FUJITSU

FUJITSU Software

FUJITSU
SSL II 拡張機能使用手引書
(科学用サブルーチンライブラリ)

J2UL-1904-01Z0(01)
2020年2月

ま　え　が　き

本マニュアルは，SSLⅡ(Scientific Subroutine Library Ⅱ)の拡張機能の使用方法について述べたものです．

SSL Ⅱは，標準機能と拡張機能で構成されています．標準機能は，“FUJITSU SSL Ⅱ使用手引書”で述べる機能を意味し，もっぱら汎用計算機上の広範囲な科学計算を狙っています．一方，本マニュアルで述べる拡張機能は，大規模な科学計算をベクトル計算機(FUJITSU VPシリーズ)上で高速に処理することを狙っています．

現在は当時の機能を互換のために提供しています．

本マニュアルの構成は次のとおりです．

第Ⅰ部　概説

分野別に機能の概要を述べ，利用者の目的にあったサブルーチンの選択方法を示します．

第Ⅱ部　サブルーチン使用方法

個々のサブルーチンの使用方法を述べます．各サブルーチンはアルファベット順に編集されています．

なお，SSL Ⅱの全般的な規約及び標準機能については，次のマニュアルを参照して下さい．

“FUJITSU SSL Ⅱ使用手引書”

本書は，旧マニュアル(マニュアルコード99SP-4070-1)をもとに作成しました．記載内容が旧マニュアルと異なる箇所の項目には，目次に＊印を付けています．

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
2020年　2月 第3版 改訂	J2UL-1904-01Z0(01)
2014年 10月 第3版	J2UL-1904-01Z0(00)
2011年　3月 第2版 改訂3	—
2008年　2月 第2版 改訂2	—
2007年　5月 第2版 改訂	—
1991年　8月 第2版	—
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3.1版

目　　次

SSL Ⅱ 拡張機能一覧表

線型計算

サブルーチン名	項　　目
VMGGM	行列の積(実行例)
VLSX	正値対称行列の連立1次方程式(変形コレスキー法)
VSLDL	正値対称行列のLDL^T分解
VLDLX	LDL^T分解された正値対称行列の連立1次方程式
VLTX	実3項行列の連立1次方程式(サイクリック・リダクション法)
VLTX1	定数型実3項行列の連立1次方程式(ディリクレ型，サイクリック・リダクション法)
VLTX2	定数型実3項行列の連立1次方程式(ノイマン型，サイクリック・リダクション法)
VLTX3	定数型実3項行列の連立1次方程式(周期型，サイクリック・リダクション法)
VLAX	実行列の連立1次方程式(ブロッキングLU分解法)
VALU	実行列のLU分解法(ブロッキングLU分解法)
VLUIV	LU分解法された実行列の逆行列

固有値固有ベクトル

サブルーチン名	項　　目
VSEG2	実対称行列の固有値固有ベクトル(並列バイセクション法，逆反復法)
VGSG2	実対称行列の一般固有値固有ベクトル(並列バイセクション法，逆反復法)

フーリエ変換

サブルーチン名	項　　目
VCOS1	離散型cosine変換(2基底FFT)
VSIN1	離散型sine変換(2基底FFT)
VRFT1	離散型実フーリエ変換(性能優先型，2基底FFT)
VRFT2	離散型実フーリエ変換(メモリ節約型，2基底FFT)
VCFT1	離散型複素フーリエ変換(性能優先型，2基底FFT)
VCFT2	離散型複素フーリエ変換(メモリ節約型，2基底FFT)

第Ⅰ部　概　　説

第１章　SSL Ⅱ拡張機能の概要

1.1　拡張機能とは

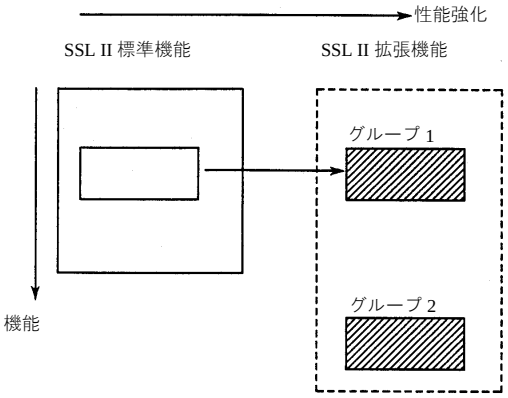
科学技術計算の分野では，膨大な計算を要する問題がある．流体力学，構造解析，分子科学，核融合などから派生する数学モデルはその例である．これらの分野では，問題の大型化は進む一方であり，必然的に高速計算の技術が必要となる．ベクトル計算機はこれに応えるためのハードウェアであり，汎用計算機とは異なったアーキテクチャを採用することにより著しい高速化を実現している．これに伴って，数値計算のアルゴリズムもベクトル計算機によく適合したアルゴリズムが必要となる．

SSLⅡ拡張機能とは，こうした需要に応えるためベクトル計算機上で高速処理することを狙ったサブルーチン群を意味し，手法の面ではハードウェアの能力を最大限に活かすアルゴリズムが用いられている．これに対して，マニュアル“FUJITSU SSLⅡ使用手引書”で述べられているサブルーチン群をSSLⅡ標準機能と呼ぶ，この標準機能は，汎用計算機上の広範囲な科学計算を狙ったものである．

本マニュアルでは単にSSLⅡと言う場合，標準機能と拡張機能を合せた集合を指すものとする．

1.2　拡張機能の構成

拡張機能は二つのグループから成る(図1.1参照)．グループ1は，ベクトルアルゴリズムを用いることによりSSLⅡ標準機能の幾つかを改造し，ベクトル計算機上での高速化を図ったものである．これらのサブルーチンは高速化の手段としてアルゴリズムの変更はもとより，データの格納方法の変更，作業用配列の増設を行っており，その結果，入出力インタフェースが標準機能の該当サブルーチンとは異なっ



ている．なお，ベクトル計算機用に提供される標準機能の多くのサブルーチンは，入出力インタフェースを変えない範囲でベクトル計算機向きに改造されている．したがって，グループ1は，別の言い方をすれば，標準機能の入出力インタフェースを変えることによりベクトル計算機用に一層高速化されたサブルーチン群であると定義することができる．

グループ2は，SSLⅡ標準機能にない，大型問題指向の機能である．ここでもベクトルアルゴリズムが用いられている．

1.3　拡張機能と標準機能の使い分け

SSLⅡは，汎用計算機にもベクトル計算機にも，全く同じ機能群が提供される．したがって，SSLⅡを呼び出している利用者プログラムは，呼出し法を変更することなく，いずれの機種でも動作させることができる．

しかし，グループ1に関しては，標準機能の中に，機能的に類似したサブルーチンがあるので，処理効率の理由から，汎用計算機とベクトル計算機の間で，次のように使い分けるのが望ましい．

(1) 標準機能のサブルーチンを呼び出しているプログラムをベクトル計算機で実行するとき，もしグループ1に同等機能のサブルーチンがあれば，それを呼び出すように変更する．

(2) 逆に，グループ1のサブルーチンを呼び出しているプログラムを汎用計算機で実行するとき，該当する標準機能のサブルーチンを呼び出すように変更する．ただし，汎用計算機で実行することが，一時的なもの，例えばデバッグを目的とする場合は，その必要はない．

グループ1のサブルーチンが，標準機能のどのサブルーチンに対応するかは，各分野の導入の章に述べてある．

利用者プログラム内のSSLⅡ呼出し文を変更することは，多少なりとも手間を要するが，機種によってサブルーチンを使い分けることは，処理効率を上げるために大切なことである．

サブルーチンを使い分けたことによる計算結果の精度の差は無いと考えてよい．SSLⅡのベクトルアルゴリズムは，精度についても十分な考慮がなされている．

第2章　線型計算

2.1　概　　要

本章では，線型計算を取り扱う．

拡張機能として用意されているサブルーチン名と，対応する標準機能のサブルーチン名を，表2.1に示す．

表2.1　線型計算のサブルーチン

機　　能	拡張機能 ルーチン名	標準機能 ルーチン名
行列の積	VMGGM	MGGM
正値対称行列の 連立1次方程式	VLSX (VSLDL) (VLDLX)	LSX (SLDL) (LDLX)
3項行列の 連立1次方程式	VLTX VLTXl VLTX2 VLTX3	LTX LSTX
実行列の連立1次 方程式・逆行列	VLAX (VALU) (VLUIV)	LAX (ALU) (LUIV)

括弧の付いたサブルーチンは，コンポーネントルーチンである．例えば，VSLDLは正値対称行列のLDL^T分解，VLDLXは分解された行列に基づく求解を行うものであり，これらはVLSXにとってコンポーネントの役割を持っている．

各サブルーチンはベクトル計算機で高速に実行されるように，ベクトルアルゴリズムを採用している．以下では，使用に当たっての一般的な注意事項，及びサブルーチンの使い分けについて述べる．

2.2　使用上の注意

ベクトルアルゴリズムを採用している関係で，拡張機能のサブルーチンは標準機能のそれらとは仕様上，以下のような異なった点がある．主なものは以下の2点である．

(1) 正値対称行例，及び3項行列の格納方法が標準機能の場合と異なっている．

(2) 作業領域は一般に，標準機能よりも幾分多い．

これは，ベクトルアルゴリズムの構築に伴い，メモリアクセスを効率良く行うためである．標準機能のサブルーチンから拡張機能のサブルーチンへ呼出しを変える際は，注意が必要である．

2.3　サブルーチンの使い分け

表2.1に挙げたように，3項行列に関するサブルーチンが四つあり，各々は扱う行列の型が違う．それらの使い分けを述べる．

四つのサブルーチンが扱う3項行列は，アルゴリズムの数値的安定性の理由で，既約優対角性を満たすものに限定している．ここで，既約優対角性とは，3項行列を(2.1)で表したとき，(2.2)の条件を満たすことをいう．

$$\begin{bmatrix} d_1 & f_1 & & & & \\ e_2 & d_2 & f_2 & & & \\ & e_3 & d_3 & f_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & f_{n-1} \\ & & & & e_n & d_n \end{bmatrix} \qquad (2.1)$$

$|d_i| \geqq |e_i| + |f_i|,\ i = 1, 2, ..., n$
かつ，少なくとも一つのiについて，　　(2.2)
狭義の不等号が成り立つ．
ただし，$(e_1 = f_n = 0)$

応用面で現れる3項行列は，だいたいこの性質を満たしている．既約優対角性を満たさない場合は，標準機能のルーチンを使用すること．

最初のサブルーチンVLTXは(2.1)の行列を扱う最も一般的なサブルーチンである．これに対して，VLTX1はVLTXの限定版であり，(2.3)の行列を扱う．

$$\begin{bmatrix} d & e & & & & & \\ e & d & e & & & \mathbf{0} & \\ & e & d & e & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & \mathbf{0} & & & \cdot & \cdot & e \\ & & & & & e & d \end{bmatrix} \qquad (2.3)$$

(2.3)のように，要素の値が行と列によらず一定である行列を，本マニュアルでは，定数型3項行列と呼ぶ．

あるいは，この行列が，ある種のディリクレ境界値問題から派生することから，より厳密に，ディリクレ型の定数型3項行列と呼ぶ．

次に，サブルーチンVLTX2，VLTX3の扱う行列は，(2.3)を若干，変形したものである．VLTX2では(2.3)における1行2列の要素，又はn行n-1列の要素が$2e$となった行列を扱い，これをノイマン型の定数型3項行列と呼ぶ．VLTX3では，1行n列の要素とn行1列の要素にeが加わった行列を扱い，これを周期型の定数型3項行列と呼ぶ．いずれも微分方程式の境界値問題から派生する行列である．

アルゴリズムとしては，ベクトル計算機向きのアルゴリズムであるサイクリック・リダクション法(Cyclic Reduction Method)を採用している．これはガウス消去法と比較して計算量自体は多いが，ベクトル計算機にとって重要な，計算の並列性が著しく高い．精度については，既約優対角性を満たす行列ならば，ガウス消去法と同等である．

サブルーチンVLTX1，VLTX2は行列の型が単純であるため，VLTXよりも処理が速い．

第3章　固有値固有ベクトル

3.1　概　　要

本章では，行列の固有値問題を扱う．

拡張機能として用意されているサブルーチン名と，対応する標準機能のサブルーチン名を，表3.1に示す．

表3.1　固有値問題のサブルーチン

問題の種類	行列の種類	拡張機能ルーチン名	標準機能ルーチン名
$\boldsymbol{Ax} = \lambda \boldsymbol{x}$	$\boldsymbol{A}$:実対称行列	VSEG2	SEIG2
$\boldsymbol{Ax} = \lambda \boldsymbol{Bx}$	$\boldsymbol{A}$:実対称行列 $\boldsymbol{B}$:正値対称行列	VGSG2	GSEG2

3.2　使用上の注意

拡張機能のサブルーチンでは，指定されたm個の部分固有値を並列バイセクション法により同時に計算する方法を採用している．そのため，拡張機能と標準機能とでは，作業領域の確保の仕方などが異なる． したがって，標準機能のサブルーチンから拡張機能のサブルーチンへ呼出しを変える際は，パラメタの修正が必要である．

第4章　フーリエ変換

4.1　概　　要

本章では，離散型フーリエ変換を取り扱う．

拡張機能として用意されているサブルーチン名と，対応する標準機能のサブルーチン名を，表4.1に示す．

表4.1　離散型実フーリエ変換のサブルーチン

変換の種類	データ数	拡張機能ルーチン名	特　徴	標準機能ルーチン名
cosine変換	2の巾	VCOS1	－	FCOST
sine変換	2の巾	VSIN1	－	FSINT
実変換	2の巾	VRFT1	性能優先型	RFT
		VRFT2	メモリ節約型	
複素変換	2の巾	VCFT1	性能優先型	CFT
		VCFT2	メモリ節約型	

4.2　使用上の注意

(1)　拡張機能と標準機能の使い分け

離散型フーリエ変換を，汎用計算機上で処理することを目的とする場合は，表4.1に示す対応する標準機能のサブルーチンを用いることが望ましい．

拡張機能のサブルーチンを汎用計算機上で用いることもできるが，処理効率は標準機能のサブルーチンに比べ低下する．

(2)　性能優先型とメモリ節約型の使い分け

性能優先型のサブルーチンは，複数組の変換を処理する場合に適している．すなわち，複数組の変換で共通に使用できる回転因子(三角関数表)やリストベクトルなどを，作業用配列に保存することにより高速化をはかっている．したがって，性能優先型のサブルーチンでは，作業用配列VW，IVWを余分に必要とする．

一方，単独の変換を処理する場合は，メモリ節約型のサブルーチンが適している．

(3)　単精度演算ルーチンの効果的な使用法

単精度演算ルーチンでは，使用するベクトル計算機で最適な性能を得るために，メモリインタリーブ数を意識したアルゴリズムを採用している．そのために，メモリインタリーブ数をSSLⅡに通知する次のような機能が用意されている．

機　能	メモリインタリーブ数の初期設定
呼び出し	CALL SETBNK(INTER)

INTERは，使用するベクトル計算機のメモリインタリーブ数を指定する入力パラメタである．

利用者プログラムでは，単精度演算のフーリエ変換ルーチンを呼び出す直前に，上記プログラムを呼び出すことにより，最適な性能を得ることができる．

なお，上記プログラムを呼び出さない場合は，インタリーブ数は64とみなして処理される．

第Ⅱ部　サブルーチン使用方法

A22－71－0202　　VALU, DVALU

実行列のLU分解(ブロッキングLU分解法)
CALL VALU (A, K, N, EPSZ, IP, IS, VW, ICON)

(1)　機能

n×nの正則な実行列$\boldsymbol{A}$をブロッキングLU分解法(ガウスの消去法)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，$\boldsymbol{L}$は下三角行列，$\boldsymbol{U}$は単位上三角行列である.

n≧1であること,

(2)　パラメタ

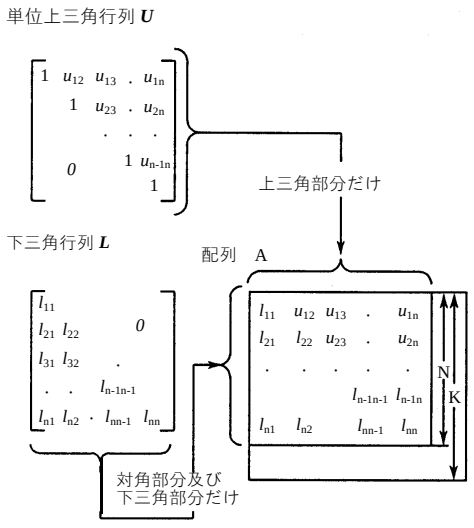
A ……**入力**．行列$\boldsymbol{A}$.
　　出力．行列$\boldsymbol{L}$と行列$\boldsymbol{U}$
　　図VALU-1参照.
　　A(K, N)なる2次元配列.

K ……**入力**．配列Aの整合寸法(≧N).

N ……**入力**．行列$\boldsymbol{A}$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VALU-1　演算後の配列A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VALU－1参照.

表VALU－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$\boldsymbol{A}$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　SSLⅡ…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に値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，選択されたピボット要素が，実行列$\boldsymbol{A}=(a_{ij})$の絶対値最大要素$\max|a_{ij}|$とEPSZの積以下なら

$$\left|a_{kk}^{k}\right| \le \max\left|a_{ij}\right| \mathrm{EPSZ}$$

　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$\boldsymbol{P}$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≧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VLAXを呼び出せば，一度に解が求められる．

c. 使用例
$n\times n$の実行列を入力して，LU分解する．
$n\leqq 100$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 VALU(A,100,N,0.0,IP,IS,VW,ICON)
        WRITE(6,610) ICON
        IF(ICON.GE.20000)GO TO 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

ブロッキングLU分解法は外積形式ガウス消去法をブロック化した方法である．

a. 外積形式ガウス消去法
$n\times n$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．以下の操作で$\boldsymbol{A}=(a_{ij})$を変形することで$\boldsymbol{L}$と$\boldsymbol{U}$に分解する．

$$\boldsymbol{A}^1=\boldsymbol{A}\rightarrow,\ldots,\rightarrow\boldsymbol{A}^k\rightarrow,\ldots,\rightarrow\boldsymbol{A}^n$$

$$u_{kj}=a_{kj}^k/a_{kk}^k,\quad j=k,\ldots,n \tag{4.2}$$

$$l_{ik}=a_{ik}^k,\quad i=k,\ldots,n \tag{4.3}$$

$$a_{ij}^{k+1}=a_{ij}^k-l_{ik}u_{kj},\quad i=k+1,\ldots,n,\quad j=k+1,\ldots,n \tag{4.4}$$

実際には，部分ピボッティングによる行の入換えを行う．

(4.4)は，(4.3)の列ベクトルと(4.2)の行ベクトルの積を作り残りの部分を更新している．

b. ブロッキング法
以上の外積形式ガウスの消去法をブロック化した以下の式で求める．

等ブロック幅blで，行と列方向を分解し，k番目にブロック化した外積ガウス消去を行うとき使う列マトリックス$\boldsymbol{L}_2^k$，行マトリックス$\boldsymbol{U}_2^k$，更新部分を$\boldsymbol{A}^k$とする．各マトリックスの配置については，図VALU-2を参照．

(4.4)に対応する更新を(4.5)で行う．

$$\boldsymbol{A}^k=\boldsymbol{A}^k-\boldsymbol{L}_2^k\boldsymbol{U}_2^k \tag{4.5}$$

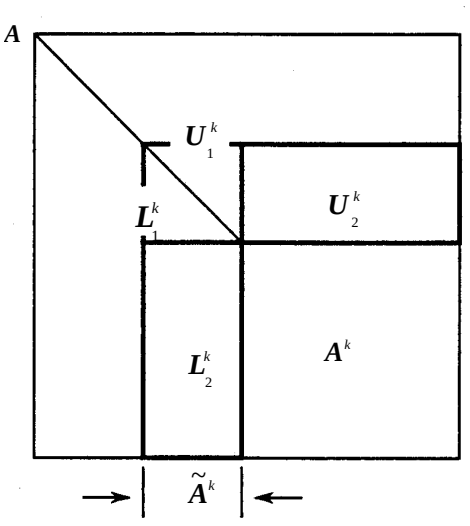
これに先立ち$\boldsymbol{L}_2^k$, $\boldsymbol{U}_2^k$は以下の式で更新する．

まず$\tilde{\boldsymbol{A}}^k$を$\boldsymbol{L}_1^k,\boldsymbol{L}_2^k$と$\boldsymbol{U}_1^k$に分解して，つぎに$\boldsymbol{U}_2^k$を更新する．

$$\tilde{\boldsymbol{A}}^k=\left(\boldsymbol{L}_1^{kt},\boldsymbol{L}_2^{kt}\right)^t\boldsymbol{U}_1^k \tag{4.6}$$

$$\boldsymbol{U}_2^k=\left(\boldsymbol{L}_1^k\right)^{-1}\boldsymbol{U}_2^k \tag{4.7}$$

これらは外積形式ガウス消去法と同じことを順序を変えただけである．



図VALU-2 ブロック化した行列$\boldsymbol{A}$の各要素の配置関係

c. 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(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【9】，【10】，【11】，及び【12】，外積形式ガウスの消去法については，【13】を参照すること.

