Algorithms in the Nashlib set in various programming languages – Part 2

John C Nash, retired professor, University of Ottawa   Peter Olsen, retired ??

11/01/2021

Contents

Abstract

Algorithms 10 and 13-15 from the book Nash (1979) are implemented in a variety of programming languages including Fortran, BASIC, Pascal, Python and R. These concern the eigensolutions of a real, symmetric matrix and its generalization to include a symmetric positive-definite metric.

?? Need to standardize the examples!!

Overview of this document

This section is repeated for each of the parts of Nashlib documentation.

A companion document **Overview of Nashlib and its Implementations** describes the process and computing environments for the implementation of Nashlib algorithms. This document gives comments and/or details relating to implementations of the algorithms themselves.

Note that some discussion of the reasoning behind certain choices in algorithms or implementations are given in the Overview document.

Algorithm 10 – Inverse iteration via Gaussian elimination

The purpose of this algorithm is to find a single eigensolution of a matrix A via inverse iteration. That is, we want solutions (e,x) of

$$Ax = ex$$

The programs do not require a symmetric matrix, which leaves open the possibility that a solution may not exist in the unsymmetric case.

Fortran

The Algorithm 10 code:

```
      SUBROUTINE A10GII(W,NW,N,N2,X,Y,SHIFT,EPS,LIMIT,EV,IPR)
C  ALGORITHM 10
C  J.C. NASH   JULY 1978, FEBRUARY 1980, APRIL 1989
C  INVERSE ITERATION VIA GAUSS ELIMINATION
C  SOLVES EIGENPROBLEM A*X=EV*B*X FOR EIGENSOLUTION (EX,X)
C  VECTOR NORMALISED SO LARGEST ELEMENT IS 1.0
C  W=WORKING ARRAY HAVING INITIALLY A IN COLUMNS 1 TO N
C                                  B IN COLUMNS N+1 TO 2*N=N2
C  NW=FIRST DIMENSION OF W
C  N=ORDER OF PROBLEM
C  N2=2*N = NO. OF COLUMNS IN W
C  X = INITIAL GUESS FOR EIGENVECTOR - SHOULD NOT BE NULL
C  Y = WORKING VECTOR
C  X & Y OF LENGTH N AT LEAST
C  SHIFT = SHIFT TO TRANSFORM PROBLEM TO ONE WITH EV CLOSEST TO SHIFT
C    WITH EV CLOSEST TO SHIFT
C  EPS=MACHINE PRECISION--SMALLEST NUMBER S.T. 1.0+EPS.GT.1.0
C  LIMIT=UPPER BOUND TO NUMBER OF ITERATIONS
C       = ON OUTPUT THE NUMBER OF ITERATIONS USED
C  EV=EIGENVALUE CALCULATED
C  IPR=PRINT CHANNEL. IPR=0 SUPPRESSES PRINTING.
C  STEP 0
      INTEGER N,N2,NW,LIMIT,COUNT,I,J,JN,K,N1,I1
      REAL W(NW,N2),X(N),Y(N),EPS,SHIFT,EV,S,T,P
C  SAFETY CHECK
      IF(N2.NE.2*N)STOP
C  STEP 1
      T=0.0
      DO 10 I=1,N
        Y(I)=0.0
        S=0.0
        DO 5 J=1,N
          JN=J+N
          W(I,J)=W(I,J)-SHIFT*W(I,JN)
          S=S+ABS(W(I,J))
   5    CONTINUE
        IF(T.LT.S)T=S
  10  CONTINUE
      T=T*EPS
C  STEP 2
      N1=N-1
```

```
      DO 100 I=1,N1
C   STEP 3
        S=ABS(W(I,I))
        K=I
        I1=I+1
        DO 20 J=I1,N
          IF(ABS(W(J,I)).LE.S)GOTO 20
          S=ABS(W(J,I))
          K=J
  20    CONTINUE
      IF(S.GT.0.0)GOTO 30
C   STEP 4
        W(I,I)=T
        GOTO 100
C   STEP 5
  30    IF(K.EQ.I)GOTO 50
C   STEP 6
        DO 40 J=I,N2
          S=W(I,J)
          W(I,J)=W(K,J)
          W(K,J)=S
  40    CONTINUE
C   STEP 7
  50    DO 80 J=I1,N
          S=W(J,I)/W(I,I)
          DO 70 K=I,N2
            W(J,K)=W(J,K)-S*W(I,K)
  70      CONTINUE
  80    CONTINUE
C   STEP 8
 100  CONTINUE
C   STEP 9
      IF(ABS(W(N,N)).EQ.0.0)W(N,N)=T
C   STEP 10
      COUNT=0
C   STEP 11
 110  COUNT=COUNT+1
      M=N
      S=X(N)
      X(N)=Y(N)
      Y(N)=S/W(N,N)
      P=ABS(Y(N))
C   STEP 12
      DO 130 JN=1,N1
        I=N-JN
        S=X(I)
        X(I)=Y(I)
        I1=I+1
        DO 120 J=I1,N
          S=S-W(I,J)*Y(J)
 120    CONTINUE
        Y(I)=S/W(I,I)
        IF(ABS(Y(I)).LE.P)GOTO 130
```

```
        P=ABS(Y(I))
        M=I
 130  CONTINUE
C  STEP 13
      EV=SHIFT+X(M)/Y(M)
C  STEP 14
      P=Y(M)
      M=0
      DO 140 I=1,N
        Y(I)=Y(I)/P
        IF(FLOAT(N)+Y(I).EQ.FLOAT(N)+X(I))M=M+1
 140  CONTINUE
      IF(IPR.GT.0)WRITE(IPR,960)COUNT,EV,M
 960  FORMAT(14H ITERATION NO.,I4,14H   APPROX. EV=,1PE16.8,5X,I4,
     *27H VECTOR ELEMENTS TEST EQUAL)
C  STEP 15
      IF(M.EQ.N)GOTO 200
      IF(COUNT.GT.LIMIT)GOTO 200
C  STEP 16
      DO 160 I=1,N
        S=0.0
        DO 150 J=1,N
          JN=J+N
          S=S+W(I,JN)*Y(J)
 150    CONTINUE
        X(I)=S
 160  CONTINUE
C  STEP 17
      GOTO 110
 200  LIMIT=COUNT
      RETURN
      END
```

