実対称行列の一般固有値問題 (Fortran90)

シキノ *

February 26, 2023

Contents

*https://slpr.sakura.ne.jp/qp/, https://twitter.com/sikinote

1 一般固有値問題

$$A\mathbf{x} = \lambda B\mathbf{x} \tag{1.1}$$

ただし、A はエルミート行列で $A_{ij} = A^*_{ji}$, B は正定値エルミート行列で $B_{ij} = B^*_{ji}$ かつ B の固有値はいずれも正である。

1.1 解法

B が正定値行列である時、コレスキー分解によって以下の形に分解される。

$$B = U^\dagger U, \quad U_{i,j<i} = 0 \tag{1.2}$$

ここで、$U^\dagger$ は行列 U のエルミート共役（転置複素共役）で $U^\dagger = U^{T^*}$, U は上三角行列である（つまり $U^\dagger$ は下三角行列となる）。

また、行列 $\mathbf{x}$ が

$$\mathbf{x} = U^{-1}\mathbf{y} \tag{1.3}$$

として書けるとすると

$$\begin{aligned} A\mathbf{x} &= \lambda B\mathbf{x} && \text{(1.4a)} \\ AU^{-1}\mathbf{y} &= \lambda U^\dagger U U^{-1}\mathbf{y} && \text{(1.4b)} \\ U^{\dagger -1}AU^{-1}\mathbf{y} &= \lambda\mathbf{y} && \text{(1.4c)} \\ C\mathbf{y} &= \lambda\mathbf{y} && \text{(1.4d)} \end{aligned}$$

となり、標準固有値問題の形に書き換えられる。ここで $C = U^{\dagger -1}AU^{-1}$ は対称行列となる。

参考は [2]。

1.2 具体例

$$A = \begin{pmatrix} 1 & 2 & 4 \\ 2 & 1 & 5 \\ 4 & 5 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 3 & 1 & 1 \\ 1 & 5 & 2 \\ 1 & 2 & 6 \end{pmatrix}, \tag{1.5}$$

の時、固有値、固有ベクトルは下記の通りになる。

$$\lambda_1 \approx -1.15215, \quad \mathbf{x}_1 \approx \begin{pmatrix} -0.584768 \\ -0.807755 \\ 1 \end{pmatrix}, \tag{1.6a}$$

$$\lambda_2 \approx -0.331689, \quad \mathbf{x}_2 \approx \begin{pmatrix} -12.7043 \\ 9.01249 \\ 1 \end{pmatrix}, \tag{1.6b}$$

$$\lambda_3 \approx 1.21623, \quad \mathbf{x}_3 \approx \begin{pmatrix} 1.25794 \\ 0.699342 \\ 1 \end{pmatrix}, \tag{1.6c}$$

途中、行列 B は下記のようにコレスキー分解されている。

$$B = U^{\dagger}U, \quad U = \begin{pmatrix} \sqrt{3} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 0 & \sqrt{\frac{14}{3}} & \frac{5}{\sqrt{42}} \\ 0 & 0 & \sqrt{\frac{71}{14}} \end{pmatrix}, \tag{1.7}$$