F16‐15‐0201　　VCFT1, DVCFT1

離散型複素フーリエ変換 (性能優先型，2基底FFT)
CALL VCFT1 (A, B, N, ISN, ISW, VW, IVW, 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，又はその逆変換をベクトル計算機向きの高速変換手法(FFT)により計算する．ただし，$n=2^l$(l：0又は正整数)であること．

a.　フーリエ変換

$\{x_j\}$を入力し，(1.1)で定義する変換を行い，$\{n\alpha_k\}$を求める．

$$n\alpha_k = \sum_{j=0}^{n-1} x_j \cdot \omega^{-jk} \quad , k = 0, 1, \ldots , n-1 \qquad (1.1)$$

$$, \omega = \exp (2\pi i/n)$$

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，$\{x_j\}$を求める．

$$x_j = \sum_{k=0}^{n-1} \alpha_k \cdot \omega^{jk} \quad , j = 0, 1, \ldots , n-1 \qquad (1.2)$$

$$, \omega = \exp (2\pi i/n)$$

(2)　パラメタ

A ……**入力**．$\{x_j\}$又は$\{\alpha_k\}$の実部．
　　出力．$\{n\alpha_k\}$又は$\{x_j\}$の実部．
　　大きさnの1次元配列．

B ……**入力**．$\{x_j\}$又は$\{\alpha_k\}$の虚部．
　　出力．$\{n\alpha_k\}$又は$\{x_j\}$の虚部．
　　大きさnの1次元配列．

N ……**入力**．変数の項数n．

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)

ISW……**入力**．変換の初期設定を制御する情報．
　　初回呼出しのとき:ISW=0
　　継続呼出しのとき:ISW=1
　　(使用上の注意②参照)

VW……**作業領域**．大きさmax(nl,1)の1次元配列．

IVW ……**作業領域**．大きさn max(l-3,2)の1次元配列．

ICON……**出力**．コンディションコード．
　　表VCFT1-1参照．

表VCFT1‐1　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，ISW≠0，1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　SSLⅡ‥UVTB1，UVF9l，UVFA1，UVFB1，UVFX1，UBANK，MGSSL

②　FORTRAN組込み関数…ALOG2，SIN，COS，ATAN，IABS，FLOAT，IAND，MOD

b.　注意

①　サブルーチンの使い分け

本サブルーチンは，複素フーリエ変換をベクトル計算機上で高速処理する．汎用計算機上で処理する場合は，サブルーチンCFT又はCFTMが適している．

本サブルーチンは，独立した複数組の変換を処理する場合に適している．反面，作業用配列を余分に必要とするので，言わば，「性能優先型」のサブルーチンである．作業用配列の確保が困難な場合は，多少性能は低下するが「メモリ節約型」のサブルーチンVCFT2が適している．

②　ISWによる制御

複数組の変換を行う場合は，2回目以降の呼出しのときISW=1と指定すると，変換に必要な三角関数表やリストベクトルなどの生成を省略するので，効率良く処理される．このとき，配列VW，IVWの内容は変更せず呼び出すこと．

複数組の変換の項数nが各々異なる場合でも，ISW=1の指定は有効である．しかし，可能な限り同一の項数の変換が連続するように呼び出すことが望ましい．

実フーリエ変換のサブルーチンVRFT1と組み合わせて本サブルーチンを呼び出す場合でも，ISW=1の指定は有効である．

③　ISNの与え方

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$\{\alpha_k\}$の実部，虚部が，共に間隔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，計算結果も間隔*I*で格納される．ただし*I*>1の場合は，作業領域VWの大きさは*n*(*l*+2)とすること．

ベクトル計算機で実行する場合，メモリアクセスを効率良くするために，間隔*I*は次のような値であることが望ましい．使用例②参照．

単精度演算(VCFT1)　：$I = 4P+2$,
$P = 0, 1, 2,\cdots$

倍精度演算(DVCFT1)：$I = 2P+1$
$P = 1, 2, 3,\cdots$

④　作業用配列の大きさ早見表

16≦*n*≦4096の場合の大きさを以下に示す．

l	*n*	VW	IVW
4	16	64(96)	32
5	32	160(224)	64
6	64	384(512)	192
7	128	896(1152)	512
8	256	2048(2560)	1280
9	512	4608(5632)	3072
10	1024	10240(12288)	7168
11	2048	22528(26624)	16384
12	4096	49152(57344)	36864

( )内は，ABS(ISN)>1のときの大きさ

⑤　一般的なフーリエ変換の定義

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．

$$\alpha_k = \frac{1}{n}\cdot\sum_{j=0}^{n-1} x_j \cdot \omega^{-jk}, \quad k = 0,1,...,n-1 \quad (3.1)$$

$$x_j = \sum_{k=0}^{n-1} \alpha_k \cdot \omega^{jk} \quad , \quad j = 0,1,...,n-1 \quad (3.2)$$

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$\{n\alpha_k\}$又は$\{x_j\}$を求める．したがって，結果の正規化は必要に応じて行うこと．

c.　使用例

①　複数組のフーリエ変換の場合

*k*組の独立したフーリエ変換(*n*項)を求める．*k*≦64，*n*≦512の場合．

```
C      **EXAMPLE**
       DIMENSION A(512,64),B(512,64),
      *          VW(4680),IVW(3072)
       READ(5,500) N,K
       READ(5,510) ((A(I,J),B(I,J),I=1,N),
      *            J=1,K)
C
       ISN=1
       ISW=0
       CALL VCFT1(A,B,N,ISN,ISW,VW,IVW,
      *           ICON)
       IF(ICON.NE.0) STOP
       ISW=1
       DO 10 J=2,K
       CALL VCFT1(A(1,J),B(1,J),N,ISN,ISW,
      *           VW,IVW,ICON)
    10 CONTINUE
C
       WRITE(6,600) K,N
       DO 20 J=1,K
    20 WRITE(6,610) J,(I,A(I,J),B(I,J),
      *             I=1,N)
C
   500 FORMAT(2I5)
   510 FORMAT(2E15.7)
   600 FORMAT(5X,'***',I3,' SET TRANSFORMS'
      *       ' OF',' TERM',I4)
   610 FORMAT(8X,I3,'-TH TRANSFORM'/
      *       (8X,I3,2E16.7))
       STOP
       END
```

②　多次元のフーリエ変換の場合

2次元のフーリエ変換(*n*1×*n*2項)を求める．*n*1≦512，*n*2≦64の場合．

プログラム中，行方向の変換を行うときのデータの間隔(配列宣言の第一寸法宣言子)は，ベクトル計算機上での性能を考慮し，ISN=514(=4p+2, p=128)としている．

```
C      **EXAMPLE**
       DIMENSION A(514,64),B(514,64),
      *          VW(4608),IVW(3072)
       READ(5,500) N1,N2
       READ(5,510) ((A(I,J),B(I,J),I=1,N1)
      *                           ,J=1,N2)
C      ---N2 SET TRANSFORMS OF TERM N1---
       ISN=1
       ISW=0
       CALL VCFT1(A,B,N1,ISN,ISW,VW,IVW,
      *           ICON)
       IF(ICON.NE.0) STOP
       ISW=1
       DO 10 J=2,N2
       CALL VCFT1(A(1,J),B(1,J),N1,ISN,ISW,
      *           VW,IVW,ICON)
    10 CONTINUE
C      ---N1 SET TRANSFORMS OF TERM N2---
       ISN=514
       CALL VCFT1(A,B,N2,ISN,ISW,VW,IVW,
      *           ICON)
       IF(ICON.NE.0) STOP
       DO 20 I=2,N1
       CALL VCFT1(A(I,1),B(I,1),N2,ISN,ISW,
      *           VW,IVW,ICON)
    20 CONTINUE
C
       WRITE(6,600) N1,N2
       DO 30 J=1,N2
    30 WRITE(6,610) J,(I,A(I,J),B(I,J),
      *             I=1,N1)
C
   500 FORMAT(2I5)
   510 FORMAT(2E15.7)
   600 FORMAT(5X,'*** 2 DIMENSIONAL '
      *       'TRANSFORM OF TERM',I4,
      *       ' BY ',I4)
   610 FORMAT(8X,I3,'-TH COLUMN'//
      *       (8X,I3,2E16.7))
       STOP
       END
```

(4)　手法概要

離散型複素フーリエ変換を，ベクトル計算機向きの高速変換手法(Isogeometric型及びSelf-sorting型のFFT)により計算する．

本サブルーチンでは，ベクトル計算機の特性を考慮し，単精度演算ルーチンではIsogeometric型を，倍精度演算ルーチンではSelf-sorting型を採用している．

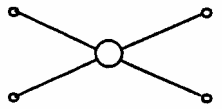
一般に，FFTの算法には，計算を行う領域の扱いに応じて2種類の方法がある．一つはデータを入力した領域だけを用いて計算するIn-place型である．もう一つはデータを入力した領域と作業用の領域を用いて計算するNot-in-place型である．汎用計算機向きのFFTは，通常In-place型に属する．本サブルーチンで採用するFFTは，共にNot-in-place型に属する．Not-in-place型のFFTの算法は，内在する並列性を十分に活かすことができ，ベクトル計算機に適したものといえる．

FFTの算法は，バタフライ演算が計算の核である．バタフライ演算は，任意の二つの入力データをa，b，二つの出力データをc，dとすると，(4.1)で定義する．

$$\begin{aligned} c &= a + b, \\ d &= (a - b)\times\omega^{\xi} \end{aligned} \qquad (4.1)$$

ここで，データa，b，c，d及びω^{ξ}は複素数，又，ω^{ξ}はフーリエ変換固有の係数(回転因子)である．

いま，(4.1)の演算に，入力と出力のデータの位置を考慮し，次のような表記法を導入する．



(4.2)

(4.2)において，◦はデータの位置を表す．左側の2点が入力(上段:a，下段:b)，右側の2点が出力(上段:c，下段:d)である．〇はバタフライ演算を表し，〇内に数字を記入した場合はξを表す．

この表記法を用いて，Isogeometric型，及びSelf-sorting型におけるバタフライ演算の処理を図VCFT1-1及び図VCFT1-2に示す(n=16の場合)．一般に，$n=2^l$とすると，FFTはl回のバタフライステージから構成される．図の例では$n=16=2^4$であるから4回のステージで構成されている．両方法の演算量は同じであるが，各バタフライステージでの，データの転送パターンが異なっている．両方法の特徴とベクトル計算機への適合性は次のとおりである．

Isogeometric型FFT

この方法では，全ステージを通じて入力(並びに出力)の転送パターンがすべて同一である．算法の並列性は最も高く，プログラムも並列性を完全に記述できる．しかし，バタフライ演算を終了した時点で，データの並びがビット反転の意味で逆順になるため，並べ替えの処理が必要である．更に，倍精度演算では，ベクトル計算機の特性でメモリ競合が生じる．

Self-sorting型FFT

この方法では，全ステージを通じて入力の転送パターンはすべて同じであるが，出力の転送パターンがステージごとに規則的に変化している．算法の並列性は，Isogeometric型と同等である．プログラムは，リストベクトルを用いれば並列性を完全に記述できる．しかし，単精度演算では，ベクトル計算機の特性でメモリ競合が生じる．

本サブルーチンでは，両方法の特徴とベクトル計算への適合性を考慮し，高速化を図っている．

本サブルーチンにおける計算手順

【単精度演算ルーチン】

① 三角関数表(回転因子)の作成
　各ステージで必要とする，すべての関数値を計算し，作業用配列VWに格納する．

② リストベクトルの作成
　バタフライ演算の終了後，並べ換えの処理で必要とするリストベクトルを計算し，作業用配列IVWに格納する．

③ バタフライ演算

④ データの並べ換え
　①，②は，本ルーチンが初回目に呼び出されたとき，すなわち，ISW=0のときだけ計算する．

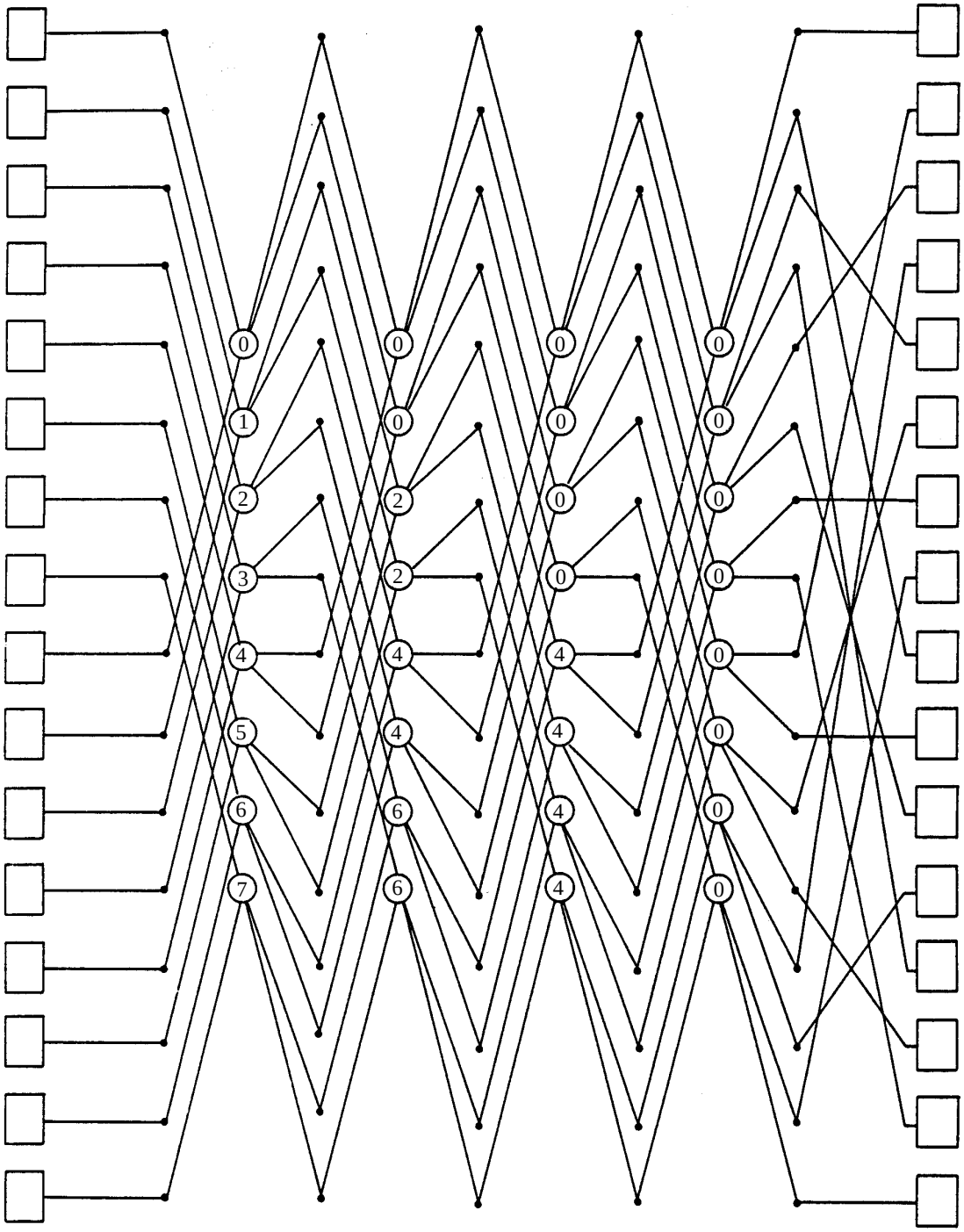
【倍精度演算ルーチン】

① 三角関数表(回転因子)の作成
　各ステージで必要とする，すべての関数値を計し，作業用配列VWに格納する．

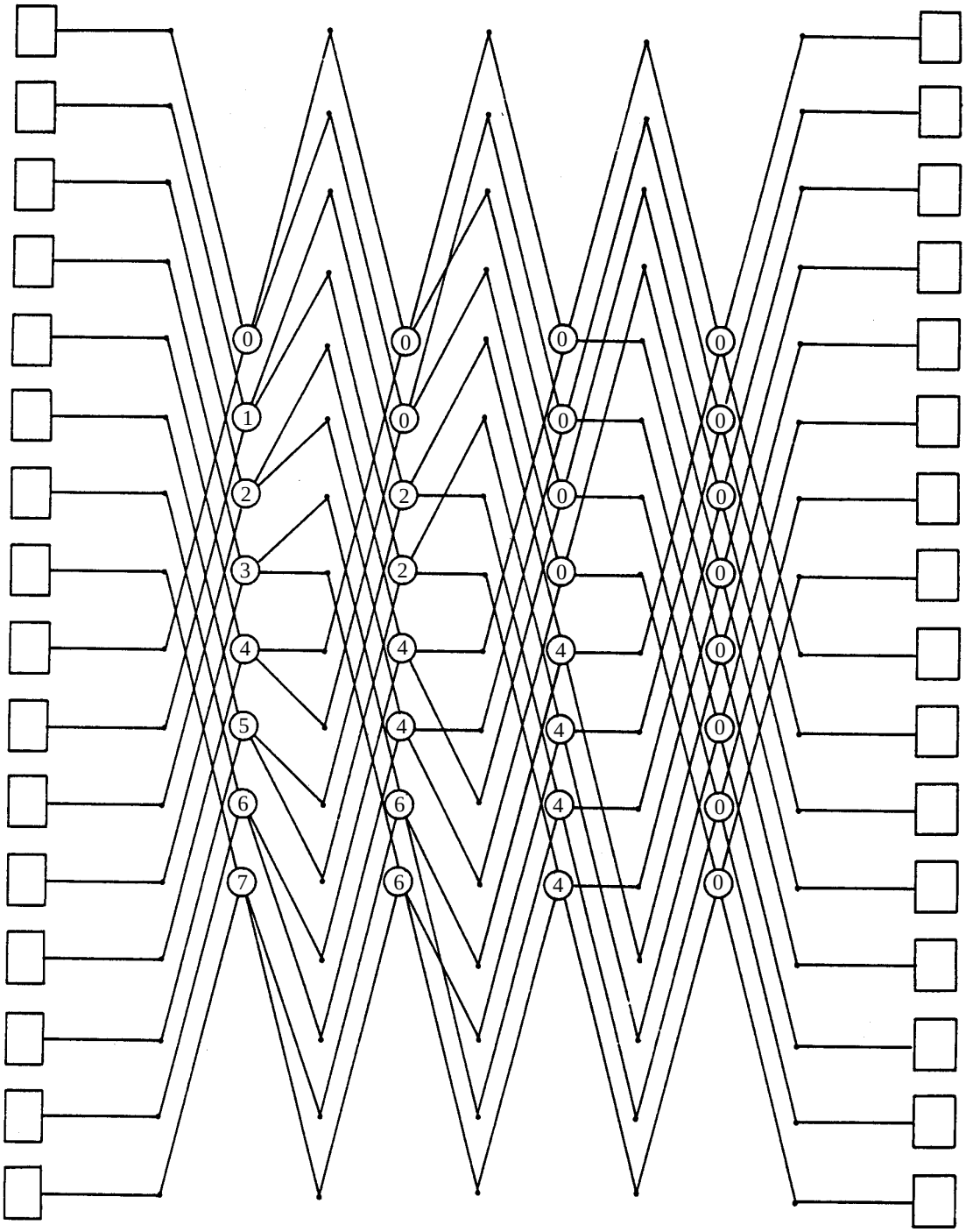
② リストベクトルの作成
　各ステージで必要とする，すべてのリストベクトルを計算し，作業用配列IVWに格納する．

③ バタフライ演算
　①，②は，本ルーチンが初回目に呼び出されたとき，すなわち，ISW=0のときだけ計算する．

なお，ベクトル計算機における様々のFFTについては，参考文献【5】，Isogeometric型のFFTについては，参考文献【4】，Self-sorting型のFFTについては，参考文献【2】，【6】を参照されたい.



図VCFT1-1　Isogeometric型FFTの流れ図(N=16)



図VCFT1-2　Self-sorting型FFTの流れ図(N=16)

F16‐15‐0301　　VCFT2, DVCFT2

離散型複素フーリエ変換 (メモリ節約型，2基底FFT)
CALL VCFT2(A, B, N, ISN, ISW, VW, IVW, 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，又はその逆変換をベクトル計算機向きの高速変換手法(FFT)により計算する．

ただし，$n=2^l$(l:0又は正整数)であること．

a.　フーリエ変換

$\{x_j\}$を入力し，(1.1)で定義する変換を行い，$\{n\alpha_k\}$を求める．

$$n\alpha_k = \sum_{j=0}^{n-1} x_i \cdot \omega^{-jk} \quad , k = 0, 1, \ldots , n-1 \quad (1.1)$$

$$, \omega = \exp(2\pi i/n)$$

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，$\{x_j\}$を求める．

$$x_j = \sum_{k=0}^{n-1} \alpha_k \cdot \omega^{jk} \quad , k = 0, 1, \ldots , n-1 \quad (1.2)$$

$$, \omega = \exp(2\pi i/n)$$

(2)　パラメタ

A ……**入力**．$\{x_i\}$又は$\{\alpha_k\}$の実部．
　　出力．$\{n\alpha_k\}$又は$\{x_j\}$の実部．
　　大きさnの1次元配列．

B……**入力**$\{x_j\}$肩又は$\{\alpha_k\}$の虚部．
　　出力．$\{n\alpha_k\}$又は$\{x_j\}$の虚部．
　　大きさnの1次元配列．

N ……**入力**．変換の項数n．

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)

ISW……**入力**．変換の初期設定を制御する情報．
　　初回呼出しのとき:ISW = 0
　　継続呼出しのとき:ISW = 1
　　(使用上の注意②参照)

VW……**作業領域**．大きさ5nの1次元配列．

IVW ……**作業領域**．大きさ3nの1次元配列．

ICON……**出力**．コンディションコード．
　　表VCFT2-1参照．

表VCFT2-1　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=0，1SW≠0，1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　SSLⅡ…UVTB2，UVF92，UVFA2，UVFB2，UVFX2，UBANK，MGSSL

②　FORTRAN組込み関数…ALOG2，SIN，COS，IABS，FLOAT，IAND，MOD

b.　注意

①　サブルーチンの使い分け

本サブルーチンは，複素フーリエ変換をベクトル計算機上で高速処理する．汎用計算機上で処理する場合は，サブルーチンCFT又はCFTMが適している．

本サブルーチンは，単独の変換を行う場合に適している．作業用配列は，最小限におさえてあり，言わば，「メモリ節約型」のサブルーチンである．複数組の変換を行う場合で，作業用配列を充分に確保できるならば，「性能優先型」のサブルーチンVCFT1が適している．

②　ISWによる制御

複数組の変換を行う場合，2回目以降の呼出しのときISW=1と指定すると，変換に必要な三角関数表などの生成を省略するので，多少効率良く処理される．このとき，配列VW，IVWの内容は変更せず呼び出すこと．

複数組の変換の項数nが各々異なる場合でも，ISW=lの指定は有効である．しかし，可能な限り同一の項数の変換が連続するように呼び出すことが望ましい．

実フーリエ変換のサブルーチンVRFT2と組み合わせて本サブルーチンを呼び出す場合でもISW=1の指定は有効である．

③　ISNの与え方

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$\{\alpha_k\}$の実部，虚部が，共に間隔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，計算結果も間隔Iで格納される．ただしI>1の場合は，作業領域VWの大きさは7nとすること．
ベクトル計算機で実行する場合，メモリアクセスを効率良くするために，間隔Iは次のような値であることが望ましい．使用例②参照．
単精度演算(VCFT2) ：I = 4P+2,
P = 0,1,2,…
倍精度演算(DVCFT2)：J = 2P+1,
P = 1,2,3,…

④ 作業用配列の大きさ早見表
16≦n≦4096の場合の大きさを以下に示す．

l	n	VW	IVW
4	16	80(112)	48
5	32	160(224)	96
6	64	320(448)	192
7	128	640(896)	384
8	256	1280(1792)	768
9	512	2560(3584)	1536
10	1024	5120(7168)	3072
11	2048	10240(14336)	6144
12	4096	20480(28672)	12288

( )内は，ABS(ISN)>1のときの大きさ

⑤ 一般的なフーリエ変換の定義
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．

$$\alpha_k = \frac{1}{n}\cdot\sum_{j=0}^{n-1} x_j\cdot\omega^{-jk}\ ,\ k = 0, 1, \ldots, n-1 \quad (3.1)$$

$$x_j = \sum_{k=0}^{n-1}\alpha_k\cdot\omega^{jk}\qquad ,\ j = 0, 1, \ldots, n-1 \quad (3.2)$$

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，{$n\alpha_k$}又は{x_j}を求める．したがって，結果の正視化は必要に応じて行うこと．

c. 使用例
1次元のフーリエ変換(n項)及びその逆変換を 連続して計算する．n≦1024の場合．

```
C      **EXAMPLE**
       DIMENSION A(1024),B(1024),VW(5120),
      *          IVW(3072)
       READ(5,500) N
       READ(5,510) (A(I),B(I),I=1,N)
C      ---FORWARD TRANSFORM---
       ISN=1
       ISW=0
       CALL VCFT2(A,B,N,ISN,ISW,VW,IVW,
      *     ICON)
       IF(ICON.NE.0) STOP
C      ---NORMALIZATION---
       ANOR=1.0/FLOAT(N)
       DO 10 I=1,N
       A(I)=ANOR*A(I)
    10 B(I)=ANOR*B(I)
       WRITE(6,600) N,(I,A(I),B(I),I=1,N)
C      ---BACKWARD TRANSFORM---
       ISN=-1
       ISW=1
       CALL VCFT2(A,B,N,ISN,ISW,VW,IVW,
      *           ICON)
       IF(ICON.NE.0)STOP
C
       WRITE(6,610) N,(I,A(I),B(I),I=1,N)
C
  500 FORMAT(I5)
  510 FORMAT(2E15.7)
  600 FORMAT(5X,
     *  '*** FORWARD TRANSFORM OF TERM',
     *  I5//(8X,I3,2E16.7))
  610 FORMAT(5X,
     *  '*** BACKWARD TRANSFORM OF TERM',
     *  I5//(8X,I3,2E16.7))
       STOP
       END
```

