プログラムのページ

担当 伊 理 正 夫

6401. 三次方程式の解法

高橋秀知（東大物性研）

実係数の三次方程式

$$x^3+Ax^2+Bx+C=0$$

の根を Cardano の方法で求めるプログラムである.

A, B, C はこの順に穿孔する.

〔注 1〕 Cardano の方法については，周知であるので，説明を省略する.

〔注 2〕 このプログラムは，ISSP-ALGOL でテストされた.

〔注 3〕 したがって入出力に関する statement はプログラムのページの規約に従って変更してある.

三次方程式の解法プログラム

```
CARDANO:
  begin real A, B, C, P, Q, R, W 1, W 2, ALPHA,
      PI; PI: =3.1415926535;
START: READ(A); READ(B); READ(C);
  Q: =C−A×B/3+2×A↑3/27; P: =B/3−A↑2/9;
  R: =Q↑2+4×P↑3; CRLF;
    if R<0 then
      begin if Q=0 then
        begin PRINTSTRING('X 1= ');
          PRINT(−A/3);
          PRINTSTRING('X 2= ');
          PRINT(sqrt(−3×P)−A/3);
          PRINTSTRING('X 3= ');
          PRINT(−sqrt(−3×P)−A/3)
      end else
        begin ALPHA: =arctan(sqrt(−R)/
            (−Q)); if ALPHA<0 then ALPHA
            : =PI+ALPHA;
          PRINTSTRING('X 1= ');
          PRINT(2×sqrt(−P)×cos(ALPHA/3)
            −A/3);
          PRINTSTRING('X 2= ');
          PRINT(−2×sqrt(−P)×cos((PI+
            ALPHA)/3)−A/3);
          PRINTSTRING('X 3= ');
          PRINT(−2×sqrt(−P)×cos((PI−
            ALPHA)/3)−A/3)
        end
      end else
      begin R: =sqrt(R);
        W 1: =if Q<0 then (−Q+R)/2 else
          (−Q−R)/2;
        W 2: =−Q−W 1;
        W 1: =sign(W 1)×abs(W 1)↑(1/3);
        W 2: =sign(W 2)×abs(W 2)↑(1/3);
        if R=0 then begin
          PRINTSTRING('X 1= ');
          PRINT(W 1+W 2−A/3);
          PRINTSTRING('X 2=X 3= ');
          PRINT(−(W 1+W 2)/2−A/3)
        end
      else begin
          PRINTSTRING('X 1= ');
          PRINT(W 1+W 2−A/3); CRLF;
          PRINTSTRING('X 2, X 3, REAL
            PART= ');
          PRINT(−(W 1+W 2)/2−A/3);
          PRINTSTRING('IMAGINARY
            PART= ');
          PRINT(1.7320508075×(W 1−W 2)/2)
        end
      end
    end
```

6402. Hermite 行列の固有値，unitary 行列

槌田 敦（東大理学部）

H を Hermite 行列，

Λ を対角行列，

U を Unitary 行列

とするとき，

$$HU=U^{+}\Lambda$$

を満たす，Λ と U を求める.

H の行列要素を $A_{ij}+B_{ij}i$,

U の行列要素を $U_{ij}+V_{ij}i$

とする.

Hermite 行列の固有値解法に，対称行列で使われている JACOBI の方法を拡張して用いる. (JACOBI

法のプログラムについては Vol. 3 の No. 6 のプログラムのページ参照のこと.)

まず行列要素が $A_{ij}+B_{ij}i$ なる (A_{ij}, B_{ij}) の形で与えられているものとして, **procedure** A plus Bi type JACOBI の中で複素数の表現をかきなおす. すなわち,

$A_{ij}+B_{ij}i \rightarrow A'_{ij}\exp(iB'_{ij})$

$U_{ij}+V_{ij}i \rightarrow U'_{ij}\exp(iV'_{ij})$

である. 次に **procedure** JACOBI によって, 二次元の unitary 変換を, くり返して, 非対角要素の絶対値 A_{ij} が, THR よりも全て小さくなるまで, つづける.

