Res. Rep. Fac. Eng. Mie Univ., Vol. 23, pp. 83−102 (1998)

論　文

ランチョス法による実対称行列用の対角化サブルーチンの作成

Eigenvalue Analysis for Real Symmetric Matrices by Lanczos Method

佐野和博
Kazuhiro SANO
(物理工学科　Department of Physics Engineering)

(Received September 14, 1998)

Abstract

Eigenvalue problems for very large sparse matrices based on the Lanczos' minimized iteration method are examined. A detailed analysis shows that false eigenvalues which rarely appear in the Lanczos method are unstable for iteration procedure. It leads us to an improved Lanczos method which can exclude the false eigenvalues and pick out more eigenvalues than the ordinary Lanczos method. A sample program of this method and some numerical results are presented.

Key words: Eigenvalue problem, Lanczos Method, Large sparse matrix

1．序論

理工系の分野では、計算機を用いて行列を対角化して固有値、固有ベクトルを求める事がしばしばある。扱いたい行列があまり大きくは時は、対角化の仕方には標準的な手法がある。これらはたいがいの計算機センターでライブラリープログラムとして利用でき、計算機上で利用しやすいようにサブルーチンの形として組み込まれている。このような既存のルーチンが使用できれば何も問題は生じないない訳であるが、問題としている行列がある程度以上大きくなると、既存のものでは間に合わなくなる事が多い。標準的な手法では、行列がある程度以上大きくなるとメモリーや計算時間がかかり過ぎて大型計算機をもってしても計算出来なくなってくるのである。

このような時には、幾つかの制限があるがランチョス法と呼ばれる方法が有効である。ただ単純なランチョス法では、たまに偽の固有値が出るという重大な欠点があり、多くの長所があるのにもかかわらず、一般的に使うには問題点が大きかった。一般に行列が大きくなればなるほど、その対角化は難しい問題となりプログラムの作成にも特別な工夫が必要となってくる。

本研究では、ランチョス法の問題点である偽の固有値の処理に工夫をこらして、一般向けに比較的使いやすい形でサブルーチン[1]を作成したので、それを紹介したいと思う。

2．ランチョス法の計算法と特徴

行列の対角化のため良く使われるハウスホルダー法と呼ばれる標準的な手法では、n行×n列の大きさの行列を対角化する際必要な配列の数はnの2乗程度に比例し、計算時間はnの3乗程度に比例する。そこで、大規模な行列（ここではnとして数千程度以上を想定している）を対角化しようとすると、非常に多くのメモリーと長時間の計算時間が必要となってくる。

近年、計算機のメモリーや計算速度はめざましく改善されているが、それでも大規模な行列の対角

化計算に対しては、まだまだ力不足である。大型計算機を使うとなると、一個人の使えるｃｐｕ時間も限られてくる。従って現状ではハウスホルダー法で扱える行列の大きさとしては、ｎが数千程度が現実的な限界となっている。

今ハウスホルダー法では扱えないような大きな行列をどうしても対角化したいとしよう。扱いたい行列が疎行列でかつ必要な固有値及び固有ベクトルが小さい方から（大きい方から）数個程度であったとすると、一般に反復法が有効である。反復法では、与えられた行列を書き換えなくても済むため、行列の要素がゼロでない所のみをメモリーに格納するだけで計算が可能となる。従って行列が疎行列であれば、全要素分のメモリーを必要とするルーチンと比べて、より大きな行列を対角化することが出来るわけである。また計算時間もハウスホルダー法より短くすることが可能である。

一口に反復法と言っても色々な手法が知られているが、最も単純な反復法では、勝手に取った初期ベクトルに対角化したい行列を何回も掛けることにより、初期ベクトルを固有ベクトルへと近づける方法が取られる。この方法は単純で解りやすいが、あまり効率的とは言えない。また固有値を複数個求めようとするとかなり面倒な操作が必要となる。反復法を改良して、もっとも効率的に対角化が行なえるようにしたものが、以下に述べるランチョス法である[1,2]。

今解きたい行列をAとし、AはN次実対称行列とする。互いに直交する適当な列ベクトルの組をV_n,(n=1,2,...,N)とする。次の関係式により定まるα_n（非対角項）とβ_n（対角項）を要素に持つ三重対角化行列Ｔを作るとする。

$$\alpha_n = V_n^{\dagger} A V_n, \qquad \beta_n = | A V_n - \beta_{n-1} V_{n-1} - \alpha_n V_n |$$

ここで，V_nは

$$V_n = (A V_{n-1} - \beta_{n-2} V_{n-2} - \alpha_{n-1} V_{n-1}) / \beta_{n-1}$$

で決まる。ただし、$\beta_0 V_0 = 0$で、V_1は任意にとることが出来る。これらからV_n、α_n、及びβ_nを順次定めることが可能である。

この時、少数個の固有値、固有ベクトルを求めるだけで良いとすると３重対角化を最後まで実行せず、途中で打ち切る事が出来る。そのため計算時間が少なくて済むことになる。この３重対角化された行列Ｔを既存のサブルーチンを用いて対角化し固有値及び固有ベクトルを求める。この時得られた固有値は、そのまま与えられた行列の固有値となり、固有ベクトルの方はランチョス法の反復時に得られるベクトルV_iを用いて、与えられた行列の固有ベクトルへと変換する。

ランチョス法では、ごく少数個の固有値を求めることに限れば対角化に必要な計算時間は行列の零ではない要素数にほぼ比例することになる。例えば、n次の行列を考え、各行に平均m個の零でない要素があるとすると、計算時間はn×mに比例する事になる。従って、ハウスホルダー法のようなnの３乗に比例する手法に比べて、大きな行列では有利になると考えられる。密行列の場合でも、計算時間はnの２乗に比例するので、大きな行列ではランチョス法のほうが計算時間が短くなる。