(4) 手法概要

離散型複素フーリエ変換を，ベクトル計算機向きの高速変換手法(Isogeometric型及びSelf-sorting型のFFT)により計算する．

本サブルーチンでは，ベクトル計算機の特性を考慮し，単精度演算ルーチンではIsogeometric型を，倍精度演算ルーチンではSelf-sorting型を採用している．

算法については，サブルーチンVCFT1の「手法概要」を参照されたい．

本サブルーチンにおける計算手順

【単精度演算ルーチン】
① 三角関数表(回転因子)の作成
初回のステージで必要とする関数値などを計算し，作業用配列VW，IVWに格納する．
② バタフライ演算
③ データの並べ換え
①は，本ルーチンが初回目に呼び出されたとき，すなわち，ISW=0のときだけ計算する．

【倍精度演算ルーチン】
① 三角関数表(回転因子)の作成
初回のステージで必要とする関数値などを計算し，作業用配列VW，IVWに格納する．
② バタフライ演算
①は，本ルーチンが初回目に呼び出されたとき，すなわち，ISW=0のときだけ計算する．

F16－11－0201　　VCOS1, DVCOS1

離散型cosine変換(2基底FFT)
CALL VCOS1(A, N, TAB, VW, IVW, 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$\{x_j\}$

$$x_j = x(\theta j) \quad , \quad j = 0, 1, \ldots , n$$

$$, \quad \theta = \pi / n \qquad (1.1)$$

が与えられたとき，離散型cosine変換，又はその逆変換をベクトル計算機向きの高速変換手法(FFT)により計算する．

ただし，$n=2^l$(l:0又は，正整数)であること．

a.　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$\{2n \cdot a_k\}$を求める．

$$2n \cdot a_k = 4 \cdot \sum_{j=0}^{n}{}'' x_j \cdot \cos kj\theta \ , k = 0, 1,..., n$$

$$, \theta = \pi / n \qquad (1.2)$$

ここで$\sum''$は，初項と末項を1/2倍して和をとることを意味する．

b.　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$\{4 \cdot x_j\}$を求める．

$$4 \cdot x_j = 4 \sum_{k=0}^{n}{}'' a_k \cdot \cos kj\theta \ , j = 0, 1,..., n$$

$$, \theta = \pi / n \qquad (1.3)$$

(2)　パラメタ

A ……**入力**．$\{x_j\}$又は$\{a_k\}$
　　出力．$\{2n \cdot a_k\}$又は$\{4 \cdot x_j\}$．
　　大きさn+2の1次元配列．
　　図VCOS1-1参照．

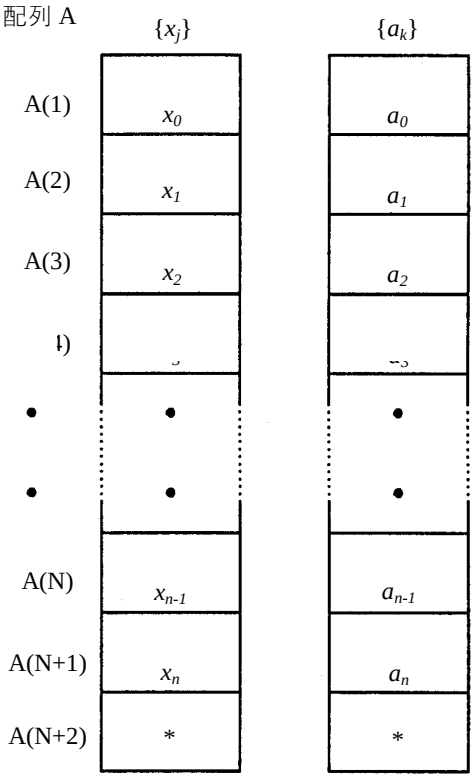
N ……**入力**．標本の数-1．

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2n+4の1次元配列．

VW……**作業領域**．
　　大きさmax(n(l+1)/2,1)の1次元配列．

IVW ……**作業領域**．
　　大きさn·max(l-4,2)/2の1次元配列．

ICON……**出力**．コンディションコード．
　　表VCOS1-1参照．



[**注意**]　2{na_k}，{4x_j}についても同様．
　　*は，入力時は任意，出力時は0.0がセットされる．

図VCOS1-1　データの格納方法

表VCOS1-1　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: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　SSLII…VRFT1，VCFT1，UVRFT，UVTB1，UVF91，UVFA1，UVFB1，UVFX1，UBANK，UVTAB，MGSSL

②　FORTRAN組込み関数…ALOG2，SIN，COS，ATAN，IABS，IAND，FLOAT

b.　注意

①　サブルーチンの使い分け

本サブルーチンは，離散型cosine変換をベクトル計算機上で高速処理する．汎用計算機上で処理する場合は，サブルーチンFCOSTが適している．

② 複数組の変換について

複数組の変換を行う場合，2回目以降の呼出しの ときは，変換に必要な三角関数表やリストベクトルなどの生成を省略するので，効率良く処理される．このとき，配列TAB, VW，IVWの内容は変更せず呼び出すこと．

複数組の変換の項数nが各々異なる場合でも，それまでに生成した，配列TAB，VW，IVWの内容は有効である．しかし，可能な限り同一の項数の変換が連続するように呼び出すことが望ましい．

③ 三角関数表及び作業用配列の大きさ早見表

16≦n≦4096の場合の大きさを以下に示す．

l	n	TAB	VW	IVW
4	16	36	40	16
5	32	68	96	32
6	64	132	224	64
7	128	260	512	192
8	256	516	1152	512
9	512	1028	2560	1280
10	1024	2052	5632	3072
11	2048	4100	12288	7168
12	4096	8196	26624	16384

④ 一般的な離散型cosine変換について

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．

$$a_k = \frac{2}{n}\sum_{j=0}^{n}{}''x_j \cdot \cos kj\theta \ , k = 0, 1, ..., n \quad (3.1)$$

$$x_j = \sum_{k=0}^{n}{}''a_k \cdot \cos kj\theta \quad , j = 0, 1, ..., n \quad (3.2)$$

本サブルーチンでは，(3.1)，(3.2)の左辺に対応して{2n·a_k}又は，{4·x_j}を求める．したがって，結果の正規化は必要に応じて行うこと．

c. 使用例

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≦512の場合．

```
C      **EXAMPLE**
       DIMENSION X(514),TAB(1028),VW(2560)
      *          ,IVW(1280)
     1 READ(5,500) N
       IF(N.EQ.0) STOP
       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 VCOS1(X,N,TAB,VW,IVW,ICON)
       IF(ICON.NE.0) GO TO 30
C      NORMALIZE
       CN=1.0/(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 TRANSRORM
       CALL VCOS1(X,N,TAB,VW,IVW,ICON)
       IF(ICON.NE.0) GO TO 30
C      NORMALIZE
       CN=0.25
       DO 20 K=1,NP1
       X(K)=X(K)*CN
    20 CONTINUE
       WRITE(6,602)
       WRITE(6,601) (X(I),I=1,NP1)
       GO TO 1
    30 WRITE(6,603) ICON
       GO TO 1
   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, ...)の離散型cosine変換をベクトル計算機向きの高速変換手法(FFT)により計算することを考える．

標本{x_j}，j=0, 1, ... , nが与えられたときの離散cosine変換は，一般的に(4.1)で表される．

$$a_j = \frac{1}{n}x_0 + \frac{2}{n}\sum_{k=1}^{n-1} x_k \cdot \cos(kj\theta) + \frac{1}{n}(-1)^j x_n$$
$$, j = 0, 1, ..., n, \ \theta = \pi / n \quad (4.1)$$

ところで，標本は偶関数であり，1周期に拡大すると

$$x_{2n-j} = x_j, \ j = 0, 1, ..., n \quad (4.2)$$

なる関係がある．したがって，x_0～x_nをx_0～x_{2n-1}に拡張して，項数2nの離散型実フーリエ変換を行えば，a_0～a_nを求めることができる．しかし，この場合(4.2)の対称性を利用すれば，効率よく変換が可能であることが知られている．

いま，標本{x_j}に，以下のような前処理を施す．

$$d_j = \frac{1}{2}\cdot(x_j + x_{n-j}) - \sin(j\theta)\cdot(x_j - x_{n-j})$$
$$, j = 0, 1, ..., n-1 \quad (4.3)$$

ここで，(4.3)に離散型cosine逆変換(4.4)を代入すると，(4.5)を得る．

$$x_j = \frac{1}{2}a_0 + \sum_{k=1}^{n-1} a_k \cdot \cos kj\theta + \frac{1}{2}(-1)^j a_n$$
$$, j = 0, 1, ..., n-1 \quad (4.4)$$

$$d_j = \frac{1}{2}a_0 + \sum_{k=1}^{n/2-1}\left[a_{2k}\cdot\cos(2\cdot kj\theta) + (a_{2k+1} - a_{2k-1})\cdot\sin(2\cdot kj\theta)\right] + \frac{1}{2}a_n\cdot(-1)^j$$
$$, j = 0, 1, ..., n-1 \quad (4.5)$$

(4.5)は，標本を{d_j}，フーリエ係数を{a_{2k}}及び{a_{2k+1}-a_{2k-1}}とするn項の離散型実フーリエ変換と等

価である．そこで，標本$\{d_j\}$に対するフーリエ係数$\{\tilde{a}_k\}$及び$\{\tilde{b}_k\}$を求めれば，恒等式

$$\tilde{a}_k = a_{2k}$$
$$\tilde{b}_k = a_{2k+1} - a_{2k-1}$$

より，$\{a_k\}$を得ることができる．
すなわち，$\{a_k\}$は(4.6)により求められる．

$$a_1 = 1/n \cdot x_0 + \frac{2}{n}\sum_{j=1}^{n-1} x_j \cdot \cos(j\theta) - 1/n \cdot x_n,$$
$$a_{2k} = \tilde{a}_k,$$
$$a_{2k+1} = a_{2k-1} + \tilde{b}_k \quad , k = 1, \ldots, n/2 - 1 \qquad (4.6)$$

ところで，(4.6)の最後の式は漸化式であり，ベクトル計算機には不向きである．そこで，本サブルーチンでは，離散型cosine変換とその逆変換とが正規化定数を除くと同一の式であることを利用して，以上の計算式を逆にたどることにより漸化式の計算をなくし，ベクトル計算機に適したアルゴリズムを実現している．

なお，本アルゴリズムの詳細は，参考文献【8】を参照されたい．

B62－21－0201　　VGSG2, DVGSG2

実行対称行列の一般固有ベクトル (並列バイセクション法，逆反復法)
CALL VGSG2(A, B, N, M, EPSZ, EPST, E, EV, K, VW, IVW, 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$\boldsymbol{B}$に対する一般固有値問題

$$\boldsymbol{Ax} = \lambda\boldsymbol{Bx} \qquad (1.1)$$

の固有値を，並列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個求める．また，対応するm個の固有ベクトル$x_1, x_2, \ldots, x_m$を逆反復法により求める．固有ベクトルは，

$$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{BX} = \boldsymbol{I} \qquad (1.2)$$

を満たす．ここで，$\boldsymbol{X} = [x_1, x_2, \ldots, x_m]$である．
ただし，$1 \leqq m \leqq n$であること．

(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$n(n+1)/2$の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$n(n+1)/2$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$\boldsymbol{A}$，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**．$\boldsymbol{B}$の$\boldsymbol{LL}^{\mathrm{T}}$分解におけるピボットの相対零判定値．零又は負の値を指定すると標準値が採用される．
(使用上の注意②参照)

EPST……**入力**．固有値の収束判定に使われる絶対誤差の上限．負の値を指定すると標準値が採用される．
(使用上の注意③参照)

E……**出力**．固有値．
大きさmの1次元配列．
Mが正の場合大きい順に，Mが負の場合小さい順に格納される．

EV……**出力**．固有ベクトル．
EV(K, m)なる2次元配列．
固有値E(J)に対応する固有ベクトルはEV(I, J)，I=1, ... , Nに格納される．

K……**入力**．配列EVの整合寸法(≧n)．

VW……**作業領域**．大きさ5nの1次元配列．

IVW……**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VGSG2-1参照．

表VGSG2-1　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), EV(1,1)=1.0/SQRT(B(1))とする．
15000	固有ベクトルの計算で求まらないものがあった．	求まらない固有ベクトルを零ベクトルとする．
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零ベクトルとする．
28000	$\boldsymbol{B}$の$\boldsymbol{LL}^{\mathrm{T}}$分解においてピボットが負になった． $\boldsymbol{B}$が正値でない．	処理を打ち切る．
29000	$\boldsymbol{B}$の$\boldsymbol{LL}^{\mathrm{T}}$解においてピボットが相対的に零となった． $\boldsymbol{B}$は非正則の可能性が強い．	処理を打ち切る．
30000	M=0，N<|M|は又は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…GSCHL，TRID1，TRBK，UVBCT，GSBK，UVTG2，UCHLS，AMACH，MGSSL

②　FORTRAN組込み関数…IABS，SQRT，SIGN, ABS，AMAX1

b.　注意

①　本サブルーチンは，機能的にはサブルーチンGSEG2と同等であるが，並列バイセクション法を採用することによりベクトル計算機上での高速化を図ったのである．両サブルーチンの間で作業領域の確保仕方が異なることに注意すること．

②　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$\boldsymbol{LL}^{\mathrm{T}}$分解の過程でピボットの値が10進sけた以上のけた落ちを生じた場合に，そのピボットを零とみ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は，EPSZに非常に小さな値を与えればよいが，結果の精度は保証されない．

一方，分解の過程でピボットの値が負となった場合は，行列Bが正値ではないとみなし，コンディションコードを設定し(ICON=28000) 処理を打ち切る．

③ パラメタEPSTの標準値は，丸め誤差の単位をuとしたとき

$$EPST = u \cdot \max(|\lambda_{max}|, |\lambda_{min}|) \qquad (3.1)$$

である．ここで，λ_{max}及びλ_{min}は，$\boldsymbol{Ax}=\lambda\boldsymbol{Bx}$の固有値の存在範囲(ゲルシュゴリンの定理により与えられる)の上限と下限である．

固有値の絶対値が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，(3.1)により収束判定すると，一般に小さい方の固有値を精度よく求めることは困難である．このような場合は，EPSTに小さな値(絶対誤差)を設定することにより，小さい方の固有値も精度よく求めることができる．ただし，反復回数が増加するため処理速度は遅くなる．

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$\boldsymbol{B}$に対する一般固有値問題$\boldsymbol{Ax}=\lambda\boldsymbol{Bx}$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≦100，m≦20の場合

```
C     **EXAMPLE**
      DIMENSION A(5050),B(5050),E(20),
     *          EV(102,20),VW(1500),
     *          IVW(700)
   10 READ(5,500,END=900) N,M,EPSZ,EPST
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      READ(5,510) (B(I),I=1,NT)
      WRITE(6,600) N,M,EPSZ,EPST
      WRITE(6,610)
      IJ=0
      DO 20 I=1,N
      IJ=IJ+I
   20 WRITE(6,620) I,(A(J),J=IJ-I+1,IJ)
      WRITE(6,630)
      IJ=0
      DO 30 I=1,N
      IJ=IJ+I
   30 WRITE(6,620) I,(B(J),J=IJ-I+1,IJ)
      CALL VGSG2(A,B,N,M,EPSZ,EPST,
     *           E,EV,102,VW,IVW,ICON)
      WRITE(6,640) ICON
      IF(ICON.GE.20000) GO TO 10
      MM=IABS(M)
      CALL SEPRT(E,EV,102,N,MM)
      GO TO 10
  900 STOP
  500 FORMAT(2I5,2E10.2)
  510 FORMAT(5E15.7)
  600 FORMAT('1'//' ***     N=',I5
     *          /' ***     M=',I5
     *          /' ***  EPSZ=',E15.7
     *          /' ***  EPST=',E15.7)
  610 FORMAT('0'//' *** INPUT MATRIX A'/)
  620 FORMAT('0',2X,I3,5E15.7/(6X,5E15.7))
  630 FORMAT('0'//' *** INPUT MATRIX B'/)
  640 FORMAT('0'//' ***  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印刷するサブルーチンである．その詳細は，サブルーチンVSEG2の使用例を参照されたい．

(4) 手法概要

n次の実対称行列$\boldsymbol{A}$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

$$\boldsymbol{Ax} = \lambda\boldsymbol{Bx} \qquad (4.1)$$

の固有値固有ベクトルを次の手順で求める．

a. 一般固有値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，

$$\boldsymbol{B} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}} \qquad (4.2)$$

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．(4.1)に(4.2)を代入し整理すると，

$$\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}(\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}) = \lambda(\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}) \qquad (4.3)$$

となる．ここで，

$$\boldsymbol{S} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}} \qquad (4.4)$$

$$\boldsymbol{y} = \boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} \qquad (4.5)$$

とおくと，$\boldsymbol{S}$は実対称行列，(4.3)は

$$\boldsymbol{S}\boldsymbol{y} = \lambda\boldsymbol{y} \qquad (4.6)$$

となり，標準形が導かれる．

b. 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3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$\boldsymbol{y}'$をそれぞれ並列バイセクション法と逆反復法により求める．$\boldsymbol{y}'$は更にSの固有ベクトル$\boldsymbol{y}$へ逆変換される．

c. 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，b.により求めた固有ベクトル$\boldsymbol{y}$から，

$$\boldsymbol{x} = \boldsymbol{L}^{\mathrm{T}}\boldsymbol{y} \qquad (4.7)$$

により得られる．

a.はGSCHL，b.はVSEG2の内部ルーチン，c.はGSBKを用いて計算する．

A22－71－0101　　VLAX, DVLAX

実行列の連立1次方程式 (ブロッキングLU分解法)
CALL VLAX(A, K, N, B, EPSZ, ISW, IS, VW, IP, ICON)

(1)　機能

実係数連立1次方程式

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

をブロッキングLU分解法(ガウスの消去法)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，xはn次元の解ベクトルである．$n\geqq 1$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(≧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(≧l)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.
1SW=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VLAX－1参照.

表VLAX－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…VALU，AMACH，LUX，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位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，選択されたピボット要素が，実行列$\boldsymbol{A}=(a_{ij})$の絶対値最大要素$\max|a_{ij}|$とEPSZの積以下なら，

$$|a_{kk}^{k}|\leq \max|a_{ij}|\,\mathrm{ESPZ}$$

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，$\mathrm{EPSZ}=16\cdot u$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のときの値が保障される.

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*≦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 VLAX(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
       GO TO 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,')',E15.8)))
   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(ブロッキングLU分解法)

ブロッキングLU分解法により，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，$\boldsymbol{P}$はピボッテイ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VALU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，後進代入によって解く.この計算はサブルーチンLUXを用いて行う.

A22－61－0302　　VLDLX, DVLDLX

LDL^{T}分解された正値対称行列の連立1次方程式
CALL VLDLX(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

$$\boldsymbol{LDL}^T\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

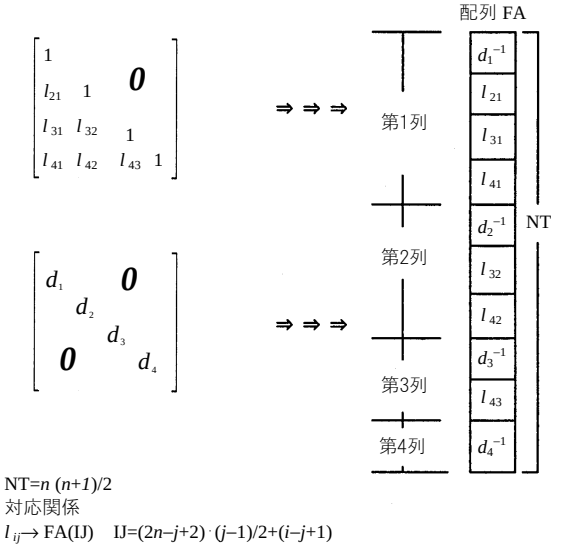
を解く．ただし，$\boldsymbol{L}$，$\boldsymbol{D}$はそれぞれ$n\times n$の単位下三角行列と対角行列，$\boldsymbol{b}$はn次元の実定数ベクトル，$\boldsymbol{x}$はn次元の解ベクトルである．$n\geqq 1$であること．

本サブルーチンは，サブルーチンVSLDLにより LDL^{T}分解された行列を受けて求解処理を行うものである．

(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$\boldsymbol{L}$及び$\boldsymbol{D}^{-1}$
　　大きさ$n(n+1)/2$の1次元配列.
　　格納の方法は，図VLDLX-1に示すように，$\boldsymbol{L}$を第1列から第n列まで列ごとに入力する．



NT=n (n+1)/2
対応関係
$l_{ij}\rightarrow$ FA(IJ)　IJ=(2n−j+2)·(j−1)/2+(i−j+1)

図VLDLX-1　行列$\boldsymbol{L}$及び$\boldsymbol{D}^{-1}$の格納方法

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VLDLX-1参照.