このとき, **procedure** JACSUPPLEMENT によって, 同時に, unitary 行列も作る. 再び,

procedure A plus Bi type JACOBI

の中で,

$A'_{ij}\exp(iB'_{ij}) \rightarrow A_{ij}+B_{ij}i$

$U'_{ij}\exp(iV'_{ij}) \rightarrow U_{ij}+V_{ij}i$

の変換をして, 答を打ち出し, 次の行列をよみこむ.

もし, Hermite 行列の要素が

$A'_{ij}\exp(iB'_{ij})$

なる (A_{ij}', B_{ij}') の形で,与えられているときは, 最後より二つめの label, LABEL 2: をつけた statement, LABEL 2: A plus Bi type JACOBI を

LABEL 2: JACOBI

に取りかえればよい.

データが何れの形で与えられているとしても, その入力テープの作成法は, 同じである.

まず, (THR), 次に必要な個数の行列を並べる.

行列の入れ方は,

次元数 (AN)

$(A_{11}\ B_{11})$

$(A_{21}\ B_{21})\ (A_{22}\ B_{22})$

$(A_{31}\ B_{31})\ (A_{32}\ B_{32})\ (A_{33}\ B_{33})$

とすればよい.

ただし, $B_{ii}=0\ (i=1, 2, \cdots\cdots AN)$

である.

データの終りに, 0を入れておくと $AN=0$ になって計算は終る.

〔注〕 このプログラムは, ISSP-ALGOL で, 試されたが, 入出力の statement は, プログラムのページの規約にかなうように取りかえてある.