Example output

We illustrate by finding a single eigensolution of the Hilbert segments of order 5 and 10. ?? Do we want to swap in the Frank matrix (the computations are generally easier)?

```
## #!/bin/bash
gfortran ../fortran/d10.f
mv ./a.out ../fortran/d10.run
../fortran/d10.run
```

```
##  ORDER=   5
##  USING SHIFT OF     0.000000
##  CONVERGED TO EV=  3.29019417E-06  IN    5  ITERATIONS
##  X(  1)= -0.80475118E-02
##  X(  2)=  0.15210588E+00
##  X(  3)= -0.65976608E+00
##  X(  4)=  0.10000000E+01
##  X(  5)= -0.49041715E+00
##  RESIDUAL(  1)= -0.74505806E-08
##  RESIDUAL(  2)=  0.00000000E+00
##  RESIDUAL(  3)= -0.74505806E-08
##  RESIDUAL(  4)= -0.37252903E-08
```

```
##  RESIDUAL(  5)= -0.37252903E-08
##  ORDER=  10
##  USING SHIFT OF     0.000000
##  CONVERGED TO EV=  1.26338462E-09  IN  101  ITERATIONS
##  X(  1)=  0.50510102E-05
##  X(  2)= -0.61139709E-03
##  X(  3)=  0.65672603E-02
##  X(  4)= -0.65278080E-02
##  X(  5)= -0.94817474E-01
##  X(  6)=  0.25418818E+00
##  X(  7)=  0.62985711E-01
##  X(  8)= -0.86295480E+00
##  X(  9)=  0.10000000E+01
##  X( 10)= -0.35897413E+00
##  RESIDUAL(  1)=  0.00000000E+00
##  RESIDUAL(  2)=  0.00000000E+00
##  RESIDUAL(  3)= -0.18626451E-08
##  RESIDUAL(  4)= -0.11175871E-07
##  RESIDUAL(  5)= -0.37252903E-08
##  RESIDUAL(  6)=  0.18626451E-08
##  RESIDUAL(  7)= -0.18626451E-08
##  RESIDUAL(  8)= -0.18626451E-08
##  RESIDUAL(  9)= -0.37252903E-08
##  RESIDUAL( 10)=  0.37252903E-08
```