表VLDLX-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	系列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　SSLⅡ…MGSSL
②　FORTRAN組込み関数…なし

b.　注意
①　本サブルーチンを，サブルーチンVSLDL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VLSXを呼び出せば，一度に解が求められる.

c.　使用例
　正値対称行列をサブルーチンVSLDLで LDL^{T}分解し，続いて本サブルーチンを用いて連立1次方程式を解く．$n\leqq 100$の場合.

```
C      **EXAMPLE**
       DIMENSION A(5050),B(100),VW(200),
      *          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)
      WRITE(6,640)
      IS=1
      IE=N
      DO 20 J=1,N
      WRITE(6,600) J,(A(I),I=IS,IE)
      IS=IE+1
   20 IE=IE+(N-J)
      CALL VSLDL(A,N,1.0E-6,VW,IVW,ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      READ(5,510) (B(I),I=1,N)
      CALL VLDLX(B,A,N,ICON)
      WRITE(6,610) ICON
      DET=1.0
      II=1
      NCOL=N
      DO 30 I=1,N
      DET=DET*A(II)
      II=II+NCOL
   30 NCOL=NCOL-1
      DET=1.0/DET
      WRITE(6,620) (B(I),I=1,N)
      WRITE(6,630) DET
      GO TO 10
```

```
500 FORMAT(I5)
510 FORMAT(5E15.7)
600 FORMAT(' ',I5/(10X,4E16.8))
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

正値対称行列$\boldsymbol{A}$のLDL$^{\mathrm{T}}$分解.

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}} \tag{4.1}$$

が得られているとする．このとき,

$$\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{b} \tag{4.2}$$

を以下の順で解く.

① $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}$を解く(前進代入).
まず，$\boldsymbol{y}$を$\boldsymbol{b}$で初期化する.

$$\boldsymbol{y} \leftarrow \boldsymbol{b} \tag{4.3}$$

次に，$j = 1, 2, \ldots, n-1$に対して(4..4)を繰り返す.

$$y_i \leftarrow y_i - y_j l_{ij}, \quad i = j+1, j+2, \ldots, n \tag{4.4}$$

② $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{D}^{-1}\boldsymbol{y}$を解く(後退代入).
まず，$\boldsymbol{x}$を$\boldsymbol{D}^{-1}\boldsymbol{y}$で初期化する.

$$\boldsymbol{x} \leftarrow \boldsymbol{D}^{-1}\boldsymbol{y} \tag{4.5}$$

次に，$i = n-1, n-2, \ldots, 1$に対して(4.6)を繰り返す.

$$x_i \leftarrow x_i - \sum_{j=i+1}^{n} l_{ji} x_j \tag{4.6}$$

実際の計算では，$\boldsymbol{y}$，$\boldsymbol{x}$は共に配列B上に作るので，上記の代入処理は，すべて配列Bの更新手続きである.

以上の計算は，ベクトル計算機上ですべてベクトル処理できる.

A22‐61‐0101　　VLSX, DVLSX

正値対称行列の連立1次方程式(変形コレスキー法)
CALL VLSX(A, N, B, EPSZ, ISW, VW, IVW, ICON)

(1)　機能

実係数連立1次方程式,

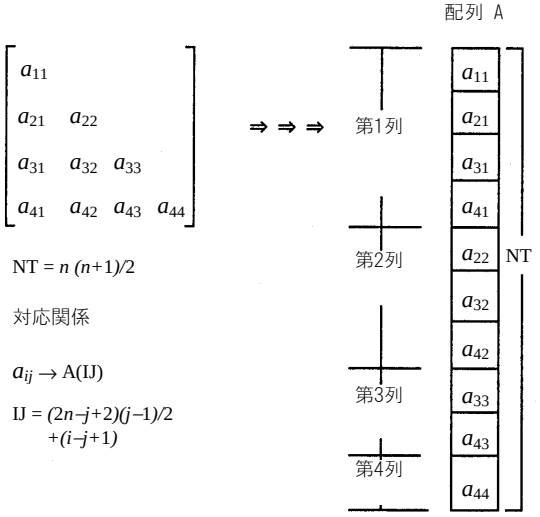
$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

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．$n\geqq 1$であること．

本サブルーチンは，サブルーチンLSXと機能的に同じであるが，係数行列の格納方法が異なり，ベクトル計算機に適した方法を採っている．

(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．
大きさ$n(n+1)/2$の1次元配列.
格納の方法は，図VLSX-1に示すように，対称行列の下三角部分を第1列から第n列まで列ごとに入力する．



図VLSX-1　対称行列の格納方法

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②参照)

ISW ……**入力**．制御情報.
同一の係数行列を持つ複数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用する.
(使用上の注意③参照)

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VLSX-1参照.

表VLSX-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　SSLⅡ‥VSLDL，VLDLX，AMACH，MGSSL

②　FORTRAN組込み関数…ABS

b.　注意

①　本サブルーチンは，サブルーチンLSXにおける行列の格納方法を変えることにより，ベクトル計算機上での高速化を図ったものである．両サブルーチンの間で格納方法，及び呼出し方法が異なることに注意すること.

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分解の過程でピボットの値が50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·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，結果の精度は保証されない．

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LDL^{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，係数行列は正値でないことになる．このとき，ICON=10000とするが処理は続行する．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注意が必要である．

⑤ 係数行列の行列式の値を求めるには，演算後の配列$\boldsymbol{A}$のn個の対角線上の値($\boldsymbol{D}^{-1}$の対角要素)を掛け合わせ，その逆数をとればよい．

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≦100の場合．

```
C     **EXAMPLE**
      DIMENSION A(5050),B(100),VW(200),
     *          IVW(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
      ISW=1
      M=1
      EPSZ=1.0E-6
   10 READ(5,510) (B(I),I=1,N)
      CALL VLSX(A,N,B,EPSZ,ISW,VW,IVW,
     *          ICON)
      WRITE(6,610) ICON
      IF(ICON.GE.20000) STOP
      WRITE(6,620) (B(I),I=1,N)
      IF(L.EQ.M) GO TO 20
      M=M+1
      ISW=2
      GO TO 10
   20 DET=1.0
      II=1
      NCOL=N
      DO 30 I=1,N
      DET=DET*A(II)
      II=II+NCOL
   30 NCOL=NCOL-1
      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. 系列行列$\boldsymbol{A}$のLDL^{T}分解(変形コレスキー法)

変形コレスキー法により，系列行列$\boldsymbol{A}$をLDL^{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VSLDL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VLDLXを用いて行う．

本サブルーチンは，サブルーチンLSXのベクトル版として位置付けられ，ベクトル計算機上での高速化が図られている．計算手順の詳細は，サブルーチンVSLDL及びVLDLXの「手法概要」を参照されたい．

A62‐11‐0101　　VLTX, DVLTX

実3項行列の連立1次方程式 (サイクリック・リダクション法)
CALL VLTX (SBD, D, SPD, N, B, ISW, IND, IVW, ICON)

(1)　機能

実3項方程式,

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

をサイクリック・リダクション法で解く．ただし，$\boldsymbol{A}$は$n\times n$の既約優対角性を満たす実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．

$n\geqq 1$であること．

ここで，$\boldsymbol{A}$が既約優対角であるとは,

$$\boldsymbol{A}=\begin{bmatrix} d_1 & f_1 & & & \mathbf{0} & & \\ e_2 & d_2 & f_2 & & & & \\ & e_3 & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & \\ & \mathbf{0} & & & \cdot & \cdot & f_{n-1} \\ & & & & & e_n & d_n \end{bmatrix} \qquad (1.2)$$

とするとき,

$$|d_i| \geqq |e_i| + |f_i|,\ i = 1, 2, \ldots, n \qquad (1.3)$$

(ただし $e_1 = f_n = 0$)

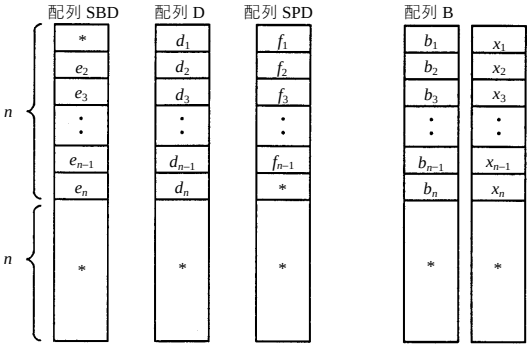
が成り立ち，かつ，少なくとも一つのiに対して狭義の不等号が成立することである．

(2)　パラメタ

SBD ……**入力**．係数行列$\boldsymbol{A}$の下副対角部分.
SBD(i) = e_i, i = 2, 3, ... , nのように格納する.
図VLTX-1参照.
演算後，内容は保存されない.
大きさ$2n$の1次元配列.
（使用上の注意④参照)

D …… **入力**．係数行列$\boldsymbol{A}$の対角部分.
D(i) = d_i, i = 1, 2, ... , nのように格納する.
図VLTX-1参照.
演算後，内容は保存されない.
大きさ$2n$の1次元配列.
(使用上の注意④参照)

SPD……**入力**．係数行列$\boldsymbol{A}$の上副対角部分.
SPD(i) = f_i, i = 1, 2, ... , n-1のように格納する.
図VLTX-1参照.
演算後，内容は保存されない.



[注意] ∗の部分は，本サブルーチン内で作業領域として使用される.

図VLTX-1　行列$\boldsymbol{A}$，並びにベクトル$\boldsymbol{b}$，$\boldsymbol{x}$の格納方法

大きさ$2n$の1次元配列.
(使用上の注意④参照)

N……**入力**．係数行列$\boldsymbol{A}$の次数n.

B……**入力**．定数ベクトル$\boldsymbol{b}$.
B(i)=b_i, i = 1, 2, ... , nのように格納する.
出力．解ベクトル$\boldsymbol{x}$.
B(i)=x_i, i = 1, 2, ... , nのように格納される.
図VLTX-1参照.
大きさ$2n$の1次元配列.

ISW ……**入力**．制御情報.
同一の係数行列を持つ複数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)

IND……**入力**．制御情報.
係数行列の既約優対角性の判定を行うか否かを次のとおり指定する.
IND=0のとき，判定を行う.
IND=1のとき，判定を行わない.
通常は，0と指定すればよい.
(使用上の注意③参照)

IVW……**作業領域**．大きさ[$\log_2 n$]+10の1次元配列．[ ]はガウス記号である.

ICON……**出力**．コンディションコード.

表VLTX-1参照.

表VLTX-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	係数行列が既約優待角でない, または，係数行列が正則でない.	処理を打ち切る.
30000	N<1，ISW≠1，2 又は，IND≠0,l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組込み関数…ALOG2，AMAX1，AMIN1，ABS，FLOAT，MIN0

b.　注意

①　本サブルーチンは，ベクトル計算機向きのアルゴリズムであるサイクリック・リダクション法を採用しており，ベクトル計算機上での処理速度については，次のことが言える.

・ガウス消去法によるサブルーチンLTXに比べ，著しく高速である.

・処理時間は，Nの増加に対してほぼ一次関数的に増加する.

・優対角性が強い問題ほど高速である.

なお，精度の面では，既約優対角行列に対しては，サブルーチンLTXと同等である.

②　同一の係数行列を持つ3項方程式をいくつか続けて解く場合，2回目以降ISW=2として解くと，係数行列の消去過程を省略するので，計算時間が少なくなる.

③　もし，係数行列が既約優対角であることがあらかじめ分かっている場合には，IND=1として解くと既約優対角性の判定を行わないので，計算時間が少なくなる．しかし，既約優対角でない係数行列に対してIND=1と指定すると解の精度は保証されない.

④　本サブルーチンで，ISW=1として演算を行うと，配列D(i)，SBD(i)，SPD(i)，i = 1, 2, ... , nにはそれぞれ$1/d_i$，e_i/d_i，f_i/d_iなる値が格納される.

c.　使用例

同一の係数行列を持つl組のn次元3項方程式を解く．n≦1000の場合

```
C     **EXAMPLE**
      DIMENSION SBD(2000),D(2000),SPD(2000),
     *          B(2000),IVW(20)
      READ(5,500) N,L
      IF(N.LE.0) GO TO 30
      NM1=N-1
      READ(5,510) (SBD(I),I=2,N)
      READ(5,510) (D(I),I=1,N)
      READ(5,510) (SPD(I),I=1,NM1)
      WRITE(6,600) N,D(1),SPD(1)
      WRITE(6,610) (I,SBD(I),D(I),SPD(I),
     *              I=2,NM1)
      WRITE(6,610) N,SBD(N),D(N)
      ISW=1
      IND=0
      DO 10 II=1,L
      READ(5,510) (B(I),I=1,N)
      WRITE(6,620) (B(I),I=1,N)
      CALL VLTX(SBD,D,SPD,N,B,ISW,IND,IVW,
     *          ICON)
      WRITE(6,630) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,640) (B(I),I=1,N)
      ISW=2
   10 CONTINUE
   30 WRITE(6,650)
      STOP
  500 FORMAT(2I5)
  510 FORMAT(5E14.7)
  600 FORMAT('1',20X,
     *     'LINEAR EQUATIONS (TRIDIAGONAL)',
     *     /' ',20X,'ORDER= ',I5,/,
     *     /' ',25X,'COEFFICIENT MATRIX',/,
     *     /' ','(',4X,'1)',21X,2(2X,E14.7))
  610 FORMAT((' ','(',I5,')',
     *5X,3(2X,E14.7)))
  620 FORMAT(/' ',78('*'),//,
     *' ',25X,'CONSTANT VECTOR',//,
     *(' ',5(1X,E15.7)))
  630 FORMAT(/' ','CONDITION CODE OF VLTX',
     *'= ',I5)
  640 FORMAT(/' ',25X,'SOLUTION VECTOR',//,
     *(' ',5(1X,E14.7)))
  650 FORMAT(//' ',30X,'** NORMAL END **')
      END
```

(4)　手法概要

係数行列の対角要素が1となるように正規化した3項方程式(4.1)をサイクリック・リダクション法で解くことを考える.

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad (4.1)$$

ここで、

$$\boldsymbol{A}=\begin{bmatrix} 1 & f_1 & & & \mathbf{0} \\ e_2 & 1 & f_2 & & \\ & e_3 & \cdot & \cdot & \\ & & \cdot & \cdot & f_{n-1} \\ \mathbf{0} & & & e_n & 1 \end{bmatrix}$$

まず，(4.1)に対するサイクリック・リダクション法の一般形を説明し，その後で優対角性が強い場合の計算の効率化について述べる.

a.　サイクリック・リダクション法の一般形

本方法の基本は，解こうとする3項方程式に対し，適当な消去処理を施すことによって偶数

番目の未知数だけに関する3項方程式を作り出すことにある．

今，便宜上，nを奇数とする．(4.1)の相続く特定の3行に注目して，以下のように書く．

$$\begin{aligned} e_{i-1}x_{i-2} + x_{i-1} + f_{i-1}x_i &= b_{i-1} \\ e_i x_{i-1} + x_i + f_i x_{i+1} &= b_i \\ e_{i+1}x_i + x_{i+1} + f_{i+1}x_{i+2} &= b_{i+1} \end{aligned} \qquad (4.2)$$

この三つの式からx_{i-1}とx_{i+1}を次の手順で消去することができる．すなわち，第1式に$(-e_i)$を，第3式に$(-f_i)$をそれぞれ乗じ，それらを第2式に加えればよい．その結果を整理すれば，(4.3)を得る．

$$e_i^{(1)} x_{i-2} + x_i + f_i^{(1)} x_{i+2} = b_i^{(1)} \qquad (4.3)$$

ここで，
$$\begin{aligned} e_i^{(1)} &= e_{i-1}\, e_i t_i \\ f_i^{(1)} &= f_i\, f_{i+1}\, t_i \\ b_i^{(1)} &= (e_i\, b_{i-1} + f_i\, b_{i+1} - b_i)\, t_i \\ t_i &= \frac{1}{e_i\, f_{i-1} + e_{i+1}\, f_i - 1} \end{aligned}$$

である．さて(4.3)において，iを偶数のインデックス，すなわち，$i = 2, 4, \ldots, n-1$の各々について(4.3)を考えれば(ただし$x_0 = x_{n+1} = 0$とする)，以下に示す元数$[n/2]$の3項方程式が得られたことになる．

$$\begin{bmatrix} 1 & f_2^{(1)} & & & & & \\ e_4^{(1)} & 1 & f_4^{(1)} & & & 0 & \\ & e_6^{(1)} & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & 0 & & & \ddots & \ddots & f_{n-3}^{(1)} \\ & & & & & e_{n-1}^{(1)} & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \\ \cdot \\ \cdot \\ \cdot \\ x_{n-3} \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} b_2^{(1)} \\ b_4^{(1)} \\ b_6^{(1)} \\ \cdot \\ \cdot \\ \cdot \\ b_{n-3}^{(1)} \\ b_{n-1}^{(1)} \end{bmatrix} \qquad (4.4)$$

このように，元数が半分の方程式を作り出す操作を，ここでは縮小(reduction)と呼ぶ．(4.4)から，もし，$x_2, x_4, \ldots, x_{n-1}$が分ると，それらを(4.1)に代入することにより奇数番目の未知数は，

$$x_{i-1} = b_{i-1} - e_{i-1}x_{i-2} - f_{i-1}x_i,\ i = 2, 4, \ldots, n+1 \qquad (4.5)$$

より求めることができる．この操作を後退代入と呼ぶ．

ところで，(4.3)の$e_i^{(1)}$からt_iまでの計算，並びに(4.5)の計算はいずれも完全な並列性を持っており，ガウス消去法に見られる漸化関係がない．ゆえに，これらの計算はベクトル計算機上で完全にベクトル処理でき，ここにサイクリック・リダクション法が同計算機上でガウス消去法よりも高速な理由がある．

次にnが偶数の場合を考える．この場合は，n-1が奇数となるので，(4.3)を適用するiの上限をn-2に止め，その代わり，

$$\begin{aligned} e_{n-1}x_{n-2} + x_{n-1} + f_{n-1}x_n &= b_{n-1} \\ e_n x_{n-1} + x_n &= b_n \end{aligned} \qquad (4.6)$$

の2式よりx_{n-1}を消去した式，

$$e_n^{(1)} x_{n-2} + x_n = b_n^{(1)} \qquad (4.7)$$

ここで，
$$\begin{aligned} e_n^{(1)} &= e_{n-1}\, e_n\, t_n \\ b_n^{(1)} &= (e_n\, b_{n-1} - b_n)\, t_n \\ t_n &= \frac{1}{e_n\, f_{n-1} - 1} \end{aligned}$$

を付加する．こうすることによってnが偶数の場合でも(4.4)と同じような$[n/2]$元の3項方程式に縮小することができる．

以上，縮小の方法について述べたが，$[n/2]$元の3項方程式に対して，同じ手順で再度，縮小操作を施すことにより元数を更に半減することができる．

このことを繰り返せば，最終的には1元の方程式が得られ，対応する一つの未知数について解ける．この解から出発して，今度は縮小の過程を逆にたどりながら，後退代入を繰り返せば，最後に(4.1)の解が得られることになる．

(4.1)から，1元の方程式にまで縮小するための回数は，$[1\log_2 n]$である．

b. 縮小の中途打切り

上で述べた縮小操作を続けてゆくと，ある条件のもとで優対角性がますます強まり，その結果，非対角要素が対角要素の1に比べ，ついには丸め誤差の程度まで小さくなることがある．この状態においては，変形を受けた右辺ベクトルが解ベクトルの幾つかの成分に収束していく．したがって，タイミング良く縮小操作を停止し，直ちに後退代入に入れば，効率が上がることが予想される．このように，縮小操作を，1元の方程式に持ち込む以前に停止することを，ここでは中途打切りと呼ぶ．

優対角性が強まるための一つの十分条件は，(4.1)の正規化された方程式において，

$$|e_i|, |f_i| < 1/2 \qquad (4.8)$$

が成り立つことである．本サブルーチンではこの条件が成り立つとき，以下のように中途打切りまでの縮小回数を決めている．

まず(4.8)の条件下で，優対角性が強まる速さの下限を調べてみる．このために，

$$e = \max_i(|e_i|, |f_i|) < 1/2 \qquad (4.9)$$

なる量を導入し，(4.1)において，すべてのe_i，f_iをeで置き換えた行列を考える．

このとき，対角要素との比|1/*e*|は，2よりも大きくそれを

$$\left|\frac{1}{e}\right| = 2+\varepsilon^{(0)} \quad (\varepsilon^{(0)} > 0) \tag{4.10}$$

と表そう．このとき1回目の縮小で非対角要素は，

$$e' = \frac{e^2}{2e^2 - 1} \tag{4.11}$$

となり，したがって対角要素との比は，

$$\left|\frac{1}{e'}\right| = 2 + \varepsilon^{(1)}, \text{ ただし，} \varepsilon^{(1)} = 4\varepsilon^{(0)} + (\varepsilon^{(0)})^2 \tag{4.12}$$

と変わる．これより，一般に第*k*段階目の比を

$$\left|\frac{1}{e^{(k)}}\right| = 2 + \varepsilon^{(k)} \tag{4.13}$$

とすると，

$$\varepsilon^{(k+1)} = 4\varepsilon^{(k)} + (\varepsilon^{(k)})^2, \; k = 1, 2, \ldots \tag{4.14}$$

が成り立つことが推察できる．これから分かるように，$\varepsilon^{(0)} < 1$のときは，最初，1次の速さで大きくなり，$\varepsilon^{(k)} > 1$となったあとは，2次の速さで大きくなる．

本サブルーチンでは(4.14)に従う$\varepsilon^{(k)}$が，

$$\varepsilon^{(k)} \geqq \frac{1}{u} \text{ (}u\text{:丸めの誤差の単位)} \tag{4.15}$$

となる最小の整数*k*を事前に見積もり，その回数分だけ縮小操作を行い，その後直ちに代入処理に移行する．ただし，$k \geqq [\log_2 n]$のときは，中途打切りは起こらない．

以上のような中途打切りは，*n*が大きいほど，また，$\max(|e_i|, |f_i|)$が小さいほど効果が大である．

なお，詳細は，参考文献[1]，[3]及び[7]を参照されたい．

A62 - 21 - 0101　　VLTX1, DVLTX1

定数型実3項行列の連立1次方程式 (ディリクレ型，サイクリック・リダクション法)
CALL VLTX 1(D, SD, N, B, ISW, VW, IVW, ICON)

(1)　機能

実3項方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

をサイクリック・リダクション法で解く．ただし，$\boldsymbol{A}$は(1.2)のような$n\times n$の既約優対角性を満たす実3項行列である．

$$\boldsymbol{A}=\begin{bmatrix} d & e & & & \mathbf{0} \\ e & d & e & & \\ & e & d & . & \\ & & . & . & . \\ & & & . & . & e \\ \mathbf{0} & & & e & d \end{bmatrix} \qquad (1.2)$$

$$d \neq 0, \ |d| \geq 2\,|e|$$

$\boldsymbol{b}$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≧1であること．

本サブルーチンは，係数行列を(1.2)に限定することにより，一般の実3項行列を扱うサブルーチンVLTXに対して高速化を図ったものである．

(2)　パラメタ

D ……**入力**．対角要素d.

SD ……**入力**．非対角要素e.

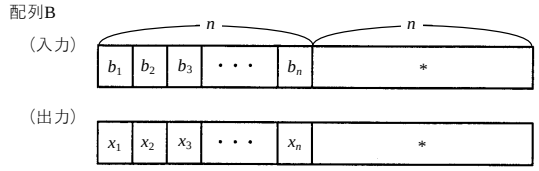
N ……**入力**．係数行列$\boldsymbol{A}$の次数n.

B ……**入力**．定数ベクトル$\boldsymbol{b}$.

B(i) = b_i, i = 1, 2, ... , nのように格納する．

出力．解ベクトル$\boldsymbol{x}$.

B(i) = x_i, i = 1, 2, ... , nのように格納される．図VLTX1-1参照．



[**注意**] ＊の部分は，本サブルーチン内で作業領域として使用される．

図VLTX1-1　ベクトル$\boldsymbol{b}$，$\boldsymbol{x}$の格納方法

大きさ2nの1次元配列．

ISW ……**入力**．制御情報．

同一の係数行列を持つ複数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).

