

5. 行列の固有値問題

$n \times n$ 正方行列 A に対する n 個の固有値 $\lambda_i (i=1, 2, \dots, n)$ と対応する固有ベクトル \mathbf{u}_i は次式を満たす。

$$A\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \quad \mathbf{u}_i = \begin{bmatrix} u_{1i} \\ u_{2i} \\ \vdots \\ u_{ni} \end{bmatrix}$$

これらはまとめて、つぎのように書ける。

$$AU = U\Lambda, \quad U = [\mathbf{u}_1 | \mathbf{u}_2 | \cdots | \mathbf{u}_n], \quad \Lambda = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

5.1 ヤコビ法

実数対称行列 A の固有値と対応する固有ベクトルをすべて求める方法であり、古典的ではあるが、行列サイズが 10 以下ならば現在でも使用されている。

直交行列 P を用いて A の相似変換 (すなわち直交変換) $\tilde{A} = P^T A P$ を行い、対角行列に収束させていくと、対角要素が固有値となる (相似変換に関して固有値は不変)。

【アルゴリズム 5.1.1】

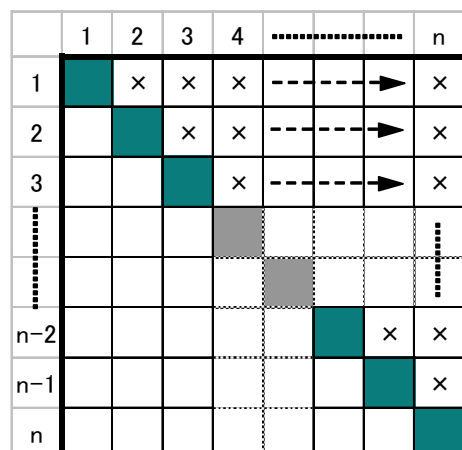
$A :=$ 与えられた行列

$U := I$ (単位行列)

```

for m=1, 2, ...           [収束条件を満たす (すべての非対角要素が  $\epsilon$  より小さくなる) まで繰り返す]
  for i=1 to n-1         [A の対角要素を除いた上三角部分を走査]
    for j=i+1 to n
      if  $|a_{ij}| \geq \epsilon$  then      [非対角要素で絶対値が  $\epsilon$  より大きい要素  $a_{ij}$  を零にする変換]
         $\theta := (1/2) \cot^{-1}((a_{ii} - a_{jj}) / (2a_{ij}))$ 
         $A := P^T A P$                 [A の直交変換 (非対角要素  $a_{ij}$  を零にする) ]
         $U := U P$                   [U の変換]
      end
    end
  end
end
end
end

```



A の直交変換 $\tilde{A}=P^TAP$ はつぎのように書き表せ,

$$\cot 2\theta = (a_{ii}-a_{jj})/(2a_{ij})$$

$$\tilde{a}_{ii} = (1/2)(a_{ii}+a_{jj}) + (1/2)(a_{ii}-a_{jj})\cos 2\theta + a_{ij}\sin 2\theta$$

$$\tilde{a}_{jj} = (1/2)(a_{ii}+a_{jj}) - (1/2)(a_{ii}-a_{jj})\cos 2\theta - a_{ij}\sin 2\theta$$

$$\tilde{a}_{ij} = \tilde{a}_{ji} = 0$$

$$\tilde{a}_{ik} = \tilde{a}_{ki} = a_{ik}\cos \theta + a_{jk}\sin \theta \quad (k=1, 2, \dots, n, \text{ただし } k \neq i, j)$$

$$\tilde{a}_{jk} = \tilde{a}_{kj} = -a_{ik}\sin \theta + a_{jk}\cos \theta \quad (k=1, 2, \dots, n, \text{ただし } k \neq i, j)$$

$$\tilde{a}_{kl} = a_{kl} \quad (k, l=1, 2, \dots, n, \text{ただし } k, l \neq i, j)$$

U の変換 $\tilde{U}=UP$ はつぎのようになる。

$$\tilde{u}_{ki} = u_{ki}\cos \theta + u_{kj}\sin \theta \quad (k=1, 2, \dots, n)$$

$$\tilde{u}_{kj} = -u_{ki}\sin \theta + u_{kj}\cos \theta \quad (k=1, 2, \dots, n)$$

$$\tilde{u}_{kl} = u_{kl} \quad (k, l=1, 2, \dots, n, \text{ただし } l \neq i, j)$$

計算誤差を少なくするための注意 :

実際の数値計算においては, θ を求めることなく, 代数式から $\cos 2\theta$, $\sin 2\theta$, $\cos \theta$, $\sin \theta$ を求めるほうが, 計算時間, 計算誤差とも少なくできる。