このように大きな行列では、ランチョス法が有利であるが、ただ問題点が幾つかある。例えば、３重対角化の過程には数値的な誤差が溜まりやすくnの大きな所ではV_nの直交性が崩れてしまうことが知られている。その為数個以上の固有値や固有ベクトルを求めようとしても、十分な精度が得られないことが多い。また、初めに与える初期ベクトルと直交する固有ベクトル（及びそれに対応する固有値）は計算されないという点、縮退している場合縮重度がわからないなどの問題点があげられる。

これらの欠点は、ランチョス法に限らず反復法全体に共通するものであり、ハウスホルダー法ではこのような欠点はなく安定している。従って行列があまり大きくない時はハウスホルダー法が一般に使われている。さらに単純なランチョス法では、たまに偽の固有値が出るという重大な欠点があり、多くの長所があるのにもかかわらず、ランチョス法はあまり一般的には使われておらず、ライブラリープログラムとしても見かける事はほとんどなかった。ただ大きな行列を扱うには反復法に頼らざる得ないので、必要に迫られた時に自家製のルーチンを作成しているのが現状と考えられる。

３．ランチョス法の改善とサブルーチンの性能評価

上記で述べたランチョス法の問題点の解決のため、再直交化の方法など幾つかの方法[3,4]が提案されている。しかしながら、これらの方法を大きな行列に用いると、少ないメモリーで、かつ少ない計算時間で対角化出来るというランチョス法の長所が損なわれる問題点があり、実用化には問題があった。ここでは、ランチョス法の長所を最大限生かしながら、出来る限り上記に述べた欠点を取り除く事を目標にしてサブルーチンの作成を行った。

まず偽の固有値が出る原因を探るため実際の行列をランチョス法で対角化し、求められた固有値の振る舞いを注意深く分析してみた。

ランチョス法の反復を繰り返して固有値の振る舞いを見てみると、真の固有値は安定であるにもか

かわらず、偽の固有値は反復に対し不安定であることがわかった。さらに、偽の固有値は反復を繰り返していくと、真の固有値に収束しその代わり新たな偽の固有値が生じてくることがわかった。反復をどんどん繰り返せば上記の繰り返しとなる。

これは、偽の固有値の起源が直交性の崩れによるものであることを示唆している。なぜなら反復を繰り返せば直交性の崩れ方が異なることになり、それに連れて偽の固有値も変化するからである。また真の固有値は直交性の崩れに影響されないことが知られているが、この事により、偽の固有値が真の固有値に収束することが理解できる。

これらの性質を利用すれば、偽の固有値の排除が可能となる。具体的には、通常のランチョス法より反復回数を若干（数倍程度以内で良い）多くして、固有値の振る舞いをモニターし、不安定な振る舞いをする固有値を除く事により真の固有値だけを取り出すわけである。この方法によれば、必要とするメモリーの大きさは、単純なランチョス法と変わらず、また計算時間も数倍程以内程度で収まり、ランチョス法の長所を十分生かす事が出来る。また、反復回数が多い分だけ通常のランチョス法よりも、より多数の固有値を取り出す事も可能となった。

このサブルーチンの性能の目安を得るため一例として、密行列（行列要素はすべて一様乱数により作成）の固有値を代数的な意味で小さい方から１０個汎用機で実際に計算してみた。すると、大体700×700程度の大きさの所で、ランチョス法と、ハウスホルダー法（NUMPACKの中にあるサブルーチンを使用）によるものとの計算時間が同じになる事がわかった。これより大きな行列ではランチョス法の方が速く計算でき、また疎な行列ではさらに早く対角化出来る事も確かめた。

なお問題点としては、数個以上の固有ベクトルまで求めようとすると、数値誤差のためランチョス法だけでは十分な精度があがらないという点がある。精度を上げるため必要に応じて逆反復法を併用しているが、そこで計算時間が掛かり、同じ行列で固有値だけを求めるのに比べ、かなりの計算時間がかかることになる。もっとも計算時間はnの２乗に比例するので、行列の大きさを大きくすれば、いずれはハウスホルダー法のサブルーチンよりは早くなる。両者のサブルーチンにより、上記の密行列で１０個固有ベクトルを求めると、ここで作成したルーチンが速くなるのは2万×2万程度以上の行列の時と見積もられた。また固有値を多く求めようとすればするだけ、余計に計算時間がかかり、数十個以上の固有値を求めようとする時などは、場合によって収束しないことがある。これらの問題点は反復法一般の限界の可能性もあるが、今後の課題としたい。

4．まとめ

ランチョス法の欠点であった偽の固有値が出るという問題点を改善して、大規模な行列を対角化するための実用的な固有値解析のサブルーチンを作成した。少ない固有値だけを求めるのであれば、従来からのハウスホルダー法のルーチンと比較して、行列の大きさが数百×数百程度より大きい所で、ランチョス法のサブルーチンのほうが有利になることが解った。このサブルーチンを使えば、疎な行列であれば数万×数万程度以上の巨大な行列が、普通のワークステーション等でも実用的なメモリー容量（数十から数百メガバイト）と計算時間（数十分から数時間）の範囲内で対角化可能である。

なお本研究で開発されたプログラムは、１９９４年度から１９９７年度末までの間、名古屋大学計算機センターのライブラリープログラム開発課題の一つとして作成されたものである。現在、ライブラリープログラムの一つとして公開され、利用可能である[5]。プログラムの作成にあたっては名古屋大学計算機センターの方にいろいろお世話になりました。感謝いたします。

参考文献

1) C. Lanczos, An iteration method for the solution of eigenvalue problem of linear differential and integral operators, J, Res, Nat. Bur. Stand., 45,(1950)255.