BASIC

Listing

```
5 DIM A(10, 20),X(10),Y(10)
10 PRINT "GII JULY 25 77 ALG 10"
20 PRINT "GAUSS ELIMINATION FOR INVERSE ITERATION"
30 PRINT "ORDER=",
40 READ N
50 PRINT N
55 IF N <= 0 THEN QUIT : REM BWBASIC VARIANT
60 GOSUB 1500: REM BUILD OR INPUT MATRIX
70 GOSUB 2000: REM PUT METRIC IN RIGHT HALF OF A
75 GOSUB 1000: REM INITIAL GUESS TO VECTOR
80 LET K9=0 : REM SHIFT OF 0 FOR THIS EXAMPLE
90 PRINT "SHIFT=",K9
95 LET E9=K9
100 REM PRINT
105 LET T2=N: REM FACTOR FOR CONVERGENCE TEST
110 LET T1=0: REM STEP 1
120 FOR I=1 TO N
130 LET Q=0
140 FOR J=1 TO N
150 LET A(I,J)=A(I,J)-K9*A(I,J+N)
160 LET S=S+ABS(A(I,J))
170 NEXT J
180 IF T1>=S THEN GOTO 200
190 LET T1=S
200 NEXT I
```

```
205 LET T1=T1*1.0E-7: REM NS 8 DIGIT BASIC
210 FOR I=1 TO N-1: REM STEP 2
218 LET S=ABS(A(I,I)): REM STEP 3
226 LET K=I
234 FOR J=I+1 TO N
242 IF ABS(A(J,I))<=S THEN GOTO 266
250 LET S=ABS(A(J,I))
258 LET K=J
266 NEXT J
274 IF S>0 THEN GOTO 298: REM STEP 4
282 LET A(I,I)=T1
290 GOTO 394
298 IF K=I THEN GOTO 346: REM STEP 5
306 FOR J=I TO 2*N: REM STEP 6
314 LET S=A(I,J)
322 LET A(I,J)=A(K,J)
330 LET A(K,J)=S
338 NEXT J
346 FOR J=I+1 TO N: REM STEP 7
354 LET S=A(J,I)/A(I,I)
362 FOR K=I TO 2*N
370 LET A(J,K)=A(J,K)-S*A(I,K)
378 NEXT K
386 NEXT J
394 NEXT I: REM STEP 8
402 IF ABS(A(N,N))>0 THEN GOTO 420: REM STEP 9
410 LET A(N,N)=T1
420 LET I9=0: REM STEP 10
430 LET I9=I9+1: REM STEP 11
440 LET M=N
445 LET S=X(N)
450 LET X(N)=Y(N)
455 LET Y(N)=S/A(N,N)
460 LET P=ABS(Y(N))
470 FOR I=(N-1) TO 1 STEP -1: REM STEP 12
480 LET S=X(I)
485 LET X(I)=Y(I)
490 FOR J=I+1 TO N
500 LET S=S-A(I,J)*Y(J)
510 NEXT J
520 LET Y(I)=S/A(I,I)
530 IF ABS(Y(I))<=P THEN GOTO 560
540 LET M=I
550 LET P=ABS(Y(I))
560 NEXT I
570 LET E8=K9+X(M)/Y(M): REM STEP 13
580 REM PRINT "APPROX EV=",E8
600 LET P=Y(M): REM STEP 14
610 LET M=0
620 FOR I=1 TO N
630 LET Y(I)=Y(I)/P
635 IF T2+Y(I)<>T2+X(I) THEN GOTO 640
636 LET M=M+1
```

```
640 NEXT I
644 IF M=N THEN GOTO 730: REM STEP 15 -- CONVERGENCE TEST
645 IF I9>100 THEN GOTO 730: REM LIMIT SET AT 100
650 FOR I=1 TO N: REM STEP 16
660 LET S=0
670 FOR J=1 TO N
680 LET S=S+A(I,J+N)*Y(J)
690 NEXT J
700 LET X(I)=S
710 NEXT I
720 GOTO 430: REM STEP 17
725 REM STEP 18 -- END AND RESIDUALS
730 PRINT "CONVERGED TO EV=",E8," IN ",I9," ITNS"
735 PRINT M," EQUAL CPNTS IN VECTOR BETWEEN ITERATIONS"
740 GOSUB 1500: REM GET MATRIX AGAIN
750 GOSUB 2000: REM GET METRIC AGAIN
755 LET S=0: REM COMPUTE VECTOR INNER PRODUCT
760 FOR I=1 TO N
770 FOR J=1 TO N
780 LET S=S+Y(I)*A(I,J+N)*Y(J)
790 NEXT J
800 NEXT I
810 LET S=1/SQR(S)
815 PRINT "VECTOR"
820 FOR I=1 TO N
830 LET Y(I)=Y(I)*S: REM VECTOR NORMALIZATION
840 PRINT Y(I);
845 IF I=5*INT(I/5) THEN PRINT
850 NEXT I
855 PRINT
860 PRINT "RESIDUALS"
870 FOR I=1 TO N
880 LET S=0
890 FOR J=1 TO N: REM MATRIX * VECTOR - VALUE * METRIC * VECTOR
900 LET S=S+(A(I,J)-E8*A(I,J+N))*Y(J)
910 NEXT J
920 PRINT S;
925 IF 5*INT(I/5)=I THEN PRINT
930 NEXT I
940 PRINT
950 GOTO 40 : REM NEXT TRY
960 DATA 5, 10, -1
1000 REM          INITIAL X
1010 FOR I=1 TO N
1020 LET X(I)=1: REM MAY BE A POOR CHOICE
1030 NEXT I
1040 RETURN
1500 REM A IN FOR FRANK MATRIX
1505 PRINT "FRANK MATRIX"
1510 FOR I=1 TO N
1520 FOR J=1 TO I
1530 A(I,J)=J
1540 A(J,I)=J
```

```
1550 NEXT J
1560 NEXT I
1570 RETURN
2000 REM UNIT B IN RIGHT HALF OF MATRIX
2010 FOR I=1 TO N
2020 FOR J=1 TO N
2030 A(I,J+N)=0
2040 NEXT J
2050 A(I,I+N)=1
2060 NEXT I
2070 RETURN
```