一般に, $\cot \alpha$ (あるいは $\tan \alpha = \cot^{-1} \alpha$) から $\cos \alpha$, $\sin \alpha$ を求める場合,

$$|\cot \alpha| \leq 1 \text{ のとき} \quad \sin \alpha = \frac{1}{\sqrt{1+\cot^2 \alpha}}, \quad \cos \alpha = \cot \alpha \cdot \sin \alpha$$

$$|\cot \alpha| \geq 1 \text{ のとき} \quad |\tan \alpha| = \frac{1}{|\cot \alpha|} \leq 1, \quad \cos \alpha = \frac{1}{\sqrt{1+\tan^2 \alpha}}, \quad \sin \alpha = \tan \alpha \cdot \cos \alpha$$

により計算すると, 1 よりはるかに大きい数の二乗を行わずにすむのでオーバーフローを避けることができ, かつ 0 による割算を避けることができる。 $\cot 2\theta$ から $\cos 2\theta$, $\sin 2\theta$ を求めるときはこの関係を用いる。さらに $\cos \theta$, $\sin \theta$ を求めるときには半角公式を用いるが, 桁落ちを避けるため倍角公式も併用して,

$$\cos 2\theta \geq 0 \text{ のとき} \quad \cos \theta = \sqrt{\frac{1}{2}(1+\cos 2\theta)}, \quad \sin \theta = \sin 2\theta / (2\cos \theta)$$

$$\cos 2\theta \leq 0 \text{ のとき} \quad \sin \theta = \sqrt{\frac{1}{2}(1-\cos 2\theta)}, \quad \cos \theta = \sin 2\theta / (2\sin \theta)$$

により計算する。

実用に耐える数値計算では, このような細部にまで注意が払われていることに留意しよう。

【プログラム 5.1.1】

```

1 /* solve eigen-value problem for a symmetric matrix */
2 /* by Jacobi's method */
3
4 #include <stdio.h>
5 #include <math.h>
6
7 #define NMAX 20
8
9 int jacobi(float);
10
11 float a[NMAX][NMAX], u[NMAX][NMAX];

```

```

12 int  n;
13
14 void main(void)
15 {
16     int mcum, i, j;
17     float eps;
18
19     printf("Order of a Symmetric matrix: n ? ");
20     scanf("%d", &n);
21     printf("Input Lower triangle of matrix\n");
22     for(i=0; i<n; i++) {
23         printf("? A(%2d, j), j=1,%2d : ", i+1, i+1);
24         for(j=0; j<=i; j++) {
25             scanf("%g", &a[i][j]);
26             a[j][i]=a[i][j];
27         }
28     }
29     printf("\nSymmetric matrix: A\n");
30     for(i=0; i<n; i++) {
31         for(j=0; j<n; j++)
32             printf(" %15.6e", a[i][j]);
33         printf("\n");
34     }
35     for(i=0; i<n; i++) {
36         for(j=0; j<i; j++)
37             u[i][j]=u[j][i]=0;
38         u[i][i]=1;
39     }
40
41     mcum= 0; /* initial set for cumulative number of transformations */
42     while(printf("\nTolerance : eps ? "), scanf("%g", &eps)!=EOF) {
43         mcum += jacobi(eps);
44         printf("cumulative no. of transformations=%d\n", mcum);
45         printf("Eigen-values:\n");
46         for(i=0; i<n; i++)
47             printf(" %15.6e", a[i][i]);
48         printf("\nEigen-vectors:\n");
49         for(i=0; i<n; i++) {
50             for(j=0; j<n; j++)
51                 printf(" %15.6e", u[i][j]);
52             printf("\n");
53         }
54     }
55 }
56
57 int jacobi(float eps)
58 {
59     int  i, j, k, m, mcycle;
60     float p, q, t, s2, c2, c, s, r;
61
62     m=0; /* initial set for transformation counter (total) */