2)戸川隼人；マトリクスの数値計算、オーム社（1971）：田口善弘、西森秀稔、量子スピン系の基底状態を有限系（N≦20）について計算するプログラム、物性研究45,(1986)299.：西森秀稔、量子スピン系の対角化プログラムTITPACK Ver.2、物性研究56,(1991)494.

3) H. Takahasi and M. Natori, Eigenvalue problem of large sparse matrices、 Rep. Compt. Centre, Univ. Tokyo, 4(1971-1972)129

4)名取　亮、　野寺　隆、ランチョス法その後、京都大学数理解析研究所数理科学講究録　５８５「並列数値計算アルゴリズムとその周辺研究集会報告集」（１９８６）275

5）佐野和博、LANVOD、　名古屋大学計算機センターライブラリープログラム（１９９７）

5. 付録（サブルーチンの使用説明とソースコード）

サブルーチン名 LANVOD 言語；FORTORAN サイズ 729行

（1） 概要

このサブルーチンは反復法の一種であるランチョス法を用いて、大規模な実対称行列（なるべく疎な行列が良い）を対角化し固有値及び固有ベクトルを固有値の代数的な意味で小さい方から（又は大きい方から）ごく少数個求めるためのものである。

大きな行列の固有値が、少数個に限られるが、比較的効率良く求められる。また行列の要素がゼロでない所のみをメモリーに格納する方式のため、疎な行列であれば必要なメモリーが少なくて済み計算時間も短かくなる。ただ、初めに与える初期ベクトルと直交する固有ベクトル（及びそれに対応する固有値）は計算されないという点、縮退している場合縮重度がわからないなどの欠点があるので、注意されたい。LANVODは倍精度用のルーチンのみである。

サブルーチンの内部で三重対角化された行列の固有値及び固有ベクトルを求めるために富士通のサブルーチンライブラリーの中のDTEIG2を呼び出している。ワークステーション等で使用する際は同等のサブルーチンを用意する必要がある。ただこれは普通のハウスホルダー法のルーチンで十分間に合うので、各自で用意されたい。なお、使用手引き書は名古屋大学計算機センターでも閲覧できる。

（2） 使用法

```
CALL LANVOD(MTMAX,MT,MAXMTE,MTE,ICON,IEN ,IU,AU,V1,V2,V3,X,E,EPS)
```

引数　型と種類　属性　内容

MTMAX　整数型　入力　対角化したい実対称行列の最大次数

MT　整数型　入力　対角化したい実対称行列の次数。MT<=MTMAX

MAXMTE　整数型　入力　対角化したい実対称行列の最大の要素数

MTE　整数型　入力　対角化したい実対称行列の要素数

ICON　整数型　入出力　入力として、ICON=-1,又は1のいずれかの値を入れる。符号に応じて固有値を代数的な意味で小さい方（符号が負の時）から、または大きい方（符号が正の時）から順に求める。また固有値のみを計算し固有ベクトルを計算しない時は、ICONの入力として-10又は10を入れる。上で指定した以外の値を入れると異常終了する。

出力として、正の値のときは、正常に終了した場合で、収束するまでの反復回数を出力する。負の値のときは、以下のように異常終了を表す。

（ICON=-9999：初期ベクトルが制限を破った。

ICON=-999：引数ICONが制限を破った。

ICON=-99：ICON以外の引数が制限を破った。

ICON=-9：このルーチンでは処理出来ない行列が入力されたか、又は行列の入力が誤っている可能性がある。

ICON=-3：β_1（本文を参照の事）が零になったため三重対角化が出来なかった。この場合行列の入力が誤っている可能性がある。

ICON=-30：β_n(n>1)が零になったため三重対角化が出来なかった。この場合初期ベクトルを変えるとうまくいく可能性がある。

ICON=-300：固有ベクトルが求まらなかった。

ICON=-10000：三重対角化が収束しなかった。この場合、求める固有値の数IENが大きすぎる場合が考えられる。また初期ベクトルを変えるとうまくいくことがある。）

IEN　整数型　入力　必要な固有値、固有ベクトルの数。代数的な意味で小さい方から（大きい方から）求る。IENとして数十以上の大きな値を入れると収束が急に遅くなり、場合によっては収束しないことがある。

IU　整数型　一次元配列　入力　実対称行列の対角線を含む右上半分の零でない行列要素の位置を指定する。各行順に、右上半分の零でない要素でかつ、もっとも左にある要素の位置（列の番号）

を順番に配列に入れていく。ただし、各行の最初の要素が位置する列番号（右上半分の零でない行列要素であることに注意）には、その要素が行の先頭に位置する事を表すためマイナス符号をつける。

なお行列の各行は、すべての要素が零であってはいけない。

AU　実数型　一次元配列　　入力　　実対称行列の対角線を含む右上半分の零でない行列要素を配列に格納する。各行順に、もっとも左にある要素から順に、IUと連動して配列に入れていく。行列の各行は、すべての要素が零であってはいけない。AU及びIUは保存されている。

以下に、行列の格納方法の一例を示す。

```
|a11,a12,   0,   0,a15|              |a11,a12,   0,   0,a15|
|a21,a22,   0,   0,   0|   右上半分   |    ,a22,   0,   0,   0|
|  0,   0,   0,a34,   0|     ===>     |          ,0,a34,   0|
|  0,   0,a43,a44,   0|              |            ,a44,   0|
|a51,   0,   0,   0,   0|              |                  ,0|
```

```
IU(1)=-1, IU(2)=2,   IU(3)=5,   IU(4)=-2, IU(5)=-4,IU(6)=-4
AU(1)=a11,AU(2)=a12,AU(3)=a15,AU(4)=a22,AU(5)=a34,AU(6)=a44,
```