VW ……**作業領域**．大きさ2([$\log_2 n$]＋1)の1次元配列．[ ]はガウス記号である．

IVW……**作業領域**．大きさ2([$\log_2 n$]＋1)＋10の1次元配列．

ICON…**出力**．コンディションコード．

表VLTX1-1参照．

表VLTX1-1　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	係数行列が既約優待角でない．	処理を打ち切る．
30000	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　SSLII‥AMACH，MGSSL

②　FORTRAN組込み関数…ALOG2，FLOAT，ABS，MIN0

b.　注意

①　本サブルーチンは，ベクトル計算機向きのアルゴリズムであるサイクリック・リダクション法を採用しており，ベクトル計算機上での処理速度については，次のことが言える．

・ガウス消去法によるサブルーチンLTX又はLSTXに比べ著しく高速である．

・処理時間は，Nの増加に対してほぼ一次関数的に増加する．

・優対角性が強い問題ほど高速である．

精度の面では，既約優対角行列に対してはサブルーチンLTX又はLSTXと同等である．

②　係数行列(1.2)は，簡単なディリクレ境界値問題を離散化するときに現れる．

③　同一の係数行列を持つ3項方程式をいくつか続けて解く場合は，2回目以降ISW=2として解くと，係数行列の消去過程を省略するので計算時間が少なくなる．

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≦1000の場合．

```
C     **EXAMPLE**
      DIMENSION B(2000),VW(20),IVW(30)
      READ(5,500) N
      READ(5,510) D,SD
      WRITE(6,600) N,D,SD
      READ(5,500) L
      ISW=1
      DO 10 II=1,L
      READ(5,510) (B(I),I=1,N)
      WRITE(6,610) (B(I),I=1,N)
      CALL VLTX1(D,SD,N,B,ISW,VW,IVW,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,630) (B(I),I=1,N)
      ISW=2
   10 CONTINUE
      WRITE(6,640)
      STOP
  500 FORMAT(I5)
  510 FORMAT(5E14.7)
  600 FORMAT('1',
     *  20X,'LINEAR EQUATIONS (TRIDIAGONAL)'
     *  /' ',20X,'ORDER= ',I5/
     *  /' ',25X,'COEFFICIENT MATRIX'/
     *  /' ',30X,'D =',E14.7/
     *  /' ',30X,'SD=',E14.7)
  610 FORMAT(/' ',78('*')//' ',
     *  25X,'CONSTANT VECTOR'//
     *  (' ',5(1X,E14.7)))
  620 FORMAT(/' ','CONDITION CODE OF ',
     *  'VLTX1= ',I5)
  630 FORMAT(/' ',25X,'SOLUTION VECTOR'//
     *  (' ',5(1X,E14.7)))
  640 FORMAT(//' ',30X,'** NORMAL END **')
      END
```

(4) 手法概要

係数行列の非対角要素が1となるように正規化した3項方程式(4.1)をサイクリック・リダクション法で解くことを考える．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

ここで，

$$\boldsymbol{A}=\begin{bmatrix} d & 1 & & & & \\ 1 & d & 1 & & 0 & \\ & 1 & d & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & 0 & & & \cdot & \cdot & 1 \\ & & & & & 1 & d \end{bmatrix} \tag{4.2}$$

$|d| \geq 2$

係数行列が一般の場合のサイクリック・リダクション法については，サブルーチンVLTXの「手法概要」に述べられているが，(4.2)のように単純な場合には，計算が著しく省力化できる．すなわち，係数行列の各段階における縮小操作はほんの少しのスカラー演算で済み，計算の大部分の手間は，右辺ベクトルの縮小操作で占められることになる．

以下では(4.2)を係数行列とするサイクリック・リダクション法について述べる．なお，優対角性が強い場合，縮小の中途打切りを行っているが，これについては，サブルーチンVLTXを参照されたい．

今，便宜上，nを奇数とする．(4.1)の相続く特定の3行に注目して，以下のように書く．

$$\begin{aligned} x_{i-2} + dx_{i-1} + x_i &= b_{i-1} \\ x_{i-1} + dx_i + x_{i+1} &= b_i \\ x_i + dx_{i+1} + x_{i+2} &= b_{i+1} \end{aligned} \tag{4.3}$$

この三つの式からx_{i-1}とx_{i+1}を次の手順で消去することができる．すなわち，第2式に$(-d)$を乗じ，それに第1式，第3式を加えると(4.4)を得る．

$$x_{i-2} + d^{(1)}x_i + x_{i+2} = b_i^{(1)} \tag{4.4}$$

ここで，

$$d^{(1)} = 2-d^2$$
$$b_i^{(1)} = b_{i-1}+b_{i+1}-db_i$$

である．さて，(4.4)において，iを偶数のインデックス，すなわち，i = 2, 4, ... , n-1の各々について考えれば(ただし$x_0 = x_{n+1} = 0$とする)，以下に示す元数[n/2]の3項方程式が得られたことになる．

$$\begin{bmatrix} d^{(1)} & 1 & & & & \\ 1 & d^{(1)} & 1 & & 0 & \\ & 1 & d^{(1)} & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & 0 & & & \cdot & \cdot & 1 \\ & & & & & 1 & d^{(1)} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \\ \cdot \\ \cdot \\ x_{n-3} \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} b_2^{(1)} \\ b_4^{(1)} \\ b_6^{(1)} \\ \cdot \\ \cdot \\ b_{n-3}^{(1)} \\ b_{n-1}^{(1)} \end{bmatrix} \tag{4.5}$$

(4.5)より，もし，$x_2, x_4, \ldots, x_{n-1}$が分ると，それを(4.1)に代入することにより奇数番目の未知数は，

$$x_{i-1} = (b_{i-1}-x_{i-2}-x_i) / d \tag{4.6}$$
$$i = 2, 4, \ldots, n+1$$

より求めることができる．

ところで(4.4)の$b^{(1)}$の計算，並びに(4.6)の計算はベクトル計算機上で完全にベクトル処理できる．

次に，nが偶数の場合を考える．この場合は，n-1が奇数となるので(4.4)を適用するiの上限をn-2に止め，

その代わり，

$$\begin{aligned} x_{n-2} + dx_{n-1} + x_n &= b_{n-1} \\ x_{n-1} + dx_n &= b_n \end{aligned} \tag{4.7}$$

の2式よりx_{n-1}を消去した式，

$$x_{n-2} + c^{(1)}x_n = b_n{}^{(1)}, \qquad (4.8)$$

ここで，　$c^{(1)} = 1-d^2$

$$b_n{}^{(1)} = b_{n-1}-db_n$$

を付加する．この場合は(4.5)に代わって

$$\begin{bmatrix} d^{(1)} & 1 & & & & \\ 1 & d^{(1)} & 1 & & 0 & \\ & 1 & d^{(1)} & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & 0 & & & \cdot & \cdot & 1 \\ & & & & & 1 & c^{(1)} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \\ \cdot \\ \cdot \\ x_{n-2} \\ x_n \end{bmatrix} = \begin{bmatrix} b_2^{(1)} \\ b_4^{(1)} \\ b_6^{(1)} \\ \cdot \\ \cdot \\ b_{n-2}^{(1)} \\ b_n^{(1)} \end{bmatrix} \qquad (4.9)$$

となり，依然として[n/2]元の3項方程式に縮小することができる．

以上，1回目の縮小操作について述べた．この操作を繰り返すことにより最終的には1元の方程式を作り出す．縮小の各段階における係数行列は，非対角要素がすべて1であり，対角要素については最後の要素を除いて同じである．最後の対角要素が特別に扱われるのは縮小の各段階における係数行列の次数が奇数にも偶数にもなるためである．

そこで，一般の段階における縮小操作を述べると次のようになる．今，縮小しようとしている方程式を(4.10)で表し，元数は新たにnであるとする．

$$\begin{bmatrix} d & 1 & & & & \\ 1 & d & 1 & & 0 & \\ & 1 & d & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & 0 & & & \cdot & d & 1 \\ & & & & & 1 & c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \cdot \\ \cdot \\ b_{n-1} \\ b_n \end{bmatrix} \qquad (4.10)$$

縮小操作は(4.10)より偶数番目の未知数に関する方程式を作ることである．それには，nが奇数か偶数かに応じて前述の処理を行えばよい．その結果，縮小された方程式は(4.11)のようになる．

$$\begin{bmatrix} d^{(1)} & 1 & & & & \\ 1 & d^{(1)} & 1 & & 0 & \\ & 1 & d^{(1)} & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & 0 & & & \cdot & \cdot & 1 \\ & & & & & 1 & c^{(1)} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \\ \cdot \\ \cdot \\ x_{l-2} \\ x_l \end{bmatrix} = \begin{bmatrix} b_2^{(1)} \\ b_4^{(1)} \\ b_6^{(1)} \\ \cdot \\ \cdot \\ b_{l-2}^{(1)} \\ b_l^{(1)} \end{bmatrix} \qquad (4.11)$$

ここで，　$d^{(1)} = 2-d^2$

$$b_{2r}{}^{(1)} = b_{2r-1}+b_{2r+1}-db_{2r},$$

$$r = 1, 2, \ldots, [n/2]-1$$

nが奇数のとき

$$l = n-1,\ c^{(1)} = 1-d^2 + d / c,$$

$$b_l^{(1)} = b_{n-2}-db_{n-1}+ (d / c)\, b_n$$

nが偶数のとき

$$l = n,\ c^{(1)} = 1 - dc,$$

$$b_l^{(1)} = b_{n-1}-db_n \qquad (4.12)$$

このように(4.10)から(4.11)への縮小を繰り返せば最終的に1元の方程式が得られ，それを解いて，後退代入の処理に移行すれば(4.1)の解を求めることができる．

本サブルーチンは，以上述べたように，係数行列の縮小においては，手間が少ない．したがって計算の主要部は，右辺ベクトルの縮小の計算と後退代入であり，共にベクトル計算機上でベクトル処理される．

A62‐31‐0101　　VLTX2, DVLTX2

定数型実3項行列の連立1次方程式 (ノイマン型，サイクリック・リダクション法)
CALL VLTX2(D, SD, N, B, ISW, IND, VW, IVW, ICON)

(1)　機能

実3項方程式，

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

をサイクリック・リダクション法で解く．ただし，$\boldsymbol{A}$は(1.2)，(1.3)若しくは(1.4)のような$n\times n$の既約優対角性を満たす実3項行列である．

$$\begin{bmatrix} d & 2e & & & 0 \\ e & d & e & & \\ & e & d & \cdot & \\ & & \cdot & \cdot & \cdot \\ & 0 & & \cdot & \cdot & e \\ & & & & e & d \end{bmatrix} \quad d \neq 0,\ |d| \geqq 2|e| \qquad (1.2)$$

$$\begin{bmatrix} d & e & & & & \\ e & d & e & & 0 & \\ & e & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & e \\ & 0 & & & 2e & d \end{bmatrix} \quad d \neq 0,\ |d| \geqq 2|e| \qquad (1.3)$$

$$\begin{bmatrix} d & 2e & & & \\ e & d & e & & 0 & \\ & e & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & e \\ & 0 & & & 2e & d \end{bmatrix} \quad d \neq 0,\ |d| > 2|e| \qquad (1.4)$$

$\boldsymbol{b}$はn次元の実定数ベクトル，xはn次元の解ベクトルである．n≧1であること．

本サブルーチンは，係数行列を上記に限定することにより，一般の実3項行列を扱うサブルーチンVLTXに対して高速化を図ったものである．

(2)　パラメタ

D ……**入力**．対角要素d.

SD ……**入力**．非対角要素e.

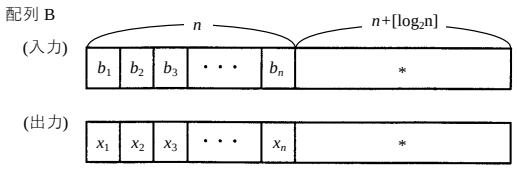
N ……**入力**．係数行列$\boldsymbol{A}$の次数n.

B……**入力**．定数ベクトル$\boldsymbol{b}$.

B(i) = b_i, i = 1, 2, ... , nのように格納する．

出力．解ベクトルx.

B(i) = x_i, i = 1, 2, ... , nのように格納される．図VLTX2-1参照．



[**注意**] *の部分は，本サブルーチン内で作業領域として使用される．

図VLTX2-1　ベクトル$\boldsymbol{b}$，$\boldsymbol{x}$の格納方法

大きさ$2n$+[$\log_2 n$]の1次元配列.

ISW ……**入力**．制御情報.

同一の係数行列を持つ複数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).

IND……**入力**．係数行列の型を指定する制御情報.

IND=1のとき(1.2)

IND=2のとき(1.3)

IND=3のとき(1.4)

VW ……**作業領域**．大きさ2([$\log_2 n$]+1)の1次元配列．[ ]はガウス記号である.

IVW……**作業領域**．大きさ2([$\log_2 n$]+1)+10の1次元配列.

ICON ……**出力**．コンディションコード.

表VLTX2-1参照.

表VLTX2-1　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	係数行列が既約優待角でない.	処理を打ち切る.
30000	N＜1，1ND≠1,2,3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　SSLII…AMACH，MGSSL

②　FORTRAN組込み関数…ALOG2，FLOAT，ABS，MIN0

b. 注意

① 本サブルーチンは，ベクトル計算機向きのアルゴリズムであるサイクリック・リダクション法を採用しており，ベクトル計算機上での処理速度については，次のことが言える．

・ガウス消去法によるサブルーチンLTXに比べ著しく高速である．

・処理時間は，Nの増加に対してほぼ一次関数的に増加する．

・優対角性が強い問題ほど高速である．

精度の面では，既約優対角行列に対してはサブルーチンLTXと同等である．

② 係数行列の(1.2)から(1.4)は，簡単なノイマン境界値問題を離散化するときに現れる．

③ 同一の係数行列を持つ3項方程式をいくつか続けて解く場合は，2回目以降ISW=2として解くと，係数行列の消去過程を省略するので計算時間が少なくなる．

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．ここでは係数行列を(1.2)の型とする．

$n \leqq 1000$の場合．

```
C      **EXAMPLE**
       DIMENSION B(2010),VW(20),IVW(30)
       READ(5,500) N
       READ(5,510) D,SD
       WRITE(6,600) N,D,SD
       READ(5,500) L
       ISW=1
       IND=1
       DO 10 II=1,L
       READ(5,510) (B(I),I=1,N)
       WRITE(6,610) (B(I),I=1,N)
       CALL VLTX2(D,SD,N,B,ISW,IND,VW,IVW,
      *ICON)
       WRITE(6,620) ICON
       IF(ICON.NE.0) STOP
       WRITE(6,630) (B(I),I=1,N)
       ISW=2
    10 CONTINUE
       WRITE(6,640)
       STOP
   500 FORMAT(I5)
   510 FORMAT(5E14.7)
   600 FORMAT('1',
      *  20X,'LINEAR EQUATIONS (TRIDIAGONAL)'
      *  /' ',20X,'ORDER= ',I5/
      *  /' ',25X,'COEFFICIENT MATRIX'/
      *  /' ',30X,'D= ',E14.7/
      *  /' ',30X,'SD=',E14.7)
   610 FORMAT(/' ',78('*')//' ',
      *  25X,'CONSTANT VECTOR'//
      *  (' ',5(1X,E14.7)))
   620 FORMAT(/' ','CONDITION CODE OF VLTX2',
      *  '= ',I5)
   630 FORMAT(/' ',25X,'SOLUTION VECTOR'//
      * (' ',5(1X,E14.7)))
   640 FORMAT(//' ',30X,'** NORMAL END **')
       END
```

(4) 手法概要

係数行列の非対角要素が1となるように正規化した3項方程式(4.1)をサイクリック・リダクション法で解くことを考える．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

ここで，$\boldsymbol{A}$は以下のいずれかであるとする．

$$\begin{bmatrix} d & 2 & & & 0 \\ 1 & d & 1 & & \\ & 1 & d & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & 1 \\ & 0 & & & 1 & d \end{bmatrix}, \quad |d| \geqq 2 \tag{4.2}$$

$$\begin{bmatrix} d & 1 & & & 0 \\ 1 & d & 1 & & \\ & 1 & d & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & 1 \\ & 0 & & & 2 & d \end{bmatrix}, \quad |d| \geqq 2 \tag{4.3}$$

$$\begin{bmatrix} d & 2 & & & 0 \\ 1 & d & 1 & & \\ & 1 & d & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & 1 \\ & 0 & & & 2 & d \end{bmatrix}, \quad |d| > 2 \tag{4.4}$$

(4.3)の行列に対しては，n行目を更に2で正規化すると，非対角要素がすべて1となり最後の対角要素は$d/2$となる．このような行列はサブルーチンVLTX1の「手法概要」に述べてある方法で解ける．したがって，以下では(4.2)と(4.4)の場合だけ考える．

(4.4)のn行目を2で正規化すると，(4.2)とは最後の対角要素だけ異なる行列にすることができる．そこで(4.2)と(4.4)を一般的に次の(4.5)の行列と見なし，これに対するサイクリック・リダクション法を述べることにする．

$$\begin{bmatrix} d & 2 & & & 0 \\ 1 & d & 1 & & \\ & 1 & d & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & 1 \\ & 0 & & & 1 & c \end{bmatrix}, \quad c=d, \text{ 又は} c=d/2 \tag{4.5}$$

(4.5)を係数行列とする方程式の定数ベクトルを新たに$\boldsymbol{b}$とする．ここでのサイクリック・リダクション法は奇数番目の未知数$x_1, x_3, x_5, \ldots$に関する方程式 ―元数は$[(n-1)/2]+1$― を導く．この点，サブルーチンVLTX1と異なる．次のように進める．

まず，

$$\begin{aligned} dx_1 + 2x_2 &= b_1 \\ x_1 + dx_2 + x_3 &= b_2 \end{aligned} \qquad (4.6)$$

の2式からx_2を消去して

$$(2-d^2)x_1 + 2x_3 = 2b_2 - db_1 \qquad (4.7)$$

を得る．次に，(4.5)の第$2j$行，第$2j+1$行，及び第$2j+2$行に関する計三つの式から，未知数x_{2j}, x_{2j+2}を消去して

$$x_{2j-1} + (2-d^2)x_{2j+1} + x_{2j+3} = b_{2j} + b_{2j+2} - db_{2j+1} \qquad (4.8)$$

を得る．これを$j=1,2,\ldots,m$(ただしmは$2j+1 \le n-2$を満たす最大のj)の各々について行う．最後にあと一つの方程式を，nが偶数か奇数かに応じて次のように作る．

nが偶数のときは，

$$\begin{aligned} x_{n-3} + dx_{n-2} + x_{n-1} &= b_{n-2} \\ x_{n-2} + dx_{n-1} + x_n &= b_{n-1} \\ x_{n-1} + cx_n &= b_n \end{aligned} \qquad (4.9)$$

の三つの式からx_{n-2}，x_nを消去して

$$x_{n-3} + (1-d^2 + d/c)x_{n-1} = b_{n-2} + (d/c)\, b_n - db_{n-1} \qquad (4.10)$$

を得る．nが奇数のときは，(4.9)の第2，第3式からx_{n-1}を消去して

$$x_{n-2} + (1-dc)x_n = b_{n-1} - db_n \qquad (4.11)$$

を得る．

以上，(4.7)，(4.8)，及び(4.10)又は(4.11)により，奇数番目の未知数だけに関する$[(n-1)/2]+1$元の方程式が得られた．それは(4.12)のように書ける．

$$\begin{bmatrix} d^{(1)} & 2 & & & 0 \\ 1 & d^{(1)} & 1 & & \\ & 1 & d^{(1)} & . & \\ & & . & . & . \\ & & & . & . & 1 \\ & 0 & & & 1 & c^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \\ x_5 \\ . \\ x_{l-2} \\ x_l \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_3^{(1)} \\ b_5^{(1)} \\ . \\ b_{l-2}^{(1)} \\ b_l^{(1)} \end{bmatrix} \qquad (4.12)$$

ここで，

$$\begin{aligned} d^{(1)} &= 2 - d^2 \\ b_1^{(1)} &= 2b_2 - db_1 \\ b_{2j+1}^{(1)} &= b_{2j} + b_{2j+2} - db_{2j+1} \\ & \quad j = 1, 2, \ldots, m \end{aligned}$$

nが偶数のとき

$$\begin{aligned} l &= n-1 \\ c^{(1)} &= 1 - d^2 + d/c \\ b_l^{(1)} &= b_{n-2} + (d/c)\, b_n - d\, b_{n-1} \end{aligned}$$

nが奇数のとき

$$\begin{aligned} l &= n \\ c^{(1)} &= 1 - dc \\ b_l^{(1)} &= b_{n-1} - d\, b_n \end{aligned}$$

これから分かるように，1回の縮小で作られた係数行列は依然，(4.5)の形と同じであり，これが本方法の特徴である．(4.12)の解が分かると，それを元の方程式に代入することにより偶数番目の未知数が求まる．

(4.12)に対して同じ縮小操作を適用すれば，更に半分の元数の方程式を導くことができ，これを繰り返せば最終的に(4.13)を係数行列とする方程式が得られる．

$$\begin{bmatrix} d^{(k)} & 2 \\ 1 & c^{(k)} \end{bmatrix} \qquad (4.13)$$

これを解き，続いて代入処理に移行すれば元の方程式の解が得られる．

なお，$|d|>2$が成り立つとき，サブルーチンVLTXと同じような中途打切りを行っている．

A62－41－0101　　VLTX3, DVLTX3

定数型実3項行列の連立1次方程式 (周期型，サイクリック・リダクション法)
CALL VLTX3(D, SD, N, B, ISW, VW, IVW, ICON)

(1)　機能

実3項方程式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (1.1)$$

をサイクリック・リダクション法で解く．ただし，$\boldsymbol{A}$は(1.2)のような $n \times n$の既約優対角性を満たす擬似実3項行列である．

$$\begin{bmatrix} d & e & & & & e \\ e & d & e & & 0 & \\ & e & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & 0 & & \cdot & \cdot & e \\ e & & & & e & d \end{bmatrix} \qquad \begin{matrix} d \neq 0 \\ , \mid d \mid > 2 \mid e \mid \end{matrix} \qquad (1.2)$$

$\boldsymbol{b}$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．$n \geqq 1$であること．

(2)　パラメタ

D ……**入力**．対角要素d.

SD ……**入力**．非対角要素e.

N ……**入力**．係数行列$\boldsymbol{A}$の次数n.

B ……**入力**．定数ベクトル$\boldsymbol{b}$.

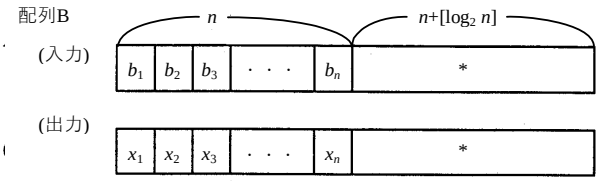
B(i)=b_i, i = 1,2,..., nのように格納する．

出力．解ベクトル$\boldsymbol{x}$.

B(i)=x_i, i = 1,2,..., nのように格納される．

図VLTX3-1参照．

大きさ$2n$+[$\log_2 n$]の1次元配列．



[**注意**]＊の部分は、本サブルーチン内で作業領域として使用される．

図VLTX 3-1　ベクトル$\boldsymbol{b}$，$\boldsymbol{x}$の格納方法

ISW ……**入力**．制御情報．

同一の係数行列を持つ複数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).