```

```

63  while(1) {
64      mcycle=0;          /* initial set for transformation counter (one cycle) */
65      for(i=0;i<n-1;i++)
66          for(j=i+1;j<n;j++)
67              if( fabs(a[i][j])>=eps ) {
68                  m++; mcycle++;      /* increment of transformation counters */
69                  p= (a[i][i]-a[j][j])/2;  q=a[i][j];
70                  if( fabs(p)<fabs(q) ) {
71                      t=p/q;            /* t=cot(2 theta) */
72                      s2= 1/sqrt(1+t*t);
73                      if(q<0) s2= -s2;
74                      c2= t*s2;
75                  }else{
76                      t=q/p;            /* t=tan(2 theta) */
77                      c2= 1/sqrt(1+t*t);
78                      if(p<0) c2= -c2;
79                      s2= t*c2;
80                  }
81                  if(c2>0) {
82                      c= sqrt( (1+c2)/2 );  s= s2/c/2;
83                  }else{
84                      s= sqrt( (1-c2)/2 );  c= s2/s/2;
85                  }
86                  r= (a[i][i]+a[j][j])/2;
87                  a[i][i]= r + p*c2 + q*s2;
88                  a[j][j]= r - p*c2 - q*s2;
89                  a[i][j]= a[j][i]=0;
90                  for(k=0;k<n;k++)
91                      if( (k!=i)&&(k!=j) ) {
92                          p= a[i][k];  q=a[j][k];
93                          a[i][k]= a[k][i]= p*c+q*s;
94                          a[j][k]= a[k][j]= -p*s+q*c;
95                      }
96                  for(k=0;k<n;k++) {
97                      p= u[k][i];  q= u[k][j];
98                      u[k][i]= p*c+q*s;
99                      u[k][j]= -p*s+q*c;
100                 }
101             }
102     if(mcycle==0) return(m);      /* return no. of transformations (total) */
103 }
104 }

```

【実行例】

教科書章末問題【2】の質点・ばね系の固有値問題をヤコビ法により解く。

(1) 運動方程式はつぎのように行列・ベクトル表示される。

$$M\ddot{\mathbf{s}} = -K\mathbf{s}$$

$$\mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix}, \quad M = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix}, \quad K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}$$

(2) 系は固有振動数 ω の調和運動をしているとすれば, 解は 振幅を $\mathbf{x} = [x_1, x_2, x_3]^T$ として

$$\mathbf{s} = \mathbf{x}e^{i\omega t}$$

と表され, これを運動方程式に代入すると, 一般固有値問題

$$K\mathbf{x} = \omega^2 M\mathbf{x}$$

となるので, 結局, 標準固有値問題

$$A\mathbf{x} = \lambda\mathbf{x} \quad (\text{ここに, } A = M^{-1}K, \lambda = \omega^2)$$

を得る。 $m_1=m_2=m_3=m$, $k_1=k_2=k_3=k$ とすると, 行列 A は

$$A = \frac{k}{m} \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}$$

と表される。いま $k/m=1$ として, 標準固有値問題を解く。

```
-----実行開始-----
Order of a Symmetric matrix: n ? 3
Input Lower triangle of matrix
? A( 1, j), j=1, 1 : 2
? A( 2, j), j=1, 2 : -1 2
? A( 3, j), j=1, 3 : 0 -1 1

Symmetric matrix: A
  2.000000e+00  -1.000000e+00   0.000000e+00
 -1.000000e+00   2.000000e+00  -1.000000e+00
  0.000000e+00  -1.000000e+00   1.000000e+00

Tolerance : eps ? 0.1
cumulative no. of transformations=5
Eigen-values:
  3.246979e+00   1.554825e+00   1.981960e-01
Eigen-vectors:
 -5.907426e-01  -7.404091e-01   3.206514e-01
  7.370955e-01  -3.335710e-01   5.877248e-01
 -3.281969e-01   5.835449e-01   7.428069e-01

Tolerance : eps ? 0.01
cumulative no. of transformations=6
Eigen-values:
  3.246979e+00   1.554958e+00   1.980622e-01
Eigen-vectors:
 -5.907426e-01  -7.371892e-01   3.279864e-01
  7.370955e-01  -3.277196e-01   5.910075e-01
 -3.281969e-01   5.908908e-01   7.369769e-01

Tolerance : eps ? 0.0001
```

cumulative no. of transformations=7

Eigen-values:

3.246980e+00 1.554958e+00 1.980622e-01

Eigen-vectors:

-5.910084e-01 -7.369762e-01 3.279864e-01

7.369773e-01 -3.279853e-01 5.910075e-01

-3.279838e-01 5.910090e-01 7.369769e-01

Tolerance : eps ? 1.0e-6

cumulative no. of transformations=8

Eigen-values:

3.246980e+00 1.554958e+00 1.980622e-01

Eigen-vectors:

-5.910090e-01 -7.369762e-01 3.279853e-01

7.369761e-01 -3.279853e-01 5.910090e-01

-3.279853e-01 5.910090e-01 7.369762e-01

Tolerance : eps ? ^Z (controlとZを同時に押す) ↵ (Enter Keyを押す)

…… 入力終了コードの入力

-----おしまい-----

参考文献

高倉葉子：数値計算の基礎—解法と誤差—，コロナ社（2007）

森口繁一，伊理正夫，武市正人 編：Cによる算法痛論，東京大学出版会（2000）

Heath, Michael T.: Scientific Computing, An Introductory Survey, McGraw-Hill (2002)