V1　実数型　一次元配列　入力及び作業領域　大きさMTMAXの一次元配列。ノルムを１に規格化した初期ベクトル又はゼロベクトルを入れる。これら以外の値を入れると異常終了となる。ゼロベクトルを入れた場合、ルーチン内で疑似乱数により作り出したベクトルが、初期ベクトルとして入る。この初期ベクトルは、ルーチンを呼び出すごとに変化する。

なお初期ベクトルと直交する固有ベクトル及びそれに対応する固有値は抜け落ちるので注意されたい。良い初期ベクトルを用いる事が出来れば反復回数が減り精度もあげる事が出来る。

V2　実数型　一次元配列　作業領域　大きさMTMAXの一次元配列

V3　実数型　一次元配列　作業領域　大きさMTMAXの一次元配列

X　　実数型　２次元配列　出力　　それぞれの固有値に対応する固有ベクトルが入る。第一添字の値はMTMAX、第２添字の値はIENである。固有値だけを求めるときには使用しないので、その時は何を書いてもよい。

E　　実数型　一次元配列　固有値が入る。配列の大きさはIENである。固有値は縮重している場合でも、縮重度が分からないので縮重していないときと同様１個の固有値として入る。

EPS　実数型　　入力　　収束判定定数　固有値に対しては

||E||･EPSにより収束を判定し、固有ベクトルに対しては、||X||･EPSにより収束を判定している。EPSとして10^{-15}以下の値を入れても10^{-15}として処理される。求まる固有値の精度は、行列や初期ベクトルにもよるが10^{-13}程度である。EPSやルーチンの精度を越えて、近接した固有値がある場合、固有値を分離することが出来ない。またルーチンの精度を越えてEPSを小さくすると、偽の固有値が現れることがある。

```
      SUBROUTINE LANV0D(MTMAX,MT,MAXMTE,MTE,
     &                  ICON,IEN  ,IU,AU,V1,V2,V3,X,ES,EPS)
C#
C#  LANV0D.. LANCZS--TRIDIG2(SLL2)   .....EIGEN VALUES
C#               |
C#             LANV1                .....EIGEN VECTOR
C#               |
C#              CHK3   <------]      .....CHECK EIGEN VECTOR
C#            ___|____        |
C#           |        |       |
C#        (IF OK)  (IF NO)--> CG     .....IMPROVE EIGEN VECTOR
C#           |
C#           |
C#          END
C#
C#
C    MTMAX          % MAXIMUM DIMENSION OF MATRIX
C    MT             % DIMENSION OF MATRIX
C    MAXMTE         % MAXIMUM NUMBER OF  ELEMENT
C    MTE            % NUMBER OF  ELEMENT
C    IEN            % NUMBER OF EIGEN-VALUES TO BE EXCUTED
C    IVN100         % THE FIRST NUMBER OF WORKING AREA
C    ICON           % INPUT(OUTPUT) CONDITION NUMBER
C    IU(MTE)        % LOCATION OF NONZERO ELEMENT OF THE MATRIX
C    AU(MTE)        % NONZERO ELEMENT OF THE MATRIX
C    V1(MT)         % INPUT INITAL VECTOR & WORKING AREA
C    V2(MT),V3(MT)  % WORKING AREA
C    X(MT,IEN)      % EIGEN VECTORS
C    E(IEV)         % EIGENVALUES
C    EV(IVN100,IEV) % WORKING AREA FOR EIGEN VECTOR
C#
C# ICON:  POSITIVE(NEGATIVE).....MAXIMUM (MINMUM) EIGNVALUE
C# ICON:  10000....MESSEGE ON    1000....EIGEN VECTOR PASS
C#        100......INVERTH  ITERATION METHOD PASS
C#        10.......DIVIDE AU(MAXMTE) BY 2   PASS
C#
C#
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER( IVN100= 400,IEV=500,IVN10=200)
      DIMENSION  V1(MTMAX),V2(MTMAX),V3(MTMAX), ES(IEN),E(IEV)
      DIMENSION    X(MTMAX,IEN) ,EV(IVN100,IEV)
      INTEGER*4  IU(MAXMTE)
      DIMENSION AU(MAXMTE)
      COMMON/COM1/ IESET(IVN10),ENAMA(IVN100)

C......INITAL CHECK .............

         IF(EPS.LE.1.0D-15) EPS=1.0D-15


             IVSIGN=SIGN(1,ICON)
```

```
          MES=ABS(ICON/10000)
        IAUPAS=ABS(MOD(ICON,10000)/1000)
        ICGPAS=ABS(MOD(ICON,1000 )/100 )
        IEIPAS=ABS(MOD(ICON,100  )/10  )

      ICONAB=ABS(ICON)
    IF(ICONAB.EQ.1.OR.ICONAB.EQ.10.OR.ICONAB.EQ.10010.OR
&    .ICONAB.EQ.10001)THEN
           ELSE
       ICON=-999
       IF(MES.EQ.1) WRITE(*,*) 'CHECK ICON  !!!!!'
           RETURN
      END IF

    IF( (MES.EQ.0.OR.MES.EQ.1).AND.(ICGPAS.EQ.0.OR.ICGPAS.EQ.1)
&   .AND.(IAUPAS.EQ.0.OR.IAUPAS.EQ.9)  ) THEN
           ELSE
       ICON=-999
                WRITE(*,*) 'CHECK ICON  !!!!!'
           RETURN
    END IF


    IF(MTMAX.LT.MT.OR.MAXMTE.LT.MTE) THEN
         ICON=-1
         IF(MES.EQ.1) WRITE(*,*) 'CHECK MTMAX & MAXMTE '
         RETURN
    END IF


    IF(MT.GT.MTE+1) THEN
         ICON=-1
         IF(MES.EQ.1) WRITE(*,*) 'CHECK MT & MTE '
         RETURN
    END IF

       MTCHK=0
       MTCHK2=0
       MTCHK3=1
    DO  2 I=1,MTE
    MTCHK=MAX(ABS(IU(I)),MTCHK)
    IF(IU(I).EQ.0)   MTCHK2=1
    IF(IU(I).LT.0)   MTCHK3=0
2   CONTINUE

    IF(MTCHK.NE.MT)  THEN
         ICON=-1
         IF(MES.EQ.1) WRITE(*,*) 'CHECK MT & MTE & IU(MTE)'
         RETURN
    END IF
```

```
