathrm{Im}(z)| < -2.25\ \mathrm{Re}(z) + 4.5$
倍精度：$|\mathrm{Im}(z)| < -4\ \mathrm{Re}(z) + 8$
のときには，次式により，$K_0(z)$ 及び $K_1(z)$を計算する．

$$K_0(z) = -\left\{\gamma + \log\left(\frac{z}{2}\right)\right\} I_0(z) + 2\sum_{k=0}^{\infty} \frac{I_{2k}(z)}{k} \qquad (4.2)$$

$$K_1(z) = \left(\frac{1}{z} - I_1(z)K_0(z)\right) \Big/ I_0(z) \qquad (4.3)$$

ただし，$I_k(z)$ は漸化式による方法で求める．

② 単精度：$|\mathrm{Im}(z)| \geq -2.25\mathrm{Re}(z) + 4.5$
倍精度：$|\mathrm{Im}(z)| \geq -4\mathrm{Re}(z) + 8$
のときには，τ-method を用いて計算する(参考文献 [84] 参照)．この方法では，$K_n(z)$ を

$$K_n(z) = \sqrt{\frac{\pi}{2z}} e^{-z} f_n\left(\frac{1}{z}\right) \qquad (4.4)$$

の形として，$f_n(1/z)$ についての近似式を求めている．以下，$t=1/z$ とする．$f_n(t)$ は，

$$t^2 f_n''(t) + 2(t+1) f_n'(t) - \left(n^2 - \frac{1}{4}\right) f_n(t) = 0 \qquad (4.5)$$

を満足する．上式の右辺に，直行区間 $[0,\eta]$ のずらし超球（shifted Ultraspherical）多項式を τ 倍したものを付加すると，

$$t^2 f_n''(t) + 2(t+1) f_n'(t) - \left(n^2 - \frac{1}{4}\right) f_n(t) = \tau C_m^{*(\alpha)}\left(\frac{t}{\eta}\right) \qquad (4.6)$$

となる．ここで，

$$C_m^{*(\alpha)}(t) = \sum_{i=0}^{m} C_{mi}^{*(\alpha)} t^i \qquad (4.7)$$

は，ずらし超球多項式である（α=0 のとき，ずらしチェビシェフ多項式，α=0.5 のとき，ずらしルジャンドル多項式に相当する）．（4.6）は次の形の m 次の多項式の解をもつ．

$$f_{nm}(t) = \tau \sum_{k=0}^{m} \frac{C_{mk}^{*(\alpha)} \sum_{i=0}^{k} a_i t^i}{(k+1) a_{k+1} \eta^k} \qquad (4.8)$$

ただし，

$$\left.\begin{aligned} a_0 &= 1 \\ a_k &= \frac{(4n^2 - 1^2)(4n^2 - 3^2)..(4n^2 - (2k-1)^2)}{k! 8^k} \end{aligned}\right\} \quad (k \geq 1) \qquad (4.9)$$

である．初期条件 $f_{nm}(0)=1$ ($t \to 0$ のとき，$f_n(t) \to 1$) から τ を決めると，

$$f_{nm}(t) = \frac{\sum_{k=0}^{m} \frac{C_{mk}^{*(\alpha)} \sum_{i=0}^{k} a_i t^i}{(k+1) a_{k+1} \eta^k}}{\sum_{k=0}^{m} \frac{C_{mk}^{*(\alpha)}}{(k+1) a_{k+1} \eta^k}} \qquad (4.10)$$

が得られる．上式は，未知数として，α 及び η を含んでいるが，α=0.5 (ずらしルジャンドル多項式の場合)，$\eta = t$ と選んだときに最も精度が高いことが既に分っている (参考文献 [84] 参照)．そのように選べば，

$$f_{nm}(t) = \frac{\sum_{k=0}^{m} \frac{P_{mk}^{*} \sum_{i=0}^{k} a_i t^i}{(k+1) a_{k+1} t^k}}{\sum_{k=0}^{m} \frac{P_{mk}^{*}}{(k+1) a_{k+1} t^k}} \qquad (4.11)$$

となる．ただし，P^*_{mk} は，ずらしルジャンドル多項式，

$$P_m(t) = \sum_{i=0}^{m} P_{mi}^{*} t^i \qquad (4.12)$$

の係数である．（4.11）の分母，分子に t^m を乗じ，t のべきでまとめれば，

$$f_{nm}(t) = \frac{\sum_{i=0}^{m} G_i(m,n)\, t^i}{\sum_{i=0}^{m} H_i(m,n)\, t^i} \qquad (4.13)$$

となる．ただし，

$$G_i(m,n) = \sum_{k=0}^{i} \frac{P_{m,m-i+k}^{*} a_k}{(m+1-i+k) a_{m+1-i+k}} \qquad (4.14)$$

$$H_i(m,n) = \frac{P_{m,m-i}}{(m-i+1) a_{m-i+1}} \qquad (4.15)$$

である．（4.4）及び（4.13）より，$K_n(z)$の計算式として，

$$K_n(z)=\sqrt{\frac{1}{z}}e^{-z}\frac{\sum_{i=0}^{m}\tilde{G}_i(m,n)\left(\frac{1}{z}\right)^i}{\sum_{i=0}^{m}\tilde{H}_i(m,n)\left(\frac{1}{z}\right)^i} \qquad (4.16)$$

が得られる．ただし，

$$\tilde{G}_i(m,n)=G_i(m,n)/G_m(m,n) \qquad (4.17)$$

$$\tilde{H}_i(m,n)=\sqrt{\frac{2}{\pi}}H_i(m,n)/G_m(m,n) \qquad (4.18)$$

である．m，n を決めれば，$\tilde{G}_i(m,n)$ 及び $\tilde{H}_i(m,n)$ は定数となる．本サブルーチンでは，n=0 及び 1 について，単精度の場合には $|z|\geq 17$ のとき，m=3,それ以外のとき，m=7，倍精度の場合には，$(\mathrm{Re}(z))^2+0.425(\mathrm{Im}(z))^2\geq 15^2$ のとき，m=10，それ以外のとき，m=19 として，あらかじめ $\tilde{G}_i(m,n)$ 及び $\tilde{H}_i(m,n)$ の値をデータ文により表としてもっている．

b. $\mathrm{Re}(z)<0$ の場合

$K_n(z)$ の切断線（cut）を負の実軸に選ぶことにする．

そのために，$\mathrm{Im}(z)\geq 0$ の場合には，関係式

$$K_n(z)=(-1)^nK_n(-z)-\pi i I_n(-z) \qquad (4.19)$$

を $\mathrm{Im}(z)<0$ の場合には，関係式

$$K_n(z)=(-1)^nK_n(-z)+\pi i I_n(-z) \qquad (4.20)$$

を利用する．ここで，$K_n(-z)$ の値は，①で述べた $\mathrm{Re}(z)\geq 0$ での計算法により求め，$I_n(-z)$ の値は，サブルーチン CBIN により求める．このように，$\mathrm{Re}(z)<0$ の場合には，$\mathrm{Re}(z)\geq 0$ の場合と異なり，$I_n(-z)$ の計算が余分に必要である．

B21-15-0202 CBLNC, DCBLNC

複素行列の平衡化
CALL CBLNC (ZA, K, N, DV, ICON)

(1) 機能

n次の複素行列$\boldsymbol{A}$を（1.1）の対角相似変換により，平衡化する．

$$\tilde{\boldsymbol{A}} = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D} \qquad (1.1)$$

ここで，平衡化するとは，変換後の複素行列$\tilde{\boldsymbol{A}}$について第i行 (i = 1, ..., n) 要素のノルムの和と，第i列要素のノルムの和をほぼ等しくすることである．また，要素のノルムとは，複素数$z = x + i\cdot y$に対して，$\|z\|_1=|x|+|y|$ を考える．$\boldsymbol{D}$は実対角行列である．$n \geq 1$なること．

(2) パラメタ

ZA……**入力**．複素行列$\boldsymbol{A}$.
出力．平衡化後の複素行列$\tilde{\boldsymbol{A}}$.
ZA (K,N) なる2次元配列.

K……**入力**．配列ZAの整合寸法.

N……**入力**．複素行列$\boldsymbol{A}$及び$\tilde{\boldsymbol{A}}$の次数n.

DV……**出力**．スケーリングファクタ（$\boldsymbol{D}$の対角要素）.
大きさnの1次元配列.

ICON……**出力**．コンディションコード.
表CBLNC- 1参照.

表CBLNC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1であった.	平衡化は行わない.
30000	N < 1又はK < Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... IRADIX, MGSSL

② FORTRAN 基本関数 ... REAL, AIMAG, ABS

b. 注意

① 行列の各要素のノルムが不揃いな場合，この行列の固有値及び固有ベクトルの計算値の精度が悪くなりやすい．本サブルーチンは，これを防ぐためのものである.

② 行列の各要素の大きさがほぼ揃っているときには，本サブルーチンにより何等の変換も行われないので，本サブルーチンの実行を省略した方がよい.

③ 本サブルーチンは行列の平衡化にあたり，行又は列の要素が，対角要素を除いてすべて零となるような行（又は列）がある場合，その行（又は列）と対応する列（又は行）については平衡化を省略する.

④ 行列 $\boldsymbol{A}$ の固有ベクトル x を求めるときは，本サブルーチンにより平衡化した行列 $\tilde{\boldsymbol{A}}$ の固有ベクトル$\tilde{x}$ に対して，(3.1) の逆変換を行う必要がある.

$$\boldsymbol{x} = \boldsymbol{D}\tilde{\boldsymbol{x}} \qquad (3.1)$$

(3.1) の逆変換はサブルーチン CHBK2 により実行できる．(CNBK2 参照のこと).

c. 使用例

n 次の複素行列 $\boldsymbol{A}$ を平衡化した後，サブルーチン CHES2 により複素ヘッセンベルグ行列に変換し，固有値をサブルーチン CHSQR で求める.

$n \leq 100$ の場合.

```
C     **EXAMPLE**
      COMPLEX ZA(100,100),ZE(100)
      DIMENSION DV(100),IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),
     *            J=1,N)
      WRITE (6,600) N
      DO 20 I=1,N
   20 WRITE(6,610)(I,J,ZA(I,J),J=1,N)
      CALL CBLNC(ZA,100,N,DV,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
      CALL CHES2(ZA,100,N,IP,ICON)
      CALL CHSQR(ZA,100,N,ZE,M,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,630) (ZE(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',
     *       5X,'N=',I3/)
  610 FORMAT(/2(5X,'A(',I3,',',I3,')=',
     *       2E15.7))
  620 FORMAT('0',20X,'ICON=',I5)
  630 FORMAT('0',5X,'EIGENVALUE'/'0',
     *       20X,'REAL',16X,'IMAG'/
     *       ('0',15X,2(E15.7, 5X)))
      END
```

(4) 手法概要

n次の複素行列$\boldsymbol{A}$を，(4.1) の対角相似変換を反復的に行うことにより平衡化する.

$$\boldsymbol{A}_s = \boldsymbol{D}_s^{-1}\boldsymbol{A}_{s-1}\boldsymbol{D}_s \qquad s = 1, 2, \cdots \qquad (4.1)$$

ここに，$\boldsymbol{A}_0 = \boldsymbol{A}$であり，$\boldsymbol{D}_s$は (4.2) で表される実対角行列，$s$は反復回数である.

$$\boldsymbol{D}_s = \begin{bmatrix} d_1^{(s)} & & & 0 \\ & d_2^{(s)} & & \\ & & \cdots & \\ 0 & & & d_n^{(s)} \end{bmatrix} \tag{4.2}$$

(4.1) の平衡化は，$\boldsymbol{A}_s$の第i行（ i =1,2,⋯,n）要素から対角要素を除いたもののノルムの和と，第i列要素から対角要素を除いたもののノルムの和がほぼ等しくなるように行う．

すなわち， $\boldsymbol{A}_s = \left(a_{ij}^{(s)}\right)$として

$$\sum_{\substack{k=1 \\ k\neq i}}^{n} \left\| a_{ik}^{(s)} \right\|_1 \approx \sum_{\substack{k=1 \\ k\neq i}}^{n} \left\| a_{ki}^{(s)} \right\|_1 \tag{4.3}$$

を満足するようにする．

いま$\boldsymbol{D}_{si}$を

$$\boldsymbol{D}_{si} = \begin{bmatrix} 1 & & & & & & \\ & \cdots & & & & 0 & \\ & & 1 & & & & \\ & & & d_i^{(s)} & & & \\ & & & & 1 & & \\ & 0 & & & & \cdots & \\ & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ i \\ \\ \\ \\ \end{matrix} \qquad \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \end{matrix} \tag{4.4}$$

$$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad i$$

と定義すれば，(4.2) の$\boldsymbol{D}_s$は，

$$\boldsymbol{D}_s = \boldsymbol{D}_{s1}\boldsymbol{D}_{s2}\cdots\cdots\cdots\boldsymbol{D}_{sn} \tag{4.5}$$

と表される．

(4.5) より，(4.1) の変換は，(4.6) で示す変換を，i = l, 2, ..., n と次々に行うことにより実行できることが分かる．

$$\boldsymbol{A}_{si} = \boldsymbol{D}_{si}^{-1}\boldsymbol{A}_{si-1}\boldsymbol{D}_{si} \tag{4.6}$$

ただし， $\boldsymbol{A}_{s0} = \boldsymbol{A}_{s-1}, \boldsymbol{A}_{sn} = \boldsymbol{A}_s$ である．

ここで$\boldsymbol{D}_{si}$の対角要素 $d_i^{(s)}$は，変換後のi行とi列が，(4.3)を満足するように定め，変換前に，既にi行とi列が(4.3)を満足していれば， $d_i^{(s)} = 1$すなわち$\boldsymbol{D}_{si} = \boldsymbol{I}$とする．

(4.1) の反復は，すべての行と列に対して，(4.3) が満足されたとき終了する．

本サブルーチンでは，この操作を以下に示す手順で実行する．

① 第 i 行と第 i 列の各要素のノルムの和を，対角要素を除いて計算する．

$$C = \sum_{\substack{k=1 \\ k\neq i}}^{n} \left\| a_{ki} \right\|_1 \tag{4.7}$$

$$R = \sum_{\substack{k=1 \\ k\neq i}}^{n} \left\| a_{ik} \right\|_1 \tag{4.8}$$

② $d_i^{(s)}$ を次のように定める．

$$d_i^{(s)} = \rho^k \tag{4.9}$$

ここに， $\rho = \begin{cases} 2, & \text{2進法の計算機のとき} \\ 16, & \text{16進法の計算機のとき} \end{cases}$

なる定数である．

k は，次の条件を満足するように定める．

$$R \cdot \rho > C \cdot \rho^{2k} \geq R/\rho \tag{4.10}$$

(4.10)より k は，$C < R/\rho$ のとき，$k > 0$ であり，$C \geq R/\rho$ のとき，$k \leq 0$ である．

③ (4.11) の条件により変換の要不要を判定する．

$$\left(C \cdot \rho^{2k} + R\right) \Big/ \rho^k < 0.95\left(C + R\right) \tag{4.11}$$

(4.11) が満足されれば， $d_i^{(s)} = \rho^k$ として変換を行い，満足されなければ変換を省略する．

④ すべての行と列について，全く変換が行われなくなったとき，平衡化の処理を終了する．

このとき，配列 DV には，

$$\boldsymbol{D} = \boldsymbol{D}_1\boldsymbol{D}_2\ldots\boldsymbol{D}_s \tag{4.12}$$

で与えられる $\boldsymbol{D}$ の対角要素がスケーリングファクタとして格納されている．

なお，詳細については，参考文献 [13] pp.315 – 326を参照すること．

I11-82-1401 CBYN, DCBYN

複素変数第2種整数次ベッセル関数$Y_n(z)$
CALL CBYN (Z, N, ZBY, ICON)

(1) 機能

複素変数zの第2種整数次ベッセル関数$Y_n(z)$

$$
\begin{aligned}
Y_n(z) &= (-1)^n Y_{-n}(z) \\
&= \frac{2}{\pi}\left\{\gamma + \log\left(\frac{z}{2}\right)\right\} J_n(z) \\
&\quad - \frac{1}{\pi}\left(\frac{z}{2}\right)^n \sum_{k=0}^{\infty} \frac{\left(-\frac{z^2}{4}\right)^k}{k!(n+k)!}\left(\Phi_k + \Phi_{k+n}\right) \\
&\quad - \frac{1}{\pi}\left(\frac{z}{2}\right)^{-n} \sum_{k=0}^{n-1} \frac{(n-k-1)!}{k!}\left(\frac{z^2}{4}\right)^k
\end{aligned}
$$

の値を漸化式による方法及び τ-methodを用いて計算する．ただし，上式において，$n = 0$のとき，最後の項は0であり，γ はオイラーの定数，$J_n(z)$ は第1種ベッセル関数，また，

$$
\begin{aligned}
\Phi_0 &= 0 \\
\Phi_k &= \sum_{m=1}^{k} \frac{1}{m} \quad (k \geq 1)
\end{aligned}
$$

である．

(2) パラメタ

Z …… **入力**．独立変数z．複素数型変数．
N …… **入力**．$Y_n(z)$ の次数n．
ZBY …… **出力**．関数値$Y_n(z)$．複素数型変数．
ICON …… **出力**．コンディションコード．
表CBYN-1参照．

表CBYN-1コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	$\lvert\mathrm{Re}(z)\rvert > \log(fl_{max})$ 又は， $\lvert\mathrm{Im}(z)\rvert > \log(fl_{max})$ であった．	ZBY = (0.0, 0.0)とする．
30000	$\lvert z\rvert = 0.0$であった．	ZBY = (0.0, 0.0)とする．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, CBIN, CBKN, MGSSL, ULMAX, UTLIM

② FORTRAN 基本関数 ... REAL, AIMAG, IABS, CONJG, CMPLX, MOD

b. 注意

① $|z| \neq 0$ であること．

② $\lvert\mathrm{Re}(z)\rvert \leq \log(fl_{max})$ 及び $\lvert\mathrm{Im}(z)\rvert \leq \log(fl_{max})$ であること．

③ $Y_n(z)$, $Y_{n+1}(z)$, $Y_{n+2}(z)$, …, $Y_{n+M}(z)$ の値をいっせいに必要とするときには，手法概要で述べる方法を用いるとよい．本サブルーチンにより，$Y_n(z)$ 及び $Y_{n+1}(z)$ を求め，漸化式を反復適用して次数の高いものを求めていくことは，zが正の実数の場合を除いて不安定を生ずるので避けなければならない．

c. 使用例

$Y_1(1+2i)$ の値を計算する．

```
C     **EXAMPLE**
      COMPLEX Z,ZBY
      Z=(1.0,2.0)
      N=1
      CALL CBYN(Z,N,ZBY,ICON)
      WRITE(6,600)Z,N,ZBY,ICON
      STOP
600   FORMAT(' ',2F8.2,I6,5X,2E17.7,I7)
      END
```

(4) 手法概要

$$
Y_{-n}(z) = (-1)^n Y_n(z) \tag{4.1}
$$

であるので，$n < 0$の場合は，$n \geq 0$の場合に帰着させる．また，

$$
Y_n(\bar{z}) = \overline{Y_n(z)} \tag{4.2}
$$

であるので，Im (z) < 0の場合は，Im (z) ≥ 0の場合に帰着させる．

関係式

$$
Y_n(z) = i^{n+1} I_n(-iz) - \frac{2}{\pi} i^n (-1)^n K_n(-iz) \tag{4.3}
$$

$$
, \; i = \sqrt{-1}
$$

を利用して$Y_n(z)$ の値を計算する．ここで，$I_n(-iz)$ の値は，漸化式による方法で計算を行っているサブルーチンCBINを用いて計算し，$K_n(-iz)$ の値は，漸化式による方法及び τ-methodで計算を行っているサブルーチンCBINを用いて計算する．

なお，$Y_n(z)$, $Y_{n+1}(z)$, $Y_{n+2}(z)$, …, $Y_{n+M}(z)$ の値をいっせいに必要とするとき，これらの値を能率的に求めるには以下のようにすればよい．CBINにより，$I_{n+M}(-iz)$ 及び$I_{n+M-1}(-iz)$ の値を求め，漸化式を反復適用して次数の低いものへと，$I_n(-iz)$ まで順次求める．また，CBINにより，$K_n(-iz)$ 及び $K_{n+1}(-iz)$ の値を求め，漸化式を反復適用して次数の高いものへと，$K_{n+M}(z)$ まで順次求めて，(4.3) により，必要とするものを計算する．

B21-15-0101 CEIG2, DCEIG2

複素行列の固有値固有ベクトル (QR 法)
CALL CEIG2 (ZA, K, N, MODE, ZE, ZEV, VW, IVW, ICON)

(1) 機能

*n*次の複素行列***A***の固有値及び対応する固有ベクトルを求める．固有ベクトルは $\|x\|_2=1$となるように正規化される．$n \geq 1$であること．

(2) パラメタ

ZA …… **入力**．複素行列***A***.
ZA (K, N) なる複素数型2次元配列.
演算後，内容は保存されない.

K …… **入力**．配列ZA及びZEVの整合寸法 (≥ *n*).

N …… **入力**．複素行列***A***の次数*n*.

MODE … **入力**．平衡化の省略の指定.
MODE = 1のときは，平衡化を省略する.
MODE ≠ 1のときは，平衡化を行う.

ZE …… **出力**．固有値.
大きさ*n*の複素数型1次元配列.

ZEV …… **出力**．固有ベクトル.
固有ベクトルは，固有値と対応する列に格納される.
ZEV (K, N) なる複素数型2次元配列.

VW …… **作業領域**．大きさ*n*の1次元配列.

IVW …… **作業領域**．大きさ*n*の1次元配列.

ICON …… **出力**．コンディションコード.
表CEIG2-1参照.

表CEIG2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1であった.	ZE (1) = ZA (1,1) ZEV (1) = (1.0, 0.0) とする.
20000	三角行列に変換できず，固有値・固有ベクトルは求められなかった.	処理を打ち切る.
30000	N < 1又はK < Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, CBLNC, CHES2, CNRML, IRADIX, MGSSL

② FORTRAN 基本関数 ... ABS, REAL, AIMAG, AMAX1, CSQRT, SIGN, SQRT, CONJG

b. 注意

① 平衡化について
複素行列の各要素の大きさが，大きく異なっているような場合には，一般に行列の平衡化によって結果の精度を上げることができる．このため，本サブルーチンでは，サブルーチン CBLNC により平衡化を行うようにしている．他方，もし行列の各要素の大きさがほぼそろっているような場合には，行列の平衡化による効果を期待できない．このような場合には，MODE=1 とすることにより，平衡化を省略できる.

② 本サブルーチンは，複素行列の固有値及び固有ベクトルを全体として求めるためのものである.
固有値だけが必要な場合には，サブルーチン CBLNC, CHES2, CHSQR により求めること.
また，固有ベクトルの一部だけが必要な場合には，サブルーチンCBLNC, CHES2, CHSQR, CHVEC, CHBK2, CNRMLにより求めること.

c. 使用例
n 次の複素行列 ***A*** の固有値及び固有ベクトルを求める．$n \leq 100$ の場合

```
C     **EXAMPLE**
      COMPLEX ZA(100,100),ZE(100),
     *        ZEV(100,100)
      DIMENSION IND(100),VW(100),
     *          IVW(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),
     *             J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,ZA(I,J),J=1,N)
      CALL CEIG2(ZA,100,N,0,ZE,ZEV,VW,
     *           IVW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      DO 30 I=1,N
   30 IND(I)=1
      CALL CEPRT(ZE,ZEV,100,N,IND,N)
      GO TO 10
  500 FORMAT(I3)
  510 FORMAT(4E15.7)
  600 FORMAT('0',5X,'ORIGINAL MATRIX',
     *5X,'N=',I3)
  610 FORMAT(/2(5X,'A(',I3,',',I3,')=',
     *2E15.7))
  620 FORMAT('0',5X,'ICON=',I5)
      END
```

本使用例中のサブルーチン CEPRT は，複素行列の固有値及び固有ベクトルを出力するためのサブルーチンである．以下にその内容を示す.

```
      SUBROUTINE CEPRT(ZE,ZEV,K,N,IND,M)
      COMPLEX ZE(M),ZEV(K,M)
      DIMENSION IND(M)
      WRITE(6,600)
      MM=0
      DO 20 J=1,M
      IF(IND(J).EQ.0) GO TO 20
      MM=MM+1
      ZE(MM)=ZE(J)
      DO 10 I=1,N
      ZEV(I,MM)=ZEV(I,J)
   10 CONTINUE
   20 CONTINUE
      IF(MM.EQ.0) GO TO 50
      DO 40 INT=1,MM,3
      LST=MIN0(INT+2,MM)
      WRITE(6,610) (J,J=INT,LST)
      WRITE(6,620) (ZE(J),J=INT,LST)
      DO 30 I=1,N
      WRITE(6,630) (ZEV(I,J),J=INT,LST)
   30 CONTINUE
   40 CONTINUE
   50 RETURN
  600 FORMAT('1',30X,'**EIGENVECTORS**')
  610 FORMAT('0',26X,I3,29X,I3,29X,I3)
  620 FORMAT('0',' EIGENVALUES',
     *3(2X,2E15.7))
  630 FORMAT(12X,3(2X,2E15.7))
      END
```

(4) 手法概要

*n*次の複素行列$\boldsymbol{A}$の固有値と対応する固有ベクトルを求める.

*n*次の複素行列の固有値は，次の3段階の変換を行うことにより，上三角行列$\boldsymbol{R}$の対角要素として求められる.

① 対角相似変換による複素行列 $\boldsymbol{A}$ の平衡化

$$\tilde{\boldsymbol{A}} = \boldsymbol{B}^{-1}\boldsymbol{A}\boldsymbol{B} \tag{4.1}$$

ここに，$\boldsymbol{B}$ はスケーリングファクターを要素とする対角行列である．詳細については，サブルーチン CBLNC を参照のこと.

② 安定化基本相似変換による複素行列 $\tilde{\boldsymbol{A}}$ の複素ヘッセンベルグ行列 $\boldsymbol{H}$ への変換,

$$\boldsymbol{H} = \boldsymbol{S}^{-1}\tilde{\boldsymbol{A}}\boldsymbol{S} \tag{4.2}$$

ここに $\boldsymbol{S}$ は，変換行列 $\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_{n-2}$ の積

$$\boldsymbol{S} = \boldsymbol{S}_1\boldsymbol{S}_2\ldots\boldsymbol{S}_{n-2} \tag{4.3}$$

で与えられ，各々の $\boldsymbol{S}_i$ は置換行列 $\boldsymbol{P}_i$ と消去行列 $\boldsymbol{N}_i$ とによって,

$$\boldsymbol{S}_i = \boldsymbol{P}_i\boldsymbol{N}_i^{-1}, i = 1,2,\ldots,n-2 \tag{4.4}$$

で与えられる．詳細については，サブルーチン CHES2 を参照のこと.

③ 複素 QR 法による複素ヘッセンベルグ行列 $\boldsymbol{H}$ から複素上三角行列 $\boldsymbol{R}$ への変換,

$$\boldsymbol{R} = \boldsymbol{Q}_R^*\boldsymbol{H}\boldsymbol{Q}_R \tag{4.5}$$

ここに $\boldsymbol{Q}_R$ は，複素 QR 法で用いる変換行列 $\boldsymbol{Q}_1, \boldsymbol{Q}_2, \ldots, \boldsymbol{Q}_L$ の積

$$\boldsymbol{Q}_R = \boldsymbol{Q}_1\boldsymbol{Q}_2\ldots\boldsymbol{Q}_L \tag{4.6}$$

で与えられるユニタリ行列である.
詳細は，サブルーチン CHSQR を参照すること.

一方，固有ベクトルは，(4.7)の相似変換により上三角行列$\boldsymbol{R}$を対角行列$\boldsymbol{D}$に変換する行列$\boldsymbol{F}$が存在すれば，(4.8)によって得られる行列$\boldsymbol{X}$の列ベクトルとして求めることができる.

$$\boldsymbol{D} = \boldsymbol{F}^{-1}\boldsymbol{R}\boldsymbol{F} \tag{4.7}$$
$$\boldsymbol{X} = \boldsymbol{B}\boldsymbol{S}\boldsymbol{Q}_R\boldsymbol{F} \tag{4.8}$$

(4.8) で与えられる行列$\boldsymbol{X}$の列ベクトルが，行列$\boldsymbol{A}$の固有ベクトルとなることは，(4.7) に (4.1)，(4.2) 及び (4.5) を代入すれば(4.9)が得られることから分かる.

$$\boldsymbol{D} = \boldsymbol{F}^{-1}\boldsymbol{Q}_R^*\boldsymbol{S}^{-1}\boldsymbol{B}^{-1}\boldsymbol{A}\boldsymbol{B}\boldsymbol{S}\boldsymbol{Q}_R\boldsymbol{F} = \boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} \tag{4.9}$$

ここで，$\boldsymbol{B}\boldsymbol{S}\boldsymbol{Q}_R$を$\boldsymbol{Q}$で表せば，(4.3) 及び (4.6) より

$$\boldsymbol{Q} = \boldsymbol{B}\boldsymbol{S}_1\boldsymbol{S}_2\ldots\boldsymbol{S}_{n-2}\boldsymbol{Q}_1\boldsymbol{Q}_2\ldots\boldsymbol{Q}_L \tag{4.10}$$

となる.

(4.10)より，$\boldsymbol{Q}$の計算は，変換の進行につれて逐次変換行列の積を作ることにより実行できることが分かる.

さて$\boldsymbol{F}$は単位上三角行列として求めることができる．すなわち，(4.7)より,

$$\boldsymbol{F}\boldsymbol{D} = \boldsymbol{R}\boldsymbol{F} \tag{4.11}$$

ここで，$\boldsymbol{D}$, $\boldsymbol{R}$, $\boldsymbol{F}$ の要素をそれぞれ$\boldsymbol{D}$=diag(λ_i), $\boldsymbol{R}$=(γ_{ij}), $\boldsymbol{F}$=(f_{ij}) とすると，f_{ij}は (4.11) より，$j=n, n-1, \ldots, 2$に対して(4.12)で求められる.

$$f_{ij} = \sum_{k=i+1}^{j}\gamma_{ik}f_{kj}/(\lambda_j - \lambda_i) \quad i = j-1, j-2, \ldots, 1 \tag{4.12}$$

ただし,

$$\gamma_{ij} = 0\,(i > j), \quad \gamma_{ii} = \lambda_i$$
$$f_{ij} = 0\,(i > j), \quad f_{ii} = 1$$

もしも，$\lambda_i=\lambda_j$の時は(4.13)で計算するものとする．

$$f_{ij} = \sum_{k=i+1}^{j} \gamma_{ik} f_{kj} \Big/ \left(u \left\| \boldsymbol{A} \right\|_{\infty}\right) \tag{4.13}$$

ここにuは丸めの誤差の単位であり，$\boldsymbol{A}=(a_{ij})$ として $\|\boldsymbol{A}\|_{\infty}$ は，

$$\left\| \boldsymbol{A} \right\|_{\infty} = \max_{j} \sum_{i=1}^{n} \left\| a_{ij} \right\|_1 \tag{4.14}$$

で定義されるノルムである．ただし複素数$z = x + iy$ に対するノルム $\|z\|_1$は，

$$\left\| z \right\|_1 = \left| x \right| + \left| y \right|$$

とする．

なお，詳細については，参考文献 [12] 及び [13] pp.372 – 395を参照すること．

I11-11-0101 CELI1, DCELI1

第1種完全楕円積分　$K(x)$
CALL CELIl (X, CELI, ICON)

(1) 機能

第1種完全楕円積分$K(x)$の値

$$K(x)=\int_0^{\frac{\pi}{2}}\frac{d\theta}{\sqrt{1-x\sin^2\theta}}$$

を近似式を用いて計算する．ただし，$0 \le x < 1$であること．

(2) パラメタ

X……**入力**．独立変数x.
CELI……**出力**．関数値$\boldsymbol{K}(x)$.
ICON……**出力**．コンディションコード.
表CELI1-1参照

表 CELI1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	X < 0又はX ≥ 1であった.	CELI ＝ 0.0とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... DLOG

b. 使用例

$K(x)$の値を x が 0.00 から 0.99 まで 0.01 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,100
      A=K-1
      X=A/100.0
      CALL CELI1(X,CELI,ICON)
      IF(ICON.EQ.0)WRITE(6,610)X,CELI
      IF(ICON.NE.0)WRITE(6,620)X,CELI,
     *             ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF COMPLETE ',
     *'ELLIPTIC INTEGRAL OF THE FIRST ',
     *'KIND'///6X,'X',9X,'K(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'CELI=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4) 手法概要

第1種完全楕円積分$K(x)$の計算は，定義域$0 \le x < 1$で次の近似式を用いて計算する.

単精度：
$$K(x)=\sum_{k=0}^{4}a_k t^k-\log(t)\sum_{k=0}^{4}b_k t^k \qquad (4.1)$$
ただし$t = 1 - x$
理論精度7.87桁

倍精度：
$$K(x)=\sum_{k=0}^{10}a_k t^k-\log(t)\sum_{k=0}^{10}b_k t^k \qquad (4.2)$$
ただし$t = 1 - x$
理論精度17.45桁

なお，詳細については，参考文献 [78] のpp.150 ~ 154を参照のこと.

I11-11-0201 CELI2, DCELI2

第2種完全楕円積分　$E(x)$
CALL CELI2 (X, CELI, ICON)

(1)　機能

第2種完全楕円積分$E(x)$ の値

$$E(x)=\int_0^{\frac{\pi}{2}}\sqrt{1-x\sin^2\theta}\,d\theta$$

を近似値を用いて計算する．ただし，$0 \le x \le 1$であること．

(2)　パラメタ

X…… **入力**．独立変数x.
CELI…… **出力**．関数値$E(x)$.
ICON…… **出力**．コンディションコード.
表CELI2-1参照

表 CELI2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	X < 0又はX > 1であった.	CELI = 0.0とする.

(3)　使用上の注意

a.　使用する副プログラム
①　SSL II ... MGSSL
②　FORTRAN 基本関数 ... DLOG

b.　使用例
$E(x)$ の値を x が 0.00 から 1.00 まで 0.01 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      A=K-1
      X=A/100.0
      CALL CELI2(X,CELI,ICON)
      IF(ICON.EQ.0)WRITE(6,610)X,CELI
      IF(ICON.NE.0)WRITE(6,620)X,CELI,
     *             ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF COMPLETE ',
     *'ELLIPTIC INTEGRAL OF THE ',
     *'SECOND KIND'///6X,'X',9X,'E(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'CELI=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4)　手法概要

第2種完全楕円積分$\boldsymbol{E}(x)$ の計算は，定義域$0 \le x \le 1$で次の近似式を用いて計算する.

単精度：
$$E(x)=\sum_{k=0}^{4}a_k t^k-\log(t)\sum_{k=1}^{4}b_k t^k \qquad (4.1)$$
ただし$t=1-x$
理論精度7.80桁

倍精度：
$$E(x)=\sum_{k=0}^{10}a_k t^k-\log(t)\sum_{k=1}^{10}b_k t^k \qquad (4.2)$$
ただし$t=1-x$
理論精度17.42桁

ただし，x=1 のときは $\boldsymbol{E}(x)$ =1 とする.

なお，詳細については，参考文献 [78] のpp.150-154を参照のこと.

I11-51-0201 CFRI, DCFRI

余弦フレネル積分　$C(x)$
CALL CFRI (X, CF, ICON)

(1)　機能

余弦フレネル積分$C(x)$ の値

$$C(x)=\frac{1}{\sqrt{2\pi}}\int_0^x\frac{\cos(t)}{\sqrt{t}}dt=\int_0^{\sqrt{\frac{2}{\pi}}x}\cos\left(\frac{\pi}{2}t^2\right)dt$$

を多項式及び漸近形式の近似式を用いて計算する.
ただし，$x\geq 0$であること.

(2)　パラメタ

X …… **入力**. 独立変数x.
CF …… **出力**. 関数値$C(x)$.
ICON …… **出力**. コンディションコード.
表CFRI-1参照.

表 CFRI-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	X ≥ t_{max}であった.	CF = 0.5
30000	X<0であった.	CF = 0.0

(3)　使用上の注意

a.　使用する副プログラム
①　SSL II ... MGSSL, UTLIM
②　FORTRAN 基本関数 ... SIN, COS, SQRT.

b.　注意
①　引数 X の範囲は

$$0\leq X<t_{max}$$

であること.
Xが大きくなるとsin(x), cos(x) の値が正確に計算出来なくなることに対する処理である.
(手法概要 (4.4) 参照).

c.　使用例
$C(x)$ の値を x が 0 から 100 まで，1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=K-1
      CALL CFRI(X,CF,ICON)
      IF(ICON.EQ.0)WRITE(6,610)X,CF
      IF(ICON.NE.0)WRITE(6,620)X,CF,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF FRESNEL ',
     *'INTEGRAL'///6X,'X',9X,'C(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     *E17.7,5X,'C=',E17.7,5X,
     *'CONDITION=',I10)
      END
```

(4)　手法概要

余弦フレネル積分$C(x)$ の計算は，x = 4を境にして，近似式の形が異なる.

a.　$0\leq x<4$ の場合
$C(x)$ のべき級数展開

$$C(x)=\sqrt{\frac{2}{\pi}}x\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n)!(4n+1)}x^{2n} \qquad (4.1)$$

を次の近似式を用いて計算する.

単精度：$C(x)=\sqrt{x}\sum_{k=0}^{7}a_k z^{2k}, z=x/4$ (4.2)

倍精度：$C(x)=\sqrt{x}\sum_{k=0}^{12}a_k x^{2k}$ (4.3)

b.　$x\geq 4$ の場合
$C(x)$ の漸近展開

$$C(x)=\frac{1}{2}+\sin(x)P(x)+\cos(x)Q(x) \qquad (4.4)$$

より計算する．ただし，$P(x)$, $Q(x)$ は次の近似式を用いる.

単精度：$P(x)=\frac{2}{\sqrt{x}}\sum_{k=0}^{11}a_k z^k,\quad z=4/x$ (4.5)

$$Q(x)=\frac{2}{\sqrt{x}}\sum_{k=0}^{10}b_k z^{k+1},\quad z=4/x \qquad (4.6)$$

倍精度：$P(x)=\frac{-1}{\sqrt{x}}\sum_{k=0}^{10}a_k z^k\Big/\sum_{k=0}^{10}b_k z^k,\quad z=4/x$ (4.7)

$$Q(x)=\frac{1}{\sqrt{x}}\sum_{k=0}^{10}c_k z^{k+1}\Big/\sum_{k=0}^{11}d_k z^k,\quad z=4/x \qquad (4.8)$$

F12-15-0101 CFT, DCFT

多次元離散型複素フーリエ変換 (8, 2基底FFT)
CALL CFT (A, B, N, M, ISN, ICON)

(1) 機能

*M*次元 (各元の項数*N* l, *N* 2,⋯, *NM*) の複素時系列データ $\{x_{J1,\ldots,JM}\}$ が与えられたとき，離散型複素フーリエ変換，又はその逆変換を，高速変換手法 (FFT) により行う．ただし，*N*l, *N* 2,⋯, *NM*は各々 2^l (*l*: 0又は正整数) であること．また，$M \geq 1$であること．

① フーリエ変換

$\{x_{J1,\ldots,JM}\}$を入力し，(1.1) で定義する変換を行い$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$ を求める．

$$
\begin{aligned}
N1\ldots NM\alpha_{K1,\ldots,KM} = & \sum_{J1=0}^{N1-1}\cdots\sum_{JM=0}^{NM-1} x_{J1,\ldots,JM}\omega_1^{-J1.K1} \\
& \ldots\omega_M^{-JM.KM} \\
& ,K1 = 0,1,\ldots,N1-1 \\
& \ldots \\
& ,KM = 0,1,\ldots,NM-1 \\
& ,\omega_1 = \exp(2\pi i/N1) \\
& \ldots \\
& ,\omega_M = \exp(2\pi i/NM)
\end{aligned}
\qquad (1.1)
$$

② フーリエ逆変換

$\{\alpha_{K1,\ldots,KM}\}$を入力し，(1.2) で定義する変換を行い$\{x_{J1,\ldots,JM}\}$を求める．

$$
\begin{aligned}
x_{J1,\ldots,JM} = & \sum_{K1=0}^{N1-1}\cdots\sum_{KM=0}^{NM-1}\alpha_{K1,\ldots,KM}\cdot\omega_1^{J1.K1}\ldots\omega_M^{JM\cdot KM} \\
& ,J1 = 0,1,\ldots,N1-1 \\
& \ldots \\
& ,JM = 0,1,\ldots,NM-1 \\
& ,\omega_1 = \exp(2\pi i/N1) \\
& \ldots \\
& ,\omega_M = \exp(2\pi i/NM)
\end{aligned}
\qquad (1.2)
$$

(2) パラメタ

A ····· **入力**．$\{x_{J1,\ldots,JM}\}$又は$\{\alpha_{K1,\ldots,KM}\}$の実部．
出力．$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$又は，$\{x_{J1,\ldots,JM}\}$の実部．M次元配列．
(使用上の注意②参照)

B ····· **入力**．$\{x_{J1,\ldots,JM}\}$又は，$\{\alpha_{K1,\ldots,KM}\}$の虚部．
出力．$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$又は，$\{x_{J1,\ldots,JM}\}$の虚部．M次元配列．
(使用上の注意②参照)

N ····· **入力**．M次元変換の項数を，N (1) = *N*1, N(2) = *N*2, … N(M) = *NM*と与える．
大きさMの1次元配列．

M ····· **入力**．次元数*M*.

ISN ····· **入力**．変換か逆変換かを指定する (≠ 0)
変換　: ISN = +1
逆変換　: ISN = –1
(使用上の注意③参照)

ICON ····· **出力**．コンディションコード．
表CFT-1参照．

表 CFT-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	M < 1, ISN = 0又は*N*1, *N*2, …, *NM*のいずれかが2^l (*l* : 0又は正整数)	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II … CFTN, PNR, MGSSL

② FORTRAN 基本関数 … ATAN, ALOG, SQRT, SIN, IABS

b. 注意

① 一般的なフーリエ変換の定義について多次元離散型複素フーリエ変換及び，逆変換は一般的に (3.1), (3.2) で定義される．

$$
\alpha_{K1,\ldots,KM} = \frac{1}{N1\ldots NM}\sum_{J1=0}^{N1-1}\cdots \sum_{JM=0}^{NM-1} x_{J1,\ldots,JM}\omega_1^{-J1\cdot K1}\ldots\omega_M^{-JM\cdot KM} \qquad (3.1)
$$

$$
x_{J1,\ldots,JM} = \sum_{K1=0}^{N1-1}\cdots\sum_{KM=0}^{NM-1}\alpha_{K1,\ldots,KM}\omega_1^{J1\cdot K1}\ldots\omega_M^{JM\cdot KM} \qquad (3.2)
$$

ここで K1,…,KM,J1,…,JM,$\omega_1,\cdots,\omega_M$ の定義は (1.1), (1.2) に準じる．

本サブルーチンでは，(3.1), (3.2) の左辺に対応して$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$又は，$\{x_{J1,\ldots,JM}\}$を求めるので結果の正規化は必要に応じて行うこと．

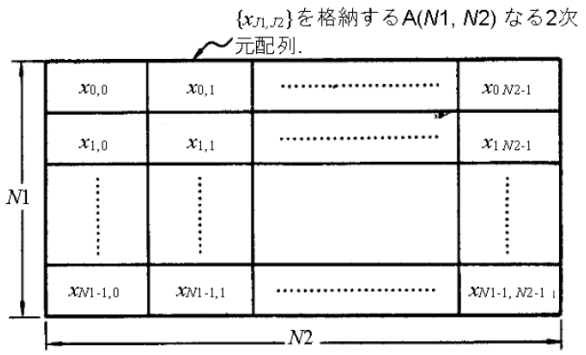
ちなみに，本サブルーチンにより変換・逆変換を正規化せずに連続して実行すると，入力データの各要素は $N1\cdots NM$ 倍されて出力される．

② データの格納方法について．

入力$\{x_{J1,\ldots,JM}\}$の実部は，M 次元配列 A に図 CFT-1 で示されるように格納する．虚部も，M 次元配列 B に同様に格納する．

また，入力$\{\alpha_{K1,\ldots,kM}\}$も同様に格納する．

一方，出力$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$，$\{x_{J1,\ldots,JM}\}$も同様に与えられる．



2 次元変換 (M=2) の例.

図CFT-1 {$x_{J1,\ldots,JM}$}格納方法

一般に M 次元の場合，FORTRAN で扱う M 次元配列と同様な格納順序を持つならば，パラメタ A, B は 1 次元配列でもよい.

③ ISN の与え方について

ISNでは，変換か逆変換かを指定するが，更に次のように使い分けることができる.

すなわち，{$x_{J1,\ldots,JM}$}又は{$\alpha_{K1,\ldots,KM}$}の実部，虚部が，それぞれ間隔 I で格納されている場合は，次のように指定する.

変換　: ISN = $+I$
逆変換 : ISN = $-I$

この場合，変換結果も間隔 I で格納される.

c. 使用例

① l 次元変換の場合

N l項の複素時系列データ{x_{J1}}を入力して，フーリエ変換し結果の{$N1\alpha_{K1}$}を求める.

Nl ≤ 1024 (= 2^{10}) の場合.

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
      CALL CFT(A,B,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,620) (I,A(I),B(I),I=1,N)
      STOP
  500 FORMAT(I5/(2E20.7))
  600 FORMAT('0',10X,'INPUT DATA N=',I5/
     *(15X,I5,2E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5)
  620 FORMAT(15X,I5,2E20.7)
      END
```

② 3 次元変換の場合

N l, N 2, N 3項の複素時系列データ{$x_{J1,J2,J3}$}を入力し，フーリエ変換し，その結果をフーリエ逆変換して{$x_{J1,J2,J3}$}を求める.

ここでは，N l = 2, N 2 = 4, N 3 = 4とする.

```
C     **EXAMPLE**
      DIMENSION A(2,4,4),B(2,4,4),N(3)
      DATA N/2,4,4/
      READ(5,500)(((A(I,J,K),B(I,J,K),
     *            I=1,2),J=1,4),K=1,4)
      WRITE(6,600)N,(((I,J,K,
     *            A(I,J,K),B(I,J,K),
     *            I=1,2),J=1,4),K=1,4)
C     NORMAL TRANSFORM
      CALL CFT(A,B,N,3,1,ICON)
      IF(ICON.NE.0)STOP
C     INVERSE TRANSFORM
      CALL CFT(A,B,N,3,-1,ICON)
      IF(ICON.NE.0)STOP
      NT=N(1)*N(2)*N(3)
      DO 10 I=1,2
      DO 10 J=1,4
      DO 10 K=1,4
      A(I,J,K)=A(I,J,K)/FLOAT(NT)
      B(I,J,K)=B(I,J,K)/FLOAT(NT)
   10 CONTINUE
      WRITE(6,610)(((I,J,K,
     *            A(I,J,K),B(I,J,K),
     *            I=1,2),J=1,4),K=1,4)
      STOP
  500 FORMAT(8E5.0)
  600 FORMAT('0',10X,'INPUT DATA',5X,
     *'(',I3,',',I3,',',I3,')'/
     *(15X,'(',I3,',',I3,',',I3,')',
     *2E20.7))
  610 FORMAT('0',10X,'OUTPUT DATA'/
     *(15X,'(',I3,',',I3,',',I3,')',
     *2E20.7))
      END
```

(4) 手法概要

多次元離散型複素フーリエ変換を，8及び2を基底とする高速変換手法 (高速フーリエ変換・FFT) により行う.

① 多次元変換

(3.1) で定義される多次元変換は，共通項を整理する事により，2次元の場合 (4.1) のように変形できる.

$$\alpha_{K1,K2} = \sum_{J1=0}^{N1-1} \omega_1^{-J1,K1} \sum_{J2=0}^{N2-1} x_{J1,J2} \omega_2^{-J2,K2} \qquad (4.1)$$

ただし，(4.1) では，通常の正規化定数 1/N l·N 2 は省略している.

(4.1) において$\sum_{J2}$は N 2 項の 1 次元変換を $J1$ に関して，$N1$ 組行い，その結果に対して$\sum_{J1}$は $N1$ 項の 1 次元変換を $J2$ に関して $N2$ 組行う.

同様にして，多次元離散型複素フーリエ変換は，1 次元変換を各元において複数組ずつ行うことにより達成される．本サブルーチンでは，各元に対する 1 次元変換を行うために，8 及び 2 を基底とする高速複素フーリエ変換を適用している.

② 高速フーリエ変換の原理

1次元の離散型複素フーリエ変換を，(4.2) のように定義する．

$$\alpha_K = \sum_{j=0}^{n-1} x_j \omega^{-jk}, k = 0,1,...,n-1 \quad , \omega = \exp\left(2\pi i/n\right) \tag{4.2}$$

ただし (4.2) では，通常の正規化定数 1/ n は省略している．

(4.2) を直接計算する場合，その複素数の乗算回数は n^2 となる．ところで今 $n = r \cdot r$ なる場合，指数関数ω^{-jk} の性質を考慮し計算すると，乗算回数は $2nr$ 程度になることが知られている．

ところで，(4.2) における k, j は (4.3) のように表現できる．

$$\left.\begin{aligned} j &= j_0 + r \cdot j_1, 0 \le j_0 \le r-1, 0 \le j_1 \le r-1 \\ k &= k_0 + r \cdot k_1, 0 \le k_0 \le r-1, 0 \le k_1 \le r-1 \end{aligned}\right\} \tag{4.3}$$

(4.3)を(4.2)に代入すると(4.4)となる．

$$\alpha_{k0+r \cdot k1} = \sum_{j_0=0}^{r-1}\sum_{j=0}^{r-1} x_{j_0+r \cdot j_1} \cdot \exp\left\{-2\pi i \frac{\left(j_0 + r \cdot j_1\right)\left(k_0 + r \cdot k_1\right)}{r^2}\right\} \tag{4.4}$$

(4.4)の右辺を $j0$ 及び $j1$ に関して整理し，共通項をまとめると(4.5)となる．

$$\alpha_{k0+r \cdot k} = \sum_{j_0=0}^{r-1} \exp\left(-2\pi i \frac{j_0 \cdot k_1}{r}\right) \cdot \exp\left\{-2\pi i \frac{j_0 \cdot k_1}{r^2}\right\} \sum_{j_1=0}^{r-1} x_{j_0+r \cdot j_1} \exp\left(-2\pi i \frac{j_1 \cdot k_0}{r}\right) \tag{4.5}$$

(4.5)の計算で $\sum_{j1}$ 及び $\sum_{j0}$ の項は，部分的な r 項のフーリエ変換を r 組行っていることがわかる．また $\exp\left(-2\pi i \frac{j_0 \cdot k_0}{r^2}\right)$ は $\sum_{j1}$ の結果に対する回転因子である．したがって (4.5) の計算での乗算回数 C_n は (4.6) となり，n が大きい場合にその計算量が減少することがわかる．

$$\begin{aligned} C_n &= r \cdot \left(r^2\right) + r \cdot \left(r^2\right) + \left(r-1\right)\left(r-1\right) \\ &= 2nr + \left(r-1\right)^2 \end{aligned} \tag{4.6}$$

更に r が因数分解されるなら，より効率よく計算することができる．

なお，具体例がCFTNに示されているので参照すること．また，詳細については，参考文献 [55], [56] 及び，[57] を参照すること．

F12-11-0101 CFTM, DCFTM

多次元離散型複素フーリエ変換 (混合基底FFT)
CALL CFTM (A, B, N, M, ISN, ICON)

(1) 機能

*M*次元 (各元の項数*N*1, *N*2, …, *NM*) の複素時系列データ$\{x_{J1,\ldots,JM}\}$が与えられたとき，離散型複素フーリエ変換，又はその逆変換を，高速変換手法 (FFT) により行う．ただし，各元の項数は1若しくは，次の条件を満たす数であること．

- 素因数 p ($p\in\{4, 3, 5, 7,11,13,17,19,23,2\}$)の積 (同一素因数が重複してもよい) で表せること．
- 素因数の個数は，最大 11 個であること．
- 2 乗因数で除した残りの因数 (square free factors) の積は，最大 210 であること．

また，$M\geq 1$であること．

① フーリエ変換

$\{x_{J1,\ldots,JM}\}$を入力し，(1.2) で定義する変換を行い$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$を求める．

$$
\begin{aligned}
&N1\ldots NM\alpha_{K1,\ldots,KM}\\
&=\sum_{J1=0}^{N1-1}\cdots\sum_{JM=0}^{NM-1}x_{J1,\ldots,JM}\,\omega_1^{-J1\cdot K1}\cdots\omega_M^{-JM\cdot KM}\\
&,K1=0,1,\ldots,N1-1\\
&\ldots\\
&,KM=0,1,\ldots,NM-1\\
&,\omega_1=\exp(2\pi i/N1),\ldots,\omega_M=\exp(2\pi i/NM)
\end{aligned}
\qquad (1.2)
$$

② フーリエ逆変換

$\{\alpha_{K1,\ldots,KM}\}$を入力し，(1.3) で定義する変換を行い$\{x_{J1,\ldots,JM}\}$を求める．

$$
\begin{aligned}
&x_{J1,\ldots,JM}=\sum_{K1=0}^{N1-1}\cdots\sum_{KM=0}^{NM-1}a_{K1,\ldots,KM}\,\omega_1^{J1\cdot K1}\cdots\omega_M^{JM\cdot KM}\\
&,J1=0,1,\ldots,N1-1\\
&\ldots\\
&,JM=0,1,\ldots,NM-1\\
&,\omega_1=\exp(2\pi i/N1),\ldots,\omega_M=\exp(2\pi i/NM)
\end{aligned}
\qquad (1.3)
$$

(2) パラメタ

A ····· **入力**．$\{x_{J1,\ldots,JM}\}$又は，$\{\alpha_{K1,\ldots,KM}\}$の実部．
出力．$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$又は，$\{x_{J1,\ldots,JM}\}$の実部．*M*次元配列．
(使用上の注意③参照)

B ····· **入力**．$\{x_{J1,\ldots,JM}\}$又は，$\{\alpha_{K1,\ldots,KM}\}$の虚部．
出力．$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$又は，$\{x_{J1,\ldots,JM}\}$の虚部．*M*次元配列．
(使用上の注意③参照)

N ····· **入力**．*M*次元変換の項数を，N(1) = *N*1, N(2) = *N*2 …, N(M) = *NM*と与える．
大きさMの1次元配列．

M ····· **入力**．次元数*M*.

ISN ····· **入力**．変換か逆変換かを指定する (≠ 0).
変換　: ISN = +1
逆変換 : ISN = –1
(使用上の注意④参照)

ICON ····· **出力**．コンディションコード．
表 CFTM-1参照．

表CFTM-1 コンディションコード

コード	意味		処理内容
0	エラーなし．		
29100	項数*N*1,…, *N*Mのいずれかが，23以上の素因数*r*を持つ．	かつその項数*N*が$N \pmod{r^2}=0$	処理を打ち切る．
29200		かつその項数*N*が$N \pmod{r}=0$	
29300	素因数の個数が11個以上である．		
29400	2乗因数で除した残りの因数の積が210以上である．		
30000	M ≤ 0, ISN = 0 又はいずれかの項数≤ 0であった．		

(3) 使用上の注意

a. 使用する副プログラム

① SSL II … MGSSL, UCFTM

② FORTRAN 基本関数 … ATAN, COS, SIN, SQRT, MOD, FLOAT

b. 注意

① 一般的なフーリエ変換の定義について

多次元離散型複素フーリエ変換及び，逆変換は一般的に (3.1), (3.2)で定義される．

$$a_{K1,\ldots,KM}=\frac{1}{N1\ldots NM}\sum_{J1=0}^{N1-1}\cdots\sum_{JM=0}^{NM-1}x_{J1,\ldots,JM}\,\omega_1^{-J1\cdot K1}\cdots\omega_M^{-JM\cdot KM}\qquad (3.1)$$

$$x_{J1,\ldots,JM}=\sum_{K1=0}^{N1-1}\cdots\sum_{KM=0}^{NM-1}a_{k1,\ldots,KM}\,\omega_1^{J1\cdot K1}\cdots\omega_M^{JM\cdot KM}\qquad (3.2)$$

ここで $K1, \ldots, KM, J1, \ldots, JM, \omega_1,\ldots,\omega_M$ の定義は (1.2), (1.3) に準じる．

本サブルーチンでは，(3.1), (3.2) の左辺に対応して$\{N1\cdots NM\alpha_{K1,\ldots,KM}\}$又は，$\{x_{J1,\ldots,JM}\}$を求めるので結果の正規化は必要に応じて行うこと．

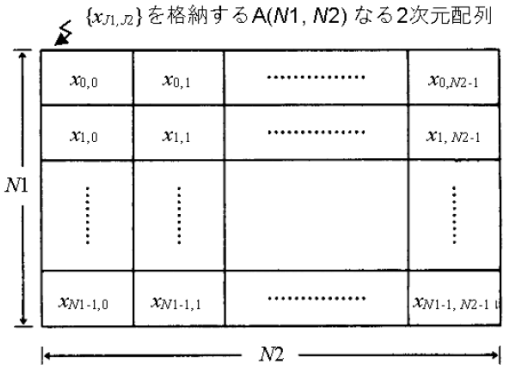
ちなみに，本サブルーチンにより変換･逆変換を正規化せずに連続して実行すると，入力データの各要素は $N1\ldots NM$ 倍されて出力される．

② 項数の決定と処理速度について
各元の項数を決定する場合，「**(1)機能**」で示された条件を満たせばよいが，可能な限り，素因数が5以下 ($p \le 5$) となるように決定することが望ましい．5以上 ($p > 5$) の素因数を持つ場合よりも，一般的に処理速度が速い．ちなみに,素因数が5以下で表現できる10000以下の数は,表CFTM-2で示される．
なお，項数が 2 の巾 (2^I, $I \ge 0$ なる整数) で表現できる場合には，サブルーチン CFT の方が処理速度が速い．

表 CFTM-2 5以下の素因数で表現できる10000以下の数

2	90	405	1215	2916	6075
3	96	432	1250	3000	6144
4	100	450	1280	3072	6250
5	108	480	1296	3125	6400
6	120	486	1350	3200	6480
8	125	500	1440	3240	6561
9	128	512	1458	3375	6750
10	135	540	1500	3456	6912
12	144	576	1536	3600	7200
15	150	600	1600	3645	7290
16	160	625	1620	3750	7500
18	162	640	1728	3840	7680
20	180	648	1800	3888	7776
24	192	675	1875	4000	8000
25	200	720	1920	4050	8100
27	216	729	1944	4096	8192
30	225	750	2000	4320	8640
32	240	768	2025	4374	8748
36	243	800	2048	4500	9000
40	250	810	2160	4608	9216
45	256	864	2187	4800	9375
48	270	900	2250	4860	9600
50	288	960	2304	5000	9720
54	300	972	2400	5120	10000
60	320	1000	2430	5184	
64	324	1024	2500	5400	
72	360	1080	2560	5625	
75	375	1125	2592	5760	
80	384	1152	2700	5832	
81	400	1200	2880	6000	

③ データの格納方法について
入力$\{x_{J1,\ldots,JM}\}$の実部は，M 次元配列 A に図 CFTM-1 で示されるように格納する．虚部も，M次元配列 B に同様に格納する．
また，入力$\{\alpha_{K1,\ldots,KM}\}$も同様に格納する．
一方，出力$\{N1 \cdot N2 \cdots NM\alpha_{K1,\ldots,KM}\}$, $\{x_{J1,\ldots,JM}\}$も同様に与えられる．



2次元変換 (M=2) の例

図 CFTM-1 $\{x_{J1,\ldots,JM}\}$の格納方法

一般には M 次元の変換を行う場合，データの並びが FORTRAN で扱う M 次元配列と同様な格納順序を持つならば，パラメタ A, B は 1 次元配列でもよい．

④ ISN の与え方について
ISNでは，変換か逆変換かを指定するが，さらに次のように使い分けることができる．
すなわち，$\{x_{J1,\ldots,JM}\}$又は$\{\alpha_{K1,\ldots,KM}\}$の実部，虚部が配列A, B上に間隔$I$で格納されている場合は，次のように指定する．
変換 ：ISN = $+I$
逆変換 ：ISN = $-I$
この場合，変換結果も間隔 I で格納される．

c. 使用例

① 1 次元変換の場合
$N1$項の複素時系列データ$\{x_{j1}\}$を入力して，フーリエ変換し結果の$\{N1\alpha_{K1}\}$を求める．
$N1 \le 1000$の場合．

```
C     **EXAMPLE**
      DIMENSION A(1000),B(1000)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
      CALL CFTM(A,B,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,620) (I,A(I),B(I),I=1,N)
      STOP
500 FORMAT(I5/(2E20.7))
600 FORMAT('0',10X,'INPUT DATA N=',I5/
     *      (15X,I5,2E20.7))
610 FORMAT('0',10X,'RESULT   ICON=',
     *       I5)
620 FORMAT('0',10X,'OUTPUT DATA'/
     *      (15X,I5,2E20.7))
      END
```

② 3 次元変換の場合
$N1$，$N2$, $N3$項の複素時系列データ$\{x_{J1,J2,J3}\}$を 入力し，フーリエ変換し，その結果をフーリエ逆変換して$\{x_{J1,J2,J3}\}$を求める．
ここでは，$N1 = 5$, $N2 = 12$, $N3 = 7$とする．

```
C     **EXAMPLE**
      DIMENSION A(5,12,7),B(5,12,7),N(3)
      DATA N/5,12,7/
      READ(5,500) (((A(I,J,K),B(I,J,K),
     *             I=1,N(1)),J=1,N(2)),
     *             K=1,N(3))
      WRITE(6,600) N,(((I,J,K,A(I,J,K),
     *             B(I,J,K),I=1,N(1)),
     *             J=1,N(2)),K=1,N(3))
C     NORMAL TRANSFORM
      CALL CFTM(A,B,N,3,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0)STOP
C     INVERSE TRANSFORM
      CALL CFTM(A,B,N,3,-1,ICON)
      NT=N(1)*N(2)*N(3)
      DO 10 K=1,N(3)
      DO 10 J=1,N(2)
      DO 10 I=1,N(1)
      A(I,J,K)=A(I,J,K)/FLOAT(NT)
      B(I,J,K)=B(I,J,K)/FLOAT(NT)
   10 CONTINUE
      WRITE(6,620) (((I,J,K,A(I,J,K),
     *             B(I,J,K),I=1,N(1)),
     *             J=1,N(2)),K=1,N(3))
      STOP
  500 FORMAT(2E20.7)
  600 FORMAT('0',10X,'INPUT DATA',5X,
     * '(',I3,',',I3,',',I3,')'/
     * (15X,'(',I3,',',I3,',',I3,')',
     * 2E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5)
  620 FORMAT('0',10X,'OUTPUT DATA'/
     * (15X,'(',I3,',',I3,',',I3,')',
     *  2E20.7))
      END
```

(4) 手法概要

多次元離散型複素フーリエ変換を，素因数p ($2 \le p \le 23$) を基底とする高速変換手法 (高速フーリエ変換･FFT) により行う.

① 多次元変換

(1.1) で定義される多次元変換は，共通項を整理することにより，2次元の場合(4.1) のように変形できる.

$$N1\cdot N2\alpha_{K1,K2}=\sum_{J1=0}^{N1-1}\omega_1^{-J1\,K1}\sum_{J2=0}^{N2-1}x_{J1,J2}\,\omega_2^{-J2\,K2} \tag{4.1}$$

(4.1) において$\sum_{J2}$は $N2$ 項の 1 次元変換を $J1$ に関して $N1$ 組行い，その結果に対して$\sum_{J1}$は $N1$ 項の 1 次元変換を $J2$ に関して $N2$ 組行う.

同様にして，多次元離散型複素フーリエ変換は，1 次元変換を各元において複数組ずつ行うことにより達成される．本ルーチンでは，各元に対する 1 次元変換を行うために，素因数 p を基底とする高速フーリエ変換を適用している.

② 混合基底の高速フーリエ変換の原理

1次元の離散型複素フーリエ変換を，(4.2) のように定義する.

$$\alpha_k=\sum_{j=0}^{n-1}x_j\omega^{-jk},k=0,1,...,n-1 \qquad ,\omega=exp(2\pi i/n) \tag{4.2}$$

ただし (4.2) では，通常の正規化定数 1/ n は省略している．(4.2) を直接計算する場合，その複素数の乗算回数は n^2 となる.

ところで n が任意の因数 r, q により, $n = r\cdot q$ と表せる場合は，指数関数ω^{jk} の性質を考慮すると，乗算回数は $n\,(r+q)$ 程度になる.

今，(4.2) における k, j を (4.3) のように表現する.

$$k = k_0 + k_1\cdot q,\ \ 0 \le k_0 \le q - 1,\ \ 0 \le k_1 \le r - 1$$
$$j = j_0 + j_1\cdot r,\ \ 0 \le j_0 \le r - 1,\ \ 0 \le j_1 \le q - 1 \tag{4.3}$$

(4.3) を (4.2) に代入すると (4.4) となる.

$$\alpha_{k_0+k_1\cdot q}=\sum_{j_0=0}^{r-1}\sum_{j_1=0}^{q-1}x_{j_0+j_1\cdot r}\cdot exp\left\{-2\pi j\frac{(k_0+k_1\cdot q)(j_0+j_1\cdot r)}{r\cdot q}\right\} \tag{4.4}$$

(4.4) の右辺を j_0 及び j_1 に関して整理し共通項をまとめると (4.5) を得る.

$$\alpha_{k_0+k_1\,q}=\sum_{j_0=0}^{r-1}\exp\left\{-2\pi i\frac{k_1 j_0}{r}\right\}\exp\left\{-2\pi i\frac{k_0 j_0}{r}\right\}\cdot\sum_{j_1=0}^{q-1}\exp\left\{-2\pi i\frac{k_0 j_1}{q}\right\}x_{j_0+j_1\,r} \tag{4.5}$$

(4.5) の計算で$\sum_{j1}$は q 項のフーリエ変換を j_0 に関して r 組，また$\sum_{j0}$は r 項のフーリエ変換を q 組行っていることが分かる．また $exp\{-2\pi I\cdot k_0 j_0/n\}$は$\sum_{j1}$の結果に対する回転因子である．したがって (4.5) の計算での乗算回数 C_nは (4.6) となり，n が大きい場合にその計算量は，(4.6) を直接計算する場合の n^2 に比して，減少することが分かる.

$$C_n = r\cdot q^2 + q\cdot r^2 + (q-1)(r-1) = n(r+q+1)-(r+q)+1 \tag{4.6}$$

この変換は，基底 r 及び q の高速フーリエ変換という．r, q が更に素因数分解されるなら，より効率よく計算することができる．本サブルーチンでは，素因数として最大 23 を基底とする，混合基底の高速フーリエ変換手法を導入している．

詳細については，参考文献 [57] を参照すること．

F12-15-0202 CFTN, DCFTN

離散型複素フーリエ変換 (8, 2基底FFT, 逆順出力)
CALL CFTN (A, B, NT, N, NS, ISN, ICON)

(1) 機能

1次元 (項数n) の複素時系列データ$\{x_j\}$が与えられたとき，離散型複素フーリエ変換，又はその逆変換を高速変換手法 (FFT) により行う．
ただし，nは$n=2^l$(l : 0 又は正整数) であること．

a. フーリエ変換
$\{x_i\}$を入力し，(1.1) で定義する変換を行い，$\{n\tilde{\alpha}_k\}$ を求める．

$$n\tilde{\alpha}_k = \sum_{j=0}^{n-1} x_j \omega^{-jk}, k = 0,1,...,n-1, \quad \omega = \exp(2\pi i/n) \tag{1.1}$$

b. フーリエ逆変換
$\{\alpha_k\}$を入力し，(1.2) で定義する変換を行い，$\{\tilde{x}_k\}$ を求める．

$$\tilde{x}_j = \sum_{k=0}^{n-1} a_k \omega^{jk}, j = 0,1,...,n-1, \quad \omega = \exp(2\pi i/n) \tag{1.2}$$

ここで，$\{\tilde{\alpha}_k\}, \{\tilde{x}_j\}$ は，データを入力した領域上で変換を行うために，変換されたデータの並びが正順となっていないことを意味する．すなわち，(3.1), (3.2) で定義されるフーリエ変換，又はその逆変換の結果$\{\alpha_k\}$, $\{x_j\}$の並びを正順としたとき，$\{\tilde{\alpha}_k\}$, $\{\tilde{x}_j\}$ はビット逆転の意味で逆順となる．

(2) パラメタ

A …… **入力**．$\{x_j\}$又は，$\{\alpha_k\}$の実部．
出力．$\{n\tilde{\alpha}_k\}$ 又は，$\{\tilde{x}_j\}$ の実部．
大きさNTの1次元配列．

B …… **入力**．$\{x_j\}$又は，$\{\alpha_k\}$の虚部．
出力．$\{n\tilde{\alpha}_k\}$ 又は，$\{\tilde{x}_j\}$ の虚部．
大きさNTの1次元配列．

NT …… **入力**．変換の対象となる$\{x_j\}$又は，$\{\alpha_j\}$が含まれるデータの総数 (≥ N, NS)
通常，NT = Nとして指定する．
(使用上の注意③参照)

N …… **入力**．変換の項数n．

NS …… **入力**，NT個のデータのうち，n 項の変換の対象となる$\{x_j\}$又は，$\{\alpha_k\}$のあい隣り合うデータの間隔 (≥ 1かつ≤ NT)．
通常，NS = 1として指定する．
(使用上の注意③参照)

ISN …… **入力**．変換か逆変換かを指定する (≠ 0)．
変換　: ISN = +1
逆変換 : ISN = −1
(使用上の注意④参照)

ICON …… **出力**．コンディションコード．
表 CFTN-1参照．

表 CFTN-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	ISN = 0, NS < 1, NT < N, NT < NS又は，$N \neq 2^l$ (l : 0 又は正整数) であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数...ATAN, ALOG, SQRT, SIN

b. 注意
① 一般的なフーリエ変換の定義について
離散型フーリエ変換及び，逆変換は一般的に (3.1), (3.2) で定義される．

$$a_k = \frac{1}{n}\sum_{j=0}^{n-1} x_j \omega^{-jk}, \quad k = 0,1,...,n-1 \tag{3.1}$$

$$x_j = \sum_{k=0}^{n-1} a_k \omega^{jk}, \quad j = 0,1,...,n-1 \tag{3.2}$$

ここで，$\omega = \exp(2\pi i/n)$

本サブルーチンでは，(3.1), (3.2) の左辺に対応して，$\{n\tilde{\alpha}_k\}$ 又は$\{\tilde{x}_j\}$ を求める．ここで，$\{\tilde{\alpha}_k\}$, $\{\tilde{x}_j\}$ はデータを入力した領域上で変換を行うために，変換されたデータの並びが正順となっていないことを意味する．すなわち，$\{n\tilde{\alpha}_k\}$ は$\{\alpha_k\}$に対して各要素が n 倍されており，かつそのデータの並びがビット逆転の意味で逆順となっている．一方，$\{\tilde{x}_j\}$は$\{x_j\}$に対してそのデータの並びがビット逆転の意味で逆順となっている．
したがって，結果の正規化及びデータの並べかえは必要に応じて行うこと．
データの並べかえは，サブルーチン PNR により行うことができる．
使用例①を参照すること．

② 本サブルーチンの用途について
通常，フーリエ変換又は，フーリエ逆変換を行う場合はサブルーチンCFTを用いればよいが，変換，逆変換を連続処理するような場合は，本サブルーチン及びサブルーチンCFTRを併用する事により，効率よく行うことができる．
すなわち，本サブルーチンによるフーリエ

変換の結果の$\{n\tilde{\alpha}_k\}$に対して，そのデータの並びを正順にもどすことなく必要な処理を行い，その後フーリエ逆変換をサブルーチンCFTRにより行う．
したがって，通常行われるデータの並べかえを必要としないために，処理速度が向上する．
ちなみに，CFTR では (3.3) (3.4) で定義されるフーリエ変換，又はその逆変換を行う．

$$n\alpha_k = \sum_{j=0}^{n-1} \tilde{x}_j \omega^{-jk}, \quad k = 0,1,...,n-1 \tag{3.3}$$

$$x_j = \sum_{k=0}^{n-1} \tilde{\alpha}_k \omega^{jk}, \quad j = 0,1,...,n-1 \tag{3.4}$$

使用例②を参照すること．

③ 多次元変換への応用について
本サブルーチンは多次元変換への応用が可能である．多次元変換は2次元の場合，(3.5), (3.6) でフーリエ変換及びフーリエ逆変換が定義される．

$$\alpha_{K1,K2} = \frac{1}{N1 \cdot N2} \sum_{J1=0}^{N1-1} \sum_{J2=0}^{N2-1} x_{J1,J2}\, \omega_1^{-J1 \cdot K1} \cdot \omega_2^{-J2 \cdot K2} \tag{3.5}$$
$$, K1 = 0,..., N1-1, K2 = 0,..., N2-1$$

$$x_{J1,J2} = \sum_{K1=0}^{N1-1} \sum_{K2=0}^{N2-1} \alpha_{K1,K2}\, \omega_1^{J1 \cdot K1} \cdot \omega_2^{J2 \cdot K2} \tag{3,6}$$
$$, J1 = 0,..., N1-1, \quad J2 = 0,..., N2-1$$

ここで，$\omega_1 = \exp(2\pi i / N1), \omega_2 = \exp(2\pi i / N2)$

(3.5) を変形すると，(3.7) となる．

$$\alpha_{K1,K2} = \frac{1}{N1 \cdot N2} \sum_{J1=0}^{N1-1} \omega_1^{-J1 \cdot K1} \sum_{J2=0}^{N2-1} x_{J1,J2} \omega_2^{-J2 \cdot K2} \tag{3.7}$$

多次元変換 (3.7) は，まず，$\sum_{J2}$により $N2$ 項の 1 次元変換を $J1$ に関して $N1$ 組行い、その結果に対して$\sum_{J1}$により $N1$ 項の 1 次元変換を $J2$ に関して $N2$ 組行うことにより達成されることがわかる．本サブルーチンでは，1 回の呼出しで$\sum_{J2}$による $N2$ 項の 1 次元変換を $N1$ 組行う．(又は，1 回の呼出しで$\sum_{J1}$による $N1$ 項の 1 次元変換を $N2$ 組行う．)
具体的には，以下のように指定し本サブルーチンを呼びだせばよい．

(a) $\{x_{j1,j2,j3}\}$を，2 次元配列 A 及び B に，図 CFTN-1 で示されるように格納する．

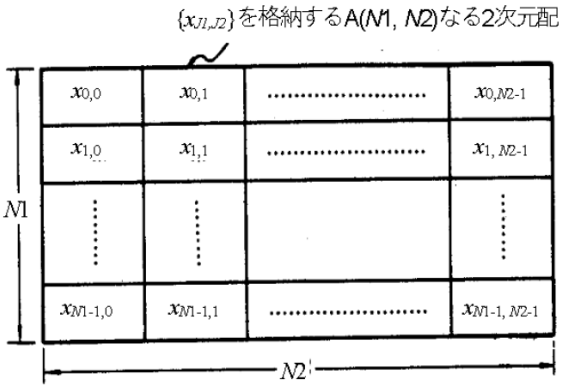


図 CFTN-1 $\{x_{j1,j2}\}$の格納方法

(b) 次のように，本サブルーチンを 2 回呼び出す．

```
   …
NT=N1*N2
NS=1
CALL CFTN(A,B,NT,N1,NS,1,ICON)
NS=N1
CALL CFTN(A,B,NT,N2,NS,1,ICON)
   …
```

ただし，これによって得られる結果は$\{N1 \cdot N2\tilde{\alpha}_{K1,K2}\}$であるので，1 次元変換の場合と同様に，正規化及びデータの並べかえは必要に応じて行うこと．
データの並べかえは，サブルーチン PNR により行うことができる．
(3.6) で定義される逆変換の場合も，同様に処理が可能である．
また 2 次元以上も可能であり，3 次元の場合については使用例③を参照すること．

④ ISN の与え方について
ISNでは，変換か逆変換かを指定するが，更に次のように使い分けることができる．すなわちNT個のデータの実部，虚部が，各々NT $\cdot I$なる大きさの領域に，間隔Iで格納されている場合は，次のように指定する．

変換　　: ISN = $+I$
逆変換　: ISN = $-I$

この場合，変換結果も間隔 I で格納される．

c. 使用例

① 1 次元変換の場合
n項の複素時系列データ$\{x_j\}$を入力して，フーリエ変換する．その結果のデータをサブルーチンNRにより並べかえ$\{n\alpha_k\}$を求める．

$n \leq 1024\ (=2^{10})$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
      CALL CFTN(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      CALL PNR(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,620) (I,A(I),B(I),I=1,N)
      STOP
  500 FORMAT(I5 /(2E20.7))
  600 FORMAT('0',10X,'INPUT DATA N=',I5/
     *      /(15X,I5,2E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5)
  620 FORMAT(15X,I5,2E20.7)
      END
```

② 変換･逆変換連続処理の場合

n項の複素時系列データ$\{x_j\}$入力して，本サブルーチンによりフーリエ変換し$\left\{n\tilde{\alpha}_k\right\}$を求める．$\left\{n\tilde{\alpha}_k\right\}$に対して，そのデータの並びを正順にもどすことなく必要な処理を行い (例中 …部)，その結果に対してサブルーチンCFTRによりフーリエ逆変換する．

$n \leq 1024\ (=2^{10})$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
C     NORMAL TRANSFORM
      CALL CFTN(A,B,N,N,1,1,ICON)
      IF(ICON.NE.0)STOP
         :
C     INVERSE TRANSFORM
      CALL CFTR(A,B,N,N,1,-1,ICON)
      IF(ICON .NE. 0)STOP
      DO 10 I=1,N
      A(I)=A(I)/FLOAT(N)
      B(I)=B(I)/FLOAT(N)
   10 CONTINUE
      WRITE(6,610) (I,A(I),B(I),I=1,N)
      STOP
  500 FORMAT(I5/(2E20.7))
  600 FORMAT('0',10X,'INPUT DATA N=',I5/
     *      /(15X,I5,2E20.7))
  610 FORMAT('0',10X,'OUTPUT DATA'/
     *      /(15X,I5 ,2E20.7))
      END
```

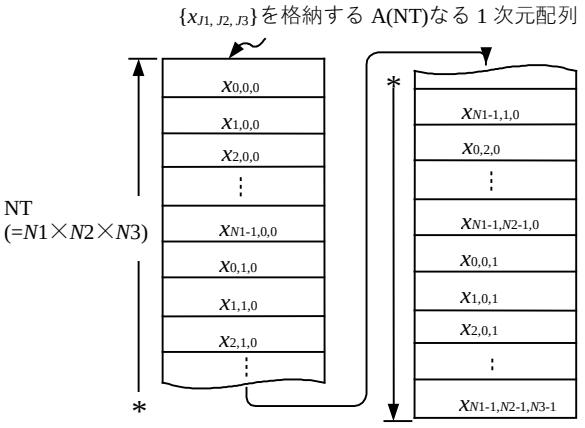
③ 3 次元変換の場合

$N1, N2, N3$項の複素時系列データ$\{x_{J1,J2,J3}\}$を入力して，本サブルーチンによりフーリエ変換し$\{N1 \cdot N2 \cdot N3\tilde{\alpha}_{K1,K2,K3}\}$を求める．

$\{N1 \cdot N2 \cdot N3\tilde{\alpha}_{K1,K2,K3}\}$に対して，そのデータの並びを正順にもどすことなく必要な処理を行い (例中…部)，その結果に対してサブルーチンCFTRによりフーリエ逆変換する．

$N1 \cdot N2 \cdot N3 \leq 1024\ (=2^{10})$ の場合.

なおデータは，図 CFTN-2で示されるように，1次元配列上に格納してもよい.



[注意]このように格納した場合のNSの与え方は，3回の呼び出し時に順に，1, $N1$, $N1$*$N2$とすればよい.

図 CFTN-2 $\{x_{J1,J2,J3}\}$の格納方法

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024),N(3)
      READ(5,500) N
      NT=N(1)*N(2)*N(3)
      READ(5,510) (A(I),B(I),I=1,NT)
      WRITE(6,600) N,(I,A(I),B(I),I=1,
     *             NT)
C     NORMAL TRANSFORM
      NS=1
      DO 10 I=1,3
      CALL CFTN(A,B,NT,N(I),NS,l,ICON)
      IF(ICON.NE.0) STOP
      NS=NS *N(I)
   10 CONTINUE
         :
C     INVERSE TRANSFORM
      NS=1
      DO 20 I=1,3
      CALL CFTR(A,B,NT,N(I),NS,-1,ICON)
      IF(ICON.NE.0) STOP
      NS=NS*N(I)
   20 CONTINUE
C     NORMALIZATION
      DO 30 I=1,NT
      A(I)=A(I)/FLOAT(NT)
      B(I)=B(I)/FLOAT(NT)
   30 CONTINUE
      WRITE(6,610) (I,A(I),B(I),I=1,NT)
      STOP
  500 FORMAT(3I5)
  510 FORMAT(2E20.7)
  600 FORMAT('0',10X,'INPUT DATA N=',
     * 3I5//(15X,I5,2E20.7))
  610 FORMAT('0,10X,'OUTPUT DATA'/
     * /(15X,I5,2E20.7))
      END
```

(4) 手法概要

離散型複素フーリエ変換を，8及び2を基底とする高速変換手法 (高速フーリエ変換･FFT) により行う．ただし，変換結果は，ビット逆転の意味で逆順とする．高速フーリエ変換の原理については，サブルーチンCFTの項を参照すること.

ここでは，n = 16なる場合の具体例について説明する．まず，この場合フーリエ変換は (4.1) に表される．

$$\alpha_k = \sum_{j=0}^{15} x_j \omega^{-jk}, \quad k = 0,1,...15, \qquad \omega = \exp(2\pi i/16) \tag{4.1}$$

ただし (4.1) では通常の正規化定数1/16は省略している．本サブルーチンでは，nを8と2の因数に分解する ($n = 8 \times 2$).
さて，k, j は (4.2) のように表わすことができる．

$$k = k_0 + k_1 \cdot 8, 0 \le k_0 \le 7, 0 \le k_1 \le 1$$
$$j = j_0 + j_1 \cdot 2, 0 \le j_0 \le 1, 0 \le j_1 \le 7 \tag{4.2}$$

(4.2) を (4.1) に代入し，共通項を整理すると (4.3) となる．ただし，$\alpha(k_0 + k_1 \cdot 8) \equiv a_{k_0 + k_1 8}$ と表示する．

$$\alpha(k_0 + k_1 \cdot 8) = \sum_{j_0=0}^{1} \exp\left(-2\pi i \frac{j_0 k_1}{2}\right) \cdot \exp\left(-2\pi i \frac{j_0 \cdot k_0}{16}\right) \sum_{j_1=0}^{7} \exp\left(-2\pi i \frac{j_1 k_0}{8}\right) \cdot x(j_0 + j_1 \cdot 2) \tag{4.3}$$

本サブルーチンでは (4.3) を逐次計算する．結果は同一領域に得るものとする．
手順は以下のようになるが，図 CFTN-3も参照すること．

a. 手順 l : $x(j_0 + k_0 \cdot 2) \Leftarrow$

$$\exp\left(-2\pi i \frac{j_0 k_0}{16}\right) \sum_{j_1=0}^{7} \exp\left(-2\pi i \frac{j_1 k_0}{8}\right) x(j_0 + j_1 \cdot 2) \tag{4.4}$$

(4.4)を実行する．
すなわち，$\sum_{j1}$ による 8 項のフーリエ変換を，まず j_0=0 に関して行う．すなわち，x_0, x_2, x_4, x_6, x_8, x_{10}, x_{12}, x_{14} のデータに対して行う．
次に，j_0=1 に関して行う．すなわち，x_1, x_3, x_5, x_7, x_9, x_{11}, x_{13}, x_{15} のデータに対して変換を行う．
この結果に対して回転因子 $\exp\left\{-2\pi i \frac{j_0 k_0}{16}\right\}$ を掛けるが j_0=0 の場合は 1 であるので実質的には不要である． 一方 j_0=1 の場合は k_0=0 の場合を除いて，次の回転因子を順に掛け合せる．すなわち，$\xi = \exp(-2\pi i/16)$ として$\xi^1, \xi^2, \xi^3, \xi^4, \xi^5, \xi^6, \xi^7$ である．以上の結果を $x(j_0 + k_0 \cdot 2)$ に格納する．

b. 手順 2 : $x(k_1 + k_0 \cdot 2) \Leftarrow$

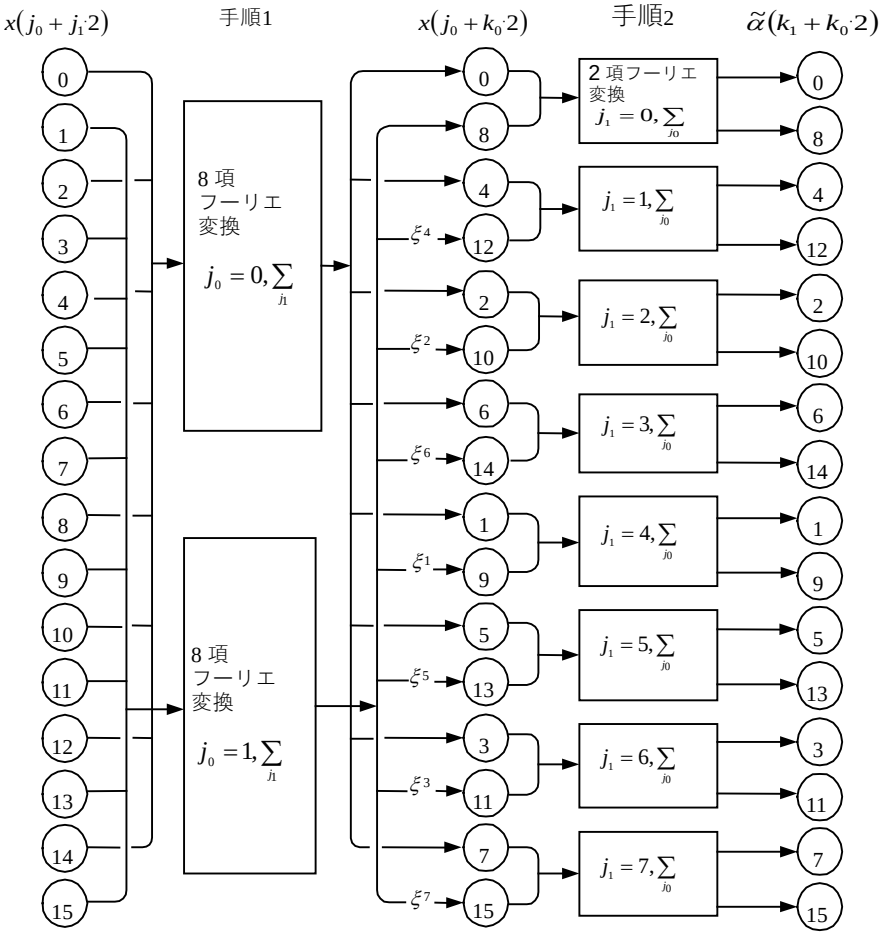
$$\sum_{j_0=0}^{1} \exp\left(-2\pi i \frac{j_0 k_1}{2}\right) x(j_0 + k_0 \cdot 2) \tag{4.5}$$

(4.5) を実行する．
すなわち，手順 1 の結果に対して $\sum_{j_0}$ による 2 項のフーリエ変換を，まず k_0=0 に関して行う．すなわち x_0, x_1 のデータに対して変換を行う．
同様に，k_0=1,…,7 に関して各々2 項のフーリエ変換を行う．この結果に対する回転因子の掛合せは不要である．
以上の結果を $x(k_1 + k_0 \cdot 2)$ に格納する．

これにより得られた$x(k_1 + k_0 \cdot 2)$ は，求める“16項の離散型複素フーリエ変換”の結果$\alpha(k_0 + k_1 \cdot 8)$ に対して，データの順位が逆順となり，$\tilde{\alpha}(k_1 + k_0 \cdot 2)$ で表される．本サブルーチンでは，8項フーリエ変換の結果はビット逆転の意味で逆順となるようにすることにより，最終的な結果 $\tilde{\alpha}(k_1 + k_0 \cdot 2)$ は $\alpha(k_0 + k_1 \cdot 8)$ に対して，ビット逆転の意味で逆順となるようにしている．

なお，詳細については，参考文献 [55], [56], 及び, [57] を参照すること．

$$\alpha(k_0 + k_1 \cdot 8) = \sum_{j_0=0}^{1} \exp\left(-2\pi i \frac{j_0 k_1}{2}\right) \exp\left(-2\pi i \frac{j_0 k_0}{16}\right) \sum_{j_1=0}^{7} x(j_0 + j_1 \cdot 2) \exp\left(-2\pi i \frac{j_1 k_0}{8}\right)$$



[注意] ○印内の数字はデータ順位を示す．また，ξ=exp(–2πi/16)，8項フーリエ変換の結果は，ビット逆転の意味で逆順となるようにしているので，$\tilde{\alpha}(k_1 + k_0 \cdot 2)$ も $\alpha(k_0 + k_1 \cdot 8)$ に対してビット逆転の意味で逆順となっている．

図 CFTN-3 16項複素フーリエ変換の流れ図 (逆順出力)

F12-15-0302 CFTR, DCFTR

離散型複素フーリエ変換 (8, 2基底FFT, 逆順入力)
CALL CFTR (A, B, NT, N, NS, ISN, ICON)

(1) 機能

1次元 (項数n) の複素時系列データ$\{x_j\}$が，ビット逆転の意味で逆順であるデータ$\{\tilde{x}_j\}$として与えられたとき，離散型複素フーリエ変換，又はその逆変換を高速変換手法 (FFT) により行う.
ただし，$n = 2^l$ (l : 0 又は正整数) であること.

① フーリエ変換
$\{\tilde{x}_j\}$を入力し，(1.1) で定義する変換を行い，$\{n\alpha_k\}$を求める.

$$n\alpha_k = \sum_{j=0}^{n-1} \tilde{x}_j \omega^{-jk}, k = 0,1,...,n-1 \quad (1.1)$$
$$,\omega = \exp(2\pi i/n)$$

② フーリエ逆変換
$\{\tilde{\alpha}_k\}$ を入力し，(1.2) で定義する変換を行い，$\{x_j\}$を求める.

$$x_j = \sum_{k=0}^{n-1} \tilde{\alpha}_k \omega^{jk}, j = 0,1,...,n-1, \quad (1.2)$$
$$\omega = \exp(2\pi i/n)$$

ここで，変換された$\{n\alpha_k\}$, $\{x_j\}$の並び順位は，正順となる.

(2) パラメタ

A …… **入力**. $\{\tilde{x}_j\}$ 又は，$\{\tilde{\alpha}_k\}$の実部.
出力. $\{n\alpha_k\}$又は，$\{x_j\}$の実部.
大きさNTの1次元配列.

B …… **入力**. $\{\tilde{x}_j\}$ 又は，$\{\tilde{\alpha}_k\}$の虚部.
出力. $\{n\alpha_k\}$又は，$\{x_j\}$の虚部.
大きさNTの1次元配列.

NT …… **入力**. 変換の対称となる$\{\tilde{x}_j\}$ または，$\{\tilde{\alpha}_k\}$が含まれるデータの総数 ($\geq$ N, NS).
通常，NT = Nとして指定する.
(使用上の注意③参照)

N …… **入力**. 変換の項数n.

NS …… **入力**. NT個のデータのうちn項の変換の対象となる$\{\tilde{x}_j\}$ 又は，$\{\tilde{\alpha}_k\}$のあい隣り合うデータの間隔 (≥ 1かつ$\leq$ NT).
通常，NS = 1として指定する.
(使用上の注意③参照)

ISN …… **入力**. 変換か逆変換かを指定する ($\neq 0$).
変換 : ISN = +l
逆変換 : ISN = –l
(使用上の注意④参照)

ICON …… **出力**. コンディションコード.
表 CFTR-1参照.

表 CFTR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	ISN = 0, NS < 1, NT < N, NT < NS又は, N≠2^l (l : 0 又は正整数) であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ...ATAN, ALOG, SQRT, SIN

b. 注意
① 一般的なフーリエ変換の定義について
離散型複素フーリエ変換及び，逆変換は一般的に (3.1), (3.2) で定義される.

$$\alpha_k = \frac{1}{n}\sum_{j=0}^{n-1} x_j \omega^{-jk}, \quad k = 0,1,...,n-1 \quad (3.1)$$

$$x_j = \sum_{k=0}^{n-1} \alpha_k \omega^{jk}, \quad j = 0,1,...,n-1 \quad (3.2)$$
$$\text{ここで}, \omega = \exp(2\pi i/n)$$

本サブルーチンでは，(3.1), (3.2) の右辺の$\{\tilde{x}_j\}$又は$\{\alpha_k\}$に対応して，データの順位がビット逆転の意味で逆順である$\{\tilde{x}_j\}$ 又は$\{\tilde{\alpha}_k\}$ に対する変換を行う．変換された結果は，(3.1), (3.2) の左辺$\{\alpha_k\}$又は$\{x_j\}$に対応して，$\{n\alpha_k\}$又は$\{x_j\}$が得られる．すなわち，データの順位は正順となり，かつ$\{\alpha_k\}$の要素はn倍された値となる.
結果の正規化は必要に応じて行うこと.

② 本ルーチンの用途について
通常，フーリエ変換又は，フーリエ逆変換を行う場合はサブルーチンCFTを用いればよいが，変換･逆変換を連続処理するような場合は，サブルーチンCFTN，及び本サブルーチンを併用することにより，効率よく行うことができる.
(サブルーチンCFTN「**使用上の注意**」参照)

③ 多次元変換への応用について
本サブルーチンは，多次元変換への応用が可能である．例えば，2次元変換は，(3.5), (3.6) でフーリエ変換及び逆変換が定義される.

$$\alpha_{K1,K2}=\frac{1}{N1\cdot N2}\sum_{J1=0}^{N1-1}\sum_{j2=0}^{N2-1}x_{j1,j2}\,\omega_1^{-J1\cdot K1}\,\omega_2^{-J2\cdot K2} \qquad (3.5)$$

$$,K1=0,1,...,N1-1,K2=0,1,...,N2-1$$

$$x_{J1,J2}=\sum_{K1=0}^{N1-1}\sum_{K2=0}^{N2-1}\alpha_{K1,K2}\,\omega_1^{J1\cdot K1}\,\omega_2^{J2\cdot K2} \qquad (3.6)$$

$$,J1=0,1,...,N1-1,J2=0,1,...,N2-1$$

ここで，$\omega_1=\exp(2\pi i/N1),\omega_2=\exp(2\pi i/N2)$

(3.5) を変形すると (3.7) となる．

$$\alpha_{K1,K2}=\frac{1}{N1\cdot N2}\sum_{J1=0}^{N1-1}\omega_1^{-J1\cdot K1}\sum_{J2=0}^{N2-1}x_{J1,J2}\,\omega_2^{-J2\cdot K2} \qquad (3.7)$$

2 次元変換 (3.7) は，まず$\sum_{j2}$ により $N2$ 項の 1 次元変換を $J1$ に関して $N1$ 組行い，その結果に対して$\sum_{j1}$ により $N1$ 項の 1 次元変換を $J2$ に関して $N2$ 組行うことにより達成されることが分る．

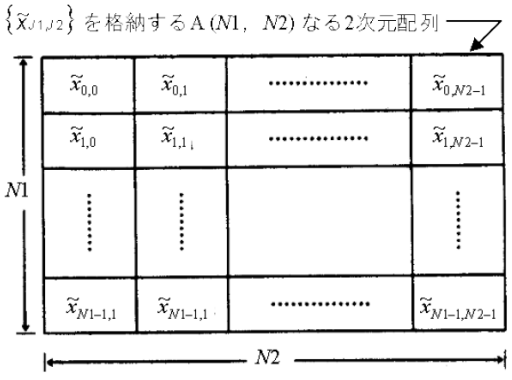
本サブルーチンでは，1 回の呼出しで$\sum_{j2}$ による $N2$ 項の 1 次元変換を $N1$ 組まとめて行うことが可能である．これにより，$N2$ 項の 1 次元変換を個別に行う場合よりも計算の手間が少なく，効率よく多次元変換を達成できる．

具体的には，以下のように指定し本サブルーチンを呼び出せばよい．

(a) $\{\tilde{x}_{J1,J2}\}$ を，2 次元配列 A 及び B に，図 CFTR-1 で示されるように格納する．．

(b) 次のように，本ルーチンを 2 回呼び出す．

```
      …
      NT=N1*N2
      NS=1
      CALL CFTR(A,B,NT,N1,NS,1,ICON)
      NS=N1
      CALL CFTR(A,B,NT,N2,NS,1,ICON)
      …
```



［注意］配列Aには，$\{\tilde{x}\}$ の実部を格納する．虚部は，B ($N1$, $N2$) なる2次元配列に，同様に格納する．

図 CFTR-1 $\{\tilde{x}_{J1,J2}\}$ の格納方法

ただし，これによって得られる結果は $\{N1\cdot N2\alpha_{K1,K2}\}$であるので，1 次元変換の場合と同様に，正規化は必要に応じて行うこと．

(3.6) で定義される逆変換の場合も，同様に処理が可能である．

また 2 次元以上も可能であり，3 次元の場合については使用例②を参照すること．

④ ISN の与え方について

ISNでは，変換か逆変換かを指定するが，更に次のように使いわけることができる．すなわちNT個の $\{\tilde{x}_j\}$ 又は，$\{\tilde{\alpha}_k\}$ の実部，虚部が，各々 $NT\cdot I$ なる大きさの領域に間隔I で格納されている場合は，次のように指定する．

変換　：ISN = $+I$

逆変換：ISN = $-I$

この場合，変換結果も間隔 I で格納される．

c. 使用例

① 1 次元変換の場合

n項の複素時系列データ$\{x_j\}$を入力し、サブルーチンPNRにより，データの順位をビット逆転の意味で逆順に並べ換える．その結果の$\{\tilde{x}_j\}$ に対して本サブルーチンによりフーリエ変換し$\{n\alpha_k\}$を求める．

$n \le 1024$ (= 2^{10}の場合)

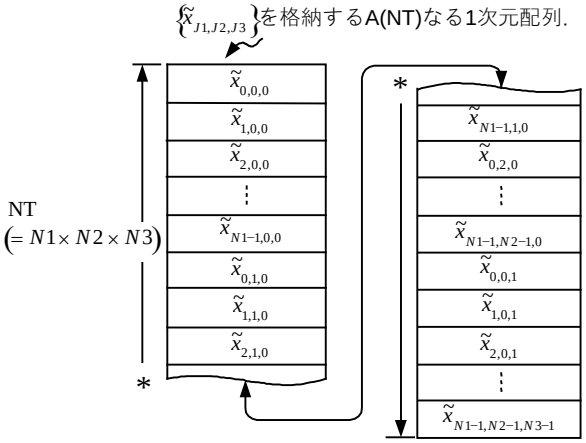
```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
      CALL PNR(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      CALL CFTR(A,B,N,N,1,1,ICON)
      WRITE(6,620) (I,A(I),B(I),I=1 ,N)
      STOP
500   FORMAT(I5/(2E20.7))
600   FORMAT('0',10X,'INPUT DATA N=',I5/
     *      /(15X,I5 ,2E20.7))
610   FORMAT('0',10X,'RESULT ICON=',I5)
620   FORMAT('0',10X,'OUTPUT DATA'//
     *      (15X,I5 ,2E20.7))
      END
```

② 3 次元変換の場合

$N1$, $N2$, $N3$項の3次元複素時系列データ$\{x_{J1,J2,J3}\}$を入力し，サブルーチンPNR荷より，データの順位をビット逆転の意味で逆順に並べ換える．

その結果の $\{\tilde{x}_{J1,J2,J3}\}$ に対して本サブルーチンによりフーリエ変換し$\{N1\cdot N2\cdot N3\alpha_{K1,K2,K3}\}$を求める．

なおデータ$\{x_{J1,J2,J3}\}$は，図 CFTR-2で示されるように，1次元配列上に格納してもよい．



［注意］配列Aには, $\{\tilde{x}\}$ の実部を格納する． 虚部はB (NT) なる1次元配列に，同様に格納する．このように格納した場合の NSの与え方は，3回の呼出し時に順に，1, N1, N1*N2とすればよい.

図 CFTR-2 $\{\tilde{x}_{J1,J2,J3}\}$ の格納方法

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024),N(3)
      READ(5,500) N
      NT=N(1)*N(2)*N(3)
      READ(5,510) (A(I),B(I),I=1,NT)
      WRITE(6,600) N,(I,A(I),B(I),I=1,
     *              NT)
      NS=1
      DO 10 I=1,3
      CALL PNR(A,B,NT,N(I),NS,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      CALL CFTR(A,B,NT,N(I),NS,1,ICON)
      NS=NS*N(I)
   10 CONTINUE
      WRITE(6,620) (I,A(I),B(I),I=1,NT)
      STOP
  500 FORMAT(3I5)
  510 FORMAT(2E20.7)
  600 FORMAT('0',10X,'INPUT DATA N=',
     * 3I5/ /(15X,I5 ,2E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5)
  620 FORMAT('0',10X,'OUTPUT DATA'//
     * (15X,I5 ,2E20.7))
      END
```

(4) 手法概要

離散型複素フーリエ変換を 8及び 2を基底とする高速変換手法 (高速フーリエ変換･FFT) により行う. ただし入力データはビット逆転の意味で逆順に与えられるものとする.

高速複素フーリエ変換の原理については，サブルーチンCFTの項を参照すること.

ここでは，n = 16なる場合の具体例について説明する．まず，この場合フーリエ変換は (4,1) で表される.

$$\alpha_k=\sum_{j=0}^{15}x_j\omega^{-jk},\ k=0,1,...,15,\qquad \omega=\exp(2\pi i/16) \tag{4.1}$$

ただし，(4.1) では正規化定数1/16は省略している.

本ルーチンでは，nを2 と8 の因数に分解する (n = 2×8)．さて，k, j は (4.2) のように表すことができる.

$$k=k_0\cdot 2+k_1, 0\le k_0\le 7, 0\le k_1\le 1$$
$$j=j_0\cdot 8+j_1, 0\le j_0\le 1, 0\le j_1\le 7 \tag{4.2}$$

(4.2) を (4.1) に代入し，共通項を整理すると (4.3) となる．ただし，$\alpha(k_0\cdot 2+k_1)\equiv\alpha_{k_0\cdot 2+k_1}$ と表示する.

$$\alpha(k_0\cdot 2+k_1)=\sum_{j_0=0}^{7}\exp\left\{-2\pi j\frac{j_1k_0}{8}\right\}\cdot\exp\left\{-2\pi i\frac{j_1k_1}{16}\right\}\cdot\sum_{j_0=0}^{1}\exp\left\{-2\pi i\frac{j_0k_1}{2}\right\}x(j_0\cdot 8+j_1) \tag{4.3}$$

本サブルーチンでは，入力データがビット逆転の意味で逆順に与えられる．すなわち$x(j_0\cdot 8+j_1)$ の代りに $\tilde{x}(j_0+j_1\cdot 2)$ で与えられることを考慮して (4.3) を逐次計算する．結果は同一領域に得るものとする.

手順は以下のようになるが，図 CFTR-3も参照すること.

a. 手順 1 : $\tilde{x}(k_1+j_1\cdot 2)\Leftarrow$

$$\sum_{j_0=0}^{1}\exp\left\{-2\pi i\frac{j_0k_1}{2}\right\}\tilde{x}(j_0+j_1\cdot 2) \tag{4.4}$$

(4.4) を実行する.
すなわち, $\sum_{j0}$ による2項のフーリエ変換を，まず j_1=0に関して行う．具体的には $\tilde{x}_0, \tilde{x}_1$ に対して変換を行う．同様に，j_1=1,～, 7に関して2項のフーリエ変換を行い，$\tilde{x}(k_1+j_1\cdot 2)$のように格納する.

b. 手順 2 : $\tilde{x}(k_1+k_0\cdot 2)\Leftarrow$

$$\sum_{j_1=0}^{7}\exp\left\{-2\pi i\frac{j_1k_0}{8}\right\}\exp\left\{-2\pi i\frac{j_1k_1}{16}\right\}\cdot\tilde{x}(k_1+j_1\cdot 2) \tag{4.5}$$

(4.5) を実行する.

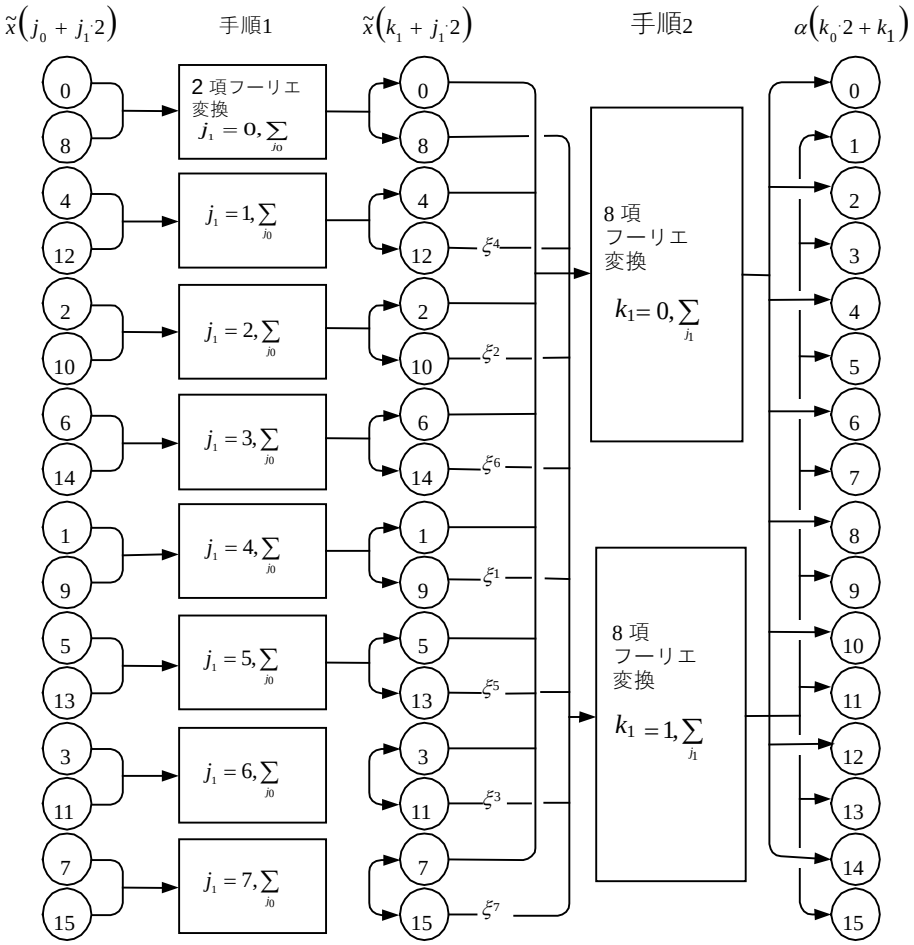
すなわち，手順1の結果に対して回転因子 $\exp\left\{-2\pi i\frac{j_1k_1}{16}\right\}$ を掛け $\sum_{j_1}$ による8項のフーリエ変換を行う．まず，k_1=0の場合，回転因子は1であるので掛け合せは不要であり，$\tilde{x}_0,\tilde{x}_2,\tilde{x}_4,\tilde{x}_6,\tilde{x}_8,\tilde{x}_{10},\tilde{x}_{12},\tilde{x}_{14}$ に対して変換を行う．K_1=1の場合は，$\tilde{x}_1,\tilde{x}_3,\tilde{x}_5,\tilde{x}_7,\tilde{x}_9,\tilde{x}_{11},\tilde{x}_{13},\tilde{x}_{15}$ に対して回転因子を掛けた後，変換を行う．回転因子は，$\xi=\exp(-2\pi i/16)$ としたとき $\xi^0,\xi^1,\xi^2,\xi^3,\xi^4,\xi^5,\xi^6,\xi^7$ である．

この2組の8項のフーリエ変換では，各々の入力データのならびはビット反転の意味で逆順となっているため，各々の変換結果は正順にし，$\tilde{x}(k_1+k_0\cdot 2)$ のように格納する．

これによって得られた $\tilde{x}(k_1+k_0\cdot 2)$ は，求める“16項の離散型複素フーリエ変換”の結果 $\alpha(k_0\cdot 2+k_1)$ に一致する．

なお，詳細については，参考文献 [55], [56] 及び [57] を参照すること．

$$\alpha(k_0\cdot 2+k_1)=\sum_{j_1=0}^{7}\exp\left\{-2\pi i\frac{j_1k_0}{8}\right\}\exp\left\{-2\pi i\frac{j_1k_1}{16}\right\}\sum_{j_0=0}^{1}\exp\left\{-2\pi i\frac{j_0k_1}{8}\right\}\tilde{x}(j_0+j_1\cdot 2)$$



［注意］○印内の数字はデータ順位を示す．また $\xi=\exp\{-2\pi i/16\}$.

図 CFTR-3 16項複素フーリエ変換の流れ図 (逆順入力)

A11-40-0101 CGSBM, DCGSBM

行列格納モードの変換 (一般モード→対称バンド行列用圧縮モード)
CALL CGSBM (AG, K, N, ASB, NH, ICON)

(1) 機能

一般モードで格納された $n \times n$，バンド幅hの実対称バンド行列を，対称バンド行列用圧縮モードに変換する.

$n > h \geq 0$であること.

(2) パラメタ

AG……**入力**. 一般モードで格納された対称バンド行列. AG (K, N) なる2次元配列.
(使用上の注意①参照)

K……**入力**. 配列AGの整合寸法 (≥ N)

N……**入力**. 行列の次数n.

ASB……**出力**. 対称バンド行列用圧縮モードで格納された対称バンド行列.
大きさ$n(h+1)–h(h+1)/2$の1次元配列.

NH……**入力**. 行列のバンド幅 h.

ICON……**出力**. コンディションコード.
表 CGSBM-1参照.

表 CGSBM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	NH < 0, N ≤ NH 又は K<Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... MAX0

b. 注意

① 一般モードの対称行列の格納方法
配列AGには，下バンド部分及び対角部分だけを与えればよい．上バンド部分は，サブルーチン内で下バンド部分が複写される.

② 領域の節約について
配列AG上の内容を保存する必要のない場合は，EQUIVALENCE文によりEQUIVALENCE (AG (1,1), ASB (1)) と宣言すれば領域が節約できる．使用例参照.

c. 使用例
$n \times n$，バンド幅 hの正値対称バンド行列が一般モードで与えられたとき，これを対称バンド行列用圧縮モードに移してLDL^T分解する．サブルーチンSBDLを用いて分解を行い，必要なモード変換を本サブルーチン並びにサブルーチンCSBGMで行う．$n \leq 100$, $h \leq 20$の場合.

```
C     **EXAMPLE**
      DIMENSION AG(100,100),ASB(1890)
      EQUIVALENCE(AG(1,1),ASB(1))
   10 READ(5,500) N,NH
      IF(N.EQ.0)STOP
      K=100
      READ(5,510) ((AG(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,NH,((I,J,AG(I,J),
     *             I=1,N),J=1,N)
      CALL CGSBM(AG,K,N,ASB,NH,ICON)
      WRITE(6,610) ICON
      IF(ICON.EQ.30000) GO TO 10
      EPSZ=0.0
      CALL SBDL(ASB,N,NH,EPSZ,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL CSBGM(ASB,N,NH,AG,K,ICON)
      WRITE(6,630) N,NH,((I,J,AG(I,J),
     *             J=1,N),I=1,N)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT(//10X,'** INPUT MATRIX **'/
     *10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     *(2X,4('(',I3,',',I3,')',E17.8)))
  610 FORMAT('1'/10X,'CGSBM ICON=',I5)
  620 FORMAT(/10X,'SBDL ICON=',I5)
  630 FORMAT('1'//10X,'DECOMPOSED MATRIX'/
     *10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     *(2X,4('(',I3,',',I3,')',E17.8)))
      END
```

(4) 手法概要

2次元配列AG上に一般モードで格納された実対称バンド行列を，次に示す手順により，1次元配列上に対称バンド行列用圧縮モードで格納する.

a. 対角部分を対称軸として，下バンド部分を上バンド部分に転送する.

$$AG(I,J) \rightarrow AG(J,I),\ I - h \leq J \leq I - 1$$

b. AGの対角要素及び上バンド要素AG (I,J) を，ASB上に転送する.
転送はAGの1列目から列ごとに行う．対応関係は次のとおりである.

一般モード上の要素	行列要素	対称バンド行列用圧縮モード上の要素

$$AG(1, J) \longrightarrow a_{ij} \longrightarrow ASB\ (J\,(J-1)/2+I)$$
$$, I = 1, 2, \cdots, J\ ,\ \ J = 1, 2, \cdots, h+1$$
$$AG(1, J) \longrightarrow a_{ij} \longrightarrow ASB\ (hJ - h\,(h+1)/2+I)$$
$$, I = J-h, J-h+1, \cdots, J\ ,\ J = h+2, h+3, \cdots, N$$

A11-10-0101 CGSM, DCGSM

行列格納モードの変換 (一般モード→対称行列用圧縮モード)
CALL CGSM (AG, K, N, AS, ICON)

(1) 機能

一般モードで格納された $n \times n$の実対称行列を，対称行列用圧縮モードに変換する．$n \geq 1$であること．

(2) パラメタ

AG ……**入力**．一般モードで格納された対称行列．
AG (K, N) なる2次元配列．
(使用上の注意①参照)

K ……**入力**．配列AGの整合寸法 (≥ N).

N ……**入力**．行列の次数 n.

AS ……**出力**．対称行列用圧縮モードで格納された対称行列．大きさn (n +1)/2の1次元配列.

ICON ……**出力**．コンディションコード．
表 CGSM-1参照.

表 CGSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N < 1 又は K < Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... なし.

b. 注意

① 一般モードの対称行列の格納方法
配列AGには，下三角部分及び対角部分だけを与えればよい．上三角部分は，サブルーチン内において下三角部分が複写される.

② 領域の節約について
配列AG上の内容を保存する必要のない場合は，EQUIVALENCE文により

EQUIVALENCE (AG(1,1), AS(1))

と宣言すれば領域が節約できる．使用例参照.

c. 使用例
$n \times n$の正値対称行列が一般モードで与えられたとき，その逆行列を求める.
サブルーチンSLDL, LDIVを用いて求め，必要なモード変換を，本サブルーチン並びにサブルーチンCSGMで行う．$n \leq 100$の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),B(5050)
      EQUIVALENCE(A(1,1),B(1))
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      K=100
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,A(I,J),J=1,N),
     *             I=1,N)
      CALL CGSM(A,K,N,B,ICON)
      WRITE(6,610) ICON
      IF(ICON.EQ.30000) GOTO 10
      EPSZ=0.0
      CALL SLDL(B,N,EPSZ,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GOTO 10
      CALL LDIV(B,N,ICON)
      WRITE(6,630) ICON
      CALL CSGM(B,N,A,K,ICON)
      WRITE(6,640) ICON
      WRITE(6,650)N,((I,J,A(I,J),J=1,N),
     *             I=1,N)
      GOTO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'//
     *10X,'** INPUT MATRIX **'/
     *10X,'ORDER=',I5/(2X,
     *4('(',I3,',',I3,')',E17.8)))
  610 FORMAT(10X,'CGSM ICON=',I5)
  620 FORMAT(10X,'SLDL ICON=',I5)
  630 FORMAT(10X,'LDIV ICON=',I5)
  640 FORMAT(10X,'CSGM ICON=',I5)
  650 FORMAT('1'//10X,'** INVERSE ',
     *'MATRIX **'/10X,'ORDER=',I5/(2X,
     *4('(',I3,',',I3,')',E17.8)))
      END
```

(4) 手法概要

2次元配列AG上に一般モードで格納された実対称行列を，次に示す手順により，1次元配列上に対称行列用圧縮モードで格納する.

a. 対角部分を対称軸として，下三角部分を上三角部分に転送する.

$$AG(I,J) \rightarrow AG(J,I),\ J<I$$

b. AGの対角要素及び上三角要素AG (I,J) を，ASのJ (J-1)/2 + I番目に転送する．ここでJ ≥ Iである．転送は，AGの1列目から列ごとに行う. NT = n (n + 1)/2としたとき以下に対応関係を示す.

一般モード上の要素		行列の要素		圧縮モード上の要素
AG (1, 1)	→	a_{11}	→	AS (1)
AG (1, 2)	→	a_{21}	→	AS (2)
AG (2, 2)	→	a_{22}	→	AS (3)
…		…		…
AG (I, J)	→	a_{ji}	→	AS (J (J – 1)/2+I)
…		…		…
AG (N– 1, N)	→	a_{nn-1}	→	AS (NT – 1)
AG (N, N)	→	a_{nn}	→	AS (NT)

B21-15-0602 CHBK2, DCHBK2

複素行列の固有ベクトルへの逆変換
CALL CHBK2 (ZEV, K, N, IND, M, ZP, IP, DV, ICON)

(1) 機能

*n*次の複素ヘッセンベルグ行列***H***の*m*個の固有ベクトルを，複素行列***A***の固有ベクトルへ逆変換する.ただし，***H***は***A***に安定化基本相似変換を適用して得られたものとする．複素行列***A***の固有ベクトルの正規化は行わない.

$n \geq 1$であること.

(2) パラメタ

ZEV …… **入力**．複素ヘッセンベルグ行列***H***の*m*個の固有ベクトル.
出力．複素行列***A***のm_l個の固有ベクトル.
ここで，m_lは，INDの要素のうち，その値が1であるものの個数である.
ZEV (K,M) なる複素数型2次元配列.

K …… **入力**．配列ZEV, ZPの整合寸法 ($\geq n$).

N …… **入力**．行列***A***と***H***の次数*n*.

IND …… **入力**．固有ベクトル逆変換要不要の指定.
IND (J) = 1なら，J番目の固有値に対応する固有ベクトルの逆変換を行う.
IND (J) = 0なら，行わない.
大きさMの1次元配列.
(使用上の注意①参照).

M …… **入力**．複素ヘッセンベルグ行列の全固有ベクトルに対応する固有値の総数*m*.

ZP …… **入力**．***A***から***H***への変換行列のための情報.
ZP (K,N) なる複素数型2次元配列.
(使用上の注意②参照)

IP …… **入力**．***A***から***H***への変換行列のための情報.
大きさ*n*の1次元配列.
(使用上の注意②参照)

DV …… **入力**．複素行列***A***の平衡化に使われたスケーリングファクタ.
大きさ*n*の1次元配列.
平衡化が行われなかったとき，DVは1次配列である必要はなく，このとき，DV=0.0と指示できる.
(使用上の注意③参照)

ICON …… **出力**．コンディションコード.
表 CHBK2-l参照.

表 CHBK2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1であった.	ZEV(1,1)=(1.0,0.0) とする.
30000	N<M, M<1 又は K<Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... なし.

b. 注意

① サブルーチンCHVEC呼出し後のパラメタZEV, IND, Mは，本サブルーチンでそのまま入力できるようになっている.

② サブルーチンCHES2のパラメタZAとIPは，本サブルーチンのパラメタZPとIPに対応しているので，そのまま使用できる．ZPとIPの内容については，CHES2参照のこと.

③ 平衡化によるスケーリングファクタDVの内容については，サブルーチンCBLNC参照のこと.

c. 使用例

*n*次の複素行列の固有値と固有ベクトルを，以下のサブルーチンを用いて計算する．ただし,固有ベクトルは，固有値の求められた順に計算する.

CBLNC ..... 複素行列の平衡化.
CHES2 ..... 複素ヘッセンベルグ行列への変換.
CHSQR ..... 複素ヘッセンベルグ行列の固有値.
CHVEC ..... 複素ヘッセンベルグ行列の固有ベクトル.
CHBK2 ..... 複素ヘッセンベルグ行列の固有ベクトルを複素行列の固有ベクトルへ逆変換する.
CNRML ..... 複素行列の固有ベクトルの正規化.

$n \leq 100$の場合.

```
C     **EXAMPLE**
      COMPLEX ZA(100,100),ZE(100),
     *ZAW(100,101),ZEV(100,100)
      DIMENSION IND(100),DV(100),IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,ZA(I,J),J=1,N)
      CALL CBLNC(ZA,100,N,DV,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
      CALL CHES2(ZA,100,N,IP,ICON)
```

```
      DO 30 J=1,N
      IM=MIN0(J+1,N)
      DO 30 I=1,IM
   30 ZAW(I,J)=ZA(I,J)
      CALL CHSQR(ZAW,100,N,ZE,M,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      DO 40 I=1,M
   40 IND(I)=1
      CALL CHVEC(ZA,100,N,ZE,IND,M,
     *           ZEV,ZAW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL CHBK2(ZEV,100,N,IND,M,ZA,IP,
     *           DV,ICON)
      CALL CNRML(ZEV,100,N,M,2,ICON)
      CALL CEPRT(ZE,ZEV,100,N,IND,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1',10X,'ORIGINAL MATRIX'
     *//11X,'ORDER=',I5)
  610 FORMAT(/2(5X,'A(',I3,',',I3,')=',
     *2E15.7))
  620 FORMAT(/11X,'CONDITION CODE=',I5/)
      END
```

本使用例中のサブルーチンCEPRTは，複素行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチンCEIG2の使用例参照のこと．

(4) 手法概要

n次の複素ヘッセンベルグ行列$\boldsymbol{H}$のm個の固有ベクトルを，平衡化された複素行列 $\tilde{\boldsymbol{A}}$ の固有ベクトルへ逆変換し，更に平衡化を行う前の複素行列$\boldsymbol{A}$の固有ベクトルへ逆変換する．

複素行列$\boldsymbol{A}$の平衡化は，(4.1) に示す対角相似変換により行い，平衡化された複素行列 $\tilde{\boldsymbol{A}}$ の複素ヘッセンベルグ行列$\boldsymbol{H}$への変換は，(4.2) に示すn–2回の安定化基本相似変換により行う．

$$\tilde{\boldsymbol{A}} = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D} \tag{4.1}$$

$$\boldsymbol{H} = \boldsymbol{S}_{n-2}^{-1}...\boldsymbol{S}_2^{-1}\boldsymbol{S}_1^{-1}\tilde{\boldsymbol{A}}\boldsymbol{S}_1\boldsymbol{S}_2...\boldsymbol{S}_{n-2} \tag{4.2}$$

ただし，$\boldsymbol{D}$は対角行列であり，$\boldsymbol{S}_i$は置換行列$\boldsymbol{P}_i$と消去行列$\boldsymbol{N}_i$とにより，

$$\boldsymbol{S}_i = \boldsymbol{P}_i\boldsymbol{N}_i^{-1} \quad i = 1,2,...,n-2 \tag{4.3}$$

で表される．

$\boldsymbol{H}$の固有値をλ，固有値λに対応する固有ベクトルを$\boldsymbol{y}$とすると，

$$\boldsymbol{H}\boldsymbol{y} = \lambda\boldsymbol{y} \tag{4.4}$$

が成り立つ．(4.1) と (4.2) の関係から，(4.4) は，

$$\boldsymbol{S}_{n-2}^{-1}...\boldsymbol{S}_2^{-1}\boldsymbol{S}_1^{-1}\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}\boldsymbol{S}_1\boldsymbol{S}_2...\boldsymbol{S}_{n-2}\boldsymbol{y} = \lambda\boldsymbol{y} \tag{4.5}$$

となり，(4.5) の両辺に$\boldsymbol{D}\boldsymbol{S}_1\boldsymbol{S}_2\cdots\boldsymbol{S}_{n-2}$を左側より乗ずると (4.6) となる．

$$\boldsymbol{A}\boldsymbol{D}\boldsymbol{S}_1\boldsymbol{S}_2...\boldsymbol{S}_{n-2}\boldsymbol{y} = \lambda\,\boldsymbol{D}\boldsymbol{S}_1\boldsymbol{S}_2...\boldsymbol{S}_{n-2}\boldsymbol{y} \tag{4.6}$$

したがって，$\boldsymbol{A}$の固有ベクトル$\boldsymbol{x}$は，(4.7) となる．

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{S}_1\boldsymbol{S}_2...\boldsymbol{S}_{n-2}\boldsymbol{y} \tag{4.7}$$

(4.7) の計算は，(4.8), (4.9) により行う．ただし，

$$\boldsymbol{y}=\boldsymbol{x}_{n-1}$$

とする．

$$\boldsymbol{x}_i = \boldsymbol{S}_i\boldsymbol{x}_{i+1} = \boldsymbol{P}_i\boldsymbol{N}_i^{-1}\boldsymbol{x}_{i+1}, \quad i = n-2,...,2,1 \tag{4.8}$$

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{x}_1 \tag{4.9}$$

平衡化及び安定化基本相似変換の詳細は，サブルーチンCBLNC, CHES2参照のこと．

なお，詳細については，参考文献 [13] の pp.339 – 358を参照すること．

B21-15-0302 CHES2, DCHES2

複素行列の複素ヘッセンベルグ行列への変換 (安定化基本相似変換)
CALL CHES2 (ZA, K, N, IP, ICON)

(1) 機能

*n*次の複素行列***A***を，安定化基本相似変換 (部分ピボッティングを伴うガウスの消去法) により，複素ヘッセンベルグ行列***H***へ変換する．

$$\boldsymbol{H} = \boldsymbol{S}^{-1}\boldsymbol{A}\boldsymbol{S}$$

ただし***S***は変換行列である．$n \geq 1$であること．

(2) パラメタ

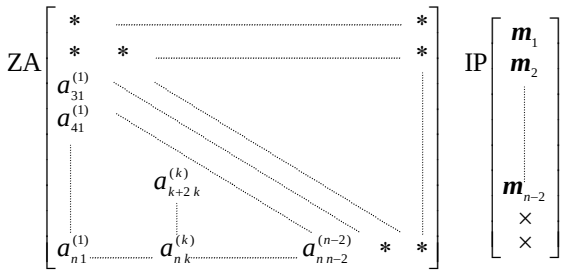
ZA …… **入力**．複素行列***A***.
出力．複素ヘッセンベルグ行列***H***及び変換行列***S***．(図 CHES2-1参照).
ZA (K, N) なる複素数塑2次元配列.

K …… **入力**．配列ZAの整合寸法 ($\geq n$).

N …… **入力**．行列***A***の次数*n*.

IP …… **出力**．変換行列***S***のための情報 (図 CHES2–1参照)．大きさ*n*の1次元配列.

ICON …… **出力**．コンディションコード.
表 CHES2-1参照.



［注意］*部分は，ヘッセンベルグ行列であり，他は変換行列のための情報である．×は作業領域.

図 CHES2-1 変換後の配列ZAとIP

表 CHES2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N =1 又は N = 2であった.	変換しない.
30000	K < N 又は N < 1であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... CSUM, AMACH, MGSSL

② FQRTRAN 基本関数 ... REAL, AIMAG, ABS, AMAX1

b. 注意

① 出力された配列ZA及びIPは，行列***A***の固有ベクトルを求めるために必要となる.

② 固有値の精度は，複素ヘッセンベルグ行列に変換した時点である程度決定される．そのため，できるだけ精度よく変換を行うことが必要であり，本サブルーチンではそのための考慮が払われている．しかし，固有値に非常に大きいものと小さいものがあるとき，小さい方の固有値は変換の影響を受けやすく，したがって，小さい固有値を精度よく求めることが困難な場合がある.

c. 使用例

*n*次の複素行列を複素ヘッセンベルグ行列に変換した後，サブルーチンCHSQRにより固有値を計算する．$n \leq 100$の場合.

```
C     **EXAMPLE**
      COMPLEX ZA(100,100),ZE(100)
      DIMENSION IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,ZA(1,J),J=1,N)
      CALL CHES2(ZA,100,N,IP,ICON)
      WRITE(6,620)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,610) ((I,J,ZA(I,J),
     *              J=1,N),I=1,N)
      CALL CHSQR(ZA,100,N,ZE,M,ICON)
      WRITE(6,640)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,650) (I,ZE(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1',10X,'ORIGINAL MATRIX'
     * //11X,'ORDER=',I5/)
  610 FORMAT(/2(5X,'A(',I3,',',I3,
     * ')=',2E15.7))
  620 FORMAT('0'//11X,
     * 'HESSENBERG MATRIX')
  630 FORMAT(/11X,'CONDITION CODE=',I5/)
  640 FORMAT('0'/11X,'EIGENVALUES')
  650 FORMAT(5X,'E(',I3,')=',2E15.7)
      END
```

(4) 手法概要

*n*次の複素行列***A***の複素ヘッセンベルグ行列***H***への変換は，次式で示されるような$n-2$回の安定化基本相似変換により行われる.

$$\boldsymbol{A}_k = \boldsymbol{S}_k^{-1}\boldsymbol{A}_{k-1}\boldsymbol{S}_k, \quad k = 1,2,\ldots,n-2 \qquad (4.1)$$

ただし，$\boldsymbol{A}_0$=***A***である.

変換の終了した時点で$\boldsymbol{A}_{n-2}$は，複素ヘッセンベルグ行列となる.

第k回目の変換は，次の手順で行う．ただし，$\boldsymbol{A}_{k-1}=\left(a_{ij}^{(k-1)}\right)$とする．

① $\boldsymbol{A}_{k-1}$の第k列の第k+1行以降の要素$a_{ik}^{(k-1)}$, $i = k + 1, ..., n$ の中でノルム最大の要素を見出す．ただし，ここでは $z = x + iy$のノルムとして$\|z\|_1=|x|+|y|$を考えるものとする．ノルム最大の要素が$a_{m_k k}^{(k-1)}$であったとする．番号m_kをIPの第k要素として格納しておく．

② $m_k = k$+1ならば直ちに③へ行く．$m_k > k$+1ならば，$\boldsymbol{A}_{k-1}$の第$k + 1$行と第m_k行の第k列以降の部分を入れかえる．

③ $a_{k+1,k}^{(k+1)}$をピボットとして，第k+2行以降の第k列要素を消去する．実際には，

$$a_{ik}^{(k)}=-a_{ik}^{(k-1)}\Big/a_{k+1,k}^{(k-1)},\quad i=k+2,...,n \tag{4.2}$$

及び，

$$\tilde{a}_{ij}^{(k)}=a_{ij}^{(k-1)}+a_{ik}^{(k)}a_{k+1,j}^{(k-1)},\quad \begin{matrix} i=k+2,...,n \\ j=k+1,...,n \end{matrix} \tag{4.3}$$

という計算を行う．

④ m_k=k+1ならば直ちに⑤へ行く．$m_k > k$+1ならば，第k +1列全体と第m_k列全体とを入れかえる．

⑤ この時点での第k+1列以降の要素を$\tilde{a}_{ij}^{(k)}$とする．第k +1列を次のように変形する．

$$a_{ik+1}^{(k)}=\tilde{a}_{ik+1}^{(k)}-\sum_{j=k+2}^{n}\tilde{a}_{ij}^{(k)}a_{ik}^{(k)},\quad i=1,2,...,n \tag{4.4}$$

以上で第k回目の変換が終わる．②の段階での行の入れかえは，図 CHES2-2の置換行列$\boldsymbol{P}_k$を$\boldsymbol{A}_{k-1}$の左から乗ずることによって実現される．また③の消去の計算は図 CHES2-3の消去行列$\boldsymbol{N}_k$を左から乗ずることによって実現される．したがって，

$$\boldsymbol{S}_k^{-1}=\boldsymbol{N}_k\boldsymbol{P}_k,\quad \boldsymbol{S}_k=\boldsymbol{P}_k\boldsymbol{N}_k^{-1} \tag{4.5}$$

となる．ただし，$\boldsymbol{P}_k^{-1}=\boldsymbol{P}_k$であり，また$\boldsymbol{N}_k^{-1}$は$\boldsymbol{N}_k$の非対角要素の符号を反転したもので与えられる．

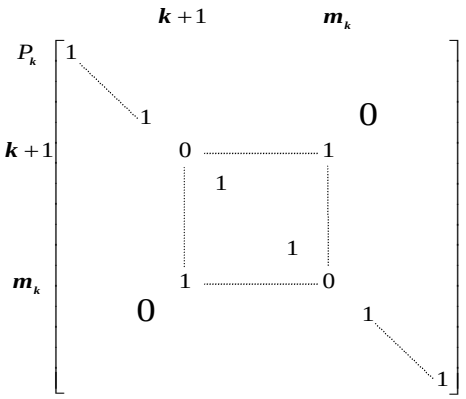
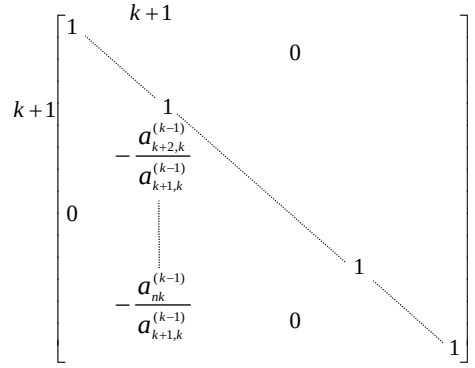


図 CHES2-2 置換行列$\boldsymbol{P}_k$



［注意］$a_{ij}^{(k-1)}$は，変換行列 P_k を掛け合わせた後の要素である．

図 CHES2-3 消去行列N_k

$\boldsymbol{P}_k$のための情報 m_kはIPの第k要素として，そして$\boldsymbol{N}_k$のk +1列のk +2行以降の要素は，そのまま$a_{ij}^{(k)}$として$\boldsymbol{A}$のk列のk +2行以下に保存される．

$n = 2$又は$n = 1$のときは変換は行わない．

なお，詳細については，参考文献 [13] の pp.339 – 358を参照すること．

B21-15-0402 CHSQR, DCHSQR

複素ヘッセンベルグ行列の固有値 (QR法)
CALL CHSQR (ZA, K, N, ZE, M, ICON)

(1) 機能

*n*次の複素ヘッセンベルグ行列***A***の固有値を，QR法により求める．ただし$n \geq 1$であること．

(2) パラメタ

ZA …… **入力**．複素ヘッセンベルグ行列***A***.
演算後，内容は保存されない．
ZA (K,N) なる複素数型2次元配列.

K …… **入力**．配列ZAの整合寸法 ($\geq n$).

N …… **入力**．複素ヘッセンベルグ行列***A***の次数*n*.

ZE …… **出力**．固有値．J番目の固有値はZE (*J*) である．(*J* = 1 ,2,..., M).
大きさ*n*の複素数型1次元配列.

M …… **出力**．求まった固有値の個数.

ICON …… **出力**．コンディションコード.
表 CHSQR-1参照.

表 CHSQR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1であった.	ZE (1) = ZA (1, 1) とする.
15000	固有値のすべてを求めつくすことはできなかった.	Mに求まった固有値の個数をセットする. 1≤M<N.
20000	固有値が，一つも求められなかった.	M = 0とする.
30000	K < N 又は N < 1であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数 ... REAL, AIMAG, CONJG, ABS, SIGN, AMAXl, SQRT, CSQRT

b. 注意

① 通常CHES2を実行した後，本サブルーチンによって固有値を求める.

② 固有ベクトルをも必要とする場合は，本サブルーチンを呼び出す前に，配列ZAの内容を他の領域へ移しておくこと.

c. 使用例

*n*次の複素行列を，サブルーチンCHES2によって複素ヘッセンベルグ行列へ変換した後，固有値を計算する．$n \leq 100$の場合.

```
C     **EXAMPLE**
      COMPLEX ZA(100,100),ZE(100)
      DIMENSION IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,ZA(I,J),J=1,N)
      CALL CHES2(ZA,100,N,IP,ICON)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      CALL CHSQR(ZA,100,N,ZE,M,ICON)
      WRITE(6,630)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,640) (I,ZE(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1',10X,
     * '** ORIGINAL MATRIX'//11X,
     * '** ORDER=',I5/)
  610 FORMAT(/2(5X,'A(',I3,',',I3,
     * ')=',2E15.7))
  620 FORMAT(/11X,'** CONDITION CODE=',
     * I5/)
  630 FORMAT('0'/11X,'** EIGENVALUES')
  640 FORMAT(5X,'E(',I3,')=',2E15.7)
      END
```

(4) 手法概要

複素ヘッセンベルグ行列***A***に対するQR法は，(4.1)のユニタリ相似変換を逐次施すことによって，Aの下副対角要素を零に収束させ，そのときの主対角要素を固有値とする方法である.

$$\boldsymbol{A}_{s+1} = \boldsymbol{Q}_s^* \boldsymbol{A}_s \boldsymbol{Q}_s \,, s = 1,2,... \qquad (4.1)$$

ただし$\boldsymbol{A}_1$=***A***とする.

ここで$\boldsymbol{Q}_s$は，(4.2) により$\boldsymbol{A}_s$をQR分解したときに，一意的に定まるユニタリ行列である.

$$\boldsymbol{A}_s = \boldsymbol{Q}_s \boldsymbol{R}_s \qquad (4.2)$$

(4.2) で$\boldsymbol{R}_s$は主対角要素が正の実数である上三角行列である.

QR法では通常収束を速めるために，$\boldsymbol{A}_s$の代りに原点移動を行った$\boldsymbol{A}_s - k_s\boldsymbol{I}$に対してQR分解を行う．ここで$k_s$は原点移動量である．すなわち，(4.2) の代りに，(4.3) により$\boldsymbol{Q}_s$, $\boldsymbol{R}s$を定める.

$$\boldsymbol{A}_s - k_s \boldsymbol{I} = \boldsymbol{Q}_s \boldsymbol{R}_s \qquad (4.3)$$

以下に複素QR法の手順を示す．ただし $\boldsymbol{A}_s = \left(a_{ij}^{(s)}\right)$, $\boldsymbol{A} = \left(a_{ij}\right)$ とする.

① $\boldsymbol{A}_s$の下副対角要素$a^{(s)}_{n,n-1},...,a^{(s)}_{21}$に，零と見なすことのできる要素があるかどうかを (4.4) により調べる．

$$\left\|a^{(s)}_{l,l-1}\right\|_1 < u\|\mathbf{A}\|, l = n, n-1, ..., 2 \qquad (4.4)$$

ただし，uは丸め誤差の単位，$\|\ \ \|_1$は複素数 z= $x + iy$に対して，

$$\|z\|_1 = |x| + |y| \qquad (4.5)$$

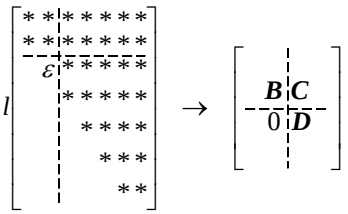
で定義されるノルム，$\|\boldsymbol{A}\|$は

$$\|A\| = \max_j \sum_{i=1}^{n} \left\|a_{ij}\right\|_1 \qquad (4.6)$$

で定義される$\boldsymbol{A}$のノルムである．
(4.4) を満足したとき，$a^{(s)}_{l,l-1}$を零と見なす．満足しないときは，②へ進む．

(a) もしも$l = n$であれば，$a^{(s)}_{nn}$を固有値とし，行列の次数nを1だけ減らして①へ戻る．もしも$n = 0$となれば処理を終了する．

(b) $2 \le l \le n - 1$のときは，図 CHSQR-1のように行列を分解する．そして，$\boldsymbol{D}$を$\boldsymbol{A}_s$とし，②へ進む．



［注意］εは零と見なした要素

図 CHSQR-1 ヘッセンベルグ行列の直和分解

② $\boldsymbol{A}_s$の右下すみの2×2の小行列の二つの固有値の中で小さいノルム$\|\ \ \|_1$を持つ方を原点移動量k_sとする．$\boldsymbol{A}_s$の対角要素からk_sを引いて$\boldsymbol{A}_s - k_s\boldsymbol{I}$を作る．

③ l =1, 2, . .., n –1の順に，2次元のギブンスのユニタリ変換$\boldsymbol{P}_l^*$を施し，$\boldsymbol{A}_s - k_s\boldsymbol{I}$を右上三角行列$\boldsymbol{R}_s$に変換する．

$$\boldsymbol{P}^*_{n-1}\boldsymbol{P}^*_{n-2}...\boldsymbol{P}^*_2\boldsymbol{P}^*_1(\boldsymbol{A}_s - k_s\boldsymbol{I}) = \boldsymbol{R}_s \qquad (4.7)$$

すなわち，(4.3) の$\boldsymbol{Q}_s$は，

$$\boldsymbol{Q}_s = \boldsymbol{P}_1\boldsymbol{P}_2...\boldsymbol{P}_{n-1} \qquad (4.8).$$

で表される．
ユニタリ行列$\boldsymbol{P}_l$は，そのときまでの変換を受けた行列$\boldsymbol{A}_s$ - $k_s\boldsymbol{I}$の第 l列の主対角要素 xと下副対角要素 yから次のように定められる．

$$\mathbf{P}_l = \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccccccc} & & & & l & l+1 & & & \end{array} \\ \begin{bmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & \mathrm{O} & \\ & & 1 & & & & & & \\ & & & \bar{c} & -\bar{s} & & & & \\ & & & s & c & & & & \\ & & & & & 1 & & & \\ & \mathrm{O} & & & & & \ddots & & \\ & & & & & & & 1 & \end{bmatrix} \end{array} \begin{array}{l} \\ \\ \\ \\ l \\ l+1 \\ \\ \\ \end{array}$$

ただし，

$$c = \bar{x}/r \qquad (4.9)$$
$$c = y/r \qquad (4.10)$$
$$r = \sqrt{|x|^2 + |y|^2} \qquad (4.11)$$

である．

④ l =1, 2, ..., n –1の順に，2次元のギブンスのユニタリ変換$\boldsymbol{P}_l$を，$\boldsymbol{R}_s$の右に施した後，対角要素に原点移動量k_sを加えて$\boldsymbol{A}_{s+1}$とする．

$$\boldsymbol{A}_{s+1} = \boldsymbol{R}_s\boldsymbol{P}_1\boldsymbol{P}_2...\boldsymbol{P}_{n-1} + k_s\boldsymbol{I} \qquad (4.12)$$

本サブルーチンでは，実際には④の処理を，③の処理の中に適当に組み入れて計算の高速化と記憶容量の節約を図っている．

なお．詳細については，参考文献 [12] 及び [13] pp.372-395を参照すること．

B21-15-0502 CHVEC, DCHVEC

複素ヘッセンベルグ行列の固有ベクトル (逆反復法)
CALL CHVEC (ZA, K, N, ZE, IND, M, ZEV, ZAW, ICON)

(1) 機能

n次の複素ヘッセンベルグ行列$\boldsymbol{A}$の指定された固有値 μに対応する固有ベクトル$\boldsymbol{x}$を，逆反復法により求める．固有ベクトルの正規化は行わない．

$n \geq 1$なること．

(2) パラメタ

ZA …… **入力**．複素ヘッセンベルグ行列$\boldsymbol{A}$．
ZA (K, N) なる複素数型2次元配列．

K …… **入力**．配列ZA, EV及びZAWの整合寸法 ($\geq n$).

N …… **入力**．行列$\boldsymbol{A}$の次数n.

ZE …… **入力**．固有値 μ．第 j番目の固有値μ_jは ZE (j) に格納する．
大きさMの複素数型1次元配列.

IND …… **入力**．固有ベクトル要不要の指定．μ_jに対応する固有ベクトルを要求するときIND (j) = 1とし，不要のときはIND (j) = 0とする．
大きさMの1次元配列.

M …… **入力**．配列ZEに格納されている固有値の総数 ($\leq n$).

ZEV …… **出力**．固有ベクトル$\boldsymbol{x}$.
INDの指示に従って求められた固有ベクトルは，ZEVの第1列から，列ごとに格納されるZEV (K, MK) なる複素数型2次元配列．ただし，MKは求める固有ベクトルの本数.
(使用上の注意参照).

ZAW …… **作業領域**．ZAW (K, N + 1) なる複素数型2次元配列.

ICON …… **出力**．コンディションコード.
表 CHVEC- 1参照.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, CSUM, MGSSL

② FORTRAN 基本関数 ... REAL, AIMAG, ABS, MIN0, AMAXl, FLOAT, SQRT

b. 注意

① 求められた固有ベクトルの本数MKは，演算後におけるINDの要素のうち，その値が1であるものの個数である．ところで，本サブルーチンは，ある固有ベクトルが求められなかったとき，その固有ベクトルのINDの情報を消去しているので，MKは演算前に指定した求めたい固有ベクトルの本数以下である.

表 CHVEC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1 であった.	ZEV (1, 1) = (1.0, 0.0) とする.
15000	ある固有値に対応する固有ベクトルが求められなかった.	求められなかった固有ベクトルのINDの情報は消去される.
20000	すべての固有ベクトルが求められなかった.	INDの情報はすべて消去される.
30000	M < 1, N < M 又は，K < Nであった.	処理を打ち切る.

② 本サブルーチンで用いる固有値は，サブルーチンCHSQRを用いて求めることができる．サブルーチンCHSQRで固有値を求めたときは，CHSQRのパラメタZE, Mが，そのまま本サブルーチンのパラメタとして使用できる.

③ 本サブルーチンは複素ヘッセンベルグ行列の固有ベクトルを求めるためのものである．複素行列の固有ベクトルを部分的に求める場合は,
(a) まず，サブルーチンCHES2により複素行列を複素ヘッセンベルグ行列に変換し,
(b) 次に，サブルーチンCHSQRにより複素行列の固有値を求め,
(c) 本サブルーチンにより複素ヘッセンベルグ行列の固有ベクトルを求め,
(d) 最後にサブルーチンCHBK2により元の複素行列の固有ベクトルへ逆変換すること．
なお，複素行列の固有値固有ベクトルをすべて求めるときは，サブルーチンCEIG2を用いた方がよい.

④ 本サブルーチンは，③に示した手順で複素行列の固有ベクトルを部分的に求める場合にも用いることを前提としている．したがって，固有ベクトルの正規化は行わない．正規化が必要な場合はサブルーチンCNRMLで行うことができる.

⑤ サブルーチンCHBK2やCNRMLを使用するときは，本サブルーチンのパラメタIND, M, ZEVはそのままCHBK2やCNRMLの入力パラメタとして使用できる.

c. 使用例

n次の複素行列の固有値をCHES2とCHSQRの各サブルーチンにより求め，固有ベクトルを本サブルーチンとCHBK2により求め，サブルーチン，CNRMLにより長さを1に ($\|\boldsymbol{x}\|_2=1$) 正規化する.

$n \leq 100$の場合.

```
C     **EXAMPLE**
      COMPLEX ZA(100,100),ZAW(100,101),
     *        ZE(100),ZEV(100,100)
      DIMENSION IND(100),IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,ZA(I,J),J=1,N)
      CALL CHES2(ZA,100,N,IP,ICON)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      DO 30 J=1,N
      IM=MIN0(J+1,N)
      DO 30 I=1,IM
   30 ZAW(I,J)=ZA(I,J)
      CALL CHSQR(ZAW,100,N,ZE,M,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      DO 40 I=1,M
   40 IND(I)=1
      CALL CHVEC(ZA,100,N,ZE,IND,M,ZEV,
     *           ZAW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL CHBK2(ZEV,100,N,IND,M,ZA,
     *           IP,0.0,ICON)
      CALL CNRML(ZEV,100,N,IND,M,2,ICON)
      CALL CEPRT(ZE,ZEV,100,N,IND,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',
     *       5X,'N=',I3)
  610 FORMAT(/2(5X,'A(',I3,',',I3,')=',
     *       2E15.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチンCEPRTは，複素行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチンCEIG2の使用例参照のこと．

(4) 手法概要

n次の複素ヘッセンベルグ行列$\boldsymbol{A}$の固有値 λに対応する固有ベクトル$\boldsymbol{x}$を逆反復法により求める．

逆反復法は，行列$\boldsymbol{A}$とその固有値λ_jの近似値μ_jが与えられたとき，

$$\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)\boldsymbol{x}_r=\boldsymbol{x}_{r-1},\ r=1,2,... \tag{4.1}$$

を，適当な初期ベクトル$\boldsymbol{x}_0$を与えて，反復的に解いてゆき，収束条件を満足したときの$\boldsymbol{x}_r$を固有ベクトルとして採用する方法である．

いま，n次の行列$\boldsymbol{A}$の固有値をλ_i (i=1,2,...,n), λ_iに対応する固有ベクトルを$\boldsymbol{u}_i$とする．

初期ベクトル$\boldsymbol{x}_0$が，行列$\boldsymbol{A}$の固有ベクトル$\boldsymbol{u}_i$の1次結合，

$$\boldsymbol{x}_0=\sum_{i=1}^{n}\alpha_i\boldsymbol{u}_i \tag{4.2}$$

で表せるから，$\boldsymbol{x}_r$は，すべての固有値λ_iが異なるものとすると，

$$x_r=1/\left(\lambda_j-\mu_j\right)^r \left[\alpha_j\boldsymbol{u}_j+\sum_{\substack{i=1\\ i\neq j}}^{n}\alpha_i\boldsymbol{u}_i\left(\lambda_j-\mu_j\right)^r/\left(\lambda_i-\mu_i\right)^r\right] \tag{4.3}$$

と書けることがわかる．$i\neq j$ なる i については，一般に$\left|\left(\lambda_j-\mu_j\right)/\left(\lambda_i-\mu_i\right)\right|\ll 1.0$ と考えられるので，(4.3) は，$\alpha_j\neq 0$であれば，r が大きくなると，$\boldsymbol{x}_r$は $\boldsymbol{u}_j$ の定数倍に接近することを示している．

(4.1) の連立方程式は，$\boldsymbol{A}-\mu_j\boldsymbol{I}$ を単位下三角行列$\boldsymbol{L}$と上三角行列$\boldsymbol{U}$とに分解し，

$$\boldsymbol{LU}\boldsymbol{x}_r=\boldsymbol{P}\boldsymbol{x}_{r-1} \tag{4.4}$$

(Pは，軸交換のための置換行列である).

として解く．(4.4) を解くことは，

$$\boldsymbol{L}\boldsymbol{y}_{r-1}=\boldsymbol{P}\boldsymbol{x}_{r-1}\quad \text{(前進代入)} \tag{4.5}$$

$$\boldsymbol{U}\boldsymbol{x}_r=\boldsymbol{y}_{r-1}\quad \text{(後退代入)} \tag{4.6}$$

なる二つの連立方程式を解くことに帰着する．

初期ベクトル$\boldsymbol{x}_0$ はどのように与えてもよいから，

$$\boldsymbol{y}_0=\boldsymbol{L}^{-1}\boldsymbol{P}\boldsymbol{x}_0 \tag{4.7}$$

なる$\boldsymbol{y}_0$が特定の形，例えば，$\boldsymbol{y}_0$=(1,1,1,...,1)^T となるように$\boldsymbol{x}_0$が与えられたとしてもよい．したがって，第1回目には，(4.5) の前進代入を省略できる．一般には，2回目以降は前進代入と後退代入を繰り返すことにより固有ベクトルを求めること ができる．

本サブルーチンでは，逆反復法を適用するにあたり，次のようにしている．

a. 初期ベクトルの選定

初期ベクトル $\boldsymbol{y}_0$ としては，

$$\boldsymbol{y}_0=\mathrm{EPS1}\cdot\boldsymbol{r} \tag{4.8}$$

を用いている．ここにベクトル$\boldsymbol{r}$ は，

$$r_k=k\Phi-\left[k\Phi\right],\ k=1,2,..... \tag{4.9}$$

で定義される．一種の乱数の相連続したn個の値を成分とするベクトルである．ただし，Φ は

$$\Phi=\left(\sqrt{5}+1\right)/2 \tag{4.10}$$

で与えられる黄金分割比である．また，EPS1は

$$\mathrm{EPS1}=u\cdot\|\boldsymbol{A}\| \tag{4.11}$$

である．ここに，uは丸め誤差の単位，ノルム $\|\boldsymbol{A}\|$は，

$$\|\boldsymbol{A}\|=\max_j\cdot\sum_{i=1}^{n}\|a_{ij}\|_1 \tag{4. 12}$$

で与えられる．ただし，複素数 $z = x + iy$に対して

$$\|z\|_1 = |x| + |y| \qquad (4.13)$$

と定義する．

b. 2回目以降の反復ベクトルの正規化

$$\mathbf{x}_r = (x_{r1}, x_{r2}, \ldots, x_{rn})^{\mathrm{T}}$$

とおくとき，$r \geq 1$に対しては，

$$\|\boldsymbol{x}_r\|_1 = \sum_{i=1}^{n} \|x_{ri}\|_1 = \mathrm{EPS1} \qquad (4.14)$$

となるように正規化する．

c. 収束判定の方法
後退代入が終了したとき，収束が達成されたかどうかを，b.の正規化を行う前の$\boldsymbol{x}_r$について，

$$\|\boldsymbol{x}_r\|_1 \geq 0.1/\sqrt{n} \qquad (4.15)$$

により判定している．すなわち，(4.15) が満足されれば，$\boldsymbol{x}_r$を固有ベクトルとして採用する．(4.15) が成り立たなかったときは，b.の正規化を行って反復を続ける．5回反復しても収束が達成されないときは，それ以上の処理を断念し，非収束のまま次の固有ベクトルの処理に移る．

d. 固有値が重根又は接近根である場合の対策
本サブルーチンでは初期ベクトル $\boldsymbol{y}_0$ は一種の乱数で作られるので，固有値が重根又は接近根である場合にも特別の処置を行う必要がない．ただし，数値解の再現性を確保するために，乱数は本サブルーチンの呼び出しごとに初期化される．

なお，詳細については，参考文献 [13] pp.418-439 を参照のこと．

C22-15-0101 CJART, DCJART

複素係数高次代数方程式（ヤラット法）
CALL CJART (ZA, N, Z, ICON)

(1) 機能

複素係数高次代数方程式

$$a_0 z^n + a_1 z^{n-1} + \ldots\ldots + a_n = 0 \qquad (1.1)$$

(a_i : 複素数, $|a_0| \neq 0$)

の根をヤラット法により求める．$n \geq 1$.

(2) パラメタ

ZA…… **入力**．代数方程式の係数.
大きさ$n+1$の複素数型1次元配列.
ZA (1) = a_0
ZA (2) = a_1
…
ZA(N+1)=a_n
の順に与える.
演算後内容は保存されない.

N…… **入力**．代数方程式の次数n.
出力．求まった根の個数.
(使用上の注意参照).

Z …… **出力**．n 個の根．大きさnの複素数型1次元配列.
根は求まった順にZ (1), Z (2), ... に出力される.
したがって求まった根の個数をNとするとそれらの根はZ (1) からZ (N) までにセットされている.

ICON …… **出力**．コンディションコード.
表 CJART-1参照.

表 CJART-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	n根を，すべては求めつくすことができなかった.	求まった根の個数をパラメタNに，そして，根をZ(1) ～ Z(N) に出力する.
30000	n < 1，又は $|a_0|=0$であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, CQDR, MGSSL, UCJAR
② FORTRAN 基本関数 ... CMPLX, SQRT, CABS, CONJG, AIMAG, REAL, ABS.

b. 注意
① 配列ZA, Zは本サブルーチンを呼び出す側のプログラムにおいてCOMPLEX宣言をすること.
② nが1または2の場合は，ヤラット法を使わず，根の公式を使って計算している.
③ n次代数方程式は n個の根を持つが，ごくまれに，必ずしもそのすべてを求めることができないことがある．利用者はこのことに注意され，パラメタICONあるいはNの演算後の値を確認されたい.

c. 使用例
次数n，複素係数 a_iを入力し，根を求める.
$1 \leq n \leq 50$の場合.

```
C       **EXAMPLE**
        DIMENSION ZA(51),Z(50)
        COMPLEX ZA,Z
        READ(5,500) N
        N1=N+1
        READ(5,510) (ZA(I),I=1,N1)
        DO 10 I=1,N1
        K=I-1
   10   WRITE(6,600) K,ZA(I)
        CALL CJART(ZA,N,Z,ICON)
        WRITE(6,610) N,ICON
        IF(ICON.EQ.30000) STOP
        WRITE(6,620) (I,Z(I),I=1,N)
        STOP
  500   FORMAT(I2)
  510   FORMAT(2F10.0)
  600   FORMAT(10X,'A(',I2,')=',2E20.8)
  610   FORMAT(10X,'N=',I2,5X,'ICON=',I5)
  620   FORMAT(10X,'Z(',I2,')=',2E20.8)
        END
```

(4) 手法概要

本サブルーチンが適用している手法は，反復解法の一種であるGarside-Jarratt-Mackの方法を多少，修正したものである．代数方程式を

$$f(z) \equiv a_0 z^n + a_1 z^{n-1} + \ldots\ldots + a_n = 0 \qquad (4.1)$$

とすると，この根と

$$1/F(z) \equiv f(z)/f'(z) = 0 \qquad (4.2)$$

の根とは一致し，しかも (4.2) は単根しか持たないので (4.2) を解いた方が収束の速さにおいて有利である．このことが本サブルーチンの基本である.

反復公式について

z_1, z_2, z_3 をある根に対する三つの初期値とし，$|f(a_3)| \leq |f(a_2)| \leq |f(a_1)|$ を満たしているものとする．このとき反復公式として以下の三つを使い分ける．以下 $F_i \equiv F(z_i)$ $i = 1, 2, 3$としておく.

方法1

ある根αの近傍では

$$1/F(z) \approx (z-a)/(b+cz) \qquad (4.3)$$

a, b, cは定数

と近似できる．ゆえに z_1, z_2, z_3において (4.3) が成り立つように aを決めれば，その aは新しい近似根となり得る．すなわち，

$$a = z_3 + \frac{(z_2 - z_3)(z_3 - z_1)(F_2 - F_1)}{(z_3 - z_2)(F_2 - F_1) + (z_1 - z_2)(F_3 - F_2)} \quad (4.4)$$

方法2 (ニュートン法)

次のa'を新しい近似値とする．

$$a' = z_3 - 1/F_3 \quad (4.5)$$

方法3 (修正ニュートン法)

条件，

$$|f(z_3)| < \sqrt{u_1 |a_n|} \quad (4.6)$$

ここで $u_1 = \sqrt{u}$ ，u は丸め誤差の単位

を満たすとき，求めようとしている根の多重度 mを $m \approx (z_3 - a)F_3$と近似し，次のa''を新しい近似根とする．

$$a'' = z_3 - m/F_3 \quad (4.7)$$

常に方法1，方法2を併用し修正量の小さい方の結果を新しい近似根z_4として採用し，z_4, z_3, z_2を新たにz_3, z_2, z_1とみなして繰り返す．なお方法3は収束を速めるために使う．

初期値のとり方

$$|a_n / a_0|^{1/n} = 5w \quad (4.8)$$

とし，初期値 z_1, z_2, z_3 として

$$iw, -w + iw, 2iw \quad (4.9)$$

をとる．その順序は $|f(z_3)| \le |f(z_2)| \le |f(z_1)|$ となるようにする．もし50回の反復を行っても収束しなかった場合は，再び初期値として

$$-3w + 2iw, -2w + 2iw, -3w + 3iw$$

をとり実行する．根αが求まったなら

$$g(z) = f(z)/(z - \alpha)$$

と次数を下げ，この$g(z)$ を新たに $f(z)$ と見なし，このときのw を求め，

$$-w \pm iw, -w \pm 2iw, \overline{\alpha} \quad (4.10)$$

を $f(z)=0$を解くための初期値とする．複合は$\overline{\alpha}$ の虚部の符号に一致するようにとる．

オーバフローの防止

$f(z)$ あるいは $f'(z)$ を計算するときオーバフローを起す恐れがあるのでそれを防止するためここでは一つの手段として，$f(z)$ の係数を，それらの絶対値の相加平均で正規化する．

収束判定法

$$|f(z_3)| \le 10nu \sum_{i=0}^{n} |a_i z_3^{n-i}| \quad (4.11)$$

を満たしたとき収束したと判断しz_3 を根として採用する．

なお，詳細については，参考文献 [26], [27] を参照すること．

A22-15-0202 CLU, DCLU

複素行列のLU分解（クラウト法）
CALL CLU (ZA, K, N, EPSZ, IP, IS, ZVW, ICON)

(1) 機能

$n \times n$ の正則な複素行列$\boldsymbol{A}$をクラウト法によりLU分解する.

$$\boldsymbol{PA} = \boldsymbol{LU} \qquad (1.1)$$

ただし，$\boldsymbol{P}$ は部分ピボッティングによる行の入れ換えを行う置換行列，$\boldsymbol{L}$ は下三角行列，$\boldsymbol{U}$ は単位上三角行列である．$n \geq 1$であること.

(2) パラメタ

ZA……**入力**．行列$\boldsymbol{A}$.
出力．行列$\boldsymbol{L}$と行列$\boldsymbol{U}$.
図 CLU-1参照.
ZA(K, N) なる複素数型2次元配列.

K……**入力**．配列ZAの整合寸法 (≥ N).

N……**入力**．行列$\boldsymbol{A}$の次数 n.

EPSZ……**入力**．ピボットの相対零判定値 (≥0.0).
0.0のときは標準値が採用される.
(使用上の注意①参照)

IP……**出力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさ nの1次元配列.
(使用上の注意②参照)

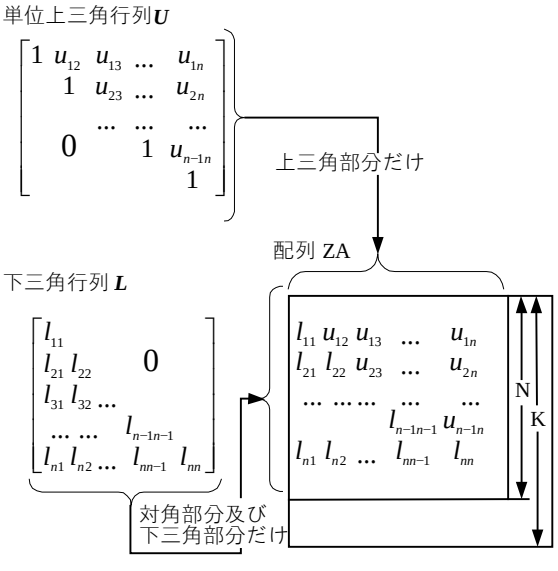


図 CLU-1 演算後の配列ZAにおける$\boldsymbol{L}$及び$\boldsymbol{U}$の各要素の格納方法

IS……**出力**．行列 $\boldsymbol{A}$の行列式を求めるための情報．演算後の配列ZAの n個の対角要素とISの値を掛け合わせると行列式が得られる.

ZVW……**作業領域**．大きさ nの複素数型1次元配列.

ICON……**出力**．コンディションコード.
表 CLU-1参照.

表 CLU-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	行列$\boldsymbol{A}$のある行の要素がすべて零であったか又はピボットが相対的に零となった．行列$\boldsymbol{A}$は非正則の可能性が強い.	処理を打ち切る.
30000	K < N, N < 1 又は, EPSZ < 0.0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... AMACH, CSUM, MGSSL

② FORTRAN 基本関数 ..... REAL, AIMAG, ABS

b. 注意

① ピボットの相対零判定値EPSZに10^{-s}を設定したとすると，この値は次の意味を持っている.
すなわち，クラウト法によるLU分解の過程でピボットの実部と虚部の値が，同時に10進s桁以上の桁落ちを生じた場合に，そのピボットを相対的に零と見なし，ICON = 20000として処理を打ち切る.
EPSZの標準値は丸め誤差の単位をu としたとき，EPSZ = $16 \cdot u$である.
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZへ極小の値を与えればよいが，その結果は保証されない.

② トランスポジションベクトルとは，部分ピボッティングを行うLU分解

$$\boldsymbol{PA} = \boldsymbol{LU}$$

における置換行列に相当する.
本サブルーチンでは，部分ピボッティングに伴い配列ZAの内容を実際に交換している．すなわち，分解のJ段階目 (J = 1 , ..., n) において第 I 行がピボット行として選択された場合には，配列ZAの第 I行と第J行の内容が交換される．そしてその履歴を示すためにIP(J) にIが格納される.

③ 本サブルーチンに続けて，サブルーチンCLUXを呼び出すことにより，連立1次方程式を解くことができる．しかし，通常はサブルーチンLCXを呼び出せば，1度に解が求められる.

c. 使用例

$n \times n$の複素行列を入力して，LU分解する．

$n \leq 100$の場合．

```
C     **EXAMPLE**
      DIMENSION ZA(100,100),ZVW(100),
     *          IP(100)
      COMPLEX ZA,ZVW,ZDET
      READ(5,500) N
      IF(N.LE.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,ZA(I,J),J=1,N),
     *             I=1,N)
      CALL CLU(ZA,100,N,0.0,IP,IS,ZVW,
     *         ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      ZDET=CMPLX(FLOAT(IS),0.0)
      DO 10 I=1,N
      ZDET=ZDET*ZA(I,I)
   10 CONTINUE
      WRITE(6,620) (I,IP(I),I=1,N)
      WRITE(6,630) ((I,J,ZA(I,J),J=1,N),
     *             I=1,N)
      WRITE(6,640) ZDET
      STOP
  500 FORMAT(I5)
  510 FORMAT(5(2F8.3))
  600 FORMAT('1'//10X,'**INPUT MATRIX **',
     * /12X,'ORDER=',I5/(5X,
     * 2('(',I3,',',I3,')',2E15.8,2X)))
  610 FORMAT('0',10X,'CONDITION CODE =',
     * I5)
  620 FORMAT('0',10X,'TRANSPOSITION ',
     * 'VECTOR'/(10X,7('(',I3,')',I5,5X)))
  630 FORMAT('0',10X,'OUTPUT MATRIX'/
     * (5X,2('(',I3,',',I3,')',2E15.8,
     * 2X)))
  640 FORMAT('0',10X, 'DETERMINANT OF ',
     * 'MATRIX',2E20.8)
      END
```

(4) 手法概要

a. クラウト法

$n \times n$の正則な複素行列$\boldsymbol{A}$は通常，部分ピボッティングによる行の入れ換えを行って，下三角行列$\boldsymbol{L}$と単位上三角行列$\boldsymbol{U}$の積に分解することができる．

$$\boldsymbol{PA} = \boldsymbol{LU} \tag{4.1}$$

ただし，$\boldsymbol{P}$ は部分ピボッティングによる行の入換えを行う置換行列である．この$\boldsymbol{L}$と$\boldsymbol{U}$ の要素を逐次求める方法の１つがクラウト法である．本サブルーチンでは，次の等式により，$\boldsymbol{L}$の j 番目の列，$\boldsymbol{U}$ の j 番目の列 (j = l, ... , n) の順に値を求める．

$$u_{ij} = \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}\right) \Big/ l_{ij}, \quad i = 1,..., j-1 \tag{4.2}$$

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} ll_{kj}, \quad i = j,..., n \tag{4.3}$$

ただし，$\boldsymbol{A}$ = (a_{ij}) , $\boldsymbol{L}$ = (l_{ij}) , $\boldsymbol{U}$ = (u_{ij}) である．また，実際には置換行列による行の入換えが必要である．

クラウト法はガウス消去法の変形であって，両者は同じ計算を行うので計算量は等しいが，計算順序は異なっている．クラウト法では$\boldsymbol{L}$と$\boldsymbol{U}$の要素を求める際に (4.2), (4.3) で一度に計算するが，このときの積和計算を精度を上げて計算することにより，丸め誤差の影響を少なくしている．

b. 部分ピボッティング

例えば，行列$\boldsymbol{A}$が

$$A = \begin{bmatrix} 0.0 + i \cdot 0.0 & 1.0 + i \cdot 0.0 \\ 1.0 + i \cdot 0.0 & 0.0 + i \cdot 0.0 \end{bmatrix}$$

で与えられたとき，これは数値的に極めて安定であるにもかかわらず，このままではもはやLU分解は不可能である． また行列が数値的に安定であっても，そのまま単純にLU分解したのでは大きな誤差が生じる場合もある．本サブルーチンでは，このような事態を回避するために，行を均衡化 (row equilibrated) した部分ピボッティングを行っている．

なお，詳細については，参考文献 [1], [3] 及び [4] を参照すること．

A22-15-0602 CLUIV, DCLUIV

LU分解された複素行列の逆行列
CALL CLUIV (ZFA, K, N, IP, ICON)

(1) 機能

LU分解された$n \times n$の複素行列$\boldsymbol{A}$の逆行列$\boldsymbol{A}^{-1}$を求める.

$$\boldsymbol{A}^{-1} = \boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P}$$

ただし，$\boldsymbol{L}$, $\boldsymbol{U}$ はそれぞれ $n \times n$ の下三角行列と単位上三角行列であり，$\boldsymbol{P}$ はLU分解におけるピボッティングによる行の入換えを示す置換行列である.

$n \geq 1$であること.

(2) パラメタ

ZFA ····· **入力**. 行列$\boldsymbol{L}$と行列$\boldsymbol{U}$.
出力. 逆行列$\boldsymbol{A}^{-1}$.
ZFA (K, N) なる2次元配列.
図 CLUIV-1参照.

K ····· **入力**. 配列ZFAの整合寸法 (≥ N).

N ····· **入力**. 行列$\boldsymbol{L}$, $\boldsymbol{U}$ の次数n.

IP ····· **入力**. ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさnの１次元配列.
(使用上の注意③参照)

ICON ····· **出力**. コンディションコード.
表 CLUIV-1参照.

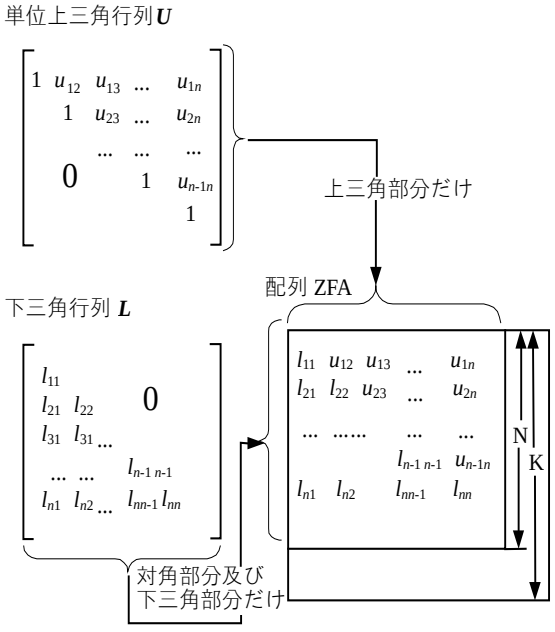


図 CLUIV-1 配列ZFAにおける$\boldsymbol{L}$及び$\boldsymbol{U}$の各要素の格納方法

表 CLUIV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	複素行列が非正則であった.	処理を打ち切る.
30000	K < N, N < 1 又は，IPに誤りがあった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ..... CSUM, MGSSL
② FORTRAN 基本関数 ..... なし.

b. 注意
① 本サブルーチンは，LU分解された複素行列を入力して，分解された元の複素行列の逆行列を計算する．分解はサブルーチンCLUにより行い，続けて本サブルーチンを呼び出すことにより逆行列を求めることができる.
② 連立１次方程式を解く場合には，サブルーチンLCXを用いること．逆行列を求めて解くことは，LCXを用いるときよりも演算回数が多くなるので避けた方がよい．本サブルーチンは逆行列が必要なときだけ使用すること.
③ トランスポジションベクトルとは，部分ピボッティングを行うLU分解

$$\boldsymbol{PA} = \boldsymbol{LU}$$

における置換行列$\boldsymbol{P}$に相当する.
サブルーチンCLUの使用上の注意②を参照すること.

c. 使用例
$n \times n$の複素行列を入力して，その逆行列を求める．$n \leq 100$の場合.

```
C     **EXAMPLE**
      DIMENSION ZA(100,100),ZVW(100),
     *          IP(100)
      COMPLEX ZA,ZVW
      READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,ZA(I,J),J=1,N),
     *             I=1,N)
      CALL CLU(ZA,100,N,0.0,IP,IS,ZVW,
     *         ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      CALL CLUIV(ZA,100,N,IP,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,630) ((I,J,ZA(I,J),I=1,N),
     *             J=1,N)
      STOP
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT(//11X,'**INPUT MATRIX**'/12X,
     *'ORDER=',I5/(2X,4('(',I3,',',I3,')',
     *2E16.8)))
  610 FORMAT('0',10X,'CONDITION ',
     *'CODE(CLU)=',I5)
  620 FORMAT('0',10X,'CONDITION ',
```

```
     *'CODE(CLUIV)=',I5)
630 FORMAT('0',10X,'**INVERSE MATRIX**',
     */(2X,4('(',I3,',',I3,')',2E16.8)))
      END
```

(4) 手法概要

$n \times n$の複素行列$\boldsymbol{A}$のLU分解された行列$\boldsymbol{L}$, $\boldsymbol{U}$ 及びLU分解におけるピボッティングによる行の入換えを示す置換行列$\boldsymbol{P}$が与えられたとき，$\boldsymbol{A}$の逆行列$\boldsymbol{A}^{-1}$を求めることを考える．
ところで

$$\boldsymbol{PA} = \boldsymbol{LU} \tag{4.1}$$

であるから

$$\boldsymbol{A}^{-1} = \left(\boldsymbol{P}^{-1}\boldsymbol{LU}\right)^{-1} = \boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P} \tag{4.2}$$

となる．そこで，$\boldsymbol{L}$と$\boldsymbol{U}$の逆行列を求めて，$\boldsymbol{U}^{-1}$に対して$\boldsymbol{L}^{-1}$を右側から掛け，その結果に対して，$\boldsymbol{P}$を右側から掛けて$\boldsymbol{A}$の逆行列$\boldsymbol{A}^{-1}$を計算する．

なお，以降の説明のために$\boldsymbol{L}$及び$\boldsymbol{U}$を(4.3)で表す．

$$\boldsymbol{L} = \left(l_{\boldsymbol{ij}}\right),\ U = \left(u_{\boldsymbol{ij}}\right) \tag{4.3}$$

a. $\boldsymbol{L}^{-1}$の計算

下三角行列$\boldsymbol{L}$の逆行列$\boldsymbol{L}^{-1}$も下三角行列となるので，$\boldsymbol{L}^{-1}$を (4.4) で表すと，

$$\boldsymbol{L}^{-1} = \left(\tilde{l}_{ij}\right) \tag{4.4}$$

$\boldsymbol{L}\boldsymbol{L}^{-1}=\boldsymbol{I}$より，(4.5) が成り立つ．

$$\sum_{k=1}^{n} l_{ij}\tilde{l}_{ij} = \delta_{ij},$$

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{4.5}$$

(4.5) を

$$\sum_{k=j}^{i-1} l_{ik}\tilde{l}_{kj} + l_{ii}\tilde{l}_{ij} = \delta_{ij}$$

と変形させて，$\boldsymbol{L}^{-1}$の第j列目 (j=1,...,n) の要素 $\tilde{l}_{ij}$ を次のように求める．

$$\left.\begin{aligned} \tilde{l}_{ij} &= \left(-\sum_{k=j}^{i-1} l_{ik}\tilde{l}_{kj}\right) \Big/ l_{ii},\ \ i = j+1,...,n \\ \tilde{l}_{jj} &= 1/l_{jj} \end{aligned}\right\} \tag{4.6}$$

ここで，$l_{ii} \neq 0\ (i = j,...,n)$とする．

b. $\boldsymbol{U}^{-1}$の計算

単位上三角行列$\boldsymbol{U}$ の逆行列$\boldsymbol{U}^{-1}$ も単位上三角行列となるので，$\boldsymbol{U}^{-1}$を (4.7) で表すと，

$$\boldsymbol{U}^{-1} = \left(\tilde{u}_{ij}\right) \tag{4.7}$$

$\boldsymbol{U}\boldsymbol{U}^{-1}=\boldsymbol{I}$ より，(4.8) が成り立つ．

$$\sum_{k=1}^{n} u_{ik}\tilde{u}_{kj} = \delta_{ij}$$

$$\delta_{ij} = \begin{cases} 1 & , \quad i = j \\ 0 & , \quad i \neq j \end{cases} \tag{4.8}$$

$u_{ii} = 1$であるので，(4.8) を

$$\tilde{u}_{ij} + \sum_{k=i+1}^{j} u_{ik}\tilde{u}_{kj} = \delta_{ij}$$

と変形させて，かつ$\tilde{u}_{jj}$=1を考慮して$\boldsymbol{U}^{-1}$の第 j列目 (j=n,...,2) の要素 $\tilde{u}_{ij}$ を次のように求める．

$$\tilde{u}_{ij} = -u_{ij} - \sum_{k=i+1}^{j-1} u_{ik}\tilde{u}_{kj}, \quad i = j-1,...,1 \tag{4.9}$$

c. $\boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P}$の計算

行列$\boldsymbol{U}^{-1}$と$\boldsymbol{L}^{-1}$の掛け合わせた結果を行列$\boldsymbol{B}$とするとき，行列$\boldsymbol{B}$の要素 b_{ij}は

$$b_{ij} = \sum_{k=i}^{n} \tilde{u}_{ik}\tilde{l}_{kj}, \quad i = 1,..., j-1$$

$$b_{ij} = \sum_{k=i}^{n} \tilde{u}_{ik}\tilde{l}_{kj}, \quad i = j,..., n$$

より求められる．そこで，$\tilde{u}_{ii}$=1を考慮して$\boldsymbol{B}$の第 j列目 (j=1, ...,n) の要素b_{ij}は (4.10) の計算をして求める．

$$\left.\begin{aligned} b_{ij} &= \sum_{k=j}^{n} \tilde{u}_{ik}\tilde{l}_{kj}, \quad i = 1,..., j-1 \\ b_{ij} &= \tilde{l}_{ij} + \sum_{k=i+1}^{n} \tilde{u}_{ik}\tilde{l}_{kj}, \quad i = j,..., n \end{aligned}\right\} \tag{4.10}$$

次に，行列$\boldsymbol{B}$に対して置換行列を掛け合わせて，$\boldsymbol{A}^{-1}$の要素 a_{ij} を求める．しかし，実際には，トランスポジションベクトル$\boldsymbol{IP}$の値を参照して行列$\boldsymbol{B}$の列の入換えだけで$\boldsymbol{A}^{-1}$の要素を求める．

なお，(4.6), (4.9) 及び (4.10) に関する計算のうち積和計算部分は精度を上げて行うことにより，丸め誤差の影響を少なくしている．

また，詳細については，参考文献 [1] を参照すること．

A22-15-0302 CLUX, DCLUX

LU分解された複素行列の連立１次方程式
CALL CLUX (ZB, ZFA, K, N, ISW, IP, ICON)

(1) 機能

LU分解された複素係数行列を持つ連立1次方程式

$$\boldsymbol{LUx = Pb} \qquad (1.1)$$

を解く．ただし，$\boldsymbol{L}$, $\boldsymbol{U}$ はそれぞれ $n \times n$の下三角行列と単位上三角行列，$\boldsymbol{P}$ は置換行列 (係数行列をLU分解したときの部分ピボッティングによる行の入換えを行う)，$\boldsymbol{b}$は n次元の複素定数ベクトル，$\boldsymbol{x}$は n次元の解ベクトルである．なお，方程式 (1.1) の代わりに

$$\boldsymbol{Ly = Pb} \qquad (1.2)$$
$$\boldsymbol{Uz = b} \qquad (1.3)$$

のいずれかを解くこともできる．
$n \geq 1$であること．

(2) パラメタ

ZB……**入力**．定数ベクトル$\boldsymbol{b}$.
出力．解ベクトル$\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$のうちいずれか.
大きさnの１次元配列.

ZFA ……**入力**．行列 $\boldsymbol{L}$と行列$\boldsymbol{U}$.
図 CLUX-1参照.
ZFA (K, N) なる複素数型2次元配列.

K……**入力**．配列ZFAの整合寸法 (≥N).

N……**入力**．行列 $\boldsymbol{L}$, $\boldsymbol{U}$ の次数 n.

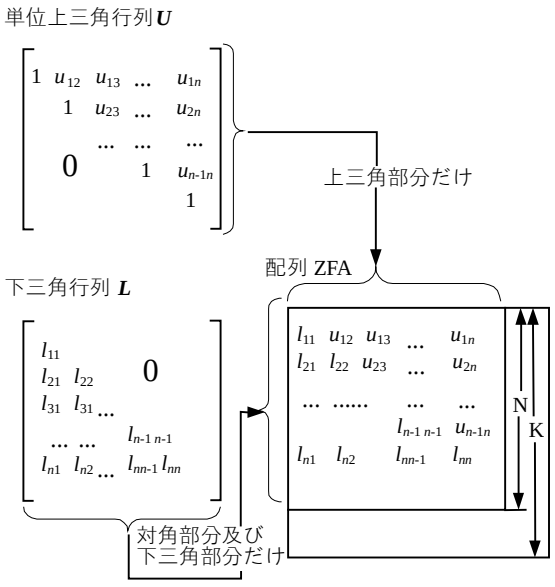


図 CLUX-1 配列ZFAにおける$\boldsymbol{L}$及び$\boldsymbol{U}$の各要素の格納方法

ISW……**入力**．制御情報.
1のとき，$\boldsymbol{x}$を求める.
2のとき，$\boldsymbol{y}$を求める.
3のとき，$\boldsymbol{z}$を求める.

IP ……**入力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさnの1次元配列.
(サブルーチンCLUの使用上の注意②参照)

ICON……**出力**．コンディションコード.
表 CLUX-1参照.

表 CLUX- 1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列が非正則であった.	処理を打ち切る.
30000	K<N, N<1, ISW ≠ 1, 2, 3 又は, IPに誤りがあった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ..... CSUM, MGSSL
② FORTRAN 基本関数 ..... なし.

b. 注意
① 連立1次方程式を解く場合，サブルーチンCLUを呼び出して，係数行列をLU分解させてから，本サブルーチンを呼び出せば方程式を解くことができる．しかし，通常サブルーチンLCXを呼び出せば，一度に解が求められる.

c. 使用例
$n \times n$の複素係数行列をサブルーチンCLUによりLU分解させてから，複素係数連立1次方程式を解く．$n \leq 100$の場合.

```
C       **EXAMPLE**
        DIMENSION ZA(100,100),ZB(100),
       *    ZVW(100),IP(100)
        COMPLEX ZA,ZB,ZVW
        READ(5,500) N
        READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
        READ(5,510) (ZB(I),I=1,N)
        WRITE(6,600) N,((I,J,ZA(I,J),J=1,N),
       *             I=1,N)
        WRITE(6,610) (I,ZB(I),I=1,N)
        CALL CLU(ZA,100,N,0.0,IP,IS,ZVW,
       *         ICON)
        WRITE(6,620) ICON
        IF(ICON.GE.20000) STOP
        CALL CLUX(ZB,ZA,100,N,1,IP,ICON)
        WRITE(6,630) ICON
        IF(ICON.GE.20000) STOP
        WRITE(6,640) (I,ZB(I),I=1,N)
        STOP
```

```
500 FORMAT(I5)
510 FORMAT(5(2F8.3))
600 FORMAT(///10X,'**COMPLEX MATRIX **',
   * /12X,'ORDER=',I5/(5X,2('(',I3,',',
   * I3,')',2E15.8,2X)))
610 FORMAT('0',10X,'CONSTANT VECTOR',
   * /(5X,3('(',I3,')',2E15.8,2X)))
620 FORMAT('0',10X,'CONDITION ',
   * 'CODE(CLU)=',I5)
630 FORMAT('0',10X,'CONDITION ',
   * 'CODE(CLUX)=',I5)
640 FORMAT('0',10X,'SOLUTION VECTOR',
   * /(5X,3('(',I3,')',2E15.8,2X)))
    END
```

(4) 手法概要

連立1次方程式

$$\boldsymbol{LUx} = \boldsymbol{Pb} \tag{4.1}$$

を解くことは，次の二つの方程式

$$\boldsymbol{Ly} = \boldsymbol{Pb} \tag{4.2}$$
$$\boldsymbol{Ux} = \boldsymbol{y} \tag{4.3}$$

を解くことに帰着される．

a. $\boldsymbol{Ly} = \boldsymbol{Pb}$を解く　(前進代入)

$$y_i = \left(b'_i - \sum_{k=1}^{i-1} l_{ik} y_k \right) \Big/ l_{ii}, \quad i = 1,...,n \tag{4.4}$$

なる式で逐次求める．ただし，$\boldsymbol{L}=(l_{ij})$, $\boldsymbol{y}^{\mathrm{T}}=(y_1,\ldots,y_n)$, $(\boldsymbol{Pb})^{\mathrm{T}}=(b'_1,\ldots, b'_n)$ である．

b. $\boldsymbol{Ux} = \boldsymbol{y}$を解く　(後退代入)

$$x_i = y_i - \sum_{k=i+1}^{n} u_{ik} x_k, \quad i = n,...,1 \tag{4.5}$$

なる式で逐次求める．ただし，$\boldsymbol{U}=(u_{ij})$, $\boldsymbol{x}^{\mathrm{T}}=(x_1,\ldots,x_n)$ である．

なお，(4.4) と (4.5) の積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている．

また，詳細については，参考文献 [1], [3] 及び [4] を参照すること．

B21-15-0702 CNRML, DCNRML

複素行列の固有ベクトルの正規化
CALL CNRML (ZEV, K, N, IND, M, MODE, ICON)

(1) 機能

*n*次の複素行列の *m*個の固有ベクトル$\boldsymbol{x}_i$ (*i*= l, ..., *m*) が与えられたとき，(1.1) 又は (1.2) の正規化を行いベクトル$\boldsymbol{y}_i$を求める.

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_\infty \qquad (1.1)$$

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_2 \qquad (1.2)$$

ただし，$n \geq 1$であること.

(2) パラメタ

ZEV ····· **入力**．*m*個の固有ベクトル$\boldsymbol{x}_i$ (*i*= l, 2, ..., *m*). ベクトル$\boldsymbol{x}_i$は，ZEVの第1列から，列ごとに格納すること．(使用上の注意①参照).
出力．正規化された*m*個の固有ベクトル$\boldsymbol{y}_i$. ベクトル$\boldsymbol{y}_i$は，対応するベクトル$\boldsymbol{x}_i$上に格納される.
ZEV(K, M) なる複素数型2次元配列.

K ····· **入力**．配列ZEVの整合寸法 (≥ *n*).

N ····· **入力**．複素行列の次数*n*.

IND ····· **入力**． 正規化を行う固有ベクトルの指定.
IND (*j*) = 1のときは，*j*番目の固有値に対応する固有ベクトルの正規化を行う.
IND (*j*) = 0のときは，行わない.
大きさMの1次元配列.
(使用上の注意①参照).

M ····· **入力**．配列INDの大きさ.
(使用上の注意①参照).

MODE ···· **入力**．正規化の方法の指示.
MODE=1のときは，(1.1) の正規化を行う.
MODE=2のときは，(1.2) の正規化を行う.

ICON ····· **出力**．コンディションコード.
表 CNRML-1参照.

表 CNRML-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
1000	N=1であった.	ZEV(1, 1) = (1.0. 0.0) とする.
3000	N<M, M<1, K<N, MODE ≠ 1, 2又はINDに誤りがあった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... REAL, AIMAG, AMAXl, SQRT

b. 注意

① サブルーチンCHVECあるいはCHBK2を呼び出した後のパラメタZEV, IND, Mは，本サブルーチンでそのまま入力できるようになっている.

c. 使用例

サブルーチンCEIG2により求めた *n*次の複素行列の固有ベクトルを，本サブルーチンにより$\|\boldsymbol{x}\|_\infty$ =1となるように正規化しなおす.

$n \leq 100$の場合.

```
C      **EXAMPLE**
       COMPLEX ZA(100,100),ZE(100),
      *        ZEV(100,100)
       DIMENSION IND(100),VW(100),IVW(100)
    10 READ(5,500) N
       IF(N.EQ.0) STOP
       READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
       WRITE(6,600) N
       DO 20 I=1,N
    20 WRITE(6,610) (I,J,ZA(I,J),J=1,N)
       CALL CEIG2(ZA,100,N,1,ZE,ZEV,
      *           VW,IVW,ICON)
       WRITE(6,620) ICON
       IF(ICON.GT.10000) GO TO 10
       DO 30 I=1,N
    30 IND(I)=1
       CALL CNRML(ZE,ZEV,100,N,IND,N,1,
      *           ICON)
       CALL CEPRT(ZE,ZEV,100,N,IND,N)
       GO TO 10
   500 FORMAT(I5)
   510 FORMAT(4E15.7)
   600 FORMAT('1',5X,'ORIGINAL MATRIX',
      *       5X,'N=',I3/)
   610 FORMAT(/2(5X,'A(',I3,',',I3,')=',
      *       2E15.7))
   620 FORMAT('0',20X,'ICON=',I5)
       END
```

本使用例中のサブルーチンCEPRTは，複素行列の固有値及び固有ベクトルを出力するサブルーチンである．この詳細についてはCEIG2の使用例参照のこと.