1.3 プログラム

MKL[1] を利用したプログラムと実行結果を載せる。

ソースコード 1: MKL を利用した実対称行列の一般固有値問題を解くプログラム

```
1  program main
2    implicit none
3    integer::i,j,N
4    double precision,allocatable::A(:,:),B(:,:),E(:),evec(:)
5
6    N = 3
7    allocate(A(1:N,1:N),B(1:N,1:N))
8    A = 0d0
9    B = 0d0
10
11 ! A(1,1:N) = (/1d0,2d0,4d0/)
12 ! A(2,1:N) = (/2d0,1d0,5d0/)
13 ! A(3,1:N) = (/4d0,5d0,2d0/)
14 ! B(1,1:N) = (/3d0,1d0,1d0/)
15 ! B(2,1:N) = (/1d0,5d0,2d0/)
16 ! B(3,1:N) = (/1d0,2d0,6d0/)
17
18   A(1,1:N) = (/1d0,2d0,4d0/)
19   A(2,1:N) = (/0d0,1d0,5d0/)
20   A(3,1:N) = (/0d0,0d0,2d0/)
21   B(1,1:N) = (/3d0,1d0,1d0/)
22   B(2,1:N) = (/0d0,5d0,2d0/)
23   B(3,1:N) = (/0d0,0d0,6d0/)
24
25   allocate(E(1:N))
26   E = 0d0
27
28   ! A \mathbf{z} = \lambda B \mathbf{z}
29   call generalEvpSymMat(N,A,B,E)
30
31   allocate(evec(1:N))
32   evec = 0d0
33   do j = 1, N
34      write(6,*)E(j)
35      evec(1:N) = A(1:N,j)
36      do i=1, N
37         write(6,*)j, evec(i)/evec(N)
38      enddo
39      write(6,*)
40   enddo
41
42   stop
43 end program main
```

```
44
45 subroutine generalEvpSymMat(N,A,B,E)
46   !---sikinote
47   implicit none
48   integer,intent(in)::N
49   double precision,intent(inout)::B(1:N,1:N)
50   double precision,intent(inout)::A(1:N,1:N)
51   double precision,intent(out)::E(1:N)
52   !
53   ! Solve General Eigenvalue problems of
54   ! A \mathbf{z} = \lambda B \mathbf{z}
55   !
56   ! Input | A: N x N, Real symmetric matrix (Hermite matrix)
57   ! Only need the upper triangular elements
58   ! B: N x N, Real symmetric matrix (Hermite matrix)
59   ! Only need the upper triangular elements
60   ! Output | E: N x 1, N Eigenvalues
61   ! A: N x N, Eigenvectors A(1:N,j) of E(j)
62   !
63
64   ! Reduce general EVP to standard EVP
65   call reducedToStandardForm(N,A,B)
66
67   ! Slove standard EVP for real symmetric matrix
68   call diagonalizeHermite(N,A,E)
69
70   ! Obtain eigenvectors of general EVP from its of standard EVP
71   call solveLseUpperTriangleMatrix(N,B,N,A)
72
73   return
74 end subroutine generalEvpSymMat
75
76 subroutine reducedToStandardForm(N,A,B)
77   implicit none
78   integer,intent(in)::N
79   double precision,intent(inout)::B(1:N,1:N)
80   double precision,intent(inout)::A(1:N,1:N)
81   !
82   ! Make matrix C from A and B of
83   ! from: A \mathbf{z} = \lambda B \mathbf{z}
84   ! to : C \mathbf{y} = \lambda \mathbf{y}
85   !
86   ! B = U^{T}U
87   ! \mathbf{z} = U^{-1}\mathbf{y}
88   !
89   ! Input | A: N x N, Real symmetric matrix (Hermite matrix)
90   ! Only need the upper triangular elements
91   ! | B: N x N, Real symmetric matrix (Hermite matrix)
92   ! Only need the upper triangular elements
93   ! Positive definite matrix
94   !
95   ! Output | B: N x N, Overwrited by the Cholesky matrix U of input B=U^{T}U
96   ! U is Upper triangular matrix (U(i,j<i)=0)
97   ! | A: N x N, Overwrited by the Real symmetric matrix C, C=(U^{T})^{-1}AU^{-1}
98   ! Only return the upper triangular elements
99   ! Upper triangular matrix
100  !
101  integer::lda,info
102  character(1)::uplo
103  integer::itype,ldb
104
105  uplo="U" ! B is Upper triangular matrix
106  lda=N
107  call dpotrf(uplo, N, B, lda, info)
108  if(info.ne.0)then
109     write(6,'(A)')"***Error at subroutine dpotrf"
110     write(6,'(A,i0)')" program stop, info --> ",info
111     stop
112  endif
113
114  itype=1 ! Type: A \mathbf{z} = \lambda B \mathbf{z}