      IF(MTCHK2.EQ.1)  THEN
           ICON=-1
           IF(MES.EQ.1) WRITE(*,*) 'IU(MTE) CONTAINS zero !!! '
           RETURN
      END IF

      IF(MTCHK3.EQ.1)  THEN
           ICON=-1
           IF(MES.EQ.1) WRITE(*,*) 'IU(MTE) IS INVALID  '
           RETURN
      END IF



      IVECTN=IEN*5
      V1CHEK=0.0D0
      DO 10 I=1,MT
10    V1CHEK=V1CHEK+V1(I)**2
      IF(SQRT(V1CHEK).LE.1.0D-9) THEN

C  #  RANDOM  INITAL VECTOR  #

            IM=2**31
      DO 20 I=1,MT
           IRAN=MOD(32771*IRAN+1234567891,IM)
20         V1(I)=(IRAN*1.00D0)/IM-0.5D0

           V1NN=0.0D0
      DO 22 I=1,MT
22         V1NN=V1NN+V1(I)**2
           V1NN=SQRT(V1NN)
      DO 23 I=1,MT
23         V1(I)=V1(I)/V1NN


      ELSE
           IF( ABS( SQRT(V1CHEK)-1.0D0 ).GE.1.0D-7) THEN
                  ICON=-1
                  IF(MES.EQ.1) WRITE(*,*) 'CHECK INITAL VECTOR  '
                  RETURN
           END  IF

      END  IF

            IF(IEIPAS.NE.1)  THEN
               DO 30 I=1,MT
30              X(I,1)=V1(I)
            END  IF


C.... DIVIDE DIAGONAL ELEMENT OF AU(J) BY 2
```

```
C     BECAUSE OF DOUBLE COUNTING IN    DO LOOP
C                              ........  AU(J)*V(ABS(IU(J)) ........

          IF(IAUPAS.EQ.0)  THEN
            I=0
      DO 40 J=1,MTE
      I=I+(1-SIGN(1,IU(J)))/2
      IF(ABS(IU(J)).EQ.I) AU(J)=AU(J)/2.0D0
 40   CONTINUE
        END  IF

C..... EIGEN VALUES ................

            ICON=IVSIGN

        CALL  LANCZS(MTMAX,MT,MAXMTE,MTE,IVECTN,ICON,IEN,IVN100,IEV,
     &    IU,AU,V1,V2,V3,  E,  EV,EPS)

          DO 35 J=1,IEN
35        ES(J)=E(J)

        IF(ICON.EQ.-3) THEN
             IF(MES.EQ.1) WRITE(*,*)'TRIDIAGONALIZATION FAILLING B1=0'
             RETURN
        END  IF


        IF(ICON.EQ.-30) THEN
             IF(MES.EQ.1) WRITE(*,*)'TRIDIAGONALIZATION FAILLING BN=0'
             RETURN
        END  IF


        IF(ICON.EQ.-1000) THEN
        IF(MES.EQ.1) WRITE(*,*)'INCREASE PARAMETER(IVN100) IN LANCZS'
                   RETURN
        END  IF


        IF(ICON.EQ.-10000) THEN
             IF(MES.EQ.1) WRITE(*,*) 'LANCZS DONT CONVERGE  '
                  RETURN
        END  IF


      IF(IEIPAS.NE.1)  THEN

C      WRITE(*,*) 'ICON=',ICON
       ICONV=ICON
       IF(ICON.GE.300)  IVONV=300
```

```
C..... EIGEN VECTORS ..............

        CALL  LANV1(MTMAX,MT,MAXMTE,MTE,IEN,IEV,ICONV,IVN100
     &                                        ,IU,AU,V1,V2,V3,X,EV)

       DO 100 IV=1,IEN

      CALL  CHK3(MTMAX,MT,MAXMTE,MTE,IEN,IV,IU,AU,X,V2,V3,EE,EPSCHK)
C
       IF(EPSCHK.GT.EPS.AND.ICGPAS.NE.1)  THEN

C   ##  INVERSE ITERATION   METHOD   WITH  CG    ##

              DO 140 ICG =1,100

      CALL CG(MTMAX,MT,MAXMTE,MTE,IEN,IV
     &                     ,ICON,IU,AU,X,E,V1,V2,V3,EPS)

        IF(ICON.EQ.-300) THEN
             IF(MES.EQ.1) THEN
                WRITE(*,*) 'CG DONT CONVERGE  '
                WRITE(*,*) 'I=',IV,'EPSCHK= ',EPSCHK
             END  IF
             RETURN
        END  IF

      CALL  CHK3(MTMAX,MT,MAXMTE,MTE,IEN,IV,IU,AU,X,V2,V3,EE,EPSCHK)

                  IF(EPSCHK.LE.EPS) GOTO 110

 140          CONTINUE

         ICON=IV
        IF(MES.EQ.1)  THEN
         WRITE(*,*) 'EIGEN VECTOR DONT CONVERGE !!!!    No=',IV
         WRITE(*,*) 'EPSCHK= ',EPSCHK
        END  IF

                  END IF

 110          CONTINUE

         EECHK=ABS(E(IV)-EE)
         EV(1,IV)=EE


       IF(EPSCHK.GT.EPS)  THEN

           ICON=IV
```

```
                    IF(MES.EQ.1)  THEN
            WRITE(*,*) 'CHECK EIGEN VALUE & VECTOR !!!!!  No=',IV
C           WRITE(*,*) '<X AU X>=',EE,'   ZANSA=',EPSCHK
                    END  IF
C           RETURN
        END  IF

100      CONTINUE

        END  IF
C.....  REMAKE   AU(J) .......