(4) 手法概要

*n*次の複素行列の*m*個の固有ベクトル$\boldsymbol{x}_i$ (*i*=1, ..., *m*) が与えられたとき，正規化された固有ベクトル$\boldsymbol{y}_I$を求める．ここで，$\boldsymbol{x}_i = (x_{1i}, \ldots, x_{ni})^T$とする.

MODE=1のときは，ベクトル$\boldsymbol{x}_i$の絶対値最大の要素を1とする正規化を行う.

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_\infty, \quad \|\boldsymbol{x}_i\|_\infty = \max_k |x_{ki}| \qquad (4.1)$$

MODE=2のときは，ベクトル$\boldsymbol{x}_i$の各要素の絶対値の2乗の和を1とする正規化を行う.

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_2, \quad \|\boldsymbol{x}_i\|_2 = \sqrt{\sum_{k=1}^{n} |x_{ki}|^2} \qquad (4.2)$$

I11-41-0201 COSI, DCOSI

余弦積分 $C_i(x)$
CALL COSI (X, CI, ICON)

(1) 機能

余弦積分 $C_i(x)$ の値

$$C_i(x) = -\int_x^{\infty} \frac{\cos(t)}{t} dt$$

を近似多項式を用いて計算する．ただし $x \neq 0$であること．$x < 0$のときは，上の積分は主値をとるものとする．

(2) パラメタ

X…… **入力**．独立変数 ***x***.
CI…… **出力**．関数値 $C_i(x)$.
ICON…… **出力**．コンディションコード.
表 COSI-1参照.

表 COSI-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$\|x\| \geq t_{max}$であった.	CI = 0.0とする.
30000	x=0であった.	CI = 0.0とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, UTLIM
② FORTRAN 基本関数 ... ABS, SIN, COS, ALOG

b. 注意
① 引数Xの範囲は
$|X| < t_{max}$
であること.
$|X|$ が大きくなるとsin (x), cos (x) の値が正確に計算できなくなることに対する処理である.
(手法概要 (4.4) 参照)

c. 使用例
$C_i(x)$ の値をxが0.1から10.0まで，0.1おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,100
      X=FLOAT(K)/10.0
      CALL COSI(X,CI,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,CI
      IF(ICON.NE.0) WRITE(6,620) X,CI,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF COSINE ',
     * 'INTEGRAL FUNCTION'///6X,'X',9X,
     * 'CI(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'CI=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

余弦積分$C_i(x)$ の計算は，$|x| = 4$ を境にして，近似式の形が異なる.

$C_i(x)=C_i(-x)$ であるから，以下では$x > 0$の場合の計算法についてだけ述べる.

a. $0 < x < 4$の場合
$C_i(x)$ のべき級数展開

$$C_i(x) = \gamma + \log(x) + \sum_{n=1}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!2n} \qquad (4.1)$$

ただし，γ：オイラー定数

をそれぞれ (4.2), (4.3) の近似式を用いて計算する.

単精度： $C_i(x) = \log(x) + \sum_{k=0}^{7} a_k z^{2k}, z = x/4$ (4.2)

倍精度： $C_i(x) = \log(x) + \sum_{k=0}^{12} a_k x^{2k}$ (4.3)

b. $x \geq 4$の場合
$C_i(x)$ の漸近展開

$$C_i(x) = -\{P(x)\sin(x) + Q(x)\cos(x)\}/x \qquad (4.4)$$

より計算する．ただし$P(x)$, $Q(x)$ は次の近似式を用いる.

単精度： $P(x) = \sum_{k=0}^{11} a_k z^k, z = 4/x$ (4.5)

$$Q(x) = \sum_{k=0}^{11} b_k z^k, z = 4/x \qquad (4.6)$$

倍精度： $P(x) = \sum_{k=0}^{11} a_k z^k \Big/ \sum_{k=0}^{11} b_k z^k, z = 4/x$ (4.7)

$$Q(x) = -\sum_{k=0}^{10} c_k z^{k+1} \Big/ \sum_{k=0}^{11} d_k z^k, z = 4/x \qquad (4.8)$$

C21-15-0101 CQDR, DCQDR

複素係数2次方程式
CALL CQDR (Z0, Z1, Z2, Z,ICON)

(1) 機能

複素係数2次方程式の根を求める.

$a_0 z^2 + a_1 z + a_2 = 0 \quad (|a_0| \neq 0)$

(2) パラメタ

Z0, Zl, Z2 ……**入力**. 2次方程式の係数.
複素数型変数.
Z0 = a_0, Zl = a_1, Z2 = a_2と与える.

Z ……**出力**. 2次方程式の根.
大きさ2の複素数型1次元配列.

ICON……**出力**. コンディションコード.
表 CQDR-1参照.

表 CQDR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	$\|a_0\| = 0.0$であった.	Z (1) には$-a_2/a_1$ Z (2) は保証しない.
30000	$\|a_0\| = 0.0$かつ $\|a_1\| = 0.0$であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... CSQRT, REAL, AIMAG, ABS

b. 使用例

複素係数を入力し，根zを求める.

```
C      **EXAMPLE**
       DIMENSION Z(2)
       COMPLEX Z0,Z1,Z2,Z
       READ(5,500) Z0,Z1,Z2
       CALL CQDR(Z0,Z1,Z2,Z,ICON)
       WRITE(6,600) ICON,Z0,Z1,Z2
       IF(ICON.EQ.30000) STOP
       WRITE(6,610) (Z(I),I=1,2)
       STOP
  500 FORMAT(6F10.0)
  600 FORMAT(10X,'ICON=',I5/10X,'A=',
     * 2E15.6/(12X,2E15.6))
  610 FORMAT(10X,'Z=',2E15.6/12X,2E15.6)
       END
```

(4) 手法概要

複素係数2次方程式$a_0x^2+a_1z+a_2=0$の根は $P_1 = a_1/a_0$ $P_2 = a_2/a_0$ とすれば $z = (-P_1 \pm \sqrt{P_1^2 - 4P_2})$ で与えられる.

もし $|P_1^2| >> |4P_2|$ の場合

$$-P_1 + \sqrt{P_1^2 - 4P_2}, \quad -P_1 - \sqrt{P_1^2 - 4P_2}$$

のどちらかの計算において大きな精度の損失を招く．この問題を防止するために，根の公式で分子を有理化した式を使い計算している．それを以下に示す.

$D = P_1^2 - 4P_2$ とする.

$|D| = 0$の場合

$z_1 = -P_1/2$
$z_2 = -P_1/2$

$|D| \neq 0$の場合

P_1の実数部をx，虚数部をyとすれば,

$x > 0$の場合

$$z_1 = \left(-P_1 - \sqrt{D}\right)\Big/2$$
$$z_2 = -2P_2\Big/\left(P_1 + \sqrt{D}\right) \qquad (5.1)$$

$x < 0$の場合

$$z_1 = 2P_2\Big/\left(-P_1 + \sqrt{D}\right)$$
$$z_2 = \left(-P_1 + \sqrt{D}\right)\Big/2 \qquad (5.2)$$

$x = 0$の場合

$$\left.\begin{aligned} z_1 &= \left(-P_1 - \sqrt{D}\right)\Big/2 \\ z_2 &= -2P_2\Big/\left(P_1 + \sqrt{D}\right) \end{aligned}\right\} y \geq 0$$

$$\left.\begin{aligned} z_1 &= 2P_2\Big/\left(-P_1 + \sqrt{D}\right) \\ z_2 &= \left(-P_1 + \sqrt{D}\right)\Big/2 \end{aligned}\right\} y < 0 \qquad (5.3)$$

なお，判別式$D=P_1^2-4P_2$ の計算において$|P_1|$が大きいとき，P_1^2がオーバフローを起こすことがある.この問題を防止するために，P_1 の実数部 $>10^{35}$又はP_1 の虚数部 $>10^{35}$の判定条件を設け，この条件を満たさないときは上述の計算方式で対処し，この条件を満たすときは $\sqrt{D} = P_1\sqrt{1 - 4P_2/P_1/P_1}$ より計算する.

All-40-0201 CSBGM, DCSBGM

行列格納モードの変換 (対称バンド行列用圧縮モード→一般モード)
CALL CSBGM (ASB, N, NH, AG, K, ICON)

(1) 機能

対称バンド行列用圧縮モードで格納された$n \times n$, バンド幅 hの実対称バンド行列を一般モードに変換する.

$n > h \geq 0$であること.

(2) パラメタ

ASB ……**入力**. 対称バンド行列用圧縮モードで格納された対称バンド行列.
大きさ$n(h+1)-h(h+1)/2$の1次元配列.

N……**入力**. 行列の次数n.

NH……**入力**. 行列のバンド幅h.

AG……**出力**. 一般モードで格納された対称バンド行列.
AG (K, N) なる2次元配列.
(使用上の注意①参照)

K……**入力**. 配列AGの整合寸法 (≥ N).

ICON……**出力**. コンディションコード.
表 CSBGM-1参照.

表 CSBGM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	NH < 0, N ≤ NH又はK < Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSLII ... MGSSL

② FORTRAN 基本関数 ... なし

b. 注意

① 一般モードの対称バンド行列の格納方法
本サブルーチンにより変換された一般モードの対称バンド行列とは，下バンド部分，対角部分だけではなく，上バンド部分やその他 (零要素) も格納されている.

② 領域の節約について
配列ASB上の内容を保存する必要のない場合は，EQUIVALENCE文により

EQUIVALENCE (ASB (1), AG (1, 1))

と宣言すれば領域が節約できる．使用例参照.

c. 使用例
$n \times n$, バンド幅 hの実対称バンド行列が圧縮モードで与えられたとき，これを一般モードに移して逆行列を求める.
サブルーチンALU, LUIVを用いて求め，必要なモード変換を本サブルーチンで行う.
$n \leq 100$, $h \leq 20$の場合.

```
C      **EXAMPLE**
       DIMENSION ASB(1890),AG(100,100),
      *          VW(100),IP(100)
       EQUIVALENCE(ASB(1),AG(1,1))
   10  READ(5,500) N,NH
       IF(N.EQ.0) STOP
       NT=(NH+1)*(N+N-NH)/2
       READ(5,510) (ASB(I),I=1,NT)
       K=100
       CALL CSBGM(ASB,N,NH,AG,K,ICON)
       WRITE(6,600) ICON
       IF(ICON.EQ.30000) GO TO 10
       WRITE(6,610) N,NH,((I,J,AG(I,J),
      *             J=1,N),I=1,N)
       EPSZ=0.0
       CALL ALU(AG,K,N,EPSZ,IP,IS,VW,ICON)
       WRITE(6,620) ICON
       IF(ICON.GE.20000) GO TO 10
       CALL LUIV(AG,K,N,IP,ICON)
       WRITE(6,630) ICON
       WRITE(6,640) N,NH,((I,J,AG(I,J),
      *             J=1,N),I=1,N)
       GO TO 10
  500  FORMAT(2I5)
  510  FORMAT(4E15.7)
  600  FORMAT('1'/10X,'CSBGM ICON=',I5)
  610  FORMAT(//10X,'** INPUT MATRIX **'/
     *10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     *(2X,4('(',I3,',',I3,')',E17.8)))
  620  FORMAT(/10X,'ALU ICON=',I5)
  630  FORMAT(/10X,'LUIV ICON=',I5)
  640  FORMAT('1'//10X,'** INVERSE ',
     *'MATRIX **'/10X,'ORDER=',I5,5X,
     *'BANDWIDTH=',I5/(2X,4('(',I3,',',I3,
     *')',E17.8)))
       END
```

(4) 手法概要

1次元配列ASB上に対称バンド行列用圧縮モードで格納された実対称バンド行列を次に示す手順により，2次元配列上に一般モードで格納する.

a. ASBの後から順に，AGの対角部分と上三角部分とに転送する．転送は，n列目から列ごとに行う．対応関係は次のとおりである.

対称バンド行列用圧縮モード上の要素	行列要素	一般モード上の要素
$\mathrm{ASB}(hJ-h(h+1)/2+I)$	$\rightarrow a_{ij}$	$\rightarrow \mathrm{AG(I,J)}$
$,I=J,J-1,...,J-h,J=N,N-1,...h+2$		
$\mathrm{ASB}(J(J-1)/2+I)$	$\rightarrow a_{ij}$	$\rightarrow \mathrm{AG(I,J)}$
$,I=J,J-1,...,1$	$,J=h+1,h,...,1$	

b. 対角部分を対称軸として，上三角部分を下三角部分に，AG (I, J) = AG(J, I) となるように転送する．ただし，I>Jとする.

All-50-0201 CSBSM, DCSBSM

行列格納モードの変換 (対称バンド行列用圧縮モード →対称行列用圧縮モード)
CALL CSBSM (ASB, N, NH, AS, ICON)

(1) 機能

対称バンド行列用圧縮モードで格納された$n \times n$, バンド幅hの対称バンド行列を対称行列用圧縮モードに変換する．$n>h\geq 0$であること.

(2) パラメタ

ASB ……**入力**．対称バンド行列用圧縮モードで格納された対称バンド行列.
大きさ$n(h+1)-h(h+1)/2$の1次元配列.

N ……**入力**．行列の次数 n.

NH ……**入力**．行列のバンド幅 h.

AS ……**出力**．対称行列用圧縮モードで格納された対称バンド行列.
大きさ$n(n+1)/2$の1次元配列.
(使用上の注意①参照).

ICON ……**出力**．コンディションコード.
表 CSBSM- 1参照.

表 CSBSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	NH < 0 又は NH ≥ Nであった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ...MGSSL

② FORTRAN 基本関数 ... なし.

b. 注意

① 領域の節約について
配列ASB上の内容を保存する必要のない場合は，EQUIVALENCE文により

EQUIVALANCE (ASB (1), AS (1))

と宣言すれば領域が節約できる．使用例参照.

c. 使用例
$n \times n$, バンド幅 hの正値対称バンド行列が対称バンド行列用圧縮モードで与えられたとき，これを対称行列用圧縮モードに移して逆行列を求める.
本サブルーチンによりモード変換を行い，SLDL, LDIVを用いて逆行列を求める.
$n \leq 100$, $h \leq 20$の場合.

```
C     **EXAMPLE**
      DIMENSION AS(5050),ASB(1890)
      EQUIVALENCE(AS(1),ASB(1))
   10 READ(5,500) N,NH
      IF(N.EQ.0) STOP
      NS=(N+1)*N/2
      NT=(NH+1)*(N+N-NH)/2
      READ(5,510) (ASB(I),I=1,NT)
      CALL CSBSM(ASB,N,NH,AS,ICON)
      WRITE(6,600) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,610) N,NH,(I,AS(I),I=1,NS)
      EPSZ=0.0
      CALL SLDL(AS,N,EPSZ,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL LDIV(AS,N,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,640) N,NH,(I,AS(I),I=1,NS)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'/10X,'CSBSM ICON=',I5)
  610 FORMAT(//10X,'** INPUT MATRIX **'/
     *10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     *(2X,5('(',I4,')',E17.8)))
  620 FORMAT(/10X,'SLDL ICON=',I5)
  630 FORMAT(/10X,'LDIV ICON=',I5)
  640 FORMAT('1'//10X,'** INVERSE ',
     * 'MATRIX **'/
     * 10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     * (2X,5('(',I4,')',E17.8)))
      END
```

(4) 手法概要

1次元配列ASB上に対称バンド行列用圧縮モードで格納された実対称バンド行列を，以下のとおり1次元配列AS上に対称行列用圧縮モードで格納する．必要な行列要素は，行ごとに転送され．バンド部分の外側へは零要素が補われる．補われる零要素を除く各要素の対応関係は次のとおりである.

対称バンド行列用圧縮モード上の要素	行列要素	対称行列用圧縮モード上の要素

$$\mathrm{ASB}\left(\mathrm{I}(\mathrm{I}-1)/2+\mathrm{J}\right)\rightarrow a_{ij} \qquad \rightarrow \mathrm{AS}(\mathrm{I}(\mathrm{I}-1)/2+\mathrm{J})$$
$$,\mathrm{J}=1,2,...,\mathrm{I}=1,2,...,h+1$$
$$\mathrm{ASB}\left(\mathrm{I}\cdot h+\mathrm{J}-h(h+1)/2\right)\rightarrow a_{ij} \rightarrow \mathrm{AS}(\mathrm{I}(\mathrm{I}-1)/2+\mathrm{J}))$$
$$,\mathrm{J}=\mathrm{I},\mathrm{I}-1...,\mathrm{I}-h \quad ,\mathrm{I}=\mathrm{N},\mathrm{N}+1,...h+2$$

A11-10-0201 CSGM, DCSGM

行列格納モードの変換 (対称行列用圧縮モード→一般モード)
CALL CSGM (AS, N, AG, K, ICON)

(1) 機能

対称行列用圧縮モードで格納された $n \times n$の実対称行列を一般モードに変換する．$n \geq 1$であること．

(2) パラメタ

AS…… **入力**．対称行列用圧縮モードで格納された対称行列．
大きさ $n(n+1)/2$の1次元配列．

N…… **入力**．行列の次数 n．

AG…… **出力**．一般モードで格納された対称行列．
AG (K,N) なる2次元配列．
(使用上の注意①参照)．

K…… **入力**．配列AGの整合寸法 (≥ N)．

ICON…… **出力**．コンディションコード．
表 CSGM-1参照．

表 CSGM-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	N < 1 又は K < Nであった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... なし．

b. 注意

① 一般モードの対称行列の格納方法
本サブルーチンにより変換された一般モードの対称行列とは，下三角部分，対角部分だけではなく，上三角部分も格納されている．

② 領域の節約について
配列AS上の内容を保存する必要のない場合は，EQUIVALENCE文により

EQUIVALENCE (AS(1), AG(1,1))

と宣言すれば領域が節約できる．使用例参照．

c. 使用例
$n \times n$の実対称行列$\boldsymbol{A}$が，圧縮モードで与えられたとき，その逆行列を求める．
サブルーチンALU, LUIVを用いて求め，必要なモード変換を，本サブルーチンで行う．
$n \leq 100$の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),B(100,100),
     *          VW(100),IP(100)
      EQUIVALENCE (A(1),B(1,1))
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      K=100
      CALL CSGM(A,N,B,K,ICON)
      WRITE(6,600) ICON
      IF(ICON.EQ.30000) GOTO 10
      WRITE(6,610) N,((I,J,B(I,J),
     *             J=1,N),I=1,N)
      EPSZ=0.0
      CALL ALU(B,K,N,EPSZ,IP,IS,VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GOTO 10
      CALL LUIV(B,K,N,IP,ICON)
      WRITE(6,630) ICON
      WRITE(6,640) N,((I,J,B(I,J),
     *             J=1,N),I=1,N)
      GOTO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'/10X,'CSGM ICON=',I5)
  610 FORMAT(//10X,'** INPUT MATRIX **'/
     *10X,'ORDER=',I5/
     *(2X,4('(',I3,',',I3,')',E17.8)))
  620 FORMAT(/10X,'ALU ICON=',I5)
  630 FORMAT(/10X,'LUIV ICON=',I5)
  640 FORMAT('1'//10X,'** INVERSE ',
     *'MATRIX **'/10X,'ORDER=',I5/
     *(2X,4('(',I3,',',I3,')',E17.8)))
      END
```

(4) 手法概要

1次元配列AS上に対称行列用圧縮モードで格納された実対称行列を，次に示す手順により，2次元配列上に一般モードで格納する．

a. ASの後から順に，AGの対角部分と上三角部分とに転送する．転送は，n列目から列ごとに行う．
NT = $n(n+1)/2$としたとき，以下に対応関係を示す．

圧縮モード上の要素		行列要素		一般モード上の要素
AS(NT)	→	a_{nn}	→	AG(N,N)
AS(NT-1)	→	a_{nn-1}	→	AG(N-1,N)
.		.		.
.		.		.
.		.		.
AS(I(I-1)/2+J)	→	a_{ij}	→	AG(J,I)
.		.		.
.		.		.
.		.		.
AS(2)	→	a_{21}	→	AG(1,2)
AS(1)	→	a_{11}	→	AG(1,1)

b. 対角部分を対称軸として，上三角部分を下三角部分に，AG(I,J) = AG(J,I) となるように転送する．ただし，I>Jとする．

A11-50-0101 CSSBM, DCSSBM

行列格納モードの変換 (対称行列用圧縮モード →対称バンド行列用圧縮モード)
CALL CSSBM (AS, N, ASB, NH, ICON)

(1) 機能

対称行列用圧縮モードで格納された $n \times n$，バンド幅 hの実対称バンド行列を，対称バンド行列用圧縮モードに変換する．$n \times h \geq 1$であること．

(2) パラメタ

AS……**入力**．対称行列用圧縮モードで格納された対称バンド行列．
大きさ$n\,(n+1)/2$の1次元配列．

N……**入力**．行列の次数 n.

ASB……**出力**．対称バンド行列用圧縮モードで格納された対称バンド行列．
大きさ$n\,(h+1) - h\,(h+1)/2$の1次元配列．

NH……**入力**．行列のバンド幅 h.

ICON……**出力**．コンディションコード．
表 CSSBM-1参照．

表 CSSBM-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	NH < 0 又は NH ≤ Nであった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... なし．

b. 注意
① 領域の節約について
配列AS上の内容を保存する必要のない場合は，EQUIVALENCE文により

EQUIVALENCE (AS (1), ASB (1))

と宣言すれば領域が節約できる．使用例参照．

c. 使用例
$n \times n$，バンド幅 hの正値対称バンド行列が対称行列用圧縮モードで与えられたとき，これを対称バンド行列用圧縮モードに移してLDL^T分解する．
サブルーチンBDLを用いて分解を行い，必要なモード変換を本サブルーチン並びにサブルーチンCSBSMで行う．$n \leq 100, h \leq 20$の場合．

```
C     **EXAMPLE**
      DIMENSION AS(5050),ASB(1890)
      EQUIVALENCE(AS(1),ASB(1))
   10 READ(5,500) N,NH
      IF(N.EQ.0) STOP
      NT=(N+1)*N/2
      READ(5,510) (AS(I),I=1,NT)
      WRITE(6,600) N,NH,(I,AS(I),I=1,NT)
      CALL CSSBM(AS,N,ASB,NH,ICON)
      WRITE(6,610) ICON
      IF(ICON.EQ.30000) GO TO 10
      EPSZ=0.0
      CALL SBDL(ASB,N,NH,EPSZ,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL CSBSM(ASB,N,NH,AS,ICON)
      WRITE(6,630) N,NH,(I,AS(I),I=1,NT)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT(//10X,'** INPUT MATRIX **'/
     *10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     *(2X,5('(',I4,')',E17.8)))
  610 FORMAT('1'//10X,'CSSBM ICON=',I5)
  620 FORMAT(10X,'SBDL ICON=',I5)
  630 FORMAT('1'/10X,'DECOMPOSED MATRIX'/
     *10X,'ORDER=',I5,5X,'BANDWIDTH=',I5/
     *(2X,5('(',I4,')',E17.8)))
      END
```

(4) 手法概要

1次元配列AS上に対称行列用圧縮モードで格納されたバンド幅 hの実対称バンド行列を，以下のとおり1次元配列ASB上に対称バンド行列用圧縮モードで格納する．必要な行列要素は，第1行から行ごとに転送される．各要素の対応関係は次のとおりである．

対称行列用圧縮モード上の要素　行列要素　対称バンド行列用圧縮モード上の要素

$$\mathrm{AS}\left(\mathrm{I}(\mathrm{I}-1)/2+\mathrm{J}\right) \rightarrow a_{ij} \rightarrow \mathrm{ASB}\left(\mathrm{I}(\mathrm{I}-1)/2+\mathrm{J}\right)$$
$$,\mathrm{J}=1,2,...,\mathrm{I} \quad ,\mathrm{I}=1,2,...,h+1$$

$$\mathrm{AS}\left(\mathrm{I}(\mathrm{I}-1)/2+\mathrm{J}\right) \rightarrow a_{ij} \rightarrow \mathrm{ASB}\left(\mathrm{I}\cdot h+\mathrm{J}-h(h+1)/2\right)$$
$$,\mathrm{J}=\mathrm{I}-h,\mathrm{I}-h+1,...,\mathrm{I}$$
$$,\mathrm{I}=h+2,h+3,...,\mathrm{N}$$

C23-15-0101 CTSDM, DCTSDM

複素超越方程式 $f(z) = 0$ (マラー法)
CALL CTSDM (Z, ZFUN, ISW, EPS, ETA, M, ICON)

(1) 機能

与えられた初期値をもとに複素超越方程式,

$$f(z) = 0$$

の1根をマラー (Muller) 法により求める.

(2) パラメタ

Z ……**入力**. 求めようとする根の初期値.
出力. 近似根.

ZFUN …**入力**. 解こうとする方程式の関数 $f(z)$ を計算する複素数型関数副プログラム名.

ISW ……**入力**. 制御情報.
求めようとする根の収束判定法を指定する.
ISW= 1,2, 3のいずれかであること.
ISW = 1のとき,
$|f(z_j)| \le$ EPS : 収束判定法I
を満たすz_jを根とする.
ISW=2のとき,
$|z_j - z_{j-1}| \le$ ETA $\cdot |z_j|$: 収束判定法II
を満たすz_jを根とする.
ISW=3のとき,
上の二つの収束判定方を同時に採用し少くとも一方が満たされたとき，z_jを根とする.
(使用上の注意参照)

EPS……**入力.** 収束判定法I (パラメタISWの項参照) で用いられる収束判定値 (≥ 0.0).

ETA……**入力.** 収束判定法II (パラメタISWの項参照) で用いられる収束判定値 (≥ 0.0).

M ……**入力**. 求めようとする根の反復回数の上限 (> 0).
(使用上の注意参照)
出力. 実際に反復した回数.

ICON…**出力**. コンディションコード.
表 CTSDM-1参照.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMAX, AFMIN, AMACH, MGSSL
② FORTRAN 基本関数 ... CABS, CMPLX, EXP, ALOG, CSQRT, SQRT, REAL, AIMAG

表 CTSDM-1 コンディションコード

コード	意味	処理内容
1	求まった近似根が収束判定法 I (パラメタISWの項参照) を満たした.	正常終了
2	求まった近似根が収束判定法II (パラメタISWの項参照) を満たした.	正常終了
10	反復の回数は無条件に m回 (M = -m) 繰り返した.	正常終了
11	反復の回数は無条件に m回繰り返すと指定されたが (M = -m), その回数の反復以前に $\|f(z_j)\| = 0.0$ となったので反復を止めて，z_j を根とした.	正常終了
12	反復の回数は無条件に m回繰り返すと指定されたが (M = -m), その回数の反復以前に $\|z_j - z_{j-1}\| \le u \cdot \|z_j\|$ となったので反復を止めて，z_j を根とした.	正常終了
10000	与えられた反復回数内で指定された収束条件を満たさなかった.	Zには最後の反復値が格納される.
20000	計算の過程で，マラー法では反復を続行できない事態が起きた. (手法概要参照)	処理を打ち切る.
30000	入力パラメタにエラーがあった.すなわち，M>0のとき, ① ISW=1かつEPS<0 又は ② ISW=2かつETA<0 又は ③ ISW=3かつEPS<0又はETA<0 のいずれであるか，又は M = 0 又はISW≠1, 2, 3であった.	処理を打ち切る.

b. 注意

① 本サブルーチンを呼び出す側のプログラム内で引数ZFUNに相当する複素数型関数副プログラムに対してEXTERNAL文で宣言すること.

② 本サブルーチンは，ISW = 1と与えられたときでも，反復の過程で
$|z_j - z_{j-1}| \le u|z_j|$, u は丸め誤差の単位
が満たされたときは反復を止め，ICON=2と出力する．同様に，ISW=2と与えられたときでも，反復の過程で $|f(z_j)| = 0.0$ となったときは反復を止め，ICON=1と出力する.

③ 反復の回数MをM = -m (m > 0) と与えると反復は無条件に m回繰り返す.
ただし，注意②と同様に，反復された近似根が収束判定法I (ISWの項参照) の左辺で0.0となった場合，又は収束判定法II (ISWの項参照) の左辺で丸め誤差の単位uよりも小さくなった場合は反復を止める.

c. 使用例
$f(z) = e^z - i$ の1根を初期値 $z_0 = 0$として求める.

```
C      **EXAMPLE**
       COMPLEX Z,ZFEXP
       EXTERNAL ZFEXP
       Z=CMPLX(0.0,0.0)
       ISW=3
       EPS=0.0
       ETA=1.0E-6
       M=100
       CALL CTSDM(Z,ZFEXP,ISW,EPS,ETA,M,
      *ICON)
       WRITE(6,600) ICON,M,Z
       STOP
  600 FORMAT(10X,'ICON=',I5/13X,'M=',I5/
     *       13X,'Z=',2E15.7)
       END
       FUNCTION ZFEXP(Z)
       COMPLEX Z,ZFEXP
       ZFEXP=CEXP(Z)-CMPLX(0.0,1.0)
       RETURN
       END
```

(4) 手法概要

本サブルーチンはマラー (Muller) 法を用いている．マラー法は求めようとする根の三つの近似値を用いて，$f(z)$を2次の補間多項式$P(z)$で近似し，$P(z)$ =0の根の一方を$f(z)$ の次の近似根として採用することにより，次々と反復的に根を近似していくものである.

本アルゴリズムの特徴として，次のことが挙げられる.

a. 微係数は不要である.
b. 反復の過程での関数の評価は，その反復ごとに1回で済む.
c. 収束の速さは1.84次である(ただし単根の場合).

マラー法

$f(z)$ の求めようとする根αの三つの近似値を z_{i-2}, z_{i-1}, z_iとする (出発値 z_1, z_2, z_3については後述).

この3点を用いて$f(z)$ を2次のニュートン補間多項式で近似する．それを$P(z)$ とすると次のようになる.

$$P(z) = f_1 + f[z_i, z_{i-1}](z - z_i) + f[z_i, z_{i-1}, z_{i-2}](z - z_i)(z - z_{i-1}) \tag{4.1}$$

ただし，$f_i = f(z_i)$ であり，$f[z_i, z_{i-1}]$,$f[z_i, z_{i-1}, z_{i-2}]$は，それぞれ1階，2階の差分商 (devided difference) で,

$$f[z_i, z_{i-1}] = \frac{f_i - f_{i-1}}{z_i - z_{i-1}}$$
$$f[z_i, z_{i-1}, z_{i-2}] = \frac{f[z_i, z_{i-1}] - f[z_{i-1}, z_{i-2}]}{z_i - z_{i-2}} \tag{4.2}$$

である.

ここで，2次方程式$P(z) = 0$を解くと，その2根は次のようになる.

$$z = z_i - \frac{2f_i}{\omega \pm \left\{\omega^2 - 4f_i f[z_i, z_{i-1}, z_{i-2}]\right\}^{1/2}}$$
$$\omega = f[z_i, z_{i-1}] + (z_i - z_{i-1}) f[z_i, z_{i-1}, z_{i-2}] \tag{4.3}$$

ここで，この (4.3) 式の2根のうち，第2項の分母の絶対値の大きい方の根を次の反復値 z_{i+1}とする．これは，z_{i+1}として，z_iに近い方の根を採用することを意味する.

(4.1) 式で，z^2の項が0の場合，すなわち，$f[z_i, z_{i-1}, z_{i-2}] = 0$の場合は，(4.3)式を用いず,

$$z = z_i - \frac{f_i}{f[z_i, z_{i-1}]} = z_i - \frac{z_i - z_{i-1}}{f_i - f_{i-1}} f_i \tag{4.4}$$

を用いる．これはセカント (secant) 法である.

また (4.2) 式で，z^2, zの項が共に0の場合は,

$P(z)$ = 定数

となり，本アルゴリズムを用いることが不可能となる (この場合については後述).

アルゴリズムの考慮

a. 出発値 z_1, z_2, z_3
z_1, z_2, z_3は次のようにして決定する.
利用者が入力パラメタZに与えた初期値をzとすると,

$|z| \neq 0$のとき,
$$\begin{cases} z_1 = 0.9z \\ z_2 = 1.1z \\ z_3 = z \end{cases}$$

$|z| \neq 0$のとき,
$$\begin{cases} z_1 = -1.0 \\ z_2 = 1.0 \\ z_3 = 0.0 \end{cases}$$
とする.

b. $f(z_{i-2}) = f(z_{i-1}) = f(z_i)$ の場合の処置
この場合は (4.1) 式の z^2, z の項が0となった場合で，マラー法ではこれ以降の処理は不可能である．本サブルーチンでは z_{i-2}, z_{i-1}, z_i に摂動を与えて，$f(z_{i-2}) = f(z_{i-1}) = f(z_i)$ の状態を避けるよう次のような試みを行っている.
すなわち,

$z'_{i-2} = (1 + p^n) z_{i-2}$
$z'_{i-1} = (1 + p^n) z_{i-1}$
$z'_i = (1 + p^n) z_i$

ただし，$p = -u^{-1/10}$, u は丸め誤差の単位.

nは摂動を与えた回数.

として，z'_{i-2}, z'_{i-1}, z'_i についてマラー法を続行する．ここで，この摂動が5回以上行われた場合は本サブルーチンではこれ以上の続行をあきらめ，ICON =20000として処理を打ち切る.

収束判定法

次の二つの収束判定法を用いている.

収束判定法I

近似根 z_iが $|f(z_i)| \leq$ EPSを満たしたとき，z_iを根として採用する.

収束判定法II

近似根 z_iが $|z_i - z_{i-1}| \leq$ ETA$\cdot |z_i|$ を満たしたとき，z_iを根として採用する.

この判定法におけるETAは，求めようとする根が多重根であったり，近接根である場合は大きめにとる必要がある．なお，ETA < u (uは丸め誤差の単位) であったときは，サブルーチン内部でETA = uと置き換えている.

なお，詳細については，参考文献 [32] を参照すること.

B5l-30-0201 ECHEB, DECHEB

チェビシェフ級数の求和
CALL ECHEB (A, B, C, N, V, F, ICON)

(1) 機能

区間 [*a*, *b*] で定義された *n* 項のチェビシェフ級数

$$f(x)=\sum_{k=0}^{n-1}{}'c_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \qquad (1.1)$$

が与えられたとき，区間内の任意の点 *v* における級数の値 *f*(*v*) を求める．

ここで，Σ’ は初項を 1/2 倍にして和をとることを意味する．

ただし，$a\neq b$, $v\in[a,b]$, $n\geq 1$ であること．

(2) パラメタ

A …… **入力**．チェビシェフ級数の定義域の下限 *a*.
B …… **入力**．チェビシェフ級数の定義域の上限 *b*.
C …… **入力**．係数 $\{c_k\}$.
　　大きさ *n* の1次元配列.
　　$C(1)=c_0$, $C(2)=c_1$, ..., $C(N)=c_{n-1}$
　　のように格納する.
N …… **入力**．級数の項数 *n*.
V …… **入力**．区間 [*a*, *b*] 内の点 *v*.
F …… **出力**．点 *v* における級数の値 *f*(*v*).
ICON … **出力**．コンディションコード
　　表 ECHEB-1 参照.

表 ECHEB-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	次のいずれかであった. ① N<1 ② A = B ③ $v\notin$ [a, b]	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... なし.

b. 注意
① 本サブルーチンは，チェビシェフ級数の値 *f*(*v*) を求める．任意の滑らかな関数 *f*(*x*) をチェビシェフ級数展開する場合は，サブルーチンFCHEBを用いればよい.

c. 使用例
区間 [0, π] で，正弦関数

$$f(x)=\sin x\left(=\sum_{n=0}^{\infty}(-1)^n\frac{x^{2n+1}}{(2n+1)!}\right)$$

をサブルーチン FCHEB により級数展開する．（要求精度：絶対誤差 $5\cdot 10^{-5}$）
次に，求めた展開係数を用いて，区間 [0, π] 上の32個のチェビシェフ点

$$x_j=\frac{\pi}{2}+\frac{\pi}{2}\cos\frac{j}{32}\pi\left(=\pi\cos^2\frac{j}{64}\pi\right)$$
$$,j=0,1,\ldots,31$$

の上で級数の値を本サブルーチンにより求める．同時に誤差も計算し，それらを印字する.

```
C     **EXAMPLE**
      DIMENSION C(257),TAB(255)
      EXTERNAL FUN
      EPSA=5.0E-5
      EPSR=0.0
      NMIN=9
      NMAX=257
      A=0.0
      B=ATAN(1.0)*4.0
      CALL FCHEB(A,B,FUN,EPSA,EPSR,NMIN,
     *           NMAX,C,N,ERR,TAB,ICON)
      IF(ICON.NE.0) GOTO 20
      WRITE(6,600) N,ERR,ICON
      WRITE(6,601) (C(K),K=1,N)
      WRITE(6,602)
      X=A
      NS=32
      H=ATAN(1.0)*2.0/FLOAT(NS)
      DO 10 J=1,NS
      X=B*COS(H*FLOAT(J-1))**2
      CALL ECHEB(A,B,C,N,X,Y,ICON)
      IF(ICON.NE.0) GOTO 20
      ERROR=FUN(X)-Y
      WRITE(6,603) X,Y,ERROR
   10 CONTINUE
      STOP
   20 WRITE(6,604) ICON
      STOP
  600 FORMAT('0',3X,'EXPANSION OF',
     * ' FUNCTION FUN(X)',3X,'N=',I4,3X,
     * 'ERROR=',E13.3,3X,'ICON=',I6)
  601 FORMAT (/(5E15.5))
  602 FORMAT ('0',10X,'X',7X,'EVALUATION',
     * 8X,'ERROR'/)
  603 FORMAT(1X,3E15.5)
  604 FORMAT('0',5X,'CONDITION CODE',I8)
      END
```

```
      FUNCTION FUN(X)
      REAL*8 SUM,XP,TERM
      EPS=AMACH(EPS)
      SUM=X
      P=X*X
      XP=X*P
      XN=-6.0
      N=3
   10 TERM=XP/XN
      SUM=SUM+TERM
      IF(DABS(TERM).LE.EPS) GO TO 20
      N=N+2
      XP=XP*P
      XN=-XN*FLOAT(N)*FLOAT(N-1)
      GOTO 10
   20 FUN=SUM
      RETURN
      END
```

(4)　手法概要

n 項のチェビシェフ級数 (1.1) の値を，後退漸化式により求める．

後退漸化式

チェビシェフ級数

$$f(x)=\sum_{k=0}^{n-1}{}'c_k T_k(x)$$

の，$x = v$ ($v\in$ [-1,1]) における値 $f(v)$ は，次のように補助的な数列 $\{b_k\}$ を生成することにより，効率よく計算することができる．

今，$\{b_k\}$ について，

$$b_{n+2}=b_{n+1}=0$$
$$b_k=2vb_{k+1}-b_{k+2}+c_k,\ k=n,n-1,\ldots 0 \qquad (4.1)$$

を定義すると，級数の値 $f(v)$ は，

$$f(v)=\left(b_0-b_2\right)/2 \qquad (4.2)$$

により求まる．

本サブルーチンでは，まず変数 $v\in[a,b]$ を変数$s\in$[-1,1] に変換する．

$$s=\frac{2v-(b+a)}{b-a} \qquad (4.3)$$

その後，(4.1)，(4.2) を適用し $f(v)$ を求める．n 項のチェビシェフ級数の値を求めるに必要な乗算回数は，約 n 回である．

E51-10-0201 ECOSP, DECOSP

cosine 級数の求和
CALL ECOSP (TH, A, N, V, F, ICON)

(1) 機能

周期 $2T$ をもつ n 項の cosine 級数

$$f(t)=\frac{1}{2}a_0+\sum_{k=1}^{n-1}a_k\cos\frac{\pi}{T}kt \tag{1.1}$$

が与えられたとき，任意の点 v における級数の値$f(v)$を求める．

ただし，$T>0$, $n\geq 1$とする．

(2) パラメタ

TH ……**入力**．cosine級数の半周期 T.

A ……**入力**．係数 $\{a_k\}$.
大きさ N の1次元配列.
A(1) = a_0, A(2) = a_1, ..., A(N) = a_{n-1}
のように格納する．

N ……**入力**．級数の項数 n.

V ……**入力**．点 v.

F ……**出力**．点 v における級数の値 $f(v)$.

ICON ……**出力**．コンディションコード
表 ECOSP-1 参照.

表 ECOSP-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	次のいずれかであった． ① N<1 ② TH ≤ 0	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... COS, ATAN

b. 注意

① 本サブルーチンは，cosine 級数の値 $f(v)$ を求める．任意の周期 $2T$をもつ滑らかな偶関数 $f(t)$ を cosine 級数展開する場合は，サブールチン FCOSFを用いればよい．

c. 使用例

助変数 ω を含む奇関数の積分

$$F(x)=\int_0^x\frac{\sin\omega t}{\sqrt{1+\sin^2\omega t}}dt \tag{3.1}$$
$$,\ \omega=\frac{\pi}{4},\frac{\pi}{2},\pi,2\pi,4\pi$$

の値を，$x=h$, $2h$, ..., $10h$ と x を動かしながら求める．ただし $h=\pi/(10\omega)$.

まず被積分関数は，周期 $2\pi/\omega$ をもつ奇関数であるので，これをサブルーチン FSINF によって，要求精度 $\varepsilon_a=\varepsilon_r=5\cdot10^{-5}$ の下で sine 級数

$$f(t)\approx\sum_{k=0}^{n-1}b_k\sin k\omega t \tag{3.2}$$

に展開し，次にこれを項別積分して，

$$F(x)=\sum_{k=0}^{n-1}{}'a_k\cos k\omega x,$$
$$a_k=-\frac{b_k}{k\omega},\quad k=1,2,\ldots,n-1 \tag{3.3}$$

を得る．ただし初項の a_0 だけは $F(0)=0$ となるように定める．

そこで $x=h$, $2h$, ..., $10h$ と動かしながらサブルーチン ECOSP を呼び，不定積分 (3.1) の近似値を計算する．

この解析解は，

$$F(x)=\frac{1}{\omega}\left(\frac{\pi}{4}-\sin^{-1}\left(\cos\omega x/\sqrt{2}\right)\right)$$

であるので，本サブルーチンによる結果と比較する．

```
C       **EXAMPLE**
        DIMENSION A(257),TAB(127)
        EXTERNAL FUN
        COMMON W,PI
        PI=ATAN(1.0)*4.0
        EPSA=0.5E-04
        EPSR=EPSA
        NMIN=0
        NMAX=257
        W=PI*0.25
        DO 1 I=1,5
        TH=PI/W
C       EXPANSION OF INTEGRAND
        CALL FSINF(TH,FUN,EPSA,EPSR,NMIN,
     *             NMAX,A,N,ERR,TAB,ICON)
        IF(ICON.GT.10000) GO TO 10
        WRITE(6,600) N,ERR,ICON,W
        WRITE(6,601) (A(K),K=1,N)
C       TERMWISE INTEGRATION
        DO 2 K=2,N
        A(K)=-A(K)/(FLOAT(K-1)*W)
    2 CONTINUE
C       EVALUATION OF COSINE SERIES
        CALL ECOSP(TH,A,N,0.0,P,ICON)
        IF(ICON.NE.0) GO TO 10
        A(1)=-P*2.0
        WRITE(6,610)
        H=TH*0.1
```

```
    DO 3 K=1,10
    X=H*FLOAT(K)
    CALL ECOSP(TH,A,N,X,P,ICON)
    IF(ICON.NE.0) GO TO 10
    Q=TRFN(X)
    WRITE(6,611) X,P,Q
  3 CONTINUE
    W=W+W
  1 CONTINUE
    STOP
 10 WRITE(6,620) ICON
    STOP
600 FORMAT('0',5X,'EXPANSION OF',
   *' INTEGRAND',3X,'N=',I4,5X,'ERR=',
   *E15.5,5X,'ICON=',I5,5X,'W=',E15.5)
601 FORMAT(/(5E15.5))
610 FORMAT('0',5X,'EVALUATION OF',
   *' COSINE SERIES'/'0',6X,
   *'AUGUMENT',7X,'COMPUTED',11X,'TRUE')
611 FORMAT(3E15.5)
620 FORMAT('0',5X,'CONDITION CODE',I6)
    END
    FUNCTION FUN(T)
    COMMON W
    X=W*T
    P=SIN(X)
    FUN=P/SQRT(P*P+1.0)
    RETURN
    END
    FUNCTION TRFN(T)
    COMMON W,PI
    TRFN=(PI*.25-ASIN(COS(W*T)*
   *     SQRT(0.5)))/W
    RETURN
    END
```

(4) 手法概要

n 項のcosine 級数 (1.1) の値を後退漸化式により求める.

変数 t を

$$\theta = \frac{\pi}{T} t$$

と変換すれば, $f(t) = g(\theta)$ は, θ に関し周期2π の cosine 級数となる. そこで点 $\theta = \frac{\pi}{T} v$ における$g(\theta)$ の値を求めればよい. これは, 次の後退漸化式によって効率よく計算できる. すなわち

$$\begin{aligned} & b_{n+2} = b_{n+1} = 0 \\ & b_k = 2\cos\theta \cdot b_{k+1} - b_{k+2} + a_k \\ & k = n, n-1, \ldots, 1, 0 \end{aligned} \tag{4.1}$$

をつくるとき, 級数の値は

$$f(v) = g(\theta) = \left(b_0 - b_1\right)/2 \tag{4.2}$$

により求まる.

n 項の cosine 級数の求和のために必要な乗算回数は約 n 回, cosine関数の評価が1回である.

B21-11-0101 EIG1, DEIG1

実行列の固有値及び固有ベクトル（2段 QR 法）
CALL EIG1 (A, K, N, MODE, ER, EI, EV, VW, ICON)

(1) 機能

n 次の実行列 ***A*** の固有値と対応する固有ベクトルを求める．固有ベクトルは $\|\boldsymbol{x}\|_2 = 1$ となるように正規化される．

$n \geq 1$ であること．

(2) パラメタ

A …… **入力**．実行例 ***A***.
A (K, N) なる2次元配列.
演算後内容は保存されない.

K …… **入力**．配列 A 及び EV の整合寸法 ($\geq n$).

N …… **入力**．実行列 ***A*** の次数 *n*.

MODE … **入力**．平衡化の省略を指定.
MODE=1 のときは平衡化を省略する.
MODE≠1 のときは平衡化を行う.

ER,EI …… **出力**．固有値.
固有値は，実部と虚部に分けられ，実部は ER に，虚部は EI に格納される．第 *j* 番目の固有値が複素固有値であれば，第 *j*+1 番目の固有値は，第 *j* 番目の固有値と共役な固有値である．(図 EIG1-1 参照).
大きさ *n* の1次元配列.

EV …… **出力**．固有ベクトル.
固有ベクトルは，固有値と対応する列に格納される.
ただし，固有値が複素固有値のときは，固有ベクトルも複素ベクトルとなるので，図 EIG1-1 のように格納する.
詳細は「(3) 使用上の注意 ①」を参照のこと.
EV (K,N) なる2 次元配列.

VW …… **作業領域**．大きさ *n* の1次元配列.

ICON …… **出力**．コンディションコード.
表 EIG1-1 参照

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, BLNC, IRADIX, HES1, MGSSL

② FORTRAN 基本関数 ... ABS, SQRT, SIGN, DSQRT.

表 EIG1-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	N = 1であった.	ER (1) = A (1, 1), EV (1,1) = 1.0 とする.
20000	三角行列に変換できず，固有値・固有ベクトルは求められなかった.	処理を打ち切る.
30000	N < 1 又は K < N であった.	処理を打ち切る.

b. 注意

① 複素固有値と対応する固有ベクトルについて.
一般に実行列の固有値には，実固有値と複素固有値が存在し，複素固有値は共役な複素固有値と対になっている.
本サブルーチンでは，ある *j* 番目の固有値 (λ_j) が複素固有値のとき，λ_j 及び λ^*_j を ER 及び EI に，次のように格納する.

λ_j = ER (J) + *i*·EI (J) (*i*は虚数単位)
λ_j^* = ER (J + 1) + *i*·EI (J+1)
= ER (J) – *i*·EI (J)

一方，λ_j に対応する固有ベクトル($\boldsymbol{x}_j$) は，

$$\boldsymbol{x}_j = \boldsymbol{u}_j + i\boldsymbol{v}_j$$

なる複素ベクトルとなり，λ_j^* に対応する固有ベクトル $\boldsymbol{x}_j^*$ は $\boldsymbol{x}_j^* = \boldsymbol{u}_j + i\boldsymbol{v}_j$ となる.
したがって，$\boldsymbol{x}_j$ の実部 ($\boldsymbol{u}_j$) と虚部 ($\boldsymbol{v}_j$) をそれぞれ保存しておけば，$\boldsymbol{x}_j^*$ は簡単に求まる．したがって本ルーチンでは，λ_j に対応する固有ベクトル $\boldsymbol{x}_j$ だけを求め，λ_j^* に対応する固有ベクトル $\boldsymbol{x}_j^*$ は計算していない.
なお，$\boldsymbol{x}_j$ は，その実部 ($\boldsymbol{u}_j$) を EV の第 J 列に，また，虚部 ($\boldsymbol{v}_j$) を第 J + 1 列に格納する.
図 EIG1-1 参照のこと.

② 平衡化について
実行列の各要素の大きさが，オーダの意味で大きく異なっているような場合には，一般に行列の平衡化によって結果の精度を上げることができる．このため，本サブルーチンでは，サブルーチンBLNCにより平衡化を行うようにしている.
他方，もし行列の各要素のオーダがほぼ揃っているような場合には，行列の平衡化による効果を期待できない．このような場合には，
MODE = 1 とすることにより，平衡化を省略できる.

③ 本サブルーチンは，実行列の固有値及び固有ベクトルを全体として求めるためのものである.
実行列の固有値だけ必要な場合は，サブルーチン BLNC, HES1, HSQR により求めること.

また，実行列の固有ベクトルの一部だけ必要な場合は，サブルーチン BLNC, HES1, HSQR HVEC, HBK1 により求めること.

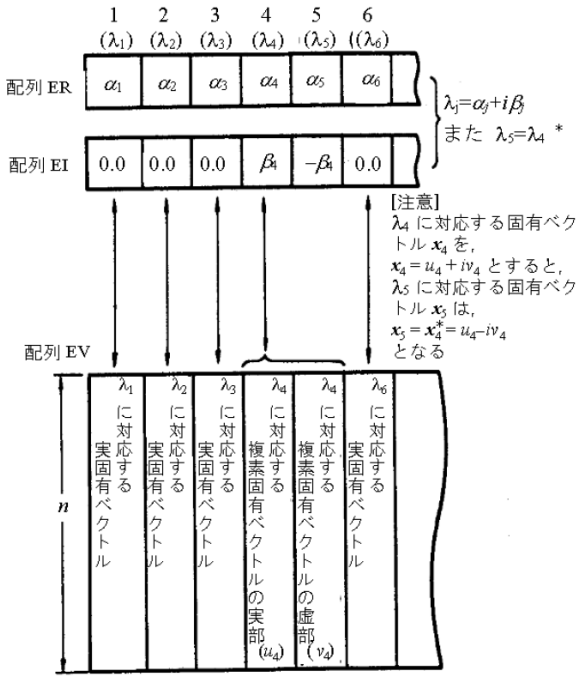


図 EIG1-1 固有値と固有ベクトルの対応

c. 使用例

n 次の実行列 ***A*** の固有値及び固有ベクトルを求める．$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),ER(100),
     * EI(100),EV(100,100),VW(100),
     * IND(100)
   10 CONTINUE
      READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610)(I,J,A(I,J),J=1,N)
      CALL EIG1(A,100,N,0,ER,EI,EV,VW,
     *          ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      IND(1)=0
      CALL EPRT(ER,EI,EV,IND,100,N,N)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('0',5X,'ORIGINAL MATRIX', 5X,
     * 'N=',I3)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E15.7))
  620 FORMAT('0',5X,'ICON=',I5)
      END
```

本使用例中のサブルーチン EPRT は，実行列の固有値及び固有ベクトルを出力するサブルーチンである．以下にその内容を示す.

```
      SUBROUTINE EPRT(ER,EI,EV,IND,K,N,M)
      DIMENSION ER(M),EI(M),EV(K,M),IND(M)
      WRITE(6,600)
      IF(IABS(IND(1)).EQ.1) GO TO 40
      J=0
      DO 30 I=1,M
      IF(J.EQ.0) GO TO 10
      IND(I)=0
      J=0
      GO TO 30
   10 IF(EI(I).NE.0.0) GO TO 20
      IND(I)=1
      GO TO 30
   20 IND(I)=-1
      J=1
   30 CONTINUE
   40 MM=0
      DO 50 I=1,M
      IF(IND(I).EQ.0) GO TO 50
      MM=MM+1
      ER(MM)=ER(I)
      IND(MM)=IND(I)
      IF(EI(I).EQ.0.0) GO TO 50
      MM=MM+1
      ER(MM)=EI(I)
      IND(MM)=IND(I+1)
   50 CONTINUE
      KAI=(MM-1)/5+1
      LST=0
      DO 70 L=1, KAI
      INT=LST+1
      LST=LST+5
      IF(LST.GT.MM) LST=MM
      WRITE(6,610) (J,J=INT,LST)
      WRITE(6,620) (ER(J),J=INT,LST)
      WRITE(6,630) (IND(J),J=INT,LST)
      DO 60 I=1,N
      WRITE(6,640) I,(EV(I,J),J=INT,LST)
   60 CONTINUE
   70 CONTINUE
      RETURN
  600 FORMAT('1',10X,'**EIGENVECTORS**')
  610 FORMAT('0',5I20)
  620 FORMAT('0',1X,'EIGENVALUE',5E20.8)
  630 FORMAT(2X,'IND',6X,I10,4I20)
  640 FORMAT(6X,I5,5E20.8)
      END
```

(4) 手法概要

n 次の実行列 ***A*** の固有値と対応する固有ベクトルを求める.

n 次の実行列の固有値は，次の2段階の変換を行うことにより，上三角行列 ***R*** の対角要素として求める.

① ハウスホルダー法による実行列 ***A*** の実ヘッセンベルグ行列 ***H*** への変換.

この操作を (4.1) で表す.

$$\boldsymbol{H} = \boldsymbol{Q}_H^T \boldsymbol{A} \boldsymbol{Q}_H \tag{4.1}$$

ここに $\boldsymbol{Q}_H$ は，ハウスホルダー法で用いる変換行列 $\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots, \boldsymbol{P}_{n-2}$ の積

$$\boldsymbol{Q}_H = \boldsymbol{P}_1 \cdot \boldsymbol{P}_2 \cdot \ldots \cdot \boldsymbol{P}_{n-2} \tag{4.2}$$

で与えられる直交行列である.
ハウスホルダー法の詳細については，サブルーチン HESl を参照すること.

② 2 段 QR 法による実ヘッセンベルグ行列 $\boldsymbol{H}$ から上三角行列 $\boldsymbol{R}$ への変換.
この操作を (4.3) で表す.

$$\boldsymbol{R} = \boldsymbol{Q}_R^T \boldsymbol{H} \boldsymbol{Q}_R \tag{4.3}$$

ここに $\boldsymbol{Q}_R$ は，2 段階 QR 法で用いる変換行列 $\boldsymbol{Q}_1, \boldsymbol{Q}_2, ..., \boldsymbol{Q}_s$ の積

$$\boldsymbol{Q}_R = \boldsymbol{Q}_1 \cdot \boldsymbol{Q}_2 \cdot \ldots \cdot \boldsymbol{Q}_s \tag{4.4}$$

で与えられる直交行列である.
2 段階 QR 法の詳細については，サブルーチン HSQR を参照すること.

一方，固有ベクトルは，(4.5) の相似変換により上三角行列 $\boldsymbol{R}$ を対角行列 $\boldsymbol{D}$ に変換する行列 $\boldsymbol{F}$ が存在すれば，(4.6) により得られる行列 $\boldsymbol{X}$の列ベクトルとして求めることができる.

$$\boldsymbol{D} = \boldsymbol{F}^{-1} \boldsymbol{R} \boldsymbol{F} \tag{4.5}$$
$$\boldsymbol{X} = \boldsymbol{Q}_H \boldsymbol{Q}_R \boldsymbol{F} \tag{4.6}$$

(4.6) で与えられる行列 $\boldsymbol{X}$ の列ベクトルが，実行列 $\boldsymbol{A}$ の固有ベクトルとなることは，(4.5) に (4.1) 及び (4.3) を代入すれば (4.7) が得られることにより確かめられる.

$$\boldsymbol{D} = \boldsymbol{X}^{-1} \boldsymbol{A} \boldsymbol{X} = \boldsymbol{F}^{-1} \boldsymbol{Q}_R^T \boldsymbol{Q}_H^T A \boldsymbol{Q}_H \boldsymbol{Q}_R \boldsymbol{F} \tag{4.7}$$

ここで，$\boldsymbol{Q}_H \boldsymbol{Q}_R$ を $\boldsymbol{Q}$ で表せば，(4.2) 及び(4.4) より，

$$\boldsymbol{Q} = \boldsymbol{P}_1 \cdot \boldsymbol{P}_2 \cdot \ldots \cdot \boldsymbol{P}_{n-2} \cdot \boldsymbol{Q}_1 \cdot \boldsymbol{Q}_2 \cdot \ldots \cdot \boldsymbol{Q}_s \tag{4.8}$$

となる.

(4.8) より $\boldsymbol{Q}$ の計算は，変換の進行につれて逐次変換行列の積を作ることにより実行できることがわかる.

(4.8) により得られた $\boldsymbol{Q}$ と行列 $\boldsymbol{F}$ の積を作れば，$\boldsymbol{X}$ が得られる.

上述の $\boldsymbol{F}$ は上三角行列として求めることができる．すなわち，(4.5) より

$$\boldsymbol{F}\boldsymbol{D} = \boldsymbol{R}\boldsymbol{F} \tag{4.9}$$

ここで，$\boldsymbol{D}$, $\boldsymbol{R}$, $\boldsymbol{F}$ の要素をそれぞれ $\boldsymbol{D}$ = diag(λ_i), $\boldsymbol{R}$ = (r_{ij}), $\boldsymbol{F}$=(f_{ij}) とすると，f_{ij} は (4.9) より $j = n, n-1, ..., 2$ に対して (4.10) で求められる.

$$f_{ij} = \sum_{k=i+1}^{j} r_{ik} f_{kj} \Big/ (\lambda_j - \lambda_i) \quad (i = j-1, j-2, \ldots, 1) \tag{4.10}$$

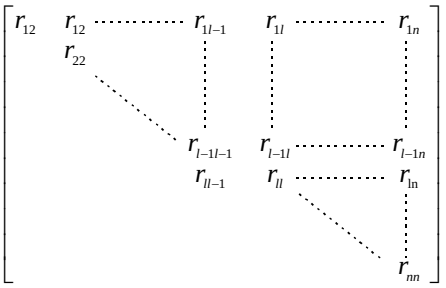


図 EIG1-2 (λ_{l-1}, λ_l) が複素数となるときの $\boldsymbol{R}$ の形

ただし，$r_{ij} = 0$ ($i>j$), $r_{ii} = \lambda_i$
$f_{ij} = 0$ ($i>j$), $f_{ii} = 1$

もし，$\lambda_i = \lambda_j$ のときは (4.11) で計算する.

$$f_{ij} = \sum_{k=i+1}^{j} r_{ik} f_{kj} \Big/ u \|\boldsymbol{R}\|_\infty \tag{4.11}$$

ここに u は丸め誤差の単位である.

さて以上は固有値がすべて実数の場合であるが，$\boldsymbol{A}$ が複素固有値を持つ場合，$\boldsymbol{R}$ は厳密な意味で上三角行列とはならない．複素固有値 λ_{l-1} , $\lambda_l (= \lambda^*_{l-1})$ を持つときの $\boldsymbol{R}$ の形を図EIG1-2 に示す．この場合，f_{ij} の求め方は少し複雑になる．今，f_{ij} を (4.10) より求める過程において，複素数である λ_i ($i = l$) に遭遇した場合，f_{lj} , f_{l-1j} に対しては次の連立方程式を解いて求める.

$$\left.\begin{aligned} (r_{l-1l-1} - \lambda_j) f_{l-1j} + r_{l-1l} f_{lj} &= -\sum_{k=l+1}^{j} r_{l-1k} f_{kj} \\ r_{ll-1} f_{l-1j} + (r_{ll} - \lambda_j) f_{lj} &= -\sum_{k=l+1}^{j} r_{lk} f_{kj} \end{aligned}\right\} \tag{4.12}$$

また，$\boldsymbol{F}$ の第 l 列及び第 l–1 列については両者とも複素ベクトルになるが，λ_l と λ_{l-1} が互いに共役関係にあるので，それら複素ベクトルも互いに共役である．それで本サブルーチンでは λ_{l-1}に対応する f_{il-1} ($i=l,l-1,\ldots,1$) だけを求め f_{il} は求めていない.

f_{il-1} は次のようにして求める．すなわち，i=l, l–1 に対しては (4.13) より求め，他 (i=l–2, l–3, ...1) は，(4.10) より求める.

$$f_{ll-1} = 1$$
$$f_{l-1l-1} = \begin{cases} -r_{l-1l} \big/ (r_{l-1l-1} - \lambda_{l-1}) & (r_{ll-1} \le r_{l-1l}) \\ -(r_{ll} - \lambda_{l-1}) \big/ r_{ll-1} & (r_{ll-1} > r_{l-1l}) \end{cases} \tag{4.13}$$

なお，(4.10) を適用している時点で，λ_i が更に複素数（すなわち λ_j , λ_i とも複素数）の場合，それに該当する $f_{il\text{-}1}$ については (4.12) を用いる．以上のようにして得られた $f_{il\text{-}1}$ の実部は $\boldsymbol{F}$ の 第 l–1 列，虚部は第 l 列に格納される．

このようにして求めた $\boldsymbol{F}$ と (4.8) より得られた $\boldsymbol{Q}$ から，最終的に (4.14) により$\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ が $\boldsymbol{X}$ の列ベクトルとして求まる．

$$\boldsymbol{X} = \boldsymbol{QF} \tag{4.14}$$

この $\boldsymbol{X}$ は $\|\boldsymbol{x}\|_2 = 1$ となるように正規化されている．

なお，本サブルーチンでは，実行列のヘッセンベルグ行列への変換に先立ち，サブルーチン BLNC により行列の平衡化を行っている．ただし，MODE=1 となっているときは平衡化を省略する．

なお，詳細については，参考文献 [12] 及び [13] pp 372-395を参照すること．

E51-20-0201 ESINP, DESINP

sine 級数の求和
CALL ESINP (TH, B, N, V, F, ICON)

(1) 機能

周期 $2T$ をもつ n 項の sine 級数

$$f(t)=\sum_{k=0}^{n-1} b_k \sin\frac{\pi}{T}kt, \quad b_0=0 \tag{1.1}$$

が与えられたとき，任意の点 v における級数の値 $f(v)$ を求める．

ただし，$T>0$，$n\geq 1$ とする．

(2) パラメタ

TH ····· **入力**．sine 級数の半周期 T．

B ····· **入力**．係数 $\{b_k\}$．
大きさ N の 1 次元配列．
B(1)=0.0, B(2)=b_1, ..., B(N)=b_{n-1}
のように格納する．

N ····· **入力**．級数の項数 n．

V ····· **入力**．点v．

F ····· **出力**．点 v における級数の値 $f(v)$．

ICON ····· **出力**．コンディションコード
表 ESINP-1 参照．

表 ESINP-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	次のいずれかであった． ① N < 1 ② TH ≤ 0	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... COS, ATAN, SIN

b. 注意

① 本サブルーチンは，sine 級数の値 $f(v)$ を求める．任意の周期 $2T$ をもつ滑らかな奇関数 $f(t)$ を sine 級数展開する場合は，サブルーチン FSINF を用いればよい．

c. 使用例

助変数 ω を含む偶関数の積分

$$F(x)=\int_0^x \frac{\cos\omega t}{\sqrt{1+\cos^2\omega t}}dt$$
$$,\omega=\frac{\pi}{4},\frac{\pi}{2},\pi,2\pi,4\pi \tag{3.1}$$

の値を，$x=h, 2h, ..., 10h$ とx を動かしながら求める．ただし，$h=\pi/(10\omega)$.

まず被積分関数は，周期 $2\pi/\omega$ をもつ偶関数であるので，これをサブルーチン FCOSF によって，要求精度 $\varepsilon_a=\varepsilon_r=10^{-5}$ の 下で cosine 級数

$$f(t)\approx\frac{1}{2}a_0+\sum_{k=1}^{n-1}a_k\cos k\omega t \tag{3.2}$$

に展開し，次にこれを項別積分して，

$$F(x)=\sum_{k=0}^{n-1}b_k\sin k\omega x,$$
$$b_0=0, b_k=\frac{a_k}{k\omega}, k=1,2,\ldots,n-1 \tag{3.3}$$

を得る．

そこで，$x=h, 2h, ..., 10h$ と動かしながらサブルーチン ESINP を呼び，不定積分 (3.1) の近似値を計算する．

この解析解は，

$$F(x)=\frac{1}{\omega}\sin^{-1}\left(\frac{\sin\omega x}{\sqrt{2}}\right)$$

であるので，本サブルーチンによる結果と比較する．

```
C     **EXAMPLE**
      DIMENSION A(257),TAB(127)
      EXTERNAL FUN
      COMMON W
      PI=ATAN(1.0)*4.0
      EPSA=0.5E-04
      EPSR=EPSA
      NMIN=0
      NMAX=257
      W=PI*0.25
      DO 1 I=1,5
      TH=PI/W
C     EXPANSION OF INTEGRAND
      CALL FCOSF(TH,FUN,EPSA,EPSR,NMIN,
     *           NMAX,A,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GO TO 10
      WRITE(6,600) N,ERR,ICON,W
      WRITE(6,601) (A(K),K=1,N)
C     TERMWISE INTEGRATION
      DO 2 K=2,N
      A(K)=A(K)/(FLOAT(K-1)*W)
    2 CONTINUE
C     EVALUATION OF SINE SERIES
      WRITE(6,610)
      H=TH*0.1
      DO 3 K=1,10
      X=H*FLOAT(K)
      CALL ESINP(TH,A,N,X,P,ICON)
      IF(ICON.NE.0) GO TO 10
      Q=TRFN(X)
      WRITE(6,611) X,P,Q
    3 CONTINUE
      W=W+W
    1 CONTINUE
      STOP
   10 WRITE(6,620) ICON
      STOP
```

```
600 FORMAT('0',5X,'EXPANSION OF',
   *' INTEGRAND',3X,'N=',I4,5X,'ERR=',
   *E15.5,5X,'ICON=',I5,5X,'W=',E15.5)
601 FORMAT(/(5E15.5))
610 FORMAT('0',5X,'EVALUATION OF',
   *' SINE SERIES'/'0',6X,
   *'AUGUMENT',7X,'COMPUTED',11X,'TRUE')
611 FORMAT(3E15.5)
620 FORMAT('0',5X,'CONDITION CODE',I6)
    END
    FUNCTION FUN(T)
    COMMON W
    X=W*T
    P=COS(X)
    FUN=P/SQRT(P*P+1.0)
    RETURN
    END
    FUNCTION TRFN(T)
    COMMON W
    TRFN=ASIN(SIN(W*T)*SQRT(0.5))/W
    RETURN
    END
```

(4) 手法概要

n 項の sine 級数 (1.1) の値を後退漸化式により求める.

変数 t を

$$\theta = \frac{\pi}{T} t$$

と変換すれば，$f(t)=g(\theta)$ は，θ に関し周期 2π の sine 級数となる．そこで $\theta = \frac{\pi}{T} v$ における $g(\theta)$ の値を求めればよい．これは，次の後退漸化式によって効率よく計算できる．すなわち,

$$\begin{aligned} & c_{n+2} = c_{n+1} = 0 \\ & c_k = 2\cos\theta \cdot c_{k+1} - c_{k+2} + b_k \\ & k = n, n-1, \ldots, 1 \end{aligned} \tag{4.1}$$

をつくるとき，級数の値は,

$$f(v) = g(\theta) = c_1 \sin\theta \tag{4.2}$$

により求まる.

n 項の sine 級数の求和のために必要な乗算回数は約 n 回，cosine関数，sine関数の評価がそれぞれ1回である.

I11-31-0101 EXPI, DEXPI

指数積分 $E_i(x)$, $\overline{E}_i(x)$
CALL EXPI (X, EI, ICON)

(1) 機能

次のように定義される指数積分 $E_i(x)$，$\overline{E}_i(x)$ の値を，近似式を用いて計算する．

$x < 0$の場合：

$$E_i(x) = -\int_{-x}^{\infty} \frac{e^{-t}}{t} dt = \int_{-\infty}^{x} \frac{e^{t}}{t} dt$$

$x > 0$の場合：

$$\overline{E}_i(x) = \text{P.V.}\int_{\infty}^{-x} \frac{e^{-t}}{t} dt = \text{P.V.}\int_{-\infty}^{x} \frac{e^{t}}{t} dt$$

ここに，P.V. は t=0 で主値をとって積分することを意味する．ただし，$x \neq 0$ であること．

(2) パラメタ

X …… **入力**．独立変数 x.

EI …… **出力**．関数値 $E_i(x)$ 又は $\overline{E}_i(x)$.

ICON …… **出力**．コンディションコード．
表 EXPI-1 参照.

表 EXPI-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
20000	X > log(fl_{max}) であった. X < - log(fl_{max}) であった.	EI = fl_{max} とする. EI = 0.0 とする.
30000	X=0 であった.	EI = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AFMAX, MGSSL, ULMAX

② FORTRAN 基本関数 ... EXP, ALOG, ABS

b. 注意

① 引数 X の範囲

(a) $|X| \leq \log(fl_{max})$ であること.
$|X|$ の値が上記の範囲を超えると，$E_i(x)$，$\overline{E}_i(x)$ はそれぞれ，e^x の計算においてアンダフロー，オーバフローをおこす．これを本サブルーチン内で前もって防ぐためである.

(b) $X \neq 0$ であること.
$E_i(x)$，$\overline{E}_i(x)$ は x=0 では定義されていない.

c. 使用例

$\overline{E}_i(x)$ の値を x が 0.01 から 0.50 まで，0.01 おきに計算して数表を作成する.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,50
      X=FLOAT(K)/100.0
      CALL EXPI(X,EI,ICON)
      IF(ICON.EQ.0) WRITE(6,610)X,EI
      IF(ICON.NE.0) WRITE(6,620)X,EI,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF ',
     * 'EXPONENTIAL INTEGRAL FUNCTION'///
     * 6X,'X',9X,'EI(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'EI=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

指数積分 $E_i(x)$ 及び $\overline{E}_i(x)$ の計算は x の区間 [-174.673,-4), [-4,-1), [-1, 0), (0, 6], (6, 12], (12, 24], (24, 174.673] により近似式の形が異なる．以下では，$s=|x|$, $t=1/|x|$ とする.

a. $-\log(fl_{max}) \leq x < -4$ の場合,

単精度： $$E_i(x) = -te^x\left(1+\sum_{k=0}^{3} a_k t^{k+1} \Big/ \sum_{k=0}^{3} b_k t^k\right) \quad (4.1)$$

理論精度 9.09 桁

倍精度： $$E_i(x) = -te^x\left(1+\sum_{k=0}^{8} a_k t^{k+1} \Big/ \sum_{k=0}^{8} b_k t^k\right) \quad (4.2)$$

理論精度 18.88 桁

b. $-4 \leq x < -1$ の場合,

単精度： $$E_i(x) = -e^x \sum_{k=0}^{4} a_k t^k \Big/ \sum_{k=0}^{4} b_k t^k \quad (4.3)$$

理論精度 10.04 桁

倍精度： $$E_i(x) = -e^x \sum_{k=0}^{8} a_k t^k \Big/ \sum_{k=0}^{8} b_k t^k \quad (4.4)$$

理論精度 19.20 桁

c. $-1 \leq x < 0$ の場合,

単精度： $$E_i(x) = \log(s) - \sum_{k=0}^{3} a_k s^k \Big/ \sum_{k=0}^{3} b_k s^k \quad (4.5)$$

理論精度 10.86 桁

倍精度： $$E_i(x) = \log(s) - \sum_{k=0}^{5} a_k s^k \Big/ \sum_{k=0}^{5} b_k s^k \quad (4.6)$$

理論精度 18.54 桁

d. $0 < x \le 6$ の場合,
単精度：

$$\overline{E}_i(x) = \log(x/x_0) + (x - x_0)\sum_{k=0}^{5} a_k z^k \Big/ \sum_{k=0}^{5} b_k z^k \tag{4.7}$$

理論精度 9.94 桁
ただし $z = x/6$, x_0=0.37250 7410

倍精度：

$$\overline{E}_i(x) = \log(x/x_0) + (x - x_0)\sum_{k=0}^{9} a_k z^k \Big/ \sum_{k=0}^{9} b_k z^k \tag{4.8}$$

理論精度 20.76 桁
ただし $z = x/6$,
x_0= 0.37250 7410 81366 63446

e. $6 < x \le 12$ の場合,
単精度：

$$\overline{E}_i(x) = \frac{e^x}{x}\left\{a_0 + \frac{b_0}{a_1 + x} + \frac{b_1}{a_2 + x} + \frac{b_8}{a_3 + x} + \frac{b_3}{a_4 + x}\right\} \tag{4.9}$$

理論精度 8.77 桁

倍精度：

$$\overline{E}_i(x) = \frac{e^x}{x}\left\{a_0 + \frac{b_0}{a_1 + x} + \frac{b_1}{a_2 + x} + \ldots + \frac{b_8}{a_9 + x}\right\} \tag{4.10}$$

理論精度 19.21 桁

f. $12 < x \le 24$ の場合,
単精度：

$$\overline{E}_i(x) = \frac{e^x}{x}\left\{a_0 + \frac{b_0}{a_1 + x} + \frac{b_1}{a_2 + x} + \frac{b_2}{a_3 + x} + \frac{b_3}{a_4 + x}\right\} \tag{4.11}$$

理論精度 9.45 桁

倍精度：

$$\overline{E}_i(x) = \frac{e^x}{x}\left\{a_0 + \frac{b_0}{a_1 + x} + \frac{b_1}{a_2 + x} + \ldots + \frac{b_8}{a_9 + x}\right\} \tag{4.12}$$

理論精度 19.22 桁

g. $24 < x \le \log(fl_{max})$ の場合,
単精度：

$$\overline{E}_i(x) = \frac{e^x}{x}\left\{1 + \frac{1}{x}\left(a_0 + \frac{b_0}{a_1 + x} + \frac{b_1}{a_2 + x}\right)\right\} \tag{4.13}$$

理論精度 8.96 桁

倍精度：

$$\overline{E}_i(x) = \frac{e^x}{x}\left\{1 + \frac{1}{x}\left(a_0 + \frac{b_0}{a_1 + x} + \ldots + \frac{b_8}{a_9 + x}\right)\right\} \tag{4.14}$$

理論精度 18.11 桁

なお，詳細については，参考文献 [79] , [80] を参照のこと.

E51-30-0101 FCHEB, DFCHEB

実関数のチェビシェフ級数展開 （関数入力，高速cosine変換）
CALL FCHEB (A, B, FUN, EPSA, EPSR, NMIN, NMAX, C, N, ERR, TAB, ICON)

(1) 機能

区間 [*a*, *b*] で滑らかな関数 $f(x)$と，要求精度 ε_a，ε_r が与えられたとき，$f(x)$ をチェビシェフ級数展開し，

$$\left| f(x) - \sum_{k=0}^{n-1}{}' c_k T_k\left(\frac{2x-(b+a)}{b-a}\right)\right| \le \max\left\{\varepsilon_a, \varepsilon_r \cdot \|f(x)\|\right\} \tag{1.1}$$

を満たすn個の係数 $\{c_k\}$ を求める．

ここで，Σ' は初項を1/2倍して和をとることを意味する．また，$\|f\|$は，区間 [*a*, *b*] での関数の標本点

$$x_j = \frac{a+b}{2} + \frac{b-a}{2}\cos\frac{\pi}{n-1}j, \quad j = 0,1,...,n-1 \tag{1.2}$$

における関数値を用いて，

$$\|f\| = \max_{0\le j\le n-1}\left|f(x_j)\right| \tag{1.3}$$

と定義する．

ただし，$a \neq b$，$\varepsilon_a \ge 0$，$\varepsilon_r \ge 0$ であること．

(2) パラメタ

A …… **入力**．関数 $f(x)$ の定義域の下限 *a*.

B…… **入力**．関数 $f(x)$ の定義域の上限 *b*.

FUN …… **入力**．展開しようとする関数 $f(x)$ を計算する関数副プログラム名（使用例参照）．

EPSA…… **入力**．級数展開の絶対誤差の上限 ε_a.

EPSR…… **入力**．級数展開の相対誤差の上限 ε_r.

NMIN…… **入力**．級数展開の項数 *n* の下限（≥0）．(2のべき) + 1 の値で与える．標準値は9.
（使用上の注意④参照）

NMAX … **入力**．級数の項数 *n* の上限（≥NMIN）．(2のべき) + 1 の値で与える．標準値は257.
（使用上の注意④参照）

C…… **出力**．係数 $\{c_k\}$.
大きさ NMAX の 1 次元配列.
C(1) = c_0. C(2) = c_1, ..., C(N) = c_{n-1}
のように格納される．

N…… **出力**．級数展開の項数 *n*（≥5）．(2のべき) + 1 の値をとる．

ERR …… **出力**．級数展開の絶対誤差の推定値.

TAB …… **出力**．級数展開で使用された三角関数表が格納される．
1 次元配列．大きさは 3 以上でかつ，
a ≠ 0 のとき　(NMAX-3)/2
a = 0 のとき　NMAX-2
（使用上の注意②，③参照）

ICON …… **出力**．コンディションコード．
表 FCHEB-1 参照.

表 FCHEB-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	丸め誤差のために，要求精度が得られなかった．要求精度が厳しすぎる．	C には，得られた係数を出力する．級数展開の誤差は達成可能な限度まで達している．
20000	級数展開の項数が，その上限に達しても要求精度が得られなかった．	処理を打ち切る． C には，そのときまでに得られた係数を出力する．また，ERR には，この時点での絶対誤差の推定値を出力する．
30000	次のいずれかであった． ①A = B ②EPSA < 0.0 ③EPSR < 0.0 ④NMIN < 0 ⑤NMAX < NMIN	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, AMACH, UTABT, UCOSM

② FORTRAN 基本関数 ... ABS, AMAX1, AMIN1, FLOAT

b. 注意

① パラメタ FUN に対応する関数副プログラムは，独立変数 *x* だけを引数に持つ関数副プログラムとして定義し，本サブルーチンを呼び出す側のプログラムで，その関数名を EXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図る．

② 三角関数表の扱いについて
本サブルーチンを繰り返し呼ぶ場合，必要な三角関数表は，1 回だけしか計算されない．すなわち，三角関数表に不足が生じた場合，その都度不足分だけ計算し追加される．
したがって，パラメタ TAB の内容は，常に保存したまま呼び出すこと．

③ 本サブルーチンでは通常，区間 [*a*, *b*] を [-1, 1] に変数変換し，$f(x)$ をチェビシェフ多項式系により展開する．しかし，区間の端点 *a* が零の場合は，変数変換での計算誤差（桁落ち）の発生を防止するために，$f(x)$ をずらしチェビシェフ多項式系により展開している．ただし，求める係数 c_k は，同一である．
その為，内部で使用する三角関数表は，前者の場合(NMAX-3)/2，後者の場合 NMAX-2 となっている．

④ NMIN，NMAX が (2のべき) + 1 の値でない場合，それらを超えない最大の (2のべき) + 1 の値とみなす．ただし，NMAX < 5 の場合，NMAX = 5 とみなす．

⑤ 展開の項数 *n* の増大とともに，誤差が小さくなる度合は，関数 $f(x)$ の滑らかさと区間 [*a*, *b*] の巾の大きさに著しく影響される．$f(x)$ が解析関数ならば，誤差は指数関数的オーダ $O(r^n)$，$0 < r < 1$ で減少し，$f(x)$ が *k* 回連続微分可能ならば，有理式のオーダ $O\left(\left|\frac{a-b}{n}\right|^k\right)$ で減少する．*k* = 0, 1の場合，展開項数が多くなり，誤差の推定も正確さを期しがたい．したがって，本サブルーチンを適用する関数は，少なくとも 2 回連続微分可能であることが望ましい．

⑥ 級数の精度
本サブルーチンは，与えられた ε_a, ε_r に対して (1.1) を満たす級数を求める．したがって，ε_r = 0 とおけば絶対誤差 ε_a 以内の，また ε_a = 0 とおけば相対誤差 ε_r 以内の精度で級数展開をすることになる．しかし，関数の性質と ε_a, ε_r の値によってはこの目的が達成されないこともある．例えば，関数の計算精度に比べて，ε_a, ε_r が小さすぎる場合には展開の項数がその上限に達しない場合でも，丸め誤差の影響の方が大きくなり，これ以上計算を続行しても無意味である．このような状態を見出したときは ICON を 10000 にして処理を打ち切る．このときの級数の精度は，使用計算機で達成可能な限度まで達している．また，展開の項数が NMAX 以内では収束しないこともあり，この場合 ICON を 20000 にして処理を打ち切る．このときの係数の値は，それまでに得られている近似値であり，級数の精度は保証されない．
なお，本サブルーチンではいずれの場合にも，級数の絶対誤差の推定値をパラメタ ERR に出力するので精度の目安にすることができる．

⑦ チェビシェフ級数展開の逆変換を行う場合，サブルーチン FCOST による cosine 変換を用いればよい．
ここで逆変換とは，(1.2) で示される区間 [*a*, *b*] 上の標本点 x_j における関数値

$$f(x_j)=\sum_{k=0}^{n-1}{}''c_k T_k\left(\frac{2x_j(b+a)}{b-a}\right),\quad j=0,1,\ldots,n-1$$

を求めることに相当する．ただし，Σ" は初項と末項を 1/2 倍して和をとることを意味する．したがって，サブルーチン FCOST を呼ぶ前に，末項の係数 c_{n-1} だけは 2 倍しておく必要がある．
この逆変換を行う場合，本サブルーチンにより得られた三角関数表はサブルーチン FCOST においても使用されるので，パラメタ TAB の内容は保存しておくこと．
（使用例②参照）

c. 使用例

① 区間 [-2,2] 上で，指数関数

$$f(x)=e^x\left(=\sum_{n=0}^{\infty}\frac{x^n}{n!}\right)$$

を，要求精度 $\varepsilon_a = 0$，$\varepsilon_r = 5\cdot10^{-5}$ 及び NMIN = 9，NMAX = 257 の下で，チェビシェフ多項式系 $\{T_k(x/2)\}$ で展開する．

```
C     EXAMPLE 1
      DIMENSION C (257), TAB (127)
      EXTERNAL FUN
      EPSA=0.0
      EPSR=5.0E-5
      NMIN=9
      NMAX=257
      A=-2.0
      B=2.0
      CALL FCHEB(A,B,FUN,EPSA,EPSR,NMIN,
     *           NMAX,C,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GOTO 10
      WRITE(6,600) N,ERR,ICON
      WRITE(6,601) (C(I),I=1,N)
      STOP
   10 WRITE(6, 602) ICON
      STOP
  600 FORMAT('0',5X,'EXPANSION OF',
     * ' FUNCTION FUN(X)',3X,'N=',I4,
     * 5X,'ERR=',E15.5,5X,'ICON=',I5)
  601 FORMAT(/(5E15.5))
  602 FORMAT('0',5X,'CONDITION CODE',I8)
      END
      FUNCTION FUN(X)
      REAL*8 SUM,XP,TERM
      EPS=AMACH(EPS)
      SUM=1.0
```

```
      XP=X
      XN=1.0
      N=1
   10 TERM=XP/XN
      SUM=SUM+TERM
      IF(DABS(TERM).LE.
     *   DABS(SUM)*EPS) GOTO 20
      N=N+1
      XP=XP*X
      XN=XN*FLOAT(N)
      GOTO 10
   20 FUN=SUM
      RETURN
      END
```

② チェビシェフ級数展開の逆変換
区間 [0, π] 上で，正弦関数

$$f(x)=\sin x\left(=\sum_{n=0}^{\infty}(-1)^n\frac{x^{2n+1}}{(2n+1)!}\right)$$

を，要求精度 $\varepsilon_a = 5\cdot10^{-5}$，$\varepsilon_r = 0$ 及び NMIN = 9，NMAX = 513 の下で，チェビシェフ多項式系 $\{T_k(2x/\pi\text{ -}1)\}$ で展開する．
その後，求めた展開係数を用いて，n 個の点

$$x_j=\pi/2+\pi/2\cos\frac{\pi}{n-1}j,\ j=0,1,...,n-1$$

におけるチェビシェフ級数の値を，サブルーチン FCOST により確認する．

```
C     EXAMPLE 2
      DIMENSION C(513),TAB(511)
      EXTERNAL FUN
      EPSA=5.0E-5
      EPSR=0.0
      NMIN=9
      NMAX=513
      A=0.0
      B=ATAN(1.0)*4.0
      CALL FCHEB(A,B,FUN,EPSA,EPSR,NMIN,
     *           NMAX,C,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GOTO 10
      WRITE(6,600) N,ERR,ICON
      C(N)=C(N)*2.0
      NM1=N-1
      WRITE(6,601) (C(I),I=1,N)
      CALL FCOST(C,NM1,TAB,ICON)
      WRITE(6,602)
      WRITE(6,601) (C(I),I=1,N)
      STOP
   10 WRITE(6,603) ICON
      STOP
  600 FORMAT('0',5X,'EXPANSION OF',
     * ' FUNCTION FUN(X)',3X,'N=',I4,
     * 5X,'ERR=',E15.5,5X,'ICON=',I5)
  601 FORMAT(/(5E15.5))
  602 FORMAT('0',5X,'INVERSE TRANS',
     * 'FORM BY ''FCOST''')
  603 FORMAT('0',5X,'CONDITION CODE',I8)
      END
```

```
      FUNCTION FUN(X)
      REAL*8 SUM,XP,TERM
      EPS=AMACH(EPS)
      SUM=X
      P=X*X
      XP=X*P
      XN=-6.0
      N=3
   10 TERM=XP/XN
      SUM=SUM+TERM
      IF(DABS(TERM).LE.EPS) GOTO 20
      N=N+2
      XP=XP*P
      XN=-XN*FLOAT(N)*FLOAT(N-1)
      GOTO 10
   20 FUN=SUM
      RETURN
      END
```

(4) 手法概要

本方法はクレンショウ・カーチスのチェビシェフ級数展開の方法を，離散型 cosine 変換を用いて高速化したものである．

a. チェビシェフ級数展開法
簡単のためチェビシェフ級数展開しようとする関数 $f(x)$ の定義域は [-1,1] とする．
$f(x)$ の真のチェビシェフ級数展開は

$$f(x)=\sum_{k=0}^{\infty}{}'c_kT_k(x) \tag{4.1}$$

$$c_k=\frac{2}{\pi}\int_{-1}^{1}\frac{f(x)T_k(x)}{\sqrt{1-x^2}}dx \tag{4.2}$$

で表せる．
ここで，$x=\cos\theta$ と変数変換すると，

$$c_k=\frac{2}{\pi}\int_0^{\pi}f(\cos\theta)\cos k\theta d\theta \tag{4.3}$$

となり，c_k は 2π を周期とする連続な偶関数 $f(\cos\theta)$ の cosine 級数の係数とみなすことができる．
チェビシェフ展開係数 (4.2) をガウス・チェビシェフ数値積分公式で求めることは，実は偶関数のフーリエ係数 (4.3) を中点公式で計算することに他ならない．ガウス型積分公式は，n-1 個の標本点を用いて 2n-3 次の多項式まで正確に積分できるという意味で最良である．積分 (4.3) に対する台形公式は，n 個の標本点を用い，2n-3 次の cosine 多項式まで正確に積分できる．すなわち最良の積分公式ではないが，殆どそれに近い．

本サブルーチンでは，入力は関数であるため要求精度に応じ，標本点を倍々と増し，必要な項数のチェビシェフ級数を求める．そこで，この目的に適した台形公式を用いる．得られた n 項のチェビシェフ級数は，n 個の標本点

$$x_j = \cos\frac{\pi}{n-1} j, \quad j = 0,1,...,n-1 \tag{4.4}$$

で，補間条件

$$f(x_j) = \sum_{k=0}^{n-1}{}'' c_k T_k(x_j)$$

を満たす．
チェビシェフ級数の表記上の統一を図るため，末尾の係数だけ 1/2 倍し，(4.1) のように書く．
本サブルーチンでは，後述する級数展開の誤差評価に基づき，区間 [-1,1]で

$$\left| f(x) - \sum_{k=0}^{n-1}{}' c_k T_k(x) \right| < \max\{\varepsilon_a, \varepsilon_r \| f \|\} \tag{4.5}$$

を満たすように項数 n と $\{c_k\}$ を FFT の手法で決定する．
今，ある項数 n ($=n_p+1$, $n_p=2^p$) に対する台形公式による係数 $\{c_k^p\}$ が (4.6) のように求められているとする．

$$c_k^p = \sum_{j=0}^{n_p}{}'' f\left(\cos\frac{\pi}{n_p} j\right) \cos\frac{\pi}{n_p} kj, \quad k = 0,1,...,n_p \tag{4.6}$$

ただし，正規化定数 $2/n_p$ は省略している．
項数を倍に拡張した $n = n_{p+1}+1$ に対する係数 $\{c_k^{p+1}\}$ は，台形公式と中点公式との補完的な性質により効率よく求められる．すなわち，$\{c_k^p\}$ を得た標本点の中点で $f(x)$ を (4.7) のように標本化し，

$$f\left(\cos\frac{\pi}{n_p}\left(j+\frac{1}{2}\right)\right), j = 0,1,.....,n_p - 1 \tag{4.7}$$

これに対する離散型 cosine 変換（中点公式）を (4.8) で定義する．

$$\tilde{c}_k^p = \sum_{j=0}^{n_p-1} f\left(\cos\frac{\pi}{n_p}\left(j+\frac{1}{2}\right)\right) \cos\frac{\pi}{n_p} k\left(j+\frac{1}{2}\right), \quad k = 0,1,.....,n_p - 1 \tag{4.8}$$

このとき $\{c_k^{p+1}\}$ は (4.9) の漸化式により求められる．

$$\left.\begin{aligned} c_k^{p+1} &= c_k^p + \tilde{c}_k^p \\ c_{n_{p+1}-k}^{p+1} &= c_k^p - \tilde{c}_k^p \\ c_{n_p}^{p+1} &= c_{n_p}^p \end{aligned}\right\}, k = 0,1,...,n_p - 1 \tag{4.9}$$

以上のように，(4.6)，(4.8)，(4.9) に基づき，p を漸次増加していくことにより $f(x)$ の（離散型）チェビシェフ級数

$$f(x) \cong \sum_{k=0}^{n_p}{}'' c_k^p T_k(x) \tag{4.10}$$

の係数 $\{c_k^p\}$ を生成することができる．
収束判定が満たされたとき，最後に各係数 $\{c_k^p\}$ に $2/n_p$ をかけて，それらを正規化する．

b. チェビシェフ級数の誤差評価
真のチェビシェフ級数展開 (4.1) の係数 $\{c_k\}$ と，p 段階の離散型チェビシェフ級数展開 (4.10) の係数 $\{c_k^p\}$ の間には，次の関係が成り立つ．

$$c_k^p = c_k + \sum_{m=1}^{\infty}\left(c_{2mn_p-k} + c_{2mn_p+k}\right), \quad k = 0,1,...,n_{p-1} \tag{4.11}$$

(4.11) より，p 段階のチェビシェフ級数の誤差評価

$$\left| f(x) - \sum_{k=0}^{n_p}{}'' c_k^p T_k(x) \right| \le 2\sum_{k=n_p+1}^{\infty} |c_k| \tag{4.12}$$

が導かれる．
区間上で $f(x)$ が解析的ならば，その展開係数 c_k は番号 k の増加に伴い指数関数的オーダ $O(r^k)$，$0 < r < 1$ で減少するので，p 段階の離散型チェビシェフ係数 c_k^p から r を推定することができる．
いま，$c_k^p = Ar^k$ （A はある定数）と表す．k は高々 n_p であるから，末尾の番号 n_p と $3/4n_p$ の係数の比から $r^{n_p/4}$ を知ることができる．本サブルーチンでは，偶然に c_{n_p} あるいは $c_{3/4n_p}$ が 0 となると都合が悪いので，その対策上 n_p -1 及び $3/4n_p$-1 番目の係数も併用して，r を

$$r = \min\left\{\left(\frac{\left|c_{n_p-1}^p\right| + \frac{1}{2}\left|c_{n_p}^p\right|}{\left|c_{\frac{3}{4}n_p-1}^p\right| + \left|c_{\frac{3}{4}n_p}^p\right|}\right)^{4/n_p}, 0.99\right\}$$

で推定している．r の上限を 0.99 とするのは，これ以上 r が 1 に近くなると，収束は極めて緩くなり事実上 $f(x)$ のチェビシェフ展開は不可能だからである．
さて，r が得られたならば，(4.12) より p 段階の誤差 e_p は，(4.13) で推定できる．

$$e_p = \frac{r}{1-r}\left(\left|c^p_{n_p-1}\right| + \frac{1}{2}\left|c^p_{n_p}\right|\right) \qquad (4.13)$$

ここで，末尾の係数 $c^p_{n_p}$ だけ 1/2 倍されているのは，これが台形公式による離散型チェビシェフ級数だからである．

c.　計算の手順

手順 1……初期設定

① 三角関数表の初期値設定
区間 $[0, \pi/2]$ を等分割した点上で，cosine 関数の値をビット逆転の順序に 3 個だけ求める．
三角関数表は，関数 $f(x)$ の標本化の処理において，標本点として利用する．

② 初期のチェビシェフ級数展開
(4.6) において n_p=4 とした，5 項のチェビシェフ級数展開を行い，c^2_0，c^2_1，c^2_2，c^2_3，c^2_4 を求める．
同時に，(1.3) に基づき関数のノルム $\|f\|$ を求める．

手順 2……収束判定

n_p+1 ≤ NMIN ならば，収束判定は省略し無条件に手順 3 へ進む．n_p+1 > NMIN ならば，以下のように収束判定を行なう．
まず計算誤差の限界 ρ

$$\rho = \sqrt{n_p}\,\|f\|(2u)，\ u \text{ は丸め誤差の単位} \qquad (4.14)$$

収束の判定定数 ε

$$\varepsilon = \max\left\{\varepsilon_a, \varepsilon_r \|f\|\right\} \qquad (4.15)$$

を求める．
p 段階の末尾 2 項の展開係数 $c^p{}_{n_p-1}$，$c^p_{n_p}$ が有効桁を失っている場合，すなわち

$$\left|c^p_{n_p-1}\right| + \frac{1}{2}\left|c^p_{n_p}\right| < \rho \qquad (4.16)$$

ならば，これ以上計算を続行しても精度の向上は期待できないので収束したと判定し，チェビシェフ級数の絶対誤差 e_p を ρ で推定する．$p < \varepsilon$ ならば，ICON = 0 として終わる．
ρ ≥ ε ならば，要求精度 ε_a，ε_r が小さすぎるとみなし ICON = 10000 として処理を終わる．

一方，末尾 2 項の展開係数が有効桁をもつ場合，誤差 e_p を (4.13) で推定し

$$e_p < \varepsilon \qquad (4.17)$$

を満たすか否かを判定する．満たされるならば，ICON = 0 として終わる．満たされない場合で，かつ $2n_p + 1 \le$ NMAX ならば，手順 3 へ進む．一方，$2n_p + 1 >$ NMAX ならば，与えられた展開項数の上限に達しても要求精度は得られなかったとみなし，ICON = 20000として処理を終わる．
なお，正常，異常にかかわらず終了時に係数を正規化する．

手順 3……関数の標本化とノルムの計算

既に得られている $n_p + 1$ 個の標本の中点で，関数を

$$f\left(\frac{a+b}{2} + \frac{b-a}{2}\cos\frac{\pi}{n_p}\left(j+\frac{1}{2}\right)\right), \qquad j = 0,1,.....,n_p - 1 \qquad (4.18)$$

のように標本化し，ビット逆転の順序に求める．
ただし，$a = 0$ の場合は，変数変換における桁落ちを防止するために，次のように標本化する．

$$f\left(b\cos^2\frac{\pi}{2n_p}\left(j+\frac{1}{2}\right)\right), \qquad j = 0,1,.....,n_p - 1 \qquad (4.19)$$

ここで新たに用いられる三角関数表を追加計算する．
また，関数のノルムを (1.3) に基づき求める．

手順 4……離散型 cosine 変換（中点公式）

手順 3 で求めた標本に対して，離散型 cosine 変換を FFT の手法で行い，$\{\tilde{c}^p_k\}$ を求める．

手順 5……$\{c^{p+1}_k\}$ の計算

既に求められている $\{c^p_k\}$ と，手順 4 により得られた $\{\tilde{c}^p_k\}$ に基づき，漸化式 (4.9) により，$2n_p$+1 項のチェビシェフ級数の係数 $\{c^{p+1}_k\}$ を合成する．p を 1 つ増加して手順 2 へ戻る．

計算の手間の大部分は，手順3，4 で費される．関数の標本化の手間を除けば，n 項のチェビシェフ展開係数を求めるに必要な乗算回数は，約 $n\log_2 n$ である．

なお詳細については，参考文献 [59] を参照すること.
また，離散型 cosine 変換については，サブルーチン FCOST，FCOSM を参照のこと.

E51-10-0101 FCOSF, DFCOSF

偶関数の cosine 級数展開 （関数入力，高速 cosine 変換）
CALL FCOSF (TH, FUN, EPSA, EPSR, NMIN, NMAX, A, N, ERR, TAB, ICON)

(1) 機能

任意の周期 $2T$ をもつ滑らかな偶関数 $f(t)$ を要求精度 ε_a，ε_r に基づき cosine 級数展開する．すなわち

$$\left| f(t) - {\sum_{k=0}^{n-1}}' a_k \cos\frac{\pi}{T}kt \right| \le \max\left\{\varepsilon_a, \varepsilon_r \|f\|\right\} \qquad (1.1)$$

を満たす n 個の係数 $\{a_k\}$ を求める．

ここで Σ'は，初項を 1/2 倍して和をとることを意味する．また，関数のノルム $\|f\|$ は，半周期 $[0, T]$ での関数の標本点，

$$t_j = \frac{T}{n-1} j, \quad j = 0,1,...,n-1 \qquad (1.2)$$

における関数値を用いて

$$\|f\| \le \max_{0 \le j \le n-1} \left| f(t_j) \right| \qquad (1.3)$$

と定義する．

ただし，$T > 0$，$\varepsilon_a \ge 0$，$\varepsilon_r \ge 0$ であること．

(2) パラメタ

TH……**入力**．関数 $f(t)$ の半周期 T.

FUN……**入力**．展開しようとする関数 $f(t)$ を計算する関数副プログラム名（使用例参照）

EPSA……**入力**．級数展開の絶対誤差の上限 ε_a.

EPSR……**入力**．級数展開の相対誤差の上限 ε_r.

NMIN……**入力**．級数展開の項数 n の下限 (≥0).
(2 のべき) + 1 の値で与える．標準値は 9．(使用上の注意③参照).

NMAX…**入力**．級数展開の項数の上限(≥NMIN).
(2 のべき) + 1 の値で与える．
標準値は 257．(使用上の注意③参照).

A ……**出力**．係数 $\{a_k\}$.
大きさ NMAX の 1次元配列.
A(1) = a_0，A(2) = a_1，…，A(N) = a_{n-1}
のように格納される．

N ……**出力**．級数展開の項数 n(≥5).
(2 のべき) + 1 の値をとる．

ERR……**出力**．級数展開の絶対誤差の推定値.

TAB……**出力**．級数展開で使用された三角関数表が格納される．
大きさ 3 以上でかつ (NMAX - 3)/2 の 1 次元配列

ICON……**出力**．コンディションコード.
表 FCOSF-1 参照.

表 FCOSF-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	丸め誤差のために要求精度が得られなかった. 要求精度が厳しすぎる.	A には得られた係数を出力する．級数展開の誤差は達成可能な限度まで達している.
20000	級数展開の項数が，その上限に達しても要求精度が得られなかった.	処理を打ち切る．この時点で得られた係数と絶対誤差の推定値を，それぞれ A と ERR に出力する.
30000	次のいずれかであった. ①TH ≤ 0 ②EPSA < 0.0 ③EPSR < 0.0 ④NMIN < 0 ⑤NMAX < NMIN	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… MGSSL, AMACH, UTABT, UCOSM, UNIFC

② FORTRAN 基本関数… ABS, AMAX1, AMIN1, FLOAT

b. 注意

① パラメタ FUN に対応する関数副プログラムは，独立変数 t だけを引数にもつ関数副プログラムとして区間 $[0, T]$ 上で定義し，本サブルーチンを呼び出す側のプログラムで，その関数名をEXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図る．(使用例参照)

② 三角関数表の扱いについて
本サブルーチンを繰返し呼ぶ場合，必要な三角関数表は，1 回だけしか計算されない．すなわち，三角関数表に不足が生じた場合，その都度不足分だけ計算し表に追加される．
したがって，パラメタ TAB の内容は，常に保存したまま呼び出すこと.

③ NMIN，NMAX が (2のべき) + 1 の値でない場合，それらを超えない最大の (2 のべき) + 1 の値とみなす．ただし NMAX < 5 の場合，NMAX = 5 とみなす.

④ 展開の項数 n の増大とともに，誤差が小さくなる度合は，全区間（-∞,∞）における関数 $f(t)$ の滑らかさに著しく影響される．$f(t)$ が解析的周期関数ならば，誤差は指数関数のオーダー $O(r^n)$，$0 < r < 1$ で減少し，$f(x)$ が k 回連続微分可能ならば，有理式のオーダ $O(n^{-k})$ で減少する．$k = 0, 1$ の場合展開項数が多くなり，誤差の推定も正確さを期しがたい．
したがって本サブルーチンを適用する関数は少なくとも 2 回連続微分可能であることが望ましい．

⑤ 級数の精度
本サブルーチンは，与えられた ε_a，ε_r に対して (1.1) を満たす級数を求める．したがって $\varepsilon_r = 0$ とおけば絶対誤差 ε_a 以内の，また $\varepsilon_a = 0$ とおけば相対誤差 ε_r 以内の精度で級数展開することになる．
しかし ε_a と ε_r の値によっては，この目的が達成されないことがある．例えば，関数の計算誤差に比べて，ε_a，ε_r が小さすぎる場合には，展開項数がその上限に達しない場合でも丸め誤差の影響が大きくなり，これ以上計算を続行しても無意味である．
このような状態を見出したときは ICON を 10000 として処理を打ち切る．このときの級数の精度は，使用計算機で達成可能な限度まで達している．また，関数の性質によって展開の項数が NMAX 以内では収束しないこともあり，この場合 ICON を 20000 にして処理を打ち切る．このときの係数の値は，それまでに得られている近似値であり，級数の精度は保証されない．
しかしながら，いずれの場合でも級数の絶対誤差の推定値はパラメタ ERR に出力される．

⑥ サブルーチン FCOST を用い逆変換を行う場合，末尾の係数 a_{n-1} だけは 2 倍しておかなければならない．また，順変換，逆変換いずれの場合もパラメタ TAB の内容は保存しておくこと．（使用例②参照）

⑦ 関数 $f(t)$ が単に周期関数であって，その cosine 変換を計算する場合，偶関数 $(f(t) + f(-t))/2$ に対して本サブルーチンを適用すればよい．

⑧ 偶関数 $f(t)$ が周期をもたず絶対積分可能なとき，$f(t)$ の（理論的）cosine 変換は

$$F(\omega) = \int_0^\infty f(t)\cos\omega t\, dt \qquad (3.1)$$

で定義される．$f(t)$ が $O(e^{-at})$，$a < 0$ で減衰するならば，次のようにして上のフーリエ積分の近似値を求めることができる．

まず T を十分大きくとり区間 $[T, \infty)$ において $|f(t)|$ は無視できるものとする．例えば，u を丸め誤差の単位として

$$|f(t)| < u, t \geq T \qquad (3.2)$$

を満たすように T を定め，$f(t)$ を周期 $2T$ の関数とみなし，本サブルーチンによって $f(t)$ の cosine 展開係数 $\{a_k\}$ を求める．ところで

$$a_k = \frac{2}{T}\int_0^T f(t)\cos\frac{\pi}{T}kt\, dt \qquad (3.3)$$

であるから (3.1)，(3.2) より

$$F\left(\frac{\pi}{T}k\right) \approx \frac{T}{2}a_k, k = 0,1,\ldots,n-1 \qquad (3.4)$$

が成り立つ．この関係を用いて離散型 cosine 変換から，cosine 変換 (3.1) の近似値を計算できる．
なお逆変換

$$f(t) = \frac{2}{\pi}\int_0^\infty F(\omega)\cos\omega t\, d\omega \qquad (3.5)$$

を計算する場合は，n 個のデータ

$$\frac{2}{T}F\left(\frac{\pi}{T}k\right), k = 0,1,\ldots,n-2$$
$$\frac{4}{T}F\left(\frac{\pi}{T}(n-1)\right)$$

を入力としてサブルーチン FCOST を呼べば，出力として

$$f\left(\frac{T}{n-1}j\right), j = 0,1,\ldots,n-1$$

の近似値が得られる．
（使用例②参照）

c. 使用例

① 助変数 p を含む周期 2π の偶関数

$$f(t) = \frac{1-p^2}{1-2p\cos t + p^2}$$

を要求精度 $\varepsilon_a = \varepsilon_r = 5\cdot 10^{-5}$ 及び NMIN = 9, NMAX = 257 の下で cosine 級数に展開する．

f(*t*) の真の cosine 級数展開は

$$f(t)=1+2\sum_{k=1}^{\infty}p^k\cos kt$$

である．p を 1/4，1/2，3/4 と変えて展開係数を出力する．

```
C     **EXAMPLE**
      DIMENSION A(257),TAB(127)
      EXTERNAL FUN
      COMMON P
      TH=ATAN(1.0)*4.0
      EPSA=0.5E-4
      EPSR=EPSA
      NMIN=9
      NMAX=257
      P=0.25
    1 CALL FCOSF(TH,FUN,EPSA,EPSR,NMIN,
     *           NMAX,A,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GO TO 10
      WRITE(6,600) N,ERR,ICON,P
      WRITE(6,601) (A(I),I=1,N)
      P=P+0.25
      IF(P.LT.1.0) GO TO 1
      STOP
   10 WRITE(6,602) ICON
      STOP
  600 FORMAT('0',5X,'EXPANSION OF',
     * ' FUNCTION FUN(T)',3X,'N=',I4,
     * 5X,'ERR=',E15.5,5X,'ICON=',I16,5X,
     * 'P=',E15.5)
  601 FORMAT(/(5E15.5))
  602 FORMAT('0',5X,'CONDITION CODE',I8)
      END
      FUNCTION FUN(T)
      COMMON P
      FUN=(1.0-P*P)/(1.0-2.0*P*COS(T)+P*P)
      RETURN
      END
```

② cosine 変換およびその逆変換
無限区間における偶関数の cosine 変換

$$F(\omega)=\int_0^{\infty}\operatorname{sech}\frac{\pi}{2}x\cos\omega x\,dx$$

を $\varepsilon_a=\varepsilon_r=5\cdot10^{-5}$ の下で行い，解析解 $F(\omega)=\operatorname{sech}\omega$ と比較する．
引続きその逆変換

$$\frac{2}{\pi}\int_0^{\infty}F(\omega)\cos\omega x\,dx$$

をサブルーチン FCOST を用いて行い，解析解と比べ精度を調べる．

```
C     **EXAMPLE**
      DIMENSION A(257),TAB(127),
     *          ARG(257),T(257)
      EXTERNAL FUN
      COMMON HP
      HP=ATAN(1.0)*2.0
      TH=ALOG(1.0/AMACH(TH))/HP
      EPSA=0.5E-04
      EPSR=EPSA
      NMIN=9
      NMAX=257
C     COSINE TRANSFORM
      CALL FCOSF(TH,FUN,EPSA,EPSR,NMIN,
     *           NMAX,A,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GO TO 10
      TQ=TH*0.5
      H=(HP+HP)/TH
      DO 1 K=1,N
      ARG(K)=H*FLOAT(K-1)
      A(K)=A(K)*TQ
      T(K)=TRFN(ARG(K))
    1 CONTINUE
      WRITE(6,600) N,ERR,ICON
      WRITE(6,610)
      WRITE(6,601) (ARG(K),A(K),T(K),
     *              K=1,N)
C     INVERSE TRANSFORM
      Q=1.0/TQ
      DO 2 K=1,N
      A(K)=A(K)*Q
    2 CONTINUE
      A(N)=A(N)*2.0
      NM1=N-1
      CALL FCOST(A,NM1,TAB,ICON)
      IF(ICON.NE.0) GO TO 10
      H=TH/FLOAT(NM1)
      DO 3 K=1,N
      ARG(K)=H*FLOAT(K-1)
      T(K)=FUN(ARG(K))
    3 CONTINUE
      WRITE(6,620)
      WRITE(6,610)
      WRITE(6,601) (ARG(K),A(K),T(K),
     *              K=1,N)
      STOP
   10 WRITE(6,602) ICON
      STOP
  600 FORMAT('0',5X,'CHECK THE COSINE',
     * ' TRANSFORM OF FUNCTION FUN(T)',
     * 3X,'N=',I4,5X,'ERR=',E15.5,5X,
     * 'ICON=',I5)
  610 FORMAT('0',6X,'ARGUMENT',7X,
     * 'COMPUTED',11X,'TURE')
  620 FORMAT('0',5X,'CHECK THE INVERSE',
     * ' TRANSFORM')
  601 FORMAT(/(3E15.5))
  602 FORMAT('0',5X,'CONDITION CODE',I6)
      END
      FUNCTION FUN(T)
      COMMON HP
      FUN=1.0/COSH(T*HP)
      RETURN
      END
      FUNCTION TRFN(W)
      TRFN=1.0/COSH(W)
      RETURN
      END
```

(4) 手法概要

本方法は，離散点入力の高速 cosine 変換（台形公式）を関数入力の cosine 変換に応用したものである．

a. cosine 級数展開法

簡単のため偶関数 $f(t)$ の周期は 2π とする．

$f(t)$ の cosine 級数は

$$f(t)=\frac{1}{2}a_0+\sum_{k=1}^{\infty}a_k\cos kt \tag{4.1}$$

$$a_k=\frac{2}{\pi}\int_0^{\pi}f(x)\cos kt\,dt \tag{4.2}$$

と表される．積分区間 [0, π] を 2 等分，4 等分，8 等分，……と細分しながら (4.2) を台形公式で求める．

得られた，3 個，5 個，9 個……の係数 a_k により，(4.1) を有限の項数で近似する．

被積分関数が滑らかならば，要求精度 ε_a , ε_r に応じて等分割数を倍々と増し，適当に大きく標本数 n (=(2のべき) + 1) をとれば

$$\left|f(t)-\sum_{k=0}^{n-1}{}''a_k\cos kt\right|<\max\left\{\varepsilon_a,\varepsilon_r\|f\|\right\} \tag{4.3}$$

が満たされる．ただし Σ'' は，初項と末項を 1/2 倍して和をとることを意味する．

得られた n-1 次の三角多項式は，台形公式の各分点を補間点とする三角補間多項式になっている．すなわち

$$f\left(\frac{\pi}{n-1}j\right)=\sum_{k=0}^{n-1}{}''a_k\cos\frac{\pi}{n-1}kj,\quad j=0,1,...,n-1 \tag{4.4}$$

となる．

上述したことを，更に詳しく説明する．

今，標本数 $n(=n_p+1,\ n_p=2^p)$ の台形公式による係数

$$a_k^p=\sum_{j=0}^{n_p}{}''f\left(\frac{\pi}{n_p}j\right)\cos\frac{\pi}{n_p}kj,\ k=0,1,...,n_p \tag{4.5}$$

が求められているとする．ここで正規化定数 $2/n_p$ は省略している．

$n_{p+1}=2n_p$ として，項数を倍に拡張したときの係数 $\{a_k^{p+1}\}$ は，台形公式に対する中点公式の補完的性質により効率よく求められる．すなわち台形公式 (4.5) の標本点の中点で $f(t)$ を (4.6) のように標本化し

$$f\left(\frac{\pi}{n_p}\left(j+\frac{1}{2}\right)\right),\quad j=0,1,...,n_p-1 \tag{4.6}$$

これに対する離散型 cosine 変換（中点公式）を (4.7) で定義する．

$$\tilde{a}_k^p=\sum_{j=0}^{n_p-1}f\left(\frac{\pi}{n_p}\left(j+\frac{1}{2}\right)\right)\cos\frac{\pi}{n_p}k\left(j+\frac{1}{2}\right),\quad k=0,1,...,n_p-1 \tag{4.7}$$

このとき，(4.8) の関係が成り立ち，$\{a_k^{p+1}\}$ を求めることができる．

$$\left.\begin{aligned}a_k^{p+1}&=a_k^p+\tilde{a}_k^p\\ a_{n_{p+1}-k}^{p+1}&=a_k^p-\tilde{a}_k^p\\ a_{n_p}^{p+1}&=a_{n_p}^p\end{aligned}\right\},k=0,1,...,n_p-1 \tag{4.8}$$

この $\{a_k^p\}$ に関する漸化関係を用い，要求精度に応じ，項数を倍々に増しながら $f(t)$ の高次の cosine 級数を求めることができる．ただし $\{a_k^p\}$ は，最後の段階で $2/n_p$ をかけられ正規化される．なお末尾の係数だけは $1/n_p$ 倍し，cosine 級数の表記を (4.1) のように統一する．

b. cosine 級数の誤差評価

$f(t)$ の理論的な cosine 係数 a_k と p 段階の離散型 cosine 係数 a_k^p の間には，それらの定義式 (4.2)，(4.5) 及び三角関数の選点直交関係より

$$a_k^p=a_k+\sum_{m=1}^{\infty}\left(a_{2mn_p-k}+a_{2mn_p+k}\right),\quad k=0,1,...,n_p \tag{4.9}$$

が成り立つ．これより p 段階の cosine 級数の誤差評価

$$\left|f(t)-\sum_{k=0}^{n_p}{}''a_k^p\cos kt\right|\le 2\sum_{k=n_p+1}^{\infty}|a_k| \tag{4.10}$$

が導かれる．

ところで $f(t)$ が解析的周期関数のとき，その展開係数 $\{a_k\}$ は番号 k の増加に伴い指数関数的オーダ $O(r^k)$，$0<r<1$ で減少する．そこで p 段階の離散型 cosine 係数から r を推定することができる．いま $a_k^p=Ar^k$（A はある定数）と表す．k は高々 n_p であるから末尾の番号 n_p と $3/4n_p$ の係数の比から $r^{np/4}$ を知ることができる．本サブルーチンでは，偶然にそれらが 0

となると都合がわるいので n_p - 1 及び 3/4n_p - 1 番の係数も併用して r を

$$r=\min\left\{\left(\frac{\left|a^p_{n_p-1}\right|+\frac{1}{2}\left|a^p_{n_p}\right|}{\left|a^p_{\frac{3}{4}n_p-1}\right|+\left|a^p_{\frac{3}{4}n_p}\right|}\right)^{4/n_p},0.99\right\}$$

で推定している．r の上限を 0.99 としたのは，これ以上 r が 1 に近くなれば，収束は極めて緩くなり事実上 $f(t)$ の cosine 展開は不可能だからである．
さて，r が得られたならば (4.10) より p 段階の誤差は

$$e_p=\frac{r}{1-r}\left(\left|a^p_{n_p-1}\right|+\frac{1}{2}\left|a^p_{n_p}\right|\right) \tag{4.11}$$

で推定できる．ここで末尾の係数 $a^p_{n_p}$ だけ 1/2 倍されているのは，これが台形公式による離散型 cosine 係数だからである．

c.　計算の手順

手順1…初期設定

① 三角関数表の初期値設定
区間 [0, π/2] を等分割した点上で cosine 関数の値をビット逆転の順序に 3 個だけ求める．本サブルーチンが既に呼ばれ，三角関数表が作成済みならば，この初期化の操作は省略される．三角関数表は，離散型 cosine 変換のために用いられる．

② 初期の cosine 級数展開
(4.5) において，n_p = 4(p=2) とした 5 項のcosine 級数展開を行い，a_0^2, a_1^2, a_2^2, a_3^2, a_4^2 を計算する．同時にノルムの定義 (1.3) に基づき $\|f\|$ を求める．

手順2…収束判定

n_p + 1 ≤ NMIN ならば，収束判定は省略し無条件に手順 3 に進む．n_p + 1 > NMIN ならば以下の収束判定を行う．
まず計算誤差の限界

$$\rho=\sqrt{n_p}\,\|f\|(2u)，\ u \text{ は丸め誤差の単位} \tag{4.12}$$

と収束判定定数

$$\varepsilon=\max\{\varepsilon_a,\varepsilon_r\|f\|\} \tag{4.13}$$

を求める．
p 段階の末尾 2 項の係数が有効桁を失っている場合，すなわち

$$\left|a^p_{n_p-1}\right|+\frac{1}{2}\left|a^p_{n_p}\right|<\rho \tag{4.14}$$

ならば，これ以上続行しても精度の向上は期待できないので収束したと判定し，求める級数の絶対誤差 e_p を計算誤差 ρ で置き換える．$\rho<\varepsilon$ ならば ICON = 0 として，又は $\rho\geq\varepsilon$ ならば ε_a, ε_rが，丸め誤差の単位 u に比べ相対的に小さすぎるとみなし ICON = 10000 として処理を終わる．
次に (4.14) が成立しない通常の場合，誤差 e_p を (4.11) で推定し

$$e_p<\varepsilon \tag{4.14}$$

を満たすか否か判定する．満たされるならば，ICON = 0 として処理を終わる．満たされない場合で，かつ 2n_p + 1 ≤ NMAX ならば，手順 3 へ進む．そうでなければ，与えられた展開項数の上限に達しても要求精度は得られなかったとみなし，ICON = 20000 として処理を終わる．正常，異常にかかわらず，終了時には係数を正規化する．

手順3…標本点の計算

p 段階において $f(t)$ を標本化するための標本点は

$$t_j=\frac{\pi}{n_p}\left(j+\frac{1}{2}\right),\quad j=0,1,...,n_p-1$$

であるが，これをビット逆転の順序に，次の漸化式に従って求める．

$$\begin{aligned}&t_0=\pi/2^{p+1}\\&t_{(2j+1)2^{p-l-1}}=t_{j2^{p-l}}+2^{-p+l}\pi\\&j=0,1,...,2^l-1,l=0,1,...,p-1\end{aligned} \tag{4.16}$$

ただし $n_p=2^p$ である．

手順4…関数の標本化とノルムの計算

n_p 個の標本点 (4.16) の上で $f(t)$ の関数値を求め，標本点の上に重ね書きする．
また関数のノルムの定義 (1.3) に従い，ノルム $\|f\|$ を計算する．

手順5…三角関数表の作成

次の手順 6 で必要な三角関数表を計算し保存する．三角関数の有効利用のため本サブルーチンが繰返し呼ばれる場合でも一度求めた表（またはその一部）は，二度と計算しない．

手順6…離散型 cosine 変換（中点公式）

手順 4 で求めた標本に対し，離散型 cosine 変換を FFT の手法で行い，$\left\{\tilde{a}_k^p\right\}$ を求める．

手順7… $\left\{a_k^{p+1}\right\}$ の計算

既に求められている $\left\{a_k^p\right\}$ と，手順 6 で得られた $\left\{\tilde{a}_k^p\right\}$ を，(4.8) によって合成し項数 $2n_p + 1$ の離散型 cosine 展開係数 $\left\{a_k^{p+1}\right\}$ を得る．

p を 1 増して手順 2 にもどる．

計算の手間の大部分は，手順 4，6 で費される．関数の標本化の手間を除けば，n 項の cosine 展開係数を求めるに必要な乗算回数は，約 $n\log_2 n$ である．

記憶領域の節約を図るため，標本点，標本及び展開係数は，1 次元配列 A において三重に重ね書きされる．

なお詳細については，参考文献 [59] を参照すること．また離散点入力の cosine 変換については，サブルーチン FCOST，FCOSM を参照のこと．

F11-11-0201 FCOSM, DFCOSM

離散型 cosine 変換（中点公式，2 基底 FFT）
CALL FCOSM (A, N, ISN, TAB, ICON)

(1) 機能

周期 2πの偶関数 $x(t)$ の半周期を等分割した n 個の標本 $\{x_{j+1/2}\}$

$$x_{j+1/2} = x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right), j = 0,1,...,n-1 \qquad (1.1)$$

が与えられたとき，中点公式による離散型 cosine 変換，又はその逆変換を高速変換手法 (FFT) により行う．

ただし，$n = 2^l$（l: 0 又は，正整数）であること．

① cosine 変換
$\{x_{j+1/2}\}$ を入力し，(1.2) で定義する変換を行い，フーリエ係数 $\{n/2 \cdot a_k\}$ を求める．

$$\frac{n}{2}a_k = \sum_{j=0}^{n=1} x_{j+1/2} \cos\frac{\pi}{n}k\left(j+\frac{1}{2}\right), k = 0,1,...,n-1 \quad (1.2)$$

② cosine 逆変換
$\{a_k\}$ を入力し，(1.3) で定義する変換を行い，フーリエ級数の値 $\{x_{j+1/2}\}$ を求める．

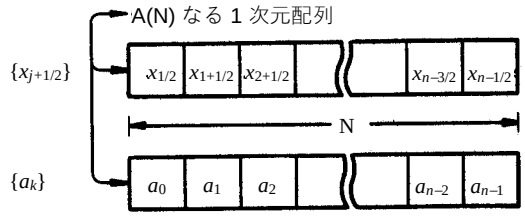
$$x_{j+1/2} = \sum_{k=0}^{n-1}{}' a_k \cos\frac{\pi}{n}k\left(j+\frac{1}{2}\right), j = 0,1,...,n-1 \quad (1.3)$$

ここで Σ' は，初項だけを 1/2 倍して和をとることを意味する．

(2) パラメタ

A ····· **入力**．$\{x_{j+1/2}\}$ 又は $\{a_k\}$．
出力．$\{n/2 \cdot a_k\}$ 又は $\{x_{j+1/2}\}$．
大きさ n の 1次元配列．
図 FCOSM-1 参照．

N ····· **入力**．標本数 n．



[注意] $\{\frac{n}{2}a_k\}$ は $\{a_k\}$ に準じる．

図 FCOSM-1 データの格納方法

ISN ····· **入力**．変換か逆変換かを指定する（+1 又は -1）．
変換　: ISN = +1
逆変換 : ISN = -1

TAB ····· **出力**．変換で使用された三角関数表が格納される．
大きさ n-1 の 1 次元配列．
（使用上の注意②参照）

ICON ····· **出力**．コンディションコード
表 FCOSM-1 参照．

表 FCOSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	ISN ≠ 1，ISN ≠ -1 又は N ≠ 2^l （l: 0 又は正整数）であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II···MGSSL, UPNR2, UTABT, UCOSM
② FORTRAN 基本関数···SQRT, MAX0

b. 注意
① 一般的なフーリエ変換の定義について
中点公式による離散型 cosine 変換及び，逆変換は，一般的に (3.1)，(3.2) で定義される．

$$a_k = \frac{2}{n}\sum_{j=0}^{n-1} x_{j+1/2}\cos\frac{\pi}{n}k\left(j+\frac{1}{2}\right), \quad k = 0,1,...,n-1 \qquad (3.1)$$

$$x_{j+1/2} = \sum_{k=0}^{n-1}{}' a_k \cos\frac{\pi}{n}k\left(j+\frac{1}{2}\right), \quad j = 0,1,...,n-1 \qquad (3.2)$$

本サブルーチンでは，(3.1)，(3.2) の左辺に対応して $\{n/2 \cdot a_k\}$，$\{x_{j+1/2}\}$ を求めるので，結果の正規化は必要に応じて行うこと．

② 三角関数表の扱いについて
同一項数の下で本サブルーチンを繰り返し呼ぶ場合，三角関数表の計算は最初の 1 回だけ行われる．
したがって 2 回目以降，パラメタ TAB の内容は保存したまま呼び出すこと．
一方，変換の項数が異なる場合でも，三角関数表に不足が生じた場合だけ，その都度不足分を計算し表に追加するので効率がよい．

c. 使用例

n 個の標本 $\{x_{j+1/2}\}$ を入力し，本サブルーチンにより変換したのち正規化し，離散型フーリエ係数 $\{a_k\}$ を求める．引続き逆変換することによって $\{x_{j+1/2}\}$ を求める．

$n \le 512$ の場合．

```
C       **EXAMPLE**
        DIMENSION X(512),TAB(511)
C       COSINE TRANSFORM
        ISN=1
        READ(5,500) N,(X(I),I=1,N)
        WRITE(6,600) N
        WRITE(6,601) (X(I),I=1,N)
        CALL FCOSM(X,N,ISN,TAB,ICON)
        IF(ICON.NE.0) GO TO 20
C       NORMALIZE
        CN=2.0/FLOAT(N)
        DO 10 K=1,N
        X(K)=X(K)*CN
   10   CONTINUE
        WRITE(6,602) ISN
        WRITE(6,601) (X(I),I=1,N)
C       COSINE INVERSE TRANSFORM
        ISN=-1
        CALL FCOSM(X,N,ISN,TAB,ICON)
        IF(ICON.NE.0) GO TO 20
        WRITE(6,602) ISN
        WRITE(6,601) (X(I),I=1,N)
   20   WRITE(6,603) ICON
        STOP
  500   FORMAT(I5/(6F12.0))
  600   FORMAT('0',5X,'INPUT DATA N=',I5)
  601   FORMAT(5F15.7)
  602   FORMAT('0',5X,'OUTPUT DATA',5X,
     *         'ISN=',I2)
  603   FORMAT('0',5X,'CONDITION CODE',
     *         I8)
        END
```

(4) 手法概要

項数 n (=2^l, l = 0, 1, ⋯) の中点公式による離散型 cosine 変換を，2 を基底とする高速変換手法 (FFT) により行う．

ところで，中点公式による変換は，変換の対象となる偶関数 $x(t)$ を複素数値関数とみなし，それに対して項数 $2n$ の中点公式による離散型複素フーリエ変換を行えば得られる．しかしこの場合，複素変換の性質を利用すると効率よく変換が可能であることが知られている．

いま，周期 2π の複素数値関数 $x(t)$ に対する $2n$ (= 2^{l+1}, l = 0, 1, 2, ⋯) 項の中点公式による離散型フーリエ変換を (4.1) で定義する．

$$2na_k = \sum_{j=0}^{2n-1} x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right)\exp\left(-\frac{\pi i}{n}k\left(j+\frac{1}{2}\right)\right), \quad k = 0,1,...,2n-1 \tag{4.1}$$

FFT の基本は，入力データに対し少ない項数の適当な離散型フーリエ変換（2 基底ならば項数は 2）を繰り返し行い目的の変換を達成するところにある．

すなわち，j + 1 番目の標本から間隔 $\tilde{n}_p$, (= 2^{l+1-p}, P = 1, 2, ⋯) で，与えられた標本を間引きし，n_p = 2^p 項の変換，

$$x^p(j,k) = \sum_{r=0}^{n_p-1} x\left(\frac{\pi}{n}\left(r\tilde{n}_p + j + \frac{1}{2}\right)\right) \cdot \exp\left(-\frac{2\pi i}{n_p}\left(r + (j+\frac{1}{2}) \Big/ \tilde{n}_p\right)k\right), \quad j = 0,1,...,\tilde{n}_p - 1, \quad k = 0,1,...,n_p - 1 \tag{4.2}$$

を定義すれば，これは基底 2 の FFT の算法 (4.3) に従う．

初期値

$$\begin{aligned} &x^0(j,0) = x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right), \quad j = 0,1,...,2n-1 \\ &x^p(j,k) = x^{p-1}(j,k) + x^{p-1}\left(j+\tilde{n}_p,k\right) \\ &x^p\left(j,k+n_{p-1}\right) = \left\{x^{p-1}(j,k) - x^{p-1}\left(j+\tilde{n}_p,k\right)\right\} \cdot \exp\left(-\frac{\pi i}{\tilde{n}_p}\left(j+\frac{1}{2}\right)\right), \\ &j = 0,1,...,\tilde{n}_p - 1, k = 0,1,...,n_{p-1} - 1, \\ &p = 1,2,...,l+1 \end{aligned} \tag{4.3}$$

この漸化式の最終段階の値が離散型フーリエ係数 (4.4) である．

$$2na_k = x^{l+1}(0,k), \quad k = 0,1,...,2n-1 \tag{4.4}$$

ところで関数 $x(t)$ が偶関数ならば，$x(t)$ の実数性

$x(t) = \bar{x}(t)$ （$\bar{x}(t)$ は $x(t)$ の共役複素数）

及び対称性

$$x(2\pi - t) = \bar{x}(t)$$

は，FFT の中間結果 $\{x^p(j,k)\}$ に対して，次のように波及する．

実数性: $x^p(j, 0)$ は実数

$$\left.\begin{aligned} & x^p\left(j, n_p-k\right)=\bar{x}^p(j, k) \exp\left(-\frac{2 \pi i}{\tilde{n}_p}\left(j+\frac{1}{2}\right)\right), \\ & j=0,1,\ldots, \tilde{n}_p-1, \quad k=1,2,\ldots, n_p-1 \end{aligned}\right\}$$

対称性:

$$\left.\begin{aligned} & x^p\left(\tilde{n}_p-j-1, k\right)=\bar{x}^p(j, k), \\ & j=0,1,\ldots, \tilde{n}_p-1, \quad k=0,1,\ldots, n_p-1 \end{aligned}\right\} \tag{4.5}$$

一方，偶関数 $x(t)$ の複素フーリエ係数 $\{a_k\}$ と，離散型 cosine 変換

$$a_k=\frac{2}{n} \sum_{j=0}^{n-1} x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right) \cos \frac{\pi}{n} k\left(j+\frac{1}{2}\right) \quad k=0,1,\ldots, n-1 \tag{4.6}$$

で定義されるフーリエ係数の間には

$a_k=2a_k$, $k=0,1,\cdots,n-1$

が成立するので，FFT の算法 (4.3) において，性質 (4.5) を利用することにより，演算回数と使用する領域を 1/4 に減らすことができる．すなわち，(4.3) における j, k の範囲はそれぞれ半分となり，離散型 cosine 変換 (4.6) に対する基底 2 の FFT の算法は，(4.7) で表せる．

$$\begin{aligned} & x^p(j, k)=x^{p-1}(j, k)+\bar{x}^{p-1}\left(\tilde{n}_p-j-1, k\right) \\ & x^p\left(j, n_{p-1}-k\right)=\left\{\bar{x}^{p-1}(j, k)-x^{p-1}\left(\tilde{n}_p-j-1, k\right)\right\} \\ & \qquad \cdot \exp\left(-\frac{\pi i}{\tilde{n}_p}\left(j+\frac{1}{2}\right)\right) \\ & j=0,1,\ldots, \frac{\tilde{n}_p}{2}-1, k=0,1,\ldots, \frac{n_{p-1}}{2}-1 \\ & p=1,2,\ldots, l \end{aligned} \tag{4.7}$$

中間結果 $\{x^p(j, k)\}$ を格納するためには，大きさ n の 1 次元配列が必要であるが，入力データをあらかじめビット逆転によって並べ換えれば，上の計算を入力データの同一領域上で行うことができる．

n 項の cosine 変換のために必要な実数乗算回数は約 $n\log_2 n$ である．

a. 本サブルーチンにおける変換の手順

① 変換に必要な三角関数表（大きさ n-1）を，ビット逆転の順序に作成する．

② 標本 $\{x(\pi/n(j+1/2))\}$（大きさ n）をビット逆転の順序に並べ換える．

③ 入力データの対称性を利用した FFT (4.7) の計算を同一領域上で行い，フーリエ係数 $\{a_k\}$ を番号順に求める．

b. 逆変換の手順

① 逆変換に必要な三角関数表を作成する．変換①で用いる表と同じものである．

② 偶関数 $x(t)$ の n 個のフーリエ係数 $\{a_k\}$ を入力し，p に関する漸化式 (4.7) を逆にたどることにより関数値 $\{x(\pi/n(j+1/2))\}$ をビット逆転の順序に求める．

③ 得られた n 個のデータをビット逆転の順序に並べ変えることによって関数値 $\{x(\pi/n(j+1/2))\}$ を番号順に求める．

なお，詳細については，参考文献 [58] を参照すること．

F11-11-0101 FCOST, DFCOST

離散型 cosine 変換（台形公式，2 基底 FFT）
CALL FCOST (A, N, TAB, ICON)

(1) 機能

周期 2π の偶関数 $x(t)$ の半周期を n 等分した $n+1$ 個の標本 $\{x_j\}$,

$$x_j = x\left(\frac{\pi}{n} j\right), j = 0,1,...,n \qquad (1.1)$$

が与えられたとき，台形公式による離散型 cosine 変換，又はその逆変換を高速変換手法 (FFT) により行う.

ただし，$n = 2^l$（l: 0 又は，正整数）であること.

① cosine 変換

$\{x_j\}$ を入力し，(1.2) で定義する変換を行い，フーリエ係数 $\{n/2 \cdot a_k\}$ を求める.

$$\frac{n}{2} a_k = \sum_{j=0}^{n}{}'' x_j \cos\frac{\pi}{n} kj, \ k = 0,1,...,n \qquad (1.2)$$

ここで Σ" は，初項と末項を 1/2 倍して和をとることを意味する.

② cosine 逆変換

$\{a_k\}$ を入力し，(1.3) で定義する変換を行い，フーリエ級数の値 $\{x_j\}$ を求める.

$$x_j = \sum_{j=0}^{n}{}'' a_k \cos\frac{\pi}{n} kj, j = 0,1,...,n \qquad (1.3)$$

(2) パラメタ

A ····· **入力**. $\{x_j\}$ 又は $\{a_k\}$.
出力. $\{n/2\ a_k\}$ 又は $\{x_j\}$.
大きさ $n+1$ の 1 次元配列.
図 FCOST1-1 参照.

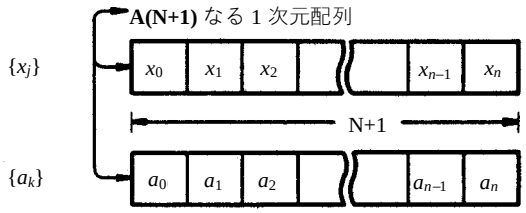
N ····· **入力**. 標本数 - 1.

TAB ····· **出力**. 変換で使用された三角関数表が格納される.
大きさ $n/2$-1 の 1 次元配列.
（使用上の注意③参照）

ICON····· **出力**. コンディションコード.
表 FCOST-1 参照.

表 FCOST-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	$N \neq 2^l$ (l: 0 又は，正整数）であった.	処理を打ち切る.



[注意] $\{\frac{n}{2} a_k\}$ は $\{a_k\}$ に準じる

図 FCOST-1 データの格納方法

(3) 使用上の注意

a. 使用する副プログラム

① SSL II···UPNR2，UTABT，UCOSM，MGSSL

② FORTRAN 基本関数···SQRT，MAX0

b. 注意

① 一般的なフーリエ変換の定義について

台形公式による離散型 cosine 変換及び，逆変換は，一般的に (3.1)，(3.2) で定義される.

$$a_k = \frac{2}{n}\sum_{j=0}^{n}{}'' x_j \cos\frac{\pi}{n} kj, k = 0,1,...,n \qquad (3.1)$$

$$x_j = \sum_{k=0}^{n}{}'' a_k \cos\frac{\pi}{n} kj, j = 0,1,...,n \qquad (3.2)$$

本サブルーチンでは，(3.1)，(3.2) の左辺に対応して $\{n/2 \cdot a_k\}$，$\{x_j\}$ を求めるので，結果の正規化は必要に応じて行うこと.

② 三角多項式の計算

逆変換により，n 次の三角多項式

$$x(t) = \frac{1}{2} a_0 + a_1 \cos t + ... + a_n \cos nt \qquad (3.3)$$

の値を $x\left(\frac{\pi}{n} j\right), j = 0,1,...,n,$ で求める場合は，最高次の係数 a_n は，あらかじめ 2 倍する必要がある.

使用例②を参照すること.

③ 三角関数表の扱いについて

同一項数の下で本サブルーチンを繰り返し呼ぶ場合，三角関数表の計算は最初の 1 回だけ行われる．したがって 2 回目以降，パラメタ TAB の内容は保存したまま呼び出すこと.

一方，変換の項数が異なる場合でも，三角関数表に不足が生じた場合だけ，その都度不足分を計算し表に追加するので効率がよい.

c. 使用例

① $n+1$ 個の標本 $\{x_j\}$ を入力し，本サブルーチンにより変換したのち正規化し，離散型フーリエ係数 $\{a_k\}$ を求める．引続き逆変換することにより，$\{x_j\}$ を求める.

$n \leq 512$ の場合.

```
C      **EXAMPLE**
       DIMENSION X(513),TAB(255)
       READ(5,500) N
       NP1=N+1
       READ(5,501) (X(I),I=1,NP1)
C      COSINE TRANSFORM
       WRITE(6,600) N
       WRITE(6,601) (X(I),I=1,NP1)
       CALL FCOST(X,N,TAB,ICON)
       IF(ICON.NE.0) GO TO 20
C      NORMALIZE
       CN=2.0/FLOAT (N)
       DO 10 K=1,NP1
       X(K)=X(K)*CN
    10 CONTINUE
       WRITE(6,602)
       WRITE(6,601) (X(I),I=1,NP1)
C      COSINE INVERSE TRANSFORM
       CALL FCOST(X,N,TAB,ICON)
       IF(ICON.NE.0) GO TO 20
       WRITE(6,602)
       WRITE(6,601) (X(I),I=1,NP1)
    20 WRITE(6,603) ICON
       STOP
   500 FORMAT(I5)
   501 FORMAT(6F12.0)
   600 FORMAT('0',5X,'INPUT DATA N=',I5)
   601 FORMAT(5F15.7)
   602 FORMAT('0',5X,'OUTPUT DATA')
   603 FORMAT('0',5X,'CONDITION CODE',
      *       I8)
       END
```

② 台形公式による cosine 係数 $\{a_k\}$ を入力し，n 次の三角多項式

$$x(t)=\frac{1}{2}a_0+a_1\cos t+\ldots+a_{n-1}\cos(n-1)t + a_n\cos nt$$

の標本点 $\{\pi j/n\}$ における値 $\{x(\pi j/n)\}$ を求める．この場合，末項の係数だけ 2 倍して入力すること.

$n \leq 512$ の場合.

```
C      **EXAMPLE**
       DIMENSION A(513),TAB(255)
       READ(5,500) N
       NP1=N+1
       READ(5,501) (A(K),K=1,NP1)
       WRITE(6,600) N
       WRITE(6,601) (A(K),K=1,NP1)
       A(NP1)=A(NP1)*2.0
       CALL FCOST(A,N,TAB,ICON)
       IF(ICON.NE.0) GO TO 20
       WRITE(6,602)
       WRITE(6,601) (A(K),K=1,NP1)
    20 WRITE (6,603) ICON
       STOP
   500 FORMAT(I5)
   501 FORMAT(6F12.0)
   600 FORMAT('0',5X,'INPUT DATA N=',I5)
   601 FORMAT(5F15.7)
   602 FORMAT('0',5X,'OUTPUT DATA')
   603 FORMAT('0',5X,'CONDITION CODE',
      *       I8)
       END
```

(4) 手法概要

項数 $n+1$（$=2^l+1$, $l = 0, 1, \cdots$）の台形公式による離散型 cosine 変換を，2 を基底とする高速変換手法 (FFT) により行う.

ところで，台形公式による変換は，中点公式による変換を利用して効率よく変換できる．本サブルーチンでは，この方法を採用している.

いま，周期 2π の偶関数 $x(t)$ の半周期 $[0, \pi]$ を n_p（$=2^P$, $p = 0,1, \cdots$）等分した場合の台形公式による離散型 cosine 変換を (4.1) で定義する.

$$a_k^p=\sum_{j=0}^{n_p}{}''x\left(\frac{\pi}{n_p}j\right)\cos\frac{\pi}{n_p}kj, k=0,1,\ldots,n_p \qquad (4.1)$$

また，項数 n_p の中点公式による離散型 cosine 変換を (4.2) で定義する.

$$\hat{a}_k^p=\sum_{j=0}^{n_p-1}x\left(\frac{\pi}{n_p}\left(j+\frac{1}{2}\right)\right)\cos\frac{\pi}{n_p}k\left(j+\frac{1}{2}\right), \quad k=0,1,\ldots,n_p-1 \qquad (4.2)$$

ただし，(4.1)，(4.2) では通常の正規化定数 $2/n_p$ は省略している.

ところで分割数を 2 倍に増し n_{p+1} としたとき $\{a_k^{p+1}\}$ は，項数がその半分の $\{a_k^p\}$ と $\{\hat{a}_k^p\}$ を用いて次のように表現できる.

$$\left.\begin{aligned} a_k^{p+1}&=a_k^p+\hat{a}_k^p \\ a_{n_{p+1}-k}^{p+1}&=a_k^p-\hat{a}_k^p \end{aligned}\right\} k=0,1,\ldots,n_p-1 \qquad (4.3)$$
$$a_{n_p}^{p+1}=a_{n_p}^p$$

この式は，$\{a_k^p\}$ から $\{a_k^{p+1}\}$ への p に関する漸化式とみることができる.

したがって，分割数が 2^l の変換を行う場合，初期条件 $\{a_k^1; k = 0, 1, 2\}$ を与え，各 p 段階（$p = 1, 2, \cdots, l-1$）で項数 n_p の中点公式による離散型 cosine 変換を行えば，(4.3) により台形公式による離散型 cosine 変換の列 $\{a_k^p; k = 0, 1, ..., n_p\}$（$p = 1, 2, \cdots, l$）が得られる.

実数の乗算回数は，約 $n\log_2 n$ ($n = 2^l$) であり，$\{a_k^l\}$ を求める上記の算法は高速となる.

本サブルーチンにおける手順

番号順に並べられた $n + 1$ 個の標本 x_0，x_1，…，x_n の初めの n 個をビット逆転によって並べ換え，これをあらためて $x(0)$，$x(1)$，…，$x(n)$ と表す．こうすることによって漸化式 (4.3) に必要なデータを順番にとり出せる．

次に変換に必要な三角関数表（大きさ $n/2 - 1$）を，標本の番号に対応してビット逆転の順序に作成する．

以上で予備の処理を終わり cosine 変換に移る．

① 初期条件の設定
3 個の標本 $x(0)$，$x(1)$，$x(n)$ を用い，$p = 1$ として $\{a_k^1; k = 0,1,2\}$ を求め，それぞれを $x(\cdot)$ の上に格納する．

② 2^p 項の中点公式による cosine 変換
n_p 個の標本 $x(n_p)$，$x(n_p + 1)$，…，$x(n_{p+1} - 1)$ を入力して中点公式による cosine 変換を行い $\{\hat{a}_k^p\}$ を同一領域上に求める．中点公式による cosine 変換については，サブルーチン FCOSM の手法概要参照．

③ $\{a_k^p\}$ に関する漸化式の計算
中間結果はすべて入力データの配列上に格納されるので補助的な作業領域を必要としない．すなわち p 段階の四つの要素 a_k^p，$a_{n_p-k}^p$，$\hat{a}_k^p$，$\hat{a}_{n_p-k}^p$ $(k = 0,1,\cdots, n_{p-1} - 1)$ から $p + 1$ 段階の四つの要素 a_k^{p+1}，$a_{n_p-k}^{p+1}$，$a_{n_p+k}^{p+1}$，$a_{n_{p+1}-k}^{p+1}$ を漸化式に従って計算し，p 段階の対応する要素の上に格納する．
手順②，③を $p = 1$，2，…，$l - 1$ と動かしながら繰り返すことにより $\{a_k\}$ を求める．

なお，詳細については参考文献 [58] を参照すること．

E51-20-0101 FSINF, DFSINF

奇関数の sine 級数展開 （関数入力，高速 sine 変換）
CALL FSINF (TH, FUN, EPSA, EPSR, NMIN, NMAX, B, N, ERR, TAB, ICON)

(1) 機能

任意の周期 $2T$ をもつ滑らかな奇関数 $f(t)$ を要求精度 ε_a，ε_r に基づき sine 級数展開する．すなわち，

$$\left| f(t) - \sum_{k=0}^{n-1} b_k \sin \frac{\pi}{T} kt \right| \le \max\left\{ \varepsilon_a, \varepsilon_r \| f \| \right\} \qquad (1.1)$$

を満たす n 個の係数 $\{b_k\}$ を求める．$\{b_k\}$ は自明の係数 $b_0 = 0$ を含むので，実質的な係数の個数は n - 1 である．

関数のノルム $\|f\|$ は，半周期 [0, T] での関数の標本点

$$t_j = \frac{T}{n-1} j,\ j = 0,1,...,n-1 \qquad (1.2)$$

における関数値を用いて

$$\|f\| \le \max_{0 \le j \le n-1} \left| f(t_j) \right| \qquad (1.3)$$

と定義する．

ただし， T > 0，$\varepsilon_a \ge 0$，$\varepsilon_r \ge 0$ であること．

(2) パラメタ

TH …… **入力**．関数 $f(t)$ の半周期 T.

FUN …… **入力**．展開しようとする関数 $f(t)$ を計算する関数副プログラム名（使用例参照）．

EPSA …… **入力**．級数展開の絶対誤差の上限 ε_a.

EPSR …… **入力**．級数展開の相対誤差の上限 ε_r.

NMIN …… **入力**．級数展開の項数 n の下限 (≥0).
2 のべきの値で与える．
標準値は 8.（使用上の注意③参照）．

NMAX … **入力**．級数展開の項数の上限 (≥NMIN)．2 のべきの値で与える．
標準値は 256.（使用上の注意③参照）．

B …… **出力**．係数 $\{b_k\}$.
大きさ NMAX の 1 次元配列．
B(1)=b_0，B(2)=b_1，…，B(N)=b_{n-1}のように格納される．

N …… **出力**．級数展開の項数 n (≥4).
2 のべきの値をとる．

ERR …… **出力**．級数展開の絶対誤差の推定値．

TAB …… **出力**．級数展開で使用された三角関数表が格納される．
大きさ 3 以上で，かつ NMAX/2 - 1 の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 FSINF-1 参照．

表 FSINF-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	丸め誤差のために要求精度が得られなかった．要求精度が厳しすぎる．	B には得られた係数を出力する．級数展開の誤差は達成可能な限度まで達している．
20000	級数展開の項数が，その上限に達しても要求精度が得られなかった．	処理を打ち切る．この時点で得られた係数と絶対誤差の推定値を，それぞれ B と ERR に出力する．
30000	次のいずれかであった． ①TH ≤ 0 ②EPSA < 0.0 ③EPSR < 0.0 ④NMIN < 0 ⑤NMAX < NMIN	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II … MGSSL, AMACH, UTABT, USINM, UNIFC

② FORTRAN 基本関数 … ABS, AMAX1, AMIN1, FLOAT

b. 注意

① パラメタ FUN に対応する関数副プログラムは，独立変数 t だけを引数にもつ関数副プログラムとして区間 [0, T] 上で定義し，本サブルーチンを呼び出す側のプログラムで，その関数名を EXTERNAL 文で宣言しなければならない．与えられた関数が助変数を含むときは，これらを COMMON 文で宣言することにより，呼び出す側のプログラムとの連絡を図る．（使用例参照）．

② 三角関数表の扱いについて
本サブルーチンを繰返し呼ぶ場合，必要な三角関数表は，1 回だけしか計算されない．すなわち，三角関数表に不足が生じた場合，その都度不足分だけ計算し表に追加される．
したがって，パラメタ TAB の内容は，常に保存したまま呼び出すこと．

③ NMIN，NMAX が 2 のべきの値でない場合，それらを超えない最大の 2 のべきの値とみなす．ただし NMAX < 4 の場合は，NMAX = 4 とみなす．

④ 展開の項数 n の増大とともに，誤差が小さくなる度合は，全区間 $(-\infty, \infty)$ における関数 $f(t)$ の滑らかさに著しく影響される．$f(t)$ が解析的周期関数ならば，誤差は指数関数のオーダ $O(r^n)$，$0 < r < 1$ で減少し，$f(t)$ が k 回連続微分可能ならば，有理式のオーダ $O(n^{-k})$ で減少する．$k = 0, 1$ の場合展開項数が多くなり，誤差の推定も正確さを期しがたい．したがって本サブルーチンを適用する関数は少なくとも 2 回連続微分可能であることが望ましい．

⑤ 級数の精度

本サブルーチンは，与えられた ε_a，ε_rに対して，(1.1) を満たす級数を求める．したがって，$\varepsilon_r = 0$ とおけば絶対誤差 ε_a 以内の，また $\varepsilon_a = 0$ とおけば相対誤差 ε_r 以内の精度で級数展開することになる．

しかし ε_a と ε_r の値によっては，この目的が達成されないことがある．例えば，関数の計算誤差に比べて，ε_a，ε_r が小さすぎる場合には，展開項数がその上限に達しない場合でも丸め誤差の影響が大きくなり，これ以上計算を続行しても無意味である．

このような状態を見出したときは ICON を 10000 として処理を打ち切る．このときの級数の精度は，使用計算機で達成可能な限度まで達している．また，関数の性質によって展開の項数が NMAX 以内では収束しないこともあり，この場合 ICON を 20000にして処理を打ち切る．このときの係数の値は，それまでに得られている近似値であり，級数の精度は保証されない．

しかしながら，いずれの場合でも級数の絶対誤差の推定値はパラメタ ERR に出力される．

⑥ サブルーチン FSINT を用い逆変換を行うことができる．順変換，逆変換いずれの場合もパラメタ TAB の内容は保存しておくこと．（使用例②参照）

⑦ 関数 $f(t)$ が単に周期関数であって，その sine 変換を計算する場合，奇関数 $(f(t) - f(-t))/2$ に対して本サブルーチンを適用すればよい．

⑧ 奇関数 $f(t)$ が周期をもたず絶対積分可能なとき，$f(t)$ の（理論的）sine 変換は，

$$F(\omega) = \int_0^\infty f(t) \sin \omega t dt \tag{3.1}$$

で定義される．$f(t)$ が $O(e^{-at})$，$a > 0$ で減衰するならば，次のようにして上のフーリエ積分の近似値を求めることができる．

まず T を大きくとり区間 $[T, \infty)$ において，$|f(t)|$ は無視できるものとする．例えば，u を丸め誤差の単位として，

$$|f(t)| < u,\ t \geq T \tag{3.2}$$

を満たすように T を定め，$f(t)$ を周期 $2T$ の関数とみなし，本サブルーチンによって $f(t)$ の sine 展開係数 $\{b_k\}$ を求める．ところで，

$$b_k = \frac{2}{T}\int_0^T f(t) \sin \frac{\pi}{T} kt\, dt \tag{3.3}$$

であるから，(3.1)，(3.2) より，

$$F\left(\frac{\pi}{T}k\right) \approx \frac{T}{2} a_k, k = 0,1,..., n-1 \tag{3.4}$$

が成り立つ．この関係を用いて離散型 sine 変換から，sine 変換 (3.1) の近似値を計算できる．

なお，逆変換

$$f(t) = \frac{2}{\pi}\int_0^\infty F(\omega) \sin \omega t d\omega \tag{3.5}$$

を計算する場合は，n 個のデータ

$$\frac{2}{T} F\left(\frac{\pi}{T}k\right), k = 0,1,..., n-1$$

を入力としてサブルーチン FSINT を呼べば，出力として，

$$f\left(\frac{T}{n}j\right), j = 0,1,..., n-1$$

の近似値が得られる．

（使用例②参照）．

c. 使用例

① 助変数 p を含む周期 2π の奇関数

$$f(t) = \frac{\sin t}{1 - 2p\cos t + p^2}$$

を要求精度 $\varepsilon_a = \varepsilon_r = 5 \cdot 10^{-5}$ 及び NMIN = 8, NMAX = 256 の下で sine 級数に展開する．

$f(t)$ の真の sine 級数展開は，

$$f(t) = \sum_{k=1}^{\infty} p^{k-1} \sin kt$$

である．p を 1/4，1/2，3/4 と変えて展開係数を出力する．

```
C     **EXAMPLE**
      DIMENSION B(256),TAB(127)
      EXTERNAL FUN
      COMMON P
      TH=ATAN(1.0)*4.0
      EPSA=0.5E-04
      EPSR=EPSA
      NMIN=8
      NMAX=256
      P=0.25
    1 CALL FSINF(TH,FUN,EPSA,EPSR,NMIN,
     *           NMAX,B,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GO TO 10
      WRITE(6,600) N,ERR,ICON,P
      WRITE(6,601) (B(I),I=1,N)
      P=P+0.25
      IF(P.LT.1.0) GO TO 1
      STOP
   10 WRITE(6,602) ICON
      STOP
  600 FORMAT('0',5X,'EXPANSION OF',
     * ' FUNCTION FUN(T)',3X,'N=',I4,
     * 5X,'ERR=',E15.5,5X,'ICON=',I6,5X,
     * 'P=', E15.5)
  601 FORMAT(/(5E15.5))
  602 FORMAT('0',5X,'CONDITION CODE',I8)
      END
      FUNCTION FUN(T)
      COMMON P
      FUN=SIN(T)/(1.0-2.0*P*COS(T)+P*P)
      RETURN
      END
```

② sine 変換およびその逆変換
無限区間における奇関数の sine 変換

$$F(\omega)=\int_0^{\infty} te^{-x^2}\sin\omega t\,d\,t$$

を $\varepsilon_a=\varepsilon_r=5\cdot10^{-5}$ の下で行い，解析解

$$F(\omega)=\frac{\sqrt{\pi}}{4}\omega\cdot e^{-\omega^2/4}$$

と比較する．引続きその逆変換,

$$\frac{2}{\pi}\int_0^{\infty}F(\omega)\sin\omega t\,d\omega$$

をサブルーチン FSINT を用いて行い，解析解と比べ精度を調べる．

```
C     **EXAMPLE**
      DIMENSION B(256),TAB(127),
     *          ARG(256),T(256)
      EXTERNAL FUN
      COMMON PI,SQPI
      PI=ATAN(1.0)*4.0
      SQPI=SQRT(PI)
      TH=SQRT(ALOG(4.0/AMACH(TH)))
      EPSA=0.5E-04
      EPSR=0.0
      NMIN=8
      NMAX=256
C     SINE TRANSFORM
      CALL FSINF(TH,FUN,EPSA,EPSR,NMIN,
     *           NMAX,B,N,ERR,TAB,ICON)
      IF(ICON.GT.10000) GO TO 10
      TQ=TH*0.5
      H=PI/TH
      DO 1 K=1,N
      ARG(K)=H*FLOAT(K-1)
      B(K)=B(K)*TQ
      T(K)=TRFN(ARG(K))
    1 CONTINUE
      WRITE(6,600) N,ERR,ICON
      WRITE(6,610)
      WRITE(6,601) (ARG(K),B(K),T(K),
     *              K=1,N)
C     INVERSE TRANSFORM
      Q=1.0/TQ
      DO 2 K=1,N
      B(K)=B(K)*Q
    2 CONTINUE
      CALL FSINT(B,N,TAB,ICON)
      IF(ICON.NE.0) GO TO 10
      H=TH/FLOAT(N)
      DO 3 K=1, N
      ARG(K)=H*FLOAT(K-1)
      T(K)=FUN(ARG(K))
    3 CONTINUE
      WRITE(6,620)
      WRITE(6,610)
      WRITE(6,601) (ARG(K),B(K),T(K),
     *              K=1,N)
      STOP
   10 WRITE(6,602)ICON
      STOP
  600 FORMAT('0',5X,' CHECK THE SINE',
     * ' TRANSFORM OF FUNCTION FUN(T)',
     * 3X,'N=',I4,5X,'ERR=',E15.5,5X,
     * 'ICON=',I5)
  610 FORMAT('0',6X,'ARGUMENT',7X,
     * 'COMPUTED',11X,'TRUE')
  620 FORMAT('0',5X,'CHECK THE INVERSE',
     * ' TRANSFORM')
  601 FORMAT(/(3E15.5))
  602 FORMAT('0',5X,'CONDITION CODE',I6)
      END
      FUNCTION FUN(T)
      FUN=T*EXP(-T*T)
      RETURN
      END
      FUNCTION TRFN(W)
      COMMON PI,SQPI
      TRFN=W*EXP(-W*W*0.25)*SQPI*0.25
      RETURN
      END
```

(4) 手法概要

本方法は，離散点入力の高速 sine 変換（台形公式）を関数入力の sine 変換に応用したものである．

a. sine 級数展開法
簡単のため奇関数 $f(t)$ の周期は 2π とする．
$f(t)$ の sine 級数は,

$$f(t)=\sum_{k=1}^{\infty}b_k\sin kt \tag{4.1}$$

$$b_k=\frac{2}{\pi}\int_0^{\pi}f(t)b_k\sin kt\,dt \tag{4.2}$$

と表される．積分区間 [0, π] を 2 等分，4 等分，8 等分，……と細分しながら (4.2) を台形公式で求める．得られた，3 個，5 個，9 個，……の係数 b_k により，(4.1) を有限の項数で近似する．被積分関数が滑らかならば，要求精度 ε_a，ε_r に応じて等分割数を倍々と増し，適当に大きく標本数 n（= (2 のべき) + 1）をとれば，

$$\left| f(t) - \sum_{k=0}^{n-1} b_k \sin kt \right| < \max\left\{ \varepsilon_a, \varepsilon_r \| f \| \right\} \qquad (4.3)$$

が満たされる．ただし，b_0=0.
得られた n - 1 次の三角多項式は，台形公式の各分点を補間点とする三角補間多項式になっている．すなわち，

$$f\left(\frac{\pi}{n} j\right) = \sum_{k=0}^{n-1} b_k \sin \frac{\pi}{n} kj, \quad j = 0,1,\ldots,n-1 \qquad (4.4)$$

となる．
上述したことを，更に詳しく説明する．
今，標本数 (n=n_p, n_p=2^p) の台形公式による係数

$$b_k^p = \sum_{j=0}^{n_p-1} f\left(\frac{\pi}{n_p} j\right) \sin \frac{\pi}{n_p} kj, \quad k = 1,2,\ldots,n_p - 1 \qquad (4.5)$$

が求められているとする．ここで正規化定数 2/n_p は省略している．
n_{p+1} = 2n_p として，項数を倍に拡張したときの係数 $\left\{b_k^{p+1}\right\}$ は，台形公式に対する中点公式の補完的性質により効率よく求められる．すなわち台形公式 (4.5) の標本点の中点で $f(x)$ を (4.6) のように標本化し，

$$f\left(\frac{\pi}{n_p}\left(j + \frac{1}{2}\right)\right), j = 0,1,\ldots,n_p - 1 \qquad (4.6)$$

これに対する離散型 sine 変換（中点公式）を (4.7) で定義する．

$$\tilde{b}_k^p = \sum_{j=0}^{n_p-1} f\left(\frac{\pi}{n_p}\left(j + \frac{1}{2}\right)\right) \sin \frac{\pi}{n_p} k\left(j + \frac{1}{2}\right), \quad k = 1,2,\ldots,n_p \qquad (4.7)$$

このとき，(4.8) の関係が成り立ち，$\left\{b_k^{p+1}\right\}$ を求めることができる．

$$\left.\begin{aligned} b_k^{p+1} &= b_k^p + \tilde{b}_k^p \\ b_{n_{p+1}-k}^{p+1} &= \tilde{b}_k^p - b_k^p \\ b_{n_p}^{p+1} &= \tilde{b}_{n_p}^{p+1} \end{aligned}\right\}, k = 1,2,\ldots,n_p - 1 \qquad (4.8)$$

この $\left\{b_k^p\right\}$ に関する漸化関係を用い，要求精度に応じ項数を倍々に増しながら $f(t)$ の高次の sine 級数を求めることができる．ただし，$\left\{b_k^p\right\}$ は，最後の段階で，2/n_p をかけられ正規化される．

b. sine 級数の誤差評価
$f(t)$ の理論的な sine 係数 b_k と p 段階の離散型 sine 係数 b_k^p の間には，それらの定義式 (4.2)，(4.5) 及び三角関数の選点直交関係より，

$$b_k^p = b_k + \sum_{m-1}^{\infty} \left(b_{2mn_p-k} + b_{2mn_p+k} \right), \quad k = 1,2,\ldots,n_p - 1 \qquad (4.9)$$

が成り立つ．これより p 段階の sine 級数の誤差評価

$$\left| f(t) - \sum_{k=0}^{n_p-1} b_k^p \sin kt \right| \le \left| b_{n_p} \right| + 2 \sum_{k=n_p+1}^{\infty} |b_k| \qquad (4.10)$$

が導かれる．
ところで，$f(t)$ が解析的周期関数のとき，その展開係数 $\{b_k\}$ は番号 k の増加に伴い指数関数的オーダ $O(r^k)$，$0 < r < 1$ で減少する．そこで p 段階の離散型 sine 係数から r を推定することができる．いま $b_k^p \approx Ar^k$（A はある定数）と表す．
k は高々 n_p - 1 だから末尾の番号 n_p - 1 と 3/4n_p - 1 の係数の比から $r^{n_p/4}$ を知ることができる．
本サブルーチンでは，偶然にそれらが 0 となると都合がわるいので n_p - 2 および 3/4n_p - 2 番の係数も併用して r を，

$$r = \min\left\{ \left(\frac{\left|b_{n_p-2}^p\right| + \left|b_{n_p-1}^p\right|}{\left|b_{\frac{3}{4}n_p-2}^p\right| + \left|b_{\frac{3}{4}n_p-1}^p\right|} \right)^{\frac{4}{n_p}}, 0.99 \right\}$$

で推定している．r の上限を 0.99 としたのは，これ以上 r が 1 に近くなれば，収束は極めて緩くなり事実上 $f(x)$ の sine 展開は不可能だからである．
さて，r が得られたならば (4.10) より p 段階の誤差は

$$e_p = \frac{r}{1-r}\left(\left|b_{n_p-2}^p\right| + \left|b_{n_p-1}^p\right| \right) \qquad (4.11)$$

で推定できる．

c. 計算の手順

手順 1 ……初期設定

① 三角関数表の初期値設定
区間 [0, π/2] を等分割した点上で cosine 関数の値をビット逆転の順序に 3 個だけ求める．本サブルーチンが既に呼ばれ，三角関数表が作成済みならば，この初期化の操作は省略される．三角関数表は，離散型 sine 変換のために用いられる．

② 初期の sine 級数展開
(4.5) において，$n_p = 4(p = 2)$ とした 4 項の sine 級数展開（実質的には 3 項）を行い，
$0.0, b_1^2, b_2^2, b_3^2$ を計算する．同時にノルムの定義 (1.3) に基づき $\|f\|$ を求める．

手順 2 ……収束判定

$n_p \le$ NMIN ならば，収束判定は省略し無条件に手順 3 に進む．$n_p >$ NMIN ならば以下の収束判定を行う．
まず計算誤差の限界

$$\rho = \sqrt{n_p}\,\|f\|(2u), \quad u \text{ は丸め誤差の単位} \qquad (4.12)$$

と収束判定定数，

$$\varepsilon = \max\{\varepsilon_a, \varepsilon_r \|f\|\} \qquad (4.13)$$

を求める．
p 段階の末尾 2 項の係数が有効桁を失っている場合，すなわち，

$$\left|b^p_{n_p-2}\right| + \frac{1}{2}\left|b^p_{n_p-1}\right| < \rho \qquad (4.14)$$

ならば，これ以上続行しても精度の向上は期待できないので収束したと判定し，求める級数の絶対誤差 e_p を計算誤差 ρで置き換える．$\rho < \varepsilon$ ならば ICON = 0 として，又は $\rho \ge \varepsilon$ ならば ε_a，ε_r が，丸め誤差の単位 u に比べ相対的に小さすぎるとみなし ICON = 10000 として処理を終わる．
次に (4.14) が成立しない通常の場合，誤差 e_p を (4.11) で推定し，

$$e_p < \varepsilon \qquad (4.15)$$

を満たすか否か判定する．満たされるならば，ICON = 0 として処理を終わる．満たされない場合で，かつ $2n_p \le$ NMAX ならば，手順 3 へ進む．そうでなければ，与えられた展開項数の上限に達しても要求精度は得られなかったとみなし，ICON = 20000 として処理を終わる．正常，異常にかかわらず，終了時には係数を正規化する．

手順 3 ……標本点の計算

p 段階において $f(x)$ を標本化するための標本点は，

$$t_j = \frac{\pi}{n_p}\left(j + \frac{1}{2}\right), j = 0,1,\ldots,n_p - 1$$

であるが，これをビット逆転の順序に，次の漸化式に従って求める．

$$\begin{aligned} &t_0 = \pi / 2^{p+1} \\ &t_{(2j+1)2^{p-l-1}} = t_{j2^{p-l}} + 2^{-p+l}\pi, \\ &j = 0,1,\ldots,2^l - 1, l = 0,1,\ldots,p-1, \end{aligned} \qquad (4.16)$$

ただし，$n_p = 2^p$ である．

手順 4 ……関数の標本化とノルムの計算

n_p 個の標本点 (4.16) の上で $f(t)$ の関数値を求め，標本点の上に重ね書きする．
また関数のノルムの定義 (1.3)に従い，ノルム $\|f\|$ を計算する．

手順 5 ……三角関数表の作成

次の手順 6 で必要な三角関数表を計算し保存する．三角関数の有効利用のため本サブルーチンが繰返し呼ばれる場合でも一度求めた表（またはその一部）は，二度と計算しない．

手順 6 ……離散型 sine 変換（中点公式）

手順 4 で求めた標本に対し，離散型 sine 変換を FFT の手法で行い，$\{\tilde{b}_k^p\}$ を求める．

手順 7 ……$\{b_k^{p+1}\}$ の計算

既に求められている $\{b_k^p\}$ と，手順 6 で得られた $\{\tilde{b}_k^p\}$ を，(4.8) によって合成し項数 $2n_p + 1$ の離散型 sine 展開係数 $\{b_k^{p+1}\}$ を得る．
p を 1 増して手順 2 にもどる．

計算の手間の大部分は，手順 4，6 で費される．関数の標本化の手間を除けば，n 項の sine 展開係数を求めるに必要な乗算回数は，約 $n\log_2 n$ である．
記憶領域の節約を図るため，標本点，標本及び展開係数は，1 次元配列 B において三重に重ね書きされる．
なお詳細については，参考文献 [59] を参照すること．
また離散点入力の sine 変換については，サブルーチンのFSINT，FSINM を参照のこと．

F11-21-0201 FSINM, DFSINM

離散型 sine 変換（中点公式，2 基底 FFT）
CALL FSINM (A, N, ISN, TAB, ICON)

(1) 機能

周期 2π の奇関数 $x(t)$ の半周期を等分割した n 個の標本 $\{x_{j+1/2}\}$,

$$x_{j+1/2} = x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right), j = 0,1,...,n-1 \qquad (1.1)$$

が与えられたとき，中点公式による離散型 sine 変換，又はその逆変換を高速変換手法 (FFT) により行う.

ただし， $n = 2^l$（l: 0 又は，正整数）であること.

① sine 変換

$\{x_{j+1/2}\}$ を入力し，(1.2) で定義する変換を行い，フーリエ係数 $\{n/2 \cdot b_k\}$ を求める.

$$\frac{n}{2}b_k = \sum_{j=0}^{n-1} x_{j+1/2} \sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right), k = 1,2,...,n \qquad (1.2)$$

② sine 逆変換

$\{b_k\}$ を入力し，(1.3) で定義する変換を行い，フーリエ級数の値 $\{x_{j+1/2}\}$ を求める.

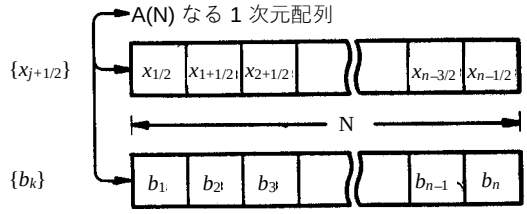
$$x_{j+1/2} = \sum_{k=0}^{n-1} b_k \sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right) + \frac{1}{2}b_n \sin\pi\left(j+\frac{1}{2}\right), \quad j = 0,1,...,n-1 \qquad (1.3)$$

(2) パラメタ

A ····· **入力**. $\{x_{j+1/2}\}$ 又は $\{b_k\}$.
出力. $\{n/2 \cdot b_k\}$ 又は $\{x_{j+1/2}\}$.
大きさ n の 1 次元配列.
図 FSINM-1 参照.

N ····· **入力**. 標本数 n.

ISN ····· **入力**. 変換か逆変換かを指定する（+1 又は -1）.
変換: ISN = +1
逆変換: ISN = -1



[注意] $\{\frac{n}{2}b_k\}$ は $\{b_k\}$ に準じる.

図 FSINM-1 データの格納方法

TAB ····· **出力**. 変換で使用された三角関数表が格納される.
大きさ n - 1 の 1 次元配列.
（使用上の注意③参照）

ICON ····· **出力**. コンディションコード.
表 FSINM-1 参照.

表 FSINM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	ISN ≠ 1，ISN ≠ -1 又は N ≠ 2^l（l: 0 又は正整数）であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II … UPNR2, UTABT, USINM，MGSSL

② FORTRAN 基本関数 … SQRT，MAX0

b. 注意

① 一般的なフーリエ変換の定義について

中点公式による離散型 sine 変換及び，その逆変換は，一般的に (3.1)，(3.2) で定義される.

$$b_k = \frac{2}{n}\sum_{j=0}^{n-1} x_{j+1/2} \sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right), \quad k = 1,2,...,n \qquad (3.1)$$

$$x_{j+1/2} = \sum_{k=1}^{n-1} b_k \sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right) + \frac{1}{2}\sin b_n \sin\pi\left(j+\frac{1}{2}\right), \quad j = 0,1,...,n-1 \qquad (3.2)$$

本サブルーチンでは，(3.1)，(3.2) の左辺に対応して $\{n/2 \cdot b_k\}$，$\{x_{j+1/2}\}$ を求めるので，結果の正規化は必要に応じて行うこと.

② 三角多項式の計算

逆変換により，n 次の三角多項式

$$x(t) = b_1 \sin t + b_2 \sin 2t + ... + b_n \sin nt$$

の値 $x(\pi/n(j+1/2))$ を求める場合は，最高次の係数 b_n はあらかじめ 2 倍する必要がある.
使用例②を参照すること.

③ 三角関数表の扱いについて

同一項数の下で本サブルーチンを繰り返し呼ぶ場合，三角関数表の計算は最初の 1 回だけ行われる．したがって 2 回目以降，パラメタ TAB の内容は保存したまま呼び出すこと.
一方，変換の項数が異なる場合でも，三角関数表に不足が生じた場合だけ，その都度不足分を計算し表に追加するので効率がよい.

c. 使用例

n 個の標本 $\{x_{j+1/2}\}$ を入力し，本サブルーチンにより変換したのち正規化し，離散型フーリエ係数 $\{b_k\}$ を求める．引続き逆変換することによって $\{x_{j+1/2}\}$ を求める．

$n \le 512$ の場合．

```
C      **EXAMPLE**
       DIMENSION X(512),TAB(511)
C      SINE TRANSFORM
       ISN=1
       READ(5,500) N,(X(I),I=1,N)
       WRITE(6,600) N
       WRITE(6,601) (X(I),I=1,N)
       CALL FSINM(X,N,ISN,TAB,ICON)
       IF(ICON.NE.0) GO TO 20
C      NORMALIZE
       CN=2.0/FLOAT(N)
       DO 10 K=1,N
       X(K)=X(K)*CN
    10 CONTINUE
       WRITE(6,602) ISN
       WRITE(6,601) (X(I),I=1,N)
C      SINE INVERSE TRANSFORM
       ISN=-1
       CALL FSINM(X,N,ISN,TAB,ICON)
       IF(ICON.NE.0) GO TO 20
       WRITE(6,602) ISN
       WRITE(6,601) (X(I),I=1,N)
    20 WRITE(6,603) ICON
       STOP
   500 FORMAT(I5/(6F12.0))
   600 FORMAT('0',5X,'INPUT DATA N=',I5)
   601 FORMAT(5F15.7)
   602 FORMAT('0',5X,'OUTPUT DATA',5X,
     *       'ISN=',I2)
   603 FORMAT('0',5X,'CONDITION CODE',
     *       I8)
       END
```

④ 中点公式による sine 係数 $\{b_k\}$ を入力し，n 次の三角多項式

$$x(t) = b_1 \sin t + b_2 \sin 2t + \ldots + b_n \sin nt$$

の標本点 $\{\pi(j+1/2)/n\}$ における値 $\{x(\pi(j+1/2)/n)\}$ を求める．この場合，末項の係数 b_n だけ 2 倍して入力すること．

$n \le 512$ の場合．

```
C      **EXAMPLE**
       DIMENSION B(512),TAB(511)
       ISN=-1
       READ(5,500) N,(B(I),I=1,N)
       WRITE(6,600) N
       WRITE(6,601) (B(I),I=1,N)
       B(N)=B(N)*2.0
       CALL FSINM(B,N,ISN,TAB,ICON)
       IF(ICON.NE.0) GO TO 20
       WRITE(6,602) ISN
       WRITE(6,601) (B(I),I=1,N)
    20 WRITE(6,603) ICON
       STOP
   500 FORMAT(I5/(6F12.0))
   600 FORMAT('0',5X,'INPUT DATA N=',I5)
   601 FORMAT(5F15.7)
   602 FORMAT('0',5X,'OUTPUT DATA',5X,
     *       'ISN=',I2)
   603 FORMAT('0',5X,'CONDITION CODE',
     *       I8)
       END
```

(4) 手法概要

項数 n（$= 2^l$, l=0, 1, …）の中点公式による離散型 sine 変換を，2 を基底とする高速変換手法 (FFT) により行う．

ところで，中点公式による変換は，変換の対象となる奇関数 $x(t)$ を複素数値関数とみなし，それに対して項数 $2n$ の中点公式による離散型複素フーリエ変換を行えば得られる．しかしこの場合，複素変換の性質を利用すると効率よく変換が可能であることが知られている．

いま，周期 2π の複素数値関数 $x(t)$ に対する $2n$（$=2^{l+1}$, l=0, 1, 2, …）項の中点公式による離散型フーリエ変換を (4.1) で定義する．

$$2n\alpha_k = \sum_{j=0}^{2n-1} x\left(\frac{\pi}{n}k\left(j+\frac{1}{2}\right)\right)\exp\left(-\frac{\pi i}{n}k\left(j+\frac{1}{2}\right)\right), \quad k = 0,1,\ldots,2n-1 \tag{4.1}$$

FFT の基本は，入力データに対し少ない項数の適当な離散型フーリエ変換（2 基底ならば項数は 2）を繰り返し行い目的の変換を達成するところにある．

すなわち，$j+1$ 番目の標本から間隔 $\tilde{n}_p \left(= 2^{l+1-p}, p = 1, 2, \ldots\right)$ で，与えられた標本を間引きし，$n_p = 2^p$ 項の変換，

$$x^p(j,k) = \sum_{j=0}^{n_p-1} x\left(\frac{\pi}{n}\left(r\tilde{n}_p + j + \frac{1}{2}\right)\right) \cdot \exp\left(-\frac{2\pi i}{n_p}\left(r + \left(j+\frac{1}{2}\right)/\tilde{n}_p\right)k\right)$$
$$j = 0,1,\ldots,\tilde{n}_p - 1, k = 0,1,\ldots n_p - 1 \tag{4.2}$$

を定義すれば，これは基底 2 の FFT の算法 (4.3) に従う．

初期値

$$x^0(j,0) = x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right), \quad j = 0,1,\ldots,2n-1$$

$$x^p(j,k) = x^{p-1}(j,k) + x^{p-1}\left(j+\tilde{n}_p, k\right)$$
$$x^p\left(j, k+n_{p-1}\right) = \left\{x^{p-1}(j,k) - x^{p-1}\left(j+\tilde{n}_p,k\right)\right\}\cdot\exp\left(-\frac{\pi i}{\tilde{n}_p}\left(j+\frac{1}{2}\right)\right)$$
$$j = 0,1,\ldots,\tilde{n}_p - 1, k = 0,1,\ldots,n_{p-1} - 1$$
$$p = 1,2\ldots,l-1 \tag{4.3}$$

この漸化式の最終段階の値が離散型フーリエ係数 (4.4) である.

$$2n\alpha_k = x^{l+1}(0,k), \quad k = 0,1,\ldots,2n-1 \tag{4.4}$$

ところで関数 $x(t)$ が奇関数ならば，$x(t)$ の実数性

$$x(t) = \bar{x}(t) \quad （\bar{x}(t) \text{ は } x(t) \text{ の共役複素数）}$$

及び歪対称性

$$x(2\pi - t) = -\bar{x}(t)$$

は，FFT の中間結果 $\{x^p(j, k)\}$ に対して，次のように波及する.

実数性: $x^p(j, 0)$ は実数

歪対称性

$$\left.\begin{aligned} &x^p\left(j, n_p - k\right) = \bar{x}^p\left(j, k\right)\exp\left(-\frac{2\pi i}{\tilde{n}_p}\left(j+\frac{1}{2}\right)\right), \\ &\quad j = 0,1,\ldots,\tilde{n}_p - 1, \quad k = 1, 2,\ldots,n_p - 1 \\ &x^p\left(\tilde{n}_p - j - 1, k\right) = -\bar{x}^p\left(j, k\right) \\ &j = 0,1\ldots,\tilde{n}_p - 1, \quad k = 0,1,\ldots,n_p - 1 \end{aligned}\right\} \tag{4.5}$$

一方，奇関数 $x(t)$ の複素フーリエ係数 $\{a_k\}$ と，離散型 sine 変換

$$b_k = \frac{2}{n}\sum_{j=0}^{n-1} x\left(\frac{\pi}{n}\left(j+\frac{1}{2}\right)\right)\sin\frac{\pi}{n}k\left(j+\frac{1}{2}\right) \qquad k = 1,2,\cdots, n \tag{4.6}$$

で定義されるフーリエ係数の間には

$$b_k = 2i\alpha_k, \quad k = 1,2,\ldots,n$$

が成立するので，FFT の算法 (4.3) において，性質 (4.5) を利用することにより，演算回数と使用する領域を 1/4 に減らすことができる．すなわち，(4.3) における j, k の範囲はそれぞれ半分となり，離散型 sine 変換 (4.6) に対する基底 2 の FFT の算法は，(4.7) で表せる.

$$\begin{aligned} &x^p\left(j, k\right) = x^{p-1}\left(j, k\right) - \bar{x}^{p-1}\left(\tilde{n}_p - j - 1, k\right) \\ &x^p\left(j, n_{p-1} - k\right) = \left\{\bar{x}^{p-1}\left(j, k\right) + x^{p-1}\left(\tilde{n}_p - j - 1, k\right)\right\} \\ &\qquad \cdot\exp\left(-\frac{\pi i}{\tilde{n}_p}\left(j+\frac{1}{2}\right)\right) \\ &j = 0,1,\ldots,\frac{\tilde{n}_p}{2} - 1, \quad k = 0,1\ldots,\frac{n_{p-1}}{2} - 1 \\ &p = 1, 2,\ldots,l \end{aligned} \tag{4.7}$$

中間結果 $\{x^p(j, k)\}$ を格納するためには，大きさ n の 1 次元配列が必要であるが，入力データをあらかじめビット逆転によって並べ換えれば，上の計算を入力データと同一領域上で行うことができる.

n 項の sine 変換のために必要な実数乗算回数は約 $n\log_2 n$ である.

a. 本サブルーチンにおける変換の手順

① 変換に必要な三角関数表（大きさ n - 1）を，ビット逆転の順序に作成する.

② 標本 $\{x(\pi/n(j+1/2))\}$（大きさ n）をビット逆転の順序に並べ換える.

③ 入力データの歪対称性を利用した FFT (4.7) の計算を同一領域上で行い，フーリエ係数を b_n，b_{n-1}，…，b_1 の順に求める.

④ フーリエ係数 $\{b_k\}$ を番号順に並べ換える.

b. 逆変換の手順

① 逆変換に必要な三角関数表を作成する．変換①で用いる表と同じものである.

② フーリエ係数 $\{b_k\}$ を，b_n，b_{n-1}，…，b_1 の順に並び換え，$\{b_{n-k}\}$ を得る.

③ $\{b_{n-k}\}$ を入力し，p に関する漸化式 (4.7) を逆にたどることにより奇関数 $x(t)$ の関数値 $\{x(\pi/n(j+1/2))\}$ をビット逆転の順序に求める.

④ 得られた n 個のデータをビット逆転の順序に並べ変えることによって関係値 $\{x(\pi/n(j+1/2))\}$ を番号順に求める.

なお，詳細については，参考文献 [58] を参照すること.

F11-21-0101, FSINT, DFSINT

離散型 sine 変換（台形公式，2 基底 FFT）
CALL FSINT (A, N, TAB, ICON)

(1) 機能

周期 2π の奇関数 $x(t)$ の半周期を n 等分した n 個の標本 $\{x_j\}$

$$x_j = x\left(\frac{\pi}{n} j\right), \quad j = 0,1,...,n-1 \tag{1.1}$$

が与えられたとき，台形公式による離散型 sine 変換，又はその逆変換を高速変換手法 (FFT) により行う．

ただし n = 2^l（l: 正整数）であること．

① sine 変換

$\{x_j\}$ を入力し，(1.2) で定義する変換を行い，フーリエ係数 $\{n/2 \cdot b_k\}$ を求める．

$$\frac{n}{2} b_k = \sum_{j=0}^{n-1} x_j \sin\frac{\pi}{n} kj, \quad k = 0,1,...,n-1 \tag{1.2}$$

ただし $x_0 = 0$.

② sine 逆変換

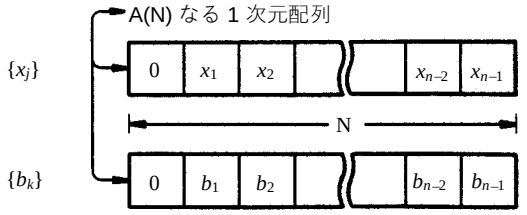
$\{b_k\}$ を入力し，(1.3) で定義する変換を行い，フーリエ級数の値 $\{x_j\}$ を求める．

$$x_j = \sum_{k=0}^{n-1} b_k \sin\frac{\pi}{n} kj, \quad j = 0,1,...,n-1$$

ただし $b_0 = 0$. (1.3)

(2) パラメタ

A ····· **入力**．$\{x_j\}$ 又は $\{b_k\}$.
出力．$\{n/2\ b_k\}$ 又は $\{x_j\}$.
大きさ n の 1 次元配列.
図 FSINT-1 参照.



[注意] $\{\frac{n}{2} b_k\}$ は $\{b_k\}$に準じる.

図 FSINT-1 データの格納方法

N ····· **入力**．標本数 n.
TAB ····· **出力**．変換で使用された三角関数表が格納される．
大きさ $n/2-1$ の 1 次元配列.
（使用上の注意②参照）
ICON ····· **出力**．コンディションコード.
表 FSINT-1 参照.

表 FSINT-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N ≠ 2^l（l: 正整数）であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ··· UPNR2, UTABT, USINM, MGSSL

② FORTRAN 基本関数 ··· SQRT and MAX0

b. 注意

① 一般的なフーリエ変換の定義について

台形公式による離散型 sine 変換及び，その逆変換は，一般的に (3.1)，(3.2) で定義される．

$$b_k = \frac{2}{n} \sum_{j=0}^{n-1} x_j \sin\frac{\pi}{n} kj, \quad k = 1,2,...,n-1 \tag{3.1}$$

$$x_j = \sum_{j=0}^{n-1} b_k \sin\frac{\pi}{n} kj, \quad j = 1,2,...,n-1 \tag{3.2}$$

本サブルーチンでは，(3.1)，(3.2) の左辺に対応して，$\{n/2 \cdot b_k\}$，$\{x_j\}$ を求めるので結果の正規化は必要に応じて行うこと．

② 三角関数表の扱いについて

同一項数の下で本サブルーチンを繰り返し呼ぶ場合，三角関数表の計算は最初の 1 回だけ行われる．したがって 2 回目以降，パラメタ TAB の内容は保存したまま呼び出すこと．

一方，変換の項数が異なる場合でも，三角関数表に不足が生じた場合だけ，その都度不足分を計算し表に追加するので効率がよい．

c. 使用例

n 個の標本 $\{x_j\}$ を入力し，本サブルーチンにより変換したのち正規化し，離散型フーリエ係数 $\{b_k\}$ を求める．引続き逆変換することにより，$\{x_j\}$ を求める．

$n \le 512$ の場合.

```
C       **EXAMPLE**
        DIMENSION X(512),TAB(255)
C       SINE TRANSFORM
        READ(5,500) N,(X(I),I=1,N)
        WRITE(6,600) N
        WRITE(6,601) (X(I),I=1,N)
        CALL FSINT(X,N,TAB,ICON)
        IF(ICON.NE.0) GO TO 20
C       NORMALIZE
```

```
      CN=2.0/FLOAT(N)
      DO 10 K=1,N
   10 X(K)=X(K)*CN
      WRITE(6,602)
      WRITE(6,601)(X(I),I=1,N)
C     SINE INVERSE TRANSFORM
      CALL FSINT(X,N,TAB,ICON)
      IF(ICON.NE.0) GO TO 20
      WRITE(6,602)
      WRITE(6,601) (X(I),I = 1,N)
   20 WRITE(6,603) ICON
      STOP
  500 FORMAT(I5/(6F12.0))
  600 FORMAT('0',5X,'INPUT DATA N=',I5)
  601 FORMAT(5F15.7)
  602 FORMAT('0',5X,'OUTPUT DATA')
  603 FORMAT('0',5X,'CONDITION CODE',
     *       I8)
      END
```

(4) 手法概要

項数 n (= 2^l, l = 1, 2, ⋯) の台形公式による離散型 sine 変換を，2 を基底とする高速変換手法 (FFT) により行う．

ところで，台形公式による変換は，中点公式による変換を利用して効率よく変換できる．本サブルーチンでは，この方法を採用している．

いま，周期 2πの奇関数 $x(t)$ の半周期 $[0, \pi]$ を n_p (=2^p, p = 1,2, ⋯) 等分した場合の台形公式による離散型 sine 変換を (4.1) で定義する．

$$b_k^p = \sum_{j=0}^{n_p-1} x\left(\frac{\pi}{n_p} j\right) \sin\frac{\pi}{n_p} kj, \quad k = 1,2,...,n_p - 1 \tag{4.1}$$

一方，項数 n_p の中点公式による離散型 sine 変換を (4.2) で定義する．

$$\hat{b}_k^p = \sum_{j=0}^{n_p-1} x\left(\frac{\pi}{n_p} k\left(j+\frac{1}{2}\right)\right) \sin\frac{\pi}{n_p} k\left(j+\frac{1}{2}\right), \quad k = 1,2,...,n_p \tag{4.2}$$

ただし，(4.1)，(4.2) では通常の正規化定数 $2/n_p$ は省略している．

ところで分割数を 2 倍に増し n_{p+1} としたとき $\{b_k^{p+1}\}$ は，項数がその半分の $\{b_k^p\}$ と $\{\hat{b}_k^p\}$ を用いて次のように表現できる．

$$\left.\begin{aligned} b_k^{p+1} &= \hat{b}_k^p + b_k^p \\ b_{n_{p+1}-k}^{p+1} &= \hat{b}_k^p + b_k^p \end{aligned}\right\} k = 1,2,..., n_p - 1$$
$$b_{n_p}^{p+1} = \hat{b}_{n_p}^p \tag{4.3}$$

この式は，$\{b_k^p\}$ から $\{b_k^{p+1}\}$ への p に関する漸化式とみなすことができる．

したがって分割数が 2^l の変換を行う場合，初期条件として $b_1^1 = x(\pi/2)$ を与え，各 p 段階（p=1, 2, ⋯ , l-1）で項数 n_p の中点公式による離散型 sine 変換を行えば，(4.3) より台形公式による離散型 sine 変換の列 $\{b_k^p; k = 1, 2, \cdots, n_{p-1}\}$ (p = 1, 2, ⋯, l) が得られる．実数乗算回数は約 $n\log_2 n$ ($n = 2^l$) である．

本サブルーチンにおける手順

番号順に並べられた n 個の標本 x_0，x_1，⋯，x_{n-1} をビット逆転によって並べ換え，これをあらためて $x(0)$，$x(1)$，⋯，$x(n\text{-}1)$ と表す．ただし $x(0) = x_0 = 0$.