Example output

In this case we use the Frank matrix for our test.

```
bwbasic ../BASIC/a10.bas >../BASIC/a10.out
# echo "done"
```

```
Bywater BASIC Interpreter/Shell, version 2.20 patch level 2

Copyright (c) 1993, Ted A. Campbell

Copyright (c) 1995-1997, Jon B. Volkoff


GII JULY 25 77 ALG 10
GAUSS ELIMINATION FOR INVERSE ITERATION
ORDER=
 5
FRANK MATRIX
SHIFT=        0
CONVERGED TO EV=            0.2715541     IN            101            ITNS
 1             EQUAL CPNTS IN VECTOR BETWEEN ITERATIONS
FRANK MATRIX
VECTOR
 0.3260187 -0.5485287 0.5968848 -0.4557341 0.1698911

RESIDUALS
 -0 0 -0 -0 0

 10
FRANK MATRIX
SHIFT=        0
CONVERGED TO EV=            0.2556738     IN            101            ITNS
 1             EQUAL CPNTS IN VECTOR BETWEEN ITERATIONS
FRANK MATRIX
VECTOR
 0.1281224 -0.2449948 0.3403202 -0.405639 0.4350771
 -0.4258922 0.3787616 -0.2977698 0.1900823 -0.0653188

RESIDUALS
 -0.0000087 0.0000141 -0.0000143 0.000009 -0
 -0.0000097 0.0000166 -0.0000183 0.0000141 -0.0000053
```

```
-1
```

Pascal

Listing

```
procedure gii(nRow : integer;
             var A : rmatrix;
             var Y : rvector;
             var shift : real;
             var itcount: integer);

var
  i, itlimit, j, m, msame, nRHS :integer;
  ev, s, t, tol : real;
  X : rvector;

begin
  itlimit:=itcount;
  nRHS:=nRow;
  tol:=Calceps;
  s:=0.0;
  for i:=1 to nRow do
  begin
    X[i]:=Y[i];
    Y[i]:=0.0;
    for j:=1 to nRow do
    begin
      A[i,j]:=A[i,j]-shift*A[i,j+nRow];
      s:=s+abs(A[i,j]);
    end;
  end;
  tol:=tol*s;
  gelim(nRow, nRHS, A, tol);
  itcount:=0;
  msame :=0;
  while (msame<nRow) and (itcount<itlimit) do
  begin
    itcount:=itcount+1;
    m:=nRow; s:=X[nRow];
    X[nRow]:=Y[nRow];
    if abs(A[nRow,nRow])<tol then Y[nRow]:=s/tol
                             else Y[nRow]:=s/A[nRow,nRow];
    t:=abs(Y[nRow]);
    for i:=(nRow-1) downto 1 do
    begin
      s:=X[i]; X[i]:=Y[i];
      for j:=(i+1) to nRow do
      begin
        s:=s-A[i,j]*Y[j];
      end;
      if abs(A[i,i])<tol then Y[i]:=s/tol else  Y[i]:=s/A[i,i];
      if abs(Y[i])>t then
```

```
      begin
        m:=i; t:=abs(Y[i]);
      end;
    end;
    ev:=shift+X[m]/Y[m];
(*    writeln('Iteration ',itcount,'  approx. ev=',ev);*)

    t:=Y[m]; msame:=0;
    for i:=1 to nRow do
    begin
      Y[i]:=Y[i]/t;
      if reltest+Y[i] = reltest+X[i] then msame:=msame+1;

    end;

    if msame<nRow then
    begin
      for i:=1 to nRow do
      begin
        s:=0.0;
        for j:=1 to nRow do s:=s+A[i,j+nRow]*Y[j];
        X[i]:=s;
      end;
    end;
  end;
  if itcount>=itlimit then itcount:=-itcount;
  shift:=ev;
end;
```