          IF(IAUPAS.EQ.0)  THEN
            I=0
       DO 50 J=1,MTE
       I=I+(1-SIGN(1,IU(J)))/2
       IF(ABS(IU(J)).EQ.I) AU(J)=AU(J)*2.0D0
 50    CONTINUE
          END  IF


                    END


C********** lanczos  method                              ********

      SUBROUTINE LANCZS(MTMAX,MT,MAXMTE,MTE,IVECTN,ICON,IEN,IVN100,IEV
     &                                        ,IU,AU,V1,V2,V3,E,EV,EPS)

C    MTMAX          % MAXIMUM DIMENSION OF MATRIX
C    MT             % DIMENSION OF MATRIX
C    MAXMTE         % MAXIMUM NUMBER OF  ELEMENT
C    IVECTN         % NUMBER OF EIGEN-VALUES TO BE EXCUTED
C    IVN100         % THE FIRST NUMBER OF WORKING AREA
C    ICON           % INPUT(OUTPUT) CONDITION NUMBER
C    IEN            % INPUT(OUTPUT) CONDITION NUMBER2
C    IU(MTE)        % LOCATION OF NONZERO ELEMENT OF THE MATRIX
C    AU(MTE)        % NONZERO ELEMENT OF THE MATRIX
C    V1(MT)         % INPUT INITAL VECTOR & WORKING AREA
C    V2(MT),V3(MT)  % WORKING AREA
C    X(MT,IEN)      % EIGEN VECTORS
C    E(IVECTN)      % EIGENVALUES
C EV(IVN100,IVECTN) % WORKING AREA FOR EIGEN VECTOR
C    ISTEP          # INTERVAL TO CHECK CONVERGENCE


      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (ISTEP= 8,  IV2=1, IA=1000, IVN10=200)
      DIMENSION E(IVECTN) ,EOLD(IA)
      DIMENSION V1(MTMAX),V2(MTMAX),V3(MTMAX)
      INTEGER*4 IU(MAXMTE)
```

```
      DIMENSION          AU(MAXMTE)
      DIMENSION ALPHA(IA),BETA(IA+1),WK(5*IA)
      DIMENSION EV(IVN100,IEV)
      COMMON/COM1/ IESET(IVN10),ENAMA(IA)



C*** INITIALIZATION

      IS=ICON
      DO  5 I=1,IVN100
       ALPHA(I)=0.0D0
        BETA(I)=0.0D0
 5    CONTINUE

      DO 10 I=1,MT
          V2(I)=0.0D0
          V3(I)=0.0D0
10    CONTINUE


C*** ALPHA(1) AND BETA(2)
      CALL     AX(MTMAX,MAXMTE,MTE,IU,AU,V2,V1)


      ALPHA(1)=0.0D0
      DO 15 I=1,MT
15    ALPHA(1)=ALPHA(1)+V1(I)*V2(I)

      BETA1=0.0D0
      DO 50 I=1,MT
      V2(I)=V2(I)-ALPHA(1)*V1(I)
50    BETA1=BETA1+V2(I)**2
      BETA(2)=SQRT(BETA1)
      IF(BETA(2).LT.0.5D-20)THEN
C       WRITE(*,*) 'LANCZS DE BETA1 GA ZERO ]]]]] '
            ICON=-3
            RETURN
        ELSE
          DO 65 I=1,MT
65        V2(I)=V2(I)/BETA(2)
      END IF

C*** ITERATION
      ET=1.0D+30
      IC=0

      DO 100 I=2,IVN100

        IF(I.GT.IVN100) THEN
```

```
C          WRITE(*,*)' ITERATION NUMBER OVER !!! '
           ICON=-1000
           RETURN
          END  IF

          DO 105 II=1,MT
 105      V3(II)=0.0D0

C        DO 11 JJJ=1,MTE

       CALL     AX(MTMAX,MAXMTE,MTE,IU,AU,V3,V2)

          ALPHAI=0.0D0
          DO 120 J=1,MT
 120      ALPHAI=ALPHAI+V2(J)*V3(J)

          ALPHA(I)=ALPHAI
          BETAI1=BETA(I)
          DO 130 J=1,MT
 130      V3(J)=V3(J)-ALPHAI*V2(J)-BETAI1*V1(J)

          BETAI=0.0D0
          DO 140 J=1,MT
              V1(J)=V2(J)
              BETAI=BETAI+V3(J)**2
 140      CONTINUE
          BETA(I+1)=SQRT(BETAI)
          IF(BETA(I+1).LT.0.5D-20)THEN
C           WRITE(*,*)' TRIDIAGONALIZATION UNSUCCESSFUL IN LANCZS'
            ICON=-30
            RETURN
          END IF

          IF(I.GT.15.AND.MOD(I,ISTEP).EQ.0)THEN
                IF(I.GT.IVECTN)      THEN
                     IMM=IVECTN
                ELSE
                     IMM=I
                END  IF
               CALL DTEIG2(ALPHA,BETA,I,IS*IMM,E,EV,IVN100,WK,IER)
                ICON=I
                IF( ABS(EBEFOR-E(IV2)) .LT.EPS*ABS(E(IV2)) ) THEN
C
             DO 220   IIII=1,IMM
 220         ENAMA(IIII)=E(IIII)

             EOLD(1)=E(1)
                 ICON2=IEN
           CALL EIGENS(IVECTN,ICON2,IC,IMM,E,EOLD,EPS,ET)

             IF(ICON2.GE.IEN) THEN
```

```
                    DO 240 ISELC1=1,IEN
                    SABUN=1000000000.0