次に変換に必要な三角関数表（大きさ $n/2$ - 1）を標本の番号に対応してビット逆転の順序に作成する．

以上で予備の処理を終わり sine 変換に移る．

① 初期条件の設定
$b_1^1 = x(1)$，$p = 1$ とおく．

② n_p 項の中点公式による sine 変換
n_p 個の標本 $x(n_p)$，$x(n_{p+1})$，⋯，$x(n_{p+1}\text{-}1)$ を入力し，中点公式による sine 変換を行い，$\{\hat{b}_k^p\}$ を $\hat{b}_{n_p}^p, \hat{b}_{n_{p-1}}^p, ..., \hat{b}_1^p$ の順序で同一領域上に求める．
中点公式による sine 変換についてはサブルーチン FSINM 手法概要の項参照．

③ $\{b_k^p\}$ に関する漸化式の計算
p 段階においては $x(0)$，$x(1)$，⋯，$x(n_{p+1}\text{-}1)$ の上に，中間結果 $x(0), b_1^p, b_2^p, ... b_{n_p-1}^p, \hat{b}_{n_p}^p, ..., \hat{b}_1^p$ が格納されている．この領域の上だけで $\{b_k^{p+1}\}$ を求める．
すなわち，p 段階の二つの要素 b_k^p，$\hat{b}_k^p (k = 1,2,..., n_p - 1)$ から，p + 1 段階の二つの要素 b_k^{p+1}，$b_{n_{p+1}-k}^{p+1}$ が漸化式 (4.3) にしたがって計算され，p 段階の対応する要素の上に，それらを格納する．
手順②，③を p = 1，2，⋯，l - 1 と動かしつつ繰り返すことにより $\{b_k\}$ を求める．

なお，詳細については参考文献 [58] を参照すること．

1B52-11-0101 GBSEG, DGBSEG

実対称バンド行列の一般固有値及び固有ベクトル（ジェニングス法）
CALL GBSEG (A, B, N, NH, M, EPSZ, EPST, LM, E, EV, K, IT, VW, ICON)

(1) 機能

次数 n，バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ と $\boldsymbol{B}$ とに対する一般固有値問題

$$Ax = \lambda Bx \qquad (1.1)$$

の，絶対値の大きい方から又は小さい方からの m 個の固有値及びそれらに対応する固有ベクトルを，m 個の与えられた初期ベクトルをもとにして，ジェニングスのベクトル加速法を伴う，ジェニングスの同時反復法によって求める．ただし，絶対値の大きい方から求めるときは $\boldsymbol{B}$ が，小さい方から求める時は $\boldsymbol{A}$ が正値行列でなければならない．なお固有ベクトルは，それぞれの場合に応じて，

$$\boldsymbol{X}^T\boldsymbol{B}\boldsymbol{X} = \boldsymbol{I} \qquad (1.2)$$

又は，

$$\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X} = \boldsymbol{I} \qquad (1.3)$$

の意味で正規化されている．$1 \le m << n$ ，

$0 \le h << n$ であること．

(2) パラメタ

A …… **入力**．実対称バンド行列 $\boldsymbol{A}$．絶対値の小さい方から求めるときには，演算後，その内容は保存されない．
対称バンド行列用圧縮モード．
大きさ $n(h+1) - h(h+1)/2$ の 1 次元配列．

B …… **入力**．実対称バンド行列 $\boldsymbol{B}$．演算後，内容は保存されない．
対称バンド行列用圧縮モード．
大きさ $n(h+1) - h(h+1)/2$ の 1 次元配列．
（使用上の注意①参照）．

N …… **入力**．行列 $\boldsymbol{A}$，$\boldsymbol{B}$ の次数 n．

NH …… **入力**．行列 $\boldsymbol{A}$，$\boldsymbol{B}$ のバンド幅 h．
（使用上の注意②参照）．

M …… **入力**．求める固有値及び固有ベクトルの個数 m．
M = m のときは絶対値の大きい方から求める．
M = $-m$ のときは絶対値の小さい方から求める・

EPSZ …… **入力**．$\boldsymbol{B}$ 又は $\boldsymbol{A}$ の LL^T 分解におけるピボットの相対零判定値．零又は負のときは標準値が設定される（使用上の注意③参照）．

EPST …… **入力**．固有ベクトルの収束判定に用いられる定数 **ε**．零又は負のときは標準値が設定される．
（使用上の注意④参照）．

LM …… **入力**． 反復回数の上限．反復回数がこれを超えると処理を中断する．
（使用上の注意⑤参照）．

E …… **出力**．固有値．指定された順序に格納される．大きさ m の 1 次元配列．

EV …… **入力**． 始めの m 列に列方向に格納された m 個の初期ベクトル．
（使用上の注意⑥参照）．
出力．固有ベクトル．始めの m 列に列方向に格納される．大きさ EV (K, m+2) の 2 次元配列．

K …… **入力**．EV の整合寸法．

IT …… **出力**．固有値固有ベクトルが求まるまでの反復回数

VW …… **作業領域**．大きさ $2n+m(3m+1)/2$ 以上の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 GBSEG-1 参照．

表 GBSEG-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
20000	反復回数が上限 LM を超えた．	処理を終了する．E と EV には，そのときまでに得られた固有値と固有ベクトルの近似値が入っている．
25000	反復ごとの固有ベクトルの直交化処理が不可能となった．	処理を打ち切る．
28000	行列 B 又は A が正値行列でない．	処理を打ち切る．
29000	行列 B 又は A が正則行列でない．	処理を打ち切る．
30000	NH < 0, NH ≥ N, N > K, M = 0 又は |M| > N であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MSBV, TRID1, TEIG1, TRBK, UCHLS, UBCHL, UBCLX, UESRT, MGSSL

② FORTRAN 基本関数 ... IABS, ABS, AMAX1, FLOAT, SQRT

b. 注意

① 絶対値の小さい方から求めた場合，行列 $\boldsymbol{B}$ の内容は配列 $\boldsymbol{A}$ に保存される．よって，行列 $\boldsymbol{B}$ が同一な複数組の一般固有値問題を続けて扱う場合には，これを利用できる．

② $\boldsymbol{A}$ と $\boldsymbol{B}$ のバンド幅は同一であること．もし，$\boldsymbol{A}$ と $\boldsymbol{B}$ のバンド幅が異なる場合は大きいバンド幅に合わせるために，パラメタ $\boldsymbol{A}$ 若しくは $\boldsymbol{B}$ へ拡大された分だけ零を追加すること．

③ パラメタ EPSZ の標準値は，丸め誤差の単位 u としたとき，$16 \cdot u$ である．
本サブルーチンにおいて EPSZ に 10^{-s} を設定し

たとすると，行列 ***B*** 又は ***A*** の LL^T 分解の過程でピボットの値が 10 進 s 桁以上の桁落ちを生じた場合に，そのピボットを零と見なしコンディションコードを設定し(ICON=29000)処理を打ち切る．なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ に非常に小さな値(例えば 10^{-70})を与えればよい．

行列 ***B*** 又は ***A*** の LL^T 分解の過程でピボットが負となった場合，行列を非正則であると見なしコンディションコードを設定し(ICON= 28000)処理を打切る．

④ パラメタ EPST は $\|x\|_2 = 1$ の意味で正規化された固有ベクトルの各成分の収束を判定するために用いられる．固有ベクトルが判定定数 ε に対して収束したとき，固有値は少なくとも $\|A\| \cdot \varepsilon$ の精度で，そして多くの場合よりも高い精度で収束している．このことを考慮して幾分大きめに EPST を選ぶのがよい．丸め誤差の単位を u とするとき，標準値は $16 \cdot u$ である．しかし，固有値が密集した問題では，収束が得られないことがあるので，そのような場合には $\varepsilon \geq 100u$ ぐらいにとるのが安全である．

⑤ 反復回数の上限 LM は，何らかの理由で収束が達成されないとき，強制的に反復を中断するためのものである．要求精度や固有値の密集の度合などを考慮して定められるべきものであるが，500～1000 あたりが妥当な標準値である．

⑥ 初期固有ベクトルは，求めようとする固有値に対応する固有ベクトルの良い近似であることが望ましい．しかしそのような近似ベクトルが得られないときには，単位行列 ***I*** の始めの m 個の列ベクトルを初期ベクトルとして採用するのが標準的である．固有値及び固有ベクトルの個数 m は，n に比べて小さく，例えば $m/n < 1/10$ 程度にとるのがよい．固有値を絶対値の大きい方から，あるいは小さい方から番号をつけて，$\lambda_1, \lambda_2, ..., \lambda_n$ とするとき，もし選んだ m が $|\lambda_{m-1}/\lambda_m| << 1$ あるいは $|\lambda_{m+1}/\lambda_m| >> 1$ を満足するならば，収束が速い．

c. 使用例

次数 n，バンド幅 h の実対称行列 ***A*** と ***B*** とに対する，一般固有値問題 ***Ax*** = λ***Bx*** の固有値と固有ベクトルを求める．

$n \leq 100, h \leq 10, m \leq 10$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(1100),B(1100),E(10),
     *          EV(100,12),VW(400)
   10 READ(5,500) N,NH,M,EPSZ,EPST
      IF(N.EQ.0) STOP
      MM=IABS(M)
      NN=(NH+1)*(N+N-NH)/2
      READ(5,510) (A(I),I=1,NN),
     *            (B(I),I=1,NN),
     * ((EV(I,J),I=1,N),J=1,MM)
      WRITE(6,600) N,NH,M
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=MIN0(NH+1,I)+NE
   20 WRITE (6,610) I,(A(J),J=NI,NE)
      WRITE(6,620) N,NH,M
      NE=0
      DO 30 I=1,N
      NI=NE+1
      NE=MIN0(NH+1,I)+NE
   30 WRITE(6,610) I,(B(J),J=NI,NE)
      CALL GBSEG(A,B,N,NH,M,EPSZ,EPST,500,
     *     E,EV,100,IT,VW,ICON)
      WRITE(6,630) ICON,IT
      IF(ICON.GE.20000) GO TO 10
      CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(3I5,2E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',20X,'ORIGINAL MATRIX A',
     * 5X,'N=',I3,5X,'NH=',I3,5X,'M=',I3/)
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT('0',20X,'ORIGINAL MATRIX B',
     * 5X,'N=',I3,5X,'NH=',I3,5X,'M=',I3/)
  630 FORMAT('0',20X,'ICON=',I5,
     * 5X,'IT=',I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン SEIG1 の使用例を参照のこと．

(4) 手法概要

次数 n，バンド幅 h の実対称バンド行列 ***A*** と ***B*** とに対する一般固有値問題

$$\boldsymbol{Ax} = \lambda \boldsymbol{Bx} \tag{4.1}$$

の，絶対値の大きい方から又は小さい方からの m 個の固有値とそれらに対応する固有ベクトルを，m 個の与えられた初期ベクトルをもとにして，ジェニングスのベクトル加速を伴う，ジェニングスの同時反復法によって求める．

ジェニングスの同時反復法の詳細についてはサブルーチン BSEGJ を参照すること．

a. 本サブルーチンにおける計算手順

以下に本サブルーチンの手順を示す．

なお，小さい方から m 個の固有値を求める時には，(4.1)を

$$\boldsymbol{Bx} = \frac{1}{\lambda}\boldsymbol{Ax} = \mu \boldsymbol{Ax} \tag{4.2}$$

と変形し (4.2) について絶対値の大きい方から m 個の固有値 $\mu_1, \mu_2, ..., \mu_m$ を求め，

$$\lambda_i = 1/\mu_i, \quad i = 1,...,m$$

とすれば良い．

以上のことを念頭において，以後の説明は，絶対値の大きい方から求める場合についてのみ行う．

① サブルーチン UBCHL を用いて ***B*** を LL^T 分解して

$$\boldsymbol{B} = \tilde{\boldsymbol{B}}\tilde{\boldsymbol{B}}^{T} \tag{4.3}$$

とする．これは一般固有値問題 (4.1)を，標準問題

$$\tilde{\boldsymbol{B}}^{-1}\boldsymbol{A}\tilde{\boldsymbol{B}}^{-T}\boldsymbol{u} = \lambda\boldsymbol{u} \tag{4.4}$$

に変換するためのものである．ただし，u は

$$\boldsymbol{u} = \tilde{\boldsymbol{B}}^{T}\boldsymbol{x} \tag{4.5}$$

である．

② m 個の近似固有ベクトル $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_m$ を m 個の列とし，n 行 m 列の行列を U とする．これらのベクトルは正規直交系をつくるものと仮定する．すなわち，

$$\boldsymbol{U}^{T}\boldsymbol{U} = \boldsymbol{I}_m \tag{4.6}$$

ただし，$\boldsymbol{I}_m$ は m 次の単位行列である．
$\boldsymbol{U}$ に $\tilde{\boldsymbol{B}}^{-T}$，$\boldsymbol{A}$，$\tilde{\boldsymbol{B}}^{-1}$ をこの順に左から乗じて

$$\boldsymbol{V} = \tilde{\boldsymbol{B}}^{-1}\boldsymbol{A}\tilde{\boldsymbol{B}}^{-T}\boldsymbol{U} \tag{4.7}$$

とする．次に $\boldsymbol{V}$ の左から $\boldsymbol{U}^T$ を乗じて

$$\boldsymbol{C} = \boldsymbol{U}^{T}\boldsymbol{V} = \boldsymbol{U}^{T}\tilde{\boldsymbol{B}}^{-1}\boldsymbol{A}\tilde{\boldsymbol{B}}^{T}\boldsymbol{U} \tag{4.8}$$

をつくる．
以上の処理は，領域を節約するために実際には次のように行われる．すなわち，i=1, 2, ..., m に対して，補助ベクトル $\boldsymbol{y}$ を用いて

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{u}_i \\ \boldsymbol{v}_i &= \tilde{\boldsymbol{B}}^{-1}\boldsymbol{A}\tilde{\boldsymbol{B}}^{-T}\boldsymbol{y} \end{aligned} \tag{4.9}$$

とする．このことをサブルーチン UBCLX を MSBV によって行う．次に $\boldsymbol{C}$ は m 次の実対称行列であるから，その下三角部分のみを対象として

$$c_{ii} = \boldsymbol{y}^{T}\boldsymbol{v}_i, \tag{4.10}$$

$$c_{ji} = \boldsymbol{u}_{\mathrm{j}}^{\mathrm{T}}\boldsymbol{v}_{\mathrm{i}}, j = i+1,...,m \tag{4.11}$$

とする．このようにして，$\boldsymbol{U}$ から列ごとに $\boldsymbol{u}_i$ を取り出して，$\boldsymbol{v}_i$，$\boldsymbol{c}_{ii}$，$\boldsymbol{c}_{ji}$ を求めることにより，$\boldsymbol{V}$ はそのまま $\boldsymbol{U}$ の領域につくられる．

③ $\boldsymbol{C}$ の固有値問題を解き，

$$\boldsymbol{C} = \boldsymbol{PMP}^{T} \tag{4.12}$$

と分解する．ただし $\boldsymbol{M}$ は $\boldsymbol{C}$ の固有値を対角要素とする対角行列，$\boldsymbol{P}$ は m 次の直交行列である．このことを順次サブルーチン TRID1，TEIG1，TRBK によって行う．次に，固有値と対応する固有ベクトルを，固有値の絶対値の大きい方から整列する．これをサブルーチン UESRT によって行う．

④ $\boldsymbol{V}$ の右から $\boldsymbol{P}$ を乗じて，n 行 m 列の行列 $\boldsymbol{W}$ をつくる．

$$\boldsymbol{W} = \boldsymbol{VP} \tag{4.13}$$

⑤ $\boldsymbol{W}$ の各列の正規直交化を行うために，m 次の実対称正値行列 $\boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{W}$ をつくり，これを LL^{T} 分解する．

$$\boldsymbol{W}^{T}\boldsymbol{W} = \boldsymbol{LL}^{T} \tag{4.14}$$

このことをサブルーチン UCHLS によって行う．もしも LL^{T} 分解が不可能なときは，ICON=25000 として処理を打ち切る．

⑥ 方程式 $\boldsymbol{U}^{*}\boldsymbol{L}^{\boldsymbol{T}} = \boldsymbol{W}$ を解いて，

$$\boldsymbol{U}^{*} = \boldsymbol{W}\boldsymbol{L}^{-T} \tag{4.15}$$

を計算する．この $\boldsymbol{U}^{*}$は(4.6)の意味の正規直交性を有している．$\boldsymbol{U}$ と $\boldsymbol{U}^{*}$の第 m 列ベクトル $\boldsymbol{u}_m$ と $\boldsymbol{u}_m{}^{*}$について

$$d = \left\| \boldsymbol{u}_m^{*} - \boldsymbol{u}_m \right\|_{\infty} \le \varepsilon \tag{4.16}$$

が成り立つかどうかを調べる．
この収束条件が成立しないときは，$\boldsymbol{U}^{*}$をあらためて $\boldsymbol{U}$ として手順②へ行く．

⑦ 収束条件が成立した場合は，反復を止めて，③で得られた M の対角要素を固有値とする．
又，固有ベクトルは

$$\boldsymbol{X} = \tilde{\boldsymbol{B}}^{-T}\boldsymbol{U}^{*} \tag{4.17}$$

の始めの m 列とする．(4.17)の計算はサブルーチン UBCLX によって行う．
以上がジェニングス法の算法の大要である．なお，参考文献[18], [19]を参照のこと．

b. ジェニングスの加速法
本サブルーチンでは，上に述べたジェニングスの同時反復法を加速するために，ベクトル列に関するジェニングスの加速法を組み入れている．加速法の原理及びその実際的な適用については，サブルーチン BSEGJ の手法概要を参照すること．

E51-30-0301　GCHEB, DGCHEB

チェビシェフ級数の導関数
CALL GCHEB (A, B, C, N, ICON)

(1) 機能

区間[a, b]で定義された n 項のチェビシェフ級数,

$$f(x)=\sum_{k=0}^{n-1}{}'c_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \qquad (1.1)$$

が与えられたとき，その導関数をチェビシェフ級数

$$f'(x)=\sum_{k=0}^{n-2}{}' c'_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \qquad (1.2)$$

に展開し，その係数$\{c'_k\}$を求める.

ここで，Σ′は初項を 1/2 倍して和をとることを意味する．ただし，$a \neq b$，$n \geq 1$ であること.

(2) パラメタ

A…… **入力**．チェビシェフ級数の定義域の下限 a.

B…… **入力**．チェビシェフ級数の定義域の上限 b.

C…… **入力**．係数$\{c_k\}$.
C(1) =c_0, C(2) =c_1, ..., C(N) =c_{n-1}のように格納する.
出力．導関数の係数 $\{c'_k\}$
C(1) =c'_0, C(2) =c'_1, ..., C(N-1) =c'_{n-2}のように格納される.
大きさ N の 1 次元配列.

N…… **入力**．項数 n .
出力．導関数の項数 n–1.
（使用上の注意③参照）.

ICON…… **出力**．コンディションコード.
表 GCHEB-1 参照.

表 GCHEB-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	次のいずれかであった. ① N < 1 ② A = B	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数... FLOAT

b. 注意

① 任意の関数の導関数を求める場合，チェビシェフ級数展開するサブルーチン FCHEB と，本サブルーチンを順次呼び出せばよい.
更に，任意の点における微係数を求める場合は，チェビシェフ級数の求和のサブルーチン ECHEB を続けて呼び出せばよい.
（使用例参照）.

② 高階の導関数を必要とする場合，本サブルーチンを繰り返し呼び出せばよい.
導関数の誤差は，末尾 2 項の係数の絶対値和で推定できるが，高階になる程誤差が大きくなることを考慮する必要がある.

③ 入力の項数nが1の場合は，出力の項数も1となる.

c. 使用例

区間[-2, 2]上で，指数関数

$$f(x)=e^x\left(=\sum_{n=0}^{\infty}\frac{x^n}{n!}\right)$$

を，サブルーチン FCHEB により要求精度 $\varepsilon_a = 0$，$\varepsilon_r = 5\cdot 10^{-5}$ の下でチェビシェフ級数展開し，本サブルーチンによりその導関数を求める．さらに，x =-2 から，0.05 刻みで２までの分点上における微係数の値をサブルーチン ECHEB により求め真値と比較する.

```
C     **EXAMPLE**
      DIMENSION C(257),TAB(127)
      EXTERNAL FUN
      EPSR=5.0E-5
      EPSA=0.0
      NMIN=9
      NMAX=257
      A=-2.0
      B=2.0
      CALL FCHEB(A,B,FUN,EPSA,EPSR,NMIN,
     *           NMAX,C,N,ERR,TAB,ICON)
      IF(ICON.NE.0) GOTO 20
      WRITE(6,600) N,ERR,ICON
      WRITE(6,601) (C(K),K=1,N)
      CALL GCHEB(A,B,C,N,ICON)
      IF(ICON.NE.0) GOTO 20
      WRITE(6,602)
      WRITE(6,601) (C(K),K=1,N)
      WRITE(6,603)
      H=0.05
      X=A
   10 CALL ECHEB(A,B,C,N,X,Y,ICON)
      IF(ICON.NE.0) GOTO 20
      ERROR=FUN(X)-Y
      WRITE(6,604) X,Y,ERROR
      X=X+H
      IF(X.LE.B) GOTO 10
      STOP
   20 WRITE(6,605) ICON
      STOP
```

```
600 FORMAT('0',3X,'EXPANSION OF',
   * ' FUNCTION FUN(X)',3X,'N=',I4,3X,
   * 'ERROR=',E13.3,3X,'ICON=',I6)
601 FORMAT(/(5E15.5))
602 FORMAT('0',5X,'DERIVATIVE OF',
   * ' CHEBYSHEV SERIES')
603 FORMAT('0',10X,'X',7X,
   * 'DIFFERENTIAL',6X,'ERROR'/)
604 FORMAT(1X,3E15.5)
605 FORMAT('0',5X,'CONDITION CODE',I8)
    END
    FUNCTION FUN(X)
    REAL*8 SUM,XP,TERM
    EPS=AMACH(EPS)
    SUM=1.0
    XP=X
    XN=1.0
    N=1
 10 TERM=XP/XN
    SUM=SUM+TERM
    IF(DABS(TERM).LE.
   *   DABS(SUM)*EPS) GOTO 20
    N=N+1
    XP=XP*X
    XN=XN*FLOAT(N)
    GOTO 10
 20 FUN=SUM
    RETURN
    END
```

(4) 手法概要

区間[a, b]で定義された n 項のチェビシェフ級数を項別微分し，再びそれをチェビシェフ級数で表現する．即ち，

$$\frac{d}{dx}\sum_{k=0}^{n-1}{}'c_k T_k\left(\frac{2x-(b+a)}{b-a}\right)=\sum_{k=0}^{n-2}{}'c'_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \tag{4.1}$$

とすると，係数間には次の関係が成りたつ．

$$\begin{aligned} c'_{n-1} &= 0 \\ c'_{n-2} &= \left(\frac{4}{b-a}\right)(n-1)c_{n-1} \\ c'_{n-2} &= \left(\frac{4}{b-a}\right)kc_k + c'_{k+1'} \\ k &= n-2,n-3,...,1 \end{aligned} \tag{4.2}$$

本サブルーチンでは，チェビシェフ多項式の微分公式 (4.2)により $\{c'_k\}$ を求めている．

n 項の級数の導関数を求める必要な乗算回数は，約 $2n$ 回である．

なお，(4.2)式の漸化関係については，サブルーチン ICHEB の「手法概要」参照．

A25-31-0101 GINV, DGINV

実行列の一般逆行列（特異値分解法）
CALL GINV (A, KA, M, N, SIG, V, KV, EPS, VW, ICON)

(1) 機能

$m \times n$ の実行列 $\boldsymbol{A}$ の一般逆行列 $\boldsymbol{A}^+$を特異値分解法によって求める．$m \geq 1$, $n \geq 1$ なること．

(2) パラメタ

A……**入力**．行列 $\boldsymbol{A}$.
出力．一般逆行列 $\boldsymbol{A}^+$の転置行列.
A (KA, N) なる 2 次元配列.
（使用上の注意①参照）.

KA……**入力**．配列 A の整合寸法 (≥ M).

M……**入力**．行列 $\boldsymbol{A}$ の行数 m.

N……**入力**．行列 $\boldsymbol{A}$ の列数，行列 $\boldsymbol{V}$ の行数 n.

SIG……**出力**．行列 $\boldsymbol{A}$ の特異値.
大きさ n の 1 次元配列.
（使用上の注意②参照）.

V……**出力**．特異値分解による直交行列． V(KV, K) なる 2 次元配列.
K = min (M + l, N).

KV……**入力**． 配列 V の整合寸法 (≥ N).

EPS……**入力**．特異値の相対零判定値 (≥ 0.0).
0.0 のときは標準値が採用される.
（使用上の注意③参照）.

VW……**作業領域**．大きさ n の 1 次元配列.

ICON……**出力**．コンディションコード.
表 GINV-1 参照.

表 GINV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
15000	ある特異値が求められたなかった.	処理を打ち切る.
30000	KA < M, M < 1, N < 1, KV < N 又は EPS < 0.0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... ASVD1, AMACH, MGSSL

② FORTRAN 基本関数... なし

b. 注意

① パラメタ A の一に出力されるのは，$\boldsymbol{A}$ の一般逆行列の転置行列$(\boldsymbol{A}^+)^{\boldsymbol{T}}$であるので注意すること.

② 特異値はすべて非負で，大きさの減少する順に格納される．ICON=15000 の場合には，求められなかった特異値は-1 とし，大小順に整列しない.

③ 入力パラメタ EPS は $\boldsymbol{A}$ の階数を決定するための重要な役目を果たすので，その選択は慎重に行わなければならない．手法概要を参照のこと.

④ 連立 1 次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ の最小二乗最小ノルム解は，一般逆行列 $\boldsymbol{A}^+$によって，$\boldsymbol{X} = \boldsymbol{A}^+ \boldsymbol{b}$ として表されるが，この目的のためには，専用サブルーチン LAXLM を用いる方がはるかに有利である．結局，本サブルーチンは，一般逆行列 $\boldsymbol{A}^+$ 自体が必要な場合以外は用いない方がよい.

c. 使用例

$m \times n$ の行列 A の一般逆行列を求める.
$1 \leq n \leq m \leq 100$ の場合.

```
C      **EXAMPLE**
       DIMENSION A(100,100),SIG(100),
      *          V(100,100),VW(100)
    10 READ(5,500) M,N
       IF(M.EQ.0) STOP
       READ(5,510) ((A(I,J),I=1,M),J=1,N)
       WRITE(6,600) M,N,
      *  ((I,J,A(I,J),J=1,N),I=1,M)
       CALL GINV(A,100,M,N,SIG,V,100,0.0,
      *          VW,ICON)
       WRITE(6,610) ICON
       IF(ICON.NE.0) GO TO 10
       WRITE(6,620) N,M,
      *  ((I,J,A(J,I),I=1,M),J=1,N)
       GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX'/
     *  6X,'ROW NUMBER=',I4,5X,
     *  'COLUMN NUMBER=',I4/
     *  (10X,4('(',I3,',',I3,')',E17.7,
     *  3X)))
  610 FORMAT(' ',10X,'CONDITION CODE='
     *  ,I6)
  620 FORMAT('1',5X,
     *  'GENERALIZED INVERSE'/6X,
     *  'ROW NUMBER=',I4,5X,
     *  'COLUMN NUMBER=',I4/(10X,
     *  4('(',I3,',',I3,')',E17.7,3X)))
       END
```

(4) 手法概要

$m \times n$ の行列 $\boldsymbol{A}$ が対して，次の関係を満足する $n \times m$ の行列 $\boldsymbol{X}$ を，$\boldsymbol{A}$ の (Moore-Penrose の) 一般逆行列という.

$$\left.\begin{aligned} \boldsymbol{AXA} &= \boldsymbol{A} \\ \boldsymbol{XAX} &= \boldsymbol{X} \\ (\boldsymbol{AX})^T &= \boldsymbol{AX} \\ (\boldsymbol{XA})^T &= \boldsymbol{XA} \end{aligned}\right\} \tag{4.1}$$

一般逆行列は $\boldsymbol{A}$ に対して一意的に定まり，これを $\boldsymbol{A}^+$と書く．なお $\boldsymbol{A}$ が正方行列で正則ならば $\boldsymbol{A}^+$ は $\boldsymbol{A}^{-1}$ である．一般逆行列の一意性と(4.1)から,

$$\left(\boldsymbol{A}^{+}\right)^{+} = \boldsymbol{A}, \tag{4.2}$$

$$\left(\boldsymbol{A}^{T}\right)^{+} = \left(\boldsymbol{A}^{+}\right)^{T} \tag{4.3}$$

が成り立つ．(4.3)により，一般性を失うことなく，$m \geq n$ と仮定してもよい．

a. 特異値分解と一般逆行列
今，$\boldsymbol{A}$ の特異値分解

$$\boldsymbol{A} = \boldsymbol{U}\sum\boldsymbol{V}^{T} \tag{4.4}$$

が与えられているものとする．ここに，$\boldsymbol{U}$ は $\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U} = \boldsymbol{I}$ なる $m \times n$ の行列，$\boldsymbol{V}$ は n 次の直交行列，Σ は n 次の対角行列である．$\boldsymbol{U}$ の右側に m–n 個の列ベクトルを補って，m 次の直交行列 $\boldsymbol{U_c}$ を作り，Σ の下側に m–n 行 n 列の零行列を付加してΣ_cを作ると，$\boldsymbol{A}$ の特異値分解は

$$\boldsymbol{A} = \boldsymbol{U}_c\sum_c\boldsymbol{V}^{T} \tag{4.5}$$

と書きかえることができる．

$$\sum = \mathrm{diag}\left(\sigma_1, \sigma_2, ..., \sigma_n\right) \tag{4.6}$$

とすると明らかに，n 次の対角行列$\boldsymbol{\Sigma}^{+}$を次のように表すことができる．

$$\sum^{+} = \mathrm{diag}\left(\sigma_1^{+}, \sigma_2^{+}, ..., \sigma_n^{+}\right) \tag{4.7}$$

ただし，

$$\sigma_i^{+} = \begin{cases} 1/\sigma_i, \sigma_i > 0 \\ 0 \quad , \sigma_i = 0 \end{cases} \tag{4.8}$$

とする．$\boldsymbol{\Sigma}^{+}$の右側に $n \times (m\text{-}n)$の零行列を付加した行列は，$\sum_c^{+}$ に等しい．$\boldsymbol{X} = \boldsymbol{V}\sum_c^{+}\boldsymbol{U}_c^{\mathrm{T}}$ と仮定して，この $\boldsymbol{X}$ と(4.5)を (4.1)に代入し，$\boldsymbol{U_c}$ と $\boldsymbol{V}$ とが直交行列であることを利用すると，(4.1)は

$$\left.\begin{aligned} &\sum_c\sum_c^{+}\sum_c = \sum_c \\ &\sum_c^{+}\sum_c\sum_c^{+} = \sum_c^{+} \\ &\left(\sum_c\sum_c^{+}\right)^{\mathrm{T}} = \sum_c\sum_c^{+} \\ &\left(\sum_c^{+}\sum_c\right)^{\mathrm{T}} = \sum_c^{+}\sum_c \end{aligned}\right\} \tag{4.9}$$

となる．
したがって，一般逆行列の一意性から $\boldsymbol{A}^{+}$ は

$$\boldsymbol{A}^{+} = \boldsymbol{V}\sum_c^{+}\boldsymbol{U}_c^{T} \tag{4.10}$$

で与えられるが，$\sum_c^{+}$ の性質と $\boldsymbol{U_c}$ の定義から

$$\boldsymbol{A}^{+} = \boldsymbol{V}\sum^{+}\boldsymbol{U}^{T} \tag{4.11}$$

と書きかえることができる．

b. 本サブルーチンにおける計算手順
① A の特異値分解を行い，$\boldsymbol{U}$，$\boldsymbol{\Sigma}$，$\boldsymbol{V}$ を求める．$\boldsymbol{U}$ は A の位置に作られる．このことを，サブルーチン ASVD1 によって行う．詳細については，サブルーチン ASVD1 を参照のこと．
② $\boldsymbol{A}^{+}$ の転置行列$(\boldsymbol{A}^{+})^{\boldsymbol{T}}$を

$$\left(\boldsymbol{A}^{+}\right)^{T} = \boldsymbol{U}\sum^{+}\boldsymbol{V}^{T} \tag{4.12}$$

によって，$\boldsymbol{A}$ の位置に生成する．
Σ^{+}を乗ずるときには特異値 σ_i に対する零判定を行う．判定基準としては $\sigma_1 \cdot$ EPS を用いる．ここに，σ_1 は最大の特異値である．この判定基準より小さい特異値は 0 とみなされる．EPS として 0.0 が入力されたときは，$16u$ を基準値として用いる．ただし，u は丸め誤差の単位である．

なお，詳細については，サブルーチン ASVD1 の手法概要及び参考文献[11]を参照すること．

B22-21-0402 GSBK, DGSBK

一般形の固有ベクトルへの逆変更 （実対称行列の一般固有値問題）
CALL GSBK (EV, K, N, M, B, ICON)

(1) 機能

n 次の実対称行列 $\boldsymbol{S}$ の m 個の固有ベクトル $\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_m$ を，一般固有値問題 $\boldsymbol{Ax} = \lambda\boldsymbol{Bx}$ の固有ベクトル $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m$ へ変更する．

ただし，$\boldsymbol{S}$ は $\boldsymbol{B} = \boldsymbol{LL}^T$ として

$$\boldsymbol{S}=\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T} \qquad (1.1)$$

により得られたものであること．ここで $\boldsymbol{L}$ は下三角行列である．$1 \le m \le n$ であること．

(2) パラメタ

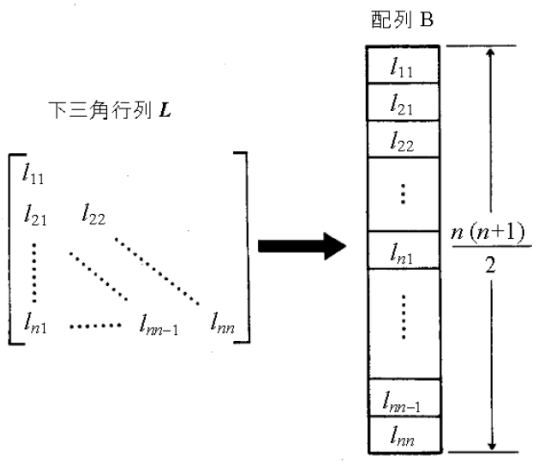
EV ……**入力**．実対称行列 $\boldsymbol{S}$ の m 個の固有ベクトル．
出力．一般固有値問題 $\boldsymbol{Ax} = \lambda\boldsymbol{Bx}$ の固有ベクトル．
EV (K, m) なる 2 次元配列．
（使用上の注意②参照）

K……**入力**．入力 EV の整合寸法．

N……**入力**．実対称行列 $\boldsymbol{S}$，$\boldsymbol{A}$ 及び $\boldsymbol{B}$ の次数 n．

M……**入力**．|M|は，固有ベクトルの個数 m．
（使用上の注意③参照）

B……**入力**．下三角行列 $\boldsymbol{L}$．（図 GSBK-1 参照）
大きさ $n(n+1)/2$ の 1 次元配列．
（使用上の注意①参照）



[注意] 配列 B には，下三角行列 L が対称行列用圧縮モードで格納されていること．

図 GSBK-1 配列 B の内容

ICON……**出力**．コンディションコード．
表 GSBK-1 参照．

表 GSBK-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N = 1 であった	EV (1, 1) = 1.0 / B(1) とする．
30000	N < |M|, K < N 又は M = 0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数... IABS

b. 注意
① サブルーチン GSCHL の出力パラメタ B は，本サブルーチンの入力パラメタ B と対応しているのでそのまま使用できる．
② 固有ベクトル $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m$ は，入力のベクトル $\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_m$ が $\boldsymbol{Y}^T\boldsymbol{Y} = \boldsymbol{I}$ を満足するように正規化されていれば，$\boldsymbol{X}^T\boldsymbol{BX} = \boldsymbol{I}$ を満足するように出力される．ただし，$\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_m]$，$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m]$である．
③ パラメタ M が負であるときはその絶対値をとって使う．

c. 使用例
n 次の実対称行列 $\boldsymbol{A}$ 及び n 次の正値対称行列 $\boldsymbol{B}$ で与えられる一般固有値問題を，一連のサブルーチン GSCHL，TRID1，TEIG1，TRBK 及び GSBK により，すべての固有値固有ベクトルを求める．
$n \le 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),B(5050),
     *          SD(100),E(100),
     *EV(100,100),D(100)
   10 READ(5,500) N,M,EPSZ,EPST
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      READ(5,510) (B(I),I=1,NN)
      WRITE(6,600) N,M,EPSZ,EPST
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      WRITE(6,620)
      NE=0
      DO 30 I=1,N
      NI=NE+1
      NE=NE+I
   30 WRITE(6,610) I,(B(J),J=1,NI,NE)
      CALL GSCHL(A,B,N,EPSZ,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
```

```
      CALL TRID1(A,N,D,SD,ICON)
      CALL TEIG1(D,SD,N,E,EV,100,M,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL TRBK(EV,100,N,M,A,ICON)
      CALL GSBK(EV,100,N,M,B,ICON)
      MM=IABS(M)
      CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(2I5,2E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'**ORIGINAL ',
     * 'MATRIX A**',11X,'** ORDER =',I5,
     * 10X,'** M =',I3/46X,'EPSZ=',E15.7,
     * 'EPST=',E15.7)
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT('1',10X,'**ORIGINAL ',
     * 'MATRIX B**')
  630 FORMAT(/11X,'** CONDITION CODE =',
     * I5/)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細についてはサブルーチン SEIG1 の使用例参照のこと．

(4) 手法概要

n 次の実対称行列 $\boldsymbol{S}$ の固有ベクトル $\boldsymbol{y}$ を，一般固有値問題（$\boldsymbol{A}$: 対称行列，$\boldsymbol{B}$: 正値対称行列）

$$\boldsymbol{Ax} = \lambda \boldsymbol{Bx} \tag{4.1}$$

の固有ベクトル $\boldsymbol{x}$ へ変換する．
この場合，(4.1)から標準形(4.2)の変換は前もって行われていなければならない．

$$\boldsymbol{Sy} = \lambda \boldsymbol{y} \tag{4.2}$$

この過程は(4.3)～(4.6)によって示される．
正値対称行列 $\boldsymbol{B}$ を

$$\boldsymbol{B} = \boldsymbol{LL}^{\boldsymbol{T}} \tag{4.3}$$

のように分解し，これを(4.1)へ代入すると

$$\begin{aligned} &\boldsymbol{Ax} = \lambda \boldsymbol{LL}^{T}\boldsymbol{x} \\ &\boldsymbol{L}^{-1}\boldsymbol{Ax} = \lambda \boldsymbol{L}^{T}\boldsymbol{x} \\ &\boldsymbol{L}^{-1}\boldsymbol{A}\left(\boldsymbol{L}^{-T}\boldsymbol{L}^{T}\right)\boldsymbol{x} = \lambda \boldsymbol{L}^{T}\boldsymbol{x} \\ &\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}\left(\boldsymbol{L}^{T}\boldsymbol{x}\right) = \lambda\left(\boldsymbol{L}^{T}\boldsymbol{x}\right) \end{aligned} \tag{4.4}$$

となる．したがって

$$\boldsymbol{S} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\boldsymbol{T}} \tag{4.5}$$

$$\boldsymbol{y} = \boldsymbol{L}^{\boldsymbol{T}}\boldsymbol{x} \tag{4.6}$$

$\boldsymbol{L}$ は既知であるから，$\boldsymbol{x}$ は(4.6)より次のように求めることができる．

$$\boldsymbol{x} = \boldsymbol{L}^{-T}\boldsymbol{y} \tag{4.7}$$

詳細については，参考文献 [13]の pp.303－314 参照のこと．

B22-21-0302 GSCHL, DGSCHL

一般形から標準形への変換 （実対称行列の一般固有値問題）
CALL GSCHL (A, B, N, EPSZ, ICON)

(1) 機能

n 次の実対称行列 $\boldsymbol{A}$，及び n 次の正値対称行列 $\boldsymbol{B}$ に対して，一般固有値問題

$$\boldsymbol{Ax} = \lambda\boldsymbol{Bx} \qquad (1.1)$$

を標準固有値問題

$$\boldsymbol{Sy} = \lambda\boldsymbol{y} \qquad (1.2)$$

に変換する．ここで $\boldsymbol{S}$ は実対称行列である．

$n \geq 1$ なること．

(2) パラメタ

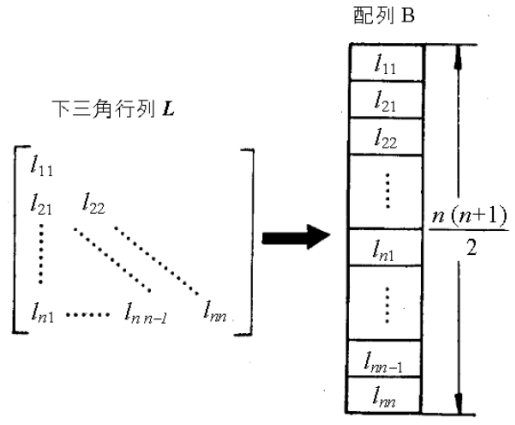
A……**入力**．実対称行列 $\boldsymbol{A}$.
出力．実対称行列 $\boldsymbol{S}$.
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

B……**入力**．正値対称行列 $\boldsymbol{B}$.
出力．下三角行列 $\boldsymbol{L}$.（図 GSCHL-1 参照)
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

N……**入力**．行列の次数 n.

EPSZ……**入力**．$\boldsymbol{B}$ の LL^T 分解におけるピボットの相対零判定値.
0.0 又は負を指定すると，標準値が採用される．（使用上の注意①参照）

ICON……**出力**．コンディションコード.
表 GSCHL-1 参照．.



[注意] 行列 $\boldsymbol{L}$ の下三角行部分を，対象行列圧縮モードで 1 次元配列 B に格納する.

図 GSCHL-1 行列 $\boldsymbol{L}$ の格納方法

表 GSCHL-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1 であった.	A(1) = A (1) / B(1) B(1)=SQRT (B(1)) とする.
28000	行列 $\boldsymbol{B}$ の LL^T 分解において，ピボットが負となった. 入力行列 $\boldsymbol{B}$ は正値ではない.	処理を打ちきる.
29000	行列 $\boldsymbol{B}$ の LL^T 分解において，ピボットが相対的に零となった. 入力行列 $\boldsymbol{B}$ は非正則の可能性が強い.	処理を打ちきる.
30000	N < 1 であった.	処理を打ちきる.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL, UCHLS

② FORTRAN 基本関数 ... SQRT

b. 注意

③ パラメタ EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である（サブルーチン LSX 参照）.
本サブルーチンにおいて EPSZ に 10^{-S} を設定したとすると，正値対称行列 $\boldsymbol{B}$ の LL^T 分解の過程でピボットの値が 10 進 $\boldsymbol{s}$ 桁以上の桁落ちを生じた場合に，そのピボットを零と見なしコンディションコードを設定し (ICON＝29000) 処理を打ち切る．なおピボットが小さくなっても計算を続行させる場合には，EPSZ に極小の値（例えば 10^{-70}）を与えればよいが，その結果は保証されない.
行列の $\boldsymbol{B}$ の LL^T 分解の過程でピボットが負となった場合，$\boldsymbol{B}$ は正値行列ではない．本サブルーチンでは，このときコンディションコードを設定し(ICON＝28000) 処理を打ち切る.

c. 使用例

n 次の実対称行列 $\boldsymbol{A}$ 及び n 次の正値対称行列 $\boldsymbol{B}$ の一般固有値問題 $\boldsymbol{Ax} = \lambda\boldsymbol{Bx}$ を，サブルーチン GSCHL により標準形に変換し，更にサブルーチン TRID1 により実対称 3 重対角行列に変換する．その後，サブルーチン BSCT1 により m 個の固有値を求める．$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),B(5050),VW(300),
     *          E(100),D(100),SD(100)
   10 CONTINUE
      READ(5,500) N,M,EPSZ,EPST
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      READ(5,510) (B(I),I=1,NN)
      WRITE(6,600) N,M,EPSZ,EPST
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      WRITE(6,620)
      NE=0
      DO 30 I=1,N
      NI=NE+1
      NE=NE+I
   30 WRITE(6,610) I,(B(J),J=1,NI,NE)
      CALL GSCHL(A,B,N,EPSZ,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL TRID1(A,N,D,SD,ICON)
      CALL BSCT1(D,SD,N,M,EPST,E,VW,ICON)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,640)
      MM=IABS(M)
      WRITE(6,650) (I,E(I),I=1,MM)
      GO TO 10
  500 FORMAT(2I5,2E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL ',
     * 'MATRIX A'/11X,'** ORDER =',I5,10X,
     * '** M =',I3,10X,'** EPSZ =',E15.7,
     * 10X,'EPST =',E15.7)
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT('0',10X,'** ORIGINAL ',
     * 'MATRIX B')
  630 FORMAT(/11X,'** CONDITION CODE =',
     * I5/)
  640 FORMAT('0'/11X,'** EIGENVALUES')
  650 FORMAT(5X,'E(',I3,')=',E15.7)
      END
```

(4) 手法概要

$\boldsymbol{A}$ を n 次の対称行列，$\boldsymbol{B}$ を正値対称行列とするとき，一般固有値問題

$$\boldsymbol{Ax} = \lambda \boldsymbol{Bx} \tag{4.1}$$

の標準形

$$\boldsymbol{Sy} = \lambda \boldsymbol{y} \tag{4.2}$$

への変換は (4.3)～(4.6)によって行われる．
$\boldsymbol{B}$ は正値対称行列であるので，

$$\boldsymbol{B} = \boldsymbol{L}\boldsymbol{L}^T \tag{4.3}$$

の形に分解できる．ここで $\boldsymbol{L}$ は下三角行列である．しかもこの分解は $\boldsymbol{L}$ の対角要素を正になるように定めれば，一意に定まる．(4.1)及び(4.3)より，

$$\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}\left(\boldsymbol{L}^T\boldsymbol{x}\right) = \lambda\left(\boldsymbol{L}^T\boldsymbol{x}\right) \tag{4.4}$$

となる．ここで，$\boldsymbol{L}^{-T}$ は $(\boldsymbol{L}^{-1})^T$ 又は $(\boldsymbol{L}^{\mathrm{T}})^{-1}$ を意味する．
したがって，

$$\boldsymbol{S} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T} \tag{4.5}$$

$$\boldsymbol{y} = \boldsymbol{L}^T\boldsymbol{x} \tag{4.6}$$

とすれば，(4.2)が導き出される．

(4.3)の分解は，コレスキー法により行われる．すなわち，行列 $\boldsymbol{L}$ の要素は (4.7)，(4.8)により，行ごとに逐次求められる．

$$l_{11} = \left(b_{11}\right)^{\frac{1}{2}} \tag{4.7}$$

$$\left.\begin{aligned} l_{ij} &= \left(b_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk}\right) \Big/ l_{jj}\,,\ j = 1,\ldots,i-1 \\ l_{ii} &= \left(b_{ii} - \sum_{k=1}^{i-1} l_{ik}^2\right)^{\frac{1}{2}} \end{aligned}\right\} i = 2,\ldots,n \tag{4.8}$$

ここで，$\boldsymbol{L} = (l_{ij})$, $\boldsymbol{B} = (b_{ij})$である．

詳細については，参考文献 [13]の PP.303-314 を参照のこと．

B22-21-0201 GSEG2, DGSEG2

実対称行列の一般固有値及び固有ベクトル （バイセクション法，逆反復法）
CALL GSEG2 (A, B, N, M, EPSZ, EPST, E, EV, K, VW, ICON)

(1) 機能

n 次実対称行列 A，及び n 次の正値対称行列 $\boldsymbol{B}$ に対して一般固有値問題

$$\boldsymbol{Ax} = \lambda \boldsymbol{Bx} \qquad (1.1)$$

の m 個の固有値を，バイセクション法により，大きい方から又は小さい方から求め，対応する m 個の固有ベクトル $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m$ を逆反復法により求める．固有ベクトルは，

$$\boldsymbol{X}^T\boldsymbol{BX} = \boldsymbol{I} \qquad (1.2)$$

となっている．ただし，$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m]$ である．$1 \le m \le n$ なること．

(2) パラメタ

A …… **入力**．実対称行列 $\boldsymbol{A}$．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ なる 1 次元配列．
演算後，内容は保存されない．

B …… **入力**．正値対称行列 $\boldsymbol{B}$．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ なる 1 次元配列．
演算後，内容は保存されない．

N …… **入力**．実対称行列 $\boldsymbol{A}$ 及び正値対称行列 $\boldsymbol{B}$ の次数 n．

M …… **入力**．求める固有値の個数 m．
M = + m のときは大きい方から求める．
M = – m のときは小さい方から求める．

EPSZ …… **入力**．B の LL^T 分解におけるピボットの相対零判定値．
0.0 又は負を指定すると，標準値が採用される．（使用上の注意①参照）

EPST …… **入力**．固有値の収束判定に使われる絶対誤差の上限．
0.0 又は負を指定すると，標準値が採用される．（使用上の注意②参照）

E …… **出力**．m 個の固有値．
M が正の場合大きい順に，**M** が負の場合小さい順に並ぶ．
大きさ m の 1 次元配列．

EV …… **出力**．固有ベクトル．
固有ベクトルは列方向に格納される．
EV (K, m)なる 2 次元配列．

K …… **入力**．配列 EV の整合寸法．

VW …… **作業領域**．大きさ $7n$ の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 GSEG2-1 参照．

表 GSEG2-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N = 1 であった．	E(1)=A(1)/B(1) X(1, 1)=1.0/SQRT (B(1))とする
15000	固有ベクトルのうち求まらないものがあった	その固有ベクトルを 0 ベクトルとする．
20000	固有ベクトルはすべて求まらなかった．	固有ベクトルはすべて 0 ベクトルとする．
28000	$\boldsymbol{B}$ の LL^T 分解において，ピボットが負となった． 入力行列 $\boldsymbol{B}$ は正値でない．	処理を打ち切る．
29000	$\boldsymbol{B}$ の LL^T 分解において，ピボットが相対的に零となった． 入力行列 $\boldsymbol{B}$ は非正則の可能性が強い．	処理を打ち切る．
30000	M = 0，N < |M| 又は K < N であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... GSCHL, TRID1, UTEG2, TRBK, GSBK, AMACH, UCHLS, MGSSL

② FORTRAN 基本関数 ... IABS, SQRT, SIGN, ABS, AMAX1, DSQRT

b. 注意

① パラメタ EPSZ の標準値は，丸め誤差の単位を u としたとき EPSZ=16·u である．
本サブルーチンにおいて EPSZ に 10^{-S} を設定したとすると，正値対称行列 $\boldsymbol{B}$ の LL^T 分解の過程でピボットの値が 10 進 s 桁以上の桁落ちを生じた場合に，そのピボットを零と見なしコンディションコードを設定し(ICON=29000) 処理を打ち切る．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ に非常に小さな値（例えば 10^{-70}）を与えればよいが，結果は保証されない．正値対称行列 $\boldsymbol{B}$ の LL^T 分解の過程でピボットが負となった場合，係数行列は正値ではない．本サブルーチンでは，このときのコンディションコードを設定し(ICON=28000) 処理を打ち切る．

② パラメタ EPST の標準値は，丸め誤差の単位を u としたとき

$$\mathrm{EPST} = u \cdot \max(|\lambda_{\max}|, |\lambda_{\min}|),$$

である．ここで，$\lambda_{\max}$ 及び $\lambda_{\min}$ は $\boldsymbol{Ax} = \lambda\boldsymbol{Bx}$ の固有値の存在範囲（ゲルシュゴリン定理により与えられる）の上限及び下限である．

固定値の非常に大きいものと非常に小さいものとが混在しているときには，小さい方の固有値を精度よく求めることは難しいことがある．この場合，EPST に小さな値を設定することにより，小さい方の固有値を精度よく求めることができる．
ただし，処理速度は遅くなる．
EPST の与え方の詳細についてはサブルーチン BSCT1 参照のこと．

c. 使用例

n 次の実対称行列 A 及び n 次の正値対称行列 B についての一般固有値問題 $\boldsymbol{Ax} = \lambda\boldsymbol{Bx}$ の大きい方又は小さい方から m 個の固有値及び対応する固有ベクトルを求める．
$n \leq 100$, $m \leq 100$ の場合．

```
C     ** EXAMPLE**
      DIMENSION A(5050),B(5050),E(10),
     *          EV(100,10),VW(700)
   10 CONTINUE
      READ(5,500) N,M,EPSZ,EPST
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      READ(5,510) (B(I),I=1,NN)
      WRITE(6,600) N,M,EPSZ,EPST
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
      WRITE(6,610) I,(A(J),J=NI,NE)
   20 CONTINUE
      WRITE(6,620)
      NE=0
      DO 30 I=1,N
      NI=NE+1
      NE=NE+I
      WRITE(6,610) I,(B(J),J=1,NI,NE)
   30 CONTINUE
      CALL GSEG2(A,B,N,M,EPSZ,EPST,E,
     *           EV,100,VW,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      MM=IABS(M)
      CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(2I5,2E15.7)
  510 FORMAT(5E15.7)
  600 FORMAT('1','OROGINAL MATRIX A',5X,
     * 'N=',I3,' M=',I3,' EPSZ=',
     * E15.7,' EPST=',E15.7)
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT('1','OROGINAL MATRIX B')
  630 FORMAT('0',20X 'ICON=',I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細についてはサブルーチン SEIG1 の使用例参照のこと．

(4) 手法概要

n 次の実対称列 A 及び n 次の正値対称行列 $\boldsymbol{B}$ についての一般固有値問題

$$\boldsymbol{Ax} = \lambda\boldsymbol{Bx} \tag{4.1}$$

の固有値固有ベクトルを次の手順で求める．

① 一般固有問題の標準形への変換
(4.1)の $\boldsymbol{B}$ は正値対称行列であるから，

$$\boldsymbol{B} = \boldsymbol{LL}^{T} \tag{4.2}$$

の形に分解できる．ここで $\boldsymbol{L}$ は n 次の下三角行列である．この分解は $\boldsymbol{L}$ の対角要素を正になるように定めれば，一意に定まる．
(4.1)及び(4.2)より，$\boldsymbol{A}$ の左から $\boldsymbol{L}^{-1}$ を，右から $\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{\mathrm{T}}$ を掛けて，

$$\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}\left(\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}\right) = \lambda\left(\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}\right) \tag{4.3}$$

となる．ここで $\boldsymbol{L}^{-\mathrm{T}}$ は $(\boldsymbol{L}^{\mathrm{T}})^{-1}$ 又は $(\boldsymbol{L}^{-1})^{\mathrm{T}}$ の代わりに使われている．ここで，

$$\boldsymbol{S} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}} \tag{4.4}$$

$$\boldsymbol{y} = \boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} \tag{4.5}$$

とおくと，$\boldsymbol{S}$ は実対称行列，(4.3)は

$$\boldsymbol{Sy} = \lambda\boldsymbol{y} \tag{4.6}$$

となり，標準形が導かれる．

② 実対称行列の固有値，固有ベクトル
実対称行列 $\boldsymbol{S}$ を直交相似変換により実対称三重対角行列 $\boldsymbol{T}$ とし，$\boldsymbol{T}$ の固有値 λ とそれに対応する固有ベクトル $\boldsymbol{y}'$ をそれぞれバイセクション法と逆反復法により求める． $\boldsymbol{y}'$は更に $\boldsymbol{S}$ の固有ベクトル $\boldsymbol{y}$ へ逆変換される．

③ 一般固有値問題の固有ベクトル
(4.1)の固有ベクトル $\boldsymbol{x}$ は②より求めた固有ベクトル $\boldsymbol{y}$ から，

$$\boldsymbol{x} = \boldsymbol{L}^{-\mathrm{T}}\boldsymbol{y} \tag{4.7}$$

により得られる．

①は GSCHL ②は TRID1 と UTEG2，③は GSBK の各サブルーチンを用いて行う．

詳細については，参考文献 [13]の pp. 303-314 を参照のこと．

H11-20-0121 HAMNG, DHAMNG

連立1階常微分方程式（ハミング法）
CALL HAMNG (Y, N1, H, XEND, EPS, SUB, OUT, VW, ICON)

(1) 機能

連立1階常微分方程式の初期値問題,

$$\left.\begin{aligned} y_1' &= f_1(x, y_1, y_2, ..., y_n),\ y_{10} = y_1(x_0) \\ y_2' &= f_2(x, y_1, y_2, ..., y_n),\ y_{20} = y_2(x_0) \\ &\cdots \\ &\cdots \\ y_n' &= f_n(x, y_1, y_2, ..., y_n),\ y_{n0} = y_n(x_0) \end{aligned}\right\} \quad (1.1)$$

における関数系 f_1, $\cdots f_n$ と初期値 x_0, y_{10}, $\cdots$, y_{n0} が与えられたとき，ハミング法により $x=x_0$ から $x=x_e$ にわたり解 y_i と微係数 $y'_i(i=1, \cdots, n)$を求める.

キザミは，指定されたキザミを採用するが，解に対する所要の精度を達成するために小さくされることもあり，逆に計算速度を上げるために（所要の精度は達成するという条件で）大きくされることもある.

(2) パラメタ

Y…… **入力**．初期値 $x_0, y_{10}, y_{20}, ..., y_{n0}$.
大きさ n+1 の 1 次元配列.
演算後内容は保存されない.

N1…… **入力**．（連立常微分方程式の元数 n）+1.

H…… **入力**．初期キザミ．H ≠ 0.0
演算後内容は保存されない.

XEND…… **入力**．独立変数 x の最終値 x_e.
x_e における解が求まったところで計算は終了する.

EPS…… **入力**．解に対する相対誤差の上限.
0.0 の場合は，標準値を採用する.
（使用上の注意，手法概要参照）

SUB…… **入力**．(1.1) 式における f_i (i = 1, 2, ..., n)を計算するサブルーチン副プログラム名.
［副プログラムの用意のし方］
SUBROUTINE SUB (YY, FF)
パラメタ
YY: **入力**．YY(1) = x, YY(2) = y_1,
YY(3) = y_2, ...,
YY(n+1) = y_n なる対応を持つ大きさ n + 1, の 1 次元配列.
FF: **出力**．FF(2) = f_1, FF(3) = f_2,
FF(4) = f_3 ,...,
FF(n+1) = f_n なる対応を持つ大きさ n + 1 の 1 次元配列.
(使用例参照.)
なお FF (1) には何も代入してはならない.

OUT …… **入力**．計算結果を受けとるサブルーチン副プログラム名．すなわちキザミ HH (必ずしも初期キザミと同じではない)で 1 ステップ計算するたびに，HAMNG はこの副プログラムに結果を渡す.
［副プログラムの用意のし方］
SUBROUTINE OUT (YY, FF, N1, HH, IS)
パラメタ
YY… **入力**．$x, y_1, ..., y_n$ の計算結果
YY(1) = x, YY(2) = y_1, ...,
YY(n+1) = y_n なる対応をもつ大きさ n+1 の 1 次元配列.
FF… **入力**．$y'_1, ..., y'_n$ の計算結果.
FF(2) = y'_1, FF(3) = y'_2 ,...,
FF(n+1) = y'_n なる対応をもつ大きさ n+1 の 1 次元配列.
なお FF (1) には常に 1 がセットされる.
N1… **入力**．n+1.
HH… **入力**．y_i, y'_i を求めたときのキザミ.
IS…‥ **入力**．初期キザミを H とするとき，上記の HH が HH = 2^{IS}*H であることを示す.
（使用上の注意参照）.
またこの副プログラムの内部で IS に -11 を代入することにより，その時点で計算をただちに終了させることもできる.
なおこの副プログラムの内部では，IS 以外のパラメタの値を変更してはならない.
（使用例参照）

VW …… **作業領域**．大きさ 18 (n+1) の 1 次元配列.

ICON …… **出力**．コンディションコード
表 HAMNG-1 参照.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, RKG, MGSSL

② FORTRAN 基本関数...ABS, AMAX1

b. 注意

① パラメタ SUB, OUT は本サブルーチンを呼び出す側のプログラムで EXTERNAL 宣言をすること.

② 本ルーチンでは IS<-10 となったとき ICON = 20000 として計算を打ち切る．ただし利用者自身が，サブルーチン OUT の中で IS = -11 と代入したときは ICON = 0 として終了する.

表 HAMNG-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	0.0 < EPS < 64 *u* であった. ただし *u* は丸め誤差の単位.	EPS として標準値 (64*u*) を採用して処理を続ける.
20000	与えられた初期キザミ H の 2^{-10} 倍のキザミで計算しても相対誤差を EPS 以下にできないことが起こった.	その時点で処理を打ち切る.
30000	N1 < 2, EPS < 0.0, 1.0 < EPS, (XEND-Y(1)) *H ≤ 0.0 のいずれかであった.	処理を打ち切る.

③ EPS について
EPS は,
$64\,n \le \mathrm{EPS} < 1$
を満たす値であること. ただし EPS = 0.0 とした場合は, 本ルーチンの内部で標準値,
$64u$
の値を採用する. ここで *u* は丸め誤差の単位である.

c. 使用例
連立一階常微分方程式の初期値問題 (3.1) を解く.

$$\left.\begin{aligned} y_1' &= y_2 \qquad\quad , y_{10} = y_1(1) = 5 \\ y_2' &= \frac{4}{x^2} y_1 + \frac{2}{x} y_2 , y_{20} = y_2(1) = 3 \end{aligned}\right\} \tag{3.1}$$

H=0.1, XEND=5.0, EPS=10^{-4} の場合

```
C     **EXAMPLE**
      DIMENSION Y(3),VW(54)
      EXTERNAL SUB,OUT
      Y(1)=1.0
      Y(2)=5.0
      Y(3)=3.0
      H=0.1
      EPS=1.0E-4
      WRITE(6,600)
      CALL HAMNG(Y,3,H,5.0,EPS,SUB,OUT,VW,
     *           ICON)
      WRITE(6,610) ICON
      STOP
  600 FORMAT('1'/' ',17X,'X',15X,'Y1',14X,
     * 'Y2',14X,'F1',14X,'F2',14X,'HH',
     * 12X,'IS'//)
  610 FORMAT(' ',20X,'ICON=',I5)
      END
      SUBROUTINE SUB(YY,FF)
      DIMENSION YY(3),FF(3)
      FF(2)=YY(3)
      FF(3)=4.0*YY(2)/(YY(1)*YY(1))+2.0
     *      *YY(3)/YY(1)
      RETURN
      END
      SUBROUTINE OUT(YY,FF,N1,HH,IS)
      DIMENSION YY(N1),FF(N1)
      WRITE(6,600) (YY(I),I=1,N1),(FF(I),
     *             I=2,N1),HH,IS
      RETURN
  600 FORMAT(' ',10X,6E16.8,I10)
      END
```

(4) 手法概要

連立 1 階常微分方程式の初期値問題 (1.1) は, 独立変数そのものを一つの関数,

$$y_0(x) = x, y_{00} = y_0(x_0) = x_0 \tag{4.1}$$

と見なすことにより次の (4.2) のように書くことができる.

$$\begin{aligned} &\text{ただし } f_0(y_0, y_1, \cdots\cdots, y_n) = 1 \text{とする} \\ &y_0' = f_0(y_0, y_1, \cdots\cdots, y_n), y_{00} = x_0 \\ &y_1' = f_1(y_0, y_1, \cdots\cdots, y_n), y_{10} = y_1(x_0) \\ &y_2' = f_2(y_0, y_1, \cdots\cdots, y_n), y_{20} = y_2(x_0) \\ &\cdots \qquad\qquad\qquad \cdots \\ &y_n' = f_n(y_0, y_1, \cdots\cdots, y_n), y_{n0} = y_n(y_0) \end{aligned} \tag{4.2}$$

更に, 記号を簡潔にするために, いくつかのベクトルを導入する.

$$\begin{aligned} &\boldsymbol{y} = (y_0, y_1, y_2, \cdots\cdots, y_n)^T \\ &\boldsymbol{y_0} = (y_{00}, y_{10}, y_{20}, \cdots\cdots, y_{n0})^T \\ &\boldsymbol{f}(\boldsymbol{y}) = (f_0(\boldsymbol{y}), f_1(\boldsymbol{y}), \cdots\cdots, f_n(\boldsymbol{y}))^T \\ &\text{ただし} f_i(\boldsymbol{y}) = f_i(y_0, y_1, \cdots\cdots, y_n) \\ &\qquad\qquad i = 0,1,\cdots\cdots, n \end{aligned} \tag{4.3}$$

また, y'_i (i=0, 1,⋯, n) を要素にもつベクトルを $\boldsymbol{y}'$ で表すことにする. このような記号を用いれば, (4.2) は次の (4.4) のように簡潔に書くことができる.

$$\boldsymbol{y}' = \boldsymbol{f}(\boldsymbol{y}), \boldsymbol{y_0} = \boldsymbol{f}(x_0) \tag{4.4}$$

a. ハミングの公式
x=x_j における計算解と真の解をそれぞれ $\boldsymbol{y_j}$, $\boldsymbol{y}(x_j)$ と区別して書き, それに対応して微係数も $\boldsymbol{y}'_j$=$\boldsymbol{f}(\boldsymbol{y}_j)$, $\boldsymbol{y}'(x_j)$=$\boldsymbol{f}(\boldsymbol{y}(x_j))$ とを区別して書くことにする.
今, x_{k-3}=x_k-3h, x_{k-2}=x_k-2h, x_{k-1}=x_k-h, x_k の等間隔な 4 点における計算解と微係数の組,

$$\boldsymbol{y}_{k-3}, \boldsymbol{y}_{k-2}, \boldsymbol{y}_{k-1}, \boldsymbol{y}_k$$
$$\boldsymbol{y}'_{k-3}, \boldsymbol{y}'_{k-2}, \boldsymbol{y}'_{k-1}, \boldsymbol{y}'_k$$

が既知であるとする. この時, $\boldsymbol{y}_{k+1}$ は $\boldsymbol{p}_{k+1}$, $\boldsymbol{m}_{k+1}$, $\boldsymbol{c}_{k+1}$ なる量を次の手順で計算することにより求める.

① $\boldsymbol{p}_{k+1}$ を計算する.

$$\boldsymbol{p}_{k+1} = \boldsymbol{y}_{k-3} + \frac{4h}{3}\left(2\boldsymbol{y}'_k - \boldsymbol{y}'_{k-1} + 2\boldsymbol{y}'_{k-2}\right) \quad (4.5)$$

この $\boldsymbol{P}_{k+1}$ は求めようとする $\boldsymbol{y}_{k+1}$ をある程度“予測”する値である．この意味で予測子(predictor)と呼ばれている．テイラーの定理を使えば,

$$\begin{aligned} \boldsymbol{y}\left(x_{k+1}\right) &= \boldsymbol{y}\left(x_{k-3}\right) + \frac{4h}{3}\left[2\boldsymbol{y}'\left(x_k\right) - \boldsymbol{y}'\left(x_{k-1}\right)\right. \\ &\left. + 2\boldsymbol{y}'\left(x_{k-2}\right)\right] + \frac{14}{45}h^5\boldsymbol{y}^{(5)}\left(\eta_1\right) \end{aligned} \quad (4.6)$$

と表せる．ここで η_1 は，$\boldsymbol{y}$ の要素により異なり x_{k-3} と x_{k+1} の間の値である． (4.6) 式より，$\boldsymbol{p}_{k+1}$ は打切り誤差 $(14/45)h^5\boldsymbol{y}^{(5)}(\eta_1)$ を持つという．このことは，(4.5) の右辺に現われる $\boldsymbol{y}_i$, $\boldsymbol{y}_i$’ が仮に真値であったとしても，かつ丸め誤差の影響が全くないとしても $\boldsymbol{p}_{k+1}$ は $(14/45)h^5\boldsymbol{y}^{(5)}(\eta_1)$だけの誤差を持つことを意味する．

② $\boldsymbol{m}_{k+1}$ を計算する．

$$\boldsymbol{m}_{k+1} = \boldsymbol{p}_{k+1} + \frac{112}{121}\left(\boldsymbol{c}_k - \boldsymbol{p}_k\right) \quad (4.7)$$

（ただし $\boldsymbol{c}_k$ は後で明らかにする）

この $\boldsymbol{m}_{k+1}$ は $\boldsymbol{p}_{k+1}$ を“改良”した値である．この意味でここでは改良子 (modifier) と呼んでおく．

(4.7) の右辺第 2 項は $\boldsymbol{p}_{k+1}$ がもつ打切り誤差の推定量である．（その導出方法は省略する）．

③ $\boldsymbol{c}_{k+1}$ を計算する．

$$\begin{aligned} &\boldsymbol{m}'_{k+1} = \boldsymbol{f}\left(\boldsymbol{m}_{k+1}\right)\text{として} \\ &\boldsymbol{c}_{k+1} = \frac{1}{8}\left(9\boldsymbol{y}_k - \boldsymbol{y}_{k-2}\right) + \frac{3h}{8}\left(m'_{k+1} + 2\boldsymbol{y}'_k - \boldsymbol{y}'_{k-1}\right) \end{aligned} \quad (4.8)$$

この $\boldsymbol{c}_{k+1}$ は $\boldsymbol{p}_{k+1}$ を最終的に“修正”した値である．この意味で修正子 (corrector) と呼ばれている．(4.6) と同様な考え方で，$\boldsymbol{c}_{k+1}$ の打切り誤差は $(-1/40)h^5\boldsymbol{y}^{(5)}(\eta_2)$ となることが導かれる．

そしてこの $\boldsymbol{c}_{k+1}$ が所要の精度をもてば $x=x_{k+1}$ における解，すなわち $\boldsymbol{y}_{k+1}$ として採用する．

b. 出発値の計算

上記の方法は計算をすすめるのに過去の 4 点における情報を使う．したがって計算の初めにおいては

$\boldsymbol{y}'_0, \boldsymbol{y}'_1, \boldsymbol{y}'_2, \boldsymbol{y}'_3$
$\boldsymbol{y}'_0, \boldsymbol{y}'_1, \boldsymbol{y}'_2, \boldsymbol{y}'_3$

を特別な方法であらかじめ計算しておかなくてはならない．本ルーチンはそれらをルンゲ・クッタ･ギル法(サブルーチン RKG) により計算する．更に $\boldsymbol{m}_4$ を計算するのに
$\boldsymbol{c}_3$-$\boldsymbol{p}_3$
なる量を必要とするがこのときに限り，特に $\boldsymbol{c}_3$-$\boldsymbol{p}_3$=0 としている．

c. $\boldsymbol{c}_{k+1}$ の誤差評価

前に述べた予測子 $\boldsymbol{p}_{k+1}$, 修正子 $\boldsymbol{c}_{k+1}$ の打切り誤差は，それぞれ $(14/45)\ h^5\boldsymbol{y}^{(5)}\ (\eta_1)$, $(-1/40)\ h^5\boldsymbol{y}^5\ (\eta_2)$, であった．すなわち丸め誤差がこれらの誤差に比べて無視できるなら (4.9) のように書ける．

$$\begin{aligned} \boldsymbol{y}\left(x_{k+1}\right) &= \boldsymbol{p}_{k+1} + \frac{14}{45}h^5\boldsymbol{y}^{(5)}\left(\eta_1\right) \\ \boldsymbol{y}\left(x_{k+1}\right) &= \boldsymbol{c}_{k+1} - \frac{1}{40}h^5\boldsymbol{y}^{(5)}\left(\eta_2\right) \end{aligned} \quad (4.9)$$

更に $\boldsymbol{y}^{(5)}(x)$ が η_1 と η_2 の間で大きく変化しないと仮定すれば，(4.9)より

$$h^5\boldsymbol{y}^{(5)}\left(\eta_1\right) \approx h^5\boldsymbol{y}^{(5)}\left(\eta_2\right) \approx \frac{360}{121}\left(\boldsymbol{c}_{k+1} - \boldsymbol{p}_{k+1}\right)$$

となり結局 $\boldsymbol{c}_{k+1}$ の打切り誤差は

$$-\frac{1}{40}h^5\boldsymbol{y}^{(5)}\left(\eta_2\right) \approx -\frac{9}{121}\left(\boldsymbol{c}_{k+1} - \boldsymbol{p}_{k+1}\right) \quad (4.10)$$

となる．

d. キザミのコントロール

一定のキザミで計算を進めていくと，ある場所では，所要の精度を達成できないことがある．また逆に所要の精度よりもはるかに精度の良い解を求めすぎて場合によっては打切り誤差よりも丸め誤差の影響が大きくなり，その結果，計算解の下位の桁が振動することもある.この理由により必要に応じてキザミの大きさをコントロールしなければならない.

解として採用する修正子 $\boldsymbol{c}_{k+1}$ の持つ打切り誤差が (4.10) で与えられることからキザミを次のようにコントロールする.

(a) もしすべての要素に対して

$$\frac{9}{121}\left|\boldsymbol{c}_{k+1}-\boldsymbol{p}_{k+1}\right| \le \mathrm{EPS}\cdot\max\left(\left|\boldsymbol{y}_k\right|,\left|\boldsymbol{c}_{k+1}\right|\right) \quad (4.11)$$

となったときは $\boldsymbol{c}_{k+1}$ を解として採用する．ここで EPS は指定された相対誤差の上限であり，標準値は $64u$ である．

(b) もしすべての要素に対して

$$\frac{9}{121}\left|\boldsymbol{c}_{k+1}-\boldsymbol{p}_{k+1}\right| \le \mathrm{TOL}\cdot\max\left(\left|\boldsymbol{y}_k\right|,\left|\boldsymbol{c}_{k+1}\right|\right) \quad (4.12)$$

ただし TOL=EPS/32
となったら $\boldsymbol{c}_{k+1}$ を解として採用するが，次の段階ではキザミを 2 倍にして計算を試みる．
このとき新たに以前の $\boldsymbol{y}_{k-5}$ の値を必要とする．このため過去の $\boldsymbol{y}_{k-5}$ までの情報は可能なかぎり蓄わえている．

(c) もしある要素に対して

$$\frac{9}{121}\left|\boldsymbol{c}_{k+1}-\boldsymbol{p}_{k+1}\right| > \mathrm{EPS}\cdot\max\left(\left|\boldsymbol{y}_k\right|,\left|\boldsymbol{c}_{k+1}\right|\right) \quad (4.13)$$

となったときは $\boldsymbol{c}_{k+1}$ を解とせず，キザミを 1/2 倍にして計算しなおす．このとき新たに $\boldsymbol{y}_{k-1/2}$， $\boldsymbol{y}_{k-3/2}$ が必要となるが，それらはそれぞれ $\boldsymbol{y}_{k-1}$，$\boldsymbol{y}_{k-2}$ を出発値としてルンゲ・クッタ･ギル法により求める．
以上の (b)，(c) のいずれかによりキザミが変化したときは (4.7) の $\boldsymbol{c}_k$-$\boldsymbol{p}_k$ なる量を次の (4.14) より計算し直す．

$$c_k - p_k = -\frac{242}{27}\left[y_k - y_{k-3} - \frac{3\mathrm{h}}{8}\left(y'_{k-3} + 3y'_{k-2} + 3y'_{k-1} + y'_k\right)\right] \quad (4.14)$$

ここで右辺の h， $\boldsymbol{y}_i$， $\boldsymbol{y}'_i$ はすべてキザミが変化したあとの量である．

e. XEND における解の計算法
今まで述べた方法により計算を進めていくと，XEND における解は必ずしも求まらない．これを補うため，本サブルーチンでは XEND における解を以下に示す方法で 求める．
独立変数 x が XEND を越える一歩手前の状態を考え，その時の x の値を x_i，キザミを h とする．（図 HAMG-1 参照）

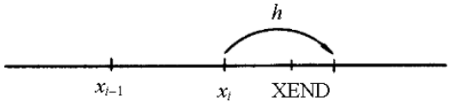


図 HAMNG-1 最終の段階 (h>0 のとき)

XEND, x_i, h の間には

$[\mathrm{XEND}-(x_i+h)]h<0$

なる関係が成り立っている．この状態に達したらハミング法をもはや使わず，x_i を出発値としてルンゲ･クッタ･ギル法を使う．このときキザミを (XEND-x_i)/2 とした場合と (XEND -x_i) とした場合との二つの近似解を計算することにより，XEND における解 $\boldsymbol{y}_e$ を求める．それを (4.15)に示す．

$$\boldsymbol{y}_e = \boldsymbol{y}_e^{(2)} + \left(\boldsymbol{y}_e^{(2)} - \boldsymbol{y}_e^{(1)}\right)/15 \quad (4.15)$$

ただし

$\boldsymbol{y}_e^{(2)}$ ： キザミを(XEND-x_i)/2 として得られた XEND における近似解

$\boldsymbol{y}_e^{(1)}$ ： キザミを(XEND-x_i)として得られた XEND における近似解

なお，詳細については，参考文献 [70], を参照すること．

B21-11-0602　　HBK1, DHBK1

実行列の固有ベクトルへの逆変換と正規化
CALL HBK1 (EV, K, N, IND, M, P, PV, DV, ICON)

(1)　機能

n 次への実ヘッセンベルグ行列 $\boldsymbol{H}$ の m 個の固有ベクトルを，実行列 $\boldsymbol{A}$ の固有ベクトルへ逆変換し，$\|x\|_2 = 1$ となるように正規化する．ただし，$\boldsymbol{H}$ は $\boldsymbol{A}$ にハウスホルダー法を適用して得られたものであること．かつ $n \geq 1$ であること．

(2)　パラメタ

EV ……**入力**．実ヘッセンベルグ行列 $\boldsymbol{H}$ の m 個の固有ベクトル（使用上の注意参照）．
EV(K，M) なる 2 次元配列．
出力．実行行列 $\boldsymbol{A}$ の固有ベクトル．

K……**入力**．配列 EV，P の整合寸法（$\geq n$）．

N……**入力**．実ヘッセンベルグ行列 $\boldsymbol{H}$ の次数 n.

IND……**入力**．EV に格納する各固有ベクトルが，実固有ベクトルか複素固有ベクトルかの指示．
EV の J 列目が，実固有ベクトルならば IND(J)=1，複素固有ベクトルの実部ならば IND(J)=-1，虚部ならば IND(J)=0 と与えること．大きさ M の 1 次元配列．

M……**入力**．配列 IND の大きさ($m \leq M \leq n$).

P……**入力**．ハウスホルダー法による $\boldsymbol{A}$ から $\boldsymbol{H}$ への変換行列（使用上の注意参照）．
P(K，N) なる 2 次元配列．

PV……**入力**．ハウスホルダー法による $\boldsymbol{A}$ から $\boldsymbol{H}$ への変換行列（使用上の注意参照）．
大きさ n の 1 次元配列．

DV……**入力**．実行列 $\boldsymbol{A}$ の平衡化に使われたスケーリングファクタ．大きさ n の 1 次元配列．
平衡化が行われなかったときは，DV は 1 次元配列である必要はなく，このとき DV=0.0 と指示できる．

ICON……**出力**．コンディションコード
表 HBK1-1 参照．

表 HBK1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N=1であった．	EV(1, 1)=1.0 とする．
30000	N<M，M<1 又は K<N であった．	処理を打ち切る．

(3)　使用上の注意

a.　使用する副プログラム

①　SSL II…NRML，MGSSL

②　FORTRAN 基本関数…ABS，SQRT

b.　注意

①　固有ベクトルの EV への格納は，実固有ベクトルは 1 列毎に，複素固有ベクトルは実部と虚部を対にして 2 列毎に格納すること．
（図 HBK1-1 参照のこと）
なお，サブルーチン HVEC 呼び出し後のパラメタ EV, IND, M は，本サブルーチンでそのまま入力できるようになっている．

②　サブルーチン HES1 のパラメタ A と PV は，本サブルーチンのパラメタ P と PV に対応しているので，そのまま使用できる．

③　ハウスホルダー法による変換行列の情報を配列 P と PV へ，図 HBK1-2 のように与えること．
ただし，$a_{ij}^{(k)}$ は，

$$\boldsymbol{A}_{k+1} = \boldsymbol{P}_k^T \boldsymbol{A}_k \boldsymbol{P}_k, k = 1,2,...,n-2 \qquad (3.1)$$

の計算を行うときに用いた A_k の要素であり，σ_k は，(3.2)により求める．

$$\sigma_k = \left(a_{k+1k}^{(k)}\right)^2 + \left(a_{k+2k}^{(k)}\right)^2 + ... + \left(a_{nk}^{(k)}\right)^2 \qquad (3.2)$$

P_k については，HES1 参照のこと．

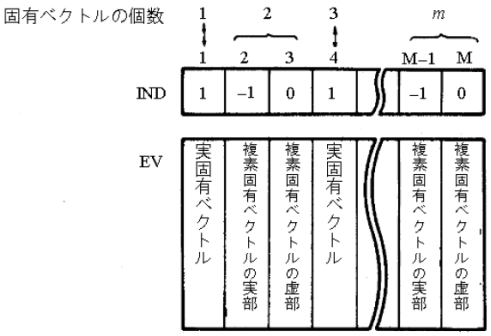
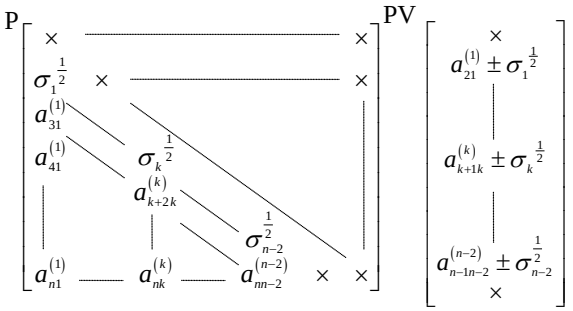


図 HBK1-1 IND とEV の関係



［注意］ × は使用されない．

図 HBK1-2 P，PVの入力方法

④　平衡化によるスケーリングファクタ DV の内容については，サブルーチン BLNC 参照のこと．

c. 使用例

n 次の実行列の固有値と固有ベクトルを，以下のサブルーチンを用いて計算する．ただし固有ベクトルは，固有値の求まった順に計算する．

BLNC... 実行列の平衡化.

HES1... 平衡化した実行列から，実ヘッセンベルグ行列へ変換.

HSQR... 実ヘッセンベルグ行列の固有値.

HVEC... 実ヘッセンベルグ行列の固有ベクトル.

HBK1... 実ヘッセンベルグ行列の固有ベクトルを，実行列の固有ベクトルへ逆変換して，正規化を行う.

$n \leq 100$， $m \leq 10$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),DV(100),
     * PV(100),IND(100),ER(100),EI(100),
     * AW(100,104),EV(100,100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL BLNC(A,100,N,DV,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
      CALL HES1(A,100,N,PV,ICON)
      MP=N
      DO 35 II=1,N
      I=1+N-II
      DO 30 J=1,MP
   30 AW(J,I)=A(J,I)
   35 MP=I
      CALL HSQR(AW,100,N,ER,EI,M,ICON)
      WRITE(6,620)ICON
      IF(ICON.EQ.20000) GO TO 10
      DO 40 I=1,M
   40 IND(I)=1
      CALL HVEC(A,100,N,ER,EI,IND,M,EV,
     *          100,AW,ICON)
      WRITE(6,620)ICON
      IF(ICON.GE.20000) GO TO 10
      CALL HBK1(EV,100,N,IND,M,A,PV,DV,
     *          ICON)
      CALL EPRT(ER,EI,EV,IND,100,N,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'
     * //11X,'** ORDER =',I5)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT(/11X,'** CONDITION CODE =',
     * I5/)
      END
```

本使用例中のサブルーチン EPRT は，実行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン EIG1 の使用例参照のこと.

(4) 手法概要

n 次の実ヘッセンベルグ行列 $\boldsymbol{H}$ の *m* 個の固有ベクトルを，平衡化した実行列 $\tilde{\boldsymbol{A}}$ の固有ベクトルへ逆変換し，更に平衡化を行う前の実行列 A の固有ベクトルへ逆変換して $\|\boldsymbol{x}\|_2 = 1$ の正規化を行う.

実行列 $\boldsymbol{A}$ の平衡化は，(4.1)に示す対角相似変換により行い，平衡化された実行列 $\tilde{\boldsymbol{A}}$ の実ヘッセンベルグ行列への変換は，ハウスホルダー法により (4.2)に示す *n*-2 回の直交相似変換により行う.

$$\tilde{\boldsymbol{A}} = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D} \tag{4.1}$$

(ただし，$\boldsymbol{D}$ は対角行列.)

$$\boldsymbol{H} = \boldsymbol{P}_{n-2}^{T}\ldots\boldsymbol{P}_{2}^{T}\boldsymbol{P}_{1}^{T}\tilde{\boldsymbol{A}}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2} \tag{4.2}$$

(ただし， $\boldsymbol{P}_i = \boldsymbol{I} - \boldsymbol{u}_i\boldsymbol{u}_i^{T}/h_i$)

H の固有値，固有ベクトルを，λ , $\boldsymbol{y}$ とすると，

$$\boldsymbol{H}\boldsymbol{y}=\lambda\boldsymbol{y} \tag{4.3}$$

が成り立つ．(4.1)と(4.2)の関係から，(4.3)は，

$$\boldsymbol{P}_{n-2}^{T}\ldots\boldsymbol{P}_{2}^{T}\boldsymbol{P}_{1}^{T}\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{y} = \lambda\,\boldsymbol{y} \tag{4.4}$$

となり，(4.4)の両辺に $\boldsymbol{D}\boldsymbol{P}_1\boldsymbol{P}_2\ldots\boldsymbol{P}_{n-2}$, を左側より掛けると(4.5)となる.

$$\boldsymbol{A}\boldsymbol{D}\boldsymbol{P}_1\boldsymbol{P}_2\ldots\boldsymbol{P}_{n-2}\boldsymbol{y} = \lambda\boldsymbol{D}\boldsymbol{P}_1\boldsymbol{P}_2\ldots\boldsymbol{P}_{n-2}\boldsymbol{y} \tag{4.5}$$

したがって, A の固有ベクトル $\boldsymbol{x}$ は, (4.6)となる.

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{P}_1\boldsymbol{P}_2\ldots\boldsymbol{P}_{n-2}\boldsymbol{y} \tag{4.6}$$

(4.6)の計算は，(4.7), (4.8)により行う．ただし，$\boldsymbol{y}=\boldsymbol{x}_{n-1}$ とする.

$$\begin{aligned}\boldsymbol{x}_i &= P_i\boldsymbol{x}_{i+1} \\ &= \boldsymbol{x}_{i+1} - \left(\boldsymbol{u}_i\boldsymbol{u}_i^{T}/h_i\right)\boldsymbol{x}_{i+1}, \quad i = n-2,\ldots,2,1\end{aligned} \tag{4.7}$$

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{x}_1 \tag{4.8}$$

ハウスホルダー法及び平衡化の詳細は，サブルーチン HES1, BLNC 参照のこと．固有ベクトルの正規化は，サブルーチン NRML を用いて行っている.

なお，詳細については，参考文献 [13] の pp339-358 を参照すること.

B21-25-0201　HEIG2, DHEIG2

エルミート行列の固有値及び固有ベクトル（ハウスホルダー法, バイセクション法, 逆反復法）
CALL HEIG2　(A, K, N, M, E, EVR, EVI, VW, ICON)

(1)　機能

n 次のエルミート行列 ***A*** の固有値を，バイセクション法により，大きい方から又は小さい方から *m* 個求め，対応する固有ベクトルを逆反復法により求める．固有ベクトルは，$\|\boldsymbol{x}\|_2 = 1$ となるように正規化する．

$1 \le m \le n$ なること．

(2)　パラメタ

A……**入力**．エルミート行列 *A*.
エルミート行列用圧縮モード.
A(K，N)なる 2 次元配列.
演算後，内容は保存されない.

K……**入力**．配列 A，EVR，EVI の整合寸法 (≥*n*).

N……**入力**．エルミート行列の次数 *n* .

M……**入力**．求める固有値の個数 *m.*
M=+*m* のときは大きい方から求める.
M=- *m* のときは小さい方から求める.

E……**出力**．エルミート行列 *A* の固有値.
大きさ *m* の 1 次元配列.

EVR，EVI……**出力**．EVR には固有ベクトルの実数部，EVI には固有ベクトルの虚数部を，列方向に格納する.
第 *j* 番目の固有値 E(J) に対応する固有ベクトルの，第 *l* 番目の要素は
EVR(L, J)+*i* · EVI(L, J)と表せる．ただし，
$i = \sqrt{-1}$ である．EVR(K, *m*),
EVI(K, *m*)なる 2 次元配列.

VW……**作業領域**．大きさ *9n* の 1 次元配列.

ICON……**出力**．コンディションコード.
表 HEIG2-1 参照.

表 HEIG2-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	N=1 であった.	E(1)=A(1, 1) EVR(1, 1) =1.0 EVI(1.1)=0.0 とする.
15000	固有ベクトルのうち求まらないものがあった.	その固有ベクトルを 0 ベクトルとする.
20000	固有ベクトルは全て求まらなかった.	固有ベクトル は全て 0 ベクトルとする.
30000	M=0，N<|M| 又は K<N であった.	処理を打ち切る.

(3)　使用上の注意

a.　使用する副プログラム

① SSL II... AMACH, TEIG2, TRBKH, TRIDH, MGSSL, UTEG2

② FORTRAN 基本関数 ... IABS, SQRT, ABS, AMAX1

b.　注意

① 本サブルーチンはエルミート行列用であり，一般複素行列については使用できない.

② 本サブルーチンは固有値および固有ベクトルを求める場合に使用し，固有値だけを求める場合は，サブルーチン TRIDH と BSCT1 を用いること.

c.　使用例

n 次のエルミート行列 ***A*** の固有値を大きい方から（または小さい方から） *m* 個，及びそれに対応する固有ベクトルをそれぞれ求める.
n≤100，*m*≤ 10 の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),E(10),
     * EVR(100,10),EVI(100,10),VW(900)
   10 CONTINUE
      READ(5,500) N,M
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,M
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL HEIG2(A,100,N,M,E,EVR,EVI,VW,
     *           ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      MM=IABS(M)
      CALL HEPRT(E,EVR,EVI,100,N,MM)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'///40X,'**ORIGINAL ',
     * 'MATRIX**',5X,'N=',I3,5X,'M=',I3//)
  610 FORMAT(/4(4X,'A(',I3,',',I3,')=',
     * E15.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン HEPRT は，エルミート行列の固有値及び固有ベクトルを出力するサブルーチンである．以下にその内容を示す.

```
      SUBROUTINE HEPRT(E,EVR,EVI,K,N,M)
      DIMENSION E(M),EVR(K,M),EVI(K,M)
      WRITE(6,600)
      DO 10 I=1,M
   10 WRITE(6,610) I,E(I)
      KAI=(M-1)/3+1
      LST=0
      WRITE(6,620)
      DO 20 I=1,KAI
      INT=LST+1
      LST=LST+3
      IF(LST.GT.M) LST=M
      WRITE(6,630) (J,J=INT,LST)
      WRITE(6,640)
```

```
      DO 20 J=1,N
   20 WRITE(6,650) J,(EVR(J,L),EVI(J,L),
     *             L=INT,LST)
      RETURN
  600 FORMAT('1'///5X,'**HERMITIAN ',
     * 'MATRIX**',///
     * 5X,'**EIGENVALUES**'/)
  610 FORMAT(5X,'E(',I3,')=',E20.8)
  620 FORMAT('0',///5X,
     * '**EIGENVECTORS**'//)
  630 FORMAT('0',3X,3(20X,'X(',I3,')',
     * 14X))
  640 FORMAT(//13X,3('REAL PART',10X,
     * 'IMAG. PART',11X))
  650 FORMAT(1X,I3,6E20.8)
      END
```

(4) 手法概要

n 次のエルミート 行列 A の固有値，固有ベクトルを次の手順で求める．

① エルミート 行列 の実対称 3 重対角行列への変換
エルミート 行列 $\boldsymbol{A}$ をハウスホルダー法により，エルミート 3 重対角行列 $\boldsymbol{H}$ に変換し，

$$\boldsymbol{H}=\boldsymbol{P}^{*}\boldsymbol{A}\boldsymbol{P} \tag{4.1}$$

更に対角ユニタリ変換により，実対称 3 重対角行列 $\boldsymbol{T}$ に変換する．

$$\boldsymbol{T}=\boldsymbol{V}^{*}\boldsymbol{H}\boldsymbol{V} \tag{4.2}$$

ただし $\boldsymbol{P}$ はユニタリ行列であり，また $\boldsymbol{V}$ は対角ユニタリ行列である．

② 実対称 3 重対角行列の固有値，固有ベクトル
バイセクション法により $\boldsymbol{T}$ の m 個の固有値を求め，続いて逆反復法により対応する固有ベクトル $\boldsymbol{y}$ を求める．逆反復法は，

$$\left(\boldsymbol{T}-\mu\boldsymbol{I}\right)\boldsymbol{y}_{r}=\boldsymbol{y}_{r-1}\boldsymbol{r}=1,2,... \tag{4.3}$$

を反復的に解いて固有ベクトル $\boldsymbol{y}$ を求める方法である．ただし，(4.3)で μ はバイセクション法により求めた固有値で，$\boldsymbol{y}_0$ は適当な初期ベクトルである．

③ エルミート行列の固有ベクトル
$\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ は②により求めた固有ベクトル $\boldsymbol{y}$ から，

$$\boldsymbol{x}=\boldsymbol{P}\boldsymbol{V}\boldsymbol{y} \tag{4.4}$$

により得られる．(4.4) は (4.1)と (4.2)の逆変換である．

①は TRIDH，②は TEIG2，③は TRBKH の各サブルーチンを用いて行う．

詳細については，参考文献 [12], [13] pp. 259-269， 及び [17] を参照すること．

B21-11-0302 HES1, DHES1

実行列の実ヘッセンベルグ行列への変換 (ハウスホルダー法)
CALL HES1(A, K, N, PV, ICON)

(1) 機能

n 次の実行列 ***A*** を，ハウスホルダー法（直交相似変換）により実ヘッセンベルグ行列 ***H*** へ変換する．

$$\boldsymbol{H}=\boldsymbol{P}^{T}\boldsymbol{A}\boldsymbol{P}$$

ただし ***P*** は変換行列である．$n \geq 1$ であること．

(2) パラメタ

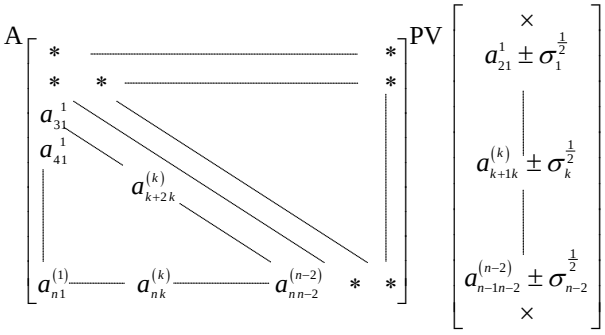
A…… **入力**．実行列 ***A***.
出力．実ヘッセンベルグ行列 ***H*** 及び変換行列 ***P*** (図 HES1-1 参照).
A (K, N)なる 2 次元配列.

K…… **入力**．配列 A の整合寸法 (≥*n*).

N…… **入力**．実行列 ***A*** の次数 *n*.

PV…… **出力**．変換行列 ***P*** (図 HES1-1 参照).
大きさ *n* の 1 次元配列.

ICON…… **出力**．コンディションコード.
表 HES1-1 参照.



[注意] * 部分はヘッセンベルグ行列であり，ほかは変換行列のための情報である．×は作業領域.

図 HES1-1 ハウスホルダー変換後の A と PV

表 HES1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1 又は N=2であった.	変換しない.
30000	K<N 又は N<1であった.	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH， MGSSL
② FORTRAN 基本関数 ... ABS， DSQRT， SIGN

b. 注意
① 出力された配列 A 及び PV は，実行列 ***A*** の固有値固有ベクトルを求めるために必要となる.
② 固有値の精度は，実ヘッセンベルグ行列化を行った時点である程度決定される．そのため，できるだけ精度よく，実ヘッセンベルグ行列を求めることが必要であり，本サブルーチンでも若干の考慮が払われている．しかし，固有値に非常に大きいものと小さいものがあるとき，小さい方の固有値は，大きい固有値に比べて変換の影響を受けやすい．したがって，小さい固有値を精度よく求めることが難しい場合がある.

c. 使用例
n 次の実行列 を実ヘッセンベルグ行列に変換した後，サブルーチン HSQR により固有値を計算する． $n \leq 100$ の場合，

```
C     **EXAMPLE**
      DIMENSION A(100,100),PV(100),
     *          ER(100),EI(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL HES1(A,100,N,PV,ICON)
      WRITE(6,620)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,610) ((I,J,A(I,J),J=1,N),
     *              I=1,N)
      CALL HSQR(A,100,N,ER,EI,M,ICON)
      WRITE(6,640)
      WRITE(6,630) ICON
      IF(ICON.EQ.20000) GO TO 10
      WRITE(6,650) (I,ER(I),I,EI(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'
     * //11X,'** ORDER =',I5/)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT('0'/11X,'** HESSENBERG ',
     * 'MATRIX')
  630 FORMAT(/11X,'** CONDITION CODE =',
     * I5/)
  640 FORMAT('0'/11X,'** EIGENVALUES')
  650 FORMAT(5X,'ER(',I3,')=',E14.7,
     * 5X,'EI(',I3,')=',E14.7)
      END
```

(4) 手法概要

n 次の実行列 $\boldsymbol{A}$ の実 ヘッセンベルグ 行列への変換は，次式で示されるような n-2 回の直交相似変換により行われる．

$$\boldsymbol{A}_{k+1} = \boldsymbol{P}_k{}^T \boldsymbol{A}_k \boldsymbol{P}_k, k = 1,2,...,n-2 \qquad (4.1)$$

ただし $\boldsymbol{A}_1$=$\boldsymbol{A}$,$\boldsymbol{P}_k$ は変換行列（かつ直交行列）である．

変換の終了した時点で $\boldsymbol{A}_{n\text{-}1}$ は，実ヘッセンベルグ行列となる．

第 k 回目の変換は，次の手順により行う．

$\boldsymbol{A}_k = \left(a_{ij}^{(k)}\right)$ として

$$\sigma_k = \left(a_{k+1k}^{(k)}\right)^2 + \left(a_{k+2k}^{(k)}\right)^2 + ... + \left(a_{nk}^{(k)}\right)^2 \qquad (4.2)$$

$$\mathbf{u}_k{}^T = \left(0,..,0,a_{k+1k}^{(k)} \pm \sigma_k{}^{\frac{1}{2}}, a_{k+2k}^{(k)}, ..., a_{nk}^{(k)}\right) \qquad (4.3)$$

$$h_k = \sigma_k \pm a_{k+1k}^{(k)} \sigma_k{}^{\frac{1}{2}} \qquad (4.4)$$

$$\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k{}^T / h_k \qquad (4.5)$$

(4.5) の $\boldsymbol{P}_k$ を (4.1) へ適用することによって $\boldsymbol{A}_k$ の $a_{k+2\,k}^{(k)} \sim a_{n\,k}^{(k)}$ を消去する．

実際の変換にあたっては，次のような考慮を払っている．

① (4.2)～(4.4)の計算におけるアンダフロー，オーバフローを防ぐために，(4.2) の右辺の各要素は，

$$\sum_{i=k+1}^{n} \left|a_{ik}^{(k)}\right|$$

でスケーリングしたものを用いる．

② (4.3)の $\boldsymbol{u}_k{}^T$ を求めるとき，$a_{k+1\,k}^{(k)} \pm \sigma_k{}^{1/2}$ の計算で桁落ちが生じないようにするため $\pm\sigma_k{}^{1/2}$ は，$a_{k+1\,k}^{(k)}$ と同符号になる方を採用する．

③ (4.1)の変換は，$\boldsymbol{P}_k$ を求めて行うかわりに，(4.1) を (4.6), (4.7)のように展開して $\boldsymbol{A}_{k+1}$ を求める．

$$\boldsymbol{B}_{k+1} = \boldsymbol{P}_k{}^T \boldsymbol{A}_k = \boldsymbol{A}_k - \boldsymbol{u}_k \left(\boldsymbol{u}_k{}^T \boldsymbol{A}_k\right) / h_k \qquad (4.6)$$

$$\boldsymbol{A}_{k+1} = \boldsymbol{B}_{k+1} \boldsymbol{P}_k = \boldsymbol{B}_{k+1} - \left(\boldsymbol{B}_{k+1} \boldsymbol{u}_k / h_k\right) \boldsymbol{u}_k{}^T \qquad (4.7)$$

また変換行列 $\boldsymbol{P}_k$. は，実行列 A の固有ベクトルを求めるときにも必要となる．そのため (4.3) によって求まる u_k の要素を配列 $\boldsymbol{A}$ 及び 1 次元配列 PV に図 HES1-2 のような形で保存しておく．

n=2 又は n=1 のときは変換しない．

なお，詳細については，参考文献 [13] の pp.339-358 を参照すること．

F20-02-0201 HRWIZ, DHRWIZ

Hurwitz 多項式の判定
CALL HRWIZ (A, NA, ISW, IFLG, SA, VW, ICON)

(1) 機能

n 次の実係数多項式

$$P(s)=a_1s^n+a_2s^{n-1}+.....+a_ns+a_{n+1}$$

が Hurwitz 多項式（すべての零点が複素平面の左半面，すなわち，Re(*s*)<0 の領域に存在する多項式）であるかどうかを判定する．*P*(*s*)が Hurwitz 多項式でない場合は$\alpha>\alpha_0(\geq 0)$で $P(s+\alpha)$が Hurwitz 多項式となるα_0を探索する．

(2) パラメタ

A…… **入力**．*P*(*s*)の係数．
大きさ *n*+1 の 1 次元配列．A(1)=a_1,
A(2)=a_2,…, A(*n*+1)=a_{n+1}の順に与える．

NA…… **入力**．*P*(*s*)の次数 *n*.

ISW …… **入力**．制御情報．
ISW=0: *P*(*s*)が Hurwitz 多項式か否かの判定だけを行う．.
ISW=1: *P*(*s*)が Hurwitz 多項式か否かの判定を行い，かつ Hurwitz 多項式でない場合には α_0 を探索する．
上記以外の値を入力すると，ISW は ISW=1 とみなして処理する．

IFLG…… **出力**．多項式の判定結果．
IFLG=0: Hurwitz 多項式である．
IFLG=1: Hurwitz 多項式でない．

SA…… **出力**． α_0 の値．*P*(*s*)が Hurwitz 多項式の場合は SA= 0.0 と出力する．

VW…… **作業領域**．大きさ *n*+1 の 1 次元配列．

ICON…… **出力**．コンディションコード．
表 HRWIZ-1 参照．

表 HRWIZ-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	α_0 の値が見つからなかった．	処理を打ち切る．
30000	NA<1，又は A(1)=0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH， MGSSL

② FORTRAN 基本関数...ABS，ALOG10

b. 注意

① 本サブルーチンの機能は，有理関数 *F*(*s*)=*Q*(*s*)/*P*(*s*)のラプラス逆変換 *f*(*t*) を求めることに関連し，*f*(*t*) の大ざっぱな振舞を調べる目的で利用できる．すなわち，*P*(*s*)が Hurwitz 多項式でないときは，*F*(*s*) が Re(*s*)≥0 の領域に特異点を持つことになり，その結果，逆変換 *f*(*t*) は *t* →∞のとき指数関数的に増大する．
有理関数 *F*(*s*)のラプラス逆変換 *f*(*t*) を求めるには，Re(*s*)>0 における *F*(*s*)の正則性が既知か否かに応じてサブルーチン LAPS1，又は LAPS2 を使用すればよい．ラプラス逆変換の数値計算法については，8 章の「ラプラス変換」に述べてある．

c. 使用例
多項式 $P(s)=s^4-12s^3+54s^2-108s+81$ が， Hurwitz 多項式であるか否かを判定する．

```
C     **EXAMPLE**
      DIMENSION A(5),VW(5)
      NA=4
      NA1=NA+1
      A(1)=1.0
      A(2)=-12.0
      A(3)=54.0
      A(4)=-108.0
      A(5)=81.0
      ISW=1
      WRITE(6,600) NA,(I,A(I),I=1,NA1)
      CALL HRWIZ(A,NA,ISW,IFLG,SA,VW,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0)STOP
      WRITE(6,620)ISW,IFLG,SA
      STOP
  600 FORMAT(//24X,'NA=',I3//
     *(20X,'A(',I3,')=',E15.8/))
  610 FORMAT(/24X,'ICON=',I6)
  620 FORMAT(/24X,'ISW=',I2,5X,
     *'IFLG=',I2,5X,'SA=',E15.8)
      END
```

(4) 手法概要

Hurwitz 多項式の判定法については「8.6 ラプラス変換」で述べてあるので，そこを参照すること．

B21-11-0402 HSQR, DHSQR

実ヘッセンベルグ行列の固有値 (2 段 QR 法)
CALL HSQR (A, K, N, ER, EI, M, ICON)

(1) 機能

n 次の実ヘッセンベルグ行列 $\boldsymbol{A}$ の固有値を，2 段 QR 法により求める．ただし $n \geq 1$ であること．

(2) パラメタ

A …… **入力**．実ヘッセンベルグ行列 $\boldsymbol{A}$.
演算後，内容は保存されない．
A(K，N)なる 2 次元配列.

K …… **入力**．配列 A の整合寸法 ($\geq n$).

N …… **入力**．実ヘッセンベルグ行列 $\boldsymbol{A}$ の次数 n.

ER, EI …… **出力**．固有値． J 番目の固有値は，
ER(J)+i EI(J) (J=1，2， ...，M)である．
ただし $i=\sqrt{-1}$
大きさ n の 1 次元配列.

M …… **出力**．求まった固有値の個数.

ICON …… **出力**．コンディションコード.
表 HSQR-1 参照.

表 HSQR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1 であった.	ER(1)=A(1, 1)，EI(1)=0.0 とする.
15000	固有値のすべてを求めつくすことはできなかった.	M に求まった固有値の個数をセットする. 1≤M<N.
20000	固有値が，一つも求まらなかった.	M = 0 とする.
30000	K<N 又は N <1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSLII...AMACH，MGSSL
② FORTRAN 基本関数...ABS，SQRT，SIGN

b. 注意
① 通常，サブルーチン HES1 を実行した後，本サブルーチンによって固有値.を求める.
② 固有ベクトルをも必要とする場合は，本サブルーチンを呼び出す前に，配列 A の内容を他の領域へ移しておくこと.

c. 使用例
n 次の実行列を，サブルーチン HES1 によって実ヘッセンベルグ行列へ変換した後，固有値を計算する． $n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),PV(100),
     *          ER(100),EI(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL HES1(A,100,N,PV,ICON)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      CALL HSQR(A,100,N,ER,EI,M,ICON)
      WRITE(6,630)
      WRITE(6,620) ICON
      IF(ICON.EQ.20000) GO TO 10
      WRITE(6,640) (I,ER(I),I,EI(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'
     * //11X,'** ORDER =',I5/)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT(/11X,'** CONDITION CODE =',
     * I5/)
  630 FORMAT('0'/11X,'** EIGENVALUES')
  640 FORMAT(5X,'ER(',I3,')=',E14.7,
     * 5X,'EI(',I3,')=',E14.7)
      END
```

(4) 手法概要

QR 法は，ヘッセンベルグ行列 $\boldsymbol{A}$ に，(4.1) の直交相似変換を逐次施すことによって，$\boldsymbol{A}$ の下副対角要素を零に収束させ，そのときの主対角要素を固有値とする方法である.

$$\boldsymbol{A}_{s+1}=\boldsymbol{Q}_s^T\boldsymbol{A}_s\boldsymbol{Q}_s \tag{4.1}$$

ここで $\boldsymbol{Q}_s$ は，(4.2) により $\boldsymbol{A}s$ を QR 分解したときに，一意的に定まる直交行列であり，$\boldsymbol{R}s$ は主対角要素が正の実数である上三角行列である.

$$\boldsymbol{A}_s=\boldsymbol{Q}_s\boldsymbol{R}_s \tag{4.2}$$

この(4.2)によって定まる $\boldsymbol{Q}_s$ を (4.1) に適用することによって，$\boldsymbol{A}$ 下副対角要素は徐々に零に収束する.

QR 法では通常収束を速めるため，$\boldsymbol{A}_s$ の代りに原点移動を行った $(\boldsymbol{A}_s-k_s\boldsymbol{I})$ に対して QR 分解を行い，そして直交相似変換を行う．ここで k_s は原点移動量である．しかしながら実行列の場合の固有値は，一般に実数と複素数とが混在するので k_s も複素数を選ぶ必要があり，複素行列の演算となる．そこで実行列の演算だけで行える 2 段 QR 法を利用する．これによると (4.1), (4.2) はそれぞれ (4.3), (4.4)のようになる.

$$\boldsymbol{A}_{s+2} = \left(\boldsymbol{Q}_s\boldsymbol{Q}_{s+1}\right)^T \boldsymbol{A}_s\left(\boldsymbol{Q}_s\boldsymbol{Q}_{s+1}\right) \tag{4.3}$$

$$\left(\boldsymbol{A}_s - k_s I\right)\left(\boldsymbol{A}_s - k_{s+1} I\right) = \left(\boldsymbol{Q}_s\boldsymbol{Q}_{s+1}\right)\left(\boldsymbol{R}_{s+1}\boldsymbol{R}_s\right) \tag{4.4}$$

すなわち k_s と k_{s+1} が複素共役なら (4.4) の左辺は，常に実行列として扱えることを示している．直交行列の積は，やはり直交行列となるから，(4.3)を行うにあったっては，

$$\boldsymbol{Q} = \boldsymbol{Q}_s\boldsymbol{Q}_{s+1} = \boldsymbol{P}_1\boldsymbol{P}_2.....\boldsymbol{P}_{n-1} \tag{4.5}$$

として(4.6)の形で行う．

$$\boldsymbol{A}_{s+2} = \boldsymbol{P}_{n-1}^T...\boldsymbol{P}_2^T\boldsymbol{P}_1^T\boldsymbol{A}_s\boldsymbol{P}_1\boldsymbol{P}_2...\boldsymbol{P}_{n-1} \tag{4.6}$$

ここで P_1 は (4.4) において QR 分解を行うとき，$R_{s+1}R_s$，の第 1 列の要素を定める変換行列である．また $\boldsymbol{P}_i(i=2, 3, ..., n-1)$は，(4.6) において $\boldsymbol{P}_1\boldsymbol{A}_s\boldsymbol{P}_1$ を $\boldsymbol{A}_{s+2}$ へ変換するために，ハウスホルダー法を適用するとき定まる一連の変換行列である．

以下に 2 段 QR 法の手順を示す．

① A_s の下副対角要素 $a^{(s)}{}_{nn-1}$，…,$a_{21}{}^{(s)}$に，零と見なせる要素があるか，(4.7)により調べる．

$$\left|a_{ll-1}^{(s)}\right| \le u\left(\left|a_{l-1l-1}^{(s)}\right| + \left|a_{ll}^{(s)}\right|\right), \quad l = n, n-1, ..., 2 \tag{4.7}$$

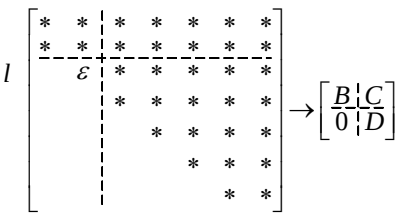
ただし u は丸め誤差の範囲である．

(4.7)を満足したとき，$\boldsymbol{a}_{l\,l-1}^{(s)}$ を零と見なす．満足しないときは，②へ進む．

(a) もし $l = n$であれば,$a_{nn}^{(s)}$を固有値とし， n を n-1 と行列の次数を減らし，①へ戻る．
もし n=0 となれば処理終了．

(b) もし l=n-1 であれば，右下すみの（2 × 2）の小行列より二つの固有値を求め，n を n-2 と行列の次数を減らし①へ戻る．
もし n=0 となれば処理終了．

(c) 2≤ l <n-1 のときは，
図 HSOR-1 のように行列を分解する．
そして $\boldsymbol{D}$ を $\boldsymbol{A}_s$ とし，②へ進む．



[注意]: ε は零と見なした要素.

図 HSQR-1 ヘッセンベルグ行列の直和分解

② $\boldsymbol{A}_s$ の右下すみの (2×2) の小行列の二つの固有値を移動量とし，それぞれ$\boldsymbol{k}_s$, $\boldsymbol{k}_{s+1}$ とする．

③ (4.4) の左辺 $\left(\boldsymbol{A}_s - k_s\boldsymbol{I}\right)\left(\boldsymbol{A}_s - k_{s+1}\boldsymbol{I}\right)$ の第 1 列を求める．$\boldsymbol{A}_s$ はヘッセンベルグ行列であるからその第 1 列目 $\boldsymbol{m}_1$ は，

$$\boldsymbol{m}_1=(x_1,\ y_1,\ z_1,\ 0,\ ,\ 0)^T \tag{4.8}$$

となる
ただし，

$$\left.\begin{aligned} x_1 &= a_{11}^{(s)2} - a_{11}^{(s)}\left(k_s + k_{s+1}\right) + k_s k_{s+1} + a_{12}^{(s)}a_{21}^{(s)} \\ y_1 &= a_{21}^{(s)}\left(a_{11}^{(s)} + a_{22}^{(s)} - k_s - k_{s+1}\right) \\ z_1 &= a_{32}^{(s)}a_{21}^{(s)} \end{aligned}\right\} \tag{4.9}$$

である．

④ $\boldsymbol{P}_1$を求める．$\boldsymbol{R}_{s+1}\boldsymbol{R}_s$の第1列を

$$\boldsymbol{r}_1 = \left(\sigma_1, 0, ..., 0\right)^T \tag{4.10}$$

$$\sigma_1 = \pm\sqrt{x_1{}^2 + y_1{}^2 + z_1{}^2} \tag{4.11}$$

とすると，次のように定めることができる．

$$\boldsymbol{P}_1 = \boldsymbol{I} - 2\boldsymbol{w}_1\boldsymbol{w}_1{}^T \tag{4.12}$$

ただし,

$$\boldsymbol{w}_1 = \left(\boldsymbol{m}_1 - \boldsymbol{r}_1\right)\big/\left\|\boldsymbol{m}_1 - \boldsymbol{r}_1\right\|_2 \tag{4.13}$$

ここで σ_1 の符号は，x_1 と異符号を選び桁落ちを防ぐ．

⑤ $\boldsymbol{P}_1{}^T\boldsymbol{A}_s\boldsymbol{P}_1$ を行う．$\boldsymbol{P}_1{}^T\boldsymbol{A}_s\boldsymbol{P}_1$ の形は，図 HSQR-2 のようになる．

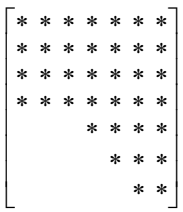


図 HSQR-2 $P_1{}^TA_sP_1$の形

⑥ $\boldsymbol{P}_1{}^T\boldsymbol{A}_s\boldsymbol{P}_1$ をハウスホルダー法によって，ヘッセンベルグ行列に変換し，①へ戻る．（ハウスホルダー法については，サブルーチン HES1 の手法概要参照のこと．）
この手順を何回か行うことによって，右下すみの下副対角要素は徐々に零に収束する．したがって，①の (a), (b) によって固有値が決まる．ただし連続 30 回行っても，①の (a), (b) によって固有値が求まらない場合，ICON=15000 若しくは，20000 とする．

なお，詳細については，参考文献 [12], [13] の PP.359-371 及び [16] の PP.177-206 を参照すること．

B21-11-0502 HVEC, DHVEC

実ヘッセンベルグ行列の固有ベクトル固有ベクトル(逆反復法)
CALL HVEC (A, K, N, ER, EI, IND, M, EV, MK, AW, ICON)

(1) 機能

n 次の実ヘッセンベルグ行列 ***A*** の指定された固有値 μ に対応する固有ベクトル ***x*** を，逆反復法により求める．固有ベクトルの正規化は行わない．

n≥1 なること．

(2) パラメタ

A ····· **入力**．実ヘッセンベルグ行列 ***A***.
A(K, N)なる 2 次元配列.

K ····· **入力**．配列 **A,** EV 及び AW の整合寸法 (≥*n*)

N ····· **入力**．実ヘッセンベルグ行列 ***A*** の次数 *n* .

ER, EI ····· **入力**．固有値 μ.
固有値 μ を実部と虚部とに分け，それぞれ ER 及び EI に格納しておくこと．
すなわち，第 *j* 番目の固有値 μ_j は，

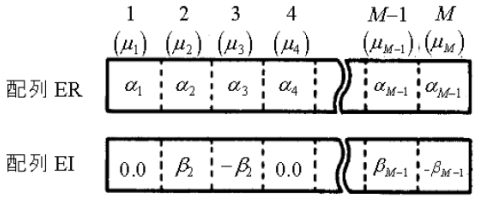
$$\mu_j=\mathrm{ER}(j)+i\cdot\mathrm{EI}(j)$$

であらわせる（ただし $i=\sqrt{-1}$）．
μ_j が複素数(EI(*j*)≠0)のときは，図 HVEC-1 のように μ_{j+1} と共役複素数の対になっていなければならない．
大きさ M の 1 次元配列.

IND ····· **入力**．固有ベクトルの要不要の指定.
μ_j に対応する固有ベクトルを要求するとき IND(J)=1 とし，不要のときは IND(J)=0 とする．ただし，μ_j と μ_{j+1} が複素共役関係にあるとき，IND(J+1)=1 としても 0 と見なされる．
（使用上の注意①参照）
大きさ M の 1 次元配列.

M ····· **入力**．配列 ER, EI に格納されている固有値の総数（≤*n*）.

EV ····· **出力**．固有ベクトル ***x***.
固有ベクトルを列方向に格納する．実固有値に対応する実固有ベクトルは，EV の 1 列に，複素固有値に対応する複素固有ベクトルは，実部と虚部とに分けて 2 列に格納する．
使用上の注意②参照のこと．
EV(K，MK)なる 2 次元配列.



ただし，固有値 μ_jは$\mu_j=\alpha_j+i\beta_j$とする.

図 HVEC-1 固有値 の格納法

MK ····· **入力**．配列 **EV** の列数.
使用上の注意参照のこと.

AW ····· **作業領域.**
AW(K，N+4)なる 2 次元配列.

ICON ····· **出力**．コンディションコード.
表 HVEC-1 参照.

表 HVEC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1 であった.	EV(1, 1)=1.0 とする.
15000	ある固有値に対応する固有ベクトルが求まらなかった.	求まらなかった固有ベクトルの IND の情報は消去される.
16000	配列 EV の列数が，要求されたすべての固有ベクトルを格納するには小さすぎた.	配列 EV に格納できるだけ計算する．計算されなかったものの IND の情報は消去される.
20000	すべての固有ベクトルが求まらなかった.	IND の情報はすべて消去される.
30000	M<1，N<M 又は K<N であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II...AMACH, MGSSL.
② FORTRAN 基本関数...ABS, SQRT, SIGN

b. 注意
① パラメタ IND と固有ベクトルの格納法.
図 HVEC-2 の配列 ER，EI のように，固有値が与えられたとき，同図のように IND を与えると，$\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6$, 及び μ_8 に対応する固有ベクトルが計算され，同図の EV のように格納される． μ_3 及び μ_7 に対応する固有ベクトルは計算されず IND(3)=0，IND (7)=0 とされる．なお，μ_2 及び μ_6 は複素固有値であるので，IND(2)=-1，IND(6) =-1 と変更される．
また図 HVEC-3 のように IND を与えると，固有ベクトルは同図の EV のように第 1 列から順に格納される．このとき，固有値の並びと対応した列には格納されないので注意すること．

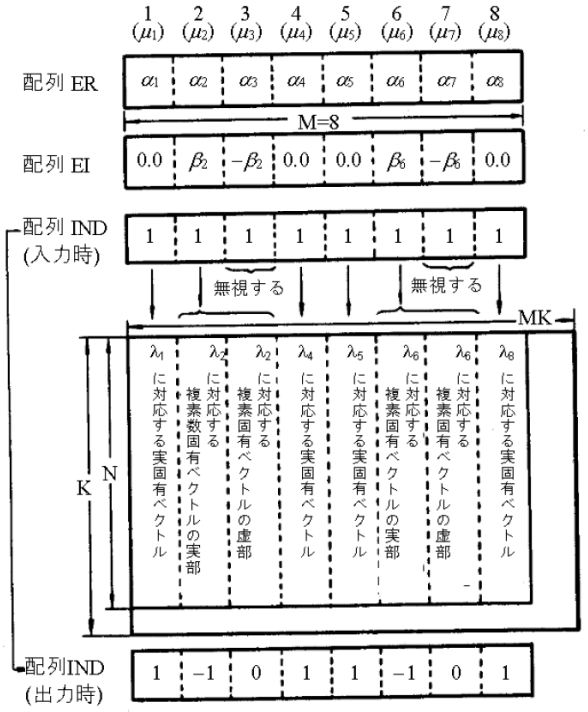


図 HVEC-2 INDの情報と固有ベクトルの格納のされ方(例 1)

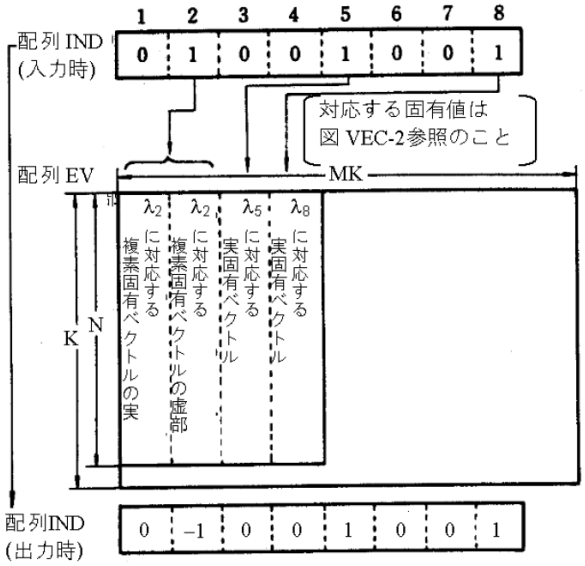


図 HVEC-3 IND の情報と固有ベクトルの格納のされ方 (例 2)

固有ベクトルは，①で述べたように格納されるので,パラメタ MK は，この点を考慮して固有ベクトルの格納のために必要な列数を指定すればよい．ただし，もし固有ベクトルの格納に必要な列数が，MK で指定した列数より大きいときは,
MK で指定した列数までは計算して格納されるが，越える部分の指定は無視される．この場合，ICON は 16000 となる．

② 本サブルーチンで用いる固有値は，サブルーチン HSQR を用いて求めることができる．
サブルーチン HSQR により固有値を求めたときは，HSQR のパラメタ ER, EI, M が，そのまま本サブルーチンのパラメタとして使用できる．

③ 本サブルーチンは実ヘッセンベルグ行列の固有ベクトルを求めるためのものである．
実行列の固有ベクトルを部分的に求める場合は,
(a) まず，サブルーチン HES1 により実ヘッセンベルグ行列に変換し,
(b) 次にサブルーチン HSQR により固有値を求め,
(c) 本サブルーチンにより固有ベクトルを求め,
(d) 最後にサブルーチン HBK1 により実行列の固有ベクトルへ逆変換すること．
なお，実行列の固有値固有ベクトルをすべて求めるときは，サブルーチン EIG1 を用いた方がよい．

④ 本サブルーチンは，④に示した手順で実行列の固有ベクトルを部分的に求める場合に用いることを前提としている．したがって，固有ベクトルの正規化は行わない．実ヘッセンベルグ行列の固有ベクトルを求めるときは，必要に応じてサブルーチン NRML により正規化すること．

⑤ サブルーチン HBK1 や NRML を使用するときは，本サブルーチンのパラメタ IND, M, EV はそのまま HBK1 や NRML の入力パラメタとして使用できる．

c. 使用例
n 次の実行列の固有値を HES1 と HSQR の各サブルーチンにより求め，固有ベクトルを本サブルーチンと HBK1 により求める．
n≤100 の場合,

```
C     **EXAMPLE**
      DIMENSION A(100,100),AW(100,104),
     *          ER(100),EI(100),IND(100),
     *          EV(100,100),PV(100)
   10 CONTINUE
      READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510)((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL HES1(A,100,N,PV,ICON)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      MP=N
      DO 35 II=1,N
      I=1+N-II
      DO 30 J=1,MP
   30 AW(J,I)=A(J,I)
   35 MP=I
      CALL HSQR(AW,100,N,ER,EI,M,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      DO 40 I=1,M
   40 IND(I)=1
      CALL HVEC(A,100,N,ER,EI,IND,M,EV,
     *     100,AW,ICON)
      WRITE(6,620) ICON
```

```
      IF(ICON.GE.20000) GO TO 10
      CALL HBK1(EV,100,N,IND,M,A,PV,0.0,
     *     ICON)
      CALL EPRT(ER,EI,EV,IND,100,N,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',5X,
     * 'N=',I3)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン ERPT は，実行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン EIG1 の使用例参照のこと．

(4) 手法概要

n 次の実 ヘッセンベルグ 行列 $\boldsymbol{A}$ の固有値 λ に対応する固有ベクトル $\boldsymbol{x}$ を逆反復法により求める．

逆反復法は，行列 $\boldsymbol{A}$ とその固有値 λ_j の近似解 μ_j が与えられたとき，

$$\left(\boldsymbol{A}-\mu_j\boldsymbol{I}\right)x_r=\boldsymbol{x}_{r-1},\quad r=1,2,3,... \tag{4.1}$$

（ここに，r は反復回数，$\boldsymbol{x}_r$ は第 r 回目の反復で得られるベクトル）

を適当な初期ベクトル $\boldsymbol{x}_0$ を与えて，反復的に解いてゆき，収束条件を満たしたときの $\boldsymbol{x}_r$ を固有ベクトルとして採用する方法である．

いま，n 次の行列 $\boldsymbol{A}$ の固有値を λ_i (i=1，2，...，n)，λ_i に対応する固有ベクトルを $\boldsymbol{u}_i$ とする．

適当な初期ベクトル $\boldsymbol{x}_0$ は，行列 $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{u}_i$ の 1 次結合，

$$\boldsymbol{x}_0=\sum_{i=1}^{n}\alpha_i\boldsymbol{u}_i \tag{4.2}$$

で表せるから，$\boldsymbol{x}_r$ は，すべての固有値 λ_i が異なるものとすれば，

$$\boldsymbol{x}_r=1/\left(\lambda_j-\mu_j\right)^r\left[\alpha_i\boldsymbol{u}_i+\sum_{\substack{i=1\\ i\neq j}}^{n}\alpha_i\boldsymbol{u}_i\left(\lambda_j-\mu_j\right)^r/\left(\lambda_j-\mu_j\right)^r\right] \tag{4.3}$$

と書ける．ここに $1/(\lambda_j-\mu_j)^r$ は定数であるから，ここでは省略して次のように書く．

$$\boldsymbol{x}_r=\alpha_i\boldsymbol{u}_i+\sum_{\substack{i=1\\ i\neq j}}^{n}\alpha_i\boldsymbol{u}_i\left(\lambda_j-\mu_j\right)^r\Big/\left(\lambda_j-\mu_j\right)^r \tag{4.4}$$

一般に $\left|\left(\lambda_j-u_j\right)/\left(\lambda_j-u_j\right)\right|<<1.0$，とみなせるから，(4.4) は，$\alpha_j\neq0$ であれば，r が大きくなる程，x_r は $\alpha_j\boldsymbol{u}_j$ に接近することを示している．

(4.1) の連立方程式は，$(\boldsymbol{A}-\mu_j\boldsymbol{I})$ を単位下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ とに分解し，

$$\boldsymbol{LUx}_r=\boldsymbol{Px}_{r-1} \tag{4.5}$$

（$\boldsymbol{P}$ は，軸交換のための置換行列である．）

として解く．(4.5) を解くことは，

$$\boldsymbol{Ly}_{r-1}=\boldsymbol{Px}_{r-1}\text{(前進代入)} \tag{4.6}$$

$$\boldsymbol{Ux}_r=\boldsymbol{y}_{r-1}\text{(後退代入)} \tag{4.7}$$

なる二つの連立方程式を解くことに帰着する．

初期ベクトル $\boldsymbol{x}_0$ はどのように与えてもよいから，

$$\boldsymbol{y}_0=\boldsymbol{L}^{-1}\boldsymbol{Px}_0 \tag{4.8}$$

なる，$\boldsymbol{y}_0$ が特定の形，例えば，$\boldsymbol{y}_0=(1,\ 1,\ 1,\ ...,\ 1)^T$ となるように $\boldsymbol{x}_0$ が与えられたとしてよい．したがって，第 1 回目には，(4.6) の前進代入は省略できる．一般には，2 回目以降は前進代入と後退代入をくり返すことにより固有ベクトルを求めることができる．

本サブルーチンでは，逆反復法を適用するにあたり，次のようにしている．

① 初期ベクトルの選定

初期ベクトル $\boldsymbol{y}_0$ としては，

$$\boldsymbol{y}_0=(\mathrm{EPS1},\mathrm{EPS1},...,\mathrm{EPS1})^T \tag{4.10}$$

を用いている．ここに EPS1 は

$$\mathrm{EPS1}=u\|\boldsymbol{A}\|_\infty \tag{4.11}$$

である．ただし u は丸め誤差の単位である．

② 収束判定の方法

後退代入の終了したとき，固有ベクトルが求まったか否かを，

$$\|\boldsymbol{x}_r\|_1\geq0.1/\sqrt{n} \tag{4.12}$$

により判定している．すなわち，(4.12) を満足すれば，$\boldsymbol{x}_r$ を固有ベクトルとして採用する．(4.12) が満足されなかったときは，初期ベクトルが適切でないと考えられるので，初期ベクトルを交換して再び後退代入を行う．

③ 固有値が重根又は近接根を持つ場合の対策

いま，固有ベクトルを求めようとする固有値 μ_j が，すでに固有ベクトルの計算を終了した固有値 μ_i との間で，

$$\left|\mu_j-\mu_i\right|\leq\mathrm{EPS1}(i=1,2,...,j-1) \tag{4.13}$$

を満足すれば，反復回数を増しても正しい固有ベクトルは得られない．

いま，$\lambda_j=\lambda_i$ であれば，(4.4) より，

$$\boldsymbol{x}_1 = \alpha_j \boldsymbol{u}_j + \alpha_i \boldsymbol{u}_i + \sum_{k=1}^{n} \alpha_k \boldsymbol{u}_k \left(\lambda_j - \mu_j\right) \Big/ \left(\lambda_k - \mu_j\right) \quad (k \neq i, j) \tag{4.14}$$

となることが分かる．このとき，$\boldsymbol{x}_1$ が $\boldsymbol{u}_j$ の定数倍となるためには，｜α_j｜≫｜α_i｜でなければならない．したがって，同一の初期ベクトルを用いる限り，μ_j と μ_i とに対応して計算された固有ベクトル $\boldsymbol{x}_1^{(j)}, \boldsymbol{x}_1^{(i)}$ は，ほぼ同じベクトルになる．
そこで本サブルーチンでは，このような場合には，μ_j を EPS1 ずつ修正して，

$$\left|\tilde{\mu}_j - \mu_i\right| > \mathrm{EPS1} \left(i = 1,2,..., j-1\right) \tag{4.15}$$

となるような $\tilde{\mu}_j$ を用いて固有ベクトルを計算している．

なお，詳細については，参考文献 [12] 及び [13] PP.418-439 を参照すること．

E51-30-0401 ICHEB, DICHEB

チェビシェフ級数の不定積分
CALL ICHEB(A,B,C,N,ICON)

(1) 機能

区間 [*a*,*b*] で定義された *n* 項のチェビシェフ級数

$$f(x)=\sum_{k=1}^{n-1}{}'c_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \qquad (1.1)$$

が与えられたとき，その不定積分をチェビシェフ級数

$$\int f(x)dx=\sum_{k=0}^{n}{}'\bar{c}_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \qquad (1.2)$$

で表現し，その係数 $\{\bar{c}_k\}$ を求める．ここで，任意定数 $\bar{c}_0$ は便宜上 0 とする．また，Σ' は初項を 1/2 倍して和をとることを意味する．

ただし， $a\neq b$, $n\geq 1$ であること．

(2) パラメタ

A …… **入力**．チェビシェフ級数の定義域の下限 *a* .

B …… **入力**．チェビシェフ級数の定義域の上限 *b* .

C …… **入力**．係数 $\{c_k\}$.
C(1)=c_0,C(2)=c_1,...,C(N)=c_{n-1} のように格納する．
出力．不定積分の係数 $\{\bar{c}_k\}$.
C(1)=0.0, C(2)= $\bar{c}_1$,...,C(N+1)= $\bar{c}_n$ のように格納される．
大きさ N+1 の 1次元配列.

N …… **入力**．項数 *n*.
出力．不定積分の項数 *n*+1.

ICON … **出力**．コンディションコード
表 ICHEB-1 参照.

表 ICHEB-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① N<1 ② A=B	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II.....MGSSL

② FORTRAN 基本関数...FLOAT

b. 注意

① 任意の関数の不定積分を求める場合，チェビシェフ級数展開するサブルーチン FCHEB と，本サブルーチンを順次呼び出せばよい.

更に，区間内の任意の点 $\nu\in[a,b]$ に対する積分値を求める場合は，チェビシェフ級数の求和のサブルーチン ECHEB を続けて呼び出せばよい．（使用例参照）.

② 積分の任意定数 $\bar{c}_0$ の決定法

本サブルーチンでは，任意定数 $\bar{c}_0$ を便宜上 0 として出力する．そこで，区間内の任意の点 $\nu\in[a,b]$ における不定積分が値 y_ν をとるように定数を定める場合，チェビシェフ級数の求和のサブルーチン ECHEB を併用して次のように計算する.

```
      …
CALL ICHEB(A,B,C,N,ICON)
CALL ECHEB(A,B,C,N,V,Y,ICON)
C(1)=(YV-Y)*2.0
      …
```

ここで，C(1) が $\bar{c}_0$, V が ν , YV が y_ν に対応する.

チェビシェフ級数に展開された関数 *f*(*x*) について，積分区間の上限を変えながら定積分

$$\int_a^{x_i} f(t)dt,\quad x_i\in[a,b],\quad i=1,2,...,m \qquad (3.1)$$

を求める場合，端点 *a* における不定積分の値が，0 となるように任意定数 $\bar{c}_0$ の値を決定し，以後 *m* 回チェビシェフ級数の求和を行えばよい.

（使用例参照）.

③ 不定積分の誤差は，末尾 2 項の係数の絶対値和で推定できる.

c. 使用例

関数

$$\tilde{f}(x)=\frac{1}{4}+\int_0^x\frac{dt}{1+100t^2},\quad x\in[0,1] \qquad (3.2)$$

の値を，*x*=0 から 0.05 刻みで求める.

まず，被積分関数を，サブルーチン FCHEB により級数展開する（要求精度：絶対誤差 $5\cdot10^{-5}$）．その後，本サブルーチンにより不定積分を求める．積分の定数は，$x=0$ で不定積分の値が 1/4 となるように定める.

各刻みごとの関数値は，サブルーチン ECHEB により求め，真値

$$f(x)=\frac{1}{4}+\frac{1}{10}\tan^{-1}10x \qquad (3.3)$$

に対する誤差も計算する.

```
C     **EXAMPLE**
      DIMENSION C(258),TAB(255)
      EXTERNAL FUN
      EPSA=5.0E-5
      EPSR=5.0E-5
      NMIN=9
      NMAX=257
      A=0.0
      B=1.0
      CALL FCHEB(A,B,FUN,EPSA,EPSR,NMIN,
     *           NMAX,C,N,ERR,TAB,ICON)
      IF(ICON.NE.0) GO TO 20
      WRITE(6,600) N,ERR,ICON
      WRITE(6,601) (C(K),K=1,N)
      CALL ICHEB(A,B,C,N,ICON)
      IF(ICON.NE.0) GO TO 20
      WRITE(6,602)
      WRITE(6,601) (C(K),K=1,N)
      CALL ECHEB(A,B,C,N,A,F,ICON)
      IF(ICON.NE.0) GO TO 20
      C(1)=(0.25-F)*2.0
      WRITE(6,603)
      H=0.05
      X=A
   10 CALL ECHEB(A,B,C,N,X,Y,ICON)
      IF(ICON.NE.0) GO TO 20
      ERROR=G(X)-Y
      WRITE(6,604) X,Y,ERROR
      X=X+H
      IF(X.LE.B) GO TO 10
      STOP
   20 WRITE(6,604) ICON
      STOP
  600 FORMAT('0',3X,'EXPANSION OF',
     * ' FUNCTION FUN(X)',3X,'N=',I4,3X,
     * 'ERROR=',E13.3,3X,'ICON=',I6)
  601 FORMAT(/(5E15.5))
  602 FORMAT('0',5X,'INTEGRATION OF',
     * ' CHEBYSHEV SERIES')
  603 FORMAT('0',10X,'X',7X,
     * 'INTEGRATION ',6X,'ERROR'/)
  604 FORMAT(1X,3E15.5)
  605 FORMAT('0',5X,'CONDITION CODE',I8)
      END
      FUNCTION FUN(X)
      FUN=1.0/(1.0+100.0*X*X)
      RETURN
      END
      FUNCTION G(X)
      G=0.25+ATAN(10.0*X)/10.0
      RETURN
      END
```

(4) 手法概要

区間 [*a*,*b*] で定義された *n* 項のチェビシェフ級数を項別積分し，再びそれをチェビシェフ級数で表現する．

すなわち，

$$\int \sum_{k=0}^{n-1}{}' c_k T_k\left(\frac{2x-(b+a)}{b-a}\right)dx = \sum_{k=0}^{n} \bar{c}_k T_k\left(\frac{2x-(b+a)}{b-a}\right) \tag{4.1}$$

とする．いま，

$$\begin{aligned}
&\int T_0(y)\,dx = \frac{b-a}{2} T_1(y) \\
&\int T_1(y)\,dx = \frac{b-a}{2} T_2(y) \\
&\int T_k(y)\,dx = \frac{b-a}{2}\left\{\frac{T_{k+1}(y)}{k+1} - \frac{T_{k-1}(y)}{k-1}\right\}, k \geq 2
\end{aligned} \tag{4.2}$$

ここで，積分定数は省略している．

また， $y = \dfrac{2x-(b+a)}{b-a}$.

なる関係を (4.1) の左辺に代入し，$T_k(y)$ に関して整理すると，係数 $\{c_k\}$ と $\{\bar{c}_k\}$ の間には次の関係が成り立つ．

$$\begin{aligned}
&\bar{c}_{n+1} = 0 \\
&\bar{c}_n = \frac{b-a}{4n} c_{n-1} \\
&\bar{c}_k = \frac{b-a}{4k}(c_{k-1} + c_{k+1}), \\
&k = n-1, n-2, \ldots\ldots, 1
\end{aligned} \tag{4.3}$$

本サブルーチンでは，チェビシェフ多項式の積分公式 (4.3) により，$\{\bar{c}_k\}$ を求めている．なお，任意定数 $\bar{c}_0$ は便宜上 0 としている．

n 項の級数の不定積分を求めるに，必要な乗算，除算回数は，それぞれ，約 *n* 回である．

I11-71-0301 IERF, DIERF

逆誤差関数 $\mathrm{erf}^{-1}(x)$
CALL IERF (X,F,ICON)

(1) 機能

誤差関数 $\mathrm{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ の逆関数 $\mathrm{erf}^{-1}(x)$ の値を，多項式及び有理関数の最良近似式を用いて計算する．だたし $|x|<1$ であること．

(2) パラメタ

X……**入力**．独立変数 *x*.
F……**出力**．関数値 $\mathrm{erf}^{-1}(x)$.
ICON……**出力**．コンディションコード.
表 IERF-1 参照.

表 IERF-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	|x|≥1 であった.	F = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II...IERFC,MGSSL
② FORTRAN 基本関数...ABS,SQRT,ALOG

b. 注意
① 引数 X の範囲は $|x|<1$ であること.
この範囲がこの関数の定義領域である.
② $\mathrm{erf}^{-1}(x)=\mathrm{erfc}^{-1}(1-x)$ という関係を利用すれば，逆誤差関数の値をサブルーチン IERFC を用いて計算することもできるが，$|x|\le 0.8$ の範囲の *x* については本サブルーチン IERF を用いる方が，値が正確で計算時間も短い.

c. 使用例
$\mathrm{erf}^{-1}(x)$ の値を，0 から 0.99 までの 0.01 刻みの *x* の値に対して計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,100
      X=FLOAT(K-1)/100.0
      CALL IERF(X,F,ICON)
      IF(ICON.EQ.0)WRITE(6,610)X,F
      IF(ICON.NE.0)WRITE(6,620)X,F,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF INVERSE ',
     * 'ERROR FUNCTION'///6X,'X',7X,
     * 'IERF(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'IERF=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

逆誤差関数 $\mathrm{erf}^{-1}(x)$ の計算には，$|x|\le 0.8$ であるか，そうでないかに応じて異なる近似式を用いる.

a. $|x|\le 0.8$ の場合
$\mathrm{erf}^{-1}(x)$ の $x=0$ を中心とするテーラー級数展開（参考文献 [87] 参照）に基づく相対誤差の意味での最良有理近似式を用いる.

単精度：$\mathrm{erf}^{-1}(x)=x\cdot\sum_{k=0}^{3}a_k t^k \Big/ \sum_{k=0}^{4}b_k t^k$ (4.1)
$t=\left(1-(x/0.8)^2\right)$
理論精度 9.2 桁

倍精度：$\mathrm{erf}^{-1}(x)=x\cdot\sum_{k=0}^{7}a_k t^k \Big/ \sum_{k=0}^{8}b_k t^k$ (4.2)
$t=\left(1-(x/0.8)^2\right)$
理論精度 18.8 桁

b. $|x|>0.8$ の場合
$\mathrm{erf}^{-1}(x)=\mathrm{erfc}^{-1}(1-x)$ の関係を利用して，サブルーチン IERFC を引用することにより計算する．詳細については，IERFC の項を参照のこと.

I11-71-0401 IERFC,DIERFC

逆余誤差関数 $\mathrm{erfc}^{-1}(x)$
CALL IERFC(X,F,ICON)

(1) 機能

余誤差関数 $\mathrm{erfc}(x)=\frac{2}{\sqrt{\pi}}\int_x^\infty e^{-t^2}dt$ の逆関数

$\mathrm{erfc}^{-1}(x)$ の値を，多項式及び有理関数の最良近似式を用いて計算する．ただし，$0<x<2$ であること．

(2) パラメタ

X…… **入力**．独立変数 x.
F …… **出力**．関数値 $\mathrm{erfc}^{-1}(x)$.
ICON… **出力**．コンディションコード.
表 IERFC-1 参照.

表 IERFC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	X≤0 又は X≥2 であった.	F = 0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II...IERF,MGSSL
② FORTRAN 基本関数...ABS, SQRT, ALOG

b. 注意
① 引数 X の範囲は 0<X<2 であること.
この範囲がこの関数の定義領域である.
② $\mathrm{erfc}^{-1}(x)=\mathrm{erf}^{-1}(1-x)$ という関係を利用すれば，逆余誤差関数の値をサブルーチン IERF を用いて計算することもできるが，$0<x<0.2$ の範囲の x については本サブルーチン IERFC を用いる方が，値が正確で計算時間も短い.

c. 使用例
$\mathrm{erfc}^{-1}(x)$ の値を，0.01 から 1.00 までの 0.01 刻みの x の値に対して計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,100
      X=FLOAT(K)/100.0
      CALL IERFC(X,F,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,F
      IF(ICON.NE.0) WRITE(6,620) X,F,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF INVERSE',
     * ' COMPLEMENTARY ERROR FUNCTION',
     * ///6X,'X',6X,'IERFC(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'IERFC=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

逆余誤差関数 $\mathrm{erfc}^{-1}(x)$ の計算には，$0<x<0.2$ 又は，$1.8<x<2$ である場合とそうでない場合に応じて異なる近似式を用いる.

a. $0<x<0.2$ 又は $1.8<x<2$ の場合
$\beta=\sqrt{-\log(x(2-x))}$ に対して $\mathrm{erfc}^{-1}(x)=\beta\cdot R(\beta)$ とおき，補助関数 R(β) を，Strecok （参考文献 [87] 参照）の理論に基づく相対誤差の意味での最良近似式を用いて計算する.

(a) $0<x<5.10^{-16}$ 又は $(2\text{-}5\cdot10^{-16})<x<2$の場合
$t=c/\sqrt{\beta}+d$ として,

単精度： $\mathrm{erfc}^{-1}(x)=\beta\cdot\sum_{k=0}^{7}a_k t^k$ (4.1)

理論精度 9.7 桁
ただし c=-4.5999961
d=1.8974954

倍精度： $\mathrm{erfc}^{-1}(x)=\beta\cdot\sum_{k=0}^{16}a_k t^k$ (4.2)

理論精度 18.9 桁
ただし c = -4.59999617941507552
d = 1.89749541006229975

(b) $5.10^{-16}\le x<2.5\cdot10^{-3}$ 又は
$2\text{-}2.5\cdot10^{-3}<x\le2\text{-}5\cdot10^{-16}$ の場合
$t=c\cdot\beta+d$ として,

単精度： $\mathrm{erfc}^{-1}(x)=\beta\cdot\sum_{k=0}^{4}a_k t^k/\sum_{k=0}^{4}b_k t^k$ (4.3)

理論精度 8.4 桁
ただし c = 0.27972881
d = -0.64395786

倍精度： $\mathrm{erfc}^{-1}(x)=\beta\cdot\sum_{k=0}^{11}a_k t^k/\sum_{k=0}^{11}b_k t^k$ (4.4)

理論精度 18.7 桁
ただしc = 0.279728815664916161
d= -0.643957858131678820

(c) $2.5\ 10^{-3}\le x<0.2$ 又は $1.8<x\le1.9975$ の場合
$t=c\cdot B+d$ として

単精度： $\mathrm{erfc}^{-1}(x)=\beta\cdot\sum_{k=0}^{7}a_k t^k$ (4.5)

理論精度 8.9 桁
ただし c = -0.77440652
d = 1.7827451

倍精度： $\mathrm{erfc}^{-1}(x)=\beta\cdot\sum_{k=0}^{20}a_k t^k$ (4.6)

理論精度 18.5 桁
ただし c=-0.774406521186630830
d=1.78274506157390808

b. $0.2 \le x \le 1.8$ の場合
$\mathrm{erfc}^{-1}(x) = \mathrm{erf}^{-1}(1-x)$ の関係を利用して，サブルーチン IERF を引用することにより計算する．詳細については，IERF の項を参照のこと．

I11-61-0101 IGAM1, DIGAM1

第1種不完全ガンマ関数 $\gamma(v,x)$
CALL IGAM1(V,X,F,ICON)

(1) 機能

第1種不完全ガンマ関数 $\gamma(v,x)$ の値

$$\gamma(v,x)=\int_0^x e^{-t}t^{v-1}dt$$

を級数展開，漸近展開及び数値積分により求める．ただし，$v>0$, $x\geq 0$ であること．

(2) パラメタ

V…… **入力**．独立変数 *v*.
X…… **入力**．独立変数 *x*.
F …… **出力**．関数値 $\gamma(v,x)$.
ICON … **出力**．コンディションコード.
表 IGAM1-1 参照.

表 IGAM1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	V≤0 又は X<0 であった.	F=0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... AFMAX, AMACH, EXPI, IGAM2, MGSSL, ULMAX
② FORTRAN 基本関数...FLOAT, EXP, GAMMA

b. 注意
① 引数 X の範囲が，X≥23.0 （単精度），X≥46.0 （倍精度）であれば，$\gamma(v,x)\approx\Gamma(v)$ となるので，FORTRAN 基本関数の完全ガンマ関数 GAMMA(*v*) を利用した方がよい.

c. 使用例
$\gamma(v,x)$ の値を $v,x=a+b$, a=0,1,2, b=0.1,0.2,0.3 に対して求める.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 20 IV=1,9
      V=FLOAT(IV+7*((IV-1)/3))/10.0
      DO 10 IX=1,9
      X=FLOAT(IX+7*((IX-1)/3))/10.0
      CALL IGAM1(V,X,F,ICON)
      IF(ICON.EQ.0) WRITE(6,610) V,X,F
      IF(ICON.NE.0) WRITE(6,620) V,X,F,
     *                           ICON
   10 CONTINUE
   20 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF',
     * ' INCOMPLETE GAMMA FUNCTION'
     * //5X,'V',9X,'X',10X,'FUNC'/)
  610 FORMAT(' ',2F10.5,E17.7)
  620 FORMAT(/' ','** ERROR **',5X,'V=',
     * F10.5,5X,'X=',F10.5,5X,'IGAM1=',
     * E17.7,5X,'CONDITION=',I10/)
      END
```

(4) 手法概要

計算式は x_1=3.5（倍精度では，x_1=5.5）を境にして異なる.

a. $x\leq 2(v+1)$ 又は $x<x_1$ の場合，次の級数展開による.

$$\gamma(v,x)=x^v e^{-x}\left\{1+\frac{x}{v+1}+\frac{x^2}{(v+1)(v+2)}+\ldots\right\}/v \quad (4.1)$$

級数は，初項からの累和に対して，その項が影響しなくなるまで計算する.

b. $x>2$(v+1) かつ $x\geq x_1$ の場合，

$$\gamma(v,x)=\Gamma(v)-\Gamma(v,x) \quad (4.2)$$

を用い，$\Gamma(v)$ は FORTRAN 基本関数 GAMMA, $\Gamma(v,x)$ はサブルーチン IGAM2 を用いて計算する.

なお，詳細については，参考文献 [85] pp.14-16 を参照のこと.

I11-61-0201 IGAM2, DIGAM2

第2種不完全ガンマ関数 $\Gamma(v,x)$
CALL IGAM2(V,X,F,ICON)

(1) 機能

第2種不完全ガンマ関数 $\Gamma(v,x)$ の値

$$\Gamma(v,x)=\int_x^\infty e^{-t}t^{v-1}dt=e^{-x}\int_0^\infty e^{-t}(x+t)^{v-1}dt$$

を級数展開，漸近展開及び数値積分により求める．ただし，$v\geq 0$, $x\geq 0$ ($v=0$ のとき $x\neq 0$) であること．

(2) パラメタ

V…… **入力**．独立変数v.
X…… **入力**．独立変数 x.
F…… **出力**．関数値 $\Gamma(v.x)$.
ICON … **出力**．コンディションコード．
表 IGAM2-1 参照．

表 IGAM2-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	$x^{v-1}e^{-x}>fl_{max}$ であった．	$F=fl_{max}$ とする．
30000	V<0 または X<0, V=0 かつ x=0 であった．	F=0.0 とする．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... AFMAX, AMACH, EXPI, MGSSL, ULMAX
② FORTRAN 基本関数...FLOAT, EXP, GAMMA, ATAN

b. 注意
① 引数 X の範囲が $X\geq\log(fl_{max})$ となると $\Gamma(v,x)$ はアンダフローするほど小さくなる．
② $x^{v-1}e^{-x} > fl_{max}$ は $x>1$ で，v が非常に大きいときで，そのとき $\Gamma(v,x)$ はオーバフローする．

c. 使用例
$\Gamma(v,x)$ の値をv,x=a+b, a=0,1,2 ,b=0.1,0.2,0.3 に対して求める．

```
C      **EXAMPLE**
       WRITE(6,600)
       DO 20 IV=1,9
       V=FLOAT(IV+7*((IV-1)/3))/10.0
       DO 10 IX=1,9
       X=FLOAT(IX+7*((IX-1)/3))/10.0
       CALL IGAM2(V,X,F,ICON)
       IF(ICON.EQ.0) WRITE(6,610) V,X,F
       IF(ICON.NE.0) WRITE(6,620) V,X,F,
      *                           ICON
    10 CONTINUE
    20 CONTINUE
       STOP
   600 FORMAT('1','EXAMPLE OF',
      * ' INCOMPLETE GAMMA FUNCTION'
      * //5X,'V',9X,'X',10X,'FUNC'/)
   610 FORMAT(' ',2F10.5,E17.7)
   620 FORMAT(/' ','** ERROR **',5X,'V=',
      * F10.5,5X,'X=',F10.5,5X,'IGAM2=',
      * E17.7,5X,'CONDITION=',I10/)
       END
```

(4) 手法概要

v=0, x>0 のとき$\Gamma(v,x)$ は本質的に指数積分となるので，サブルーチン EXPI を利用して計算する．また，v>0, x=0 のときは $\Gamma(v,x)$ は v についての完全ガンマ関数となるので FORTRAN 基本関数 GAMMA より計算する．

以下では，v>0, x>0 の場合の計算法を述べる．

計算式は x_1=20.0（倍精度はx_1= 40.0）を境にして異なる．

a. v=整数又は，$x>x_1$ の場合，次の漸近展開による．

$$\Gamma(v,x)=x^{v-1}e^{-x}\left\{1+\frac{v-1}{x}+\frac{(v-1)(v-2)}{x^2}+\cdots+\frac{(v-1)\cdots(v-n)}{x^n}+\cdots\right\} \tag{4.1}$$

級数は初項からの累和に対して，その項が影響しなくなるまで計算するか，$|(v-n)/x|$ が1を超える前，すなわち $n\leq v+x$ を満す最大の整数まで計算する．

b. $v\neq$整数かつ $x\leq x_1$ の場合，
次の様に変形する．

$$\Gamma(v,x)=x^{v-1}e^{-x}\left\{1+\frac{v-1}{x}+\cdots+\frac{(v-1)(v-2)\cdots(v-m+1)}{x^{m-1}}\right\}+(v-1)(v-2)\cdots(v-m)\Gamma(v-m,x) \tag{4.2}$$

ただし，m は $m\leq v$ となる最大の整数である．

v<1 なら，第1項は0とし，$(v\text{-}1)(v\text{-}2)\cdots\cdots(v\text{-}m)=1$ とする．

第2項の $\Gamma(v\text{-}m,x)$ は，$0<v\text{-}m<1$ であり，次のように表せる．

$$\Gamma(v-m,x)=e^{-x}\int_0^\infty e^{-t}(x+t)^{v-m-1}dt \tag{4.3}$$

この積分は，$(x+t)^{v-m-1}$ が特異性を有するので，その場合有力な二重指数関数型数値積分公式（参考文献 [68]pp.21-27参照）を用いて計算する．

なお，詳細については，参考文献 [85]pp.14-16を参照すること．

I11-91-0301 INDF, DINDF

逆正規分布関数 $\phi^{-1}(x)$
CALL INDF(X,F,ICON)

(1) 機能

正規分布関数 $\phi(x)=\frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}}dt$ の逆関数 $\phi^{-1}(x)$ の値を，関係式

$$\phi^{-1}(x)=\sqrt{2}\,\mathrm{erf}^{-1}(2x) \qquad (1.1)$$

により，計算する．ただし $|x|<1/2$ であること．

(2) パラメタ

X…… **入力**．独立変数 x.
F …… **出力**．関数値 $\phi^{-1}(x)$.
ICON… **出力**．コンディションコード.
表 INDF-1 参照.

表 INDF-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	$\|x\|\geq 1/2$ であった.	F=0.0 とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... IERF, IERFC, MGSSL
② FORTRAN 基本関数...ABS,SQRT,ALOG

b. 注意
① 引数 X の範囲は定義領域，$|x|<1/2$ であること.
② 逆余正規分布関数 $\psi^{-1}(x)$ との間の関係

$$\phi^{-1}(x)=\psi^{-1}(1/2-x) \qquad (3.1)$$

を利用すれば，$\phi^{-1}(x)$ の値をサブルーチン INDFC を用いて計算することもできる．ただし，このように計算すると $|x|\leq 0.4$ の範囲の x については本サブルーチン INDF を直接用いるのに比べ精度が悪く，計算時間も長くなるので注意すること.

c. 使用例
$\phi^{-1}(x)$ の値を，0.0 から 0.49 までの 0.01 刻みの x の値に対して計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,50
      X=FLOAT(K-1)/100.0
      CALL INDF(X,F,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,F
      IF(ICON.NE.0) WRITE(6,620) X,F,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF INVERSE ',
     * 'NORMAL DISTRIBUTION FUNCTION'
     * //6X,'X',7X,'INDF(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'INDF=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

正規分布関数 $\phi(t)$ と誤差関数 erf(t) との間には

$$\phi(t)=\mathrm{erf}\left(t/\sqrt{2}\right)/2,\quad -\infty<t<\infty \qquad (4.1)$$

なる関係がある．この式の両辺を $x(|x|<1/2)$ と置くと $\phi(t)=x$ より $t=\phi^{-1}(x)$, および $\mathrm{erf}\left(t/\sqrt{2}\right)/2=x$ より $t=\sqrt{2}\,\mathrm{erf}^{-1}(2x)$ となる．ゆえに

$$\phi^{-1}(x)=\sqrt{2}\,\mathrm{erf}^{-1}(2x) \qquad (4.2)$$

が導かれる．本サブルーチンではサブルーチン IERF を使用して (4.2) より $\phi^{-1}(x)$ の計算を行う.

I11-91-0401 INDFC, DINDFC

逆余正規分布関数 $\psi^{-1}(x)$
CALL INDFC(X,F,ICON)

(1) 機能

余正規分布関数 $\psi(x)=\frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{t^2}{2}}\,dt$ の逆関数 $\psi^{-1}(x)$ の値を，関係式

$$\psi^{-1}(x)=\sqrt{2}\,\mathrm{erfc}^{-1}(2x) \qquad (1.1)$$

により計算する．ただし，$0<x<1$であること．

(2) パラメタ

X…… **入力**．独立変数 x.
F…… **出力**．関数値 $\psi^{-1}(x)$.
ICON … **出力**．コンディションコード．
表 INDFC-1 参照

表 INDFC-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	X≤0 又は X≥1 であった．	F=0.0 とする．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II...IERFC,IERFC,MGSSL
② FORTRAN 基本関数...ABS, SQRT, ALOG

b. 注意
① 引数 X の範囲は定義領域 0< X <1 であること．
② 逆正規分布関数 $\phi^{-1}(x)$ との間の関係

$$\psi^{-1}(x)=\phi^{-1}(1/2-x) \qquad (3.1)$$

を利用すれば，$\psi^{-1}(x)$ の値をサブルーチン INDF を用いて計算することもできる．ただし，このように計算すると $0<x<0.1$ の範囲の x については本サブルーチン INDFC を直接用いるのに比べ精度が悪くなるので注意すること．

c. 使用例
$\psi^{-1}(x)$ の値を，0.01 から 0.50 までの 0.01刻みの x の値に対して計算し数表を作る

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,50
      X=FLOAT(K)/100.0
      CALL INDFC(X,F,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,F
      IF(ICON.NE.0) WRITE(6,620) X,F,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF INVERSE ',
     * 'COMPLEMENTARY NORMAL ',
     * 'DISTRIBUTION FUNCTION'//
     * 6X,'X',7X,'INDFC(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'INDFC=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

余正規分布関数 $\psi(t)$ と 余誤差関数 $\mathrm{erfc}(t)$ との間には

$$\psi(t)=\mathrm{erfc}\left(t/\sqrt{2}\right)/2, \quad -\infty<t<\infty \qquad (4.1)$$

なる関係がある．この式の両辺を $x(0<x<1)$ と置くと $\psi(t)=x$ より $t=\psi^{-1}(x)$, および $\mathrm{erfc}\left(t/\sqrt{2}\right)/2=x$ より $t=\sqrt{2}\,\mathrm{erfc}^{-1}(2x)$ となる．ゆえに

$$\psi^{-1}(x)=\sqrt{2}\,\mathrm{erfc}^{-1}(2x) \qquad (4.2)$$

が導かれる．本サブルーチンでは，サブルーチン IERFC を使用して (4.2) より $\psi^{-1}(x)$ の計算を行う．

E12-21-0101　　INSPL, DINSPL

3次 spline 補間式
CALL INSPL(X,Y,DY,N,C,D,E,ICON)

(1)　機能

離散点$x_1, x_2, ..., x_n$ $(x_1<x_2<...<x_n)$ に対して，関数値 $y_i=f(x_i)$, i=1,...,n と，両端における2階微係数y_1" , y_n"が与えられたとき，3次 spline 関数を用いて， $f(x)$ に対する (1.1) で表される補間式 S(x)（以後3次 spline 補間式という）を求める．ただし n≥2 であること．

$$S(x)=\begin{cases} x<x_1 \text{のとき} \\ \quad y_1+c_1(x-x_1)+d_1(x-x_1)^2 \\ x_i\le x\le x_{i+1}, i=1,...n-1 \text{のとき} \\ \quad y_i+c_i(x-x_i)+d_i(x-x_i)^2 \\ \quad +e_i(x-x_i)^3 \\ x>x_n \text{ のとき} \\ \quad y_n+c_n(x-x_n)+d_n(x-x_n)^2 \end{cases} \qquad (1.1)$$

(2)　パラメタ

X…… **入力**．離散点 x_i.
　　大きさ n の1次元配列.

Y…… **入力**．関数値 y_i.
　　大きさ n の1次元配列.

DY…… **入力**．両端における2階微係数 y_1" , y_n".
　　大きさ2 の 1次元配列.
　　DY(1) = y_1" DY(2) = y_n"と格納する.
　　(使用上の注意参照)

N…… **入力**．離散点の個数 n .
　　n≥2.

C…… **出力**．(1.1) 式における係数 c_i.
　　大きさ n の1次元配列.

D…… **出力**．(1.1) 式における係数 d_i.
　　大きさ n の1次元配列.

E …… **出力**．(1.1) 式における係数 e_i .
　　常に E(N)=0.0 となる.
　　大きさ n の1次元配列.

ICON… **出力**．コンディションコード.
　　表INSPL-1 参照.

(3)　使用上の注意

a.　使用する副プログラム

①　SSL II ... MGSSL

②　FORTRAN 基本関数 ... なし.

表 INSPL-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	n<2 ,又は $x_i\ge x_{i+1}$ であった.	処理を打ち切る.

b.　注意

①　両端の2階微係数 y_1" , y_n" は任意に指定することができるが，それらの値があらかじめ分からない場合は，y_1"=0.0, y_n"=0.0 とするのが良い．このとき，積分 $\int_{x_1}^{x_n}\left[S''(x)\right]^2 dx$ が最小となる．ここで S"(x) は求めようとする3次 spline 補間式の2階導関数である.

②　補間しようとする関数 $f(x)$ が， x_n-x_1 の 周期を持つと考えるならば，両端の2階微係数は y_1"=y_n" となる値を与えるとよい.

c.　使用例

離散点の個数 n, 離散点 x_i, 関数値 y_i, i=1,...,n を入力し，3次 spline 補間式を求め，それによりある区間 [x_i,x_{i+1}] 内のある点 x=ν における補間値を求める．y_1"=0.0, y_n"=0.0, n≤10の場合.

```
C      **EXAMPLE**
       DIMENSION X(10),Y(10),DY(2),C(10),
      *          D(10),E(10)
       READ(5,500) N
       READ(5,510) (X(I),Y(I),I=1,N)
       DY(1)=0.0
       DY(2)=0.0
       CALL INSPL(X,Y,DY,N,C,D,E,ICON)
       WRITE(6,600) ICON
       IF(ICON.GT.10000) STOP
       READ(5,520) I,V
       XX=V-X(I)
       YY=Y(I)+(C(I)+(D(I)+E(I)*XX)*XX)*XX
       WRITE(6,610) YY
       STOP
  500 FORMAT(I2)
  510 FORMAT(2E16.8)
  520 FORMAT(I2,E16.8)
  600 FORMAT('1',10X,'ICON=',I5)
  610 FORMAT('0',10X,'INTERPOLATED ',
     * 'VALUE=',E16.8)
       END
```

(4)　手法概要

離散点 $x_1,x_2...,x_n$ $(x_1<x_2<...<x_n)$ に対して，関数値 $y_i=f(x_i)$, i=1,...,n が与えられたとき，(1.1) で表せる補間式を求めることを考える．ところで (1.1) は，区間 [x_i,x_{i+1}] ごとに異なった高々3次 ((-∞,x_1) , (x_n, ∞) では高々2次) の多項式を表している.

説明の都合上，求めようとする補間式を S(x) で表し，さらにそれを区分的に次の (4.1) で表す.

$$S(x)=\begin{cases} S_0(x), x<x_1 \\ S_i(x), x_i\le x\le x_{i+1}, i=1,...,n-1 \\ S_n(x), x>x_n \end{cases} \qquad (4.1)$$

ただし $S(x_i)=y_i$, i=1,…,n

ここで $S_0(x)$, $S_n(x)$ は高々2次の多項式であり，$S_1(x),...,S_{n-1}(x)$ は高々3次の多項式である．

本ルーチンでは，$S(x)$ を，その2階までの導関数が，$(-\infty,\infty)$ で連続になるよう決定する．そのためには $S_i(x)$ が (4.2) を満たせばよい．

$$\left.\begin{aligned} &S_{i-1}(x_i)=S_i(x_i)=y_i \\ &S'_{i-1}(x_i)=S'_i(x_i) \\ &S''_{i-1}(x_i)=S''_i(x_i) \\ &i=1,...,n \end{aligned}\right\} \tag{4.2}$$

(1.1) における係数 c_i, d_i, e_i は (4.2) の条件により決定される．それを (4.3) に示す．

$i=1,...,n-1$のとき

$$\left.\begin{aligned} c_i &= \frac{y_{i+1}-y_i}{h_i}-\frac{h_i}{6}\left(y''_{i+1}+2y''_i\right) \\ d_i &= \frac{y''_i}{2} \\ e_i &= \frac{y''_{i+1}-y''_i}{6h_i} \end{aligned}\right\} \tag{4.3}$$

$i=n$のとき

$$\left.\begin{aligned} c_n &= \frac{y_n-y_{n-1}}{h_{n-1}}+\frac{h_{n-1}}{6}\left(2y''_n+y''_{n-1}\right) \\ d_n &= \frac{y''_n}{2} \end{aligned}\right\}$$

ここで $h_i=x_{i+1}-x_i$, $y_i''=S''(x_i)$ である．y_i''は (4.2) の2番目の条件より，(4.4) の3項関係を満たすものである．

$$\begin{aligned} &h_{i-1}y''_{i-1}+2(h_{i-1}+h_i)y''_i+h_iy''_{i+1} \\ &\quad=6\left(\frac{y_{i+1}-y_i}{h_i}-\frac{y_i-y_{i-1}}{h_{i-1}}\right) \\ &\quad,i=2,...,n-1 \end{aligned} \tag{4.4}$$

(4.4) は y_1'', y_n'' が与えられれば，y_2'', y_3'', ..., y''_{n-1} に関する連立方程式となり，その係数行列は正値対称行列であるので，本ルーチンでは，変形コレスキー法を使って解く．

$y_1''=y_n''=0$ なるときの $S(x)$ の意味

$y_1''=y_n''=0$ と与えられたとき，$S(x)$ は $(-\infty,x_1)$, (x_n, ∞) において1次式になるが，このときは，特に次の性質をもつ3次自然 spline (cubic natural spline) 補間式となる．

$g(x)$ を (4.5) を満たす任意の補間式とする．

$$\left.\begin{aligned} &g(x_i)=y_i, i=1,...,n \\ &g(x),\ g'(x),\ g''(x) \text{ は } [x_1, x_n] \text{ で連続である．} \end{aligned}\right\} \tag{4.5}$$

そして $S(x)$ を，$y''_1=y_n''=0$ としたときの3次 spline補間式とすると，

$$\int_{x_1}^{x_n}\left\{g''(x)\right\}^2dx \geq \int_{x_1}^{x}\left\{S''(x)\right\}^2dx \tag{4.6}$$

が成り立つ（ここで等号は $g(x)=S(x)$ のときに限り成り立つ)．故に，$S(x)$ は

$$\int_{x_1}^{x_n}\left\{g''(x)\right\}^2dx$$

を最小にするという意味において，“最も滑らかな”補間式である．

なお，詳細については，参考文献 [48], [49], 及び [50] pp.349 – 356 を参照すること．

F20-01-0101　　LAPS1, DLAPS1

ラプラス変換（複素右半平面で正則な有理関数）
CALL LAPS1(A,NA,B,NB,T,DELT,L, EPSR,FT, T1,NEPS,ERRV,ICON)

(1)　機能

(1.1) で表される有理関数 $F(s)$ が与えられたとき，そのラプラス逆変換 $f(t)$ の値，$f(t_0)$，$f(t_0+\Delta t)$，…，$f(t_0+\Delta t(L-1))$ を求める．

$$F(s)=\frac{Q(s)}{P(s)}$$

$$Q(s) = b_1 s^m + b_2 s^{m-1} + \cdots + b_m s + b_{m+1}$$

$$P(s) = a_1 s^n + a_2 s^{n-1} + \cdots + a_n s + a_{n+1}$$

$n \geqq m$, $a_1, a_2, \cdots, a_{n+1}, b_1, b_2, \cdots,$

b_{m+1}: 実数　　(1.1)

ただし，$F(s)$ は Re(s) > 0 で正則であるとする．

(2)　パラメタ

A ……**入力**．$P(s)$ の係数．大きさ $n+1$ 次元配列．A(1)=a_1，A(2)=a_2，...，A(n+1)=a_{n+1} の順に与える．

NA……**入力**．$P(s)$ の次数 n.

B……**入力**．$Q(s)$ の係数．
大きさ $m+1$ の 1 次元配列．B(1)=b_1，B(2)=b_2,⋯,B(m+1)=b_{m+1} の順に与える．

NB……**入力**．$Q(s)$ の次数 m.

T……**入力**．$f(t)$ の計算開始値 t_0 (≥0).

DELT ……**入力**．変数 t の増分 Δt (> 0).
DELT = 0.0 の場合は，$f(t_0)$ だけが計算される．

L……**入力**．$f(t)$ の値を求めようとする点の個数．ただし L≥1.

EPSR……**入力**．$f(t)$ の値に対する相対誤差の上限 (≥0.0)．単精度では 10^{-2} ～ 10^{-4}，倍精度では 10^{-2} ～ 10^{-7} 程度がよい．EPSR = 0.0 と入力したときは標準値 10^{-4} が採用される．

FT……**出力**．$f(t_0)$，$f(t_0+\Delta t)$，…，$f(t_0+\Delta t$(L-1)) の値．大きさ L の 1 次元配列．

T1……**出力**．t_0，$t_0+\Delta t$，…，$t_0+\Delta t$(L-1) の列．大きさ L の 1 次元配列．

NEPS……**出力**．打切り項数 N（手法概要 参照）．大きさ L の 1 次元配列．FT(I) を計算するために用いた項数 N が NEPS(I) に格納される．

ERRV……**出力**．各結果 FT の相対誤差．
大きさ L の 1 次元配列．FT(I) における相対誤差が ERRV(I) に格納される．

ICON……**出力**．コンディションコード．
表 LAPS1-1 参照．

表 LAPS1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	結果 $f(t)$ の中に要求精度を満足しないものがある.	処理を続ける. $f(t_0+\Delta t\cdot l)$: l=0,1,…,L-1の精度は配列 ERRV に出力される.
30000	次のいずれかであった. ① NB<0，又は NB>NA ② T<0，又は DELT<0 ③ L<1 ④ EPSR<0 ⑤ A(1) = 0	処理を打ち切る.

(3)　使用上の注意

a.　使用する副プログラム

① SSL II…　MGSSL, AFMAX

② FORTRAN 基本関数 …　FLOAT, ALOG, ALOG10, CMPLX, AIMAG, EXP, ABS, INT, ATAN

b.　注意

① 有理関数 F(s) は Re(s)>0 で正則であること．Re(s)>0 で非正則であったり，あるいは正則性について不明である場合は，本サブルーチンの代りに LAPS2 を使用すること．

② t_0=0 と与えたとき $f(0)$ の値は，次の初期値定理より計算する．

$$f(0)=\begin{cases} b_1/a_1 & (n=m+1) \\ 0 & (n>m+1) \end{cases}$$

③ NA=NB の場合，(1.1) は

$$F(s)=\frac{Q(s)}{P(s)}=F_1(s)+F_2(s) \qquad (3.1)$$

ただし，

$$F_1(s)\equiv b_1/a_1$$

$$F_2(s)\equiv\frac{c_2 s^{n-1}+c_3 s^{n-2}+\ldots+c_{n+1}}{a_1 s^n+a_2 s^{n-1}+\ldots\ldots+a_{n+1}}$$

と分解できる．

$F_1(s)$ の逆変換 $f_1(t)$ は

$$f_1(t)=\frac{b_1}{a_1}\delta(t) \text{ は } \delta(t)\text{: デルタ関数}$$

となる．$F_2(s)$ は 8 章の式 (8.20) の条件を満足しているので $F_2(s)$ の逆変換 $f_2(t)$ は式 (8.22) から計算できる．

NA = NB のとき，本サブルーチンを用いて逆変換を計算すると t > 0 で $F_2(s)$ の逆変換の値が出力される．t = 0 では浮動小数点の最大値を出力する．

c. 使用例

Re(s)>0 で正則な有理関数,

$$F(s)=\frac{s^2+4}{s^4+12s^3+54s^2+108s+81}$$

のラプラス逆変換 $f(t)$ をt_i = 0.2 + 0.2(i-1), i=1, 2, …, 9 の各点で求める．EPSR=10^{-4} とする．

```
C      **EXAMPLE**
       DIMENSION A(5),B(3),T1(9),FT(9),
      *ERRV(9),NEPS(9)
       NB=2
       NB1=NB+1
       B(1)=1.0
       B(2)=0.0
       B(3)=4.0
       NA=4
       NA1=NA+1
       A(1)=1.0
       A(2)=12.0
       A(3)=54.0
       A(4)=108.0
       A(5)=81.0
       T=0.2
       DELT=0.2
       L=9
       EPSR=1.0E-4
       WRITE(6,600) NB,(I,B(I),I=1,NB1)
       WRITE(6,610) NA,(I,A(I),I=1,NA1)
       WRITE(6,620) EPSR
       CALL LAPS1(A,NA,B,NB,T,DELT,L,EPSR,
      *           FT,T1,NEPS,ERRV,ICON)
       WRITE(6,630) ICON
       IF(ICON.EQ.30000) STOP
       WRITE(6,640) (T1(I),FT(I),ERRV(I),
      *              NEPS(I),I=1,L)
       STOP
  600 FORMAT(//24X,'NB=',I3//
     * (20X,'B(',I3,')=',E15.8/))
  610 FORMAT(/24X,'NA=',I3//
     * (20X,'A(',I3,')=',E15.8/))
  620 FORMAT(22X,'EPSR=',E15.8/)
  630 FORMAT(22X,'ICON=',I6//)
  640 FORMAT(15X,'F(',F8.5,')=',E15.8,2X,
     * 'ERROR=',E15.8,2X,'N=',I3/)
       END
```

(4) 手法概要

ラプラス逆変換の計算法については，8 章の「ラプラス変換」のところで述べた．すなわち，f (t) は，N 項の σ_0 近似

$$f_N(t,\sigma_0)=\frac{e^{\sigma_0}}{t}\sum_{n=1}^{N}F_n$$
$$=\frac{e^{\sigma_0}}{t}\left\{\sum_{n=1}^{k-1}F_n+\frac{1}{2^{p+1}}\sum_{r=0}^{p}A_{p,r}F_{k+r}\right\} \quad (4.1)$$

で近似する．ここで

$$F_n=(-1)^n\,\mathrm{Im}\left[F\left(\frac{\sigma_0+i(n-0.5)\pi}{t}\right)\right]$$
$$N=k+p$$
$$A_{p,p}=1,\quad A_{p,r-1}=A_{p,r}+\left(\frac{p+1}{r}\right)$$

である．本サブルーチンでは，σ_0，N，p を以下のように決めている．σ_0 近似 $f(t,\sigma_0)$ は

$$f(t,\sigma_0)=f(t)-e^{-2\sigma_0}f(3t)+e^{-4\sigma_0}f(5t)-...$$

を満たすことから，σ_0 を，利用者の与える相対誤差的要求精度 EPSR より，

$$\mathrm{EPSR}\approx\left|\frac{f_N(t,\sigma_0)-f_{N-1}(t,\sigma_0)}{f_N(t,\sigma_0)}\right|=e^{-2\sigma_0} \quad (4.2)$$

となるように

$$\sigma_0=-\left[\frac{\log(\mathrm{EPSR})}{2}\right]+2$$

と定める．ただし [·] はガウス記号である．
(4.2) の条件を満足させる N は，t の値によっても異なるので，経験的に得られた次の関係式より決める．

$$N=[5\sigma_0]+[\sigma_0\cdot t/2.5] \quad (4.3)$$

また，p は

$\sigma_0\le 4$ のとき $p=5$
$4<\sigma_0\le 9$ のとき $p=[\sigma_0+1]$
$9<\sigma_0$ のとき $p=9$

としている．

パラメタ ERRV には，結果 $f_N(t,\sigma_0)$ の相対誤差が出力され，次の式より見積る．

$$\mathrm{ERRV}=\left|\frac{f_N(t,\sigma_0)-f_{N-1}(t,\sigma_0)}{f_N(t,\sigma_0)}\right|$$

F20-02-0101　　LAPS2, DLAPS2

ラプラス変換（一般の有理関数）
CALL LAPS2(A,NA,B,NB,T,DELT,L,EPSR, FT, T1,NEPS,ERRV,IFLG,VW,ICON)

(1)　機能

(1.1) で表される有理関数 $F(s)$ が与えられたとき，そのラプラス逆変換 $f(t)$ の値，$f(t_0)$，$f(t_0+\Delta t)$，…，$f(t_0+\Delta t(L\text{-}1))$ を求める．

$$\left.\begin{aligned} F(s)&=\frac{Q(s)}{P(s)}\\ Q(s)&=b_1s^m+b_2s^{m-1}+\ldots+b_ms+b_{m+1}\\ P(s)&=a_1s^n+a_2s^{n-1}+\ldots+a_ns+a_{n+1}\\ n&\geq m, a_1,a_2,\ldots,a_{n+1},b_1,b_2,\ldots,b_{m+1}:\text{実数}\end{aligned}\right\} \quad (1.1)$$

ここで $F(s)$ は $\mathrm{Re}(s)>0$ で正則であってもなくてもよい．

(2)　パラメタ

A ……**入力**．$P(s)$ の係数．
大きさ $n+1$ の 1 次元配列．A(1)=a_1，A(2)=a_2，…，A(n+1)=a_{n+1} の順に与える．

NA……**入力**．$P(s)$ の次数 n．

B……**入力**．$Q(s)$ の係数．
大きさ $m+1$ の 1 次元配列．B(1)=b_1，B(2)=b_2，…，B(m+1)=b_{m+1} の順に与える．

NB……**入力**．$Q(s)$ の次数 m．

T……**入力**．$f(t)$ の計算開始値 $t_0(\geq 0.0)$．

DELT ……**入力**．変数 t の増分 $\Delta t(>0.0)$．
DELT = 0.0 の場合は $f(t_0)$ だけが計算される．

L……**入力**．$f(t)$ の値を求めようとする点の個数．ただし L ≥ 1．

EPSR……**入力**．$f(t)$ の値に対する相対誤差の上限 (≥ 0.0)．単精度では 10^{-2} ～ 10^{-4}，倍精度では 10^{-2} ～ 10^{-7} 程度がよい．EPSR = 0.0 と入力したときは標準値 10^{-4} が採用される．

FT……**出力**．$f(t_0)$，$f(t_0+\Delta t)$，…，$f(t_0+\Delta t(\text{L-}1))$ の値．大きさ L の 1 次元配列．

T1……**出力**．t_0，$t_0+\Delta t$，…，$t_0+\Delta t$(L-1) の列．
大きさ L の 1 次元配列．

NEPS……**出力**．打切り項数 N．
大きさ L の 1 次元配列．FT(I) を計算するために用いた項数 N が NEPS(I) に格納される．

ERRV……**出力**．各結果 FT の相対誤差．
大きさ L の 1 次元配列．FT(I) における相対誤差が ERRV(I) に格納される．

IFLG……**出力**．正則性の判定結果．
$F(s)$ が $\mathrm{Re}(s)>0$ で正則であったとき，IFLG=0，そうでないとき IFLG=1 と出力する（使用上の注意② 参照）．

VW……**作業領域**．大きさ $n+m+2$ の 1 次元配列．

ICON……**出力**．コンディションコード．
表 LAPS2-1 参照．

表 LAPS2-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	結果 $f(t)$ の中に要求精度を満足しないものがある．	処理を続ける．$f(t_0+\Delta t\cdot l)$: l=0,1,…,L-1 の精度は配列 ERRV に出力される．
20000	$F(s)$ が $\mathrm{Re}(s)>\gamma_0$ で正則となる非負な実数 γ_0 が見つからなかった．	処理を打ち切る．
30000	次のいずれかであった． ① NB<0，又は ② NB>NA ③ T<0，又は ④ DELT<0 ⑤ L<1 EPSR<0 A(1) = 0	処理を打ち切る．

(3)　使用上の注意

a.　使用する副プログラム

① SSL II…　AFMAX，AMACH，HRWIZ，LAPS1，MGSSL

② FORTRAN 基本関数…　FLOAT，ALOG，ALOG10，CMPLX，AIMAG，EXP，ABS，INT，ATAN

b.　注意

① 有理関数 $F(s)$ は，$\mathrm{Re}(s)>0$ で正則であってもなくてもよい．しかし，$\mathrm{Re}(s)>0$ で正則であることが分っているならば，本サブルーチンよりも LAPS1 を用いた方が処理が速い．

② IFLG = 1 と出力されたときは，$F(s)$ が $\mathrm{Re}(s)>0$ で正則でないことを表す．このことは，$f(t)$ が $t\rightarrow\infty$ のとき指数関数的に増大することをも意味する．

③ t_0=0 と与えたときは $f(0)$ の値は，次の初期値定理より計算する．

$$f(0)=\begin{cases} b_1/a_1 & (n=m+1)\\ 0 & (n>m+1)\end{cases}$$

④ NA=NB の場合，(1.1) は

$$F(s)=\frac{Q(s)}{P(s)}=F_1(s)+F_2(s) \qquad (3.1)$$

ただし

$$F_1(s)\equiv b_1/a_1$$

$$F_2(s)\equiv \frac{c_2 s^{n-1}+c_3 s^{n-2}+\ldots+c_{n+1}}{a_1 s^n+a_2 s^{n-1}+\ldots\ldots+a_{n+1}}$$

と分解できる．
$F_1(s)$ の逆変換 $f_1(t)$ は

$$f_1(t)=\frac{b_1}{a_1}\delta(t) \qquad \delta(t)\text{: デルタ関数}$$

となる．$F_2(s)$ は 8 章の式 (8.20) の条件を満足しているので $F_2(s)$ の逆変換 $f_2(t)$ は式 (8.22) から計算できる．
NA=NB のとき本サブルーチンを用いて逆変換を計算すると，t>0 で $F_2(s)$ の逆変換の値が出力される．t=0 では浮動小数点数の最大値を出力する．

c. 使用例
有理関数

$$F(s)=\frac{s^2+4}{s^4-12s^3+54s^2-108s+81}$$

のラプラス逆変換 $f(t)$ を t_i=0.2+0.2(i-1)，i=1, 2，…，9 の各点で求める．EPSR=10^{-4} とする．

```
C     **EXAMPLE**
      DIMENSION A(5),B(3),T1(9),FT(9),
     *          ERRV(9),NEPS(9),VW(8)
      NB=2
      NB1=NB+1
      B(1)=1.0
      B(2)=0.0
      B(3)=4.0
      NA=4
      NA1=NA+1
      A(1)=1.0
      A(2)=-12.0
      A(3)=54.0
      A(4)=-108.0
      A(5)=81.0
      T=0.2
      DELT=0.2
      L=9
      EPSR=1.0E-4
      WRITE(6,600) NB,(I,B(I),I=1,NB1)
      WRITE(6,610) NA,(I,A(I),I=1,NA1)
      WRITE(6,620) EPSR
      CALL LAPS2(A,NA,B,NB,T,DELT,L,EPSR,
     *     FT,T1,NEPS,ERRV,IFLG,VW,ICON)
      WRITE(6,630) ICON,IFLG
      IF(ICON.GE.20000) STOP
      WRITE(6,640) (T1(I),FT(I),ERRV(I),
     *NEPS(I),I=1,L)
      STOP
  600 FORMAT(//24X,'NB=',I3//
     * (20X,'B(',I3,')=',E15.8/))
  610 FORMAT(/24X,'NA=',I3//
     * (20X,'A(',I3,')=',E15.8/))
  620 FORMAT(22X,'EPSR=',E15.8/)
  630 FORMAT(22X,'ICON=',I6,20X,'IFLG=',
     * I2)
  640 FORMAT(15X,'F(',F8.5,')=',E15.8,2X,
     * 'ERROR=',E15.8,2X,'N=',I3/)
      END
```

(4) 手法概要

ラプラス逆変換の計算法については，「8.6 ラプラス変換」のところで述べた．ここでは，本サブルーチンの計算手順について述べる．

① $F(s+\gamma)$ が Re(s)>0 で正則となるような実数 γ を求める．これはサブルーチン HRWIZ を使用して行う．

② $G(s)\equiv F(s+\gamma)$ に対するラプラス逆変換 $g(t)$ を計算する．これはサブルーチン LAPS 1 を使用して行う．

③ $f(t)=e^{\gamma t}g(t)$ の関係より $f(t)$ を計算する．

F20-03-0101　　LAPS3, DLAPS3

ラプラス変換（一般関数）
CALL LAPS3(FUN,T,DELT,L,EPSR,R0,FT, T1, NEPS,ERRV,ICON)

(1)　機能

関数 $F(s)$（非有理関数でもよい）が与えられたとき，そのラプラス逆変換 $f(t)$ の値 $f(t_0)$，$f(t_0+\Delta t)$，…，$f(t_0+\Delta t(\mathrm{L}-1))$ を求める．

ここで，$F(s)$ は $\mathrm{Re}(s)>\gamma_0$（γ_0: 収束座標）で正則であるとし，γ_0 は既知であるとする．

(2)　パラメタ

FUN …… **入力**．複素変数 s に対して関数 $F(s)$ の虚数部を計算する関数副プログラム名．
〔副プログラムの用意の仕方〕
FUNCTION FUN(S)
ここで S は複素数型変数である（使用例参照）

T …… **入力**．$f(t)$ の計算開始値 t_0（>0.0）

DELT …… **入力**．変数 t の増分 Δt（≥0.0）．
DELT=0.0 の場合は，$f(t_0)$ だけが計算される．

L …… **入力**．$f(t)$ の値を求めようとする点の個数．ただし L ≥ 1.