The Eigenvalue Problem of Hermite Matrix:

```
begin integer AN; real THR;
START: READ (THR);
LABEL1: READ (AN);
begin array A, B, U, V[1 : AN, 1 : AN];
  real RR, C, S, W, WW, X, XX;
  procedure JACSUPPLEMENT;
    begin
    real CC, CCC, SS, SSS, WC, WS, XC, XS;
    CC: =cos (WW+RR); CCC: =cos (XX-RR);
    SS: =sin (WW+RR); SSS: =sin (XX-RR);
    WC: =W×C×CC-X×S×CCC; XC: =W×S
      ×CC+X×C×CCC;
    WS: =W×C×SS-X×S×SSS; XS: =W×S×
      SS+X×C×SSS;
    W: =sqrt (WC↑2+WS↑2); X: =sqrt (XC↑2
      +XS↑2);
    WW: =if WC≠0 then arctan (WS/WC)
      else if WS>0 then 1.570796333
        else-1.570796333;
      if WC<0 then WW: =WW+3.14159265;
      XX: =if XC≠0 then arctan (XS/XC)
        else if XS>0 then 1.570796333
          else -1.570796333;
      if XC<0 then XX: =XX+3.14159265
      end JACSUPPLEMENT;
    procedure JACOBI;
      begin
      real P, Q, R, W1;
      integer I, J, K, L, M;
      for L: =1 step 1 until AN do
        for M: =1 step 1 until AN do U[L, M]:
          =V[L, M]: =0;
      for L: =1 step 1 until AN do U[L, L]: =1;
LABEL1: W: =THR;
      for M: =2 stpe 1 until AN do
      for L: =1 step 1 until M-1 do
        begin
        W1: =abs(A[L, M]);
        if W1>W then begin W: =W1; I: =L;
          J: =M end
        end;
      if W=THR then go to LABEL2;
      P: =A[I, I]; Q: =A[J, J]; R: =A[I, J];
      W: =Q-P; RR: =0.5×B[I, J];
      if W=0 then C: =S: =0.7071067812
        else begin
```

```
        W:=0.5×arctan(2.0×R/W);
        C:=cos (W);
        S:=sin (W)
        end;
    for K:=1 step 1 until AN do
      begin
      W:=U[K, I]; WW:=V[K, I];
      X:=U[K, J]; XX:=V[K, J];
      JACSUPPLEMENT;
      U[K, I]:=W; V[K, I]:=WW;
      U[K, J]:=X; V[K, J]:=XX;
      W:=A[K, I]; WW:=B[K, I];
      X:=A[K, J]; XX:=B[K, J];
      JACSUPPLEMENT;
      A[K, I]:=A[I, K]:=W; B[K, I]:=WW;
        B[I, K]:=-WW;
      A[K, J]:=A[J, K]:=X; B[K, J]:=XX;
        B[J, K]:=-XX
      end;
    A[I, I]:=W:=P×C↑2-2×R×C×S+Q×
      S↑2;
    A[J, J]:=P+Q-W;
    A[I, J]:=A[J, I]:=B[I, J]:=B[J, I]:=0;
    go to LABEL 1;
LABEL 2: end JACOBI;
  procedure A plus Bi type JACOBI;
    begin integer I, J;
    for J:=2 step 1 until AN do
    for I:=1 step 1 until J-1 do
      begin
        W:=A[I, J]; WW:=B[I, J];
        X:=sqrt(W↑2+WW↑2);
        XX:=if W≠0 then arctan (WW/W)
          else if WW>0 then 1.5707963
            else -1.5707963;
      if W<0 then XX:=XX+3.14159265359;
    A[I, J]:=A[J, I]:=X; B[I, J]:=XX; B[J,
      I]:=-XX
    end;
    JACOBI;
    for J:=1 step 1 until AN do
    for I:=1 step 1 until AN do
      begin W:=U[I, J]; WW:=V[I, J]; U[I,
        J]:=W×cos (WW); V[I, J]:=W×sin
          (WW)
      end
  end A plus Bi type JACOBI;
  procedure HERMITE MATRIX PRINT;
  begin integer I, J;
    PRINTSTRING(' HERMITE MATRIX ');
    for I:=1 step 1 until AN do
      begin for J:=1 step 1 until AN do
        begin PRINT(A[I, J]); PRINTSTRING
        (' + '); PRINT(B[I, J]);
        PRINTSTRING(' i ')
        end;
        CRLF
      end
  end HERMITE MATRIX PRINT;
  procedure EIGEN VALUE PRINT;
  begin integer I;
    PRINTSTRING(' EIGEN VALUES ');
    for I:=1 step 1 until AN do
      begin PRINT(A[I, I]); SPACE(10) end
    CRLF
  end; EIGEN VALUE PRINT;
  procedure UNITARY MATRIX PRINT;
  begin integer I, J;
    PRINTSTRING(' UNITARY MATRIX ');
      CRLF;
    for I:=1 step 1 until AN do
      begin for J:=1 step 1 until AN do
        begin PRINT(U[I, J]); PRINTSTRING
          (' + ');
          PRINT(V[I, J]); PRINTSTRING(' i ')
        end;
        CRLF
      end
  end UNITARY MATRIX PRINT;
    procedure READ and FORM HERMITE
      MATRIX;
    begin integer I, J;
      for J:=1 step 1 until AN do
      for I:=1 step 1 until J do
        begin READ (A[I, J]); READ(B[I, J])
          end;
      for J:=2 step 1 until AN do
      for I:=1 step 1 until J-1 do
```

```
    begin A[J, I]:=A[I, J]; B[J, I]:=-B[I,
      J] end
   end READ and FORM HERMIX MATRIX;
    if AN=0 then go to LABEL 3;
    READ and FORM HERMITE MATRIX;
    HERMITE MATRIX PRINT;
LABEL 2: A plus Bi type JACOBI;
    EIGEN VALUE PRINT;
    UNITARY MATRIX PRINT;
    CRLF;
    go to LABEL 1;
LABEL 3:
end
```