VW ……**作業領域**．大きさは 3([$\log_2 n$]+1)の1次元配列．[ ]はガウス記号である．

IVW……**作業領域**．大きさは 4([$\log_2 n$]+1)+10の1次元配列．

ICON……**出力**．コンディションコード．

表VLTX3-1参照

表VLTX3-1　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	係数行列が既約優対角でない．	処理を打ち切る．
30000	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　SSLⅡ・・・AMACH，MGSSL

②　FORTRAN組込み関数・・・ALGO2, FLOAT，ABS, MIN0

b.　注意

①　本サブルーチンは，ベクトル計算機向きのアルゴリズムであるサイクリック・リダクション法を採用しており，ベクトル機上での処理速度については，次のことが言える．

・ガウス消去法に比べ著しく高速である．

・処理時間は，Nの増加に対してほぼ一次関数的に増加する．

・優対角性が強い問題ほど高速である．

精度の面では，ガウス消去法と同じである．

②　係数行列(1.2)は，簡単な周期的境界値問題を離散化するときに現れる．

③　同一の係数行列を持つ3項方程式をいくつか続けて解く場合は，2回目以降ISW=2として解くと，係数行列の消去過程を省略するので計算時間が少なくなる．

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≦1000の場合．

```
C     **EXAMPLE**
      DIMENSION B(2010),VW(30),IVW(50)
      READ(5,500) N
      READ(5,510) D,SD
      WRITE(6,600) N,D,SD
      READ(5,500) L
      ISW=1
      DO 10 II=1,L
      READ(5,510) (B(I),I=1,N)
      WRITE(6,610) (B(I),I=1,N)
      CALL VLTX3(D,SD,N,B,ISW,VW,IVW,ICON)
      WRITE(6,620) ICON
      IF(ICON.NE.0) STOP
      WRITE(6,630) (B(I),I=1,N)
      ISW=2
   10 CONTINUE
      WRITE(6,640)
      STOP
  500 FORMAT(I5)
  510 FORMAT(5E14.7)
  600 FORMAT('1',
     *  20X,'LINEAR EQUATIONS ',
     *  '(TRIDIAGONAL)'
     *  /' ',20X,'ORDER= ',I5/
     *  /' ',25X,'COEFFICIENT MATRIX'/
     *  /' ',30X,'D= ',E14.7/
     *  /' ',30X,'SD=',E14.7)
  610 FORMAT(/' ',78('*')//' ',
     *  25X,'CONSTANT VECTOR'//
     *  (' ',5(1X,E14.7)))
  620 FORMAT(/' ','CONDITION CODE OF ',
     *  'VLTX3= ',I5)
  630 FORMAT(/' ',25X,'SOLUTION VECTOR'//
     *  (' ',5(1X,E14.7)))
  640 FORMAT(//' ',30X,'** NORMAL END **')
      END
```

(4) 手法概要

係数行列の非対角要素が1となるように正規化した3項方程式(4.1)をサイクリック・リダクション法で解くことを考える．

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.1}$$

ここで

$$\boldsymbol{A}=\begin{bmatrix} d & 1 & & & & 1 \\ 1 & d & 1 & & 0 & \\ & 1 & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & 0 & & \cdot & \cdot & 1 \\ 1 & & & & 1 & d \end{bmatrix}, \quad |d|>2 \tag{4.2}$$

この方程式は，$(n,1)$要素と$(1,n)$要素が非零であるためサイクリック・リダクション法を直接的に適用することができない．しかし，適当に変数変換すれば，二つの独立な3項方程式に分離することができ，各々は，サブルーチンVLTX1あるいはサブルーチンVLTX2の「手法概要」に述べたサイクリック・リダクション法で解ける．ここでは，分離の方法についてだけ述べる．その処理はnが偶数か奇数かに応じて異なるので以下，場合分けする．

① nが偶数の場合

$n=2l$とするとき，次のように新しい変数$\boldsymbol{y}$と$\boldsymbol{z}$を導入する．

$$y_j = x_{l-j} - x_{l+j},\ j = 1,2,\ldots,l-1 \tag{4.3}$$
$$z_{j+1} = x_{l-j} + x_{l+j},\ j = 0,1,\ldots,l$$

ただし$x_0 = x_n$とする．このとき，$\boldsymbol{y}$に関する方程式，及び$\boldsymbol{z}$に関する方程式がそれぞれ(4.4)，(4.5)のように得られる．

$$\begin{bmatrix} d & 1 & & & & 0 \\ 1 & d & 1 & & & \\ & 1 & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & 1 \\ 0 & & & & 1 & d \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \cdot \\ y_{l-2} \\ y_{l-1} \end{bmatrix} = \begin{bmatrix} b_{l-1}-b_{l+1} \\ b_{l-2}-b_{l+2} \\ b_{l-3}-b_{l+3} \\ \cdot \\ b_2-b_{n-2} \\ b_1-b_{n-1} \end{bmatrix} \tag{4.4}$$

$$\begin{bmatrix} d & 2 & & & & 0 \\ 1 & d & 1 & & & \\ & 1 & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & 1 \\ 0 & & & & 1 & c \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \cdot \\ z_l \\ z_{l+1} \end{bmatrix} = \begin{bmatrix} 2b_l \\ b_{l-1}+b_{l+1} \\ b_{l-2}-b_{l+2} \\ \cdot \\ b_1+b_{n-1} \\ b_n \end{bmatrix} \tag{4.5}$$

$,c=d/2$

(4.4)，(4.5)はそれぞれサブルーチンVLTX1，VLTX2の手法で解ける．

$\boldsymbol{y}$，$\boldsymbol{z}$が求まると$\boldsymbol{x}$は

$$x_l = z_1 / 2,\ x_n = z_{l+1} / 2$$
$$x_{l-j} = (y_j + z_{j+1}) / 2,\ x_{l+j} = (z_{j+1} - y_j) / 2$$
$$j = 1, 2, \ldots, l-1 \tag{4.6}$$

より求めることができる．

② nが奇数の場合

$n = 2l-1$とするとき，次のように新しい変数$\boldsymbol{y}$と$\boldsymbol{z}$を導入する．

$$y_j = x_{l-j} - x_{l+j},\ j = 1,2,\ldots,l-1 \tag{4.7}$$
$$z_{j+1} = x_{l-j} + x_{l+j},\ j = 0,1,\ldots,l$$

このとき，$\boldsymbol{y}$に関する方程式，及び$\boldsymbol{z}$に関する方程式がそれぞれ(4.8)，(4.9)のように得られる．

$$\begin{bmatrix} d & 1 & & & & 0 \\ 1 & d & 1 & & & \\ & 1 & d & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & 1 \\ 0 & & & & 1 & c_1 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \cdot \\ y_{l-2} \\ y_{l-1} \end{bmatrix} = \begin{bmatrix} b_{l-1}-b_{l+1} \\ b_{l-2}-b_{l+2} \\ b_{l-3}-b_{l+3} \\ \cdot \\ b_2-b_{n-1} \\ b_1-b_n \end{bmatrix}, c_1=d-1 \tag{4.8}$$

$$\begin{bmatrix} d & 2 & & & 0 \\ 1 & d & 1 & & \\ & 1 & d & \cdot & \\ & & \cdot & \cdot & \cdot \\ 0 & & & \cdot & 1 \\ & & & 1 & c_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \cdot \\ z_{l-1} \\ z_l \end{bmatrix} = \begin{bmatrix} 2b_l \\ b_{l-1}+b_{l+1} \\ b_{l-2}+b_{l+2} \\ \cdot \\ b_2+b_{n-1} \\ b_1+b_n \end{bmatrix} \quad (4.9)$$

, $c_2=d+1$

これら二つの方程式も同様にサブルーチンVLTX1，VLTX2の手法で解ける．

$\boldsymbol{y}$，$\boldsymbol{z}$が求まると$\boldsymbol{x}$は

$$\begin{aligned} x_l &= z_1 / 2 \\ x_{l-j} &= (y_j + z_{j+1}) / 2,\ x_{l+j} = (z_{j+1} - y_j) / 2 \\ & j = 1, 2, \ldots, l-1 \end{aligned} \quad (4.10)$$

より求めることができる．

A22－71－0602　　VLUIV, DVLUIV

LU分解された実行列の逆行列
CALL VLUIV (FA, K, N, IP, AI, 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.

$n\geqq 1$であること.

(2)　パラメタ

FA ……**入力**．行列$\boldsymbol{L}$と行列$\boldsymbol{U}$.
図VLUIV-1参照.
FA(K, N)なる2次元配列.

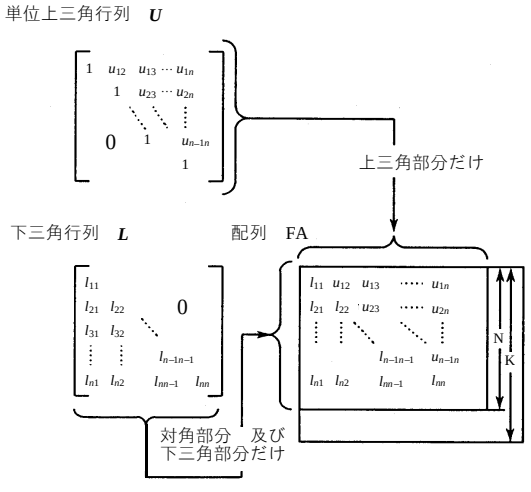
K ……**入力**．配列FA，AIの整合寸法(≧N)

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)

AI ……**出力**．逆行列$\boldsymbol{A}^{-1}$．AI(K, N)なる2次元配列

ICON……**出力**．コンディションコード.
表VLUIV－1参照.



図VLUIV-1　配列FAにおける$\boldsymbol{L}$及び$\boldsymbol{U}$の各要素の収納方法

表VLUIV-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　SSLⅡ…MGSSL

②　FORTORAN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VALU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VLAXを用いること．逆行列を求めて解くことは､VLAX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VALU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.

$n\leqq 100$の場合.

```
C      **EXAMPLE**
       DIMENSION  A(100,100),VW(100),IP(100),
      *           AI(100,100)
       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 VALU(A,100,N,0.0,IP,IS,VW,ICON)
       WRITE(6,610) ICON
       IF(ICON.GE.20000) STOP
       CALL VLUIV(A,100,N,IP,AI,ICON)
       WRITE(6,620) ICON
       IF(ICON.GE.20000) STOP
       WRITE(6,630) ((I,J,AI(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(VALU)=',I5)
620 FORMAT('0',10X,'CONDITION ',
   *'CODE(VLUIV)=',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

$$\boldsymbol{A}^{-1} = (\boldsymbol{P}^{-1}\boldsymbol{LU})^{-1} = \boldsymbol{U}^{-1}\boldsymbol{L}^{-1}\boldsymbol{P} \tag{4.2}$$

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$\boldsymbol{B}$に対する方程式$\boldsymbol{UB} = \boldsymbol{L}^{-1}$を解いて，$\boldsymbol{B}= \boldsymbol{U}^{-1}\boldsymbol{L}^{-1}$を求め，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，

$$\boldsymbol{L}^{-1} = \left(\tilde{l}_{ij}\right) \tag{4.4}$$

$\boldsymbol{LL}^{-1} = \boldsymbol{I}$より，(4.5)が成り立つ．

$$\sum_{k=1}^{n} l_{ik}\tilde{l}_{kj} = \delta_{ij}, \tag{4.5}$$

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

(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($j = 1, \ldots, n$)の要素$\tilde{l}_{ij}$を次のように求める．

$$\left.\begin{aligned} \tilde{l}_{ij} &= \left(-\sum_{k=j}^{i-1} l_{ik}\tilde{l}_{kj} \right) / l_{ii}, i = j+1, \ldots, n \\ \tilde{l}_{jj} &= 1/l_{jj} \end{aligned}\right\} \tag{4.6}$$

ここで，$l_{ii} \neq 0 (i = j, \ldots, n)$とする．

b.　$\boldsymbol{UB} = \boldsymbol{L}^{-1}$の求解．

$\boldsymbol{B}$に対する方程式$\boldsymbol{UB} = \boldsymbol{L}^{-1}$は，$\boldsymbol{B}$，$\boldsymbol{L}^{-1}$の各ベクトル$b_j$，$\tilde{l}_j$に対する方程式

$$\boldsymbol{U}b_j = \tilde{l}_j \tag{4.7}$$

を解くことで求める．ただし，

$$b_j = (b_{1j}, \ldots, b_{nj}),\quad \tilde{l}_j = \left(\tilde{l}_{1j}, \ldots, \tilde{l}_{nj}\right)$$

である．

$$b_{ij} = \tilde{l}_{ij} - \sum_{k=i+1}^{n} u_{ik}\tilde{b}_{kj} \tag{4.8}$$

から$i = n, \ldots, 1$の順に逐次求める．

A61－11－0301　　VMGGM, DVMGGM

行列の積(実行例)
CALL VMGGM(A, KA, B, KB, C, KC, M, N, L, 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≧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．
ICON……**出力**．コンディションコード．
　　表VMGGM-1　参照．

表VMGGM-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　SSLⅡ…MGSSL
②　FORTRAN基本関数…FLOAT，MOD

b.　注意
①　標準機能のサブルーチンMGGMとの相違について
　本サブルーチンは，MGGMと比較しベクトル計算機上で高速に処理をすることができる．
　MGGMでは，配列A，B，Cの整合寸法の与え方に依存して性能にばらつきが生じるが，本ルーチンでは本質的に一定である．
②　領域の節約について
　領域の節約のため，配列Aに行列$\boldsymbol{C}$を得たい場合はMGGMを使用すること．

c.　使用例
　実行列$\boldsymbol{A}$，$\boldsymbol{B}$の積を求める．m≦200，n≦400，l≦300の場合．

```
C      **EXAMPLE**
       DIMENSION A(202,400),B(402,300),
      *          C(202,300)
       CHARACTER*4 IA,IB,IC
       DATA IA/'A   '/,IB/'B   '/,
      *     IC/'C   '/
       DATA KA/202/,KB/402/,KC/202/
   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 VMGGM(A,KA,B,KB,C,KC,M,N,L,
      *           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

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
```

　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。

F15‐31‐0201　　VRFT1, DVRFT1

離散型実フーリエ変換 (性能優先型，2基底FFT)
CALL VRFT1(A, N, ISN, ISW, VW, IVW, 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，又はその逆変換をベクトル計算機向きの高速変換手法(FFT)により計算する．

ただし，$n=2^l$(l:0又は正整数)であること．

a.　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．

$$na_k = 2\cdot\sum_{j=0}^{n-1} x_j\cdot\cos kj\theta, k=0,1,...,n/2,$$

$$nb_k = 2\cdot\sum_{j=0}^{n-1} x_j\cdot\sin kj\theta, k=1,2,...,n/2-1 \quad (1.1)$$

$$,\theta = 2\pi/n$$

b.　フーリエ逆変換

$\{a_k\}$，$\{b_k\}$を入力し，(1.2)で定義する変換を行い，フーリエ級数の値$\{2x_j\}$を求める．

$$2x_j = a_0 + a_{n/2}\cdot\cos\pi j + 2\cdot\sum_{k=1}^{n/2-1}\left(a_k\cdot\cos kj\theta + b_k\cdot\sin kj\theta\right), \quad (1.2)$$

$$j=0,1,...,n-1, \quad \theta = 2\pi/n$$

(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+2の1次元配列．
図VRFT1-1参照．

N ……**入力**．変換の項数n

ISN……**入力**．変換か逆変換かを指定する(≠0)
変換　：ISN=+1
逆変換：ISN=‐1
(使用上の注意④参照)

ISW……**入力**．変換の初期設定を制御する情報．
初回呼出しのとき：ISW=0
継続呼出しのとき：ISW=1
(使用上の注意②参照)

VW……**作業領域**．
大きさmax(n(l+1)/2,1)の1次元配列．

IVW ……**作業領域**．大きさn max(l -4,2)/2の1次元配列．

ICON……**出力**．コンディションコード．
表VRFT1-1参照．

配列A

	$\{x_j\}$	$\{a_k\}$ $\{b_k\}$
A(1)	x_0	a_0
A(2)	x_1	*
A(3)	x_2	a_1
A(4)	x_3	b_1
・	・	・
・	・	・
・	・	・
A(N−1)	x_{n-2}	$a_{\frac{n}{2}}$ -1
A(N)	x_{n-1}	$b_{\frac{n}{2}}$ -1
A(N+1)	*	$a_{\frac{n}{2}}$
A(N+2)	*	*

[**注意**]＊は，入力時は任意，出力時は0. 0がセットされる．

図VRFT1-1　データの格納方法

表VRFT1-1　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，ISW≠0，1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　SSLⅡ…VCFT1，UVRFT，UVTB1，UVF91，UVFA1，UVFB1，UVFX1，UBANK，MGSSL

②　FORTRAN 組込み関数…ALOG2，SIN，COS，ATAN，IABS，IAND，MOD，FLOAT

b.　注意

①　サブルーチンの使い分け

本サブルーチンは，実フーリエ変換をベクトル計算機上で高速処理する．汎用計算機上で処理する場合は，サブルーチンRFTが適している．

本サブルーチンは，ベクトル計算機用のもう一つのサブルーチンVRFT2と同等の機能を持つ．本サブルーチンは，独立した複数組の変換を行う場合に適している．反面，作業用配列を余分に必要とするので，言わば「性能優先型」のサブルーチンである．作業用配列の確保が困難な場合は，多少性能は低下するが「メモリ節約型」のサブルーチンVRFT2が適している．

② ISWによる制御

複数組の変換を行う場合，2回目以降の呼出しのときISW=1と指定すると，変換に必要な三角関数表やリストベクトルなどの生成を省略するので，効率良く処理される．このとき，配列VW，IVWの内容は変更せず呼び出すこと．

複数組の変換の項数nが各々異なる場合でも，ISW=1の指定は有効である．しかし，可能な限り同一の項数の変換が連続するように呼び出すことが望ましい．

複素フーリエ変換のサブルーチンVCFT1と組み合わせて本サブルーチンを呼び出す場合でも，ISW=1の指定は有効である．

③ 作業用配列の大きさ早見表

l6≦n≦4096の場合の大きさを以下に示す．

l	n	VW	IVW
4	16	40	16
5	32	96	32
6	64	224	64
7	128	512	192
8	256	1152	512
9	512	2560	1280
10	1024	5632	3072
11	2048	12288	7168
12	4096	26624	16384

④ ISNの与え方

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，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，計算結果も間隔Iで格納される．

ベクトル計算機で実行する場合，メモリアクセスを効率良くするために，間隔Iは次のような値であることが望ましい．使用例②参照．

単精度演算(VRFT1) :I=4P+2，P=0,1,2,...

倍精度演算(DVRFT1) :I=2P+1，P=1,2,3,...

⑤ 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．

$$a_k = \frac{2}{n}\sum_{j=0}^{n-1} x_j \cdot \cos kj\theta,\ k = 0,1,...,n/2$$

$$b_k = \frac{2}{n}\sum_{j=0}^{n-1} x_j \cdot \sin kj\theta,\ k = 1,2,...,n/2-1$$

$$,\theta = 2\pi/n \qquad (3.1)$$

$$x_j = \frac{1}{2}a_0 + \frac{1}{2}a_{n/2}\cdot\cos\pi j + \sum_{k=0}^{n/2-1}\left(a_k \cdot \cos kj\theta + b_k \cdot \sin kj\theta\right),$$

$$j = 0,1,...,n-1,\quad \theta = 2\pi/n \qquad (3.2)$$

本サブルーチンでは，(3.1)，(3.2)の左辺に対応して$\{na_k\}\{nb_k\}$又は$\{2x_j\}$を求める．したがって，結果の正規化は必要に応じて行うこと．

c. 使用例

① 複数組のフーリエ変換の場合

k組の独立したフーリエ変換(n項)を求める．k≦64，n≦512の場合．

```
C      **EXAMPLE**
       DIMENSION A(514,64),VW(2560),
      *          IVW(1280)
       READ(5,500) N,K
       READ(5,510) ((A(I,J),I=1,N),J=1,K)
C
       ISN=1
       ISW=0
       CALL VRFT1(A,N,ISN,ISW,VW,IVW,ICON)
       IF(ICON.NE.0) STOP
       ISW=1
       DO 10 J=2,K
       CALL VRFT1(A(1,J),N,ISN,ISW,VW,IVW
      *          ,ICON)
   10 CONTINUE
C
       WRITE(6,600) K,N
       DO 20 J=1,K
   20 WRITE(6,610) J,(I,A(I,J),I=1,N+2)
C
  500 FORMAT(2I5)
  510 FORMAT(E15.7)
  600 FORMAT(5X,'***',I3,' SET TRANSFORMS'
     *       ' OF',' TERM',I4//)
  610 FORMAT(8X,I3,'-TH TRANSFORM'/
     *      (8X,I3,E16.7))
       STOP
       END
```

② 多次元のフーリエ変換の場合

2次元のフーリエ変換(n1×n2項)を求める．n1≦512，n2≦64の場合．

プログラム中，行方向はサブルーチンVCFT1を用いて複素フーリエ変換している．その際のデータの間隔(配列宣言の第一寸法宣言子)ISN=514は，ベクトル計算機上に適している(514=4P+2，P=128)．

倍精度演算の場合は，ISN=517がよい．