EPSR …… **入力**．$f(t)$ の値に対する相対誤差の上限（≥0.0）．単精度では 10^{-2} ～ 10^{-4}，倍精度では 10^{-4} ～ 10^{-7} 程度がよい．EPSR=0.0 あるいは EPSR≥1.0 と入力された場合は，標準値 10^{-4} が採用される．

R0 …… **入力**．関数 $F(s)$ が $\mathrm{Re}(s)>\gamma_0$ で正則であるとき，$\gamma\geq\gamma_0$ を満たす γの値．R0<0.0 と入力された場合は，本サブルーチン内で R0=0.0 とされる．

FT …… **出力**．$f(t_0)$，$f(t_0+\Delta t)$，…，$f(t_0+\Delta t(\mathrm{L}-1))$ の値の列．大きさ L の 1 次元配列．

T1 …… **出力**．t_0, $t_0+\Delta t$, …，$t_0+\Delta t(\mathrm{L}-1)$ の値の列．大きさ L の 1 次元配列．

NEPS …… **出力**．打切り項数．大きさ L の 1 次元配列．FT(I) を計算するのに用いた項数が NEPS(I) に出力される．

ERRV …… **出力**．各結果 FT(I) の相対誤差．大きさ L の 1 次元配列．FT(I) における相対誤差が ERRV(I) に出力される．

ICON …… **出力**．コンディションコード．
表 LAPS3-1 参照．

表 LAPS3-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	結果 $f(t)$ の中に要求精度を満たさないものがある．	処理を続ける．$f(t_0+\Delta t\cdot l)$，l=0，1，…，L-1 の精度は配列 ERRV に出力される．
20000	ある I に対して EXP(R0*T1(I)+σ_0)/T1(I) の値がオーバフローするほど大きくなった．	処理を打ち切る．結果は保証されない．
30000	次のいずれかであった． ① T≤0，又は DELT<0 ② L<1	処理を打ち切る．

(3)　使用上の注意

a.　使用する副プログラム

①　SSL II…　MGSSL

②　FORTRAN 基本関数…　FLOAT, ALOG, CMPLX, EXP, INT, ATAN, ABS

b.　注意

①　$F(s)$ が有理関数の場合には，サブルーチン LAPS1 あるいは LAPS2 を使用する方が効率が良い．

②　$F(s)$ が $\mathrm{Re}(s)>\gamma_0$ で正則であるとき，パラメタ R0 には $\gamma\geq\gamma_0$ なる γを入力すること．$\gamma_0\leq 0.0$ のときは R0 = 0.0 とすればよい．R0 < 0.0 と入力された場合には，サブルーチン内で R0=0.0 とみなす．

③　R0 = 0.0 の場合と，R0 > 0.0 の場合とで $f(t)$ が大きく変わるときには $\gamma_0>0.0$ と考えられる．本サブルーチンを使用して γ_0 を推定するには次のようにすればよい．
R0 に適当な値（例えば R0 = 0.0，R0 = 0.5 など）を入力して $f(t)$ を計算する．$f(t)$ の曲線が R0 = 0.0，0.5，1.0 で同じにならない t を推定しそれを t_a とする．$F(s)$ の特異点を $s_0=\nu_0+i\mu_0(\nu_0>0)$ とすると（特異点が 2 個以上存在する場合は，実数部の一番大きい特異点を s_0 とする），t が $t_a\approx\sigma_0/(\nu_0-R0)$ よりも大きい領域では $f(t)$ が R0 の値によって変化する．
ここで σ_0 の値は

$$\sigma_0=\left[-\frac{\log(\mathrm{EPSR})}{2}\right]+2$$

である．ゆえに $\gamma_0(=\nu_0)$ は

$$\gamma_0\approx R0+\frac{\sigma_0}{t_a} \qquad (3.1)$$

より推定できる．以上のことを例によって示す．

例:

$$F(s)=1/\sqrt{s^2-4s+8} \qquad (3.2)$$

に対し，EPSR = 10^{-2} とし R0 = 0.0，R0 = 0.5，R0 = 1.0 としたときの $f(t)$ の結果をそれぞれ図 LAPS3-1 の (a)，(b)，(c) に実線で示す．図中の点線は正しい結果である．R0 の値によって $f(t)$ が大きく変っていることが分かる．
R0 = 0.0 では $t_a \approx 2$，R0 = 0.5 では $t_a \approx 3.5$，そして R0 = 1.0 では $t_a \approx 5$ の位置で $f(t)$ が急に変化している．EPSR = 10^{-2} のとき σ_0=5 となるので (3.1) を用いて γ_0 を推定すると

R0=0.0 のとき $\gamma_0 \approx 2.5$
R0=0.5 のとき $\gamma_0 \approx 1.93$
R0=1.0 のとき $\gamma_0 \approx 2.0$

となる．(3.2) の $F(s)$ は $s_0 = 2 \pm i2\sqrt{3}$ に特異点があるので，上記の γ_0 は良好な推定値であることを示している．ちなみに，図中の点線は R0 = 2.0 と入力したときの結果である．

④ $f(t)$ の計算式（手法概要 (4.4)）に e^{σ_0}/t の因子があるので $f(0.0)$ を計算することはできない．$f(0.0)$ の値は，$f(0.01)$ や $f(0.0001)$ などで代用すること．しかし，t を余り小さくするとオーバフローを起こすことがあるので注意を要する．

⑤ 本サブルーチンでは，関数 $F(s)$ の虚数部の値を，

$$s_n = \frac{\sigma_0 + i(n-0.5)\pi}{t}, \quad n = 1,2,3,...$$
$$\sigma_0 > 0, \quad t > 0$$

の点で評価する．このような点で $F(s)$ が多価関数である場合には，関数副プログラムの中で，$F(s)$ の正しい枝を計算するよう注意が必要である．例えば，

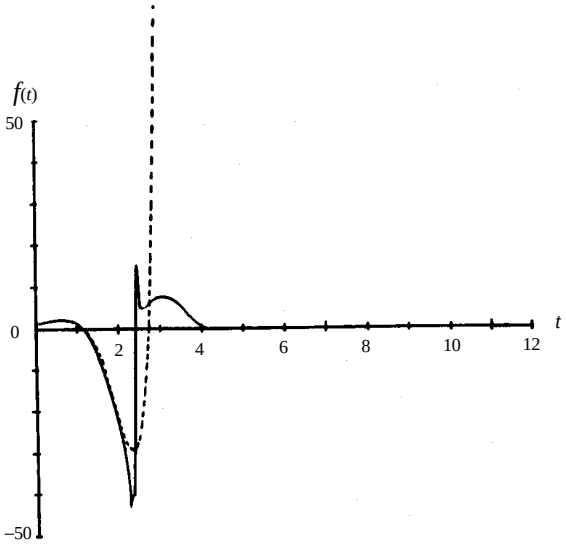
$$F(s)=1/\sqrt{s^2+1}$$

における $\sqrt{s^2+1}$ は，上記の s_n に対し，第 1 象限と第 3 象限に枝を作るが，この場合，関数副プログラム内では第 1 象限の枝を採用すること．

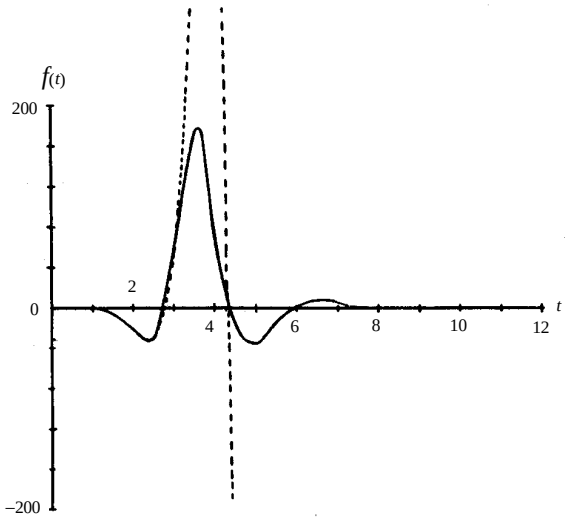
⑥
$$F(s)=\exp\left(-\sqrt{4s^2+1}\right)\Big/ s \qquad (3.4)$$

$$F(s)=1\Big/\left(s\cosh\sqrt{s^2+1}\right) \qquad (3.5)$$

などのように $s \to \infty$ のとき $F(s)$ に遅れ要素 $\exp(-as)$（$a > 0$）の項が含まれている場合には，関数副プログラムを定義する段階で注意が必要である．例えば，(3.4) をそのままの形で用いると，$f(t)$ の結果は図 LAPS3-2 の①に示すように $t < 2$ で $f(t) \neq 0.0$ となり，2 と 4 の間で振動を起こす．これは，Euler 変換の有効条件（8.6節の (8.25)）が満足されないために生ずる Gibbs の現象である．
(3.4) の場合，$t < 2$ ならば $f(t) = 0$ となることが理論的に分かるから

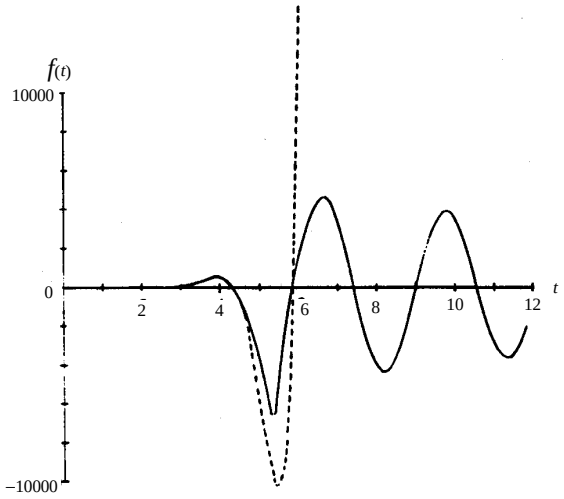


(a) EPSR=10^{-2}, R0=0.0



(b) EPSR=10^{-2}, R0=0.5

図 LAPS3-1 $F(s)= 1/\sqrt{s^2-4s+8}$ の γ_0 の推定（続く）



(c) EPSR=10^{-2}, R0=1.0

図 LAPS3-1 $F(s)= 1/\sqrt{s^2-4s+8}$ の χ の推定（続き）

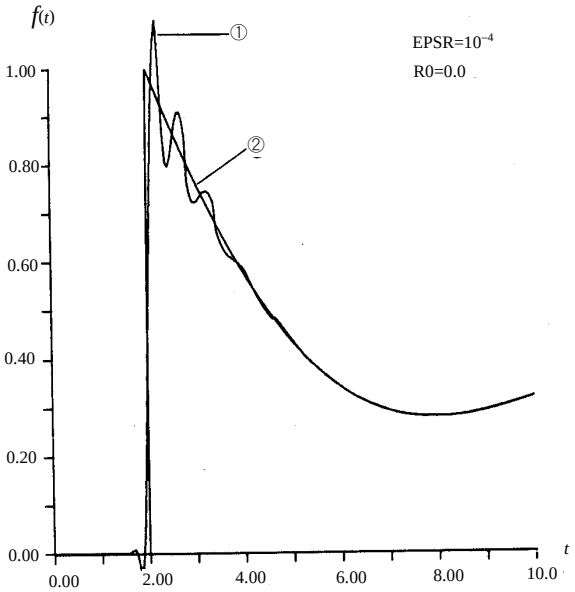


図 LAPS3-2 $F(s) = \exp(-\sqrt{4s^2+1})/s$ の変換

$$g(t') \equiv f(t'+2) \qquad t' > 0$$

の像関数

$$G(s) \equiv \exp(2s)\cdot F(s)$$
$$\exp = \left(2s-\sqrt{4s^2+1}\right)\Big/ s \qquad (3.6)$$

を扱えば十分であり，また $F(s)$ よりも $G(s)$ の方が高精度の結果を与える．ただし (3.6) の $\left(2s-\sqrt{4s^2+1}\right)$ の計算は桁落ちを生ずるので

$$2s-\sqrt{4s^2+1} = -\frac{1}{2s+\sqrt{4s^2+1}}$$

と変形した方がよい．図 LAPS3-2 の②は，このようにして $G(s)$ を用いたときの結果である．

一方，(3.5) の $F(s)$ は

$$F(s)=\frac{2}{s}\left\{\exp\left(-\sqrt{s^2+1}\right)-\exp\left(-3\sqrt{s^2+1}\right)+\exp\left(-5\sqrt{s^2+1}\right)-...\right\}$$

と展開し，(3.4) の場合にならって各項別に変換した方がよい．

c. 使用例

$$F(s)=\frac{\exp\left(-X\sqrt{s^2+1}\right)}{s^2+2} \quad , X=40$$

のラプラス逆変換 $f(t)$ を $(t_i\text{-}X)=0.2+0.2(i\text{-}1)$, $i=1, 2, \cdots, 800$ の各点で求める．EPSR=10^{-4} とする．$F(s)$ の代りに

$$G(s)=\exp(Xs)\cdot F(s)$$
$$=\frac{\exp\left\{\left(s-\sqrt{s^2+1}\right)X\right\}}{s^2+2}$$
$$=\frac{\exp\left\{\dfrac{-X}{s+\sqrt{s^2+1}}\right\}}{s^2+2}$$

を関数副プログラムで計算する．

$f(t)$ は，$G(s)$ の逆変換 $g(t)$ を用いて，

$$f(t)=\begin{cases} 0, & t<X \\ g(t-X), & t\geq X \end{cases}$$

となる．図 LAPS3-3 に，t-X を横軸としたときの $g(t\text{-}X)$ の結果を示す（文献 [101] 参照）．

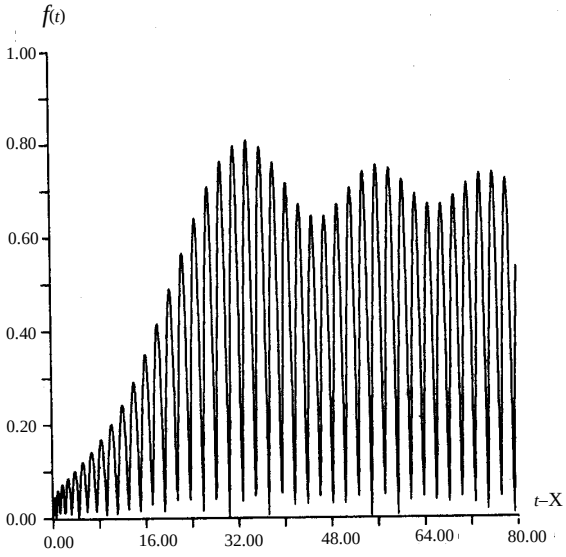


図 LAPS3-3 使用例の結果

```
C      **EXAMPLE**
       DIMENSION FT(800),T1(800),NEPS(800),
      *          ERRV(800)
       EXTERNAL FUN
       T=0.2
       DELT=0.2
       L=800
       EPSR=1.0E-4
       R0=0.0
       CALL LAPS3(FUN,T,DELT,L,EPSR,R0,FT,
      *           T1,NEPS,ERRV,ICON)
       WRITE(6,600) ICON
       WRITE(6,610) (T1(I),FT(I),NEPS(I),
      *             ERRV(I),I=1,L)
  600 FORMAT('1',20X,'ICON=',I5//)
  610 FORMAT(6X,'F(',F8.3,')=',E16.7,3X,
     * 'N=',I3,3X,'ERR=',E16.7)
       STOP
       END
       FUNCTION FUN(S)
       COMPLEX*8 S,SR
       SR=CSQRT(S*S+1.0)
       IF(REAL(SR).LT.0.0) SR=-SR
       SR=CEXP(40.0/(S+SR))*(S*S+2.0)
       FUN=AIMAG(1.0/SR)
       RETURN
       END
```

(4) 手法概要

ラプラス逆変換の計算法については，8 章の「8.6 ラプラス変換」のところで述べた．パラメタ R0 の値を γ とすると，γ>0 のとき，$G(s)=F(s+\gamma)$ は Re(s)>0 で正則となり，その逆変換 $g(t)$ を求めれば，$f(t)$ は

$$f(t)=e^{\gamma t}g(t) \tag{4.1}$$

より計算できる．以下では，Re(s)>0 で正則な関数を，あらためて $F(s)$ とし，その逆変換 $f(t)$ を計算することを考え，その場合の 8.6 節で述べたいくつかのパラメタ σ_0，p，N の決め方について述べる．

① σ_0 の値（近似誤差）

8.6 節の式 (8.23) より $f(t)$ の σ_0 近似 $f(t,\sigma_0)$ は

$$f(t,\sigma_0)=f(t)-e^{-2\sigma_0}f(3t)+e^{-4\sigma_0}f(5t)-..... \tag{4.2}$$

となることから，相対的要求精度 EPSR が

$$\mathrm{EPSR}=\left|\frac{f(t,\sigma_0)-f(t)}{f(t)}\right|\approx\left|\frac{f(t,\sigma_0)-f(t)}{f(3t)}\right|$$

$$\approx e^{-2\sigma_0}$$

となるように σ_0 を決める．

$$\sigma_0=\left[-\frac{\log(\mathrm{EPSR})}{2}\right]+2.0 \tag{4.3}$$

ただし［•］はガウス記号である．

(4.3) の右辺第 2 項に 2.0を加えてあるのは，EPSR = 10^{-1} のときでも σ_0 = 3 とするためである． σ_0 の値は，ほぼ $f_N(t,\sigma_0)$ の有効桁数と考えてよい．

② p の値

8.6 節の式 (8.28) より $f(t)$ の近似値 $f_N(t,\sigma_0)$ は，

$$f_N(t,\sigma_0)=\frac{e^{\sigma_0}}{t}\left\{\sum_{n=1}^{k-1}F_n+\sum_{q=0}^{p}\frac{1}{2^{p+1}}D^qF_k\right\} \tag{4.4}$$

から計算する．ここで (p+1) は Euler 変換の項数であり，本サブルーチンでは，p を経験的に

$$p=\sigma_0+2 \tag{4.5}$$

と決めている．$F(s)$ が $s\to\infty$ のとき $F(s)$ = O(s^{α}) である場合は，p の値が

$$p\geq(\alpha+5) \tag{4.6}$$

となるように σ_0，すなわち EPSR を与えれば，f(t) が超関数となる$F(s)$ にも (4.4) が適用できる．ここで$F(t)$ が超関数となる例を挙げる．

$$F(s)=\sqrt{s}$$

の逆変換 $f(t)$ は

$$f(t)=\frac{\delta(t)}{\sqrt{\pi t}}-\frac{U(t)}{\left(2\sqrt{\pi}\,t^{3/2}\right)}$$

となる．ここで $\delta(t)$ はディラックのデルタ関数，$U(t)$ は，

$$U(t)=\begin{cases}1, & t\geq 0\\ 0, & t<0\end{cases}$$

である．$F(s)=\sqrt{s}$ は，$\lim_{|s|\to\infty} F(s)\neq 0$ であるから，理論的には，本手法の適用外である．しかし $t>0$ なる範囲では，p を 6(>1/2+5) 以上にとれば普通の場合と同じようにして本サブルーチンを利用することができる．

③ N の値（打切り項数）

(4.4) の k，p より，

$$N=k+p$$

とすると，N は，打切り項数であり，関数 $F(s)$ の評価回数に等しい．k の値は，Euler 変換の有効条件（8.6 節の (8.25)）が満たされるように選ぶ必要がある．そのような k は，t に依存し，k_1，k_2 を定数とすると，

$$k=k_1+k_2t \tag{4.7}$$

なる関係がある．$t_0\leq t\leq t_0+\Delta t(\mathrm{L}-1)$ なる範囲で $f(t)$ を求める場合，本サブルーチンでは，k_1，k_2 を次の方法で決めている．まず，$t=t_0$ として打切り誤差 ERRV(1) が ERRV(1) < EPSR となる k を定め，これを k' とする．次に $t=t_0+\Delta t$(L-1) として同様に ERRV(L) < EPSR となる k を定め，これを k'' とする．これより連立方程式，

$$\begin{aligned}k'&=k_1+k_2t_0\\ k''&=k_1+k_2\{t_0+\Delta t(\mathrm{L}-1)\}\end{aligned} \tag{4.8}$$

が得られ，これを解いて k_1，k_2 を決める．これらを (4.7) に適用して k を計算する（ただし，小数以下は切り上げる）．N の値は，配列 NEPS に出力される．

A22-11-0101　　LAX, DLAX

実行列の連立 1 次方程式（クラウト法）
CALL LAX(A,K,N,B,EPSZ,ISW,IS,VW,IP,ICON)

(1)　機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

をクラウト法で解く．ただし，$\boldsymbol{A}$ は $n \times n$ の正則な実行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．$n \geq 1$ であること．

(2)　パラメタ

A …… **入力**．係数行列 *A*.
　　演算後，内容は保存されない．
　　A(K, N) なる 2 次元配列.
K …… **入力**．配列 A の整合寸法（≥N）.
N …… **入力**．係数行列 *A* の次数 *n*.
B …… **入力**．定数ベクトル $\boldsymbol{b}$.
　　出力．解ベクトル $\boldsymbol{x}$.
　　大きさ *n* の 1 次元配列.
EPSZ …… **入力**．ピボットの相対零判定値（≥0.0）.
　　0.0 のときは標準値が採用される.
　　（使用上の注意②参照）
ISW …… **入力**．制御情報.
　　同一の係数行列を持つ *l* (≥1) 組の方程式を解くとき，次のとおり指定する.
　　ISW = 1 のとき 1 組目の方程式を解く.
　　ISW = 2 のとき 2 組目以降の方程式を解く．ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う.
　　（使用上の注意③参照）
IS …… **出力**．行列 $\boldsymbol{A}$ の行列式を求めるための情報．演算後の配列 $\boldsymbol{A}$ の *n* 個の対角要素と IS の値を掛け合わせると行列式が得られる.
VW …… **作業領域**．大きさ *n* の 1 次元配列.
IP …… **作業領域**．大きさ *n* の 1 次元配列.
ICON …… **出力**．コンディションコード.
　　表 LAX-1 参照.

表 LAX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列のある行の要素がすべて零であったか又はピボットが相対的に零となった．係数行列は非正則の可能性が強い.	処理を打ち切る.
30000	K<N，N<1，EPSZ<0.0 又は ISW≠1，2 であった.	処理を打ち切る.

(3)　使用上の注意

a.　使用する副プログラム

①　SSL II…　ALU，LUX，AMACH，MGSSL

②　FORTRAN 基本関数…　ABS

b.　注意

①　本サブルーチンにより得られた解 $\boldsymbol{x}$ は，続けてサブルーチン LAXR を呼び出すことにより改良することができる.

②　ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，クラウト法による LU 分解の過程でピボットの値が 10 進 *s* 桁以上の桁落ちを生じた場合に，そのピボットを相対的に零と見なし，ICON=20000 として処理を打ち切る．EPSZ の標準値は丸め誤差単位を u としたとき，EPSZ = $16 \cdot u$ である．なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない.

③　同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW=2 として解くと，係数行列 *A* の LU 分解過程を省略するので計算時間が少なくなる．なお，この場合 IS の値は，ISW = 1 のときの値が保証される.

c.　使用例

同一の係数行列を持つ *l* 組 *n* 元連立 1 次方程式を解く．*n*≤100 の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),B(100),
     *          VW(100),IP(100)
      READ(5,500) N
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,A(I,J),J=1,N),
     *              I=1,N)
      READ(5,500) L
      M=1
      ISW=1
      EPSZ=1.0E-6
   10 READ(5,510) (B(I),I=1,N)
      WRITE(6,610) (I,B(I),I=1,N)
      CALL LAX(A,100,N,B,EPSZ,ISW,IS,VW,
     *         IP,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,630) (I,B(I),I=1,N)
      IF(L.EQ.M) GOTO 20
      M=M+1
      ISW=2
      GOTO 10
   20 DET=IS
      DO 30 I=1,N
      DET=DET*A(I,I)
   30 CONTINUE
      WRITE(6,640) DET
      STOP
```

```
500 FORMAT(I5)
510 FORMAT(4E15.7)
600 FORMAT('1',10X,'** COEFFICIENT ',
   * 'MATRIX'/12X,'ORDER=',I5/
   * (10X,4('(',I3,',',I3,')',E16.8)))
610 FORMAT(///10X,'CONSTANT VECTOR'
   * /(10X,5('(',I3,')',E16.8)))
620 FORMAT('0',10X,'CONDITION CODE=',I5)
630 FORMAT('0',10X,'SOLUTION VECTOR'
   * /(10X,5('(',I3,')',E16.8)))
640 FORMAT(///10X,
   * 'DETERMINANT OF COEFFICIENT ',
   * 'MATRIX=',E16.8)
    END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く.

a. 係数行列 $\boldsymbol{A}$ のLU 分解（クラウト法）
クラウト法により，係数行列 $\boldsymbol{A}$ を下三角行列 $\boldsymbol{L}$ と単位上三角行列 $\boldsymbol{U}$ の積に分解する．丸め誤差を少なくするため分解過程において部分ピボッティングを行う.

$$\boldsymbol{PA} = \boldsymbol{LU} \tag{4.2}$$

ここで，$\boldsymbol{P}$ はピボッティングによる行の入換えを行う置換行列である．この計算はサブルーチン ALU を用いて行う.

b. 求解（前進代入，後退代入）
連立 1 次方程式 (4.1) を解くことは

$$\boldsymbol{LUx} = \boldsymbol{Pb} \tag{4.3}$$

を解くことと同値であり，この方程式は,

$$\boldsymbol{Ly} = \boldsymbol{Pb} \tag{4.4}$$
$$\boldsymbol{Ux} = \boldsymbol{y} \tag{4.5}$$

として，前進代入，後退代入によって解く．この計算はサブルーチン LUX を用いて行う.

なお，詳細については，参考文献 [1]，[3] 及び，[4] を参照すること.

A25-11-0101　　LAXL, DLAXL

実行列の最小二乗解（ハウスホルダー変換）
CALL LAXL(A,K,M,N,B,ISW,VW,IVW,ICON)

(1)　機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

の最小二乗解 $\tilde{\boldsymbol{x}}$ を求める.

すなわち，$\boldsymbol{A}$ が $m \times n$，rank($\boldsymbol{A}$) = n なる実行列であり，$\boldsymbol{b}$ が m 次元の実定数ベクトルであるとき，

$$\left\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\right\|_2$$

が最小となる解を求める.

$m \geq n \geq 1$ であること.

(2)　パラメタ

A …… **入力**．係数行列 $\boldsymbol{A}$.
　　演算後，内容は保存されない.
　　A(K,N) なる 2 次元配列.

K …… **入力**．配列 A の整合寸法 (≥M).

M …… **入力**．係数行列 $\boldsymbol{A}$ の行数 m.

N …… **入力**．係数行列 $\boldsymbol{A}$ の列数 n.

B …… **入力**．定数ベクトル $\boldsymbol{b}$.
　　出力．最小二乗解 $\tilde{\boldsymbol{x}}$.
　　大きさ m の 1 次元配列.
　　（使用上の注意①参照）

ISW …… **入力**．制御情報.
　　同一の係数行列を持つ l（≥ 1）組の方程式を解くとき，次のとおり指定する.
　　ISW = 1 のとき，1 組目の方程式を解く.
　　ISW = 2 のとき，2 組目以降の方程式を解く．ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う.
　　（使用上の注意②参照）

VW …… **作業領域**．大きさ $2n$ の 1 次元配列.

IVW …… **作業領域**．大きさ n の 1 次元配列.

ICON …… **出力**．コンディションコード.
　　表 LAXL-1 参照.

表 LAXL-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	rank ($\boldsymbol{A}$) < n であった.	処理を打ち切る.
30000	K < M，M < N，N < 1 又は，ISW ≠ 1，2 であった.	処理を打ち切る.

(3)　使用上の注意

a.　使用する副プログラム

①　SSL Ⅱ …　AMACH，MGSSL，ULALH，ULALB

②　FORTRAN 基本関数…　SQRT

b.　注意

①　最小二乗解 $\tilde{\boldsymbol{x}}$ は配列 B の先頭 n 要素に格納される.

②　同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列 $\boldsymbol{A}$ の三角行列化過程を省略するので計算時間が少なくなる．（手法概要参照）.

c.　使用例

同一の係数行列を持つ l 組の連立 1 次方程式（未知数の数 n，方程式の数 m，$m \geq n$）の最小二乗解を求める．$m \leq 100$ の場合.

```
C      **EXAMPLE**
       DIMENSION A(100,100),B(100),
      *          VW(200),IVW(100)
       READ(5,500) M,N,L
       READ(5,510) ((A(I,J),I=1,M),J=1,N)
       WRITE(6,600)M,N,((I,J,A(I,J),J=1,N),
      *            I=1,M)
       LL=1
       ISW=1
    10 READ(5,510) (B(I),I=1,M)
       WRITE(6,610) (I,B(I),I=1,M)
       CALL LAXL(A,100,M,N,B,ISW,VW,IVW,
      *          ICON)
       WRITE(6,620) ICON
       IF(ICON.NE.0) STOP
       WRITE(6,630) (I,B(I),I=1,N)
       IF(L.EQ.LL) GO TO 20
       ISW=2
       LL=LL+1
       IF(LL.LE.L) GO TO 10
    20 WRITE(6,640)
       STOP
   500 FORMAT(3I5)
   510 FORMAT(4E15.7)
   600 FORMAT('1',5X,
      * 'LINEAR LEAST SQUARES SOLUTION'/
      * 6X,'ROW NUMBER=',I4,5X,
      * 'COLUMN NUMBER=',I4/6X,
      * 'COEFFICIENT MATRIX='/
      * (10X,4('(',I3,',',I3,')',E17.8,
      * 3X)))
   610 FORMAT(///10X,'CONSTANT VECTOR='
      * /(10X,4('(',I3,')',E17.8,3X)))
   620 FORMAT(' ',10X,'CONDITION CODE=',I6)
   630 FORMAT(' ',10X,'SOLUTION VECTOR='
      * /(10X,4('(',I3,')',E17.8,3X)))
   640 FORMAT('0',5X,
      * 'LINEAR LEAST SQUARES SOLUTION ',
      * 'END')
       END
```

(4) 手法概要

$\boldsymbol{A}$ を与えられた $m \times n$ なる実行列（rank($\boldsymbol{A}$) = $\boldsymbol{n}$), $\boldsymbol{b}$ を与えられた m 次元の定数ベクトルとするとき，

$$\left\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \right\|_2 \tag{4.1}$$

を最小にする解（最小二乗解）を求めることを考える．

さて，ユークリッドノルムは直交変換に対して不変であるから，$\boldsymbol{Q}$ を直交行列とすれば，

$$\left\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \right\|_2 = \left\| \boldsymbol{Q}\boldsymbol{b} - \boldsymbol{Q}\boldsymbol{A}\boldsymbol{x} \right\|_2 = \left\| \boldsymbol{C} - \boldsymbol{R}\boldsymbol{x} \right\|_2 \tag{4.2}$$

となる．ここで，$\boldsymbol{C} = \boldsymbol{Q}\boldsymbol{b}$，$\boldsymbol{R} = \boldsymbol{Q}\boldsymbol{A}$ とする．

いま，$\boldsymbol{Q}$ を

$$\boldsymbol{Q}\boldsymbol{A} = \boldsymbol{R} = \begin{bmatrix} \tilde{\boldsymbol{R}} \\ \cdots \\ 0 \end{bmatrix} \begin{matrix} \} \\ \} \end{matrix} \begin{matrix} n \\ (m-n) \end{matrix} \times n \tag{4.3}$$

ここで，$\tilde{\boldsymbol{R}}$ は $n \times n$ の上三角行列である．

となるように選べば，(4.2) は (4.4) で表せる．

$$\left\| \boldsymbol{C} - \boldsymbol{R}\boldsymbol{x} \right\|_2 = \left\| \begin{matrix} \tilde{\boldsymbol{C}} - \tilde{\boldsymbol{R}}\boldsymbol{x} \\ \cdots\cdots\cdots \\ \boldsymbol{C}_1 \end{matrix} \right\|_2 \tag{4.4}$$

ここで，$\tilde{\boldsymbol{C}}$ は $\boldsymbol{C}$ の最初の n 次元ベクトル，$\boldsymbol{C}_1$ は $\boldsymbol{C}$ から $\tilde{\boldsymbol{C}}$ 部分を除いた m - n 次元ベクトルである．

したがって，(4.4) は $\tilde{\boldsymbol{C}} - \tilde{\boldsymbol{R}}\boldsymbol{x} = \boldsymbol{0}$ となるとき最小となる．

すなわち，(4.3) のように行列 $\boldsymbol{A}$ を直交変換によって上三角行列とすることが出来るならば，(4.1) の最小二乗解 $\tilde{\boldsymbol{x}}$ は (4.5) により求めることができる．

$$\boldsymbol{x} = \tilde{\boldsymbol{R}}^{-1}\tilde{\boldsymbol{C}} \tag{4.5}$$

なお，本サブルーチンでは，次の手順により最小二乗解を求める．

a. 三角行列化 ····ハウスホルダー変換により，$\boldsymbol{A}$ を三角行列化して (4.3) により行列 $\boldsymbol{R}$ を求める．

b. 求解 ·····(4.5) により最小二乗解 $\tilde{\boldsymbol{x}}$ を求める．

実際の手続きは以下のとおりである．

a. 三角行列化

実行列を三角行列化する．すなわち，$\boldsymbol{A}^{(k)}$ の要素を $a_{ij}^{(k)}$ とするならば，

$$\begin{aligned} &\boldsymbol{A}^{(1)} = \boldsymbol{A} \\ &\boldsymbol{A}^{(k+1)} = \boldsymbol{P}^{(k)}\boldsymbol{A}^{(k)} \quad ,k = 1,...,n \end{aligned} \tag{4.7}$$

なる変換により，$a_{ik}^{(k+1)}=0$，$i=k+1$，…，m となるように直交行列 $P^{(k)}$ を (4.8) のように選ぶ（図 LAXL-1 参照）．

$$P^{(k)} = \mathrm{I} - \beta_k u^{(k)} u^{(k)T} \tag{4.8}$$

ここで $\sigma_k = \left(\sum_{i=k}^{m} \left(\alpha_{ik}^{(k)} \right)^2 \right)^{\frac{1}{2}}$

$\boldsymbol{u}^{(k)T} = \left(u_1^{(k)}, u_2^{(k)}, ..., u_m^{(k)} \right)$ として，

$\beta_k = \left[\sigma_k \left(\sigma_k + \left| a_{kk}^{(k)} \right| \right) \right]^{-1}$

$u_i^{(k)} = 0,\ i < k$

$u_k^{(k)} = \mathrm{sign}\left(a_{kk}^{(k)} \right) \left(\sigma_k + \left| a_{kk}^{(k)} \right| \right)$

$u_i^{(k)} = a_{ik}^{(k)},\ i > k$

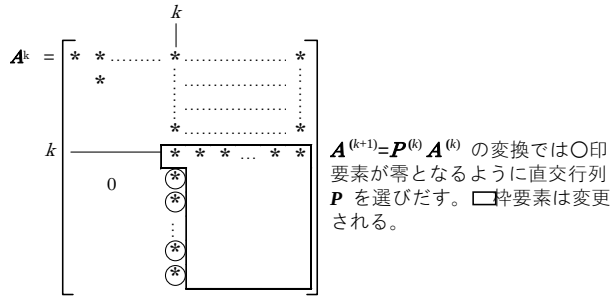


図 LAXL-1 三角行列化過程

したがって，次のようになる．

$$\boldsymbol{R} = \boldsymbol{A}^{(n+1)} = \boldsymbol{P}^{(n)}\boldsymbol{P}^{(n-1)}...\boldsymbol{P}^{(1)}\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{A}$$

この変換をハウスホルダー変換という．ところで，本サブルーチンでは，次の点を考慮している．

・ k 段目の変換に先だって，計算誤差を少なくするために (4.8) において σ_k の値が最大となるような列を選択する．すなわち，(4.9) のうちから最大となるような第 l 列を選び，それと第 k 列を入れ換える．

$$s_j^{(k)} = \sum_{i=k}^{m} \left(a_{ij}^{(k)}\right)^2, \quad i = k,...,n \qquad (4.9)$$

$s_l^{(k)}$ の値が,

$$s_l^{(k)} \le u \cdot \max_{1 \le i \le n} s_i^{(1)}$$

ここで，u は丸め誤差の単位である．であるならば，rank ($\boldsymbol{A}$) < n と見なし ICON = 20000 として処理を打ち切る．

・ 変換時における計算量を減らすために $\boldsymbol{P}^{(k)}$ を直接計算せず，次のように変換する．

$$\boldsymbol{A}^{(k+1)} = \left(\boldsymbol{I} - \beta_k \boldsymbol{u}^{(k)} \boldsymbol{u}^{(k)T}\right) \cdot \boldsymbol{A}^{(k)}$$
$$= \boldsymbol{A}^{(k)} - \boldsymbol{u}^{(k)} \boldsymbol{y}_k^T$$

ここで，　$\boldsymbol{y}_k^T = \beta_k \boldsymbol{u}^{(k)T} \boldsymbol{A}^{(k)}$

である．

なお，ベクトル $\boldsymbol{y}_k$ と $\boldsymbol{A}^{(k+1)}$ を計算するとき，$\boldsymbol{u}^{(k)}$ の先頭 (k-1) 個の要素がゼロであることを考慮する．

b. 求解

$$\boldsymbol{R} = \boldsymbol{P}^{(n)} \boldsymbol{P}^{(n-1)} ... \boldsymbol{P}^{(1)} \boldsymbol{A} \qquad (4.10)$$

であるから，定数ベクトルにも同じ変換を行う．

$$\boldsymbol{C} = \boldsymbol{Q}\boldsymbol{b} = \boldsymbol{P}^{(n)} \boldsymbol{P}^{(n-1)} ... \boldsymbol{P}^{(1)} \boldsymbol{b} \qquad (4.11)$$

この C を用いて，連立 1 次元方程式 (4.12) を解いて (4.1) の最小二乗解 $\tilde{\boldsymbol{x}}$ を求める．

$$\boldsymbol{R}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{C}} \qquad (4.12)$$

$\tilde{\boldsymbol{R}}$ は上三角行列であるから，後退代入により計算する．

なお，詳細については，参考文献 [6] を参照すること．

A25-21-0101　LAXLM, DLAXLM

実行列の最小二乗最小ノルム解（特異値分解法）
CALL LAXLM(A,KA,M,N,B,ISW,EPS,SIG,V, KV, VW,ICON)

(1) 機能

$m \times n$ の実行列 $\boldsymbol{A}$ を係数とする連立 1 次方程式

$$\boldsymbol{Ax}=\boldsymbol{b} \qquad (1.1)$$

の最小二乗最小ノルム解 $\boldsymbol{x}^+$ を求める.

すなわち，$\boldsymbol{b}$ が m 次元の実定数ベクトルであるとき,

$$\|\boldsymbol{b}-\boldsymbol{Ax}\|_2$$

が最小となる n 次元のベクトル $\boldsymbol{x}$ の中で,

$$\|\boldsymbol{x}\|_2$$

が最小となるものを求める.

$m \geq 1$，$n \geq 1$ であること.

(2) パラメタ

A ……**入力**．係数行列 $\boldsymbol{A}$.
演算後，内容は保存されない.
A(KA,N) なる 2 次元配列.

KA……**入力**．配列 A の整合寸法（≥M）.

M……**入力**．係数行列 $\boldsymbol{A}$ の行数 m.

N ……**入力**．係数行列 $\boldsymbol{A}$ の列数，行列 $\boldsymbol{V}$ の行数 n.

B……**入力**．定数ベクトル $\boldsymbol{b}$.
出力．最小二乗最小ノルム解 $\boldsymbol{x}^+$.
大きさ max(m,n) の 1 次元配列.
（使用上の注意①参照）

ISW ……**入力**．制御情報.
同一の係数行列を持つ方程式を 1 組だけ解く場合及び複数組解く場合に応じて，次のように指定する.
ISW=0 のとき，1 組だけ方程式を解く.
ISW=1 のとき，1 組目の方程式を解き，2 組目以降の方程式を解くための情報を残す.
ISW=2 のとき，2 組目以降の方程式を解く．ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタは，ISW=1 として用いたときのままで使う.
（使用上の注意②参照）

EPS……**入力**．特異値の相対零判定値（≥0.0）.
0.0 のときは標準値が採用される.
（使用上の注意④参照）

SIG ……**出力**．特異値.
大きさ n の 1 次元配列.
（使用上の注意③参照）

V……**作業領域**．V(KV,K) なる 2 次元配列.
K=min(M+1,N).
（使用上の注意②参照）

KV ……**入力**．配列 V の整合寸法（≥N）.

VW ……**作業領域**．大きさ n の 1 次元配列.

ICON ……**出力**．コンディションコード.
表 LAXLM-1 参照.

表 LAXLM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
15000	ある特異値が求められなかった.	処理を打ち切る.
30000	KA<M，M<1，N<1， KV<N，EPS<0.0 又は ISW≠0，1，2 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，ASVD1，MGSSL

② FORTRAN 基本関数… MIN0，SIGN，SQRT，AMAX1，ABS

b. 注意

① 最小二乗最小ノルム解 $\boldsymbol{x}^+$ は，配列 B の先頭 n 要素に格納される.

② 最小二乗最小ノルム解を一つだけ求めるときには，ISW=0 と指定すると，特異値分解による変換行列 $\boldsymbol{U}$ を計算しないので，計算量が少なく有利である.
なお，このとき配列 V の先頭の l 列に，行列 $\boldsymbol{V}$ が格納される．この行列 $\boldsymbol{V}$ は A の上に重ね書きができるように配慮されている.
ここに $l = \min(m, n)$.
同一の係数行列を持つ複数組の連立 1 次方程式の最小二乗最小ノルム解を続けて求める場合には，1 組目は ISW = 1 と指定し，2 組目以降は ISW = 2 と指定すれば，2 組目以降は，係数行列の特異値分解を省略するので計算量が少なくなる．なお，このときには，A の先頭の l 列には行列 $\boldsymbol{U}$ が，V の先頭の l 列には行列 $\boldsymbol{V}$ が格納されている.
手法概要参照のこと.

③ 特異値はすべて非負で，大きさの減少する順に格納される．ICON = 15000 の場合には，求められなかった特異値は -1 とし，大小順に整列しない.

④ 本サブルーチンは $\boldsymbol{A}$ に階数落ち（rank($\boldsymbol{A}$) < min(m, n)）があるか，又はそのおそれがあるときに用いるべきで，階数落ちがないことが明白な場合には，サブルーチン LAXL を用いるのが合理的である．
入力パラメタ EPS は $\boldsymbol{A}$ の階数を決定するための重要な役目を果すので，その選択は慎重に行わなければならない．手法概要④を参照のこと．

c. 使用例
m 行 n 列の係数行列 $\boldsymbol{A}$ に関する連立 1 次方程式 $\boldsymbol{Ax} = \boldsymbol{b}$ の最小二乗最小ノルム解を求める．行列 $\boldsymbol{V}$ を $\boldsymbol{A}$ の上に重ね書きする．

$1 \le m \le 100$，$1 \le n \le 100$ の場合．

```
C      **EXAMPLE**
       DIMENSION A(100,100),B(100),
      *          SIG(100),VW(100)
   10 READ(5,500) M,N
      IF(M.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,M),J=1,N)
      WRITE(6,600) M,N,((I,J,A(I,J),
     *             J=1,N),I=1,M)
      READ(5,510) (B(I),I=1,M)
      WRITE(6,610) (I,B(I),I=1,M)
      CALL LAXLM(A,100,M,N,B,0,0.0,SIG,
     *           A,100,VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
      CALL SVPRT(SIG,A,100,N,N)
      WRITE(6,630)(I,B(I),I=1,N)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',5X,
     * 'LEAST SQUARES AND MINIMAL',
     * ' NORM SOLUTION'/6X,
     * 'ROW NUMBER=',I4,5X,
     * 'COLUMN NUMBER=',I4/6X,
     * 'COEFFICIENT MATRIX='/
     * (10X,4('(',I3,',',I3,')',
     * E17.7,3X)))
  610 FORMAT(///10X,
     * 'CONSTANT VECTOR='
     * /(10X,4('(',I3,')',E17.7,3X)))
  620 FORMAT(' ',10X,'CONDITION CODE=',
     * I6)
  630 FORMAT('1',10X,
     * 'SOLUTION VECTOR='
     * /(10X,4('(',I3,')',E17.7,3X)))
      END
```

本使用例中のサブルーチン SVPRT は，特異値と固有ベクトルを印刷するものである．サブルーチン SVPRT は，サブルーチン ASVD1 の使用例に記載されている．

(4) 手法概要

$m \times n$ の行列 $\boldsymbol{A}$ と，m 次元の定数ベクトル $\boldsymbol{b}$ が与えられたとき，連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

の最小二乗最小ノルム解 $\boldsymbol{x}^+$ を求める．
すなわち残差ノルム

$$\left\| \boldsymbol{b} - \boldsymbol{Ax} \right\|_2 \tag{4.2}$$

を最小にする解 $\boldsymbol{x}$ の中で，$\boldsymbol{x}$ 自身のノルム

$$\left\| \boldsymbol{x} \right\|_2 \tag{4.3}$$

を最小にするものを求める．

本サブルーチンでは，m と n の大小関係に制限なく，どのような場合でも取り扱うことができるが，実際によく現れるという理由から，以下の説明では簡単のために $m \ge n$ と仮定する．

a. 特異値分解と最小二乗最小ノルム解
今，$\boldsymbol{A}$ の特異値分解が与えられているものとする．

$$\boldsymbol{A} = \boldsymbol{U\Sigma V}^T \tag{4.4}$$

ただし，$\boldsymbol{U}$ は (4.5) なる $m \times n$ の行列，

$$\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I} \tag{4.5}$$

$\boldsymbol{V}$ は n 次の直交行列

$$\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{VV}^T = \boldsymbol{I} \tag{4.6}$$

$\boldsymbol{\Sigma}$ は n 次の対角行列

$$\Sigma = \mathrm{diag}\left(\sigma_1, \sigma_2, ..., \sigma_n\right)$$

であり，

$$\sigma_1 \ge \sigma_2 \ge ... \ge \sigma_n \ge 0$$

となっているものとする．
U の右に m - n 個の列ベクトルを追加して，m 次の直交行列 $\boldsymbol{U}_c$ を作り，$\boldsymbol{\Sigma}$ の下に m - n 行 n 列の零行列を付加して $\boldsymbol{\Sigma}_c$ とすると，特異値分解は

$$\boldsymbol{A} = \boldsymbol{U}_c\boldsymbol{\Sigma}_c\boldsymbol{V}^T \tag{4.8}$$

の形に書き直すことができる．

$$\left.\begin{aligned}\tilde{\boldsymbol{b}}_c &= \boldsymbol{U}_c{}^T\boldsymbol{b},\\ \tilde{\boldsymbol{x}} &= \boldsymbol{V}^T\boldsymbol{x}\end{aligned}\right\} \tag{4.9}$$

とすれば，直交行列 $\boldsymbol{U}_c$ は $\|\ \|_2$ ノルムを変えないので，(4.8) 及び (4.9) により

$$\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2 = \left\|\boldsymbol{U}_c^T(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x})\right\|_2 = \left\|\tilde{\boldsymbol{b}}_c - \Sigma_c\tilde{\boldsymbol{x}}\right\|_2 \tag{4.10}$$

となる．したがって，$\left\|\tilde{\boldsymbol{b}}_c - \Sigma_c\tilde{\boldsymbol{x}}\right\|_2$ を最小にすればよい．ところが，$\boldsymbol{\Sigma}_c$ の最後の m - n 行は零行列であるので

$$\left\|\tilde{\boldsymbol{b}}_c - \boldsymbol{\Sigma}_c\tilde{\boldsymbol{x}}\right\|_2 = \left\|\begin{matrix}\tilde{b} - \Sigma\tilde{x}\\ \tilde{b}_1\end{matrix}\right\|_2 \tag{4.11}$$

の形である．ここに $\tilde{\boldsymbol{b}}$ は $\tilde{\boldsymbol{b}}_c$ の初めの n 個の要素から成る n 次元ベクトルで，

$$\tilde{\boldsymbol{b}} = \boldsymbol{U}^T\boldsymbol{b} \tag{4.12}$$

で与えられているものであり，$\tilde{\boldsymbol{b}}_1$ は $\tilde{\boldsymbol{b}}_c$ から $\tilde{\boldsymbol{b}}$ の部分を除いた m - n 次元ベクトルである．このようにして，結局

$$\left\|\tilde{\boldsymbol{b}} - \Sigma\tilde{\boldsymbol{x}}\right\|_2 \tag{4.13}$$

を最小化すればよいことが分かる．
今，行列 $\boldsymbol{A}$ の階数を r，つまり，

$$\sigma_1 \geq \ldots \geq \sigma_r > 0\,,\ \sigma_{r+1} = \ldots = \sigma_n = 0$$

と仮定する．$\tilde{\mathbf{x}} = (\tilde{x}_1,\ldots,\tilde{x}_n)^T$，$\tilde{\boldsymbol{b}} = \left(\tilde{\boldsymbol{b}}_1,\ldots,\tilde{\boldsymbol{b}}_n\right)^T$ とすると明らかに，最小二乗解の初めの r 個の成分は

$$\tilde{x}_i = \tilde{\boldsymbol{b}}_i/\sigma_i,\quad i = 1,2,\ldots,r \tag{4.14}$$

で与えられ，残りの成分は任意である．すなわち最小二乗解は一意的でない．これに対して，解自身のノルムを最小にするという条件を加えれば，解は一意的に定められる．それには，(4.14) で与えられる成分以外の成分を

$$\tilde{x}_i = 0\,,\qquad i = r+1,\ldots,n \tag{4.15}$$

とすればよい．(4.9) を考慮し，$\boldsymbol{V}$ が直交変換であって $\|\ \|_2$ ノルムを不変にすることから，

$$\boldsymbol{x}^+ = \boldsymbol{V}\tilde{\boldsymbol{x}} \tag{4.16}$$

で得られる $\boldsymbol{x}^+$ は目的の最小二乗最小ノルム解である．次に $\boldsymbol{x}^+$ を行列を用いて表すことを考える．
$\boldsymbol{\Sigma}$ の一般逆行列 $\boldsymbol{\Sigma}^+$ は明らかに次のように表される．

$$\Sigma^+ = \mathrm{diag}\left(\sigma_1^+,\sigma_2^+,\ldots,\sigma_n^+\right), \tag{4.17}$$

ここに，

$$\sigma_i{}^+ = \begin{cases}1/\sigma_i, & \sigma_i > 0\\ 0, & \sigma_i = 0\end{cases} \tag{4.18}$$

である．この $\boldsymbol{\Sigma}^+$ を使えば (4.14)，(4.15) から

$$\tilde{\boldsymbol{x}} = \boldsymbol{\Sigma}^+\tilde{\boldsymbol{b}} \tag{4.19}$$

となる．(4.16)，(4.19)，及び (4.12) より，最小二乗最小ノルム解 $\boldsymbol{x}^+$ は

$$\boldsymbol{x}^+ = \boldsymbol{V}\boldsymbol{\Sigma}^+\boldsymbol{U}^T\boldsymbol{b} \tag{4.20}$$

として与えられる．(4.20) は一般逆行列 $\boldsymbol{A}^+$ を使って $\boldsymbol{x}^+ = \boldsymbol{A}^+\boldsymbol{b}$ と書くことができる．サブルーチン GINV の手法概要参照．

b. 本サブルーチンにおける計算手順

① $\boldsymbol{A}$ に左と右から交互にハウスホルダー変換を施して，上 2 重対角行列 $\boldsymbol{J}_0$ に変換する．

$$\boldsymbol{J}_0 = \boldsymbol{P}_n\boldsymbol{P}_{n\text{-}1}\ldots\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{Q}_1\boldsymbol{Q}_2\ldots\boldsymbol{Q}_{n\text{-}2} \tag{4.21}$$

詳細については，サブルーチン ASVD1 手法概要 を参照のこと．

② 上 2 重対角行列 $\boldsymbol{J}_0$ に左と右から直交変換を施して，対角行列 $\boldsymbol{\Sigma}$ に変換する．

$$\Sigma = \boldsymbol{S}_q{}^T\ldots\boldsymbol{S}_1{}^T\boldsymbol{J}_0\boldsymbol{T}_1\ldots\boldsymbol{T}_q \tag{4.22}$$

各々の $\boldsymbol{S}_i$，$\boldsymbol{T}_i$ はそれぞれ

$$\boldsymbol{S}_i = \boldsymbol{L}_2\boldsymbol{L}_3\ldots\boldsymbol{L}_n \tag{4.23}$$
$$\boldsymbol{T}_i = \boldsymbol{R}_2\boldsymbol{R}_3\ldots\boldsymbol{R}_n \tag{4.24}$$

の形で，2 次元の回転変換の積として与えられる．詳細については，サブルーチン ASVD1 手法概要を参照のこと．

③ 行列 $\boldsymbol{U}^T$ と $\boldsymbol{V}$ は，(4.4)，(4.21) 及び (4.22) から

$$\boldsymbol{U}^T = \boldsymbol{S}_q^T \ldots \boldsymbol{S}_1^T \boldsymbol{P}_n \ldots \boldsymbol{P}_1 \qquad (4.25)$$

$$\boldsymbol{V} = \boldsymbol{Q}_1 \ldots \boldsymbol{Q}_{n-2} \boldsymbol{T}_1 \ldots \boldsymbol{T}_q \qquad (4.26)$$

で与えられる．
$\boldsymbol{U}$ と $\boldsymbol{V}$ とは，変換行列を逐次右から乗ずることにより，それぞれ配列 A と V の位置に生成される．
ただし，以上は ISW = 1 のときで，ISW = 0 の場合には，$\boldsymbol{U}$ を生成することなく，直接に $\boldsymbol{U}^T\boldsymbol{b}$ を計算する．それには，$\boldsymbol{U}^T$ を構成する変換行列を，逐次 $\boldsymbol{b}$ の左から乗ずればよい．

④ $\boldsymbol{b}$ の左から，順次 $\boldsymbol{U}^T$，$\boldsymbol{\Sigma}^+$，$\boldsymbol{V}$ を乗じて

$$\boldsymbol{x}^+ = \boldsymbol{V}\boldsymbol{\Sigma}^+\boldsymbol{U}^T\boldsymbol{b}$$

を作る．ただし，ISW = 0 の場合は $\boldsymbol{U}^T$ を乗ずる部分を省略する．

$\boldsymbol{\Sigma}^+$ を乗ずるところで，実際には特異値 σ_i に対する零判定を行う．判定基準としては $\|\boldsymbol{J}_0\|_\infty \cdot$ EPS を用いる．判定基準より小さい特異値は 0 とみなされる．EPS として 0.0 が入力されたときは，$16u$ を標準値として用いる．ここに，u は丸め誤差の単位である．

以上の処理を，ISW = 0 の場合には直接本サブルーチンで，ISW = 1 の場合には，特異値分解の部分を，サブルーチン ASVD1 により行っている．

なお，詳細については，サブルーチン ASVD1 の手法概要及び参考文献 [11] を参照すること．

A25-11-0401　LAXLR, DLAXLR

実行列の最小二乗解の反復改良
CALL LAXLR(X,A,K,M,N,FA,FD,B,IP,VW,ICON)

(1) 機能

$m \times n$ なる実行列 $\boldsymbol{A}$ (rank ($\boldsymbol{A}$)=n) の連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (1.1)$$

の近似最小二乗解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は m 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は，列の入換えを行う部分ピボッティングを伴って (1.2) のとおりハウスホルダー変換されていること．

$$\boldsymbol{R} = \boldsymbol{QA} \qquad (1.2)$$

ここで，$\boldsymbol{R}$ は上三角行列，$\boldsymbol{Q}$ は直交行列である．

$m \geq n \geq 1$ であること．

(2) パラメタ

X ……**入力**．近似最小二乗解 $\tilde{\boldsymbol{x}}$．
　　出力．反復改良された最小二乗解 $\boldsymbol{x}$．
　　大きさ n の 1 次元配列．

A ……**入力**．係数行列 $\boldsymbol{A}$．
　　A(K,N) なる 2 次元配列．

K ……**入力**．配列 A の整合寸法（≥M）．

M……**入力**．係数行列 $\boldsymbol{A}$ の行数 m．

N ……**入力**．係数行列 $\boldsymbol{A}$ の列数 n．
　　（使用上の注意②参照）

FA ……**入力**．行列 $\boldsymbol{R}$ の上三角部分と行列 $\boldsymbol{Q}$．
　　FA(K,N) なる 2 次元配列．
　　（使用上の注意①参照．）

FD ……**入力**．行列 $\boldsymbol{R}$ の対角部分．
　　大きさ n の 1 次元配列．
　　（使用上の注意①参照．）

B……**入力**．定数ベクトル $\boldsymbol{b}$．
　　大きさ m の 1 次元ベクトル．

IP……**入力**．部分ピボッティングによる列の入換えの履歴を示すトランスポジションベクトル．
　　大きさ n の 1 次元ベクトル．
　　（使用上の注意①参照．）

VW……**作業領域**．大きさ m の 1 次元配列．

ICON……**出力**．コンディションコード．
　　表 LAXLR-1 参照．

表 LAXLR-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	rank($\boldsymbol{A}$)<n であった．	処理を打ち切る．
25000	係数行列の条件が悪く，収束条件を満足しなかった．	処理を打ち切る．（収束条件については 手法概要 b. 参照のこと）
30000	K<M，M<N 又は N<1 であった．	処理を打ち切る．

(3) 使用上の注意

a.　使用する副プログラム

①　SSL II…AMACH，MAV，MGSSL，ULALB

②　FORTRAN 基本関数…　IABS，ABS，SQRT

b.　注意

①　本サブルーチンは，サブルーチン LAXL により求められた近似最小二乗解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである．
したがって，本サブルーチンを呼び出す前にサブルーチン LAXL を呼び出して近似最小二乗解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された最小二乗解を得ることができる．
この場合，パラメタ X，FA，FD，IP には直前に呼び出したサブルーチン LAXL のパラメタ B，A，VW，IVW の値（サブルーチン LAXL の項目参照）をそのまま入力すること．ただし，本サブルーチンでは係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LAXL を呼ぶに先立ってこれらを保存しておく必要がある．
具体的には，使用例を参照されたい．

②　N=-n として指定すれば，サブルーチン LAXL により得られた近似最小二乗解 $\tilde{\boldsymbol{x}}$ の残差ベクトルのユークリッドノルム $\left(\left\| \boldsymbol{b} - \boldsymbol{A}\tilde{\boldsymbol{x}} \right\|_2 \right)$ を得ることができる．
この場合，本サブルーチンでは解の反復改良は行わず，単に残差ベクトルのユークリッドノルムを計算し，VW(1) に出力する．

c.　使用例

未知数の数 m，方程式の数 n（≤ m）の連立 1 次方程式の最小二乗解 $\tilde{\boldsymbol{x}}$ をサブルーチン LAXL により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する．

$m \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(50,50),FA(50,50),X(50),
     *   B(50),VW(100),IVW(50),FD(100)
      CHARACTER*4 NT1(6),NT2(4),NT3(4)
      DATA NT1/'CO  ','EF  ','FI  ',
     *         'CI  ','EN  ','T   '/,
     *     NT2/'CO  ','NS  ','TA  ',
     *         'NT  '/,
     *     NT3/'SO  ','LU  ','TI  ',
     *         'ON  '/
      READ(5,500) M,N
      WRITE(6,600) M,N
      READ(5,510) ((A(I,J),I=1,M),J=1,N)
      CALL PGM(NT1,6,A,50,M,N)
      READ(5,510) (B(I),I=1,M)
      CALL PGM(NT2,4,B,M,M,1)
      DO 20 I=1,M
      X(I)=B(I)
      DO 10 J=1,N
      FA(I,J)=A(I,J)
   10 CONTINUE
   20 CONTINUE
      ISW=1
      CALL LAXL(FA,50,M,N,X,ISW,FD,IVW,
     *          ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      CALL LAXLR(X,A,50,M,N,FA,FD,B,IVW,
     *           VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) STOP
      CALL PGM(NT3,4,X,N,N,1)
      STOP
  500 FORMAT(2I5)
  510 FORMAT(10F8.3)
  600 FORMAT('1',/6X,'LINEAR LEAST '
     * 'SQUARES SOLUTION'/
     * 6X,'ROW NUMBER=',I4/
     * 6X,'COLUMN NUMBER=',I4)
  610 FORMAT(' ',5X,'ICON OF LAXL=',I6)
  620 FORMAT(' ',5X,'ICON OF LAXLR=',I6)
      END
```

サブルーチン PGM は，実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている．

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似最小二乗解（以降，近似解と略す）$\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える．

a. 反復改良の考え方

反復改良とは，(4.1) の解 $\boldsymbol{x}$ の s 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)}=\tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$

$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{4.3}$$

$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{4.4}$$

s=1,2,⋯

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$ (s=1,2,⋯) を逐次求めていくことである．

なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる．

ただし，係数行列 $\boldsymbol{A}$ の条件が非常に悪い場合には改良されないことがある．

（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順

サブルーチン LAXL で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする．

本サブルーチンでは，以下を繰り返す．

- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MAV を用いて行う．
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはスレーブサブルーチン ULALB を用いて行う．
- 修正された近似解 $\boldsymbol{x}^{(s+1)}$ を (4.4) から求める．

収束判定法は，以下のとおりである．

ここで丸め誤差の単位を u としたとき，s 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2\cdot u \tag{4.5}$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $\boldsymbol{x}^{(s+1)}$ を解として採用する．

ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s+1)}\right\|_\infty} > \frac{1}{2}\cdot\frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON=25000 として処理を打ち切る．

A22-11-0401　LAXR, DLAXR

実行列の連立 1 次方程式の解の反復改良
CALL LAXR(X,A,K,N,FA,B,IP,VW,ICON)

(1)　機能

$n \times n$ の実行列 $\boldsymbol{A}$ の連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (1.1)$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は (1.2) のとおり LU 分解されていること．

$$\boldsymbol{PA} = \boldsymbol{LU} \qquad (1.2)$$

ここで，$\boldsymbol{L}$ と $\boldsymbol{U}$ はそれぞれ $n \times n$ の下三角行列と単位上三角行列，$\boldsymbol{P}$ は部分ピボッティングによる行の入換えを行う置換行列である．

$n \geq 1$ であること．

(2)　パラメタ

X ……**入力**．近似解ベクトル $\tilde{\boldsymbol{x}}$．
出力．反復改良された解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

A ……**入力**．係数行列 $\boldsymbol{A}$．
A(K,N) なる 2 次元配列．

K ……**入力**．配列 A，FA の整合寸法（≥N）．

N ……**入力**．係数行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意②参照．）

FA ……**入力**．行列 $\boldsymbol{L}$ と行列 $\boldsymbol{U}$．
図 LAXR-1 参照．
FA(K,N) なる 2 次元配列．
（使用上の注意①参照．）

B……**入力**．定数ベクトル $\boldsymbol{b}$．
大きさ n の 1 次元配列．

IP……**入力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．
大きさ n の 1 次元配列．
（使用上の注意①参照）

VW……**作業領域**．大きさ n の 1 次元配列．

ICON……**出力**．コンディションコード．
表 LAXR-1 参照．

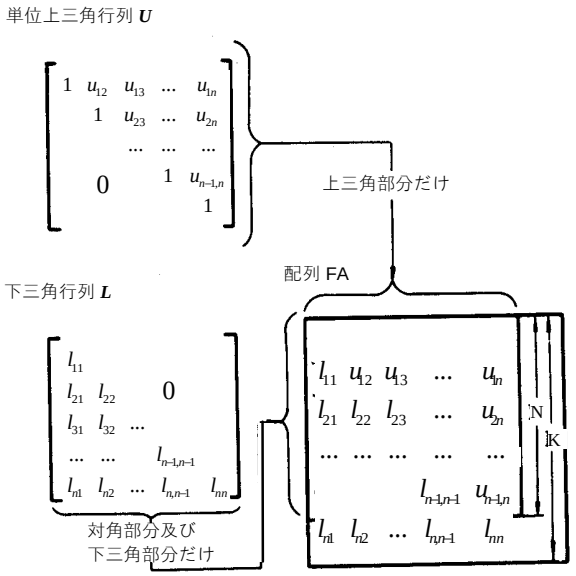


図 LAXR-1　係数行列 A を LU 分解した下三角行列 L 及び単位上三角行列 U の各要素の格納方法

表 LAXR-1　コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	係数行列が非正則であった．	処理を打ち切る．
25000	係数行列の条件が悪く，収束条件を満足しなかった．	処理を打ち切る．（収束条件については手法概要 b. 参照のこと）
30000	K<N 又は N<1 であった．	処理を打ち切る．

(3)　使用上の注意

a.　使用する副プログラム

①　SSL II…　AMACH，LUX，MAV，MGSSL

②　FORTRAN 基本関数…　ABS

b.　注意

①　本サブルーチンは，サブルーチン LAX により求められた近似解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである．
したがって，本サブルーチンを呼び出す前にサブルーチン LAX を呼び出して近似解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された解を得ることができる．
この場合，パラメタ X，FA，IP には直前に呼び出したサブルーチン LAX の結果をそのまま入力すること．ただし，本サブルーチンでは係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LAX を呼ぶに先立ってそれらを保存しておく必要がある．
具体的には使用例を参照されたい．

② N = –*n* と指定すれば，サブルーチン LAX により得られた近似解 $\tilde{\boldsymbol{x}}$ の推定精度（相対誤差）を得ることができる．
この場合，本サブルーチンでは解の反復改良は行わず，単に相対誤差を計算し，VW(1) に出力する．
精度の推定については，手法概要 c. を参照されたい．

c. 使用例
n 元の連立 1 次方程式の近似解 $\tilde{\boldsymbol{x}}$ をサブルーチン LAX により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する．
n ≤ 100 の場合．

```
C     **EXAMPLE**
      DIMENSION A(100,100),FA(100,100),
     *   X(100),B(100),VW(100),IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,A(I,J),J=1,N),
     *             I=1,N)
      READ(5,510) (B(I),I=1,N)
      WRITE(6,610) (I,B(I),I=1,N)
      DO 20 I=1,N
      X(I)=B(I)
      DO 20 J=1,N
      FA(J,I)=A(J,I)
   20 CONTINUE
      EPSZ=0.0E0
      ISW=1
      K=100
      CALL LAX(FA,K,N,X,EPSZ,ISW,IS,VW,IP,
     *         ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL LAXR(X,A,K,N,FA,B,IP,VW,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,640) (I,X(I),I=1,N)
      DET=IS
      DO 30 I=1,N
      DET=DET*FA(I,I)
   30 CONTINUE
      WRITE(6,650) DET
      STOP
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1',10X,
     * '**COEFFICIENT MATRIX'/
     * 12X,'ORDER=',I5/(10X,4('(',I3,',',
     * I3,')',E17.8)))
  610 FORMAT(///10X,'CONSTANT VECTOR'/
     * (10X,4('(',I3,')',E17.8)))
  620 FORMAT('0',10X,'LAX  ICON=',I5)
  630 FORMAT('0',10X,'LAXR ICON=',I5)
  640 FORMAT('0',10X,'SOLUTION VECTOR'/
     * (10X,4('(',I3,')',E17.8)))
  650 FORMAT(///10X,
     * 'DETERMINANT OF COEFFICIENT ',
     * 'MATRIX=',E17.8)
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える．

a. 反復改良の考え方
反復改良とは，(4.1) の解 $\boldsymbol{x}$ の s 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)} = \tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$
$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{4.3}$$
$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{4.4}$$
s=1,2,⋯

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$ (s=1,2,⋯) を逐次求めていくことである．
なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる．
ただし，係数行列 $\boldsymbol{A}$ の条件が非常に 悪い場合には改良されないことがある．
（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順
サブルーチン LAX で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする．
本サブルーチンでは，以下を繰り返す．
- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MAV を用いて行う．
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはサブルーチン LUX を用いて行う．
- 修正された近似解 $\boldsymbol{x}^{(s+1)}$ を (4.4) から求める．

収束判定法は，以下のとおりである．
ここで丸め誤差の単位を u としたとき，s 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2u \tag{4.5}$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $\boldsymbol{x}^{(s+1)}$ を解として採用する．
ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s+1)}\right\|_\infty} > \frac{1}{2}\cdot\frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON = 25000 として処理を打ち切る．

c. 近似解の精度推定
近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)}$（$= \boldsymbol{x}^{(1)} - \boldsymbol{x}$）とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ である．いま反復法が収束するならば，$\boldsymbol{e}^{(1)}$ と $\boldsymbol{d}^{(1)}$ はほぼ等しいものと考えられるから，近似解の相対誤差は $\left\|\boldsymbol{d}^{(1)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ によって推定できる．
（3.4 (2) f. 近似解の精度推定 参照）

なお，詳細については，文献 [1]，[3] 及び [5] を参照すること．

A52-11-0101 LBX1,DLBX1

実バンド行列の連立 1 次方程式（ガウス消去法）
CALL LBX1(A,N,NH1,NH2,B,EPSZ,ISW,IS, FL, VW,IP,ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b} \qquad (1.1)$$

をガウス消去法で解く．ただし，$\boldsymbol{A}$ は $n \times n$，下バンド幅 h_1，上バンド幅 h_2 のバンド行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

$n > h_1 \geq 0$，$n > h_2 \geq 0$ であること．

(2) パラメタ

A …… **入力**．係数行列 $\boldsymbol{A}$．
演算後，内容は保存されない．
バンド行列用圧縮モード．
大きさ $n \cdot \min(h_1+h_2+1,n)$ の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$ の次数 n．

NH1…… **入力**．下バンド幅 h_1．

NH2…… **入力**．上バンド幅 h_2．

B…… **入力**．定数ベクトル．
出力．解ベクトル．
大きさ n の 1 次元配列．

EPSZ …… **入力**．ピボットの相対零判定値 (≥0.0)．
0.0 のときは標準値が採用される．
（使用上の注意①参照．）

ISW …… **入力**．制御情報．
同一の係数行列を持つ l (≥1) 組の方程式を解くとき，次のとおり指定する．
ISW=1 のとき，1 組目の方程式を解く．
ISW=2 のとき，2 組目以降の方程式を解く．ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．（使用上の注意②参照）

IS…… **出力**．行列 $\boldsymbol{A}$ の行列式を求めるための情報．（使用上の注意③参照）

FL…… **作業領域**．大きさ $(n-1) \cdot h_1$ の 1 次元配列．

VW…… **作業領域**．大きさ n の 1 次元配列．

IP…… **作業領域**．大きさ n の 1 次元配列．

ICON…… **出力**．コンディションコード．
表 LBX1-1 参照．

表 LBX1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い．	処理を打ち切る．
30000	N≤NH1，N≤NH2，NH1<0，NH2<0，EPSZ<0.0 又は ISW≠1,2 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，BLUX1，BLU1，MGSSL

② FORTRAN 基本関数… ABS，MIN0

b. 注意

① 本サブルーチンでは，ピボットの相対零判定法として，ガウス消去法による LU 分解の過程でピボットの値が（係数行列の絶対値最大要素）*EPSZ より小さい場合に，そのピボットを相対的に零と見なし，ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない．

② 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW=2 として解くと，係数行列 $\boldsymbol{A}$ の LU 分解過程を省略するので計算時間が少なくなる．なお，この場合 IS の値は，ISW = 1 のときの値が保証される．

③ 行列 $\boldsymbol{A}$ の行列式は，n 個の配列要素 A ($i \cdot h$+1)，i= 0，1，…，n-1 の積と IS の値を掛け合わせて得られる．ただし，
$h = \min(h_1+h_1+1,n)$.

④ 本サブルーチンでは，バンド行列の特性を生かして，データ格納領域を節約出来るように，バンド行列を格納しているが，バンド幅の大きさによっては，密行列の連立 1 次方程式を解くサブルーチン LAX より（作業領域を含めて）データ格納領域を多く必要とする場合がある．その時は，サブルーチン LAX を用いた方がデータ格納領域を節約できる．本サブルーチンの特性を生かせるのは，次数 n の係数行列の上バンド幅と下バンド幅が等しいなら，各々のバンド幅が約 n / 3 以下の場合である．

c. 使用例

同一の係数行列を持つ l 組の n 元連立 1 次方程式を解く.

$n \le 100$, $h_1 \le 20$, $h_2 \le 20$ の場合.

```
C     ** EXAMPLE **
      DIMENSION A(4100),B(100),IP(100),
     *   FL(1980),VW(100)
      CHARACTER*4 NT1(6),NT2(4),NT3(4)
      DATA NT1/'CO  ','EF  ','FI  ',
     *         'CI  ','EN  ','T   '/,
     *     NT2/'CO  ','NS  ','TA  ',
     *         'NT  '/,
     *     NT3/'SO  ','LU  ','TI  ',
     *         'ON  '/
      READ(5,500) N,NH1,NH2,L
      WRITE(6,600) N,NH1,NH2
      NT=N*MIN0(N,NH1+NH2+1)
      READ(5,510) (A(I),I=1,NT)
      M=1
      ISW=1
      EPSZ=1.0E-6
      CALL PBM(NT1,6,A,N,NH1,NH2)
   10 READ(5,510) (B(I),I=1,N)
      CALL PGM(NT2,4,B,N,N,1)
      CALL LBX1(A,N,NH1,NH2,B,EPSZ,ISW,IS,
     *          FL,VW,IP,ICON)
      WRITE(6,610) ICON
      IF(ICON.EQ.0) GOTO 20
      WRITE(6,620)
      STOP
   20 CALL PGM(NT3,4,B,N,N,1)
      M=M+1
      ISW=2
      IF(L.GT.M) GO TO 10
      WRITE(6,630)
      STOP
  500 FORMAT(4I4)
  510 FORMAT(4E15.7)
  600 FORMAT('1'///
     * 5X,'LINEAR EQUATIONS  AX=B'/
     * 5X,'ORDER=',I4/
     * 5X,'SUB-DIAGONAL LINES=',I4/
     * 5X,'SUPER-DIAGONAL LINES=',I4)
  610 FORMAT(' ',4X,'ICON=',I5)
  620 FORMAT(' '/5X,
     * '** ABNORMAL END **')
  630 FORMAT(' '/5X,'** NORMAL END **')
      END

C     ** MATRIX PRINT (REAL BAND) **
      SUBROUTINE PBM(ICOM,L,A,N,NH1,NH2)
      DIMENSION A(1)
      CHARACTER*4 ICOM(1)
      WRITE(6,600) (ICOM(I),I=1,L)
      M=MIN0(NH1+NH2+1,N)
      IE=0
      IB=1
      DO 10 I=1,N
      J=MAX0(1,I-NH1)
      KIB=IB
      KIE=IE+MIN0(NH1+1,I)+MIN0(NH2,N-I)
      WRITE(6,610) I,J,(A(K),K=KIB,KIE)
      IE=IE+M
      IB=IB+M
   10 CONTINUE
      RETURN
  600 FORMAT(/10X,35A2)
  610 FORMAT(/3X,'(',I3,',',I3,')',
     * 3(2X,E16.7)/(12X,3(2X,E16.7)))
      END
```

サブルーチン PGM 及び PBM は，実行列及びバンド行列を印刷するものである.

サブルーチン PGM は，サブルーチン MGSM の使用例に記載されている.

(4) 手法概要

バンド行列 $\boldsymbol{A}$ を係数に持つ連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く.

a. 係数行列 $\boldsymbol{A}$ の LU 分解（ガウス消去法）

ガウス消去法により，係数行列 $\boldsymbol{A}$ を単位下バンド行列 $\boldsymbol{L}$ と上バンド行列 $\boldsymbol{U}$ に分解する.

$$\boldsymbol{A} = \boldsymbol{LU} \tag{4.2}$$

この計算はサブルーチン BLU1 を用いて行う.

b. 求解（前進代入，後退代入）

連立 1 次方程式 (4.1) を解くことは

$$\boldsymbol{LUx} = \boldsymbol{b} \tag{4.3}$$

を解くことと同値であり，この方程式を

$$\boldsymbol{Ly} = \boldsymbol{b} \tag{4.4}$$
$$\boldsymbol{Ux} = \boldsymbol{y} \tag{4.5}$$

として，前進代入，後退代入によって解く.

この計算はサブルーチン BLUX1 を用いて行う.

なお，参考文献として [1]，[3]，[4] 及び [8] を参照すること.

A52-11-0401 LBX1R, DLBX1R

実バンド行列の連立 1 次方程式の解の反復改良
CALL LBX1R (X, A, N, NH1, NH2, FL, FU, B, IP, VW, ICON)

(1) 機能

$n \times n$，下バンド幅 h_1，上バンド幅 h_2 の実バンド行列 $\boldsymbol{A}$ の連立 1 次方程式

$$\boldsymbol{Ax}=\boldsymbol{b} \qquad (1.1)$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は (1.2) のとおり LU 分解されていること．

$$\boldsymbol{A}=\boldsymbol{LU} \qquad (1.2)$$

ここで，$\boldsymbol{L}$ と $\boldsymbol{U}$ はそれぞれ $n \times n$ の単位下バンド行列と上バンド行列である．

$n > h_1 \geq 0$，$n > h_2 \geq 0$ であること．

(2) パラメタ

X …… **入力**．近似解ベクトル $\tilde{\boldsymbol{x}}$．
出力．反復改良された解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

A …… **入力**．係数行列 $\boldsymbol{A}$．
バンド行列用圧縮モード．
大きさ $n \cdot \min(h_1+h_2+1, n)$ の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意②参照）

NH1…… **入力**．係数行列 $\boldsymbol{A}$ の下バンド幅 h_1．

NH2…… **入力**．係数行列 $\boldsymbol{A}$ の上バンド幅 h_2．

FL…… **入力**．行列 $\boldsymbol{L}$．
図 LBX1R-1 参照．
大きさ $(n\text{-}1) \cdot h_1$ の 1 次元配列．
（使用上の注意①参照）

FU …… **入力**．行列 $\boldsymbol{U}$．
図 LBX1R-2 参照．
大きさ $n \cdot \min(h_1 + h_2 + 1, n)$ の 1 次元配列．
（使用上の注意①参照）

B…… **入力**．定数ベクトル $\boldsymbol{b}$．
大きさ n の 1 次元配列．

IP…… **入力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．
大きさ n の 1 次元配列．
（使用上の注意①参照）

VW…… **作業領域**．大きさ n の 1 次元配列．

ICON…… **出力**．コンディションコード．
表 LBX1R-1 参照．

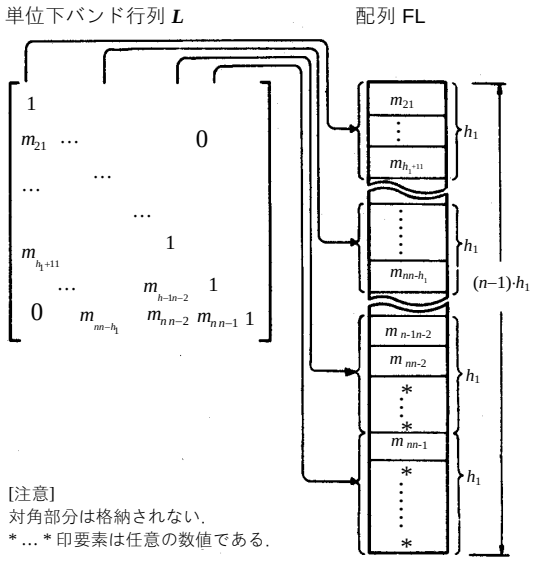


図 LBXIR-1 配列 FL における L の各要素の格納方法

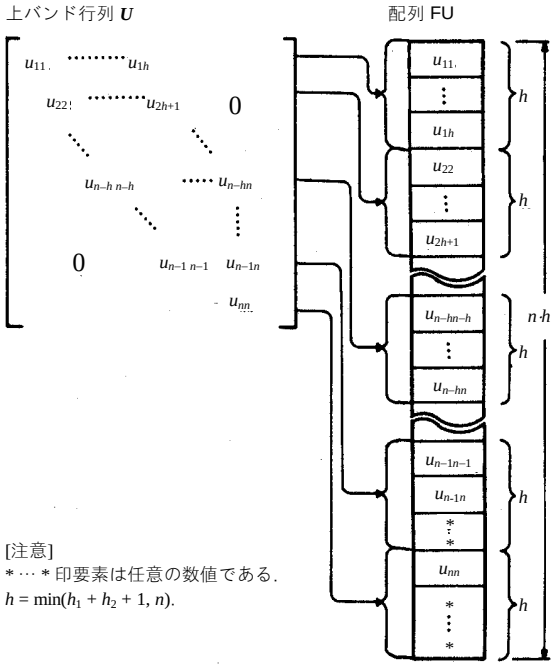


図 LBXIR-2 配列 FU における U の各要素の格納方法

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，BLUX1，MBV，MGSSL

② FORTRAN 基本関数 … ABS，MIN0

表 LBXIR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列が非正則であった.	処理を打ち切る.
25000	係数行列の条件が悪く，収束条件を満足しなかった.	処理を打ち切る.（収束条件については手法概要 b. 参照のこと）
30000	N=0，N ≤ NH1，N ≤ NH2，NH1<0 又は NH2<0 であった.	処理を打ち切る.

b. 注意

① 本サブルーチンは，サブルーチン LBX1 により求められた近似解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである.
したがって，本サブルーチンを呼び出す前にサブルーチン LBX1 を呼び出して近似解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された解を得ることができる.
この場合，パラメタ X，FL，FU，IP には直前に呼び出したサブルーチン LBX1 のパラメタ B，FL，A，IP（サブルーチン LBX1 の項目参照）の値をそのまま入力すること．ただし，本サブルーチンでは，係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LBX1 を呼ぶに先立って，それらを保存しておく必要がある.
具体的には使用例を参照されたい.

② N = –*n* として指定すれば，サブルーチン LBX1 により得られた近似解 $\tilde{\boldsymbol{x}}$ の推定精度（相対誤差）を得ることができる.
この場合，本サブルーチンでは解の反復改良は行わず，単に相対誤差を計算し，VW(1) に出力する．精度の推定については，手法概要 c. を参照されたい.

c. 使用例

n 元の連立 1 次方程式の近似解 $\tilde{\boldsymbol{x}}$ をサブルーチン LBX1 により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する.
$n \le 50$，$h_1 \le 10$，$h_2 \le 10$ の場合

```
C      **EXAMPLE**
       DIMENSION A(1050),FL(490),FU(1050),
      *  X(50),B(50),VW(50),IP(50)
       CHARACTER*4 NT1(6),NT2(4),NT3(4)
       DATA NT1/'CO  ','EF  ','FI  ',
      *         'CI  ','EN  ','T   '/,
      *     NT2/'CO  ','NS  ','TA  ',
      *         'NT  '/,
      *     NT3/'SO  ','LU  ','TI  ',
      *         'ON  '/
       READ(5,500) N,NH1,NH2
       WRITE(6,600) N,NH1,NH2
       NT0=MIN0(NH1+NH2+1,N)
       NT=N*NT0
       READ(5,510) (A(I),I=1,NT)
       CALL PBM(NT1,6,A,N,NH1,NH2)
       READ(5,510) (B(I),I=1,N)
       CALL PGM(NT2,4,B,N,N,1)
       DO 10 I=1,N
    10 X(I)=B(I)
       DO 20 I=1,NT
    20 FU(I)=A(I)
       CALL LBX1(FU,N,NH1,NH2,X,0.0,1,IS,
      *          FL,VW,IP,ICON)
       WRITE(6,610) ICON
       IF(ICON.GE.20000) STOP
       CALL LBX1R(X,A,N,NH1,NH2,FL,FU,B,IP,
      *           VW,ICON)
       WRITE(6,620) ICON
       IF(ICON.GE.20000) STOP
       CALL PGM(NT3,4,X,N,N,1)
       DET=IS
       IC=1
       DO 30 I=1,N
       DET=DET*FU(IC)
    30 IC=IC+NT0
       WRITE(6,630) DET
       STOP
   500 FORMAT(3I4)
   510 FORMAT(4E15.7)
   600 FORMAT('1'/
      * 6X,'LINEAR EQUATIONS AX=B'/
      * 6X,'ORDER=',I4/
      * 6X,'SUB-DIAGONAL LINES=',I4/
      * 6X,'SUPER-DIAGONAL LINES=',I4)
   610 FORMAT(' ',5X,'ICON OF LBX1=',I4)
   620 FORMAT(' ',5X,'ICON OF LBX1R=',I6)
   630 FORMAT(' ',5X,'DETERMINANT=',E15.7)
       END
```

サブルーチン PBM 及び PGM は，バンド行列及び実行列を印刷するものである．プログラムは，サブルーチン LBX1 及び MGSM の各使用例に記載されている.

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える.

a. 反復改良の考え方

反復改良とは，(4.1) の解 $\boldsymbol{x}$ の s 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)} = \tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$

$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{4.3}$$

$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{4.4}$$

$$s = 1,2,\ldots$$

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$ (s = 1, 2, ⋯) を逐次求めていくことである.
なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる.
ただし，係数行列 $\boldsymbol{A}$ の条件が非常に悪い場合には改良されないことがある.
（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順
サブルーチン LBX1 で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする.
本サブルーチンでは，以下を繰り返す.

- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MBV を用いて行う.
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはサブルーチン BLUX1 を用いて行う.
- 修正された近似解 $\boldsymbol{x}^{(s+1)}$ を (4.4) から求める.

収束判定法は，以下のとおりである.
ここで丸め誤差の単位を u としたとき，s 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2\cdot u \qquad (4.5)$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $\boldsymbol{x}^{(s+1)}$ を解として採用する.
ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s-1)}\right\|_\infty} > \frac{1}{2}\cdot\frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON = 25000 として処理を打ち切る.

c. 近似解の精度推定
近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)}$ ($=\boldsymbol{x}^{(1)}-\boldsymbol{x}$) とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(s)}\right\|_\infty$ である．いま反復法が収束するならば，$\boldsymbol{e}^{(1)}$ と $\boldsymbol{d}^{(1)}$ はほぼ等しいものと考えられるから，近似解の相対誤差は $\left\|\boldsymbol{d}^{(1)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ によって推定できる．（3.4 (2) f. 近似解の精度推定 参照）

なお，詳細については，文献 [1]，[3] 及び [5] を参照すること.

A22-15-0101 LCX,DLCX

複素行列の連立 1 次方程式（クラウト法）
CALL LCX (ZA, K, N, ZB, EPSZ, ISW, IS, ZVW, IP,ICON)

(1) 機能

複素係数連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (1.1)$$

をクラウト法で解く．ただし，$\boldsymbol{A}$ は $n \times n$ の正則な複素行列，$\boldsymbol{b}$ は n 次元の複素定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．$n \geq 1$ であること．

(2) パラメタ

ZA …… **入力**．係数行列 $\boldsymbol{A}$.
演算後，内容は保存されない．
ZA (K, N) なる複素数型 2 次元配列.

K …… **入力**．配列 ZA の整合寸法（≥ N）.

N …… **入力**．係数行列 $\boldsymbol{A}$ の次数 n.

ZB …… **入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の複素数型 1 次元配列.

EPSZ …… **入力**．ピボットの相対零判定値（≥ 0.0）.
0.0 のときは標準値が採用される．（使用上の注意①参照）

ISW …… **入力**．制御情報.
同一の係数行列を持つ l（≥ 1）組の方程式を解くとき，次のとおり指定する．
ISW = 1 のとき，1 組目の方程式を解く.
ISW = 2 のとき，2 組目以降の方程式を解く.
ただし，このとき ZB の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．（使用上の注意②参照）

IS …… **出力**．行列 $\boldsymbol{A}$ の行列式を求めるための情報．演算後の配列 ZA の n 個の対角要素と IS の値を掛け合わせると，行列式が得られる.

ZVW …… **作業領域**．大きさ n の複素数型 1 次元配列.

IP …… **作業領域**．大きさ n の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 LCX-1 参照.

表 LCX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列のある行の要素がすべて零であったか又はピボットが相対的に零となった．係数行列は非正則の可能性が強い.	処理を打ち切る.
30000	K<N，N<1，EPSZ<0.0 又は ISW ≠ 1,2 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，CLU，CLUX，CSUM，MGSSL

② FORTRAN 基本関数… REAL，AIMAG，ABS

b. 注意

① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，クラウト法による LU 分解の過程でピボットの実部と虚部の値が，同時に 10 進 s 桁以上の桁落ちを生じた場合にそのピボットを相対的に零と見なし，ICON = 20000 として処理を打ち切る.
EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である.
なお，ピボットが小さくなっても計算を続行させるには，EPSZ へ極小の値を与えればよいが，その結果は保証されない.

② 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列 $\boldsymbol{A}$ の LU 分解過程を省略するので計算時間が少なくなる．なお，この場合 IS の値は，ISW = 1 のときの値が保証される.

c. 使用例

同一の係数行列を持つ l 組の n 元の複素係数連立 1 次方程式を解く．$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION ZA(100,100),ZB(100),
     *          ZVW(100),IP(100)
      COMPLEX ZA,ZB,ZVW,ZDET
      READ(5,500) N,L
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,ZA(I,J),J=1,N),
     *             I=1,N)
      M=1
      ISW=1
      EPSZ=1.0E-6
   10 READ(5,510) (ZB(I),I=1,N)
      WRITE(6,610) (I,ZB(I),I=1,N)
      CALL LCX(ZA,100,N,ZB,EPSZ,ISW,IS,
     *         ZVW,IP,ICON)
```

```
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,630) (I,ZB(I),I=1,N)
      IF(L.EQ.M) GOTO 20
      M=M+1
      ISW=2
      GOTO 10
   20 ZDET=CMPLX(FLOAT(IS),0.0)
      DO 30 I=1,N
      ZDET=ZDET*ZA(I,I)
   30 CONTINUE
      WRITE(6,640) ZDET
      STOP
  500 FORMAT(2I5)
  510 FORMAT(5(2F8.3))
  600 FORMAT('1',10X,
     * '** COMPLEX MATRIX **'/
     * 12X,'ORDER=',I5/
     * (3X,2('(',I3,',',I3,')',2E15.8,
     * 2X)))
  610 FORMAT(///10X,'COMPLEX CONSTANT ',
     * 'VECTOR',/(5X,3('(',I3,')',2E15.8,
     * 2X)))
  620 FORMAT('0',10X,'CONDITION CODE=',I5)
  630 FORMAT('0',10X,'SOLUTION VECTOR'/
     * (5X,3('(',I3,')',2E15.8,2X)))
  640 FORMAT(///10X,'DETERMINANT OF ',
     * 'MATRIX',2E15.8)
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く.

a. 係数行列 $\boldsymbol{A}$ の LU 分解（クラウト法）
クラウト法により，係数行列 $\boldsymbol{A}$ を下三角行列 $\boldsymbol{L}$ と単位上三角行列 $\boldsymbol{U}$ の積に分解する．丸め誤差を少なくするため分解過程において部分ピボッティングを行う.

$$\boldsymbol{PA} = \boldsymbol{LU} \tag{4.2}$$

ここで，$\boldsymbol{P}$ はピボッティングによる行の入れ換えを行う置換行列である．この計算はサブルーチン CLU を用いて行う.

b. 求解（前進代入，後退代入）
連立 1 次方程式 (4.1) を解くことは

$$\boldsymbol{LUx} = \boldsymbol{Pb} \tag{4.3}$$

を解くことと同値であり，この方程式は

$$\boldsymbol{Ly} = \boldsymbol{Pb} \tag{4.4}$$
$$\boldsymbol{Ux} = \boldsymbol{y} \tag{4.5}$$

として，前進代入，後退代入によって解く．この計算はサブルーチン CLUX を用いて行う.

なお，詳細については，参考文献 [1]，[3] 及び [4] を参照すること.

A22-15-0401 LCXR,DLCXR

複素行列の連立 1 次方程式の解の反復改良
CALL LCXR(ZX, ZA, K, N, ZFA, ZB, IP, ZVW, ICON)

(1) 機能

$n \times n$ の複素行列 $\boldsymbol{A}$ の連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (1.1)$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は n 次元の複素定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は (1.2) のとおり LU 分解されていること．

$$\boldsymbol{PA} = \boldsymbol{LU} \qquad (1.2)$$

ここで，$\boldsymbol{L}$ と $\boldsymbol{U}$ はそれぞれ $n \times n$ の下三角行列と単位上三角行列，$\boldsymbol{P}$ は部分ピボッティングによる行の入換えを行う置換行列である．$n \geq 1$ であること．

(2) パラメタ

ZX …… **入力**．近似解ベクトル $\tilde{\boldsymbol{x}}$.
出力．反復改良された解ベクトル $\boldsymbol{x}$.
大きさ n の複素数型 1 次元配列.

ZA …… **入力**．係数行列 $\boldsymbol{A}$.
ZA (K,N) なる複素数型 2 次元配列.

K …… **入力**．配列 ZA，ZFA の整合寸法（≥ N）.

N …… **入力**．係数行列 $\boldsymbol{A}$ の次数 n.
（使用上の注意②参照）

ZFA …… **入力**．行列 $\boldsymbol{L}$ と行列 $\boldsymbol{U}$.
図 LCXR-1 参照.
ZFA (K,N) なる複素数型 2 次元配列.
（使用上の注意①参照）

ZB …… **入力**．定数ベクトル $\boldsymbol{b}$.
大きさ n の複素数型 1 次元配列.

IP …… **入力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさ n の 1 次元配列．（使用上の注意①参照）

ZVW …… **作業領域**．大きさ n の複素数型 1 次元配列.

ICON …… **出力**．コンディションコード.
表 LCXR-1 参照.

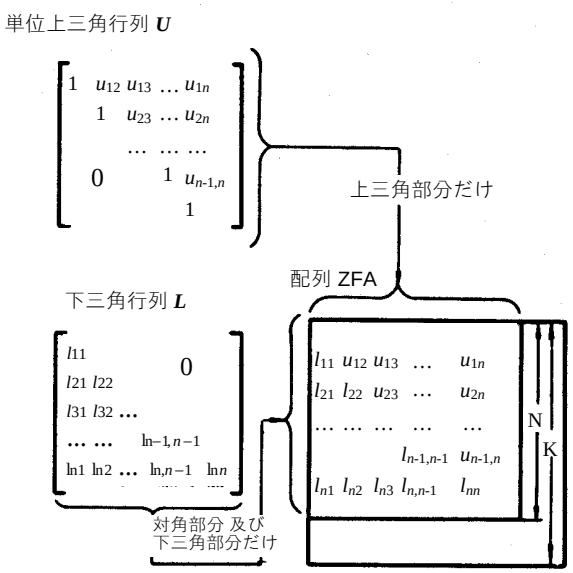


図 LCXR-1 配列 ZFA における $\boldsymbol{L}$ 及び $\boldsymbol{U}$ の各要素の格納方法

表 LCXR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	係数行列が非正則であった.	処理を打ち切る.
25000	係数行列の条件が悪く，収束条件を満足しなかった.	処理を打ち切る.（収束条件については手法概要 b. 参照のこと）
30000	K<N 又は N<1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，CLUX，CSUM，MCV，MGSSL

② FORTRAN 基本関数… REAL，AIMAG，ABS，IABS

b. 注意

① 本サブルーチンは，サブルーチン LCX により求められた近似解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである.
したがって，本サブルーチンを呼び出す前にサブルーチン LCX を呼び出して近似解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された解を得ることができる.
この場合，パラメタ ZX，ZFA，IP には直前に呼び出したサブルーチン LCX の結果をそのまま入力すること．ただし，本サブルーチンでは係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LCX を呼ぶに先立ってそれらを保存しておく必要がある.
具体的には使用例を参照されたい.

② N = - *n* と指定すれば，サブルーチン LCX により得られた近似解 $\tilde{\boldsymbol{x}}$ の推定精度（相対誤差）を得ることができる．
この場合，本サブルーチンでは解の反復改良は行わず，単に相対誤差を計算し，ZVW(1) に出力する．
精度の推定については，手法概要 c. を参照されたい．

c. 使用例
n 元の連立 1 次方程式の近似解 $\tilde{\boldsymbol{x}}$ をサブルーチン LCX により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する．$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION ZA(100,100),ZFA(100,100),
     *  ZX(100),ZB(100),ZVW(100),IP(100)
      COMPLEX ZA,ZFA,ZX,ZB,ZVW,ZDET
      READ(5,500) N
      IF(N.LE.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,ZA(I,J),J=1,N),
     *             I=1,N)
      READ(5,510) (ZB(I),I=1,N)
      WRITE(6,610) (I,ZB(I),I=1,N)
      DO 10 I=1,N
      ZX(I)=ZB(I)
      DO 10 J=1,N
      ZFA(I,J)=ZA(I,J)
   10 CONTINUE
      K=100
      CALL LCX(ZFA,K,N,ZX,0.0,1,IS,ZVW,IP,
     *         ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      CALL LCXR(ZX,ZA,K,N,ZFA,ZB,IP,ZVW,
     *          ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,640) (I,ZX(I),I=1,N)
      ZDET=IS
      DO 20 I=1,N
      ZDET=ZDET*ZFA(I,I)
   20 CONTINUE
      WRITE(6,650) ZDET
      STOP
  500 FORMAT(I3)
  510 FORMAT(4E15.7)
  600 FORMAT('1',5X,'COEFFICIENT MATRIX'/
     * 6X,'ORDER=',I5/(6X,2('(',I3,',',I3,
     * ')',2E15.8,1X)))
  610 FORMAT(' ',5X,'CONSTANT VECTOR'/
     * (6X,3('(',I3,')',2E15.8)))
  620 FORMAT(' ',5X,'ICON OF LCX=',I5)
  630 FORMAT(' ',5X,'ICON OF LCXR=',I5)
  640 FORMAT(' ',5X,'IMPROVED SOLUTION'/
     * (6X,3('(',I3,')',2E15.8,1X)))
  650 FORMAT(' ',5X,'DETERMINANT=',2E15.8)
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える．

a. 反復改良の考え方
反復改良とは，(4.1) の解 $\boldsymbol{x}$ の s 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)}= \tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)}=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$
$$\boldsymbol{A}\boldsymbol{d}^{(s)}=\boldsymbol{r}^{(s)} \tag{4.3}$$
$$\boldsymbol{x}^{(s+1)}=\boldsymbol{x}^{(s)}+\boldsymbol{d}^{(s)} \tag{4.4}$$
$s=1,2,\cdots$

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$（s = 1，2，…）を逐次求めていくことである．
なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる．
ただし，係数行列 $\boldsymbol{A}$ の条件が非常に悪い場合には改良されないことがある．
（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順
サブルーチン LCX で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする．
本サブルーチンでは，以下を繰り返す．

- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MCV を用いて行う．
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはサブルーチン CLUX を用いて行う．
- 修正された近似解 $\boldsymbol{x}^{(s+1)}$ を (4.4) から求める．

収束判定法は，以下のとおりである．
ここで丸め誤差の単位を u としたとき，s 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2u \tag{4.5}$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $\boldsymbol{x}^{(s+1)}$ を解として採用する．

ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s+1)}\right\|_\infty} > \frac{1}{2}\cdot\frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON=25000 として処理を打ち切る．

c. 近似解の精度推定
近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)}$ $(=\boldsymbol{x}^{(1)}-\boldsymbol{x})$ とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ である．いま反復法が収束するならば，$\boldsymbol{e}^{(1)}$ と $\boldsymbol{d}^{(1)}$ はほぼ等しいものと考えられるから，近似解の相対誤差は $\left\|\boldsymbol{d}^{(1)}\right\|_\infty \Big/ \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ によって推定できる．（3.4 (2) f. 近似解の精度推定 参照）

なお，詳細については，文献 [1]，[3] 及び [5] を参照すること．

A22-51-0702 LDIV, DLDIV

LDL^T 分解された正値対称行列の逆行列
CALL LDIV (FA, N, ICON)

(1) 機能

LDL^T 分解された $n \times n$ の正値対称行列 $\boldsymbol{A}$ の逆行列 $\boldsymbol{A}^{-1}$ を求める.

$$\boldsymbol{A}^{-1} = \left(\boldsymbol{L}^T\right)^{-1} \boldsymbol{D}^{-1} \boldsymbol{L}^{-1} \qquad (1.1)$$

ただし，$\boldsymbol{L}$，$\boldsymbol{D}$ はそれぞれ $n \times n$ の単位下三角行列と対角行列である.

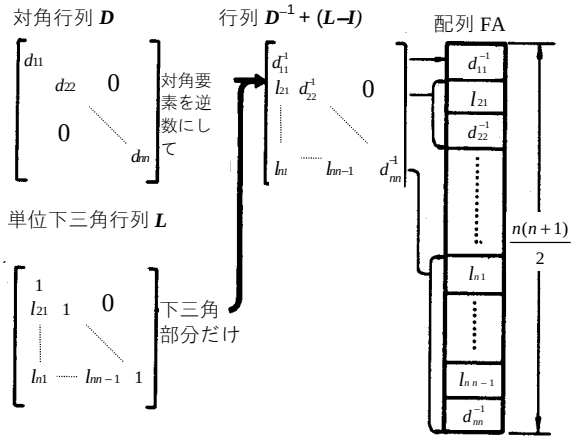
$n \geq 1$ であること.

(2) パラメタ

FA ····· **入力**. 行列 $\boldsymbol{L}$ と行列 $\boldsymbol{D}^{-1}$.
図 LDIV-1 参照.
出力. 逆行列 $\boldsymbol{A}^{-1}$.
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

N ····· **入力**. 行列 $\boldsymbol{L}$，$\boldsymbol{D}$ の次数 n.

ICON····· **出力**. コンディションコード.
表 LDIV-1 参照.



[注意] 行列 $\boldsymbol{D}^{-1} + (\boldsymbol{L}-\boldsymbol{I})$ の対角部分及び下三角部分を，対称行列用圧縮モードで 1 次元配列 FA に格納する.

図 LDIV-1 行列 $\boldsymbol{L}$ 及び $\boldsymbol{D}$ の格納方法

表 LDIV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	行列が正値でなかった.	処理は続行する.
30000	N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II··· MGSSL
② FORTRAN 基本関数··· なし

b. 注意
① 本サブルーチンは，LDL^T 分解された正値対称行列を入力して，分解された元の正値対称行列の逆行列を計算する．LDL^T 分解はサブルーチン SLDL により行い，続けて本サブルーチンを呼び出すことにより逆行列を求めることができる（使用例参照）．なお，本サブルーチンでは行列 $\boldsymbol{D}$ を，$\boldsymbol{D}^{-1}$ としてその対角要素を与える必要がある（サブルーチン SLDL では，$\boldsymbol{D}^{-1}$ が出力される）.
② 連立 1 次方程式を解く場合には，サブルーチン LSX を用いること．逆行列を求めて解くことは，LSX を用いるときよりも演算回数が多くなるので避けた方がよい．本サブルーチンは逆行列が必要なときだけ使用すること.

c. 使用例
$n \times n$ の正値対称行列を入力し，その逆行列を求める．$n \leq 100$ の場合.

```
C       **EXAMPLE**
        DIMENSION A(5050)
   10   READ(5,500) N
        IF(N.EQ.0) STOP
        NT=N*(N+1)/2
        READ(5,510) (A(I),I=1,NT)
        WRITE(6,620)
        L=0
        LS=1
        DO 20 I=1,N
        L=L+1
        WRITE(6,600) I,(A(J),J=LS,L)
   20   LS=L+1
        CALL SLDL(A,N,0.0,ICON)
        IF(ICON.GE.20000) STOP
        CALL LDIV(A,N,ICON)
        WRITE(6,630)
        L=0
        LS=1
        DO 30 I=1,N
        L=L+1
        WRITE(6,600) I,(A(J),J=LS,L)
   30   LS=L+1
        WRITE(6,610) ICON
        GOTO 10
  500   FORMAT(I5)
  510   FORMAT(5E10.2)
  600   FORMAT(' ',I5/(10X,5E16.8))
  610   FORMAT(/10X,'ICON=',I5)
  620   FORMAT(/10X,'INPUT MATRIX')
  630   FORMAT(/10X,'INVERSE MATRIX')
        END
```

(4) 手法概要

$n \times n$ の正値対称行列 $\boldsymbol{A}$ の LDL$^{\mathrm{T}}$ 分解された行列 $\boldsymbol{L}$，$\boldsymbol{D}$ が与えられたとき，$\boldsymbol{A}$ の逆行列 $\boldsymbol{A}^{-1}$ を求めることを考える．ところで，

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{T} \tag{4.1}$$

であるから

$$\boldsymbol{A}^{-1} = \left(\boldsymbol{L}^{T}\right)^{-1}\boldsymbol{D}^{-1}\boldsymbol{L}^{-1} = \left(\boldsymbol{L}^{-1}\right)^{T}\boldsymbol{D}^{-1}\boldsymbol{L}^{-1} \tag{4.2}$$

となる．そこで，$\boldsymbol{L}$ と $\boldsymbol{D}$ の逆行列を求めて（本サブルーチンでは $\boldsymbol{D}^{-1}$ を入力するので，$\boldsymbol{L}$ の逆行列だけ求める），$(\boldsymbol{L}^{-1})^{T}$ に対して，$\boldsymbol{D}^{-1}$ 及び $\boldsymbol{L}^{-1}$ を逐次右側から掛けることにより $\boldsymbol{A}^{-1}$ を求める．

ここで，以降の説明のために，$\boldsymbol{L}$ 及び $\boldsymbol{D}^{-1}$ を (4.3) で表す．

$$\boldsymbol{L} = \left(\boldsymbol{l}_{ij}\right),\ \boldsymbol{D}^{-1} = \mathrm{diag}\left(d_{i}^{-1}\right) \tag{4.3}$$

a. $\boldsymbol{L}^{-1}$ の計算

単位下三角行列 $\boldsymbol{L}$ の逆行列 $\boldsymbol{L}^{-1}$ も単位下三角行列となるので，$\boldsymbol{L}^{-1}$ を (4.4) で表すと，

$$L^{-1} = \left(\tilde{\boldsymbol{l}}_{ij}\right) \tag{4.4}$$

$\boldsymbol{L}\boldsymbol{L}^{-1}=\boldsymbol{I}$ より (4.5) が成り立つ．

$$\sum_{k=1}^{n} l_{ik}\tilde{l}_{kj} = \delta_{ij}\ ,\quad \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{4.5}$$

ところで，l_{ii}=1 であるから (4.5) は $\tilde{l}_{ij}$ に関して (4.6) のように変形できる．

$$\tilde{l}_{ij} = \delta_{ij} - \sum_{k=j}^{i-1} l_{ik}\tilde{l}_{kj} \tag{4.6}$$

そこで，$\tilde{l}_{ii} = 1$，$\tilde{l}_{jj} = 1$ を考慮することにより，$\boldsymbol{L}^{-1}$ の第 i 行目（i = 2，…，n）の要素 $\tilde{l}_{ij}$ を (4.7) により逐次求める．

$$\tilde{l}_{ij} = -l_{ij} - \sum_{k=j+1}^{i-1} l_{ik}\tilde{l}_{kj}\ ,\ j = 1,\ldots,i-1 \tag{4.7}$$

b. $(\boldsymbol{L}^{-1})^{T}\boldsymbol{D}^{-1}\boldsymbol{L}^{-1}$ の計算

逆行列 $\boldsymbol{A}^{-1}$ を (4.8) で表すと，

$$\boldsymbol{A}^{-1} = \left(\tilde{a}_{ij}\right) \tag{4.8}$$

$\boldsymbol{A}^{-1}=(\boldsymbol{L}^{-1})^{T}\boldsymbol{D}^{-1}\boldsymbol{L}^{-1}$ より (4.9) が成り立つ．

$$\tilde{a}_{ij} = \sum_{k=1}^{n} \tilde{l}_{ki} d_{k}^{-1} \tilde{l}_{kj} \tag{4.9}$$

そこで，$\tilde{l}_{ii} = 1$ であることを考慮して，$\boldsymbol{A}^{-1}$ の第 i 行目（i = 1，...，n）の要素 $\tilde{a}_{ij}$ を (4.10) により逐次求める．

$$\tilde{a}_{ij} = d_{i}^{-1}\tilde{l}_{ij} + \sum_{k=i+1}^{n} \tilde{l}_{ki} d_{k}^{-1} \tilde{l}_{kj} \qquad j = 1,\ldots,i \tag{4.10}$$

なお，(4.7)，(4.10) で積和計算部分は精度を上げて行うことにより，丸め誤差の影響を少なくしている．

また，詳細については，参考文献 [2] を参照すること．

A22-51-0302 LDLX, DLDLX

LDL^T 分解された正値対称行列の連立 1 次方程式
CALL LDLX (B, FA, N, ICON)

(1) 機能

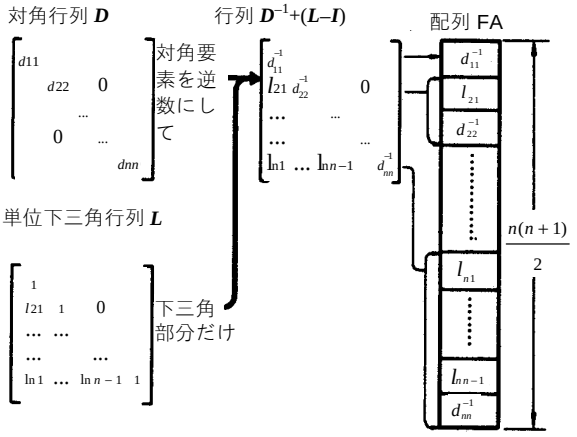
LDL^T 分解された正値対称な係数行列を持つ連立 1 次方程式

$$\boldsymbol{LDL^T x = b} \qquad (1.1)$$

を解く．ただし，$\boldsymbol{L}$，$\boldsymbol{D}$ はそれぞれ $n \times n$ の単位下三角行列と対角行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．$n \geq 1$ であること．

(2) パラメタ

B…… **入力**．定数ベクトル $\boldsymbol{b}$.
　　出力．解ベクトル $\boldsymbol{x}$.
　　大きさ n の 1 次元配列.
FA …… **入力**．行列 $\boldsymbol{L}$ と $\boldsymbol{D}^{-1}$.
　　図 LDLX-1 参照.
　　対称行列用圧縮モード.
　　大きさ $n(n+1)/2$ の 1 次元配列.
N …… **入力**．行列 $\boldsymbol{L}$，$\boldsymbol{D}$ の次数 n.
ICON…… **出力**．コンディションコード.
　　表 LDLX-1 参照.



[注意]　行列 $\boldsymbol{D}^{-1}+(\boldsymbol{L}-\boldsymbol{I})$ の対角部分及び下三角部分を，対称行列用圧縮モードで 1 次元配列 FA に格納する.

図 LDLX-1 行列 $\boldsymbol{L}$ 及び $\boldsymbol{D}$ の格納方法

表 LDLX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	係数行列が正値でなかった.	処理は続行する.
30000	N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a.　使用する副プログラム
①　SSL II…　MGSSL
②　FORTRAN 基本関数…　なし.

b.　注意
①　本サブルーチンは，行列 $\boldsymbol{D}$ については $\boldsymbol{D}^{-1}$ としてその対角要素を入力する必要があるので注意すること．本サブルーチンを，サブルーチン SLDL に続けて呼び出すことにより，連立 1 次方程式を解くことができる．しかし，通常は，サブルーチン LSX を呼び出せば，一度に解が求められる．なお，サブルーチン SLDL では $\boldsymbol{D}^{-1}$ が出力される.
②　行列 $\boldsymbol{L}$ が正値対称バンド行列の場合，バンド部分の外側にある要素にかかわる計算を省略するサブルーチン BDLX の方が速く処理できる.

c.　使用例
$n \times n$ の係数行列をサブルーチン SLDL で LDL^T 分解し，連立 1 次方程式を解く．n ≤ 100 の場合.

```
C      **EXAMPLE**
       DIMENSION A(5050),B(100)
   10  READ(5,500) N
       IF(N.EQ.0) STOP
       NTOT=N*(N+1)/2
       READ(5,510) (A(I),I=1,NTOT)
       WRITE(6,640)
       L=0
       LS=1
       DO 20 I=1,N
       L=L+1
       WRITE(6,600) I,(A(J),J=LS,L)
   20  LS=L+1
       CALL SLDL(A,N,1.0E-6,ICON)
       WRITE(6,610) ICON
       IF(ICON.GE.20000) STOP
       READ(5,510) (B(I),I=1,N)
       CALL LDLX(B,A,N,ICON)
       WRITE(6,610) ICON
       DET=A(1)
       L=1
       DO 30 I=2,N
       L=L+I
   30  DET=DET*A(L)
       DET=1.0/DET
       WRITE(6,620) (B(I),I=1,N)
       WRITE(6,630) DET
       GOTO 10
  500  FORMAT(I5)
  510  FORMAT(5E10.2)
  600  FORMAT(' ',I5/(10X,5E16.8))
  610  FORMAT(/10X,'ICON=',I5)
  620  FORMAT(/10X,'SOLUTION VECTOR'//
      * (10X,5E16.8))
  630  FORMAT(/10X,
      * 'DETERMINANT OF COEFFICIENT ',
      * 'MATRIX=',E16.8)
  640  FORMAT(/10X,'INPUT MATRIX')
       END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{LDL}^T\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

を解くことは，次の二つの方程式

$$\boldsymbol{Ly} = \boldsymbol{b} \tag{4.2}$$

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{D}^{-1}\boldsymbol{y} \tag{4.3}$$

を解くことに帰着される．

a. $\boldsymbol{Ly}=\boldsymbol{b}$ を解く（前進代入）

$$y_i = b_i - \sum_{k=1}^{i-1} l_{ik} y_k, \quad i = 1,...,n \tag{4.4}$$

なる式で逐次求める．ただし，$\boldsymbol{L}=(l_{ij})$，$\boldsymbol{y}^T=(y_1,...,y_n)$, $\boldsymbol{b}^T=(b_1,...,b_n)$ である．

b. $\boldsymbol{L}^T\boldsymbol{x} =\boldsymbol{D}^{-1}\boldsymbol{y}$ を解く（後退代入）

$$x_i = y_i d_i^{-1} - \sum_{k=i+1}^{n} l_{ki} x_k, \quad i = n,...,1 \tag{4.5}$$

なる式で逐次求める．
ただし，$\boldsymbol{D}^{-1} = \mathrm{diag}(d_i^{-1})$，$\boldsymbol{x}^T=(x_1,...,x_n)$ である．

なお，詳細については，参考文献 [2] を参照すること．

E21-20-0101 LESQ1, DLESQ1

最小二乗近似多項式
CALL LESQ1 (X, Y, N, M, W, C, VW, ICON)

(1) 機能

離散点 x_1，x_2，…，x_n に対して，観測値 y_i 及び重み関数値 $w(x_i)$ i=1，2，…，n が与えられたとき，これに対する最小二乗近似多項式を求める．

すなわち，m 次の多項式 $\bar{y}_m(x)$ を，

$$\bar{y}_m(x)=c_0+c_1x+...+c_mx^m \tag{1.1}$$

とするとき，

$$\delta_m^2=\sum_{i=1}^{n}w(x_i)\{y_i-\bar{y}_m(x_i)\}^2 \tag{1.2}$$

を最小にするような $\bar{y}_m(x)$ を求める．

ここで m は $0 \le m \le$ k なる範囲で，

$$\mathrm{AIC}=n\log\delta_m^2+2m \tag{1.3}$$

なる量を最小にするように選ばれ，このときの m を最小二乗近似多項式の最良なる次数とする．

ただし，$0 \le k \le n$-1 とする．また重み関数値 $w(x_i)$ は，

$$w(x_i)\ge 0,\ i=1,2,...,n \qquad n\ge 2 \tag{1.4}$$

なる条件を満たすものであること．

(2) パラメタ

X ····· **入力**．離散点 x_i．
大きさ n の 1 次元配列．

Y ····· **入力**．観測値 y_i．
大きさ n の 1 次元配列．

N ····· **入力**．離散点の個数 n．

M ····· **入力**．求める近似多項式の次数の上限 k．
ただし，
M = -k (k > 0)
と入力すると，無条件に k 次の近似多項式を求める．
出力．得られた近似多項式の次数 m($\le k$)．
したがって，M = -k と入力したときは，出力時の M の値は常に k となる．

W ····· **入力**．重み関数値 $w(x_i)$．
大きさ n の 1 次元配列．

C ····· **出力**．得られた近似多項式の係数 c_i．
大きさ k+1 の 1 次元配列．パラメタ M の出力時の値を m ($0 \le m \le k$) とすると，c_0，c_1，...，c_m の順に格納する．
このとき，$m < k$ となった場合は，要素 C(I+1)，I=m+1，...，k には 0.0 を格納する．

VW ····· **作業領域**．大きさ 7n の 1 次元配列．

ICON ····· **出力**．コンディションコード．
表 LESQ1-1 参照．

表 LESQ1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	M = -k(k>0) と指定されたが，k 次の近似多項式が一意的に決まらなかった．	一意的に決めることができる範囲で最高次（しかし k 次よりは低い）の近似多項式を出力する．
30000	次のいずれかであった． ① $n < 2$ ② $k > n$- 1 ③ $w(x_i)$ のなかに負のものがある．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II … AMACH，MGSSL
② FORTRAN 基本関数 … IABS, DABS, DLOG, FLOAT, AMAX1

b. 注意
① 単精度用 / 倍精度用サブルーチンの使分け．
観測値の個数と性質によっては単精度用サブルーチンと倍精度用サブルーチンとの間で，出力する近似多項式の次数が異なることがある．したがって，利用者は，特に観測値の個数が多い場合はなるべく倍精度用を使う方がよい．
② 重み関数値 $w(x_i)$ の与え方
観測値 $\{y_i\}$ がオーダ的にそろっている場合は，$w(x_i)$=1.0，i=1，2，...，n でよいが，オーダ的にまちまちである場合は $w(x_i)=1/y_i^2$（y_i=0 のときは $w(x_i)$=1.0）と与えるのがよい．
③ 離散点の個数 n は次数の上限 k に比べてなるべく多くとるべきである．理論的には $10k \le n$ を満たすのがよいとされている．

c. 使用例
離散点 x_i と観測値 y_i，i=1，2，...，n を入力し，10 次以下の範囲の最良なる最小二乗近似多項式を求める．ここでは，$10 \le n \le 50$，$w(x_i)$ = 1（i = 1，2，...，n）の場合とする．

```
C     **EXAMPLE**
      DIMENSION X(50),Y(50),W(50),
     *          C(6),VW(350)
      READ(5,500) N
      READ(5,510) (X(I),Y(I),I=1,N)
      WRITE(6,600) (I,X(I),Y(I),I=1,N)
      DO 10 I=1,N
   10 W(I)=1.0
      M=5
      MI=M
      CALL LESQ1(X,Y,N,M,W,C,VW,ICON)
      WRITE(6,610) MI,ICON
      IF(ICON.EQ.30000) STOP
      M1=M+1
      WRITE(6,620) M,(I,C(I),I=1,M1)
      STOP
  500 FORMAT(I5)
  510 FORMAT(2F10.0)
  600 FORMAT('1'//10X,'INPUT DATA'//
     * 20X,'NO.',10X,'X',17X,'Y'//
     * (20X,I3,3X,E15.7,3X,E15.7))
  610 FORMAT(10X,
     * 'THE UPPER LIMIT OF THE DEGREE',
     * 5X,I5/10X,'ICON=',I5)
  620 FORMAT(//10X,
     * 'THE RESULTANT DEGREE',14X,I5/
     * 10X,'THE RESULTANT COEFFICIENTS'/
     * (20X,'C(',I2,')=',E15.7))
      END
```

(4) 手法概要

a. 最小二乗近似多項式

離散点 x_i，i=1，2，...，n に対して観測値 y_i，i=1，2，...，n が与えられていて，それらの観測値にはある程度の観測誤差が含まれているとする．

このとき，これらの観測値に対する，m 次の最小二乗近似多項式とは，(4.1) を最小にするような m 次の多項式 $\bar{y}_m(x)$ のことをいう．

$$\delta_m^2 = \sum_{i=1}^{n} w(x_i)\{y_i - \bar{y}_m(x_i)\}^2 \tag{4.1}$$

ここで $w(x_i)$，i=1，2，...，n は重み関数値である．

$\bar{y}_m(x)$ を次のように表す．

$$\bar{y}_m(x) = \sum_{j=0}^{m} b_j^{(m)} P_j(x) \tag{4.2}$$

ここで $P_j(x)$ は j 次のある多項式である（後述）．(4.2) の $b_j^{(m)}$ を決定するには，(4.2) を (4.1) へ代入し，そのあと，右辺を $b_j^{(m)}$ で偏微分する．これを j = 0，…，m に対して行い，それらをすべて 0 と置く．このようにしてできる連立一次方程式は，

$$\left.\begin{aligned} &\sum_{j=0}^{m} d_{jk} b_j^{(m)} = \omega_k \quad k = 0,1,...,m \\ &\text{ここで,} \\ &d_{jk} = \sum_{i=1}^{n} w(x_i) P_j(x_i) P_k(x_i) \\ &\omega_k = \sum_{i=1}^{n} w(x_i) y_i P_k(x_i) \end{aligned}\right\} \tag{4.3}$$

となり，これを正規方程式という．$b_j^{(m)}$ はこの正規方程式を解けば得られる．

ところで，多項式列 $\{P_j(x)\}$ が，(4.3) の d_{jk} を，

$$d_{jk} \begin{cases} = 0, & j \neq k \\ \neq 0, & j = k \end{cases} \tag{4.4}$$

とするものであるなら，正規方程式の係数行列は，対角要素を除いて 0 となるので，$b_j^{(m)}$ は，

$$\begin{aligned} &b_j^{(m)} = b_j = \omega_j / \gamma_j \\ &\text{ここで、} \gamma_j = d_{jj} = \sum_{i=1}^{n} w(x_i)\{P_j(x_i)\}^2 \end{aligned} \tag{4.5}$$

より求まる．

(4.4) を満たす $\{P_j(x)\}$ は，次の漸化式から作り出すことができる．

$$\begin{aligned} &P_{j+1}(x) = (x - \alpha_{j+1}) P_j(x) - \beta_j P_{j-1}(x) \\ &\qquad\qquad j = 0,1,... \\ &P_0(x) = 1, P_{-1}(x) = 0 \end{aligned}$$

$$\alpha_{j+1} = \sum_{i=1}^{n} w(x_i) x_i \{P_j(x_i)\}^2 / \gamma_j \tag{4.6}$$

$$j = 0,1,...$$

$$\beta_j = \begin{cases} 0, j = 0 \\ \gamma_j / \gamma_{j-1}, j = 1,2,... \end{cases}$$

b. 最良なる次数の選択

最小二乗近似多項式を求める場合，その次数の選択は重要な問題である．本サブルーチンは以下のようにして自動的に最良なる次数を選択している．

δ_m^2 を，

$$\delta_m^2 = \sum_{i=1}^{n} w(x_i)\left\{ y_i - \sum_{j=0}^{m} b_j P_j(x_i) \right\}^2 \qquad (4.7)$$

として，AIC という量を，

$$\mathrm{AIC} = n \log \delta_m^2 + 2m \qquad (4.8)$$

と定義する．この量は次数 m の良し悪しを計る尺度であり，一般に AIC が小さくするような m ほど良い．

そこで本サブルーチンは，$0 \le m \le k$（k は利用者が与える）なる範囲で AIC を最小にする m を最良なる次数と定め，その次数の最小二乗近似多項式を出力する．出力のし方は，$\{b_j\}$，$\{P_j(x)\}$（j=0，1，...，m）を使って，

$$\bar{y}_m(x) = \sum_{j=0}^{m} b_j P_j(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_m x^m \qquad (4.9)$$

のように多項式の標準形に直して，c_0，c_1，c_2，...，c_m を出力する．

なお，詳細については，参考文献 [46] pp.228～250 を参照すること．

D11-30-0101 LMINF, DLMINF

1 変数関数の極小化 （微係数不要，2 次補間法）
CALL LMINF (A, B, FUN, EPSR, MAX, F, ICON)

(1) 機能

1 変数の実関数 $f(x)$ が与えられたとき，区間 $[a,b]$ で $f(x)$ の極小点 x^* とその関数値 $f(x^*)$ を求める．

ただし，$f(x)$ は 2 階までの連続な微係数を持つものと仮定する．

(2) パラメタ

A …… **入力**．区間 $[a,b]$ の端点 a．
出力．極小点 x^*．

B …… **入力**．区間 $[a,b]$ の端点 b．

FUN …… **入力**．$f(x)$ を計算する関数副プログラム名．
〔副プログラムの用意の仕方〕
FUNCTION FUN (X)
パラメタ
X …… **入力**．変数 x．
関数 FUN には，$f(x)$ の値を代入すること．
（使用例参照）

EPSR …… **入力**．収束判定値（≥0.0）
0.0 のときは標準値が採用される．
（使用上の注意②参照）

MAX …… **入力**．関数 $f(x)$ の評価回数の上限（≠0）．
（使用上の注意③参照）
出力．実際に評価した回数（>0）．

F …… **出力**．関数値 $f(x^*)$．

ICON …… **出力**．コンディションコード．
表 LMINF-1 参照．

表 LMINF-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	指定された関数の評価回数内で，収束条件が満たされなかった．	パラメタ A，F には最後の値を格納する．
30000	EPSR < 0.0 又は MAX = 0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II … AMACH，MGSSL
② FORTRAN 基本関数 … ABS，SQRT，AMAX1

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で，引数 FUN 及び GRAD に相当する副プログラム名を EXTERNAL 文で宣言すること．
② EPSR の与え方
本サブルーチンの収束判定は，反復の過程で点 x^* をはさむ 2 点 x_1，x_2 に関して

$$|x_1 - x_2| \le \max(1.0, \tilde{x}) \cdot \mathrm{EPSR}$$

となった場合に，点 $\tilde{x}$ を極小点 x^* とみなし反復を終了する．ここで $\tilde{x}$ は，$\tilde{x} = x_1$（$f(x_1) \le f(x_2)$ のとき），又は $\tilde{x} = x_2$（$f(x_1) > f(x_2)$ のとき）である．
本サブルーチンでは，関数 $f(x)$ が点 x^* の近傍で近似的に 2 次関数であることを前提としており，極小点 x^* を丸め誤差程度まで正しく求めるためには，

$\mathrm{EPSR} \approx \sqrt{u}$，$u$ は丸め誤差の単位

程度に与えるのが適当である．
EPSR の標準値は $2\sqrt{u}$ である．
③ MAX の与え方
関数の評価回数は，変数 x に対して，$f(x)$ を計算することを 1 回と数える．すなわち，副プログラム FUN の呼出し回数に一致する．
この評価回数は，関数 $f(x)$ の性質，区間 $[a,b]$ の与え方，収束判定値に依存する．
一般には，区間 $[a,b]$ が適当に与えられ，収束判定値として標準値を採用した場合には，MAX = 400 程度が妥当である．
指定評価回数内で収束条件が満たされず，ICON = 10000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ MAX には，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと．
④ A，B の与え方
本サブルーチンでは，$f(x)$ が区間 $[a,b]$ で唯一の極小点をもつ場合，その値を許容誤差の範囲内で求める．区間 $[a,b]$ に多くの極小点を持つ場合には，いずれの極小点に収束するかは保証されない．
したがって，求める極小点 x^* を含む区間の端点 a 及び b の値は，可能なかぎり x^* の近くにとることが望ましい．

c. 使用例

$$f(x) = x^4 - 4x^3 - 6x^2 - 16x + 4$$

の区間 [-5,5] の極小値を求める．
ただし，収束判定値は標準値を与える．

```
C       **EXAMPLE**
        EXTERNAL FUN
        A=-5.0
        B=5.0
        EPSR=0.0
        MAX=400
        CALL LMINF(A,B,FUN,EPSR,
       *           MAX,F,ICON)
        WRITE(6,600) ICON,MAX,A,F
  600 FORMAT('1'//1X,'ICON=',I6,2X,
     * 'MAX=',I5,2X,'A=',E15.7,2X,
     * 'F=',E15.7)
        STOP
        END
C       OBJECTIVE FUNCTION
        FUNCTION FUN (X)
        FUN=(((X-4.0)*X-6.0)*X-16.0)*X+4.0
        RETURN
        END
```

(4) 手法概要

1 変数の実関数 $f(x)$ と極小値を求める区間 $[a,b]$ が与えられたとき，2 次補間法により $f(x)$ の極小点 x^* とその関数値 $f(x^*)$ を求める．
ここで，区間 $[a,b]$ において $-f(x)$ は単峰であると仮定する．単峰でない場合は，いずれかの極小点を求めることになる．

本方法は，次の二つのステップから成る．

- x^* をはさむ 2 点 x_1, x_2 を求める．
- x_1, x_2 及び 2 点の中点 x_h における関数値 $f(x_1)$, $f(x_2)$ 及び $f(x_h)$ に基づき，点 $(x_1,f(x_1))$, $(x_h,f(x_h))$，$(x_2,f(x_2))$ を通る 2 次曲線をあてはめ，その公式より最小点を求める．

この二つの手順を極小点を含む区間幅が収束と判定されるまで繰り返す．

本サブルーチンの計算手順

以下の説明では簡単のために，$f(x_1)$，$f(x_2)$ 等を f_1，f_2 等と表すことにする．

ステップ 1（2 点 x_1，x_2を求める）

① a, b のうち小さい方を x_1, 大きい方を x_2 とする．
また ε=max (EPSR, $2\sqrt{u}$) とする．

② 区間 $[x_1,x_2]$ の中点 x_h を求める．
x_h=(x_1+ x_2)/2

③ もし，$f_1 > f_h < f_2$ ならば，$h = x_2 - x_h$ としてステップ 2 へ進む．

④ もし，$f_1 \le f_h \le f_2$ ならば，$x_2 = x_h$ として②へ戻る．
ただし，収束条件
$|x_1 - x_h| \le \max(1.0, |x_1|) \cdot \varepsilon$
を満す場合は，x_1 を x^*，f_1 を $f(x^*)$ とみなし ICON = 0 として処理を終了する．

⑤ もし $f_1 \ge f_h \ge f_2$ ならば，$x_1 = x_h$ として②へ戻る．
ただし，収束条件
$|x_h - x_2| \le \max(1.0, |x_2|) \cdot \varepsilon$
を満す場合は，x_2 を x^*，f_2 を $f(x^*)$ とみなし ICON = 0 として処理を終了する．

⑥ もし $f_1 < f_h > f_2$ ならば，$x_2 = x_h$（$f_1 \le f_2$ のとき）又は，$x_1 = x_h$（$f_1 > f_2$ のとき）として②へ戻る．ただし，収束条件
$|x_1 - x_h| \le \max(1.0, |x_1|) \cdot \varepsilon$　　（$f_1 \le f_2$ のとき）
又は
$|x_2 - x_h| \le \max(1.0, |x_2|) \cdot \varepsilon$　　（$f_1 > f_2$ のとき）
を満す場合は，x_1 又は x_2 を x^*，f_1 又は f_2 を $f(x^*)$ とみなし，ICON = 0 として処理を終了する．

ステップ 2（2 次補間）

⑦ 2 次補間により最小点 x_m を求める．すなわち，次の諸量を計算する．
$$\Delta h = \frac{h}{2}(f_2 - f_1)/(f_1 - 2f_h + f_2)$$
$$x_m = x_h - \Delta h$$

⑧ もし，収束条件
$|\Delta h| \le \max(1.0, |x_h|) \cdot \varepsilon/2$
を満す場合は，⑨へ進む．
満さない場合は，$x_1 = x_m$（$f_m \le f_h$ のとき）又は $x_1 = x_h$，$x_h = x_m$（$f_m > f_h$ のとき）とし，再び x^* をはさむ 2 点を求めた後⑦に戻る．

⑨ 最小値を与える点 $\tilde{x}$，すなわち $\tilde{x} = x_m$（$f_m < f_h$ のとき）又は，$\tilde{x} = x_h$（$f_m \ge f_h$ のとき）の近傍の 2 点
$\tilde{x}_1 = \tilde{x} + \max(1.0, |\tilde{x}|) \cdot \varepsilon/2$
$\tilde{x}_2 = \tilde{x} - \max(1.0, |\tilde{x}|) \cdot \varepsilon/2$
に対して
$f(\tilde{x}_1) < \min(f_m, f_h)$ かつ
$f(\tilde{x}_2) < \min(f_m, f_h)$
を満たす場合は，$\tilde{x}$ を x^*，$f(\tilde{x})$ を $f(x^*)$ とみなし，ICON = 0 として処理を終了する．
満たさない場合は，再び x^* をはさむ 2 点を求めた後⑦に戻る．

D11-40-0101 LMING, DLMING

1 変数関数の極小化（微係数要，3 次補間法）
CALL LMING (A, B, FUN, GRAD, EPSR, MAX, F, ICON)

(1) 機能

1 変数の実関数 $f(x)$ とその導関数 $g(x)$ が与えられたとき，区間 $[a, b]$ で $f(x)$ の極小点 x^* とその関数値 $f(x^*)$ を求める．
ただし，$f(x)$ は 3 階までの連続な微係数を持つものと仮定する．

(2) パラメタ

A ……**入力**．区間 $[a, b]$ の端点 a．
出力．極小点 x^*．

B……**入力**．区間 $[a, b]$ の端点 b．

FUN ……**入力**．$f(x)$ を計算する関数副プログラム名．
〔副プログラムの用意の仕方〕
FUNCTION FUN (X)
パラメタ
X… **入力**．変数 x．
関数 FUN には，$f(x)$ の値を代入すること．
（使用例参照）

GRAD ….**入力**．$g(x)$ を計算する関数副プログラム名．
〔副プログラムの用意の仕方〕
FUNCTION GRAD (X)
パラメタ
X… **入力**．変数 x．
関数 GRAD には $g(x)$ の値を代入すること．
（使用例参照）

EPSR……**入力**．収束判定値（≥0.0）．
0.0 のときは標準値が採用される．
（使用上の注意②参照）

MAX ……**入力**．関数 $f(x)$ 及び $g(x)$ の評価回数の上限（≠0）．
（使用上の注意③参照）
出力．実際に評価した回数（>0）．

F ……**出力**．関数値 $f(x^*)$．

ICON……**出力**．コンディションコード．
表 LMING-1 参照．

表 LMING-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	指定された関数の評価回数内で，収束条件が満たされなかった．	パラメタ A，F には最後の値を格納する．
20000	EPSR が小さすぎる．	処理を打ち切る．（パラメタ A，F には最後の値を格納する）
30000	EPSR<0.0 又は MAX=0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II… AMACH，MGSSL
② FORTRAN 基本関数 … ABS，SQRT，AMAX1

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で，引数 FUN 及び GRAD に相当する副プログラム名を EXTERNAL 文で宣言すること．
② EPSR の与え方
本サブルーチンの収束判定は，反復の過程で点 x^* をはさむ 2 点 x_1，x_2 に関して

$$|x_1 - x_2| \le \max(1.0, \tilde{x}) \cdot \mathrm{EPSR}$$

となった場合に，点 $\tilde{x}$ を極小点とみなし反復を終了する．ここで $\tilde{x}$ は，$\tilde{x} = x_1$（$f(x_1) \le f(x_2)$ のとき），又は $\tilde{x} = x_2$（$f(x_1) > f(x_2)$ のとき）である．本サブルーチンでは，関数 $f(x)$ が点 x^* の近傍で近似的に 3 次関数であることを前提としており，極小点 x^* を丸め誤差程度まで正しく求めるためには，
$EPSR \approx \sqrt{u}$，u は丸め誤差の単位
程度に与えるのが適当である．
EPSR の標準値は $2\sqrt{u}$ である．
③ MAX の与え方
関数の評価回数は，変数 x に対して，$f(x)$ を計算することを 1 回，$g(x)$ を計算することを 1 回と数える．すなわち，副プログラム FUN 及び GRAD の呼び出し回数に一致する．
この評価回数は，関数 $f(x)$，$g(x)$ の性質，区間 $[a, b]$ の与え方，収束判定値に依存する．
一般には，区間 $[a,b]$ が適当に与えられ，収束判定値として標準値を採用した場合には，MAX = 400 程度が妥当である．
指定評価回数内で収束条件が満たされず，ICON = 10000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ MAX には，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと．
④ A，B の与え方
本サブルーチンでは，$f(x)$ が区間 $[a,b]$ で唯一の極小点をもつ場合，その値を許容誤差の範囲内で求める．区間 $[a,b]$ に多くの極小点を持つ場合には，いずれの極小点に収束するかは保証されない．
したがって，求める極小点 x^* を含む区間の端点 a 及び b の値は，可能なかぎり x^* の近くにとることが望ましい．

c. 使用例

$$f(x) = x^4 - 4x^3 - 6x^2 - 16x + 4$$

の区間 [-5,5] の極小値を求める．ただし，$f(x)$ の導関数 $g(x)$ は

$$g(x) = 4x^3 - 12x^2 - 12x - 16$$

であり，収束判定値は標準値を与える．

```
C      **EXAMPLE**
       EXTERNAL FUN,GRAD
       A=-5.0
       B=5.0
       EPSR=0.0
       MAX=400
       CALL LMING(A,B,FUN,GRAD,EPSR,
      *           MAX,F,ICON)
       WRITE(6,600) ICON,MAX,A,F
  600 FORMAT('1'//1X,'ICON=',I6,2X,
     * 'MAX=',I5,2X,'A=',E15.7,2X,
     * 'F=',E15.7)
       STOP
       END
C      OBJECTIVE FUNCTION
       FUNCTION FUN(X)
       FUN=(((X-4.0)*X-6.0)*X-16.0)*X+4.0
       RETURN
       END
C      DERIVATIVE
       FUNCTION GRAD(X)
       GRAD=((4.0*X-12.0)*X-12.0)*X-16.0
       RETURN
       END
```

(4) 手法概要

1 変数の実関数 $f(x)$ とその導関数 $g(x)$，及び極小値を求める区間 $[a,b]$ が与えられたとき，3 次補間法により $f(x)$ の極小点 x^* とその関数値 $f(x^*)$ を求める．ここで，区間 $[a,b]$ において $-f(x)$ は単峰であると仮定する．単峰でない場合は，いずれかの極小点を求めることになる．

本方法は，次の二つのステップから成る．

- x^* をはさむ 2 点 x_1，x_2 を求める．
- x_1，x_2 における関数値 $f(x_1)$，$f(x_2)$ 及びそれらの微係数 $g(x_1)$，$g(x_2)$ に基づき，点 $(x_1, f(x_1))$，$(x_2, f(x_2))$ を通る 3 次曲線をあてはめ，その公式より最小点を求める．

この二つの手順を極小点を含む区間幅が収束と判定されるまで繰り返す．

本サブルーチンの計算手順

以下の説明では，簡単のために，$f(x_1)$，$f(x_2)$ 等を f_1，f_2 等と表わすことにする．

ステップ 1（2 点 x_1，x_2 を求める）

① a，b のうち小さい方を x_1，大きい方を x_2 とする．
また ε =max(EPSR, $2\sqrt{u}$) とする．
もし，$g_1 < 0$ かつ $g_2 > 0$ ならば，ステップ 2 へ進む．

② 区間 $[x_1, x_2]$ の中点 x_h を求める．
$x_h = (x_1 + x_2) / 2$

③ もし，$f_1 \le f_h \le f_2$ ならば，$x_2 = x_h$ として②へ戻る．ただし，収束条件
$|x_1 - x_h| \le \max(1.0, |x_1|) \cdot \varepsilon$
を満す場合は，x_1 を x^*，f_1 を $f(x^*)$ とみなし ICON = 0 として処理を終了する．

④ もし，$f_1 \ge f_h \ge f_2$ ならば，$x_1 = x_h$ として②へ戻る．ただし，収束条件
$|x_h - x_2| \le \max(1.0, |x_2|) \cdot \varepsilon$
を満す場合は，x_2 を x^*，f_2 を $f(x^*)$ とみなし ICON = 0 として処理を終了する．

⑤ もし，$f_1 < f_h > f_2$ ならば，$x_2 = x_h$（$f_1 \le f_2$ のとき）又は，$x_1 = x_h$（$f_1 > f_2$ のとき）として②へ戻る．ただし，収束条件
$|x_1 - x_h| \le \max(1.0, |x_1|) \cdot \varepsilon$　　（$f_1 \le f_2$ のとき）
又は
$|x_2 - x_h| \le \max(1.0, |x_2|) \cdot \varepsilon$　　（$f_1 > f_2$ のとき）
を満す場合は，x_1 又は x_2 を x^*，f_1 又は f_2 を $f(x^*)$ とみなし，ICON = 0 として処理を終了する．

⑥ もし，$f_1 > f_h < f_2$ で，かつ

- $g_h < 0$，$g_2 > 0$ ならば $x_1 = x_h$ としてステップ 2 へ進む．
- $g_1 < 0$，$g_h > 0$ ならば $x_2 = x_h$ としてステップ 2 へ進む．
- $g_1 \ge 0$，$g_h \ge 0$ ならば $x_2 = x_h$ として②へ戻る．
- $g_h \le 0$，$g_2 \le 0$ ならば $x_1 = x_h$ として②へ戻る．

ステップ 2（3 次補間）

⑦ 3 次補間により最小点 x_m を求める．即ち，次の諸量を計算する．
$h = x_2 - x_1$，$z = 3(f_1 - f_2) / h + g_1 + g_2$，
$w = (z_2 - g_1 g_2)^{1/2}$，
$\Delta h = h(w + z - g_1) / (g_2 - g_1 + 2w)$，
$x_m = x_1 + |\Delta h|$

⑧ もし，$x_m \ge x_2$ ならば，x_1 を x^* とみなし，ICON = 20000 として処理を打ち切る．

⑨ もし，$x_m < x_2$ で，かつ

・ 収束条件

$$|h| \le \max(1.0, |x_m|) \cdot \varepsilon$$

を満す場合は，又は $g_m = 0$ の場合は，x_m を x^*，f_m を $f(x^*)$ とみなし，ICON = 0 として処理を終了する．

・ 収束条件を満さない場合は，$x_1 = x_m$（$g_m < 0$ のとき）又は，$x_2 = x_m$（$g_m > 0$ のとき）として⑦に戻る．

C21-41-0101 LOWP, DLOWP

実係数低次代数方程式（5 次以下）
CALL LOWP (A, N, Z, ICON)

(1) 機能

実係数低次代数方程式,

$$a_0 x^n + a_1 x^{n-1} + \ldots + a_n = 0 \quad (n \le 5, a_0 \ne 0)$$

の根を逐次代入法，ニュートン法，フェラリー法，ベアストウ法，2 次方程式の根の公式により求める.

(2) パラメタ

A ····· **入力**．代数方程式の係数．大きさ $n+1$ の 1 次元配列.
$A(1) = a_0$， $A(2) = a_1$， ……， $A(N+1) = a_n$
の順に与える.

N ····· **入力**．代数方程式の次数 n.

Z ····· **出力**．n 個の根．大きさ n の複素数型 1 次元配列.

ICON···· **出力**．コンディションコード.
表 LOWP-1 参照.

表 LOWP-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	5 次方程式の一つの実根を求めるとき 50 回の逐次代入を行っても $f_k f_{k+1} < 0$ とならなかった.	x_{k+1} をニュートン法の初期値として処理を続ける.（手法概要の (4.16) 式参照）
30000	$a_0 = 0.0$，$n \le 0$ 又は $n > 5$ であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II··· AMACH，RQDR，MGSSL，U3DEG，UREDR

② FORTRAN 基本関数 ··· SQRT，ABS，CMPLX

b. 使用例
次数 n と係数 a_i（$i = 0$，1，…，n）を入力し，n 個の根を求める.

```
C     **EXAMPLE**
      DIMENSION A(6),Z(5)
      COMPLEX Z
      READ(5,500) N
      N1=N+1
      READ(5,510)(A(I),I=1,N1)
      CALL LOWP(A,N,Z,ICON)
      WRITE(6,600) N,ICON
      WRITE(6,610)(A(I),I=1,N1)
      IF(ICON.EQ.30000) STOP
      WRITE(6,620)(Z(I),I=1,N)
      STOP
  500 FORMAT(I1)
  510 FORMAT(6F10.0)
  600 FORMAT(10X,'N=',I1,5X,'ICON=',I7)
  610 FORMAT(10X,'A=',E20.8/(12X,E20.8))
  620 FORMAT(10X,'Z=',2E20.8/(12X,2E20.8))
      END
```

(4) 手法概要

以下，代数方程式の最高次の係数を 1 と仮定して説明する（こうしても一般性は失われない）.

a. $n = 2$ のとき（2 次方程式）
根の公式を使って計算する．これはサブルーチン RQDR を使用することにより行う.

b. $n = 3$ のとき（3 次方程式）
$f(x) = x^3 + c_1 x^2 + c_2 x + c_3 = 0$ の一つの実根 x_1 が分かれば,

$$f(x) = (x - x_1)(x^2 + p_1 x + p_2) \qquad (4.1)$$

ここで $\begin{cases} p_1 = c_1 + x_1 \\ p_2 = c_2 + x_1 p_1 \end{cases}$

となって，2 次方程式の問題に還元される.
一つの実根 x_1 はニュートン法により求める．その初期値 x_a は $f(x)$ の特性を調べることにより決定する.
すなわち,

$$f'(x) = 3x^2 + 2c_1 x + c_2 = 0 \qquad (4.2)$$

とおいて，次の三つの場合に分け x_a を決める.

(a) $f'(x) = 0$ が異なる 2 実根 x_{m1}，x_{m2} を持つとき.
このときは $f(x)$ が極大値 $f(x_{m1})$，極小値 $f(x_{m2})$ をもつ場合にあたる．もし $f(x_{m1})f(x_{m2}) < 0$ なら x_a は $f(x)$ の変曲点とする．逆に $f(x_{m1})f(x_{m2}) > 0$ なら，更に次に示すように二つの場合に分け，x_a を決める.

・ $f(x_{m1}) > 0$ の場合
$f(m_1 + h)$ の h^2 までのテイラー展開の式を 0 とおいた h に関する 2 次方程式を考えその正の実根を改めて h として,

$$x_a = x_{m1} - h \qquad (4.3)$$

とする.

・ $f(x_{m1}) < 0$ の場合
$f(x_{m2} + h)$ の h^2 までのテイラー展開の式を 0 とおいた h に関する 2 次方程式を考えその正の実根を改めて h として,

$$x_a = x_{m2} + h \qquad (4.4)$$

とする.

(b) $f'(x) = 0$ が重根を持つとき.
このときは極大値と極小値が変曲点に一致した場合にあたる．変曲点を x_i とすると，逐次代入法を 1 回適用して

$$x_a = x_i - f(x_i) \qquad (4.5)$$

とする.

(c) $f'(x) = 0$ が実根を持たないとき.
x_a は $f(x)$ の変曲点とする.

c. $n = 4$ のとき（4 次方程式）
一般に 4 次方程式

$$f(x) = x^4 + c_1x^3 + c_2x^2 + c_3x + c_4 = 0 \quad (4.6)$$

はフェラリー法により二つの 2 次因数に分解される.

$$f(x) = \left(x^2 + p_1x + p_2\right)\left(x^2 + q_1x + q_2\right) \quad (4.7)$$

次の 3 次方程式,

$$\mu^3 - c_2\mu^2 + (c_1c_3 - 4c_4)\mu + \left(4c_2c_4 - c_3^2 - c_1^2c_4\right) = 0 \quad (4.8)$$

の一つの実根 μ を求めれば，p_2，q_2 は 2 次方程式,

$$v^2 - \mu v + c_4 = 0 \quad (4.9)$$

の 2 根として，p_1，q_1 は 2 次方程式,

$$v^2 - c_1v + (c_2 - \mu) = 0 \quad (4.10)$$

の 2 根として求められる．もし p_2，q_2 が複素数ならば (4.8) の残りの実根について調べ，p_2，q_2 がある許容量の範囲内で実数となるものを採用しておく．その後で p_1，q_1，p_2，q_2 をベアストウ法を用いて修正する. すなわち, 次の諸量,

$$\left.\begin{aligned} q_1 &= c_1 - p_1 \\ q_2 &= c_2 - p_1q_1 - p_2 \\ q_3 &= c_1 - p_1q_2 - p_2q_1 \\ q_4 &= c_0 - p_1q_3 - p_2q_2 \end{aligned}\right\} \quad (4.11)$$

$$\left.\begin{aligned} d_1 &= q_1 - p_1 \\ d_2 &= q_2 - p_1d_1 - p_2 \\ d_3 &= -p_1d_2 - p_2d_1 \end{aligned}\right\} \quad (4.12)$$

を q_1 から q_4 へ，次に d_1 から d_3 へと計算し，次の Δp_1，Δp_2 に関する連立方程式 (4.13) の解 Δp_1，Δp_2 を求める.

$$\begin{aligned} d_2\Delta p_1 + d_1\Delta p_2 &= q_3 \\ d_3\Delta p_1 + d_2\Delta p_2 &= q_4 \end{aligned} \quad (4.13)$$

$p_1 + \Delta p_1$，$p_2 + \Delta p_2$ を新しい p_1，p_2 とし，これに対応する q_1，q_2 は (4.11) より求める.

d. $n = 5$ のとき（5 次方程式）
5 次方程式,

$$f(x) = x^5 + c_1x^4 + c_2x^3 + c_3x^2 + c_4x + c_5 = 0 \quad (4.14)$$

の一つの実根を x_1 とすると $f(x)$ は次のようにかける.

$$f(x) = (x - x_1)\left(x^4 + c_1'x^3 + c_2'x^2 + c_3'x + c_4'\right) = 0 \quad (4.15)$$

ここで
$c_1' = c_1 + x_1$
$c_2' = c_2 + x_1c_1'$
$c_3' = c_3 + x_1c_2'$
$c_4' = c_4 + x_1c_3'$
したがって x_1 を求めたならあとは 4 次方程式を解けばよいことになる．x_1 の求め方を以下に示す.

[x_1 の求め方]
まず逐次代入法,

$$\left.\begin{aligned} &x_0 = 0 && f_0 = c_5 \\ &x_{k+1} = x_k - f_k \\ &f_k = f(x_k) && k = 0, 1, \cdots, 50 \end{aligned}\right\} \quad (4.16)$$

により，$f_kf_{k+1} < 0$ となる x_k，x_{k+1} を求める.
次に擬点法（regula falsi），

$$x_a = x_{k+1} - (x_{k+1} - x_k)f_{k+1} / (f_{k+1} - f_k)$$

により求めた x_a を初期値としてニュートン法を適用する．(4.16) において k の上限は 50 とし，f_k　$f_{k+1} < 0$ とならなかった場合は，最後の x_{k+1} を，ニュートン法の初期とする.

収束判定法について

以上のように，3 次方程式あるいは 5 次方程式のひとつの実根はニュートン法を適用して求めているが，そのときの収束判定法について述べる.

3 次方程式あるいは 5 次方程式を，一般的に

$$c_0x^n + c_1x^{n-1} + \ldots\ldots + c_n = 0 \quad (4.17)$$

（ただし $c_0 = 1$，n は 3 または 5）

で表し，また (4.17) にニュートン法を適用したときの k 回目の近似値を x_k で表せば，この x_k が

$$\left|\sum_{j=0}^{n} c_j x_k^{n-j}\right| \le u \sum_{j=0}^{n} \left|c_j x_k^{n-j}\right| \quad (4.18)$$

を満たしたとき, x_k を根として採用する. ここで u は丸め誤差の単位である．なお，詳細については，参考文献 [25] を参照すること.

D21-10-0101 LPRS1, DLPRS1

線形計画問題（改訂シンプレックス法）
CALL LPRS1 (A, K, M, N, EPSZ, IMAX, ISW, NBV, B, VW, IVW, ICON)

(1) 機能

次の線形計画問題を改訂シンプレックス法により解く.

"条件 $\sum_{j=1}^{n} a_{ij}x_j \le d_i,\ i=1, 2, \ldots,\ m_l,$

$\sum_{j=1}^{n} a_{ij}x_j \ge d_i,\ i=m_l+1,\ m_l+2, \ldots,$

$m_l+m_g,$

$\sum_{j=1}^{n} a_{ij}x_j = d_i,\ i=m_l+m_g+1,\ m_l+m_g+2,$

$\ldots, m_l+m_g+m_e,$

$x_j \ge 0,\quad j=1,\ 2, \ldots,\ n$

のもとで，線形の目的関数

$$z=\sum_{j=1}^{n} c_j x_j + c_0$$

を最小（又は最大）にせよ．"

本サブルーチンでは，解法が 2 段階に分れており，第 1 段階では可能基底解を求め，（可能基底解が存在するならば）第 2 段階で最適解を求める.

初期可能基底を入力することもできる．また d_i の符号に制約はない.

以降，$m = m_l + m_g + m_e$ とし，要素 a_{ij} によりなる $m \times n$ 行列を係数行列 $\boldsymbol{A}$, $\boldsymbol{d} = (d_1, d_2, \cdots, d_m)^T$ を定数ベクトル，$\boldsymbol{c} = (c_1, c_2, \cdots, c_n)^T$ を係数ベクトル，c_0 を定数項と呼ぶ.

ここで，$n \ge 1$，$m_l \ge 0$，$m_g \ge 0$，$m_e \ge 0$，$m \ge 1$ であること.

(2) パラメタ

A ····· **入力**．係数行列 $\boldsymbol{A}$，定数ベクトル $\boldsymbol{d}$，係数ベクトル $\boldsymbol{c}$，定数項 c_0.
図 LPRS1-1 参照
A(K, N+1) なる 2 次元配列.

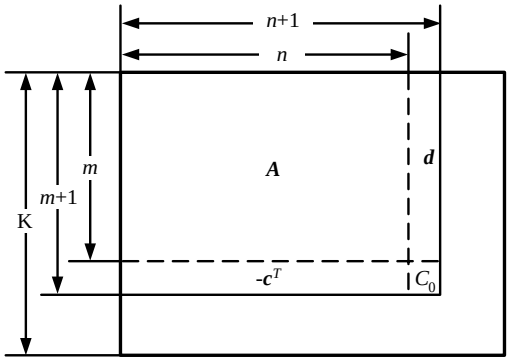


図 LPRS1-1 配列 A の内容

K ····· **入力**．配列 A，B の整合寸法（$\ge m+1$）.

M ····· **入力**．条件式の個数．M(1) = m_l, M(2) = m_g, M(3) = m_e なる対応をもつ，大きさ 3 の 1 次元配列.

N ····· **入力**．変数の個数 n.

EPSZ····· **入力**．反復過程での要素（係数及び定数項），基底逆行列 $\boldsymbol{B}^{-1}$ を求めるときのピボットに対する相対零判定値（≥0.0）.
0.0 のときは，標準値が採用される.
使用上の注意③参照.

IMAX ····· **入力**．反復回数の上限（≠ 0 ）
使用上の注意②参照.
出力．実際に反復した回数（> 0）

ISW ····· **入力**．制御情報.
最小化問題か最大化問題か，および初期可能基底を与えるか否かを指定する.
ISW = $10d_1 + d_0$ なる値を入力する.
ここで，$0 \le d_0$，$d_1 \le 1$.
$d_0 = 0$ のとき最小化問題を解く.
$d_0 = 1$ のとき最大化問題を解く.
$d_1 = 0$ のとき基底を与えない.
$d_1 = 1$ のとき基底を与える.
出力．最適解又は可能基底解が得られた場合，$d_1 = 1$ となる.
それ以外の場合は，入力した値が保存される.

NBV ····· **入力**．（ISW = 10 又は 11 と指定したとき）可能基底解の基底変数の番号.
出力．最適解又は可能基底解の基底変数の番号.
大きさ m の 1 次元配列.
使用上の注意①参照.

B ····· **出力**．最適解又は可能基底解に対する基底逆行列 $\boldsymbol{B}^{-1}$，最適解又は可能基底解 $\boldsymbol{g}$，シンプレックス乗数 $\boldsymbol{\pi}$，目的関数の値 q.
図 LPRS1-2 参照.
B(K, m+1) なる 2 次元配列.

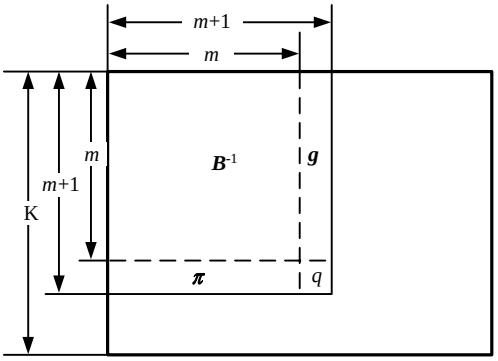


図 LPRS1-2 配列 B の内容

VW…… **作業領域**．大きさ $2n+m+m_l+m_g+1$ の 1 次元配列．

IVW …… **作業領域**．大きさ $n+m_l+m_g$ の 1 次元配列．

ICON…… **出力**．コンディションコード．表 LPRS1-1 参照．

表 LPRS1-1 コンディションコード

コード	意味	処理内容
0	最適解が得られた．	
10000	可能基底解は得られたが最適解は存在しない．	配列 B には，可能基底解と対応する基底逆行列，シンプレックス乗数，目的関数の値を格納する．
11000	第 2 段階の途中で反復回数が，指定された上限に達した． 可能基底解は得られている．	
20000	可能解は存在しない． EPSZ の値が適切でない可能性がある．	処理を打ち切る．
21000	ISW = 10 又は 11 のとき，与えられた変数の組は基底でない．	処理を打ち切る．
22000	ISW = 10 又は 11 のとき，与えられた変数の組を基底とする基底解は，可能解でない．	処理を打ち切る．
23000	第 1 段階の途中で，基底変数の交換が行えなくなった．EPSZ の値が適切でない可能性がある．	処理を打ち切る．
24000	第 1 段階の途中で反復回数が，指定された上限に達した．	処理を打ち切る．
30000	次のいずれかであった． ①M(1)，M(2)，M(3) の中に負のものがあった． ② N < 1 ③ IMAX = 0 ④ EPSZ < 0.0 ⑤ M(1) + M(2) + M(3) < 1 ⑥ M(1) + M(2) + M(3) ≥ K ⑦ NBV の要素の中に 0 又は負のものがあった． ⑧ NBV の要素の中に，N + M(1) + M(2) より大きいものがあった． ⑨ NBV の要素の中に，同じものがあった． ⑩ ISW の与え方に誤りがあった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… ALU，AMACH，LUIV，MGSSL

② FORTRAN 基本関数 … ABS，IABS

b. 注意

① NBV の内容

ISW = 10 又は 11 として基底変数の番号を NBV に指定する場合，i 番目（$i \le m_l+m_g$）の不等式条件に対するスラック変数の番号は $n+i$ とすること．

同様に計算結果の基底解に i 番目のスラック変数が含まれる場合は，$n+i$ として与えられる．一方，人為変数については，入力時に基底変数として指定することはできない．ただし，計算結果の基底解の中に人為変数が混入する場合，NBV に 0 の値が与えられる．すなわち，i 番目の基底変数が人為変数であるときは，NBV(i) = 0 になっている．可能基底解が得られているとき（ICON = 0, 10000 又は 11000 のとき），NBV(i) = 0 は i 番目の条件式がむだな条件式であることを示している．（むだな条件式とは，NBV(i) ≠ 0 なるいくつかの条件式より導びかれる条件式である．）

② IMAX の与え方．

反復回数は，可能基底解（第 1 段階では人為変数を含んだ問題の可能基底解）に対する最適性規準（手法概要 (4.7) 参照）の判定回数を指す．IMAX の標準値は $10m$ である．

指定反復回数内で最適解が得られず，ICON = 11000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ IMAX には，負の値で追加反復回数を指定し，他のパラメタの内容は保存したまま呼び出すこと．

③ EPSZ の与え方

入力データ（配列 A の内容）の要素の絶対値の最大値を $a_{\max}$ とすると，反復過程での要素の絶対値が $a_{\max}\cdot$EPSZ より小さい要素は零とみなされる．

また，基底逆行列 $\boldsymbol{B}^{-1}$ をサブルーチン ALU，LUIV により求める場合に，ピボットの相対零判定値として使用される．（サブルーチン ALU の「使用上の注意」参照）

EPSZ の標準値は丸め誤差の単位を u としたとき，

EPSZ = $16\cdot u$ である．

入力データ（配列 A の内容）は，行又は列毎に定数を乗じて，絶対値をできるだけ揃えておくことが望ましい．

なお，ICON = 20000 又は 23000 となった場合は，EPSZ の値が適切でない可能性がある．このような場合は，EPSZ の値を標準値とするか又は変更して再実行してみるとよい．

④ 変数 x_j に符号の制約がない場合の対策
本サブルーチンでは，$x_j \geq 0$，$j=1$，2，…，n を原則としている．しかし x_j に符号の制約がない場合でも以下のように問題を変換することにより本サブルーチンを使用することができる．

・ x_j を $x_j^+ - x_j^-$ に置き換える．
・ 条件 $x_j^+ \geq 0$，$x_j^- \geq 0$ を付加する．

c. 使用例
変数の個数 n が最大 10，条件式の個数 m が最大 20 の場合の線形計画問題を解く．

```
C     **EXAMPLE**
      DIMENSION A(21,11),B(21,21),M(3),
     *          NBV(20),VW(61),IVW(30)
      READ(5,500) N,M,ISW
      N1=N+1
      MM=M(1)+M(2)+M(3)
      M1=MM+1
      DO 10 I=1,M1
      READ(5,510) (A(I,J),J=1,N1)
   10 CONTINUE
      WRITE(6,600) (J,J=1,N)
      IF(M(1).EQ.0) GOTO 30
      WRITE(6,610)
      DO 20 I=1,M(1)
      WRITE(6,620) I,(A(I,J),J=1,N1)
   20 CONTINUE
   30 IF(M(2).EQ.0) GOTO 50
      WRITE(6,630)
      IS=M(1)+1
      IE=M(1)+M(2)
      DO 40 I=IS,IE
      WRITE(6,620) I,(A(I,J),J=1,N1)
   40 CONTINUE
   50 IF(M(3).EQ.0) GOTO 70
      WRITE(6,640)
      IS=M(1)+M(2)+1
      DO 60 I=IS,MM
      WRITE(6,620) I,(A(I,J),J=1,N1)
   60 CONTINUE
   70 IF(MOD(ISW,10).EQ.0) GOTO 80
      WRITE(6,650) (A(M1,J),J=1,N1)
      GOTO 90
   80 WRITE(6,660) (A(M1,J),J=1, N1)
   90 READ(5,520) EPSZ,IMAX
      WRITE(6,670) EPSZ,IMAX
      IF(ISW.LT.10) GOTO 100
      READ(5,500) (NBV(I),I=1,MM)
      WRITE(6,680) (NBV(I),I=1,MM)
  100 CALL LPRS1(A,21,M,N,EPSZ,IMAX,
     *           ISW,NBV,B,VW,IVW,ICON)
      WRITE(6,720) B(M1,M1)
      WRITE(6,690) ICON,IMAX
      IF(ICON.GE.20000) STOP
      IF(ICON.EQ.0) WRITE(6,700)
      IF(ICON.GE.10000) WRITE(6,710)
      WRITE(6,720) B(M1,M1)
      WRITE(6,730) (NBV(I),B(I,M1),I=1,MM)
      STOP
  500 FORMAT(10I4)
  510 FORMAT(11F6.0)
  520 FORMAT(E10.2,I5)
  600 FORMAT('1','INITIAL TABLEAU'
     *      /'0',10X,11(I6,4X))
  610 FORMAT(' ','LHS.LE.RHS')
  620 FORMAT(' ',I6,4X,11F10.3)
  630 FORMAT(' ','LHS.GE.RHS')
  640 FORMAT(' ','LHS.EQ.RHS')
  650 FORMAT(' ','OBJ.FUNC.(MAX)'
     *      /' ',10X,11F10.3)
  660 FORMAT(' ','OBJ.FUNC.(MIN)'
     *      /' ',10X,11F10.3)
  670 FORMAT('0','EPSZ=',E12.4,5X,
     *       'IMAX=',I4)
  680 FORMAT('0','INITIAL BASIS'
     *      /' ',20I4)
  690 FORMAT('0','ICON=',I5,12X,'IMAX=',I4)
  700 FORMAT('0','OPTIMAL SOLUTION')
  710 FORMAT('0','FEASIBLE SOLUTION')
  720 FORMAT(' ','OBJ.FUNC.',F15.4)
  730 FORMAT(' ',I6,3X,F15.4)
      END
```

(4) 手法概要

はじめに，標準形の線形計画問題

"条件　　$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{d}$　　(4.1)

$\boldsymbol{x} \geq \boldsymbol{o}$　　(4.2)

のもとで，線形の目的関数

$$z = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} + c_0 \tag{4.3}$$

を最小にせよ"

の解法について説明する．ここで，$\boldsymbol{A}$ は $m \times n$ 行列で

$$\text{rank } \boldsymbol{A} \ = \ m \leq n$$

とし，また，$\boldsymbol{x} = (x_1, x_2, ..., x_n)^T$，$\boldsymbol{d} = (d_1, d_2, ..., d_m)^T$，$\boldsymbol{c} = (c_1, c_2, ..., c_n)^T$ とする．

いま，$\boldsymbol{A}$ の第 j 列を $\boldsymbol{a}_j$ のように表すことにする．$\boldsymbol{A}$ の任意の m 個の列

$$a_{k_1}, a_{k_2}, ..., a_{k_m}$$

が 1 次独立であるとき，対応する x_{k_1}，x_{k_2}，…，x_{k_m} を基底といい，x_{k_i} を i 番目の基底変数という．非基底変数の番号の集合を L とすると，(4.1) と (4.3) と同値である連立 1 次方程式 (4.4) が存在する．

$$\begin{aligned} x_{k_i} + \sum_{j \in L} f_{ij} x_j &= g_i, \quad i = 1, 2, \cdots, m \\ z + \sum_{j \in L} p_j x_j &= q \end{aligned} \tag{4.4}$$

これを基底形式といい，一つの解

$$\begin{aligned} & x_{k_i} = g_i\,,\ \ i = 1,\ 2,\ \cdots,\ m \\ & x_j = 0\,,\ j \in L \\ & z = q \end{aligned} \qquad (4.5)$$

を基底解という．ここで，

$$g_i \geq 0,\ \ i = 1,\ 2,\ \cdots,\ m \qquad (4.6)$$

が成り立つとき，基底解 (4.5) は (4.2) も満たすので，この場合の (4.5) を可能基底解，その基底を可能基底，(4.4) を可能基底形式という．
(4.6) と

$$p_j \leq 0,\ J \in L \qquad (4.7)$$

が成りたつならば，(4.5) は最適解である．(4.7) を，可能基底解に対する最適性規準という．
　可能基底形式において，

$$p_s > 0,\ \ s \in L \qquad (4.8)$$

なる s が存在するなら，x_s をある基底と交換することにより基底に入れると，z の値をより小さくする可能基底解が得られる可能性がある．そこで，ϕ を空集合とし，

$$I^+ = \{i \mid f_{is} > 0\}$$

とするとき，$I^+ \neq \phi$ であって

$$\frac{a_r}{f_{rs}} = \min_{i \in I^+}\left(\frac{g_i}{f_{is}}\right) \qquad (4.9)$$

とすると，x_{k_r} の代りに x_s を入れた基底は可能基底であり，新しい基底に対する基底解は，

$$\begin{aligned} & x_{k'_r} = \frac{g_r}{f_{rs}}\,, \\ & x_{k'_i} = g_i - f_{is}\frac{g_r}{f_{rs}},\ \ i \neq r\,, \\ & x_j = 0 \qquad\ \ ,\ j \in L'\,, \\ & z = q - p_s\frac{g_r}{f_{rs}} \end{aligned} \qquad (4.10)$$

となる．ここで，k'_i は新しい基底変数の番号であり，L' は新しい非基底変数の番号の集合である．すなわち，

$$\begin{aligned} & k'_r = s, \\ & k'_i = k_i\ ,\ i \neq r, \qquad\qquad i \neq r\ , \\ & L' = L - \{s\} + \{k_r\} \end{aligned}$$

である．この新しい可能基底解における z の値は，$g_r > 0$ であれば，前の値より小さくなっている．一方，$I^+ = \phi$ の場合は，最適解は存在しない．
　ところで，

$$\begin{aligned} & \boldsymbol{B} = \left[\boldsymbol{a}_{k_1}, \boldsymbol{a}_{k_2}, ..., \boldsymbol{a}_{k_m}\right], \\ & \boldsymbol{C}_B = (c_{k_1}, c_{k_2}, ..., c_{k_m})\,, \\ & \boldsymbol{\pi} = \boldsymbol{C}_B\boldsymbol{B}^{-1} \end{aligned} \qquad (4.11)$$

とおくと，基底形式 (4.4) の係数及び右辺の値は，次のように表される．

$$\begin{aligned} & \boldsymbol{f}_j = \boldsymbol{B}^{-1}\boldsymbol{a}_j\,,\ j \in L \\ & \boldsymbol{g} = \boldsymbol{B}^{-1}\boldsymbol{d}\,, \\ & p_j = \boldsymbol{\pi}\,\boldsymbol{a}_j - c_j\ ,\ j \in L \\ & q = \boldsymbol{\pi}\,\boldsymbol{d} - c_0 \end{aligned} \qquad (4.12)$$

ここで $\boldsymbol{f}_j = (f_{1j}, f_{2j}, ..., f_{mj})^T$，$\boldsymbol{g} = (g_1, g_2, ..., g_m)^T$ であり，$\boldsymbol{B}$ を基底行列，$\boldsymbol{B}^{-1}$ を基底逆行列，$\boldsymbol{\pi}$ をシンプレックス乗数（ベクトル）という．

　以上が標準形の線形計画問題に対する解法の概要であるが，その手順を整理すると次のようになる．

① 可能基底に対する基底逆行列 $\boldsymbol{B}^{-1}$ と，シンプレックス定数 $\boldsymbol{\pi}$ を計算する．
② (4.12) により p_j を求める．
③ (4.7) に示した最適性規準を判定する．
　規準を満たすならば，そのときの $\boldsymbol{g}$ を最適解とする．
　満たされないならば，(4.12) により $\boldsymbol{f}_s$ を計算する．
④ (4.9) より x_s の代りに基底から出す変数を決定し，新しい可能基底を構成する．

この手順を繰り返して最適解を求める方法を，改訂シンプレックス法という．

標準形でない問題の場合の処置

不等式条件が含まれている場合は，次のように等式条件に直し，標準形の問題に変換する．

$$\sum_{j=1}^{n} a_{ij} x_j \le d_i \quad , \ i = 1,2,\ldots,\ m_l$$

については，

$$\left.\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j + x_{n+i} &= d_i \\ x_{n+i} &\ge 0 \end{aligned}\right\}, i = 1,2,\ldots,m_l \qquad (4.13)$$

とする．

$$\sum_{j=1}^{n} a_{ij} x_j \le d_i \ , i = m_l + 1 \ , m_l + 2 \ , \ldots, \ m_l + m_g$$

については，

$$\left.\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j - x_{n+i} &= d_i \\ x_{n+i} &\ge 0 \end{aligned}\right\}, \ i = m_l + 1 \ , m_l + 2 \ , \qquad (4.14)$$

$$\ldots, \ m_l + m_g$$

とする．

(4.13)，(4.14) における付加された変数 x_{n+1}，x_{n+2}，…，$x_{n+m_l+m_g}$ をスラック変数という．

一方，最大化問題は，目的関数に -1 を乗じ，最小化問題に変換する．

初期可能基底解の求め方

初期可能基底解が得られていない場合は，与えられた問題を解く前に次の問題を解く．

"条件　$\boldsymbol{A}^{*}\boldsymbol{x}^{*} + \boldsymbol{A}^{(a)}\boldsymbol{x}^{(a)} = \boldsymbol{d}$,
$\boldsymbol{x}^{*} \ge 0$，$\boldsymbol{x}^{(a)} > 0$　(4.15)

のもとで　$z_1 = \sum_{i=1}^{m} x_i^{(a)}$

を最小にせよ"

ここで，$\boldsymbol{x}^{*}$ はスラック変数も含めた $(n + m_l + m_g)$ 次元ベクトルであり，$\boldsymbol{A}^{*}$ は対応する係数行列である．また，$\boldsymbol{x}^{(a)}$ は $\boldsymbol{x}^{(a)} = \left(x_1^{(a)}, x_2^{(a)}, \ldots, x_m^{(a)}\right)^T$ なる m 次元ベクトル，$\boldsymbol{A}^{(a)}$ は $\boldsymbol{A}^{(a)} = \left(a_{ij}^{(a)}\right)$ なる m 次の対角行列で

$$a_{ij}^{(a)} = \begin{cases} 1, & \mathrm{d}_i \ge 0 \ \text{のとき} \\ -1, & \mathrm{d}_i < 0 \ \text{のとき} \end{cases} \qquad (4.16)$$

である．$x_i^{(a)}$ を人為変数という．

この最適解が得られたとき，$z_1 = 0$ すなわち $\boldsymbol{x}^{(a)} = \boldsymbol{o}$ であれば，与えられた問題の可能基底解が得られている．

一方 $z_1 > 0$ の場合は，与えられた問題に可能解（(4.1)，(4.2) を満たす $\boldsymbol{x}$）は存在しない．

また，$z_1 = 0$ であっても，人為変数が基底にのこっていることがある．この場合は，後に与えられた問題を解く段階で，該当の人為変数の値を常に 0 となるような処置を図る．

本サブルーチンの計算手順

初期可能基底が与えられる場合（ISW = 10 又は 11）は，②へ進む．

① 第 1 段階: 初期設定
(4.15) において $\boldsymbol{x}^{(a)}$ を基底として，その基底逆行列 $\boldsymbol{B}^{-1}$ を設定する．$\boldsymbol{B}^{-1}$ は (4.16) に等しい．
対応する基底解 $\boldsymbol{g}$，シンプレックス乗数 $\boldsymbol{\pi}$，目的関数の値 q を，(4.11) と (4.12) より次のように設定する．

$$g_i = |d_i|$$
$$\pi_i = \left\{\begin{array}{l} 1, d_i \ge 0 \text{のとき} \\ -1, d_i < 0 \text{のとき} \end{array}\right\}, i = 1,2,\ldots,m \qquad (4.17)$$
$$q = \sum_{i=1}^{m} |d_i|$$

④へ進む．

② 初期可能基底に対する基底逆行列を計算する．
サブルーチン ALU，LUIV により求める．もし，逆行列が求められないときは，ICON = 21000 として処理を終了する．求められたら，対応する基底解 $\boldsymbol{g}$ を (4.12) により計算する．
この基底解が可能解でないときは，ICON = 22000 として処理を終了する．

③ 第 2 段階: 初期設定
シンプレックス乗数 $\boldsymbol{\pi}$ を (4.11) より計算する．
また，目的関数の値 q を (4.12) の関係より
$q = \boldsymbol{C}_B\boldsymbol{g} + c_0$

と計算する．
ここで，スラック変数と人為変数が基底となっている場合は，対応する C_B の要素は 0 とみなす．

④ 最適性規準の判定
スラック変数も含めた非基底変数に対応した p_j を (4.12) により順次求め，(4.7) の最適性規準の判定を行う．(4.8) であれば⑤へ進む．
一方，最適性規準が成立している場合で，かつ
- ISW ≥ 10 の場合は，ここで最適解が得られたので，ICON = 0 として処理を終了する．
- ISW < 10 の場合で，かつ $q > 0$ の場合は，可能基底解は存在しないので，ICON = 20000 として処理を終了する．

・ ISW < 10 の場合で，かつ $q = 0$ の場合は，ここで初期可能基底解が得られた．第 1 段階を終了する．ISW = ISW + 10 として③へ進む．

⑤ 反復回数の判定
反復回数が指定された上限に達していない場合は⑥へ進み反復を継続する．
上限に達した場合で，かつ
・ 第 1 段階の過程の場合は ICON = 24000 として処理を終了する．
・ 第 2 段階の過程の場合は ICON = 11000 として処理を終了する．

⑥ 基底の交換
(4.12) により，x_s に対応する $\boldsymbol{f}_s$ を計算する．
もし，$f_{is} \le 0$，$i = 1$，2，…，m である場合は，最適解は存在しないので，第 1 段階の過程の場合は，ICON = 10000，第 2 段階の過程の場合は ICON = 23000 として処理を終了する．
一方，$f_{is} > 0$ の場合は，(4.9) より交換する基底 x_{k_r} を定める．

⑦ 新しい基底に対する基底逆行列等の計算
行列

$$\begin{bmatrix} \boldsymbol{B}^{-1} & \boldsymbol{g} \\ \boldsymbol{\pi} & q \end{bmatrix}$$

の第 i 行列（$i = 1$，2，…，m+1）を β_i で表すと，新しい基底に対する行列を，次のように求める．

$$\frac{1}{f_{rs}}\beta_r \to \beta_r \qquad (4.17)$$
$$\beta_i - f_{is}\beta_r \to \beta_i \quad ,i \ne r$$

この演算は，f_{rs} をピボットとするピボット演算という．④へ進む．

解法の収束性について

基底の交換において，新たに基底に入れる変数を x_s，基底から出る変数を x_{k_r} とするとき，

$$g_r > 0 \qquad (4.18)$$

であれば，目的関数の値は，(4.10) より

$$p_s \frac{g_r}{f_{rs}} \quad (>0)$$

だけ小さくなる．反復過程で (4.18) が満たされる限り同じ可能基底解が出現することはなく，可能基底解の個数は有限であるから，有限回の基底の交換により最適解に到達する（若しくは，最適解が存在しないことが分かる）．
一方，

$$g_r > 0$$

のときは，目的関数の値は不変であり，このような現象が何回か続くと，一度現れた可能基底解が再び現れる可能性がある．この場合，いくつかの可能基底解が周期的に現われて，循環現象を起こし最適解は求められない．本サブルーチンでは，この循環を防ぐために次のように s，r を定めている．
・ (4.8) で，$p_j > 0$ である j の中で最小のものを s とする．
・ (4.9) で，最小値が 0 で

$$\frac{g_{r_0}}{f_{r_0 s}} = \min_{i \in I^+}\left(\frac{g_i}{f_{is}}\right)$$

を満たす r_0 の中で，k_{r_0} を最小にするものを r とする．

なお，詳細については参考文献 [41] を参照すること．

A52-21-0101 LSBIX, DLSBIX

実対称バンド行列の連立 1 次方程式 （ブロック対角ピボッティング手法）
CALL LSBIX (A, N, NH, MH, B, EPSZ, ISW, VW, IVW, ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

をブロック対角ピボッティング手法（同名手法にはクラウト法とガウス消去法の関係に似た二つの考え方があり，ここでは後者）で解く．ただし，$\boldsymbol{A}$ は $n\times n$，バンド幅 h の対称バンド行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

$n > h \geq 0$ であること．

(2) パラメタ

A …… **入力**．係数行列 $\boldsymbol{A}$．
対称バンド行列用圧縮モード．
大きさ $n(h+1)-h(h+1)/2$ の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$，定数ベクトル $\boldsymbol{b}$ 及び解ベクトル $\boldsymbol{x}$ の次数 n.

NH …… **入力**． $\boldsymbol{A}$ のバンド幅 h.
演算後，その内容は保存されない．（使用上の注意④参照）

MH …… **入力**. 最大許容バンド幅 h_m（N > MH ≥ NH）
（使用上の注意④参照）

B …… **入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.

EPSZ …… **入力**．ピボットの相対零判定値（≥ 0.0）
0.0 のときは標準値が採用される．
（使用上の注意①参照）

ISW …… **入力**．制御情報．同一の係数行列を持つ $l(\geq 1)$ 組の方程式を解くとき，次のとおり指定する．
ISW=1 のとき，1 組目の方程式を解く．
ISW=2 のとき，2 組目以降の方程式を解く．
ただし，このとき $\boldsymbol{B}$ の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．（使用上の注意②参照）

VW …… **作業領域**. 大きさ $n(h_m+1)-h_m(h_m+1)/2$ の 1 次元配列．（使用上の注意⑤参照）

IVW …… **作業領域**．大きさ $2n$ の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 LSBIX-1 参照.

表 LSBIX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い.	処理を打ち切る.
25000	演算中にバンド幅が許容最大値を超えた.	処理を打ち切る.
30000	NH < 0，NH > MH，MH ≥ N，EPSZ < 0.0 又は ISW ≠ 1，2 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II … AMACH，BMDMX，SBMDM，MGSSL
② FORTRAN 基本関数 … AMAX1，ABS，IDIM

b. 注意
① ピボットの相対零判定 EPSZ に 10^{-s} を設定したとすると，これは次の意味を持っている．すなわち，ブロック対角ピボッティング手法による MDM^T 分解の過程でピボットの値（1×1 若しくは 2×2 ピボットが作る小行列式）が，10 進 s 桁以上の桁落ちを生じた場合に，そのピボットの値を相対的に零と見なし，ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸めの誤差の単位を u としたとき，$16\cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ に極小の値を与えればよいが，その結果は保証されない．
② 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列の MDM^T 分解過程を省略するので計算時間が少なくなる．
③ 行列 $\boldsymbol{A}$ の行列式の求め方については使用例及び本サブルーチンが使用するサブルーチン SBMDM の該当項目を参照すること．
④ ピボッティングのための行と列の交換により，行列のバンド幅は一般に増大する．許容最大バンド幅 h_m はこの増大分を見越して適当に設定すること．
もしも MDM^T 分解の途中にバンド幅が h_m を超過する場合には，ICON = 25000 として処理が打ち切られる．
⑤ A と VW が同じ領域であっても差支えがないように考慮されているので，領域を節約するために A と VW を同じ領域にとることができる．

c. 使用例
同一の係数行列を持つ l 組の連立 1 次方程式を解き，行列式を計算する．領域節約のために A と VW を同じ配列とする．
$n \le 100$， $h \le h_m \le 50$ の場合

```
C     **EXAMPLE**
      DIMENSION A(3825),B(100),IVW(200)
      READ(5,500) N,NH,MH
      WRITE(6,600) N,NH,MH
      NHP1=NH+1
      NT=(N+N-NH)*NHP1/2
      READ(5,510) (A(J),J=1,NT)
      READ(5,520) L
      EPSZ=1.0E-6
      ISW=1
      DO 10 K=1,L
      IF(K.GE.2) ISW=2
      READ(5,510) (B(I),I=1,N)
      CALL LSBIX(A,N,NH,MH,B,EPSZ,ISW,A,
     *           IVW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,620) (B(I),I=1,N)
   10 CONTINUE
      DET=1.0
      J=1
      I=1
   20 M=IVW(I)
      IF(M.NE.I) GO TO 30
      DET=A(J)*DET
      J=MIN0(I,MH)+1+J
      I=I+1
      GO TO 40
   30 JJ=MIN0(I,MH)+1+J
      DET=(A(J)*A(JJ)-A(JJ-1)*A(JJ-1))*DET
      J=MIN0(I+1,MH)+1+JJ
      I=I+2
   40 IF(I.LE.N) GO TO 20
      WRITE(6,630) DET
      STOP
  500 FORMAT(3I4)
  510 FORMAT(4E15.7)
  520 FORMAT(I4)
  600 FORMAT('1'/10X,'N=',I3,5X,'NH=',I3,
     * 5X,'MH=',I3)
  610 FORMAT('0',10X,'ICON=',I5)
  620 FORMAT(11X,'SOLUTION VECTOR'/
     * (15X,5E15.6))
  630 FORMAT('0',10X,
     * 'DETERMINANT OF COEFFICIENT ',
     * 'MATRIX=',E15.6)
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く．

a. 係数行列 $\boldsymbol{A}$ の MDM$^{\mathrm{T}}$ 分解（ブロック対角ピボッティング手法）
ブロック対角ピボッティング手法により，係数行列 A を MDM$^{\mathrm{T}}$ 分解する．

$$\boldsymbol{PAP}^{\boldsymbol{T}} = \boldsymbol{MDM}^{\boldsymbol{T}} \tag{4.2}$$

ただし，$\boldsymbol{P}$ はピボッティングによる行の入換えを行う置換行列，$\boldsymbol{M}$ は単位下バンド行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列である．この計算はサブルーチン SBMDM を用いて行う．

b. 求解
連立 1 次方程式 (4.1) を解くことは，

$$\boldsymbol{P}^{-1}\boldsymbol{MDM}^{\boldsymbol{T}}\boldsymbol{P}^{-\boldsymbol{T}}\boldsymbol{x} = \boldsymbol{b} \tag{4.3}$$

を解くことと同値であり，この方程式は

$$\boldsymbol{Mx}^{(1)} = \boldsymbol{Pb} \tag{4.4}$$
$$\boldsymbol{Dx}^{(2)} = \boldsymbol{x}^{(1)} \tag{4.5}$$
$$\boldsymbol{M}^{\boldsymbol{T}}\boldsymbol{x}^{(3)} = \boldsymbol{x}^{(2)} \tag{4.6}$$
$$\boldsymbol{P}^{-\boldsymbol{T}}\boldsymbol{x} = \boldsymbol{x}^{(3)} \tag{4.7}$$

として，前進代入，後退代入及び簡単な計算により解くことができる．この計算はサブルーチン BMDMX を用いて行う．

なお，詳細については，参考文献 [9] 及び [10] を参照すること．

A52-31-0101 LSBX, DLSBX

正値対称バンド行列の連立 1 次方程式 （変形コレスキー法）
CALL LSBX (A, N, NH, B, EPSZ, ISW, ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

を変形コレスキー法で解く．ただし $\boldsymbol{A}$ は $n \times n$，上，下バンド幅 h の正値対称バンド行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

$n > h \geq 0$ であること．

(2) パラメタ

A …… **入力**．係数行列 $\boldsymbol{A}$．
演算後，内容は保存されない．
対称バンド行列用圧縮モード
大きさ $n(h+1)-h(h+1)/2$ の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$ の次数 n．

NH …… **入力**．バンド幅 h．

B …… **入力**．定数ベクトル $\boldsymbol{b}$．
出力．解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

EPSZ …… **入力**．ピボットの相対零判定値（≥ 0.0）
0.0 のときは標準値が採用される．
（使用上の注意③ 参照）

ISW …… **入力**．制御情報．
同一の係数行列を持つ $l(\geq 1)$ 組の方程式を解くとき，次のとおり指定する．
ISW=1 のとき，1 組目の方程式を解く．
ISW=2 のとき，2 組目以降の方程式を解く．
ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．
（使用上の注意④参照）

ICON …… **出力**．コンディションコード．
表 LSBX-1 参照．

表 LSBX-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	ピボットが負となった． 係数行列が正値でない．	処理は続行する．
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い．	処理を打ち切る．
30000	NH < 0，NH ≥ N，EPSZ < 0.0 又は ISW ≠ 1，2 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，BDLX，SBDL，MGSSL

② FORTRAN 基本関数 … ABS

b. 注意

① 本サブルーチンにより得られた解 $\boldsymbol{x}$ は，続けてサブルーチン LSBXR を呼び出すことにより改良することができる．

② 本サブルーチンではバンド部分の外側にある要素にかかわる計算は省略しているので，正値対称行列用のサブルーチン LSX よりも処理が速い．

③ ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，変形コレスキー法による LDL^T 分解の過程でピボットの値が 10 進 s 桁以上の桁落ちを生じた場合にそのピボットを相対的に零と見なし，ICON = 20000 として処理を打ち切る．EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない．

④ 同一係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列 $\boldsymbol{A}$ の LDL^T 分解過程を省略するので計算時間が少なくなる．

⑤ 分解過程でピボットが負となった場合，係数行列は正値ではない．本サブルーチンでは，このとき処理は続行するが ICON = 10000 とする．ただし，ピボッティングを行っていないため計算誤差は大きい可能性があるので考慮すること．

⑥ 係数行列の行列式は，演算後の配列 A の n 個の対角線上の値を掛け合わせ，その逆数をとることにより得られる．ただし，配列 A は対称バンド行列用圧縮モードであることに注意すること．

⑦ 演算後の配列 A の内容はサブルーチン SBDL と同一である．

c. 使用例

同一係数を持つ l 組の n 元の連立 1 次方程式を解く．$n \leq 100$，$h \leq 50$ の場合．

```
C       **EXAMPLE**
        DIMENSION A(3825),B(100)
        READ(5,500) N,NH
        WRITE(6,600) N,NH
        NH1=NH+1
        NT=N*NH1-NH*NH1/2
        READ(5,510) (A(I),I=1,NT)
        READ(5,500) L
        K=1
        ISW=1
        EPSZ=1.0E-6
   10   READ(5,510) (B(I),I=1,N)
        CALL LSBX(A,N,NH,B,EPSZ,ISW,ICON)
        WRITE(6,610) ICON
        IF(ICON.GE.20000) STOP
        WRITE(6,620) (B(I),I=1,N)
        IF(K.EQ.L) GO TO 20
        K=K+1
        ISW=2
        GO TO 10
   20   DET=A(1)
        K=1
        DO 30 I=2,N
        K=K+MIN0(I,NH1)
        DET=DET*A(K)
   30   CONTINUE
        DET=1.0/DET
        WRITE(6,630) DET
        STOP
  500   FORMAT(2I5)
  510   FORMAT(4E15.7)
  600   FORMAT('1'/10X,'N=',I3,'NH=',I3)
  610   FORMAT('0',10X,'ICON=',I5)
  620   FORMAT(11X,'SOLUTION VECTOR'/
       * (15X,5E17.8))
  630   FORMAT('0',10X,
       * 'DETERMINANT OF COEFFICIENT MATRIX='
       * ,E17.8)
        END
```

(4) 手法概要

正値対称バンド行列 $\boldsymbol{A}$ を係数に持つ連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く．

a. 係数行列 $\boldsymbol{A}$ の LDT^T 分解（変形コレスキー法）

変形コレスキー法により，係数行列 $\boldsymbol{A}$ を LDT^T 分解する．

$$\boldsymbol{A} = \boldsymbol{LDL}^T \tag{4.2}$$

ただし，$\boldsymbol{L}$ は単位下バンド行列，$\boldsymbol{D}$ は対角行列である．この計算はサブルーチン SBDL を用いて行う．

b. 求解（前進代入，後退代入）

連立 1 次方程式

$$\boldsymbol{LDL}^T\boldsymbol{x} = \boldsymbol{b} \tag{4.3}$$

を解く．この計算はサブルーチン BDLX を用いて行う．

なお，詳細については，参考文献 [7] を参照すること．

A52-31-0401 LSBXR, DLSBXR

正値対称バンド行列の連立 1 次方程式の解の反復改良
CALL LSBXR (X, A, N, NH, FA, B, VW, ICON)

(1) 機能

$n\times n$，上，下バンド幅 h の正値対称バンド行列 $\boldsymbol{A}$ の連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (1.1)$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は (1.2) のとおり LDT^{T} 分解されていること．

$$\boldsymbol{A} = \boldsymbol{LDL}^{T} \qquad (1.2)$$

ここで，$\boldsymbol{L}$ と $\boldsymbol{D}$ はそれぞれ $n\times n$ の下バンド幅 h の単位下バンド行列と対角行列である．

$n > h \geq 0$ であること．

(2) パラメタ

X ····· **入力**．近似解ベクトル $\tilde{\boldsymbol{x}}$．
出力．反復改良された解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

A ····· **入力**．係数行列 $\boldsymbol{A}$．
対称バンド行列用圧縮モード．
大きさ $n(h+1) - h(h+1)/2$ の 1 次元配列．

N ····· **入力**．係数行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意②参照）

NH ····· **入力**．上，下バンド幅 h．

FA ····· **入力**．行列 $\boldsymbol{L}$ と行列 $\boldsymbol{D}^{-1}$．
図 LSBXR-1 参照．
対称バンド行列用圧縮モード．
大きさ $n(h+1) - h(h+1)/2$ の 1次元配列．

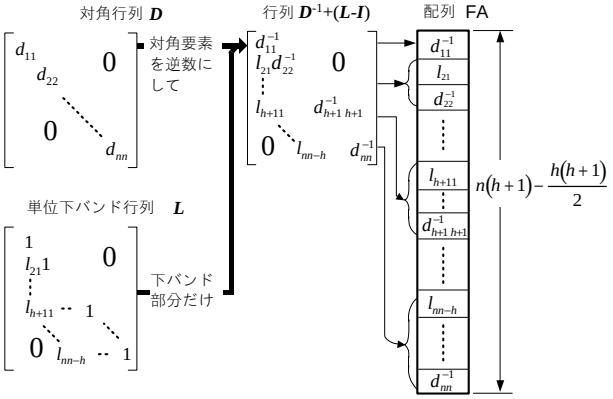
B····· **入力**．定数ベクトル $\boldsymbol{b}$．
大きさ n の 1 次元配列．

VW····· **作業領域**．大きさ n の 1 次元配列．

ICON····· **出力**．コンディションコード．
表 LSBXR-1 参照．

表 LSBXR-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	係数行列が正値でなかった．	処理は続行する．
25000	係数行列の条件が悪く，収束条件を満足しなかった．	処理を打ち切る．（収束条件については手法概要 b. 参照のこと）
30000	N = 0，NH < 0 又は NH ≥ |N| であった．	処理を打ち切る．



[注意] 配列 FA に行列 $\boldsymbol{D}^{-1}+(\boldsymbol{L}-\boldsymbol{I})$ の対角部分及び下バンド部分を，対称バンド行列用圧縮モードで格納する．

図 LSBXR-1 行列 $\boldsymbol{L}$ 及び $\boldsymbol{D}^{-1}$ の格納方法

(3) 使用上の注意

a. 使用する副プログラム

① SSL II··· AMACH，BDLX，MSBV，MGSSL

② FORTRAN 基本関数 ··· ABS

b. 注意

① 本サブルーチンは，サブルーチン LSBX により求められた近似解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである．
したがって，本サブルーチンを呼び出す前にサブルーチン LSBX を呼び出して近似解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された解を得ることができる．
この場合，パラメタ X，FA には直前に呼び出したサブルーチン LSBX の結果をそのまま入力すること．ただし，本サブルーチンでは係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LSBX を呼ぶに先立ってそれらを保存しておく必要がある．
具体的には使用例を参照されたい．

② N = -n と指定すれば，サブルーチン LSBX により得られた近似解 $\tilde{\boldsymbol{x}}$ の推定精度（相対誤差）を得ることができる．
この場合，本サブルーチンでは解の反復改良は行わず，単に相対誤差を計算し，VW(1) に出力する．精度の推定については，手法概要 c. を参照されたい．

c. 使用例

n 元の連立 1 次方程式の近似解 $\tilde{\boldsymbol{x}}$ をサブルーチン LSBX により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する．

$n \leq 100$，$h \leq 50$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(3825),FA(3825),X(100),
     *          B(100),VW(100)
   10 READ(5,500) N,NH
      IF(N.EQ.0) STOP
      NH1=NH+1
      NT=N*NH1-NH*NH1/2
      READ(5,510) (A(I),I=1,NT)
      READ(5,510) (B(I),I=1,N)
      WRITE(6,610) (I,B(I),I=1,N)
      DO 20 I=1,N
   20 X(I)=B(I)
      DO 30 I=1,NT
   30 FA(I)=A(I)
      EPSZ=0.0E0
      ISW=1
      CALL LSBX(FA,N,NH,X,EPSZ,ISW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL LSBXR(X,A,N,NH,FA,B,VW,ICON)
      WRITE (6,630) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,640) (I,X(I),I=1,N)
      DET=FA(1)
      L=1
      DO 40 I=2,N
      L=L+MIN0(I,NH1)
   40 DET=DET*FA(L)
      DET=1.0/DET
      WRITE(6,650) DET
      STOP
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  610 FORMAT(///10X,'CONSTANT VECTOR'/
     * (10X,4('(',I3,')',E17.8)))
  620 FORMAT('0','LSBX  ICON=',I5)
  630 FORMAT('0','LSBXR ICON=',I5)
  640 FORMAT('0',10X,'SOLUTION VECTOR'/
     * (10X,4('(',I3,')',E17.8)))
  650 FORMAT(///10X,
     * 'DETERMINANT OF COEFFICIENT ',
     * 'MATRIX=',E17.8)
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える．

a. 反復改良の考え方

反復改良とは，(4.1) の解 $\boldsymbol{x}$ の s 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)} = \tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$
$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{4.3}$$
$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{4.4}$$
$$s=1,2,\cdots$$

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$（s = 1，2，…）を逐次求めていくことである．

なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる．

ただし，係数行列 $\boldsymbol{A}$ の条件が非常に悪い場合には改良されないことがある．

（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順

サブルーチン LSBX で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする．

本サブルーチンでは，以下を繰り返す．

- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MSBV を用いて行う．
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはサブルーチン BDLX を用いて行う
- 修正された近似解 $\boldsymbol{x}^{(s+1)}$ を (4.4) から求める．

収束判定法は，以下のとおりである．

ここで丸め誤差の単位 u としたとき，s 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty / \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2\cdot u \tag{4.5}$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $\boldsymbol{x}^{(s+1)}$ を解として採用する．

ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s+1)}\right\|_\infty} > \frac{1}{2}\cdot\frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON = 25000 として処理を打ち切る．

c. 近似解の精度推定

近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)}$（$= \boldsymbol{x}^{(1)} - \boldsymbol{x}$）とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_\infty / \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ である．いま反復法が収束するならば，$\boldsymbol{e}^{(1)}$ と $\boldsymbol{d}^{(1)}$ はほぼ等しいものと考えられるから，近似解の相対誤差は $\left\|\boldsymbol{d}^{(1)}\right\|_\infty / \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ によって推定できる．（3.4 (2) f. 近似解の精度推定 参照）

なお，詳細については，文献 [1]，[3] 及び [5] を参照すること．

A22-21-0101 LSIX,DLSIX

実対称行列の連立 1 次方程式（ブロック対角ピボッティング手法）
CALL LSIX (A,N,B,EPSZ,ISW,VW,IP,IVW,ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{AX} = \boldsymbol{b} \qquad (1.1)$$

をブロック対角ピボッティング手法（同名手法にはクラウト法とガウス消去法の関係に似た二つの考え方があり，ここでは前者）で解く．ただし，$\boldsymbol{A}$ は $n\times n$ 対称行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

$n \geq 1$ であること．

(2) パラメタ

A …… **入力**．係数行列 $\boldsymbol{A}$．
演算後，内容は保存されない．
対称行列用圧縮モード．
大きさ $n\,(n+n)\,/\,2$ の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$，定数ベクトル $\boldsymbol{b}$ 及び解ベクトル $\boldsymbol{x}$ の次数 n．

B …… **入力**．定数ベクトル $\boldsymbol{b}$．
出力．解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

EPSZ …… **入力**．ピボットの相対零判定値（≥ 0.0）．
0.0 のときは標準値が採用される．
（使用上の注意①参照．）

ISW …… **入力**．制御情報．
同一の係数行列を持つ l（≥1）組の方程式を解くとき，次のとおり指定する．
ISW=1 のとき，1 組目の方程式を解く．
ISW=2 のとき，2 組目以降の方程式を解く．ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．
（使用上の注意②参照．）

VW …… **作業領域**．大きさ $2n$ の 1 次元配列．

IP …… **作業領域**．大きさ n の 1 次元配列．

IVW …… **作業領域**．大きさ n の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 LSIX-1 参照．

表 LSIX-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い．	処理を打ち切る．
30000	N < 1，EPSZ < 0.0 又は ISW ≠ 1，2 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… MDMX，AMACH，SMDM，MGSSL，USCHA

② FORTRAN 基本関数… ABS，SQRT，IABS，ISIGN

b. 注意

① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，ブロック対角ピボッティング手法による $\mathrm{MDM^T}$ 分解の過程でピボットの値（1×1 若しくは 2×2 ピボットが成す小行列の行列式）が，10 進 s 桁以上の桁落ちを生じた場合に，そのピボットの値を相対的に零と見なし，ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸め誤差の単位を u としたとき，$16\cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない．

② 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列の $\mathrm{MDM^T}$ 分解過程を省略するので計算時間が少なくなる．

③ 行列 $\boldsymbol{A}$ の行列式の求め方については使用例及び本サブルーチンが使用するサブルーチン SMDM の該当項目を参照すること．

④ 本サブルーチンでは，行列の対称性を生かし分解中もそれを保持するので領域が節約できる．

c. 使用例

同一の係数行列を持つ l 組の連立 1 次方程式を求める．また，係数行列の行列式も求める．
$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),B(100),VW(200),
     *          IP(100),IVW(100)
      CHARACTER*4 IA,IB,IX
      DATA IA,IB,IX/'A   ','B   ','X   '/
      READ(5,500) N,L
      NT=(N*(N+1))/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,600) N
      CALL PSM(IA,1,A,N)
      M=1
      ISW=1
      EPSZ=1.0E-6
   10 READ(5,510) (B(I),I=1,N)
      CALL PGM(IB,1,B,N,N,1)
      CALL LSIX(A,N,B,EPSZ,ISW,VW,IP,IVW,
     *          ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      CALL PGM(IX,1,B,N,N,1)
      IF(M.GE.L) GO TO 20
      M=M+1
      ISW=2
      GO TO 10
```

```
 20 DET=1.0
    I=1
    J=1
 30 IF(IP(J+1).GT.0) GO TO 40
    DET=DET*(A(I)*A(I+J+1)-A(I+J)
   *    *A(I+J))
    J=J+2
    I=I+J-1+J
    GO TO 50
 40 DET=DET*A(I)
    J=J+1
    I=I+J
 50 IF(J.LT.N) GO TO 30
    IF(J.EQ.N) DET=DET*A(I)
    WRITE(6,620) DET
    STOP
500 FORMAT(2I3)
510 FORMAT(4E15.7)
600 FORMAT('1'/
   * 6X,'LINEAR EQUATIONS AX=B'/
   * 6X,'ORDER=',I4)
610 FORMAT(' ',5X,'ICON OF LSIX=',I6)
620 FORMAT(' ',5X,'DETERMINANT OF A=',
   * E14.7)
    END
```

本使用例中のサブルーチン PSM 及び PGM は，実対称行列及び実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている．

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く．

a. 係数行列 $\boldsymbol{A}$ の $\mathrm{MDM}^{\mathrm{T}}$ 分解（ブロック対角ピボッティング手法）
ブロック対角ピボッティング手法により，係数行列 $\boldsymbol{A}$ を $\mathrm{MDM}^{\mathrm{T}}$ 分解する．

$$\boldsymbol{PAP}^{\boldsymbol{T}} = \boldsymbol{MDM}^{\boldsymbol{T}} \tag{4.2}$$

ただし，$\boldsymbol{P}$ はピボッティングによる行の入換えを行う置換行列，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列である．この計算はサブルーチン SMDM を用いて行う．

b. 求解
連立 1 次方程式 (4.1) を解くことは，

$$\boldsymbol{P}^{-1}\boldsymbol{MDM}^{\boldsymbol{T}}\left(\boldsymbol{P}^{\boldsymbol{T}}\right)^{-1}\boldsymbol{x} = \boldsymbol{b} \tag{4.3}$$

を解くことと同値であり，この方程式は，

$$\boldsymbol{M}\boldsymbol{x}^{(1)} = \boldsymbol{P}\boldsymbol{b} \tag{4.4}$$
$$\boldsymbol{D}\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} \tag{4.5}$$
$$\boldsymbol{M}^{\boldsymbol{T}}\boldsymbol{x}^{(3)} = \boldsymbol{x}^{(2)} \tag{4.6}$$
$$\left(\boldsymbol{P}^{\boldsymbol{T}}\right)^{-1}\boldsymbol{x} = \boldsymbol{x}^{(3)} \tag{4.7}$$

として，前進代入，後退代入及び簡単な計算により解く．この計算はサブルーチン MDMX を用いて行う．

なお，詳細については，参考文献 [9] 及び [10] を参照すること．

A22-21-0401 LSIXR, DLSIXR

実対称行列の連立 1 次方程式の解の反復改良
CALL LSIXR (X, A, N, FA, B, IP, VW, ICON)

(1) 機能

$n \times n$ の対称行列 $\boldsymbol{A}$ の連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (1.1)$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は (1.2) のとおり MDM^T 分解されていること．

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T \qquad (1.2)$$

ここで，$\boldsymbol{P}$ はピボッティングによる行の入換えを行う置換行列，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列で $d_{k+1,k} \neq 0$ なら $m_{k+1,k} = 0$ である．ただし，$\boldsymbol{M} = (m_{ij})$, $\boldsymbol{D} = (d_{ij})$

$n \geq 1$ であること．

(2) パラメタ

X ····· **入力**．近似解ベクトル $\tilde{\boldsymbol{x}}$．
出力．反復改良された解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

A ····· **入力**．係数行列 $\boldsymbol{A}$．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．

N ····· **入力**．係数行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意②参照）

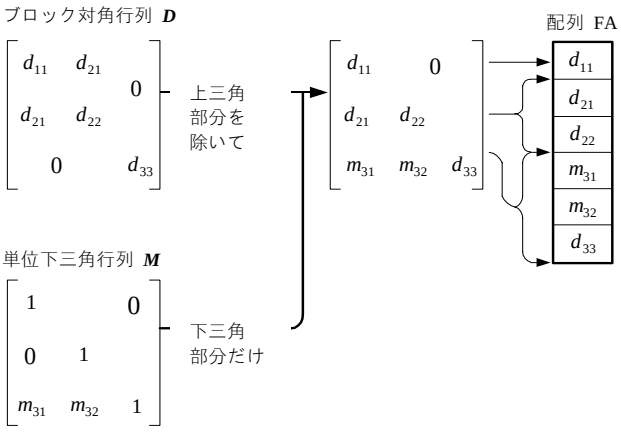
FA ····· **入力**．行列 $\boldsymbol{M}$ と行列 $\boldsymbol{D}$
図 LSIXR-1 参照．
大きさ $n(n+1)/2$ の 1 次元配列．

B····· **入力**．定数ベクトル $\boldsymbol{b}$．
大きさ n の 1 次元配列．

IP····· **入力**．ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．
大きさ n の 1 次元配列．
（使用上の注意①参照）

VW····· **作業領域**．大きさ n の 1 次元配列．

ICON····· **出力**．コンディションコード．
表 LSIXR-1 参照．



[注意] 行列 $\boldsymbol{D} + (\boldsymbol{M}-\boldsymbol{I})$ の対角部分及び下三角部分を，対称行列用圧縮モードで 1 次元配列 FA に格納する．ここで，$\boldsymbol{D}$ は次数 2 と 1 のブロックからなる場合である．

図 LSIXR-1 行列 $\boldsymbol{D}$ 及び $\boldsymbol{M}$ の格納方法

表 LSIXR-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	係数行列が正則でなかった．	処理を打ち切る．
25000	係数行列の条件が悪く，収束条件を満足しなかった．	処理を打ち切る．（収束条件については手法概要 b. 参照のこと）
30000	N = 0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II··· AMACH，MDMX，MSV，MGSSL

② FORTRAN 基本関数 ··· IABS，ABS

b. 注意

① 本サブルーチンは，サブルーチン LSIX により求められた近似解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである．
したがって，本サブルーチンを呼び出す前にサブルーチン LSIX を呼び出して近似解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された解を得ることができる．
この場合，パラメタ X，FA，IP には直前に呼び出したサブルーチン LSIX の結果をそのまま入力すること．ただし，本サブルーチンでは係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LSIX を呼ぶに先立ってそれらを保存しておく必要がある．
具体的には使用例を参照されたい．

② N = −*n* と指定すれば，サブルーチン LSIX により得られた近似解 $\tilde{\boldsymbol{x}}$ の推定精度（相対誤差）を得ることができる．
この場合，本サブルーチンでは解の反復改良は行わず，単に相対誤差を計算し，VW(1) に出力する．
精度の推定については，手法概要 c. を参照されたい．

c. 使用例
*n*元の連立 1 次方程式の近似解 $\tilde{\boldsymbol{x}}$ をサブルーチン LSIX により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する．
$n \leq 100$ の場合

```
C      **EXAMPLE**
       DIMENSION A(5050),FA(5050),X(100),
      *   B(100),VW(200),IP(100),IVW(100)
   10 READ(5,500) N
       IF(N.EQ.0) STOP
       NT=N*(N+1)/2
       READ(5,510) (A(I),I=1,NT)
       READ(5,510) (B(I),I=1,N)
       DO 20 I=1,N
       X(I)=B(I)
   20 CONTINUE
       DO 30 I=1,NT
   30 FA(I)=A(I)
       EPSZ=0.0E0
       ISW=1
       CALL LSIX(FA,N,X,EPSZ,ISW,VW,IP,IVW,
      *          ICON)
       WRITE(6,620) ICON
       IF(ICON.GE.20000) GO TO 10
       CALL LSIXR(X,A,N,FA,B,IP,VW,ICON)
       WRITE(6,630) ICON
       IF(ICON.GE.20000) STOP
       WRITE(6,640) (I,X(I),I=1,N)
       GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  620 FORMAT('0',10X,'LSIX  ICON=',I5)
  630 FORMAT('0',10X,'LSIXR ICON=',I5)
  640 FORMAT('0',10X,'SOLUTION VECTOR'/
     * (10X,4('(',I3,')',E17.8)))
       END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える．

a. 反復改良の考え方
反復改良とは，(4.1) の解 $\boldsymbol{x}$ の *s* 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)} = \tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$
$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{4.3}$$
$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{4.4}$$
$$s = 1,2,\cdots$$

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$（s = 1，2，…）を逐次求めていくことである．
なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる．
ただし，係数行列 $\boldsymbol{A}$ の条件が非常に悪い場合には改良れないことがある．
（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順
サブルーチン LSIX で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする．
本サブルーチンでは，以下を繰り返す．

- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MSV を用いて行う．
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはサブルーチン MDMX を用いて行う
- 修正された近似解 $x^{(s+1)}$ を (4.4) から求める．

収束判定法は，以下のとおりである．
ここで丸め誤差の単位を u としたとき，*s* 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty / \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2 \cdot u \tag{4.5}$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $x^{(s+1)}$ を解として採用する．
ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s+1)}\right\|_\infty} > \frac{1}{2} \cdot \frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON = 25000 として処理を打ち切る．

c. 近似解の精度推定
近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)} (= \boldsymbol{x}^{(1)} - \boldsymbol{x})$ とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_\infty / \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ である．いま反復法が収束するならば，$\boldsymbol{e}^{(1)}$ と $\boldsymbol{d}^{(1)}$ はほぼ等しいものと考えられるから，近似解の相対誤差は $\left\|\boldsymbol{d}^{(1)}\right\|_\infty / \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ によって推定できる．（3.4 (2) f. 近似解の精度推定 参照）

なお，詳細については，文献 [1]，[3] 及び [5] を参照すること．

A52-31-0501 LSTX, DLSTX

正値対称 3 項行列の連立 1 次方程式 (変形コレスキー法)
CALL LSTX (D, SD, N, B, EPSZ, ISW, ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

を変形コレスキー法で解く．ただし，$\boldsymbol{A}$ は $n \times n$ の正値対称な実 3 項行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルであること．

$n \geq 1$ であること．

(2) パラメタ

D …… **入力**．係数行列 $\boldsymbol{A}$ の対角部分．
演算後，内容は保存されない．
図 LSTX-1 参照
大きさ n の 1 次元配列．

SD …… **入力**．係数行列 $\boldsymbol{A}$ の副対角部分．
演算後，内容は保存されない．
図 LSTX-1 参照
大きさ n -1 の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$，定数ベクトル $\boldsymbol{b}$ 及び解ベクトル $\boldsymbol{x}$ の次数 n.