                    DO 240 ISELC2=1,IMM
                    IF( ABS(ENAMA(ISELC2)-E(ISELC1)).LT.SABUN ) THEN
                       SABUN=ABS(ENAMA(ISELC2)-E(ISELC1))
                       IESET(ISELC1)=ISELC2
                    END IF
 240                CONTINUE
             RETURN
             END  IF
C
          END  IF

             DO 210   IIII=1,IMM
             EOLD(IIII)=E(IIII)
 210         CONTINUE
            EBEFOR=E(IV2)
          END IF
C
          RB=1.0D0/BETA(I+1)
          DO 150 J=1,MT
 150      V2(J)=V3(J)*RB
          IF(I.EQ.15)THEN
         CALL DTEIG2(ALPHA,BETA,I,IS*IV2,E,EV,IVN100,WK,IER)
            EBEFOR=E(IV2)

          END IF
 100  CONTINUE
C     WRITE(*,*)' LANCZS DID NOT CONVERGE'
      ICON=-10000
      RETURN
      END


      SUBROUTINE LANV1(MTMAX,MT,MAXMTE,MTE,IEN,IEV,ICON,IVN100
     &                 ,IU,AU,V1,V2,V3,X,EV )

C    V1,V1,V2          WORKING AREA
C    X(MT,IEN)    *EIGEN VECTOR ...X(MT,1) IS ALSO INITAL VECTOR
C    AU(MAX)      *NONZERO   ELEMENT OF THE MATRIX
C    IU(MAX)      *NONZERO   POSTION OF THE MATRIX
C    ISTEP           @ INTERVAL TO CHECK CONVERGENCE
C    E               # FOUR LOWEST EIGENVALUES TO BE RETURNED
C    ITER            # NUMBER OF ITERATIONS TO BE RETURNED
C    MT              @ DIMENSION OF THE MATRIX


      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER(IA=1000,IVN10=200)
      DIMENSION  V1(MTMAX),V2(MTMAX),V3(MTMAX),X(MTMAX,IEN)
```

```
      DIMENSION AU(MAXMTE)
      INTEGER*4 IU(MAXMTE)
      DIMENSION ALPHA(IA),BETA(IA), EV(IVN100,IEV)
      COMMON/COM1/ IESET(IVN10),ENAMA(IA)
       IF(ICON.GT.0) ITER=ICON
       IF(ICON.EQ.-100) ITER=MT

C*** INITIALIZATION
          DO 10 I=1,MT
            V1(I)=X(I,1)
            V2(I)=V1(I)
            V3(I)=0.0D0
          DO 10 II=2,IEN
          X(I,II)=0.0D0
 10   CONTINUE

C***
          DO 12 I=1,IEN
          DO 12 II=1,MT
 12       X(II,I)=EV(1,IESET(I))*V1(II)

C*** ALPHA(1) AND BETA(1)

       CALL     AX(MTMAX,MAXMTE,MTE,IU,AU,V1,V2)

       ALPHA(1)=0.0D0
       DO 40 I=1,MT
       ALPHA(1)=ALPHA(1)+V1(I)*V2(I)
 40    CONTINUE

       DO 45 I=1,MT
 45    V1(I)=V1(I)-ALPHA(1)*V2(I)
       BETA1=0.0D0
       DO 50 I=1,MT
 50    BETA1=BETA1+V1(I)*V1(I)
       BETA(1)=SQRT(BETA1)
       IF(BETA(1).LT.0.5D-20) THEN
 C     WRITE(*,*) 'LANV1 DE      BETA(1)=0.0D0 '
         STOP
         END  IF
           DO 60 I=1,MT
 60        V3(I)=0.0D0
           DO 65 I=1,MT
 65        V3(I)=V1(I)/BETA(1)
           DO 67 II=1,IEN
           DO 67 I=1,MT
 67        X(I,II)=X(I,II)+EV(2,IESET(II))*V3(I)

 C*** ITERATION

       DO 100 I=2,ITER-1
```

```
      CALL     AX(MTMAX,MAXMTE,MTE,IU,AU,V1,V3)

          ALPHAI=0.0D0
          DO 120 J=1,MT
 120      ALPHAI=ALPHAI+V3(J)*V1(J)
          ALPHA(I)=ALPHAI
          BETAI1=BETA(I-1)
          DO 130 J=1,MT
 130      V1(J)=V1(J)-ALPHAI*V3(J)-BETAI1*V2(J)
          BETAI=0.0D0
          DO 140 J=1,MT
              V2(J)=V3(J)
              BETAI=BETAI+V1(J)*V1(J)
 140      CONTINUE
          BETA(I)=ABS(BETAI)**0.5
         DBETA=1.0D0/BETA(I)
           DO 145 J=1,MT
 145       V3(J)=V1(J)*DBETA

          DO  150  JJ=1,IEN
          DO  150  J=1,MT
 150        X(J,JJ)=X(J,JJ)+EV(I+1,IESET(JJ))*V3(J)

 100  CONTINUE

C......X(I,II)  normalization .........
          DO 200 JJ=1,IEN
                VNORM=0.0D0
            DO 210 J=1,MT
 210          VNORM=VNORM+ABS(X(J,JJ))**2
              VNORM=SQRT(VNORM)

            DO 230 J=1,MT
 230          X(J,JJ)=X(J,JJ)/VNORM
 200          CONTINUE
      RETURN
      END

      SUBROUTINE CG(MTMAX,MT,MAXMTE,MTE,IEN,IV
     &                                    ,ICON,IU,AU,X,E,P,Q,R,EPS)



      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(MTMAX),Q(MTMAX),R(MTMAX), X(MTMAX,IEN)