115  lda=N
116  ldb=N
117  call dsygst(itype, uplo, N, A, lda, B, ldb, info)
118  if(info.ne.0)then
119     write(6,'(A)')"***Error at subroutine dsygst in Diagonalize"
120     write(6,'(A,i0)')" program stop, info --> ",info
121     stop
122  endif
123
124  return
```

```
125 end subroutine reducedToStandardForm
126
127 subroutine diagonalizeHermite(N,A,ev)
128   !---sikinote
129   implicit none
130   integer::N
131   double precision::A(1:N,1:N),ev(1:N)
132   !for Hermite matrix
133
134   integer::lda,lwork,liwork,info
135   double precision,allocatable::work(:)
136   double precision::qw(1:3)
137   integer,allocatable::iwork(:)
138   integer::qiw(1:3)
139   character(1)::job,uplo
140
141   job="V"
142   uplo="U"
143   lda=N
144
145   !size query of dsyevd or zheevd parameter.
146   call dsyevd(job,uplo,N,0,lda,0,qw,-1,qiw,-1,info)
147   if(info.ne.0)then
148      write(6,'(A)')" something wrong at dsyevd(size query) in Diagonalize"
149      write(6,'(A)')" program stop"
150      write(6,'(A,i0)')" info --> ",info
151      stop
152   endif
153
154   !just to be safe, +1
155   lwork=idint(qw(1))+1
156   liwork=qiw(1)+1
157   allocate(work(1:lwork),iwork(1:liwork))
158   work(1:lwork)=0.d0
159   iwork(1:liwork)=0
160
161   !diagonalise.
162   call dsyevd(job,uplo,N,A,lda,ev,work,lwork,iwork,liwork,info)
163   if(info.ne.0)then
164      write(6,'(A)')" something wrong at subroutine dsyevd in Diagonalize"
165      write(6,'(A)')" program stop"
166      write(6,'(A,i0)')" info --> ",info
167      stop
168   endif
169
170   deallocate(work,iwork)
171
172   return
173 end subroutine diagonalizeHermite
174
175 subroutine solveLseUpperTriangleMatrix(N,A,ldb,B)
176   !---sikinote
177   implicit none
178   integer,intent(in)::N,ldb
179   double precision,intent(in)::A(1:N,1:N)
180   double precision,intent(inout)::B(1:N,1:N)
181   !
182   ! Slove linear simultaneous equation (LSE)
183   ! A(1:N,1:N) x = B(1:N,j) for each j
184   !
185   ! Input | A: N x N, Upper triangular real matrix (U(i,j<i)=0)
186   ! | B: N x ldb, Right hand side of B(1:N,j)
187   !
188   ! Output | B: N x ldb, Overwrited by the solution x
189   !
190   integer::lda,info,nrhs
191   character(1)::uplo,trans,diag
192
193   uplo = 'U' ! Upper trianglar matrix
194   trans = 'N' ! slove A(1:N,1:N) x(1:N,j) = B(1:N,j)
195   diag = 'N'
196   nrhs = N
197   lda = N
198   call dtrtrs(uplo, trans, diag, n, nrhs, A, lda, B, ldb, info)
199   if(info.ne.0)then
200      write(6,'(A)')"***Error at subroutine dtrtrs"
201      write(6,'(A,i0)')" program stop, info --> ",info
202      stop
203   endif
204
205   return
```

```
206 end subroutine solveLseUpperTriangleMatrix
```

ソースコード 2: 実行結果

```
1  $ ./a.out
2    -1.1521485211112101
3                      1 -0.58476768299560977
4                      1 -0.80775482423872369
5                      1 1.0000000000000000
6
7   -0.33168880188026734
8                      2 -12.704343950958979
9                      2 9.0124895755548664
10                     2 1.0000000000000000
11
12     1.2162316891886606
13                     3 1.2579365740025481
14                     3 0.69934198729297192
15                     3 1.0000000000000000
```

Reference

[1] インテル マス・カーネル・ライブラリーリファレンスマニュアル (2006 年)

[2] 桂田 祐史著,『一般化固有値問題』, (2003), http://nalab.mind.meiji.ac.jp/~mk/labo/text/generalized-eigenvalue-problem.pdf