```
C     **EXAMPLE**
      DIMENSION A(514,64),VW(2560),
     *          IVW(1280)
      READ(5,500) N1,N2
      READ(5,510) ((A(I,J),I=1,N1),J=1,N2)
C -- N2 SET REAL TRANSFORMS OF TERM N1 --
      ISN=1
      ISW=0
      CALL VRFT1(A,N1,ISN,ISW,VW,IVW,ICON)
      IF(ICON.NE.0)STOP
      ISW=1
      DO 10 J=2,N2
      CALL VRFT1(A(I,J),N1,ISN,ISW,VW,IVW,
     *           ICON)
   10 CONTINUE
C -- HALF SET COMPLEX TRANS. OF TERM N2 --
      ISN=514
      CALL VCFT1(A,A(2,1),N2,
     *           ISN,ISW,VW,IVW,ICON)
      IF (ICON.NE.0) STOP
      DO 20 I=3,N1+2,2
      CALL VCFT1(A(I,1),A(I+1,1),N2,
     *           ISN,ISW,VW,IVW,ICON)
   20 CONTINUE
C
      WRITE(6,600) N1,N2
      DO 30 J=1,N2
   30 WRITE(6,610) J,(I,A(I,J),
     *             A(I+1,J),I=1,N1+2,2)
C
  500 FORMAT(2I5)
  510 FORMAT(E15.7)
  600 FORMAT(5X,'***2-DIMENSIONAL ',
     *'TRANSFORM OF TERM',I4,' BY ',I4)
  610 FORMAT(8X,I3,'-TH COLUMN'//
     *(8X,I3,2E16.7))
      STOP
      END
```

(4)　手法概要

項数$n(=2^l)$の離散型実フーリエ変換を，ベクトル計算機向きの高速変換手法(Isogeometric型及びSelf-sorting型のFFT)により計算する．

実フーリエ変換は，変換の対象となる実データ$\{x_j\}$を虚部が0.0であ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を行えば得られる．しかし，この場合，複素フーリエ変換の性質を利用すると，効率良く変換が可能であることが知られている．

いま，複素変換を(4.1)で定義する．

$$\alpha_k = \sum_{j=0}^{n-1} x_j \cdot \omega^{jk}, \quad k = 0,1,...,n-1$$
$$, \quad \omega = \exp(2\pi i / n) \tag{4.1}$$

$\{x_j\}$が実データの場合は，(4.2)なる共役関係が成り立つ．

$$\alpha_{n-k} = \alpha_k^*, \quad k = 1,2,...,n-1 \tag{4.2}$$

＊は複素共役を表す．

一方，実フーリエ変換の結果$\{a_k\}$，$\{b_k\}$と，複素フーリエ変換の結果$\{a_k\}$とには，(4.3)に示すような関係がある．

$$\begin{aligned} a_0 &= 2 \cdot \alpha_0, a_{n/2} = 2 \cdot \alpha_{n/2} \\ a_k &= (\alpha_k + \alpha_{n-k}), \quad k = 1,2,...,n/2-1 \\ b_k &= i(\alpha_k - \alpha_{n-k}), \quad k = 1,2,...,n/2-1 \end{aligned} \tag{4.3}$$

したがって，実フーリエ変換を求める場合は，複素フーリエ変換

$$\alpha_k = \sum_{j=0}^{n-1} x_j \omega^{jk}, \quad k = 0,1,...,n/2 \tag{4.4}$$
$$, \quad \omega = \exp(2\pi i / n)$$

を計算し，(4.2)，(4.3)を適用すれば得られることが分かる．

本サブルーチンでは，(4.4)の複素フーリエ変換を，ベクトル計算機向きの高速変換手法により計算している．

なお，複素フーリエ変換を利用する実フーリエ変換の方法の詳細については，サブルーチンRFTの「手法概要」を，また，ベクトル計算機向きの高速変換手法については，サブルーチンVCFT1の「手法概要」を参照されたい．

F15‐31‐0301　　VRFT2, DVRFT2

離散型実フーリエ変換 (メモリ節約型，２基底FFT)
CALL VRFT2(A, N, ISN, ISW, VW, IVW, 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，又はその逆変換をベクトル計算機向きの高速変換手法(FFT)により計算する.

だだし，$n=2^l$(l: 0又は正整数)であること.

a.　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.

$$na_k = 2\cdot\sum_{j=0}^{n-1} x_j\cdot\cos kj\theta\ ,k=0,1,...,n/2$$

$$nb_k = 2\cdot\sum_{j=0}^{n-1} x_j\cdot\sin kj\theta\ ,k=1,2,...,n/2-1 \qquad (1.1)$$

$$,\theta = 2\pi/n$$

b.　フーリエ逆変換

$\{a_k\}$，$\{b_k\}$を入力し，(1.2)で定義する変換を行い，フーリエ級数の値$\{2x_j\}$を求める.

$$2x_j = a_0 + a_{n/2}\cos\pi j + 2\cdot\sum_{k=0}^{n/2-1}\left(a_k\cdot\cos kj\theta + b_k\cdot\sin kj\theta\right), \qquad (1.2)$$

$$j=0,1,...,n-1,\theta=2\pi/n$$

(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+2の1次元配列.
　　図VRFT2-1参照.

N ……**入力**. 変換の項数n.

ISN……**入力**. 変換か逆変換かを指定する(≠0)
　　変換　:ISN = +1
　　逆変換:ISN = -1
　　(使用上の注意④参照)

ISW……**入力**. 変換の初期設定を制御する情報.
　　初回呼出しのとき:ISW = 0
　　継続呼出しのとき:ISW = 1
　　(使用上の注意②参照)

VW……**作業領域**. 大きさ7n/2の1次元配列.

IVW ……**作業領域**. 大きさ3n/2の1次元配列.

ICON……**出力**. コンディションコード.
　　表VRFT2-1参照.

配列 A

	$\{x_j\}$	$\{a_k\}$ $\{b_k\}$
A(1)	x_0	a_0
A(2)	x_1	*
A(3)	x_2	a_1
A(4)	x_3	b_1
.	.	.
.	.	.
.	.	.
A(N–1)	x_{n-2}	$a_{\frac{n}{2}-1}$
A(N)	x_{n-1}	$b_{\frac{n}{2}-1}$
A(N+1)	*	$a_{\frac{n}{2}}$
A(N+2)	*	*

[**注意**]＊は，入力時は任意，出力時は0. 0がセットされる.

図VRFT2-1　データの格納方法

表VRFT2-1　コンテソションコード

コード	意　　味	処　理　内　容
0	エラーなし.	―
30000	ISN＝0，JSW≠0，1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　SSLⅡ…VCFT2，UVRFT，UVTB2，UVF92，UVFA2，UVFB2，UVFX2，UBANK，MGSSL

②　FORTRAN組込み関数…ALOG2，SIN，COS，IABS，FLOAT，IAND，MOD

b.　注意

①　サブルーチンの使い分け

本サブルーチンは，実フーリエ変換をベクトル計算機上で高速処理する．汎用計算機上で処理する場合は，サブルーチンRFTが適している.

本サブルーチンは，ベクトル計算機用のもう一つのサブルーチンVRFT1と同等の機能を持つ．本サブルーチンは，単独の変換を行う場合に適している.

作業用配列は，必要最小限におさえてあり，言わば，「メモリ節約型」のサブルーチンである．複数組の変換を行う場合で，作業用配列を十分に確保できるならば，「性能優先型」のサブルーチンVRFT1が適している.

② ISWによる制御

複数組の変換を行う場合，2回目以降の呼出しのときISW=1と指定すると，変換に必要な三角関数表やリストベクトルなどの生成を省略するので，多少効率良く処理される．このとき，配列VW，IVWの内容は変更せず呼び出すこと．

複数組の変換の項数nが各々異なる場合でも，ISW=1の指定は有効である．しかし，可能な限り同一の項数の変換が連続するように呼び出すことが望ましい．

複素フーリエ変換のサブルーチンVCFT2と組み合わせて本サブルーチンを呼び出す場合でも，ISW=1の指定は有効である．

③ 作業用配列の大きさ早見表．

$16 \leqq n \leqq 4096$の場合の大きさを以下に示す．

l	n	VW	IVW
4	16	56	24
5	32	112	48
6	64	224	96
7	128	448	192
8	256	896	384
9	512	1792	768
10	1024	3584	1536
11	2048	7168	3072
12	4096	14336	6144

④ ISNの与え方

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，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，計算結果も間隔Iで格納される．ベクトル計算機で実行する場合，メモリアクセスを効率良くするために，間隔Iは次のような値であることが望ましい．

単精度演算(VRFT2):I=4P+2, P=0,1,2,…

倍精度演算(DVRFT2):I=2P+1, P=1,2,3,…

⑤ 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．

$$a_k = \frac{2}{n}\sum_{j=0}^{n-1} x_j \cdot \cos kj\theta, \quad k = 0,1,...,n/2.$$

$$b_k = \frac{2}{n}\sum_{j=0}^{n-1} x_j \cdot \sin kj\theta, \quad k = 1,2,...,n/2-1$$

$$, \quad \theta = 2\pi/n \tag{3.1}$$

$$x_j = \frac{1}{2}a_0 + \frac{1}{2}a_{n/2} \cdot \cos \pi j + \sum_{k=0}^{n/2-1}\left(a_k \cdot \cos kj\theta + b_k \cdot \sin kj\theta\right),$$

$$j = 0,1,...,n-1, \quad \theta = 2\pi/n \tag{3.2}$$

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．したがって，結果の正規化は必要に応じて行うこと．

c. 使用例

1次元のフーリエ変換(n項)及びその逆変換連続して計算する．$n \leqq 1024$の場合．

```
C      **EXAMPLE**
       DIMENSION A(1026),VW(3584),IVW(1536)
       READ(5,500) N
       READ(5,510) (A(I),I=1,N)
C      ----FOURIER ANALYSIS----
       ISN=1
       ISW=0
       CALL VRFT2(A,N,ISN,ISW,VW,IVW,ICON)
       IF(ICON.NE.0)STOP
C      ----NORMALIZATION----
       ANOR=2.0/FLOAT(N)
       DO 10 I=1,N+2
    10 A(I)=ANOR*A(I)
       WRITE(6,600) N,(I,A(I),A(I+1),
      *             I=1,N+2,2)
C      ----FOURIER SYNTHESIS----
       ISN=-1
       ISW=1
       CALL VRFT2(A,N,ISN,ISW,VW,IVW,ICON)
       IF(ICON.NE.0) STOP
C      ----NORMALIZATION----
       ANOR=0.5
       DO 20 I=1,N
    20 A(I)=ANOR*A(I)
       WRITE(6,610) N,(I,A(I),I=1,N)
C
  500 FORMAT(I5)
  510 FORMAT(E15.7)
  600 FORMAT(5X,
     *'***FOURIER ANALYSIS OF TERM',I5//
     *(8X,I3,2E16.7))
  610 FORMAT(5X,
     *'***FOURIER SYNTHESIS OF TERM',I5//
     *(8X,I3,E16.7))
       STOP
       END
```

(4) 手法概要

項数$n(=2^l)$の離散型実フーリエ変換を，ベクトル計算機向きの高速変換手法(Isogeometric型及びSelfsorting型のFFT)により計算する．

実フーリエ変換は，変換の対象となる実データ$\{x_j\}$を虚部が0.0であ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を行えば得られる．しかし，この場合，複素フーリエ変換の性質を利用すると，効率良く変換が可能であることが知られている．

いま，複素変換を(4.1)で定義する．

$$\alpha_k = \sum_{j=0}^{n-1} x_j \cdot \omega^{jk}, k = 0,1,...,n-1 \qquad (4.1)$$
$$\omega = \exp(2\pi i / n)$$

$\{x_j\}$が実データの場合は，(4.2)なる共役関係が成り立つ。

$$\alpha_{n-k} = \alpha_k^*, \quad k = 1,2,...,n-1 \qquad (4.2)$$

＊は複素共役を表す．

一方，実フーリエ変換の結果$\{a_k\}$，$\{b_k\}$と，複素フーリエ変換の結果$\{a_k\}$とには，(4.3）に示すような関係がある．

$$a_0 = 2 \cdot \alpha_0, \quad a_{n/2} = 2 \cdot \alpha_{n/2}$$
$$a_k = (\alpha_k + \alpha_{n-k}), \quad k = 1,2,...,n/2-1 \qquad (4.3)$$
$$b_k = i(\alpha_k - \alpha_{n-k}), \quad k = 1,2,...,n/2-1$$

したがって，実フーリエ変換を求める場合は，複素フーリエ変換

$$\alpha_k = \sum_{j=0}^{n-1} x_j \cdot \omega^{jk}, \; k = 0,1,...,n/2 \qquad (4.4)$$
$$, \omega = \exp(2\pi i / n)$$

を計算し，(4.2)，(4.3)を適用すれば得られることが分かる．

本サブルーチンでは，(4.4)の複素フーリエ変換を，ベクトル計算機向きの高速変換手法により計算している．

なお，複素フーリエ変換を利用する実フーリエ変換の方法の詳細については，サブルーチンRFTの「手法概要」を，また，ベクトル計算機向きの高速変換手法については，サブルーチンVCFT1の「手法概要」を参照されたい．

B61‐21‐0201　　VSEG2, DVSEG2

実対称行列の固有値固有ベクトル (並列バイセクション法，逆反復法)
CALL VSEG2(A, N, M, EPST, E, EV, K, VW, IVW, ICON)

(1)　機能

*n*次の実対称行列***A***の固有値を，並列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*個求める．また，対応する*m*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．ただし，$1 \leqq m \leqq n$であること．

(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.
負の値を与えると標準値が採用される.
(使用上の注意②参照)

E…… **出力**．固有値.
大きさ*m*の1次元配列.
Mが正の場合大きい順に，Mが負の場合小さい順に格納される.

EV…… **出力**．固有ベクトル.
EV(K, *m*)なる2次元配列.
固有値E(J)に対応する固有ベクトルは，
EV(I,J)，I = 1,..., Nに格納される.

K …… **入力**．配列EVの整合寸法(≧*n*).

VW…… **作業領域**．大きさ15*n*の1次元配列.

IVW …… **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VSEG2-1参照.

(3)　使用上の注意

a.　使用する副プログラム

①　SSLⅡ…TRID1，UVTG2，TRBK，
UVBCT，AMACH，MGSSL

表VSEG2-1　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，E(1)＝A(1)とする.
15000	固有ベクトルの計算で求まらないものがあった.	求まらない固有ベクトルを零ベクトルとする.
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零ベクトルとする.
30000	M＝0，N＜|M|又は，K＜Nであった.	処理を打ち切る.

②　FORTRAM組込み関数…IABS，SQRT，
SIGN，ABS，
AMAX1

b.　注意

①　本サブルーチンは，機能的にはサブルーチンSEIG2と同等であるが，並列バイセクション法を採用することによりベクトル計算機上での高速化を図ったものである．両サブルーチンの間で作業領域の確保の仕方が異なることに注意すること.

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，

$$EPST = u \cdot \max(|\lambda_{\max}|, |\lambda_{\min}|) \qquad (3.1)$$

である．ここで，$\lambda_{\max}$及び$\lambda_{\min}$は，$\boldsymbol{Ax} = \lambda\boldsymbol{x}$の固有値の存在範囲(ゲルシュゴリンの定理により与えられる)の上限と下限である.
固有値の絶対値が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，(3.1)により収束判定すると，一般に小さい方の固有値を精度よく求めることは困難である．このような場合は，EPSTに小さな値(絶対誤差)を設定することにより，小さい方の固有値も精度よく求めることができる．ただし，反復回数が増加するため処理速度は遅くなる.
収束判定法については，手法概要を参照されたい.

c.　使用例

*n*次の実対行列***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.
$n \leqq 100$，$m \leqq 20$の場合.

```
C     **EXAMPLE**
      DIMENSION A(5050),E(20),EV(102,20),
     *          VW(1500),IVW(700)
   10 READ(5,500,END=900) N,M,EPST
      NT=N*(N+1)/2
      READ(5,510) (A(I),I=1,NT)
      WRITE(6,600) N,M
      IJ=0
      DO 20 I=1,N
      IJ=IJ+I
   20 WRITE(6,610) I,(A(J),J=IJ-I+1,IJ)
      CALL VSEG2(A,N,M,EPST,E,EV,102,
     *           VW,IVW,ICON)
      WRITE(6,620) ICON
      IF(ICON.GE.20000) GO TO 10
      MM=IABS(M)
      CALL SEPRT(E,EV,102,N,MM)
      GO TO 10
  900 STOP
  500 FORMAT(2I5,E10.2)
  510 FORMAT(5E15.7)
  600 FORMAT('1',//'*** ORIGINAL MATRIX ',
     *       'N=',I4,2X,'M=',I4//)
  610 FORMAT('0',2X,I3,5E15.7/(6X,5E15.7))
  620 FORMAT('0'//'*** 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印刷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$\boldsymbol{A}$の固有値を，並列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，図VSEG2-1に示す実対称3重対角行列$\boldsymbol{T}$に変換する．この操作を(4.1)で示す．

$$\boldsymbol{T} = \boldsymbol{Q}_H^{\mathrm{T}} \boldsymbol{A} \boldsymbol{Q}_H \tag{4.1}$$

ここで，$\boldsymbol{Q}_H$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．

$$\begin{bmatrix} c_1 & b_2 & & & & & \\ b_2 & c_2 & b_3 & & & & \\ & b_3 & c_3 & b_4 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & b_n \\ & & & & & b_n & c_n \end{bmatrix}$$

図VSEG2-1　実対称3重対角行列$\boldsymbol{T}$

次に，変換された行列$\boldsymbol{T}$に対して並列バイセクション法を適用し，m個の固有値を求める．さらに，求めたm個の固有値に対応する行列$\boldsymbol{T}$の固有ベクトルを逆反復法により求める．逆反復法は，

$$(\boldsymbol{T} - \lambda \boldsymbol{I}) y_r = y_{r-1}, \quad r = 1,2,... \tag{4.2}$$

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λは並列バイセクション法により求めた固有値，y_rは反復ベクトルである．並列バイセクション法については後述する．逆反復法については，サブルーチンTEIG2の手法概要を参照されたい．

次に，$\boldsymbol{A}$の固有ベクトルを求める．$\boldsymbol{T}$の固有ベクトルをyとすれば，$\boldsymbol{A}$の固有ベクトルxは，(4.1)の$\boldsymbol{Q}_H$を用いて，

$$x = \boldsymbol{Q}_H y \tag{4.3}$$

により計算できる．この操作は，サブルーチンTRBKを用いて行う．

並列バイセクション法

以下では，説明の都合上，大きい方からm個の固有値を求めることを考える．

ここでλを変数とし，行列$(\boldsymbol{T} - \lambda \boldsymbol{I})$の左上からの主小行列式の値を$p_i(\lambda)$とすれば，次のような漸化関係が成り立つ．

$$p_0(\lambda) = 1, \quad p_1(\lambda) = c_1 - \lambda,$$
$$p_i(\lambda) = (c_i - \lambda) \times p_{i-1}(\lambda) - b_i^2 \times p_{i-2}(\lambda),$$
$$i = 2, 3, \ldots, n \tag{4.4}$$

(4.4)の多項式列$p_0(\lambda), p_1(\lambda), \ldots, p_n(\lambda)$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，隣合う項の符号が反転する回数を$\alpha(\lambda)$とすればこの$\alpha(\lambda)$は$\lambda$より小さい固有値の個数と一致する．

この定理を応用して固有値の存在区間の2分割を繰り反し，固有値を一つずつ求める方法がバイセクション法である．一般には，(4.4)の計算でアンダフローやオーバフローが起きやすく，それを回避するために，(4.5)に示す多項式$q_i(\lambda)$を用いて評価する．

$$q_i(\lambda) = p_i(\lambda) / p_{i-1}(\lambda) \tag{4.5}$$

この場合は，$q_i(\lambda)$が負となる回数が，λより小さい固有値の個数と一致する．以下で，$q_i(\lambda)$が負となる回数をあらためて$\alpha(\lambda)$とする．

並列バイセクション法は，求めるm個の固有値λ_j, j = 1, 2, ... , m対して，各々の固有値に対応する存在区間を設定してm組同時にバイセクション法を適用する方法である．いま，j番目の固有値λ_jに対する存在区間を$\left[a_j^{(k)}, b_j^{(k)}\right]$と表す．$k$は反復回数を示す．初期存在区間は$\left[a_j^{(0)}, b_j^{(0)}\right]$であり，

$$\alpha\left(a_j^{(0)}\right) = j-1,$$
$$\alpha\left(b_j^{(0)}\right) = j \qquad (4.6)$$

を満たすように設定する．並列バイセクションは以下に示す手順①～③をk=0,1,2,... と反復し$\left[a_j^{(k)}, b_j^{(k)}\right]$を十分縮めて，$\lambda_j$をその区間の中点で近似する．

① λ_jを区間の中点で近似する．

$$h_j^{(k)} = \left(a_j^{(k)} + b_j^{(k)}\right)/2,\ j = 1,2,...,m \qquad (4.7)$$
$$\alpha\left(h_j^{(k)}\right) = 0,\ j = 1,2,...,m \qquad (4.8)$$

② スツルム列q_i, i=1,2,...,nを評価し，符号が負となる回数を調べる．

$$q_i\left(h_j^{(k)}\right),$$
$$q_i\left(h_j^{(k)}\right) < 0 \text{ ならば } \alpha\left(h_j^{(k)}\right) = \alpha\left(h_j^{(k)}\right) + 1,$$
$$j = 1,2,...,m \qquad (4.9)$$

③ 存在区間を改訂する．

$\alpha\left(h_j^{(k)}\right) = j-1$, ならば

$$a_j^{(k+1)} = h_j^{(k)}, b_j^{(k+1)} = b_j^{(k)},$$

$\alpha\left(h_j^{(k)}\right) = j$, ならば

$$a_j^{(k+1)} = a_j^{(k)}, b_j^{(k+1)} = h_j^{(k)},$$
$$j = 1,2,...,m \qquad (4.10)$$

固有値の収束判定法とEPSTの与え方

本サブルーチンの収束判定は，(4.11)により行う．

$$b_j^{(k)} - a_j^{(k)} \leqq 2u\left(\left|b_j^{(k)}\right| + \left|a_j^{(k)}\right|\right) + \text{EPST} \qquad (4.11)$$

ここで，uは丸め誤差の単位，EPSTは求める固有の絶対誤差の上限として指定された値である．(4.11)を満たしたならば，$\left(b_j^{(k)} - a_j^{(k)}\right)/2$を$j$番目の固有値$\lambda_j$とする．EPSTには，必要な精度での打ち切りを制御する役割がある．もし，EPST＝0.0と与えると，(4.11)は(4.12)となる．

$$b_j^{(k)} - a_j^{(k)} \leq 2u\left(\left|b_j^{(k)}\right| + \left|a_j^{(k)}\right|\right) \qquad (4.12)$$

このとき，$b_j^{(k)}$ と $a_j^{(k)}$ の最下位のけたがぼぼ一致するまで2分割していくことになる．一方，EPST＞0.0と与えると指定した精度で反復を止める．特に，固定値に零を含む場合は，EPST＞0.0と与える必要がある．

本サブルーチンではEPST＜0.0と与えると，標準値として

$$\text{EPST} = u \cdot \max(|\lambda_{\max}|, |\lambda_{\min}|)$$

を採用する．ここで，$\lambda_{\max}$, $\lambda_{\min}$は，ゲルシュゴリンの定理より求めた全固有値を含む区間の下限と上限の値である．

F16‐21‐0201　　VSIN1, DVSIN1

離散型sine変換 (2基底FFT)
CALL VSIN1(A, N, TAB, VW, IVW, 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$\{x_j\}$

$$x_j = x(\theta j), j = 0,1,...,n-1$$
$$,\theta = \pi / n \qquad (1.1)$$

が与えられたとき，離散型sine変換，又はその逆変換をベクトル計算機向きの高速変換手法(FFT)により計算する．

ただし，$n=2^l$(l:正整換)であること．

a.　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$\{2n\cdot b_k\}$を求める．

$$2n\cdot b_k = 4\cdot\sum_{j=0}^{n-1} x_j \cdot \sin kj\theta, k = 0,1,...,n-1$$
$$,\theta = \pi / n \qquad (1.2)$$

ただし，$x_0=0$.

b.　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$\{4\cdot x_j\}$を求める．

$$4\cdot x_j = 4\cdot\sum_{k=0}^{n-1} b_k \cdot \sin kj\theta, j = 0,1,...,n-1$$
$$,\theta = \pi / n \qquad (1.3)$$

ただし，$b_0=0$.

(2)　パラメタ

A ……**入力**．$\{x_j\}$又は$\{b_k\}$.
　　出力．$\{2n\cdot b_k\}$又は$\{4\cdot x_j\}$.
　　大きさn+2の1次元配列.
　　図VSIN1-1参照.

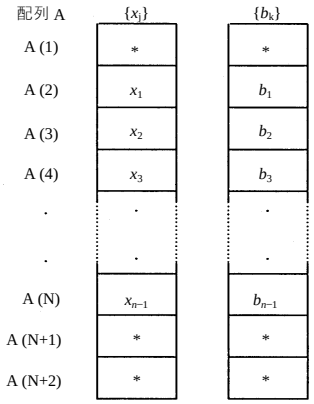
N ……**入力**．標本の数n.

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2n+4の1次元配列.

VW……**作業領域**.
　　大きさmax(n(l+1)/2,1)の1次元配列.

IVW ……**作業領域**.
　　大きさ$n\cdot$max(l−4,2)/2の1次元配列.

ICON……**出力**．コンディションコード.
　　表VSIN1-1参照.



[**注意**]　$\{2nb_k\}$，$\{4x_j\}$についても同様.
　　*は，入力時は任意．出力時は0.0がセットされる.

図VSIN-1　データの格納方法

表VSIN1-1　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　SSLⅡ…VRFT1，VCFT1，UVRFT，UVTB1，UVF91，UVFA1，UVFB1，UVFX1，UBANK，UVTAB，MGSSL

②　FORTRAN組込み関数…ALOG2，SIN，COS，ATAN，IABS，IAND，MOD，FLOAT

b.　注意

①　サブルーチンの使い分け

本サブルーチンは，離散型sine変換を、ベクトル計算機上で高速処理する．汎用計算機上で処理する場合は，サブルーチンFSINTが適している.