      DIMENSION              AU(MAXMTE)
      INTEGER*4 IU(MAXMTE)
      DIMENSION E(IEN)

C*********   CG-METHOD       ********************************
```

```
      DO 10  I=1,MT
        R(I)= X(I,IV)
 10     P(I)=-R(I)

      DO 1000 K=1,MT+100

      DO 55 I=1,MT
 55     Q(I)=-E(IV)*P(I)

      CALL     AX(MTMAX,MAXMTE,MTE,IU,AU,Q,P)

           ALP1=0.0D0
           ALP2=0.0D0

      DO 40 I=1,MT
           ALP1=ALP1-P(I)*R(I)
 40        ALP2=ALP2-P(I)*Q(I)
            ALP=ALP1/ALP2

      DO 50 I=1,MT
          X(I,IV)=X(I,IV)-ALP*P(I)
 50       R(I)=R(I)-ALP*Q(I)


           BET1=0.0D0
           BET2=0.0D0

      DO 80 I=1,MT
           BET1=BET1+R(I)*Q(I)
 80        BET2=BET2-P(I)*Q(I)

            BET=-BET1/BET2

      DO 90 I=1,MT
 90          P(I)=-R(I)+BET*P(I)


C*************   NORMALIZATION & SHUSOKU  HANTEI  **********

          EPS1=0.0D0

      DO 100 I=1,MT
 100         EPS1=EPS1+R(I)**2

           IF(SQRT(EPS1).LE.EPS    )    THEN

          VNOR=0.0D0
      DO 110 I=1,MT
 110         VNOR=VNOR+X(I,IV)**2
             RVNOR=1.0D0/SQRT(VNOR)
      DO 120 I=1,MT
```

```
120       X(I,IV)=X(I,IV)*RVNOR

               RETURN
                    END  IF


1000 CONTINUE

          IF(EPS.GE.1.0D0)  THEN
C         WRITE(*,*) 'CG DONT CONVERGE !!!!!     '
           ICON=-300
           RETURN
           END  IF

            VNOR=0.0D0
      DO 140 I=1,MT
140           VNOR=VNOR+X(I,IV)**2
              RVNOR=1.0D0/SQRT(VNOR)
      DO 130 I=1,MT
130       X(I,IV)=X(I,IV)*RVNOR

                END


      SUBROUTINE CHK3(MTMAX,MT,MAXMTE,MTE,IEN,IV
     &                                      ,IU,AU,X,V1,V2,EE,EPSCHK)
C
C#    E=<X AU X>
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION       X(MTMAX,IEN),V1(MTMAX),V2(MTMAX)
      DIMENSION       AU(MAXMTE)
      INTEGER*4       IU(MAXMTE)



C***
      DO 10 I=1,MT
            V1(I)=0.0D0
10          V2(I)=X(I,IV)

      CALL AX(MTMAX,MAXMTE,MTE,IU,AU,V1,V2)

        XHX=0.0D0
      DO 30 I=1,MT
               XHX=XHX+X(I,IV)*V1(I)
 30   CONTINUE

        EPS=0.0D0
      DO 35 I=1,MT
```

```
            EPS=EPS+(XHX*X(I,IV)-V1(I))**2
 35   CONTINUE
C        WRITE(*,*) '  ZANSA1=',EPS/XHX
                       EPSCHK=EPS/XHX**2
                       EPSCHK=SQRT(EPSCHK)/MT
                       EE=XHX

              END
      SUBROUTINE AX(MTMAX,MAXMTE,MTE,IU,AU,Y,X)

      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION    Y(MTMAX), X(MTMAX)
      INTEGER*4   IU(MAXMTE)
      DIMENSION          AU(MAXMTE)

            I=0

      DO 10 J=1,MTE
                 I=I+(1-SIGN(1,IU(J)))/2
              Y(I)= Y(I)    +AU(J)*X(ABS(IU(J)))
       Y(ABS(IU(J)))= Y(ABS(IU(J)))+AU(J)*X(I)
 10   CONTINUE
                       END

C********** eigenvalue select      ********

      SUBROUTINE EIGENS(IEV,ICON2,IC,IMM,E,EOLD,EPS,ET)

C    IVECTN         % NUMBER OF EIGEN-VALUES TO BE EXCUTED
C    ICON2          % RETUN  CONDITION NUMBER
C    E(IEV)         % EIGENVALUES
C    EOLD(IVN100)   % OLD EIGENVALUES
C    IVN100         % DIMENSION NUMBER OF OLD EIGENVALUES


      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (IVN100=1000          )
      DIMENSION E(IEV) ,EOLD(IVN100),ESET(IVN100)

C      IC.....# OF CONVERGED EIGEN VALUE
C      ET.....RESOLUTION OF  EIGEN VALUES



        DO 10 I=1,IMM

        IF(ABS(E(I)-EOLD(I)).LT.EPS*ABS(E(1)).AND.
     &          ABS( E(I) ).LT.ET                  ) THEN
           IC=IC+1
           ESET(IC)=E(I)
           ET=ABS(ESET(IC))-ABS(EPS*E(1))
```

```
      END  IF


10      CONTINUE

C.....  IF IC>=ICON2 -> END   ELSE  -> MORE ITERATION

         DO  20 I=1,IC
         E(I)=ESET(I)
20       CONTINUE


         ICON2=IC

            END
```