B…… **入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列．

EPSZ …… **入力**．ピボットの相対零判定値（≥ 0.0）．
0.0 のときは標準値が採用される．
（使用上の注意①参照）

ISW …… **入力**．制御情報．
同一の係数行列を持つ l（≥1）組の方程式を解くとき，次のとおり指定する．
ISW = 1 のとき，1 組目の方程式を解く．
ISW = 2 のとき，2 組目以降の方程式を解く．
ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．
（使用上の注意②参照）

ICON…… **出力**．コンディションコード．
表 LSTX-1 参照．

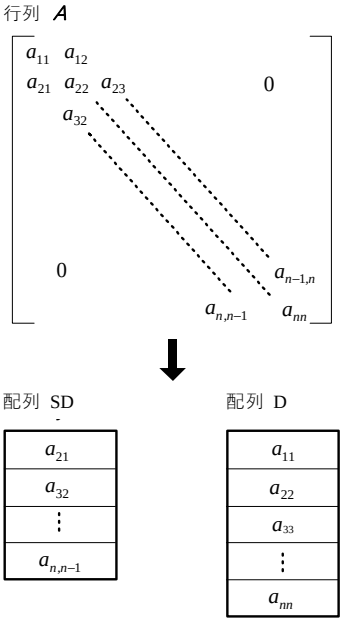


図 LSTX-1 配列 SD 及び D における行列 $\boldsymbol{A}$ の各要素の格納方法

表 LSTX-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	ピボットが負となった．係数行列は正値でない．	処理は続行する．
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い．	処理を打ち切る．
30000	N < 1，EPSZ < 0.0 又は ISW ≠ 1，2 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II… AMACH，MGSSL
② FORTRAN 基本関数 … ABS

b. 注意
① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，変形コレスキー法による LDL^T 分解の過程でピボットの値が 10進 s 桁以上の桁落ちを生じた場合にそのピボットを相対的に零と見なし，
ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸め誤差の単位を u としたとき，$16 \cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない．

② 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列 ***A*** の LU 分解過程を省略するので計算時間が少なくなる．
③ 行列 ***A*** の行列式は，n 個の配列要素 D(i)，$i = 1, \cdots, n$ を掛け合わせて得られる．
④ 分解過程でピボットが負になった場合は，ピボッティングを行っていないので計算誤差は大きい可能性があるから注意すること．
⑤ 本サブルーチンでは，繰返し演算を行う箇所で行列の特徴を生かした工夫がなされており，通常の変形コレスキー法と比べて，数学的には演算数が同じであるにもかかわらず処理速度が速い．

c. 使用例
同一の係数行列を持つ l 組の n 元連立 1 次方程式を解く．$n \leq 100$ の場合．

```
C      **EXAMPLE**
       DIMENSION D(100),SD(99),B(100)
       CHARACTER*4 IA,IB,IX
       DATA IA,IB,IX/'A   ','B   ','X   '/
       READ(5,500) N,L
       NM1=N-1
       READ(5,510) (D(I),I=1,N),(SD(I),
      *            I=1,NM1)
       WRITE(6,600) N
       CALL PTM(IA,1,SD,D,SD,N)
       ISW=1
       M=1
   10  READ(5,510) (B(I),I=1,N)
       CALL PGM(IB,1,B,N,N,1)
       CALL LSTX(D,SD,N,B,0.0,ISW,ICON)
       IF(ICON.EQ.0) GO TO 20
       WRITE(6,610) ICON
       STOP
   20  CALL PGM(IX,1,B,N,N,1)
       IF(M.EQ.L) GO TO 30
       M=M+1
       ISW=2
       GO TO 10
   30  WRITE(6,620)
       STOP
  500  FORMAT(2I3)
  510  FORMAT(4E15.7)
  600  FORMAT('1',5X,
     *  'LINEAR EQUATIONS AX=B(TRIDIAGONAL)'
     *  /6X,'(POSITIVE,SYMMETRIC)'
     *  /6X,'ORDER=',I5)
  610  FORMAT(' ',5X,'ICON OF LSTX=',I6)
  620  FORMAT(' ',5X,'* NORMA END *')
       END
```

本使用例中のサブルーチン PTM 及び PGM は，実 3 項行列及び実行列を印刷するものである．プログラムは，サブルーチン LTX 及び MGSM の使用例に記載されている．

(4) 手法概要

正値対称な実 3 項行列を係数に持つ連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

を変形コレスキー法で解くことを考える．

行列 ***A*** の特性（正値対称な実 3 項行列）を生かせば，常に行列を変形コレスキー法で (4.2) のとおり LDL^T 分解することができる（LDL^T 分解）．

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T$$

ここで，***L*** はバンド幅 1 の単位下バンド行列，***D*** は正値な対角行列である．

よって，(4.1) を解くことは，

$$\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b} \tag{4.3}$$

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{D}^{-1}\boldsymbol{y} \tag{4.4}$$

を解くことに帰着する．

(4.3) 及び (4.4) は前進代入及び後退代入により簡単に解くことができる．

a. 変形コレスキー法
行列 ***A*** が正値対称な場合には常に (4.2) なる LDL^T 分解が可能であり，その分解方法は変形コレスキー法により，次の等式で与えられる．

$$l_{ij}d_j = a_{ij} - \sum_{k=1}^{j-1} l_{ik}d_k l_{jk},\ j = 1,\ldots,i-1 \tag{4.5}$$

$$d_i = a_{ii} - \sum_{k=1}^{i-1} l_{ik}d_k l_{ik},\ i = 1,\ldots,n \tag{4.6}$$

ここで，$\boldsymbol{A} = \left(a_{ij}\right), \boldsymbol{L} = \left(l_{ij}\right)$，$\boldsymbol{D} = \mathrm{diag}\ \left(d_i\right)$ である．

なお，行列 ***A*** が 3 項行列であるから，(4.5), (4.6) は (4.7)，(4.8) となる．

$$l_{i,i-1}d_{i-1} = a_{i,i-1} \tag{4.7}$$

$$d_i = a_{ii} - l_{i,i-1}d_{i-1}l_{i,i-1} \tag{4.8}$$

b. 本サブルーチンにおける手順
本サブルーチンでは，行列の特性を生かした手順により解を求めている．

・ LDL^T 分解

通常，行列 $\boldsymbol{A}$ を変形コレスキー法で LDL^T 分解する場合, (4.7) 及び (4.8) なる式より要素 $l_{i,i-1}$ と $d_i(i=1,...,n)$ を逐次求める．しかし，本サブルーチンでは，対称 3 項行列の特性を生かし，行列 $\boldsymbol{L}$ と $\boldsymbol{D}$ の各要素を左上方要素からと右下方要素から各々中央要素まで，同時に求めることにより繰返し回数を節約する（繰返し回数節約のための考慮は，後説される (4.13) 及び (4.16) を計算するときにもなされている）．つまり，(4.9)，(4.10)，(4.11)，及び (4.12) により要素 $l_{i,i-1}, d_i, l_{n-i+1,n-i+2}$ 及び d_{n-i+1}（i = 1，…，[(n+1) / 2]）を逐次求める．

$$l_{i,i-1}d_{i-1} = a_{i,i-1} \tag{4.9}$$

$$d_i = a_{ii} - l_{i,i-1}d_{i-1}l_{ij-1} \tag{4.10}$$

$$l_{n-i+1,n-i+2}d_{n-i+2} = a_{n-i+1,n-i+2} \tag{4.11}$$

$$d_{n-i+1} = a_{n-i+1,n-i+1} - l_{n-i+2,n-i+1}$$
$$d_{n-i+2}\,l_{n-i+2,n-i+1} \tag{4.12}$$

i = 1，…，[(n+1) / 2]

・ $\boldsymbol{Ly} = \boldsymbol{b}$ を解く（前進代入，後退代入）

通常は

$$y_i = b_i - \sum_{k=1}^{i-1} l_{ik}y_k, \quad i = 1,...,n \tag{4.13}$$

なる式で逐次求める．ただし，

$\mathbf{y}^{\mathbf{T}} = (y_1,...,y_n)$，$\boldsymbol{b}^T = (b_1,...,b_n)$ である．

本サブルーチンでは，(4.14) 及び (4.15) なる式より逐次求める．

$$y_i = b_i - l_{i,i-1}y_{i-1} \tag{4.14}$$

$$y_{n-i+1} = b_{n-i+1} - l_{n-i+1,n-i+2}y_{n-i+2} \tag{4.15}$$

i = 1，…，[(n+1) / 2]

・ $\boldsymbol{L}^T\boldsymbol{x} = D^{-1}\boldsymbol{y}$ を解く（前進代入，後退代入）

通常は，

$$x_i = y_i d^{-1} - \sum_{k=i+1}^{n} l_{ki}x_k, i = n,...,1 \tag{4.16}$$

なる式で逐次求める．ただし，$\mathbf{x}^T = (x_1,...,x_n)$，

$\boldsymbol{D}^{-1} = \mathrm{diag}\,(\boldsymbol{d}_i^{-1})$ である．

本サブルーチンでは，(4.17)，(4.18) 及び (4.19) なる式より逐次求める．

$$x_{[(n+1)/2]} = y_{[(n+1)/2]}d^{-1}_{[(n+1)/2]} \tag{4.17}$$

$$x_i = y_i d_i^{-1} - l_{i+1,i}x_{i+1} \tag{4.18}$$

$$x_{n-i+1} = y_{n-i+1}d^{-1}_{n-i+1} - l_{n-i+1,n-i}x_{n-i}$$
$$i = [(\mathrm{n}+1)/2]-1,...,1 \tag{4.19}$$

A22-51-0101 LSX, DLSX

正値対称行列の連立 1 次方程式 （変形コレスキー法）
CALL LSX (A, N, B, EPSZ, ISW, ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

を変形コレスキー法で解く．ただし，$\boldsymbol{A}$ は $n \times n$ の正値対称行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．$n \geq 1$ であること．

(2) パラメタ

A……**入力**．係数行列 $\boldsymbol{A}$．
演算後，内容は保存されない．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．

N……**入力**．係数行列 $\boldsymbol{A}$ の次数 n．

B……**入力**．定数ベクトル $\boldsymbol{b}$．
出力．解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

EPSZ ……**入力**．ピボットの相対零判定値（≥0.0）．
0.0 のときは標準値が採用される．
（使用上の注意②参照）

ISW ……**入力**．制御情報．
同一の係数行列を持つ $l(\geq 1)$ 組の方程式を解くとき，次のとおり指定する．
ISW=1 のとき，1 組目の方程式を解く．
ISW=2 のとき，2 組目以降の方程式を解く．
ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う．（使用上の注意③参照）

ICON……**出力**．コンディションコード．
表 LSX-1 参照．

表 LSX-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	ピボットが負となった． 係数行列は正値でない．	処理は続行する．
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い．	処理を打ち切る．
30000	N < 1，EPSZ < 0.0 又は ISW ≠ 1，2 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II… AMACH，LDLX，SLDL，MGSSL

② FORTRAN 基本関数… ABS

b. 注意

① 本サブルーチンにより得られた解 $\boldsymbol{x}$ は，続けてサブルーチン LSXR を呼び出すことにより改良することができる．

② ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，変形コレスキー法による $\mathrm{LDL}^{\mathrm{T}}$ 分解の過程でピボットの値が 10 進 s 桁以上の桁落ちを生じた場合にそのピボットを相対的に零と見なし，ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZへ極小の値を与えればよいが，その結果は保証されない．

③ 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列 $\boldsymbol{A}$ の $\mathrm{LDL}^{\mathrm{T}}$ 分解過程を省略するので計算時間が少なくなる．

④ 分解過程でピボットが負となった場合，係数行列 ICON = 10000 として処理は続行する．ただし，ピボッティングを行っていないため計算誤差は大きい可能性があるので考慮すること．

⑤ 係数行列の行列式は，演算後の配列 A の n 個の対角線上の値（$\boldsymbol{D}^{-1}$ の対角要素）を掛け合わせ，その逆数をとることにより得られる．ただし，配列 A は対称行列用圧縮モードであることに注意すること．

⑥ 正値対称バンド行列の場合，バンド部分の外側にある要素にかかわる計算を考慮するサブルーチン LSBX の方が速く解が求まる．

c. 使用例

同一の係数行列を持つ l 組の n 元連立 1 次方程式を解く．$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),B(100)
      READ(5,500) N
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,600) N
      READ(5,500) L
      M=1
      ISW=1
      EPSZ=1.0E-6
   10 READ(5,510) (B(I),I=1,N)
      CALL LSX(A,N,B,EPSZ,ISW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,620) (B(I),I=1,N)
      IF(L.EQ.M) GOTO 20
```

```
      M=M+1
      ISW=2
      GOTO 10
   20 DET=A(1)
      L=1
      DO 30 I=2,N
      L=L+I
      DET=DET*A(L)
   30 CONTINUE
      DET=1.0/DET
      WRITE(6,630) DET
      STOP
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'/10X,'ORDER=',I5)
  610 FORMAT('0',10X,'ICON=',I5)
  620 FORMAT(11X,'SOLUTION VECTOR'/
     * (15X,5E16.8))
  630 FORMAT('0',10X,
     * 'DETERMINANT OF COEFFICIENT MATRIX='
     * ,E16.8)
      END
```

(4) 手法概要

正値対称な係数行列 $\boldsymbol{A}$ を持つ連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4.1}$$

を次の手順で解く.

a. 係数行列 $\boldsymbol{A}$ の $\mathrm{LDL}^{\mathrm{T}}$ 分解（変形コレスキー法）
変形コレスキー法により, 係数行列 $\boldsymbol{A}$ を $\mathrm{LDL}^{\mathrm{T}}$ 分解する.

$$\boldsymbol{A} = \boldsymbol{LDL}^{T} \tag{4.2}$$

ただし，$\boldsymbol{L}$ は単位下三角行列，$\boldsymbol{D}$ は対角行列である．この計算はサブルーチン SLDL を用いて行う.

b. 求解（前進代入，後退代入）
連立 1 次方程式

$$\boldsymbol{LDL}^{T}\boldsymbol{x} = \boldsymbol{b} \tag{4.3}$$

を解く．この計算はサブルーチン LDLX を用いて行う.

なお，詳細については，参考文献 [2] を参照すること.

A22-51-0401 LSXR, DLSXR

正値対称行列の連立 1 次方程式の解の反復改良
CALL LSXR (X, A, N, FA, B, VW, ICON)

(1) 機能

$n \times n$ の正値対称行列 $\boldsymbol{A}$ の連立 1 次方程式

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{1.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復修正して改良する．

ただし，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．

なお，あらかじめ，係数行列 $\boldsymbol{A}$ は (1.2) のとおり LDL^T 分解されていること．

$$\boldsymbol{A} = \boldsymbol{LDL}^T \tag{1.2}$$

ここで，$\boldsymbol{L}$ と $\boldsymbol{D}$ はそれぞれ $n \times n$ の単位下三角行列と対角行列である．

$n \geq 1$ であること．

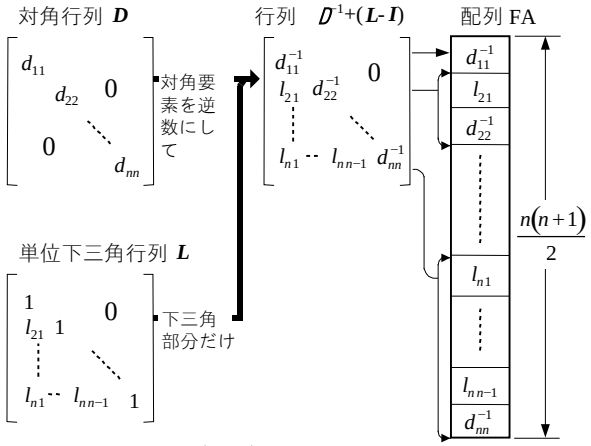
(2) パラメタ

X …… **入力**．近似解ベクトル $\tilde{\boldsymbol{x}}$．
出力．反復改良された解ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

A …… **入力**．係数行列 $\boldsymbol{A}$．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．

N …… **入力**．係数行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意②参照）

FA …… **入力**．行列 $\boldsymbol{L}$ と行列 $\boldsymbol{D}^{-1}$．
対称バンド行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．
図 LSXR-1 参照．



[注意] 行列 $\boldsymbol{D}^{-1}+(\boldsymbol{L}-\boldsymbol{I})$ の対角部分及び下三角部分を，対称行列用圧縮モードで 1 次元配列 FA に格納する．

図 LSXR-1 行列 $\boldsymbol{L}$ 及び $\boldsymbol{D}$ の格納方法

B …… **入力**．定数ベクトル $\boldsymbol{b}$．
大きさ n の 1 次元配列．

VW …… **作業領域**．
大きさ n の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 LSXR-1 参照．

表 LSXR-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	係数行列が正値でなかった．	処理は続行する．
25000	係数行列の条件が悪く，収束条件を満足しなかった．	処理を打ち切る．（収束条件については手法概要 b. 参照のこと）
30000	N = 0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II… AMACH，LDLX，MSV，MGSSL
② FORTRAN 基本関数 … ABS

b. 注意
① 本サブルーチンは，サブルーチン LSX により求められた近似解 $\tilde{\boldsymbol{x}}$ を反復修正して，その精度を改良するものである．
したがって，本サブルーチンを呼び出す前にサブルーチン LSX を呼び出して近似解 $\tilde{\boldsymbol{x}}$ を求め，続けて本サブルーチンを呼び出すことにより反復改良された解を得ることができる．
この場合，パラメタ X，FA には直前に呼び出したサブルーチン LSX の結果をそのまま入力すること．ただし，本サブルーチンでは係数行列 $\boldsymbol{A}$ と定数ベクトル $\boldsymbol{b}$ も必要とするためサブルーチン LSX を呼ぶに先立ってそれらを保存しておく必要がある．
具体的には使用例を参照されたい．

② N = -*n* と指定すれば，サブルーチン LSX により得られた近似解 $\tilde{\boldsymbol{x}}$ の推定精度（相対誤差）を得ることができる．
この場合，本サブルーチンでは解の反復改良は行わず，単に相対誤差を計算し，VW(1) に出力する．
精度の推定については，手法概要 c. を参照されたい．

c. 使用例
*n*元の連立 1 次方程式の近似解 $\tilde{\boldsymbol{x}}$ をサブルーチン LSX により求め，その $\tilde{\boldsymbol{x}}$ を本サブルーチンで反復改良する．
$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),FA(5050),X(100),
     *          B(100),VW(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      READ(5,510) (B(I),I=1,N)
      DO 20 I=1,N
      X(I)=B(I)
   20 CONTINUE
      DO 30 I=1,NT
   30 FA(I)=A(I)
      EPSZ=0.0E0
      ISW=1
      CALL LSX(FA,N,X,EPSZ,ISW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL LSXR(X,A,N,FA,B,VW,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,640) (I,X(I),I=1,N)
      DET=FA(1)
      L=1
      DO 40 I=2,N
      L=L+1
   40 DET=DET*FA(L)
      DET=1.0/DET
      WRITE(6,650) DET
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  620 FORMAT('0',10X,'LSX  ICON=',I5)
  630 FORMAT('0',10X,'LSXR ICON=',I5)
  640 FORMAT('0',10X,'SOLUTION VECTOR'/
     * (10X,4('(',I3,')',E17.8)))
  650 FORMAT(///10X,
     * 'DETERMINANT OF COEFFICIENT ',
     * 'MATRIX=',E17.8)
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

の近似解 $\tilde{\boldsymbol{x}}$ が与えられたとき，その解を反復改良することを考える．

a. 反復改良の考え方
反復改良とは，(4.1) の解 $\boldsymbol{x}$ の *s* 回目の近似解を $\boldsymbol{x}^{(s)}$ と表せば，$\boldsymbol{x}^{(1)} = \tilde{\boldsymbol{x}}$ として

$$\boldsymbol{r}^{(s)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(s)} \tag{4.2}$$
$$\boldsymbol{A}\boldsymbol{d}^{(s)} = \boldsymbol{r}^{(s)} \tag{4.3}$$
$$\boldsymbol{x}^{(s+1)} = \boldsymbol{x}^{(s)} + \boldsymbol{d}^{(s)} \tag{4.4}$$
$$s = 1, 2, \ldots$$

によって連立 1 次方程式 (4.1) の改良された近似解 $\boldsymbol{x}^{(s+1)}$（$s = 1$，2，…）を逐次求めていくことである．
なお，反復改良の原理から，(4.2) の計算を正確に行うなら，数値的に近似解 $\boldsymbol{x}^{(1)}$ の改良解が得られる．
ただし，係数行列 $\boldsymbol{A}$ の条件が非常に悪い場合には改良されないことがある．
（3.4 (2) e. 解の反復改良 参照）

b. 本サブルーチンにおける手順
サブルーチン LSX で最初の近似解 $\boldsymbol{x}^{(1)}$ は求められているものとする．
本サブルーチンでは，以下を繰り返す．

- 残差 $\boldsymbol{r}^{(s)}$ を (4.2) から計算する．これはサブルーチン MSV を用いて行う．
- 修正量 $\boldsymbol{d}^{(s)}$ を (4.3) から求める．これはサブルーチン LDLX を用いて行う．
- 修正された近似解 $\boldsymbol{x}^{(s+1)}$ を (4.4) から求める．

収束判定法は，以下のとおりである．
ここで丸め誤差の単位を *u* としたとき，*s* 回目の反復過程において

$$\left\|\boldsymbol{d}^{(s)}\right\|_\infty / \left\|\boldsymbol{x}^{(s+1)}\right\|_\infty < 2 \cdot u \tag{4.5}$$

なる関係を満たしたとき，反復改良は収束したものと見なし，そのときの $\boldsymbol{x}^{(s+1)}$ を解として採用する．
ただし，反復過程で

$$\frac{\left\|\boldsymbol{d}^{(s)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s+1)}\right\|_\infty} > \frac{1}{2} \cdot \frac{\left\|\boldsymbol{d}^{(s-1)}\right\|_\infty}{\left\|\boldsymbol{x}^{(s)}\right\|_\infty}$$

なる関係が生じた場合には，係数行列 $\boldsymbol{A}$ の条件が非常に悪く，反復改良は収束しないと見なし，ICON = 25000 として処理を打ち切る．

c. 近似解の精度推定

近似解 $\boldsymbol{x}^{(1)}$ における誤差を $\boldsymbol{e}^{(1)}\left(=\boldsymbol{x}^{(1)}-\boldsymbol{x}\right)$ とすれば，相対誤差は $\left\|\boldsymbol{e}^{(1)}\right\|_\infty / \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ である．いま反復法が収束するならば，$\boldsymbol{e}^{(1)}$ と $\boldsymbol{d}^{(1)}$ はほぼ等しいものと考えられるから，近似解の相対誤差は $\left\|\boldsymbol{d}^{(1)}\right\|_\infty / \left\|\boldsymbol{x}^{(1)}\right\|_\infty$ によって推定できる．（3.4 (2) f. 近似解の精度推定 参照）

なお，詳細については，文献 [1]，[3] 及び [5] を参照すること．

A52-11-0501 LTX, DLTX

実 3 項行列の連立 1 次方程式（ガウス消去法）
CALL LTX (SBD, D, SPD, N, B, EPSZ, ISW, IS, IP, VW, ICON)

(1) 機能

実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{1.1}$$

をガウス消去法で解く．ただし，$\boldsymbol{A}$ は $n \times n$ の正則な実 3 項行列，$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$は n 次元の解ベクトルである．$n \geq 1$ であること．

(2) パラメタ

SBD ····· **入力**．係数行列 $\boldsymbol{A}$ の下副対角部分．
演算後，内容は保存されない．
図 LTX-1 参照．
大きさ n -1 の 1 次元配列．

D ····· **入力**．係数行列 $\boldsymbol{A}$ の対角部分．
演算後，内容は保存されない．
図 LTX-1 参照．
大きさ n の 1 次元配列．

SPD ····· **入力**．係数行列 $\boldsymbol{A}$ の上副対角部分．
演算後，内容は保存されない．
図 LTX-1 参照．
大きさ n -1 の 1 次元配列．

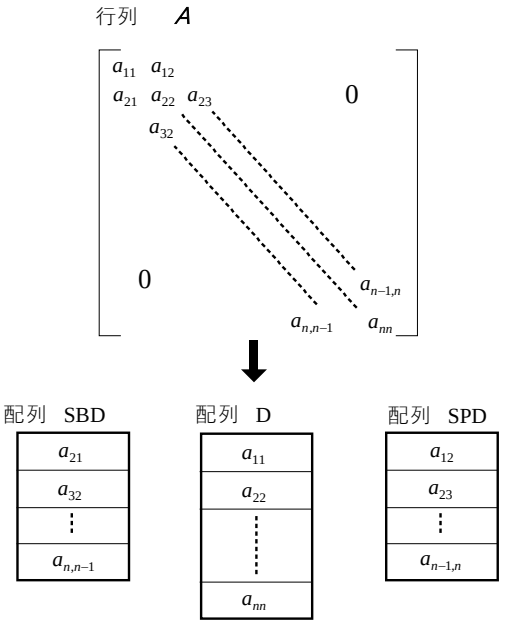


図 LTX-1 配列 SBD，D 及び SPD における
行列 $\boldsymbol{A}$ の各要素の格納方法

N ····· **入力**．係数行列 $\boldsymbol{A}$ の次数 n.

B ····· **入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.

EPSZ····· **入力**．ピボットの相対零判定値（≥0.0）.
0.0 のときは標準値が採用される.
（使用上の注意①参照）

ISW····· **入力**．制御情報.
同一の係数行列を持つ $l(\geq 1)$ 組の方程式を解くとき，次のとおり指定する.
ISW=1 のとき，1 組目の方程式を解く.
ISW=2 のとき，2 組目以降の方程式を解く．ただし，このとき B の値だけを新しい定数ベクトル $\boldsymbol{b}$ の値に変え，それ以外のパラメタはそのままで使う.
（使用上の注意②参照）

IS ····· **出力**．行列 $\boldsymbol{A}$ の行列式を求めるための情報.
（使用上の注意③参照）

IP ····· **作業領域**．大きさ n の 1 次元配列.

VW ····· **作業領域**．大きさ n の 1 次元配列.

ICON ····· **出力**．コンディションコード.
表 LTX-1 参照.

表 LTX-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	ピボットが相対的に零となった．係数行列が非正則の可能性が強い.	処理を打ち切る.
30000	N < 1 又は EPSZ < 0.0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ··· AMACH，MGSSL

② FORTRAN 基本関数 ··· AMAX1，ABS

b. 注意

① 本サブルーチンでは，ピボットの相対零判定の際，EPSZ の値を含んだ関係式にて行う（詳細手法概要参照）
EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である.
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ への極小の値を与えればよいが，その結果は保証されない.

② 同一の係数行列を持つ連立 1 次方程式をいくつか続けて解く場合には，2 回目以降 ISW = 2 として解くと，係数行列 $\boldsymbol{A}$ の LU 分解過程を省略するので計算時間が少なくなる. なお，この場合 IS の値は，ISW = 1 のときの値が保証される.

③ 行列 $\boldsymbol{A}$ の行列式は，n 個の配列要素 D(i)，i = 1, ···，n の積と IS の値を掛け合わせて得られる.

c. 使用例

同一の係数行列を持つ l 組の n 元連立 1 次方程式を解く.

$n \le 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION SBD(99),D(100),SPD(99),
     *          B(100),IP(100),VW(100)
      CHARACTER*4 NT1(6),NT2(4),NT3(4)
      DATA NT1/'CO  ','EF  ','FI  ',
     *         'CI  ','EN  ','T   '/,
     *     NT2/'CO  ','NS  ','TA  ',
     *         'NT  '/,
     *     NT3/'SO  ','LU  ','TI  ',
     *         'ON  '/
      READ(5,500) N,L
      NM1=N-1
      READ(5,510) (SBD(I),I=1,NM1),
     *   (D(I),I=1,N),(SPD(I),I=1,NM1)
      WRITE(6,600) N
      CALL PTM(NT1,6,SBD,D,SPD,N)
      ISW=1
      M=1
   10 READ(5,510) (B(I),I=1,N)
      CALL PGM(NT2,4,B,N,N,1)
      CALL LTX(SBD,D,SPD,N,B,0.0,ISW,
     *  IS,IP,VW,ICON)
      WRITE(6,610)ICON
      IF(ICON.NE.0)STOP
      CALL PGM(NT3,4,B,N,N,1)
      IF(M.EQ.L) GO TO 20
      M=M+1
      ISW=2
      GO TO 10
   20 WRITE(6,620)
      STOP
  500 FORMAT(3I2)
  510 FORMAT(4E15.7)
  600 FORMAT('1',5X,
     * 'LINEAR EQUATIONS(TRIDIAGONAL)'/
     * 6X,'ORDER=',I5)
  610 FORMAT(' ',5X,'ICON OF LTX=',I5)
  620 FORMAT(' ',5X,'** NORMAL END')
      END

      SUBROUTINE PTM(ICOM,L,SBD,D,SPD,N)
      DIMENSION SBD(1),D(N),SPD(1)
      CHARACTER*4 ICOM(L)
      WRITE(6,600) (ICOM(I),I=1,L)
      DO 30 I=1,N
      IF(I.NE.1) GO TO 10
      IC=1
      WRITE(6,610) I,IC,D(I),SPD(I)
      GO TO 30
   10 IC=I-1
      IF(I.NE.N) GO TO 20
      WRITE(6,610) I,IC,SBD(IC),D(I)
      GO TO 30
   20 WRITE(6,610)I,IC,SBD(IC),D(I),SPD(I)
   30 CONTINUE
      RETURN
  600 FORMAT(/10X,35A2)
  610 FORMAT(/5X,2(1X,I3),3(3X,E14.7))
      END
```

サブルーチン PTM 及び PGM は，実 3 項行列及び実行列を印刷するものである．サブルーチン PGM は，サブルーチン MGSM の使用例に記載されている.

(4) 手法概要

実 3 項行列 A を係数に持つ連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

を，部分ピボッティングを伴ったガウス消去法で解くことを考える.

通常, 行列は部分ピボッティングを行って単位下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ の積に分解することができる.

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U} \tag{4.2}$$

よって (4.1) を解くことは

$$\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b} \tag{4.3}$$
$$\boldsymbol{U}\boldsymbol{x} = \boldsymbol{y} \tag{4.4}$$

を解くことに帰着する.

$\boldsymbol{L}$ 及び $\boldsymbol{U}$ は三角行列であるから，(4.3) 及び (4.4) は前進代入及び後退代入により簡単に解くことができる.

a. ガウス消去法

ガウス消去法の k 段落目（$k = 1$，…，n-1）の行列を $\boldsymbol{A}^{(k)}$ と表す．ただし，$\boldsymbol{A}^{(1)} = \boldsymbol{A}$ とする.

k 段落目の消去過程は (4.5) で表現される.

$$\boldsymbol{A}^{(k+1)} = \boldsymbol{M}_k \boldsymbol{P}_k \boldsymbol{A}^{(k)} \tag{4.5}$$

ここで $\boldsymbol{P}_k$ は行列 $\boldsymbol{A}^{(k)}$ の k 列目のピボット行を選択するための置換行列である．M_k は置換された行列の k 列目の下副対角要素を消去するための行列であり，(4.6) で表現される.

$$\boldsymbol{M}_k = \begin{bmatrix} 1 & & & & \\ & \ddots & & 0 & \\ & & 1 & & \\ & & -m_{k+1,k} & 1 & \\ & 0 & & & \ddots \\ & & & & & 1 \end{bmatrix} \tag{4.6}$$

$$m_{k+1,k} = a_{k+1,k}^{(k)} / a_{kk}^{(k)}, A^{(k)} = \left(a_{\mathrm{ij}}^{(k)}\right)$$

消去過程が終了すると,

$$\boldsymbol{U} = \boldsymbol{A}^{(n)} = \boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\ldots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A}^{(1)} \tag{4.7}$$
$$\boldsymbol{L} = \left(\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\ldots\boldsymbol{M}_1\boldsymbol{P}_1\right)^{-1} \tag{4.8}$$

とおくと，(4.7) は (4.8) より

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$$

と表せる．ここで，$\boldsymbol{L}$ と $\boldsymbol{U}$ は単位下三角行列と上三角行列である.

b. 本サブルーチンにおける手順

・ 前進代入

(4.8) より (4.3) は (4.9) となる.

$$\boldsymbol{y} = \boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}...\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{b} \qquad (4.9)$$

(4.9) は次のように逐次求める.

$$\boldsymbol{y}^{(1)} = \boldsymbol{b}$$
$$\boldsymbol{y}^{(2)} = \boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{y}^{(1)}$$
$$\vdots$$
$$\boldsymbol{y}^{(n)} = \boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\boldsymbol{y}^{(n-1)}$$
$$\boldsymbol{y} = \boldsymbol{y}^{(n)}$$

本サブルーチンでは k 段落目（$k = 1, \cdots, n\text{-}1$）の消去過程において，行列 $\boldsymbol{L}$ の第 k 列，行列 $\boldsymbol{U}$ の第 k 行の各要素を (4.10)，(4.11) により求めている.

$$m_{k+1,k} = a^{(k)}_{k+1,k} / a^{(k)}_{kk} \qquad (4.10)$$
$$u_{kj} = a^{(k)}_{kj}, j = k, k+1, k+2 \qquad (4.11)$$

一方，行列 $\boldsymbol{A}^{(k+1)}$ の第 $k+1$ 行の要素は (4.12) のとおりとなる．ただし，$k+1$ 段目の消去の対象となる行列 $\boldsymbol{A}^{(k+1)}$ の他要素は行列 $\boldsymbol{A}^{(k)}$ の対応する要素と等しい値である.

$$a^{(k+1)}_{k+1,j} = a^{(k)}_{k+1} - a^{(k)}_{k,j}m^{(k)}_{k+1,k} \quad , j = k+1, k+2 \qquad (4.12)$$

なお，この消去過程に先立って，計算誤差を少なくするために，(4.10) におけるピボット要素 $\boldsymbol{a}^{(k)}_{kk}$ を次のように選択する.

すなわち，$V_l \cdot \left|a^{(k)}_{lk}\right| = \max_{l=k,k+1}\left(V_l \cdot \left|a^{(k)}_{lk}\right|\right)$ なる，スケーリングファクタ $\boldsymbol{V}_l$ を考慮した $a^{(k)}_{lk}$ をピボット要素とする．ここで，$\boldsymbol{V}_l$ は係数行列 $\boldsymbol{A}$ の第 l 行絶対値最大要素の逆数である.

選択されたピボット要素 $\boldsymbol{a}^{(k)}_{lk}$ が

$$\left|a^{(k)}_{lk}\right| < \max\left(\left|a_{ij}\right|\right) \cdot u$$

ここで $\boldsymbol{A} = \left(\boldsymbol{a}_{ij}\right)$, u : 丸め誤差の単位.

であるならば，行列 $\boldsymbol{A}$ は数値的に非正則であると見なし，ICON = 20000 として処理を打ち切る.

・ 後退代入

(4.4) は

$$x_k = \left(y_k - \sum_{j=k+1}^{k+2} u_{kj}x_j\right) / u_{kk}, k = n, ..., 1 \qquad (4.13)$$

なる式で逐次求める.

ただし，$\boldsymbol{U} = \left(u_{ij}\right)$, $\boldsymbol{x} = \left(x_1, x_2, ..., x_n\right)^T$ である.

A22-11-0602 LUIV, DLUIV

LU 分解された実行列の逆行列
CALL LUIV (FA, K, N, IP, ICON)

(1) 機能

LU 分解された $n\times n$ の実行列 $\boldsymbol{A}$ の逆行列 $\boldsymbol{A}^{-1}$ を求める.

$$\boldsymbol{A}^{-1}=\boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P}$$

ただし，$\boldsymbol{L}$，$\boldsymbol{U}$ はそれぞれ $n\times n$ の下三角行列と単位上三角行列であり，$\boldsymbol{P}$ は LU 分解におけるピボッティングによる行の入換えを示す置換行列である.

$n\geq 1$ であること.

(2) パラメタ

FA……**入力**. 行列 $\boldsymbol{L}$ と行列 $\boldsymbol{U}$.
出力. 逆行列 $\boldsymbol{A}^{-1}$.
図 LUIV-1 参照.
FA(K, N) なる 2 次元配列.

K……**入力**. 配列 FA の整合寸法（≥N）.

N……**入力**. 行列 $\boldsymbol{L}$，$\boldsymbol{U}$ の次数 n.

IP……**入力**. ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさ n の 1 次元配列.
（使用上の注意③参照）

ICON……**出力**. コンディションコード.
表 LUIV-1 参照.

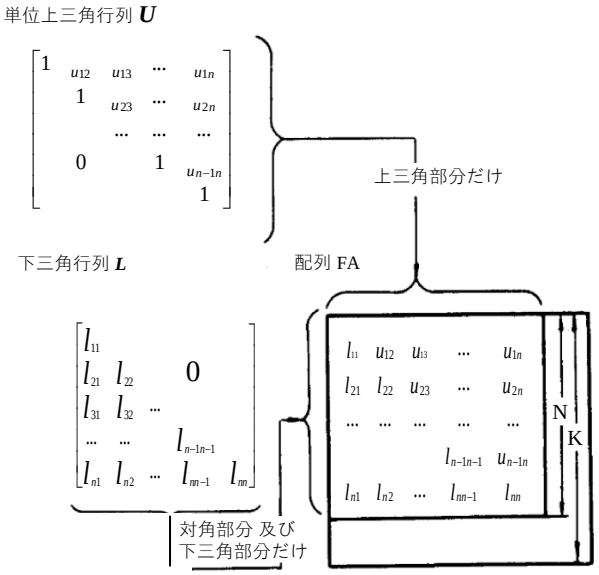


図 LUIV-1 配列 FA における $\boldsymbol{L}$ 及び $\boldsymbol{U}$ の各要素の格納方法

表 LUIV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	実行列が非正則であった.	処理を打ち切る.
30000	K < N, N < 1 又は IP に誤りがあった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II … MGSSL
② FORTRAN 基本関数 … なし.

b. 注意
① 本サブルーチンは，LU 分解された実行列を入力して，分解された元の実行列の逆行列を計算する.
分解はサブルーチン ALU により行い，続けて本サブルーチンを呼び出すことにより逆行列を求めることができる.
② 連立 1 次方程式を解く場合には，サブルーチン LAX を用いること. 逆行列を求めて解くことは，LAX を用いるときよりも演算回数が多くなるので避けた方がよい. 本サブルーチンは逆行列が必要なときだけ使用すること.
③ トランスポジションベクトルとは，部分ピボッティングを行う LU 分解

$$\boldsymbol{PA}=\boldsymbol{LU}$$

における置換行列 $\boldsymbol{P}$ に相当する.
サブルーチン ALU の使用上の注意②を参照すること.

c. 使用例
$n\times n$ の実行列を入力して，その逆行列を求める.
$n\leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),VW(100),IP(100)
      READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,((I,J,A(I,J),J=1,N),
     *             I=1,N)
      CALL ALU(A,100,N,0.0,IP,IS,VW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      CALL LUIV(A,100,N,IP,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,630) ((I,J,A(I,J),I=1,N),
     *             J=1,N)
      STOP
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT(//11X,'**INPUT MATRIX**'/12X,
     * 'ORDER=',I5/(2X,4('(',I3,',',I3,')',
     * E16.8)))
  610 FORMAT('0',10X,'CONDITION ',
     * 'CODE(ALU)=',I5)
```

```
  620 FORMAT('0',10X,'CONDITION ',
     * 'CODE(LUIV)=',I5)
  630 FORMAT('0',10X,'**INVERSE MATRIX**',
     * /(2X,4('(',I3,',',I3,')',E16.8)))
      END
```

(4)　手法概要

$n\times n$ の実行列 $\boldsymbol{A}$ の LU 分解された行列 $\boldsymbol{L}$，$\boldsymbol{U}$ 及び LU 分解におけるピボッティングによる行の入換えを示す置換行列 $\boldsymbol{P}$ が与えられたとき, $\boldsymbol{A}$ の逆行列 $\boldsymbol{A}^{-1}$ を求めることを考える.

ところで

$$\boldsymbol{PA}=\boldsymbol{LU} \tag{4.1}$$

であるから

$$\boldsymbol{A}^{-1}=\left(\boldsymbol{P}^{-1}\boldsymbol{LU}\right)^{-1}=\boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P} \tag{4.2}$$

となる．そこで，$\boldsymbol{L}$ と $\boldsymbol{U}$ の逆行列を求めて，$\boldsymbol{U}^{1}$ に対して $\boldsymbol{L}^{-1}$ を右側から掛け，その結果に対して，$\boldsymbol{P}$ を右側から掛けて $\boldsymbol{A}$ の逆行列 $\boldsymbol{A}^{-1}$ を計算する.

なお，以降の説明のために $\boldsymbol{L}$ 及び $\boldsymbol{U}$ を (4.3) で表す.

$$\boldsymbol{L}=(l_{ij})\ ,\ \boldsymbol{U}=(u_{ij}) \tag{4.3}$$

a.　$\boldsymbol{L}^{-1}$ の計算

下三角行列 $\boldsymbol{L}$ の逆行列 $\boldsymbol{L}^{-1}$ も下三角行列となるので，$\boldsymbol{L}^{-1}$ を (4.4) で表すと,

$$\boldsymbol{L}^{-1}=\left(\tilde{l}_{ij}\right) \tag{4.4}$$

$\boldsymbol{LL}^{-1}=I$ より，(4.5) が成り立つ.

$$\sum_{k=1}^{n} l_{ik}\tilde{l}_{kj}=\delta_{ij}\ ,$$

$$\delta_{ij}=\begin{cases}1, & i=j\\ 0, & i\neq j\end{cases} \tag{4.5}$$

(4.5) を

$$\sum_{k=j}^{i-1} l_{ik}\tilde{l}_{kj}+l_{ii}\tilde{l}_{ij}=\delta_{ij}$$

と変形させて，$\boldsymbol{L}^{-1}$ の第 j 列目（$j=1$，$\cdots n$）の要素 $\tilde{l}_{ij}$ を次のように求める.

$$\left.\begin{aligned}&\tilde{l}_{ij}=\left(-\sum_{k=j}^{i-1} l_{ik}\tilde{l}_{kj}\right)/l_{ii},\ \ i=j+1,..,n\\ &\tilde{l}_{jj}=1/l_{jj}\end{aligned}\right\} \tag{4.6}$$

ここで，$l_{ii}\neq 0\left(i=j,..,n\right)$ とする.

b.　$\boldsymbol{U}^{-1}$ の計算

単位上三角行列 $\boldsymbol{U}$ の逆行列 $\boldsymbol{U}^{1}$ も単位上三角行列となるので，$\boldsymbol{U}^{1}$ を (4.7) で表すと,

$$U^{-1}=\left(\tilde{u}_{ij}\right) \tag{4.7}$$

$\boldsymbol{UU}^{-1}=\boldsymbol{I}$ より，(4.8) が成り立つ.

$$\sum_{k=1}^{n} u_{ik}\tilde{u}_{kj}=\delta_{ij}\ ,$$

$$\delta_{ij}=\begin{cases}1, & i=j\\ 0, & i\neq j\end{cases} \tag{4.8}$$

$u_{ii}=1$ であるので，(4.8) を

$$\tilde{u}_{ij}+\sum_{k=i+1}^{j} u_{ik}\tilde{u}_{kj}=\delta_{ij}$$

と変形させて，かつ $\tilde{u}_{jj}=1$ を考慮して $\boldsymbol{U}^{-1}$ の第 j 列目（$j=n$，…，2）の要素 $\tilde{u}_{ij}$ を次のように求める.

$$\tilde{u}_{ij}=-u_{ij}-\sum_{k=i+1}^{j-1} u_{ik}\tilde{u}_{kj},i=j-1,...,1 \tag{4.9}$$

c.　$\boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P}$ の計算

行列 $\boldsymbol{U}^{-1}$ と $\boldsymbol{L}^{-1}$ の掛け合わせた結果を行列 $\boldsymbol{B}$ とするとき，行列 $\boldsymbol{B}$ の要素 b_{ij} は

$$b_{ij}=\sum_{k=j}^{n}\tilde{u}_{ik}\tilde{l}_{kj},i=1,..,j-1$$

$$b_{ij}=\sum_{k=i}^{n}\tilde{u}_{ik}\tilde{l}_{kj},i=j,...,n$$

より求められる．そこで，$\tilde{u}_{ii}=1$ を考慮して $\boldsymbol{B}$ の第 j 列目（$j=1,\cdots,n$）の要素 b_{ij} は (4.10) の計算をして求める.

$$\left.\begin{aligned}&b_{ij}=\sum_{k=j}^{n}\tilde{u}_{ik}\tilde{l}_{kj},\quad i=1,...,j-1\\ &b_{ij}=\tilde{l}_{ij}+\sum_{k=i+1}^{n}\tilde{u}_{ik}\tilde{l}_{kj},\qquad i=j,...,n\end{aligned}\right\} \tag{4.10}$$

次に，行列 $\boldsymbol{B}$ に対して置換行列を掛け合わせて，$\boldsymbol{A}^{-1}$ の要素 $\tilde{a}_{ij}$ を求める．しかし，実際には，トランスポジションベクトル IP の値を参照して行列 $\boldsymbol{B}$ の列の入換えだけで $\boldsymbol{A}^{-1}$ の要素を求める.

なお，(4.6)，(4.9) 及び (4.10) に関する計算のうち積和計算部分は精度を上げて行うことにより，丸め誤差の影響を少なくしている．
また，詳細については，参考文献 [1] を参照すること．

A22-11-0302 LUX, DLUX

LU 分解された実行列の連立 1 次方程式
CALL LUX (B, FA, K, N, ISW, IP, ICON)

(1) 機能

LU 分解された実係数行列を持つ連立 1 次方程式

$$\boldsymbol{LUx} = \boldsymbol{Pb} \qquad (1.1)$$

を解く．ただし，$\boldsymbol{L}$，$\boldsymbol{U}$ はそれぞれ $n \times n$ の下三角行列と単位上三角行列，$\boldsymbol{P}$ は置換行列（係数行列を LU 分解するときの部分ピボッティングによる行の入換えを行う），$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．なお，方程式 (1.1) のかわりに

$$\boldsymbol{Ly} = \boldsymbol{Pb} \qquad (1.2)$$

$$\boldsymbol{Uz} = \boldsymbol{b} \qquad (1.3)$$

のいずれかを解くこともできる．

$n \geq 1$ であること．

(2) パラメタ

B ····· **入力**．定数ベクトル $\boldsymbol{b}$．
出力．解ベクトル $\boldsymbol{x}$，$\boldsymbol{y}$，$\boldsymbol{z}$ のうちのいずれか．
大きさ n の 1 次元配列．

FA ····· **入力**．行列 $\boldsymbol{L}$ と行列 $\boldsymbol{U}$．
図 LUX-1 参照．
FA(K, N) なる 2 次元配列．

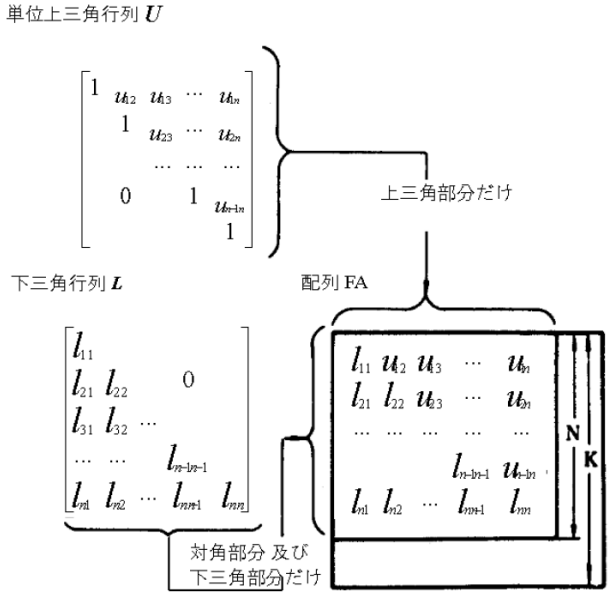


図 LUX-1 配列 FA における $\boldsymbol{L}$ 及び $\boldsymbol{U}$ の各要素の格納方法

K ····· **入力**．配列 FA の整合寸法（≥ N）．

N ····· **入力**．行列 $\boldsymbol{L}$，$\boldsymbol{U}$ の次数 n．

ISW ····· **入力**．制御情報．
ISW=1 のとき，$\boldsymbol{x}$ を求める．
ISW=2 のとき，$\boldsymbol{y}$ を求める．
ISW=3 のとき，$\boldsymbol{z}$ を求める．

IP ····· **入力**．部分ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．
大きさ n の 1 次元配列．
（サブルーチン ALU の使用上の注意②参照）

ICON ····· **出力**．コンディションコード．
表 LUX-1 参照．

表 LUX-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	係数行列が非正則であった．	処理を打ち切る．
30000	K < N，N < 1，ISW ≠ 1，2，3 又は，IP に誤りがあった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II··· MGSSL
② FORTRAN 基本関数··· なし．

b. 注意
① 連立 1 次方程式を解く場合，サブルーチン ALU を呼び出して，係数行列を LU 分解させてから，本サブルーチンを呼び出せば方程式を解くことができる．しかし，通常はサブルーチン LAX を呼び出せば，一度に解が求められる．

c. 使用例
$n \times n$ の係数行列をサブルーチン ALU で LU 分解させてから，連立 1 次方程式を解く．
$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(100,100),B(100),
     *          VW(100),IP(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      READ(5,510) (B(I),I=1,N)
      WRITE(6,600) N,((I,J,A(I,J),J=1,N),
     *             I=1,N)
      WRITE(6,610) (I,B(I),I=1,N)
      CALL ALU(A,100,N,1.0E-6,IP,IS,VW,
     *         ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL LUX(B,A,100,N,1,IP,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,640) (I,B(I),I=1,N)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT(///10X,'**COEFFICIENT ',
     * 'MATRIX**'
     * /12X,'ORDER=',I5,/(10X,4('(',I3,',',
     * I3,')',E16.8)))
  610 FORMAT('0',10X,'CONSTANT VECTOR'
     * /(10X,5('(',I3,')',E16.8)))
  620 FORMAT('0',10X,'CONDITION(ALU)'
     * ,I5)
  630 FORMAT('0',10X,'CONDITION(LUX)'
     * ,I5)
  640 FORMAT('0',10X,'SOLUTION VECTOR'
     * /(10X,5('(',I3,')',E16.8)))
      END
```

(4) 手法概要

連立 1 次方程式

$$\boldsymbol{LUx} = \boldsymbol{Pb} \tag{4.1}$$

を解くことは，次の二つの方程式

$$\boldsymbol{Ly} = \boldsymbol{Pb} \tag{4.2}$$

$$\boldsymbol{Ux} = \boldsymbol{y} \tag{4.3}$$

を解くことに帰着される．

a. $\boldsymbol{Ly} = \boldsymbol{Pb}$ を解く（前進代入）

$$y_i = \left(\mathrm{b}_i^{'} - \sum_{k=1}^{i-1} l_{ik} y_k \right) / l_{ii}, i = 1,...,n \tag{4.4}$$

なる式で逐次求める．ただし，$\boldsymbol{L} = (l_{ij})$，$\boldsymbol{y}^T = (y_1, \cdots, y_n)$，$(\boldsymbol{Pb})^T = (b'_1, \cdots, b'_n)$ である．

b. $\boldsymbol{Ux} = \boldsymbol{y}$ を解く（後退代入）

$$x_i = y_i - \sum_{k=i+1}^{n} u_{ik} x_k, i = n,...,1 \tag{4.5}$$

なる式で逐次求める．ただし，$\boldsymbol{U} = (u_{ij})$，$\boldsymbol{x}^T = (x_1, \cdots, x_n)$ である．

なお，(4.4) と (4.5) の積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている．

また，詳細については，参考文献 [1]，[3] 及び [4] を参照すること．

A21-13-0101 MAV, DMAV

実行列と実ベクトルの積
CALL MAV (A, K, M, N, X, Y, ICON)

(1) 機能

$m \times n$ の実行列 $\boldsymbol{A}$ と実ベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$ を求める.

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{1.1}$$

ここで，$\boldsymbol{x}$ は n 次元ベクトル，$\boldsymbol{y}$ は m 次元ベクトルである．m，$n \geq 1$ であること.

(2) パラメタ

A ……**入力**．行列 $\boldsymbol{A}$.
A (K, N)になる 2 次元配列.
K ……**入力**．配列 **A** の整合寸法 (≥M).
M ……**入力**．行列 $\boldsymbol{A}$ の行数 m.
N ……**入力**．行列 $\boldsymbol{A}$ の列数 n.
（使用上の注意①参照）.
X ……**入力**．ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.
Y ……**出力**．行列 $\boldsymbol{A}$ とベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$.
大きさ m の 1 次元配列.
ICON ……**出力**．コンディションコード.
表 MAV-1 参照.

表 MAV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	M<1, N=0 又は, K<M であった	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ..... MGSSL
② FORTRAN 基本関数 ... IABS

b. 注意
① 本サブルーチンは通常

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{3.1}$$

の計算を主とするが，N=$-n$ として任意のベクトル $\boldsymbol{y}'$ をパラメタ Y に与えると

$$\boldsymbol{y} = \boldsymbol{y}' - \boldsymbol{A}\boldsymbol{x} \tag{3.2}$$

なる計算ができる.
具体的には，連立 1 次方程式の残差ベクトル

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \tag{3.3}$$

の計算等を行う場合に利用できる．使用例参照.

c. 使用例
n 元の実係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.4}$$

をサブルーチン LAX により解き，その結果より残差ベクトル $\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$ を求める.
$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),X(100),Y(100),
     * VW(100),IP(100),W(100,100)
      READ(5,500) N
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      READ(5,510) (X(I),I=1,N)
      WRITE(6,600) N
      WRITE(6,610) ((I,J,A(I,J),J=1,N),
     * I=1,N)
      WRITE(6,620) (I,X(I),I=1,N)
      EPSZ=1.0E-6
      ISW=1
      DO 10 I=1,N
      Y(I)=X(I)
      DO 10 J=1,N
      W(J,I)=A(J,I)
   10 CONTINUE
      CALL LAX(A,100,N,X,EPSZ,ISW,IS,VW,
     *         IP,ICON)
      WRITE(6,630) (I,X(I),I=1,N)
      CALL MAV(W,100,N,-N,X,Y,ICON)
      IF(ICON.NE.0) GOTO 30
      WRITE(6,640) (I,Y(I),I=1,N)
      DN=0.0
      DO 20 I=1,N
      DN=DN+Y(I)*Y(I)
   20 CONTINUE
      DN=SQRT(DN)
      WRITE(6,650) DN
   30 WRITE(6,660) ICON
      STOP
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1','COEFFICIENT MATRIX'
     * /' ','ORDER=',I5)
  610 FORMAT(/4(' ','(',I3, ',',I3, ')',
     * E17.8))
  620 FORMAT(/' ','CONSTANT VECTOR'
     * /(10X,4('(',I3,')',E17.8)))
  630 FORMAT(/' ','SOLUTION VECTOR'
     * /(10X,4('(',I3,')',E17.8)))
  640 FORMAT(/' ','RESIDUAL VECTOR'
     * /(10X,4('(',I3,')',E17.8)))
  650 FORMAT(/' ','NORM=',E17.8)
  660 FORMAT(/' ','ICON=',I5)
      END
```

(4) 手法概要

$m \times n$ の実行列 $\boldsymbol{A}=\left(a_{ij}\right)$ と n 次元のベクトル $\boldsymbol{x}=\left(x_j\right)$ の積 $\boldsymbol{y}=\left(y_i\right)$ の要素を(4.1)により計算する.

$$y_i=\sum_{j=1}^{n} a_{ij}x_j, \quad i=1,...,m \tag{4.1}$$

本サブルーチンでは(4.1)の積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている.

A51-11-0101 MBV, DMBV

実バンド行列と実ベクトルの積
CALL MBV (A, N, NH1, NH2, X, Y, ICON)

(1) 機能

$n \times n$ 下バンド幅 h_1，上バンド幅 h_2 の実バンド行列 $\boldsymbol{A}$ と実ベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$ を求める．

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{1.1}$$

ここで，$\boldsymbol{x}$ と $\boldsymbol{y}$ は n 次元ベクトルである．
$n > h_1 \geq 0$, $n > h_2 \geq 0$ であること．

(2) パラメタ

A ……**入力**．行列 $\boldsymbol{A}$.
バンド行列用圧縮モード．
大きさ $n \cdot \min(h_1 + h_2 + 1, n)$ の 1 次元配列.

N ……**入力**．行列 $\boldsymbol{A}$ の次数 n.
（使用上の注意①参照）

NH1 ……**入力**．下バンド幅 h_1.

NH2 ……**入力**．上バンド幅 h_2.

X ……**入力**．ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.

Y ……**出力**．行列 $\boldsymbol{A}$ とベクトル $\boldsymbol{x}$ の積 y.
大きさ n の 1 次元配列.

ICON ……**出力**．コンディションコード.
表 MBV-1 参照.

表 MBV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	$N = 0, \lvert N \rvert \leq NH1, \lvert N \rvert \leq NH2$, $NH1 < 0$又は$NH2 < 0$であった.	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム

① SSLII ... MGSSL

② FORTRAN 基本関数 ... IABS, MIN0

b. 注意

① 本サブルーチンは通常

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{3.1}$$

の計算を主とするが，N = - n として，任意のベクトル $\boldsymbol{y}'$ をパラメタ Y に与えると

$$\boldsymbol{y} = \boldsymbol{y}' - \boldsymbol{A}\boldsymbol{x}$$

になる計算ができる．
具体的には，連立 1 次方程式の残差ベクトルの計算等を行う場合に利用できる（使用例参照）．

c. 使用例

$n \times n$ 下バンド幅 h_1，上バンド幅 h_2 の行列を係数に持つ連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

をサブルーチン LBX1 により解き，その結果より残差ベクトル $\boldsymbol{b}$ - $\boldsymbol{A}\boldsymbol{x}$ を求める．
$n \leq 100$, $h_1 \leq 20$, $h_2 \leq 20$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(4100),X(100),IP(100),
     *     FL(1980),VW(100),Y(100),W(4100)
      CHARACTER*4 NT1(6),NT2(4),NT3(4),
     *     NT4(4)
      DATA NT1/'CO  ','EF  ','FI  ',
     *         'CI  ','EN  ','T   '/,
     *     NT2/'CO  ','NS  ','TA  ',
     *         'NT  '/,
     *     NT3/'SO  ','LU  ','TI  ',
     *         'ON  '/,
     *     NT4/'RE  ','SI  ','DU  ',
     *         'AL  '/
      READ(5,500) N,NH1,NH2
      WRITE(6,600) N,NH1,NH2
      IF(N.LE.0.OR.NH1.GE.N.OR.NH2.GE.N)
     *  STOP
      NT=N*MIN0(N,NH1+NH2+1)
      READ(5,510) (A(I),I=1,NT)
      CALL PBM(NT1,6,A,N,NH1,NH2)
      READ(5,510) (X(I),I=1,N)
      CALL PGM(NT2,4,X,N,N,1)
      DO 10 I=1,NT
   10 W(I)=A(I)
      DO 20 I=1,N
   20 Y(I)=X(I)
      CALL LBX1(A,N,NH1,NH2,X,0.0,1,IS,
     *          FL,VW,IP,ICON)
      IF(ICON.GE.20000) GO TO 30
      CALL PGM(NT3,4,X,N,N,1)
      CALL MBV(A,-N,NH1,NH2,X,Y,ICON)
      IF(ICON.NE.0) GO TO 30
      CALL PGM(NT4,4,Y,N,N,1)
      RN=0.0
      DO 25 I=1,N
   25 RN=RN+Y(I)*Y(I)
      RN=SQRT(RN)
      WRITE(6,610) RN
      STOP
   30 WRITE(6,620) ICON
      STOP
  500 FORMAT(3I5)
  510 FORMAT(10F8.3)
  600 FORMAT('1','BAND EQUATIONS'
     *  /5X,'ORDER=', I5
     *  /5X,'SUB-DIAGONAL,LINES=',I4
     *  /5X,'SUPER-DIAGONAL,LINES=',I4)
  610 FORMAT(' ',4X,'RESIDUAL NORM=',
     *  E17.8)
  620 FORMAT(' ',4X,'ICON=',I5)
      END
```

サブルーチン PBM 及び PGM は，バンド行列及び実行列を印刷するものである．プログラムは，サブルーチン LBX1 及び MGSM の各使用例に記載されている．

(4) 手法概要

$n \times n$，下バンド幅 h_1，上バンド幅 h_2 のバンド行列 $\boldsymbol{A}=\left(a_{ij}\right)$ と n 次元のベクトル $\boldsymbol{x}=\left(x_j\right)$ の積 $\boldsymbol{y}=\left(y_i\right)$ の要素を(4.1)により計算する．

$$y_i=\sum_{j=1}^{n} a_{ij}x_j, \quad i=1,...,n \tag{4.1}$$

ただし，本サブルーチンでは，バンド行列であることを考慮して，(4.2)により計算する．

$$y_i=\sum_{j=\max(1,i-h_1)}^{\min(i+h_2,n)} a_{ij}x_j, \quad i=1,...,n \tag{4.2}$$

本サブルーチンでは(4.2)の積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている．

A21-15-0101 MCV, DMCV

複素行列と複素ベクトルの積
CALL MCV (ZA, K, M, N, ZX, ZY, ICON)

(1) 機能

$m \times n$ の複素行列 $\boldsymbol{A}$ と複素ベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$ を求める.

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \qquad (1.1)$$

ここで，$\boldsymbol{x}$ は n 次元の複素ベクトル，$\boldsymbol{y}$ は m 次元の複素ベクトルである.
$m, n \geq 1$ であること.

(2) パラメタ

ZA ……**入力**．複素行列 $\boldsymbol{A}$
ZA (K, N) なる複素数型 2 次元配列.
K ……**入力**．配列 ZA の整合寸法 (≥M).
M ……**入力**．行列 $\boldsymbol{A}$ の行数 m.
N ……**入力**．行列 $\boldsymbol{A}$ の列数 n.
（使用上の注意①参照）
ZX ……**入力**．複素ベクトル $\boldsymbol{x}$.
大きさ n の複素数型 1 次元配列.
ZY ……**出力**．行列 $\boldsymbol{A}$ と複素ベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$．大きさ m の複素数型 1 次元配列.
ICON ……**出力**．コンディションコード.
表 MCV-1 参照.

表 MCV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	M<1, N=0 又は K<M であった.	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... IABS

b. 注意
① 本サブルーチンは通常

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \qquad (3.1)$$

の計算を主とするが，N=−n として，任意の複素ベクトル $\boldsymbol{y}'$ をパラメタ ZY に与えると

$$\boldsymbol{y} = \boldsymbol{y}' - \boldsymbol{A}\boldsymbol{x} \qquad (3.2)$$

なる計算ができる.
具体的には，連立 1 次方程式の残差ベクトルの計算等を行う場合に利用できる（使用例参照）.

c. 使用例
n 次元の複素係数連立 1 次方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

をサブルーチン LCX により解き，その結果より残差ベクトル $\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$ を求める．$n \leq 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION ZA(50,50),ZX(50),ZY(50),
     *          ZVW(50),IP(50),ZW(50,50)
      CHARACTER*4 NT1(6),NT2(4),
     *            NT3(4),NT4(4)
      COMPLEX ZA,ZX,ZY,ZVW,ZW
      DATA NT1/'CO  ','EF  ','FI  ',
     *         'CI  ','EN  ','T   '/,
     *     NT2/'CO  ','NS  ','TA  ',
     *         'NT  '/,
     *     NT3/'SO  ','LU  ','TI  ',
     *         'ON  '/,
     *     NT4/'RE  ','SI  ','DU  ',
     *         'AL  '/
      READ(5,500) N
      IF(N.LE.0) STOP
      READ(5,510) ((ZA(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      CALL PCM(NT1,6,ZA,50,N,N)
      ISW=1
      EPSZ=1.0E-6
      READ(5,510) (ZX(I),I=1,N)
      CALL PCM(NT2,4,ZX,N,N,1)
      DO 10 I=1,N
      ZY(I)=ZX(I)
      DO 10 J=1,N
      ZW(J,I)=ZA(J,I)
   10 CONTINUE
      CALL LCX(ZA,50,N,ZX,EPSZ,ISW,IS,
     *         ZVW,IP,ICON)
      IF(ICON.GE.20000) GO TO 30
      CALL PCM(NT3,4,ZX,N,N,1)
      CALL MCV(ZW,50,N,-N,ZX,ZY,ICON)
      IF(ICON.NE.0) GO TO 30
      CALL PCM(NT4,4,ZY,N,N,1)
      RN=0.0
      DO 20 I=1, N
      CR=REAL(ZY(I))
      CI=IMAG(ZY(I))
      RN=RN+CR*CR+CI*CI
   20 CONTINUE
      RN=SQRT(RN)
      WRITE(6,610) RN
      STOP
   30 WRITE(6,620) ICON
      STOP
  500 FORMAT(I5)
  510 FORMAT(5(2F8.3))
  600 FORMAT('1','COMPLEX LINEAR ',
     * 'EQUATIONS'/5X,'ORDER=',I5)
  610 FORMAT(' ',4X,'RESIDUAL ',
     * 'NORM=',E17.8)
  620 FORMAT(' ',4X,'ICON=',I5)
      END

      SUBROUTINE PCM(ICOM,L,ZA,K,M,N)
      DIMENSION ZA(K,N)
      CHARACTER*4 ICOM(L)
      COMPLEX ZA
      WRITE(6,600) (ICOM(I),I=1,L)
      DO 10 I=1,M
      WRITE(6,610) I,(J,ZA(I,J),J=1,N)
   10 CONTINUE
```

```
      RETURN
  600 FORMAT(' ',35A2)
  610 FORMAT(' ',1X,I3,2(I3,2E17.7)
     * /(5X,2(I3,2E17.7)))
      END
```

サブルーチン PCM は，複素行列を印刷するものである．

(4) 手法概要

$m \times n$ の複素行列 $\boldsymbol{A} = \left(a_{ij}\right)$ と n 次元の複素ベクトル $\boldsymbol{x} = \left(x_j\right)$ の積 $\boldsymbol{y} = \left(y_i\right)$ の要素を(4.1)より計算する．

$$y_i = \sum_{j=1}^{n} a_{ij} x_j, \qquad i = 1,..,m \tag{4.1}$$

本サブルーチンでは(4.1)の積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている．

A22-21-0302 MDMX, DMDMX

MDM^T 分解された実対称行列の連立 1 次方程式
CALL MDMX (B, FA, N, IP, ICON)

(1) 機能

MDM^T 分解された実対称係数行列を持つ連立 1 次方程式

$$\boldsymbol{P}^{-1}\boldsymbol{M}\boldsymbol{D}\boldsymbol{M}^{T}\left(\boldsymbol{P}^{T}\right)^{-1}\boldsymbol{x}=\boldsymbol{b} \qquad (1.1)$$

を解く．ただし，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列，$\boldsymbol{P}$ は置換行列（係数行列を MDM^T 分解するときのピボッティングによる行の入換えを行う），$\boldsymbol{b}$ は n 次元の実定数ベクトル，$\boldsymbol{x}$ は n 次元の解ベクトルである．ここで $\boldsymbol{d}_{k+1,k}\neq 0$，なら $\boldsymbol{m}_{k+1,k}=0$ である．$n\geq 1$ であること．

(2) パラメタ

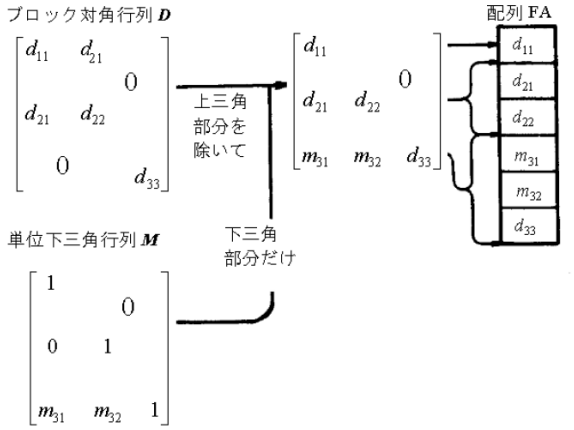
B ····· **入力**．定数ベクトル $\boldsymbol{b}$.
出力．解ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.

FA ····· **入力**．行列 $\boldsymbol{M}$ と行列 $\boldsymbol{D}$.
図 MDMX-1 参照.
大きさ $n(n+1)/2$ の 1 次元配列.

N ····· **入力**．行列 $\boldsymbol{M}$，行列 $\boldsymbol{D}$，定数ベクトル $\boldsymbol{b}$ 及び解ベクトルの $\boldsymbol{x}$ の次数 n.

IP ····· **入力**．ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル.
大きさ n の 1 次元配列.

ICON ····· **出力**．コンディションコード.
表 MDMX-1 参照.



[注意] 行列 ***D+(M-I)***の対角部分及び下三角部分を，対称行列用圧縮モードで 1 次元配列 FA に格納する．ここで $\boldsymbol{D}$ は次数 2 と 1 のブロックからなる場合である.

図 MDMX-1 行列 L 及び D の格納方法

表 MDMX-1 コンディションコード.

コード	意味	処理内容
0	エラーなし.	
20000	係数行列が非正則であった.	処理を打ち切る
30000	N<1 又は IP に誤りがあった.	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... IABS

b. 注意
① 連立 1 次方程式を解く場合，サブルーチン SMDM を呼び出して，係数行列を MDM^T 分解させてから，本サブルーチンを呼び出せば方程式を解くことができる．しかし，通常はサブルーチン LSIX を呼び出せば，一度に解が求められる.
② 本サブルーチンの入力パラメタ FA，IP は，サブルーチン SMDM の出力パラメタ A，IP と同じである.

c. 使用例
$n\times n$の実対称行列をサブルーチンSMDMでMDM^T分解させてから，連立1次方程式を解く.
$n\leq 100$の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),B(100),VW(200),
     *          IP(100),IVW(100)
      CHARACTER*4 IA,IB,IX
      DATA IA,IB,IX/'A   ','B   ','X   '/
      READ(5,500) N
      NT=(N*(N+1))/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,600) N
      CALL PSM(IA,1,A,N)
      EPSZ=0.0
      CALL SMDM(A,N,EPSZ,IP,VW,IVW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      READ(5,510) (B(I),I=1,N)
      CALL PGM(IB,1,B,N,N,1)
      CALL MDMX(B,A,N,IP,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) STOP
      CALL PGM(IX,1,B,N,N,1)
      STOP
  500 FORMAT(I3)
  510 FORMAT(4E15.7)
  600 FORMAT('1'
     *  /6X,'LINEAR EQUATIONS AX=B'
     *  /6X,'ORDER=',I4)
  610 FORMAT(' ',5X,'ICON OF SMDM=',I6)
  620 FORMAT(' ',5X,'ICON OF MDMX=',I6)
      END
```

本使用例中のサブルーチン PSM 及び PGM は，実対称行列及び実行列を印刷するものである.

プログラムは，サブルーチン MGSM の使用例に記載されている．

(4) 手法概要

MDM^T 分解された実対称行列を係数に持つ連立 1 次方程式

$$\boldsymbol{P}^{-1}\boldsymbol{M}\boldsymbol{D}\boldsymbol{M}^{T}\left(\boldsymbol{P}^{T}\right)^{-1}\boldsymbol{x}=\boldsymbol{b} \tag{4.1}$$

を解くことは，次の四つの方程式を解くことに帰着される．

$$\boldsymbol{M}\boldsymbol{x}^{(1)}=\boldsymbol{P}\boldsymbol{b} \tag{4.2}$$
$$\boldsymbol{D}\boldsymbol{x}^{(2)}=\boldsymbol{x}^{(1)} \tag{4.3}$$
$$\boldsymbol{M}^{T}\boldsymbol{x}^{(3)}=\boldsymbol{x}^{(2)} \tag{4.4}$$
$$\left(\boldsymbol{P}^{T}\right)^{-1}\boldsymbol{x}=\boldsymbol{x}^{(3)} \tag{4.5}$$

ここで，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックからなる対称ブロック対角行列，$\boldsymbol{b}$ は定数ベクトル，$\boldsymbol{x}$ は解ベクトルである．本サブルーチンは，$\boldsymbol{M}$ と $\boldsymbol{D}$ がブロック対角ピボッティング手法により分解された行列であることを前提としており，$\boldsymbol{P}$ はその際の置換行列である．（詳細はサブルーチン SMDM の手法概要参照）．

a. $\boldsymbol{M}\boldsymbol{x}^{(1)}=\boldsymbol{P}\boldsymbol{b}$ を解く（後退代入）
1 × 1 ピボットのとき（行列 $\boldsymbol{D}$ のブロックの次数が 1 であるとき）は(4.6)なる式で逐次求める．

$$x_i^{(1)}=b_i'-\sum_{k=1}^{i-1}m_{ik}x_k^{(1)},\quad i=1,\ldots,n \tag{4.6}$$

しかし，i 番目が 2×2 ピボットのとき（行列 $\boldsymbol{D}$ のブロックの次数が 2 であるとき）は $\boldsymbol{x}_i^{(1)}$,に続けて $\boldsymbol{x}_{i+1}^{(1)}$ を(4.7)で求め，i+2 番目へ進む．

$$x_{i+1}^{(1)}=b'_{i+1}-\sum_{k=1}^{i-1}m_{ik}x_k^{(1)} \tag{4.7}$$

ただし，
$M=\left(m_{ij}\right),x^{(1)^T}=\left(x_1^{(1)},\ldots,x_n^{(1)}\right),\left(\boldsymbol{P}\boldsymbol{b}\right)^T=\left(b'_1,\ldots,b'_n\right)$
である．

b. $\boldsymbol{D}\boldsymbol{x}^{(2)}=\boldsymbol{x}^{(1)}$を解く
1 × 1 ピボットの時は，(4.8)なる式で逐次求める．

$$x_i^{(2)}=x_i^{(1)}/d_{ii},\qquad i=1,\ldots,n \tag{4.8}$$

しかし，i 番目が 2×2 ピボットのときは，$\boldsymbol{x}_i^{(2)}$ と $\boldsymbol{x}_{i+1}^{(2)}$ を(4.9)なる式で求め，i+2 番目へ進む．

$$x_i^{(2)}=\left(x_i^{(1)}d_{i+1,i+1}-x_{i+1}^{(1)}d_{i+1,i}\right)/DET$$
$$x_{i+1}^{(2)}=\left(x_{i+1}^{(1)}d_{ii}-x_i^{(1)}d_{i+1,i}\right)/DET$$
$$\text{ここで，}\ DET=\det\begin{pmatrix}d_{ii} & d_{i,i+1}\\ d_{i+1,i} & d_{i+1,i+1}\end{pmatrix} \tag{4.9}$$

ただし，$\boldsymbol{D}=(d_{ij})$，$x^{(2)^T}=\left(x_1^{(2)},\ldots,x_n^{(2)}\right)$である．

c. $M^T x^{(3)}=x^{(2)}$ を解く（前進代入）
1×1 ピボットのときは，(4.10)なる式で逐次求める．

$$x_i^{(3)}=x_i^{(2)}-\sum_{k=i+1}^{n}m_{ki}x_k^{(3)}\qquad ,i=n,\ldots,1 \tag{4.10}$$

しかし，i 番目が 2 × 2 ピボットのときは，$\boldsymbol{x}_i^{(3)}$ に続けて $\boldsymbol{x}_{i-2}^{(3)}$ を(4.11)で求め，i+2 番目へ進む．

$$x_{i-1}^{(3)}=x_{i-1}^{(2)}-\sum_{k=i+1}^{n}m_{k,i-1}x_k^{(3)} \tag{4.11}$$

ただし，$x^{(3)^T}=\left(x_1^{(3)},\ldots,x_n^{(3)}\right)$である．

d. $\left(\boldsymbol{P}^{T}\right)^{-1}\boldsymbol{x}=\boldsymbol{x}^{(3)}$ を解く
ベクトル $\boldsymbol{x}^{(3)}$ に対して置換行列を掛け合わせて解ベクトル $\boldsymbol{x}$ の要素 x_i を求める．しかし，実際には，トランスポジションベクトル $\boldsymbol{IP}$ の値を参照して，ベクトル $\boldsymbol{x}^{(3)}$ の要素の入換えだけで求める．

本サブルーチンにおける積和計算部分は精度を上げて行うことにより，丸め誤差の影響を少なくしている．

A21-11-0301 MGGM, DMGGM

行列の積（実行列）
CALL MGGM (A, KA, B, KB, C, KC, M, N, L, VW, ICON)

(1) 機能

$m \times n$ 実行列 $\boldsymbol{A}$ と，$n \times l$ の実行列 $\boldsymbol{B}$ の積 $\boldsymbol{C}$ を求める．

$$\boldsymbol{C} = \boldsymbol{AB}$$

ここで $\boldsymbol{C}$ は，$m \times l$ の実行列である．
$m, n, l \geq 1$ であること．

(2) パラメタ

A ……**入力**．行列 $\boldsymbol{A}$.
A (KA, N) なる 2 次元配列.
KA ……**入力**．配列 A の整合寸法 (≥M).
B ……**入力**．行列 $\boldsymbol{B}$.
B (KB, L) なる 2 次元配列.
KB ……**入力**．配列 B の整合寸法 (≥N).
C ……**出力**．行列 $\boldsymbol{C}$.
C(KC, L)なる 2 次元配列.
（使用上の注意①参照）
KC ……**入力**．配列 **C** の整合寸法(≥M)
M ……**入力**．行列 $\boldsymbol{A}$, $\boldsymbol{C}$ の行数 m.
N ……**入力**．行列 $\boldsymbol{A}$ の列数及び行列 $\boldsymbol{B}$ の行数 n.
L ……**入力**．行列 $\boldsymbol{B}$, $\boldsymbol{C}$ の列数 l.
VW ……**作業領域**．大きさ n の 1 次元配列.
ICON ……**出力**．コンディションコード.
表 MGGM-1.参照.

表 MGGM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	M<1, N<1, L<1, KA<M, KB<N 又は，KC<M であった	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... なし

b. 注意
① 領域の節約について
配列 A 上の内容を保存する必要のない場合は，次のように呼び出すことにより領域が節約できる．
CALL MGGM (A, KA, B, KB, A, KA, M, N, L, VW, ICON)
この場合，配列 A に行列 $\boldsymbol{C}$ が得られる．
ただし配列 A は，あらかじめ A (KA, L)なる 2 次元配列として確保する必要がある．

c. 使用例
実行例 $\boldsymbol{A}$, $\boldsymbol{B}$ の積を求める．$m \leq 50$, $n \leq 60$, $l \leq 30$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(50,60),B(60,30),
     *          C(50,30),VW(60)
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
      DATA KA/50/,KB/60/,KC/50/
   10 READ(5,100) M,N,L
      IF(M.EQ.0) STOP
      WRITE(6,150)
      READ(5,200) ((A(I,J),I=1,M),J=1,N)
      READ(5,200) ((B(I,J),I=1,N),J=1,L)
      CALL MGGM(A,KA,B,KB,C,KC,M,N,L,VW,
     *          ICON)
      IF(ICON.NE.0) GOTO 10
      CALL PGM(IA,1,A,KA,M,N)
      CALL PGM(IB,1,B,KB,N,L)
      CALL PGM(IC,1,C,KC,M,L)
      GOTO 10
  100 FORMAT(3I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** MATRIX MULTIPLICATION **')
      END
```

本使用例中のサブルーチン PGM は，実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている．

A21-11-0401 MGSM, DMGSM

行列の積（実行列・実対称行列）
CALL MGSM (A, KA, B, C, KC, N, VW, ICON)

(1) 機能

$n \times n$ の実行列 ***A*** と，$n \times n$ の実対称行列 ***B*** の積 ***C*** を求める．

$$\boldsymbol{C} = \boldsymbol{AB}$$

ここで ***C*** は，$n \times n$ の実行列である．$n \geq 1$ であること．

(2) パラメタ

A ····· **入力**．行列 ***A*** ．
A (KA, N) なる 2 次元配列．
KA ····· **入力**．配列 A の整合寸法 (≥N).
B ····· **入力**．行列 ***B***.
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．
C ····· **出力**．行列 ***C***.
C(KC, N) なる 2 次元配列．
（使用上の注意①参照）
KC ····· **入力**．配列 C の整合寸法 (≥N).
N ····· **入力**．行列 ***A***, ***B*** 及び ***C*** の行数 *n*.
VW ····· **作業領域**．大きさ *n.* の 1 次元配列．
ICON ····· **出力**．コンディションコード．
表 MGSM-1 参照．

表 MGSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N<1,KA<N 又は, KC<N であった.	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... なし

b. 注意
① 領域の節約について
配列 ***A*** 上の内容を保存する必要のない場合は，次のように呼び出すことにより領域が節約できる．
CALL MGSM(A, KA, B, A, KA, N, VW, ICON)
この場合，配列 A に，一般モードで格納された行列 ***C*** が得られる．

c. 使用例
実行列 ***A*** と実対称行列 ***B*** の積 ***C*** を求める．ただし，行列 ***C*** は，***A*** を入力した領域に得るものとする．
$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(100,100),B(5050),VW(100)
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
   10 READ(5,100) N
      IF(N.EQ.0) STOP
      WRITE(6,150)
      NT=N*(N+1)/2
      READ(5,200) ((A(I,J),J=1,N),I=1,N)
      READ(5,200) (B(I),I=1,NT)
      CALL PGM(IA,1,A,100,N,N)
      CALL PSM(IB,1,B,N)
      CALL MGSM(A,100,B,A,100,N,VW,ICON)
      WRITE(6,250) ICON
      IF(ICON.NE.0) GOTO 10
      CALL PGM(IC,1,A,100,N,N)
      GOTO 10
  100 FORMAT(I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** C=A*B  GENERAL BY SYMMETRIC',
     * '**')
  250 FORMAT(//10X,'** MGSM ICON=',I5)
      END

C ** MATRIX PRINT(REAL NON-SYMMETRIC) **
      SUBROUTINE PGM(ICOM,L,A,K,M,N)
      DIMENSION A(K,N)
      CHARACTER*4 ICOM(L)
      WRITE(6,600) (ICOM(I),I=1,L)
      DO 10 I=1,M
      WRITE(6,610) I,(J,A(I,J),J=1,N)
   10 CONTINUE
      RETURN
  600 FORMAT(/10X,35A2)
  610 FORMAT(/5X,I3,3(4X,I3,E17.7),
     *(/8X,3(4X,I3,E17.7)))
      END

C ** MATRIX PRINT(REAL SYMMETRIC) **
      SUBROUTINE PSM(ICOM,L,A,N)
      DIMENSION A(1)
      CHARACTER*4 ICOM(L)
      WRITE(6,600) (ICOM(I),I=1,L)
      LS=1
      LE=0
      DO 10 I=1,N
      LE=LE+I
      WRITE(6,610) I,(A(J),J=LS,LE)
   10 LS=LE+1
      RETURN
  600 FORMAT(/10X,35A2)
  610 FORMAT(/5X,I3,3(4X,E17.7),
     *(/8X,3(4X,E17.7)))
      END
```

サブルーチン PSM 及び PGM は，実対称行列及び実行列を印刷するものである．

D11-10-0101 MINF1, DMINF1

多変数関数の極小化 （微係数不要，改訂準ニュートン法）
CALL MINF1 (X, N, FUN, EPSR, MAX, F, G, H, VW, ICON)

(1) 機能

n 変数の実関数 $f(\boldsymbol{x})$と，初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，改訂準ニュートン法により，$f(\boldsymbol{x})$の極小値を与えるベクトル $\boldsymbol{x}^*$と，その関数値 $f(\boldsymbol{x}^*)$を求める．

ただし，$f(\boldsymbol{x})$は 2 階までの連続な偏導関数を持つものと仮定する．

また，$n \geq 1$ であること．

(2) パラメタ

X ……**入力**．初期ベクトル $\boldsymbol{x}_0$.
出力．ベクトル $\boldsymbol{x}^*$.
大きさ n の 1 次元配列.

N ……**入力**．変数の個数 n.

FUN ……**入力**．$f(\boldsymbol{x})$を計算する関数副プログラム名.
［副プログラムの用意の仕方］
FUNCTION FUN(X)
パラメタ
X.....**入力**．任意な変数ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.
関数 FUN には，$f(\boldsymbol{x})$の値を代入すること.
（使用例参照）

EPSR ……**入力**．収束判定値 (≥0.0).
0.0 のときは標準値が採用される.
（使用上の注意②参照）

MAX……**入力**．関数の評価回数の上限(≠0).
（使用上の注意③参照）
出力．実際に評価した回数 (>0).

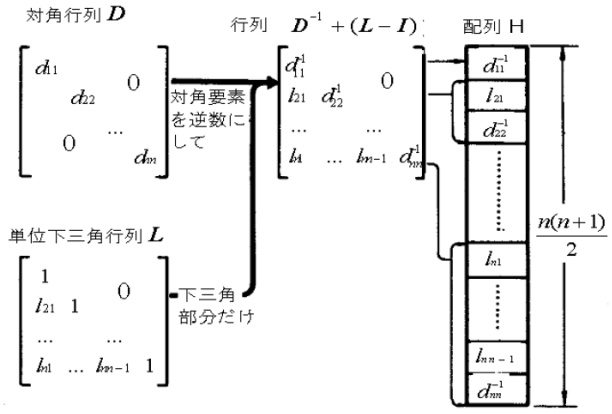
F ……**出力**．関数値 $f(\boldsymbol{x}^*)$.

G ……**出力**．$\boldsymbol{x}^*$における傾斜ベクトル.
大きさ n の 1 次元配列.

H ……**出力**．$\boldsymbol{x}^*$におけるヘシアン行列.
LDL^T 分解され，対称行列用圧縮モードで格納される．図 MINF1-1 参照・
大きさ $n(n+1)/2$ の 1 次元配列.

VW ……**作業領域**．大きさ $3n+1$ の 1 次元配列.

ICON ……**出力**．コンディションコード.
表 MINF1-1 参照.



［注意］ LDL^T 分解された近似ヘシアン行列は演算後，行列 $D^{-1}+(L-I)$の対角部分及び下三角部分が，対称行列用圧縮モードで 1 次元配列 H に格納される.

図 MINF1-1 ヘシアン行列の格納方法

表 MINF1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	指定された関数の評価回数内で，収束条件が満たされなかった.	パラメタ X, F, G, H には，最後の値を格納する.
20000	計算の過程で，$g_k^T p_k \geq 0$ となり，関数の局所的減少がみたされなかった.（手法概要（4.5）式参照）EPSR が小さすぎるか，傾斜ベクトルの差分近似の誤差が計算限界を超えた.	処理を打ち切る.（パラメタ X, F には，最後の値を格納する．）
30000	N<1, EPSR<0.0 又は MAX=0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II...AMACH, LDLX, MGSSL, UMLDL
② FORTRAN 基本関数 ... ABS, SQRT, AMAX1, AMIN1

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で，引数 FUN に相当する関数プログラムを EXTERNAL 文で宣言すること.
② EPSR の与え方
本サブルーチンの収束判定は，反復ベクトル $\boldsymbol{x}_k$ において，

$$\|x_{k+1} - x_k\|_\infty \leq \max\left(1.0, \|x_k\|_\infty\right) \cdot \mathrm{EPSR}$$

となった場合に，$\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$とみなし反復を終了する.
本サブルーチンでは，関数 $f(\boldsymbol{x})$が極小点 $\boldsymbol{x}^*$の近傍で近似的に 2 次関数であることを前提として

おり，関数値 $f(\boldsymbol{x}^*)$ を丸め誤差程度まで正しく求めるためには，

$$\text{EPSR} \approx \sqrt{u},\ u \text{ は丸め誤差の単位}$$

程度に与えるのが適当である．
EPSR の標準値は $2\cdot\sqrt{u}$ である．

③ MAX の与え方
関数の評価回数は，変数ベクトル $\boldsymbol{x}$ に対して，$f(\boldsymbol{x})$を計算することを 1 回と数える．すなわち，副プログラム FUN の呼出し回数に一致する．
関数の評価回数は，関数の性質，初期ベクトル，収束判定値などに依存する．
一般には，初期ベクトルが良く，収束判定値として標準値を採用した場合は，MAX=400$\cdot n$ 程度が妥当である．
指定評価回数内で収束条件が満たされず，ICON=10000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ MAX には，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと．

c. 使用例
$f(\boldsymbol{x}) = (1-x_1)^2 + 100\left(x_2 - x_1^{\,2}\right)^2$ の最小点 $\boldsymbol{x}^*$を初期ベクトル $\boldsymbol{x}_0 = (-1.2,\ 1.0)^{\mathrm{T}}$により求める．

```
C     **EXAMPLE**
      DIMENSION X(2),G(2),H(3),VW(7)
      EXTERNAL ROSEN
      X(1)=-1.2
      X(2)=1.0
      N=2
      EPSR=1.0E-3
      MAX=400*2
      CALL MINF1(X,N,ROSEN,EPSR,MAX,
     *           F,G,H,VW,ICON)
      WRITE(6,600) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,610) F,MAX
      WRITE(6,620) (I,X(I),I,G(I),I=1,N)
      STOP
  600 FORMAT('1','*ICON=',I5)
  610 FORMAT(' ','*F=',E15.7,' MAX=',I5/)
  620 FORMAT(/' X(',I2,')=',E15.7,2X,
     *       ' G(',I2,')=',E15.7)
      END

C     OBJECTIVE FUNCTION
      FUNCTION ROSEN(X)
      DIMENSION X(2)
      ROSEN=(1.0-X(1))**2+100.0*
     *      (X(2)-X(1)*X(1))**2
      RETURN
      END
```

(4) 手法概要

n 変数の実関数 $f(\boldsymbol{x})$と，初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，改訂準ニュートン法により，$f(\boldsymbol{x})$の極小値を与えるベクトル $\boldsymbol{x}^*$と関数値 $f(\boldsymbol{x}^*)$を求める．

本サブルーチンでは，$f(\boldsymbol{x})$の傾斜ベクトル $\boldsymbol{g}$ は差分による近似を用いて求める．

改訂準ニュートン法

関数 $f(\boldsymbol{x})$が 2 次関数の場合，極小点 $\boldsymbol{x}^*$の近傍でのテーラー級数展開は，(4.1)で与えられている．

$$f(\boldsymbol{x}) = f(\boldsymbol{x}^*) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}^*)^T \boldsymbol{B}(\boldsymbol{x}-\boldsymbol{x}^*) \qquad (4.1)$$

(4.1) で $\boldsymbol{B}$ は $\boldsymbol{x}^*$における $f(\boldsymbol{x})$のヘシアン行列である．行列 $\boldsymbol{B}$ が正定値であるならば，(4.1)は一意的に最小値を持つ．この場合，$\boldsymbol{x}_k$ を $\boldsymbol{x}^*$の近傍の任意な点とし，$\boldsymbol{x}_k$ における $f(\boldsymbol{x})$の傾斜ベクトルを $\boldsymbol{g}_k$ と表すと，(4.1) より $\boldsymbol{x}^*$は次式により求められる．

$$\boldsymbol{x}^* = \boldsymbol{x}_k - \boldsymbol{B}^{-1}\boldsymbol{g}_k \qquad (4.2)$$

一方，関数 $f(\boldsymbol{x})$が 2 次関数でない場合でも，$\boldsymbol{x}^*$の近傍では近似的に 2 次関数とみなすことができ，(4.2) に基づく反復公式を導くことができる．しかし，行列 $\boldsymbol{B}$ の逆行列を直接求めることは，計算量が多く効率的ではないため，一般にはその近似行列を設定し，反復の過程で修正していく方法が考えられる．

改訂準ニュートン法は，行列 $\boldsymbol{B}$ の近似行列を $\boldsymbol{B}_k$ として，(4.3) の反復公式法により，極小点 $\boldsymbol{x}^*$を求める方法である．

$$\left.\begin{aligned} \boldsymbol{B}_k\boldsymbol{p}_k &= -\boldsymbol{g}_k \\ \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k \\ \boldsymbol{B}_{k+1} &= \boldsymbol{B}_k + \boldsymbol{E}_k \end{aligned}\right\} k = 0,1, \qquad (4.3)$$

(4.3) で，$\boldsymbol{g}_0 = \nabla f(\boldsymbol{x}_0)$，また $\boldsymbol{B}_0$は任意の正定値行列である．$\boldsymbol{p}_k$は $\boldsymbol{x}_k$ から極小点に向かう探索方向を示すベクトルであり，α_k は $f(\boldsymbol{x}_k + \alpha_k\boldsymbol{P}_k)$ が局所的最小となるように定める（直線探索）定数である．

一方，$\boldsymbol{E}_k$ は近似ヘシアン行列 $\boldsymbol{B}_k$ を改良する，ランク 2 の行列である．$\boldsymbol{E}_k$は，関数 $f(\boldsymbol{x})$が極小点 $\boldsymbol{x}^*$の近傍で近似的に 2 次であることを前提として，セカンド条件

$$(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k) = \boldsymbol{B}_{k+1}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) \qquad (4.4)$$

を満たすように定める．

(4.3) による反復において，探索方向 $\boldsymbol{p}_k$ が下降方向（関数 $f(\boldsymbol{x})$が $\boldsymbol{x}_k$において $\boldsymbol{p}_k$の方向に局所的に減少する）にあるためには，関数 $f(\boldsymbol{x})$が $\boldsymbol{x}^*$で極小値を持つための 2 次の十分条件より，

$$\boldsymbol{g}_k^{\,T}\boldsymbol{p}_k \left(= -\boldsymbol{p}_k^{\,T}\boldsymbol{B}_k\boldsymbol{p}_k\right) < 0 \qquad (4.5)$$

が要求される．すなわち，近似ヘシアン行列 $\boldsymbol{B}_k$ の正定値性を保証することが，反復計算での要点である．

改訂準ニュートン法では，LDL$^{\mathrm{T}}$ 分解された形式で近似ヘシアン行列 $\boldsymbol{B}_k$ を表現し，$\boldsymbol{E}_k$ に基づく改良は分解要素の形で(4.6) のように行う．

$$\boldsymbol{L}_{k+1}\boldsymbol{D}_{k+1}\boldsymbol{L}^{\mathrm{T}}{}_{k+1} = \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{L}_k^{\mathrm{T}} + \boldsymbol{E}_k \qquad (4.6)$$

(4.6)で，$\boldsymbol{D}_{k+1}$ の対角要素をすべて正に保持することにより，正定値を保証することが，改訂準ニュートン法の特徴である．

本サブルーチンの計算手順

① ヘシアン行列の初期設定 $(\boldsymbol{B}_0 = \boldsymbol{I}_n)$

② 傾斜ベクトル $\boldsymbol{g}_k$ の計算

③ 探索ベクトル $\boldsymbol{p}_k$ の決定 $\left(\boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{L}_k{}^{T}\boldsymbol{p}_k = -\boldsymbol{g}_k\right)$
この方程式は，サブルーチン LDLX を用いて解く．

④ 直線探索 $\left(x_{k+1} = x_k + \boldsymbol{\alpha}_k\boldsymbol{p}_k\right)$

⑤ 近似ヘシアン行列の改良
$\left(\boldsymbol{L}_{k+1}\boldsymbol{D}_{k+1}\boldsymbol{L}^{T}{}_{k+1} = \boldsymbol{L}_k\boldsymbol{D}_k\boldsymbol{L}_k{}^{T} + \boldsymbol{E}_k\right)$

ここで，②～⑤について，k=0, 1, ...と反復する．

アルゴリズム上の留意点

① 傾斜ベクトル $\boldsymbol{g}_k$ の計算
本サブルーチンでは，$\boldsymbol{g}_k$ の計算は前進差分(4.7)及び中心差分 (4.8) により近似している．

$$g_k^i \approx \left(f\left(x_k + he_i\right) - f\left(x_k\right)\right)/h \qquad (4.7)$$

$$g_k^i \approx \left(f\left(x_k + he_i\right) - f\left(x_k - he_i\right)\right)/2h \qquad (4.8)$$

ここで，
$g_k = \left(g_k^1, g_k^2, ..., g_k^n\right)^T$,
$x_k = \left(x_k^1, x_k^2, ..., x_k^n\right)^T$,
$\boldsymbol{e_i}$ は第 i 単位ベクトル
$h = \sqrt{u}$，u: 丸め誤差の単位

反復の初めは (4.7) により近似するが，反復が収束域に近づいた場合は (4.8)による近似に切り替える．

② 直線探索 (α_k の選択)
直線探索は，原理的には探索方向 $\boldsymbol{p}_k$ に沿って，関数 $f(\boldsymbol{x})$ の最小点を求めること．すなわち α に関する関数 $\varphi(\alpha)$

$$\varphi(\alpha) = f\left(\boldsymbol{x}_k + \alpha\,\boldsymbol{p}_k\right), \quad \alpha \geq 0 \qquad (4.9)$$

を最小にする α_k を求めることである．

本サブルーチンでは，$\varphi(\alpha)$ を 2 次補間式により近似し，(4.10)により α_k を推定している．

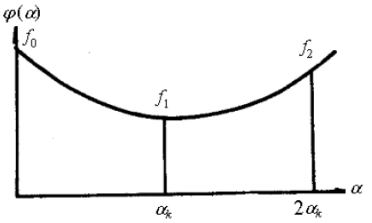
$$\boldsymbol{a_k} \approx \min\{1, 2(f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k-1}))/\boldsymbol{g}_k^T\boldsymbol{p}_k\} \qquad (4.10)$$

(4.10) においては，反復の最終過程では α_k =1 とし，ニュートン法の 2 次の局所的収束性を有効に利用することを意図している．一方(4.10)の第 2 項は，反復の初期過程で，発散を防ぐために $f\left(\boldsymbol{x}_{k+1}\right) < f\left(\boldsymbol{x}_k\right)$ を保証することを意図している．
ところで，(4.10)の第 2 項は，反復の初期過程での関数の変化率が

$$f\left(\boldsymbol{x}_{k+1}\right) - f\left(\boldsymbol{x}_k\right) \approx f\left(\boldsymbol{x}_k\right) - f\left(\boldsymbol{x}_{k-1}\right) \qquad (4.11)$$

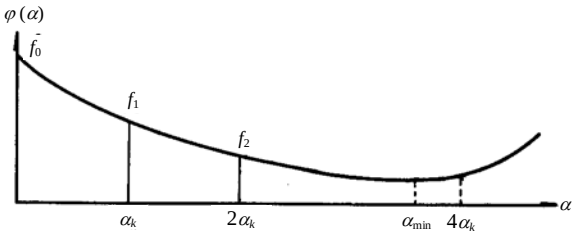
となる傾向にあることを前提としており，正確な 2 次補間式による近似ではないため，本サブルーチンでは α_k に基づき，以下に述べるような外挿，内挿による最小点の探索（直線探索）を行っている．
すなわち，点 $\boldsymbol{x}_{k+1}^{(0)} = \boldsymbol{x}_k$，$\boldsymbol{x}_{k+1}^{(1)} = \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k$，$\boldsymbol{x}_{k+1}^{(2)} = \boldsymbol{x}_k + 2\alpha_k\boldsymbol{p}_k$ に対する関数値を f_0, f_1, f_2 として

(a) $f_0 > f_1$，$f_1 < f_2$ の場合は $x_{k+1}^{(1)}$ を最小点として探索を終了する．

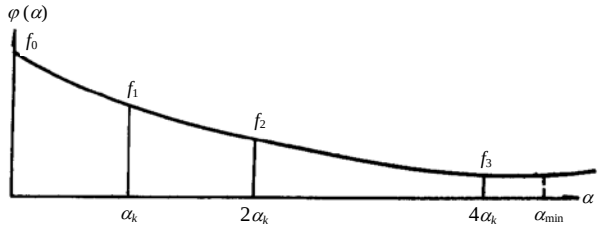


(b) $f_0 > f_1 > f_2$ の場合は，この 3 点に基づく 2 次補間式により，最小点を与える $a_{\min}$ を外挿する．

- $2\alpha_k < \alpha_{\min} < 4\alpha_k$, の場合は，$\boldsymbol{x}_{k+1}^{(2)}$ を最小点として探索を終了する．

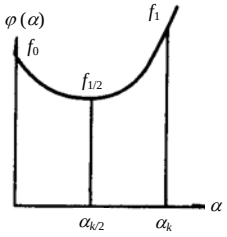


- $4\alpha_k < \alpha_{\min}$ の場合は，
$\alpha_k = 2\alpha_k$
として直線探索の先頭に戻る．



(c) $f_0 < f_1$, の場合は，$\boldsymbol{x}_{k+1}^{(1/2)} = \boldsymbol{x}_k + 1/2 \cdot \alpha_k \boldsymbol{p}_k$ に対する関数値 $f_{1/2}$ を求める．

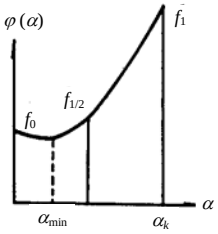
- $f_0 > f_{1/2}$，$f_{1/2} < f_1$ の場合は，$\boldsymbol{x}_{k+1}^{(1/2)}$ を最小点として探索を終了する．



- $f_0 < f_{1/2} < f_1$ の場合は，この 3 点に基づく 2 次補間式により，最小点を与える $\alpha_{\min}$ を内挿し，

$$\alpha_k = \max\left(\alpha_k / 10, \alpha_{\min}\right)$$

として直線探索の先頭に戻る．



上述のとおり，α_k に基づく直線探索を終了するが，関数 $f(\boldsymbol{x})$ が，$\boldsymbol{x}_{k+1}$ において更に減少を続ける場合，
すなわち，

$$\boldsymbol{g}_{k+1}^T \boldsymbol{p}_k < \boldsymbol{g}_k^T \boldsymbol{p}_k \qquad (4.12)$$

なる場合は，新たな探索方向 $\boldsymbol{p}_{k+1}$ と α_{k+1} を決定し，直線探策を反復する．
一方，(4.12)を満たさない場合は，近似ヘシアン行列の改良ステップへ移行する．

③ 収束判定法
本サブルーチンでは，$\boldsymbol{g}_k$ の計算が中心差分(4.8)による近似に切り替わった後の反復過程（線型探索）において，(4.13)を満たすとき反復を中止し，そのときの $\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$ とみなす．

$$\left\| x_{k+1} - x_k \right\|_\infty \le \max\left(1.0, \left\| x_k \right\|_\infty\right) \cdot \mathrm{EPSR} \qquad (4.13)$$

④ 近似ヘシアン行列の改良
改良公式として，BFGS (Broyden - Fletcher - Goldfarb - Shanno) 公式 (4.14)を採用している．

$$\boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \frac{\boldsymbol{r}_k \boldsymbol{r}_k^T}{\boldsymbol{\delta}_k^T \boldsymbol{r}_k} - \frac{\boldsymbol{B}_k \boldsymbol{\delta}_k \boldsymbol{\delta}_k^T \boldsymbol{B}_k}{\boldsymbol{\delta}_k^T \boldsymbol{B}_k \boldsymbol{\delta}_k} \qquad (4.14)$$

ここで，
$$\boldsymbol{r}_k = \boldsymbol{g}_{k+1} - \boldsymbol{g}_k$$
$$\boldsymbol{\delta}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k$$

本サブルーチンでは単位行列を初期近似ヘシアン行列 $\boldsymbol{B}_0$ として反復を開始する．
その k 階段 (k=1,2,…) におけるヘシアン行列 $\boldsymbol{B}_k$ の改良を，(4.15) に基づき，LDL^T 分解された形で次のように行っている．

$$\tilde{\boldsymbol{L}}_k \tilde{\boldsymbol{D}}_k \tilde{\boldsymbol{L}}_k^T = \boldsymbol{L}_k \boldsymbol{D}_k \boldsymbol{L}_k^T + \frac{\boldsymbol{r}_k \boldsymbol{r}_k^T}{\alpha_k \boldsymbol{p}_k^T \boldsymbol{r}_k} \qquad (4.15)$$

$$\boldsymbol{L}_{k+1} \boldsymbol{D}_{k+1} \boldsymbol{L}_{k+1}^T = \tilde{\boldsymbol{L}}_k \tilde{\boldsymbol{D}}_k \tilde{\boldsymbol{L}}_k^T + \frac{\boldsymbol{g}_k \boldsymbol{g}_k^T}{\boldsymbol{p}_k^T \boldsymbol{g}_k} \qquad (4.16)$$

ここで，(4.15)，(4.16) の第 2 項は，各々ランク 1 の行列である．

なお詳細については，参考文献 [34] を参照すること．

D11-20-0101 MING1, DMING1

多変数関数の極小化 （微係数要，準ニュートン法）
CALL MING1 (X, N, FUN, GRAD, EPSR, MAX, F, G, H, VW, ICON)

(1) 機能

n 変数の実関数 $f(\boldsymbol{x})$ とその導関数 $\boldsymbol{g}(\boldsymbol{x})$，及び初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，準ニュートン法により，$f(\boldsymbol{x})$ の極小値を与えるベクトル $\boldsymbol{x}^*$ と，その関数値 $f(\boldsymbol{x}^*)$ を求める．

ただし，$f(\boldsymbol{x})$ は 2 階までの連続な偏導関数を持つものと仮定する．ここで，$x=(x_1,x_2,...,x_n)^T$ である．

また，$n\geq 1$ であること．

(2) パラメタ

X……**入力**．初期ベクトル $\boldsymbol{x}_0$．
出力．ベクトル $\boldsymbol{x}^*$．
大きさ n の 1 次元配列．

N……**入力**．変数の個数 n．

FUN……**入力**．$f(\boldsymbol{x})$ を計算する関数副プログラム名．
［副プログラムの用意の仕方］
FUNCTION FUN(X)
パラメタ
X.....**入力**．任意な変数ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．
関数 FUN には，$f(\boldsymbol{x})$ の値を代入すること．（使用例参照）

GRAD……**入力**．$\boldsymbol{g}(\boldsymbol{x})$ を計算するサブルーチン副プログラム名．
［副プログラムの用意の仕方］
SUBROUTINE GRAD(X, G)
パラメタ
X .....**入力**．変数ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．
G.....**出力**．
$G(1)=\partial f/\partial x_1,...,G(N)=\partial f/\partial x_n$ なる対応をもつ大きさ n の 1 次元配列．
（使用例参照）

EPSR……**入力**．収束判定値 (≥0.0)
0.0 のときは標準値が採用される．
（使用上の注意②参照）

MAX……**入力**．関数 $f(\boldsymbol{x})$ 及び $g(\boldsymbol{x})$ の評価回数の上限 $(\neq 0)$．
（使用上の注意③参照）
出力．実際に評価した回数 (>0).

F ……**出力**．関数値 $f(\boldsymbol{x}^*)$．

G ……**出力**．傾斜ベクトル $g(x^*)$．
大きさ n の 1 次元配列．

H ……**出力**．$\boldsymbol{x}^*$ におけるヘシアン行列の逆行列．
対称行列用圧縮モードで格納される．
大きさ $n(n+1)/2$ の 1 次元配列．

VW ……**作業領域**．大きさ $3n+1$ の 1 次元配列．

ICON ……**出力**．コンディションコード．
表 MING1-1 参照．

表 MING1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	指定された関数の評価回数内で，収束条件が満たされなかった．	パラメタ X, F, G, H には，最後の値を格納する．
20000	計算の過程で，$g_k^T p_k \geq 0$ となり，関数の局所的減少がみたされなかった．（手法概要（4.5）式参照）EPSR が小さすぎる．	処理を打ち切る．（パラメタ X, F には，最後の値を格納する．）
25000	ある探索方向に沿って関数が単調に減少している．	処理を打ち切る．
30000	N<1, EPSR<0.0 又は MAX=0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AFMAX , AMACH, MGSSL, MSV
② FORTRAN 基本関数 ... ABS, SQRT

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で，引数 FUN 及び GRAD に相当する副プログラム名を EXTERNAL 文で宣言すること．
② EPSR の与え方
本サブルーチンの収束判定は，反復ベクトル $\boldsymbol{x}_k$ において，

$$\|x_{k+1}-x_k\|_\infty \leq \max(1.0,\|x_k\|_\infty)\cdot \mathrm{EPSR}$$

となった場合に，$\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$ とみなし反復を終了する．
本サブルーチンでは，関数 $f(\boldsymbol{x})$ が極小点 $\boldsymbol{x}^*$ の近傍で近似的に 2 次関数であることを前提としており，関数値 $f(\boldsymbol{x}^*)$ を丸め誤差程度まで正しく求めるためには，

$$\mathrm{EPSR}\approx\sqrt{u}\text{，}u\text{ は丸め誤差の単位}$$

程度に与えるのが適当である．
EPSR の標準値は $\sqrt{u}$ ／8.0 である．

③ MAX の与え方

関数の評価回数は，変数ベクトル x に対して，$f(\boldsymbol{x})$ を計算することを 1 回，$\boldsymbol{g}(\boldsymbol{x})$ を計算することを n 回と数える．

関数の評価回数は，関数の性質，初期ベクトル，収束判定値などに依存する．

一般には，初期ベクトルが良く，収束判定値として標準値採用した場合は，MAX=400$\cdot n$ 程度が妥当である．

指定評価回数内で収束条件が満たされず，ICON=10000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ MAX には，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと．

c. 使用例

$f(\boldsymbol{x})=(1-x_1)^2+100\left(x_2-x_1{}^2\right)^2$ の最小点 $\boldsymbol{x}^*$ を初期ベクトル $\boldsymbol{x}_0=(-1.2,1.0)^T$ により求める．

```
C     **EXAMPLE**
      DIMENSION X(2),G(2),H(3),VW(7)
      EXTERNAL ROSEN,ROSENG
      X(1)=-1.2
      X(2)=1.0
      N=2
      EPSR=1.0E-4
      MAX=400*2
      CALL MING1(X,N,ROSEN,ROSENG,EPSR,
     *           MAX,F,G,H,VW,ICON)
      WRITE(6,600) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,610) F,MAX
      WRITE(6,620) (I,X(I),I,G(I),I=1,N)
      STOP
  600 FORMAT('1','*ICON=',I5)
  610 FORMAT(' ','*F=',E15.5,' MAX=',I5/)
  620 FORMAT(/' X(',I2,')=',E15.7,2X,
     *       ' G(',I2,')=',E15.7)
      END

C     OBJECTIVE FUNCTION
      FUNCTION ROSEN(X)
      DIMENSION X(2)
      ROSEN=(1.0-X(1))**2+100.0*
     *      (X(2)-X(1)*X(1))**2
      RETURN
      END
C     GRADIENT VECTOR
      SUBROUTINE ROSENG(X,G)
      DIMENSION X(2),G(2)
      G(1)=-2.0*(1.0-X(1))
     *-400.0*X(1)*(X(2)-X(1)*X(1))
      G(2)=200.0*(X(2)-X(1)*X(1))
      RETURN
      END
```

(4) 手法概要

n 変数の実関数 $f(\boldsymbol{x})$ とその導関数 $\boldsymbol{g}(\boldsymbol{x})$，及び初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，準ニュートン法により，$f(\boldsymbol{x})$ の極小値を与えるベクトル $\boldsymbol{x}^*$ と関数値 $f(\boldsymbol{x}^*)$ を求める．

準ニュートン法

関数 $f(\boldsymbol{x})$ が 2 次関数の場合，極小点 $\boldsymbol{x}^*$ の近傍でのテーラー展開は，(4.1)で与えられる．

$$f(\boldsymbol{x})=f(\boldsymbol{x}^*)+\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}^*)^T\boldsymbol{B}(\boldsymbol{x}-\boldsymbol{x}^*) \qquad (4.1)$$

(4.1) で $\boldsymbol{B}$ は $\boldsymbol{x}^*$ における $f(\boldsymbol{x})$ のヘシアン行列である．行列 $\boldsymbol{B}$ が正定値であるならば，(4.1)は一意的に最小値を持つ．この場合，$\boldsymbol{x}_k$ を $\boldsymbol{x}^*$ の近傍の任意な点とし，$\boldsymbol{x}_k$ における $f(\boldsymbol{x})$ の傾斜ベクトルを $\boldsymbol{g}_k$ と表すと，(4.1) より $\boldsymbol{x}^*$ は次式により求められる．

$$\boldsymbol{x}^*=\boldsymbol{x}_k-\boldsymbol{H}\boldsymbol{g}_k \qquad (4.2)$$

ただし，$\boldsymbol{H}$ はヘシアン行列 $\boldsymbol{B}$ の逆行列である．

一方，関数 $f(\boldsymbol{x})$ が 2 次関数でない場合でも，$\boldsymbol{x}^*$ の近傍では近似的に 2 次関数とみなすことができ，(4.2) に基づく反復公式を導くことができる．しかし，行列 $\boldsymbol{B}$ の逆行列を直接求めることは，計算量が多く効率的ではないため，一般にはその近似行列を設定し，反復の過程で改良していく方法が考えられる．

準ニュートン法は，ヘシアン行列の逆行列 $\boldsymbol{H}$ の近似行列を $\boldsymbol{H}_k$ として，(4.3)，(4.4)の反復公式により極小点 $\boldsymbol{x}^*$ を求める方法である．

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\alpha_k\boldsymbol{p}_k \qquad (4.3)$$

$$\boldsymbol{H}_{k+1}=\boldsymbol{H}_k+\left(1+\boldsymbol{r}_k^T\boldsymbol{H}^k\boldsymbol{r}^k/\delta_k^T\boldsymbol{r}^k\right)\left(\delta_k\delta_k^T/\delta_k^T\boldsymbol{r}_k\right)$$

$$-\left(\boldsymbol{H}_k\boldsymbol{r}_k\delta_k^T+\delta_k\boldsymbol{r}_k^T\boldsymbol{H}_k\right)/\delta_k^T\boldsymbol{r}_k \qquad (4.4)$$

ここで，

$$\boldsymbol{p}_k=-\boldsymbol{H}_k\boldsymbol{g}_k,\qquad \boldsymbol{r}_k=\boldsymbol{g}_{k+1}-\boldsymbol{g}_k,$$
$$\delta_k=\boldsymbol{x}_{k+1}-\boldsymbol{x}_k,$$
$$k=0,1,2,\cdots\cdots$$

$\boldsymbol{H}_0$ は，任意の正定値行列である．(4.3)で，$\boldsymbol{p}_k$ は $\boldsymbol{x}_k$ から極小点に向う探索方向を示すベクトルであり，α_k は $f(\boldsymbol{x}_k+\alpha_k\boldsymbol{p}_k)$ が局所的最小となるように定める（直線探索）定数である．

(4.3)，(4.4) による反復において，探索方向 $\boldsymbol{p}_k$ が下降方向（関数 $f(\boldsymbol{x})$ が $\boldsymbol{x}_k$ において $\boldsymbol{p}_k$ の方向に局所的に減少する）にあるためには，

$$\boldsymbol{g}_k^T \boldsymbol{p}_k < 0 \qquad (4.5)$$

が要求される.

本サブルーチンの計算手順

① ヘシアン行列の近似逆行列の初期値設定 $(\boldsymbol{H}_0 = \boldsymbol{I}_n)$
② 傾斜ベクトル $\boldsymbol{g}_k$ の計算
③ 探索ベクトル $\boldsymbol{p}_k$ の計算 $\left(\boldsymbol{p}_k = -H_k \boldsymbol{g}_k\right)$
④ 直線探索 $\left(\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k\right)$
⑤ 近似逆行列 $\boldsymbol{H}_k$ の改良（(4.4) より $\boldsymbol{H}_{k+1}$ を求める）
ここで，②～⑤について，k=0, 1, ...と反復する.

アルゴリズム上の留意点

① 近似逆行列の初期設定
ヘシアン行列に対する初期近似逆行列 $\boldsymbol{H}_0$ として本サブルーチンでは単位行列 $\boldsymbol{I}_n$ を採用している．逆行列 $\boldsymbol{H}_k$ は(4.4) により反復ごとに改良を加えるが，初期の $\boldsymbol{H}_0$ から $\boldsymbol{H}_1$ を求める段階については,

$$\boldsymbol{H}_0 = s\boldsymbol{I}_n$$
ここで， $s = \boldsymbol{\delta}^T \boldsymbol{r}_0 / \boldsymbol{r}_0^T \boldsymbol{r}_0$

と再設定した後，(4.4)により $\boldsymbol{H}_1$ を求める.

② 直線探索 (α_k の選択)
直線探索は，原理的には探索方向 $\boldsymbol{p}_k$ に沿って，関数 $f(\boldsymbol{x})$ の最小点を求めること．すなわち α に関する関数 $\varphi(\alpha)$

$$\varphi(\alpha) = f\left(\boldsymbol{x}_k + \alpha \boldsymbol{p}_k\right) \quad , \quad \alpha \geq 0 \qquad (4.6)$$

を最小にする α_k を求めることである.
本サブルーチンでは，$\varphi(\alpha)$ を 2 次補間式により近似し，(4.7)により α_k を推定している.

$$\alpha_k \approx \min\left\{1, 2\left(f\left(x_k\right) - f\left(x_{k-1}\right)\right) / \boldsymbol{g}_k^T \boldsymbol{p}_k\right\} \qquad (4.7)$$

(4.7) においては，反復の最終過程では α_k=1 とし，ニュートン法の 2 次の局所的収束性を有効に利用することを意図している．一方，(4.7) の第 2 項は，反復の初期過程で，発散を防ぐために $f(\boldsymbol{x}_{k+1}) < f(\boldsymbol{x}_k)$を保証することを意図している．ところで，(4.7)の第 2 項は，反復の初期過程での関数の変化率が

$$f\left(\boldsymbol{x}_{k+1}\right) - f\left(\boldsymbol{x}_k\right) \approx f\left(\boldsymbol{x}_k\right) - f\left(\boldsymbol{x}_{k-1}\right) \qquad (4.8)$$

となる傾向にあることを前提としており，正確な 2 次補間式による近似ではないため，本サブルーチンでは α_k に基づき，以下に述べるような外挿，内挿による最小点の探索（直線探索）を行っている.

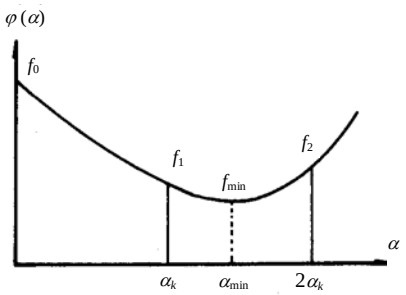
(a) まず，(4.7)より α_k を求める.

(b) 点 $\boldsymbol{x}_{k+1}{}^{(0)} = \boldsymbol{x}_k, \boldsymbol{x}^{(1)}{}_{k+1} = \boldsymbol{x}_k + \alpha_k p_k, \boldsymbol{x}_{k+1}^{(2)} = \boldsymbol{x}_k + 2\alpha_k \boldsymbol{p}_k$ に対する関数値を求め,それらを f_0, f_1, f_2 とする.

(c) $f_0 > f_1$, $f_1 < f_2$ の場合は，この 3 点に対する 2 次補間式により，最小点を与える点 $\alpha_{\min}$ を内挿する.

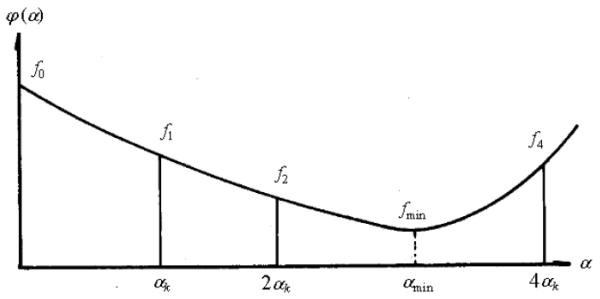
$$\alpha_{\min} = \alpha_k - \frac{\alpha_k}{2} \cdot \frac{f_2 - f_0}{f_0 - 2f_1 + f_2} \qquad (4.9)$$

この $\alpha_{\min}$ を最終的な α_k として探索を終了し，その時の関数値 $f_{\min}$ を最小値とする.

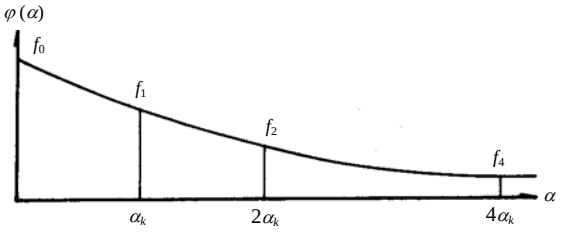


(d) $f_0 > f_1 > f_2$ の場合は，点 $\boldsymbol{x}_{k+1}^{(4)} = \boldsymbol{x}_k + 4\alpha_k \boldsymbol{p}_k$ に対する関数値を求め，それを f_4 とし，

- $f_2 < f_4$ ならば 3 点 f_0, f_1, f_2 に対する 2 次補式(4.9)より，$\alpha_{\min}$ を外挿する.
そこで，$4\alpha_k < \alpha_{\min}$ ならば $\alpha_k = 2\alpha_k$ として，(b)へ戻る.
一方，$2\alpha_k < \alpha_{\min} < 4\alpha_k$ ならば，この $\alpha_{\min}$ を最終的な α_k として探索を終了し，その時の関数値 f_{min} を最小値とする.



- $f_2 > f_4$ の場合は，$\alpha_k = 2\alpha_k$ として(b)へ戻る.

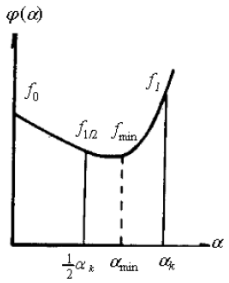


(e)　$f_0 \le f_1$ の場合は，点 $\boldsymbol{x}_{k+1}^{\left(\frac{1}{2}\right)} = \boldsymbol{x}_k + \frac{1}{2}\alpha_k \boldsymbol{p}_k$ に対する関数値を求め，それを $f_{1/2}$ とし，

- $f_0 > f_{1/2}$，$f_{1/2} < f_1$ ならば，この 3 点に対する 2 次補間式より，$\alpha_{\min}$ を内挿する．

$$\alpha_{\min} = \frac{\alpha_k}{2} - \frac{\alpha_k}{4} \cdot \frac{f_1 - f_0}{f_0 - 2f_{1/2} + f_1} \qquad (4.10)$$

この $\alpha_{\min}$ を最終的な α_k として探索を終了し，そのときの関数値 $f_{\min}$ を最小値とする．



- $f_0 < f_{1/2}$ の場合は，$f_1 = f_{1/2}$，$\alpha_k = \frac{1}{2}\alpha_k$ として (e)の先頭へ戻る．

上述のとおり，α_k に基づく直線探索を終了するが，関数 $f(\boldsymbol{x})$ が，$\boldsymbol{x}_{k+1}$ において更に減少を続ける場合，すなわち，

$$\boldsymbol{g}_{k+1}^T \boldsymbol{p}_k < \boldsymbol{g}_k^T \boldsymbol{p}_k \qquad (4.11)$$

なる場合は，新たな探索方向 $\boldsymbol{p}_{k+1}$ と α_{k+1} を決定し，直線探策を反復する．
一方，(4.11)を満たさない場合は，ヘシアン行列の近似逆行列の改良ステップへ移行する．

③　収束判定法
本サブルーチンでは，反復ベクトル $\boldsymbol{x}_k$ と $\boldsymbol{x}_{k+1}$ が，

$$\left\| \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \right\|_\infty = \max\left(1.0, \left\| \boldsymbol{x}_k \right\|_\infty\right) \cdot \mathrm{EPSR}$$

を満たすとき反復を中止し，そのときの $\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$ とみなす．

なお詳細については，参考文献 [35] を参照すること．

A51-14-0101 MSBV, DMSBV

実対称バンド行列と実ベクトルの積
CALL MSBV (A, N, NH, X, Y, ICON)

(1) 機能

$n \times n$，上，下バンド幅 h の実対称バンド行列 $\boldsymbol{A}$ と，実ベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$ を求める．

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{1.1}$$

ここで，$\boldsymbol{x}$ 及び $\boldsymbol{y}$ は n 次元ベクトルである．
$n > h \geq 0$ であること．

(2) パラメタ

A …… **入力**．行列 $\boldsymbol{A}$．
対称バンド行列用圧縮モード．
大きさ $n(h+1) - h(h+1)/2$ の 1 次元配列．

N …… **入力**．行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意①参照）．

NH …… **入力**．上，下バンド幅 h．

X …… **入力**．ベクトル $\boldsymbol{x}$．
大きさ n の 1 次元配列．

Y …… **出力**．行列 $\boldsymbol{A}$ とベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$．
大きさ n の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 MSBV-1 参照．

表 MSBV-1 コンディションコード．

コード	意味	処理内容
0	エラーなし．	
30000	N = 0, NH < 0 又は NH ≥ |N|であった．	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... MGSSL

② FORTRAN 基本関数 .... IABS

b. 注意

① 本サブルーチンは通常

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{3.1}$$

の計算を主とするが，N = $-n$ として，任意のベクトル $\boldsymbol{y}'$ をパラメタ Y に与えると

$$\boldsymbol{y} = \boldsymbol{y}' - \boldsymbol{A}\boldsymbol{x} \tag{3.2}$$

なる計算ができる．
具体的には，連立 1 次方程式の残差ベクトル

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \tag{3.3}$$

の計算等を行う場合に利用できる．使用例参照．

c. 使用例

n 元の実係数連立 1 次方程式（正値対称バンド行列）

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.4}$$

をサブルーチン LSBX により解き，その結果より残差ベクトル $\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$ を求める．
$n \leq 100$ の場合

```
C     **EXAMPLE**
      DIMENSION A(5050),X(100),
     *          Y(100),W(5050)
      READ(5,500) N,NH
      IF(N.EQ.0) STOP
      NH1=NH+1
      NT=N*NH1-NH*NH1/2
      READ(5,510) (A(I),I=1,NT)
      READ(5,510) (X(I),I=1,N)
      WRITE(6,600) N,NH
      L=1
      LE=0
      DO 10 I=1,N
      LE=LE+MIN0(I,NH1)
      JS=MAX0(1,I-NH1)
      WRITE(6,610) I,JS,(A(J),J=L,LE)
      L=LE+1
   10 CONTINUE
      WRITE(6,620) (I,X(I),I=1,N)
      EPSZ=1.0E-6
      ISW=1
      DO 20 I=1,N
      Y(I)=X(I)
   20 CONTINUE
      DO 30 I=1,NT
      W(I)=A(I)
   30 CONTINUE
      CALL LSBX(A,N,NH,X,EPSZ,ISW,ICON)
      IF(ICON.GE.20000) GOTO 50
      WRITE(6,630) (I,X(I),I=1,N)
      CALL MSBV(W,-N,NH,X,Y,ICON)
      IF(ICON.NE.0) GOTO 50
      WRITE(6,640) (I,Y(I),I=1,N)
      DN=0.0
      DO 40 I=1,N
      DN=DN+Y(I)*Y(I)
   40 CONTINUE
      DN=SQRT(DN)
      WRITE(6,650) DN
      STOP
   50 WRITE(6,660) ICON
      STOP
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1','COEFFICIENT MATRIX'
     * /' ','N=',I5,3X,'NH=',I5)
  610 FORMAT(/5X,'(',I3,',',I3,')',4E17.8
     * /(10X,4E17.8))
  620 FORMAT(/' ','CONSTANT VECTOR'
     * /(5X,4('(',I3,')',E17.8,5X)))
  630 FORMAT(/' ','SOLUTION VECTOR'
     * /(5X,4('(',I3,')',E17.8,5X)))
  640 FORMAT(/' ','RESIDUAL VECTOR'
     * /(5X,4('(',I3,')',E17.8,5X)))
  650 FORMAT(/' ','NORM=',E17.8)
  660 FORMAT(/' ','ICON=',I5)
      END
```

(4) 手法概要

$n \times n$，上，下バンド幅 h の実対称バンド行列 $A=(a_{ij})$ と n 次元のベクトル $x=(x_j)$ の積 $y=(y_i)$の要素を(4.1)により計算する．

$$y_i = \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1,...,n \tag{4.1}$$

ただし，本サブルーチンではバンド行列であることを考慮して，(4.2)により計算している．

$$y_i = \sum_{j=\max(1,i-h)}^{\min(i+h,n)} a_{ij} x_j, \quad i = 1,...,n \tag{4.2}$$

本サブルーチンでは(4.2)の積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている．

A-21-12-0401 MSGM, DMSGM

行列の積（実対称行列・実行列）
CALL MSGM (A, B, KB, C, KC, N, VW, ICON)

(1) 機能

$n \times n$ の実対称行列 $\boldsymbol{A}$ と，$n \times n$ の実行列 $\boldsymbol{B}$ の積 $\boldsymbol{C}$ を求める．

$$\boldsymbol{C} = \boldsymbol{AB}$$

ここで，$\boldsymbol{C}$ は $n \times n$ の実行列である． $n \geq 1$ であること．

(2) パラメタ

A …… **入力**．行列 $\boldsymbol{A}$.
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

B …… **入力**．行列 $\boldsymbol{B}$.
B (KB, N)なる 2 次元配列.

KB …… **入力**．配列 **B** の整合寸法 (≥ N).

C …… **出力**．行列 $\boldsymbol{C}$.
C (KC, N)なる 2 次元配列.
（使用上の注意①参照）

KC …… **入力**．配列 **C** の整合寸法 (≥ N).

N …… **入力**．行列 $\boldsymbol{A}$, $\boldsymbol{B}$ 及び $\boldsymbol{C}$ の次数 n.

VW …… **作業領域**．大きさ n の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 MSGM-1 参照..

表 MSGM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N<1, KB<N 又は KC<N であった.	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... CSGM, MGGM, MGSSL
② FORTRAN 基本関数 ... なし.

b. 注意
① 領域の節約について
配列 **A** 上の内容を保存する必要のない場合は, EQUIVALENCE 文により
EQUIVALENCE (A(1), C(1,1))
と宣言すれば領域が節約できる．使用例参照.

c. 使用例
実対称行列 $\boldsymbol{A}$ と，実行列 $\boldsymbol{B}$ の積 $\boldsymbol{C}$ を求める．ただし行列 $\boldsymbol{C}$ は，$\boldsymbol{A}$ を入力した領域に得るものとする．$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),B(100,100),
     *          C(100,100),VW(100)
      EQUIVALENCE (A(1),C(1,1))
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
   10 READ(5,100) N
      IF(N.EQ.0) STOP
      WRITE(6,150)
      NT=N*(N+1)/2
      READ(5,200) (A(I),I=1,NT)
      READ(5,200) ((B(I,J),J=1,N),I=1,N)
      CALL PSM(IA,1,A,N)
      CALL PGM(IB,1,B,100,N,N)
      CALL MSGM(A,B,100,C,100,N,VW,ICON)
      WRITE(6,250) ICON
      IF(ICON.NE.0) GOTO 10
      CALL PGM(IC,1,C,100,N,N)
      GOTO 10
  100 FORMAT(I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** MATRIX MULTIPLICATION **')
  250 FORMAT(//10X,'** MSGM ICON=',I5)
      END
```

本使用例中のサブルーチン PSM 及び PGM は，実対称行列及び実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている.

A21-12-0301 MSSM, DMSSM

行列の積 （実対称行列）
CALL MSSM (A, B, C, KC, N, VW, ICON)

(1) 機能

$n \times n$ の実対称行列 $\boldsymbol{A}$ と実対称行列 $\boldsymbol{B}$ の積 $\boldsymbol{C}$ を求める.

$$\boldsymbol{C} = \boldsymbol{AB}$$

ここで $\boldsymbol{C}$ は，$n \times n$ の実行列である．$n \geq 1$ であること.

(2) パラメタ

A …… **入力**．行列 $\boldsymbol{A}$.
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

B …… **入力**．行列 $\boldsymbol{B}$.
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

C …… **出力**．行列 $\boldsymbol{C}$.
C (KC, N)なる 2 次元配列.
（使用上の注意①参照）

KC …… **入力**．配列 **C** の整合寸法 (≥ N).

N …… **入力**．行列 $\boldsymbol{A}$, $\boldsymbol{B}$ 及び $\boldsymbol{C}$ の次数 n.

VW …… **作業領域**．大きさ n の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 MSSM-1 参照.

表 MSSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N<1 又は KC<N であった	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... CSGM, MGSM, MGSSL

② FORTRAN 基本関数 ... なし

b. 注意

① 領域の節約について
配列 $\boldsymbol{A}$ 上の内容を保存する必要のない場合は，EQUIVALENCE 文により
EQUIVALENCE (A(1), C(1.1))
と宣言すれば領域が節約できる．使用例参照.

c. 使用例
実対称行列 $\boldsymbol{A}$，$\boldsymbol{B}$ の積 $\boldsymbol{C}$ を求める．ただし行列 $\boldsymbol{C}$ は，$\boldsymbol{A}$ を入力した領域に得るものとする.
$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),B(5050),
     *          C(100,100),VW(100)
      EQUIVALENCE (A(1),C(1,1))
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
   10 READ(5,100) N
      IF(N.EQ.0) STOP
      WRITE(6,150)
      NT=N*(N+1)/2
      READ(5,200) (A(I),I=1,NT)
      READ(5,200) (B(I),I=1,NT)
      CALL MSSM(A,B,C,100,N,VW,ICON)
      IF(ICON.NE.0) GOTO 10
      CALL PSM(IA,1,A,N)
      CALL PSM(IB,1,B,N)
      CALL PGM(IC,1,C,100,N,N)
      GOTO 10
  100 FORMAT(I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** MATRIX MULTIPLICATION **')
      END
```

本使用例中のサブルーチン PSM 及び PGM は，実対称行列及び実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている.

A21-14-0101 MSV, DMSV

実対称行列と実ベクトルの積
CALL MSV (A, N, X, Y, ICON)

(1) 機能

$n \times n$ の実対称行列 $\boldsymbol{A}$ と，実ベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$ を求める．

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{1.1}$$

ここで，$\boldsymbol{x}$ 及び $\boldsymbol{y}$ は n 次元ベクトルである．
$n \geq 1$ であること．

(2) パラメタ

A …… **入力**．行列 $\boldsymbol{A}$．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．

N …… **入力**．行列 $\boldsymbol{A}$ の次数 n．
（使用上の注意①参照）

X …… **入力**．ベクトル $\boldsymbol{x}$．大きさ n の 1 次元配列．

Y …… **出力**．行列 $\boldsymbol{A}$ とベクトル $\boldsymbol{x}$ の積 $\boldsymbol{y}$．
大きさ n の 1 次元配列．

ICON …… **出力**．コンディションコード．表 MSV-1 参照．

表 MSV-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	N = 0 であった	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .... MGSSL
② FORTRAN 基本関数 ....IABS

b. 注意
① 本サブルーチンは通常

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{3.1}$$

の計算を主とするが，$N = -n$ として，任意のベクトル $\boldsymbol{y}'$ をパラメタ **Y** に与えること

$$\boldsymbol{y} = \boldsymbol{y}' - \boldsymbol{A}\boldsymbol{x} \tag{3.2}$$

なる計算ができる．
具体的には，連立 1 次方程式の残差ベクトル

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \tag{3.3}$$

の計算等を行う場合に利用できる．使用例参照．

c. 使用例
n 元の実係数連立 1 次方程式（正値対称行列）

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.4}$$

をサブルーチン LSX により解き，その結果より残差ベクトル $\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$ を求める．
$n \leq 100$ の場合

```
C     **EXAMPLE**
      DIMENSION A(5050),X(100),
     *          Y(100),W(5050)
      READ(5,500) N
      IF(N.EQ.0) STOP
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      READ(5,510) (X(I),I=1,N)
      WRITE(6,600) N
      L=1
      LE=0
      DO 10 I=1,N
      LE=LE+I
      WRITE(6,610) I,(A(J),J=L,LE)
      L=LE+1
   10 CONTINUE
      WRITE(6,620) (I,X(I),I=1,N)
      EPSZ=1.0E-6
      ISW=1
      DO 20 I=1,N
      Y(I)=X(I)
   20 CONTINUE
      DO 30 I=1,NT
      W(I)=A(I)
   30 CONTINUE
      CALL LSX(A,N,X,EPSZ,ISW,ICON)
      WRITE(6,630) (I,X(I),I=1,N)
      CALL MSV(W,-N,X,Y,ICON)
      IF (ICON.NE.0) GO TO 50
      WRITE(6,640) (I,Y(I),I=1,N)
      DN=0.0
      DO 40 I=1,N
      DN=DN+Y(I)*Y(I)
   40 CONTINUE
      DN=SQRT(DN)
      WRITE(6,650) DN
   50 WRITE(6,660) ICON
      STOP
  500 FORMAT(I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1','COEFFICIENT MATRIX'
     */' ','ORDER=',I5)
  610 FORMAT(/5X,'(',I3,')',4E17.8/
     *(10X,4E17.8))
  620 FORMAT(/' ','CONSTANT VECTOR'
     */(5X,4('(',I3,')',E17.8,5X)))
  630 FORMAT(/' ','SOLUTION VECTOR'
     */(5X,4('(',I3,')',E17.8,5X)))
  640 FORMAT(/' ','RESIDUAL VECTOR'
     */(5X,4('(',I3,')',E17.8,5X)))
  650 FORMAT(/' ','NORM=',E17.8)
  660 FORMAT(/' ','ICON=',I5)
      END
```

(4) 手法概要

$n \times n$ の実対称行列 $A = (a_{ij})$ と n 次元のベクトル $\boldsymbol{x}=(x_j)$ の積 $\boldsymbol{y}=(y_i)$ の要素を(4.1)により計算する.

$$y_i = \sum_{j=1}^{n} a_{ij} x_j, \ i = 1,...,n \qquad (4.1)$$

本サブルーチンでは (4.1) 積和計算を精度を上げて行うことにより，丸め誤差の影響を少なくしている.

I11-91-0101 NDF, DNDF

正規分布関数 $\phi(x)$
CALL NDF (X, F, ICON)

(1) 機能

正規分布関数 $\phi(x)=\frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}}dt$ の値を，関係式

$$\phi(x)=\mathrm{erf}\left(x/\sqrt{2}\right)/2 \qquad (1.1)$$

より計算する．

(2) パラメタ

X …… **入力**．独立変数 *x*.

F …… **出力**．関数値 $\phi(x)$.

ICON…… **出力**．コンディションコード．
表 NDF-1 参照.

表NDF-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ...MGSSL

② FORTRAN 基本関数 ...ERF

b. 注意

① 引数 X の範囲に制限はない.

② 余正規分布関数 $\psi(x)$ との間の関係

$$\phi(x)=1/2-\psi(x) \qquad (3.1)$$

を利用すれば $\phi(x)$ の値をサブルーチン NDFC を用いて計算することもできる．ただし，このように計算すると $|x|>2$ の範囲の *x* については本サブルーチン NDF を直接用いるのに比べ精度が悪くなるので注意すること.

c. 使用例

$\phi(x)$ の値を，0.0 から 10.0 での 0.1 刻みの *x* の値に対して計算して数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=FLOAT(K-1)/10.0
      CALL NDF(X,F,ICON)
      WRITE(6,610) X,F
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF NORMAL ',
     * 'DISTRI','BUTION FUNCTION'//
     * 6X,'X',7X,'NDF(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
      END
```

(4) 手法概要

正規分布関数:

$$\phi(x)=\frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}}dt \qquad (4.1)$$

において，$t/\sqrt{2}=u$ と変数変換すれば，(4.1)は，

$$\phi(x)=\frac{1}{\sqrt{2\pi}}\int_0^{\frac{x}{\sqrt{2}}} e^{-u^2}\sqrt{2}du$$

$$=\frac{1}{2}\frac{2}{\sqrt{\pi}}\int_0^{\frac{x}{\sqrt{2}}} e^{-u^2}du$$

したがって，$\mathrm{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-u^2}du$ より，

$$\phi(x)=\frac{1}{2}\mathrm{erf}\left(x/\sqrt{2}\right) \qquad (4.2)$$

となる．本サブルーチン NDF では FORTRAN 関数 ERF を用いて，(4.2)より $\phi(x)$ の計算を行う.

I11-91-0201 NDFC, DNDFC

余正規分布関数 $\psi(x)$
CALL NDFC (X, F, ICON)

(1) 機能

余正規分布関数

$$\psi(x)=\frac{1}{\sqrt{2\pi}}\int_x^{\infty} e^{-\frac{t^2}{2}}dt$$

の値を，関係式

$$\psi(x)=\mathrm{erfc}\left(x/\sqrt{2}\right)/2 \qquad (1.1)$$

より計算する．

(2) パラメタ

X ....**入力**．独立変数 x.

F .....**出力**．関数値 $\psi(x)$.

ICON .... **出力**．コンディションコード．
表 NDFC-1 参照.

表NDFC-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... ERFC

b. 注意

① 引数 X の範囲に制限はない．

② 正規分布関数 $\phi(x)$ との間の関係.

$$\psi(x)=\frac{1}{2}-\phi(x) \qquad (3.1)$$

を利用すれば $\psi(x)$ の値をサブルーチン NDF を用いて計算することもできる．ただし，このように計算すると $|x|>2$ の範囲の x については本サブルーチン NDFC を直接用いるのに比べ精度が悪くなるので注意すること．

c. 使用例

$\psi(x)$ の値を，0.0 から 10.0 までの 0.1 刻みの x の値に対して計算して数表を作る．

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=FLOAT(K-1)/10.0
      CALL NDFC(X,F,ICON)
      WRITE(6,610) X,F
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF ',
     * 'COMPLEMENTARY ',
     * 'NORMAL DISTRIBUTION FUNCTION'
     * //6X,'X',7X,'NDFC(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
      END
```

(4) 手法概要

余正規分布関数:

$$\psi(x)=\frac{1}{\sqrt{2\pi}}\int_x^{\infty} e^{-\frac{t^2}{2}}dt \qquad (4.1)$$

において，$t/\sqrt{2}=u$ と変数変換すれば，(4.1)は

$$\psi(x)=\frac{1}{\sqrt{2\pi}}\int_{\frac{x}{\sqrt{2}}}^{\infty} e^{-u^2}\sqrt{2}du$$

$$\frac{1}{2}\frac{2}{\sqrt{\pi}}\int_{\frac{x}{\sqrt{2}}}^{\infty} e^{-u^2}du$$

したがって，$\mathrm{erfc}\,(x)=\frac{2}{\sqrt{\pi}}\int_x^{\infty} e^{-u^2}du$ より

$$\psi(x)=\frac{1}{2}\mathrm{erfc}\left(x/\sqrt{2}\right) \qquad (4.2)$$

となる．本サブルーチン NDFC では FORTRAN 関数 ERFC を用いて，(4.2)より $\psi(x)$ の計算を行う．

D31-20-0101 NLPG1, DNLPG1

非線形計画問題(微係数要，パウエル法)
CALL NLPG1 (X, N, FUN, GRAD, FUNC, JAC, M, EPSR, MAX, F, VW, K, IVW, ICON)

(1) 機能

n 変数の実関数 $f(\boldsymbol{x})$ とその導関数 $\boldsymbol{g}(\boldsymbol{x})$,及び初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，制約条件

$$c_i(\boldsymbol{x})=0,\ i = 1, 2, ..., m_1, \qquad (1.1)$$
$$c_i(\boldsymbol{x})\geq 0,\ i = m_1 + 1,\ m_1 + 2\ , ...,\ m_1 + m_2\ . \qquad (1.2)$$

のもとで，$f(\boldsymbol{x})$の極小値を与えるベクトル $\boldsymbol{x}^*$とその関数値 $f(\boldsymbol{x}^*)$を求める．

ただし，$\{c_i(\boldsymbol{x})\}$のヤコビアン $\boldsymbol{J}(\boldsymbol{x})$も関数として与えられるものとする．また，$f(\boldsymbol{x})$は 2 階までの連続な偏微係数を持つものと仮定する．

ここで，$x=(x_1, x_2,... x_n)^T$ である．また，m_1，m_2 はそれぞれ等式制約，不等式制約の個数である．以降 $(m = m_1 + m_2)$ とする．

$n\geq 1$，$m_1\geq 0$，$m_2\geq 0$ かつ $m\geq 1$ であること．

(2) パラメタ

X ····· **入力**．初期ベクトル $\boldsymbol{x}_0$.
出力．ベクトル $\boldsymbol{x}^*$.
大きさ n の 1 次元配列.

N ····· **入力**．変数の個数 n.

FUN ····· **入力**．$f(\boldsymbol{x})$を計算する関数副プログラム名.
[副プログラムの用意の仕方]
FUNCTION FUN (X)
パラメタ
X ... **入力**．変数ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.
関数 FUN には，$f(\boldsymbol{x})$の値を代入すること．(使用例参照)

GRAD ···· **入力**．$\boldsymbol{g}(\boldsymbol{x})$ を計算するサブルーチン副プログラム名.
[副プログラムの用意の仕方]
SUBROUTINE GRAD (X, G)
パラメタ
X ····**入力**．変数ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.
G·····**出力**．G(1)= $\partial f/\partial x_1$,...,G(N)= $\partial f/\partial x_n$ なる対応をもつ大きさ n の 1 次元配列.
(使用例参照)

FUNC ····· **入力**．関数 $c_i(\boldsymbol{x})$を計算するサブルーチン副プログラム名.
[副プログラムの用意の仕方]
SUBROUTINE FUNC (X, C)
パラメタ
X ···· **入力**．変数ベクトル x.
大きさ n の 1 次元配列.
C····· **出力**．C(1) = $c_1(\boldsymbol{x})$, ..., C(M(1)) = $c_{m_1}(\boldsymbol{x})$, ..., C(M(1) + M(2)) = $c_m(\boldsymbol{x})$.なる対応をもつ大きさ m の 1 次元配列.
(使用例参照)

JAC····· **入力**．関数 $\boldsymbol{J}(\boldsymbol{x})$ を計算するサブルーチン副プログラム名.
[副プログラムの用意の仕方]
SUBROUTINE JAC (X, CJ, K)
パラメタ
X ···· **入力**．変数ベクトル x.
大きさ n の 1 次元配列.
CJ ··· **出力**．ヤコビアン行列.
$CJ(I,J)=\partial c_i/\partial x_j$ なる対応をもつ
CJ (K, N)なる 2 次元配列.
K ···· **入力**．配列 CJ の整合寸法.
(使用例参照)

M ····· **入力**．制約式の個数.
$M(1)=m_1$, $M(2)=m_2$ なる対応をもつ大きさ 2 の 1 次元配列.

EPSR ····· **入力**．収束判定値(≥ 0.0).
0.0 のときは標準値が採用される.
(使用上の注意②参照)

MAX····· **入力**．関数 $f(\boldsymbol{x})$，$\boldsymbol{g}(\boldsymbol{x})$，$\boldsymbol{c}(\boldsymbol{x})$，及び $\boldsymbol{J}(\boldsymbol{x})$の評価回数の上限(≠0).
(使用上の注意③参照)
出力．実際に評価した回数(>0).

F ····· **出力**．関数値 $f(x^*)$.

VW····· **作業領域**．$VW(K,\ M(1)+M(2)+2*N+12)$ なる 2 次元配列.

K····· **入力**．配列 VW の整合寸法 $(\geq M(1)+M(2)+N+4)$.

IVW····· **作業領域**．大きさ $2*(M(1)+M(2)+N+4)$ の 1 次元配列.

ICON····· **出力**．コンディションコード.
表 NLPG1-1 参照.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, MGSSL, UQP, UNLPG
② FORTRAN 基本関数 ... ABS, SQRT, AMAX1

表NLPG1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	指定された関数の評価回数内で，収束条件が満たされなかった.	パラメタX，Fには，最後の値を格納する.
20000	計算の過程で，関数の局所的減少が満たされなかった. (手法概要③参照) EPSRが小さすぎる.	処理を打ち切る. (パラメタX，Fには，最後の値を格納する．)
21000	制約式を満す解が存在しないか，又は初期値x_0の選び方が適切でない可能性がある．初期値x_0を変えて再実行するとよい.	処理を打ち切る.
30000	N < 1, EPSR < 0.0, M(1) < 0, M(2) <0, M (1) + M(2)<1, K < M (1) + M(2) + N + 4，又はMAX = 0であった.	処理を打ち切る.

b. 注意

① 本サブルーチンを呼び出す側のプログラム内で，引数 FUN，GRAD，FUNC 及び JAC に相当する副プログラム名を EXTERNAL 文で宣言すること.

② EPSR の与え方
本サブルーチンの収束判定は，反復ベクトル $\boldsymbol{x}_k$ において，

$$\|x_{k+1}-x_k\|_\infty \le \max(1.0,\|x_k\|_\infty)\cdot \mathrm{EPSR}$$

となった場合に，$\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$とみなし反復を終了する.
本サブルーチンでは，関数 $f(\boldsymbol{x})$ が極小点 $\boldsymbol{x}^*$の近傍で近似的に 2 次関数であることを前提としており，関数値 $f(\boldsymbol{x}^*)$ を丸め誤差程度まで正しく求めるためには，
EPSR $\approx \sqrt{u}$，u は丸め誤差の単位
程度に与えるのが適当である.
EPSR の標準値は $2\sqrt{u}$ である.

③ MAX の与え方
関数の評価回数は，変数ベクトル $\boldsymbol{x}$ に対して，$f(\boldsymbol{x})$を計算することを 1 回，$\boldsymbol{g}(\boldsymbol{x})$ を計算することを n 回，$\boldsymbol{c}(\boldsymbol{x})$ を計算することを m 回，$\boldsymbol{J}(\boldsymbol{x})$ を計算することを mn 回と数える.
関数の評価回数は，関数の性質，初期ベクトル，収束判定値などに依存する.
一般には，初期ベクトルが良く，収束判定値として標準値を採用した場合は，MAX = 800*mn 程度が妥当である.
指定評価回数内で収束条件が満たされず，ICON = 10000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ MAX には，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと.

c. 使用例
2 変数の実関数

$$f(x_1,x_2)=x_1^2-2x_1x_2+2x_2^2-10x_1+x_2$$

を，制約条件

$$c_1(x_1,x_2)=0.5x_1^2+1.5x_2^2-2=0$$
$$c_2(x_1,x_2)=-x_1+x_2\ge 0$$

のもとで極小化する．初期ベクトルは $x_0=(-2,2)^{\mathrm{T}}$ とする.

```
C     **EXAMPLE**
      DIMENSION X(2),M(2),VW(8,18),
     *          IVW(16)
      EXTERNAL TEST,GRAD,TESTC,JAC
      X(1)=-2.0
      X(2)=2.0
      N=2
      M(1)=1
      M(2)=1
      EPSR=1.0E-3
      MAX=800*2*2
      K=8
      CALL NLPG1(X,N,TEST,GRAD,TESTC,
     *     JAC,M,EPSR,MAX,F,VW,K,IVW,ICON)
      WRITE(6,600) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,610) F,MAX
      WRITE(6,620) (I,X(I),I=1,N)
      STOP
  600 FORMAT('1','*ICON=',I5)
  610 FORMAT(' ','*F=',E15.7,1X,'MAX=',I5)
  620 FORMAT('0',(/2X,'X(',I2,')=',E15.7))
      END
C     OBJECTIVE FUNCTION
      FUNCTION TEST(X)
      DIMENSION X(2)
      TEST=(X(1)-2.0*X(2)-10.0)*X(1)+
     *     (2.0*X(2)+1.0)*X(2)
      RETURN
      END
C     DERIVATIVE
      SUBROUTINE GRAD(X,G)
      DIMENSION X(2),G(2)
      G(1)=2.0*X(1)-2.0*X(2)-10.0
      G(2)=-2.0*X(1)+4.0*X(2)+1.0
      RETURN
      END
C     CONSTRANTS
      SUBROUTINE TESTC(X,C)
      DIMENSION X(2),C(2)
      C(1)=0.5*X(1)*X(1)+1.5*X(2)*X(2)-2.0
      C(2)=-X(1)+X(2)
      RETURN
      END
C     JACOBIAN
      SUBROUTINE JAC(X,CJ,K)
      DIMENSION X(2),CJ(K,2)
      CJ(1,1)=X(1)
      CJ(2,1)=-1.0
      CJ(1,2)=3.0*X(2)
      CJ(2,2)=1.0
      RETURN
```

```
END
```

(4) 手法概要

非線形計画問題

$$f(\boldsymbol{x}) \longrightarrow \text{最小化} \tag{4.1}$$

制約条件

$$c_i(\boldsymbol{x}) = 0, i = 1,2,...,m_1 \quad \text{等式制約} \tag{4.2}$$
$$c_i(\boldsymbol{x}) \geq 0, i = m_1+1,...,m_1+m_2 \quad \text{不等式制約} \tag{4.3}$$

を Powell の可変計量法により解く.

以下の説明の都合上，次のように記号を定める.

M_1 .... 等式制約式の添字の集合 $(1,2,...,m_1)$ (空集合でもよい.)

M_2 .... 不等式制約式の添字の集合 $(m_1+1,...,m_1+m_2)$ (空集合でもよい.)

M .... 制約式の添字の集合 $(1,2,...,m_1+m_2)$(空集合ではいけない.)

$\boldsymbol{g}$..... fの傾斜ベクトル

∇c_i .. c_iの傾斜ベクトル

まず，制約条件のない場合の問題(unconstrained minimization)の解法との関連に基づき，この非線形計画問題の解法を解説する.

制約式(4.2)，(4.3)がない場合，目的関数 $f(\boldsymbol{x})$を極小化するための可変計量法―改訂準ニュートン法(revised quasi-Newton method)―は次のようになる. 極小点の近似点 $\boldsymbol{x_k}$において $f(\boldsymbol{x})$を(4.4)のように 2 次の項まで展開する.

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_k) + \boldsymbol{y}^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{x}_k)\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{y} \tag{4.4}$$

ここで

$$\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{x}_k \tag{4.5}$$

であり，$\boldsymbol{B}$ は $\boldsymbol{x}_k$ における $f(\boldsymbol{x})$ のヘシアン行列である. そこで，(4.4)を極小化する $\boldsymbol{y}$ を求め(それを $\boldsymbol{y}_k$ とする)，更に 1 次元探索により

$$\min_{\alpha} f(\boldsymbol{x}_k + \alpha \boldsymbol{y}_k) \tag{4.6}$$

を満すαを求める(それを α_k とする). この α_k に基づき

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{y}_k \tag{4.7}$$

によって，次の近似値 $\boldsymbol{x}_{k+1}$ を得る. ただし，ヘシアン行列 $\boldsymbol{B}$ を直接計算することはせず，反復の過程でベクトル $\boldsymbol{g}(\boldsymbol{x}_{k+1}) - \boldsymbol{g}(\boldsymbol{x}_k)$及び $\alpha_k \boldsymbol{y}_k$ をもとに近似する.

さて，制約(4.2)，(4.3)がある場合は，$\boldsymbol{y}$ を決定する際に制約が付くことになる. 即ち，(4.2)に対しては，その 1 次の項までの展開式により

$$c_i(\boldsymbol{x}) = c_i(\boldsymbol{x}_k) + \boldsymbol{y}^{\mathrm{T}}\nabla c_i(\boldsymbol{x}_k) = 0, \quad i \in M_1 \tag{4.8}$$

が要求され，同様に(4.3)に対しては

$$c_i(\boldsymbol{x}) = c_i(\boldsymbol{x}_k) + \boldsymbol{y}^{\mathrm{T}}\nabla c_i(\boldsymbol{x}_k) \geq 0, \quad i \in M_2 \tag{4.9}$$

が要求される.

したがって，非線形計画問題(4.1)，(4.2)，(4.3)を解くためには，(4.8)，(4.9)を満す $\boldsymbol{y}$ のうち(4.4)を極小化するものを求める必要がある. これは，$\boldsymbol{y}$ に関する 2 次計画問題である.

一方，1 次元探索についても(4.6)式をそのまま用いるのではなく，ペナルティ関数

$$\boldsymbol{W}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{P}[\boldsymbol{c}(\boldsymbol{x})] \tag{4.10}$$

を導入し，$W(\boldsymbol{x})$ を最小化するようにαを決定する. ここで $P[\boldsymbol{c}(\boldsymbol{x})]$は，制約がすべて満されていれば零，それ以外の場合は正の値をとるものであるが，詳細については後述される.

更に，ヘシアン行列の近似行列 $\boldsymbol{B}_k$ の改訂についても $f(\boldsymbol{x})$ のみならず $\boldsymbol{c}(\boldsymbol{x})$ に関する情報も考慮して行う. 仮に，制約が等式条件だけの場合を考えると，極小点では

$$\boldsymbol{g}(\boldsymbol{x}) - \sum_{i \in M_1} \lambda_i \nabla c_i(\boldsymbol{x}) = 0 \tag{4.11}$$

が要求される. ここで λ_i はラグランジュ乗数である. (4.11)は $n+m_1$ 次の連立非線型方程式であり，これをニュートン法で解く場合,

$$\phi(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) - \sum_{i \in M_1} \lambda_i c_i(\boldsymbol{x}) \tag{4.12}$$

に関する 2 階の偏微分情報が必要であり，$f(\boldsymbol{x})$ のそれのみでは不十分であることがわかる.

そこで，$\boldsymbol{B}_k$ の改訂は，ベクトル$\alpha_k \boldsymbol{y}_k$ と

$$\gamma_k = \nabla_x \phi(\boldsymbol{x}_{k+1}, \lambda) - \nabla_x \phi(\boldsymbol{x}_k, \lambda) \tag{4.13}$$

ここで $\nabla_x \phi$ は，ϕ の $\boldsymbol{x}$ に関する傾斜ベクトルから得られるランク 2 の行列に基づいて行う.

ところで，非線形等式制約がある場合，$W(\boldsymbol{x})$に関する 1 次元探索を厳密に行うと，$\boldsymbol{x}_k$ の進み方(歩幅)が著しく小さくなり，真の極小値へ収束しない場合がある. そこで，解 $\boldsymbol{x}_k$ の動きを監視する番犬(watchdog)を立てることにより，上記のような不都合を回避する.

以下に本サブルーチンの計算手順の概要を述べるが，そこで用いる代表的な関数を次のように定義する．

- ペナルティ関数

$$W(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i\in M_1} \mu_i |c_i(\boldsymbol{x})| + \sum_{i\in M_2} \mu_i |\min(0, c_i(\boldsymbol{x}))| \tag{4.14}$$

点 $\boldsymbol{x}$ が制約条件から外れるにつれてペナルティがつくようになっている．係数 μ_i は，後述の方法により決定する．

- ペナルティ関数の 1 次展開
近似点 $\boldsymbol{x}_k$ において $W(\boldsymbol{x})$を，(4.15)のように 1 次の項まで展開した関数．

$$\begin{aligned} W_k(\boldsymbol{x}) &= f(\boldsymbol{x}_k) + (\boldsymbol{x}-\boldsymbol{x}_k)^{\mathrm{T}} \boldsymbol{g}(\boldsymbol{x}_k) \\ &+ \sum_{i\in M_1} \mu_i \left| c_i(\boldsymbol{x}_k) + (\boldsymbol{x}-\boldsymbol{x}_k)^{\mathrm{T}} \nabla c_i(\boldsymbol{x}_k) \right| \\ &+ \sum_{i\in M_2} \mu_i \left| \min\left(0, c_i(\boldsymbol{x}_k) + (\boldsymbol{x}-\boldsymbol{x}_k)^{\mathrm{T}} \nabla c_i(\boldsymbol{x}_k)\right) \right| \end{aligned} \tag{4.15}$$

- ラグランジュ関数

$$L_k(\boldsymbol{x}) = f(\boldsymbol{x}) - \sum_{i\in M} \lambda_i c_i(\boldsymbol{x}) \tag{4.16}$$

λ_i は $\boldsymbol{x}_k$ により決まるラグランジュ乗数で，後述の方法により決定する．

本サブルーチンの計算手順

ステップ 1(初期値設定)

① 以下の諸量を設定する．
k = 0 (反復回数のカウンタ)
$\boldsymbol{H}_0 = \boldsymbol{I}_n$
l = 0 (番犬の位置を示す番号)
lt = 5 (番犬の位置を見直す間隔)
θ =0.25

$$\varepsilon = \begin{cases} \mathrm{EPSR}, & \mathrm{EPSR} \neq 0 \\ 2\sqrt{u}, & \mathrm{EPSR} = 0 \end{cases}$$

(収束判定定数)

ステップ 2(2 次計画問題)

② 次のような $\boldsymbol{y}$ に関する 2 次計画問題を解く．

QP_k:

$$\left.\begin{aligned} &\frac{1}{2}\boldsymbol{y}^{\mathrm{T}} B_k \boldsymbol{y} + \boldsymbol{y}^{\mathrm{T}} g(\boldsymbol{x}_k) \longrightarrow \text{最小化} \\ &\text{制約式} \\ &\boldsymbol{y}^{\mathrm{T}} \nabla c_i(\boldsymbol{x}_k) + c_i(\boldsymbol{x}_k) = 0,\ i \in M_1 \\ &\boldsymbol{y}^{\mathrm{T}} \nabla c_i(\boldsymbol{x}_k) + c_i(\boldsymbol{x}_k) \geq 0,\ i \in M_2 \end{aligned}\right\} \tag{4.17}$$

③ QP_kに $\boldsymbol{y}_k$が存在する場合

- $\|\boldsymbol{y}_k\|_\infty < \max\left(1.0, \|\boldsymbol{x}_k\|_\infty\right) \cdot \varepsilon$ (4.18)
を満し，かつ $\boldsymbol{x}_k$ が可能点ならば，$\boldsymbol{x}_k$ を $\boldsymbol{x}^*$，$f(\boldsymbol{x}_k)$ を $f(\boldsymbol{x}^*)$とみなし ICON = 0 として処理を終了する．
- (4.18)を満さない場合は，ペナルティ関数 $W(\boldsymbol{x})$に関する 1 次元探索を行ない，$\boldsymbol{y}_k$方向上に
$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha^* \boldsymbol{y}_k \tag{4.19}$$
を決定する．
ここで，もし，$\|\alpha^* \boldsymbol{y}_k\|$ が収束とみなされる程小さく，かつ $\boldsymbol{x}_k$ が可能点ならば ICON = 20000 として処理を終了する．これは収束判定定数 ε が小さすぎるためであるが，多くの場合極小点とみなすことができる．一方，$\boldsymbol{x}_k$が不能点ならば ICON = 21000 として処理を終了する．
$\|\alpha^* \boldsymbol{y}_k\|$ が大きい場合にはステップ 3 に進む．

④ QP_kに可能解が存在しない場合
次のような $\boldsymbol{y}$ 及び z に関する新たな 2 次計画問題を解く．
QP_k:

$$\left.\begin{aligned} &\frac{1}{2}\boldsymbol{y}^{\mathrm{T}} B_k \boldsymbol{y} + \boldsymbol{y}^{\mathrm{T}} g(\boldsymbol{x}_k) - \beta z \rightarrow \text{最小化} \\ &\text{制約式} \\ &\boldsymbol{y}^{\mathrm{T}} \nabla c_i(\boldsymbol{x}_k) + c_i(\boldsymbol{x}_k) z = 0,\ i \in M_1 \\ &\boldsymbol{y}^{\mathrm{T}} \nabla c_i(\boldsymbol{x}_k) + c_i(\boldsymbol{x}_k) z_i \geq 0,\ i \in M_2 \\ &\text{ここで} \\ &0 \leq z \leq 1, \\ &z_i = \begin{cases} 1, & c_i(\boldsymbol{x}_k) > 0, \\ z, & c_i(\boldsymbol{x}_k) < 0, \end{cases} \end{aligned}\right\} \tag{4.20}$$

βは十分大なる正数
この 2 次計画問題の最適解を $\boldsymbol{y}_k$，$\bar{z}$ とする．

- もし $\bar{z} < \varepsilon$ で，かつ $\boldsymbol{x}_k$が可能点ならば $\boldsymbol{x}_k$ を $\boldsymbol{x}^*$，$f(\boldsymbol{x}_k)$ を $f(\boldsymbol{x}^*)$とみなし ICON =0 として処理を終了する．
- もし $\bar{z} < \varepsilon$ で，かつ $\boldsymbol{x}_k$が不能点ならば，非線形計画問題(4.1)，(4.2)，(4.3)には可能解は存在しない(制約条件が互いに矛盾している)か，又は初期値 $\boldsymbol{x}_0$の選び方が適切でない可能性があるため，ICON = 21000 として処理を終了する．
- もし $\bar{z} \geq \varepsilon$ で，
$\|\boldsymbol{y}_k\|_\infty < \max\left(1.0, \|\boldsymbol{x}_k\|_\infty\right) \cdot \varepsilon$ (4.21)
を満し，かつ $\boldsymbol{x}_k$ が可能点ならば，$\boldsymbol{x}_k$ を $\boldsymbol{x}^*$，$f(\boldsymbol{x}_k)$ を $f(\boldsymbol{x}^*)$とみなし ICON = 0 として処理を終了する．

(4.21)を満さない場合はステップ 3 に進む.

ステップ 3(番犬プロセス)

① $k = k + 1$ とする.
解 $\boldsymbol{x}_k$ の動きを監視し，$\boldsymbol{x}_k$ の進み幅(歩幅)が適切であるか否かを判定する．すなわち，もし

$$W(x_k) \le W(x_l) - \theta\left(W(x_l) - W_l(x_{l+1})\right) \tag{4.22}$$

ならばステップ 5 に進む.

ステップ 4(番犬と $\boldsymbol{B}_k$ の改訂)

⑥ もし，

$$W(x_k) \le W(x_l)$$

ならば，$l = k$，$\boldsymbol{x}_l = \boldsymbol{x}_k$ とする.

⑦ もし，k が lt の倍数ならば，$\boldsymbol{x}_k = \boldsymbol{x}_l$，$l = k$ とする.

⑧ 次のように $\boldsymbol{B}_k$ を改訂する.

$$\left.\begin{aligned} &\boldsymbol{\delta} = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}\left(= \alpha^* \boldsymbol{y}_{k-1}\right) \\ &\boldsymbol{\gamma} = \boldsymbol{g}_k - \boldsymbol{g}_{k-1} \\ &- \sum_{i \in M} \lambda_i \left(\nabla \mathrm{c}_i(\boldsymbol{x}_k) - \nabla \mathrm{c}_i(\boldsymbol{x}_{k-1})\right) \end{aligned}\right\} \tag{4.23}$$

として,

$$\xi = \begin{cases} 1 & , \ \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\gamma} \ge 0.2\boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{B}_{k-1}\boldsymbol{\delta} \\ 0.8\dfrac{\boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{B}_{k-1}\boldsymbol{\delta}}{\boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{B}_{k-1}\boldsymbol{\delta} - \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\gamma}}, & \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\gamma} < 0.2\boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{B}_{k-1}\boldsymbol{\delta} \end{cases} \tag{4.24}$$

を求め，このξを用いて

$$\boldsymbol{\eta} = \xi\boldsymbol{\gamma} + (1-\xi)\boldsymbol{B}_{k-1}\boldsymbol{\delta} \tag{4.25}$$

を計算する.
これらの諸量をもとに $\boldsymbol{B}_k$ を(4.25)により求める.

$$B_k = B_{k-1} \frac{B_{k-1}\delta\delta^{\mathrm{T}}B_{k-1}}{\delta^{\mathrm{T}}B_{k-1}\delta} + \frac{\eta\eta^{\mathrm{T}}}{\delta^{\mathrm{T}}\eta} \tag{4.26}$$

ステップ 2 に戻る.

ステップ 5(歩幅の改訂)

⑨ ③で得られた α^* を，次のような方法により拡大できるか否かを調べる.

- $W(\boldsymbol{x}) \le W(\boldsymbol{x}_{k-1})$
  又は
  $\mathrm{L}(\boldsymbol{x}) \ge \mathrm{L}(\boldsymbol{x}_{k-1})$
  を満す $\boldsymbol{x}$ を $\boldsymbol{y}_k$ 方向上に見つける. (4.27)
- もし(4.27)を満す $\boldsymbol{x}$ がない場合はステップ 4 に戻る
- もし(4.27)を満す $\boldsymbol{x}$ があれば
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \tilde{\alpha}\boldsymbol{y}_k$
  とする.
- もし，$\tilde{\alpha} = \alpha^*$ ならばステップ 4 に戻る．$\tilde{\alpha} \ne \alpha^*$ ならばステップ 3 に戻る.

アルゴリズム上の留意点

① λ_i の決定法
QP_k の解のうち，制約式に対するラグランジュ乗数を用いて λ_i を決定する.

② μ_i の決定法
初期値は，$\mu_i = 2|\lambda_i|$,とし，その後
$\mu_i < 1.5|\lambda_i|$ (λ_i はその時点での値)となった μ_i に関して $\mu_i = 2|\lambda_i|$ とする.

なお，詳細については，参考文献[94]，[95]を参照すること.