6403. 行列の固有値と固有ベクトル

清水公子（東大物性研究所）

n 次の行列 $A=(a_{ij})$ の固有値と固有ベクトルを power method で絶対値最大のものから m 個求めるプログラムである.

$$X=(1,00\cdots\cdots 0)^T \qquad (1)$$

を与え,

$$Y\leftarrow AX,\ X\leftarrow Y \qquad (2)$$

を適当にくり返し，第一根 λ_1 を X, Y の比として求め，対応するベクトル U_1 も求める．次に λ_1 に対応する A^T の固有ベクトル V_1 を求め

$$U_1^T V_1=1$$

となるように規格化する.

$$A_2=A-\lambda_1 U_1 V_1^T$$

を新しいAと考えて上のプロセスをくり返し，第二根と，対応する固有ベクトルを求める．以下同様にして第 m 根まで求める.

（2） のくり返しの収束判定には

$$\sum|x_i-y_i|<\mathrm{eps}$$

を使う．(x_i, y_i はベクトル X, Y の i 番目のエレメントである.)

固有ベクトルは長さが1に等しくなるように規格化されている.

データは次の順序に入れる.

n, m, eps, $a_{11}, a_{12}, \cdots\cdots a_{1n}, a_{21}, \cdots\cdots\cdots a_{nm}$

〔注〕 このプログラムは ISSP-ALGOL で通したものであるが，入出力の命令は投稿規定に従って変更した.

```
POWER METHOD:
  begin integer n, m; real eps;
    READ(n); READ(m); READ(eps);
  begin integer i, j, k, h;
    real w, SQ, delta, lambda, sigma;
    array x, y, u[1:n], A[1:n, 1:n];
    switch sw 1:=normal, trans;
    switch sw 2:=result, prep;
    procedure INNERPRODUCT(p, a, b, r);
      integer p; real a, b, r;
      begin real s; s:=0;
      fof p:=1 step 1 until n do s:=s+a×b;
        r:=s
      end INNERPRODUCT;
start: for i:=1 step 1 until n do
      for j:=1 step 1 until n do READ(A[i, j]);
      h:=1;
      for i:=1 step 1 until m do
  begin
  st 1: x[1]:=1.0;
      for j:=2 step 1 until n do x[j]:=0;
  st 2: go to sw 1[h];
  normal: for j:=1 step 1 until n do
      INNERPRODUCT(k, A[j, k], x[k], y[j]);
      go to test;
  trans: for j:=1 step 1 until n do
      INNERPRODUCT(k, A[k, j], x[k], y[j]);
  test: if y[1]≠0 then
      begin lambda:=y[1]; w:=1.0/lambda;
        y[1]:=1.0; delta:=0;
        for j:=2 step 1 until n do
          begin y[j]:=y[j]×w;
                delta:=delta+abs(x[j]-y[j])
          end;
        if delta<eps then go to sw 2[h]
      end;
      for j:=1 step 1 until n do x[j]:=y[j];
      go to st 2;
  prep: INNERPRODUCT(j, u[j], y[j], sigma);
      lambda:=lambda/sigma;
      for j:=1 step 1 until n do
      for k:=1 step 1 until n do
        A[j, k]:=A[j, k]-lambda×u[j]×y[k];
      h:=1;
      go to st 1;
  result: sigma:=0;
      for j:=1 step 1 until n do
        sigma:=sigma+y[j]↑2;
      SQ:=sqrt(sigma); SQ:=1.0/SQ;
      for j:=1 step 1 until n do u[j]:
          =y[j]×SQ;
      CRLF; PRINTSTRING('i='); PRINT(i);
        SPACE(3);
      PRINTSTRING('eigen value=');
      PRINT(lambda); CRLF;
      PRINTSTRING('eigen vector'); CRLF;
      for j:=1 step 1 until n do
        begin PRINT(u[j]); if (j÷6)×6=j then
          CRLF
        end;
      h:=2
    end i loop
  end
end
```