②　複数組の変換について

複数組の変換を行う場合，2回目以降の呼出しのときは，変換に必要な三角関数表やリストベクトルなどの生成を省略するので，効率良く処理される．このとき，配列TAB，VW，IVWの内容は変更せず呼び出すこと.

複数組の変換の項数nが各々異なる場合でも，それまでに生成した，配列TAB，VW，IVWの内容は有効である．しかし，可能な限り同一の項数の変換が連続するように呼び出すことが望ましい.

③　三角関数表及び作業用配列の大きさ早見表

16≦n≦4096の場合の大きさを以下に示す.

l	n	TAB	VW	IVW
4	16	36	40	16
5	32	68	96	32
6	64	132	224	64
7	128	260	512	192
8	256	516	1152	512
9	512	1028	2560	1280
10	1024	2052	5632	3072
11	2048	4100	12288	7168
12	4096	8196	26624	16384

④ 一般的な離散型sine変換の定義について

離散型sine変換及び，逆変換は一般的に(3.1)，(3.2)で定義される．

$$b_k = \frac{2}{n}\sum_{j=1}^{n-1} x_j \cdot \sin kj\theta, \quad k = 1,2,...,n-1 \tag{3.1}$$

$$x_j = \sum_{k=1}^{n-1} b_k \cdot \sin kj\theta, \quad j = 1,2,...,n-1 \tag{3.2}$$

本サブルーチンでは，(3.1)，(3.2)の左辺に対応して$\{2n \cdot b_k\}$又は，$\{4 \cdot x_j\}$を求める．したがって，結果の正規化は必要に応じて行うこと．

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≧512の場合．

```
C      **EXAMPLE**
       DIMENSION X(514),TAB(1028),VW(2560),
      *          IVW(1280)
    1  READ(5,500) N
       IF(N.EQ.0) STOP
       READ(5,501) (X(I),I=1,N)
C      SINE TRANSFORM
       WRITE(6,600) N
       WRITE(6,601) (X(I),I=1,N)
       CALL VSIN1(X,N,TAB,VW,IVW,ICON)
       IF(ICON.NE.0) GO TO 30
C      NORMALIZE
       CN=1.0/(2.0*FLOAT(N))
       DO 10 K=1,N
       X(K)=X(K)*CN
   10  CONTINUE
       WRITE(6,602)
       WRITE(6,601) (X(I),I=1,N)
C      SINE INVERSE TRANSFORM
       CALL VSIN1(X,N,TAB,VW,IVW,ICON)
       IF(ICON.NE.0) GO TO 30
C      NORMALIZE
       CN=0.25
       DO 20 K=1,N
       X(K)=X(K)*CN
   20  CONTINUE
       WRITE(6,602)
       WRITE(6,601) (X(I),I=1,N)
       GO TO 1
   30  WRITE(6,603) ICON
       GO TO 1
  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$n(=2^l,\ l = 1,2,...)$の離散型sine変換をベクトル計算機向きの高速変換法(FFT)により計算することを考える．

標本$\{x_j\}$，j = 0,1,...,n-1が与えられたときの離散型sine変換は，一般的に(4.1)で表される．

$$b_k = \frac{2}{n}\sum_{k=1}^{n-1} x_j \cdot \sin(kj\theta), \quad j = 0,1,...,n-1$$
$$, \quad \theta = \pi / n \tag{4.1}$$

ところで，標本は奇関数であり，1周期に拡大すると

$$x_{2n-j} = -x_j, \quad j = 0,1,...,n-1$$
$$\text{及び,} \quad x_0 = x_n = 0 \tag{4.2}$$

なる関係がある．したがって，x_0～x_{n-1}をx_0～x_{2n-1}に拡張し，項数$2n$の離散型実フーリエ変換を行えば，b_0～b_nを求めることができる．しかし，この場合(4.2)の対称性を利用すれば，効率よく変換が可能であることが知られている．

いま，標本$\{x_j\}$に，以下のような前処理を施す．

$$d_j = \frac{1}{2}\cdot\left(x_j - x_{n-j}\right) + \sin(j\theta)\cdot\left(x_j + x_{n-j}\right)$$
$$j = 0,1,...,n-1 \tag{4.3}$$

ここで，(4.3)に離散型sine逆変換(4.4)を代入すると(4.5)を得る．

$$b_k = \frac{2}{n}\sum_{k=1}^{n-1} x_j \cdot \sin(kj\theta), j = 0,1,...,n-1 \tag{4.4}$$

$$d_j = b_1 + \sum_{k=1}^{n/2-1} [(b_{2k+1} - b_{2k-1})\cdot\cos(2\cdot kj\theta) + b_{2k}\cdot\sin(2\cdot kj\theta)] - (-1)^j \cdot b_{n-1}$$
$$, j = 0,1,...,n-1 \tag{4.5}$$

(4.5)は，標本を$\{d_j\}$，フーリエ係数を$\{b_{2k+1}$-$b_{2k-1}\}$及び$\{b_{2k}\}$とするn項の離散型実フーリエ変換と等価ある．そこで，標本$\{d_j\}$に対するフーリエ係数$\{\tilde{a}_k\}$及び$\{\tilde{b}_k\}$を求めれば，恒等式

$$\tilde{a}_k = b_{2k+1} - b_{2k-1}$$
$$\tilde{b}_k = b_{2k}$$

より，$\{b_k\}$を得ることができる．
すなわち，$\{b_k\}$は(4.6)により求められる．

$$b_1 = \frac{1}{2}\cdot\tilde{a}_0, b_{n-1} = -\frac{1}{2}\cdot\tilde{a}_{n/2},$$
$$b_{2k} = \tilde{b}_k,$$
$$b_{2k+1} = b_{2k-1} + \tilde{a}_k, \quad k = 1,\ldots,n/2-1 \qquad (4.6)$$

ところで，(4.6)の最後の式は漸化式であり，ベクトル計算機には不向きである．そこで，本サブルーチンでは，離散型sine変換とその逆変換とが正規化定数を除くと同一の式であることを利用して，以上の計算式を逆にたどることにより漸化式の計算を回避し，ベクトル計算機に適したアルゴリズムを実現している．

なお，本アルゴリズムの詳細は，参考文献【8】を参照されたい．

A22－61－0202　　VSLDL, DVSLDL

正値対称行列のLDL^{T}分解
CALL VSLDL(A, N, EPSZ, VW, IVW, ICON)

(1)　機能

n×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．

$$\boldsymbol{A} = \boldsymbol{LDL}^T \qquad (1.1)$$

ただし，$\boldsymbol{L}$は単位下三角行列，$\boldsymbol{D}$は対角行列である．n≦1であること．

本サブルーチンは，サブルーチンSLDLと機能的に同じであるが，係数行列の格納方法が異なり，ベクトル計算機に適した方法を用いている．

(2)　パラメタ

A ……**入力**．係数行列$\boldsymbol{A}$．
出力．行列$\boldsymbol{L}$及び$\boldsymbol{D}^{-1}$．
大きさ$n(n+1)/2$の1次元配列．
格納の方法は，図VSLDL-1に示すように，対称行列の下三角部分を第1列から第n列まで列ごとに入力する．

N ……**入力**．係数行列$\boldsymbol{A}$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)

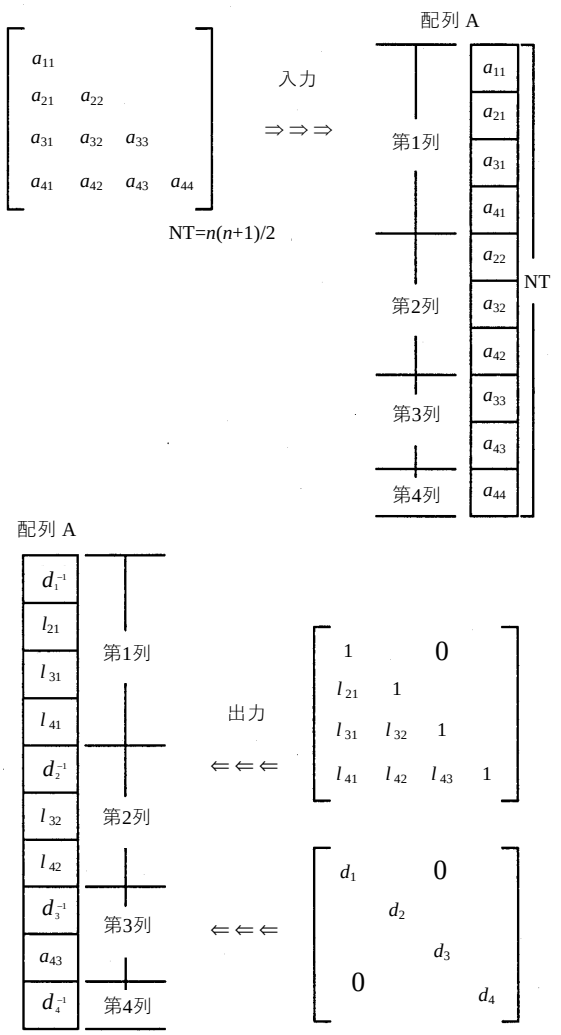
VW……**作業領域**．大きさ$2n$の1次元配列．

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VSLDL-1　参照．

表VSLDL-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，又はEPSZ＜0.0であった．	処理を打ち切る．



図VSLDL-1　対称行列の格納方法

(3)　使用上の注意

a.　使用する副プログラム

①　SSLII…AMACH，MGSSL

②　FORTRAN組込み関数…ABS

b.　注意

①　本サブルーチンは，サブルーチンSLDLにおける行列の格納方法を変えることにより，ベクトル計算機上での高速化を図ったものである．両サブルーチンの間で格納方法及び呼出し法が異なることに注意すること．

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}分解の過程でピボットの値が10進sけた以上のけた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，結果の精度は保証されない．

③ 分解過程でピボットが負となった場合，係数行列は正値でないことになる．このとき，lCON=10000とするが，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注意が必要である．

④ 係数行列の行列式の値を求めるには，演算後の配列Aのn固の対角線上の値($\boldsymbol{D}^{-1}$の対角要素)を掛け合わせ，その逆数をとればよい．

c. 使用例

$n\times n$の行列を入力し，LDL^{T}分解する．

$n\leqq 100$の場合．

```
C      **EXAMPLE**
       DIMENSION A(5050),VW(200),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)
       WRITE(6,630)
       IS=1
       IE=N
       DO 20 J=1,N
       WRITE(6,600) J,(A(I),I=IS,IE)
       IS=IE+1
   20  IE=IE+(N-J)
       CALL VSLDL(A,N,1.0E-6,VW,IVW,ICON)
       WRITE(6,610) ICON
       IF(ICON.GE.20000) GO TO 10
       WRITE(6,640)
       IS=1
       IE=N
       DET=1.0
       DO 30 J=1,N
       WRITE(6,600) J,(A(I),I=IS,IE)
       DET=DET*A(IS)
       IS=IE+1
   30  IE=IE+(N-J)
       DET=1.0/DET
       WRITE(6,620) DET
       GO TO 10
  500  FORMAT(I5)
  510  FORMAT(5E15.7)
  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

変形コレスキー法によるLDL^{T}分解は，サブルーチンSLDLの「手法概要」に述べてあるが，本サブルーチンでは，ベクトル計算機に適した手順を用いている．すなわち，分解処理の中核となる計算が，本質的には行列とベクトルの積で表現できることに着目し，これを効率良くベクトル処理する，というものである．

これに関連して，係数行列の格納方法が重要になる．ここでは，効率良くベクトル処理するため，係数行列の下三角部分を列ごとに格納する方法を用いている．

正値対称行列$\boldsymbol{A}$のLDL^{T}分解，

$$\boldsymbol{A}=\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{T} \tag{4.1}$$

において，$\tilde{\boldsymbol{L}}=\boldsymbol{L}\boldsymbol{D}$とする．$\boldsymbol{L}=(l_{ij})$，$\boldsymbol{D}=\mathrm{diag}(d_i)$とすると，$\tilde{\boldsymbol{L}}$は(4.2)の形をしている．

$$\tilde{L}=\begin{bmatrix} d_1 & & & & & 0 \\ l_{21}d_1 & d_2 & & & & \\ l_{31}d_1 & l_{32}d_2 & d_3 & & & \\ \cdot & \cdot & \cdot & \cdot & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \\ l_{n1}d_1 & l_{n2}d_2 & \cdot & \cdot & \cdot & d_n \end{bmatrix} \tag{4.2}$$

分解処理を進める過程で，本サブルーチンでは1次元配列**A**上に，係数行列$\boldsymbol{A}$，$\tilde{\boldsymbol{L}}$，$\boldsymbol{L}$及び$\boldsymbol{D}^{-1}$の，それぞれ部分的な要素を同居させ，最終的に$\boldsymbol{L}$と$\boldsymbol{D}^{-1}$の要素だけにする．

図VSLDL-2は分解の第r段階目(r=2,3,...,n)における配列**A**の内容を示したものである．図では，分かりやすくするため，配列Aを行列の下三角部分の形で書いてある．×印は既に求まっている$\boldsymbol{L}$の要素，*印は$\boldsymbol{D}^{-1}$の要素，○印は$\tilde{\boldsymbol{L}}$の要素，△印は係数行列$\boldsymbol{A}$の要素を，それぞれ表す($\boldsymbol{M}_r$，$\boldsymbol{a}_r$については以下の②で定義する)．

このとき，第r段階目では以下の計算を行う．

① 配列Aの第r行から，$\boldsymbol{L}$の第r行を決定する．これには次のように行う．配列Aの第r行は

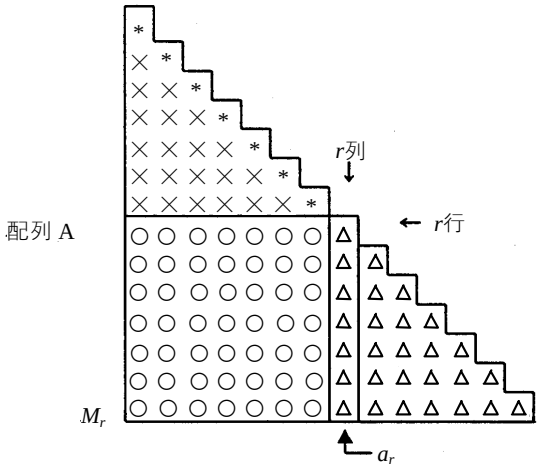
$$(l_{r1}d_1, l_{r2}d_2, \ldots\ldots\ldots, l_{r,r-1}d_{r-1}, a_{rr})$$

なる内容となっているので，l_{rj}は，

$$l_{rj}=(l_{rj}\,d_j)\,d_j^{-1},\ j=1,2,3,\ldots,\ r-1 \tag{4.3}$$

より簡単に求まる．これらの要素は，作業用の配列VWへ一時的に格納しておく．

② 配列Aの第r列を更新することにより$\tilde{\boldsymbol{L}}$の第r列を決定する．ここが本方法の中核部分であり，本質的に行列とベクトルの積の計算となる．



図VSLDL-2　配列Aの内容

準備として，いくつかの記号を導入する．まず，決定しようとする $\tilde{\boldsymbol{L}}$ の第r列ベクトルを $\tilde{\boldsymbol{l}}_r$ とする．すなわち，

$$\tilde{\boldsymbol{l}}_r = (d_r, l_{r+1,r}d_r, ..., l_{nr}d_r)^{\mathrm{T}} \tag{4.4}$$

である．次に，ベクトル$\boldsymbol{l}_r$，行列$\boldsymbol{M}_r$，及びベクトル$\boldsymbol{a}_r$を各々以下のように定義する．

$$\boldsymbol{l}_r = (l_{r1}, l_{r2}, \ldots\ldots, l_{r,r-1})^{\mathrm{T}} \tag{4.5}$$

$$\mathrm{M}_r = \begin{bmatrix} l_{r1}d_1 & \ldots\ldots\ldots\ldots & l_{r,r-1}d_{r-1} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ l_{n1}d_1 & \ldots\ldots\ldots\ldots & l_{n,r-1}d_{r-1} \end{bmatrix} \tag{4.6}$$

$$\boldsymbol{a}_r = (a_{rr}, a_{r+1,r}, ..., a_{nr})^{\mathrm{T}} \tag{4.7}$$

$\boldsymbol{l}r$は①で求めた$\boldsymbol{L}$の第r行ベクトル，$\boldsymbol{M}_r$は $\tilde{\boldsymbol{L}}$ の部分行列，$\boldsymbol{a}_r$は係数行列$\boldsymbol{A}$の第r列である(図VSLDL-2参照)．

このとき，関係，

$$\boldsymbol{a}_r = \left[\boldsymbol{M}_r : \tilde{\boldsymbol{l}}_r\right] \begin{bmatrix} \boldsymbol{l}_r \\ \ldots \\ 1 \end{bmatrix}$$

が成り立つことが分る．したがって，

$$\tilde{\boldsymbol{l}}_r = \boldsymbol{a}_r - \boldsymbol{M}_r \boldsymbol{l}_r \tag{4.8}$$

が得られ，$\tilde{\boldsymbol{l}}_r$ は，本質的に行列とベクトルの積から計算できる．これをベクトル計算機に適した手順で行う．

③　配列Aの第r行を$\boldsymbol{L}$の第r行に更新し，かつ，対角要素には d_r^{-1} を格納する．前者については，配列VWに一時保存しておいた $\{ l_{rj} ; j = 1, 2, ..., r-1\}$を配列Aの第$r$行に複写する．後者については，$d_r$が②で求めた $\tilde{\boldsymbol{l}}_r$ の第1要素となっているので，その逆数をとって格納すればよい．このとき，d_rは係数行列の正則性と正値性を決定するので，そのチェックをここで行う．

以上，第r段階目の手順を述べた．$r = 2, 3, ..., n$として①，②，③を繰り返せば，最終的に配列Aには行列$\boldsymbol{L}$の下三角部分(対角要素の1は除く)，及び対角行列$\boldsymbol{D}$の逆行列が得られる．

付　　録

付録1　SSLII拡張機能サブルーチン名一覧表

付1.1　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
VALU	A22-71-0202	AMACH
VCFT1	F16-15-0201	UVTB1, UVF91, UVFA1, UVFB1, UVFX1, UBANK
VCFT2	F16-15-0301	UVTB2, UVF92, UVFA2, UVFB2, UVFX2, UBANK
VCOS1	F16-11-0201	VRFT1, VCFT1, UVRFT UVTB1, UVF91, UVFA1, UVFB1, UVFX1, UBANK, UVTAB
VGSG2	B62-21-0201	GSCHL, TRID1, TRBK , GSBK , UVTG2, UCHLS, AMACH, UVBCT
VLAX	A22-71-0101	VALU, LUX, AMACH
VLDLX	A22-61-0302	
VLSX	A22-61-0101	AMACH, VSLDL, VLDLX
VLTX	A62-11-0101	AMACH,
VLTX1	A62-21-0101	AMACH,
VLTX2	A62-31-0101	AMACH,
VLTX3	A62-41-0101	AMACH,
VLUIV	A22-71-0602	
VMGGM	A61-11-0301	
VRFT1	F15-31-0201	VCFT1, UVTB1, UVF91, UVFA1, UVFB1, UVFX1 UBANK, UVRFT
VRFT2	F15-31-0301	VCFT2, UVTB2, UVF92, UVFA2, UVFB2, UVFX2, UVRFT, UBANK
サブルーチン名	分類コード	使用する副プログラム名
VSEG2	B61-21-0201	TRID1, UVTG2, TRBK, AMACH, UVBCT
VSIN1	F16-21-0201	VRFT1, VCFT1, UVRFT, UVTB1, UVF91, UVFA1, UVFB1, UVFX1, UBANK, UVTAB
VSLDL	A22-61-0202	AMACH

付1.2　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
UBANK	VCFT1, VRFT1, VCFT2, VRFT2, VCOS1, VSIN1
UVFA1	VCFT1, VRFT1, VCOS1, VSIN1
UVFA2	VCFT2, VRFT2
UVFB1	VCFT1, VRFT1, VCOS1, VSIN1
UVFB2	VCFT2, VRFT2
UVFX1	VCFT1, VRFT1, VCOS1, VSIN1
UVFX2	VCFT2, VRFT2
UVF91	VCFT1, VRFT1, VCOS1, VSIN1
UVF92	VCFT2, VRFT2
UVRFT	VRFT1, VRFT2, VCOS1, VSIN1
UVTAB	VCOS1, VSIN1
UVTB1	VCFT1, VRFT1, VCOS1, VSIN1
UVTB2	VCFT2, VRFT2
UVTG2	VGSG2, VSEG2

付録2　分類コードとサブルーチン名対応表

線型計算

分類コード	サブルーチン名
A22-61-0101	VLSX
A22-61-0202	VSLDL
A22-61-0302	VLDLX
A22-71-0101	VLAX
A22-71-0202	VALU
A22-71-0602	VLUIV
A61-11-0301	VMGGM
A62-11-0101	VLTX
A62-21-0101	VLTX1
A62-31-0101	VLTX2
A62-41-0101	VLTX3

固有値固有ベクトル

分類コード	サブルーチン名
B61-21-0201	VSEG2
B62-21-0201	VGSG2

フーリエ変換

分類コード	サブルーチン名
F15-31-0201	VRFT1
F15-31-0301	VRFT2
F16-11-0201	VCOS1
F16-21-0201	VSIN1
F16-15-0201	VCFT1
F16-15-0301	VCFT2

付録3　参考文献一覧表

【1】Stone,H.S.
Parallel Tridiagonal Equation Solvers,
ACM Trans. on Math. Soft., Vol.1,No.4,1975,
pp.289-307

【2】Temperton, C.
Fast Fourier Transforms and Poisson
solvers on CRAY-1.
INFOTECH,1979

【3】平岩健三
並列計算機により三項方程式を解くための
modified cyclic reduction algorithm,
情報処理学会論文誌，Vol.20，No.2，1979,
pp.190-193

【4】平岩健三
ビットベクトルとインダイレクトベクトル
を持つパイプライン計算機上のFFTのため
の新しい並列算法,
情報処理学会論文誌,Vol.21，No.2，1980

【5】三上次郎，伊奈博，秋田典伸，山下真一郎
ベクトル計算機におけるFFTアルゴリズム,
情報処理学会第28回全国大会，1984

【6】松浦俊彦，三浦謙一
スーパコンピュータのデータ編集機能と
FFTの高速化,
情報処理学会第28回全国大会，1984

【7】秋田典伸，三上次郎，伊奈博，山下真一郎
ベクトル計算機における三項方程式の解法
情報処理学会第28回全国大会，1984

【8】Swarztrauber, P.N.
Vectorizing the FFTs,
Parallel Computations,
Academic Press, 1982, PP.51-83

【9】Forsythe.G.E. and Moler,C.B.
Computer Solution of Linear Algebraic
Systems,
Prentice-Hall, Inc., 1967

【10】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

【11】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

【12】P.AMESTOY,M.DAYDE and I.DUFF
Use of computational kernels in the solution of
full and sparse linear equations M.COSNARD,
Y.ROBERT,Q.QUINTON and M.RAYNAL,
PARALLEL & DISTRIBUTED
ALGORITHMS,
North-Holland, 1989, pp.13-19

【13】島崎眞昭
スーパーコンピュータとプログラミング
共立出版株式会社，1989