C24-11-0101 NOLBR, DNOLBR

連立非線型方程式 (ブレント法)
CALL NOLBR (X, N, FUN, EPSZ, EPST, FC, M, FNOR, VW, ICON)

(1) 機能

連立非線型方程式(1.1)をブレント(Brent)法により解く.

$$\left.\begin{aligned} &f_1(x_1, x_2, ..., x_n) = 0 \\ &f_2(x_1, x_2, ..., x_n) = 0 \\ &... \\ &f_n(x_1, x_2, ..., x_n) = 0 \end{aligned}\right\} \qquad (1.1)$$

すなわち，$\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_n(\boldsymbol{x}))^T$，$\boldsymbol{x} = (x_1, x_2, ..., x_n)^T$ として，与えられた初期ベクトル $\boldsymbol{x_0}$ より，(1.2)の解を求める.

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \qquad (1.2)$$

ここで $\boldsymbol{0}$ は $\boldsymbol{n}$ 次元の 0 ベクトルである．$n \geq 1$ であること.

(2) パラメタ

X ……**入力**．求めようとする解の初期ベクトル $\boldsymbol{x}_0$.
出力．解ベクトル.
大きさ n の 1 次元配列.

N ……**入力**．方程式の元数 n.

FUN ……**入力**． $f_k(\boldsymbol{x})$ を計算する関数副プログラム名．利用者は呼出し側のプログラムの中で EXTERNAL 宣言すると共に関数副プログラムを用意すること.
[関数副プログラムの用意の仕方]
FUNCTION FUN (X, K)
パラメタ
X……**入力**．変数ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.
K……**入力**．方程式における式番号を与える($1 \leq K \leq n$)．例えば K=k と入力されたとき関数 FUN には $f_k(\boldsymbol{x})$ の値を代入すること.
(使用例参照)

EPSZ ……**入力**．収束判定値(≥ 0.0).
$\|\boldsymbol{f}(\boldsymbol{x}_i)\|_\infty \leq$ EPSZ となったとき，収束したと見なす．$\boldsymbol{x}_i$ とは第 i 段階目の解の近似ベクトルである(使用上の注意参照).

EPST ……**入力**．収束判定値(≥ 0.0).
$\|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\|_\infty \leq \text{EPST} \cdot \|\boldsymbol{x}_i\|_\infty$ となったとき収束したと見なす(使用上の注意参照).

FC ……**入力**．解ベクトルの探索範囲を示す数値(≥ 0.0)．反復の過程で
$\|\boldsymbol{x}_i\|_\infty > \text{FC} \cdot \max(\|\boldsymbol{x}_0\|_\infty, 1.0)$ となったとき解ベクトルの探索を止める(使用上の注意参照).

M ……**入力**．反復回数の上限(> 0)(使用上の注意参照)
出力．実際に反復した回数(> 0)

FNOR ……**出力**．求まった解ベクトルにおける $\|\boldsymbol{f}(\boldsymbol{x})\|_\infty$ の値.

VW ……**作業領域**.
大きさ n (n + 3)の 1 次元配列.

ICON ……**出力**．コンディションコード.
表 NOLBR-1 参照.

表NOLBR-1 コンディションコード

コード	意味	処理内容
1	収束判定 $\|\boldsymbol{f}(\boldsymbol{x}_i)\| \leq$ EPSZ が満たされた.	正常終了.
2	収束判定 $\|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\|_\infty \leq \text{EPST} \cdot \|\boldsymbol{x}_i\|_\infty$ が満たされた.	正常終了.
10000	与えられた反復回数内で指定された収束条件を満たさなかった.	パラメタXには最後の反復ベクトルが格納される.
20000	解ベクトルの探索範囲内で解ベクトルを見つけられなかった．(パラメタFCの項参照)	処理を打ち切る.
25000	計算の過程で関数$\boldsymbol{f}(\boldsymbol{x})$のヤコビアンが0になった．(手法概要参照)	処理を打ち切る．(パラメタXには最後の反復ベクトルが格納される)
30000	N≤0, EPSZ<0, EPST<0, FC≤0, 又はM≤0であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH，MGSSSL
② FORTRAN 基本関数... ABS, AMAX1, SIGN, SQRT

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で引数 FUN に相当する関数副プログラム名に対して，EXTERNAL 宣言をすること.
② EPSZ と EPST の与え方
本サブルーチンは，収束判定法として二とおり用意しており，少なくともどちらか一方が満たされたとき反復を止める．もし利用者がどちらか一方の判定法だけを適用したい場合は，他方の判定値を 0.0 と与えればよい．具体的には以下のような与え方ができる.
a. EPSZ = $\varepsilon_A(> 0)$, EPST = 0 のとき，
$\|\mathbf{x}_i - \mathbf{x}_{i-1}\|_\infty = 0$ とならない限り，

$\|f(x_i)\|_\infty \le \varepsilon_A$ が満たされるか反復回数が M になるまで反復する.

b. EPSZ = 0 , EPST = $\varepsilon_B (>0)$ のとき,
$\|f(x_i)\|_\infty = 0$ とならない限り,
$\|x_i - x_{i-1}\|_\infty \le \varepsilon_B \cdot \|x_i\|_\infty$ が満たされるか反復回数が M になるまで反復する.

c. EPSZ = 0 , EPST = 0 のとき,
$\|f(x_i)\|_\infty = 0$ 又は $\|x_i - x_{i-1}\|_\infty = 0$ とならない限り M 回反復する.

無条件に M 回反復したいときは，上記の c.の与え方をすればよい.

③ FC の意味
方程式の性質によっては，初期ベクトル x_0 の近傍に解ベクトルが無いことがある．このようなとき，反復ベクトル x_i は x_0 から大きく離れ，その結果 $f(x)$ の計算時にオーバフローなどの不都合を起こす場合がある.
パラメタ FC はこれを防ぐために設けられ，解ベクトルの探索範囲を指定するものである.
標準値としては，100.0 程度でよい.

④ M の与え方
解ベクトルへ収束するまでの反復回数は，方程式の性質及び収束判定値に依存する．一般に初期ベクトルが悪いとき，あるいは収束判定値が小さいときは，パラメタ M は大きめに与えた方がよい.
元数 n が 10 程度なら M は 50 程度でよい.

⑤ 単精度／倍精度の使分け
一般に連立非線型方程式を反復的に解く場合，単精度サブルーチンでは解けないが，倍精度サブルーチンでは解けることがある.

c. 使用例
2 元の連立非線型方程式

$$\left.\begin{aligned} x_1 \cdot (1 - x_2^2) &= 2.25 \\ x_1 \cdot (1 - x_2^3) &= 2.625 \end{aligned}\right\}$$

の解を初期ベクトル $x_0 = (5.0,\ 0.8)^T$ より求める．真の解は $x = (3.0,\ 0.5)^T$ 及び $x = (81/32,\ -1/3)^T$ である.

```
C     **EXAMPLE**
      DIMENSION X(2),VW(10)
      EXTERNAL FUN
      X(1)=5.0
      X(2)=0.8
      N=2
      EPSZ=1.0E-5
      EPST=0.0
      FC=100.0
      M=20
      CALL NOLBR(X,N,FUN,EPSZ,EPST,FC,
     *           M,FNOR,VW,ICON)
      WRITE(6,600) ICON,M,FNOR,(I,X(I),
     *             I=1,N)
      STOP
  600 FORMAT(' ','ICON=',I5/' ','M=',I5/
     *       ' ','FNOR=',E15.7/
     *      (' ','X(',I2,')=',E15.7))
      END

      FUNCTION FUN(X,K)
      DIMENSION X(2)
      GO TO(10,20),K
   10 FUN=X(1)*(1.0-X(2)**2)-2.25
      RETURN
   20 FUN=X(1)*(1.0-X(2)**3)-2.625
      RETURN
      END
```

(4) 手法概要

連立非線型方程式,

$$f(x) = O \tag{4.1}$$

の解ベクトルをブレンド(Brent)法を用いて求める.

この方法の 1 回めの反復過程では，$y_1 = x_{i-1}$ とし n 段階の操作から，中間的な反復ベクトル $y_2, y_3, ...y_n, y_{n+1}$ を作る．そして y_{n+1} を x_i とする．各々の y_{k+1} $(1 \le k \le n)$は $f_j(y)$ の y_j におけるテイラー展開の 1 次項までの近似式,

$$f_j(y) \approx f_j(y_j) + g_j^T (y - y_j),\quad j = 1,2,..,k$$

の右辺を0にするように選ばれる．ここで,

$$g_j^T = \left(\frac{\partial f_j}{\partial x_1}, ..., \frac{\partial f_j}{\partial x_n} \right)_{x = y_j} \tag{4.2}$$

とする.

ブレント法の手順

ある反復ベクトル x_{i-1} から x_i を求めるときの手順を述べる.

まず，適当な直交行列 Q_1 が与えられているとする．(x_0 から x_1 を求めるときは，$Q_1 = I$ としている).

① 第 1 段階
$y_1 = x_{i-1}$
として， $f_1(y)$ を y_1 におけるテイラー展開の 1 次の項までで近似すると,

$$f_1(y) \approx f_1(y_1) + g_1^T (y - y_1) \tag{4.3}$$

となる．第 1 段階では(4.3)の右辺を 0 とするひとつの y を以下の要領で求め，それを y_2 とする.
まず， $g_1^T Q_1 = w_1^T$ とし,

$$w_1^T P_1 = \alpha_1 e_1^T,\quad \alpha_1 = \pm \|w_1\|_2,\quad e_1^T = (1,0,...,0)$$

とする直交行列 P_1 をハウスホルダー法で求める.

ここで複号は $\boldsymbol{w}^T$ の第 1 成分の符号と同じ方を採用するものとする．このとき $\boldsymbol{y}_2$ は，

$$\boldsymbol{y}_2 = \boldsymbol{y}_1 - \frac{f_1(y_1)}{\alpha_1} Q_2 \boldsymbol{e}_1, \quad Q_2 = Q_1 P_1 \tag{4.4}$$

と計算する．

② 第 2 段階

．
．
．

③ 第 $\boldsymbol{k}$ 段階

$f_k(\boldsymbol{y})$ を $\boldsymbol{y}_k$ におけるテイラー展開の 1 次の項まで近似すると，

$$f_k(\boldsymbol{y}) \approx f_k(\boldsymbol{y}_k) + \mathbf{g}_k^T(\boldsymbol{y} - \boldsymbol{y}_k) \tag{4.5}$$

となる．

まず，横ベクトル $\boldsymbol{g}_k^T \boldsymbol{Q}_k$ の最初から k-1 個の成分を 0 で置き換えたものを $\boldsymbol{w}_k^T$ とする．

$$w_k^T = (\overbrace{0,...,0}^{k-1},*,*,...,*)$$

次に，

$$\boldsymbol{w}_k^T P_k = \alpha_k \boldsymbol{e}_k^T, \alpha_k = \pm \left\| \boldsymbol{w}_k \right\|_2,$$

$$\boldsymbol{e}_k^T = (\overbrace{0,...,0}^{k-1},1,0,...,0)$$

とする直交行列 $\boldsymbol{P}_k$ をハウスホルダー法で求める．

ここで複号は $\boldsymbol{w}_k^T$ の第 k 番目の成分の符号と同じ方を採用するものとする．

また，$\boldsymbol{P}_k$ の形状は以下のとおりである．

$$P_k = \begin{matrix} \\ \\ \\ \\ k-1 \end{matrix} \left\{ \begin{bmatrix} 1 & 0 & & \\ & \ddots & & 0 \\ 0 & 1 & & \\ \hline & 0 & & \hat{P}_k \\ & & & \end{bmatrix} \right.$$

このとき，$\boldsymbol{y}_{k+1}$ は，

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k - \frac{f_k(\boldsymbol{y}_k)}{\alpha_k} Q_{k+1} \boldsymbol{e}_k \ , Q_{k+1} = Q_k P_k \tag{4.6}$$

と計算する．

(4.6)より求めた $\boldsymbol{y}_{k+1}$ は(4.7)を満たすことが証明される．

$$\left.\begin{aligned} f_1(y_1) + g_1^T(y_{k+1} - y_1) = 0 \\ f_2(y_2) + g_2^T(y_{k+1} - y_2) = 0 \\ \cdots \\ f_k(y_k) + g_k^T(y_{k+1} - y_k) = 0 \end{aligned}\right\} \tag{4.7}$$

このようにして第 n 段階目($k = n$)に得られる $\boldsymbol{y}_{n+1}$ を $\boldsymbol{x}_i$ と置き，これが収束判定を満たすときは反復を止めて，その $\boldsymbol{x}_i$ を解ベクトルとする．さもなければ，$\boldsymbol{x}_i$ を中間的な反復ベクトルの出発値 $\boldsymbol{y}_1$，$\boldsymbol{Q}_{n+1}$ を $\boldsymbol{Q}_1$ として，①から繰返す．

アルゴリズム上の考慮

a. 偏微係数の近似法

先に述べた $\boldsymbol{w}_k^T$ の計算法について述べる．

$\boldsymbol{w}_k^T$ の j 番目($k \le j \le n$)の成分は，

$$\boldsymbol{g}_k^T \boldsymbol{Q}_k \boldsymbol{e}_j$$

であり，これは $\boldsymbol{f}_k$ の $\boldsymbol{y}_k$ における $\boldsymbol{Q}_k \boldsymbol{e}_j$($\boldsymbol{Q}_k$ の第 j 列)が示す方向の微係数である．

本サブルーチンでは，これを差分により近似する．

すなわち，

$$\mathbf{w}_k \approx \frac{1}{h_i} \begin{matrix} \\ \\ \\ \\ k-1 \end{matrix} \left\{ \begin{bmatrix} 0 \\ \cdots \\ 0 \\ f_k(\boldsymbol{y}_k + h_i \mathbf{Q}_k \boldsymbol{e}_k) - f_k(\boldsymbol{y}_k) \\ \cdots \\ f_k(\boldsymbol{y}_k + h_i \mathbf{Q}_k \boldsymbol{e}_k) - f_k(\boldsymbol{y}_k) \end{bmatrix} \right. \tag{4.8}$$

ここで h_i は，

$$h_i = \left\| \boldsymbol{x}_{i-1} \right\|_\infty \cdot \sqrt{u} \tag{4.9}$$

とする．u は丸め誤差の単位である．

b. α_k が 0 のとき

先に述べた第 k 段階目において

$$\alpha_k = 0 \tag{4.10}$$

となると，$\boldsymbol{y}_{k+1}$ は求まらない．このようなときは，

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k \tag{4.11}$$

としている．これは，$\boldsymbol{y}_k$ の修正を行なわないことを意味する．

もし，

$$\alpha_k = 0, \quad k = 1,2,...,n$$

となったとすると，$\boldsymbol{y}_1$ すなわち $\boldsymbol{x}_{i-1}$ には，何の修正もできないことになる．この事態が起きたときは，$\boldsymbol{x}_{i-1}$ における $\boldsymbol{f}(\boldsymbol{x})$のヤコビアン行列

$$J=\left(\frac{\partial f_i}{\partial x_j}\right) \tag{4.12}$$

が非正則である可能性が強い．このとき ICON = 25000 として処理を打ち切る．
(4.10)の判定は，実際は，

$$\left\| f_k\left(\boldsymbol{y}_k+h_i\boldsymbol{Q}_k\boldsymbol{e}_j\right)-f_k\left(\boldsymbol{y}_k\right)\right\|_\infty \le u\cdot\left\|f_k\left(\boldsymbol{y}_k\right)\right\|_\infty \quad ,j=k,k+1,...,n \tag{4.13}$$

で行っている．

なお，詳細については，参考文献[33]を参照すること．

D15-10-0101 NOLF1, DNOLF1

関数二乗和の極小化 (微係数不要，改訂マルカート法)
CALL NOLF1 (X, N, FUN, M, EPSR, MAX, F, SUMS, VW, K, ICON)

(1) 機能

n 変数の m 個の実関数 $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}),\ldots, f_m(\boldsymbol{x})$ と初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，改訂マルカート法(Levenberg-Marquardt-Morrison 法．以下 LMM 法と略す)により，

$$F(x)=\sum_{i=1}^{m}\{f_i(x)\}^2 \qquad (1.1)$$

の極小値を与えるベクトル $\boldsymbol{x}^*$ とその関数値 $F(\boldsymbol{x}^*)$ を求める．本サブルーチンでは微係数は不要であるが，$f_i(\boldsymbol{x})$，i=1,…, m は 1 階までの連続な偏微分をもつものと仮定する．また，$m \geq n \geq 1$ であること．

(2) パラメタ

X……**入力**．初期ベクトル $\boldsymbol{x}_0$.
出力．ベクトル $\boldsymbol{x}^*$.
大きさ n の1次元配列.

N……**入力**．変数の個数 n.

FUN……**入力**．$f_i(\boldsymbol{x})$ を計算するサブルーチン副プログラム名.
[副プログラムの用意の仕方]
SUBROUTINE FUN (X, Y)
パラメタ
X……**入力**．変数ベクトル $\boldsymbol{x}$.
大きさ n の 1 次元配列.
Y……**出力**．変数ベクトル $\boldsymbol{x}$ に対する関数値 $f_i(\boldsymbol{x})$.
$F(1)=f_1(\boldsymbol{x}),\ldots,F(M)=f_m(\boldsymbol{x})$
なる対応を持つ，大きさ m の 1 次元配列.

M……**入力**．関数の個数 m.

EPSR……**入力**．収束判定値(≥ 0.0).
0.0のときは，標準値が採用される.
(使用上の注意②参照).

MAX……**入力**．関数の評価回数の上限(≠0).
(使用上の注意③参照).
出力．実際に評価した回数(> 0).

F……**出力**．関数値 $f_i(\boldsymbol{x}^*)$.
$F(1)=f_1(\boldsymbol{x}^*),\ldots,F(M)=f_m(\boldsymbol{x}^*)$
なる対応を持つ．大きさ m の1次元配列.

SUMS……**出力**．関数二乗和 $F(x^*)$ の値.

VW……**作業領域**．VW(K, N + 2)なる2次元配列.

K……**入力**．配列VWの整合寸法(≥ $m + n$).

ICON……**出力**．コンディションコード.
表NOLF1-1参照.

表NOLF1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	指定された関数の評価回数内で収束条件が満たされなかった.	パラメタX，F, SUMSには最後の値を格納する.
20000	計算の過程で，Marquardt数 v_k が上界を超えた. (手法概要(4.14)参照). EPSRが小さすぎるか，ヤコビアン行列の差分近似の誤差が計算限界を超えた.	処理を打ち切る. (パラメタX，F, SUMSには最後の値を格納する．)
30000	N < 1, M < N, EPSR < 0.0, MAX = 0又は, K< $m+n$であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, MGSSL
② FORTRAN 基本関数... ABS, SQRT, FLOAT

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で，引数 FUN に相当する副プログラム名を EXTERNAL 文で宣言すること.
② EPSR の与え方
本サブルーチンでは，$F(\boldsymbol{x})$が極小点 $\boldsymbol{x}^*$の近傍で近似的に 2 次関数であることを前提としており，$F(\boldsymbol{x}^*)$を丸め誤差程度まで正しく求めるためには,
EPSR $\approx \sqrt{u}$，u は丸め誤差の単位
程度に与えるのが適当である.
EPSR の標準値は $2\cdot\sqrt{u}$ である.
③ MAX の与え方
関数の評価回数は，変数ベクトル $\boldsymbol{x}$ に対して，$f_i(\boldsymbol{x})$，i = 1, ..., m を計算することを 1 回と数える．すなわち，副プログラム FUN の呼出し回数に一致する.
この関数の評価回数は，関数{ $f_i(\boldsymbol{x})$ }の性質，初期ベクトル，収束判定値などに大きく依存する.
一般には，初期ベクトルが良く，収束判定値として標準値を採用した場合は，MAX = $100\cdot n\cdot m$ 程度が妥当である.
指定評価回数内で収束条件が満たされず ICON = 10000 として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタ MAX には，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと.

c. 使用例

$$F(x_1,x_2) = f_1^2(x_1,x_2) + f_2^2(x_1,x_2)$$

ここで,

$$f_1(x_1,x_2) = 1 - x_1$$
$$f_2(x_1,x_2) = 10(x_2 - x_1^2)$$

の極小点 $\boldsymbol{x}^*$を，$\boldsymbol{x}_0 = (-1.2, 1.0)^T$ から出発して求める.

```
C     **EXAMPLE**
      DIMENSION X(2),F(2),VW(4,4)
      EXTERNAL ROSEN
      X(1)=-1.2
      X(2)=1.0
      N=2
      M=2
      EPSR=1.0E-3
      MAX=100*2*2
      CALL NOLF1(X,N,ROSEN,M,EPSR,MAX,
     *           F,SUMS,VW,4,ICON)
      WRITE(6,600) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,610) SUMS,MAX
      WRITE(6,620) (I,X(I),I=1,N)
      WRITE(6,630) (I,F(I),I=1,M)
      STOP
  600 FORMAT('1','*ICON=',I5)
  610 FORMAT(' ',' SUM OF SQUARES= ',
     *E15.7,' MAX= ',I5/)
  620 FORMAT(1X,' X(',I2,')=',E15.7)
  630 FORMAT(1X,' F(',I2,')=',E15.7)
      END
C     OBJECTIVE FUNCTION
      SUBROUTINE ROSEN (X,Y)
      DIMENSION X(2),Y(2)
      Y(1)=1.0-X(1)
      Y(2)=(X(2)-X(1)*X(1))*10.0
      RETURN
      END
```

(4) 手法概要

n 変数の m 個の実関数 $f_1(x), f_2(x), \ldots, f_m(x)$ に対して,

$$f(x) = \|f(x)\|^2 = f^T(x) f(x) = \sum_{i=1}^{m} \{f_i(x)\}^2 \tag{4.1}$$

の極小値を与えるベクトル $\boldsymbol{x}^*$と関数値 $F(\boldsymbol{x}^*)$を求める．ただし,

$$f(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T$$
$$x = (x_1, x_2, \ldots, x_n)^T \tag{4.2}$$

とする．本サブルーチンでは，この問題を LMM 法により解く．この方法を述べる前に，この方法の原型である Levenberg-Marquardt 法と，Newton-Gauss 法，最急降下法との関連について説明する.

今，極小値を与えるベクトル $\boldsymbol{x}^*$の近似ベクトル $\boldsymbol{x}_k$ が得られているとする．そこで,

$$\boldsymbol{x}^* = \boldsymbol{x}_k + \Delta\boldsymbol{x}_k \tag{4.3}$$

とし，$\boldsymbol{f}(\boldsymbol{x})$ を $\boldsymbol{x}_k$の近傍で，1 次項までテーラー展開すると(4.4)となる.

$$\boldsymbol{f}(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k) = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{J}(\boldsymbol{x}_k)\Delta\boldsymbol{x}_k \tag{4.4}$$

ここで，$\boldsymbol{J}(\boldsymbol{x}_k)$は(4.5)で示される $\boldsymbol{f}(\boldsymbol{x})$ のヤコビアン行列である.

$$\boldsymbol{J}(\boldsymbol{x}_k) \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \cdots & & \cdots \\ \cdots & & \cdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}_{\boldsymbol{x} = \boldsymbol{x}_k} \tag{4.5}$$

以降 $\boldsymbol{J}(\boldsymbol{x}_k)$ は，$\boldsymbol{J}_k$ と表す.

(4.4)より，$F(x_k + \Delta x_k)$ の値は $|F(x_k)|$ が十分小であると仮定すれば，(4.6)で近似される.

$$\begin{aligned} & F(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k) \\ &= \boldsymbol{f}^T(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k)\boldsymbol{f}(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k) \\ &\approx \boldsymbol{f}^T(\boldsymbol{x}_k)\boldsymbol{f}(\boldsymbol{x}_k) + 2\boldsymbol{f}^T(\boldsymbol{x}_k)\boldsymbol{J}_k\Delta\boldsymbol{x}_k \\ &\quad + \Delta\boldsymbol{x}_k^T\boldsymbol{J}_k^T\boldsymbol{J}_k\Delta\boldsymbol{x}_k \end{aligned} \tag{4.6}$$

この値を極小化する $\Delta\boldsymbol{x}_k$ の値は，(4.6)の右辺を $\Delta\boldsymbol{x}_k$ に関して微分して得られる連立 1 次方程式(4.7)の解として与えられる.

$$\boldsymbol{J}_k^T\boldsymbol{J}_k\Delta\boldsymbol{x}_k = -\boldsymbol{J}_k^T\boldsymbol{f}(\boldsymbol{x}_k) \tag{4.7}$$

(4.7)は正規方程式と呼ばれ，この $\Delta\boldsymbol{x}_k$ により,

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta\boldsymbol{x}_k$$

と反復するのが，Newton-Gauss 法である.

この方法は，$\Delta\boldsymbol{x}_k$ の方向としては $F(\boldsymbol{x})$ の降下方向ではあるが，$\Delta\boldsymbol{x}_k$ 自体は発散する可能性がある.

一方，$F(\boldsymbol{x})$の $\boldsymbol{x}_k$における傾斜ベクトル $\nabla F(\boldsymbol{x}_k)$ は,

$$\nabla F(\boldsymbol{x}_k) = 2\boldsymbol{J}_k^T\boldsymbol{f}(\boldsymbol{x}_k) \tag{4.8}$$

で与えられる．$-\nabla F(\boldsymbol{x}_k)$は，$F(\boldsymbol{x})$の $\boldsymbol{x}_k$ における最急降下方向であり,

$$\Delta\boldsymbol{x}_k = -\nabla F(\boldsymbol{x}_k) \tag{4.9}$$

とするのが，最急降下法である.

(4.9)の $\Delta \boldsymbol{x}_k$ は，$F(\boldsymbol{x})$の減少を最も確実に保証するが，反復を繰り返すとジグザグ運動を始めるという欠点が，多くの数値例で指摘されている．

そこで，Levenberg，Marquardt，Morrison は，次の方程式より $\Delta \boldsymbol{x}_k$ を決定することを提案した．

$$\left\{\boldsymbol{J}_k^T \boldsymbol{J}_k + v_k^2 \boldsymbol{I}\right\} \Delta \boldsymbol{x}_k = -\boldsymbol{J}_k^T \boldsymbol{f}\left(\boldsymbol{x}_k\right) \qquad (4.10)$$

ここで，v_k は正数(Marquardt 数と呼ぶ)である．(4.10)より決まる $\Delta \boldsymbol{x}_k$ は，明らかに v_k の値に依存する．即ち，$v_k \to 0$ とすれば，$\Delta \boldsymbol{x}_k$ の方向は Newton-Gauss 法による方向となり，v_k が 0 から増加するにつれて $\left\|\Delta \boldsymbol{x}_k\right\|$ は単調に減少する．更に，v_k が増加するにつれて，$\Delta \boldsymbol{x}_k$ と最急降下方向 $-\boldsymbol{J}_k^T \boldsymbol{f}\left(\boldsymbol{x}_k\right)$ との間の角度は単調に減少する．そして，$v_k \to \infty$ とすれば，$\Delta \boldsymbol{x}_k$ の方向は最急降下法による方向となる．

Levenberg-Marquardt 系の方法の特徴は，最適な v_k の値を反復の過程で動的に決定し，効率よく $F(\boldsymbol{x})$の極小化を図るものである．

LMM法

(4.10)の方法は，正規方程式系を陽に形成するため，数値的安定性に関する欠陥があることが知られている．ところで，(4.10)は次の系に対する最小二乗法と同値である．

$$\begin{bmatrix} \boldsymbol{J}_k \\ \cdots\cdots \\ v_k \boldsymbol{I} \end{bmatrix} \Delta \boldsymbol{x}_k = -\begin{bmatrix} \boldsymbol{f}\left(\boldsymbol{x}_k\right) \\ \cdots\cdots\cdots \\ \boldsymbol{O} \end{bmatrix} \qquad (4.11)$$

すなわち，(4.11)の残差の 2 乗和の最小化の問題が(4.10)で表現される．

LMM 法は，正規方程式系を陽に形成せず，(4.11)に数値的安定性の高い直交変換を適用する最小二乗法により，$\Delta \boldsymbol{x}_k$ を求める方法である．

本サブルーチンでは(4.11)により $\Delta \boldsymbol{x}$ を求め，

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$$

として，

$$F(\boldsymbol{x}_{k+1}) < F(\boldsymbol{x}_k)$$

を満たすように反復し，極小点 $\boldsymbol{x}^*$を求める．

本サブルーチンの計算手順

① 初期設定
Marquardt 数 v_0 を設定する．
$\boldsymbol{f}\left(\boldsymbol{x}_0\right)$，$\boldsymbol{F}\left(\boldsymbol{x}_0\right)$ を求める．
$k = 0$ とする．

② $\boldsymbol{J}_k$ を差分近似により求める．

③ (4.11)を最小二乗法により解き，$\Delta \boldsymbol{x}_k$ を求める．
$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$ とする．

$\boldsymbol{f}\left(\boldsymbol{x}_{k+1}\right), F\left(\boldsymbol{x}_{k+1}\right)$ を求める．

④ $F(\boldsymbol{x}_{k+1}) < F(\boldsymbol{x}_k)$を満たすか否かを判定する．満たす場合は⑧へ進む．

⑤ 収束判定を行う．
満たす場合は，$\boldsymbol{x}_k$ を極小点 $\boldsymbol{x}^*$とみなし，ICON = 0 として処理を終了する．

⑥ Marquardt 数を大きくする．すなわち，

$$v_k = 1.5\ v_k$$

とする．

⑦ Marquardt 数の上界を判定する．すなわち，

$$v_k \le 1/u,\ u \text{ は丸め誤差の単位} \qquad (4.14)$$

を満たす場合は,③へ進み反復を継続する．満たさない場合は，ICON = 20000 として処理を終了する．

⑧ 収束判定を行う．
満たす場合は，$\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$とみなし，ICON = 0 として処理を終了する．

⑨ ⑥の過程を通過しなかった場合は，Marquardt 数を小さくする．すなわち，

$$v_k = 0.5\ v_k$$

とする．
$k = k + 1$ として②へ進み反復を継続する．

アルゴリズム上の留意点

① Marquardt 数 v_0 の設定
Marquardt 数の初期値として，$\boldsymbol{x}_0$ におけるヤコビアン行列のノルムを用いている．すなわち，

$$v_0 = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\partial f_i / \partial x_j\right)^2 / (m \cdot n)} \qquad (4.15)$$

② ヤコビアン行列 $\boldsymbol{J}_k$ の差分近似法

$$\boldsymbol{J}_k = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots\cdots & \frac{\partial f_n}{\partial x_n} \\ \cdots & & \cdots \\ \cdots & & \cdots \\ \frac{\partial f_m}{\partial x_1} & \cdots\cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}_{\boldsymbol{x} = \boldsymbol{x}_k} \qquad (4.16)$$

を計算するために，前進差分(4.17)を用いている．

$$\frac{\partial f_i}{\partial x_j} \approx \left\{ f_i\left(\boldsymbol{x}_k + h\boldsymbol{e}_j\right) - f_i\left(\mathbf{x}_k\right) \right\} / h \qquad (4.17)$$

ここで，$\boldsymbol{e}_j$は第 j 単位ベクトル

$h = \sqrt{u}$，u は丸め誤差の単位

③ 最小二乗法による $\Delta \boldsymbol{x}_k$ の計算

(4.11)を最小二乗法により解き $\Delta \boldsymbol{x}_k$ を求めるために，本サブルーチンではハウスホルダー法を用いている．

すなわち，(4.11)の左辺に対し左からハウスホルダー変換の直交行列 $\boldsymbol{Q}$ をかけて，上三角行列化する．

$$\boldsymbol{Q}\begin{bmatrix} \boldsymbol{J}_k \\ \cdots\cdots \\ v_k \boldsymbol{I} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} \\ \cdots \\ \boldsymbol{O} \end{bmatrix} \qquad (4.18)$$

ここで，$\boldsymbol{R}$ は $n \times n$ の上三角行列である．

同様に(4.11)の右辺にも直交変換を行う．

$$-\boldsymbol{Q}\begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}_k) \\ \cdots\cdots\cdots \\ \boldsymbol{O} \end{bmatrix} = -\begin{bmatrix} \boldsymbol{g}_1 \\ \cdots\cdots \\ \boldsymbol{g}_2 \end{bmatrix} \qquad (4.19)$$

ここで，$\boldsymbol{g}_1$ は n 次元ベクトル，$\boldsymbol{g}_2$ は m 次元ベクトルである．

直交変換に対してノルムは不変であるから，(4.11)の最小二乗解 $\Delta \boldsymbol{x}_k$ を(4.20)により求める．

$$\boldsymbol{R}\Delta \boldsymbol{x}_k = -\boldsymbol{g}_1 \qquad (4.20)$$

$\boldsymbol{R}$ は上三角行列であるから，(4.20)は後退代入により計算できる．

④ 収束判定法

本サブルーチンでは反復過程において，次のような収束判定を行っている．

- $F(\boldsymbol{x}_{k+1}) < F(\boldsymbol{x}_k)$かつ
  $\left\| \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \right\|_\infty \le \max\left(1.0, \left\| \boldsymbol{x}_k \right\|_\infty\right) \cdot \mathrm{EPSR}$ を満たすとき，$\boldsymbol{x}_{k+1}$ を極小点 $\boldsymbol{x}^*$とみなす．
- $F(\boldsymbol{x}_{k+1}) \ge F(\boldsymbol{x}_k)$かつ
  $\left\| \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \right\|_\infty \le \max\left(1.0, \left\| \boldsymbol{x}_k \right\|_\infty\right) \cdot \mathrm{EPSR}$ を満たすとき，$\boldsymbol{x}_k$ を極小点 $\boldsymbol{x}^*$とみなす．

なお詳細については，参考文献[36]，[37]を参照すること．

D15-20-0101 NOLG1, DNOLG1

関数二乗和の極小化 （微係数要，改訂マルカート法）
CALL NOLG1 (X, N, FUN, JAC, M, EPSR, MAX, F, SUMS, VW, K, ICON)

(1) 機能

n 変数の m 個の実関数 $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ とそのヤコビアン $\boldsymbol{J}(\boldsymbol{x})$，及び初期ベクトル $\boldsymbol{x}_0$ が与えられたとき，改訂マルカート法 (Levenberg-Marquardt-Morrison 法．以下 LMM 法と略す)により，

$$F(\boldsymbol{x}) = \sum_{i=1}^{m} \left\{ f_i(\boldsymbol{x}) \right\}^2 \qquad (1.1)$$

の極小値を与えるベクトル $\boldsymbol{x}^*$ とその関数値 $F(\boldsymbol{x}^*)$ を求める．本サブルーチンでは $f_i(\boldsymbol{x}), i = 1, \ldots, m$ は 1 階までの連続な偏微分をもつものと仮定する．また，$m \geq n \geq 1$ であること．

(2) パラメタ

X…… **入力**．初期ベクトル $\boldsymbol{x}_0$.
出力．ベクトル $\boldsymbol{x}^*$.
大きさ n の1次元配列.

N…… **入力**．変数の個数 n.

FUN…… **入力**． $f_i(\boldsymbol{x})$ を計算するサブルーチン副プログラム名.
[副プログラムの用意の仕方]
SUBROUTINE FUN (X, Y)
パラメタ
X ..... **入力**．変数ベクトル $\boldsymbol{x}$.
大きさ n の1次元配列.
Y ..... **出力**．変数ベクトル $\boldsymbol{x}$ に対する関数値 $f_i(\boldsymbol{x})$.
$\mathrm{F}(1) = f_1(\boldsymbol{x}), \ldots, \mathrm{F}(M) = f_m(\boldsymbol{x})$ なる対応を持つ，大きさ m の1次元配列.

JAC…… **入力**．ヤコビアン $\boldsymbol{J}(\boldsymbol{x})$ を計算するサブルーチン副プログラム名.
[副プログラムの用意の仕方]
SUBROUTINE JAC (X, G, K)
パラメタ
X……**入力**． 変数ベクトル $\boldsymbol{x}$.
大きさ n の1次元配列.
G……**出力**．変数ベクトル $\boldsymbol{x}$ に対するヤコビアン行列.
$\mathrm{G}(\mathrm{I}, \mathrm{J}) = \partial f_i / \partial x_j$ なる対応を持つ，G (K, N)なる2次元配列.
K……入力．配列 G の整合寸法.

M …… **入力**．関数の個数 m.

EPSR …… **入力**．収束判定値(≥ 0.0).
0.0のときは，標準値が採用される.
(使用上の注意②参照).

MAX…… **入力**．関数の評価回数の上限(≠0).
(使用上の注意③参照).
出力．実際に評価した回数(> 0).

F…… **出力**．関数値 $f_i(\boldsymbol{x}^*)$.
$\mathrm{F}(1) = f_1(\boldsymbol{x}^*), \ldots, \mathrm{F}(\mathrm{M}) = f_m(\boldsymbol{x}^*)$ なる対応を持つ．大きさ m の1次元配列.

SUMS …… **出力**．関数二乗和 $F(\boldsymbol{x}^*)$ の値.

VW …… **作業領域**．VW (K, N + 2)なる2次元配列.

K…… **入力**．配列VWの整合寸法(≥ $m + n$).

ICON …… **出力**．コンディションコード.
表NOLG1-1参照.

表NOLG1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	指定された反復回数内で収束条件が満たされなかった.	パラメタ X，F, SUMSには最後の値を格納する.
20000	計算の過程で，Marquardt数 v_k が上界を超えた. (手法概要(4.14)参照). EPSRが小さすぎるか，ヤコビアン行列の差分近似の誤差が計算限界を超えた.	処理を打ち切る. (パラメタ X，F, SUMSには最後の値を格納する．)
30000	N < 1, M < N, EPSR < 0.0, MAX = 0 又は, K < $m+n$ であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II .. AMACH, MGSSL
② FORTRAN基本関数... ABS, SORT, FLOAT

b. 注意
① 本サブルーチンを呼び出す側のプログラム内で，引数FUN及びJACに相当する副プログラム名をEXTERNAL文で宣言すること.
② EPSRの与え方
本サブルーチンでは，$F(\boldsymbol{x})$ が極小点 $\boldsymbol{x}^*$ の近傍で近似的に2次関数であることを前提としており，$F(\boldsymbol{x}^*)$ を丸め誤差程度まで正しく求めるためには,

$\mathrm{EPSR} \approx \sqrt{u}$ ，u は丸め誤差の単位

程度に与えるのが適当である.
EPSRの標準値は $2 \cdot \sqrt{\mathrm{u}}$ である.
③ MAXの与え方
関数の評価回数は，変数ベクトル $\boldsymbol{x}$ に対して，$\mathrm{f}_i(x), \mathrm{i} = 1, \ldots, \mathrm{m}$ を計算することを1回，$\boldsymbol{J}(\boldsymbol{x})$ を計算することを n 回と数える.
この関数の評価回数は，関数 $\left\{ f_i(\boldsymbol{x}) \right\}$ の性質，初期ベクトル，収束判定値などに大きく依存する.
一般には，初期ベクトルが良く，収束判定値として標準値を採用した場合は，MAX = 100•n m 程度が妥当である.

指定評価回数内で収束条件が満たされずICON = 10000として戻った場合でも，再度本サブルーチンを呼び出すことにより，継続して反復することができる．しかし，この場合はパラメタMAXには，負の値で追加評価回数を指定し，他のパラメタの内容は保存したままで呼び出すこと．

c. 使用例

$F(x_1,x_2)=f_1^2(x_1,x_2)+f_2^2(x_1,x_2)$

ここで，$f_1(x_1,x_2)=1-x_1$

$f_2(x_1,x_2)=10(x_2-x_1^2)$

の極小点$\boldsymbol{x}^*$を$\boldsymbol{x}_0=(-1.2, 1.0)^T$から出発して求める．

```
C     **EXAMPLE**
      DIMENSION X(2),F(2),VW(4,4)
      EXTERNAL ROSEN,ROSENJ
      X(1)=-1.2
      X(2)=1.0
      N=2
      M=2
      EPSR=1.0E-3
      MAX=100*2*2
      CALL NOLG1(X,N,ROSEN,ROSENJ,M,EPSR,
     *           MAX,F,SUMS,VW,4,ICON)
      WRITE(6,600) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,610) SUMS,MAX
      WRITE(6,620) (I,X(I),I=1,N)
      WRITE(6,630) (I,F(I),I=1,M)
      STOP
  600 FORMAT('1','*ICON=',I5)
  610 FORMAT(' ',' SUM OF SQUARES= ',
     * E15.7,' MAX= ',I5/)
  620 FORMAT(1X,' X(',I2,')=',E15.7)
  630 FORMAT(1X,' F(',I2,')=',E15.7)
      END
C     OBJECTIVE FUNCTION
      SUBROUTINE ROSEN (X,Y)
      DIMENSION X(2),Y(2)
      Y(1)=1.0-X(1)
      Y(2)=(X(2)-X(1)*X(1))*10.0
      RETURN
      END
C     JACOBIAN
      SUBROUTINE ROSENJ(X,G,K)
      DIMENSION X(2),G(K,2)
      G(1,1)=-1.0
      G(2,2)=-20.0*X(1)
      G(1,2)=0.0
      G(2,2)=10.0
      RETURN
      END
```

(4) 手法概要

n 変数の m 個の実関数$f_1(x),f_2(x),\ldots,f_m(x)$に対して，

$$F(\boldsymbol{x})=\|\boldsymbol{f}(\boldsymbol{x})\|^2=\boldsymbol{f}^T(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x})=\sum_{i=1}^{m}\{f_i(\boldsymbol{x})\}^2 \tag{4.1}$$

の極小値を与えるベクトル $\boldsymbol{x}^*$と関数値 $F(\boldsymbol{x}^*)$を求める．ただし，

$$\boldsymbol{f}(\boldsymbol{x})=(f_1(\boldsymbol{x}),f_2(\boldsymbol{x}),\ldots,f_m(\boldsymbol{x}))^T$$
$$x=(x_1,x_2,\ldots,x_n)^T \tag{4.2}$$

とする．本サブルーチンでは，この問題を LMM 法により解く．この方法を述べる前に，この方法の原型である Levenberg-Marquardt 法と，Newton-Gauss 法，最急降下法との関連について説明する．

今，極小値を与えるベクトル $\boldsymbol{x}^*$の近似ベクトル $\boldsymbol{x}_k$が得られているとする．そこで，

$$\boldsymbol{x}^*=\boldsymbol{x}_k+\Delta\boldsymbol{x}_k \tag{4.3}$$

とし，$\boldsymbol{f}(\boldsymbol{x})$を $\boldsymbol{x}_k$の近傍で，1 次項までテーラー展開すると(4.4)となる．

$$\boldsymbol{f}(\boldsymbol{x}_k+\Delta\boldsymbol{x}_k)=\boldsymbol{f}(\boldsymbol{x}_k)+\boldsymbol{J}(\boldsymbol{x}_k)\Delta\boldsymbol{x}_k \tag{4.4}$$

ここで，$\boldsymbol{J}(\boldsymbol{x}_k)$は(4.5)で示される $\boldsymbol{f}(\boldsymbol{x})$のヤコビアン行列である．

$$J(x_k)=\left[\begin{array}{ccc}\frac{\partial f_1}{\partial x_1} & \cdots\cdots & \frac{\partial f_1}{\partial x_n}\\ \cdots & & \cdots\\ \cdots & & \cdots\\ \frac{\partial f_m}{\partial x_1} & \cdots\cdots & \frac{\partial f_m}{\partial x_n}\end{array}\right]\boldsymbol{x}=\boldsymbol{x}_k \tag{4.5}$$

以降$\boldsymbol{J}(\boldsymbol{x}_k)$は，$\boldsymbol{J}_k$と表す．

(4.4)より，$F(x_{0k}+\Delta x_k)$の値は$|F(\boldsymbol{x}_k)|$が十分小であると仮定すれば，(4.6)で近似される．

$$\begin{aligned}&F(\boldsymbol{x}_k+\Delta\boldsymbol{x}_k)\\&=\boldsymbol{f}^T(\boldsymbol{x}_k+\Delta\boldsymbol{x}_k)\boldsymbol{f}(\boldsymbol{x}_k+\Delta\boldsymbol{x}_k)\\&\approx\boldsymbol{f}^T(\boldsymbol{x}_k)f(\boldsymbol{x}_k)+2\boldsymbol{f}^T(\boldsymbol{x}_k)\boldsymbol{J}_k\Delta\boldsymbol{x}_k\\&+\Delta\boldsymbol{x}_k^T\boldsymbol{J}_k^T\boldsymbol{J}_k\Delta x_k\end{aligned} \tag{4.6}$$

この値を極小化する $\Delta\boldsymbol{x}_k$ の値は，(4.6)の右辺を $\Delta\boldsymbol{x}_k$ に関して微分して得られる連立 1 次方程式(4.7)の解として与えられる．

$$\boldsymbol{J}_k^T\boldsymbol{J}_k\Delta\boldsymbol{x}_k=-\boldsymbol{J}_k^T\boldsymbol{f}(\boldsymbol{x}_k) \tag{4.7}$$

(4.7)は正規方程式と呼ばれ，この $\Delta\boldsymbol{x}_k$ により，

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\Delta\boldsymbol{x}_k$$

と反復するのが，Newton-Gauss 法である．

この方法は，$\Delta\boldsymbol{x}_k$ の方向としては $F(\boldsymbol{x})$の降下方向ではあるが，$\Delta\boldsymbol{x}_k$ 自体は発散する可能性がある．

一方，$F(\boldsymbol{x})$の $\boldsymbol{x}_k$における傾斜ベクトル$\nabla\boldsymbol{F}(\boldsymbol{x}_k)$は

$$\nabla\boldsymbol{F}(\boldsymbol{x}_k)=2\boldsymbol{J}_k^T\boldsymbol{f}(\boldsymbol{x}_k) \tag{4.8}$$

で与えられる．$-\nabla F(\boldsymbol{x}_k)$ は $F(\boldsymbol{x})$の $\boldsymbol{x}_k$ における最急降下方向であり，

$$\Delta \boldsymbol{x}_k = -\nabla \boldsymbol{F}(\boldsymbol{x}_k) \tag{4.9}$$

とするが，最急降下法である．
(4.9)の $\Delta \boldsymbol{x}_k$ は，$F(\boldsymbol{x})$の減少を最も確実に保証するが，反復を繰り返すとジグザグ運動を始めるという欠点が，多くの数値例で指摘されている．

そこで，**Levenberg, Marquardt, Morrison** は，次の方程式により $\Delta \boldsymbol{x}_k$ を決定することを提案した．

$$\left\{\boldsymbol{J}_k^T \boldsymbol{J}_k + \nu_k^2 \boldsymbol{I}\right\}\Delta \boldsymbol{x}_k = -\boldsymbol{J}_k^T \boldsymbol{f}(\boldsymbol{x}_k) \tag{4.10}$$

ここで，ν_kは正数(Marquardt数と呼ぶ)である．
(4.10)より決まる $\Delta \boldsymbol{x}_k$ は，明らかに ν_k の値に依存する．即ち，$\boldsymbol{\nu}_k \to 0$ とすれば，$\Delta \boldsymbol{x}_k$ の方向は Newton-Gauss 法による方向となり，ν_k が 0 から増加するにつれて $\|\Delta \boldsymbol{x}_k\|$ は単調に減少する．更に，ν_k が増加するにつれて，$\Delta \boldsymbol{x}_k$ と最急降下方向 $-\boldsymbol{J}_k^T \boldsymbol{f}(\boldsymbol{x}_k)$ との間の角度は単調に減少する．そして，$\nu_k \to \infty$ とすれば，$\Delta \boldsymbol{x}_k$ の方向は最急降下法による方向となる．

Levenberg-Marquardt 系の方法の特徴は，最適な ν_k の値を反復の過程で動的に決定し，効率よく $F(\boldsymbol{x})$の極小値を図るものである．

LMM法

(4.10)の方法は，正規方程式系を陽に形成するため，数値的安定性に関する欠点があることが知られている．ところで，(4.10)は次の系に対する最小二乗法と同値である．

$$\begin{bmatrix} \boldsymbol{J}_k \\ \cdots\cdots \\ \nu_k \boldsymbol{I} \end{bmatrix} \Delta \boldsymbol{x}_k = -\begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}_k) \\ \cdots\cdots\cdots \\ \boldsymbol{O} \end{bmatrix} \tag{4.11}$$

すなわち，(4.11)の残差の 2 乗和の最小化の問題が (4.10)で表現される．

LMM 法は，正規方程式系を陽に形成せず，(4.11)に数値的安定性の高い直交変換を適用する最小二乗法により，$\Delta \boldsymbol{x}_k$ を求める方法である．

本サブルーチンでは(4.11)により $\Delta \boldsymbol{x}$ を求め，

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$$

として，

$$F(\boldsymbol{x}_{k+1}) < F(\boldsymbol{x}_k)$$

を満たすように反復し，極小点 $\boldsymbol{x}^*$を求める．

本サブルーチンの計算手順

① 初期設定
Marquardt数ν_0を設定する．
$\boldsymbol{f}(\boldsymbol{x}_0)$， $F(\boldsymbol{x}_0)$を求める．
$k = 0$とする．

② $\boldsymbol{J}_k$を求める．

③ (4.11)を最小二乗法により解き，$\Delta \boldsymbol{x}_k$ を求める．
$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta \boldsymbol{x}_k$ とする．
$\boldsymbol{f}(\boldsymbol{x}_{k+1}), F(\boldsymbol{x}_{k+1})$ を求める．

④ $F(\boldsymbol{x}_{k+1}) < F(\boldsymbol{x}_k)$を満たすか否かを判定する．満たす場合は⑧へ進む．

⑤ 収束判定を行う．
満たす場合は，$\boldsymbol{x}_k$を極小点$\boldsymbol{x}^*$とみなし，ICON = 0として処理を終了する．

⑥ Marquardt数を大きくする．すなわち，

$$\nu_k = 1.5\ \nu_k$$

とする．

⑦ Marquardt数の上界を判定する．すなわち，

$$\nu_k \le 1/u,\ u \text{ は丸め誤差の単位} \tag{4.14}$$

を満たす場合は,③へ進み反復を継続する．満たさない場合は，ICON = 20000として処理を終了する．

⑧ 収束判定を行う．
満たす場合は，$\boldsymbol{x}_{k+1}$を極小点$\boldsymbol{x}^*$とみなし，ICON = 0として処理を終了する．

⑨ ⑥の過程を通過しなかった場合は，Marquardt数を小さくする．すなわち，

$$\nu_k = 0.5\ \nu_k$$

とする．
$k = k + 1$として②へ進み反復を継続する．

アルゴリズム上の留意点

① Marquardt数ν_0の設定
Marquardt数の初期値として，$\boldsymbol{x}_0$におけるヤコビアン行列のノルムを用いている．すなわち，

$$\nu_0 = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\partial f_i / \partial x_j\right)^2 / (m \cdot n)} \tag{4.15}$$

② 最小二乗法による $\Delta \boldsymbol{x}_k$ の計算
(4.11)を最小二乗法により解き $\Delta \boldsymbol{x}_k$ を求めるために，本サブルーチンではハウスホルダー法を用いている．
すなわち，(4.11)の左辺に対し左からハウスホルダー変換の直交行列$\boldsymbol{Q}$をかけて，上三角行列化する．

$$\boldsymbol{Q}\begin{bmatrix}\boldsymbol{J}_k\\ \cdots\cdots\\ v_k\boldsymbol{I}\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}\\ \cdots\\ \boldsymbol{O}\end{bmatrix} \tag{4.16}$$

ここで，$\boldsymbol{R}$は$n\times n$の上三角行列である．
同様に(4.11)の右辺にも直交変換を行う．

$$-Q\begin{bmatrix}f(x_k)\\ \cdots\cdots\cdots\\ O\end{bmatrix}=-\begin{bmatrix}g_1\\ \cdots\cdots\\ g_2\end{bmatrix} \tag{4.17}$$

ここで，$\boldsymbol{g}_1$はn次元ベクトル，$\boldsymbol{g}_2$はm次元ベクトルである．
直交変換に対してノルムは不変であるから，(4.11)の最小二乗解 $\Delta\boldsymbol{x}_k$ を(4.18)により求める．

$$\boldsymbol{R}\Delta\boldsymbol{x}_k=-\boldsymbol{g}_1 \tag{4.18}$$

$\boldsymbol{R}$は上三角行列であるから，(4.18)は後退代入により計算できる．

③ 収束判定法
本サブルーチンでは反復過程において，次のような収束判定を行っている．

- $F(\boldsymbol{x}_{k+1})<F(\boldsymbol{x}_k)$かつ
  $\left\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_k\right\|_\infty\le\max\left(1.0,\left\|\boldsymbol{x}_k\right\|_\infty\right)\cdot\mathrm{EPSR}$ を満たすとき，$\boldsymbol{x}_{k+1}$を極小点$\boldsymbol{x}^*$とみなす．
- $F(\boldsymbol{x}_{k+1})\ge F(\boldsymbol{x}_k)$かつ
  $\left\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_k\right\|_\infty\le\max\left(1.0,\left\|\boldsymbol{x}_k\right\|_\infty\right)\cdot\mathrm{EPSR}$ を満たすとき，$\boldsymbol{x}_k$を極小点$\boldsymbol{x}^*$とみなす．

なお詳細については，参考文献 [36]，[37]を参照すること．

B21-11-0702 NRML, DNRML

実行列の固有ベクトルの正規化
CALL NRML (EV, K, N, IND, M, MODE, ICON)

(1) 機能

n 次の実行列の m 個の固有ベクトル $\boldsymbol{x}_i$ (i = 1, …, m)が与えられたとき，(1.1)，又は(1.2)の正規化を行い $\boldsymbol{y}_i$ を求める．

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_\infty \qquad (1.1)$$

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_2 \qquad (1.2)$$

ただし，$n \geq 1$ であること．

(2) パラメタ

EV ……**入力**．m個の固有ベクトル$\boldsymbol{x}_i$ (i = 1, …, m) (使用上の注意参照).
出力．正規化された固有ベクトル$\boldsymbol{y}_i$.
EV(K, M)なる2次元配列.

K……**入力**．配列EVの整合寸法($\geq n$).

N……**入力**．実行列の次数n.

IND……**入力**．EVに格納する各固有ベクトルが，実固有ベクトルか複素固有ベクトルかの指示.
EVのJ列目が，実固有ベクトルならばIND(J) = 1，複素固有ベクトルの実部ならば，IND(J) = -1，虚部ならばIND(J) = 0と与えること.
大きさMの1次元配列.

M……**入力**．配列INDの大きさ($\boldsymbol{m} \leq \mathbf{M} \leq \boldsymbol{n}$)

MODE…**入力**．正規化の方法の指示
MODE = 1のときは(1.1)の正規化を行う.
MODE = 2のときは(1.2)の正規化を行う.

ICON……**出力**．コンディションコード.
表NRML-1参照.

表NRML-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	n = 1であった.	EV (1,1) =1.0 とする.
30000	N<M，M<1，K<N，MODE≠1，2又はINDに誤りがあった.	処理を打ち切る.

(3) 使用上の注意

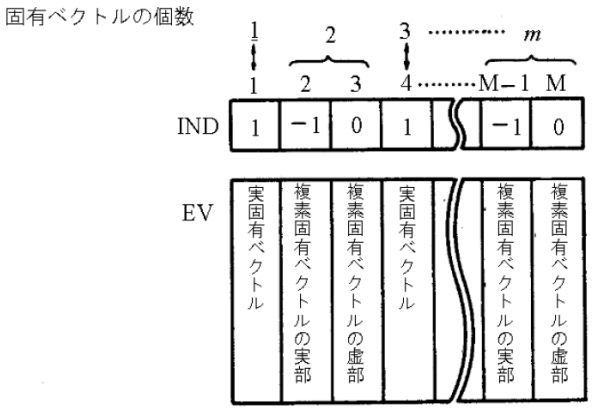
a. 使用する副プログラム

① SSL II.....MGSSL

② FORTRAN基本関数..... ABS, SQRT

b. 注意

① 固有ベクトルEVへの格納は，実固有ベクトルは1列ごとに，複素固有ベクトルは実部と虚部を対にして2列ごとに格納すること．(図NRML-1参照のこと).



図NRML-1 INDとEVの関係

なお，サブルーチンHVECあるいはサブルーチンHBK1呼び出し後のパラメタEV，IND，Mは本サブルーチンでそのまま入力できるようになっている.

② 実対称行列の固有ベクトルの正規化を行う場合は，INDの内容をすべて1で与えればよい.

c. 使用例

サブルーチンEIG1により求めたn次の実行列の固有ベクトルを，本サブルーチンにより $\|\boldsymbol{x}\|_\infty = 1$ となるように正規化しなおす．n≤100の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),ER(100),
     *          EI(100),EV(100,100),
     *          VW(100),IND(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N
      DO 20 I=1,N
      WRITE(6,610) (I,J,A(I,J),J=1,N)
   20 CONTINUE
      CALL EIG1(A,100,N,0,ER,EI,EV,VW,
     *          ICON)
      WRITE(6,620) ICON
      IF(ICON.GT.10000) GO TO 10
      DO 30 I=1,N
   30 IND(I)=1
      DO 40 I=1,N
      IF(EI(I).EQ.0.0) GO TO 40
      IF(IND(I).EQ.0) GO TO 40
      IND(I)=-1
      IND(I+1)=0
   40 CONTINUE
      CALL NRML(E,V,100,N,IND,N,1,ICON)
      CALL EPRT(ER,EI,EV,IND,100,N,N)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',
     * 5X,'N=',I3/)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチンEPRTは，実行列の固有値及び固有ベクトルを出力するサブルーチンである．この詳細についてはEIG1の使用例参照のこと．

(4) 手法概要

n 次の実行列の m 個の固有ベクトル $\boldsymbol{x}_i (i = 1, ..., m)$ が与えられたとき，正規化された固有ベクトル y_i を求める．ここで，$\boldsymbol{x}_i = (x_{1i}, ..., x_{ni})^{\mathrm{T}}$ とする．

MODE = 1 のときは，ベクトル $\boldsymbol{x}_i$ の絶対値最大の要素を 1 とする正規化を行う．

$$\boldsymbol{y}_i = \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_\infty, \|\boldsymbol{x}_i\|_\infty = \max_k |x_{ki}| \qquad (4.1)$$

MODE = 2 のときは，ベクトル $\boldsymbol{x}_i$ の各要素の 2 乗の和を 1 とする正規化を行う．

$$y_i = x_i / \|x_i\|_2, \|x_i\|_2 = \sqrt{\sum_{k=1}^{n} |x_{ki}|^2} \qquad (4.2)$$

H11-20-0141 ODAM, DODAM

連立1階常微分方程式（アダムス法）
CALL ODAM (X, Y, FUN, N, XEND, ISW, EPSA, EPSR, VW, IVW, ICON)

(1) 機能

連立1階常微分方程式の初期値問題

$$\left.\begin{aligned} y_1' &= f_1(x, y_1, y_2, ..., y_N), \quad y_1(x_0) = y_{10} \\ y_2' &= f_2(x, y_1, y_2, ..., y_N), \quad y_2(x_0) = y_{20} \\ &\cdots \qquad\qquad \cdots \\ y_N' &= f_N(x, y_1, y_2, ..., y_N), \quad y_N(x_0) = y_{N0} \end{aligned}\right\} \qquad (1.1)$$

における関数系 $f_1, f_2, ..., f_N$ と，初期値 x_0, y_{10}, y_{20}, ..., y_{N0}, 及び xの最終値 x_e が与えられたとき，アダムス法により $x_m = x_0 + \sum_{j=1}^{m} h_j; m = 1,2,...,e$ における解 $(y_{1m}, y_{2m}, ..., y_{Nm})$ を求める（図 ODAM-1 参照）．

ただし，キザミ h_j は，求めるべき解が要求精度を満たすように制御される．

本サブルーチンは解の出力方法，すなわち利用者プログラムに戻るタイミングとして以下の2通りを用意しており，利用者は目的に応じて選ぶことができる．

a. 最終値出力…… 最終値 x_e における解が求まったときはじめて戻る．

b. ステップ出力…途中の解，すなわち，x_1, x_2, ...における解が求まるたびに戻る．

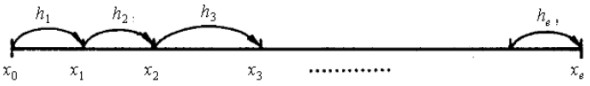


図 ODAM-1 解を求める点 x_m ($x_0 < x_e$) の場合

(2) パラメタ

X…… **入力**．初期値 x_0.
出力．最終値 x_e. ただし，ステップ出力が指定されている場合は，1ステップ前進したときの独立変数の値 x_m.

Y…… **入力**．初期値 y_{10}, y_{20}, ...,y_{N0} をY (1) = y_{10}, Y(2) = y_{20}, ..., Y(N) = y_{N0} の順に与える．大きさ N の1次元配列.
出力．最終値 x_e における解ベクトル．ただし，ステップ出力が指定されている場合は，1ステップ前進したときの x=x_m における解ベクトル.

FUN…… **入力**．(1.1) における f_i; i=1,2,⋯,Nを計算するサブルーチン副プログラム名.
［副プログラムの用意のし方］
SUBROUTINE FUN (X, Y, YP)
ここで,
X…… **入力**．独立変数 x.
Y…… **入力**．Y(1) = y_1, Y(2) = y_2, ..., Y(N) = y_N なる対応を持つ大きさ N の1次元配列.
YP…… **出力**.
$YP(1) = f_1(x, y_1, y_2, ..., y_N)$,
$YP(2) = f_2(x, y_1, y_2, ..., y_N)$,...,
$YP(N) = f_N(x, y_1, y_2, ..., y_N)$なる対応を持つ大きさ N の1次元配列.

N…… **入力**．連立1階常微分方程式の元数.

XEND…… **入力**．独立変数の最終値 x_e.

ISW …… **入力**．積分する際の条件を指定する．
ISW は10進3桁の非負の整数であり，いま，それを

$$ISW = 100d_3 + 10d_2 + d_1$$

と表すと，各 d_i は次のように指定する．

d_1: 初回呼出しであるか否かを指定する．
0 ....初回呼出しである.
1 ....継続呼出しである.
初回呼出しとは，与えられた微分方程式に対して初めて本サブルーチンを呼び出すときを言う.

d_2: 解の出力方法を指定する.
0 ...最終値出力
1 ...ステップ出力

d_3: 最終値 x_e を超えた点で，関数系 $f_1, f_2, ..., f_N$ が計算できるか否かを指定する.
0 ....計算できる.
1 .... 計算できない.
d_3 = 1 と指定する状況は，微係数が x_e を超えた点では定義されないか，あるいは x_e の近くに不連続点を持つ場合である．このような状況でないとき，むやみに d_3 = 1 とすると計算効率を著しく低下させるので注意すること.

出力．x_e における解，若しくはステップごとの解を求めて利用者プログラムに戻るとき，d_1, d_3 は次のように変更される.

d_1: d_1 = 1 とする．本サブルーチンをくり返し呼び出すときは，d_1 を1のままにしておくこと.
利用者が再び d_1 = 0 と設定し直すときは，別の方程式を解き始めるときである.

d_3: 入力時に d_3 = 1 であった場合は，は x_e における解が求まった時点で d_3 = 0 とする.

EPSA ····· **入力**．絶対誤差の上限 (≥ 0.0).
出力．入力時に EPSA が小さすぎた場合は，適当な大きさに変更された値.
（使用上の注意④参照）.

EPSR ····· **入力**．相対誤差の上限 (≥ 0.0).
出力．入力時に EPSR が小さすぎた場合は，適当な大きさに変更された値.
（使用上の注意④参照）.

VW ····· **作業領域**．大きさ 21N + 110 の1次元配列．繰り返し本サブルーチンを呼び出すときは，内容を変えてはならない.

IVW····· **作業領域**．大きさ 11 の整数型1次元配列. 繰り返し本サブルーチンを呼び出すときは，内容を変えてはならない.

ICON ····· **出力**．コンディションコード.
表 ODAM-1 参照.

表 ODAM-1 コンディションコード

コード	意味	処理内容
0	（ステップ出力の場合だけ）1ステップ積分した.	続けて呼び出すことができる.
10	XEND における解を求めた.	XEND を変えて続けて呼び出すことができる.
100	1ステップ積分した．しかし XEND に到達するのに 500 回を超えるステップが必要となることが判明した.	続けて呼び出せば，ステップ回数のカウンタを0にして，新たに回数を数えながら処理を進める.
200	1ステップ積分した．しかし解いている方程式が強いスティフ性を持つことが判明した.	続けて呼び出すこともできるが，効率の点を考えると，スティフな方程式を解くサブルーチンを利用した方がよい.
10000	EPSR 及び EPSA が演算精度に比べ小さすぎた.	EPSR 及び EPSA を適当に大きくした. 続けて呼び出すことができる.（大きくされた EPSR 及び EPSA は確認しておくのがよい．）
30000	次のいずれかであった. ① N≤ 0 ② X = XEND ③ ISW の指定に誤りがある. ④ EPSA < 0 又は EPSR < 0 ⑤ IVW の内容が変えられている（ただし，2回目以降の呼び出し時）	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL, UDE, USTE1, UNIT1

② FORTRAN 基本関数... MOD, AMIX1, AMIN1, ABS, SQRT, SIGN

b. 注意

① 本サブルーチンは，ルンゲ・クッタ法のサブルーチン ODRK1 と並んで非スティフ又は弱スティフな微分方程式を解くための標準的なプログラムである．特に，以下に示すいずれかの状況にある場合は，ODRK1よりも効果的に解を計算する.

- 関数系 $f_1, f_2, ..., f_N$ の計算に非常な時間がかかる.
- 高精度の解が要求される.
- 解の数表を作るときなどのように，独立変数の多くの値に対して解を求める.
- 解の微係数，すなわち $f_1, f_2, ..., f_N$ が不連続点を持つ.

方程式が強いスティフであることがあかじめ分っているならば，サブルーチン ODGE を利用したほうがよい.

② パラメタ FUN に対応する副プログラムの名前は本サブルーチンを呼び出す側のプログラムで，EXTERNAL 文で宣言しなければならない.

③ ICON の判定
利用者が ISW の10の位を 0 とすることにより最終値出力を指定した場合は，ICON が 10 と出力されたときに限り x_e における解を受け取ることができる．しかし，それ以前に ICON が 100, 200 あるいは 10000 として利用者プログラムに戻ることもあるので注意すること.
一方，ISW の10の位を1とすることにより，ステップ出力を指定した場合は，ICON が0と出力されたときは勿論のこと，100あるいは200と出力されたときでもステップごとの解を受け取ることができる．そのあといずれ ICON = 10 と出力されるが，このときは，x_e における解が求まったことを意味する.

④ EPSA と EPSR
微分方程式の解の成分を Y(L)，誤差（正確には局所誤差）を l_e(L) とすると，本サブルーチンは，L = 1, 2, ..., N に対して，

$$\left|l_e(\mathrm{L})\right| \le \mathrm{EPSR} * \left|\mathrm{Y}(\mathrm{L})\right| + \mathrm{EPSA} \qquad (3.1)$$

を満たすように，誤差を制御する．この式において，EPSA = 0.0 と指定すると，相対誤差判定となり，EPSR = 0.0 と指定すると，絶対誤差判定となる.

利用者は，これらの判定値を指定するに当たり，以下のことに注意する必要がある．

a. 解の大きさが非常に変化する問題では相対誤差判定が望ましい．
b. 解の大きさがさほど変化しないか，あるいは，小さな解に対しては関心が無い場合には，絶対誤差判定としてもよい．
c. しかし，安全で無駄の無い判定は，EPSA≠0 かつ EPSR≠0 として指定する混合判定である．この場合，大きい解に対しては実質的に相対判定となり，小さい解に対しては実質的に絶対判定となる．

また，本サブルーチンで達成できる最大の精度を望むときは，EPSA, EPSR を必要以上に小さく指定すればよい．このとき，サブルーチン内部で，それらの値は適当に大きくされ，利用者はこのことを，ICON = 10000 の出力で知らされる．そのあと，続けてサブルーチンを呼び出せばよい．なお，EPSA = EPSR = 0.0 と入力されたときは，EPSR = 16*u* と変更する．ただし *u* は丸め誤差の単位である．

⑤ パラメタ XEND

独立変数の連続したいくつかの値に対して解を求めたい場合は，順番に XEND を変えながら本サブルーチンを呼び出せばよい．サブルーチンは，このように繰り返し呼ばれることを想定して，利用者プログラムに戻るときはすべてのパラメタを次の呼出し時に必要な値に設定している．したがって，利用者は，単に XEND を変えて呼び出すだけでよい．あるいは必要ならば，EPSA, EPSR を変えてもよい．それ以外は変えてはならない．

⑥ 微係数の不連続点及び非定義域の処置

解及び微係数が不連続点を持つ場合には，その点を検出し，それに適応した計算をしない限り，満足のゆく精度の解は得られない．

例．方程式，

$$y' = \begin{cases} y, & 0 \le x \le 1 \\ -y, & 1 < x \le 2 \end{cases}, y(0) = 1$$

は，$0 \le x \le 1$で $y = e^x$，$1 \le x \le 2$で $y = e^{2-x}$ という解を持つが，$x = 1$ で1階微係数は不連続跳躍を持つ．

本サブルーチンは不連続点を自動的に検出し，それに適応した計算を行う能力を持っている．したがって，利用者は不連続点の存在を意識する必要はない．しかし，もし利用者が不連続点の位置を以下に示す要領で指定するならば，検出に要する手間を省くことができ，しかもより正確に計算することができる．

a. XEND を不連続点の位置に置き，同時に ISW の100 の位を1と設定してサブルーチンを呼び出す．ISW の下位2桁を変えないで，100の位を1にするには，

ISW = MOD (ISW, 100) + 100

により行える．

b. 不連続点における解が求まると，サブルーチンは ISW の100の位を0と設定して利用者プログラムに戻る．XENDを適当に前進させて，続けてサブルーチンを呼び出すときは，ISW の1の位を0に設定しておく．これは例えば，

ISW = (ISW / 10)*10

により行える．この ISW の設定は，不連続点における解を新たな初期値として，あたかも別の微分方程式を解き始めることをサブルーチンに指示する意味を持つ．

c. 使用例

連立2階常微分方程式，

$$\begin{cases} y_1'' = -\dfrac{y_1}{r^3}, y_1(0) = 1, y_1'(0) = 0 \\ y_2'' = -\dfrac{y_2}{r^3}, y_2(0) = 0, y_2'(0) = 1 \end{cases}$$

を解く．ここで，$r = \left(y_1^2 + y_2^2\right)^{1/2}$ である．この2階の方程式は，$y_3 = y'_1$, $y_4 = y'_2$ と置くことにより，次のように1階の方程式に書き直すことができる，

$$\begin{cases} y_1' = y_3, y_1(0) = 1 \\ y_2' = y_4, y_2(0) = 0 \\ y_3' = -\dfrac{y_1}{r^3}, y_3(0) = 0 \\ y_4' = -\dfrac{y_2}{r^3}, y_4(0) = 1 \end{cases} \tag{3.2}$$

以下の例は，(3.2) を，EPSA =10^{-8} , EPSR = 10^{-5} として区間 [0,2π] 上で解くものである．解は

$$x_j = \frac{2\pi}{64} \cdot j, j = 1,2,...,64$$

の各点で出力するものとする．このため，パラメタ XEND を 2π/64 ずつ増やしながら，本サブルーチンを繰り返し呼び出している．

```
C     **EXAMPLE**
      DIMENSION Y(4),VW(200),IVW(11)
      EXTERNAL FUN
      X=0.0
      Y(1)=1.0
      Y(2)=0.0
      Y(3)=0.0
      Y(4)=1.0
      N=4
      EPSA=1.0E-8
      EPSR=1.0E-5
      ISW=0
      PAI=4.0*ATAN(1.0)
      DX=PAI/32.0
C
      WRITE(6,600)
C
      DO 30 I=1,64
      XEND=DX*FLOAT(I)
   10 CALL ODAM(X,Y,FUN,N,XEND,ISW,EPSA,
     *          EPSR,VW,IVW,ICON)
      IF(ICON.EQ.10) GO TO 20
      IF(ICON.EQ.100) WRITE(6,620)
      IF(ICON.EQ.200) WRITE(6,630)
      IF(ICON.EQ.10000) WRITE(6,640)
     *                  EPSA,EPSR
      IF(ICON.EQ.30000) STOP
      GO TO 10
   20 WRITE(6,610) X,(Y(L),L=1,4)
   30 CONTINUE
      STOP
  600 FORMAT('1',12X,'X',22X,'Y(1)',
     * 16X,'Y(2)',16X,'Y(3)',16X,'Y(4)'/)
  610 FORMAT(6X,E15.8,10X,4(E15.8,5X))
  620 FORMAT(10X,'TOO MANY STEPS')
  630 FORMAT(10X,'THE EQUATIONS',
     * 1X,'APPEAR TO BE STIFF')
  640 FORMAT(10X,'TOLERANCE RESET',
     * 5X,'EPSA=',E12.5,5X,'EPSR=',E12.5)
  650 FORMAT(10X,'INVALID INPUT')
      END

      SUBROUTINE FUN(X,Y,YP)
      DIMENSION Y(4),YP(4)
      R3=(Y(1)*Y(1)+Y(2)*Y(2))**1.5
      YP(1)=Y(3)
      YP(2)=Y(4)
      YP(3)=-Y(1)/R3
      YP(4)=-Y(2)/R3
      RETURN
      END
```

(4) 手法概要

本サブルーチンは，キザミ幅制御及び次数制御付きのアダムス法を適用しており，非スティフ又は弱スティフな初期値問題を解くための標準的なサブルーチンと考えてよい．特に高精度の解を必要とする場合や，(1.1) の微係数 $f_1, f_2, ..., f_N$ が複雑で，その評価に多くの時間を費やす場合は，他の手法に比べ最も高率が良い．

以下，説明を簡単にするため，暫くの間，単一の方程式を考えることにし，それを

$$y' = f(x,y), y(x_0) = y_0 \tag{4.1}$$

と書く．点 x_m における計算解（以後，単に解という）を y_m，真の解を $y(x_m)$ で表す．また $f(x_m, y_m)$ の値を f_m で表し，$f(x_m, y(x_m))$ とは区別する．

① アダムス法の原理

解 $y_0, y_1, ..., y_m$ が既知で，

$$x_{m+1} = x_m + h_{m+1}$$

における解 y_{m+1} を求めることを考える．まず (4.1) より，

$$y(x_{m+1}) = y(x_m) + \int_{x_m}^{x_{m+1}} f(x,y)dx \tag{4.2}$$

と書ける．この式の右辺の被積分関数 $f(x,y)$ を，既に計算されているいくつかの微係数の値を使った補間多項式で近似すれば，一つの公式が得られる．今，k個の微係数を使った補間多項式 $P_{k,m}(x)$ を考え，これは，(4.3) を満たすものとする．

$$P_{k,m}(x_{m+1-j}) = f_{m+1-j}, j = 1,2,...,k \tag{4.3}$$

この $P_{k,m}(x)$ を (4.2) における $f(x,y)$ の近似として使い，更に $y(x_m)$ を，y_m で代用すると，次のような一つの解 p_{m+1} が得られる．

$$p_{m+1} = y_m + \int_{x_m}^{x_{m+1}} P_{k,m}(x)dx \tag{4.4}$$

この式は k 次の Adams-Bashforth の公式と呼ばれる．$P_{k,m}(x)$ の表現法は，いろいろ考えられるが，ここでは差分を使ったニュートンの形式を採用する．補間点 x_{m+1-j}，j = 1,2, ..., k は，キザミ幅制御の結果として不等間隔に分布するのが一般であるが，ここではまず，幅 h で等間隔に分布している場合を考える．

ニュートンの後退差分補間式による $P_{k,m}(x)$ は，

$$P_{k,m}(x) = f_m + \frac{x - x_m}{h}\nabla f_m + ... + \frac{(x-x_m)...(x-x_{m+2-k})}{h^{k-1}(k-1)!}\nabla^{k-1} f_m$$

である．これを (4.4) に代入して，

$$\left.\begin{aligned} &p_{m+1} = y_m + h\sum_{i=1}^{k}\gamma_{i-1}\nabla^{i-1}f_m \\ &\text{ここで，} \nabla^0 f_m = f_m \\ &\qquad \nabla^i f_m = \nabla^{i-1}f_m - \nabla^{i-1}f_{m-1}, i \ge 1 \\ &\qquad \gamma_0 = 1 \\ &\qquad \gamma_i = \frac{1}{i!h}\int_{x_m}^{x_{m+1}} \frac{(x-x_m)...(x-x_{m+1-i})}{h^{i-1}}dx \\ &\qquad\quad = \frac{1}{i!}\int_0^1 s(s+1)...(s+i-1)ds, i \ge 1 \end{aligned}\right\} \tag{4.5}$$

を得る．特に $k = 1$ の場合は，$P_{1,m}(x) = f_m$ となるから

$$p_{m+1} = y_m + hf_m$$

となり，これはオイラー法(Euler's method) である．
さて(4.4) の p_{m+1} を使って，$f(x_{m+1}, p_{m+1})$ を計算すれば，x_{m+1} における微係数の近似値を得たことになるが，それを使って (4.4) の $P_{k,m}(x)$ を修正し，再度積分すれば，より良い精度の解が得られることが予想できる．このような考えに基づき，(4.4) の p_{m+1} を予測子 (predictor) と呼び，修正された解を修正子 (corrector) と呼ぶ．修正子は c_{m+1} と書くことにする．$P^*_{k,m}(x)$ を

$$P^*_{k,m}(x_{m+1-j}) = f_{m+1-j}, j = 1,2,...,k-1$$

$$P^*_{k,m}(x_{m+1}) = f(x_{m+1}, P_{m+1})$$

を満たす k-1 次の補間多項式とする．この多項式は，(4.3) で使った k 個の微係数のうち，最も旧い f_{m+1-k} を捨て，その代り $f(x_{m+1}, p_{m+1})$ を使っている．
このとき，修正子 c_{m+1} は

$$c_{m+1} = y_m + \int_{x_m}^{x_{m+1}} P^*_{k,m}(x)dx \qquad (4.6)$$

より計算する．この式は k 次の Adams-Moulton の公式と呼ばれる．$P^*_{k,m}(x)$ をニュートンの後退差分補間式の形で表せば，(4.6)は，前と同様にして

$$\left.\begin{aligned} c_{m+1} &= y_m + h\sum_{i=1}^{k} \gamma^*_{i-1} \nabla^{i-1} f^p_{m+1} \\ \text{ここで, } \nabla^0 f^p_{m+1} &= f^p_{m+1} \equiv f(x_{m+1}, p_{m+1}) \\ \nabla^i f^p_{m+1} &= \nabla^{i-1} f^p_{m+1} - \nabla^{i-1} f_m, i \geq 1 \\ \gamma^*_0 &= 1 \\ \gamma^*_i &= \frac{1}{i!}\int_0^1 (s-1)(s)..(s+i-2)ds, i \geq 1 \end{aligned}\right\} \qquad (4.7)$$

と書ける．特に $k = 1$ の場合は，$P^*_{1,m}(x) = f^p_{m+1}$ であるから

$$c_{m+1} = y_m + hf^p_{m+1} \qquad (4.8)$$

となり，これは，後退オイラー法 (backward Euler's method) である．
以上がアダムス法の原理である．

実際の計算では，点列 $\{x_{m+1-j}\}$ はキザミ幅制御のため不等間隔に分布しているため (4.5)，(4.7) は使わず，その代り $P_{k,m}(x)$，$P^*_{k,m}(x)$ を，変形差分商に基づく形で表し，それを積分した公式を 用いる（後述）．ところで，修正子 c_{m+1} の誤差推定は次のような考えに基づいてなされる．
$P^*_{k,m}(x)$ を形成するとき，f_{m+1-k} を使わなかったが，もしこれを使って，次数が1次だけ高い補間多項式 $P^*_{k+1,m}(x)$ を用いて積分すれば，c_{m+1} よりも1次高い次数の修正子（仮に $c_{m+1}(k+1)$ で表す）が得られる．誤差解析によれば，

$$E_{m+1} \equiv c_{m+1}(k+1) - c_{m+1} \qquad (4.9)$$

は，c_{m+1} の局所誤差の推定値として使えることが証明される．したがって，本サブルーチンでは，もし E_{m+1} が，許容誤差以内であれば，c_{m+1} は，要求精度を満たしていると見なし，更に，x_{m+1} における最終的な解 y_{m+1} としては，c_{m+1} よりも1次高い $c_{m+1}(k+1)$ を採用する．
これ以降，c_{m+1} を$y_{m+1}(k)$で，$c_{m+1}(k+1)$ を y_{m+1}で表すことにする．
x_{m+1}における解を求める手順をまとめると次のようになる．

I. 予測 (Prediction) ... p_{m+1}
II. 評価 (Evaluation) ... $f(x_{m+1},p_{m+1})$
III. 修正 (Correction) ... y_{m+1}
IV. 評価 (Evaluation)... $f(x_{m+1},y_{m+1})$

この手順を称して，PECE 法と呼ぶこともある．

② 変形差分商に基づくアダムス法

点列 $\{x_{m+1-j}\}$ は一般に不等間隔に分布するので，このようなときの予測子 (4.4)，修正子(4.6) の計算の具体化について述べる．$P_{k,m}(x)$ は差分商を使って

$$\begin{aligned} P_{k,m}(x) = f[x_m] &+ (x-x_m)f[x_m, x_{m-1}] + ... \\ &+ (x-x_m)(x-x_{m-1}).. \\ &(x-x_{m+2-k})f[x_m, x_{m-1}, ..., x_{m+1-k}] \end{aligned} \qquad (4.10)$$

と表せる．ただし，

$$f[x_m] = f_m$$

$$f[x_m, x_{m-1}, ..., x_{m-j}] = \frac{f[x_{m-1}, ..., x_{m-j}] - f[x_m, ..., x_{m+1-j}]}{x_m - x_{m-j}}$$

$$, j = 1,2,...,k-1$$

である．(4.10)の表現に基づく積分公式では，点列 $\{x_{m+1-j}\}$ と差分商をプログラムの中で使うことになるが，本サブルーチンでは，
$\{x_{m+1-j}\}$ の代りに，キザミ幅の列 $\{h_{m+1-j}\}$ を，また差分商の代りに以下に示す変形差分商を使う

公式を用いる．その公式は，キザミ幅が一定のとき，先に示した (4.5) に帰着されるものである．準備としていくつかの記号を導入する．

$$
\begin{aligned}
&h_i = x_i - x_{i-1}\\
&s = (x - x_m)/h_{m+1}\\
&\Psi_i(m+1) = h_{m+1} + h_m + \ldots + h_{m+2-i}, i = 1,2,\ldots\\
&\alpha_i(m+1) = h_{m+1}/\Psi_i(m+1), i = 1,2,\\
&\beta_1(m+1) = 1\\
&\beta_i(m+1) = \frac{\Psi_1(m+1)\Psi_2(m+1)\ldots\Psi_{i-1}(m+1)}{\Psi_1(m)\Psi_2(m)\ldots\Psi_{i-1}(m)}, i = 2,3,\ldots\\
&\Phi_1(m) = f[x_m] = f_m\\
&\Phi_i(m) = \Psi_1(m)\Psi_2(m)\ldots\Psi_{i-1}(m) f[x_m, x_{m-1}, \ldots, x_{m+1-i}]\\
&\qquad\qquad i = 2,3,\ldots
\end{aligned}
\tag{4.11}
$$

ここでは，$\Phi_i(m)$を変形差分商と呼ぶ．もしキザミ幅が一定のとき $\Psi_i(m+1) = ih, \alpha_i(m+1) = 1/i$ ，$\beta_i(m+1) = 1$ であるから $\Phi_i(m) = \nabla^{i-1} f_m$ となる．
このとき (4.10) の一般項は

$$
\begin{aligned}
&(x - x_m)(x - x_{-1})\ldots(x - x_{m+2-i}) f[x_m, x_{m-1}, \ldots, x_{m+1-i}]\\
&= \left(\frac{sh_{m+1}}{\Psi_1(m+1)}\right)\cdot\left(\frac{sh_{m+1} + \Psi_1(m)}{\Psi_2(m+1)}\right)\\
&\quad \left(\frac{sh_{m+1} + \Psi_{i-2}(m)}{\Psi_{i-1}(m+1)}\right)\beta_i(m+1)\Phi_i(m)
\end{aligned}
\tag{4.12}
$$

と表せる．更に，この式の右辺を簡略化するため，
$\Phi_i^*(m) = \beta_i(m+1)\Phi_i(m)$
及び

$$
c_{i,m}(s) = \begin{cases}
1 & ,i = 1\\
\dfrac{sh_{m+1}}{\Psi_1(m+1)} = s & ,i = 2\\
\left(\dfrac{sh_{m+1}}{\Psi_1(m+1)}\right)\cdot\left(\dfrac{sh_{m+1} + \Psi_1(m)}{\Psi_2(m+1)}\right)\ldots & \\
\quad \left(\dfrac{sh_{m+1} + \Psi_{i-2}(m)}{\Psi_{i-1}(m+1)}\right), i = 3,4,\ldots &
\end{cases}
$$

を導入すれば，(4.12) は $c_{i,m}(s)\Phi_i^*(m)$ と表せ，ゆえに，

$$
P_{k,m}(x) = \sum_{i=1}^{k} c_{i,m}(s)\Phi_i^*(m) \tag{4.13}
$$

を得る．ここで，$c_{i,m}(s)$ は s についての i –1 次の多項式であり，点列 $\{x_{m+1-i}\}$ の分布だけに依存して決まる．(4.13) を (4.4) に代入し，積分変数 x を s に変換すれば，

$$
p_{m+1} = y_m + h_{m+1}\sum_{i=1}^{k}\left(\int_0^1 c_{i,m}(s)ds\right)\Phi_i^*(m) \tag{4.14}
$$

を得る．この式は，キザミ幅が一定の場合の (4.5) に対応する．$c_{i,m}(s)$ の積分は部分積分を繰り返し適用して求められ，その結果は次のようになる．$i \geq 1$, $q \geq 1$として数列 $\{g_{i,q}\}$ を (4.15) より作る．

$$
\begin{aligned}
&g_{1,q} = 1/q,\ g_{2,q} = 1/\left(q(q+1)\right)\\
&g_{i,q} = g_{i-1,q} - \alpha_{i-1}(m+1)g_{i-1,q+1},\ i \geq 3
\end{aligned}
\tag{4.15}
$$

このとき，

$$
g_{i,1} = \int_0^1 c_{i,m}(s)ds
$$

が成り立つ．(4.15) は $g_{i,q}$ の三角テーブルを形成するが，その一例として，キザミ幅一定の場合を下に示す．

$i \longrightarrow$

q ↓	1	2	3	4	…
1	1	1/2	5/12	3/8	
2	1/2	1/6	1/8		
3	1/3	1/12			
4	1/4				
…					

この表の第1行に並ぶ数値が $\{g_{i,1}\}$ である．
したがって (4.14) は

$$
p_{m+1} = y_m + h_{m+1}\sum_{i=1}^{k} g_{i,1}\Phi_i^*(m) \tag{4.16}
$$

となる．
次に修正子の計算について述べる．修正子は，前にも述べたとおり予測子よりも1次高い k + 1次の公式を用いる．すなわち，修正子y_{m+1}は

$$
y_{m+1} = y_m + \int_{x_m}^{x_{m+1}} P_{k+1,m}^*(x)dx \tag{4.17}
$$

に基づく．ここで $p_{k+1,m}^*(x)$ は，$P_{k,m}(x)$ が満たす補間条件に，もう一つの条件

$$
P_{k+1,m}^*(x_{m+1}) = f_{m+1}^p = f(x_{m+1}, p_{m+1})
$$

を加えてできる補間多項式であり，差分商を用いれば

$$P^{*}_{k+1,m}(x)=P_{k,m}(x)+(x-x_m)(x-x_{m-1})..(x-x_{m+1-k})\cdot f^{\,p}[x_{m+1},...,x_{m+1-k}] \quad (4.18)$$

と書ける．添字のpは，差分商を表す式中で，$f_{m+1}=f(x_{m+1},y_{m+1})$ の代りに f^{p}_{m+1} を使うことを意味する．(4.18) を (4.17) へ代入すれば，

$$y_{m+1}=p_{m+1}+h_{m+1}g_{k+1,1}\Phi^{p}_{k+1}(m+1) \quad (4.19)$$

というように，予測子 p_{m+1} に補正項を加えるという形で表せる．ここでも添字 p は Φ_{k+1}(m+1) 中の差分商の部分で $f_{m+1}=f(x_{m+1},y_{m+1})$ の代りに f^{p}_{m+1} を使うことを意味する．k 次の修正子 $y_{m+1}(k)$ も同様にして，

$$y_{m+1}(k)=p_{m+1}+h_{m+1}g_{k,1}\Phi^{p}_{k+1}(m+1) \quad (4.20)$$

と計算できる．

③ 独立変数の任意の点における解

キザミ幅は，誤差の許す限り大きくとられるので，一般に，解を求めたい点 x_e が

$$x_m < x_e \le x_{m+1} \quad (4.21)$$

となる状況が起こる．x_e における解を求める一つの方法として，キザミ幅を x_e に命中するように決めることもできるが，本サブルーチンでは，特別の事情（例えば，x_e を超えた領域で$f(x,y)$ が定義されない）がない限り，最適なキザミ幅で x_e を通過し，x_{m+1} における解が求まったあと，x_e における解を以下に示すように計算する．

$P_{k+1,m+1}(x)$ を $P_{k+1,m+1}(x_{m+2-j})=f_{m+2-j},\ j=1,2,...,k+1$ を満たす k 次の補間多項式として，点 x_e における解 y_e を，

$$y_e=y_{m+1}+\int_{x_{m+1}}^{x_e}P_{k+1,m+1}(x)dx \quad (4.22)$$

より計算する．x_{m+1} における解が要求精度を満たしているならば，y_e も同様に要求精度を満たすことが証明されている．

(4.22) を具体的に計算するには，$P_{k+1,m+1}(x)$ を変形差分商を使った形で表現する．まず，

$$h_I=x_e-x_{m+1},\ s=(x-x_{m+1})/h_I$$

とする．$P_{k+1,m+1}(x)$ は，

$$P_{k+1,m+1}(x)=f[x_{m+1}]+(x-x_{m+1})f[x_{m+1},x_m]+...+(x-x_{m+1})..(x-x_{m+2-k})f[x_{m+1},...,x_{m+1-k}]$$

と書けるが，その一般項は，

$$(x-x_{m+1})(x-x_m)..(x-x_{m+3-i})f[x_{m+1},x_m,...,x_{m+2-i}]=\left(\frac{sh_I}{\Psi_1(m+1)}\right)\cdot\left(\frac{sh_I+\Psi_1(m+1)}{\Psi_2(m+1)}\right)...\left(\frac{sh_I+\Psi_{i-2}(m+1)}{\Psi_{i-1}(m+1)}\right)\Phi_i(m+1)$$

となる．$P_{k+1,m+1}(x)$ をこのように変形差分商を使って表し，それを (4.22) へ代入すれば，

$$y_e=y_{m+1}+h_I\sum_{i=1}^{k+1}g^{I}_{i,1}\Phi_i(m+1) \quad (4.23)$$

を得る．ここで，$g^{I}_{i,1}$ は，次の漸化式より作り出される．

$$g^{I}_{1,q}=1/q$$
$$g^{I}_{i,q}=\xi_{i-1}g^{I}_{i-1,q}-\eta_{i-1}g^{I}_{i-1,q+1},\ i\ge 2 \quad (4.24)$$

ただし，

$$\xi_i=\begin{cases}h_I/\Psi_1(m+1), & i=1\\ \dfrac{h_I+\Psi_{i-1}(m+1)}{\Psi_i(m+1)}, & i\ge 2\end{cases}$$

$$\eta_i=h_I/\Psi_i(m+1),\ i\ge 1$$

④ 解の受入れ

k 次の修正子 $y_{m+1}(k)$ の局所誤差を $le_{m+1}(k)$ とすると，それは，$k+1$ 次の修正子 y_{m+1} と$y_{m+1}(k)$ との差で見積ることができる．すなわち (4.19) と (4.20) より，

$$|le_{m+1}(k)|\approx|h_{m+1}(g_{k+1,1}-g_{k,1})\Phi^{p}_{k+1}(m+1)| \quad (4.25)$$

である．この右辺を ERR と 定義すると，許容誤差 ε に対して，

$$\mathrm{ERR}\le\varepsilon \quad (4.26)$$

が成り立てば，x_{m+1} における解として y_{m+1} を採用し，$f(x_{m+1},y_{m+1})$ を評価して，そのステップは終了する．

⑤ 次数の制御

次数の制御とは，各ステップでアダムス法の公式次数を選ぶことであり，このことは $f(x,y)$ を近似する補間多項式の次数を選ぶことにほかならない．この選択はキザミ幅の選択に先立って行われる．

本サブルーチンでは，解の挙動に応じて1次から12次までの公式を使い分けるが，各ステップにおける次数は，前のステップの次数を k とすると，$k-1, k, k+1$ のいずれかより選ばれる．

今，k 次のアダムス法により x_{m+1} における解 y_{m+1} が受け入れられ，次のステップのための次数を選ぶ場面を考える．x_{m+2} における局所誤差は，あとで選ばれるキザミ幅 h_{m+2} と，x_{m+2} における微係数に依存して決まるものであり，次数選択の段階ではそのどちらも未知であるから誤差を正確に知ることはできない．そこで，それ

までに利用できる数値だけを使って，k –1, k, 及び k + 1 の各次数の，x_{m+2} における局所誤差をそれぞれ以下のERKM1, ERK, , ERKP1 により予測する．

$$\left.\begin{aligned}\mathrm{ERKM1}&=\left|h\gamma^{*}_{k-1}\sigma_{k}(m+1)\Phi^{p}_{k}(m+1)\right|\\ \mathrm{ERK}&=\left|h\gamma^{*}_{k}\sigma_{k+1}(m+1)\Phi^{p}_{k+1}(m+1)\right|\\ \mathrm{ERKP1}&=\left|h\gamma^{*}_{k+1}\Phi_{k+2}(m+1)\right|\end{aligned}\right\} \qquad (4.27)$$

ただし，

$$\sigma_{i}(m+1)=\begin{cases}1, i=1\\ \dfrac{h\cdot 2h\ldots(i-1)h}{\Psi_{1}(m+1)\Psi_{2}(m+1)\ldots\Psi_{i-1}(m+1)}\\ \qquad\qquad ,i=2,3,4,\ldots\end{cases}$$

$h=h_{m+1}=x_{m+1}-x_{m}$

γ^{*}_{i} は (4.7) で定義された係数である．

ERKP1 は，過去の k + 1 回のステップがすべて一定キザミ幅 h で進められた場合だけ見積るものとする．その理由は，もしキザミ幅が一定でないと，ERKP1 の信頼性が悪くなるからである．

(4.27) のほかに，k -2 次の局所誤差も考慮され，以下の ERKM2 のように見積る．

$$\mathrm{ERKM2}=\left|h\gamma^{*}_{k-2}\sigma_{k-1}(m+1)\Phi^{p}_{k-1}(m+1)\right|$$

さて，以上のように導入した諸量を使って次数は次のルールに従って制御する．

a. 以下のいずれかが満たされたとき次数を k-1 次に下げる．
   - k > 2 かつ max (ERKM1, ERKM2) ≤ ERK
   - k = 2 かつ ERKM1 ≤ 0.5ERK
   - 過去の k + 1 回のステップが一定キザミ幅で進められ，かつ
     ERKM1 ≤ min (ERK, ERKP1)

b. 過去のk + 1 回のステップが一定キザミ幅で進められ，以下のいずれかが満たされたとき次数を k + 1 次に上げる．
   - 1 < k < 12 かつ
     ERKP1 < ERK< max (ERKM1, ERKM2)
   - k = 1 かつ ERKP1 < 0.5ERK

c. 上記の a. , b. のいずれも満たされなかったときは，次数は変えない．

各ステップにおいて，上記のa., b., c. の順に検査される．

⑥ キザミ幅の制御

キザミ幅の制御は，各ステップで次数の選択がなされたあと行う．今，次のステップへ進むときの次数 k が決定されたとする．また，最後に使ったキザミ幅を h とする．このとき，次のステップをキザミ幅 $r\cdot h$ で進めたとすると，局所誤差の程度は

$$r^{k+1}\,\mathrm{ERK} \qquad (4.28)$$

と予測することができる．したがって，倍率 r は，理論的には (4.28) が誤差許容値 ε を超えない範囲で，できる限り大きくとれるが，安全性を考え，ε の代りに 0.5ε を採用し，

$$r^{k+1}\,\mathrm{ERK}\le 0.5\,\varepsilon \qquad (4.29)$$

を満たす範囲で r を決定する．

ところで，キザミ幅の変化は予測子及び修正子の公式の係数に影響を及ぼし，キザミ幅の大きな変動は，公式の誤差伝播特性を悪くする可能性がある．本サブルーチンでは，このことを考慮に入れ，キザミ幅を大きくする場合は，せいぜい 2倍に，小さくする場合はせいぜい 1/2 倍に止めるようにしている．キザミ幅の制御は，具体的に次のように行う．

a. キザミ幅を大きくする場合．

   $2^{k+1}\,\mathrm{ERK}\le 0.5\,\varepsilon$

   ならば，キザミ幅を2倍にする．

b. キザミ幅を小さくする場合．

   $\mathrm{ERK}>0.5\,\varepsilon$

   ならば，r = $(0.5\,\varepsilon/\ \mathrm{ERK})^{1/(k+1)}$ として，実際の倍率を以下の r’ とする．

   $r' = \min(0.9, \max(1/2, r))$

c. 上記 a. , b. が共に満たされなかった場合は，キザミ幅は変えない．

⑦ 解が受け入れられなかった場合

解の受入れの判定 (4.26) が満たされなかったとき，そのステップはやり直される．この場合，次数は，⑤のa. の検査を行うことにより選ぶ．すなわち，次数は，変えないか下げるかのどちらかである．一方，キザミ幅については無条件に1/2倍する．

もし，連続して3回やり直し，なおも (4.26) が満たされなかった場合は，1次のアダムス法，すなわちオイラー法に切り換え，(4.26) が満たされるまでキザミ幅を半減させる．

⑧ 初期手続き

与えられた初期値問題を解く最初の段階では，1次のアダムス法が適用されるが，ここでは，初期キザミ幅の決め方，及び，次数とキザミ幅の特別な増し型について述べる．

初期キザミ幅を h_1 としたとき，オイラー法による点 x_1 (= x_0 + h_1) における局所誤差を

$$h_1^2 \left| f\left(x_0, y_0\right) \right|$$

で予測し，これが，0.5 ε を超えないように h_1 を選べるが，$f(x_0,y_0)$ が 0 の場合，及び，その他の安全性を考え，

$$h_1 = \min\left(0.25 \left| \frac{0.5\varepsilon}{f\left(x_0, y_0\right)} \right|^{1/2}, H \right) \qquad (4.30)$$

ただし，

$$H = \max\left(4u\left|x_0\right|, \left|x_e - x_0\right|\right)$$

u: 丸め誤差の単位

としている．

初期段階においては，次数とキザミ幅の制御は，特別に次のように行う．すなわち，ステップが進むにつれ，次数を常に1次ずつ上げ，キザミ幅を常に2倍してゆく．このように次数とキザミ幅を常に増やす理由は，次数を早めに上げた方が効果的であり，また，ε が小さいとき (4.30) の h_1 は保守的になりやすいからである．

この特別の制御が終了するのは，あるステップで，解の受入れ判定が不合格となるか，次数が12次となるか，あるいは先に述べた⑤の条件 a. が満たされるかのいづれかが起きたときである．そのあとの次数とキザミ幅は，前述の一般的なルールに従って制御する．

⑨ 連立微分方程式への拡張

連立微分方程式の場合は，解の各成分に今までの議論を適用すればよい．

ただし，ERR をはじめ ERK, ERKM1, ERKM2 及び ERKP1 の各種誤差推定量は各成分に対して定義する代りに，いわばベクトルのノルムとして以下のように定義する．すなわち，利用者の指定した EPSA, EPSR, より，

$$\left.\begin{aligned} \mathrm{EPS} &= \max\left(\mathrm{EPSA}, \mathrm{EPSR}\right) \\ \mathrm{W}\left(I\right) &= \left(\left|\mathrm{Y}\left(I\right)\right| \mathrm{EPSR} + \mathrm{EPSA}\right) / \mathrm{EPS} \end{aligned}\right\} \qquad (4.31)$$

を導入し，ERRを

$$\mathrm{ERR} = \left(\sum_{I=1}^{N} \left(\frac{le\left(I\right)}{W\left(I\right)} \right)^2 \right)^{1/2} \qquad (4.32)$$

と定義する．ここで，Y(I), l_e(I) は第 I 成分の解と，その局所誤差である．ERK , ERKM1 なども同様なノルムで定義する．

解の受入れのための判定は，

$$\mathrm{ERR} \le \mathrm{EPS}$$

で行う．これが満たされると，(4.31), (4.32) より，

$$le\left(I\right) \le \left|Y\left(I\right)\right| \mathrm{EPSR} + \mathrm{EPSA}$$
$$, I = 1,2,..., \mathrm{N}$$

が成り立つことが分かる．

なお，本サブルーチンは，L.F. Shampine と M.K. Gordon のよるプログラム（文献 [71]）に基づいている．

H11-20-0151 ODGE, DODGE

スティフ連立1階常微分方程式（ギア法）
CALL ODGE (X, Y, FUN, N, XEND, ISW, EPSV, EPSR, MF, H, JAC, VW, IVW, ICON)

(1) 機能

連立1階常微分方程式の初期値問題

$$\left.\begin{aligned} y_1' &= f_1(x, y_1, y_2, ..., y_N),\ y_1(x_0) = y_{10} \\ y_2' &= f_2(x, y_1, y_2, ..., y_N),\ y_2(x_0) = y_{20} \\ &\cdots \qquad\qquad \cdots \\ y_N' &= f_N(x, y_1, y_2, ..., y_N),\ y_N(x_0) = y_{N0} \end{aligned}\right\} \qquad (1.1)$$

における関数系 $f_1, f_2, ..., f_N$ と，初期値 $x_0, y_{10}, y_{20}, ..., y_{N0}$, 及び x の最終値 x_e が与えられたとき，ギア法又はアダムス法により，

$x_m = x_0 + \sum_{j=1}^{m} h_j ; m = 1,2,...,e$ における解 ($y_{1m}, y_{2m}, ..., y_{Nm}$) を求める（図 ODGE-1 参照）．ただし，キザミ h_j は，求めるべき解が要求精度を満たすように制御される．

ギア (Gear) 法はスティフな方程式に適し，アダムス法は非スティフな方程式に適している．利用者は，スティフ性の有無に応じて二つの手法を使い分けることができる．

本サブルーチンは解の出力方法，すなわち利用者プログラムに戻るタイミングとして以下の2通りを用意しており，利用者は目的に応じて選ぶことができる．

a. 最終値出力 ... 最終値 x_e における解が求まったときはじめて戻る．

b. ステップ出力 ... 途中の解，すなわち，$x_1, x_2, ...$ における解が求まるたびに戻る．

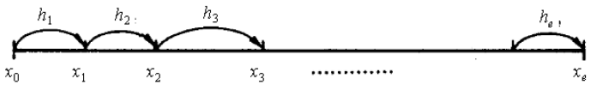


図 ODGE-1 解を求める点 x_m ($x_0 < x_e$ の場合)

(2) パラメタ

X ····· **入力**．初期値 x_0.
出力．最終値 x_e．ただし，ステップ出力が指定されている場合は，1ステップ前進したときの独立変数の値 x_m .

Y ····· **入力**．初期値 $y_{10}, y_{20}, ..., y_{N0}$ を Y(1) = y_{10}, Y(2) = y_{20}, ..., Y(N) = y_{N0} の順に与える．大きさ N の1次元配列.
出力．最終値 x_e における解ベクトル．ただし，ステップ出力が指定されている場合は，1ステップ前進したときの $x = x_m$ における解ベクトル.

FUN ····· **入力**．(1.1) における $f_i, i = 1,2,...,N$ を計算するサブルーチン副プログラム名.
［副プログラムの用意のし方］
SUBROUTINE FUN (X, Y,YP)
ここで
X ····· **入力**．独立変数 x.
Y ····· **入力**．Y(1) = y_1, Y(2) = y_2, ..., Y(N) = y_N なる対応を持つ大きさNの1次元配列.
YP ····· **出力**．
YP(1) = $f_1(x, y_1, y_2, ..., y_N)$,
YP(2) = $f_2(x, y_1, y_2, ..., y_N)$,...,
YP(N) = $f_N(x, y_1, y_2, ..., y_N)$ なる対応を持つ大きさ N の1次元配列.

N ····· **入力**．連立1階常微分方程式の元数.

XEND ····· **入力**．独立変数の最終値 x_e.

ISW ····· **入力**．積分する際の条件を指定する．
ISW は10進4桁の非負の整数であり，いま，それを
ISW= $1000d_4 + 100d_3 + 10d_2 + d_1$
と表すと，各 d_i は次のように指定する．

d_1 ： 初回呼出しであるか否かを指定する．
0 ..... 初回呼出しである．
1 ..... 継続呼出しである．
初回呼出しとは，与えられた微分方程式に対して初めて本サブルーチンを呼び出すときを言う．

d_2 ： 解の出力方法を指定する．
0 ..... 最終値出力.
1 ..... ステップ出力.

d_3 ： 最終値 x_e を超えた点で，関数系 $f_1, f_2, ..., f_N$ が計算できるか否かを指定する．
0 ..... 計算できる.
1 ..... 計算できない.
$d_3 = 1$ と指定する状況は，微係数が x_e を超えた点では定義されないか，あるいは，x_e を超えた近くに不連続点を持つ場合である．このような状況でないとき，むやみに $d_3 = 1$ と指定すると計算効率を著しく低下させるので注意すること.

d_4 ： 解法の途中で，手法を切り換えたり（パラメタ MF），要求精度（パラメタ EPSV, EPSR），若しくは N を変えたりするとき意味を持つ.
0 .... 変えなかった.
1 ..... 変えた.
$d_4 = 1$と指定する状況とタイミングについては，使用上の注意⑥，⑦参照.

出力．x_e における解，若しくはステップごとの解を求めて利用者プログラムに戻るとき，d_1, d_3, d_4 は次のように変更される．

d_1 ： $d_1 = 1$ とする．本サブルーチンを繰り返し呼び出すときは d_1 を1のままにしておくこと．
利用者が再び $d_1 = 0$ と設定し直すときは，別の方程式を解き始めるときである．

d_3 ： 入力時に $d_3 = 1$ であった場合は，x_e における解が求まった時点で $d_3 = 0$ とする．

d_4 ： 入力時に $d_4 = 1$ であった場合は，$d_4 = 0$ とする．

EPSV ····· **入力**．絶対誤差の上限．解の成分ごとに与える．大きさ N の1次元配列．
EPSV(L) ≥ 0.0, L = 1,2, ..., Nであること．
出力．入力時に EPSV が小さすぎた場合は適当な大きさに変更された値．
（使用上の注意④参照）．

EPSR ····· **入力**．相対誤差の上限 (≥ 0.0)
出力．入力時に EPSR が小さすぎた場合は適当な大きさに変更された値．
（使用上の注意④参照）．

MF ····· **入力**．ギア法とアダムス法のいずれを使うか，及び，修正子の反復法をいずれにするかを指定する．MF は10進2桁の非負の整数であり，いま，それを，
MF = 10*METH + ITER
と表すと，METH , ITER は次のように指定する．

METH... 手法を使用する．
1...ギア法を使用する．方程式がスティフな場合に適している．
2...アダムス法を使用する．方程式が非スティフな場合に適している．

ITER ... 修正子の反復法を指定する．
0...解析的ヤコビアン行列 $\boldsymbol{J} = \left(\partial f_i / \partial y_j\right)$ を使ったニュートン法．このとき利用者は，ヤコビアン行列を計算するためのサブルーチンを用意する必要がある．
（パラメタ JAC 参照）
1...差分法により内部的に近似したヤコビアン行列を使ったニュートン法．
2...上記1と同じであるが，この場合，ヤコビアン行列を対角行列で近似する．
3...ヤコビアン行列を使わない反復法（関数反復）．

方程式がスティフなら ITER = 0, 1 又は 2 とする．このうち ITER = 0 が最も望ましいが，解析的ヤコビアン行列が用意できないなら，ITER = 1 とすればよい．ITER = 2 は，ヤコビアン行列が優対角であることが分っている場合に有用である．
方程式が非スティフなら ITER = 3 とする．

H····· **入力**．初回呼出しのとき，試行すべき初期キザミ幅を指定する (H≠0)．符号は $x_e - x_0$ と同じであること．|H| の標準的な値は

$$|\mathrm{H}| = \min\left(10^{-5}, \max\left(10^{-4}|x_0|, |x_e - x_0|\right)\right)$$

である．H は要求精度を満たすべく制御される．
出力．最後に使われたキザミ幅．

JAC····· **入力**．ヤコビアン行列

$$\mathrm{J} = \begin{bmatrix} \frac{\partial f_1}{\partial y_1} & \frac{\partial f_1}{\partial y_2} & \cdots & \frac{\partial f_1}{\partial y_N} \\ \frac{\partial f_2}{\partial y_1} & \frac{\partial f_2}{\partial y_2} & \cdots & \frac{\partial f_2}{\partial y_N} \\ \cdots & \cdots & \cdots & \\ \frac{\partial f_N}{\partial y_1} & \frac{\partial f_N}{\partial y_2} & \cdots & \frac{\partial f_N}{\partial y_N} \end{bmatrix}$$

を計算するサブルーチン副プログラム名．
［副プログラムの用意のし方］
SUBROUTINE JAC (X, Y, PD, K)
ここで，

X····· **入力**．独立変数 x.

Y····· **入力**．Y(1) = y_1, Y(2) =y_2, ..., Y(N) = y_N なる対応を持つ大きさ N の1次元配列．

PD····· **出力**．ヤコビアン行列．
PD (K, K) なる2次元配列．

$$\mathrm{PD}(i, j) = \frac{\partial f_i}{\partial y_j}$$

$1 \le i \le N$, $1 \le j \le N$

K····· **入力**．2次元配列 PD の整合寸法．
（使用例参照）．

利用者が ITER を 0 以外の値に指定した場合でも，JAC は以下のようにダミーのサブルーチンとして用意しておく必要がある．

```
SUBROUTINE JAC (X, Y, PD, K)
RETURN
END
```

VW····· **作業領域**．大きさ N(N +17) + 70 の 1次元配列．繰り返し本サブルーチンを呼び出すときは内容を変えてはならない．

IVW····· **作業領域**．大きさ N + 25 の1次元配列．繰り返し本サブルーチンを呼び出すときは内容を変えてはならない．

ICON····· **出力**．コンディションコード
表 ODGE-1 参照．

表 ODGE-1 コンディションコード

コード	意味	処理内容
0	（ステップ出力の場合だけ）	続けて呼び出すこと

	1ステップ積分した.	ができる.
10	XEND における解を求めた.	XEND を変えて続けて呼び出すことができる.
10000	EPSR 及び EPSV(L): L = 1, 2, .., N が演算精度に比べて小さすぎた.	EPSR 及び EPSV(L): L = 1, 2, ..., N を適当に大きくした. （大きくされた EPSR 及び EPSV(L) は確認しておくのがよい. ）
15000	初期キザミ幅の 10^{-10} 倍のキザミ幅を使っても要求精度が得られなかった.	
16000	初期キザミ幅の 10^{-10} 倍のキザミ幅を使っても修正子を求めるための反復計算が収束しなかった.	パラメタ MF により指定されている手法，若しくは修正子反復法が与えられた方程式にとって適していない可能性が強い. MF を変えて呼び出すとよい.
30000	次のいずれかであった. ① N≤ 0 ② X = XEND ③ ISW の指定に誤りがある. ④ EPSR < 0 又は EPSV(I) < 0 なる I がある. ⑤ (XEND – X)*H≤0 ⑥ IVW の内容が変えられている（ただし2回目以降の呼出し時）.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL ... MGSSL, AMACH, USDE, UNIT2, USTE2, USETC, USETP, USOL, UADJU, UDEC

② FORTRAN 基本関数 ... MOD, FLOAT, AMAX1, AMIN1, ABS, SORT

b. 注意

① 本サブルーチンは，方程式がスティフであるか，あるいは，解こうとしている区間で非スティフな状態からスティフな状態に変化する場合に有効に利用することができる．しかし，もし方程式が非スティフならば，サブルーチン ODAM 又は ODRK1 を使用した方が効率の面で良い.

② パラメタ FUN 及び JAC に対応する副プログラムの名前は，本サブルーチンを呼び出す側のプログラムで，EXTERNAL 文により宣言しなければならない.

③ ICON の判定

利用者が ISW の10の位を0とすることにより最終値出力を指定した場合は，ICON が 10 と出力されたときに限り x_e における解を受け取ることができる.

一方，ISW の10の位を1とすることによりステップ出力を指定した場合は，ICON が 0 と出力されたときに解を受け取ることができる．この過程を続けて ICON が 10 と出力されたときは，x_e における解が求まったことを意味する.

④ パラメタ EPSV と EPSR

微分方程式の解の成分を Y(L)，誤差（正確には局所誤差）を l_e(L) とすると，本サブルーチンは，L = 1,2,...,Nに対して

$$\left|l_e(L)\right| \le \mathrm{EPSR}*\left|\mathrm{Y(L)}\right| + \mathrm{EPSV(L)} \qquad (3.1)$$

を満たすように，誤差を制御する．この式において，EPSV(L) = 0 とすれば，相対誤差判定となり，EPSR = 0 とすれば，解の各成分ごとに判定値が異なる絶対誤差判定となる．利用者は，EPSV(L) 及び EPSR を指定するに当り，以下のことに注意する必要がある.

a. 解の大きさが非常に変化する問題では，相対誤差判定が望ましい.

b. 解の大きさがさほど変化しないか，あるいは，小さな解に対しては関心が無い場合には，絶対誤差判定としてもよい.

c. しかし，安全で無駄のない判定は，
EPSV(L)≠0, L=1, 2, ...,N
EPSR≠0
とする混合判定である．この場合，大きい解に対しては実質的に相対判定となり，小さい解に対しては実質的に絶対判定となる．方程式がスティフなときは，一般に，解の各成分の間で大きさがまちまちになるので，EPSV(L) もそれに応じて成分ごとに異なる値を与えた方がよい.

なお，入力時に EPSR = 0.0 でかつ，EPSV(L) のある要素が0.0であったときは，EPSR = 16u と変更する．ただし u は丸め誤差の単位である.

⑤ パラメタ XEND

独立変数の連続したいくつかの値に対して解を求めたい場合は，順番に XEND を変えながら本サブルーチンを呼び出せばよい．サブルーチンは，このように繰返し呼ばれることを想定して，利用者プログラムに戻るときはすべてのパラメタを次の呼出し時に必要な値に設定している．したがって利用者は，単に XEND を変えて呼び出すだけでよい.

⑥ 解法の途中におけるパラメタ MF の切換え

方程式が，解こうとしている区間で，非スティフな状態からスティフな状態へ変化する場合は，スティフ性が強くなってきた時点で，MF を 23 から 10 （あるいは 11, 12）へ切り換えるのがよい．そのときの手順は以下のとおりである．

(a) ISW の 1000 の位，つまり d_4 を 1 とする．これは，ISW = ISW + 1000 なる代入文で行える．

(b) MF を切り換える．

(c) XEND を次に望む出力点に進めて，本サブルーチンを呼び出す．

本サブルーチンは，(c) のあと利用者プログラムに戻るとき，利用者の要求どおりに手法を切り換えたことを通知する意味で，

ISW= MOD (ISW, 1000)

により，ISW の1000の位をクリアする．

ただし，XEND における解を，手法を切り換えるまでもなく少ない手間で計算できるときは，その方法で計算し，利用者プログラムに戻る（この場合，ISWの1000の位はクリアされず，MF の値も呼び出す前と同じである）．本サブルーチンは，XEND が，手法を切り換えるのに十分な位置まで進められたとき，はじめて手法を切り換える．

⑦ 解法の途中におけるパラメタ N, EPSV, 及び EPSR の変更

利用者は解法の途中で，パラメタ N, EPSV, 及び EPSR の値を変更することができる．ただし，N の変更には，以下に述べるように，解の挙動に関する十分な知識を必要とする．ここでは，N の変更の意義とその方法について述べる．

方程式がスティフである場合，積分が進むにつれ，解の成分のいくつかが他の成分に比べてさほど変化しないか，あるいは，無視できるほど小さくなることがある．このとき，もし利用者が，それらの成分について興味がないならば，以降それらの成分を定数と見なし，残りの成分だけを積分することにすれば，多少厳密さを失うが，少ない計算量で必要な解を得ることができる．パラメタ N の変更は，このように，積分する解の成分を減らすことを意味する．ただし，本サブルーチンは，成分を減らす場合，方程式 (1.1) の最後の成分のいくつかを減らす．したがって利用者は，必要に応じて成分の順序を並べ換えておく必要がある．

以上を前提として，N の縮小は次のように行う．

いま，本サブルーチンの初回呼出し時の N の値を N_0 とし，これを解法途中で N_c ($<N_0$) に変更したとすると N_0 元の連立微分方程式の最後の ($N_0 - N_c$) 個の方程式を削除し，縮小された N_c 個の方程式からなる系を解く．その際，除かれた方程式に対応する解の成分 Y(L); L = Nc + 1, N_c + 2, ..., N_0, は，縮小された系において一定値に保たれる．

利用者の用意するサブルーチン FUN, JAC においては，N_c 個の方程式についての微係数又はヤコビアン行列を計算するだけでよい．これらのサブルーチン内で N の変更を確認するには，パラメタ N を COMMON 文で宣言することにより，利用者の主プログラムと連絡を図ればよい．

以上が N の変更についてのあらましであるが，EPSV および EPSR も必要ならば任意な時点で変更することができる．

これらのパラメタの変更は⑥で述べたパラメタ MF の切換えと同じ手順で行うこと．

c. 使用例

連立1階常微分方程式

$$\begin{cases} y_1' = y_2 & , y_1(0) = 1 \\ y_2' = -11y_2 - 10y_1 & , y_2(0) = -1 \end{cases}$$

を区間 [0, 100] で解く．この方程式はスティフである（ヤコビアン行列の固有値は-1と-10である）．以下の例は，MF = 10, EPSR = 10^{-4}, EPSV(1) = EPSV(2) = 0 とし，解を

$x_j = 10^{-3+j}$, j = 1, 2, ..., 5

の各点で出力する．このため，パラメタ XEND に順次 x_j を与えて本サブルーチンを繰返し呼び出している．

```
C     **EXAMPLE**
      DIMENSION Y(2),EPSV(2),VW(110),
     *          IVW(30)
      EXTERNAL FUN,JAC
      X=0.0
      Y(1)=1.0
      Y(2)=-1.0
      N=2
      EPSR=1.0E-4
      EPSV(1)=0.0
      EPSV(2)=0.0
      MF=10
      ISW=0
      H=1.0E-5
C
      WRITE(6,600)
C
      XEND=1.0E-3
      DO 40 I=1,5
      XEND=XEND*10.0
   10 CALL ODGE(X,Y,FUN,N,XEND,ISW,EPSV,
     *          EPSR,MF,H,JAC,VW,IVW,ICON)
      IF(ICON.EQ.10) GO TO 30
      IF(ICON.EQ.10000.OR.ICON.EQ.15000)
     *   GO TO 20
      IF(ICON.EQ.16000) WRITE(6,630)
```

```
      STOP
   20 WRITE(6,620) EPSR,EPSV(1),EPSV(2)
      GO TO 10
   30 WRITE(6,610) X,Y(1),Y(2)
   40 CONTINUE
      STOP
  600 FORMAT('1',12X,'X',22X,'Y(1)',
     * 16X,'Y(2)'/)
  610 FORMAT(6X,E15.8,10X,2(E15.8,5X))
  620 FORMAT(10X,'TOLERANCE RESET',5X,
     * 'EPSR=',E12.5,5X,'EPSV(1)=',E12.5,
     *5X,'EPSV(2)=',E12.5)
  630 FORMAT(10X,'NO CONVERGENCE IN',
     * ' CORRECTOR ITERATION')
  640 FORMAT(10X,'INVALID INPUT')
      END

      SUBROUTINE FUN(X,Y,YP)
      DIMENSION Y(2),YP(2)
      YP(1)=Y(2)
      YP(2)=-11.0*Y(2)-10.0*Y(1)
      RETURN
      END

      SUBROUTINE JAC(X,Y,PD,K)
      DIMENSION Y(2),PD(K,K)
      PD(1,1)=0.0
      PD(1,2)=1.0
      PD(2,1)=-10.0
      PD(2,2)=-11.0
      RETURN
      END
```

(4) 手法概要

本サブルーチンは，キザミ幅制御及び次数制御付きのギア法とアダムス法を適用している．前者の手法は，スティフな方程式を，また，後者の手法は非スティフな方程式を解くためのものである．

二つの手法は共に多段法である．本サブルーチンは，過去の計算解の保存のしかたや，キザミ幅及び次数の制御のしかたを，二つの手法の間で共通化している．

以下では，まずギア法を説明し，次いでキザミ幅及び次数の制御について述べる．そのあと，アダムス法を適用するときの変更点について述べる．

暫くの間，単一の方程式

$$y'=f(x,y)\ ,\ y(x_0)=y_0$$

を考える．点 x_m における計算解（以後，単に解という）をy_m, 真の解を $y(x_m)$ で表す．また，$f(x_m, y_m)$ の値を f_m で表し，$f(x_m, y(x_m))$とは区別する．

① ギア法の原理

今，$y_0, y_1, ..., y_m$ が既知で

$$x_{m+1} = x_m + h_{m+1}$$

における解 y_{m+1} を求めることを考える．

ギア法の基本的な考え方は次のとおりである．

真の解 $y(x)$ に対する補間多項式は $\pi_{m+1}(x)$ を考え，これは (4.1) に示すように，$k + 1$ 個の条件を満たす k 次の多項式とする．

$$\left.\begin{aligned}&\pi_{m+1}(x_{m+1-j})= y_{m+1-j}\\ &,j=0,1,2,...,k\end{aligned}\right\} \tag{4.1}$$

(4.1) は，求めようとする解 y_{m+1} を未知パラメタとして含んでいる．このとき，y_{m+1} は，多項式 $\pi_{m+1}(x)$ の x_{m+1} における微係数が $f(x_{m+1}, y_{m+1})$ に等しくなるように，すなわち

$$\pi'_{m+1}(x_{m+1})= f(x_{m+1},y_{m+1}) \tag{4.2}$$

を満たすように決定する．

$\pi_{m+1}(x)$ をラグランジュの形式で表現し，そのあと，$x = x_{m+1}$ で微分して (4.2) へ代入すれば,

$$h_{m+1} f(x_{m+1},y_{m+1}) = -\sum_{j=0}^{k}\alpha_{m+1,j}\, y_{m+1-j} \tag{4.3}$$

を得る．ここで $\alpha_{m+1,j}$ は $\{x_{m+1-j}\}$ の分布に依存して決まる定数である．一般に $f(x_{m+1}, y_{m+1})$ はy_{m+1} について非線形であるから (4.3) は，y_{m+1} に関する非線形な方程式と見なすことができる．したがって y_{m+1} は，適当な初期値から出発し，反復法により求めることになる．その初期値は次のように選ぶ．多項式 $\pi_m(x)$ を (4.4) を満たす k 次の補間多項式とする．

$$\left.\begin{aligned}&\pi_m(x_{m+1-j})= y_{m+1-j},\ j=1,2,...k\\ &\pi'_m(x_m)= f(x_m,y_m)\end{aligned}\right\} \tag{4.4}$$

このとき,

$$p_{m+1} = \pi_m(x_{m+1}) \tag{4.5}$$

により得られる p_{m+1} は x_{m+1} における一つの解と見なせる．そこで，この p_{m+1} を，反復法の初期値とする．

以上がギア法の原理である．p_{m+1} を予測子，反復法により得られる解 y_{m+1} を修正子と呼ぶ．特に $k = 1$, すなわち1次のギア法は

$$p_{m+1} = y_m + h_{m+1} f(x_m,y_m) \tag{4.6}$$

を予測子とし，

$$h_{m+1} f(x_{m+1},y_{m+1}) = y_{m+1} - y_m \tag{4.7}$$

により解 y_{m+1} を求める．(4.6), (4.7) はそれぞれオイラー法，後退オイラー法である，

また，ギア法は，(4.3)に示されるように，後退微分の公式に基づくところから，後退微分法 (Backward Differentiation Formula) と呼ばれ，略して BDF 法と言うこともある．

② Nordsieck 形に基づくギア法

予測子は $\pi_m(x)$ より計算し，修正子は $\pi_{m+1}(x)$ に基づいて計算することは先に述べた．ところで，修正子 y_{m+1} が求まれば $\pi_{m+1}(x)$ が決まる．この意味で，修正子の計算は $\pi_{m+1}(x)$ を作り出す計算であると考えることができる．そして，決定された $\pi_{m+1}(x)$ は，次のステップで予測子の計算に使われる．このように，積分を1ステップ進めることは，予測子，修正子の計算を通じて，$\pi_m(x)$ から $\pi_{m+1}(x)$ を作り出すことである．
以下では，$\pi_m(x)$ の表現法を述べ，それに基づく予測子，修正子の計算，更に $\pi_{m+1}(x)$ の作り方について述べる．
まず，$\pi_m(x)$ は，次のような Taylor 展開で書ける．

$$\pi_m(x) = y_m + (x-x_m)y'_m + (x-x_m)^2\frac{y''_m}{2!} + \\ ... + (x-x_m)^k\frac{y_m^{(k)}}{k!} \qquad (4.8)$$
$$\text{ただし} y_m^{(q)} = \frac{d^q}{dx^q}\pi_m(x_m)$$

本サブルーチンは，(4.8) の係数と，キザミ幅 $h = h_{m+1} = x_{m+1} - x_m$ を使って，横ベクトル

$$z_m = \left(y_m, hy'_m, ..., h^k y_m^{(k)}/k!\right) \qquad (4.9)$$

を導入し，これを $\pi_m(x)$ の表現として使う．したがって $\pi_m(x)$ から $\pi_{m+1}(x)$ を作る問題は，z_m から，

$$z_{m+1} = \left(y_{m+1}, hy'_{m+1}, ..., h^k y_{m+1}^{(k)}/k!\right) \qquad (4.10)$$

を作る問題に置き換えられる．z_m は，解の履歴に関する Nordsieckの表現と呼ばれている．予測子 p_{m+1} は (4.8) より

$$p_{m+1} = \pi_m(x_{m+1}) = \sum_{i=0}^{k} h^i y_m^{(i)}/i! \qquad (4.11)$$

と計算でき，右辺は z_m の要素の和にほかならない．しかし，予測の段階では，z_m から z_{m+1} への変更を容易にするために，(4.11) の代りに，

$$\boldsymbol{z}_{m+1(0)} = \boldsymbol{z}_m \boldsymbol{A} \qquad (4.12)$$

で定義される $z_{m+1(0)}$ を計算する．ここで $\boldsymbol{A}$ は $(k+1)\times(k+1)$ の単位下三角行列で，
$\boldsymbol{A} = (a_{ij})$ とすれば

$$a_{ij} = \begin{cases} 0 & , i < j \\ \binom{i}{j} = \dfrac{i!}{i!(i-j)!} & , i \geq j \end{cases}$$

で定義される．例として，$k = 5$ の場合の $\boldsymbol{A}$ は次のようになる．

$$A = \begin{bmatrix} 1 & & & & & \\ 1 & 1 & & & 0 & \\ 1 & 2 & 1 & & & \\ 1 & 3 & 3 & 1 & & \\ 1 & 4 & 6 & 4 & 1 & \\ 1 & 5 & 10 & 10 & 5 & 1 \end{bmatrix}$$

ちなみに，$\boldsymbol{A}$ の下三角部分が，パスカルの三角形を成すことから，$\boldsymbol{A}$ はパスカルの三角行列と呼ばれる．
(4.12) により得られる $z_{m+1(0)}$ の第1要素を $y_{m+1(0)}$ とすると，

$$y_{m+1(0)} = y_m + hy'_m + ... + h^k y_m^{(k)}/k!$$

となり，これは p_{m+1} に等しい．ゆえに，以降，p_{m+1} を $y_{m+1(0)}$ で表すことにする．また，$z_{m+1(0)}$ の第$(i+1)$ 要素は，$h^i\pi_m^{(i)}(x_{m+1})/i!$ に等しい．
これにより，(4.12) の計算は結局，x_{m+1} における解及び高階の微係数を予測する意味を持っている．
次に修正子の計算について述べる．修正子の計算は，$\boldsymbol{z}_{m+1(0)}$ と $\boldsymbol{z}_{m+1}$ の間の関係式を導くことにより得られるので，まず，その関係式を以下で導く．いま，仮に修正子 y_{m+1} が求まって，多項式 $\pi_{m+1}(x)$ が決まったとする．このとき，
$\boldsymbol{z}_{m+1} - \boldsymbol{z}_{m+1(0)}$ の第 $(i+1)$ 要素は，$\boldsymbol{z}_{m+1}$, $\boldsymbol{z}_{m+1(0)}$の定義より，

$$\frac{h^i\pi_{m+1}^{(i)}(x_{m+1})}{i!} - \frac{h^i\pi_m^{(i)}(x_{m+1})}{i!} \\ = \frac{h^i}{i!}\frac{d^i}{dx^i}\left\{\pi_{m+1}(x) - \pi_m(x)\right\}_{x=x_{m+1}} \qquad (4.13)$$

である.多項式 $\Delta_{m+1}(x) \equiv \pi_{m+1}(x) - \pi_m(x)$ は k 次の多項式であり，条件

$$\Delta_{m+1}(x_{m+1-j}) = \begin{cases} 0 & , j = 1,2,...,k \\ y_{m+1} - y_{m+1(0)} & , j = 0 \end{cases}$$

を満たす．したがって $\Delta_{m+1}(x)$ は一意的に定まり，それを，x_{m+1} における Taylor 展開で表せば

$$\Delta_{m+1}(x) = \left(y_{m+1} - y_{m+1(0)}\right)\sum_{q=0}^{k} l_q (x - x_{m+1})^q / h^q \qquad (4.14)$$

となる．ここで，l_q は，$\{x_{m+1-j}\}$ の分布に依存して決まる定数で，具体的には，
$s_j = (x_{m+1} - x_{m+1-j})/h$ として，多項式

$$\left(1+\frac{t}{s_1}\right)\left(1+\frac{t}{s_2}\right)...\left(1+\frac{t}{s_k}\right) \qquad (4.15)$$

を展開したときの t^q の係数である．実際，l_q は，次数 k 及び m にも依存するが，ここではその添字を省いた．(4.14) を使えば，(4.13) の右辺は，$(y_{m+1}-y_{m+1(0)})l_i$ となる．これが z_{m+1}- $z_{m+1(0)}$ の第 $(i+1)$ 要素である．したがって，z_{m+1} と $z_{m+1(0)}$ の関係は，横ベクトル

$$\mathbf{l}=(l_0,l_1,...,l_k) \tag{4.16}$$

を導入して，

$$\boldsymbol{z}_{m+1}=\boldsymbol{z}_{m+1(0)}+\left(y_{m+1}-y_{m+1(0)}\right)\boldsymbol{l} \tag{4.17}$$

と書き表せる．これは，同時に，修正子 y_{m+1} が求まったあとの，$\boldsymbol{z}_{m+1}$ の作り方を与える式である．

さて，修正子 y_{m+1} の計算は，(4.17) の第2要素の関係，すなわち

$$\left.\begin{aligned} hy'_{m+1}&=hy'_{m+1(0)}+\left(y_{m+1}-y_{m+1(0)}\right)l_1\\ \text{ここで } y'_{m+1}&=f\left(x_{m+1},y_{m+1}\right)\end{aligned}\right\} \tag{4.18}$$

に基づいて行う．(4.18) を書き直せば

$$\left(y_{m+1}-y_{m+1(0)}\right)-\frac{h}{l_1}\left(f\left(x_{m+1},y_{m+1}\right)-y'_{m+1(0)}\right)=0 \tag{4.19}$$

となり，したがってy_{m+1} は，

$$G(u)\equiv\left(u-y_{m+1(0)}\right)-\frac{h}{l_1}\left(f\left(x_{m+1},u\right)-y'_{m+1(0)}\right) \tag{4.20}$$

なる関数 $G(u)$ の零点として計算される．

(4.20) の零点の計算は，ニュートン法を基本として行うが，その説明は，あとの⑧で述べる．

③ 独立変数の任意な点における解

キザミ幅は，誤差の許す限り大きくとられるので，一般に，解を求めたい点 x_e が，

$$x_m<x_e\le x_{m+1} \tag{4.21}$$

となる状況が起こる．本サブルーチンでは，x_e における解 y_e を，x_{m+1} における解 y_{m+1} が求まったあと，z_{m+1} の要素を使って，

$$\begin{aligned} y_e&=\pi_{m+1}\left(x_e\right)\\ &=\sum_{i=0}^{k}\left\{\left(x_e-x_{m+1}\right)/h\right\}^i\left(h^i y^{(i)}_{m+1}/i!\right)\\ &\quad ,\ h=h_{m+1}\end{aligned} \tag{4.22}$$

と計算する．

④ 解の受入れ

修正子 y_{m+1} の局所誤差を $le_{m+1}(k)$ とすると，本サブルーチンは，これを，予測子と修正子の差を使って，

$$le_{m+1}(k)=-\frac{1}{l_1}\left\{1+\prod_{j=2}^{k}\left(\frac{x_{m+1}-x_{m+1-j}}{x_m-x_{m+1-j}}\right)\right\}^{-1}\left(y_{m+1}-y_{m+1(0)}\right) \tag{4.23}$$

と見積る．そして，許容誤差 ε に対して，

$$\left|le_{m+1}(k)\right|\le\varepsilon \tag{4.24}$$

が満足されれば，y_{m+1} を x_{m+1} における解として採用し，z_{m+1} を (4.17) から作って，そのステップは終了する．

⑤ 次数とキザミ幅の制御

本サブルーチンは，解の挙動に応じて，1次から5次までのギア法を使い分ける．各ステップにおける次数は前のステップの次数を k とすると，$k-1, k, k+1$ のいずれかより選ばれる．

今，k 次のギア法により，x_{m+1} における解 y_{m+1} が受け入れられ，次のステップのための次数を選ぶ場面を考える．次数を選ぶにあたって，本サブルーチンは，(4.23) の $le_{m+1}(k)$ のほかに，$k-1$ 次及び $k+1$ 次の局所誤差 $le_{m+1}(k-1)$, $le_{m+1}(k+1)$ を見積り，これらを比較する．ここで，$le_{m+1}(k-1)$ とは，x_{m+1} における解を仮に $k-1$ 次のギア法で求めた場合の局所誤差のことであり，$le_{m+1}(k+1)$ も同様に解釈する．

$le_{m+1}(k-1)$ 及び $le_{m+1}(k+1)$ を(4.25), (4.26) により見積る．

$$le_{m+1}(k-1)=-\left(s_1\cdot s...s_{k-1}/l_{k-1,1}\right)\left(h^k y^{(k)}_{m+1}/k!\right) \tag{4.25}$$

$$le_{m+1}(k+1)=\frac{-s_{k+1}\left(e_{m+1}-Q_{m+1}\cdot e_m\right)}{(k+2)l_{k+1,1}\left\{1+\prod_{2}^{k}\left(\frac{x_{m+1}-x_{m+1-j}}{x_m-x_{m+1-j}}\right)\right\}} \tag{4.26}$$

ここで，$l_{k\text{-}1,1}$ 及び $l_{k+1,1}$ は (4.15) において，k を $k-1$ 又は $k+1$ に置き換えた場合の t^1 の係数であり，更に，

$$e_{m+1} = y_{m+1} - y_{m+1(0)}$$
$$Q_{m+1} = (c_{m+1}/c_m)(h/h_m)^{k+1}$$
$$c_{m+1} = s_1 \cdot s_2 \ldots s_k \left\{ 1 + \prod_2^k \left(\frac{x_{m+1} - x_{m+1-j}}{x_m - x_{m+1-j}} \right) \right\} / (k+1)!$$

である．
これら三つの誤差推定量 $le_{m+1}(q)$, $q = k-1, k, k+1$ を使って，次数は以下のように選ぶ．まず，許容誤差 ε より，

$$\eta_q = \left(\varepsilon \Big/ \left| le_{m+1}(|q|) \right| \right)^{1/(q+1)}, \quad q = k-1, k, k+1 \qquad (4.27)$$

を計算する．η_q は，次のステップをq 次で進むときの，キザミ幅の許容倍率である．そこでいま，

$$\eta_{q'} = \max_q (\eta_q) \qquad (4.28)$$

とすると，q' を，次のステップの次数として採用する．すなわち，最も大きなキザミ幅をもたらす次数を採用する訳である．
これに伴って，次のステップのキザミ幅を，

$\eta_{q'} \cdot h$

とする．
以上が，次のステップへ進むときの，次数とキザミ幅の選び方である．一方，解の受入れの判定 (4.24) が満たされず，y_{m+1} が解として採用されなかったときは，そのステップをやり直す．そのとき，次数は変えず，キザミ幅を $\eta_k \cdot h$ に縮小する．

⑥ アダムス法

アダムス法については，サブルーチン ODAM の手法概要で述べてあるが，本サブルーチンにおけるアダムス法による計算過程は，ODAM のそれとは異なる．ODAM では，アダムス法を変形差分商に基づいて表現するのに対し，本サブルーチンでは，Nordsieck 形に基づいて表現する．これにより，本サブルーチンは，ギア法とアダムス法の計算過程の多くを共通化している．
ここでは，アダムス法の適用に際し，ギア法の説明で定義した諸量がいかに変更されるかを述べる．以下にその変更点を順に挙げる．

a. アダムス法における Nordsieck 形
アダムス法では，真の解 $y(x)$ を (4.29) を満たす k 次の補間多項式 $P_{m+1}(x)$ で近似する．

$$\left.\begin{aligned} P'_{m+1}(x_{m+1-j}) &= f(x_{m+1}, y_{m+1-j}) \\ & \qquad j = 0,1,\ldots,k-1 \\ P_{m+1}(x_m) &= y_m \end{aligned}\right\} \qquad (4.29)$$

(4.29) は，求めようとする解 y_{m+1} を未知パラメタとして含んでいる．このとき，y_{m+1} は，

$$y_{m+1} = y_m + \int_{x_m}^{x_{m+1}} P'_{m+1}(x)dx \qquad (4.30)$$

が成り立つように決められる．(4.30) は右辺に $f(x_{m+1}, y_{m+1})$ を含むので，一般に y_{m+1} に関する非線形方程式となる．これを解くに当っての初期値として

$$\left.\begin{aligned} P'_m(x_{m+1-j}) &= f(x_{m+1}, y_{m+1-j}) \\ & \qquad j = 1,2,\ldots,k \\ P_m(x_m) &= y_m \end{aligned}\right\} \qquad (4.31)$$

なる k 次の多項式 $P_m(x)$ より，

$$p_{m+1} = P_m(x_{m+1}) = y_m + \int_{x_m}^{x_{m+1}} P'_m(x)dx \qquad (4.32)$$

として計算される p_{m+1} を採用する．
以上のように，アダムス法は，多項式 $P_m(x)$ から $P_{m+1}(x)$ を作り出す計算であると見ることができる．
さて，$P_m(x)$ は，次のように，x_m における Taylor 展開で表現することができる．

$$P_m(x) = y_m + (x - x_m)y'_m + \ldots + (x - x_m)^k y_m^{(k)}/k! \qquad (4.33)$$

ここで，$y_m^{(q)}$ q=0,1,..., k は $P_m^{(q)}(x_m)$ を意味する．(4.33) の係数と，キザミ幅 $h = h_{m+1} = x_{m+1} - x_m$ を使って，横ベクトル

$$\boldsymbol{z}_m = \left(y_m, hy'_m, \ldots, h^k y_m^{(k)}/k! \right) \qquad (4.34)$$

を導入すれば，これがアダムス法における多項式 $P_m(x)$ の Nordsieck 形となる．

b. $\boldsymbol{z}_{m+1(0)}$ の計算
予測の段階での $\boldsymbol{z}_{m+1(0)}$ の計算は $\boldsymbol{z}_m$ として，(4.34) を使うことにすれば，ギア法における (4.12) と全く同じように行える．

c. $\boldsymbol{z}_{m+1(0)}$ と $\boldsymbol{z}_{m+1}$の関係
$\boldsymbol{z}_{m+1} - \boldsymbol{z}_{m+1(0)}$ の第 $(i+1)$ 要素は，

$$\frac{h^i}{i!} \frac{d^i}{dx^i} \{ P_{m+1}(x) - P_m(x) \}_{x = x_{m+1}} \qquad (4.35)$$

となるが，多項式 $\Delta_{m+1}(x) \equiv P_{m+1}(x) - P_m(x)$ を，x_{m+1} における Taylor 展開，

$$\Delta_{m+1}(x) = (y_{m+1} - y_{m+1(0)}) \sum_{k=0}^{k} l_q (x - x_{m+1})^q / h^q \qquad (4.36)$$

で表せば，(4.35) は $(y_{m+1} - y_{m+1(0)})l_i$ となる．
ただし，ここでの l_q は

$s_j = (x_{m+1} - x_{m+1-j})/h_{m+1}$ としたときの，t に関する k 次の多項式

$$\int_{-1}^{t}\prod_{1}^{k-1}(u+s_j)du \Big/ \int_{-1}^{0}\prod_{1}^{k-1}(u+s_j)du \qquad (4.37)$$

の t^q の係数である．この l_q を使えば，$\boldsymbol{z}_{m+1(0)}$ と $\boldsymbol{z}_{m+1}$ の関係は，形式的にギア法の場合の (4.17) に一致する．

d. 局所誤差

解の受入れ判定，若しくは，次数とキザミ幅の制御のために，アダムス法においても，$k-1$, k, $k+1$ の各次数における局所誤差を見積る．今，それらを順に $le_{m+1}(k-1)$, $le_{m+1}(k)$, $le_{m+1}(k+1)$ としたとき，アダムス法では，以下の(4.38), (4.39), (4.40) のように見積る．

$$le_{m+1}(k-1) = \left\{k\int_{-1}^{0}\prod_{0}^{k-2}(u+s_j)du\right\}\left(h^k y_{m+1}^{(k)}/k!\right) \qquad (4.38)$$

$$le_{m+1}(k) = \left\{kl_k\int_{-1}^{0}\prod_{0}^{k-1}(u+s_j)du \Big/ s_k\right\}e_{m+1} \qquad (4.39)$$

$$le_{m+1}(k+1) = \left\{kl_k\int_{-1}^{0}\prod_{0}^{k}(u+s_j)du \Big/ Ls_k\right\}\left(e_{m+1} - Q_{m+1}e_m\right) \qquad (4.40)$$

ここで

$$e_{m+1} = y_{m+1} - y_{m+1(0)},\quad L = k+1,$$

$$Q_{m+1} = \frac{s_k \cdot l_k(m)}{s_k(m)\cdot l_q}\left(h/h_m\right)^{k+1} \qquad (4.41)$$

また，(4.41) において，$s_k(m)$ は，s_k の定義において，$m+1$ を m に変えたものであり，$l_k(m)$ も，l_k の定義において $m+1$ を m に変えたものである．

解の受入れ判定，次数及びキザミ幅の制御は (4.38) ～ (4.40) を使って，ギア法の場合と同じように行う．

⑦ 連立微分方程式への拡張

連立微分方程式の場合は，解の各成分に対して，今までの議論を適用すればよい．ただし，解の受入れ判定，次数及びキザミ幅の制御で使われる $le_{m+1}(k)$ などの誤差推定量は，連立の場合，以下のようにベクトルのノルムとして定義する．すなわち，利用者の指定した

EPSV(I), I = 1, 2, ..., N 及び EPSR より

EPS = max (EPSR, max (EPSV(I)))

W(I)=(|Y(I)|・EPSR+EPSV(I))/EPS

を導入し，ERK を

$$\mathrm{ERK} = \left(\sum_{I=1}^{N}\left(\frac{le(\mathrm{I})}{\mathrm{W}(\mathrm{I})}\right)^2\right)^{1/2} \qquad (4.42)$$

と定義する．ここで，Y(I), le(I) は，第I成分の解と，その局所誤差であり，単一の微分方程式の場合の y_{m+1}, $le_{m+1}(k)$ に相当する．このとき，連立の場合の解の受入れ判定を，

$$\mathrm{ERK} \le \mathrm{EPS}$$

とする．これが満たされると，

$$|le(\mathrm{I})| \le |\mathrm{Y}(\mathrm{I})|\cdot\mathrm{EPSR} + \mathrm{EPSV}(\mathrm{I})$$
$$\mathrm{I} = 1,2,\ldots,\mathrm{N}$$

が成り立つことが分かる．

また，単一の微分方程式の場合の $le_{m+1}(k-1)$, $le_{m+1}(k+1)$ に対応するノルムERKM1, ERKP1 を (4.42) と同じように定義する．そして次数及びキザミ幅の制御は，(4.27) において，

$$\begin{aligned}
&\varepsilon && \rightarrow \mathrm{EPS}\\
&|le_{m+1}(k)| && \rightarrow \mathrm{ERK}\\
&|le_{m+1}(k-1)| && \rightarrow \mathrm{ERKM1}\\
&|le_{m+1}(k+1)| && \rightarrow \mathrm{ERKP1}
\end{aligned}$$

と置き換えて行うものとする．

⑧ 修正子反復

本サブルーチンは，修正子を非線形関数の零点として計算することは先に延べた．ここでは，連立微分方程式の場合，その零点の求め方について触れる．

(4.20) は，連立の場合には，次のように書き換えられる．

$$\boldsymbol{G}(\boldsymbol{u}) \equiv \left(\boldsymbol{u} - \boldsymbol{y}_{m+1(0)}\right) - \frac{h}{l_1}\left(\boldsymbol{f}\left(x_{m+1},\boldsymbol{u}\right) - \boldsymbol{y}'_{m+1(0)}\right) \qquad (4.43)$$

本サブルーチンは，(4.43) をニュートン法，すなわち，(4.44) を基本にして解く．

$$\left.\begin{aligned}
&\boldsymbol{u}_{r+1} = \boldsymbol{u}_r - \boldsymbol{P}_{m+1,r}^{-1}\boldsymbol{G}(\boldsymbol{u}_r)\\
&\qquad ,\ r = 0,1,2,\ldots,\\
&\text{ここで，}\ \boldsymbol{u}_0 = \boldsymbol{y}_{m+1(0)}\\
&\boldsymbol{P}_{m+1,r} = \left.\frac{\partial G}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}_r} = I - \frac{h}{l_1}\left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}_r}
\end{aligned}\right\} \qquad (4.44)$$

実際の計算では，$\boldsymbol{P}_{m+1,r}$ は，反復するたびに計算しないで，$\boldsymbol{P}_{m+1,0}$ すなわち，

$$\boldsymbol{P}_{m+1,0} = \boldsymbol{I} - \frac{h}{l_1}\boldsymbol{J}_{m+1}\ \text{ただし，}\ \boldsymbol{J}_{m+1} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{y}_{m+1(0)}} \qquad (4.45)$$

を，終始使う．こうすることにより，厳密さは多少失うが，計算量を著しく節約できる．

さて，$\boldsymbol{P}_{m+1,0}$ を計算するに当たり，本サブルーチンは，いくつかの方法を用意しており，利用者は，状況に応じて選ぶことができる．すなわち，利用者の指定するパラメタ MF の値を，

MF = 10*METH + ITER

とすると，本サブルーチンは，ITER の値に応じて $\boldsymbol{P}_{m+1,0}$ の計算を以下のように行う．

- ITER = 0 のとき，解析的ヤコビアン行列$\boldsymbol{J}_{m+1}$ を使って$\boldsymbol{P}_{m+1,0}$ を作り出す．このとき，利用者は $\boldsymbol{J}_{m+1}$ を計算するサブルーチン JAC を用意する．
- ITER = 1 のとき，$\boldsymbol{J}_{m+1}$ を，差分により近似して，$\boldsymbol{P}_{m+1,0}$ を作り出す．
- ITER = 2 のとき，$\boldsymbol{J}_{m+1}$ を，対角行列 (4.46) により近似して $\boldsymbol{P}_{m+1,0}$ を作り出す．

  $$\boldsymbol{D}_{m+1} = \mathrm{diag}\,(d_1, d_2, \ldots, d_N) \qquad (4.46)$$

  ここで，
  $d_i=[f_i(x_{m+1}, \boldsymbol{y}_{m+1(0)}+\boldsymbol{v}) - f_i(x_{m+1}, \boldsymbol{y}_{m+1(0)})] / v_i$
  $\boldsymbol{v} = -0.1\,\boldsymbol{G}(y_{m+1(0)})$
  $= (v_1, v_2, \ldots, v_N)^T$
  である．この場合は，$\boldsymbol{J}_{m+1}$ が優対角であるとき効率の良い近似となる．
- ITER = 3 のとき，$\boldsymbol{P}_{m+1,0}$ を単位行列 $\boldsymbol{I}$ とする．
  これにより，(4.44) は，単純な関数反復

  $$\boldsymbol{u}_{r+1} = \boldsymbol{u}_r - \boldsymbol{G}(\boldsymbol{u}_r) \qquad (4.47)$$

  となる．非スティフな微分方程式を解く場合は (4.47) の反復で十分である．
  反復 (4.44) の収束判定は

  $$\left(\sum_{I=1}^{N} \left(\frac{\boldsymbol{u}_{r+1}^I - \boldsymbol{u}_r^I}{\mathrm{W}(I)} \right)^2 \right)^{1/2} \le c^* \cdot \mathrm{EPS} \qquad (4.48)$$

  より行う．ここでは，$\boldsymbol{u}^I$ はベクトル $\boldsymbol{u}$ の第I要素である．
  また，c^* は，問題に応じて決まる定数である（詳細省略）．

なお，本サブルーチンは，A.C. Hindmarsh と G.D.Byrne のよるプログラム （文献 [76] , [77]） に基づいている．

H11-20-0131 ODRK1, DODRK1

連立 1 階常微分方程式 （ルンゲ・クッタ・ヴァーナー法）
CALL ODRK1 (X, Y, FUN, N, XEND, ISW, EPSA, EPSR, VW, IVW, ICON)

(1) 機能

連立 1 階常微分方程式の初期値問題,

$$\left.\begin{aligned} y_1' &= f_1(x, y_1, y_2, ..., y_N),\ y_1(x_0) = y_{10} \\ y_2' &= f_2(x, y_1, y_2, ..., y_N),\ y_2(x_0) = y_{20} \\ &... \\ y_N' &= f_N(x, y_1, y_2, ..., y_N),\ y_N(x_0) = y_{N0} \end{aligned}\right\} \quad (1.1)$$

における関数系 $f_1, f_2, ..., f_N$ と，初期値 $x_0, y_{10}, y_{20}, ..., y_{N0}$ 及び x の最終値 x_e が与えられたとき，ルンゲ・クッタ・ヴァーナー (Runge-Kutta-Verner) 法により，

$x_m = x_0 + \sum_{j=1}^{m} h_j$; m=1,2,..., e における解 ($y_{1m}, y_{2m}, ..., y_{Nm}$) を求める（図 ODRK 1-1 参照）

ただし，キザミ h_j は，求めるべき解が要求精度を満たすように制御される.

本サブルーチンは解の出力方法，すなわち利用者プログラムに戻るタイミングとして以下の2通りを用意しており，利用者は目的に応じて選ぶことができる.

a. 最終値出力... 最終値 x_e における解が求まったときはじめて戻る.

b. ステップ出力... 途中の解，すなわち，$x_1, x_2, ...$ における解が求まるたびに戻る.

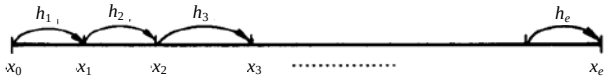


図 ODRK1-1 解を求める点 x_m ($x_0 < x_e$ の場合)

(2) パラメタ

X …… **入力**. 初期値 x_0.
出力. 最終値 x_e. ただし，ステップ出力が指定されている場合は，1ステップ前進したときの独立変数の値 x_m .

Y …… **入力**. 初期値 $y_{10}, y_{20}, ... y_{N0}$ を Y(1) = y_{10}, Y(2) = y_{20}, ..., Y(N) = y_{N0} の順に与える.
大きさ N の1次元配列.
出力. 最終値 x_e における解ベクトル.
ただし，ステップ出力が指定されている場合は，1ステップ前進したときの $x = x_m$ における解ベクトル.

FUN …… **入力**. (1.1) における f_i (i = 1, 2, ..., N) を計算するサブルーチン副プログラム名.
［副プログラムの用意のし方］
SUBROUTINE FUN (X, Y, YP)
ここで,
X ……**入力**. 独立変数 x.
Y ……**入力**. Y(1) = y_1, Y(2) = y_2, ... , Y(N) = y_N なる対応を持つ大きさ N の1次元配列.
YP …**出力**. YP(1)=$f_1(x, y_1, y_2, ..., y_N)$,
YP(2)=$f_2(x, y_1, y_2, ..., y_N)$, ... ,
YP(N)=$f_N(x, y_1, y_2, ..., y_N)$ なる 対応を持つ大きさ N の1次元配列.

N …… **入力**. 連立1階常微分方程式の元数

XEND …… **入力**. 独立変数の最終値 x_e .

ISW …… **入力**. 積分する際の条件を指定する.
ISWは10進2桁の非負の整数であり，いまそれを

$$ISW = 10d_2 + d_1$$

と表すと，各d_i は次のように指定する.

d_1: 初回呼出しであるか否かを指定する.
0: 初回呼出しである.
1: 継続呼出しである.
初回呼出しとは，与えられた微分方程式に対して初めて本サブルーチンを呼び出すときを言う.

d_2: 解の出力方法を指定する.
0: 最終値出力
1: ステップ出力

出力. x_e における解，もしくはステップごとの解を求めて利用者プログラムに戻るとき,
$d_1 = 1$
と変更される．本サブルーチンを繰返し呼び出すときは，d_1 を1のままにしておくこと.
利用者が再び $d_1 = 0$ と設定し直すときは，別の方程式を解き始めるときである.

EPSA …… **入力**. 絶対誤差の上限 (≥ 0.0)
手法概要の b. ② 参照.

EPSR …… **入力**. 相対誤差の上限 (≥ 0.0)
出力. 入力時に EPSR が小さすぎた場合は，適当な大きさに変更された値.
使用上の注意⑤参照

VW …… **作業領域**. 大きさ 9 N + 40 の1次元配列.
繰返し本サブルーチンを呼び出すときは内容を変えてはならない.

IVW …… **作業領域**. 大きさ 5の1次元配列.
繰返し本サブルーチンを呼び出すときは内容を変えてはならない.

ICON …… **出力**. コンディションコード.
表 ODRK1-1 参照.

表 ODRK1-1 コンディションコード

コード	意味	処理内容
0	（ステップ出力の場合） 1ステップ積分した.	正常. 続けて呼び出すことができる.
10	XEND における解を求めた.	正常. XEND を変えて続けて呼び出すことができる.
10000	EPSR が使用計算機の演算精度に比べて小さ過ぎたので，適当に大きくした（使用上の注意④参照）.	1ステップ進める前に利用者プログラムに戻る．続けて呼び出すことができる.
11000	微係数の計算回数が 4000 回を越えても XEND に到達できないことが判明した.	1ステップ進める前に利用者プログラムに戻る．続けて呼び出せば，微係数の計算回数カウンタを0にして，新たに回数を数えながら処理を進める.
15000	本サブルーチンで定める最小キザミを用いても，要求精度が得られないことが判明した.	1ステップ進める前に利用者プログラムに戻る. EPSA 又はEPSRを増せば続けて呼び出すことができる.
16000	（EPSA = 0のとき） 解のある成分が0になり，相対誤差判定が不可能になった.	1ステップ進める前に利用者プログラムに戻る. EPSA を 0 以外の値に変更すれば，続けて呼び出すことができる.
30000	次のいずれかであった. ① N ≤ 0 ② X = XEND ③ ISW の与え方に誤りがあった. ④ EPSA < 0 又は EPSR < 0 ⑤ ICON = 15000 又は 16000 と出力された後，EPSA 又は EPSR を何ら変更せずに呼び出した.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL, URKV, UVER

② FORTRAN 基本関数 ... ABS, SIGN, AMAX1, AMIN1

b. 注意

① 本サブルーチンは，関数評価が比較的安価な非スティフ (non-stiff) 又は弱スティフな微分方程式を解くために利用できる．しかし，高精度が要求される場合には利用すべきではない.

② パラメタ FUNに対応するプログラムの名前は，本サブルーチンを呼び出す側のプログラムで，EXTERNAL 文で宣言しなければならない.

③ 利用者は ICON が 0 又は 10 と出力されたときだけ解を得ることができる.
ICON = 10000 ～ 11000 は，その事象が判明したとき1ステップ進める前に利用者プログラムに戻る．利用者は，この事象を確認した上で，続けて本サブルーチンを呼び出すことができる.
ICON = 15000 ～ 16000 は，その事象が判明したとき，1ステップ進める前に利用者プログラムに戻るが，このとき利用者は，EPSA 又は EPSR を変更した上で，続けて本サブルーチンを呼び出すことができる（使用例参照）.

④ 相対誤差に対する上限 EPSR は

$$\mathrm{EPSR} \geq \varepsilon_{r\min} = 10^{-12} + 2u$$

を満足するように与えなければならない．ここで u は丸め誤差の単位である．EPSR が上式を満足しない場合には，本サブルーチンは ICON = 10000 と出力すると共に，

$$\mathrm{EPSR} = \varepsilon_{r\min}$$

と置いて利用者プログラムに戻る．利用者がさらに計算の続行を希望する場合には続けて本サブルーチンを呼び出せばよい.

⑤ 本サブルーチンでは，最小キザミ $h_{\min}$ を

$$h_{\min} = 26u \cdot \max\left(|x|, |d|\right)$$

を満足するように定めている．ここで x は独立変数，また，$d = (x_e - x_0)/100$ である．もし，最小キザミを用いても，なお要求精度が得られない場合，本サブルーチンは ICON = 15000 として利用者プログラムに戻る．利用者がさらに計算の続行を希望する場合には，EPSA 又は EPSR を適当に増した後，再び本サブルーチンを呼び出せばよい.

c. 使用例
連立1次階常微分方程式

$$\begin{cases} y_1' = y_1^2 y_2 \quad , \; y_1(0) = 1.0 \\ y_2' = -1/y_1 \; , \; y_2(0) = 1.0 \end{cases}$$

を EPSA = 0.0, EPSR = 10^{-5} として，$x_0 = 0.0$ から $x_e = 4.0$ まで積分する．解は各ステップ毎に印刷するものとする.

```
C     **EXAMPLE**
      DIMENSION Y(2),VW(58),IVW(5)
      EXTERNAL FUN
      X=0.0
      Y(1)=1.0
      Y(2)=1.0
      N=2
      XEND=4.0
      EPSA=0.0
      EPSR=1.0E-5
      ISW=10
   10 CALL ODRK1(X,Y,FUN,N,XEND,ISW,EPSA,
     *           EPSR,VW,IVW,ICON)
      IF(ICON.EQ.0.OR.ICON.EQ.10) GO TO 20
      IF(ICON.EQ.10000) GO TO 30
      IF(ICON.EQ.11000) GO TO 40
      IF(ICON.EQ.15000) GO TO 50
      IF(ICON.EQ.16000) GO TO 60
      IF(ICON.EQ.30000) GO TO 70
   20 WRITE(6,600) X,Y(1),Y(2)
      IF(ICON.NE.10) GO TO 10
      STOP
   30 WRITE(6,610)
      GO TO 10
   40 WRITE(6,620)
      GO TO 10
   50 WRITE(6,630)
      EPSR=10.0*EPSR
      GO TO 10
   60 WRITE(6,630)
      EPSA=1.0E-5
      GO TO 10
   70 WRITE(6,640)
      STOP
  600 FORMAT('0',10X,'X=',E15.7,10X,
     * 'Y(1)=',E15.7,10X,'Y(2)=',E15.7)
  610 FORMAT('0',10X,'RELATIVE ERROR',
     * ' TOLERANCE TOO SMALL')
  620 FORMAT('0',10X,'TOO MANY STEPS')
  630 FORMAT('0',10X,'TOLERANCE RESET')
  640 FORMAT('0',10X,'INVALID INPUT')
      END

      SUBROUTINE FUN(X,Y,YP)
      DIMENSION Y(2),YP(2)
      YP(1)=Y(1)**2*Y(2)
      YP(2)=-1.0/Y(1)
      RETURN
      END
```

(4) 手法概要

連立1階常微分方程式の初期値問題 (1.1) は，解関数ベクトル $\boldsymbol{y}(x)$, 関数ベクトル $\boldsymbol{f}(x,\boldsymbol{y})$, 初期ベクトル $\boldsymbol{y}_0$ を

$$\begin{aligned}&\boldsymbol{y}(x)=(y_1,y_2,...,y_N)^T,\\&\boldsymbol{f}(x,\boldsymbol{y})=(f_1(x,\boldsymbol{y}),f_2(x,\boldsymbol{y}),...,f_N(x,\boldsymbol{y}))^T,\\&\boldsymbol{y}_0=(y_{10},y_{20},...,y_{N0})^T\end{aligned}\qquad(4.1)$$

によって定義すれば,

$$\boldsymbol{y}'(x)=\boldsymbol{f}(x,\boldsymbol{y}),\ \boldsymbol{y}(x_0)=\boldsymbol{y}_0\qquad(4.2)$$

と表わすことができる.

a. ルンゲ・クッタ・ヴァーナー (Runge-Kutta-Verner) 法

本サブルーチンにおいては，次に示す打切り誤差の評価が可能なルンゲ・クッタ・ヴァーナー法を使用する（この方法が性能的に特にすぐれていることが，T.E.Hull らによる文献 [73] に述べられている．）．

点 $x_{m+1}=x_m+h_{m+1}$ における解を求める公式は以下のとおりである.

$$\begin{cases}\boldsymbol{k}_1=h_{m+1}\boldsymbol{f}(x_m,\boldsymbol{y}_m)\\\boldsymbol{k}_2=h_{m+1}\boldsymbol{f}\left(x_m+\frac{1}{18}h_{m+1},\boldsymbol{y}_m+\frac{1}{18}\boldsymbol{k}_1\right)\\\boldsymbol{k}_3=h_{m+1}\boldsymbol{f}\left(x_m+\frac{1}{6}h_{m+1},\boldsymbol{y}_m-\frac{1}{12}\boldsymbol{k}_1+\frac{1}{4}\boldsymbol{k}_2\right)\\\boldsymbol{k}_4=h_{m+1}\boldsymbol{f}\left(x_m+\frac{2}{9}h_{m+1},\boldsymbol{y}_m-\frac{2}{81}\boldsymbol{k}_1+\frac{4}{27}\boldsymbol{k}_2+\frac{8}{81}\boldsymbol{k}_3\right)\\\boldsymbol{k}_5=h_{m+1}\boldsymbol{f}\left(x_m+\frac{2}{3}h_{m+1},\boldsymbol{y}_m+\frac{40}{33}\boldsymbol{k}_1-\frac{4}{11}\boldsymbol{k}_2+\frac{56}{11}\boldsymbol{k}_3+\frac{54}{11}\boldsymbol{k}_4\right)\\\boldsymbol{k}_6=h_{m+1}\boldsymbol{f}\left(x_m+h_{m+1},\boldsymbol{y}_m-\frac{369}{73}\boldsymbol{k}_1+\frac{72}{73}\boldsymbol{k}_2+\frac{5380}{219}\boldsymbol{k}_3\right.\\\qquad\left.-\frac{12285}{584}\boldsymbol{k}_4+\frac{2695}{1752}\boldsymbol{k}_5\right)\\\boldsymbol{k}_7=h_{m+1}\boldsymbol{f}\left(x_m+\frac{8}{9}h_{m+1},\boldsymbol{y}_m-\frac{8716}{891}\boldsymbol{k}_1+\frac{656}{297}\boldsymbol{k}_2+\frac{39520}{891}\boldsymbol{k}_3\right.\\\qquad\left.-\frac{416}{11}\boldsymbol{k}_4+\frac{52}{27}\boldsymbol{k}_5\right)\\\boldsymbol{k}_8=h_{m+1}\boldsymbol{f}\left(x_m+h_{m+1},\boldsymbol{y}_m+\frac{3015}{256}\boldsymbol{k}_1-\frac{9}{4}\boldsymbol{k}_2-\frac{4219}{78}\boldsymbol{k}_3\right.\\\qquad\left.+\frac{5985}{128}\boldsymbol{k}_4-\frac{539}{384}\boldsymbol{k}_5+\frac{693}{3328}\boldsymbol{k}_7\right)\\\boldsymbol{y}^*_{m+1}=\boldsymbol{y}_m+\frac{3}{80}\boldsymbol{k}_1+\frac{4}{25}\boldsymbol{k}_3+\frac{243}{1120}\boldsymbol{k}_4+\frac{77}{160}\boldsymbol{k}_5+\frac{73}{700}\boldsymbol{k}_6\\\boldsymbol{y}_{m+1}=\boldsymbol{y}_m+\frac{57}{640}\boldsymbol{k}_1-\frac{16}{65}\boldsymbol{k}_3+\frac{1377}{2240}\boldsymbol{k}_4+\frac{121}{320}\boldsymbol{k}_5+\frac{891}{8320}\boldsymbol{k}_7\\\qquad+\frac{2}{35}\boldsymbol{k}_8\\T=\boldsymbol{y}_{m+1}-\boldsymbol{y}^*_{m+1}\\\quad=\frac{33}{640}\boldsymbol{k}_1-\frac{132}{325}\boldsymbol{k}_3+\frac{891}{2240}\boldsymbol{k}_4-\frac{33}{320}\boldsymbol{k}_5-\frac{73}{700}\boldsymbol{k}_6\\\qquad+\frac{891}{8320}\boldsymbol{k}_7+\frac{2}{35}\boldsymbol{k}_8\end{cases}\qquad(4.3)$$

上式において $\boldsymbol{y}^*_{m+1}$ と $\boldsymbol{y}_{m+1}$ は，それぞれ5次および6次の打切り精度を持つ近似解で，$\boldsymbol{T}$ は $\boldsymbol{y}^*_{m+1}$ の局所打切り誤差の推定値である．本サブルーチンでは，$\boldsymbol{y}^*_{m+1}$ が要求精度を満たしたとき，より高精度の近似解 $\boldsymbol{y}_{m+1}$ を解として受け入れる.

b. キザミの制御

① 初期キザミの決定

(4.3) によって与えられる $\boldsymbol{y}^*_{m+1}$ は 5次の近似解であるから，$x_1=x_0+h$ における局所打切り誤差を，解 $\boldsymbol{y}(x_0+h)$ の x_0 に関するテイラー展開の

h についての一次の項 $h\boldsymbol{f}(x_0,\boldsymbol{y}_0)$ に h^5 を乗じたもので見積り，次式によって初期キザミ h_1 を決定する．

$$h_1 = \max\left\{h_1', 26u\cdot\max\left(|x_0|,|d|\right)\right\} \tag{4.4}$$

ここで

$$h' = \min_{1\le i\le N}\left\{h_i\left|h_i^6\right| f_i(x_0,\boldsymbol{y}_0)\right| = \mathrm{EPSA} + \mathrm{EPSR}\cdot\left|y_i(x_0)\right|\right\} \tag{4.5}$$

$$d = \left(x_e - x_0\right)/100\ ,\ \mathrm{EPSR} \ge 10^{-12} + 2u \tag{4.6}$$

（u は丸め誤差の単位）

である．

② 解の受入れと棄却

(4.3) によって与えられる局所打切り誤差の推定値 $\boldsymbol{T}$ が，次の条件を満足するとき解 $\boldsymbol{y}_{m+1}$ は受け入れられる．

$$|T_i| \le \mathrm{EPSA} + \mathrm{EPSR}\cdot\frac{|y_i(x_m)| + |y_i(x_{m-1})|}{2} \tag{4.7}$$
$$,i = 1,2,\ldots,N$$

この判定は，EPSA = 0 と与えられたとき相対誤差判定となる．精度について十分な保証が求められるときには，相対誤差判定によるのが望ましい．一方，EPSR = 0.0 と与えられた場合は，サブルーチンの内部で自動的に EPSR = 10^{-12}+2u と訂正される．このとき，EPSA は，解の絶対値が余り大きくない場合には，絶対誤差のほぼ上限を与える．解の絶対値が大きい場合には，(4.7) 式右辺の第2項が絶対誤差の上限を与え，実質的に相対誤差判定となる．

③ キザミの変更

$\boldsymbol{y}^*_{m+1}$ の第 i 成分の打切り誤差項の主項は，キザミを h とすれば h^6C_i と表わすことができる．ここで C_i は定数である．この場合，h が十分に小さければ

$$h^6C_i \approx T_i \tag{4.8}$$

が成り立つ．したがって，今，キザミを h から sh に変化すれば，打切り誤差の推定値は T_i から $s^6\,T_i$ に変化することになる．このことを利用して，キザミの変更は次のように行う．

- 前のキザミが失敗であった場合

  すなわち，ある整数 i_0 が存在して，

$$|T_{i0}| > \mathrm{EPSA} + \mathrm{EPSR}\cdot\frac{|y_{i0}(x_m)| + |y_{i0}(x_{m+1})|}{2} \tag{4.9}$$

が成り立つ場合，次に試みるキザミは以下のようにして，決定される．解の第 i 成分が受け入れられるためには，キザミの倍率 s_i は次の条件を満足しなければならない．すなわち，

$$s_i^6|T_i| = \mathrm{EPSA} + \mathrm{EPSR}\cdot\frac{|y_i(x_m)| + |y_i(x_{m+1})|}{2} \tag{4.10}$$

(4.10) を満足する s_i の中で最小のものを選び，安全係数0.9をかければ，キザミの倍率 s が次式のように得られる．

$$s = 0.9\sqrt[6]{\min_{1\le i\le N}\frac{\mathrm{EPSA} + \mathrm{EPSR}\cdot\frac{|y_i(x_m)| + |y_i(x_{m+1})|}{2}}{|T_i|}} \tag{4.11}$$

(4.11) によって定められる s が 0.1 以下の場合には s = 0.1 とする．s > 0.1 の場合，すなわち

$$\max_{1\le i\le N}\frac{|T_i|}{\mathrm{EPSA} + \mathrm{EPSR}\cdot\frac{|y_i(x_m)| + |y_i(x_{m+1})|}{2}} > 9^6 \tag{4.12}$$

ならば s は (4.11) を計算して求める．

- 前のキザミが成功であった場合

  (4.11) によって決定される s が5以上の場合 s = 5.0 とする．s < 5.0 の場合，すなわち，

$$\max_{1\le i\le N}\frac{|T_i|}{\mathrm{EPSA} + \mathrm{EPSR}\cdot\frac{|y_i(x_m)| + |y_i(x_{m+1})|}{2}} > \left(\frac{9}{50}\right)^6$$

ならば s は (4.11) を計算して求める．

なお，ルンゲ・クッタ・ヴァーナー法についての詳細は参考文献 [72] を参照すること．

F12-15-0402 PNR, DPNR

ビット逆転によるデータの置換
CALL PNR (A, B, NT, N, NS, ISN, ICON)

(1) 機能

n 個の複素データ $\{\alpha_k\}$ が与えられたとき，そのデータの並び順位を，ビット逆転の意味で逆順に並べかえた $\{\tilde{\alpha}_k\}$ を求める，又は，逆に $\{\tilde{\alpha}_k\}$ が与えられたとき，$\{\alpha_k\}$ を求める．ただし，$n = 2^l$ (l: 0 又は，正整数）であること．

ここでビット逆転の意味で逆順に並びかえるとは次のことを意味する．すなわち，$\{\alpha_k\}$ 又は $\{\tilde{\alpha}_k\}$ のある要素を，その並び順位を 2 進表現により(1.1)で表すとき，

$$k = k_0 + k_1 \cdot 2 + \cdots\cdots + k_{l-1} \cdot 2^{l-1} \qquad (1.1)$$

補助変数の並びを逆転した(1.2)なる並び順位に置き換えることである．

$$\tilde{k} = k_{l-1} + k_{l-2} \cdot 2 + \ldots + k_0 \cdot 2^{l-1} \qquad (1.2)$$

並べかえは，両順位の要素を互換することにより行う．

本ルーチンは，高速複素フーリエ変換で必要となるデータの並べかえを行うものである．

(2) パラメタ

A ……**入力**．$\{\alpha_k\}$ 又は $\{\tilde{\alpha}_k\}$ の実部．
出力．$\{\tilde{\alpha}_k\}$ 又は $\{\alpha_k\}$ の実部．
大きさ NT の 1 次元配列．

B ……**入力**．$\{\alpha_k\}$ 又は $\{\tilde{\alpha}_k\}$ の虚部．
出力．$\{\tilde{\alpha}_k\}$ 又は $\{\alpha_k\}$ の虚部．
大きさ NT の 1 次元配列．

NT ……**入力**．並べかえの対象となる $\{\alpha_k\}$ 又は，$\{\tilde{\alpha}_k\}$ が含まれるデータの総数 (≥N, ≥NS)
通常，NT = N として指定する．
（使用上の注意②参照）

N ……**入力**．データの個数 n.

NS……**入力**．NT 個のデータのうち，並べかえの対象となる $\{\alpha_k\}$ 又は，$\{\tilde{\alpha}_k\}$ のあい隣り合うデータの間隔(≥ 1 かつ ≤ NT).
通常，NS = 1 として指定する．
（使用上の注意②参照）

ISN ……**入力**．NT 個のデータの，あい隣り合うデータの間隔 (≠0).
通常，ISN =±1 として指定する．
（使用上の注意③参照）

ICON ……**出力**．コンディションコード．
表 PNR-1 参照

表 PNR-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	ISN = 0, NS < 1, NT < N, NT < NS, 又は $N \neq 2^l$ (l : 0 又は正整数)であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... MGSSL

② FORTRAN 基本関数 ..... ALOG, IABS

b. 注意

① 本サブルーチンの用途について(1)
本サブルーチンは，通常サブルーチン CFTN 若しくは CFTR と併用して用いる．
ところで，一般的に離散型複素フーリエ変換及び，逆変換は(3.1), (3.2) で定義される．

$$\alpha_k = \frac{1}{n}\sum_{j=0}^{n-1} x_j \omega^{-jk} \quad , k = 0,1,\ldots,n-1 \qquad (3.1)$$

$$x_j = \sum_{k=0}^{n-1} \alpha_k \omega^{jk} \, , j = 0,1,\ldots,n-1 \qquad (3.2)$$

CFTN では，(3.1), (3.2) に対応して，(3.3), (3.4) なる変換を行う．

$$n\tilde{\alpha}_k = \sum_{j=0}^{n-1} x_j \omega^{-jk}, k = 0,1,\ldots,n-1 \qquad (3.3)$$

$$\tilde{x}_j = \sum_{k=0}^{n-1} \alpha_k \omega^{jk} \, , j = 0,1,\ldots,n-1 \qquad (3.4)$$

一方，CFTR では，(3.1), (3.2) に対応して，(3.5), (3.6)なる変換を行う．

$$n\alpha_k = \sum_{j=0}^{n-1} \tilde{x}_j \omega^{-jk}, k = 0,1,\ldots,n-1 \qquad (3.5)$$

$$x_j = \sum_{k=0}^{n-1} \tilde{\alpha}_k \omega^{jk} \, , j = 0,1,\ldots,n-1 \qquad (3.6)$$

すなわち本サブルーチンは，CFTN と併用して (3.3), (3.4) による変換の後に $\{\tilde{\alpha}_k\}$, $\{\tilde{x}_j\}$ の並べかえを行う．若しくは，CFTR と併用して，(3.5), (3.6) による変換の直前に $\{x_j\}$, $\{\alpha_k\}$ の並べかえを行う．
なお，本サブルーチンのパラメタの内容は，本質的には CFTN，CFTR と同じであるので，指定方法は同一と考えてよい．
使用例①，②参照のこと．

② 本サブルーチンの用途について (2)
本サブルーチンは，①で述べたように CFTN, CFTR と併用して用いるが，CFTN，CFTR により多次元のフーリエ変換又は逆変換を行う場合も，同様に用いることができる．

ところで一般的に多次元離散型複素フーリエ変換は，(3.7)で指定される．（2 次元の場合）

$$\alpha_{K1,K2}=\frac{1}{N1\cdot N2}\sum_{J1=0}^{N1-1}\sum_{J2=0}^{N2-1}x_{J1,J2}\omega_1^{-J1,K1}\omega_2^{-J2,K2} \qquad (3.7)$$
$$,K1=0,1,\ldots,N1-1,K2=0,1,\ldots,N2-1$$

CFTN では，(3.7)に対応して，(3.8)なる変換を行うことができる．

$$N1\cdot N2\tilde{\alpha}_{K1,K2}=\sum_{J1=0}^{N1-1}\sum_{J2=0}^{N2-1}x_{J1,J2}\omega_1^{-J1\cdot K1}\omega_2^{-J2\cdot K2} \qquad (3.8)$$
$$,K1=0,1,\ldots,N1-1,K2=0,1,\ldots,N2-1$$

一方，CFTR では，(3.7)に対応して，(3.9) なる変換を行うことができる．

$$N1\cdot N2\alpha_{K1,K2}=\sum_{J1=0}^{N1-1}\sum_{J2=0}^{N2-1}\tilde{x}_{J1,J2}\omega_1^{-J1,K1}\omega_2^{-J2,K2} \qquad (3.9)$$
$$,K1=0,1,\ldots,N1-1,K2=0,1,\ldots,N2-1$$

逆変換の場合も,1 次元の逆変換と同様な変換を行うことができる．
すなわち，本サブルーチン CFTN と併用して，(3.8) による変換の後に$\{N1\cdot N2\ \tilde{\alpha}_{k1,k2}\}$の並べかえを行う，若しくは，CFTR と併用して，(3.9) による変換の直前に$\{x_{J1J2}\}$ の並べかえを行う．使用例③参照のこと．

③ ISN の与え方について
NT 個のデータの実部，虚部が各々NT･I なる大きさの領域に，間隔 I で格納されている場合は，次のように指定する．

ISN =± I

ここで，I の符号は正負どちらでもよい．
この場合，並べかえた結果も間隔 I で格納される．

c. 使用例

① 1 次元変換に伴う並べかえ(1)
n 項の複素時系列データ$\{x_j\}$に対するフーリエ変換をサブルーチン CFTN により行い，その結果の$\{n\tilde{\alpha}_k\}$について本サブルーチンにより並べかえを行い，正規化して$\{\alpha_k\}$を求める．
$n \le 1024\ (=2^{10})$の場合．

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
      CALL CFTN(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      CALL PNR(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      DO 10 I=1,N
      A(I)=A(I)/FLOAT(N)
      B(I)=B(I)/FLOAT(N)
   10 CONTINUE
      WRITE(6,620) (I,A(I),B(I),I=1,N)
      STOP
  500 FORMAT(I5/(2E20.7))
  600 FORMAT('0',10X,'INPUT DATA N=',I5/
     *       /(15X,I5,2E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5)
  620 FORMAT(15X,I5,2E20.7)
      END
```

② 1 次元変換に伴う並べかえ(2)
n 項の複素時系列データ$\{x_j\}$に対するフーリエ変換を行う前に，本サブルーチンにより並べかえを行い，その結果の$\{\tilde{x}_j\}$についてサブルーチン CFTR により変換し，正規化して$\{\alpha_k\}$を求める．
$n \le 1024\ (=2^{10})$の場合．

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024)
      READ(5,500) N,(A(I),B(I),I=1,N)
      WRITE(6,600) N,(I,A(I),B(I),I=1,N)
      CALL PNR(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      CALL CFTR(A,B,N,N,1,1,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      DO 10 I=1,N
      A(I)=A(I)/FLOAT(N)
      B(I)=B(I)/FLOAT(N)
   10 CONTINUE
      WRITE(6,620) (I,A(I),B(I),I=1,N)
      STOP
  500 FORMAT(I5/(2E20.7))
  600 FORMAT('0',10X,'INPUT DATA N=',I5/
     *       /(15X,I5,2E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5)
  620 FORMAT(15X,I5,2E20.7)
      END
```

③ 2 次元変換に伴う並べかえ
$N1\cdot N2$ 項の複素時系列データ$\{x_{j1,\,j2}\}$に対するフーリエ変換をサブルーチン CFTN により行い，その結果の$\{N1\cdot N2\ \tilde{\alpha}_{k1,k2}\}$について本サブルーチンにより並べかえを行い，正規化して$\{\alpha_{k1,k2}\}$を求める．
$N1\cdot N2 \le 1024\ (=2^{10})$の場合．
なお，データ$\{x_{j1,\,j2}\}$は，図 PNR-1 で示されるように格納すればよい．

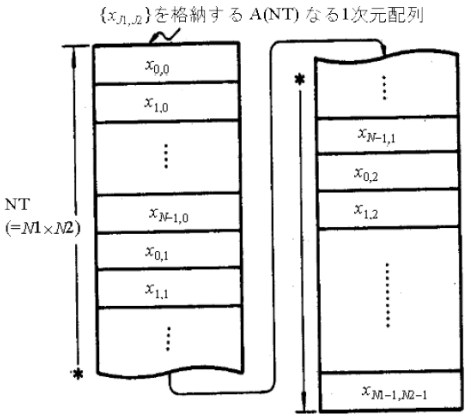


図 PNR-1 $\{x_{J1,J2}\}$の格納方法

```
C     **EXAMPLE**
      DIMENSION A(1024),B(1024),N(2)
      READ(5,500) (N(I),I=1,2)
      NT=N(1)*N(2)
      READ(5,510) (A(I),B(I),I=1,NT)
      WRITE(6,600) N,(I,A(I),B(I),I=1,NT)
      NS=1
      DO 10 I=1,2
      CALL CFTN(A,B,NT,N(I),NS,1,ICON)
      IF(ICON.NE.0) STOP
      CALL PNR(A,B,NT,N(I),NS,1,ICON)
      NS=NS*N(I)
   10 CONTINUE
      DO 20 I=1,NT
      A(I)=A(I)/FLOAT(NT)
      B(I)=B(I)/FLOAT(NT)
   20 CONTINUE
      WRITE(6,610) (I,A(I),B(I),I=1,NT)
      STOP
  500 FORMAT(2I5)
  510 FORMAT(2E20.7)
  600 FORMAT('0',10X,'INPUT DATA N=',2I5/
     *      /(15X,I5,2E20.7))
  610 FORMAT('0',10X,'OUTPUT DATA'/
     *      /(15X,I5,2E20.7))
      END
```

(4) 手法概要

2 を基底とする高速フーリエ変換で必要となる，ビット逆転の意味での逆順となるようにデータの並べかえを行う．

まず，離散型複素フーリエ変換を(4.1)で定義する．

$$\alpha_k = \sum_{j=0}^{n-1} x_j \exp\left(-2\pi i \frac{jk}{n}\right), k = 0,1,\ldots,n-1 \qquad (4.1)$$

ここで，(4.1)では通常の正規化定数 $1/n$ は省略している．

今，$n = 2^l$ である場合，高速フーリエ変換を適用することを考える．（高速フーリエ変換の原理については，サブルーチン CFT の項を参照のこと）．

$\{x_j\}$ を与えた領域だけで変換を行う場合には，変換する直前若しくは，変換した直後にデータの並べかえが必要となる．

すなわち，(4.1)における k, j を

$$k = k_0 + k_1 \cdot 2 + \ldots + k_{l-1} \cdot 2^{l-1}, k_0,\ldots,k_{l-1} = 0,1$$
$$j = j_0 + j_1 \cdot 2 + \ldots + j_{l-1} \cdot 2^{l-1}, j_0,\ldots,j_{l-1} = 0,1 \qquad (4.2)$$

と表現したとき，高速フーリエ変換の結果は，

$$\alpha\,(k_0 + k_1 \cdot 2 + \cdots + k_{l-1} \cdot 2^{l-1})$$

に対し

$$\tilde{\alpha}\,(k_{l-1} + k_{l-2} \cdot 2 + \ldots + k_0 \cdot 2^{l-1})$$

となる．

ここで，$\alpha\,(k_0 + k_1 \cdot 2 + \ldots + k_{l-1} \cdot 2^{l-1}) \equiv \alpha_{k_0 + k_1 \cdot 2 + \ldots k_{l-1} \cdot 2^{l-1}}$ は，データ $\{\alpha_k\}$ の並び順位を 2 進法で表現したものと解釈できる．したがって，$\{\tilde{\alpha}_k\}$ は $\{\alpha_k\}$ に対してビット逆転の意味でそのデータの並びが逆順となり，最終的にデータの並べかえが必要となる．一方

$$k = k_{l-1} + k_{l-2} \cdot 2 + \ldots + k_0 \cdot 2^{l-1}, k_0,\ldots,k_{l-1} = 0,1$$
$$j = j_{l-1} + j_{l-2} \cdot 2 + \ldots + j_0 \cdot 2^{l-1}, j_0,\ldots,j_{l-1} = 0,1 \qquad (4.3)$$

と表現した場合の高速複素フーリエ変換の結果は，

$$\alpha\left(k_0 + k_1 \cdot 2 + \ldots + k_{l-1} \cdot 2^{l-1}\right)$$

となり，ビット逆転はせず正順となる．

すなわち，変換の直前にデータ$\{x_j\}$の並びを(4.3) の j で示されるようにビット逆転の意味で逆順となるように並びかえるならば，最終的にデータの並べかえは不要である．

以上述べたように，高速複素フーリエ変換では，本質的にはあるデータ $\{\alpha_k\}$ において(4.4)，(4.5)で示される順位のデータの互換が必要となる．

$$k = k_0 + k_1 \cdot 2 + \ldots + k_{l-1} \cdot 2^{l-1} \qquad (4.4)$$
$$\tilde{k} = k_{l-1} + k_{l-2} \cdot 2 + \ldots + k_0 \cdot 2^{l-1} \qquad (4.5)$$

本サブルーチンでは，$\{\alpha_k\}$ 又は $\{\tilde{\alpha}_k\}$ が与えられたとき，ビット逆転の意味で逆順となるようにデータを並べかえ，$\{\tilde{\alpha}_k\}$ 又は $\{\alpha_k\}$ を求める．

並べかえのための基本条件は，次のとおりである．

(a) $k = \tilde{k}$ の場合は互換は不要である

(b) $k \neq \tilde{k}$ の場合は，$\alpha(k)$ と $\alpha(\tilde{k})$（又は，$\alpha(\tilde{k})$ と $\tilde{\alpha}\,(\tilde{k})$）を互換する．

なお，詳細については，参考文献[55], [56]及び，[57]を参照すること．

J12-20-0101 RANB2

二項乱数の生成
CALL RANB2 (M, P, IX, IA, N, VW, IVW, ICON)

(1) 機能

与えられた母数 m, p で決まる二項分布の確率密度関数(1.1)から，指定された n 個の擬似乱数（整数）を生成する．

$$P_k = \binom{m}{k} p^k (1-p)^{m-k} \qquad (1.1)$$

$$,0 < p < 1, \quad k = 0,1,\ldots,m, \quad m = 1,2,\ldots$$

$n \geq 1$ であること．

(2) パラメタ

M …… **入力**．母数 m.
P …… **入力**．母数 p.
IX …… **入力**．初期値．非負の整変数（INTEGER*4 であること）．
出力．次の本サブルーチン呼出しのための初期値．
（使用上の注意参照）．
IA …… **出力**．n 個の二項乱数．
大きさ n の 1 次元配列．
N …… **入力**．生成する二項乱数の個数 n.
VW …… **作業領域**．大きさ $m+1$ の 1 次元配列．
IVW …… **作業領域**．大きさ $m+1$ の 1 次元配列．
ICON …… **出力**．コンディションコード．
表 RANB2-1 参照．

表 RANB2-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	M < 1, P ≤ 0, P ≥ 1, IX < 0 又は N < 1 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 .... ALOG, EXP, DMOD, FLOAT

b. 注意
① 初期値 IX について
本サブルーチンは，一様乱数から二項乱数への変換を行っている．パラメタ IX は，一様乱数を生成するときの初期値として与えるもので,その扱いは RANU2 の場合と同じである．（RANU2 の「使用上の注意」参照）．
② パラメタ M, P を同じ値で使用している間は，VW, IVW の内容を変更してはならない．

c. 使用例
m = 20, p = 0.75 の二項分布から擬似乱数を 10000 個生成し，その頻歩分布を示すヒストグラムをプロットする．

```
C     **EXAMPLE**
      DIMENSION IA(10000),VW(21),IVW(21),
     *          HSUM(21)
      INTEGER*4 IX
      DATA X1,X2/-0.5,20.5/
      DATA NINT,XINT/21,1.0/
      DATA M,P,IX,N/20,0.75,0,10000/
      DO 10 I=1,21
   10 HSUM(I)=0.0
      CALL RANB2(M,P,IX,IA,N,VW,IVW,ICON)
C     SUM NOS. IN HISTGRAM FORM
      ISW=1
      DO 20 I=1,N
      X=IA(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,HINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'BINOMIAL RANDOM ',
     * 'NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST については，サブルーチン RANU2 の「使用例」参照のこと．

(4) 手法概要

二項乱数の生成は次のようにして行われる，すなわち区間(0, 1)での一様分布から生成された擬似乱数 u が，(4. 1)を満たすように整数 l を決定するとき，l が二項乱数となる．

$$F_{l-1} \leq u < F_l \qquad (4.1)$$

ここで，F_l は (4.2)で示す二項累積分布関数である．

$$F_{-1} = 0$$
$$F_l = \sum_{k=0}^{l} \binom{m}{k} p^k (1-p)^{m-k}, \quad l = 0,1,\ldots,m \qquad (4.2)$$

(4.1)の方法を m = 6, p = 0.5 の場合について図 RANB2-1 に示す．例えば u = 0.74321 とすれば l = 4 となる．

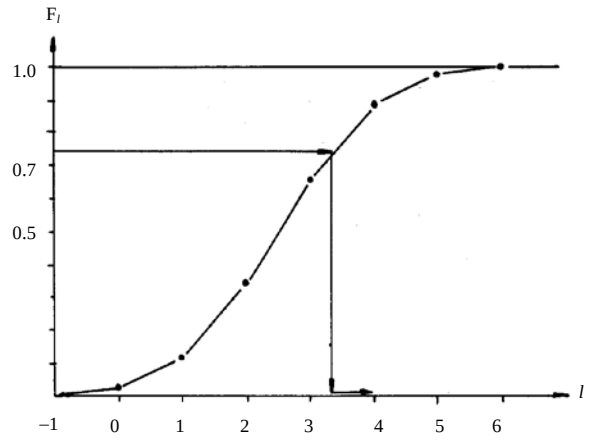


図 RANB2-1 二項累積分布関数 F_l

この累積分布は，m，p が決まれば一意的に定まるものであるから，一個の二項乱数に対して毎回(4.2)を計算するのは無駄である．したがって，最初に累積分布表を作成しておけば，後はこの表を参照するだけでよい．パラメタ VW はこのために使用される．更に，もし u が 1 に近い値である場合，(4.1)において，l =0, 1, 2, ...と調べてゆくのも時間的に無駄である．パラメタ IVW は，u の値によって l を適当な値から開始するための索引表として使用される．

なお詳細については，参考文献[93]を参照すること．

11-30-0101 RANE2

指数乱数の生成
CALL RANE2 (AM, IX, A, N, ICON)

(1) 機能

与えられた平均 m を持つ指数分布の確率密度関数(1.1)から，指定された n 個の擬似乱数を生成する．

$$g(x)=\frac{1}{m}e^{-\frac{x}{m}} \quad (1.1)$$

ここに，$x \geq 0, m > 0$ である．また $n \geq 1$ であること．

(2) パラメタ

AM ……**入力**．指数分布の平均 m.

IX ……**入力**．初期値．非負の整変数（INTEGER *4 であること）．
出力．次の本サブルーチン呼出しのための初期値．
（使用上の注意参照）．

A……**出力**．n 個の指数乱数．
大きさ n の 1 次元配列・

N……**入力**．生成する指数乱数の個数 n.

ICON……**出力**．コンディションコード．
表 RANE2-1 参照.

表 RANE2-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	AM ≤ 0, IX < 0 又は, N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... RANU2, MGSSL

② FORTRAN 基本関数 ... ALOG, DMOD

b. 注意

① 初期値 IX について
本サブルーチンは一様乱数生成サブルーチン RANU2 を使用して，一様乱数から指数乱数への変換することを行っている．
パラメタ IX は一様乱数を生成するときの初期値として与えるものであり，その扱いは RANU2 の場合と同じである（RANU2 の「使用上の注意」参照）．

c. 使用例
平均 1.0 の指数分布から擬似乱数を 10000 個生成し，その頻度分布を示すヒストグラムをプロットする．

```
C     **EXAMPLE**
      DIMENSION A(10000),HSUM(12)
      INTEGER*4 IX
      DATA X1,X2,NINT,XINT/0.0,6.0,12,0.5/
      DATA AM,IX,N/1.0,0,10000/
      DO 10 I=1,12
   10 HSUM(I)=0.0
      CALL RANE2(AM,IX,A,N,ICON)
C     SUM NOS. IN HISTGRAM FROM
      ISW=1
      DO 20 I=1,N
      X=A(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'EXPONENTIAL',
     * ' RANDOM NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST についてはサブルーチン RANU2 の「使用例」参照のこと．

(4) 手法概要

指数乱数 y は次の関係式を使って生成される．

$$y = -m\log u \quad (4.1)$$

ここに，u は区間(0, 1) での一様分布から生成された擬似乱数であり，m は (1.1)式で示されている平均である．

(4.1)は次のように導かれる．すなわち，指数分布の累積分布関数は(1.1)式より(4.2)となる．

$$F(y)=\int_0^y g(x)dx=\int_0^y \frac{1}{m}e^{-\frac{1}{m}x}dx=1-e^{-\frac{1}{m}y} \quad (4.2)$$

ところで,

$$u = F(y)$$

なる関係より，いまひとつの一様乱数を u_1 とすると，(4.2) より (4.3)が得られる．

$$y_1 = -m\log(1-u_1) \quad (4.3)$$

1- u_1 を改めて u_1 とおくと (4.1)が導かれる．
このようにして，$u_1, u_2, u_3, ...$から変換された $y_1, y_2, y_3, ...$は指数分布に従って分布する擬似乱数列と考えることができる．

J11-20-0301 RANN1

正規乱数の生成（高速型）
CALL RANN1 (AM, SD, IX, A, N, ICON)

(1) 機能

与えられた平均 m と標準偏差 σを持つ正規分布の確率密度関数(1.1)から，指定された n 個の擬似乱数を生成する．

$$g(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-m)^2/2\sigma^2} \qquad (1.1)$$

$n \geq 1$ であること．

(2) パラメタ

AM ……**入力**．正規分布の平均 m.
SD……**入力**．正規分布の標準偏差 σ.
IX……**入力**．初期値．非負の整変数
(INTEGER *4 であること).
出力．次の本サブルーチン呼出しのための初期値.
（使用上の注意参照）.
A……**出力**．n 個の正規乱数.
大きさ n の 1 次元配列・
N……**入力**．生成する正規乱数の個数 n.
ICON……**出力**．コンディションコード.
表 RANN1-1 参照 .

表 RANN1-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	IX < 0 又は N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL
② FORTRAN 基本関数 ... SQRT, ALOG, ABS, SIGN, DFLOAT, DINT

b. 注意
① 初期値 IX について
本サブルーチンは一様乱数を生成し，これを正規乱数へ変換することを行っている.
パラメタ IX は一様乱数を生成するときの初期値として与えるものであり，その扱いは RANU2 の場合と同じである（RANU2 の「使用上の注意」参照）.

c. 使用例
平均 0，標準偏差 1.0 の正規分布から擬似乱数を 10000 個生成し，その頻度分布を示すヒストグラムをプロットする.

```
C     **EXAMPLE**
      DIMENSION A(10000),HSUM(12)
      INTEGER*4 IX
      DATA X1,X2/-3.0,3.0/
      DATA NINT,XINT/12,0.5/
      DATA AM,SD,IX,N/0.0,1.0,0,10000/
      DO 10 I=1,12
   10 HSUM(I)=0.0
      CALL RANN1(AM,SD,IX,A,N,ICON)
C     SUM NOS. IN HISTGRAM FORM
      ISW=1
      DO 20 I=1,N
      X=A(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'NORMAL RANDOM',
     * ' NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST についてはサブルーチン RANU2 の「使用例」参照のこと.

(4) 手法概要

正規乱数の生成は逆関数法によっている．すなわち，区間(0, 1)での一様分布から生成された一様乱数 u_i (i = 1, 2, ..., n)に対して変換

$$z_i = G^{-1}(u_i)\sigma + m \qquad (4.1)$$

を施すことにより，正規乱数 z_i (i = 1, 2, ..., n)を得る．ここに $G^{-1}(u)$ は，累積正規分布関数

$$G(z)=\int_{-\infty}^{x} g(x)dx$$

の逆関数である.
本サブルーチンでは，二宮による最良近似式を用いることにより高速度で $G^{-1}(u)$ を計算している.

a. $|u-0.5| \leq 0.46875$ の場合

$$G^{-1}(u) = cx\left(e/\left(x^2+d\right)+x^2+b\right)$$

ただし，$x = u - 0.5$，理論絶対精度 $7.9 \cdot 10^{-4}$

b. $|u-0.5| > 0.46875$ の場合

$G^{-1}(u) = \text{sign}(u\text{-}0.5) \cdot p(v+q+r/v)$

ただし， $v = \sqrt{-\log\left(0.5-|u-0.5|\right)}$,

理論絶対精度 $9.3 \cdot 10^{-4}$

$G^{-1}(u)$の計算には殆んどの場合（確立 15/16），

a. 式が用いられるので非常に高速である.

J11-20-0101 RANN2

正規乱数の生成
CALL RANN2 (AM, SD, IX, A, N, ICON)

(1) 機能

与えられた平均 m と標準偏差 σを持つ正規分布の確率密度関数(1.1)から，指定された n 個の擬似乱数を生成する．

$$g(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-m)^2/2\sigma^2} \qquad (1.1)$$

$n \geq 1$ であること．

(2) パラメタ

AM ……**入力**．正規分布の平均 m.
SD……**入力**．正規分布の標準偏差 σ.
IX……**入力**．初期値．非負の整変数 (INTEGER *4 であること).
出力．次の本サブルーチン呼出しのための初期値.
（使用上の注意参照）
A……**出力**．n 個の正規乱数.
大きさ n の 1 次元配列.
N……**入力**．生成する正規乱数の個数 n.
ICON……**出力**．コンディションコード.
表 RANN2-1 参照.

表 RANN2-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	IX < 0 又は N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... RANU2, MGSSL
② FORTRAN 基本関数 ... SQRT, ALOG, SIN, COS , DMOD

b. 注意
① 初期値 IX について
本サブルーチンは一様乱数生成サブルーチン RANU2 を使用して，一様乱数から正規乱数へ変換することを行っている.
パラメタ IX は一様乱数を生成するときの初期値として与えるものであり，その扱いは RANU2 の場合と同じである．（RANU2 の「使用上の注意」参照）.

c. 使用例
平均 0，標準偏差 1.0 の正規分布から擬似乱数を 10000 個生成し，その頻度分布を示すヒストグラムをプロットする.

```
C     **EXAMPLE**
      DIMENSION A(10000),HSUM(12)
      INTEGER*4 IX
      DATA X1,X2/-3.0,3.0/
      DATA NINT,XINT/12,0.5/
      DATA AM,SD,IX,N/0.0,1.0,0,10000/
      DO 10 I=1,12
   10 HSUM(I)=0.0
      CALL RANN2(AM,SD,IX,A,N,ICON)
C     SUM NOS. IN HISTGRAM FORM
      ISW=1
      DO 20 I=1,N
      X=A(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'NORMAL RANDOM',
     * ' NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST についてはサブルーチン RANU2 の「使用例」参照のこと.

(4) 手法概要

正規乱数の生成はボックス・ミュラー(Box and Müllar) の方法によっている．すなわち，区間 (0, 1) の一様分布から生成された二つの一様乱数 u_i , u_{i+1} に，(4.1)及び (4.2)の変換を施すことにより正規乱数 z_i , z_{i+1} を得る.

$$z_i=\sigma\left(-2\log u_i\right)^{\frac{1}{2}}\cos 2\pi u_{i+1}+m \qquad (4.1)$$

$$z_{i+1}=\sigma\left(-2\log u_i\right)^{\frac{1}{2}}\sin 2\pi u_{i+1}+m \qquad (4.2)$$

ここに m, σ はそれぞれ正規分布の平均と標準偏差である．これにより一様乱数 u_1, u_2, u_3, ...から，これと同数の正規乱数 z_1, z_2, z_3, ...が得られる．ただし，奇数個の正規乱数を生成する場合は，当然のことながら一つ余計に一様乱数が必要となる.

J12-10-0101 RANP2

ポアソン乱数の生成
CALL RANP2 (AM, IX, IA, N, VW, IVW, ICON)

(1) 機能

与えられた平均 m を持つポアソン分布の確率密度関数(1.1)から，指定された n 個の擬似乱数（整数）を生成する．

$$p_k = e^{-m}\frac{m^k}{k!} \qquad (1.1)$$

ここに，$m < 0$, k は非負の整数である．また，$n \geq 1$ である．(図 RANP2-1 参照).

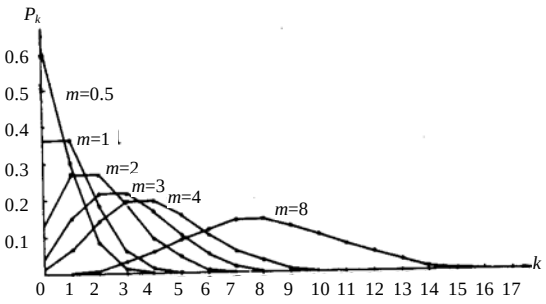


図 RANP2-1 ポアソン分布の確率密度関数 P_k

(2) パラメタ

AM ……**入力**．ポアソン分布の平均 m.
（使用上の注意参照）

IX ……**入力**．初期値．非負の整変数（INTEGER*4 であること）．
出力．次の本サブルーチン呼出しのための初期値.
（使用上の注意参照）

IA……**出力**．n 個のポアソン乱数.
大きさ n の 1 次元配列・

N……**入力**．生成するポアソン乱数の個数 n.

VW……**作業領域**．大きさ $[2m + 10]$の 1 次元配列.

IVW……**作業領域**．大きさ$[2m + 10]$の 1 次元配列.

ICON……**出力**．コンディションコード.
表 RANP2-1 参照.

表 RANP2-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	AM ≤ 0, AM > log(fl_{max}) IX < 0 又は N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... EXP, DMOD

b. 注意

① パラメタ AM ≤log(fl_{max})であること.
AM の値が log(fl_{max})を越えると(4.2)式の e^{-m} の計算においてアンダフローを起こす．このとき F_n の値が正確に計算できなくなるのでこれを防ぐためである.

② パラメタ AM の値 (m)が大きいとき（およそ AM≥ 20）は，ポアソン分布を，平均 m 標準偏差 m の正規分布で近似し，正規乱数としての擬似乱数（切上げ整数）を生成した方がよい（手法概要の［Fn の計算打切りの与える影響について］参照）.

③ 初期値 IX について
本サブルーチンは，一様乱数よりポアソン乱数へ変換することを行っている．パラメタ IX は一様乱数を生成するときの初期値として与えるもので，その扱いは RANU2 の場合と同じである（RANU2 の「使用上の注意」参照）.

④ パラメタ AM を同じ値で使用している間は，VW, IVW の内容を変更してはならない.

c. 使用例
平均 1.0 のポアソン分布から擬似乱数を 10000 個生成し，その頻度分布を示すヒストグラムをプロットする.

```
C     **EXAMPLE**
      DIMENSION IA(10000),VW(12),
     *          IVW(12),HSUM(6)
      INTEGER*4 IX
      DATA X1,X2/-0.5,5.5/
      DATA NINT,XINT/6,1.0/
      DATA AM,IX,N/1.0,0,10000/
      DO 10 I=1,6
   10 HSUM(I)=0.0
      CALL RANP2(AM,IX,IA,N,VW,IVW,ICON)
C     SUM NOS. IN HISTGRAM FORM
      ISW=1
      DO 20 I=1,N
      X=IA(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'POISSON RANDOM',
     * ' NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST については，サブルーチン RANU2 の「使用例」参照のこと.

(4) 手法概要

ボアソン乱数の生成は次のようにして行われる．すなわち，区間(0, 1)での一様分布から生成された擬似乱数 u が (4.1)を満たすように整数 l を決定するとき，l がポアソン乱数となる．

$$F_{l-1} \le u < F_l \tag{4.1}$$

ここで，F_l は (4.2) で示すポアソン累積分布関数である．

$$F_{-1} = 0$$

$$F_l = \sum_{k=0}^{l} p_k = \sum_{k=0}^{l} e^{-m} \frac{m^k}{k!}, \quad l = 0,1,2\cdots \tag{4.2}$$

(4.1) の方法を図 RANP2-2 に示す．図中，例えば $u = 0.84321$ とすれば $l = 2$ として生成される．

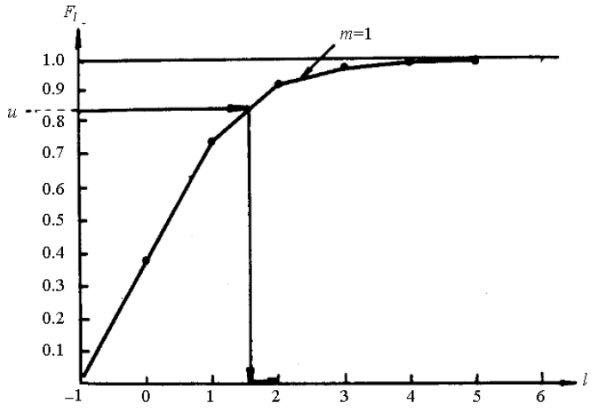


図 RANP2-2 ポアソン累積分布関数 F_l

この累積分布は，m が決まれば一意的に定まるものであるから, 一個の一様乱数に対して毎回(4.2)を計算するのは無駄である．したがって，最初に累積分布表を作成しておけば，後はこの表を参照するだけでよい．パラメタ VW はこのために使用される．更に，もし u が 1 に近い値であるとすると，(4.1)において，l =0, 1, 2, ...と調べてゆくのも時間的に無駄である．パラメタ IVW は，u の値によって l を適当な値から開始するための索引表として使用される．

［F_l の計算打切りの与える影響について］

ポアソン分布の範囲が本来無限である以上，(4.2)を計算するとき，どうしても計算の打切りが必要となる．すなわち，l = 1, 2, 3, ...をどこで打切るかである．このことは(4.2)の演算精度に依存しており，

$F_{l-1} = F_l$

となったときはこれ以上 l を増やしても意味がない．m が 20 になってくると，わずかではあるがこれによる悪影響を受けやすくなるので,この場合，ポアソン分布を正規分布（平均 m，標準偏差 m）で近似し，正規乱数（切上げ整数）として求めた方が，よりよい擬似乱数が得られる．

J11-10-0101 RANU2

一様乱数 (0, 1)の生成
CALL RANU2 (IX, A, N, ICON)

(1) 機能

区間(0,1)の一様分布から，与えられた初期値をもとに，指定された *n* 個の擬似乱数を合同法により生成する．*n*≥ 1 であること.

(2) パラメタ

IX …… **入力**．初期値．非負の整変数（INTEGER*4 であること）．
出力．次の本サブルーチン呼出しのための初期値.
（使用上の注意参照）

A …… **出力**．*n* 個の一様乱数．大きさ *n* の 1 次元配列.

N …… **入力**．生成する一様乱数の個数 *n*.

ICON …… **出力**．コンディションコード.
表 RANU2-1 参照.

表 RANU2-1 コンディションコード

コード	意味	処理内容
0	エラーなし .	
30000	IX < 0 又は N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数 ... DMOD

b. 注意

① 初期値 IX について

(3.1)の合同法により，擬似乱数列（整数）{IX_i} を求めるとき，利用者は初期値として IX_0 を与えなければならない（通常は 0 でよい）.

$$IX_{i+1} \equiv a \times IX_i + c \,(\mathrm{mod}\ m),\ i = 0,1,\cdots,n-1 \qquad (3.1)$$

ここに， IX_i, *a*, *c* 及び *m* はすべて非負の整数である．この数列{IX_i}を (0, 1) に正規化することにより，*n* 個の擬似乱数（実数）をパラメタ A に生成する．擬似乱数生成後，パラメタ IX には IX_n が与えられる．したがって，更に続けて擬似乱数列を生成する場合，パラメタ IX の値を変更しないかぎり，この IX_n がそのまま初期値として使用される.

また，パラメタ N=1 として本サブルーチンの *n* 回連続呼び出しで得た数列{IX_n} は整数一様乱数列としても用いることができる.

② 一様乱数の検定

一様乱数の性質は大きくわけて二つの性質（等出現性，無規則性）があるが，本サブルーチンを使用するにあたってはこれらの性質を把握しておいた方がよい.

IX = 0 を与えたときの本サブルーチンの統計的仮説検定結果を表 RANU2-2 に示す.

通常，擬似乱数の生成において,すべての検定項目を十分満足させることは不可能であるが，本サブルーチンは等出現性の検定にも無規則性の検定にも一応の満足を得られるよう *a*, *c* の値（手法概要参照）を採用している.

表 RANU2-2 RANU2 のχ^2の検定結果

検定項目	サンプルの大きさ*	サンプル数**	有意水準 a%で棄却されなかったサンプル数			備 考
			10%	5%	1%	
1 次元頻度	1000	100	94	99	99	部分等区間の個数 10
	2000	50	45	48	50	〃
	10000	10	10	10	10	〃
1 次元頻度	10000x2	20	18	19	20	部分等区間の個数 10x10
1 次元頻度	5000x3	20	19	19	20	部分等区間の個数 5x5x5
系列相関	1000	100	100	100	100	lag k = 60
	10000	10	10	10	10	lag k = 600
ギャップ	1000	100	68	84	91	ギャップの長さ≥ 8 なるものは一つの階級としてまとめた.
	10000	10	6	6	10	ギャップの長さ≥ 9 なるものは一つの階級としてまとめた.
上昇・下降連	1000	100	91	94	97	連の長さ≥ 4 なるものは一つの階級としてまとめた.
	10000	10	10	10	10	連の長さ≥ 5 なるものは一つの階級としてまとめた.
平均値より上か下かの連	1000	100	92	94	100	連の長さ≥ 7 なるものは一つの階級としてまとめた.
	10000	10	10	10	10	連の長さ≥ 10 なるものは一つの階級としてまとめた.

* 一つのサンプルに含まれる乱数の個数.

** 最初の相続くサンプル系列について検定した.

c. 使用例

区間(0,1)の一様分布から擬似乱数を 10000 個生成し，その頻度分布を示すヒストグラムをプロットする．

```
C     **EXAMPLE**
      DIMENSION A(10000),HSUM(10)
      INTEGER*4 IX
      DATA X1,X2,NINT,XINT/0.0,1.0,10,0.1/
      DATA IX,N/0,10000/
      DO 10 I=1,10
   10 HSUM(I)=0.0
      CALL RANU2(IX,A,N,ICON)
C     SUM NOS. IN HISTGRAM FORM
      ISW=1
      DO 20 I=1,N
      X=A(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'UNIFORM RANDOM ',
     * 'NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST は頻度分布を計算し，そのヒストグラムをプロットするサブルーチンである．以下にその内容を示す．

```
      SUBROUTINE HIST(X,X1,X2,NINT,XINT,
     *               HSUM,ISW)
      DIMENSION HSUM(1)
      CHARACTER*4 IMAGE(31),II,IBLK,IAST
      DATA II,IBLK,IAST/'I   ','    ',
     *     '*   '/
      IF(ISW.NE.1) GO TO 30
C     TO SET UP HISTGRAM FOR RANDOM NOS.
      J=0
      IF(X.GT.(X1+X2)/2.0) J=NINT/2
      BK=X1+J*XINT
   10 J=J+1
      BK=BK+XINT
      IF(X.LT.X1) RETURN
      IF(X.LT.BK) GO TO 20
      IF(X.LT.X2) GO TO 10
      RETURN
   20 HSUM(J)=HSUM(J)+1.0
      RETURN
C     TO GET MAX X FOR PLOTTING
   30 Y=HSUM(1)
      DO 40 I=2,NINT
   40 Y=AMAX1(HSUM(I),Y)
      IMAGE(1)=II
      DO 50 I=2,31
   50 IMAGE(I)=IBLK
      BK=X1-XINT
      WRITE(6,600)
      DO 60 I=1,NINT
      BK=BK+XINT
      WRITE(6,610) BK
      J=30.0*HSUM(I)/Y+1.5
      IMAGE(J)=IAST
      WRITE(6,620) HSUM(I),IMAGE
      IMAGE(J)=IBLK
      IMAGE(1)=II
   60 CONTINUE
      BK=BK+XINT
      WRITE(6,610) BK
      RETURN
  600 FORMAT(2X,'BREAK PT',3X,'PT SUM',
     * 11X,'GRAPH OF DISTRIBUTION')
  610 FORMAT(2X,F5.1,1X,48('-'))
  620 FORMAT(12X,F7.1,5X,31A1)
      END
```

(4) 手法概要

一様乱数を生成させる方法は，現在ではほとんどレーマー(Lehmer)による合同法である．合同法は(4.1)で示される基本的合同関係に基づいている．

$$\mathrm{IX}_{i+1} \equiv a \times \mathrm{IX}_i + c \pmod{m} \qquad (4.1)$$

ここに，IX_i, a, c 及び m はすべて非負の整数である．

この合同法は決定論的なものであって確率的でない．すなわち，数列$\{\mathrm{IX}_i\}$の第 i 項の値をあらかじめ計算することができる．しかし生ずる数列が統計的検定を矛盾なくパスするならば，実用的にはこれを確率的に扱えると考えてよい．

(4.1)において，IX_0, a, c が与えられるならば，数列$\{\mathrm{IX}_i\}$は m を法(modulus)とする剰余の列をつくる．このことはすべての IX_i に対して $\mathrm{IX}_i < m$ を意味する．数列$\{\mathrm{IX}_i\}$から $r_i = \mathrm{IX}_i / m$ として，数列$\{r_i\}$ をつくることによって区間(0,1)の擬似乱数列が得られる．

ここで $\mathrm{IX}_h = \mathrm{IX}_0$ となるような h を数列$\{\mathrm{IX}_i\}$の周期という．これは $\mathrm{IX}_{h+1} = \mathrm{IX}_1$, $\mathrm{IX}_{h+2} = \mathrm{IX}_2$, ... となる事実からきている．

このような h は常に存在し，しかも最大値は m に関係するという定理がある．これは合同法では繰返しのない数列を得ることはできないことを示しているが，実用上は十分大きな法を選ぶことによって周期を大きくすることができる．本サブルーチンでは $m = 2^b$ として $b = 31$ すなわち，

$$m = 2^{31} \qquad (4.2)$$

を採用した．

一方，a の値に対してはグリーンバーガー（Greenberger）の公式を適用している．この公式によると，$m^{1/2}$ に近い a の値は擬似乱数の 1 次系列相関を最小にする．擬似乱数生成法の特性として系列相関が小さいということは極めて望ましい性質であるから(4.2)より，

$$a = 2^{\frac{b}{2}} + 3 = 2^{\frac{31}{2}} + 3 = 32771 \qquad (4.3)$$

とした．また，c 値は m と互いに素である適当な値 $c = 1234567891$ を与えている．

しかし，上記 a, c の値は単に一意的に決定したものではなく，いくつかの候補に対して検定を繰り返すことにより，最良のものを採用した．このことは a, c の値は最終的には経験的な要素が大きく関与していることを示すものである．

なお，詳細については参考文献[89] pp.43-57 を参照すること．

J11-10-0201 RANU3

一様乱数 (0,1)の生成（シャフル型）
CALL RANU3 (IX, A, N, ISW, IVW, ICON)

(1) 機能

区間 (0,1)の一様分布から，与えられた初期値をもとに，指定された n 個の擬似乱数をシャフル型の合同法により生成する． $n \geq 1$ であること．

(2) パラメタ

IX …… **入力**．初期値．非負の整変数（INTEGER*4 であること）．
出力．次の本サブルーチン呼出しのための初期値．
（使用上の注意参照）．

A …… **出力**．n 個の一様乱数．大きさ n の 一次元配列．

N …… **入力**．生成する一様乱数の個数 n．

ISW …… **入力**．初回呼出しのとき 0 を指定する．
2 回目以降のとき 1 を指定する．

IVW …… **作業領域**．大きさ 128 の 1 次元配列（INTEGER*4 であること）．

ICON …… **出力**．コンディションコード．
表 RANU3-1 参照．

表 RANU3-1　コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	IX < 0, N≤0, ISW < 0 又は ISW > 1 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... RANU2, MGSSL

② FORTRAN 基本関数 ... DMOD

b. 注意

① 本サブルーチンは，乱数が持つ二つの性質（等出現性，無規則性）のうち，特に無規則性を高めたものである．したがって，サブルーチン RANU2 に対して，一般に乱数の性質（特に多次元分布の乱数としての性質）が良いが，処理速度は若干遅い．
高速度に乱数を生成する必要のある場合は,サブルーチン RANU2 を使用するとよい．

② 初期値 IX について
(3.1)の合同法により，整数型一様乱数列$\{IX_i\}$を求めるときの，初期値 IX_0 をパラメタ IX に入力する．

$$IX_{i+1} = a \times IX_i + c \pmod m,\ i = 0,1,\ldots \qquad (3.1)$$

ここに，IX_i, a, c 及び m はすべて非負の整数である．乱数発生後，パラメタ IX には最終的な IX_i が与えられる．したがって，更に続けて乱数を発生する場合，そのまま初期値として利用することができる．

③ 乱数の連続発生
本サブルーチンを繰り返し呼び出すことにより，乱数を連続的に発生させることができる．
すなわち，2 回目以降の呼出し時に，ISW = 1 と入力する．この場合，パラメタ IX, IVW の内容は保存したまま呼び出すこと．
もし，2 回目以降 ISW = 0 と入力すると，そのときのパラメタ IX の値を初期値とする別系列の乱数が発生されるので注意すること．

c. 使用例
区間 (0.1)の一様分布から擬似乱数を 10000 個生成し，その頻度分布を示すヒストグラムをプロットする．

```
C     **EXAMPLE**
      DIMENSION A(10000),HSUM(10),IVW(128)
      INTEGER*4 IX,IVW
      DATA X1,X2,NINT,XINT/0.0,1.0,10,0.1/
      DATA IX,N/0,10000/,ISW /0/
      DO 10 I=1,10
   10 HSUM(I)=0.0
      CALL RANU3(IX,A,N,ISW,IVW,ICON)
C     SUM NOS. IN HISTGRAM FORM
      ISW=1
      DO 20 I=1,N
      X=A(I)
   20 CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
C     PLOT DISTRIBUTION
      WRITE(6,600)
      ISW=2
      CALL HIST(X,X1,X2,NINT,XINT,HSUM,
     *          ISW)
      STOP
  600 FORMAT('1',10X,'UNIFORM RANDOM',
     *' NUMBER DISTRIBUTION'//)
      END
```

本使用例中のサブルーチン HIST については，サブルーチン RANU2 の「使用例」参照のこと．

(4) 手法概要

一様乱数の性質には，大きく分けて等出現性と無規則性とがある．ところで，レーマー法

$$IX_{i+1} = a \times IX_i + c \pmod m,\ i = 0,1,2,\ldots \qquad (4.1)$$

による擬似乱数列は，多次元分布において，高次元になればなるほど著しい規則性を示し，擬似乱数としての性質が低下することが知られている．

たとえば，合同式(4.1)から得られる乱数列$\{IX_i\}$を 2 つずつ組み合わせた点 P_0 (IX_0, IX_1), P_1 (IX_1, IX_2) を，xy 座標系内でプロットすると，これらの点はいくつかの平行な直線上に並ぶ．

本サブルーチンでは,レーマー法のこの欠点を補うために,シャフル型の乱数発生方法を採用している.

この方法は，乱数の発生方法自体はレーマー法と同じであるが,乱数列の順番を乱数で決定することにより，無規則性を高めるものである.

乱数列発生の手順は以下のとおりである.

① 基本乱数表の作成
レーマー法により，長さの 80 の順序づけられた整数型一様乱数を発生する．それを

$$\mathrm{IX}_i^B,\ i=1, 2, \cdots\cdots, 80 \qquad (4.2)$$

と表す.

② 乱数列の発生
レーマー法により，引き続く乱数を 1 個発生し，それを IX_0 とする.
以下 l = 1, 2, ..., n と繰り返し，n 個の擬似乱数を発生する.

- l-1 番目の乱数を用いて，基本乱数表から乱数を 1 個取りだし，それを IX_l とする．すなわち,

$$\mathrm{IX}_l = \mathrm{IX}_j^B,\ j = \mathrm{IX}_{l-1} \pmod{80} + 1 \qquad (4.3)$$

この IX_l を区間(0, 1)に正規化し，求める l 番目の擬似一様乱数とし，A(l)に格納する.

- 引き続き l + 1 番目の乱数 IX_{l+1} を発生し，基本乱数表の j 番目に格納する．すなわち,

$$\mathrm{IX}_j^B = \mathrm{IX}_{l+1} \qquad (4.4)$$

J21-10-0101 RATF1

一様乱数 (0, 1)の頻度テスト
CALL RATF1 (A, N, L, ALP, ISW, IFLG, VW, IVW, ICON)

(1) 機能

区間 (0, 1)の，n 個の擬似一様乱数列$\{x_i\}$が与えられたとき，統計的仮説検定に基づく 1 次元頻度テストを行う．

すなわち，“擬似一様乱数は，一様乱数である”という仮説のもとに，区間 (0, 1)を l 個の部分区間に分割し，各区間に落ちる乱数の実現度数と期待度数とからカイ二乗検定を行い，有意水準 α%で仮説が棄却されるか否かを判定する．

なお，乱数列の個数が多い場合は，乱数列を分割し，本サブルーチンを繰り返し呼び出すことにより，継続的に検定することもできる．

ここで，$n \geq l \geq 2$, $100 > \alpha > 0$ であること．

(2) パラメタ

A…… **入力**．擬似一様乱数列$\{x_i\}$．
大きさ n の 1 次元配列．

N…… **入力**．乱数の個数 n．使用上の注意①参照..

L …… **入力**．部分区間の数 l．使用上の注意①参照．

ALP …… **入力**．有意水準α．使用上の注意②参照．

ISW …… **入力**．制御情報．
複数組の乱数列に対して，本サブルーチンを繰り返し呼び出す場合に，継続的に検定を行うか否かを指定する．
ISW = 0 のとき，継続しない．
ISW = 1 のとき，継続する．
ただし，初回目の呼び出し時には，
ISW = 0 とする．使用上の注意③参照．

IFLG…… **出力**．検定結果．
IFLG = 0 : 仮説が棄却されなかった．
IFLG = 1 : 仮説が棄却された．

VW …… **作業領域**．大きさ 2 の 1 次元配列．

IVW…… **作業領域**．大きさ L + 1 の 1 次元配列．（INTEGER*4 であること）．

ICON…… **出力**．コンディションコード．
表 RATF1-1 参照．

表 RATF1-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	期待度数 Fi が小さく，カイ二乗分布の近似が悪い．n を大きくするか，l を小さくするとよい．	処理は続行する．検査結果の信頼性は低い．
30000	N < L, L < 2, ALP ≥ 100, ALP ≤ 0, ISW < 0 又は ISW > 1 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL，UX2UP

② FORTRAN 基本関数 ... SQRT, EXP, ALOG, FLOAT, ABS, ERFC, ATAN

b. 注意

① 乱数の個数 n と部分区間 l の設定基準
検定の信頼性を高めるためには，乱数の個数 n を十分に大きくとることが望ましい．一般には，期待度数 F_i が，

$$F_i\left(=n/l\right) > 10 \qquad (3.1)$$

を満たすことが望ましい．
本サブルーチンでは，(3.1)を満たさない場合は， ICON = 10000 とする．
更に，n が十分に大きい場合，l の値は標準的に，

$$l = \left[1 + 3.322\, log_{10}\, n\right] \qquad (3.2)$$

が適当である．例えば，n = 1000 の時は l = 10, n = 10000 の時は l = 14 となる．
(3.2) は，各部分区間に落ちる乱数の度数分布を視覚的にとらえた場合の妥当性を，経験に基づいて定めた式である．

② 有意水準 αの設定基準
有意水準は，統計的仮説検定の理論より，任意に設定できるが，“仮説が真であるにもかかわらず，これを棄却する過誤（第 1 種の過誤）をおかす確率”が有意水準であるので，1%～10%の値を選択することが望ましい．
標準的には 5%又は 1%が適当である．

③ 継続的検定について
いま m 個の一様乱数列 $\{y_i\}$ が，s 組の乱数列に分割されているとする．すなわち，

$$\{y_i\} = \{y_{i_1}\} + \{y_{i_2}\} + \ldots + \{y_{i_s}\}$$

とし，個々の乱数列に含まれる乱数の個数を $m_1, m_2, \ldots, m_s$ とする．

$$m = m_1 + m_2 + \ldots + m_s$$

本サブルーチンの 1 回の呼出しで，乱数列$\{y_i\}$の検定を行うことができるが，個々の乱数列に対して繰り返し呼び出すことにより，最終的に$\{y_i\}$に対する検定結果を得ることもできる．
後者の場合，呼出しごとの各パラメタの与え方と検定対象の関係を表 RATF1-2 に示す．

表 RATF1-2 継続的検定の方法

呼出し	A	N	ISW	検定対象
1	$\{y_{i1}\}$	m_1	0	$\{y_{i1}\}$
2	$\{y_{i2}\}$	m_2	1	$\{y_{i1}\} + \{y_{i2}\}$
...	...	...	...	...
S	$\{y_{is}\}$	m_s	1	$\{y_{i1}\} + \{y_{i2}\}$ + ... + $\{y_{is}\}$

（注）パラメタ L, ALP の値は一定であること.

一方，呼出しごとに ISW = 0 と指定すると，各乱数列ごとの検定が行われる.
以上のように，繰り返し本サブルーチンを呼び出す場合は，作業領域 VW 及び IVW の内容を保存したまま呼び出すこと.

④ 作業領域の内容
本サブルーチンを実行した後，作業領域には次の値が格納されている.

- VW(1): 自由度 $l - 1$ のχ^2 分布の χ_0^2 における上側確率.
- VW(2): 乱数列$\{x_i\}$の度数分布に対する χ_0^2 値.
- IVW(I): 第 I 番目の区間に落ちた乱数の実現度数．I = 1, 2, ..., L

c. 使用例
10000 個の擬似一様乱数列$\{x_i\}$をサブルーチン RANU2 により生成し，その結果を先頭から 1000 個ずつ計 10 組の乱数列を構成し，各々に対して頻度テストを実施する.
ここで，l = 10, α = 5%とする.

```
C       **EXAMPLE**
        DIMENSION A(10000),VW(2),IVW(11)
        INTEGER*4 IX,IVW
        DATA IX,N/0,1000/,IOK/10/
        DATA L,ALP/10,5.0/,ISW/0/
        NS=10
        LL=L-1
        WRITE(6,600) IX,N,NS,LL,ALP
        IS=1
        IE=N
        DO 20 I=1,NS
        CALL RANU2(IX,A,N,ICON)
        CALL RATF1(A,N,L,ALP,ISW,IFLG,VW,
       *           IVW,ICON)
        IF(ICON.EQ.0) GOTO 10
        WRITE(6,610) I,IS,IE
        IOK=IOK-1
        GOTO 15
   10   IOK=IOK-IFLG
        IF(IFLG.EQ.0) WRITE(6,620) I,IS,IE
        IF(IFLG.EQ.1) WRITE(6,630) I,IS,IE
   15   IS=IE+1
   20   IE=IE+N
        RATE=IOK*100.0/NS
        WRITE(6,640) IOK,RATE
        STOP
  600   FORMAT('1',60('*')/6X,
       *'FREQUENCY TEST FOR UNIFORM',
       *' RANDOM NUMBERS.'/1X,60('*')//
       *6X,'INITIAL VALUE    IX=',I5/
       *6X,'SAMPLING LENGTH   N=',I5/
       *6X,'SAMPLE NUMBER      =',I5/
       *6X,'FREE DEGREE     L-1=',I5/
       *6X,'SIGNIFICANCE LEVEL =',F5.1,'%'/
       *6X,'RESULTANTS :'//
       *9X,'NO.   SAMPLE    JUDGEMENT')
  610   FORMAT(9X,I2,I7,'-',I5,4X,'ERROR')
  620   FORMAT(9X,I2,I7,'-',I5,4X,'SAFE')
  630   FORMAT(9X,I2,I7,'-',I5,4X,'FAIL')
  640   FORMAT(//6X,'TOTAL CONCLUSION:'//
       *9X,I2,' (',F5.1,
       *' %) SAMPLES ARE SAFE.')
        END
```

(4) 手法概要

区間 (0, 1) を l 個の部分区間に等分割し，検定すべき n 個の擬似乱数のうち，i 番目の区間に落ちる個数を f_i とする．更に，f_i に対する一様分布の期待度数を F_i とすれば,

$$\chi_0^2 = \sum_{i=1}^{l} \frac{(f_i - F_i)^2}{F_i} \tag{4.1}$$

なる値は，多くの標本$\{x_i\}$に関して自由度 l-1 のカイ二乗分布をなす.

そこで本サブルーチンでは，(4.1)より χ_0^2 値を求め，有意水準 α%として，帰無仮説“ n 個の擬似乱数は一様乱数である．”を検定する.

(4.1)において期待度数 F_i は

$$F_i = n / l \tag{4.2}$$

である.

なお，詳細については参考文献 [93]を参照すること.

J21-10-0201 RATR1

一様乱数 (0.1) の上昇・下降連テスト
CALL RATR1 (A, N, L, ALP, ISW, IFLG, VW, IVW, ICON)

(1) 機能

区間 (0, 1)の，n 個の擬似一様乱数列$\{x_i\}$が与えられたとき，統計的仮説検定に基づく上昇・下降連テストを行う．

すなわち，“擬似一様乱数は，一様乱数である”という仮説のもとに，長さ r (高々 l, $1 \le r \le l$, l 以上のものは l に含める) の連の実現度数と期待度数とからカイ二乗検定を行い，有意水準 α%で仮説が棄却されるか否かを判定する．

ここで，長さ r の連とは，乱数列の中で連続した r 個の要素が単調増加（又は減少）する部分列である．

なお，乱数列の個数が多い場合は，乱数列を分割し本サブルーチンを繰り返し呼び出すことにより，継続的に検定することもできる．

ここで， $n \ge l+2$, $l \ge 2$, $100 > \alpha > 0$ であること．

(2) パラメタ

A…… **入力**．擬似一様乱数列$\{x_i\}$．
大きさ n の 1 次元配列．

N…… **入力**．乱数の個数 n．

L …… **入力**．最大の連の長さ l．使用上の注意①参照

ALP …… **入力**．有意水準α．使用上の注意②参照．

ISW …… **入力**．制御情報
複数組の乱数列に対して，本サブルーチンを繰り返し呼び出す場合に，継続的に検定を行うか否かを指定する．
ISW = 0 のとき，継続しない．
ISW = 1 のとき，継続する．
ただし，初回目の呼び出し時には，
ISW = 0 とする．
使用上の注意③参照．

IFLG…… **出力**．検定結果．
IFLG = 0 : 仮説が棄却されなかった．
IFLG = 1 : 仮説が棄却された．

VW …… **作業領域**．大きさ 3 の 1 次元配列．

IVW…… **作業領域**．大きさ L + 8 の 1 次元配列．
（INTEGER*4 であること）

ICON…… **出力**．コンディションコード．
表 RATR1-1 参照．

表 RATR1-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	期待度数 Fi の小さいものがあり，カイ二乗分布の近似が悪い． n を大きくするか，l を小さくするとよい．	処理は続行する． 検査結果の信頼性は低い．
30000	N < L+2, L < 2, ALP ≥ 100, ALP ≤ 0, ISW < 0 又は ISW > 1 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, UX2UP

② FORTRAN 基本関数 ... SQRT, EXP, ALOG, FLOAT, ATAN, ERFC, MAX0, MIN0

b. 注意

① 最大の連の長さ l の設定基準
連の長さが r となる期待度数 F_r は，

$$F_r = 2n\frac{r^2+3r+1}{(r+3)!} - 2\frac{r^3+3r^2-r-4}{(r+3)!} \qquad (3.1)$$
$$,r = 1,2,\cdots,n-2$$

で表される．すなわち，

$$F_{r+1} \approx F_r/(r+1)$$

程度に減少していく．例えば，n = 10000 の場合，F_1 = 4160 に対して F_2 = 1834, ..., F_5 = 20, F_6 = 3 となる．したがって，この場合，
l = 5
程度にとるのが適当である．
本サブルーチンでは,期待度数 F_i が
$F_i > 10$
を満たさない場合は ICON = 10000 とする．

② 有意水準 αの設定基準
有意水準は，統計的仮説検定の理論より，任意に設定できるが，“仮説が真であるにもかかわらず，これを棄却する過誤（第 1 種の過誤）をおかす確率”が有意水準であるので，1%～10%の値を選択することが望ましい．
標準的には 5%又は 1%が適当である．

③ 継続的検定について
いま m 個の一様乱数列 $\{y_i\}$ が，s 組の乱数列に分割されているとする．すなわち，

$\{y_i\} = \{y_{i1}\} + \{y_{i2}\} + ... + \{y_{is}\}$

とし，個々の乱数列に含まれる乱数の個数を，$m_1, m_2, ..., m_s$ とする．

$m = m_1 + m_2 + ... + m_s$

本サブルーチンの 1 回の呼出しで，乱数列$\{y_i\}$の検定を行うことができるが，個々の乱数列に対して繰り返し呼び出すことにより，最終的に$\{y_i\}$に対する検定結果を得ることもできる．
後者の場合，呼出しごとの各パラメタの与え方と検定対象の関係を表 RATF1-2 に示す．

表 RATR1-2 継続的検定の方法

呼出し	A	N	ISW	検定対象
1	$\{y_{i1}\}$	m_1	0	$\{y_{i1}\}$
2	$\{y_{i2}\}$	m_2	1	$\{y_{i1}\} + \{y_{i2}\}$
…	…	…	…	…
S	$\{y_{is}\}$	m_s	1	$\{y_{i1}\} + \{y_{i2}\} + \ldots + \{y_{is}\}$

（注） パラメタ L, ALP の値は一定であること．

一方，呼出しごとに ISW = 0 と指定すると，各乱数列ごとの検定が行われる．
以上のように，繰り返し本サブルーチンを呼び出す場合は，作業領域 VW 及び IVW の内容を保存したまま呼び出すこと．

④ 作業領域の内容
本サブルーチンを実行した後，作業領域には次の値が格納されている．

- VW(1): 自由度 $l - 1$ のχ^2 分布の χ_0^2 における上側確率．
- VW(2): 乱数列$\{x_i\}$の度数分布に対する χ_0^2 値．
- IVW(I): 長さが I である連の実現度数．I = 1, 2, ..., L

c. 使用例
1000 個の擬似一様乱数列$\{x_i\}$をサブルーチン RANU2 により生成し，その結果を先頭から 1000 個ずつ計 10 組の乱数列を構成し，各々に対して上昇・下降連テストを実施する．
ここで，$l = 4$, $\alpha = 5\%$とする．

```
C       **EXAMPLE**
        DIMENSION A(1000),VW(3),IVW(12)
        DATA IX,N/0,1000/,IOK/10/
        DATA L,ALP/4,5.0/,ISW/0/
        NS=10
        LL=L-1
        WRITE(6,600) IX,N,NS,LL,ALP
        IS=1
        IE=N
        DO 20 I=1,NS
        CALL RANU2(IX,A,N,ICON)
        CALL RATR1(A,N,L,ALP,ISW,IFLG,VW,
       *           IVW,ICON)
        IF(ICON.EQ.0) GOTO 10
        WRITE(6,610) I,IS,IE
        IOK=IOK-1
        GOTO 15
   10 IOK=IOK-IFLG
      IF(IFLG.EQ.0) WRITE(6,620) I,IS,IE
      IF(IFLG.EQ.1) WRITE(6,630) I,IS,IE
   15 IS=IE+1
   20 IE=IE+N
      RATE=IOK*100.0/NS
      WRITE(6,640) IOK,RATE
      STOP
  600 FORMAT('1',60('*')/6X,
     *'RUNS TEST FOR UNIFORM',
     *' RANDOM NUMBERS.'/1X,60('*')//
     *6X,'INITIAL VALUE    IX=',I5/
     *6X,'SAMPLING LENGTH   N=',I5/
     *6X,'SAMPLE NUMBER      =',I5/
     *6X,'FREE DEGREE     L-1=',I5/
     *6X,'SIGNIFICANCE LEVEL =',F5.1,'%'/
     *6X,'RESULTANTS :'//
     *9X,'NO.   SAMPLE    JUDGEMENT')
  610 FORMAT(9X,I2,I7,'-',I5,4X,'ERROR')
  620 FORMAT(9X,I2,I7,'-',I5,4X,'SAFE')
  630 FORMAT(9X,I2,I7,'-',I5,4X,'FAIL')
  640 FORMAT(//6X,'TOTAL CONCLUSION:'//
     *9X,I2,' (',F5.1,
     *' %) SAMPLES ARE SAFE.')
      END
```

(4) 手法概要

検定すべき n 個の擬似乱数列 $\{x_i\}$における，長さ r の上昇連とは，$\{x_i\}$の連続した r 個の要素からなる部分列があって，その先頭要素を $x_k\,(1 \le k < n)$ としたとき，

$$\cdots\cdots x_{k-1} > x_k < x_{k+1} < \cdots < x_{k+r} > x_{k+r+1} \cdots$$

を満たすものをいう．ただし，$k = 1$ 又は $k + r = n$ のとき端の不等号は考えないものとする．

一方，下降連についても同様に定義する．

そこで，$\{x_i\}$において，長さ r(高々 l, $1 \le r \le l$) の連の個数を f_r と表す．ただし，f_l は長さが l 以上の連の個数の総和とする．一方，長さ r の期待度数を F_r とすると，

$$\chi_0^2 = \sum_{i=1}^{l} \frac{(f_i - F_i)^2}{F_i} \tag{4.1}$$

なる値は，多くの標本$\{x_i\}$に関して自由度 l-1 のカイ二乗分布をなす．

そこで本サブルーチンでは，(4.1)より χ_0^2 値を求め，有意水準 α%として，帰無仮説“ n 個の擬似乱数は一様乱数である．”を検定する．

(4.1)において期待度数 F_r は

$$F_r = 2n\frac{r^2 + 3r + 1}{(r+3)!} - 2\frac{r^3 + 3r^2 - r - 4}{(r+3)!} \tag{4.2}$$

$$, r = 1, 2, \ldots, n-2$$

であり，その総和は

$$\sum_r F_r = (2n-7)/3 \tag{4.3}$$

となることが知られている．
ところで，カイ二乗検定では，

$$\sum_r f_r = \sum_r F_r \tag{4.4}$$

となることを前提としているが，実際には(4.4)が満たされないのが普通である．

そこで本サブルーチンでは，期待度数 F_r を実限度数 f_r により修正した値 F^*_r

$$F_r^* = \sum_r f_r \frac{F_r}{\sum_r F_r} \tag{4.5}$$

を採用することにより，検定の精度を高めている．

なお，詳細については参考文献 [93]を参照すること．

F11-31-0101 RFT, DRFT

離散型実フーリエ変換
CALL RFT (A, N, ISN, ICON)

(1) 機能

1 次元（項数 n）の実時系列データ$\{x_j\}$が与えられたとき，離散型実フーリエ変換，又はその逆変換を高速変換手法 (FFT)により行う.

ただし，項数 $n = 2^l$ (l: 正整数)であること.

① フーリエ変換

$\{x_j\}$を入力し，(1.1)で定義する変換を行い，フーリエ係数$\{na_k\}$, $\{nb_k\}$を求める.

$$\begin{aligned} na_k &= 2\sum_{j=0}^{n-1} x_j \cos\frac{2\pi kj}{n}, k = 0,\ldots,\frac{n}{2} \\ nb_k &= 2\sum_{j=0}^{n-1} x_j \sin\frac{2\pi kj}{n}, k = 1,\ldots,\frac{n}{2}-1 \end{aligned} \qquad (1.1)$$

② フーリエ逆変換

$\{a_k\}$, $\{b_k\}$ を入力し，(1.2)で定義する変換行い，フーリエ級数の値$\{2x_j\}$を求める.

$$\begin{aligned} 2x_j = a_0 &+ 2\sum_{k=1}^{n/2-1}\left(a_k\cos\frac{2\pi kj}{n} + b_k\sin\frac{2\pi kj}{n}\right) \\ &+ a_{n/2}\cos\pi j \\ j&=0,\ldots,n-1 \end{aligned} \qquad (1.2)$$

(2) パラメタ

A …… **入力**. $\{x_j\}$又は$\{a_k\}$, $\{b_k\}$.
出力. $\{na_k\}$, $\{nb_k\}$又は$\{2x_j\}$.
大きさ n の 1 次元配列.
図 RFT-1 参照.

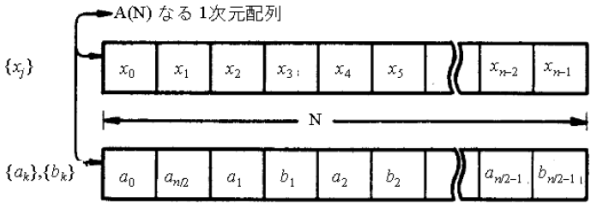
N …… **入力**. 変換の項数 n.

ISN …… **入力**. 変換か逆変換かを指定する ($\neq 0$).
変換: ISN = + 1
逆変換: ISN = - 1
（使用上の注意②参照）.

ICON …… **出力**. コンディションコード.
表 RFT-1 参照.

表 RFT-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	ISN = 0 又は N ≠ 2^l (l: 正整数) であった.	処理を打ち切る.



［注意］$\{na_k\}$, $\{nb_k\}$は，$\{a_k\}$, $\{b_k\}$に準じる.

図 RFT-1 データの格納方法

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... CFTN, PNR, MGSSL, URFT

② FORTRAN 基本関数 ... ATAN, ALOG, SQRT, SIN, IABS

b. 注意

① 一般的なフーリエ変換の定義について

離散型実フーリエ変換及び，逆変換は一般的に (3.1), (3.2) で定義される.

$$\left.\begin{aligned} a_k &= \frac{2}{n}\sum_{j=0}^{n-1} x_j\cos\frac{2\pi kj}{n}, \quad k = 0,\ldots,\frac{n}{2} \\ b_k &= \frac{2}{n}\sum_{j=0}^{n-1} x_j\sin\frac{2\pi kj}{n}, \quad k = 1,\ldots,\frac{n}{2}-1 \end{aligned}\right\} \qquad (3.1)$$

$$\begin{aligned} x_j = \frac{1}{2}a_0 &+ \sum_{k=1}^{n/2-1}\left(a_k\cos\frac{2\pi kj}{n} + b_k\sin\frac{2\pi kj}{n}\right) \\ &+ \frac{1}{2}a_{n/2}\cos\pi j, \quad j = 0,\ldots,n-1 \end{aligned} \qquad (3.2)$$

本サブルーチンでは，(3.1), (3.2)の左辺に対応して，$\{na_k\}$, $\{nb_k\}$又は，$\{2x_j\}$を求めるので結果の正規化は必要に応じて行うこと.

ちなみに本サブルーチンにより変換・逆変換を正規化せずに連続して実行すると，入力データの各要素は $2n$ 倍されて出力される.

② ISN の与え方について

ISN では，変換か逆変換かを指定するが，更に次のように使い分けることができる.

すなわち，$\{x_j\}$又は，$\{a_k\}$, $\{b_k\}$が $N{\cdot}I$ なる大きさの領域に間隔 I で格納されている場合は，次のように指定する.

変換: ISN = + I
逆変換: ISN = - I
この場合，変換結果も間隔 I で格納される.

c. 使用例

n 項の実時系列データ$\{x_j\}$を入力して，本サブルーチンにより変換したのち正規化 $\{a_k\}$, $\{b_k\}$を求める．$n \le 1024$ (= 2^{10}) の場合.

```
C     **EXAMPLE**
      DIMENSION A(1024)
      READ(5,500) N,(A(I),I=1,N)
      WRITE(6,600) N,(I,A(I),I=1,N)
      ISN=1
      CALL RFT(A,N,ISN,ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,620) A(1)
      N1=N/2-1
      IF(N1.NE.0)
     * WRITE(6,630) (I,A(2*I+1),A(2*I+2),
     *              I=1,N1)
      N1=N1+1
      WRITE(6,630) N1,A(2)
      STOP
  500 FORMAT(I5/(2E20.7))
  600 FORMAT('0',10X,'INPUT DATA N=',I5/
     * /(15X,I5,E20.7))
  610 FORMAT('0',10X,'RESULT ICON=',I5/)
  620 FORMAT('0',17X,'K',10X,'A(K)',16X,
     * 'B(K)'//19X,'0',E20.7)
  630 FORMAT(/15X,I5,2E20.7)
      END
```

(4) 手法概要

項数 n (=2·m) の離散型実フーリエ変換（以下，実変換という）を，8 及び 2 を基底とする高速変換手法 (FFT)により行う．

ところで実変換は，変換の対象となる実データ$\{x_j\}$を虚部が零なる複素データとみなし，それに対して離散型複素フーリエ変換（以下，複素変換という）を行えば得られる．しかし，この場合，複素変換の性質を利用すると効率よく変換が可能であることが知られている．すなわち，複素変換を(4.1)で定義すると，

$$\alpha_k = \sum_{j=0}^{n-1} x_j \exp\left(-2\pi i \frac{kj}{n}\right), \quad k = 0,\ldots,n-1 \qquad (4.1)$$

$\{x_j\}$が実データの場合，(4.2)なる共役関係があり，それを利用するわけである．

$$\alpha_{n-k} = \alpha_k^*, \quad k = 1,\ldots n-1 \qquad (4.2)$$

* は複素共役を表す．

ところで，実変換の結果の$\{a_k\}$, $\{b_k\}$と，複素変換による結果の$\{\alpha_k\}$との関連は(4.3)で示される．

$$\left.\begin{aligned} a_0 &= 2\cdot\alpha_0,\ a_{n/2} = 2\cdot\alpha_{n/2} \\ a_k &= (\alpha_k + \alpha_{n-k}), k = 1,\ldots, n/2-1 \\ b_k &= i(\alpha_k - \alpha_{n-k}), k = 1,\ldots n/2-1 \end{aligned}\right\} \qquad (4.3)$$

したがって実変換を複素変換により行う場合には，(4.2), (4.3)より，a_k, k = 0, ..., n/2 だけを求めれば，$\{a_k\}$, $\{b_k\}$を求められることがわかる．

本サブルーチンでは，この冗長度を考慮しつつ，高速変換手法(FFT) を用いた複素変換を利用することにより，実変換を行う．

以下にその方法を述べる．

a. 複素変換を利用する実変換の方法

まず，複素変換 (4.1)に高速変換手法 (FFT)の原理を適用し，展開することを考える．すなわち，$n = 2\cdot m$ であるから，k, j は(4.4)で表せる．

$$\begin{aligned} k &= k_0 + k_1\cdot m, 0 \le k_0 \le m-1,\ 0 \le k_1 \le 1 \\ j &= j_0 + j_1\cdot 2, 0 \le j_0 \le 1,\ 0 \le j_1 \le m-1 \end{aligned} \qquad (4.4)$$

(4.4)を(4.1)に代入し，共通項を整理すると (4.5)となる．

$$\alpha_{k_0+k_1\cdot m} = \sum_{j0=0}^{1} \exp\left(-2\pi i \frac{j_0 k_1}{2}\right) \exp\left(-2\pi i \frac{j_0 k_0}{n}\right) \cdot \sum_{j_1=0}^{m-1} \exp\left(-2\pi i \frac{j_0 k_0}{m}\right) x_{j_0+j_{1\cdot 2}} \qquad (4.5)$$

(4.5)は，$\sum_{j_1}$ により m 項の，また$\sum_{j_0}$ により 2 項の部分的な複素変換を各々複数組ずつ行うことにより(4.1)なる複素変換が達成されることを意味する．（高速変換手法の原理については CFT の項を参照すること．）ここでは，虚部が零なる複素データに対して，(4.5)を逐次計算していくものとする．

(a) $\sum_{j_1}$ による変換

$\sum_{j_1}$ による m 項の部分的な複素変換は，j_0 に関して 2 組行う．すなわち，$\{x_j\}$の偶数番目のデータ$\{x1_j\}$と奇数番目のデータ$\{x2_j\}$に対して各々複素変換し，$\{\alpha_k^{x1}\}$, $\{\alpha_k^{x2}\}$を求めればよい．ところで，$\{x1_j\}$, $\{x2_j\}$は各々虚部が零なる複素データであるから，(4.2)と同様に,(4.6)なる共役関係がある．

$$\left.\begin{aligned} \alpha_{m-k}^{x1} &= (\alpha_k^{x1})^* \\ \alpha_{m-k}^{x2} &= (\alpha_k^{x2})^* \end{aligned}\right\}, k = 1,\ldots, m-1 \qquad (4.6)$$

この 2 組の複素変換は，次のように 1 組の複素変換と若干の操作により行う．

すなわち $\{x1_j\}$, $\{x2_j\}$の実部の対を (4.7)のように新たな複素データ$\{z_j\}$の実部・虚部と見なし，

$$z_j = x1_j + i\cdot x2_j, j = 0,\ldots,m-1 \qquad (4.7)$$

この$\{z_j\}$に対して m 項の複素変換を行い，$\{\alpha_k^z\}$を求める．(4.8)により，この$\{\alpha_k^z\}$から，$\{\alpha_k^{x1}\}$, $\{\alpha_k^{x2}\}$を求める．

$$\left.\begin{aligned}\alpha_k^{x1} &= \frac{1}{2}\left(\alpha_k^z + \left(\alpha_{m-k}^z\right)^*\right)\\ \alpha_k^{x2} &= \frac{1}{2i}\left(\alpha_k^z - \left(\alpha_{m-k}^z\right)^*\right)\end{aligned}\right\}, k = 1,\ldots,\frac{m}{2}\Bigg\} \quad (4.8)$$
$$\alpha_0^{x1} = \mathrm{Re}\left(\alpha_0^z\right), \alpha_0^{x2} = \mathrm{Im}\left(\alpha_0^z\right)$$

(b) $\sum_{j0}$ による変換

$\sum_{j0}$ による 2 項の部分的な複素変換は，j_1 に関して m 組行う．その結果は虚部が零なる複素データに対する複素変換の結果の $\{\alpha_k\}$である．ところで，$\{a_k\}$, $\{b_k\}$を求めるには，α_k , k = 0, ..., n/2 を求めればよいわけで，$\sum_{j0}$ による 2 項の複素変換においては，共役部分の計算を除く．

b. 本サブルーチンにおける手順

以下に本サブルーチンにおける実変換の手順をのべるが，n=16 の場合の流れ図が，図 RFT-2 に示されるので，共に参考にすること．

(a) n 項の実データのうち，偶数番目を$\{x1_j\}$, 奇数番目を$\{x2_j\}$ とし，(4.7) なる$\{z_j\}$ に対して m 項の複素変換を行い，$\{a_k\}$を求める．

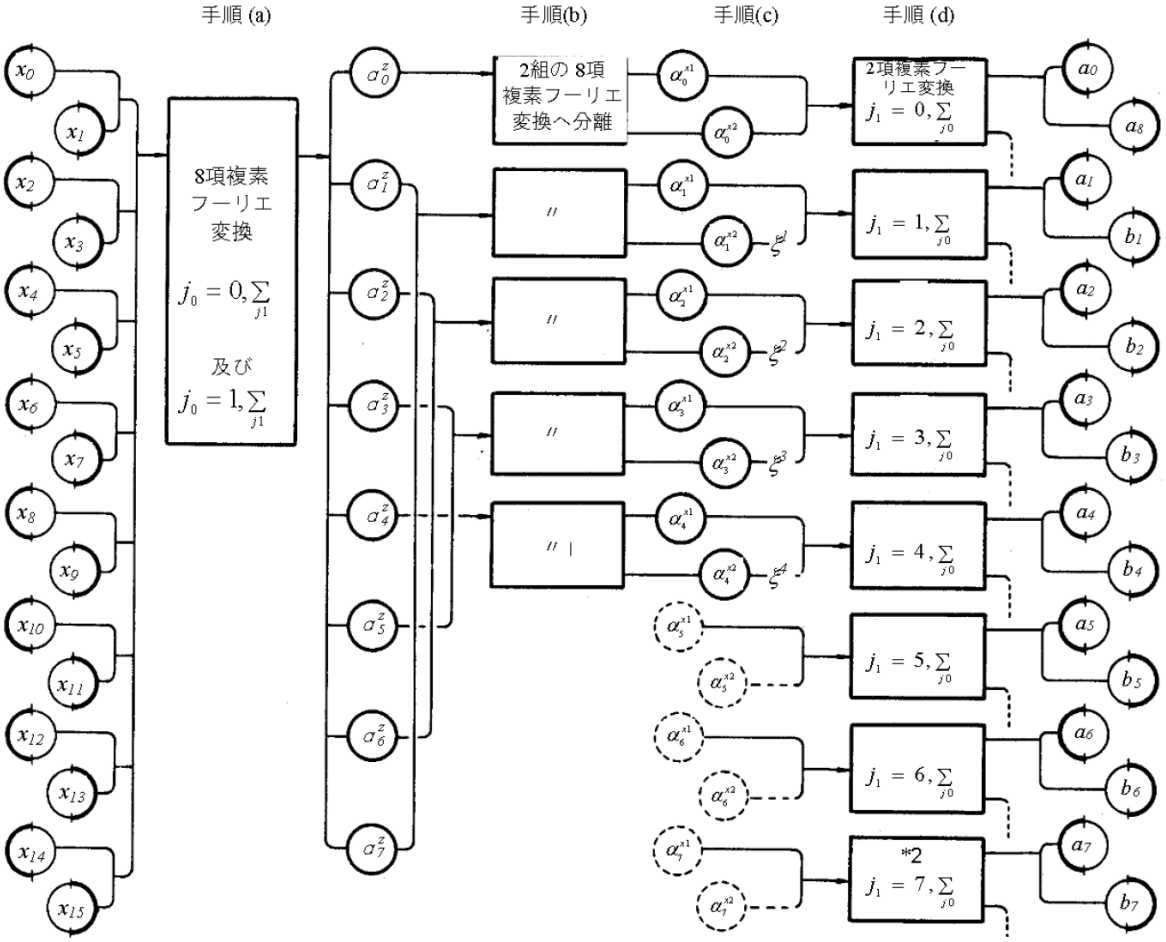
(b) (4.8)により，$\{\alpha_k^z\}$ から$\{x1_j\}$, $\{x2_j\}$に対する m 項の複素変換の結果$\{\alpha_k^{x1}\}$, $\{\alpha_k^{x2}\}$を求める．

(c) $\{\alpha_k^{x2}\}$ に対して，回転因子を掛ける．

(d) 2 項の複素変換を行う．

(e) 本サブルーチンでは，(a)の複素変換はサブルーチン CFTN, PNR により行う．

なお，詳細についていは，参考文献[55], [56]及び，[57]を参照すること．



［注意］ ○ 印は複素データ，◯印又は ○ 印は実データを示す．また，
◌ 印又は .... 印は，実質的な演算は不要であることを示す．
ξは回転因子であり，$\xi = \exp(-2\pi i/16)$である．

図 RTF-2 16 項実フーリエ変換の流れ図

C22-11-0111 RJETR, DRJETR

実係数高次代数方程式 （ジェンキンス・トラウブの方法）
CALL RJETR (A, N, Z, VW, ICON)

(1) 機能

実係数高次代数方程式

$$a_0 x^n + a_1 x^{n-1} + \cdots\cdots + a_n = 0$$
$$\left(a_i : \text{実数}, a_0 \neq 0, n \geq 1\right)$$

の根をジェンキンス･トラウブの 3 段アルゴリズムにより求める.

(2) パラメタ

A …… **入力**. 代数方程式の係数.
大きさ $n+1$ の 1 次元配列.
$A(1) = a_0$, $A(2) = a_1$, ..., $A(N+1) = a_n$ の順に与える.
演算後内容は保存されない.

N …… **入力**. 代数方程式の次数 n.
出力. 求まった根の個数.
（使用上の注意参照）.

Z …… **出力**. n 個の根. 大きさ n の複素数型 1 次元配列. 根は求まった順に Z(1), Z(2) ... に出力される.
したがって求まった根の個数を N とするとそれらの根は Z(1) から Z(N)までにセットされる.

VW... …… **作業領域**. 大きさ $6(n+1)$の 1 次元配列.

ICON.. …. **出力**. コンディションコード.
表 RJETR-1 参照.

表 RJETR-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
10000	n 根を，すべては求めつくすことができなかった.	決まった根の個数をパラメタ N に，そして，根を Z(1)～Z(N)に出力する.
30000	$n < 1$ 又は $a_0 = 0$ であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL, AMACH, RQDR, IRADIX, AFMAX, AFMIN, UJET

② FORTRAN 基本関数 ... ABS, ALOG, EXP, SQRT, CMPLX, SIN, ATAN, REAL, AIMAG, AMAX1

b. 注意

① 配列 Z は本サブルーチンを呼び出す側のプログラムにおいて COMPLEX 宣言をすること.

② n 次代数方程式は n 個の根を持つが，ごくまれに必ずしもそのすべてを求めることができないことがある. 利用者はこのことに注意し，パラメタ ICON あるいは N の演算後の値を確認すること.

c. 使用例
次数 n，実係数 a_i を入力し,根を求める.
$1 \leq n \leq 50$ の場合.

```
C       **EXAMPLE**
        DIMENSION A(51),Z(50),VW(306)
        COMPLEX Z
        READ(5,500) N
        N1=N+1
        READ(5,510) (A(I),I=1,N1)
        DO 10 I=1,N1
        K=I-1
   10   WRITE(6,600) K,A(I)
        CALL RJETR(A,N,Z,VW,ICON)
        WRITE(6,610) N,ICON
        IF(ICON.EQ.30000) STOP
        WRITE(6,620) (I,Z(I),I=1,N)
        STOP
  500   FORMAT(I2)
  510   FORMAT(5F10.0)
  600   FORMAT(10X,'A(',I2,')=',E20.8)
  610   FORMAT(10X,'N=',I2,5X,'ICON=',I5)
  620   FORMAT(10X,'Z(',I2,')=',2E20.8)
        END
```

(4) 手法概要

本サブルーチンは，ジェンキンス(M.A.Jenkins) とトラウプ(J.F.Traub) による 3 段アルゴリズムを用いている.

3 段アルゴリズムとは,各々三つの段階で異なった列として定義される K 多項式（後述）を用いて，絶対値の小さい方の根から得られるように注意を払いながら根を追求し（第 1 段階），収束の状況に入ったと確認したら（第 2 段階），収束を速めて根を得る（第 3 段階）手順である. 特に，与えられた方程式が実係数の場合は,根を 1 次又は 2 次因子として求めるため，第 2 段階で 1 次因子へ収束しているか，2 次因子へ収束しているかの判定を行い,それに応じて第 3 段階は二つの場合に分けて収束を速めている. 2 次因子が求まったときは，2 次方程式の根の公式により根を求めている.

特徴

a. 複素根はその共役根とともに 2 次因子として計算するため，複素演算を用いていない.

b. 根は比較的絶対値の小さい方から計算されるので，次数低下に伴う不安定性を避けることができる.

c. 第 3 段階での収束率は 2 次よりも速い.

K 多項式

アルゴリズムを説明する前に，ここで二つの重要な多項式を導入し，その性質を明らかにしておく.

与えられた方程式を

$$f(x) \equiv a_0 x^n + a_1 x^{n-1} + \cdots\cdots + a_n = \prod_{i=1}^{j} (x-\rho_i)^{mi} = 0, \rho_i \neq \rho_k (i \neq k) \quad (4.1)$$

とする．ここで $a_0 = 1$, $a_n \neq 0$ と仮定するが，それによって一般性が失われることはない．

さて，$(n-1)$ 次の任意の多項式 $K^{(0)}(x)$ から出発し，λ= 0, 1, ... に対して，$K^{(\lambda)}(x)$ を次のように定義する．

$$K^{(\lambda+1)}(x) = \frac{1}{x}\left[K^{(\lambda)}(x) - \frac{K^{(\lambda)}(0)}{f(0)} f(x) \right] \quad (4.2)$$

この式から明らかなように,すべての $K^{(\lambda)}(x)$は，たかだか $(n-1)$次の多項式である．

いま，$K^{(0)}(x) = \sum_{i=1}^{j} c_i f_i(x)$, ただし $f_i(x) = \dfrac{f(x)}{x-\rho_i}$ (4.3)

とすると，$K^{(\lambda)}(x)$ は次のようになる．

$$K^{(\lambda)}(x) = \sum_{i=1}^{j} c_i \rho_i^{-\lambda} f_i(x) \quad (4.4)$$

(4.4)式より，もし $|\rho_1| < |\rho_i|$ $(i \geq 2)$ かつ $c \neq 0$ となる ρ_1があると

$$\lim_{\lambda\to\infty} f(x) / \overline{K}^{(\lambda)}(x) = x - \rho_1 \quad (4.5)$$

が成り立つ．ただし $\overline{K}^{(\lambda)}(x)$ は $K^{(\lambda)}(x)$ をその最高次の係数で割った多項式である．ここで(4.5) の収束の速さは ρ_1 と ρ_i $(i \geq 2)$ の比に依存していることもわかる．(4.2) で定義された多項式をシフトなし多項式(no-sift polynomials)と呼ぶ．

第 2 の多項式は次のように定義する．$(n-1)$ 次の多項式 $K^{(0)}(x)$ から出発し，λ = 0, 1, ... に対し (4.2) とは異なって次のように定義する．

$$K^{(\lambda+1)}(x) = \frac{1}{\sigma(x)}\left[K^{(\lambda)}(x) + \left(A^{(\lambda)} x + B^{(\lambda)} \right) f(x) \right] \quad (4.6)$$

ここで$\sigma(x)$ 2 次式 $x^2 + ux + v$ であり，その 2 根 s_1 , s_2 は $s_1 \neq s_2$, $|s_1| = |s_2| = \beta$, $\beta \leq \min |\rho_i|$, $f(s_1)\, f(s_2) \neq 0$ を満たすとする．また $A^{(\lambda)}$, $B^{(\lambda)}$ は (4.6)の[ ] の中が $\sigma(x)$ で割り切れるように選ばれ，それぞれ

$$A^{(\lambda)} = \begin{vmatrix} f(s_1) & f(s_2) \\ K^{(\lambda)}(s_1) & K^{(\lambda)}(s_2) \end{vmatrix} \Bigg/ \begin{vmatrix} s_1 f(s_1) & s_2 f(s_2) \\ f(s_1) & f(s_2) \end{vmatrix}$$

$$B^{(\lambda)} = \begin{vmatrix} K^{(\lambda)}(s_1) & K^{(\lambda)}(s_2) \\ s_1 f(s_1) & s_2 f(s_2) \end{vmatrix} \Bigg/ \begin{vmatrix} s_1 f(s_1) & s_2 f(s_2) \\ f(s_1) & f(s_2) \end{vmatrix} \quad (4.7)$$

である．このとき(4.6) のすべての $K^{(\lambda)}(x)$ は，たかだか $(n-1)$ 次の多項式となる．$K^{(0)}(x)$を前と同様に (4.3) で表わすと (4.6) の多項式は次のように書ける．

$$K^{(\lambda)}(x) = \sum_{i=1}^{j} c_i \sigma_i^{-\lambda} f_i(x), \sigma_i = \sigma(\rho_i) \quad (4.8)$$

(4.6)で定義された多項式を固定シフト多項式(fixed-shift polynomials)と呼ぶ．この固定シフト多項式は次に示す二つの重要な性質を持っている．

a. もし，ある σ(x) に対して

$$|\sigma_1| < |\sigma_i|, i \geq 2 \quad (4.9)$$

となる ρ_1 があれば，ρ_1 は実数であり，かつ

$$\lim_{\lambda\to\infty} f(x) / \overline{K}^{(\lambda)}(x) = x - \rho_1 \quad (4.10)$$

が成り立つ．

ここで $K^{(\lambda)}(x)$は(4.8)の $K^{(\lambda)}(x)$をその最高次の係数で割った多項式である．

b. もし，ある σ(x) に対して

$$|\sigma_1| = |\sigma_2| < |\sigma_i|, i \geq 3 \quad (4.11)$$

となる ρ_1, ρ_2 があれば

$$K_0^{(\lambda)}(x) = K^{(\lambda)}(x),$$

$$K_{v+1}^{(\lambda)}(x) = \frac{1}{x}\left[K_v^{(\lambda)}(x) - \frac{K_v^{(\lambda)}(0)}{f(0)} f(x) \right], \quad v = 0,1$$

$$\sigma^{(\lambda)}(x) = \begin{vmatrix} K_0^{(\lambda)}(s_1) & K_0^{(\lambda)}(s_2) & x^2 \\ K_1^{(\lambda)}(s_1) & K_1^{(\lambda)}(s_2) & x \\ K_2^{(\lambda)}(s_1) & K_2^{(\lambda)}(s_2) & 1 \end{vmatrix} \Bigg/ \begin{vmatrix} K_1^{(\lambda)}(s_1) & K_1^{(\lambda)}(s_2) \\ K_2^{(\lambda)}(s_1) & K_2^{(\lambda)}(s_2) \end{vmatrix} \quad (4.12)$$

とするとき,

$$\lim_{\lambda\to\infty} \sigma^{(\lambda)}(x) = (x-\rho_1)(x-\rho_2) \quad (4.13)$$

が成り立つ.

3 段アルゴリズム

次に 3 段アルゴリズムの各段階について説明する．3 段アルゴリズムは三つの段階から成っている．各段階を通じて基本的な役割を果す(n-1)次の多項式の列 $K^{(\lambda)}(x)$は，それぞれの段階で異なった列として生成されるが，第 1 段階を終了したときの多項式 $K^{(M)}(x)$が第 2 段階の出発の多項式として，また，第 2 段階の終了時の多項式 $K^{(L)}(x)$が第 3 段階の出発の多項式として用いられる．

① 第 1 段階（シフトなし過程）
K 多項式 $K^{(0)}(x) = f'(x)$より出発し,(4.2)により M 回繰り返す.
（反復回数 M については，収束判定法参照のこと）
第 1 段階の目的は $K^{(M)}(x)$において，絶対値の小さい p_i を含む項を支配的にすることである.

② 第 2 段階（固定シフト過程）
(4.6) に従い，$\lambda = M, M+1, ..., L-1$ に対して固定シフト多項式の列 $K^{(\lambda)}(x)$を作る.
（L については，収束判定法参照のこと．）
固定シフト多項式を作る目的は，(4.9)が成り立つのか，それとも (4.11) が成り立つのかを監視することにある．しかしσ_k, $k = 1, ..., j$ は計算できない量であるから，固定シフト多項式を作るとともに，$f(x)/\overline{K}^{(\lambda)}(x)$ と$\sigma^{(\lambda)}(x)$の両方を作り，そのいずれも監視することによって 1 次因子へ収束しているのか，2 次因子へ収束しているのかを判断する.
(4.10)と (4.13)のどちらかが成り立つかは，s_1, s_2 の選び方に依存している.そのどちらへも収束しなかったときは s_1, s_2 の決定が不適当であったときで，改めて s_1, s_2 を選び直す（アルゴリズム上の考慮参照）.
さて第 2 段階で $f(x)/\overline{K}^{(\lambda)}(x)$ が収束し始めると，その時の最良の近似値だけシフトすることによって収束を速められ，また $\sigma^{(\lambda)}(x)$が収束し始めると，(4.6)の $\sigma(x)$を $\sigma^{(\lambda)}(x)$で置き換えることによって収束を速められる．これが変動シフト過程への動機となっている.

③ 第 3 段階（変動シフト過程）
この段階は，第 2 段階での 1 次因子か 2 次因子かへの収束に応じて，次の二つの手順に分ける.

a. 1 次因子への反復

$$s^{(L)} = \mathrm{Re}\left(s_1 - f(s_1)/\overline{K}^{(L)}(s_1)\right)$$

$$\boldsymbol{K}^{(\lambda+1)}(x) = \frac{1}{x - s^{(\lambda)}}\left[K^{(\lambda)}(x) - \frac{K^{(\lambda)}(s^{(\lambda)})}{f(s^{(\lambda)})} f(x)\right] \quad (4.14)$$

$$s^{(\lambda+1)} = s^{(\lambda)} - f(s^{(\lambda)})/\overline{K}^{(\lambda)}(s^{(\lambda)}) \quad (4.15)$$
$$\lambda = L, L+1, \cdots\cdots$$

として $s^{(\lambda)}$ を繰り返し求め，多項式 $f(x)$ の一つの根 ρ_1 へ収束させる.

b. 2 次因子への反復
$\lambda = L, L+1, ...$に対して次に示す変動シフト多項式 $K^{(\lambda+1)}(x)$をもとに $\sigma^{(\lambda+1)}(x)$を作る.

$$K^{(\lambda+1)}(x) = \frac{1}{\sigma^{(\lambda)}(x)}$$
$$\left[K^{(\lambda)}(x) + \frac{\begin{vmatrix} f(s_1^{(\lambda)}) & f(s_2^{(\lambda)}) \\ K^{(\lambda)}(s_1^{(\lambda)}) & K^{(\lambda)}(s_2^{(\lambda)}) \end{vmatrix} x + \begin{vmatrix} K^{(\lambda)}(s_1^{(\lambda)}) & K^{(\lambda)}(s_2^{(\lambda)}) \\ s_1^{(\lambda)} f(s_1^{(\lambda)}) & s_2^{(\lambda)} f(s_2^{(\lambda)}) \end{vmatrix}}{\begin{vmatrix} s_1^{(\lambda)} f(s_1^{(\lambda)}) & s_2^{(\lambda)} f(s_2^{(\lambda)}) \\ f(s_1^{(\lambda)}) & f(s_2^{(\lambda)}) \end{vmatrix}} f(x)\right] \quad (4.16)$$

$$K_0^{(\lambda+1)}(x) = K^{(\lambda+1)}(x)$$
$$K_{v+1}^{(\lambda+1)}(x) = \frac{1}{x}\left[K_v^{(\lambda+1)} - \frac{K_v^{(\lambda+1)}(0)}{f(0)} f(x)\right], v = 0,1$$

$$\sigma^{(\lambda+1)}(x) = \frac{\begin{vmatrix} K_0^{(\lambda+1)}(s_1^{(\lambda)}) & K_0^{(\lambda+1)}(s_2^{(\lambda)}) & x^2 \\ K_1^{(\lambda+1)}(s_1^{(\lambda)}) & K_1^{(\lambda+1)}(s_2^{(\lambda)}) & x \\ K_2^{(\lambda+1)}(s_1^{(\lambda)}) & K_2^{(\lambda+1)}(s_2^{(\lambda)}) & 1 \end{vmatrix}}{\begin{vmatrix} K_1^{(\lambda+1)}(s_1^{(\lambda)}) & K_1^{(\lambda+1)}(s_2^{(\lambda)}) \\ K_2^{(\lambda+1)}(s_1^{(\lambda)}) & K_2^{(\lambda+1)}(s_2^{(\lambda)}) \end{vmatrix}} \quad (4.17)$$

ここに $s_1^{(\lambda)}$, $s_2^{(\lambda)}$ は $\sigma^{(\lambda)}(x)$の 2 根である.
$\sigma^{(\lambda)}(x)$ を反復することにより,$f(x)$の 2 次因子である $(x-\rho_1)(x-\rho_2)$ へ収束させる.
（a. 及び b. の反復の終了については，収束判定法参照のこと）

アルゴリズム上の考慮

a. 初期値 s_1, s_2
(4.6) における $\sigma(x)$ の s_1, s_2 は

$$x^n + |a_1|x^{n-1} + \cdots\cdots + |a_{n-1}|x - |a_n| = 0$$

の唯一の正根 β をニュートン法を用いて解き

$$s_1 = \beta(\cos\theta + i\sin\theta), \quad s_2 = \beta(\cos\theta + i\sin\theta) \quad (4.18)$$

とする．ただし，$\theta \neq k\pi$, $k = 0, \pm1, \pm2, \cdots$

b. $K^{(\lambda)}(x)$ の正規化
$K^{(0)}(x) = f'(x)/n$ とし，また(4.6)の右辺を(4.7)の $A^{(\lambda)}$ で割って正規化した多項式を $K^{(\lambda)}(x)$として，演算中のオーバフローを避ける手段としている.

c. $K^{(\lambda)}(x)$, $\delta^{(\lambda)}(x)$ の計算方法
第 2 段階の $K^{(\lambda)}(x)$については,
$f(x) = Q_f(x)\sigma(x) + b(x+u) + a,$
$K^{(\lambda)}(x) = Q_K^{(\lambda)}(x)\sigma(x) + d(x+u) + c.$
とし，この $a, b, c, d, u, v, \boldsymbol{Q}_f(x), \boldsymbol{Q}_k^{(\lambda)}(x)$ を用いて $K^{(\lambda+1)}(x)$を次のように計算する.

$$K^{(\lambda+1)}(x) = \left(\frac{a^2 + uab + vb^2}{bc - ad}\right) Q_K^{(\lambda)}(x) + \left(x - \frac{ac + uad + vbd}{bc - ad}\right) Q_f(x) + b$$

なお，$\sigma^{(\lambda)}(x)$ も同じように a, b, c, d, u, v を用いて計算する．

収束収定法

a. 第 1 段階
この段階の目的は,絶対値の小さな根を強調することであり,反復回数は数値経験的に 5 回 ($M = 5$) としている．

b. 第 2 段階
$t_\lambda = -f(0)/\overline{K}^{(\lambda)}(0)$ とすると

$$\left|t_{\lambda+1} - t_\lambda\right| \le \frac{1}{2}\left|t_\lambda\right| \text{ かつ } \left|t_{\lambda+2} - t_{\lambda+1}\right| \le \frac{1}{2}\left|t_{\lambda+1}\right|$$

を満足したときこの段階の反復をやめ，第 3 段階の 1 次因子への反復に入る．
また，$\sigma^{(\lambda)}(x) = x^2 + u_\lambda x + v_\lambda$ とするとき，v_λについても t_λと 同じ判定を行い，もし満足したなら，第 3 段階の 2 次因子への反復に入る．ただし t_λ, v_λの両方が $20 \times I$ 回の反復を行っても収束しない場合には，(4.18)においてθ を適当に回転することによって s_1, s_2 を作り直す．I は s_1, s_2 を選び直した回数で，その限界を 20 回とし，それを超えたときは，ICON = 10000 として処理を打ち切る．

c. 第 3 段階
(a) 1 次因子への反復
$s = s^{(\lambda)}$とすると

$$\left|\sum_{i=0}^{n} a_i s^{n-i}\right| \le u \cdot \sum_{i=0}^{n}\left|a_i s^{n-i}\right|$$

を満したとき，その s を根として採用する．u は丸め誤差の単位である．

(b) 2 次因子への反復
いま多項式 $f(x)$ を 2 次因子 $x^2 + u_\lambda x + v_\lambda$ で割ったときの商を $\boldsymbol{Q}(x)$,
剰余を $B(x + u_\lambda)$ +A とすると

$f(x)=(x^2 + u_\lambda x + v_\lambda)\, \boldsymbol{Q}(x)+ B(x + u_\lambda) + A$

と書ける．ここで，B, A を組立除法により計算して，それらが共に有効桁を完全に失ったとき，2 次因子が収束したと判定する．すなわち，

$$b_0 = a_0, \qquad c_0 = |a_0|$$
$$b_1 = a_1 - u_\lambda \cdot b_0, \qquad c_1 = \max\{|a_1|, |u_\lambda| \cdot c_0\}$$

として，$k = 2, ..., n$-2 に対して，

$$b_k = a_k - u_\lambda \cdot b_{k-1} - v_\lambda \cdot b_{k-2},$$
$$c_k = \max\{|a_k|, |u_\lambda| \cdot c_{k-1}, |v_\lambda| \cdot c_{k-2}\}$$

を求め，更に

$$B = b_{n-1} = a_{n-1} - u_\lambda \cdot b_{n-2} - v_\lambda \cdot b_{n-2} - v_\lambda \cdot b_{n-3},$$
$$A = b_n = a_n - u_\lambda \cdot b_{n-1} - v_\lambda \cdot b_{n-2}$$

及び

$$D = c_{n-1} = \max\{|a_{n-1}|, |u_\lambda| \cdot c_{n-2}, |v_\lambda| \cdot c_{n-3}\},$$
$$C = c_n = \max\{|a_n|, |u_\lambda| \cdot c_{n-1}, |v_\lambda| \cdot c_{n-2}\}$$

を計算する．
このとき，

$$|B| \le D \cdot u \text{ かつ } |A| \le C \cdot u$$

（u は丸め誤差の単位）

を満たしたならば，$x^2 + u_\lambda x + v_\lambda$ を $f(x)$の 2 次因子として採用する．
反復の回数の限界は，(a), (b)共に 20 回とし，もしこの回数で収束しなかった場合は入ってきた第 2 段階の収束条件を強め，残りの第 2 段階を繰り返す．
なお，詳細については，参考文献 [30], [31] を参照のこと．

H11-20-0111 RKG, DRKG

連立 1 階常微分方程式 （ルンゲ・クッタ・ギル法）
CALL RKG (Y, F, K, N1, H, M, SUB, VW, ICON)

(1) 機能

連立 1 階常微分方程式の初期値問題

$$\left.\begin{aligned} y_1' &= f_1(x, y_1, y_2, \ldots, y_n), & y_{10} &= y_1(x_0) \\ y_2' &= f_2(x, y_1, y_2, \ldots, y_n), & y_{20} &= y_2(x_0) \\ \cdots & & \cdots & \\ y_n' &= f_n(x, y_1, y_2, \ldots, y_n), & y_{n0} &= y_n(x_0) \end{aligned}\right\} \quad (1.1)$$

ただし $n \geq 1$

における関係系 $f_1, \ldots, f_n$ と初期値 $x_0, y_{10}, \ldots, y_{n0}$ が与えられたとき，ルンゲ・クッタ・ギル法により，$x + h, x_0 + 2h, \ldots, x_0 + (m-1)h$ の各点における解 y_i と微係数 y'_i $(i=1, \ldots, n)$ を求める．

(2) パラメタ

Y …… **入力**．初期値 $x_0, y_{10}, y_{20}, \ldots, y_{n0}$ を Y(1,1) = x_0, Y(2,l) = y_{10}, Y(3,1) = y_{20}, ..., Y(N1, 1) = y_{n0} の順に与える．Y(K,M)なる 2 次元配列．
出力．$x_j = x_0 + jh$ $(j=1, \ldots, m-1)$ における $y_1, y_2, \ldots, y_n$ の値．
出力のしかたは次のとおりとする．
J = 1, 2, ..., $m-1$ とするとき，
Y(1, J + 1): x_j
Y(2, J + 1): $y_{1j} = y_1(x_j)$
... ...
Y(N1, J + 1): $y_{nj} = y_{nj}(x_j)$

F …… **出力**．$x_j = x_0 + jh$ $(j = 0, 1, \ldots, m-1)$ における $y'_1, y'_2, \ldots, y'_n$ すなわち $f_1, f_2, \ldots, f_n$ の値．
F(K,M)なる 2 次元配列．
出力のしかたは次のとおりとする．
J = 0, 1, ..., $m-1$ とするとき，
F(1, J + 1) : すべて 1.0
F(2, J + 1) : $y'_{1j} = y'_1(x_j)$
... ...
F(N1, J + 1) : $y'_{nj} = y'_n(x_j)$

K …… **入力**．2 次元配列 Y, F の整合寸法値．

N1 …… **入力**．(連立常微分方程式の元数 n) + 1.

H …… **入力**．キザミ h.

M …… **入力**．解 y_i $(i=1, \ldots, n)$を求めるところの独立変数 x の離散点の個数 m.
出発点 x_0 も個数に含める．$m \geq 2$.

SUB …… **入力**．(1.1)式における $f_i$$(i = 1, 2, \ldots, n)$を計算する副プログラム名．
［副プログラムの用意のし方］
SUBROUTINE SUB (YY,FF)
パラメタ

YY .. **入力**．YY(1) = x, YY(2) = y_1, YY(3)=y_2, ..., YY(n +1) = y_n なる対応を持つ大きさ n + 1 の 1 次元配列．

FF... **出力**．FF(2) = f_1, FF(3) = f_2, FF(4) = f_3, ..., FF(n +l) = f_n なる対応を持つ大きさ n +1 の 1 次元配列．
（使用例参照）．
なお FF(1) には何も代入してはならない．

VW …… **作業領域**．大きさ n+ 1 の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 RKG-1 参照．

表 RKG-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
30000	N1 < 2, K < N1, M <2 H = 0.0 のいずれかであった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... MGSSL

② FORTRAN 基本関数... なし

b. 注意

① パラメタ SUB は本サブルーチンを呼び出す側のプログラムで EXTERNAL 宣言をすること．

② 本ルーチンはキザミを一定に保って，一定間隔の x_i における解を求めるものである．しかし次のような短所を持っている．すなわち，解が順次求まってゆく過程で x_0 から遠ざかるにつれ計算解の，真の解に対する誤差の程度は大きくなっていく.よって h の大きさにもよるが，一般に x_0 からあまり遠くの点における解を求めるには適さない．
しかしながら逆に次のような長所を持っている．すなわち一段法であること，言いかえれば，x_0+jh における解を求めるのに，$x_0+(j-1)h$ における解だけしか必要としない．多段法を適用するときに必要な，ほんのいくつかの出発値を計算するのに適している．また，そのためにだけ適用すべきものである．

c. 使用例
連立 1 解常微分方程式の初期値問題 (3.1) を解く．

$$\left.\begin{aligned} y_1' &= y_2, & y_{10} &= y_1(1) = 5 \\ y_2' &= \frac{4}{x^2} y_1 + \frac{2}{x} y_2, & y_{20} &= y_2(1) = 3 \end{aligned}\right\} \quad (3.1)$$

h=0.1,m=10 の場合．

```
C     **EXAMPLE**
      DIMENSION Y(3,10),F(3,10),VW(3)
      EXTERNAL SUB
      Y(1,1)=1.0
      Y(2,1)=5.0
      Y(3,1)=3.0
      H=0.1
      CALL RKG(Y,F,3,3,H,10,SUB,VW,ICON)
      IF(ICON.NE.0) STOP
      WRITE(6,600)
      WRITE(6,610) ((Y(I,J),I=1,3),
     * (F(I,J),I=2,3),J=1,10)
      STOP
  600 FORMAT('1'/' ',20X,'X',19X,'Y1',18X,
     * 'Y2',18X,'F1',18X,'F2'//)
  610 FORMAT(' ',10X,5E20.8)
      END
      SUBROUTINE SUB(YY,FF)
      DIMENSION YY(3),FF(3)
      FF(2)=YY(3)
      FF(3)=4.0*YY(2)/(YY(1)*YY(1))+2.0*
     *      YY(3)/YY(1)
      RETURN
      END
```

(4)　手法概要

連立 1 階常微分方程式の初期値問題 (1.1) は，独立変数 x そのものを一つの関数,

$$y_0(x)=x,\ y_{00}=y_0(x_0)=x_0 \tag{4.1}$$

と見なすことにより次の (4.2) のように書くことができる．ただし $f_0(y_0, y_1,....., y_n)=1$ とする．

$$\begin{aligned} y_0' &= f_0(y_0,y_1,\ldots,y_n),\ y_{00}=x_0 \\ y_1' &= f_1(y_0,y_1,\ldots,y_n),\ y_{10}=y_1(x_0) \\ y_2' &= f_2(y_0,y_1,\ldots,y_n),\ y_{20}=y_2(x_0) \\ &\ldots \qquad\qquad \ldots \\ y_n' &= f_n(y_0,y_1,\ldots,y_n),\ y_{n0}=y_n(x_0) \end{aligned} \tag{4.2}$$

さらに記号を簡潔にするために，いくつかのベクトルを導入する．

$$\begin{aligned} &\mathbf{y}=(y_0,y_1,y_2\ldots,y_n)^T \\ &\mathbf{y}_0=(y_{00},y_{10},y_{20}\ldots,y_{n0})^T \\ &\mathbf{f}(\mathbf{y})=(f_0(\mathbf{y}),f_1(\mathbf{y}),\ldots,f_n(\mathbf{y}))^T \\ &\quad \text{ただし } f_i(\mathbf{y})=f_i(y_0,y_1\ldots,y_n)^T \\ &\qquad\qquad i=0,1,\ldots,n \end{aligned} \tag{4.3}$$

また，y'_i $(i = 0, 1, ..., n)$ を要素に持つベクトルを $\boldsymbol{y}'$ で表わすことにする．このような記号を用いれば (4.2)は次の(4.4)のように簡潔に書くことができる．

$$\boldsymbol{y}'=\boldsymbol{f}(\boldsymbol{y}),\ \boldsymbol{y}_0=\boldsymbol{y}(x_0) \tag{4.4}$$

この (4.4) に対する，$\boldsymbol{x} = x_0 + h$ における解 $\boldsymbol{y}(x_0 + h)$をルンゲ・クッタ・ギル法により求める手順を(4.5) に示す．式中，$\boldsymbol{q}_i$, $\boldsymbol{k}_i$ は $\boldsymbol{y}_i$, $\boldsymbol{f}_i$ と同じく (n+1) 次元のベクトルである．最初 $\boldsymbol{q}_0$は零ベクトルである．

$$\begin{aligned} \boldsymbol{k}_1 &= h\boldsymbol{f}(\boldsymbol{y}_0) \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0+\frac{1}{2}(\boldsymbol{k}_1-2\boldsymbol{q}_0) \\ \boldsymbol{q}_1 &= \boldsymbol{q}_0+3(\boldsymbol{y}_1-\boldsymbol{y}_0)-\frac{1}{2}\boldsymbol{k}_1 \\ \boldsymbol{k}_2 &= h\boldsymbol{f}(\boldsymbol{y}_1) \\ \boldsymbol{y}_2 &= \boldsymbol{y}_1+\left(1-\sqrt{\frac{1}{2}}\right)(\boldsymbol{k}_2-\boldsymbol{q}_1) \\ \boldsymbol{q}_2 &= \boldsymbol{q}_1+3(\boldsymbol{y}_2-\boldsymbol{y}_1)-\left(1-\sqrt{\frac{1}{2}}\right)\boldsymbol{k}_2 \\ \boldsymbol{k}_3 &= h\boldsymbol{f}(\boldsymbol{y}_2) \\ \boldsymbol{y}_3 &= \boldsymbol{y}_2+\left(1+\sqrt{\frac{1}{2}}\right)(\boldsymbol{k}_3-\boldsymbol{q}_2) \\ \boldsymbol{q}_3 &= \boldsymbol{q}_2+3(\boldsymbol{y}_3-\boldsymbol{y}_2)-\left(1+\sqrt{\frac{1}{2}}\right)\boldsymbol{k}_3 \\ \boldsymbol{k}_4 &= h\boldsymbol{f}(\boldsymbol{y}_3) \\ \boldsymbol{y}_4 &= \boldsymbol{y}_3+\frac{1}{6}(\boldsymbol{k}_4-2\boldsymbol{q}_3) \\ \boldsymbol{q}_4 &= \boldsymbol{q}_3+3(\boldsymbol{y}_4-\boldsymbol{y}_3)-\frac{1}{2}\boldsymbol{k}_4 \end{aligned} \tag{4.5}$$

ここで $\boldsymbol{y}_4$ を $\boldsymbol{y}(x_0 + h)$の近似解とする．更に，$x + 2h$ における解を求めるときは

$$\begin{aligned} \boldsymbol{q}_0 &= \boldsymbol{q}_4 \\ \boldsymbol{y}_0 &= \boldsymbol{y}_4 \end{aligned}$$

とし，(4.5) を繰り返す．以下同様な手順で，$x_0 + 3h$,....., $x_0 + (m\text{-}1)\ h$ における解が順次求まる．

なお，手順については，参考文献 [69] を参照すること．

C21-11-0101 RQDR, DRQDR

実係数 2 次方程式
CALL RQDR (A0, A1, A2, Z, ICON)

(1) 機能

実係数 2 次方程式の根を求める.

$$a_0x^2+a_1x+a_2=0\,(a_0\neq 0)$$

(2) パラメタ

A0, Al, A2 ……**入力**. 2 次方程式の係数.
Z …… **出力**. 2 次方程式の根.
大きさ 2 の複素数型 1 次元配列.
ICON …… **出力**. コンディションコード.
表 RQDR-1 参照.

表 RQDR-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	$a_0=0.0$ であった.	Z (1)の実部には-a_2/a_1 虚部には 0.0 を入れる. Z (2)は保証しない.
30000	$a_0=0.0$ かつ $a_1=0.0$ であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... AMACH, MGSSL
② FORTRAN 基本関数 ... CMPLX, SQRT, ABS

b. 使用例
2 次方程式の係数を入力し，根 Z を求める.

```
C     **EXAMPLE**
      DIMENSION Z(2)
      COMPLEX Z
      READ(5,500) A0,A1,A2
      CALL RQDR(A0,A1,A2,Z,ICON)
      WRITE(6,600) ICON,A0,A1,A2
      IF(ICON.EQ.30000) STOP
      WRITE(6,610) (Z(I),I=1,2)
      STOP
  500 FORMAT(3F10.0)
  600 FORMAT(10X,'ICON=',I5/10X,'A=',
     * 3E15.6)
  610 FORMAT(10X,'Z=',2E15.6)
      END
```

(4) 手法概要

2 次方程式 $(a_0x^2 + a_1x + a_2 = 0)$ の根は $P_1 = a_1/a_0$, $P_2 = a_2/a_0$ とすれば

$$x=\frac{-P\pm\sqrt{P_1^2-4P_2}}{2}$$

で与えられる.

$|P_1^2| >> |4P_2|$ の場合，$-P_1+\sqrt{P_1^2-4P_2}$ ，$-P_1-\sqrt{P_1^2-4P_2}$ のどちらかの計算において大きな精度の損失を招く．この問題を防止するために，根の公式で分子を有理化した式を使い計算している．それを以下に示す.

$D= P_1^2-4P_2$ とすると

$D \le 0$ の場合（共役複素数，重根）

$$x_1=-P_1/2+i\sqrt{-D}/2$$
$$x_2=-P_1/2+i\sqrt{-D}/2 \qquad (4.1)$$

$D > 0$ の場合（二つの実根）

$P_1 > 0$ ならば

$$x_1=-2P_2/\left(-P_1+\sqrt{D}\right)$$
$$x_2=-\left(P_1+\sqrt{D}\right)/2$$

$P_1 \le 0$ ならば

$$x_1=(-P_1+\sqrt{D})/2$$
$$x_2=2P_2/(-P_1+\sqrt{D}) \qquad (4.2)$$

なお，判別式 $D=P_1^2-4P_2$ の計算において $|P_1|$ が大きいとき，P_1^2 がオーバーフローを起こすことがある．この問題を防止するために，$|P_1|>10^{35}$ の判定条件を設け，この条件を満たさないときは上述の計算方法で対処し，この条件を満たすときは $D_0 = 1- 4P_2/P_1/P_1$ で根の判別を行い，$\sqrt{|D|}$ は $|P_1|\sqrt{|D_0|}$ として計算する.

A52-31-0202 SBDL, DSBDL

正値対称バンド行列の LDL^T 分解 （変形コレスキー法）
CALL SBDL (A, N, NH, EPSZ, ICON)

(1) 機能

$n \times n$, 上，下バンド幅 h の正値対称バンド行列 $\boldsymbol{A}$ を変形コレスキー法により LDL^T 分解する.

$$\boldsymbol{A} = \boldsymbol{LDL}^T \qquad (1.1)$$

ただし，$\boldsymbol{L}$ は下バンド幅 h の単位下バンド行列，$\boldsymbol{D}$ は対角行列である.

$n > h \geq 0$ であること.

(2) パラメタ

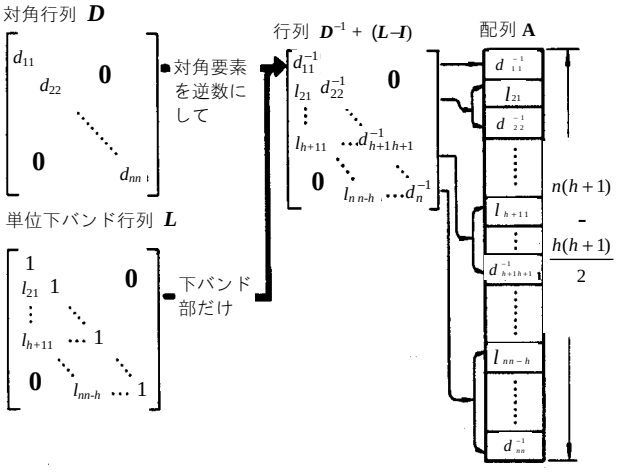
A ····· **入力**. 行列 $\boldsymbol{A}$.
出力. 行列 $\boldsymbol{L}$ と行列 $\boldsymbol{D}^{-1}$.
図 SBDL-1 参照.
対称バンド行列用圧縮モード.
大きさ $n(h+1)- h(h+1)/2$ の 1 次元配列.

N ····· **入力**. 行列 $\boldsymbol{A}$ の次数 n.

NH ····· **入力**. 下バンド幅 h.

EPSZ ····· **入力**. ピボットの相対零判定値 (≥ 0.0).
0.0 のときは標準値が採用される.
（使用上の注意②参照. ）

ICON····· **出力**. コンディションコード.
表 SBDL-1 参照.



［注意］ 演算後，行列 $D^{-1} + (L-I)$ の対角部分及び下バンド部分が，対称バンド行列用圧縮モードで 1 次元配列 A に格納される.

図 SBDL-1 分解要素の格納方法

表 SBDL-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	ピボットが負となった. 行列 $\boldsymbol{A}$ は正値でない.	処理は続行する.
20000	ピボットが相対的に零となった．行列 $\boldsymbol{A}$ は非正則の可能性が強い.	処理を打ち切る.
30000	NH<0, NH ≥ N 又は, EPSZ < 0.0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, MGSSL

② FORTRAN 基本関数... ABS

b. 注意

① 本サブルーチンではバンド部分の外側にある要素にかかわる計算は省略しているので，正値対称行列用のサブルーチン SLDL よりも処理が速い.

② ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，変形コレスキー法による LDL^T 分解過程でピボットの値が 10 進数 s 桁以上の桁落ちを生じた場合にそのピボットを相対的に零と見なし，ICON = 20000 として処理を打ち切る．EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である.
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない.

③ 分解過程でピボットが負となった場合，係数行列は正値ではない．本サブルーチンでは，このとき処理は続行するが ICON = 10000 とする.
ただし，ピボッティングを行っていないため計算誤差が大きい可能性があるので考慮すること.

④ 本サブルーチンでは，LDL^T 分解を行うが，$\boldsymbol{D}$ に対応して $\boldsymbol{D}^{-1}$ を出力するので注意すること.

⑤ 行列 $\boldsymbol{A}$ の行列式は，演算後の配列 A の n 個の対角線上の値を掛け合わせ，その逆数をとることにより得られる．ただし，配列 A は対称バンド行列用圧縮モードであることに注意すること.

c. 使用例

$n \times n$, バンド幅 h の行列を入力し，LDL^T 分解する．$n \leq 100$，$h \leq 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(3825)
   10 READ(5,500) N,NH
      IF(N.EQ.0) STOP
      NH1=NH+1
      NT=N*NH1-NH*NH1/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,630)
      L=0
      LS=1
      DO 20 I=1,N
      L=L+MIN0(I,NH1)
      JS=MAX0(1,I-NH1)
      WRITE(6,600) I,JS,(A(J),J=LS,L)
   20 LS=L+1
      CALL SBDL(A,N,NH,1.0E-6,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,640)
      L=0
      LS=1
      DET=1.0
      DO 30 I=1,N
      L=L+MIN0(1,NH1)
      JS=MAX0(1,I-NH1)
      WRITE(6,600) I,JS,(A(J),J=LS,L)
      DET=DET*A(L)
   30 LS=L+1
      DET=1.0/DET
      WRITE(6,620) DET
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT(' ','(',I3,',',I3,')'/
     * (10X,5E17.8))
  610 FORMAT(/10X,'ICON=',I5)
  620 FORMAT(//10X,
     * 'DETERMINANT OF MATRIX=',E17.8)
  630 FORMAT(/10X,'INPUT MATRIX')
  640 FORMAT(/10X,'DECOMPOSED MATRIX')
      END
```

(4) 手法概要

a. 変形コレスキー法

行列 $\boldsymbol{A}$ が，正値対称な場合には常に

$$\boldsymbol{A} = \tilde{\boldsymbol{L}}\tilde{\boldsymbol{L}}^T \tag{4.1}$$

の形に分解できる．ここで $\tilde{\boldsymbol{L}}$ は下三角行列である．

更に，$\tilde{\boldsymbol{L}} = \boldsymbol{L}\,\mathrm{diag}(\tilde{l}_{ii})$ で $\boldsymbol{L}$ を定めれば，

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T \tag{4.2}$$

の形に分解できる．

ここで，$\boldsymbol{L}$ は単位下三角行列，$\boldsymbol{D}$ は正値な対角行列である．

（4.2）の形に分解する方法は，変形コレスキー法により，次の等式で与えられる．

$$l_{ij} d_j = a_{ij} - \sum_{k=1}^{j-1} l_{ik} d_k l_{jk}, \; j = 1,..., \; i-1 \tag{4.3}$$

$$d_i = a_{ii} - \sum_{k=1}^{i-1} l_{ik} d_k l_{jk} \tag{4.4}$$

ここで，$i = 1, ..., n$.

（詳しくは，サブルーチン SLDL 手法概要参照).

行列 $\boldsymbol{A}$ が，正値対称バンド行列である場合には，(4.2) において，行列 $\boldsymbol{L}$ は同じバンド幅の下バンド行列となる．そこで，本サブルーチンではバンド部分の外側にある要素にかかわる計算を省略し，(4.5)，(4.6) により計算する．

$$l_{ij} d_j = a_{ij} - \sum_{k=\max(1,\,i-h)}^{j-1} l_{ik} d_k l_{jk}, \quad j = i-h,...,n \tag{4.5}$$

$$d_i = a_{ii} - \sum_{k=\max(1,\,i-h)}^{j-1} l_{ik} d_k l_{ik} \tag{4.6}$$

ここで，$i = 1, ..., n$.

なお，詳細については参考文献 [7] を参照すること．

A52-21-0202 SBMDM, DSBMDM

実対称バンド行列の MDM^T 分解 （ブロック対角ピボッティング手法）
CALL SBMDM (A, N, NH, MH, EPSZ, IP, IVW, ICON)

(1) 機能

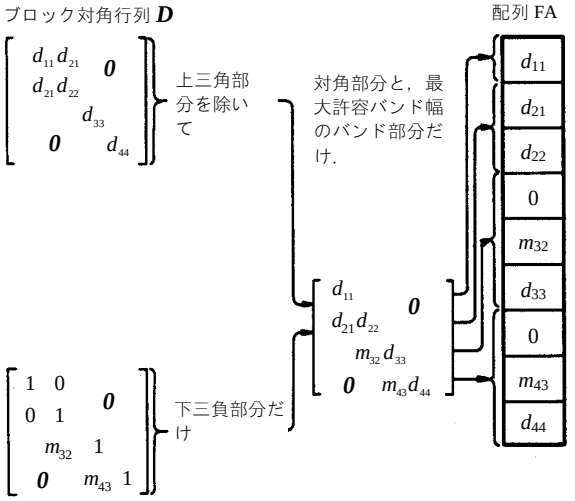
$n \times n$, バンド幅 h の正則な実対称バンド行列 $\boldsymbol{A}$ をブロック対角ピボッティング手法（同名手法にはクラウト法とガウス消去法の関係に似た二つの考え方があり，ここでは後者）により MDM^T 分解する．

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T \qquad (1.1)$$

ただし，$\boldsymbol{P}$ はピボッティングによる行の入換えを行う置換行列，$\boldsymbol{M}$ はバンド幅 $\tilde{h}$ の単位下バンド行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列で $d_{k+1,\,k} \neq 0$ なら $m_{k+1,k}=0$ である．ただし，$\boldsymbol{M} = (m_{ij})$，$\boldsymbol{D} = (d_{ij})$，$n \geq h \geq 0$ であること．

(2) パラメタ

A ····· **入力**．行列 $\boldsymbol{A}$．
$\boldsymbol{A}$ をバンド幅 h_m の行列とみなし対称バンド行列用圧縮モードで与える．
（使用上の注意④参照）
出力．行列 $\boldsymbol{M}$ と行列 $\boldsymbol{D}$．
図 SBMDM-l 参照．



［注意］ $\boldsymbol{D}$ の各ブロック次数は 2．1 及び 1，$\boldsymbol{M}$ のバンド幅は 1，最大許容バンド幅は 2 の場合である．

図 SBMDM-1 分解要素の格納方法

大きさ $n(h_m+1)-h_m(h_m+1)/2$ の 1 次元配列．

N ····· **入力**．行列 $\boldsymbol{A}$ の次数 n．
NH ····· **入力**．行列 $\boldsymbol{A}$ のバンド幅 h．
出力．行列 $\boldsymbol{M}$ のバンド幅 $\tilde{h}$．
（使用上の注意④参照）
MH ····· **入力**．最大許容バンド幅 h_m(N > MH ≥ NH)（使用上の注意④参照）
EPSZ····· **入力**．ピボットの相対零判定値 (≥ 0.0) 0.0 のときは標準値が採用される．
（使用上の注意①参照）
IP ····· **出力**．ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．大きさ n の 1 次元配列．
（使用上の注意②参照）
IVW····· **作業領域**．大きさ n の 1 次元配列．
ICON ····· **出力**．コンディションコード．
表 SBMDM-l 参照．

表 SBMDM-l コンディションコード

コード	意味	処理内容
0	エラーなし．	
20000	ピボットが相対的に零となった．係数行列は非正則の可能性が強い．	処理を打ち切る．
25000	演算中にバンド幅が許容最大値を超えた．	処理を打ち切る．
30000	NH < 0, NH > MH, MH ≥ N 又は EPSZ <0.0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... AMACH, MGSSL
② FORTRAN 基本関数... MAX0, MIN0, ABS, AMAX1

b. 注意
① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，ブロック対角ピボッティング手法による MDM^T 分解の過程でピボットの値（1×1 若しくは，2×2 ピボットが作る小行列式）が，10 進数 s 桁以上の桁落ちを生じた場合に，そのピボットの値を相対的に零と見なし，ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸め誤差の単位を u としたとき $16 \cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ に極小の値を与えればよいが，その結果は，保証されない．
② トランスポジションベクトルとは，ブロック対角ピボッティング手法による MDM^T 分解

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T$$

における置換行列 $\boldsymbol{P}$ に相当するものである．

本サブルーチンでは，ピボッティングに伴い配列 A の内容を実際に交換しており，その履歴を IP に格納している．しかし，1 × 1 ピボットと 2 × 2 ピボットの場合では，IP の格納法が異なるので注意すること．分解の k 段階目における格納方法は次のとおりである．
1 × 1 ピボットのときは，交換は行われず，k を IP(k) に格納する．2×2 ピボットのときは，$-k$ を IP(k) に，第 k + 1 行（及び列）と交換される行（及び列）の番号に負号をつけて IP (k + 1) に格納する．

③ 行列 $\boldsymbol{A}$ の行列式は，演算後の $\boldsymbol{D}$ の行列式に等しい．なお，行列 $\boldsymbol{M}$, $\boldsymbol{D}$ の要素は配列 A 上に図 SBMDM-l に示すとおり格納されている．サブルーチン LSBIX の使用例参照.

④ ピボッティングのための行と列の交換により，行列のバンド幅は一般に増大する．よって，行列 $\boldsymbol{A}$ は適当に零要素を補って，実際のバンド幅 h よりも大きなバンド幅 h_m を持ったバンド行列として入力すること．もしも，分解の途中でバンド幅が h_m を超える場合，ICON = 25000 として処理が打ち切られる．NH の出力の値は正にこの h_m としての必要十分な値に外ならない.

⑤ 本サブルーチンに続けて，サブルーチン BMDMX を呼び出すことにより連立 1 次方程式を解くことができる．しかし，通常はサブルーチン LSBIX を呼び出せば，一度に解が求められる.

⑥ 行列 $\boldsymbol{A}$ の正，負固有値の数を各々求めることができる．使用例参照.

c. 使用例
$n \times n$, バンド幅 h の実対称バンド行列の正と負の固有値の数を，本サブルーチンを用いて求める.
$n \le 100$, $h \le h_m \le 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(3825),IP(100),IVW(100)
      READ(5,500) N,NH,MH
      WRITE(6,600) N,NH,MH
      MHP1=MH+1
      NT=(N+N-MH)*MHP1/2
      READ(5,510) (A(J),J=1,NT)
      EPSZ=0.0
      CALL SBMDM(A,N,NH,MH,EPSZ,IP,IVW,
     *           ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      INEIG=0
      IPEIG=0
      I=1
      J=1
   10 M=IP(J)
      IF(M.EQ.J) GO TO 20
      IPEIG=IPEIG+1
      INEIG=INEIG+1
      I=MIN0(MH,J)+MIN0(MH,I+1)+2+I
      J=J+2
      GO TO 30
   20 IF(A(I).GT.0.0) IPEIG=IPEIG+1
      IF(A(I).LT.0.0) INEIG=INEIG+1
      I=MIN0(MH,J)+1+I
      J=J+1
   30 IF(J.LE.N) GO TO 10
      WRITE(6,620) IPEIG,INEIG
      STOP
  500 FORMAT(3I4)
  510 FORMAT(4E15.7)
  600 FORMAT('1'/10X,'N=',I3,5X,'NH=',I3,
     * 5X,'MH=',I3)
  610 FORMAT(/10X,'ICON=',I5)
  620 FORMAT('0',5X,'POSITIVE ',
     * 'EIGENVALUE=',I4/
     * 6X,'NEGATIVE EIGENVALUE=',I4)
      END
```

(4) 手法概要

a. ブロック対角ピボッティング手法
行列 $\boldsymbol{A}$ が正値対称な場合には，通常 (4.1) なる分解ができる（変形コレスキー法）.

$$\boldsymbol{A} = \boldsymbol{M}_1 \boldsymbol{D}_1 \boldsymbol{M}_1^T \tag{4.1}$$

ここで，$\boldsymbol{M}_1$ は単位下三角行列，$\boldsymbol{D}_1$ は対角行列である.
しかし行列 $\boldsymbol{A}$ が正値でない場合には (4.1) の分解は一般に不可能であり，また可能である場合にも，数値的安定性が保証されない．ところが，(4.1) を (4.2) にすることにより，安定に分解できることが知られている.

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T \tag{4.2}$$

ここで，$\boldsymbol{P}$ はピボッティングに伴う行列の行の入換えを行う置換行列，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列であり，(4.2) の形に分解する方法をブロック対角ピボッティング手法という.

b. 本サブルーチンにおける手順
本サブルーチンでは，ピボッティングのための行（及び列）の入換えによるバンド幅の増大を極力避けるために，算法 $\boldsymbol{D}$（参考文献 [9]，[10] 参照）を使用している．この算法では係数行列の要素は分解の各段階毎に更新される．第 k 段目が終了したときの係数行列を $\boldsymbol{A}^{(k)} = (a_{ij}^{(k)})$ と表わす．ただし，$\boldsymbol{A}^{(0)} = A$ とする．このとき，第 k 段目 (k=l,2,..., n) の処理は次のように表される.

(a) $\rho = \left|a_{mk}^{(k-1)}\right| = \max_{k<i\le n} \left|a_{ik}^{(k-1)}\right|$ (4.3)
を決定する.

(b) $\left|a_{kk}^{(k-1)}\right| \ge \alpha\rho, (\alpha = 0.525)$ (4.4)
が成立すれば (e) へ，さもなければ (c) へ行く.

(c) $\sigma = \max_{k<j\le n} \left| a_{mj}^{(k-1)} \right|$ (4.5)

を決定する.

(d) $\alpha\rho^2 \le |a_{kk}^{(k-1)}| \sigma$ (4.6)

が成立すれば (e) へ，さもなければ (f) へ行く.

(e) $\left| a_{kk}^{(k-1)} \right| < \varepsilon, \left(\varepsilon = \max |a_{ij}| \cdot \mathrm{EPSZ} \right)$ (4.7)

が成立すれば，特異と判断して処理を打ち切る．さもなければ，$a_{kk}^{(k-1)}$ を 1×1 ピボットとして使用し，以下の計算を行う.

$$s = a_{ik}^{(k-1)} / a_{kk}^{(k-1)}, \quad (4.8)$$
$$a_{ji}^{(k)} = a_{ik}^{(k-1)} \cdot s + a_{ji}^{(k)}, \; j = i,\ldots,n$$
$$a_{ij}^{(k)} = -s \quad (4.9)$$
$$, i = k+1,\ldots,n$$

k を 1 だけ増し (a) に戻る.

(f) 第 $k+1$ 行（及び列）と第 m 行（及び列）とを交換する.

$$\left| a_{k+1,k}^{(k-1)} \right| < \varepsilon$$

が成立すれば，特異と判断して処理を打ち切る．さもなければ，$\begin{pmatrix} a_{kk}^{(k-1)} & a_{k,k+1}^{(k-1)} \\ a_{k+1,k}^{(k-1)} & a_{k+1,k+1}^{(k-1)} \end{pmatrix}$ を 2×2 ピボットとして使用し，以下の計算を行う.

$$r = a_{kk}^{(k-1)} / a_{k+1,k}^{(k-1)} \quad (4.12)$$
$$d = r \cdot a_{k+1,k+1}^{(k-1)} / a_{k+1,k}^{(k-1)} - a_{k+1,k}^{(k-1)} \quad (4.13)$$
$$s = \left(a_{i,k+1}^{(k-1)} - a_{ik}^{(k-1)} \cdot a_{k+1,k+1}^{(k-1)} / a_{k+1,k}^{(k-1)} \right) / d \quad (4.14)$$
$$t = \left(a_{ik}^{(k-1)} - r \cdot a_{i,k+1}^{(k-1)} \right) / d \quad (4.15)$$
$$a_{ji}^{(k+1)} = s \cdot a_{jk}^{(k-1)} + t \cdot a_{j,k+1}^{(k-1)} + a_{ji}^{(k-1)} \quad (4.16)$$
$$, j = i,\ldots,n$$
$$a_{ik}^{(k+1)} = -s, \quad (4.17)$$
$$a_{i,k+1}^{(k+1)} = -t \quad (4.18)$$
$$i = k+2,\ldots,n$$

k を 2 だけ増して (a) に戻る.

分解が終了したとき，$\boldsymbol{A}^{(n)}$ の対角ブロック部分が $\boldsymbol{D}$, それ以外の部分が $\boldsymbol{M}$ となっている.

以上の説明は簡単のために係数行列のバンド構造を無視して行ったが，実際の計算はバンド構造を利用して能率的に進められる.

2×2 ピボットが用いられる場合には一般にバンド幅が増大する．本サブルーチンではバンド幅の変化を正確に追跡することにより，不要な計算を省略している.

なお，詳細については参考文献 [9], [10] を参照のこと.

B21-21-0101 SEIG1, DSEIG1

実対称行列の固有値及び固有ベクトル(QL 法)
CALL SEIG1 (A, N, E, EV, K, M, VW, ICON)

(1) 機能

n 次の実対称行列 ***A*** の固有値及び固有ベクトルを，QL 法により求める．固有ベクトルは，$\|\boldsymbol{x}\|_2 = 1$ となるように正規化する．

$n \geq 1$ なること．

(2) パラメタ

A ····· **入力**．実対称行列 ***A***.
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．
演算後，内容は保存されない．

N ····· **入力**．行列の次数 *n*.

E ····· **出力**．固有値．
大きさ *n* の 1 次元配列．

EV ····· **出力**．固有ベクトル．
固有ベクトルは列方向に格納される．
EV(K,N) なる 2 次元配列．

K ····· **入力**．配列 EV の整合寸法 ($\geq n$).

M ····· **出力**．求まった固有値・固有ベクトルの数．

VW ····· **作業領域**.
大きさ $2n$ の 1 次元配列．

ICON ····· **出力**．コンディションコード．
表 SEIG1-1 参照．

表 SEIG1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N = 1であった．	E(1) = A(1), EV (1, 1) = 1.0 とする．
15000	固有値・固有ベクトルのすべては求めつくすことができなかった．	M に求まっている固有値・固有ベクトルの数をセットして処理を打ち切る．
20000	固有値・固有ベクトルは 1 つも求められなかった．	M = 0とする．
30000	N < 1 又は K < N であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, TEIG1, TRBK, TRID1, MGSSL

② FORTRAN 基本関数... SQRT, SIGN, ABS, DSQRT

b. 注意

① 固有値及び固有ベクトルは固有値の求まった順に格納される．

② パラメタ M には，ICON = 0 のとき *n* が，また，ICON = 15000 のとき，求まった固有値及び固有ベクトルの数がセットされている．

③ 本サブルーチンは実対称行列用である．
実対称 3 重対角行列の固有値及び固有ベクトルを求める場合は，サブルーチン TEIG1 を用いること．

④ 実対称行列 ***A*** の固有値だけを求めるときは，サブルーチン TRID1 とサブルーチン TRQL を続けて使用すること．

c. 使用例

n 次の実対称行列 ***A*** の固有値及び固有ベクトルを求める．
$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),E(100),
     *          EV(100,100),VW(200)
   10 CONTINUE
      READ(5,500) N
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1, NN)
      WRITE(6,600) N
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL SEIG1(A,N,E,EV,100,M,VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL SEPRT(E,EV,100,N,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',20X,'ORIGINAL MATRIX',
     * 15X,'ORDER=',I3/'0')
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．以下にその内容を示す．

```
      SUBROUTINE SEPRT(E,EV,K,N,M)
      DIMENSION E(M),EV(K,M)
      WRITE(6,600)
      KAI=(M-1)/5+1
      LST=0
      DO 10 KK=1,KAI
      INT=LST+1
      LST=LST+5
      IF(LST.GT.M) LST=M
      WRITE(6,610) (J,J=INT,LST)
      WRITE(6,620) (E(J),J=INT,LST)
      DO 10 I=1,N
      WRITE(6,630) I,(EV(I,J),J=INT,LST)
   10 CONTINUE
      RETURN
  600 FORMAT('1',20X,
     *       'EIGENVALUE AND EIGENVECTOR')
  610 FORMAT('0',5I20)
  620 FORMAT('0',5X,'ER',3X,5E20.8/)
  630 FORMAT(5X,I3,3X,5E20.8)
      END
```

(4) 手法概要

n 次の実対称行列 $\boldsymbol{A}$ の固有値及び固有ベクトルを求める.

実対称行列 $\boldsymbol{A}$ は，(4.1) の直交相似変換により，対角行列 $\boldsymbol{D}$ に変換できる.

$$\boldsymbol{D} = \boldsymbol{Q}^T \boldsymbol{A} \boldsymbol{Q} \tag{4.1}$$

ここに $\boldsymbol{Q}$ は直交行列である．(4.1) によって求められる対角行列 $\boldsymbol{D}$ の対角要素は，すべて実対称行列 $\boldsymbol{A}$ の固有値であり，$\boldsymbol{Q}$ の第 i 列は，$\boldsymbol{D}$ の i 番目の対角要素に対応する固有ベクトルである.

本サブルーチンでは，次の手順で固有値（すなわち $\boldsymbol{D}$）と固有ベクトル（すなわち $\boldsymbol{Q}$）を求める.

① ハウスホルダー法により実対称行列 $\boldsymbol{A}$ を 3 重対角行列 $\boldsymbol{T}$ に変換する．この変換を (4.2) で表す.

$$\boldsymbol{T} = \boldsymbol{Q}_H^T \boldsymbol{A} \boldsymbol{Q}_H \tag{4.2}$$

ここに $\boldsymbol{Q}_H$ は，ハウスホルダー法の変換行列の積,

$$\boldsymbol{Q}_H = \boldsymbol{P}_1 \boldsymbol{P}_2 \cdots\cdots \boldsymbol{P}_{n-2} \tag{4.3}$$

で与えられる直交行列である.
この処理は，TRID 1 を用いて行う.

② (4.3) の $\boldsymbol{Q}_H$ を計算する.

③ QL 法により 3 重対角行列 $\boldsymbol{T}$ を対角行列 $\boldsymbol{D}$ に変換して固有値を求める．QL 法の詳細については TRQL 参照のこと.
この変換を (4.4) で表す.

$$\boldsymbol{D} = \boldsymbol{Q}_L^T \boldsymbol{T} \boldsymbol{Q}_L \tag{4.4}$$

ここに $\boldsymbol{Q}_L$ は，QL 法の変換行列の積,

$$\boldsymbol{Q}_L = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \cdots\cdots \boldsymbol{Q}_s \tag{4.5}$$

で与えられる直交行列である.
固有ベクトル $\boldsymbol{Q}$ は，(4.1), (4.2) 及び (4.4) より,

$$\boldsymbol{Q} = \boldsymbol{Q}_H \boldsymbol{Q}_L \tag{4.6}$$

と表わせる．したがって，(4.4) の変換と同時に，(4.6) の計算を行うことにより，固有値と固有ベクトル全体が同時に求まる．この処理は，TEIG1 を用いて行う.

なお，固有ベクトルは，$\|\boldsymbol{x}\|_2 = 1$ となっている.

なお，詳細については，参考文献 [12], [13] pp 191-195, [13] pp.212-248, 及び [16] pp.177-206 を参照すること.

B21-21-0201 SEIG2, DSEIG2

実対称行列の固有値及び固有ベクトル (バイセクション法，逆反復法)
CALL SEIG2 (A, N, M, E, EV, K, VW, ICON)

(1) 機能

n 次の実対称行列 ***A*** の固有値を，バイセクション法により大きい方から又は小さい方から *m* 個求め，対応する固有ベクトルを逆反復法により求める．固有ベクトルは，$||\boldsymbol{x}||_2 = 1$ となるように正規化する．

$1 \le m \le n$ なること．

(2) パラメタ

A ····· **入力**．実対称行列 ***A***．
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列．
演算後内容は保存されない．

N ····· **入力**．実対称行列 ***A*** の次数 *n*．

M····· **入力**．求める固有値の個数 *m*．
M = + *m* のときは大きい方から求める．
M = – *m* のときは小さい方から求める．

E····· **出力**．固有値
大きさ *m* の 1 次元配列．

EV ····· **出力**．固有ベクトル．
固有ベクトルは列方向に格納される．
EV(K, *m*) なる 2 次元配列

K ····· **入力**．配列 EV の整合寸法 (≥ *n*)

VW····· **作業領域**．大きさ 7*n* の 1 次元配列．

ICON····· **出力**．コンディションコード．
表 SEIG2-1 参照．

表 SEIG2-l コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N = 1であった．	E(1) = A(1), EV(1, 1) = 1.0 とする．
15000	*m* 個の固有値を求めた後，固有ベクトルを求めるとき，求まらないものがあった．	その固有ベクトルを 0 ベクトルとする．
20000	固有ベクトルは求められなかった．	固有ベクトルはすべて 0 ベクトルとする．
30000	M = 0, N < |M| 又は K < N であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, TEIG2, TRBK, TRID1, MGSSL, UTEG2

② FORTRAN 基本関数 ... IABS, SQRT, SIGN, ABS, AMAX1, DSQRT

b. 注意

① 本サブルーチンは，実対称行列用である．
実対称 3 重対角行列の *m* 個の固有値及び固有ベクトルを求める場合は，サブルーチン TEIG2 を用いること．

② 実対称行列 ***A*** の *m* 個の固有値だけを求める場合は，サブルーチン TRID1 とサブルーチン BSCT1 を続けて用いること．

c. 使用例

n 次の実対称行列 ***A*** の大きい方から又は小さい方から *m* 個の固有値及び対応する固有ベクトルを求める．

$n \le 100$, $m \le 10$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),E(10),
     *          EV(100,10),VW(700)
   10 CONTINUE
      READ(5,500) N,M
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N,M
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
      WRITE(6,610) I,(A(J),J=NI,NE)
   20 CONTINUE
      CALL SEIG2(A,N,M,E,EV,100,VW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      MM=IABS(M)
      CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',20X,'ORIGINAL MATRIX',5X,
     *       'N=',I3,5X,'M=',I3/'0')
  610 FORMAT('0',7X,I3,5E20.7/
     *       (11X,5E20.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン SEIG1 の使用例を参照のこと．

(4) 手法概要

n 次の実対称行列 $\boldsymbol{A}$ の固有値を，バイセクション法により大きい方から又は小さい方から m 個求め，対応する固有ベクトルを逆反復法により求める．

まず実対称行列 $\boldsymbol{A}$ をハウスホルダー法により 3 重対角行列 $\boldsymbol{T}$ に変換する．この操作を(4.1) で示す．

$$\boldsymbol{T} = \boldsymbol{Q}_H^T \boldsymbol{A} \boldsymbol{Q}_H \tag{4.1}$$

ここに，$\boldsymbol{Q}_H$ は直交行列である．
この操作は，サブルーチン TRID1 を用いて行う．

次にバイセクション法により m 個の固有値を求め，逆反復法により，対応する $\boldsymbol{T}$ の固有ベクトルを求める．逆反復法は，

$$\left(\boldsymbol{T} - \mu \boldsymbol{I}\right)\boldsymbol{x}_r = \boldsymbol{x}_{r-1}, r = 1,2, \tag{4.2}$$

を反復的に解いて固有ベクトルを求める方法である．ただし，(4.2) で，μ はバイセクション法により求めた固有値，$\boldsymbol{x}_0$ は適当な初期ベクトルである．

この操作は，サブルーチン TEIG2 を用いて行う．

T の固有ベクトルを $\boldsymbol{y}$ とすれば，$\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ は，(4.1) の $\boldsymbol{Q}_H$ を用いて，

$$\boldsymbol{x} = \boldsymbol{Q}_H \boldsymbol{y} \tag{4.3}$$

により得られる．(4.3) はハウスホルダー法の逆変換である．この操作はサブルーチン TRBK を用いて行う．

なお固有ベクトルは，$\|\boldsymbol{x}\|_2 = 1$ となっている．

なお，詳細については，参考文献 [12] 及び [13] pp. 418-439 を参照すること．

I11-51-0101 SFRI, DSFRI

正弦フレネル積分 $S(x)$
CALL SFRI (X, SF, ICON)

(1) 機能

正弦フレネル積分 $S(x)$ の値

$$S(x)=\frac{1}{\sqrt{2\pi}}\int_0^x \frac{\sin(t)}{\sqrt{t}}dt=\int_0^{\sqrt{\frac{2}{\pi}x}}\sin\left(\frac{\pi}{2}t^2\right)dt$$

を多項式及び漸近形式の近似式を用いて計算する.

ただし, $x\geq 0$ であること.

(2) パラメタ

X …… **入力**. 独立変数 x.
SF …… **出力**. 関数値 $S(x)$.
ICON …… **出力**. コンディションコード
表 SFRI-1 参照

表 SFR1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	$X\geq t_{max}$であった.	SF = 0.5とする.
30000	x < 0であった.	SF = 0.0とする.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... MGSSL, UTLIM
② FORTRAN 基本関数 ... SIN, COS, SQRT

b. 注意
① 引数 X の範囲は
$0\leq X < t_{max}$
であること.
X が大きくなると sin(x), cos(x)の値が正確に計算できなくなることに対する処理である.
（手法概要 (4.4) 参照）.

c. 使用例
$S(x)$ の値を x が 0 から 100 まで，1 おきに計算し数表を作る.

```
C     **EXAMPLE**
      WRITE(6,600)
      DO 10 K=1,101
      X=K-1
      CALL SFRI(X,SF,ICON)
      IF(ICON.EQ.0) WRITE(6,610) X,SF
      IF(ICON.NE.0) WRITE(6,620) X,SF,ICON
   10 CONTINUE
      STOP
  600 FORMAT('1','EXAMPLE OF FRESNEL ',
     * 'INTEGRAL',///6X,'X',9X,'S(X)'/)
  610 FORMAT(' ',F8.2,E17.7)
  620 FORMAT(' ','** ERROR **',5X,'X=',
     * E17.7,5X,'S=',E17.7,5X,
     * 'CONDITION=',I10)
      END
```

(4) 手法概要

正弦フレネル積分 $S(x)$ の計算は，x = 4 を境にして，近似式の形が異なる.

a. $0\leq x < 4$ の場合
$S(x)$ のべき級数展開

$$S(x)=\sqrt{\frac{2}{\pi}}x\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n+1)!(4n+3)}x^{2n+1} \qquad (4.1)$$

を次の近似式を用いて計算する.

単精度： $S(x)=\frac{x\sqrt{x}}{2}\sum_{k=0}^{6}a_k z^{2k}, z=x/4$ (4.2)

倍精度： $S(x)=\sqrt{x}\sum_{k=0}^{11}a_k x^{2k+1}$ (4.3)

b. $x\geq 4$ の場合
$S(x)$ の漸近展開

$$S(x)=\frac{1}{2}+\sin(x)P(x)+\cos(x)Q(x) \qquad (4.4)$$

により計算する. ただし $P(x)$, $Q(x)$ は次の近似式を用いる.

単精度： $P(x)=\frac{2}{\sqrt{x}}\sum_{k=0}^{10}a_k z^{k+1}, z=4/x$ (4.5)

$$Q(x)=-\frac{2}{\sqrt{x}}\sum_{k=0}^{11}a_k z^{k}, z=4/x \qquad (4.6)$$

倍精度： $P(x)=\frac{1}{\sqrt{x}}\sum_{k=0}^{10}a_k z^{k+1}\Big/\sum_{k=0}^{11}b_k z^{k}, z=4/x$ (4.7)

$$Q(x)=\frac{1}{\sqrt{x}}\sum_{k=0}^{10}c_k z^{k}\Big/\sum_{k=0}^{10}d_k z^{k}, z=4/x \qquad (4.8)$$

A21-11-0201 SGGM, DSGGM

行列の差（実行列）
CALL SGGM (A, KA, B, KB, C, KC, M, N, ICON)

(1) 機能

$m \times n$ の実行列 ***A*** と実行列 ***B*** の差 ***C*** を求める.

$$\boldsymbol{C} = \boldsymbol{A} - \boldsymbol{B}$$

ここで ***C*** は，$m \times n$ の実行列である.

$m, n \geq 1$ であること.

(2) パラメタ

A …… **入力**. 行列 ***A***.
A(KA, N)なる 2 次元配列.

KA …… **入力**. 配列 A の整合寸法 (≥ M).

B…… **入力**. 行列 ***B***.
B(KB, N)なる 2 次元配列.

KB…… **入力**. 配列 B の整合寸法 (≥ M).

C…… **出力**. 行列 ***C***.
C(KC, N)なる 2 次元配列.
（使用上の注意①参照）

KC…… **入力**. 配列 C の整合寸法 (≥ M).

M…… **入力**. 行列 ***A***, ***B***, ***C*** の行数 *m*.

N …… **入力**. 行列 ***A***, ***B***, ***C*** の行数 *n*.

ICON…… **出力**. コンディションコード.
表 SGGM-1 参照.

表 SGGM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	M<1, N<1, KA < M, KB < M 又は KC < M であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... MGSSL

② FORTRAN 基本関数 ...なし.

b. 注意

① 領域の節約について

配列 A あるいは，B 上の内容を保存する必要のない場合は，次のように呼び出すことにより領域が節約できる.

- 配列 A の内容が不要の場合.
  CALL SGGM (A, KA, B, KB, A, KA, M, N, ICON)
- 配列 B の内容が不要の場合.
  CALL SGGM (A, KA, B, KB, B, KB, M, N, ICON)

この場合，行列 ***C*** は配列 A あるいは，B 上に得られる.

c. 使用例

実行列 ***A***, ***B*** の差を求める． $m, n \leq 50$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(50,50),B(60,60),
     *          C(100,100)
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
      DATA KA/50/,KB/60/,KC/100/
   10 READ(5,100) M,N
      IF(M.EQ.0) STOP
      WRITE(6,150)
      READ(5,200) ((A(I,J),I=1,M),J=1,N)
      READ(5,200) ((B(I,J),I=1,M),J=1,N)
      CALL SGGM(A,KA,B,KB,C,KC,M,N,ICON)
      IF(ICON.NE.0) GOTO 10
      CALL PGM(IA,1,A,KA,M,N)
      CALL PGM(IB,1,B,KB,M,N)
      CALL PGM(IC,1,C,KC,M,N)
      GOTO 10
  100 FORMAT(2I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** MATRIX ADDITION **')
      END
```

本使用例中のサブルーチン PGM は，実行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている.

G21-11-0101 SIMP1, DSIMP1

1 次元有限区間積分 （等間隔離散点入力，シンプソン則）
CALL SIMP1 (Y, N, H, S, ICON)

(1) 機能

等間隔離散点 $x_i = x_1 + (i\text{-}1)\ h,\ i=1,...,\ n$ における関数値 $y_i = f(x_i)$ が与えられたときシンプソン則により

$$S = \int_{x_1}^{x_n} f(x)dx \qquad n \geq 3,\ h > 0.0$$

を求める．ここで h はキザミ値である．

(2) パラメタ

Y ····· **入力**．関数値 y_i .
　　大きさ n の 1 次元配列.
N ····· **入力**．離散点の個数 n.
H ····· **入力**．キザミ値 h.
S ····· **出力**．積分値 S.
ICON····· **出力**．コンディションコード.
　　表 SIMP1-1 参照.

表 SIMPI-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	$n = 2$であった.	台形則により計算する.
30000	$n < 2$ または $h \leq 0.0$であった.	$S = 0.0$として処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II… MGSSL
② FORTRAN 基本関数... なし.

b. 使用例
関数値 y_i とキザミ値 h を入力し，積分値 S を求める.

```
C     **EXAMPLE**
      DIMENSION Y(100)
      READ(5,500) N,H
      READ(5,510) (Y(I),I=1,N)
      CALL SIMP1(Y,N,H,S,ICON)
      WRITE(6,600) ICON,S
      STOP
  500 FORMAT(I3,F10.0)
  510 FORMAT(6F10.0)
  600 FORMAT(10X,'ILL CONDITION =',I5
     *      /10X,'INTEGRAL VALUE =',E15.7)
      END
```

(4) 手法概要

区間 $[x_1, x_n]$ について，等間隔に取られた離散点に対する関数値 y_i を用い，シンプソン則により積分する.

まず，最初の 3 分点について，2 次補間式で近似し，区間 $[x_1, x_3]$ にわたり積分をする.

$$\int_{x_1}^{x_3} f(x)dx \approx \frac{h}{3}(y_1 + 4y_2 + 2y_3) \tag{4.1}$$

次に続く 3 分点についても同様に計算は進められる.

$$\int_{x_1}^{x_n} f(x)dx \approx \frac{h}{3}(y_1 + 4y_2 + 2y_3 + ... \\ ... + 4y_{n-1} + y_n) \tag{4.2}$$

離散点の個数が奇数の場合は，上述のように，3 分点によって最後まで計算するが離散点の個数が偶数の場合は，区間 $[x_1, x_{n\text{-}3}]$ までは上述の方法で計算し，最後の区間 $[x_{n\text{-}3}, x_n]$ にわたっては，ニュートン・コーツ $\frac{3}{8}$ 則を適用して計算する.

$$\int_{x_{n-3}}^{x_n} f(x)dx \approx \frac{3}{8}h(y_1 + 3y_2 + 3y_3 + y_4) \tag{4.3}$$

ただし $n = 2$ の場合は，シンプソン則は適用できないので特に台形則を適用する.

$$\int_{x_1}^{x_2} f(x)dx \approx \frac{h}{2}(y_1 + y_2) \tag{4.4}$$

なお，詳細については，参考文献 [46] pp.114-121 を参照すること.

G23-11-0101 SIMP2, DSIMP2

1 次元有限区間積分 （関数入力，適応型シンプソン則）
CALL SIMP2 (A, B, FUN, EPS, S, ICON)

(1) 機能

関数 $f(x)$, a, b, ε, が与えられたとき,

$$\left| S - \int_a^b f(x)dx \right| \le \varepsilon \qquad (1.1)$$

を満たす積分の近似値 S をシンプソン則に基く適応型手法により求める．ここで $f(x)$は積分区間において有限な値を持つこと．

(2) パラメタ

A …… **入力**．積分区間の下限 a.

B…… **入力**．積分区間の上限 b.

FUN …… **入力**．被積分関数 $f(x)$ を計算する関数副プログラム名（使用例参照）．

EPS …… **入力**．求めようとする積分の近似値に対する絶対誤差の上限 ε (≥ 0.0).
EPS = 0.0 と与えれば，本サブルーチンが達成できる最大の精度の近似値を計算する．
出力．実際に求まった積分の近似値 S が持つ絶対誤差の上限．（使用上の注意 b ③参照）．

S …… **出力**．積分の近似値.

ICON …… **出力**．コンディションコード.
表 SIMP2-1 参照.

表 SLMP2-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	$\varepsilon > 0.0$ のとき $\left\|s-\int_a^b f(x)dx\right\| \le \varepsilon$ を満たす S を求めることができなかった.	実際に求まった近似値と，その絶対誤差の上限をパラメタ S , EPS に出力する.
30000	$\varepsilon < 0.0$であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, MGSSL

② FORTRAN 基本関数... ABS

b. 注意

① パラメタ FUN は本サブルーチンを呼び出す側のプログラムで EXTERNAL 宣言をすること.

② 本サブルーチンは，被積分関数 $f(x)$ が次にしめす(a), (b) を共に満たすとき有効に働く.

(a) $f(x)$ とその 5 階までの導関数が積分区間において連続である.

(b) $f(x)$ が積分区間内で激しく振動しない.

また $f(x)$ が $g(x)|x-x_0|^{\alpha}$ なる形をしている場合でも，多くの場合有効に働く．ただし，$g(x)$ は上記 (a), (b) の性質を共に満たし，α は非負であり，x_0 は a, b, $(a+b)/2$ のいずれかであるものとする.

利用者はこのことに注意し，もし $f(x)$ あるいはその 5 階までの導関数のどれかが a, b, $(a+b)/2$ の 3 点以外の点で不連続になることが分かっていれば，その不連続点を境にして積分区間を分割すべきである．そして結果としてできる部分区間の各々に対して本ルーチンを適用すればよい.

③ 出力された S の精度

本サブルーチンは，与えられた ε に対して (1.1) を満たす S を求めるのであるが，被積分関数 $f(x)$ の形あるいは ε の大きさによってはそのような S を計算できないことがある.

このような場合には，本サブルーチンが達成できるかぎりの精度を持つ近似値と，それがもつ絶対誤差の上限を計算しそれぞれパラメタ S, EPS に出力する．このとき，ICON = 10000 となる.

また，パラメタ EPS の入力時の値を

EPS = 0.0

とすることもでき，これは上に述べた場合にあたるので，パラメタ S, EPS の出力時の値は上の意味をもつ(このときは ICON = 0 となる).

c. 使用例

積分

$$\int_0^1 \frac{1}{x^2+10^{-6}}dx$$

を求める．EPS=0.0 の場合.

```
C     **EXAMPLE**
      EXTERNAL FUN
      A=0.0
      B=1.0
      EPS=0.0
      WRITE(6,600) A,B,EPS
      CALL SIMP2(A,B,FUN,EPS,S,ICON)
      WRITE(6,610) ICON,S,EPS
      STOP
  600 FORMAT('1'/' ',30X,'A=',E16.8,5X,
     * 'B=',E16.8,5X,'INPUT EPS=',E16.8//)
  610 FORMAT(' ',30X,'***RESULT***'/
     * ' ',30X,'ICON=',I5,5X,'S=',E16.8,
     * 5X,'EPS=',E16.8)
      END
      FUNCTION FUN(X)
      FUN=1.0/(X*X+1.0E-6)
      RETURN
      END
```

(4) 手法概要

本サブルーチンは，シンプソン則に基く適応型手法を適用している．適応型手法とは，被積分関数 $f(x)$ のふるまいが激しいところでは $f(x)$ を計算する点が密にとられ，反対に穏やかなところでは粗にとられる手法のことをいう．説明の便宜上 (1.1) において $a < b$ と仮定しておく．

与えられた積分区間 $[a, b]$ を以下の b で述べる方法により分割していき，分割された各々の部分区間に対してシンプソン則を適用する．その部分区間のうち，あるものはさらに分割されていく．最終的に，部分区間ごとに求めた積分値の総和を $[a, b]$ に渡る積分値とする．このとき絶対誤差が ε を上まわらないように配慮される．

a. シンプソン則と誤差評価

積分区間 $[a, b]$ 内のある部分区間を $[\alpha,\beta]$ で，その長さを $h=\beta-\alpha$ で表すとき，積分，

$$\int_{\alpha}^{\beta} f(x)dx \tag{4.1}$$

を近似する公式とその誤差評価について述べる．

$R[\alpha, \beta]f(x)$ を (4.2) で定義する．

$$R[\alpha,\beta]f(x) \equiv \frac{h}{6}\left\{ f(\alpha)+4f\left(\frac{\alpha+\beta}{2}\right)+f(\beta)\right\} \tag{4.2}$$

これは 3 点に基くシンプソン則である．

さらに $R^{(2)}[\alpha, \beta]f(x)$ を (4.3) で定義する．

$$\begin{aligned} R^{(2)}[\alpha,\beta]f(x) &\equiv R\left[\alpha,(\alpha+\beta)/2\right]f(x) \\ &\quad + R\left[(\alpha+\beta)/2,\beta\right]f(x) \end{aligned} \tag{4.3}$$

これは区間 $[\alpha, \beta]$ を 2 等分割し各々の区間に，3 点に基くシンプソン則を適用したものである．

本サブルーチンにおいては，(4.1) に対する近似として (4.3) を適用している．

$R^{(2)}[\alpha, \beta]f(x)$ の誤差評価は，オイラー・マクローリンの展開公式より導くことができる．すなわち $f(x)$ とその 5 階までの導関数が $[\alpha, \beta]$ で連続であるとすると，

$$\begin{aligned} &R^{(2)}[\alpha,\beta]f(x)-\int_{\alpha}^{\beta} f(x)dx \\ &=\frac{C_4h^4}{2^4}\int_{\alpha}^{\beta} f^{(4)}(x)dx+O(h^6) \end{aligned} \tag{4.4}$$

が成り立つ．ここで C_4 は $f(x)$ に依存しない定数である．一方，同様に

$$\begin{aligned} &R[\alpha,\beta]f(x)-\int_{\alpha}^{\beta} f(x)dx \\ &=C_4h^4\int_{\alpha}^{\beta} f^{(4)}(x)dx+O(h^6) \end{aligned} \tag{4.5}$$

が成り立つので，(4.4), (4.5) から，$O(h^6)$ が無視できると仮定すれば，

$$\begin{aligned} &R^{(2)}[\alpha,\beta]f(x)-\int_{\alpha}^{\beta} f(x)dx \\ &\approx\frac{1}{15}\left\{R[\alpha,\beta]f(x)-R^{(2)}[\alpha,\beta]f(x)\right\} \end{aligned} \tag{4.6}$$

となる．(4.6) の右辺は計算可能な量なので $R^{(2)}[\alpha,\beta]\ f(x)$ の誤差評価として利用できる．ここで，(4.6) は丸め誤差の大きさが小さいことを仮定している．

部分区間 $[\alpha,\beta]$ に対して，$\varepsilon(\beta-\alpha)/(b-a)$ なる量を割当て，この量を $R^{(2)}[\alpha,\beta]\ f(x)$ の誤差 (4.6) が上まわることを許されない限度とする．すなわち，

$$\frac{1}{15}\left|R[\alpha,\beta]f(x)-R^{(2)}[\alpha,\beta]f(x)\right| \le \frac{\beta-\alpha}{b-a}\varepsilon \tag{4.7}$$

ならば $R^{(2)}[\alpha,\beta]\ f(x)$ を (4.1) の近似値として採用する．逆に (4.7) が成り立たないときは，$[\alpha,\beta]$ は更に 2 等分割される．実際の計算では，(4.7) の式で判定せず，それと等価な式 (4.8) より判定する．

$$\left|D[\alpha,\beta]f(x)\right| \le \mathrm{E} \tag{4.8}$$

ただし

$$\begin{aligned} D[\alpha,\beta]f(x) &= \frac{12}{\beta-\alpha}\left\{R[\alpha,\beta]f(x)\right. \\ &\quad \left.-R^{(2)}[\alpha,\beta]f(x)\right\} \end{aligned} \tag{4.9}$$

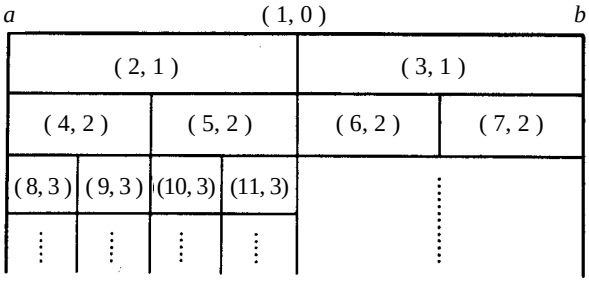
$$E=\frac{180\varepsilon}{b-a} \tag{4.10}$$

b. 積分区間 $[a, b]$ の分割方法

与えらた積分区間 $[a, b]$ の分割のしかたと積分公式 (4.3) を適用する部分区間の選択順序について述べる．説明の都合上，各部分区間に対して“番号”と“レベル”を次のように割当てる．

積分区間 $[a, b]$ を番号 1 でレベル 0 の区間と定義する．区間 $[a, (a + b) / 2]$ を番号 2 でレベル 1 の区間，区間 $[(a + b) / 2, b]$ を番号 3 でレベル 1 の区間と定義する．一般に区間 $[\alpha, \beta]$ を番号 N でレベル L の区間とすれば区間 $[\alpha,(\alpha + \beta)/2]$ は番号 $2N$ でレベル $(L + 1)$ の区間，区間

[(α + β)/2, β] は番号 (2*N*+l) でレベル(*L* + 1) の区間であると定義する（図 SIMP2-1 参照）.



［注意］ 括弧内の左の数字が番号を，右の数字がレベルを示す.

図 SIMP2-1 番号とレベル

分割は次のように進められる．まず番号 2 でレベル 1 の区間に対して (4.3) を適用する．そして (4.8) の判定に合格すれば $R^{(2)}[a,(a+b)/2]\,f(x)$ を $[a,(a+b)/2]$ にわたる積分値として採用する．逆に合格しなければ，番号 4 でレベル 2 の区間を考え (4.3) を適用する.

一般に計算のある段階で番号 *N* でレベル *L* の区間に対して (4.3) を適用し，(4.8) の判定を行う．もし合格しなければ，次には番号 2*N* でレベル (*L* + 1) の区間に対して同じことをする．反対に合格すればその時の $R^{(2)}[\alpha,\beta]f(x)$ を区間 $[\alpha,\beta]$ に渡る積分値として採用しこれを，すでに部分区間ごとに求めた積分値の累計に加えておき，次には番号が *M*(*K*)+ 1, レベルが *L* – *K* である区間に移る．ここで *M*(*K*) とは，数列,

$$M(0)=N,\ M(1)=\frac{M(0)-1}{2},\ldots\ldots,$$

$$M(J+1)=\frac{M(J)-1}{2},\ldots\ldots$$

において最初に出会う偶数である．そして *L* – *K* = 0 となったとき，積分区間 [*a*, *b*] にわたる積分が終了したことになり計算は終了する.

このようにして，[*a*, *b*] にわたる積分の近似値として

$$\sum_{i=1}^{n} R^{(2)}[a_{i-1},a_i]f(x) \qquad (a_0=a,\ a_n=b) \tag{4.11}$$

を出力する．ここで各 $R^{(2)}[a_{i-1}, a_i]f(x)$ は (4.8) , したがって (4.7) を満たすものである．(4.6), (4.7) より,

$$\left|\sum_{i=1}^{n} R^{(2)}[a_{i-1},a_i]f(x)-\int_a^b f(x)dx\right|\le\varepsilon \tag{4.12}$$

が成り立つ.

c. 丸め誤差に対する配慮

ある部分区間 $[\alpha,\beta]$ における $R^{(2)}[\alpha,\beta]\,f(x)$ を採用するか否かは (4.8) の判定の合否によることを述べた．もし (4.8) が満たされない場合は $[\alpha,\beta]$ は 2 等分割され，$[\alpha,(\alpha+\beta)/2]$ を考えることになる訳であるが，このように区間が小さくされてゆく過程のある時点からは，(4.8) の $D[\alpha,\beta]f(x)$ なる量が，誤差ばかりになる現象が起りかねない．それは $f(x)$ の計算値が持つ有効桁が $D\,[\alpha,\beta]\,f(x)$ の計算過程で失われるためである．ある部分区間でこの状態に落ち入ったなら，それ以上 2 等分割の作業を進めるのは好ましくない．この問題に対して以下のような配慮をしている.

理論的には，$[\alpha,\beta]$ において $f(x)$ と 4 階までの導関数が連続でかつ $f^{(4)}(x)$ が定符号ならば (4.13) が成り立つ.

$$\left|D[\alpha,(\alpha+\beta)/2]f(x)\right|\le\left|D[\alpha,\beta]f(x)\right| \tag{4.13}$$

したがって計算中に (4.13) が満たされなかったなら，その原因は，$[\alpha,\beta]$ 内に $f^{(4)}(x)$ の零点があったか，あるいは丸め誤差が，$D[\alpha,\beta]\,f(x)$ もしくは $D\,[\alpha,(\alpha+\beta)/2]\,f(x)$ の計算を完全に乱したか，のいずれかであると判断できる（しかし，いずれであるかを正しく判断するのは難しい）．多くの場合，小さな部分区間である限り，$f^{(4)}(x)$ の零点による影響が原因でなく，丸め誤差による乱調が原因である．このような状態に落ち入ったなら，2 等分割の作業をやめ (4.7) の ε を別な値 ε'に変える，すなわち (4.8) の E を別な値 E'に変えるのが得策と考える.

本サブルーチンはこの E'のコントロールを次の要領で行う.

(a) レベルが 5 以上のある部分区間 $[\alpha,\beta]$ において,

$$\left|D[\alpha,\beta]f(x)\right|>E', \tag{4.14}$$

$$\left|D[\alpha,(\alpha+\beta)/2]f(x)\right|>E' \tag{4.15}$$

かつ

$$\left|D[\alpha,(\alpha+\beta)/2]f(x)\right|\ge\left|D[\alpha,\beta]f(x)\right| \tag{4.16}$$

となったとき，E'を

$$E'=\left|D[\alpha,(\alpha+\beta)/2]f(x)\right| \tag{4.17}$$

と変える．そして $R^{(2)}[\alpha,(\alpha,\beta)/2]\,f(x)$ を $[\alpha,(\alpha+\beta)/2]$ にわたる積分値として採用する.

(b) ある部分区間 $[\alpha,\beta]$ に対して，

$$\left|D[\alpha,\beta]f(x)\right| \le E' \qquad (4.18)$$

となれば $R^{(2)}[\alpha,\beta]f(x)$ を $[\alpha,\beta]$ に渡る積分値として採用する．そして $|D[\alpha,\beta]f(x)| \neq 0$ ならば E' を

$$E' = \max\left(E, \left|D[\alpha,\beta]f(x)\right|\right) \qquad (4.19)$$

と変える．

以上のように E' をコントロールしても，なお問題が残る．それは (a) の操作において $[\alpha,\beta]$ 内に $f^{(4)}(x)$ の零点がある場合でも丸め誤差の乱調とみなしてしまうこと，(b) においては (4.18) における E' が大きすぎはしなかったかということである．本サブルーチンはこれらについてもある程度の工夫を施している（詳細は省く）．またレベルが 30 の区間においてはそのときの $R^{(2)}[\alpha,\beta]f(x)$ を無条件に採用し E' は変えない．

以上のように E' がコントロールされることになると，最終的な積分値 (4.11) はもはや (4.12) を満たさなくなり ε の代りとして

$$\varepsilon_{\text{eff}} = \frac{1}{b-a}\sum_{i=1}^{n}\varepsilon'\left(a_i - a_{i-1}\right) \qquad (4.20)$$

となる．ここで ε' は E' と

$$E' = \frac{180\varepsilon'}{b-a} \qquad (4.21)$$

で対応づけられる量である．本サブルーチンは(4.20) の ε_{eff} をパラメタ EPS に出力する．

なお，詳細については，参考文献 [61] を参照すること．

I11-41-0101 SINI DSINI

正弦積分 $S_i(x)$
CALL SINI (X, SI, ICON)

(1) 機能

正弦積分 $S_i(x)$の値

$$S_i(x)=\int_0^x \frac{\sin(t)}{t}dt$$

を多項式及び漸近形式の近似式を用いて計算する.

(2) パラメタ

X ····· **入力**. 独立変数 x
SI····· **出力**. 関数値 $S_i(x)$
ICON····· **出力**. コンディションコード.
表 SINI-1 参照.

表 SINI-1 コンディションコード

コード	意味	処理内容
0	エラーなし	
20000	$\|X\| \geq t_{max}$ であった.	SI = sign (X)·π / 2

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... MGSSL, UTLIM
② FORTRAN 基本関数 ... ABS, SIN, COS

b. 注意
① 引数 X の範囲は
$|X| < t_{max}$
であること.
|X| が大きくなると sin (*x*), cos (*x*) の値が正確に計算できなくなることに対する処理である.
（手法概要（4.4）参照）

c. 使用例
$S_i(x)$ の値を x が 0.0 から 10.0 まで，0.1 おきに計算し数表を作る.

```
C      **EXAMPLE**
       WRITE (6,600)
       DO 10 K=1,101
       A=K-1
       X=A/10.0
       CALL SINI(X,SI,ICON)
       IF(ICON.EQ.0) WRITE(6,610) X,SI
       IF(ICON.NE.0) WRITE(6,620) X,SI,ICON
    10 CONTINUE
       STOP
   600 FORMAT('1','EXAMPLE OF SINE ',
      * 'INTEGRAL FUNCTION'///6X,'X',9X,
      * 'SI(X)'/)
   610 FORMAT(' ',F8.2,E17.7)
   620 FORMAT(' ','** ERROR **',5X,'X=',
      * E17.7,5X,'SI=',E17.7,5X,
      * 'CONDITION=',I10)
       END
```

(4) 手法概要

正弦積分 $S_i(x)$ の計算は，$|x| = 4$ を境にして，近似式の形が異なる.

a. $0 \leq |x| < 4$ の場合
$S_i(x)$ のべき級数展開

$$S_i(x)=\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n+1}}{(2n+1)!(2n+1)} \tag{4.1}$$

をそれぞれ (4.2), (4.3) の近似式を用いて計算する.

単精度： $S_i(x)=\sum_{k=0}^{6} a_k z^{2k+1},\ z=x/4$ (4.2)

倍精度： $S_i(x)=\sum_{k=0}^{11} a_k x^{2k+1}$ (4.3)

b. $|x| \geq 4$ の場合
$S_i(x)$ の漸近展開

$$S_i(x)=\mathrm{sign}(x)\cdot\left[\pi/2+\left\{P(x)\cos(x)-Q(x)\sin(x)\right\}/x\right] \tag{4.4}$$

より計算する．ただし $P(x)$, $Q(x)$ は次の近似式を用いる.

単精度： $P(x)=\sum_{k=0}^{11} a_k z^k,\ z=4/x$ (4.5)

$$Q(x)=\sum_{k=0}^{11} b_k z^k,\ z=4/x \tag{4.6}$$

倍精度： $P(x)=\sum_{k=0}^{11} a_k z^k \Big/ \sum_{k=0}^{11} b_k z^k,\ z=4/x$ (4.7)

$$Q(x)=-\sum_{k=0}^{10} c_k z^k \Big/ \sum_{k=0}^{11} d_k z^k,\ z=4/x \tag{4.8}$$

A22-51-0202 SLDL, DSLDL

正値対称行列の LDL^T 分解 （変形コレスキー法）
CALL SLDL (A, N, EPSZ, ICON)

(1) 機能

$n \times n$ の正値対称行列 $\boldsymbol{A}$ を変形コレスキー法により LDL^T 分解する.

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T \qquad (1.1)$$

ただし，$\boldsymbol{L}$ は単位下三角行列，$\boldsymbol{D}$ は対角行列である.

$n \geq 1$ であること.

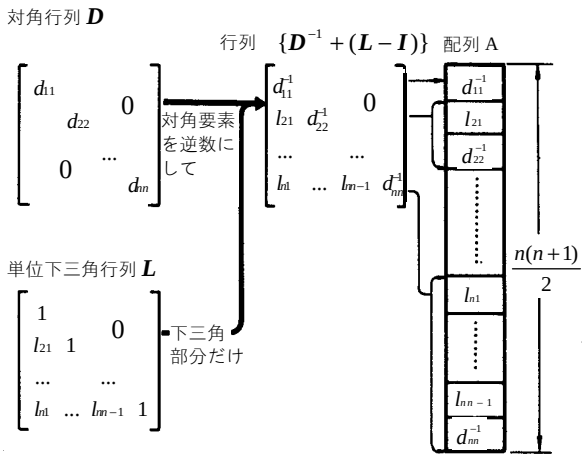
(2) パラメタ

A ····· **入力**. 行列 $\boldsymbol{A}$.
出力. 行列 $\boldsymbol{L}$ と $\boldsymbol{D}^{-1}$.
図 SLDL-1 参照.
対称行列用圧縮モード.
大きさ $n(n+1)/2$ の 1 次元配列.

N ····· **入力**. 行列 $\boldsymbol{A}$ の次数 n.

EPSZ ····· **入力**. ピボットの相対零判定値 (≥ 0.0).
0.0 のときは標準値が採用される.
（使用上の注意①参照）

ICON ····· **出力**. コンディションコード.
表 SLDL-1 参照.



［注意］ 演算後，行列 $\boldsymbol{D}^{-1}+(\boldsymbol{L}-\boldsymbol{I})$ の対角部分及び下三角部分が，対称行列用圧縮モードで 1 次元配列 A に格納される.

図 SLDL-1 分解要素の格納方法

表 SLDL-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	ピボットが負となった. 行列 $\boldsymbol{A}$ は正値でない.	処理は続行する.
20000	ピボットが相対的に零となった．行列 $\boldsymbol{A}$ は非正則の可能性が強い.	処理を打ち切る.
30000	N <1 又は，EPSZ < 0.0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, MGSSL

② FORTRAN 基本関数... ABS

b. 注意

① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，変形コレスキー法による LDL^T 分解の過程でピボットの値が 10 進 s 桁以上の桁落ちを生じた場合にそのピボットを相対的に零と見なし，ICON = 20000 として処理を打ち切る．EPSZ の標準値は丸め誤差の単位を u としたとき，EPSZ = $16 \cdot u$ である.
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない.

② 分解過程でピボットが負となった場合，係数行列は正値ではない．このとき本サブルーチンでは，ICON = 10000 として処理は続行する．ただしピボッティングを行っていないため計算誤差は大きい可能性があるので考慮すること.

③ 本ルーチンでは，LDL^T 分解を行うが，$\boldsymbol{D}$ に対応して $\boldsymbol{D}^{-1}$ を出力するので注意すること.

④ 行列 $\boldsymbol{A}$ の行列式は，演算後の配列 A の n 個の対角線上の値（$\boldsymbol{D}^{-1}$ の対角要素）を掛け合わせ，その逆数をとることにより得られる．ただし，配列 A は対称行列用圧縮モードであることに注意すること.

⑤ 正値対称バンド行列の場合，バンド部分の外側にある要素にかかわる計算を省略するサブルーチン SBDL の方が速く処理できる.

c. 使用例

$n \times n$ の行列を入力し，LDL^T 分解する.
$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,630)
      L=0
      LS=1
      DO 20 I=1,N
      L=L+I
      WRITE(6,600) I,(A(J),J=LS,L)
   20 LS=L+1
      CALL SLDL(A,N,1.0E-6,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) GOTO 10
      WRITE(6,640)
      L=0
      LS=1
      DET=1.0
      DO 30 I=1,N
      L=L+I
      WRITE(6,600) I,(A(J),J=LS,L)
      DET=DET*A(L)
   30 LS=L+1
      DET=1.0/DET
      WRITE(6,620) DET
      GOTO 10
  500 FORMAT(I5)
  510 FORMAT(5E10.2)
  600 FORMAT(' ',I5/(10X,5E16.8))
  610 FORMAT(/10X,'ICON=',I5)
  620 FORMAT(//10X,
     *'DETERMINANT OF MATRIX=',E16.8)
  630 FORMAT(/10X,'INPUT MATRIX')
  640 FORMAT(/10X,'DECOMPOSED MATRIX')
      END
```

(4) 手法概要

a. 変形コレスキー法

行列 $\boldsymbol{A}$ が正値対称な場合には常に

$$\boldsymbol{A}=\tilde{\boldsymbol{L}}\tilde{\boldsymbol{L}}^{T} \tag{4.1}$$

の形に分解できる．ここで，$\tilde{\boldsymbol{L}}$ は下三角行列である．しかもこの分解は $\tilde{\boldsymbol{L}}$ の対角行列要素を正になるように定めれば，一意的に定まる．このとき，分解手順は次の等式で与えられる（コルスキー法）．

$$\tilde{l}_{ij}=\left(a_{ij}-\sum_{k=1}^{j-1}\tilde{l}_{ik}\tilde{l}_{jk}\right)\Big/\,\tilde{l}_{jj},\ j=1,\ldots,i-1 \tag{4.2}$$

$$\tilde{l}_{ii}=\left(a_{ii}-\sum_{k=1}^{i-1}\tilde{l}_{ik}^{2}\right)^{1/2} \tag{4.3}$$

ただし，$\boldsymbol{A}=\left(a_{ij}\right)$，$\tilde{\boldsymbol{L}}=\left(\tilde{l}_{ij}\right)$.

ここで，$\tilde{\boldsymbol{L}}=\boldsymbol{L}\mathrm{diag}\left(\tilde{l}_{ii}\right)$ で $\boldsymbol{L}$ を定めれば，

$$\begin{aligned}\boldsymbol{A}=\tilde{\boldsymbol{L}}\tilde{\boldsymbol{L}}^{T}&=\boldsymbol{L}\mathrm{diag}(\tilde{l}_{ii})\mathrm{diag}(\tilde{l}_{ii})\boldsymbol{L}^{T}\\&=\boldsymbol{L}\mathrm{diag}(\tilde{l}_{ii}^{2})\boldsymbol{L}^{T}=\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{T}\end{aligned} \tag{4.4}$$

となる．ここで，$\boldsymbol{L}$ は単位下三角行列，$\boldsymbol{D}$ は正値な対角行列である．(4.4) の形に分解する方法は，変形コレスキー法により，次の等式で与えられる．

$$l_{ij}d_{j}=a_{ij}-\sum_{k=1}^{j-1}l_{ik}d_{k}l_{jk},\ j=1,\ldots,i-1 \tag{4.5}$$

$$d_{i}=a_{ii}-\sum_{k=1}^{i-1}l_{ik}d_{k}l_{ik} \tag{4.6}$$

ここで，$i=1,\ldots,n$

コレスキー法では，(4.3) の計算で平方根を求めなければならないが，変形コレスキー法ではその必要がないのが特徴である．

なお，詳細については，参考文献 [2] を参照すること．

A22-21-0202 SMDM, DSMDM

実対称行列の MDM^T 分解 （ブロック対角ピボッティング手法）
CALL SMDM (A, N, EPSZ, IP, VW, IVW, ICON)

(1) 機能

$n \times n$ の正則な実対称行列 $\boldsymbol{A}$ をブロック対角ピボッティング手法（同名手法にはクラウト法とガウス消去法の関係に似た二つの考え方があり，ここでは前者）により MDM^T 分解する．

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T \qquad (1.1)$$

ただし，$\boldsymbol{P}$ はピボッティングによる行の入換えを行う置換行列，$\boldsymbol{M}$ は単位下三角行列，$\boldsymbol{D}$ は高々次数 2 の対称なブロックから成る対称ブロック対角行列で $d_{k+1,k} \neq 0$ なら $m_{k+1,k} = 0$ である．ただし，$\boldsymbol{M}= (m_{ij})$, $\boldsymbol{D} = (d_{ij})$, $n \geq 1$ であること．

(2) パラメタ

A ····· **入力**．行列 $\boldsymbol{A}$.
対称行列用圧縮モード．
出力．行列 $\boldsymbol{M}$ と行列 $\boldsymbol{D}$.
図 SMDM-1 参照
大きさ $n(n+1)/2$ の 1 次元配列.

N ····· **入力**．行列 $\boldsymbol{A}$ の次数 n.

EPSZ ····· **入力**．ピボットの相対零判定値 (≥ 0.0).
0.0 のときは標準値が採用される．
（使用上の注意①参照）．

IP····· **出力**．ピボッティングによる行の入換えの履歴を示すトランスポジションベクトル．
大きさ n の 1 次元配列.
（使用上の注意②参照）．

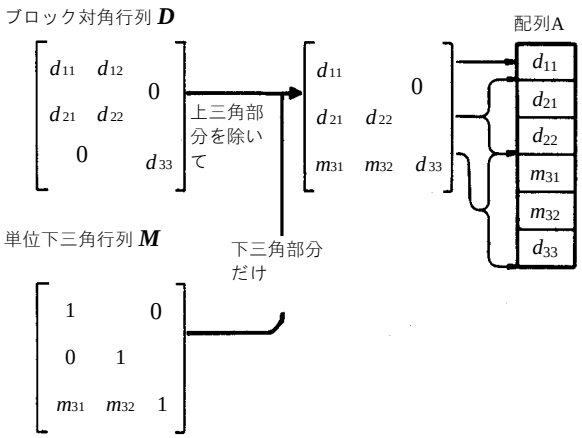
VW····· **作業領域**．大きさ $2n$ の 1 次元配列.

IVW ····· **作業領域**．大きさ n の 1 次元配列.

ICON····· **出力**．コンディションコード.
表 SMDM-1 参照.

表 SMDM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
20000	ピボットが相対的に零となった．行列は非正則の可能性が強い.	処理を打ち切る.
30000	N < 1 又は EPSZ < 0.0 であった.	処理を打ち切る.



［注意］ 演算後，行列 $\boldsymbol{D}+(\boldsymbol{M}-\boldsymbol{I})$ の対角部分及び下三角部分が，対称行列用圧縮モードで 1 次元配列 A に格納される．ここで，$\boldsymbol{D}$ は次数 2 と 1 のブロックからなる場合である．

図 SMDM-l 分解要素の格納方法

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... AMACH, MGSSL, USCHA

② FORTRAN 基本関数... ABS, SQRT, IABS, ISIGN

b. 注意

① ピボットの相対零判定値 EPSZ に 10^{-s} を設定したとすると，この値は次の意味を持っている．すなわち，ブロック対角ピボッティング手法による MDM^T 分解の過程でピボットの値（1×1 若しくは，2×2 ピボットが成す小行列の行列式）が，10 進 s 桁以上の桁落ちを生じた場合に，そのピボットの値を相対的に零と見なし，ICON = 20000 として処理を打ち切る．
EPSZ の標準値は丸め誤差の単位を u としたとき $16 \cdot u$ である．
なお，ピボットが小さくなっても計算を続行させる場合には，EPSZ へ極小の値を与えればよいが，その結果は保証されない．

② トランスポジションベクトルとは，ブロック対角ピボッティング手法による MDM^T 分解

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T$$

における置換行列 $\boldsymbol{P}$ に相当する．
本サブルーチンでは，ピボッティングに伴い配列 A の内容を実際に交換しており，その履歴を IP に格納している．しかし，1 × 1 ピボットと 2 × 2 ピボットの場合では，IP の格納法が多少異なるので注意すること．分解の k 段階目における格納方法は次のとおりである．
1 × 1 ピボットのときは，第 k 行（及び列）と交換する行（及び列）の番号 $r(r \geq k)$ を IP(k) に格納する．2 × 2 ピボットのときは，更に第 k+l 行（及び列）と交換する行（及び列）の番

号 $s(s \geq k+1)$ の負値 $(-s)$ を IP $(k+1)$ に格納する.

③ 行列 ***A*** の行列式は，演算後の行列 ***D*** の行列式に等しい．なお，行列 ***D*** の要素は配列 A 上に図 SMDM-1 に示すとおり格納されている．サブルーチン LSIX の使用例参照.

④ 本サブルーチンでは，行列の対称性を生かし分解中もそれを保持するので領域が節約できる.

⑤ 本サブルーチンに続けて，サブルーチン MDMX を呼び出すことにより連立 1 次方程式を解くことができる．しかし，通常はサブルーチン LSIX を呼び出せば，一度に解が求められる.

⑥ 行列 ***A*** の正，負固有値の数を各々求めることができる．使用例参照.

c. 使用例

$n \times n$ の実対称行列の性質が調べられる正と負の固有値の数を，本サブルーチンを用いて求める.

$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),VW(200),
     *          IP(100),IVW(100)
      CHARACTER*4 IA
      DATA IA/'A   '/
      READ(5,500) N
      NT=(N*(N+1))/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,600) N
      CALL PSM(IA,1,A,N)
      EPSZ=0.0
      CALL SMDM(A,N,EPSZ,IP,VW,IVW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      INEIG=0
      IPEIG=0
      I=1
      J=1
   10 IF(IP(J+1).GT.0) GO TO 20
      IPEIG=IPEIG+1
      INEIG=INEIG+1
      J=J+2
      I=I+J-1+J
      GO TO 30
   20 IF(A(I).GT.0.0) IPEIG=IPEIG+1
      IF(A(I).LT.0.0) INEIG=INEIG+1
      J=J+1
      I=I+J
   30 IF(J.LT.N) GO TO 10
      IF(J.NE.N) GO TO 40
      IF(A(I).GT.0.0) IPEIG=IPEIG+1
      IF(A(I).LT.0.0) INEIG=INEIG+1
   40 WRITE(6,620) IPEIG,INEIG
      STOP
  500 FORMAT(I3)
  510 FORMAT(4E15.7)
  600 FORMAT('1'
     * /6X,'CLASSIFICATION OF EIGENVALUE'
     * /6X,'ORDER OF MATRIX=',I4)
  610 FORMAT(' ',5X,'ICON OF SMDM=',I6)
  620 FORMAT(' ',5X,'POSITIVE ',
     * 'EIGENVALUE=',
     * I4/6X,'NEGATIVE EIGENVALUE=',I4)
      END
```

本使用例中のサブルーチン PSM は，実対称行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている.

(4) 手法概要

a. ブロック対角ピボッティング手法

行列 ***A*** が正値対称な場合には，通常 (4.1) なる分解ができる（変形コレスキー法）.

$$\boldsymbol{A} = \boldsymbol{M}_1 \boldsymbol{D}_1 \boldsymbol{M}_1^T \tag{4.1}$$

ここでは，$\boldsymbol{M}_1$ は単位下三角行列，$\boldsymbol{D}_1$ は対角行列である.

しかし行列 ***A*** が正値でない場合には，(4.1) の分解過程で要素が増長されることがあり不安定である．そこで，(4.1) を (4.2) とすれば，安定性は十分となる.

$$\boldsymbol{PAP}^T = \boldsymbol{MDM}^T \tag{4.2}$$

ここで，***P*** はピボッティングに伴う行列の行の入換えを行う置換行列，***M*** は単位下三角行列，***D*** は高々次数 2 の対称なブロックから成る対称ブロック対角行列であり，(4.2) の形に分解する方法をブロック対角ピボッティング手法という.

本サブルーチンは，ブロック対角ピボッティング手法を用いて，実対称行列 ***A*** が与えられたとき，(4.3) 及び (4.4) を満足する行列 ***P***, ***D*** 及び ***M*** を求める.

$$\boldsymbol{PAP}^T = \boldsymbol{MS} \tag{4.3}$$

$$\boldsymbol{S} = \boldsymbol{DM}^T \tag{4.4}$$

b. 本サブルーチンにおける手順

本サブルーチンにおける分解過程の k 段階目 (k = 1, ..., n) は次の計算手順により行列 ***M***, ***S***, ***D*** の第 k 列を求める（なお，特に値を定義しない要素は零である）．ただし，2×2 ピボットが選択されたときは第 $k+1$ 列も求める.

ここで，以降の説明の都合上 n 次元ベクトル ***Q***, ***R*** を導入する．各要素は $\boldsymbol{A} = (a_{ij})$, $\boldsymbol{M} = (m_{ij})$, $\boldsymbol{D} = (d_{ij})$, $\boldsymbol{S} = (s_{ij})$, $\boldsymbol{Q} = (q_i)$, $\boldsymbol{R} = (r_i)$ で表す.

(a)

$$s_{ik} = d_{i,i-1} m_{k,i-1} + d_{ii} m_{ki} + d_{i,i+1} m_{k,i+1}$$

$$i = 1,\ldots,k-1$$

$$q_i = a_{ik} - \sum_{l=1}^{k-1} m_{il} s_{lk}, i = k,\ldots,n$$

(b) $\lambda = |q_j| = \max_{k<i\le n} |q_i|$

$|q_k| \ge \alpha\lambda$ なら q_k を 1×1 ピボットする．

$d_{kk} = q_k$

$m_{ik} = q_i / q_k, i = k+1,..., n$

(g) へ移る．ただし$\alpha = (1+\sqrt{17})/8$（参考文献参照）．

(c) 行列 $\boldsymbol{M}, \boldsymbol{A}$ の第 $k+1$ 行（及び列）と第 j 行（及び列）を，q_{k+1} と q_j を各々交換する．

$s_{i,k+1} = d_{i,i-1}m_{k+1,i-1} + d_{ii}m_{k+1,i} + d_{i,i+1}m_{k+1,i+1}$

$i = 1,..., k-1$

$r_i = a_{i,k+1} - \sum_{l=1}^{k-1} m_{il}s_{l,k+1}, i = k+1,..., n$

(d) σ $|q_k| \ge \alpha\lambda^2$ なら q_k を 1×1 ピボットする．

$d_{kk} = q_k$

$m_{ik} = q_i / q_k, i = k+1,..., n$

(g) へ移る．

(e) $\sigma = \max(\lambda, \max_{k+1<i\le n} |r_i|)$

$|r_{k+1}| > \alpha\sigma$ なら r_{k+1} を 1×1 ピボットする．行列 $\boldsymbol{M}, \boldsymbol{A}$ の第 k 行（及び列）と第 $k+1$ 行（及び列）を交換する．

$d_{kk} = r_{k+1}$

$m_{k+1,k} = q_{k+1} / r_{k+1}$

$m_{ik} = r_i / r_{k+1}, i = k+2,..., n$

(g) へ移る．

(f) $\det \begin{pmatrix} q_k & q_{k+1} \\ q_{k+1} & r_{k+1} \end{pmatrix}$ $(= q_k r_{k+1} - q_{k+1}q_{k+1})$ を 2×2 ピボットする．

$d_{kk} = q_k$

$d_{k,k+1} = d_{k+1,k} = q_{k+1}$

$d_{k+1,k+1} = r_{k+1}$

$$\begin{cases} m_{ik} = (q_i A_{k+1} - r_i q_{k+1}) / (q_k r_{k+1} - q_{k+1}q_{k+1}) \\ m_{i,k+1} = (r_i q_k - q_i q_{k+1}) / (q_k r_{k+1} - q_{k+1}q_{k+1}) \end{cases}$$

$i = k+2,..., n$

(g) 次の計算段階を，k 段階目のピボットが 1×1 なら $k+1$ 段階目，2×2 なら $k+2$ 段階目とする．

ところで，本アルゴリズムでは，s_{ik} や $s_{i,k+1}$ を計算するとき，行列 $\boldsymbol{D}$ の要素が明らかに零であるときは，それを考慮している．また，k 段階目の実行が (d) 又は (e) で終了したなら，$k+1$ 段階目における $s_{i,k+1}$, $(i = 1, ..., k)$ や q_i $(i = k+1, ..., n)$ の値は 1 回の乗算と 1 回の加算を残して既に計算されており，この事実を (a) で反映させている．

本サブルーチンにおける積和計算部分は精度を上げて行うことにより，丸め誤差の影響を少なくしている．

なお，参考文献として [9] 及び [10] を参照すること．

E31-11-0101 SMLE1, DSMLE1

最小二乗近似多項式による平滑化 （等間隔離散点）
CALL SMLE1 (Y, N, M, L, F, ICON)

(1) 機能

独立変数の等間隔な点における観測値 y_i, i = 1,...,n が与えられたとき，局部的な最小二乗近似多項式を用いてこれらの観測値を平滑化する．

すなわち，ある一つの観測値を平滑化するとき，全体の観測値を用いた最小二乗近似多項式を適用する代わりに，その隣接する指定された個数 l（ただし，$l \leq n$）の観測値に関する指定された次数 m の局部的最小二乗近似多項式を適用して平滑化する．この手続きを全ての観測値について行うものである．

ただし，本サブルーチンでは，m と l の組合せを，表 SMLE1-1 のように制限する．

表 SMLE1-1 m と l の制限

次数 (m)	観測値の個数 (l)
1	3
	5
3	5
	7

(2) パラメタ

Y …… **入力**．観測値 y_i．
大きさ n の 1 次元配列．

N …… **入力**．観測値の個数 n.

M …… **入力**．局部的最小二乗近似多項式の次数 m.

L …… **入力**．局部的最小二乗近似多項式に用いる観測値の個数 l.

F …… **出力**．平滑値．
大きさ n の 1 次元配列．

ICON …… **出力**．コンディションコード．
表 SMLE1-2 参照．

表 SMLE 1-2 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	次のいずれかであった． 1) $m \neq 1$ かつ，$m \neq 3$ 2) $m = 1$ のとき， $l \neq 3$ かつ，$l \neq 5$ $m = 3$ のとき， $l \neq 5$ かつ，$l \neq 7$ 3) $n < l$	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... MGSSL

② FORTRAN 基本関数..... なし

b. 注意

① 本サブルーチンは，真の関数が，全体にわたり唯一の多項式で近似できないが局部的にはある次数の多項式で近似できることを前提としている．

② m と l の選択にあたっては，観測値に対する物理的情報と利用者の経験に基づいてなされるべきである．しかし,平滑化の程度は l の値に伴い増加し，m の値に伴い減少することを考慮する必要がある．

③ 本サブルーチンを繰り返し呼ぶこと，つまり，l 個に関する m 次の最小二乗近似多項式を，求まった平滑値に繰り返し適用することも可能であるが，むやみに反復させると，全体の観測値に関する m 次の最小二乗近似多項式を適用した結果に近づく傾向がある．もし反復する場合，いつ反復を止めるべきかは利用者の判断による．

④ 本サブルーチンで制限された以外の m, l を使って平滑化を行ないたい場合は，サブルーチン SMLE2 を利用されたい．

c. 使用例

独立変数の等間隔な点における観測値の個数 n, 観測値 y_i, 局部的最小二乗近似多項式の次数 m, それに用いる観測値の個数 l を入力し，各々観測値を平滑化する．
($n \leq 20$) の場合．

```
C       **EXAMPLE**
        DIMENSION Y(20),F(20)
        READ(5,500) N,M,L
        READ(5,510) (Y(I),I=1,N)
        CALL SMLE1(Y,N,M,L,F,ICON)
        WRITE(6,600)
        WRITE(6,610) ICON
        IF(ICON.NE.0) STOP
        WRITE(6,620) M,L
        WRITE(6,630) (I,Y(I),F(I),I=1,N)
        STOP
  500 FORMAT(3I5)
  510 FORMAT(5F10.0)
  600 FORMAT('1'////26X,
     *'**** SMOOTHING BY SMLE1 ****')
  610 FORMAT('0',37X,'ICON=',I5)
  620 FORMAT('0',20X,'DEGREE OF ',
     *'POLYNOMIAL =',I2/
     *' ',23X,'POINT OF SUBRANGE =',I2/
     *'0',17X,'NO.',10X,'OBSERVED VALUES',
     *10X,'SMOOTHED VALUES')
  630 FORMAT(' ',16X,I4,10X,F15.7,10X,
     *F15.7)
        END
```

(4) 手法概要

本サブルーチンは，与えられた n 個の観測値 y_i を，特定の次数の唯一の最小二乗近似多項式を用いて平滑化する代わりに，局部的な l 個の観測値に関して，m 次の各々異った最小二乗近似多項式を適用して平滑化するものである．

すなわち，ある一つの観測値 y_k は，隣接する l (= 2r +1) 個の観測値 $y_{k-r}, ..., y_{k-1}, y_k, y_{k+1}, ..., y_{k+r}$ に関して m 次の最小二乗近似多項式をあてはめ，その k における値を算出することにより平滑化する．

いま，

$$\eta_s = y_i,\ s = i-k,\ k-r \le i \le k+r$$

とする（図 SMLE1-1 参照）．

$$\begin{array}{ccccccccc} y_{k-r}, & \ldots, & y_{k-1}, & y_k, & y_{k+1}, & \cdots, & y_{k+r} \\ \downarrow & & \downarrow & \downarrow & \downarrow & & \downarrow \\ \eta_{-r}, & \ldots, & \eta_{-1}, & \eta_0, & \eta_1, & \ldots\ldots, & \eta_r \end{array}$$

図 SMLEI-1 y_k と η_s の対応

これらの η_s に関する m 次の最小二乗近似多項式は (4.1) で表わされる．

$$\bar{y}_m(s) = \sum_{t=-r}^{r}\sum_{j=0}^{m}\left\{\frac{(2j+1)[(2r)!]^2}{(2r+j+1)!(2r-j)!}\cdot P_j(t,2r)P_j(s,2r)\right\}\eta_t \quad (4.1)$$

ここで

$$P_j(\xi,2r) = \sum_{k=0}^{j}(-1)^{j+k}\frac{(j+k)^{(2k)}}{(k!)^2}\frac{(r+\xi)^{(k)}}{(2r)^{(k)}} \qquad (4.2)$$

$$(\xi)^{(k)} = \xi(\xi-1)(\xi-2).....(\xi-k+1)$$

である．(4.1) は，l 点に関する m 次の平滑化公式と呼ばれ，この公式は，サブルーチン LESQ1 の手法概要で述べた最小二乗近似多項式を $w(x_i)$ = 1 として，等間隔な x_i についてあてはめることにより得られる．

そこで，y_k は (4.3) で平滑化する．

$$\bar{y}_m(0) = \sum_{t=-r}^{r}\sum_{j=0}^{m}\left\{\frac{(2j+1)[(2r)!]^2}{(2r+j+1)!(2r-j)!}\cdot P_j(t,2r)P_j(0,2r)\right\}\eta_t \qquad (4.3)$$

ここで，y_k が $y_1, ..., y_r$ と $y_{n-r+1}, ..., y_n$ のときは，両側に等しく r 個の観測値を持たないので，$\bar{y}_m(0)$ で平滑化できない．そこで，$y_1, ..., y_r$ に対しては $\bar{y}_m(-r), ..., \bar{y}_m(-1)$ で，$y_{n-r+1}, ..., y_n$ に対しては $\bar{y}_m(1), ..., \bar{y}_m(r)$ で平滑化する．ちなみに，(4.1) において，$\bar{y}_m(s)$ の η_t の係数は，$\bar{y}_m(-s)$ の η_{-t} の係数と同じである．

本サブルーチンにおいて制限された m と l についての具体的な平滑化公式を示す．

m= l , l = 3 に関する平滑化公式 ; (r = 1).

$$\bar{y}_1(-1) = \frac{1}{6}(5\eta_{-1} + 2\eta_0 - \eta_1)$$

$$\bar{y}_1(0) = \frac{1}{3}(\eta_{-1} + \eta_0 + \eta_1)$$

……

m = 1, l = 5 に関する平滑化公式 ; (r = 2)

$$\bar{y}_1(-2) = \frac{1}{5}(3\eta_{-2} + 2\eta_{-1} + \eta_0 - \eta_2)$$

$$\bar{y}_1(-1) = \frac{1}{10}(4\eta_{-2} + 3\eta_{-1} + 2\eta_0 + \eta_1)$$

$$\bar{y}_1(0) = \frac{1}{5}(\eta_{-2} + \eta_{-1} + \eta_0 + \eta_1 + \eta_2)$$

……

m = 3 , l = 5 に関する平滑化公式 ; (r= 2)

$$\bar{y}_3(-2) = \frac{1}{70}(69\eta_{-2} + 4\eta_{-1} - 6\eta_0 + 4\eta_1 - \eta_2)$$

$$\bar{y}_3(-1) = \frac{1}{35}(2\eta_{-2} + 27\eta_{-1} + 12\eta_0 - 8\eta_1 + 2\eta_2)$$

$$\bar{y}_3(0) = \frac{1}{35}(-3\eta_{-2} + 12\eta_{-1} + 17\eta_0 + 12\eta_1 - 3\eta_2)$$

……..

m = 3 , l = 7 に関する平滑化公式 ; (r= 3)

$$\bar{y}_3(-3) = \frac{1}{42}(39\eta_{-3} + 8\eta_{-2} - 4\eta_{-1} - 4\eta_0 + \eta_1 + 4\eta_2 - 2\eta_3)$$

$$\bar{y}_3(-2) = \frac{1}{42}(8\eta_{-3} + 19\eta_{-2} + 16\eta_{-1} + 6\eta_0 - 4\eta_1 - 7\eta_2 + 4\eta_3)$$

$$\bar{y}_3(-1) = \frac{1}{42}(-4\eta_{-3} + 16\eta_{-2} + 19\eta_{-1} + 12\eta_0 + 2\eta_1 - 4\eta_2 + \eta_3)$$

$$\bar{y}_3(0) = \frac{1}{21}(-2\eta_{-3} + 3\eta_{-2} + 6\eta_{-1} + 7\eta_0 + 6\eta_1 + 3\eta_2 - 2\eta_3)$$

……

なお，詳細については，文献 [46] pp.228-254, [51] pp.314-363 を参照すること．

E31-21-0l01 SMLE2, DSMLE2

最小二乗近似多項式による平滑化 （不等間隔離散点）
CALL SMLE2 (X, Y, N, M, L, W, F, VW, ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して，観測値 y_i, 重み関数値 $w(x_i)$, $i = 1, 2, ..., n$ が与えられたとき，局部的な最小二乗近似多項式を用いて，これらの観測値を平滑化する．

すなわち，ある一つの観測値を平滑化するとき，全体の観測値を用いた最小二乗近似多項式を適用する代わりに，その隣接する指定された個数 l の観測値に関する指定された次数 m の局部的最小二乗近似多項式を適用して平滑化する．この手続きをすべての観測値について行うものである．

ただし，$n \geq l$, $w(x_i) \geq 0$ $(i= 1, ..., n)$, $l \geq m+ 2$, l は奇数，$m \geq 1$ であること．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列．

Y …… **入力**．観測値 y_i.
大きさ n の 1 次元配列．

N …… **入力**．観測値の個数 n.

M…… **入力**．局部的最小二乗近似多項式の次数 m.

L…… **入力**．局部的最小二乗近似多項式に用いる観測値の個数 l.

W …… **入力**．重み関数値 $w(x_i)$.
通常は $w(x_i) = 1$ として用いる．
大きさ n の 1 次元配列．

F…… **出力**．平滑値．
大きさ n の 1 次元配列．

VW…… **作業領域**．
大きさ $2l$ の 1 次元配列．

ICON…… **出力**．コンディションコード．
表 SMLE2-1 参照．

表 SMLE2-2 コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	次のいずれかであった． 1) $x_1 < x_2 < \ldots < x_{n-1} < x_n$ を満足しない． 2) $n < l$ 3) $m < 1$ または， $m+2 > l$ 4) $w(x_i)$ に負のものがある． 5) l が偶数である	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム
① SSL II... MGSSL
② FORTRAN 基本関数... なし

b. 注意
① 本サブルーチンは，真の関数が，全体にわたり唯一の多項式で近似できないが局部的にはある次数の多項式で近似できることを前提としている．
② m と l の選択にあたっては，観測値に対する物理的情報と利用者の経験に基づいてなされるべきである．しかし，平滑化の程度は l の値に伴い増加し，m の値に伴い減少することを考慮する必要がある．
③ 本サブルーチンを繰り返し呼ぶこと，つまり，l 個に関する m 次の最小二乗近似多項式を，求まった平滑値に繰り返し適用することも可能であるが，むやみに反復させると，全体の観測値に関する m 次の最小二乗近似多項式を適用した結果に近づく傾向がある．もし反復する場合，いつ反復を止めるべきかは利用者の判断による．
④ 本サブルーチンは，等間隔な独立変数における場合も使用できるが，サブルーチン SMLE1 に比べ処理時間が長い．

c. 使用例
離散点の個数 n, 離散点 x_i, 観測値 y_i, 局部的最小二乗近似多項式の次数 m, それに用いる観測値の個数 l を入力し，各々観測値を平滑化する．

$n \leq 20, m \leq 17, w(x_i) = 1, (i = 1,...,n)$ の場合．

```
C     **EXAMPLE**
      DIMENSION X(20),Y(20),W(20),
     *          F(20),VW(40)
      READ(5,500) N,M,L
      READ(5,510) (X(I),I=1,N)
      READ(5,510) (Y(I),I=1,N)
      DO 10 I=1,N
   10 W(I)=1.0
      CALL SMLE2 (X,Y,N,M,L,W,F,VW,ICON)
      WRITE(6,600)
      WRITE(6,610) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,620) M,L
      WRITE(6,630) (X(I),Y(I),F(I),I=1,N)
      STOP
  500 FORMAT(3I5)
  510 FORMAT(5F10.0)
  600 FORMAT('1'////26X,
     *'**** SMOOTHING BY SMLE2 ****')
  610 FORMAT('0',37X,'ICON=',I5)
  620 FORMAT('0',20X,'DEGREE OF ',
     *'POLYNOMIAL =',I2/
     *' ',23X,'POINT OF SUBRANGE =',I2/
     *'0',11X,'ABSCISSA',11X,'OBSERVED ',
     *'VALUES',10X,'SMOOTHED VALUES')
  630 FORMAT(' ',10X,F10.0,10X,F15.7,10X,
     *F15.7)
      END
```

(4) 手法概要

本サブルーチンは，離散点 $x_1 < x_2 < ... < x_n$ に対する観測値 $y_i, i=1,...,n$ を特定の次数の唯一の最小二乗

近似多項式を用いて平滑化する代わりに，局部的な l 個の観測値に関する m 次の各々異った最小二乗近似多項式を適用して平滑化するものである．

すなわち，ある一つの観測値 y_k は，隣接する $l(=2r+1)$ 個の観測値 $y_{k-r}, ..., y_{k-1}, y_k, y_{k+1}, ..., y_{k+r}$ に関して，m 次の最小二乗近似多項式をあてはめ，その $x = x_k$ における値を算出することにより平滑化する．

いま，

$$\eta_s = y_i, \xi_s = x_i,\ s = i - k$$
$$k - r \le i \le k + r$$

とする（図 SMLE2-l 参照）．

これらの η_s に関する m 次の最小二乗近似多項式は (4.1) で表される．

$$\bar{y}_m(\xi_s) = \sum_{j=0}^{m} \left\{ \frac{\sum_{t=-r}^{r} w(\xi_t)\eta_t p_j(\xi_t)}{\sum_{t=-r}^{r} w(\xi_t)[p_j(\xi_t)]^2} P_j(\xi_s) \right\} \qquad (4.1)$$

ここで，

$$P_{j+1}(\xi_s) = (\xi_s - \alpha_{j+1})P_j(\xi_s) - \beta_j P_{j-1}(\xi_s)$$
$$P_0(\xi_s) = 1, P_{-1}(\xi_s) = 0 \text{ である．}$$
$$j = 0,1,2,...$$

$$\alpha_{j+1} = \sum_{t=-r}^{r} w(\xi_t)\xi_t [P_j(\xi_t)]^2 / \sum_{t=-r}^{r} w(\xi_t)[P_j(\xi_t)]^2$$
$$j = 0,1,2,...$$
$$\beta_j = \sum_{t=-r}^{r} w(\xi_t)[P_j(\xi_t)]^2 / \sum_{t=-r}^{r} w(\xi_t)[P_{j-1}(\xi_t)]^2$$
$$j = 1,2,... \qquad (4.2)$$

(4.1) については，サブルーチン LESQ1 の手法概要を参照されたい．

そこで，y_k は (4.3) で平滑化する．

$$\bar{y}_m(\xi_0) = \sum_{j=0}^{m} \left\{ \frac{\sum_{t=-r}^{r} w(\xi_t)\eta_t P_j(\xi_t)}{\sum_{t=-r}^{r} w(\xi_t)[P_j(\xi_t)]^2} P_j(\xi_0) \right\} \qquad (4.3)$$

ここで，y_k が $y_1, ..., y_r$ と $y_{n-r+1}, ..., y_n$ のときは，両側に等しく r 個の観測値を持たないので $\bar{y}_m(\xi_0)$ で平滑化できない．そこで $y_1, ..., y_r$ に対しては $\bar{y}_m(\xi_{-r}), ..., \bar{y}_m(\xi_{-1})$ で，$y_{n-r+1}, ..., y_n$ に対しては $\bar{y}_m(\xi_1), ..., \bar{y}_m(\xi_r)$ で平滑化する．

なお，詳細については，文献 [46] pp.228 – 254, [51] pp.314 – 363 を参照すること．

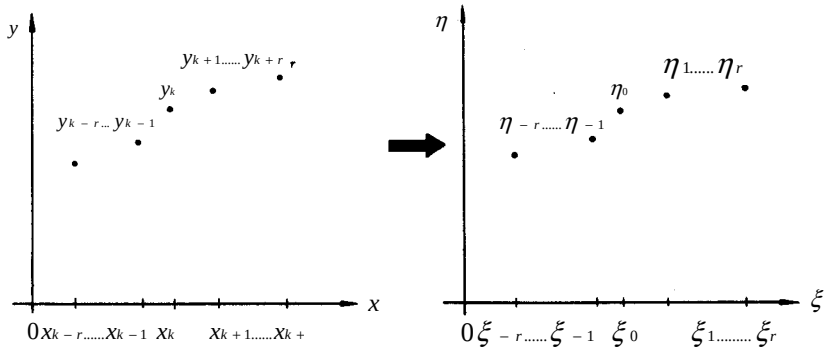


図 SMLE2-1 (x_i, y_i) と (η_s, ξ_s) の対応

E11-21-0101 SPLV, DSPLV

3 次 spline 補間式による補間・数値微分
CALL SPLV (X, Y, N, ISW, DY, V, M, DV, K, VW, ICON)

(1) 機能

離散点 $x_1, x_2, ..., x_n$ ($x_1 < x_2 < ... < x_n$) に対して，関数値 $y_i = f(x_i)$, $i = 1, 2, ..., n$ が与えられたとき，3 次 spline 補間式により，$x = v_i$, $i = 1, 2, ..., m$ における補間値，1 階微分値，2 階微分値を求める．

ただし， $n \geq 3, x_1 \leq v_1, v_2, ..., v_m \leq x_n, m \geq 1$ であること．

なお，離散点の両端点 ($x = x_1$, $x = x_n$) における微係数に関する境界条件を指定することができる．

(2) パラメタ

X ····· **入力**．離散点 x_i.
大きさ n の 1 次元配列.

Y ····· **入力**．関数値 y_i.
大きさ n の 1 次元配列.

N ····· **入力**．離散点の個数 n.

ISW ····· **入力**．制御情報.
大きさ 2 の 1 次元配列.
両端点の境界条件の型を指定する.
ISW (1), ISW (2) は共に 1, 2, 3, 4 のいずれかであること．更にこれに関連して DY (1) , DY (2) に実際に微係数を入力しなければならない.
詳細は以下のとおりである.
［境界条件の与え方］
ISW(1)=1のとき, DY(1)=$f''(x_1)$
=2のとき, DY(1) = $f'(x_1)$
=3のとき, DY(1)=$f''(x_1) / f''(x_2)$
=4のとき, DY(1) は入力不要
ISW(2)=1のとき, DY(2)=$f''(x_n)$
=2のとき, DY(2) = $f'(x_n)$
=3のとき, DY(2) =$f'(x_n) / f''(x_{n-1})$
=4のとき, DY(2) は入力不要
なお，ISW(1) = 4 （若しくは ISW(2)=4）のときは，$f'(x_1)$（若しくは $f'(x_n)$）を3 次の（n = 3 のときは2 次の）補間多項式により近似し，その値を境界条件としている.

DY ····· **入力**．両端点における微係数.
大きさ 2 の 1 次元配列.
（パラメタ ISW 参照）.

V ····· **入力**．補間値などを求めたい点
v_i , $i = 1,..., m$.
大きさ m の 1 次元配列.

M ····· **入力**．補間値などを求めたい点の個数 m .

DV ····· **出力**． v_i における補間値，1 階微分値，2 階微分値.
DV (K,3)なる 2 次元配列.
I = 1,2, ..., M とすると，
DV (I,1) には v_i における補間値，
DV (I,2) には v_i における 1 階微分値，
DV (I,3) には v_i における 2 階微分値，
をそれぞれ出力する.

K ····· **入力**．配列 DV の整合寸法 (≥M).

VW ····· **作業領域**．大きさ $2n$ の 1 次元配列.

ICON ····· **出力**．コンディションコード.
表 SPLV-1 参照.

表 SPLV-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	次のいずれかであった. ① N<3 ② $x_i \geq x_{i+1}$ なる x_i がある. ③ ISW (1) 又は ISW (2) が 1, 2, 3, 4 のいずれでもない. ④ M < 1 ⑤ $v_i < x_1$ 又は $x_n < v_i$ なる v_i がある. ⑥ K < M	処理を打ち切る

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... MGSSL, USPL
② FORTRAN 基本関数... なし

b. 注意
両端点の微係数が分からないときは ISW(1) と ISW(2) は共に 4 と指定すればよい.

c. 使用例
離散点 x_i, 関数値 y_i, $i = 1, 2, ..., n$ と境界条件, ISW(1), DY(1), ISW(2), DY(2) を入力し,

$$v_{2i-1} = x_i, \qquad i = 1,...,n$$
$$v_{2i} = (x_i + x_{i+1})/2, \qquad i = 1,...,n-1$$

の点における補間値，1 階微分値，2 階微分値を求める． $n \leq 10$ の場合.

```
C       **EXAMPLE**
        DIMENSION X(10),Y(10),ISW(2),DY(2),
       *          V(50),DV(50,3),VW(20)
        READ(5,500) N
        READ(5,510) (X(I),Y(I),I=1,N)
        WRITE(6,600) (I,X(I),Y(I),I=1,N)
        READ(5,520) (ISW(I),DY(I),I=1,2)
        WRITE(6,610) (ISW(I),DY(I),I=1,2)
        N1=N-1
        DO 10 I=1,N1
        V(2*I-1)=X(I)
   10 V(2*I)=0.5*(X(I)+X(I+1))
        M=2*N-1
        V(M)=X(N)
        CALL SPLV(X,Y,N,ISW,DY,V,M,DV,50,
       *          VW,ICON)
        WRITE(6,620) ICON
        IF(ICON.EQ.30000) STOP
        WRITE(6,630) (I,V(I),(DV(I,J),
```

```
     *                J=1,3),I=1,M)
      STOP
  500 FORMAT(I5)
  510 FORMAT(2F10.0)
  520 FORMAT(I5,F10.0)
  600 FORMAT('1'//10X,'INPUT DATA'//
     * 20X,'NO.',10X,'X',17X,'Y'//
     * (20X,I3,3X,E15.7,3X,E15.7))
  610 FORMAT(/10X,'BOUNDARY COND.'/
     * 20X,'ISW(1)=',I3,',DY(1)=',E15.7/
     * 20X,'ISW(2)=',I3,',DY(2)=',E15.7/)
  620 FORMAT(10X,'RESULTS'/20X,'ICON=',
     * I5/)
  630 FORMAT(20X,'NO.',10X,'V',17X,'Y(V)',
     * 'Y''(V)',13X,'Y''''(V)'//
     * (20X,I3,4(3X,E15.7)))
      END
```

(4) 手法概要

離散点 $x_1, x_2, ..., x_n$ ($x_1 < x_2 < ... < x_n$) に対して，関数値 $y_i = f(x_i)$, $i = 1, 2, ..., n$ が与えられたとき，$[x_1, x_n]$ 内の任意の点 v における補間値，1 階微分値，2 階微分値を 3 次 spline 補間式により求めることを考える．

3 次 spline 補間式とは次の条件を満たす，区間 $[x_1, x_n]$ で定義される補間式 $S(x)$ をいう．

a. $S(x)$ は区間 $[x_i, x_{i+1}]$ i= 1..., $n-1$ で高々3 次の多項式である．

b. $S(x)$ は区間 $[x_1, x_n]$ で，その 2 階導関数までが連続である．すなわち $S(x) \in C^2[x_1, x_n]$.

c. $S(x_i) = y_i$, $i = 1, 2, ..., n$

ここで $S(x)$ を具体的に求める．説明の都合上，$S(x)$ を区分的に次の (4.1) で表す．

$$\begin{aligned} S(x) &= S_i(x) \\ &= y_i + c_i(x-x_i) + d_i(x-x_i)^2 + e_i(x-x_i)^3 \\ & x_i \le x \le x_{i+1}, \quad i = 1,...,n-1 \end{aligned} \tag{4.1}$$

上記 a.～c.の条件は，$S_i(x)$ を使って表せば次のようになる．

$$\left.\begin{aligned} &S_i(x_i) = y_i, && i = 1,...,n-1 \\ &S_{i-1}(x_i) = y_i, && i = 2,...,n \\ &S'_{i-1}(x_i) = S'_i(x_i), && i = 2,...,n-1 \\ &S''_{i-1}(x_i) = S''_i(x_i), && i = 2,...,n-1 \end{aligned}\right\} \tag{4.2}$$

よって，(4.1) における係数 c_i, d_i, e_i は (4.2) の条件により決定される．それを (4.3) に示す．

$$\left.\begin{aligned} &i = 1,...,n-1 \text{のとき}, \\ &\quad c_i = \frac{y_{i+1}-y_i}{h_i} - \frac{h_i}{6}(y''_{i+1} + 2y''_i) \\ &\quad d_i = y''_i / 2 \\ &\quad e_i = \frac{y''_{i+1} - y''_i}{6h_i} \\ &\text{ただし}, \quad h_i = x_{i+1} - x_i \\ &\qquad\quad y''_i = S''(x_i) \end{aligned}\right\} \tag{4.3}$$

y''_i は (4.2) の 3 番目の条件より，(4.4) の 3 項関係を満たすものである．

$$\begin{aligned} &h_{i-1}y''_{i-1} + 2(h_{i-1} + h_i)y''_i + h_i y''_{i+1} \\ &\quad = 6\left(\frac{y_{i+1}-y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}\right) \\ &\quad , i = 2,...,n-1 \end{aligned} \tag{4.4}$$

したがって，(4.4) より y''_i を解けば (4.3) の c_i, d_i, e_i は求まることになる．

ところで (4.4) は，n 個の未知数 y''_i, i=1,...,n に対して (n - 2) 個の式しかなく，それを一意的に解くには y''_i についての式をあと二つ必要とする．本サブルーチンではこれを境界条件として，利用者が与えるようにしてある．

境界条件の与え方

(4.4) を解くための境界条件として，次の (4.5) 及び (4.6) の形の式が与えられるものと考える．

$$\lambda_1 y''_1 + \mu_1 y''_2 = d_1 \tag{4.5}$$

$$\lambda_n y''_{n-1} + \mu_n y''_n = d_n \tag{4.6}$$

これらの定数 λ_1, μ_1, d_1, λ_n, μ_n, d_n はいろいろ指定することができるが，本サブルーチンでは，以下の場合だけを想定している．

(a) $f''(x_1)$ を指定する(ISW(1) = 1, DY(1) =$f''(x_1)$).
このとき (4.5) は $y_1'' = f''(x_1)$ となる．

(b) $f'(x_1)$ を指定する (ISW(1) =2, DY(1) = $f'(x_1)$).
このとき (4.5) に対応する式は次のようにして作る．(4.1) , (4.3) によってきまる $S_1(x_1)$ の $x = x_1$ における 1 階微係数, $S'_1(x_1)$ は,

$$S'_1(x_1) = \frac{y_2 - y_1}{h_1} - h_1\left(\frac{2y''_1 + y''_2}{6}\right)$$

となるから， $S'_1(x_1) = f'(x_1)$ と置くことにより，

$$2y''_1 + y''_2 = \frac{6}{h_1}\left(\frac{y_2 - y_1}{h_1} - f'(x_1)\right)$$

を得る．

(c) $f''(x_1)/f''(x_2)$ を指定する (ISW(1) =3, DY (1)= $f''(x_1)/f''(x_2)$).
このとき， $y''_1 / y''_2 = f''(x_1) / f''(x_2)$ と置くことにより (4.5) は,

$$y''_1 - \left(f''(x_1) / f''(x_2)\right) y''_2 = 0$$

となる．

(d) $f'(x_1)$ を補間多項式により近似するよう指定する（ISW(1) =4, DY(1) には入力不要）．
このとき，$f'(x_1)$を，x_1, x_2, x_3, x_4 の 4 点を使った補間多項式により近似し，その後 (b) の場合をあてはめる．（ただし，与えられた離散点が 3 点しかない場合は $f'(x_1)$ はその 3 点を使って近似される）．

以上四つは，(4.5) に対応する式の与え方であるが，(4.6) に対応する式の与え方も同様な考えに基づいている．それを以下に簡略して述べる．

(e) $f''(x_n)$ を指定する (ISW(2) = 1, DY(2) = $f''(x_n)$) .

(f) $f'(x_n)$ を指定する (ISW(2) = 2, DY(2) = $f'(x_n)$) .
ちなみに，このとき (4.6) は，

$$y''_{n-1} + 2y''_n = \frac{6}{h_{n-1}}\left(f'(x_n) - \frac{y_n - y_{n-1}}{h_{n-1}} \right)$$

となる．

(g) $f''(x_n)/f''(x_{n-1})$ を指定する (ISW(2) = 3, DY (2) = $f''(x_n)/f''(x_{n-1})$).

(h) $f'(x_n)$ を近似して (f) をあてはめる（ISW(2) = 4, DY(2) には入力不要）．

これらの(a) ～ (d)，(e) ～ (h) は適当に組合せて指定することができる．例えば，(a) と (e), (a) と (g), (b) と (h) ,...という具合である．

補間値などの計算

区間 $[x_1, x_n]$ 内の任意の点 v における補間値，1 階微分値，2 階微分値は，$x_i \le v < x_{i+1}$ を満たす区間 $[x_i, x_{i+1}]$ で定義される $S_i(x)$, 及び $S'_i(x)$, $S''_i(x)$ を評価することにより求める．

なお，詳細については参考文献 [48] を参照すること．

A21-12-0201 SSSM, DSSSM

行列の差（実対称行列）
CALL SSSM (A, B, C, N, ICON)

(1) 機能

$n \times n$ の実対称行列 ***A*** と実対称行列 ***B*** の差 ***C*** を求める．

$$\boldsymbol{C} = \boldsymbol{A} - \boldsymbol{B}$$

ここで，***C*** は $n \times n$ の実対称行列である．

$n \geq 1$ であること．

(2) パラメタ

A …… **入力**．行列 ***A***.
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列.

B…… **入力**．行列 ***B***.
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列.

C…… **出力**．行列 ***C***.
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列.
（使用上の注意①参照）．

N …… **入力**．行列 ***A***, ***B*** 及び ***C*** の次数 *n*.

ICON…… **出力**．コンディションコード．
表 SSSM-1 参照．

表 SSSM-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	N < 1であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... MGSSL

② FORTRAN 基本関数... なし.

b. 注意

① 配列 A あるいは，B 上の内容を保存する必要のない場合は，次のように呼び出すことにより領域が節約できる．

- 配列 A の内容が不要の場合.
  CALL SSSM (A, B, A, N, ICON)
- 配列 B の内容が不要の場合.
  CALL SSSM (A, B, B, N, ICON)

この場合，行列 ***C*** は配列 A あるいは，B 上に得られる．

c. 使用例

実対称行列 ***A***, ***B*** の差を求める．$n \leq 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),B(5050),C(5050)
      CHARACTER*4 IA,IB,IC
      DATA IA/'A   '/,IB/'B   '/,
     *     IC/'C   '/
   10 READ(5,100) N
      IF(N.EQ.0) STOP
      WRITE(6,150)
      NT=N*(N+1)/2
      READ(5,200) (A(I),I=1,NT)
      READ(5,200) (B(I),I=1,NT)
      CALL SSSM(A,B,C,N,ICON)
      IF(ICON.NE.0) GOTO 10
      CALL PSM(IA,1,A,N)
      CALL PSM(IB,1,B,N)
      CALL PSM(IC,1,C,N)
      GOTO 10
  100 FORMAT(I5)
  200 FORMAT(4E15.7)
  150 FORMAT('1'///10X,
     * '** MATRIX SUBTRACTION **')
      END
```

本使用例中のサブルーチン PSM は，実対称行列を印刷するものである．プログラムは，サブルーチン MGSM の使用例に記載されている．

B21-21-0602 TEIG1, DTEIG1

実対称 3 重対角行列の固有値及び固有ベクトル（QL 法）
CALL TEIG1 (D, SD, N, E, EV, K, M, ICON)

(1) 機能

n 次の実対称 3 重対角行列 ***T*** の固有値及び固有ベクトルを QL 法により求める．

固有ベクトルは $\|\boldsymbol{x}\|_2=1$ となるように正規化する．*n*≥1 なること．

(2) パラメタ

D……**入力**．実対称 3 重対角行列 ***T*** の主対角要素．
大きさ *n* の 1 次元配列．
演算後，内容は保存されない．

SD……**入力**． 3 重対角行列 ***T*** の副対角要素．
SD (2)～SD (*n*)に副対角要素を格納すること．
大きさ *n* の 1 次元配列．
演算後，内容は保存されない．

N……**入力**． 3 重対角行列 ***T*** の次数 *n*．

E ……**出力**．固有値．
大きさ *n* の 1 次元配列．

EV ……**出力**．固有ベクトル．
固有ベクトルは列方向に格納される．
EV(K, N) なる 2 次元配列．

K……**入力**．配列 EV の整合寸法．

M ……**出力**．求まった固有値・固有ベクトルの数．

ICON……**出力**．コンディションコード．
表 TEIG1-1 参照．

表 TEIG1-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N=1 であった．	E(1) = D(1), EV(1,1)=1.0 とする．
15000	固有値固有ベクトルのすべてを求めつくすことはできなかった．	M に求まっている固有値固有ベクトルの数をセットする．
20000	固有値固有ベクトルは 1 つも求められなかった．	M = 0 とする．
30000	N < 1 又は K < N であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... AMACH, MGSSL

② FORTRAN 基本関数 .. ABS, SIGN, SQRT

b. 注意

① 固有値固有ベクトルは，固有値の求まった順に格納される．

② パラメタ M には，ICON=0 のとき *n* が，また，ICON=15000 のとき，求まった固有値及び固有ベクトルの数がセットされている．

③ 本サブルーチンは，実対称 3 重対角行列用である．実対称行列の固有値及び固有ベクトルを求める場合は，サブルーチン SEIG1 を用いること．

④ 実対称 3 重対角行列 ***T*** の固有値だけを求める場合は，サブルーチン TRQL を用いること．

c. 使用例

n 次の実対称 3 重対角行列 ***T*** の固有値及び固有ベクトルを求める．

$n \le 100$ の場合．

```
C     **EXAMPLE**
      DIMENSION D(100),SD(100),
     *          EV(100,100),E(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      READ(5,510) (D(I),SD(I),I=1,N)
      WRITE(6,600) N,(I,D(I),SD(I),I=1,N)
      CALL TEIG1(D,SD,N,E,EV,100,M,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL SEPRT(E,EV,100,N,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(2E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',5X,
     * 'ORDER=',I3/'0',20X,
     * '***DIAGONAL***',
     * 5X,'**SUBDIAGONAL*'//(13X,I3,5X,
     * 2(E14.7,5X)))
  610 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値固有ベクトルを出力するサブルーチンである．その詳細についてはサブルーチン SEIG1 の使用例参照のこと．

(4) 手法概要

n 次の実対称 3 重対角行列 ***T*** の固有値及び固有ベクトルを QL 法により求める．

QL 法により固有値を求める方法は，TRQL と同様であるので，TRQL「(4)手法概要」を参照すること．ここでは，QL 法により固有ベクトルを求める方法について説明する．

3 重対角行列 ***T*** の固有ベクトルは，(4.1)の直交相似変換により ***T*** を対角行列 ***D*** に変換する直交行列 ***Q*** の列ベクトルである．

$$\boldsymbol{D}=\boldsymbol{Q}^T\boldsymbol{T}\boldsymbol{Q} \qquad (4.1)$$

ところで，QL 法は， (4.2)の直交相似変換を反復的に行うことにより，***T*** を ***D*** に変換して，その対角要素として固有値を求める方法である．

$$\boldsymbol{T}_{s+1}=\boldsymbol{Q}_s^{\mathrm{T}}\boldsymbol{T}_s\boldsymbol{Q}_s \quad s=1,2,3,\ldots \qquad (4.2)$$

ここで $\boldsymbol{Q_s}$ は，(4.3)の QL 分解により得られる直交行列である．

$$\left(\boldsymbol{T}_s - k_s \boldsymbol{I}\right) = \boldsymbol{Q}_s \boldsymbol{L}_s \tag{4.3}$$

（ただし，k_s は原点移動量，$\boldsymbol{L}_s$ は下三角行列）

したがって，m 回の反復で対角行列に収束したときは，(4.2)より，

$$\boldsymbol{D} = \boldsymbol{Q}_m^T \boldsymbol{Q}_{m-1}^T \ldots \boldsymbol{Q}_2^T \boldsymbol{Q}_1^T \boldsymbol{T}_1 \boldsymbol{Q}_1 \boldsymbol{Q}_2 \ldots \boldsymbol{Q}_{m-1} \boldsymbol{Q}_m \tag{4.4}$$

となっている．ここに $T_1 = T$ である．

(4.1), (4.4)より，固有ベクトルは，

$$\boldsymbol{Q} = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \ldots \boldsymbol{Q}_{m-1} \boldsymbol{Q}_m \tag{4.5}$$

の列ベクトルとして得られる．

いま，$\boldsymbol{Q}^{(s-1)}$ を，

$$\boldsymbol{Q}^{(s-1)} = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \ldots\ldots \boldsymbol{Q}_{s-1} \tag{4.6}$$

と定義する．すると，第 s 回までの変換行列の積 $\boldsymbol{Q}^{(s)}$ は，

$$\boldsymbol{Q}^{(s)} = \boldsymbol{Q}^{(s-1)} \boldsymbol{Q}_s \tag{4.7}$$

により求めることができる．

(4.7)の $\boldsymbol{Q}_s$ は，QL 法では，

$$\boldsymbol{Q}_s = \boldsymbol{P}_{n-1}^{(s)} \boldsymbol{P}_{n-2}^{(s)} \ldots \boldsymbol{P}_2^{(s)} \boldsymbol{P}_1^{(s)} \tag{4.8}$$

により得られる．(4.7)に(4.8)を代入すれば，

$$\boldsymbol{Q}^{(s)} = \boldsymbol{Q}^{(s-1)} \boldsymbol{P}_{n-1}^{(s)} \boldsymbol{P}_{n-2}^{(s)} \ldots \boldsymbol{P}_2^{(s)} \boldsymbol{P}_1^{(s)} \tag{4.9}$$

を得る．この操作を $s = 1, 2, ..., m$ と繰り返すことにより，$\boldsymbol{Q}$ を求めることができる．

(4.9)のようにすることにより，QL 法の各変換段階で変換行列の積を作ってゆき，全固有値が求まった（すなわち対角化が完了した）時点で，$\boldsymbol{Q}$（すなわち固有ベクトル）も求まっている．

なお，QL 法は，一つの固有値に対し 30 回反復しても収束しなければ,処理を中断するが，この時点までに求まっている M 個の固有値及び固有ベクトルは，正しい固有値及び固有ベクトルとして利用できる．

以上の方法で求めた固有ベクトルは，$\boldsymbol{Q}$ が直交行列であるので，$\|\boldsymbol{x}\|_2$=1 となっている．

なお，詳細については，参考文献[12], [13] pp. 227-248 及び [16] pp. 177-206 を参照すること．

B21-21-0702 TEIG2, DTEIG2

実対称 3 重対角行列の固有値及び固有ベクトル （バイセクション法，逆反復法）
CALL TEIG2 (D, SD, N, M, E, EV, K, VW, ICON)

(1) 機能

n 次の実対称 3 重対角行列 ***T*** の大きい方から又は小さい方から *m* 個の固有値と対応する固有ベクトルを求める．

固有値はバイセクション法により，固有ベクトルは逆反復法により求める．固有ベクトルは $\|\boldsymbol{x}\|_2=1$ となるように正規化される．

$1 \le m \le n$ なること．

(2) パラメタ

D……**入力**．実対称 3 重対角行列 ***T*** の主対角要素．大きさ *n* の 1 次元配列．

SD……**入力**．実対称 3 重対角行列 ***T*** 副対角要素．大きさ *n* の 1 次元配列．
ただし，SD (2)～SD (*n*)に副対角要素を格納しておくこと．

N……**入力**．3 重対角行列 ***T*** の次数 *n*．

M……**入力**．求める固有値の個数 *m*．
M = + *m* のときは大きい方から求める．
M = - *m* のときは小さい方から求める．

E……**出力**．固有値．
大きさ *m* の 1 次元配列．

EV……**出力**．固有ベクトル．
固有ベクトルは列方向に格納される．
EV (K, *m*) なる 2 次元配列．

K……**入力**．配列 EV の整合寸法．

VW……**作業領域**．大きさ 5*n* の 1 次元配列．

ICON……**出力**．コンディションコード．
表 TEIG2-1 参照．

表 TEIG2-1 コンディションコード

コード	意味	処理内容
0	エラーなし．	
10000	N=1 であった．	E(1)=D(1), EV (1, 1) = 1.0 とする．
15000	*m* 個の固有値を求めた後，固有ベクトルを求めるとき，求まらないものがあった．	その固有ベクトルを 0 ベクトルとする．
20000	固有ベクトルが 1 つも求まらなかった．	固有ベクトルはすべて 0 ベクトルである．
30000	M = 0, N < |M| 又は，K < N であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II....AMACH, MGSSL, UTEG2

② FORTRAN 基本関数.....ABS, AMAX1, IABS

b. 注意

① 本サブルーチンは実対称 3 重対角行列用である．実対称行列の *m* 個の固有値及び固有ベクトルを求める場合は，サブルーチン SEIG2 を用いること．

② 3 重対角行列の *m* 個の固有値だけを求める場合は，サブルーチン BSCT1 を用いること．

c. 使用例

n 次の実対称 3 重対角行列の固有値を大きい方から又は小さい方から *m* 個求め，対応する固有ベクトルを求める．

$n \le 100, m \le 10$ の場合．

```
C     **EXAMPLE**
      DIMENSION D(100),SD(100),E(10),
     *          EV(100,10),VW(500)
   10 CONTINUE
      READ(5,500) N,M
      IF(N.EQ.0) STOP
      READ(5,510) (D(I),SD(I),I=1,N)
      WRITE(6,600) N,M,(I,D(I),SD(I),
     *             I=1,N)
      CALL TEIG2(D,SD,N,M,E,EV,100,VW,
     *           ICON)
      WRITE(6,610) ICON
      IF(ICON.NE.0) GO TO 10
      MM=IABS(M)
      CALL SEPRT(E,EV,100,N,MM)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(2E15.7)
  600 FORMAT('1',5X,'ORIGINAL MATRIX',5X,
     * 'N=',I3,'M=',I3/'0',20X,
     * '***DIAGONAL***','**SUBDIAGONAL*'//
     * (13X,I3,5X,2(E14.7,5X)))
  610 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値固有ベクトルを出力するサブルーチンである．その詳細についてはサブルーチン SEIG1 の使用例参照のこと．

(4) 手法概要

n 次の実対称 3 重対角行列の大きい方から又は小さい方から *m* 個の固有値及び対応する固有ベクトルを求める．

m 個の固有値は，バイセクション法により求め，対応する固有ベクトルは逆反復法により求める．

バイセクション法については，サブルーチン BSCT1 を参照されたい．

ここでは逆反復法についてだけ説明する．

いま 3 重対角行列 ***T*** の真の固有値は，

$$\lambda_1 > \lambda_2 > \lambda_3 \cdots\cdots > \lambda_n \qquad (4.1)$$

となっているとする.

バイセクション法により λ_j の近似解としてμ_j が得られたとき，逆反復法で対応する固有ベクトルを求めることを考える.

逆反復法は,

$$(\boldsymbol{T}-\mu_j\boldsymbol{I})\,\boldsymbol{x}_r=\boldsymbol{x}_{r-1},\ r=1,2,\ldots \tag{4.2}$$

（$\boldsymbol{x}_0$は適当な初期ベクトルとする）

を反復的に解いてゆき，収束条件を満足したときの $\boldsymbol{x}_r$ を固有ベクトルとする方法である.

真の固有値 $\lambda_1,\lambda_2,\ldots,\lambda_n$ に対応する真の固有ベクトルを $\boldsymbol{u}_1,\boldsymbol{u}_2,\ldots,\boldsymbol{u}_n$ とすれば，適当な初期ベクトル $\boldsymbol{x}_0$ は,

$$\boldsymbol{x}_0=\sum_{i=1}^{n}\alpha_i\boldsymbol{u}_i \tag{4.3}$$

なる 1 次結合で表すことができる.

(4.2)及び (4.3)より$\boldsymbol{x}_r$は,

$$\boldsymbol{x}_r=\frac{1}{(\lambda_j-\mu_j)^r}\left[\alpha_j\boldsymbol{u}_j+\sum_{\substack{i=1\\ i\neq j}}^{n}\alpha_i\boldsymbol{u}_i(\lambda_j-\mu_j)^r/(\lambda_i-\mu_j)^r\right] \tag{4.4}$$

と表せる.

一般に，$\left|\left(\lambda_j-\mu_j\right)/\left(\lambda_i-\mu_j\right)\right|<<1.0$ と見なせるから，(4.4)は，$a_j\neq 0$ であれば，r が大きくなる程，$\boldsymbol{x}_r$ は，$\alpha_j\boldsymbol{u}_j$に接近することを示している.

(4.2)の連立方程式は，$(\boldsymbol{T}-\mu_j\boldsymbol{I})$ を単位下三角行列 $\boldsymbol{L}$ と上三角行列 $\boldsymbol{U}$ とに分解し,

$$\boldsymbol{L}\boldsymbol{U}\boldsymbol{x}_r=\boldsymbol{P}\boldsymbol{x}_{r-1} \tag{4.5}$$

($\boldsymbol{P}$ は，軸交換のための置換行列である．)

として解く．(4.5)を解くことは,

$$\boldsymbol{L}\boldsymbol{y}_{r-1}=\boldsymbol{P}\boldsymbol{x}_{r-1}\quad\text{（前進代入）} \tag{4.6}$$

$$\boldsymbol{U}\boldsymbol{x}_r=\boldsymbol{y}_{r-1}\quad\text{（後退代入）} \tag{4.7}$$

なる二つの連立方程式を解くことに帰着する.

初期ベクトル $\boldsymbol{x}_0$はどのように与えてもよいから,

$$\boldsymbol{y}_0=\boldsymbol{L}^{-1}\boldsymbol{P}\boldsymbol{x}_0 \tag{4.8}$$

なる $\boldsymbol{y}_0$が特定の形，例えば $\boldsymbol{y}_0=(1,\,1,\,\ldots\,1)^T$ となるように $\boldsymbol{x}_0$ が与えられたとしてよい．したがって，第 1 回目には，(4.6)の前進代入は省略できる．一般には 2 回目以降は前進代入と後退代入をくり返すことにより固有ベクトルを求めることができる.

① 逆反復法における初期ベクトルと収束判定について

本ルーチンでは，初期ベクトル $\boldsymbol{y}_0$ として,

$$\boldsymbol{y}_0=\left(\sqrt{n}\cdot u\|\boldsymbol{T}\|,\sqrt{n}\cdot u\|\boldsymbol{T}\|,\ldots\sqrt{n}\cdot u\|\boldsymbol{T}\|\right)^T \tag{4.9}$$

ただし，u：丸め誤差単位

$$\|\boldsymbol{T}\|=\sum_{i=1}^{n}\left(|t_{ii}|+|t_{i-1i}|\right)$$

$$\boldsymbol{T}=(t_{ij}),\qquad t_{01}=0$$

を与え，各反復段階で，$\boldsymbol{x}_{r\text{-}1}$ を

$$\|\boldsymbol{x}_{r-1}\|_1=\|\boldsymbol{y}_0\|_2 \tag{4.10}$$

となるように調整し，$\boldsymbol{x}_r$ が,

$$\|\boldsymbol{x}_r\|_1\geq 1 \tag{4.11}$$

を満足したとき，$\boldsymbol{x}_r$ を固有ベクトルとして採用する．(4.11)は，次のようにして導かれる．(4.2)において，$\boldsymbol{x}_r$ を正規化すれば,

$$\left(\boldsymbol{T}-\mu_j\boldsymbol{I}\right)\boldsymbol{x}_r/\|\boldsymbol{x}_r\|_1=\boldsymbol{x}_{r-1}/\|\boldsymbol{x}_r\|_1 \tag{4.12}$$

となる，(4.12)の右辺は，残差ベクトルと見なせるから，これが零と見なせる程度に減少したとき，$\boldsymbol{x}_r$ は固有ベクトルに収束したとすればよい.

そこで，残差ベクトルのノルムが,

$$\|\boldsymbol{x}_{r-1}\|_1/\|\boldsymbol{x}_r\|_1\leq\|\boldsymbol{y}_0\|_2 \tag{4.13}$$

を満足したとき，(4.12)の右辺は零になったと見なすことにする．このとき $\boldsymbol{x}_{r\text{-}1}$ を (4.10)を満足するよう調整しておけば，(4.13)は，(4.11)となる.

なお，本サブルーチンでは，5 回反復しても $\boldsymbol{x}_5$ が (4.11)を満足しなかったときは，固有ベクトルは求まらなかったと見なし ICON=15000 として，対応する EV の列をすべて零とする.

② 固有ベクトルの直交化について

実対称 3 重対角行列の固有ベクトルは，すべて直交するはずであるが，逆反復法で求めた固有ベクトルは，固有値が互いに接近するにつれ，直交性が失なわれるという欠点を持っている.

そこで，本サブルーチンでは，固有ベクトルの直交性を確保するため，次のように対処している.

まず，対応する固有ベクトルを求めようとする固有値μ_i と直前の固有値$\mu_{i\text{-}1}$ が(4.15)を満足するか調べる.

$$\mu_i-\mu_{i-1}\leq 10^{-3}\|\boldsymbol{T}\| \tag{4.15}$$

もし，(4.15)を満足しているときは，μ_i について逆反復法で求めた固有ベクトル $\boldsymbol{x}_i$ を$\mu_{i\text{-}1}$ について求めた固有ベクトル $\boldsymbol{x}_{i\text{-}1}$ に対して,

$$(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}) = 0 \tag{4.16}$$

を満足するするように修正する．
以下同様の考え方により，連続して(4.15)の関係を満足する固有値を一つのグループとして，対応する固有ベクトルの直交化を行う．

③ 直和分解について
いま，$\boldsymbol{T}$ が $\boldsymbol{T}_1, \boldsymbol{T}_2, \ldots \boldsymbol{T}_m$ なる m 個の小行列の直和に分解できたとする．このとき，$\boldsymbol{T}$ の固有値及び固有ベクトルは，それぞれ(4.17) の $\boldsymbol{D}$ の対角要素及び $\boldsymbol{Q}$ の列ベクトルである．

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{D}_1 & & & \\ & \boldsymbol{D}_2 & & 0 \\ & & \cdots & \\ & 0 & & \boldsymbol{D}_m \end{bmatrix}, \quad \boldsymbol{Q} = \begin{bmatrix} \boldsymbol{Q}_1 & & & \\ & \boldsymbol{Q}_2 & & 0 \\ & & \cdots & \\ & 0 & & \boldsymbol{Q}_m \end{bmatrix} \tag{4.17}$$

ここに，$\boldsymbol{D}$ 及び $\boldsymbol{Q}$ は，

$$\boldsymbol{D} = \boldsymbol{Q}^T \boldsymbol{T} \boldsymbol{Q} \tag{4.18}$$

を満足する $\boldsymbol{D}$ 及び $\boldsymbol{Q}$ である．
(4.17)及び(4.18)より，

$$\left.\begin{aligned} \boldsymbol{D}_1 &= \boldsymbol{Q}_1^T \boldsymbol{T}_1 \boldsymbol{Q}_1 \\ \boldsymbol{D}_2 &= \boldsymbol{Q}_2^T \boldsymbol{T}_2 \boldsymbol{Q}_2 \\ &\cdots \\ \boldsymbol{D}_m &= \boldsymbol{Q}_m^T \boldsymbol{T}_m \boldsymbol{Q}_m \end{aligned}\right\} \tag{4.19}$$

を得る．(4.19)より，$\boldsymbol{T}$ が直和分解できるときは，各小行列ごとに固有値及び固有ベクトルを求めればよいことがわかる．

なお．詳細については，参考文献[12]及び[13] pp. 418-439 を参照すること．

G21-21-0l0l TRAP, DTRAP

1 次元有限区間積分 （不等間隔離散点入力，台形則）
CALL TRAP (X, Y, N, S, ICON)

(1) 機能

不等間隔離散点 $x_1, x_2, ..., x_n$ $(x_1 < x_2 < ... < x_n)$ に対して，関数値 $y_i=f(x_i)$, i=1, 2, ..., n が与えられたとき，台形則により

$$S = \int_{x_1}^{x_n} f(x)dx \qquad n \geq 2$$

を求める．

(2) パラメタ

X …… **入力**．離散点 x_i.
大きさ n の 1 次元配列.

Y …… **入力**．関数値 y_i.
大きさ n の 1 次元配列.

N …… **入力**．離散点の個数 n

S …… **出力**．積分値 S.

ICON …… **出力**．コンディションコード.
表 TRAP-1 参照.

表 TRAP-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
30000	$n < 2$ 又は $x_i \geq x_{i+1}$ であった.	S=0.0 として処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ..... MGSSL

② FORTRAN 基本関数 ... なし.

b. 注意

離散点 x_i が等間隔の場合でも本サブルーチンは適用できるが，シンプソン則による方法（サブルーチン SIMP1）を適用した方が一般的に精度が良い.

c. 使用例

離散点 x_i，関数値 y_i を入力し，積分値 S を求める.

```
C     **EXAMPLE**
      DIMENSION X(20),Y(20)
      READ(5,500) N
      READ(5,510) (X(I),Y(I),I=1,N)
      CALL TRAP(X,Y,N,S,ICON)
      WRITE(6,600) ICON,S
      STOP
  500 FORMAT(I2)
  510 FORMAT(2F10.0)
  600 FORMAT(10X,'ILL CONDITION =',I5
     * /10X,'INTEGRAL VALUE =',E15.7)
      END
```

(4) 手法概要

区間 $[x_1, x_n]$について，不等間隔に取られた離散点を用い，台形則により積分する．まず，最初の区間 $[x_1, x_2]$の 2 点 $(x_1, f(x_2))$及び $(x_1, f(x_2))$ について積分計算を行う．次に続く 2 分点についても同様に計算し，それぞれ求まった値を加算していく．

最終的には次のようになる.

$$\int_{x_1}^{x_n} f(x)dx \approx \frac{1}{2}\{ (x_2 - x_1)(f(x_1) + f(x_2)) + (x_3 - x_2)(f(x_2) + f(x_3)) + ... + (x_n - x_{n-1})(f(x_n) + f(x_{n-1}))\} = \frac{1}{2}\{ (x_2 - x_1)f(x_1) + (x_3 - x_1)f(x_2) + ... + (x_n - x_{n-2})f(x_{n-1}) + (x_n - x_{n-1})f(x_n)\} \quad (4.1)$$

なお，詳細については，参考文献 [46] pp. 114-121 を参照すること.

B21-21-0802 TRBK, DTRBK

実対称行列の固有ベクトルへの逆変換
CALL TRBK (EV, K, N, M, P, ICON)

(1) 機能

n 次の実対称 3 重対角行列 ***T*** の *m* 個の固有ベクトルを実対称行列 ***A*** の固有ベクトルへ逆変換する.

ただし，***T*** は ***A*** にハウスホルダー法を適用して得られたものであること．かつ $1 \le m \le n$ であること.

(2) パラメタ

EV ……**入力**．実対称 3 重対角行列 ***T*** の *m* 個の固有ベクトル.
出力．実対称行列 ***A*** の固有ベクトル.
EV (K, *m*) なる２次元配列.

K……**入力**．配列 EV の整合寸法 (≥ *n*).

N……**入力**．実対称 3 重対角行列の次数 *n*.

M ……**入力**．|M|は，固有ベクトルの個数 *m*（使用上の注意④参照）.

P……**入力**．ハウスホルダー法による ***A*** から ***T*** への変換行列 ***P***（使用上の注意②参照）.
大きさ *n*(*n* +1)/2 の 1 次元配列.

ICON ……**出力**．コンディションコード.
表 TRBK-1 参照.

表 TRBK-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1 であった.	EV (1, 1) = 1.0 とする.
30000	N < |M|, K < N 又は M = 0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II... MGSSL

② FORTRAN 基本関数... IABS

b. 注意

① 本サブルーチンは，通常サブルーチン TRID1 に続けて呼び出される.

② 本サブルーチンの入力パラメタ P はサブルーチン TRID1 の出力パラメタ A が与えられることを想定しているので，それをそのまま用いればよい．配列 P の内容についてはサブルーチン TRID1 を参照のこと.

③ 固定ベクトルは，$\|\boldsymbol{x}\|_2 = 1$ となるように正規化されている.

④ パラメタ Mの内容が負であるときは，その絶対値をとって使う.

c. 使用例

n 次の実対称行列をサブルーチン TRID1 により 3 重対角行列へ変換し，サブルーチン TEIG2 により *m* 個の固有値と固有ベクトルを求め，本サブルーチンにより 3 重対角行列の固有ベクトルを実対称行列の固有ベクトルへ逆変換する． *n* ≤ l00 の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),D(100),SD(100),
     *          E(100),EV(100,100),VW(500)
   10 READ(5,500) N,M
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N,M
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL TRID1(A,N,D,SD,ICON)
      IF(ICON.EQ.30000) GOTO 10
      CALL TEIG2(D,SD,N,M,E,EV,100,VW,
     *           ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GOTO 10
      CALL TRBK(EV,100,N,M,A,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GOTO 10
      MM=IABS(M)
      CALL SEPRT(E,EV,100,N,MM)
      GOTO 10
  500 FORMAT(2I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'
     * //11X,'** ORDER =',I5,10X,'** M =',
     * I3/)
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT(/11X,'** CONDITION CODE =',
     * I5/)
      END
```

本使用例中のサブルーチン SEPRT は，実対称行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン SEIG1 の使用例参照のこと.

(4) 手法概要

n 次の実対称 3 重対角行列 ***T*** の *m* 個の固有ベクトルを，実対称行列 ***A*** の固有ベクトルへ逆変換する.

A から ***T*** への変換は，ハウスホルダー法により，(4.1)に示す *n*-2 回の直交相似変換により行なわれる.

$$\boldsymbol{T} = \boldsymbol{P}_{n-2}^{\mathrm{T}} \cdots \boldsymbol{P}_2^{\mathrm{T}} \boldsymbol{P}_1^{\mathrm{T}} \boldsymbol{A} \boldsymbol{P}_1 \boldsymbol{P}_2 \cdots \boldsymbol{P}_{n-2} \qquad (4.1)$$

ただし $\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^T / h_k$ である．（詳細はサブルーチン TRID1 参照のこと）.

T の固有値，固有ベクトルを，λ, ***y*** とすると

$$\boldsymbol{T}\boldsymbol{y} = \lambda \boldsymbol{y} \qquad (4.2)$$

が成り立つ．(4.1)を(4.2)へ代入すると

$$\boldsymbol{P}_{n-2}^{\mathrm{T}}\ldots\boldsymbol{P}_{2}^{\mathrm{T}}\boldsymbol{P}_{1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{y}=\lambda\,\boldsymbol{y} \tag{4.3}$$

となり，(4.3)の両辺に $\boldsymbol{P}_1\boldsymbol{P}_{2\ldots}\boldsymbol{P}_{n\text{-}2}$ を左側より掛けると(4.4)となる．

$$\boldsymbol{A}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{y}=\lambda\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{y} \tag{4.4}$$

したがって， $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ は(4.5)となる．

$$\boldsymbol{x}=\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{y} \tag{4.5}$$

(4.5)の計算は(4.6)において $\boldsymbol{y}=\boldsymbol{x}_{n\text{-}1}$ として行うことにより，$\boldsymbol{x}=\boldsymbol{x}_1$ として求まる．

$$\begin{aligned}\boldsymbol{x}_k &= \boldsymbol{P}_k\boldsymbol{x}_{k+1}\\ &= \boldsymbol{x}_{k+1}-\left(\boldsymbol{u}_k\boldsymbol{u}_k^{T}/h_k\right)\boldsymbol{x}_{k+1},\quad k=n-2,\ldots,2,1\end{aligned} \tag{4.6}$$

固有ベクトル $\boldsymbol{x}$ は，(4.5)により求めているので，$\|\boldsymbol{x}\|_2=1$ となっている．

なお，詳細については，参考文献[13]の pp. 212-226 を参照すること．

B21-25-0402 TRBKH, DTRBKH

エルミート行列の固有ベクトルへの逆変換
CALL TRBKH (EVR, EVI, K, N, M, P, PV, ICON)

(1) 機能

n 次の実対称 3 重対角行列 $\boldsymbol{T}$ の m 個の固有ベクトル $\boldsymbol{y}$ を，エルミート行列 $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ へ逆変換する．

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{V}^{*}\boldsymbol{y}$$

ただし $\boldsymbol{P}$ 及び $\boldsymbol{V}$ はそれぞれ，ハウスホルダー法及び対角ユニタリ変換によって $\boldsymbol{A}$ から $\boldsymbol{T}$ へ変換するときの変換行列である．$1 \le m \le n$ になること．

(2) パラメタ

EVR……**入力**．EVR は実対称 3 重対角行列 $\boldsymbol{T}$ の m 個の固有ベクトル $\boldsymbol{y}$.
出力．n 次のエルミート行列 $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ の実部.
EVR (K, m)なる 2 次元配列.
（使用上の注意④参照）.

EVI……**出力**．n 次のエルミート行列 $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ の虚部.
EVI (K, m)なる 2 次元配列.
（使用上の注意④参照）.

K……**入力**．配列 EVR, EVI の整合寸法 ($\ge n$).

N……**入力**．実対称 3 重対角行列の次数 n.

M……**入力**．|M|は，固有ベクトルの個数 m .
（使用上の注意⑤参照）.

P……**入力**．$\boldsymbol{A}$ から $\boldsymbol{T}$ へハウスホルダー法を適用するときの変換行列 $\boldsymbol{P}$.
エルミート行列用圧縮モード.
P (K, N)なる 2 次元配列.
（使用上の注意②参照）.

PV……**入力**．$\boldsymbol{A}$ から $\boldsymbol{T}$ へ対角ユニタリ変換を適用するときの変換行列 $\boldsymbol{V}$.
大きさ 2 n の 1 次元配列.
（使用上の注意②参照）.

ICON……**出力**．コンディションコード.
表 TRBKH-1 参照.

表 TRBKH-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1 であった.	EV(1, 1) = 1.0 とする.
30000	N < |M|, K < N 又は M = 0 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... MGSSL

② FORTRAN 基本関数.... IABS

b. 注意

① 本サブルーチンはエルミート行列用であり，一般複素行列については適用することができない.

② 配列 P は $\boldsymbol{A}$ から $\boldsymbol{T}$ への変換行列 $\boldsymbol{P}$ を直接に表現するものではないので，注意すること.
本サブルーチンはサブルーチン TRIDH と通常は組み合わせて使う．したがって，パラメタ P, PV の内容は TRIDH の A, PV で出力される内容を想定しているので，これをそのまま用いればよい.
配列 P, PV の内容については，サブルーチン TRIDH を参照のこと.

③ 入力した固有ベクトル $\boldsymbol{y}$ が $\|\boldsymbol{y}\|_2 = 1$ に正規化されていれば，出力する固有ベクトル $\boldsymbol{x}$ も $\|\boldsymbol{x}\|_2 = 1$ となる.

④ 第 j 番目の固有値に対応する固有ベクトルの第 l 番目の要素は EVR (L, J) + i・EVI (L, J)で表される．ただし $i = \sqrt{-1}$ である.

⑤ パラメタ M の内容が負であるときは，その絶対値をとって使う.

c. 使用例

n 次のエルミート行列をサブルーチン TRIDH により，実対称 3 重対角行列に変換し，サブルーチン TEIG2 により m 個の固有値と固有ベクトルを求め，本サブルーチンにより，実対称 3 重対角行列の固有ベクトルをエルミート行列の固有ベクトルへ逆変換する.
$n \le 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),D(100),SD(100),
     * V(200),EVR(100,100),EVI(100,100),
     * VW(500),E(100)
   10 CONTINUE
      READ(5,500) N,M
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,M
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL TRIDH(A,100,N,D,SD,V,ICON)
      WRITE(6,620) ICON
      IF(ICON.EQ.30000) GO TO 10
      CALL TEIG2(D,SD,N,M,E,EVR,100,VW,
     *           ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      CALL TRBKH(EVR,EVI,100,N,M,A,V,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
```

```
      MM=IABS(M)
      CALL HEPRT(E,EVR,EVI,100,N,MM)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(4E15.7)
  600 FORMAT('1'///40X,'**ORIGINAL ',
     * 'MATRIX**',5X,'N=',I3,5X,'M=',I3//)
  610 FORMAT(/4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT('0',20X,'ICON=',I5)
      END
```

本使用例中のサブルーチン HEPRT は，エルミート行列の固有値及び固有ベクトルを出力するサブルーチンである．その詳細については，サブルーチン HEIG2 の使用例参照のこと．

(4) 手法概要

n 次の実対称 3 重対角行列 $\boldsymbol{T}$ の m 個の固有ベクトルを，エルミート行列 $\boldsymbol{A}$ の固有ベクトルへ逆変換する．

$\boldsymbol{A}$ から $\boldsymbol{T}$ への変換は，ハウスホルダー法及び対角ユニタリ変換により，(4.1)によって行われる．

$$\boldsymbol{T} = \boldsymbol{V}^{*}\left(\boldsymbol{P}_{n-2}^{*}\ldots\boldsymbol{P}_{2}^{*}\boldsymbol{P}_{1}^{*}\boldsymbol{A}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\right)\boldsymbol{V} \tag{4.1}$$

ただし $\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{u}_k\,\boldsymbol{u}_k^{*}/h_k$ であり，$\boldsymbol{V}$ は対角ユニタリ行列である（詳細はサブルーチン TRIDH 参照のこと）．

$\boldsymbol{T}$ の固有値及び固有ベクトルを，それぞれλ及び $\boldsymbol{y}$ とすると，

$$\boldsymbol{T}\boldsymbol{y} = \lambda\boldsymbol{y} \tag{4.2}$$

が成り立つ．(4.1)を(4.2)に代入すると，

$$\boldsymbol{V}^{*}\boldsymbol{P}_{n-2}^{*}\ldots\boldsymbol{P}_{2}^{*}\boldsymbol{P}_{1}^{*}\boldsymbol{A}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{V}\boldsymbol{y} = \lambda\boldsymbol{y} \tag{4.3}$$

となる．(4.3)の両辺に左側から $\boldsymbol{P}_1\boldsymbol{P}_{2\ldots}\boldsymbol{P}_{n\text{-}2}\,V$ を左側から掛けると，

$$\boldsymbol{A}\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{V}\boldsymbol{y} = \lambda\boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{V}\boldsymbol{y} \tag{4.4}$$

となる．したがって， $\boldsymbol{A}$ の固有ベクトル $\boldsymbol{x}$ は

$$\boldsymbol{x} = \boldsymbol{P}_{1}\boldsymbol{P}_{2}\ldots\boldsymbol{P}_{n-2}\boldsymbol{V}\boldsymbol{y} \tag{4.5}$$

によって求められる．

(4.5)の計算は(4.6)～(4.7)の手順で行う．

$$z = \boldsymbol{V}\boldsymbol{y} \tag{4.6}$$

$$z_k = z_{k+1} - \boldsymbol{u}_k\boldsymbol{u}_k^{*}z_{k+1}/h_k \tag{4.7}$$

$$k = n-2,\ldots,2,1$$

ただし，$\boldsymbol{z}=\boldsymbol{z}_{n\text{-}1}$，$\boldsymbol{x}=\boldsymbol{z}_1$ である．

このようにして求められた固有ベクトル $\boldsymbol{x}$ (= $\boldsymbol{z}_1$) は，(4.5)により $\|\boldsymbol{y}\|_2 = 1$ ならば $\|\boldsymbol{x}\|_2 = 1$ である．

なお，詳細については，参考文献[17]を参照すること．

B21-25-0302 TRIDH, DTRIDH

エルミート行列の実対称 3 重対角行列への変換 （ハウスホルダー法，対角ユニタリ変換）
CALL TRIDH (A, K, N, D, SD, PV, ICON)

(1) 機能

n 次のエルミート行列 ***A*** をハウスホルダー法により，エルミート 3 重対角行列 ***H*** に変換し，

$$\boldsymbol{H} = \boldsymbol{P}^{*}\boldsymbol{A}\boldsymbol{P} \qquad (1.1)$$

更に対角ユニタリ変換により，実対称 3 重対角行列 ***T*** に変換する．

$$\boldsymbol{T} = \boldsymbol{V}^{*}\boldsymbol{H}\boldsymbol{V} \qquad (1.2)$$

ただし，***P*** 及び ***V*** は変換行列である．
n≥1 であること．

(2) パラメタ

A …… **入力**．エルミート行列 ***A***.
出力．変換行列 ***P***（図 TRIDH-1 参照）．
エルミート行列用圧縮モード．
（使用上の注意②参照）．
大きさ A (K, N)なる 2 次元配列.

K …… **入力**．配列 A の整合寸法 (≥ *n*).

N …… **入力**．エルミート行列の次数 *n*.

D …… **出力**．実対称 3 重対角行列 ***T*** の主対角要素（図 TRIDH-2 参照）．
大きさ *n* の 1 次元配列.

SD …… **出力**．実対称 3 重対角行列 ***T*** の副対角要素（図 TRIDH-2 参照）．
大きさ *n* の 1 次元配列.
SD (1) は 0 とし， SD (2) ,..., SD (*n*) に格納する．

PV …… **出力**．変換行列 ***V***．（図 TRIDH-3 参照）．
大きさ 2*n* の 1 次元配列.

ICON …… **出力**．コンディションコード.
表 TRIDH-1 参照.

表 TRIDH-l コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N = 1 であった.	変換しない.
30000	K < N 又は N < 1 であった.	処理を打ち切る.

×	×	$\mathrm{Im}(a_{31}^{(3)})$	$\mathrm{Im}(a_{41}^{(2)})$	$\mathrm{Im}(a_{51}^{(1)})$
×	×	$\mathrm{Im}(a_{32}^{(2)}) \bullet (1+\sigma_3/\tau_3)$	$\mathrm{Im}(a_{42}^{(2)})$	$\mathrm{Im}(a_{52}^{(1)})$
$\mathrm{Re}(a_{31}^{(3)})$	$\mathrm{Re}(a_{32}^{(3)}) \bullet (1+\sigma_3/\tau_3)$	$h_3^{\frac{1}{2}}$	$\mathrm{Im}(a_{43}^{(2)}) \bullet (1+\sigma_2/\tau_2)$	$\mathrm{Im}(a_{53}^{(1)})$
$\mathrm{Re}(a_{41}^{(2)})$	$\mathrm{Re}(a_{42}^{(2)})$	$\mathrm{Re}(a_{43}^{(2)}) \bullet (1+\sigma_2/\tau_2)$	$h_2^{\frac{1}{2}}$	$\mathrm{Im}(a_{54}^{(1)}) \bullet (1+\sigma_1/\tau_1)$
$\mathrm{Re}(a_{51}^{(1)})$	$\mathrm{Re}(a_{52}^{(1)})$	$\mathrm{Re}(a_{53}^{(1)})$	$\mathrm{Re}(a_{54}^{(1)}) \bullet (1+\sigma_1/\tau_1)$	$h_1^{\frac{1}{2}}$

［注意］ × は作業領域， *n*=5 の場合.
h_k は (4.11) 又は (4.13) 参照のこと.

図 TRIDH-l 変換後の A の各要素

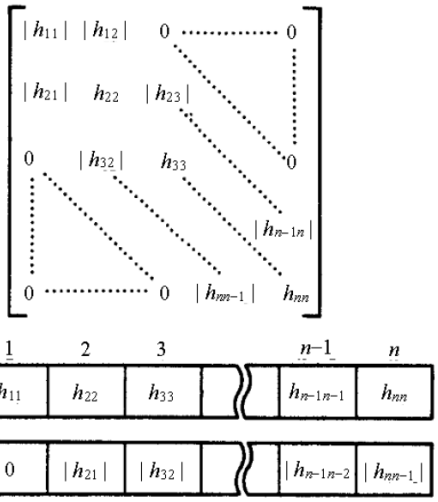


図 TRIDH-2 実対称 3 重対角行列 ***T*** と配列 D, SD の対応

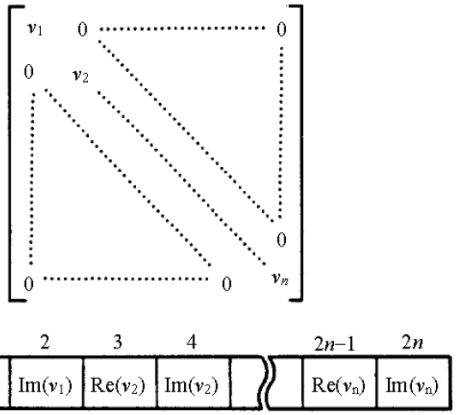


図 TRIDH-3 変換行列 V と 配列 PV の対応

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数... ABS, SQRT

b. 注意

① 本サブルーチンはエルミート行列用であり，一般複素行列については適用することができない.

② 出力された配列 A と PV はエルミート行列 $\boldsymbol{A}$ の固有ベクトルを求めるために必要となる.
この A と PV は，エルミート行列の固有ベクトルを求めるサブルーチン TRBKH の P と PV にそれぞれ対応している.

③ 固有値の精度は 3 重対角化を行った時点で，ある程度決定される.
そのため，できるだけ精度よく 3 重対角行列を求めることが必要である．そのための配慮は，本サブルーチンにおいてなされている.
しかし，固有値に非常に大きいものと，小さいものとが混在しているときは，小さい方の固有値は大きい方に比べて変換の影響を受けやすい.
したがって，そのような場合は小さい固有値を精度よく求めることは難しいことがある.

c. 使用例

n 次のエルミート行列を実対称 3 重対角行列に変換した後，サブルーチン TRQL により，固有値を計算する.
$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(100,100),D(100),SD(100),
     *          V(200),E(100)
   10 CONTINUE
      READ(5,500) N,M
      IF(N.EQ.0) STOP
      READ(5,510) ((A(I,J),I=1,N),J=1,N)
      WRITE(6,600) N,M
      DO 20 I=1,N
   20 WRITE(6,610) (I,J,A(I,J),J=1,N)
      CALL TRIDH(A,100,N,D,SD,V,ICON)
      WRITE(6,620)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,640)
      WRITE(6,650) (I,D(I),SD(1),I=1,N)
      CALL TRQL(D,SD,N,E,M,ICON)
      WRITE(6,630) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,660)
      WRITE(6,670) (I,E(I),I=1,M)
      GO TO 10
  500 FORMAT(2I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1'///,40X,'**ORIGINAL ',
     * 'MATRIX**',5X,'N=',I3,5X,'M=',I3//)
  610 FORMAT(4(5X,'A(',I3,',',I3,')=',
     * E14.7))
  620 FORMAT('0'//,11X,'**TRIDIAGONAL ',
     * 'MATRIX**')
  630 FORMAT('0',20X,'ICON=',I5)
  640 FORMAT('0',//1X,3('NO.',5X,
     * 'DIAGONAL',7X,'SUBDIAGONAL',2X),//)
  650 FORMAT(1X,3(I3,2E17.7))
  660 FORMAT('1'///,5X,'**EIGENVALUES**'/)
  670 FORMAT(1X,4('E(',I3,')=',E17.7))
      END
```

(4) 手法概要

n 次のエルミート行列 $\boldsymbol{A}$ の実対称 3 重対角行列 $\boldsymbol{T}$ への変換 (4.1)及び (4.2)の 2 段階の変更により行われる.

① n 次のエルミート行列 $\boldsymbol{A}$ をハウスホルダー法により，n 次のエルミート 3 重対角行列 $\boldsymbol{H}$ に変換する．これは次のような n-2 回のユニタリ相似変換によって行われる． $\boldsymbol{A}_1 = \boldsymbol{A}$ として，

$$\boldsymbol{A}_{k+1} = \boldsymbol{P}_k^* \boldsymbol{A}_k \boldsymbol{P}_k \quad k = 1,2,...,\mathrm{n}-2 \tag{4.1}$$

これは実対称行列の場合の直交相似変換の式における $\boldsymbol{P}_k^T$ を $\boldsymbol{P}_k^*$ に置きかえたものである（実対称行列のハウスホルダー法については TRID1 参照）．この変換の結果 $\boldsymbol{A}_{n-1}$ はエルミート 3 重対角行列 $\boldsymbol{H}$ となる.

② (4.1) で求めたエルミート 3 重対角行列 $\boldsymbol{H}$ を，対角ユニタリ相似変換(4.2)によって， n 次の実対称 3 重対角行列 $\boldsymbol{T}$ に変換する.

$$\boldsymbol{T} = \boldsymbol{V}^* \boldsymbol{H} \boldsymbol{V} \text{ ただし，} \quad \boldsymbol{H} = \boldsymbol{A}_{n-1} \tag{4.2}$$

$$\boldsymbol{V} = \mathrm{diag}(v_j) \tag{4.3}$$

いま，$\boldsymbol{H} = (h_{ij})$ として v_j を(4.4)のように表すと，

$$v_1 = 1$$
$$v_j = h_{jj-1} v_{j-1} / \left| h_{jj-1} \right| \text{ ，} j = 2,...,n \tag{4.4}$$

(4.2)の変換によって，$\boldsymbol{T}$ の各要素 t_{ij} は(4.5)，(4.6)のように実数として与えられる.

$$t_{jj} = v_j^* h_{jj} v_j = h \tag{4.5}$$

$$\begin{aligned} t_{jj-1} &= v_j^* h_{jj-1} v_{j-1} \\ &= \left(h_{jj-1}^* v_{j-1}^* / \left| h_{jj-1} \right| \right) h_{jj-1} v_{j-1} \\ &= \left| h_{jj-1} \right| \end{aligned} \tag{4.6}$$

すなわち，$\boldsymbol{H}$ の副対角要素を求める時点で $h_{jj\text{-}1}$ の絶対値 $h_{jj\text{-}1}$ の絶対値ととれば，それはそのまま $\boldsymbol{T}$ の副対角要素となる.

①，②の変更が終了した後，$\boldsymbol{T}$ は図 TRIDH-2 のようになる.

次に (4.1), (4.2)の変換の手順を示す.

第 k 回目の変換は $\boldsymbol{A}_k = (a_i^{(k)})$ として，

$$\sigma_k = \left| a_{l1}^{(k)} \right|^2 + \left| a_{l2}^{(k)} \right|^2 + ... + \left| a_{ll-1}^{(k)} \right|^2 \tag{4.7}$$

ただし， $l = n - k + 1$

$$t_{ll-1} = \sigma_k^{\frac{1}{2}} \tag{4.8}$$

ここで， $\left|a_{ll-1}^{(k)}\right| \neq 0$ の場合は，

$$\tau_k = \left(\left|a_{ll-1}^{(k)}\right|^2 \sigma_k\right)^{\frac{1}{2}} \tag{4.9}$$

$$\boldsymbol{u}_k^* = \left(a_{l1}^{(k)*},...,a_{ll-2}^{(k)*},a_{ll-1}^{(k)*}\left(1+\sigma_k/\tau_k\right),0,...,0\right) \tag{4.10}$$

$$h_k = \sigma_k + \tau_k \tag{4.11}$$

また $\left|a_{ll-1}^{(k)}\right| = 0$ の場合は，(4.10), (4.11)を，

$$\boldsymbol{u}_k^* = \left(a_{l1}^{(k)*},...,a_{ll-2}^{(k)*}\sigma_k^{\frac{1}{2}},0,...,0\right) \tag{4.12}$$

$$h_k = \sigma_k \tag{4.13}$$

として変換行列 $\boldsymbol{P}_k$ を(4.14)で構成する．

$$\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^* / h_k \tag{4.14}$$

$$\boldsymbol{A}_{k+1} = \boldsymbol{P}_k^* \boldsymbol{A}_k \boldsymbol{P}_k \tag{4.15}$$

$$\boldsymbol{t}_{ll} = \boldsymbol{a}_{ll}^{(k-1)} \tag{4.16}$$

(4.15)によって，エルミート行列 $\boldsymbol{A}$ の要素 $a_{l1}^{(k)},.....,a_{ll-2}^{(k)}$ を消去する．

実際の変換に当っては次のような考慮を払っている．

(a) (4.7) ～ (4.14) の計算中に発生するアンダフロー，オーバーフローを防ぐために (4.7) の右辺の各要素は，

$$\sum_{j=1}^{l-1}\left(\left|\mathrm{Re}\left(a_{lj}^{(k)}\right)\right| + \left|\mathrm{Im}\left(a_{lj}^{(k)}\right)\right|\right) \tag{4.17}$$

でスケーリングしたものを用いる．

(b) $\boldsymbol{P}_k$ を求めて(4.15)の変換を行う代わりに，次のように展開して $\boldsymbol{A}_{k+1}$ を求める．

$$\boldsymbol{p}_k = \boldsymbol{A}_k \boldsymbol{u}_k / \boldsymbol{h}_k,\ K_k = \boldsymbol{u}_k^* \boldsymbol{p}_k / 2\boldsymbol{h}_k \tag{4.18}$$

$$\boldsymbol{q}_k = \boldsymbol{p}_k - K_k u_k \tag{4.19}$$

$$\begin{aligned}\boldsymbol{A}_{k+1} &= \left(\boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^* / h_k\right)\boldsymbol{A}_k\left(\boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^* / h_k\right) \\ &= \boldsymbol{A}_k - \boldsymbol{u}_k \boldsymbol{q}_k^* - \boldsymbol{q}_k \boldsymbol{u}_k^*\end{aligned} \tag{4.20}$$

(c) また変換行列 $\boldsymbol{P}_k$ 及び $\boldsymbol{V}$ はエルミート行列の固有ベクトルを求めるときに必要になる．
そのために $\boldsymbol{P}_k$ として，(4.10)又は(4.12)の $\boldsymbol{u}_k$，及び (4.11)又は(4.13)の h_k をその平方根をとって図 TRIDH-1 の形に格納する．
対角ユニタリ行列 $\boldsymbol{V}$ はその対角要素 v_j を，1 次元配列 PV に図 TRIDH-3 のように格納する．

入力パラメタ N が 1 のときは，$\boldsymbol{A}$ の対角要素をそのまま配列 D に入れる．

なお，詳細については，参考文献[13] pp.212-226 及び [17] を参照のこと．

B21-21-0302 TRID1, DTRID1

実対称行列の実対称 3 重対角行列への変換 （ハウスホルダー法）
CALL TRID1 (A, N, D, SD, ICON)

(1) 機能

n 次の実対称行列 $\boldsymbol{A}$ を，ハウスホルダー法（直交相似変換）により，実対称 3 重対角行列 $\boldsymbol{T}$ へ変換する．

$$\boldsymbol{T} = \boldsymbol{P}^T \boldsymbol{A}\boldsymbol{P} \qquad (1.1)$$

ただし $\boldsymbol{P}$ は変換行列であり， $n \geq 1$ であること．

(2) パラメタ

A…… **入力**．実対称行列 $\boldsymbol{A}$.
出力．変換行列 $\boldsymbol{P}$.
対称行列用圧縮モード．
大きさ $n(n+1)/2$ の 1 次元配列.

N…… **入力**．実対称行列 $\boldsymbol{A}$ の次数 n.

D…… **出力**． 3 重対角行列 $\boldsymbol{T}$ の主対角要素.
大きさ n の 1 次元配列.

SD…… **出力**．3 重対角行列 $\boldsymbol{T}$ の副対角要素.
大きさ n の 1 次元配列.
ただし SD(2)～ SD(N) を使用し，SD (1)には零がセットされる．

ICON .. … **出力**．コンディションコード.
表 TRID1-1 参照.

表 TRID1-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1 又は N=2 であった.	変換しない.
30000	N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II ... AMACH, MGSSL
② FORTRAN 基本関数 ... ABS, DSQRT, SIGN

b. 注意
① 出力された配列 A は，実対称行列 $\boldsymbol{A}$ の固有ベクトルを求めるために必要となる．
② 固有値の精度は，3 重対角行列化によってある程度決定される．そのため，できるだけ精度よく，3 重対角行列を求めることが必要であり，本サブルーチンではその考慮がはらわれている．しかし，固有値に非常に大きいものと小さいものがあるとき，小さい方の固有値は，大きい固有値に比べて変換の影響を受けやすい．したがって小さい固有値を精度よく求めることがむずかしい場合がある．

c. 使用例
n 次実対称行列を 3 重対角行列に変換した後，サブルーチン TRQL により固有値を計算する．
$n \leq 100$ の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),D(100),
     *          SD(100),E(100)
   10 READ(5,500) N
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N
      NE=0
      DO 20 I=1,N
      NI=NE+1
      NE=NE+I
   20 WRITE(6,610) I,(A(J),J=NI,NE)
      CALL TRID1(A,N,D,SD,ICON)
      WRITE(6,620)
      WRITE(6,630) ICON
      IF(ICON.EQ.30000) GO TO 10
      WRITE(6,640) (I,D(I),SD(I),I=1,N)
      CALL TRQL(D,SD,N,E,M,ICON)
      WRITE(6,650)
      WRITE(6,630) ICON
      IF(ICON.EQ.20000) GO TO 10
      WRITE(6,660) (I,E(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',10X,'** ORIGINAL MATRIX'/
     *       11X,'** ORDER =',I5)
  610 FORMAT('0',7X,I3,5E20.7/
     * (11X,5E20.7))
  620 FORMAT('0'/11X,'** TRIDIAGONAL ',
     * 'MATRIX')
  630 FORMAT(11X,'** CONDITION CODE =',
     * I5/)
  640 FORMAT(5X,I5,2E16.7)
  650 FORMAT('0'/11X,'** EIGENVALUES')
  660 FORMAT(5X,'E(',I3,')=',E15.7)
      END
```

(4) 手法概要

n 次の実対称行列 $\boldsymbol{A}$ の 3 重対角行列への変換は，次式で示されるような $n-2$ 回の直交相似変換により行われる．

$$A_{k+1} = P_k^T A_k P_k, \quad k = 1,2,.....,n-2 \qquad (4.1)$$

ただし $\boldsymbol{A}_1 = \boldsymbol{A}$, $\boldsymbol{P}_k$ は変換行列（かつ直交行列）である．

変換の終了した時点で $\boldsymbol{A}_{n-1}$ は，実対称 3 重対角行列となる．

第 k 回目の変換は，次の手順により行う．

$\boldsymbol{A}_k = \left(a_{ij}^{(k)}\right)$ として，

$$\sigma_k = \left(a_{l1}^{(k)}\right)^2 + \left(a_{l2}^{(k)}\right)^2 + ...+ \left(a_{ll-1}^{(k)}\right)^2 \qquad (4.2)$$

ただし，$l = n - k + 1$

$$u_k^T = \left(a_{l1}^{(k)},...,a_{ll-2}^{(k)},a_{ll-1}^{(k)} \pm \sigma_k^{\frac{1}{2}},0,...,0\right) \quad (4.3)$$

$$h_k = \sigma_k \pm a_{ll-1}^{(k)}\sigma_k^{\frac{1}{2}} \quad (4.4)$$

$$\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^T / h_k \quad (4.5)$$

(4.5)の $\boldsymbol{P}_k$を(4.l)へ適用することによって $\boldsymbol{A}_k$ の $a_{l1}^{(k)}$ ～ $a_{ll-2}^{(k)}$ を消去する．

実際の変換にあたっては，次のような考慮をはらっている．

① (4.2)～(4.4)の計算におけるアンダフロー,オーバフローを防くために，(4.2)の右辺の各要素は，$\sum_{j=1}^{l-1}\left|a_{ij}^{(k)}\right|$ でスケーリングしたものを用いる．

② (4.3)の $\boldsymbol{u}_k^T$ を求めるとき， $a_{k+1\,k}^{(k)} \pm \sigma_k^{\frac{1}{2}}$ の計算で桁落ちが生じないようにするため， $\pm\sigma_k^{\frac{1}{2}}$ は $a_{k-1\;1k}^{(k)}$ と同符号になる方を採用する．

③ (4.1)の変換は，$\boldsymbol{P}_k$を求めて行うかわりに，(4.1)を(4.7), (4.8)のように展開して $\boldsymbol{A}_{k+1}$を求める．

$$\boldsymbol{p}_k = \boldsymbol{A}_k \boldsymbol{u}_k / \boldsymbol{h}_k, \quad K_k = \boldsymbol{u}_k^T \boldsymbol{p}_k / 2h_k \quad (4.7)$$

$$\boldsymbol{q}_k = \boldsymbol{p}_k - K_k \boldsymbol{u}_k \quad (4.8)$$

$$\begin{aligned}\boldsymbol{A}_{k+1} &= \left(\boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^T / h_k\right)\boldsymbol{A}_k\left(\boldsymbol{I} - \boldsymbol{u}_k \boldsymbol{u}_k^T / h_k\right) \\ &= \boldsymbol{A}_k - \boldsymbol{u}_k \boldsymbol{q}_k^T - \boldsymbol{q}_k \boldsymbol{u}_k^T \end{aligned} \quad (4.9)$$

また変換行列 $\boldsymbol{P}_k$ は，実対称行列 $\boldsymbol{A}$ の固有ベクトルを求めるときにも必要となる．そのため (4.3)の $\boldsymbol{u}_k$ の要素及び (4.4)の h_k を，図 TRID1-l の形に格納する．

$$\begin{bmatrix} \times & & & & \\ \times & \times & & & \\ a_{31}^{(3)} & a_{32}^{(3)} \pm \sigma_3^{\frac{1}{2}} & h_3 & & \\ a_{41}^{(2)} & a_{42}^{(2)} & a_{43}^{(2)} \pm \sigma_2^{\frac{1}{2}} & h_2 & \\ a_{51}^{(1)} & a_{52}^{(1)} & a_{53}^{(1)} & a_{54}^{(1)} \pm \sigma_1^{\frac{1}{2}} & h_1 \end{bmatrix}$$

[注意] ×は作業領域，n=5 の場合

図 TRID1-1 ハウスホルダー変換後の A
（ただし，A は圧縮モードである）

n= 2 又は n=1 のときは，主対角要素及び副対角要素をそのまま配列 D 及び配列 SD へ入れる．

なお，詳細については，参考文献 [13]の pp.212-226 を参照すること．

B21-21-0402 TRQL, DTRQL

実対称 3 重対角行列の固有値（QL 法）
CALL TRQL (D, SD, N, E, M, ICON)

(1) 機能

n 次の実対称 3 重対角行列 ***T*** の固有値を QL 法により求める.

n ≥ 1 になること.

(2) パラメタ

D…… **入力**. 3 重対角行列 ***T*** の主対角要素.
大きさ *n* の 1 次元配列.
演算後内容は保存されない.

SD…… **入力**. 3 重対角行列 ***T*** の副対角要素.
SD(2)～SD (N)に格納しておくこと.
大きさ *n* の 1 次元配列.
演算後内容は保存されない.

N…… **入力**. 3 重対角行列 の次数 *n*.

E …… **出力**. 固有値.
大きさ *n* の 1 次元配列.

M …… **出力**. 求まった固有値の数.

ICON…… **出力**. コンディションコード.
表 TRQL-1 参照.

表 TRQL-1 コンディションコード

コード	意味	処理内容
0	エラーなし.	
10000	N=1 であった.	E(1)=D(1) とする.
15000	固有値のすべてを求めつくすことはできなかった.	M に求まった固有値の数をセットする. 1 ≤ M < N.
20000	固有値が一つも求まらなかった.	M=0 とする.
30000	N < 1 であった.	処理を打ち切る.

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数 ... ABS, SQRT, SIGN

b. 注意

① パラメタ M は，ICON =0 のとき *n* が，また ICON = 15000 のとき，求まった固有値の数がセットされている.

② 本ルーチンは， QL 法を用いているため，3 重対角行列の左上方の要素の絶対値が，右下方の要素の絶対値に比べ小さいような行列のとき，最適な方法である.

③ 本サブルーチンは，3 重対角行列用である．実対称行列の固有値を求める場合は，まずサブルーチン TRID1 によって 3 重対角行列に変換した後，本ルーチンを呼出すこと.

④ 3 重対角行列のおよそ *n*/4 個以下の固有値を求める場合は，サブルーチン BSCT1 を用いた方が処理が速い.

⑤ 3 重対角行列の固有ベクトルまで求める場合は，サブルーチン TEIG 1 を用いること.

c. 使用例

n 次の実対称行列をサブルーチン TRID1 により 3 重対角行列に変換し，本サブルーチンにより固有値を求める.

n≤100 の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),D(100),
     *          SD(100),E(100)
   10 CONTINUE
      READ(5,500) N
      IF(N.EQ.0) STOP
      NN=N*(N+1)/2
      READ(5,510) (A(I),I=1,NN)
      WRITE(6,600) N
      LST=0
      DO 20 I=1,N
      INT=LST+1
      LST=LST+I
      WRITE(6,610) I,(A(J),J=INT,LST)
   20 CONTINUE
      CALL TRID1(A,N,D,SD,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) GO TO 10
      CALL TRQL(D,SD,N,E,M,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      WRITE(6,630) (I,E(I),I=1,M)
      GO TO 10
  500 FORMAT(I5)
  510 FORMAT(5E15.7)
  600 FORMAT('1',5X,'ORIGINAL  MATRIX',
     * 10X,'N=',I3)
  610 FORMAT('0',7X,I3,5E20.7/(11X,
     * 5E20.7))
  620 FORMAT('0',20X,'ICON=',I5)
  630 FORMAT('0',5X,'EIGENVALUE'//
     *       (8X,4('E(',I3,')=',E15.7,
     * 3X)))
      END
```

(4) 手法概要

n 次の実対称 3 重対角行列 ***T*** の固有値を QL 法により求める．QL 法の説明にはいる前に，3 重対角行列及びその固有値の記述法を定義しておく.

以後，3 重対角行列を ***T*** で表し，その全固有値を，$(\lambda_1, \lambda_2, ..., \lambda_n)$で表す．また，固有値は，すべて相異なるものとする.

3 重対角行列 ***T*** の主対角要素を $d_1, d_2, ..., d_n$ とし，副対角要素 を $e_1, e_2, ..., e_n$ とする.

すなわち ***T*** は,

$$T=\begin{bmatrix} d_1 & e_1 & & & \\ e_1 & d_2 & e_2 & & 0 \\ & e_2 & \ddots & \ddots & \\ & & \ddots & \ddots & e_{n-1} \\ & 0 & & e_{n-1} & d_n \end{bmatrix} \qquad (4.1)$$

であるとする.

QL 法は，次の三つの定理を利用して固有値を求める方法である.

① 3 重対角行列 $\boldsymbol{T}$ が実対称行列であれば,

$$\boldsymbol{D} = \boldsymbol{X}^T \boldsymbol{T} \boldsymbol{X} \tag{4.2}$$

とする直交行列 $\boldsymbol{X}$ が存在する．ここに，$\boldsymbol{D}$ は対角行列である．このとき，$\boldsymbol{D}$ の対角要素は，$\boldsymbol{T}$ の固有値である．

② $\boldsymbol{T}$ が正則であれば，直交行列 $\boldsymbol{Q}$ と対角要素がすべて正なる下三角行列 $\boldsymbol{L}$ との積,

$$\boldsymbol{T} = \boldsymbol{Q}\boldsymbol{L} \tag{4.3}$$

に一意に分解できる．

③ 与えられた実対称 3 重対角行列 $\boldsymbol{T}$ を $\boldsymbol{T}_1$ とし，第 s 回目の反復後に得られる行列を $\boldsymbol{T}_{s+1}$ とする．
ここで,

$$\boldsymbol{T}_s = \boldsymbol{Q}_s \boldsymbol{L}_s \quad s = 1,2,3,... \tag{4.4}$$

$$\boldsymbol{T}_{s+1} = \boldsymbol{Q}_s^T \boldsymbol{T}_s \boldsymbol{Q}_s \tag{4.5}$$

なる反復計算を行うことを考える．
(4.5)の直交相似変換により，$\boldsymbol{T}_{s+1}$ は実対称 3 重対角行列となる．しかも，この $\boldsymbol{T}_{s+1}$ は，その全体が $\boldsymbol{Q}_s$ の第 n 列だけによって決定され，$s \to \infty$のとき，対角行列に収束する．

QL 法は，以上の三つの定理を利用して，(4.4)及び(4.5)を反復適用することにより，固有値を求める方法である．

QL 法の実際の計算では，(4.4)を解いて $\boldsymbol{Q}_s$ を求め，次いで (4.5)により $\boldsymbol{T}_{s+1}$ を求めるという方法は用いず，(4.4)及び(4.5)の計算を同時に行う．

(4.4)より,

$$\boldsymbol{Q}_s^T \boldsymbol{T}_s = \boldsymbol{L}_s \tag{4.6}$$

であるから，$\boldsymbol{Q}_s^T$は $\boldsymbol{T}_s$に左から掛けて下三角行列に変換する行列である．したがって，いま $\boldsymbol{P}_i^{(s)T}$ を $\boldsymbol{T}_s$ の (i, i+l)要素を消去する変換行列とすれば,

$$\boldsymbol{Q}_s^T = \boldsymbol{P}_1^{(s)T} \boldsymbol{P}_2^{(s)T} \cdots\cdots\cdots \boldsymbol{P}_{n-1}^{(s)T} \tag{4.7}$$

と表せるから，(4.5)は,

$$\boldsymbol{T}_{s+1} = \boldsymbol{P}_1^{(s)T} \boldsymbol{P}_2^{(s)T} \cdots\cdots \boldsymbol{P}_{n-1}^{(s)T} \boldsymbol{T}_s \boldsymbol{P}_{n-1}^{(s)} \cdots\cdots \boldsymbol{P}_2^{(s)} \boldsymbol{P}_1^{(s)} \tag{4.8}$$

と表せる．(4.8)を内側から順次計算してゆけば，(4.4)及び(4.5)を同時に実行できる．

しかし通常，QL 法では収束の加速をするため，$\boldsymbol{T}_s$ のかわりに適当に定めた定数 k により原点移動を行った$(\boldsymbol{T}_s - k\boldsymbol{I})$を用いる．

$\boldsymbol{T}_s$ のかわりに$(\boldsymbol{T}_s - k\boldsymbol{I})$を用いれば，(4.4), (4.5)はそれぞれ

$$\left(\boldsymbol{T}_s - k\boldsymbol{I}\right) = \boldsymbol{Q}_s \boldsymbol{L}_s \tag{4.9}$$

$$\left(\boldsymbol{T}_{s+1} - k\boldsymbol{I}\right) = \boldsymbol{Q}_s^T \left(\boldsymbol{T}_s - k\boldsymbol{I}\right) \boldsymbol{Q}_s \tag{4.10}$$

となる．この(4.10)は更に

$$\boldsymbol{T}_{s+1} = \boldsymbol{Q}_s^T \left(\boldsymbol{T}_s - k\boldsymbol{I}\right) \boldsymbol{Q}_s + k\boldsymbol{I} \tag{4.11}$$

と変形でき，したがって,

$$\boldsymbol{T}_{s+1} = \boldsymbol{Q}_s^T \boldsymbol{T}_s \boldsymbol{Q}_s \tag{4.12}$$

とすることができる．（(4.5)の$\boldsymbol{Q}_s$とは違うことに注意）．したがって，$\boldsymbol{Q}_s$を(4.9)により,

$$\boldsymbol{Q}_s^T \left(\boldsymbol{T}_s - k\boldsymbol{I}\right) = \boldsymbol{L}_s \tag{4.13}$$

とするように作れば，原点移動を行った場合でも $\boldsymbol{T}_{s+1}$ は(4.12)より直接に求めることができる．

原点移動量 k は，次のように定める．

今，(4.1)において d_i 上に求まる固有値をλ_i とすればλ_i に関する(4.12)の収束速度は,

$$\left|\lambda_i - k\right| / \left|\lambda_{i+1} - k\right| \tag{4.14}$$

の比率によってきまる．
このとき，k として,

$$\boldsymbol{B}_s = \begin{bmatrix} \boldsymbol{d}_i^{(s)} & \boldsymbol{e}_i^{(s)} \\ \boldsymbol{e}_i^{(s)} & d_{i+1}^{(s)} \end{bmatrix} \tag{4.15}$$

なる2 × 2小行列$\boldsymbol{B}_s$の固有値をヤコビ法により求め，$d_i^{(s)}$ に対応する方を与えることにする．

こうすると，まず $e_i^{(s)}$ が急速に 0 に収束する．しかも，(4.12)の計算を 1 回行った後，k を計算しなおすことにすれば，更によいλ_i の近似値が得られるから，毎回 k を計算し直して，(4.12)を実行することにより，更に収束を加速できる．

このようにして計算した s 回目の k を k_s とする．

(4.12)の計算は，実際には(4.8)の形で計算する．そこで，$\boldsymbol{P}_i^{(s)T}$ (i = n-1, ..., 2, 1)の計算法を述べる．

まず，$\boldsymbol{P}_{n-1}^{(s)T}$ について考えてみると，$\boldsymbol{P}_{n-1}^{(s)T}$ は $(\boldsymbol{T}_s - k_s\boldsymbol{I})$ に左から掛けて(n-1, n)要素を零に変換する行列であるから，$\boldsymbol{P}_{n-1}^{(s)T}$ として,

$$\boldsymbol{P}_{n-1}^{(s)T} = \begin{bmatrix} 1 & & & 0 \\ & \cdots & & \\ & & 1 & \\ 0 & & & \begin{matrix} C_{n-1} & -S_{n-1} \\ S_{n-1} & C_{n-1} \end{matrix} \end{bmatrix}$$

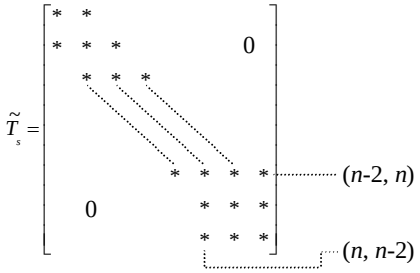
$$C_{n-1} = \left(d_n - k_s\right) / \left\{\left(d_n - k_s\right)^2 + e_{n-1}^2\right\}^{\frac{1}{2}}$$

$$S_{n-1} = e_{n-1} / \left\{\left(d_n - k_s\right)^2 + e_{n-1}^2\right\}^{\frac{1}{2}}$$

なる行列を考え，これを用いて

$$\tilde{\boldsymbol{T}}_s = \boldsymbol{P}_{n-1}^{(s)\mathrm{T}} \boldsymbol{T}_s \boldsymbol{P}_{n-1}^{(k)} \tag{4.16}$$

を計算すれば，$\tilde{\boldsymbol{T}}_s$ は次のような形になる



これは，3 重対角行列に，$(n-2, n)$及び $(n, n-2)$ の要素を付加した対称行列となっている．

この $\tilde{\boldsymbol{T}}_s$ をギブンス法により 3 重対角行列に変換する行列を，

$$\tilde{\boldsymbol{Q}}_s = \tilde{\boldsymbol{P}}_{n-2}^{(s)} \cdots\cdots \tilde{\boldsymbol{P}}_2^{(s)} \tilde{\boldsymbol{P}}_1^{(s)} \tag{4.17}$$

とすれば，$\boldsymbol{T}_s$ から $\boldsymbol{T}_{s+1}$ への変換については QL 分解の一意性により，

$$\boldsymbol{T}_{s+1} = \tilde{\boldsymbol{Q}}_s^T \tilde{\boldsymbol{T}}_s \tilde{\boldsymbol{Q}}_s \tag{4.18}$$

となる．したがって $\boldsymbol{T}_{s+1}$ は(4.16), (4.17), (4.18)より，

$$\boldsymbol{T}_{s+1} = \tilde{\boldsymbol{P}}_1^{(s)T} \tilde{\boldsymbol{P}}_2^{(s)T} \cdots\cdots \tilde{\boldsymbol{P}}_{n-2}^{(s)T} \left(\boldsymbol{P}_{n-1}^{(s)T} \boldsymbol{T}_s \boldsymbol{P}_{n-1}^{(s)} \right) \tilde{\boldsymbol{P}}_{n-2}^{(s)} \cdots\cdots \cdots\cdots \tilde{\boldsymbol{P}}_2^{(s)} \tilde{\boldsymbol{P}}_1^{(s)} \tag{4.19}$$

として計算することができる．

以下に本サブルーチンの計算手順を示す．

i=1, 2, ..., n–1 に対して①～⑥を行う．

① 次の式により原点移動量 $\boldsymbol{K}_s$ を計算する．

$$f = \left(d_{i+1}^{(s)} - d_i^{(s)} \right) / 2e_i^{(s)}$$
$$k_s = d_i^{(s)} - e_i^{(s)} / \left(f \pm \sqrt{f^2+1} \right) \tag{4.24}$$

（$\pm\sqrt{f^2+1}$ は f の符号に合わせる）

② C_{n-1} 及び S_{n-1} を計算して $\boldsymbol{P}_{n-1}^{(s)T}$ を定め，(4.16)より $\tilde{\boldsymbol{T}}_s$ を作る．

③ $\tilde{\boldsymbol{T}}_s$ をギブンス法により 3 重対角行列に変換する．

④ 次のいずれかの条件が満足されるまで，$s = s + 1$ として①～③をくり返す．

(a) $$\left| e_j \right| \le u \left(\left| d_j \right| + \left| d_{j+1} \right| \right) \left(j = i, \quad i+1, ..., n-1 \right) \tag{4.20}$$

ここに u は丸め誤差の単位である．

(b) 反復回数 $s \ge 30$.

⑤ $j = i$ で条件(a)が満足されたとき，固有値 λ_i は求まったとして，次の固有値を計算する．ただし，λ_{n-1} が求まれば λ_n は自動的に求まる．

⑥ 条件(b)が満足されたとき，固有値 λ_i は求まらなかったとして，ICON に 15000 をセットする．

本サブルーチンの実際の計算では，計算速度をあげるため，次のような考慮をしている．

① 行列 $\boldsymbol{T}$ が直和分解できるときは，各小行列ごとに計算できる．そこで，各小行毎に前記①～⑥の計算を行うことにより，計算量を減らす．
このため，(4.20)の判定条件を直和分解の判定条件と共用している．すなわち，$j \ne i$ で(4.20)を満足するとき小行列に分解する．

② 前記②と③の計算は，(4.21)のようにして，同一の処理を行っている．

$$\left.\begin{aligned} & e_n^{(s)} = 0.0, C_n = 1.0, S_n = 1.0 \\ & g_j = C_{j+1} d_{j+1}^{(s)} - S_{j+1} e_{j+1}^{(s)} \\ & w_j = \left(g_j^2 + e_j^{(s)2} \right)^{\frac{1}{2}} \\ & C_j = g_j / w_j, S_j = e_j^{(s)2} / w_j, x_j = 2 C_j S_j e_j^{(s)} \\ & d_{j+1}^{(s+1)} = C_j^2 d_{j+1}^{(s)} + S_j^2 d_j^{(s)} + x_j \\ & e_j^{(s+1)} = C_j S_j d_j^{(s)} - S_j^2 e_j^{(s)} \\ & \left(j = n-1, n-2, ..., i+1, i \right) \end{aligned}\right\} \tag{4.21}$$

$$d_i^{(s+1)} = d_{i+1}^{(s+1)} - 2x_i$$

なお，詳細については，参考文献[12], [13] pp. 227-248 及び [16] pp.177-206 を参照すること．

C23-11-0111 TSDM, DTSDM

実超越方程式 $f(x)=0$（マラー法）
CALL TSDM(X, FUN, ISW, EPS, ETA, M, ICON)

(1) 機能

与えられた初期値をもとに実超越方程式

$$f(x) = 0$$

の 1 根をマラー(Muller)法により求める.

(2) パラメタ

X …… **入力**. 求めようとする根の初期値.
出力. 近似根.

FUN …… **入力**. 解こうとする方程式の関数 $f(x)$を計算する関数副プログラム名.

ISW …… **入力**. 制御情報.
求めようとする根の収束判定法を指定する.
ISW = 1,2, 3 のいずれかであること.
ISW = 1 のとき,

$|f(x_i)| \le$ EPS: 収束判定法 I

を満たす x_i を根とする.
ISW = 2 のとき,

$|x_i - x_{i-1}| \le \text{ETA} \cdot |x_i|$: 収束判定法 II

を満たす x_i を根とする.
ISW = 3 のとき,
上の二つの判定法を同時に採用し少くとも一方が満たされたとき, x_i を根とする.
（使用上の注意②参照）

EPS …… **入力**. 収束判定法 I （パラメタ ISW の項参照）で用いられる収束判定値 (≥ 0.0).

ETA …… **入力**. 収束判定法 II（パラメタ ISW の項参照）で用いられる収束判定値 (≥ 0.0).

M …… **入力**. 根を求めるための反復回数の上限 (>0). （使用上の注意③参照）
出力. 実際に反復した回数.

ICON …… **出力**. コンディションコード.
表 TSDM-1 参照.

(3) 使用上の注意

a. 使用する副プログラム
① SSL II..... AFMAX, AFMIN, AMACH, MGSSL
② FORTRAN 基本関数 .. ABS, EXP, SQRT, ALOG

表 TSDM-1 コンディションコード

コード	意味	処理内容
1	求まった近似根が収束判定法 I（パラメタ ISW の項参照）を満たした.	正常終了
2	求まった近似根が収束判定法 II（パラメタ ISW の項参照）を満たした.	正常終了
10	反復の回数は無条件に m 回 (M=–m)繰り返した.	正常終了
11	反復の回数は無条件に m 回繰り返すと指定されたが (M=–m), その回数の反復以前に $f(x_j) = 0.0$ となったので反復を止めて x_i を根とした.	正常終了
12	反復の回数は無条件に m 回繰り返すと指定されたが (M=–m), その回数の反復以前に $\|x_i-x_{i-1}\| \le u \cdot \|x_i\|$ となったので反復を止めて x_i を根とした.	正常終了
10000	与えられた反復回数内で指定された収束条件を満たさなかった.	X には最後の反復値が格納される. .
20000	計算過程で, マラー法では反復を続行できない事態が起きた（手法概要参照）	処理を打ち切る.
30000	入力パラメタにエラーがあった. すなわち M>0 のとき ① ISW=1 かつ EPS < 0 又は ② ISW=2 かつ ETA<0 又は ③ ISW=3 かつ EPS<0 かつ ETA<0 のいずれかであるか, 又は M=0 又は ISW ≠ 1, 2, 3 であった .	処理を打ち切る.

b. 注意

① パラメタ FUN に対応する関数副プログラムは, 変数 x だけを引数に持つ関数副プログラムとして定義し, 本サブルーチンを呼び出す側のプログラムで, その関数名を EXTERNAL 文で宣言しなければならない. （使用例参照）

② 本サブルーチンは, ISW = 1 と与えられたときでも, 反復の過程で

$|x_i - x_{i-1}| \le u \cdot |x_i|$ （u は丸め誤差の単位）

が満たされたときは反復を止め, ICON = 2 と出力する. 同様に, ISW = 2 と与えられたときでも,

$f(x_i) = 0.0$

となったときは反復を止め, ICON = 1 と出力する.

③ 反復の回数 M を M = –m (m > 0)と与えると，反復は無条件に m 回繰り返す．ただし，m 回繰り返す間に，$f(x_i)$= 0.0 あるいは $|x_i - x_{i-1}| \le u \cdot |x_i|$ が満たされたときは，反復を止め，それぞれ ICON = 11 あるいは ICON = 12 と出力する．

c. 使用例

$f(x) = e^x - 1$ の 1 根を，初期値を 0 として求める．

```
C     **EXAMPLE**
      EXTERNAL FEXP
      X=0.0
      ISW=3
      EPS=0.0
      ETA=1.0E-6
      M=100
      CALL TSDM(X,FEXP,ISW,EPS,ETA,M,
     *          ICON)
      WRITE(6,600) ICON,M,X
      STOP
  600 FORMAT(10X,'ICON=',I5/13X,'M=',I5/
     * 13X,'X=',E15.7)
      END
      FUNCTION FEXP(X)
      FEXP=EXP(X)-1.0
      RETURN
      END
```

(4) 手法概要

本サブルーチンはマラー(Muller) 法を用いている．マラー法は，求めようとする根の三つの近似値を用いて，$f(x)$を 2 次の補間多項式 $P(x)$で近似し，$P(x)$ = 0 の根の一方を $f(x)$の次の近似値として採用することにより，次々と反復的に近似していくものである．

本アルゴリズムの特長として，次のことが挙げられる．

a. 微係数は不要である．
b. 反復の過程での関数の評価は，その反復ごとに 1 回で済む．
c. 収束の速さは 1.84 次である（ただし単根の場合）．

マラー法

$f(x)$ の求めようとする根 αの三つの近似値を x_{i-2}, x_{i-1}, x_i とする（出発値 x_1 , x_2, x_3 については後述），この 3 点を用いて $f(x)$ を 2 次のニュートン補間多項式で近似する．それを $P(x)$とすると次のようになる．

$$P(x) = f_i + f[x_i, x_{i-1}](x - x_i) + f[x_i, x_{i-1}, x_{i-2}](x - x_i)(x - x_{i-1}) \quad (4.1)$$

ただし，$f_i = f(x_i)$であり，$f[x_i, x_{i-1}]$，$f[x_i, x_{i-1}, x_{i-2}]$ は，それぞれ 1 階，2 階の差分商 (devided defference) で，

$$f[x_i, x_{i-1}] = \frac{f_i - f_{i-1}}{x_i - x_{i-1}}$$

$$f[x_i, x_{i-1}, x_{i-2}] = \frac{f[x_i, x_{i-1}] - f[x_{i-1}, x_{i-2}]}{x_i - x_{i-2}} \quad (4.2)$$

である．

ここで，2 次方程式 $P(x) = 0$ を解くと，その 2 根は次のように書ける．

$$x = x_i - \frac{2 f_i}{\omega \pm \left\{\omega^2 - 4 f_i f[x_i, x_{i-1}, x_{i-2}]\right\}^{1/2}}$$

$$\omega = f[x_i, x_{i-1}] + (x_i - x_{i-1}) f[x_i, x_{i-1}, x_{i-2}] \quad (4.3)$$

ここで，この(4.3)式のうち，第 2 項の分母の絶対値の大きい方の根を次の反復値 x_{i+1} とする．これは x_{i+1} として x_i に近い方の根を採用することを意味する．

(4.1) 式で，x^2 の項が 0 の場合，すなわち，$f[x_i, x_{i-1}, x_{i-2}]$ = 0 の場合は，(4.3)式を用いず，

$$x = x_i - \frac{f_i}{f[x_i, x_{i-1}]} = x_i - \frac{x_i - x_{i-1}}{f_i - f_{i-1}} \cdot f_i$$

を用いる．これはセカント (secant) 法にほかならない．

また (4.1) 式で，x^2, x の項が共に 0 の場合は，

$P(x)$ = 定数

となり，本アルゴリズムを用いることが不可能となる．（この場合については後述）．

アルゴリズム上の考慮

a. 出発値 x_1, x_2, x_3

x_1, x_2, x_3 は次のように決定する．

利用者は入力パラメタ X に与えた初期値 x とすると，

$x \neq 0$ のとき

$$\begin{cases} x_1 = 0.9x \\ x_2 = 1.1x \\ x_3 = x \end{cases}$$

x= 0 のとき

$$\begin{cases} x_1 = -1.0 \\ x_2 = 1.0 \\ x_3 = 0.0 \end{cases}$$

とする．

b. $f(x_{i-2}) = f(x_{i-1}) = f(x_i)$の場合の処置

この場合は，(4.1)式の x^2, x の項が共に 0 となった場合で，マラー法ではこれ以降の処理が不可能となる．本サブルーチンでは x_{i-2}, x_{i-1}, x_i に摂動を与えて，この状態を避けるよう次のような試みを行っている．

すなわち，

$x'_{i-2}=(1+p^n)x_{i-2}$
$x'_{i-1}=(1+p^n)x_{i-1}$
$x'_i=(1+p^n)x_i$
ただし，$p=-u^{-1/10}$, u は丸め誤差の単位，n は摂動を与えた回数.

として，x'_{i-2}, x'_{i-1}, x'_i についてマラー法を続行する．ここで，この摂動が 5 回以上行われた場合は本サブルーチンでは，これ以上の続行をあきらめ，ICON = 20000 として処理を打ち切る.

収束判定法

次の二つの収束判定法を用いている.

収束判定法 I

近似根 x_i が $|f(x_i)| \le$ EPS を満たしたとき，x_i を根として採用する.

収束判定法 II

近似根 x_i が $|x_i - x_{i-1}| \le \mathrm{ETA}\cdot|x_i|$を,満たしたとき，$x_i$ を根として採用する．この判定法における ETA は求めようとする根が多重根であったり近接根である場合は大きめにとる必要がある．なお，$0 \le \mathrm{ETA} < u$ であったときは，サブルーチン内部で，ETA = u と置き換えている.

なお詳細については，参考文献 [32]を参照すること.

C23-11-0101 TSD1, DTSD1

実超越方程式 $f(x)=0$（ブレント法）
CALL TSD1 (AI, BI, FUN, EPST, X, ICON)

(1) 機能

区間[a, b] で $f(x)$ が連続でかつ $f(a)\,f(b) \le 0$ となる 2 点 a, b が与えられたとき実超越方程式

$$f(x)=0$$

の根を区間[a, b] の中で求める．$f(x)$の導関数を必要とせず，$f(x)$のふるまいを計算中に調べることにより 2 分法，線型補間法，逆 2 次補間法を併用して求める．

(2) パラメタ

AI…… **入力**．区間の下限 a.

BI…… **入力**．区間の上限 b.

FUN…… **入力**．解こうとする方程式の関数 $f(x)$を計算する関数副プログラム名．
利用者は呼び出し側のプログラム中で EXTERNAL 宣言をすると共に関数副プログラムを用意すること．

EPST …… **入力**．求めようとする近似根の絶対誤差の上限(≥ 0.0)（使用上の注意参照）．

X…… **出力**．近似根．

ICON…… **出力**．コンディションコード．
表 TSDI-1 参照．

表 TSD1-l コンディションコード

コード	意味	処理内容
0	エラーなし．	
30000	$f(a)\,f(b) > 0$ 又は EPST < 0.0 であった．	処理を打ち切る．

(3) 使用上の注意

a. 使用する副プログラム

① SSL II ... AMACH, MGSSL

② FORTRAN 基本関数... ABS

b. 注意
本サブルーチンを呼び出す側のプログラム内で引数 FUNに相当する関数副プログラムに対して，EXTERNAL 宣言をすること．また，その関数副プログラムを用意すること．

① 例えば図 TSD1-l のように区間 [a, b]内に複数個の根を持つ場合は，出力される根がどの根であるかを保証しない．

② パラメタ EPST は求めようとする根に対する所要の精度を与えるものである．区間[a, b]が原点を含まないかぎり EPST=0.0 とすることができ，この場合本サブルーチンは，達成できるかぎりの精度を持つ根を計算する．

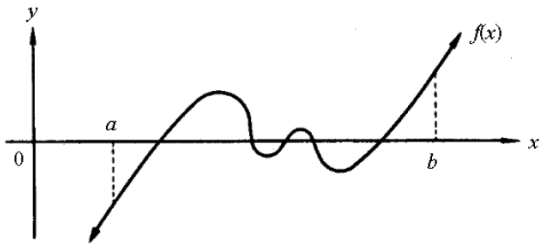


図 TSD1-1

c. 使用例
$f(x) = \sin^2(x) - 0.5 = 0$ の，区間[0.0,1.5]内の 1 根を求める．明らかに，この区間の両端で $f(x)$ は異符号を持ち，かっこの区間で 1 根しか持たない．

```
C       **EXAMPLE**
        EXTERNAL FSIN
        AI=0.0
        BI=1.5
        EPST=1.0E-6
        CALL TSD1(AI,BI,FSIN,EPST,X,ICON)
        WRITE(6,600) X,ICON
        STOP
  600   FORMAT(10X,E15.7,5X,I10)
        END
        FUNCTION FSIN(X)
        FSIN=SIN(X)**2-0.5
        RETURN
        END
```

(4) 手法概要

本サブルーチンは広くデッカー (Dekker) のアルゴリズムとして知られている手法に若干の修正を加えた手法を用いている． $f(x)$のふるまいを調べることにより，反復における各ステップにおいて 2 分法，線型補間法，逆 2 次補間法のいずれかをある判断に基づいて使い分ける．

$f(x)$を [a, b]で連続，かつ $f(a)f(b) < 0$ を満たしている実関数とする．アルゴリズムは初期の状態で $c = a$ としその後の一般的な手順は次のとおりである．

手順 1. $f(b)\,f(c) > 0$なら c を a に等しくとる．
$f(b)f(c)<0$ かつ $|f(b)| > |f(c)|$なら b と c を入れかえ，そのあとで a を cに等しくとる．
このようにして常に，

$$f(b)f(c)<0 \text{ かつ } |f(b)| \le |f(c)|$$

の条件を満足するように bと cを決める．
変数 a, b, c の意味は次のとおりである．
b... 最新の近似根．
a... 以前の b.
c... b と対をなす変数であり，b と c の間に根が存在することを示す．

手順 2. $m = (c - b)/2$ として $|m| < \delta$でない限り次の(a), (b), (c)のいずれかにより次の近似根となるべき i を計算しそれを新たに bとする（δ はbの関数で詳しくは後述する）．

(a) $|f(a)|\le|f(b)|$のとき
b と c の間で 2 分法を適用する.
すなわち,
$i = (b + c) /2$

(b) $a = c$ のとき
a と b の間で線型補間法を適用する.
すなわち,

$$i = b - \frac{(a-b) f(b) / f(a)}{1 - f(b) / f(a)} = b + p / q \qquad (4.1)$$

(c) $|f(b)| < |f(a)|\le|f(c)|$のとき
a , b, c 間で逆 2 次補間法を適用する.
すなわち,
$f_a = f(a), f_b = f(b), f_c = f(c),$
更に, $r_1=f_a/f_c, r_2=f_b/f_c, r_3=f_b/f_a,$とするとき

$$\begin{aligned} i &= b - f_b\left[\frac{(c-b) f_a (f_a - f_b) - (b-a) f_c (f_b - f_c)}{(f_a - f_c)(f_b - f_c)(f_b - f_a)}\right] \\ &= b - r_3\left[\frac{(c-b) r_1 (r_1 - r_2) - (b-a)(r_2 - 1)}{(r_1 - 1)(r_2 - 1)(r_3 - 1)}\right] \\ &= b + p / q \end{aligned} \qquad (4.2)$$

以上の手順 1, 手順 2 を繰り返す.

オーバフローの防止

(4.1), (4.2)を適用するとき, オーバフローを防止するための次の配慮が払われている. すなわち

$$|p / q| > \frac{3}{4}|c-b| = \frac{3}{4}|2m| = \frac{3}{2}|m|$$

あるいは書きなおして,

$$2|p| > 3|mq|$$

となるときは p/q の除算は行わず, b と c の間で 2 分法（すなわち上記 a））を適用する.
また $|p/q|<\delta$のときは, 特別に

$$i = b + \delta \mathrm{sign}(c-b)$$

とする（これは収束ののろまさを防ぐためである）.

収束判定法

次の (a), (b)のいずれかが満たされたとき収束したと判断し, その時の b を根とし計算を終える.

(a) $f(b) = 0$
（これはアンダフローが起きたときである.）

(b) $|m|<\delta=2u|b|+\varepsilon/2 \qquad \varepsilon>0 \qquad (4.3)$
ここで u は丸め誤差の単位であり, εは近似根の絶対誤差の許容量であり本サブルーチンのパラメタ EPST に当たる.
(4.3)の右辺の導出は経験的性格を持っている.
詳しくは文献 [8]を参照すること.
真の根をxとするとき, 近似根bの絶対誤差, $|b–x|$の上限は丸め誤差の影響のため (4.3)の右辺よりも多少大きくなる. 結果だけをかけば,

$$|b-x|\le 6u|x|+\varepsilon$$

である.
ここで注意すべきはパラメタ EPSTの値である. もし区間$[a, b]$が原点を含んでいる場合, 真の根が原点である恐れがあるので EPST=0.0とするのは危険である. そうでない場合, EPST=0.0 とすることは一向にかまわない.

なお詳細については, 参考文献 [28] を参照すること.

付　録

付録 1
補助サブルーチン

付1.1　概要

補助サブルーチンとは，SSL II の一般サブルーチン並びにスレーブサブルーチンの機能を，分野を問わず内部的に補助するサブルーチンである，例えば，2.9 節で述べたように SSL II のサブルーチンでは丸め誤差の単位 u (unit round off)を頻繁に用いているが，この値 u を設定する補助サブルーチンを用意し，各サブルーチンは必要に応じてこの補助サブルーチンを呼び出すことにより，値 u を利用している．

このように通常，補助サブルーチンは SSL II の一般サブルーチン並びにスレーブサブルーチンが呼び出すものであり，以下に述べるような性格を持っている．

［補助サブルーチンの性格］

- 補助サブルーチンには，使用する計算機に依存するものがある．（SSL IIの一般サブルーチン並びにスレーブサブルーチンには，計算機に依存するものはない．）
- 補助サブルーチンのパラメタの型には，単精度・倍精度が混在するものがある．（SSL IIの一般サブルーチン並びにスレーブサブルーチンでは，混在しない．）
- 補助サブルーチンのサブルーチン名は，SSL II の一般サブルーチン並びにスレーブサブルーチンのような命名規則はない．（2.3節参照）

表付 1.1 に補助サブルーチンを一覧する．

表付 1.1 補助サブルーチン

項目	サブルーチン名		性格*	説明箇所
	単精度	倍精度		
丸め誤差の単位	AMACH	DMACH	M,F	付 1.2
コンディションメッセージの出力	MGSSL		S	付 1.3
コンディションメッセージの出力制御	MGSET**		S	付 1.4
積和計算（実ベクトル）	ASUM,BSUM	DSUM,DBSUM	S	付 1.5
積和計算（複素ベクトル）	CSUM	DCSUM	S	付 1.6
浮動小数点演算の基底	IRADIX		M,F	付 1.7
浮動小数点数の正の最大値	AFMAX	DFMAX	M,F	付 1.8
浮動小数点数の正の最小値	AFMIN	DFMIN	M,F	

* 性格欄の記号は以下の事を意味する．
M: 計算機に依存した副プログラム．
F: 関数副プログラム．
S: サブルーチン副プログラム．

** MGSET は MGSSL の 2 次入口である．

付 1.2　AMACH, DMACH

丸め誤差の単位 (unit round off)
（関数副プログラム）AMACH(EPS)

(1)　機能

関数の値は，正規化された浮動小数点演算における，丸め誤差の単位 *u* (unit round off)の値が設定される．

ここで *u* は，*M* 進法 *L* 桁の演算で

$u = M^{1-L}/2$（四捨五入演算）

$u = M^{1-L}$（切り捨て演算）

である．具体的には，表 AMACH-1 で示される値が設定される．

(2)　パラメタ

EPS…… **入力**．演算精度に応じて，次のように指定する．

単精度:単精度実数型変数又は定数．

倍精度:培精度実数型変数又は定数．

ただし，変数又は定数の値は，いかなるものでもよい．

(3)　使用例

単，倍精度に対する丸め誤差の単位を求める．

```
C      **EXAMPLE**
       REAL*4 AEPS
       REAL*8 DEPS,DMACH
       AEPS=AMACH(AEPS)
       DEPS=DMACH(DEPS)
       WRITE(6,600) AEPS,DEPS
       STOP
  600  FORMAT(10X,'AEPS=',E16.7,
     *         5X,'DEPS=',D25.17)
       END
```

表 AMACH-1 丸め誤差の単位

演算方法		AMACH 単精度	DMACH 倍精度	適用例
2 進法: $M = 2$	四捨五入	$L = 26$ $u = {}^1/_2 \cdot 2^{-25}$	$L = 61$ $u = {}^1/_2 \cdot 2^{-60}$	
16 進法: $M = 16$	切捨て	$L = 6$ $u = 16^{-5}$	$L = 14$ $u = 16^{-13}$	FACOM M シリーズ FACOM S シリーズ SX/G 200 シリーズ
2 進法: $M = 2$	四捨五入	$L = 23$ $u = {}^1/_2 \cdot 2^{-22}$	$L = 52$ $u = {}^1/_2 \cdot 2^{-51}$	FM シリーズ SX/G シリーズ

付 1.3　MGSSL

コンディションメッセージの出力
CALL MGSSL (ICON,MCODE)

(1)　機能

SSL II の各サブルーチンのコンディションメッセージを出力する.

(2)　パラメタ

ICON ……**入力**. コンディションコード.
MCODE ……**入力**. サブルーチンの分類コード.
大きさ 6 の 1 次元配列.
分類コード 11 桁を文字定数で与える.

例　分類コード A52-22-0101 の場合.
DATA MCODE/2HA5,2H2-,2H22,2H-0,2H10,2H1 /

(3)　使用上の注意

a.　注意

① 本サブルーチンは，SSL II のすべての一般サブルーチンにより呼び出される.
すなわち，すべての一般サブルーチンで，各処理が終了した時点で必ず本サブルーチンに制御が渡される.
ただし，例外として，特殊関数のサブルーチンからは，正常終了(ICON=0) の場合は呼び出されない.

② 本サブルーチンでは，通常はメッセージを出力しない.
本サブルーチンによりメッセージを実際に出力するか否かは，メッセージ出力制御用サブルーチン MGSET より決定される.
サブルーチン MGSET 参照のこと.

b.　使用例

本サブルーチンが，SSL II の一般サブルーチンの内部で呼び出される具体例を，サブルーチン LAX（分類コード: A22-11-0101）により示す.

```
C      **EXAMPLE**
       SUBROUTINE LAX(A,K,N,B,EPSZ,ISW,IS,
      *VW,IP,ICON)
       DIMENSION A(K,N),B(N),VW(N),IP(N)
       CHARACTER  MCODE(6)*4
       DATA MCODE/'A2  ','2-  ','11  ',
      *'-0  ','10  ','1   '/
       IF(ISW.EQ.1) GOTO 1000
       IF(ISW.EQ.2) GOTO 1100
       ICON=30000
       GOTO 8000
 1000  CALL ALU(A,K,N,EPSZ,IP,IS,VW,ICON)
       IF(ICON.NE.0) GOTO 8000
 1100  CALL LUX(B,A,K,N,1,IP,ICON)
 8000  CALL MGSSL(ICON,MCODE)
       RETURN
       END
```

付 1.4　MGSET

コンディションメッセージ出力の制御
CALL MGSET(ISET,IFLE)

(1)　機能

SSL II の一般サブルーチンでのコンディションメッセージ出力を制御する.

(2)　パラメタ

ISET ……**入力**. 出力レベルを指定する.
表 MGSET-1 参照.

IFLE ……**入力**. 出力するファイル参照番号（データセット識別番号）

表 MGSET-1 出力レベル

ISET	制御内容
0	コンディションコードが 0~30000 のとき出力する.
1	コンディションコードが 10000~30000 のとき出力する.
2	コンディションコードが 20000~30000 のとき出力する.
3	コンディションコードが 30000 のとき出力する.
-1	コンディションメッセージを出力しない.

(3)　使用上の注意

a.　注意

①　コンディションメッセージの出力
SSL II の一般サブルーチンでは，処理終了時のコンディションメッセージを出力するために，補助サブルーチン MGSSL を呼び出している.
MGSSL では，通常はメッセージを出力しないが，SSL II サブルーチンを呼び出すことにより，出力させることができる.

②　出力制御の範囲
本サブルーチンによる出力制御は，利用者のプログラムが論理的に再度本サブルーチンを呼び出すまで有効である.
よって，利用者は呼び出す SSL II サブルーチンが，内部で他のサブルーチンを使用しているときは，出力制御がそれらサブルーチンにも及ぶことに注意すること.

③　出力の中断
本サブルーチンの呼出しによりメッセージ出力を開始したのち，その出力を中断したい場合には，ISET=-1 として再度本サブルーチンを呼び出す.

④　ファイル参照番号
通常は，参照番号として IFLE=6 と指定する.

⑤　本サブルーチンは，補助サブルーチン MGSSL の 2 次入口である.

b.　使用例
正値対称行列を係数に持つ連立 1 次方程式を解くサブルーチン LSX を利用する際の，使用例を示す.

```
C      **EXAMPLE**
       DIMENSION A(5050),B(100)
       CALL MGSET(0,6)
   10  READ(5,500) N
       IF(N.EQ.0) STOP
       NT=N*(N+1)/2
       READ(5,510) (A(I),I=1,NT)
       READ(5,510) (B(I),I=1,N)
       WRITE(6,600) N
       CALL LSX(A,N,B,0.0,1,ICON)
       IF(ICON.GE.20000) GOTO 20
       WRITE(6,610) (B(I),I=1,N)
   20  CALL MGSET(-1,6)
       GOTO 10
  500  FORMAT(I5)
  510  FORMAT(5E16.8)
  600  FORMAT('0',4X,'ORDER=',I5)
  610  FORMAT(' ',4X,'SOLUTION VECTOR'
     *  /(5X,E15.7))
       END
```

この例では，何組かの方程式を逐次解くようになっている．MGSET は 2 箇所で呼び出されているが，配列宣言の直後の呼び出しにより，方程式を最初に解く場合には，メッセージが出力される．更に，文番号 20 での呼出しにより，以後の出力は停止される.
なお，サブルーチン LSX は，内部でコンポーネントルーチン(SLDL, LDLX)を使用しているので出力制御はこれらのサブルーチンにも及んでいる.
以下は使用例による結果である.

```
     ORDER=3
****SSL2(A22-51-0202) CONDITION 0****
****SSL2(A22-51-0302) CONDITION 0****
****SSL2(A22-51-0101) CONDITION 0****
     SOLUTION VECTOR
     0.1352613E+01
     0.1111623E+00
     0.4999998E+01

     ORDER=4
     SOLUTION VECTOR
     0.5000001E+01
     0.3333330E-01
     0.1111110E+00
     0.2499998E+00
```

付 1.5　ASUM(BSUM), DSUM(DBSUM)

積和計算（実ベクトル）
CALL ASUM (A, B, N, IA, IB, SUM)

(1)　機能

n 次元の実ベクトル $\boldsymbol{a}$, $\boldsymbol{b}$ が与えられたとき，次のような積和を計算する．

$$\mu = \sum_{i=1}^{n} a_i b_i$$

ただし，$\boldsymbol{a}^T=(a_1, a_2, \ldots a_n)$, $\boldsymbol{b}^T=(b_1, b_2, \ldots, b_n)$である．$n\geq 1$ であること．

(2)　パラメタ

A……**入力**．ベクトル $\boldsymbol{a}$.
　　大きさ |IA|・N の 1 次元配列.

B……**入力**．ベクトル $\boldsymbol{b}$.
　　大きさ |IB|・N の 1 次元配列.

N……**入力**．次元 n.

IA……**入力**．ベクトル $\boldsymbol{a}$ の要素の間隔 ($\neq 0$).
　　通常 1 と指定する．
　　（使用上の注意②参照）

IB……**入力**．ベクトル $\boldsymbol{b}$ の要素の間隔 ($\neq 0$).
　　通常 1 と指定する．
　　（使用上の注意②参照）

SUM……**出力**．積和値 μ.
　　（使用上の注意③参照）

(3)　使用上の注意

a.　注意

①　本サブルーチンの目的
　　数値計算でしばしば現れる積和計算は，その浮動小数点演算における誤差解析の理論より，精度を上げて計算することが，有効数字を保持する上で必要であることが知られている．
　　本サブルーチンでは，使用する計算機に依存した最適な方法により精度をあげて計算するものである．

②　配列 A, B 内のデータ間隔
　　ベクトル $\boldsymbol{a}$ の要素が，配列 A 上に，間隔 p で格納されている場合，IA= p と指定すればよい．
　　同様に，ベクトル $\boldsymbol{b}$ において間隔 q ならば，IB=q と指定すればよい．
　　なお，p, q<0 の場合は，配列 A, B の与え方に注意すること．
　　使用例を参照されたい．

③　BSUM について
　　BSUM の機能は，ASUM と同等であるが，積和値 μ が倍精度で出力される．
　　以下に，ASUM, BSUM, DSUM, DBSUM の比較を示す．

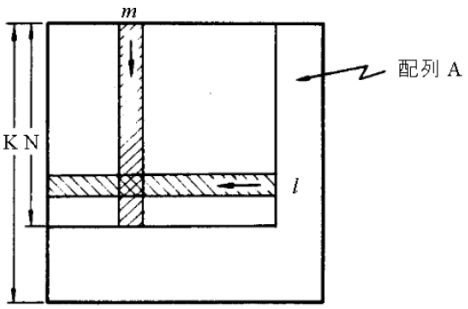
サブルーチン名	使用区分	実パラメタの相違点
ASUM	単精度用ルーチン	A, B, SUM:　単精度
BSUM		A, B:　単精度 SUM: 倍精度
DSUM	倍精度用ルーチン	A, B, SUM:　培精度
DBSUM		A, B:　倍精度 SUM: 4 倍精度

　　したがって，BSUM 又は，DBSUM を使用する場合には，パラメタ SUM を倍精度又は 4 倍精度宣言する必要がある．ただし，FORTRAN システムが 4 倍精度演算機能をサポートしていない場合，DBSUM は使用不可である．

b.　使用例
　　n 次の実行列 $\boldsymbol{A}$ (=(a_{ij}))が，A (K, N) なる 2 次元配列に与えられたとき，その m 列と l 行とによる次のような積和を計算する．

$$\mu = \sum_{i=1}^{n} a_{im} a_{l\,n+1-i}$$

　　下図参照．



$n\leq 100$ の場合．

```
C       **EXAMPLE**
        DIMENSION A(100,100)
        K=100
        READ(5,100) N,((A(I,J),I=1,N),J=1,N)
        READ(5,200) M,L
        CALL ASUM(A(1,M),A(L,N),N,1,-K,SUM)
        WRITE(6,150) M,L,SUM
        STOP
  100 FORMAT(I5/(4E15.7))
  200 FORMAT(2I5)
  150 FORMAT('1'//10X,'M=',I5,5X,'L=',I5,
     *5X,'SUM=',E16.7)
        END
```

付 1.6　CSUM, DCSUM

積和計算（複素ベクトル）
CALL CSUM (ZA, ZB, N, IA, IB, ZSUM)

(1)　機能

n 次元の複素ベクトル ***a***, ***b*** が与えられたとき，次のような積和を計算する.

$$\mu = \sum_{i=1}^{n} a_i b_i$$

ただし，$\boldsymbol{a}^T=(a_1, a_2, ..., a_n)$, $\boldsymbol{b}^T=(b_1, b_2, ...,b_n)$ である. $n\geq 1$ であること.

(2)　パラメタ

ZA …… **入力**. ベクトル ***a***.
　　大きさ |IA|・N の 1 次元配列.

ZB…… **入力**. ベクトル ***b***.
　　大きさ |IB|・N の 1 次元配列.

N…… **入力**. 次元 *n*.

IA…… **入力**. ベクトル ***a*** の要素の間隔 (≠0).
　　通常 1 と指定する.
　　（使用上の注意②参照）

IB…… **入力**. ベクトル ***b*** の要素の間隔 (≠0).
　　通常 1 と指定する.
　　（使用上の注意②参照）

ZSUM…… **出力**. 積和値 μ.

(3)　使用上の注意

a.　注意

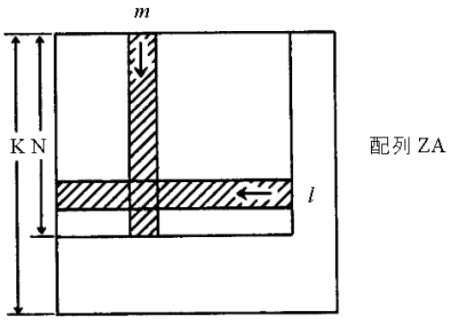
①　本サブルーチンの目的
　　数値計算でしばしば現れる内積計算は，その浮動小数点演算における誤差解析の理論より，精度を上げて計算することが，有効数字を保持する上で必要であることが知られている.
　　本サブルーチンでは，使用する計算機に依存した最適な方法により精度をあげて計算するものである.

②　配列 ZA, ZB 内のデータ間隔
　　ベクトル ***a*** の要素が，配列 ZA 上に，間隔 *p* で格納されている場合，IA= *p* と指定すればよい.
　　同様に，ベクトル ***b*** において間隔 *q* ならば，IB=*q* と指定すればよい.
　　なお，*p*, *q*<0 の場合は，配列 ZA, ZB の与え方に注意すること.
　　使用例を参照されたい.

b.　使用例
　　n 次の複素行列 ***A*** (=(a_{ij}))が，ZA (K, N) なる 2 次元配列に与えられたとき，その *m* 列と *l* 行とによる次のような積和を計算する.

$$\mu = \sum_{i=1}^{n} a_{im} a_{ln+1-i}$$

　　下図参照.



　　n≤100 の場合.

```
C     **EXAMPLE**
      DIMENSION ZA(100,100)
      COMPLEX ZA,ZSUM
      K=100
      READ(5,100) N,((ZA(I,J),I=1,N),
     *            J=1,N)
      READ(5,200) M,L
      CALL CSUM(ZA(1,M),ZA(L,N),N,1,
     *          -K,ZSUM)
      WRITE(6,150) M,L,ZSUM
      STOP
  100 FORMAT(I5/(4E15.7))
  200 FORMAT(2I5)
  150 FORMAT('1'//10X,'M=',I5,5X,'L=',I5,
     * 5X,'CSUM=',2E16.7)
      END
```

付 1.7　IRADIX

浮動小数点演算の基底
（関数副プログラム）IRADIX(RDX)

(1)　機能

関数の値は，浮動小数点演算の基底の値が，整数型で設定される．

具体的には，表 IRADIX-1 で示される値が設定される．

表 IRADIX-1 浮動小数点演算の基底

演算方法	IRADIX	適用例
2 進法	IRADIX=2	FM シリーズ SX/G 100 シリーズ
16 進法	IRADIX=16	FACOM M シリーズ FACOM S シリーズ SX/G 200 シリーズ

(2)　パラメタ

RDX……**入力**．実数型変数又は定数．実数型であれば，数値はいかなるものであってもよい．

(3)　使用例

浮動小数点演算の基底を求める．

```
C     **EXAMPLE**
      REAL*4 RDX
      RDX=IRADIX(RDX)
      WRITE(6,600) RDX
  600 FORMAT(//10X,'RDX=',E15.7)
      STOP
      END
```

付 1.8　AFMAX, DFMAX, AFMIN, DFMIN

浮動小数点数の正の最大値及び最小値	
（関数副プログラム）	最大値: AFMAX (X) 最小値: AFMIN (X)

(1)　機能

正規化された浮動小数点数の正の最大値 fl_{max} 又は最小値 fl_{min} を設定する.

これらの値は計算機の演算方法（2 進法か 16 進法か）により異なり，また最大値は演算精度（単精度か倍精度か）によっても異なる．具体的には表 AFMAX-1 で示される値を設定する.

(2)　パラメタ

X …… 入力．演算精度に応じて，次のように指定する.
単精度:単精度実数型変数又は定数.
倍精度:倍精度実数型変数又は定数.
ただし，変数又は定数の値は，いかなるものでもよい.

(3)　使用例

単，倍精度に対する最大値，最小値を求める.

```
C      **EXAMPLE**
       REAL*4 X0,X1,X2
       REAL*8 Y0,Y1,Y2,DFMAX,DFMIN
       X1=AFMAX(X0)
       X2=AFMIN(X0)
       Y1=DFMAX(Y0)
       Y2=DFMIN(Y0)
       WRITE(6,600) X1,X2,Y1,Y2
       STOP
  600  FORMAT(10X,'AFMAX=',E16.7/
     *10X,'AFMIN=',E16.7/
     *10X,'DFMAX=',D25.17/
     *10X,'DFMIN=',D25.17)
       END
```

表 AFMAX-1 浮動小数点数の正の最大値及び最小値

演算方法	最大値		最小値		適用例
	単精度	倍精度	単精度	倍精度	
	AFMAX	DFMAX	AFMIN	DFMIN	
2 進法	$(1-2^{-26})\cdot 2^{255}$	$(1-2^{-61})\cdot 2^{255}$	$2^{-1}\cdot 2^{-256}$	$2^{-1}\cdot 2^{-256}$	
16 進法	$(1-16^{-6})\cdot 16^{63}$	$(1-16^{-14})\cdot 16^{63}$	$16^{-1}\cdot 16^{-64}$	$16^{-1}\cdot 16^{-64}$	FACOM M シリーズ FACOM S シリーズ SX/G 200 シリーズ
2 進法	$(1-2^{-24})\cdot 2^{128}$	$(1-2^{-53})\cdot 2^{1024}$	$2^{-1}\cdot 2^{-125}$	$2^{-1}\cdot 2^{-1021}$	FM シリーズ
		$(1-2^{-56})\cdot 2^{252}$		$2^{-1}\cdot 2^{-259}$	SX/G 100 シリーズ

付録 2
SSL II サブルーチン名一覧表

本付録は，SSL II のサブルーチン名を，アルファベット順に並べたものである．

一覧表は，サブルーチンの利用形態に応じた分類に従って，三つに区分されている．

なお，サブルーチン名は，単精度演算用だけ表示してある．

付 2.1　一般サブルーチン

サブルーチン名	分類コード	使用する副プログラム名*
AGGM	A21-11-0101	
AKHER	E11-11-0201	AFMAX
AKLAG	E11-11-0101	AFMAX
AKMID	E11-42-0101	
AKMIN	E12-21-0201	
ALU	A22-11-0202	AMACH
AQC8	G23-11-0301	AMACH
AQE	G23-11-0401	AMACH, AFMIN
AQEH	G23-21-0101	AMACH, AFMAX
AQEI	G23-31-0101	AMACH, AFMAX
AQMC8	G24-13-0101	AMACH
AQME	G24-13-0201	AFMAX, AFMIN, UAQE1, UAQE2, UAQE3, UFN10, UFN20, UFN30, UFN40, AMACH
AQN9	G23-11-0201	AMACH
ASSM	A21-12-0101	
ASVD1	A25-31-0201	AMACH
BDLX	A52-31-0302	
BICD1	E12-32-1102	UBAS1, UCIO1,ULUI1, UMIO1
BICD3	E12-32-3302	UBAS1, UCIO3, ULUI3, UMIO3
BIC1	E12-31-0102	UBAS1, UCIO1,ULUI1, UMIO1
BIC2	E12-31-0202	UBAS1, UCIO2, ULUI1, UMIO2
BIC3	E12-31-0302	UBAS1, UCIO3, ULUI3 UMIO3
BIC4	E12-31-0402	UBAS4, UCIO4, ULUI4, UMIO4, UPEP4
BIFD1	E11-32-1101	UBAS1, UCAD1
BIFD3	E11-32-3301	UBAS1, UCAD1
BIF1	E11-31-0101	UBAS1, UCAR1
BIF2	E11-31-0201	UBAS1, UCAR1
BIF3	E11-31-0301	UBAS1, UCAR1
BIF4	E11-31-0401	UBAS4, UCAR4, UPEP4
BIN	I11-81-1201	AMACH,BI0, BI1, AFMIN, AFMAX, ULMAX
BIR	I11-83-0301	AFMIN, AMACH, ULMAX
BI0	I11-81-0601	AFMAX, ULMAX
BI1	I11-81-0701	AFMAX, ULMAX
BJN	I11-81-1001	AMACH, BJ0, BJ1, AFMIN, UTLIM
BJR	I11-83-0101	AMACH, AFMIN, UTLIM
BJ0	I11-81-0201	UTLIM
BJ1	I11-81-0301	UTLIM
BKN	I11-81-1301	BK0, BK1, ULMAX
BKR	I11-83-0401	AMACH, AFMAX, ULMAX
BK0	I11-81-0801	ULMAX
BK1	I11-81-0901	ULMAX
BLNC	B21-11-0202	IRADIX
BLUX1	A52-11-0302	
BLU1	A52-11-0202	AMACH
BMDMX	A52-21-0302	
BSCD2	E32-32-0202	UBAS0, UCDB2, UPOB2, UPCA2, UREO1
BSCT1	B21-21-0502	AMACH
BSC1	E32-31-0102	UBAR1, UCAO1, UCDB1, UNCA1, UREO1
BSC2	E32-31-0202	UBRS0, UCDB2, UPOB1, UPCA1, UREO1, AFMAX
BSEG	B51-21-0201	AMACH,BSCT1, BSVEC, BTRID
BSEGJ	B51-21-1001	AMACH, BDLX, MSBV, SBDL, TEIG1, TRBK, TRID1, UCHLS, UESRT
BSFD1	E31-32-0101	UBAS1, UCAR2
BSF1	E31-31-0101	UBAS1, UCAR1
BSVEC	B51-21-0402	AMACH
BTRID	B51-21-0302	AMACH
BYN	I11-81-1101	BY0, BY1, UBJ0, UBJ1, UTLIM
BYR	I11-83-0201	AMACH, AFMAX, UTLIM, ULMAX
BY0	I11-81-0401	UBJ0, UTLIM
BY1	I11-81-0501	UBJ1, UTLIM
CBIN	I11-82-1101	AMACH, ULMAX
CBJN	I11-82-1301	AMACH, ULMAX
CBJR	I11-84-0101	AMACH, ULMAX
CBKN	I11-82-1201	CBIN, AMACH, ULMAX, UTLIM
CBLNC	B21-15-0202	IRADIX
CBYN	I11-82-1401	CBIN, CBKN, AMACH, ULMAX, UTLIM
CEIG2	B21-15-0101	CBLNC, CHES2, CNRML, AMACH, CSUM, IRADIX
CELI1	I11-11-0101	
CELI2	I11-11-0201	
CFRI	I11-51-0201	UTLIM
CFT	F12-15-0101	CFTN, PNR
CFTM	F12-11-0101	UCFTM
CFTN	F12-15-0202	
CFTR	F12-15-0302	
CGSBM	A11-40-0101	
CGSM	A11-10-0101	
CHBK2	B21-15-0602	
CHES2	B21-15-0302	AMACH, CSUM

サブルーチン名	分類コード	使用する副プログラム名*
CHSQR	B21-15-0402	AMACH
CHVEC	B21-15-0502	AMACH, CSUM
CJART	C22-15-0101	AMACH, CQDR, UCJAR
CLU	A22-15-0202	AMACH, CSUM
CLUIV	A22-15-0602	CSUM
CLUX	A22-15-0302	CSUM
CNRML	B21-15-0702	
COSI	I11-41-0201	UTLIM
CQDR	C21-15-0101	
CSBGM	A11-40-0201	
CSBSM	A11-50-0201	
CSGM	A11-10-0201	
CSSBM	A11-50-0101	
CTSDM	C23-15-0101	AMACH, AFMAX, AFMIN
ECHEB	E51-30-0201	
ECOSP	E51-10-0201	
EIGI	B21-11-0101	AMACH, BLNC, HES1, IRADIX
ESINP	E51-20-0201	
EXPI	I11-31-0101	ULMAX, AFMAX
FCHEB	E51-30-0101	AMACH, UTABT, UCOSM
FCOSF	E51-10-0101	AMACH, UTABT, UCOSM, UNIFC
FCOSM	F11-11-0201	UCOSM, UPNR2, UTABT
FCOST	F11-11-0101	UCOSM, UPNR2, UTABT
FSINF	E51-20-0101	AMACH, UTABT, USINM, UNIFC
FSINM	F11-21-0201	USINM, UPNR2, UTABT
FSINT	F11-21-0101	USINM, UPNR2, UTABT
GBSEG	B52-11-0101	AMACH, MSBV, TRID1, TEIG1, TRBK, UCHLS, UBCHL, UBCLX, UESRT
GCHEB	E51-30-0301	
GINV	A25-31-0101	ADVD1, AMACH
GSBK	B22-21-0402	
GSCHL	B22-21-0302	AMACH, UCHLS
GSEG2	B22-21-0201	AMACH, GSBK, GSCHL, TRBK, TRID1, UCHLS, UTEG2
HAMNG	H11-20-0121	AMACH, RKG
HBK1	B21-11-0602	NRML
HEIG2	B21-25-0201	AMACH, TEIG2, TRBKH, TRIDH, UTEG2
HES1	B21-11-0302	AMACH
HRWIZ	F20-02-0201	AMACH
HSQR	B21-11-0402	AMACH
HVEC	B21-11-0502	AMACH
ICHEB	E51-30-0401	
IERF	I11-71-0301	IERFC
IERFC	I11-71-0401	IERF
IGAM1	I11-61-0101	AMACH, IGAM2, AFMAX, EXPI, ULMAX
IGAM2	I11-61-0201	AMACH, EXPI, ULMAX, AFMAX
INDF	I11-91-0301	IERF, IERFC
INDFC	I11-91-0401	IERF, IERFC
INSPL	E12-21-0101	
LAPS1	F20-01-0101	AFMAX
LAPS2	F20-02-0101	LAPS1, HRWIZ, AFMAX, AMACH
LAPS3	F20-03-0101	
LAX	A22-11-0101	ALU, AMACH, LUX
LAXL	A25-11-0101	AMACH, ULALB, ULALH
LAXLM	A25-21-0101	ADVD1, AMACH
LAXLR	A25-11-0401	AMACH, MAV, ULALB
LAXR	A22-11-0401	AMACH, LUX, MAV
LBX1	A52-11-0101	AMACH, BLUX1, BLU1
LBX1R	A52-11-0401	AMACH, BLUX1, MBV
LCX	A22-15-0101	AMACH, CLU, CLUX, CSUM
LCXR	A22-15-0401	AMACH, CLUX, CSUM, MCV

サブルーチン名	分類コード	使用する副プログラム名*
LDIV	A22-51-0702	
LDLX	A22-51-0302	
LESQ1	E21-20-0101	AMACH
LMINF	D11-30-0101	AMACH
LMING	D11-40-0101	AMACH
LOWP	C21-41-0101	AMACH, RQDR, UREDR, U3DEG
LPRS1	D21-10-0101	ALU, LUIV, AMACH
LSBIX	A52-21-0101	SBMDM, BMDMX, AMACH
LSBX	A52-31-0101	AMACH, BDLX, SBDL
LSBXR	A52-31-0401	AMACH, BDLX, MSBV
LSIX	A22-21-0101	AMACH, MDMX, SMDM, USCHA
LSIXR	A22-21-0401	MDMX, MSV, AMACH
LSTX	A52-31-0501	AMACH
LSX	A22-51-0101	AMACH, LDLX, SLDL
LSXR	A22-51-0401	AMACH, LDLX, MSV
LTX	A52-11-0501	AMACH
LUIV	A22-11-0602	
LUX	A22-11-0302	
MAV	A21-13-0101	
MBV	A51-11-0101	
MCV	A21-15-0101	
MDMX	A22-21-0302	
MGGM	A21-11-0301	
MGSM	A21-11-0401	
MINF1	D11-10-0101	AMACH, LDLX, UMLDL
MING1	D11-20-0101	AMACH, AFMAX, MSV
MSBV	A51-14-0101	
MSGM	A21-12-0401	CSGM, MGGM
MSSM	A21-12-0301	CSGM, MGSM
MSV	A21-14-0101	
NDF	I11-91-0101	
NDFC	I11-91-0201	
NLPG1	D31-20-0101	AMACH, UQP, UNLPG
NOLBR	C24-11-0101	AMACH
NOLF1	D15-10-0101	AMACH
NOLG1	D15-20-0101	AMACH
NRML	B21-11-0702	
ODAM	H11-20-0141	AMACH, UDE, USTE1, UINT1
ODGE	H11-20-0151	AMACH, USDE,UINT2, USTE2, USETC, USETP, USOL, UADJU, UDEC
ODRK1	H11-20-0131	AMACH, URKV, UVER
PNR	F12-15-0402	
RANB2	J12-20-0101	
RANE2	J11-30-0101	RANU2
RANN1	J11-20-0301	
RANN2	J11-20-0101	RANU2
RANP2	J12-10-0101	ULMAX
RANU2	J11-10-0101	
RANU3	J11-10-0201	RANU2
RATF1	J21-10-0101	UX2UP
RATR1	J21-10-0201	UX2UP
RFT	F11-31-0101	CFTN, PNR, URFT
RJETR	C22-11-0111	AFMAX, AFMIN, AMACH, IRADIX, RQDR, UJET
RKG	H11-20-0111	
RQDR	C21-11-0101	AMACH
SBDL	A52-31-0202	AMACH
SBMDM	A52-21-0202	AMACH
SEIG1	B21-21-0101	AMACH,TEIG1, TRID1, TRBK
SEIG2	B21-21-0201	AMACH, TEIG2, TRBK, TRID1, UTEG2
SFRI	I11-51-0101	UTLIM
SGGM	A21-11-0201	
SIMP1	G21-11-0101	
SIMP2	G23-11-0101	AMACH
SINI	I11-41-0101	UTLIM

サブルーチン名	分類コード	使用する副プログラム名*
SLDL	A22-51-0202	AMACH
SMDM	A22-21-0202	AMACH, USCHA
SMLE1	E31-11-0101	
SMLE2	E31-21-0101	
SPLV	E11-21-0101	USPL
SSSM	A21-12-0201	
TEIG1	B21-21-0602	AMACH
TEIG2	B21-21-0702	AMACH, UTEG2
TRAP	G21-21-0101	
TRBK	B21-21-0802	
TRBKH	B21-25-0402	
TRIDH	B21-25-0302	AMACH
TRID1	B21-21-0302	AMACH
TRQL	B21-21-0402	AMACH
TSDM	C23-11-0111	AFMAX, AFMIN, AMACH
TSD1	C23-11-0101	AMACH

* “使用する副プログラム名”の欄には，各サブルーチンが，内部で呼び出す福プログラム（ただし，補助サブルーチン MGSSL は除く）名が記入されている．

付 2.2　スレーブサブルーチン

スレーブ ルーチン名	呼出しサブルーチン名	スレーブ ルーチン名	呼出しサブルーチン名
UADJU	ODGE	ULU14	BIC4
UAQE1	AQME	UMIO1	BICD1, BIC1
UAQE2	AQME	UMIO2	BIC2
UAQE3	AQME	UMIO3	BICD3, BIC3
UBAR1	BSC1	UMIO4	BIC4
UBAS0	BSCD2, BSC2	UMLDL	MINF1
UBAS1	BICD1, BICD3, BIC1, BIC2, BIC3, BIFD1, BIFD3, BIF1, BIF2, BIF3, BSF1,BSFD1	UNCA1	BSC1
		UNIFC	FSINF, FCOSF
UBAS4	BIC4, BIF4	UNLPG	NLPG1
UBCHL	GBSEG	UPCA1	BSC2
UBCLX	GBSEG	UPCA2	BSCD2
UBJ0	BY0	UPEP4	BIF4
UBJ1	BY1	UPNR2	FCOSM, FCOST, FSINM, FSINT
UCAD1	BIFD1, BIFD3	UPOB1	BSC2
UCAO1	BSC1	UPOB2	BSCD2
UCAR1	BIF1, BIF2, BIF3, BSF1	UQP	NLPG1
UCAR2	BSFD1	UREO1	BSC1, BSC2, BSCD2
UCAR4	BIF4	UREDR	LOWP
UCDB1	BSC1	URFT	RFT
UCDB2	BSCD2, BSC2	URKV	ODRK1
UCFTM	CFTM	USCHA	SMDM
UCHLS	BSEGJ, GBSEG, GSCHL, GSEG2	USDE	ODGE
UCIO1	BICD1, BIC1	USETC	ODGE
UCIO2	BIC2	USETP	ODGE
UCIO3	BICD3, BIC3	USINM	FSINM, FSINT
UCIO4	BIC4	USOL	ODGE
UCJAR	CJART	USPL	SPLV
UCOSM	FCOSM, FCOST	USTE1	ODAM
UDE	ODAM	USTE2	ODGE
UDEC	ODGE		
UESRT	BSEGJ, GBSEG	UTABT	FCOSM, FCOST, FSINM, FSINT
UFN10	AQME	UTEG2	GSEG2, TEIG2
UFN20	AQME	UTLIM	BJR, BJ0, BJ1, BYR,BY0, BY1 ,CBKN, CFRI, COSI, SFRI, SINI
UFN30	AQME	UVER	ODRK1
UFN40	AQME	UX2UP	RATF1, RATR1
UINT1	ODAM	U3DEG	LOWP
UINT2	ODGE		
UJET	RJETR		
ULALB	LAXL, LAXLR		
ULALH	LAXL		
ULMAX	BIR, BI0, BI1, BKR, BK0, BK1, BYR, CBIN, CBJN, CBJR, CBKN, EXPI, IGAM2, RANP2		
ULUI1	BICD1, BIC1, BIC2		
ULU13	BICD3, BIC3		

［注意］ 各スレーブサブルーチンは，“呼出しサブルーチン名”の欄に記入されているサブルーチンにより直接呼び出される．

付 2.3　補助サブルーチン

補助サブルーチン名	呼出しサブルーチン名
AMACH	(多くの一般サブルーチン,スレーブサブルーチンが呼び出している．)
MGSET	(SSL II サブルーチン利用者が呼び出すものである．)
MGSSL	(すべての一般サブルーチンが呼び出している．)
IRADIX	BLNC, CBLNC, RJETR
ASUM	
BSUM	
CSUM	GEIG2, CHES2, CHVEC, CLU, CLUX, CLUIV
AFMAX	AKHER, AKLAG, AQME, BI0, BI1, BKR, BYR, CTSDM, EXPI, IGAM2, MING1, RJETR, TSDM, UPOB1
AFMIN	AQE, AQME, BIN, BIR, BJN, BJR, CTSDM, RJETR, TSDM

［注意］付録 1.を参照すること．
なお，MGSET は MGSSL の 2 次入口である．各補助サブルーチンは，“呼出しサブルーチン名”の欄に記入されているサブルーチンにより直接呼び出される．

付録 3
分類コードとサブルーチン名
対応表

A.　線型計算

分類コード	サブルーチン名
A11-10-0101	CGSM
A11-10-0201	CSGM
A11-40-0101	CGSBM
A11-40-0201	CSBGM
A11-50-0101	CSSBM
A11-50-0201	CSBSM
A21-11-0101	AGGM
A21-11-0201	SGGM
A21-11-0301	MGGM
A21-11-0401	MGSM
A21-12-0101	ASSM
A21-12-0201	SSSM
A21-12-0301	MSSM
A21-12-0401	MSGM
A21-13-0101	MAV
A21-14-0101	MSV
A21-15-0101	MCV
A22-11-0101	LAX
A22-11-0202	ALU
A22-11-0302	LUX
A22-11-0401	LAXR
A22-11-0602	LUIV
A22-15-0101	LCX
A22-15-0202	CLU
A22-15-0302	CLUX
A22-15-0401	LCXR
A22-15-0602	CLUIV
A22-21-0101	LSIX
A22-21-0202	SMDM
A22-21-0302	MDMX
A22-21-0401	LSIXR
A22-51-0101	LSX
A22-51-0202	SLDL
A22-51-0302	LDLX
A22-51-0401	LSXR
A22-51-0702	LDIV
A25-11-0101	LAXL
A25-11-0401	LAXLR
A25-21-0101	LAXLM
A25-31-0101	GINV
A25-31-0201	ASVD1
A51-11-0101	MBV
A51-14-0101	MSBV
A52-11-0101	LBX1
A52-11-0202	BLU1
A52-11-0302	BLUX1
A52-11-0401	LBX1R
A52-11-0501	LTX
A52-21-0101	LSBIX
A52-21-0202	SBMDM
A52-21-0302	BMDMX
A52-31-0101	LSBX
A52-31-0202	SBDL
A52-31-0302	BDLX
A52-31-0401	LSBXR
A52-31-0501	LSTX

B.　固有値固有ベクトル

分類コード	サブルーチン名
B21-11-0101	EIG1
B21-11-0202	BLNC
B21-11-0302	HES1
B21-11-0402	HSQR
B21-11-0502	HVEC
B21-11-0602	HBK1
B21-11-0702	NRML
B21-15-0101	CEIG2
B21-15-0202	CBLNC
B21-15-0302	CHES2
B21-15-0402	CHSQR
B21-15-0502	CHVEC
B21-15-0602	CHBK2
B21-15-0702	CNRML
B21-21-0101	SEIG1
B21-21-0201	SEIG2
B21-21-0302	TRID1
B21-21-0402	TRQL
B21-21-0502	BSCT1
B21-21-0602	TEIG1
B21-21-0702	TEIG2
B21-21-0802	TRBK
B21-25-0201	HEIG2
B21-25-0302	TRIDH
B21-25-0402	TRBKH
B22-21-0201	GSEG2
B22-21-0302	GSCHL
B22-21-0402	GSBK
B51-21-0201	BSEG
B51-21-0302	BTRID
B51-21-0402	BSVEC
B51-21-0001	BSEGJ
B52-11-0101	GBSEG

C.　非線型計算

分類コード	サブルーチン名
C21-11-0101	RQDR
C21-15-0101	CQDR
C21-11-0101	LOWP
C22-11-0111	RJETR
C22-15-0101	CJART
C23-11-0101	TSD1
C23-11-0111	TSDM
C23-15-0101	CTSDM
C24-11-0101	NOLBR

D.　極値問題

分類コード	サブルーチン名
D11-10-0101	MINF1
D11-20-0101	MING1
D11-30-0101	LMINF
D11-40-0101	LMING
D15-10-0101	NOLF1
D15-20-0101	NOLG1
D21-10-0101	LPRS1
D31-20-0101	NLPG1

E. 補間・近似

分類コード	サブルーチン名
E11-11-0101	AKLAG
E11-11-0201	AKHER
E11-21-0101	SPLV
E11-31-0101	BIF1
E11-31-0201	BIF2
E11-31-0301	BIF3
E11-31-0401	BIF4
E11-32-1101	BIFD1
E11-32-3301	BIFD3
E11-42-0101	AKMID
E12-21-0101	INSPL
E12-21-0201	AKMIN
E12-31-0102	BIC1
E12-31-0202	BIC2
E12-31-0302	BIC3
E12-31-0402	BIC4
E12-32-1102	BICD1
E12-32-3302	BICD3
E21-20-0101	LESQ1
E31-11-0101	SMLE1
E31-21-0101	SMLE2
E31-31-0101	BSF1
E32-31-0102	BSC1
E32-31-0202	BSC2
E31-32-0101	BSFD1
E32-32-0202	BSCD2
E51-10-0101	FCOSF
E51-10-0201	ECOSP
E51-20-0101	FSINF
E51-20-0201	ESINP
E51-30-0101	FCHEB
E51-30-0201	ECHEB
E51-30-0301	GCHEB
E51-30-0401	ICHEB

F. 変換

分類コード	サブルーチン名
F11-11-0101	FCOST
F11-11-0201	FCOSM
F11-21-0101	FSINT
F11-21-0201	FSINM
F11-31-0101	RFT
F12-11-0101	CFTM
F12-15-0101	CFT
F12-15-0202	CFTN
F12-15-0302	CFTR
F12-15-0402	PNR
F20-01-0101	LAPS1
F20-02-0101	LAPS2
F20-02-0201	HRWIZ
F20-03-0101	LAPS3

G. 数値微積分

分類コード	サブルーチン名
G21-11-0101	SIMP1
G21-21-0101	TRAP
G23-11-0101	SIMP2
G23-11-0201	AQN9
G23-11-0301	AQC8
G23-11-0401	AQE
G23-21-0101	AQEH
G23-31-0101	AQEI
G24-13-0101	AQMC8
G24-13-0201	AQME

H. 微分方程式

分類コード	サブルーチン名
H11-20-0111	RKG
H11-20-0121	HAMNG
H11-20-0131	ODRK1
H11-20-0141	ODAM
H11-20-0151	ODGE

I. 特殊関数

分類コード	サブルーチン名
I11-11-0101	CELI1
I11-11-0201	CELI2
I11-31-0101	EXPI
I11-41-0101	SINI
I11-41-0201	COSI
I11-51-0101	SFRI
I11-51-0201	CFRI
I11-61-0101	IGAM1
I11-61-0201	IGAN2
I11-71-0301	IERF
I11-71-0401	IERFC
I11-81-0201	BJ0
I11-81-0301	BJ1
I11-81-0401	BY0
I11-81-0501	BY1
I11-81-0601	BI0
I11-81-0701	BI1
I11-81-0801	BK0
I11-81-0901	BK1
I11-81-1001	BJN
I11-81-1101	BYN
I11-81-1201	BIN
I11-81-1301	BKN
I11-82-1101	CBIN
I11-82-1201	CBKN
I11-82-1301	CBJN
I11-82-1401	CBYN
I11-83-0101	BJR
I11-83-0201	BYR
I11-83-0301	BIR
I11-83-0401	BKR
I11-84-0101	CBJR
I11-91-0101	NDF
I11-91-0201	NDFC
I11-91-0301	INDF
I11-91-0401	INDFC

J. 擬似乱数

分類コード	サブルーチン名
J11-10-0101	RANU2
J11-10-0201	RANU3
J11-20-0101	RANN2
J11-20-0301	RANN1
J11-30-0101	RANE2
J12-10-0101	RANP2
J12-20-0101	RANB2
J21-10-0101	RATF1
J21-10-0201	RATR1

付録 4
参考文献一覧表

[1] Forsythe. G.E and Moler, C.B.
Computer Solution of Linear Algebraic Systems,
Prentice-Hall, Inc., 1967

[2] Martin R.S.,Peters, G. and Wilkinson, J.H.
Symmetric Decomposition of A Positive Definite Matrix, Linear Algebra,
Handbook for Automatic Computation, Vol.2, pp.9-30,
Springer-Verlag, Berlin-Heidelberg-New York,1971

[3] Bowdler,H.J., Martin, R.S. and Wilkinson, J.H.
Solution of Real and Complex Systems of Linear Equations, Linear Algebra,
Handbook for Automatic Computation, Vol.2,pp.93-110,
Springer-Verlag, Berlin-Heidelberg-New York,1971

[4] Parlett,B.N. and Wang,Y.
The Influence of The Compiler on The Cost of Mathematical Software - in Particular on The Cost of Triangular Factorization,
ACM Transactions on Mathematical Software,Vol.1,No.1,pp.35-46, March, 1975

[5] Wilkinson, J.H.
Rounding Errors in Algebraic Process,
Her Britannic Majesty's Stationary Office,London, 1963

[6] Peter Businger and Gene H. Golub.
Linear Least Squares Solutions by Householder Transformatinos,Linear Algebra,
Handbook for Automatic Computation,Vol.2,pp.111-118,
Springer-Verlag, Berlin-Heidelberg-New York,1971

[7] Martin,R.S and Wilkinson,J.H.
Symmetric Decomposition of Posive Definite Band Matrices,
Linear Algebra,
Handbook for Automatic Computation, Vol.2,pp.50-56,
Springer-Verlag, Berlin- Heidelberg-New York ,1971

[8] Martin,R.S. and Wilkinson,J.H.
Solution of Symmetric and Unsymmetric Band Equations and the Calculations of Eigenvectors of Band Matrices,
Linear Algebra,
Handbook for Automatic Computation,Vol.2,pp.70-92,
Springer-Verlag, Berlin Heidelberg New York,1971

[9] Bunch,J.R and Kaufman,L.
Some Stable Methods for Calculation Inertia and Solving Symmetric,Linear Systems,
Mathematics of Computation, Vol.13,No.137,January 1977,pp.163-179

[10] Bunch,J.R. and Kaufman, L.
Some Stable Methods for Calculation Inertia and Solving Symmertric Linear Systems,
Univ. of Colorado Tech. Report 63, CU:CS:06375

[11] Golub,G.H. and Reinsch,C.
Singular Value Decomposition and Least Squares Solutions, Linear Algebre,
Handbook for Automatic Computation,Vol.2,pp.134-151,
Springer-Verlag,Berlin Heidelberg-New York, 1971

[12] Wilkinson,J.H.
The Algebraic Eigenvalue Problem,
Clarendon Press, Oxford, 1965

[13] Wilkinson,J.H.and Reinsch, C.
Linear Algebra,
Handbook for Automatic Computation,Vol.2,
Springer-Verlag, Berlin-Heidelberg-New York,1971

[14] Smith,B.T., Boyle, J.M, Garbow, B.S, Ikebe, Y, Kleme, V.C and Moler,C.B. Matrix Eigensystem Routine-EISPACK Guide 2nd edit. Lecture Note in Computer Science 6,
Springer-Verlag,Berlin-Heidelberg-New York,1976.

[15] Ortega,J.
The Givens-Householder Method for Symmetric Matrix, Mathematical Methods for Digital Computors,Vol.2,
John wiley&Sons,
New York-London-Sydney,pp.94-115,1967

[16] 戸川隼人
マトリクスの数値計算　3章　pp.146-210
オーム社, 東京, 1971

[17] Mueller,D.J,
Householder's Method for Complex Matrices and Eigensystems of Hermitian Matrices,
Numer.Math.8,pp.72-92,1996

[18] Jennings,A.
Matrix Computation for Engineers and Scientists,
J.Wiley, pp.301-310,1977

[19] 戸川隼人
有限要素法による振動解析　4章　pp.156-162
サイエンス社, 1970

[20] Brezinski, C.
Accélération de la Convergence en Analyse Numérique, Lecture Notes in Mathematics 584, Springer,
pp. 136-159, 1977.

[21] Wynn, P.
Acceleration Techniques for Iterated Vectol and Matrix Problems,
Mathematics of Computation, Vel. 16,
pp. 301-322, 1962.

[22] 山下真一郎
浮動小数点演算における誤差評価について,
情報処理, Vol.15, No.12, pp.935-939, 1974

[23] 山下真一郎, 佐竹誠也
高次代数方程式の根の計算限界について,
情報処理, Vol.7, No.4, pp.197-201, 1966

[24] 平野菅保
代数方程式の誤差, (「数値計算における誤差」より) bit 臨時増刊 12, pp.1364-1378, 1975

[25] 清野 武, 松井照子
低次代数方程式の解法について,
京都大学大型計算機センター広報, Vol.4, No.9,
pp.9-32, 1971

[26] Garside, G.R.,Jarratt,P. and Mack, C.
A New Method for Solving Polynomial Equations,Computer Journal, Vol. 11,pp.87-90,1968

[27] 二宮市三
代数方程式の GJM 解法について,
情報処理学会予稿集, p.33, 1969

[28] Brent, Richard P.
Algorithms for Minimization without Derivatives,
Prentice-Hall, London-Sydney-ToKyo, pp. 47-60, 1973

[29] Peters, G. and Wilkinson, J.H.
Eigenvalues of Ax=λBx with Band Symmetric A and B,
Comp. J. 12,pp.398-404,1969

[30] Jenkins, M. A. and Traub, J.F.
A three-stage algorithm for real polynomials using quadratic iteration,
SIAM J. Number. Anal., Vol.17,pp.545-566, 1970

[31] Jenkins, M. A.
Algorithm 493 Zeros of a real polynomial,
ACM Transaction on Mathematical Software, Vol.1, No.2, pp.178-187,1975

[32] Traub, J. F.
The Solution of Transcendental Equations,
Mathematical Methods for Digital Computers, Vol. 2,
1967, pp. 171-184

[33] Cosnard, M. Y.
A Comparison of Four Methods for Solving Systems of Nonlinear Equations,
Cornell University Computer Science Technical Report TR75-248, 1975

[34] Fletcher, R.
Fortran subroutines for minimization by quasi-Newton methods,
Report R7125 AERE, Harwell, England, 1972

[35] Shanno, D. F. and Phua, K. H.
Numerical comparison of several varible metric algorithms, J. of Optimization Theory and Applications, Vol. 25, No. 4, 1978

[36] Marquardt, D. W.
An algorithm for least squares estimatron of nonlinear parameters,
SIAM J. Appl. Math., 11, pp. 431-441, 1963

[37] Osborne, M. R.
Nonlinear least squares-The Levenberg algorithm revisited, J. of the Australian Mathematical Society, 19, pp. 343-357, 1976

[38] 森口繁一
線形計画法入門
日科技連出版, 1957 (改装版 1973)

[39] 小野勝章
計算を中心とした線形計画法
日科技連出版, 1967 (改装版 1976)

[40] 刀根 薫
数理計画法
朝倉書店, 1978

[41] 古林 隆
線形計画法入門
産業図書, 1980

[42] 古林 隆
ネットワーク理論
日科技連出版, 1976

[43] Dantzig, G.
Linear Programming and Extensions,
Princeton University Press, 1963.

[44] Simonnard, M. (translated by W. S. Jewell)
Linear Programming,
Prentice-Hall,1966.

[45] Vande Panne, C.
Linear Programming and Related Techniqes,
North-Holland, 1971,

[46] Ralston, A.
A First Conrse In Numerical Analysis,
Mc Graw-Hill, New York-Tronto-London, 1965.

[47] Gershinsky, M. and Levine, D. A.
Aitken-Hermite Interpolation,
Journal of the ACM, Vol. 11, No. 3, 1964, pp. 352-356.

[48] Greville, T. N. E.
Spline Functon, Interpolation and Numerical Quadrature, Mathematical Methods for Digital Computers, Vol. 2,
Johe Wiley & Sons, New York-London-Sydney, 1967, pp. 156-168.

[49] Ahlberg, J. H., Nilson, E:N. and Walsh, J. L.
The Theory of Splines and Their Applications.
Academic Press, New York and London, 1967.

[50] Hamming R. W.
Numerical Methods for Scientists and Engineers,
McGraw-Hill, New York-Tronto-London, 1973

[51] Hildebrand, F. B.
Introduction to Numerical Analysis, sec, ed.
McGraw-Hill, 1974.

[52] Akima, H.
A New Method of Interpolation and Smooth Curve

Fitting Based on Local Procedure
Journal of the ACM, Vol. 17, No. 4, 1970. pp. 589-602.

[53] Carl de Boor.
On Calculating with B-splines,
J.Approx. Theory, Vol.6, 1972, pp.50-62.

[54] Akima, H.
Bivariate Interpolation and Smooth Surface Fitting Based on Local Procedures,
Comm. of the ACM., Vol.17, No.1, 1974, pp.26-31

[55] Singlton, R.C.
On Computing the Fast Fourier Transform,
Communications of The ACM,
Vol.10, No.10, pp. 647-654, October, 1967

[56] Singlton, R.C.
An ALGOL Convolution Procedure Based On The Fast Fouier Tronsform,
Communications of The ACM,
Vol.12, No.3, pp.179-184, March, 1969

[57] Singlton, R.C.
An Algorithm for Computing The Mixed Radix Fast Fourier Transform,
IEEE Transactions on Audio and Electroacoustics,
Vol.AU-17, No.2, pp.93-103, June, 1969

[58] 鳥居達生
高速 sine 変換, cosine 変換とその数値積分への応用,
情報処理，Vol.15, No.9, pp.670-679,1974

[59] 鳥居達生
フーリエ変換サブルーチンパッケージの作成
（その 1），（その 2）
名古屋大学大型計算機
センターニュース，Vol.10, No.2, 1979

[60] Fox, L. and Parker, I.B.
Chebyshev Polynomials in Numerical Analysis,
Oxford University Press, 1972

[61] Lyness, J.N.
Notes on the Adaptive Simpson Quadrature Routine,
Journal of the ACM, Vol.16, No.3, 1969, PP.483-495

[62] Davis, P.J. and Rabinowitz,P.
Methods of Numerical Intergration,
Academic Press, 1975

[63] Kahaner, D.K.
Comparison of numerical quadrature formulas.
In “Mathematical Software” (Ed. J.R. Rice),
Academic Press, 1971, pp.229-259

[64] De Boor, C.
CADRE:An algorithm for numerical quadrature.
In “Mathematical Software” (Ed. J.R. Rice),
Academic Press, 1971, pp.417-449

[65] Clenshaw, C.W. and Curtis, A.R.
A method for numerical integration on an automatic computer.
Numer.Math.2, 1960, pp.197-205

[66] 鳥居達生，長谷川武光，二宮市三
等差数列的に標本点を増す補間的自動積分法,
情報処理　Vol.19, No.3, 1978, pp.248-255

[67] Takahashi, H. and Mori, M.
Double Exponential Formulas for Numerical Integration. Publications of R.I.M.S, Kyoto Univ. 9, 1974, pp.721-741

[68] 森 正武
曲線と曲面
教育出版，1974

[69] Romanelli, M.J.
Runge-Kutta Methods for Solution of Ordinary Differential Equations, Mathematical Methods for Digital Computer,
Vol.2, John Wiley & Sons,
New York-London-Sydney, 1967, PP.110-120

[70] Hamming, R.W.
Stable Predictor Corrector Methods for Ordinary Differential Equations,
Jouranl of the ACM, Vol.6, 1956, PP.37-47

[71] Shampine, L.F. and Gordon, M.K.
Computer Solution of Ordinary Differential Equations,
Freeman, 1975

[72] Verner, J.H.
Explicit Runge-Kutta methods with estimate of the local truncation error,
SIAM J. Numer. Anal.
Vol.15, No.4, 1978 pp.772-790

[73] Jackson, K.R., Enright, W.H. and Hull, T.E.
A theoretical criterion for comparing Runge-Kutta Formulas,
SIAM J.Numer, Anal., Vol.15, No.3, 1978, pp.618-641

[74] Gear, C.W.
Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, 1971

[75] Lambert, J.D.
Computational Methods in Ordinary Differential Equations, Wiley, 1973

[76] Hindmarsh, A.C. and Byrne, G.D.
EPISODE:An Effective Package for the Integration of Systems of Ordinary Differential Equations, UCID-30112, Rev.1, Lawrence Livermore Laboratory, April 1977

[77] Byrne, G.D. and Hindmarsh, A.C.
A Polyalgorithm for the Numerical Solution of Ordinary Differential Equations,
ACM Transaction of Math. Soft., Vol.1, No.1, pp.71-96, March 1975.

[78] Hart, J.F.
Complete Elliptic Integrals,
John Wiley & Sons,
Computer Approximations, 1968.

[79] Cody, W.J. and Thacher Jr, C.H.
Rational Chebyshev Approximations for the Exponential Integral $E_i(x)$,
Mathematics of Computation, Vol.22, pp.641-649, July 1968.

[80] Cody, W.J. and Thacher Jr, C.H.
Chebyshev Approximations for the Expontial Integral $E_i(x)$,
Mathematics of Computation, Vol.23,
pp.289-303, Apr. 1969.

[81] 二宮市三
漸化式による Bessel 関数の計算,
培風館，電子計算機のための数値計算法Ⅱ,
pp.103-120, 1967
[82] 宇野利雄
Bessel 関数,
培風館，電子計算機のための数値計算法Ⅲ,
pp.166-186, 1972.
[83] 吉田年雄，浅野道雄，海野正義，三木七郎
漸化式を用いる 複素変数のベッセル関数 $I_n(z)$の数値計算,
情報処理，Vol.14, No.1, pp.23-29, Jan.1973
[84] 吉田年雄，二宮市三
τ-method による複素変数のベッセル関数 $K_n(z)$ の数値計算,
情報処理，Vol.14, No.8, pp.569-575, Aug.1973.
[85] 森口繁市，宇田川銈久，一松信
数学公式Ⅲ ー特殊関数ー,
岩波全書，1967.
[86] Forsythe,G.E.著，森 正武訳
計算機のための数値計算法（コンピュータ・サイエンス研究所シリーズ）
科学技術出版社, 1978.
[87] Streck, A.J.
On the Calculation of the Inverse of Error Function,
Mathematics of Computation, Vol.22, 1968.
[88] 吉田年雄，二宮市三
x が小さい場合の変形ベッセル関数 $K_\nu(x)$の計算,
情報処理学会論文誌，Vol.21, No.3, pp.238-245, May.1980
[89] Nayler, T.H.
Computer Simulation Techiques,
John Wiley & Sons,
Inc., 1966, pp.43-67.
[90] Algorithm 334, Normal Random Deviates,
Comm. ACM, 11 (July 1968) , p.498.
[91] Pike, M.C.
Algorithm 267, Random Normal Deviate,
Comm. ACM, 8 (Oct. 1965), p.606.
[92] 脇本和昌
乱数の知識（森北出版，1970）
[93] 宮竹 修，脇本和昌
乱数とモンテカルロ法（森北出版，1978）
[94] Powell, M.J.D.
A Fast Algorithm for Nonlinearly constranied Optimization Calculations.
Proceedings of the 1977 Dundee Conference on Numerical Analysis,
Lecture Notes in Mathematics, Springer-Verlag, 1978
[95] Powell, M.J.D., et al.
The Watchdog Technique for forcing Convergence in Algorithms for Constrained Optimization, Presented at the Tenth International Symposium on Mathematical Programming, Montreal, 1979
[96] 二宮市三
適応型ニュートン・コーツ積分法の改良,
情報処理，Vol.21, No.5, 1980, pp.504-512.
[97] Ninomiya, I.
Improvements of Adaptive Newton-Cotes Quadrature Methods,
Joumal of Information Processing, Vol.3, No.3, 1980, pp.162-170.
[98] 細野敏夫
“数値ラプラス変換”
電気学会論文誌 A,
Vol.99-A, No.10, Oct.,1979
[99] 細野敏夫
“線形ブラックボックスの基礎”
コロナ社，1980
[100] Bromwich, T.J.I’A.
Introduction to the Theory of Infinite Series.
Macmillan.1926
[101] Hosono, T.
Numerical inversion of Laplace transform and some applications to wave optics.
International U.R.S.I-Symposium 1980 on Electromagnetic Waves, Munich, 1980.

著作者氏名と著作物の索引

著作者氏名	サブルーチン名	項目
田 中 正 次	ODRK1	連立 1 階常微分方程式（ルンゲ・タック・ヴァーナー法）
二 宮 市 三	CGSBM	行列格納モードの変換（一般モード→対称バンド行列用圧縮モード）
	CSBGM	行列格納モードの変換（対称バンド行列用圧縮モード→一般モード）
	CSSBM	行列格納モードの変換（対称行列用圧縮モード→対称バンド行列用圧縮モード）
	CSBSM	行列格納モードの変換（対称バンド行列用圧縮モード→対称行列用圧縮モード）
	LSBIX	実対称バンド行列の連立 1 次方程式（ブロック対角ピボッティング手法）
	SBMDM	実対称バンド行列の MDM^T 分解（ブロック対角ピボッティング手法）
	BMDMX	MDM^T 分解された実対称バンド行列の連立 1 次方程式
	LAXLM	実行列の最小二乗最小ノムル解（特異値分解法）
	ASVD1	特異値分解（ハウスホルダー法，QR 法）
	GINV	一般逆行列（特異値分解法）
	CEIG2	複素行列の固有値及び固有ベクトル（QR 法）
	CBLNC	複素行列の平衡化
	CHES2	複素行列のヘッセルベルグ行列化（安定基本相似変換）
	CHSQR	複素ヘッセンベルグ行列の固有値（QR 法）
	CHVEC	複素ヘッセンベルグ行列の固有ベクトル（逆反復法）
	CHBK2	複素行列の固有ベクトルへの逆変換
	CNRML	複素行列の固有ベクトルの正規化
	BSEG	実対称バンド行列の固有値及び固有ベクトル （ルティスハウザー・シュワルツ法，バイセクション法，逆反復法）
	BTRID	実対称バンド行列の三重対角行列への変換（ルティスハウザー・シュワルツ法）
	BSVEC	実対称バンド行列の固有ベクトル（逆反復法）
	BSEGJ	実対称バンド行列の固有値及び固有ベクトル（ジェニングス法）
	GBSEG	実対称バンド行列の一般固有値及び固有ベクトル（ジェニングス法）
	AQN9	1 次元有限区間積分（関数入力，適応型ニュートン・コーツ 9 点則）
	IERF	逆誤差関数 $\mathrm{erf}^{-1}(x)$
	IERFC	逆余誤差関数 $\mathrm{erfc}^{-1}(x)$
	NDF	正規分布関数 $\phi(x)$
	INDF	逆正規分布関数 $\phi^{-1}(x)$
	NDFC	余正規分布関数 $\psi(x)$
	INDFC	逆余正規分布関数 $\psi^{-1}(x)$
	RANNI	正規乱数の生成（高速型）
鳥 居 達 生	FCOSF	偶関数の cosine 級数展開（関数入力，高速 cosine 変換）
	ECOSP	cosine 級数の求和
	FSINF	奇関数の sine 級数展開（関数入力，速度 sine 交換）
	ESINP	sine 級数の求和
	FCHEB	実関数のチェビシェフ級数展開
	ECHEB	チェビシェフ級数の求和
	GCHEB	チェビシェフ級数の導関数
	ICHEB	チェビシェフ級数の不定積分
	FCOST	離散型 consine 変換（台形公式，2 基底 FFT）
	FCOSM	離散型 consine 変換（中点公式，2 基底 FFT）
	FSINT	離散型 sine 変換（台形公式，2 基底 FFT）
	FSINM	離散型 sine 変換（中点公式，2 基底 FFT）
長 谷 川 武 光	AQC8	1 次元有限区間積分（関数入力，クレンショー・カーチス型積分法）
	AQMC8	多次元有限領域積分（関数入力，クレンショー・カーチス型積分法）
秦 野 和 郎	BICD1	B-spline 2 次元補間式 (I -I)
	BICD3	B-spline 2 次元補間式 (III-III)
	BIC1	B-spline 補間式 (I)
	BIC2	B-spline 補間式 (II)
	BIC3	B-spline 補間式 (III)

著作者氏名	サブルーチン名	項目
	BIC4	B-spline 補間式 (IV)
秦 野 和 郎	BIFD1	B-spline 2 次元補間式 (I-I) による補間，数値微分，数値積分
	BIFD3	B-spline 2 次元補間式 (III-III) による補間，数値微分，数値積分
	BIF1	B-spline 補間式 (I) による補間，数値微分，数値積分
	BIF2	B-spline 補間式 (II) による補間，数値微分，数値積分
	BIF3	B-spline 補間式 (III) による補間，数値微分，数値積分
	BIF4	B-spline 補間式 (IV) による補間，数値微分，数値積分
	BSCD2	B-spline 2 次元平滑化式（節点追加方式）
	BSC1	B-spline 平滑化式（固定節点）
	BSC2	B-spline 平滑化式（節点追加方式）
	BSFD1	B-spline 2 次元平滑化式による平滑化，数値微分，数値積分
	BSF1	B-spline 平滑化式による平滑化，数値微分，数値積分
秦 野 甯 世	AKMID	2 次元準エルミート補間式による補間
	AQE	1 次元有限区間積分（関数入力，二重指数関数型積分公式）
	AQEH	1 次元半無限区間積分（関数入力，二重指数関数型積分公式）
	AQEI	1 次元全無限区間積分（関数入力，二重指数関数型積分公式）
	AQME	多次元積分（関数入力，二重指数関数型積分公式）
吉 田 年 雄	CBIN	複素変数第 1 種整数次変形ベッセル関数 $I_n(z)$
	CBKN	複素変数第 2 種整数次変形ベッセル関数 $K_n(z)$
	CBJN	複素変数第 1 種整数次ベッセル関数 $J_n(z)$
	CBYN	複素変数第 2 種整数次ベッセル関数 $Y_n(z)$
	BJR	第 1 種実数次ベッセル関数 $Jv(x)$
	BYR	第 2 種実数次ベッセル関数 $Yv(x)$
	BIR	第 1 種実数次変形ベッセル関数 $Iv(x)$
	BKR	第 2 種実数次変形ベッセル関数 $Kv(x)$
	CBJR	複素変数第 1 種実数次ベッセル関数 $Jv(x)$
刀 根 薫	LMINF	1 変数関数の極小化（微係数不要，2 次補間法）
	LMING	1 変数関数の極小化（微係数要，3 次補間法）
	MING1	多変数関数の極小化（微係数要，準ニュートン法）
	NOLF1	関数二乗和の極小化（微係数不要，改訂マルカート法）
	NOLG1	関数二乗和の極小化（微係数要，改訂マルカート法）
	NLPG1	非線形計画問題（微係数要，パウエル法）
古 林 隆	LPRS1	線形計画問題（改訂シンプレックス法）
細 野 敏 夫	LAPS1	ラプラス変換（複素右半平面で正則な有理関数）
	LAPS2	ラプラス変換（一般の有理関数）
	LAPS3	ラプラス変換（一般関数）
	HRWIZ	Hurwitz 多項式の判定
L.F. Shampine	ODAM 注）	連立１階常微分方程式（アダムス法）
A.C. Hindmarsh	ODGE 注）	スティフ連立１階常微分方程式（ギア法）

注）このプログラムは，米国のアルゴンヌ国立研究所コードセンタ (ANL-NESC) に登録されているものに基づいている．このプログラムの原版は，同センタから直接入手することができる．

NESC: National Energy Software Center
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IIIinois 60439
U.S.A.