Example output

```
fpc ../Pascal2021/dr10.pas
# copy to run file
mv ../Pascal2021/dr10 ../Pascal2021/dr10.run
../Pascal2021/dr10.run <../Pascal2021/dr10p.in >../Pascal2021/dr10p.out
```

```
## Free Pascal Compiler version 3.0.4+dfsg-23 [2019/11/25] for x86_64
## Copyright (c) 1993-2017 by Florian Klaempfl and others
## Target OS: Linux for x86-64
## Compiling ../Pascal2021/dr10.pas
## Linking ../Pascal2021/dr10
## /usr/bin/ld.bfd: warning: link.res contains output sections; did you forget -T?
## 402 lines compiled, 0.3 sec
```

```
order of problem (n) = 5
Provide the A matrix
A matrix
   1.00000    1.00000    1.00000    1.00000    1.00000
   1.00000    2.00000    2.00000    2.00000    2.00000
   1.00000    2.00000    3.00000    3.00000    3.00000
   1.00000    2.00000    3.00000    4.00000    4.00000
   1.00000    2.00000    3.00000    4.00000    5.00000
B matrix set to unit matrix
B matrix
```

```
   1.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    1.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    1.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    1.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    1.00000
shift=? 0.0000000000000000E+000
alg05.pas -- Gauss elimination with partial pivoting
Gauss elimination complete -- determinant =  2.4000000000000000E+001

Not converged. Approximate eigenvalue= 2.7155412933904033E-001 after 100 iterations
Eigenvector
   0.54620   -0.91899    1.00000   -0.76352    0.28463
Rayleigh quotient =  2.7155412933882112E-001   sumsquared err= 0.000000000
shift=? 1.0000000000000001E+032
order of problem (n) = 10
Provide the A matrix
A matrix
   1.00000    1.00000    1.00000    1.00000    1.00000    1.00000    1.00000
   1.00000    1.00000    1.00000
   1.00000    2.00000    2.00000    2.00000    2.00000    2.00000    2.00000
   2.00000    2.00000    2.00000
   1.00000    2.00000    3.00000    3.00000    3.00000    3.00000    3.00000
   3.00000    3.00000    3.00000
   1.00000    2.00000    3.00000    4.00000    4.00000    4.00000    4.00000
   4.00000    4.00000    4.00000
   1.00000    2.00000    3.00000    4.00000    5.00000    5.00000    5.00000
   5.00000    5.00000    5.00000
   1.00000    2.00000    3.00000    4.00000    5.00000    6.00000    6.00000
   6.00000    6.00000    6.00000
   1.00000    2.00000    3.00000    4.00000    5.00000    6.00000    7.00000
   7.00000    7.00000    7.00000
   1.00000    2.00000    3.00000    4.00000    5.00000    6.00000    7.00000
   8.00000    8.00000    8.00000
   1.00000    2.00000    3.00000    4.00000    5.00000    6.00000    7.00000
   8.00000    9.00000    9.00000
   1.00000    2.00000    3.00000    4.00000    5.00000    6.00000    7.00000
   8.00000    9.00000   10.00000
B matrix set to unit matrix
B matrix
   1.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000
   0.00000    1.00000    0.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000
   0.00000    0.00000    1.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    1.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    1.00000    0.00000    0.00000
   0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    0.00000    1.00000    0.00000
   0.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    1.00000
   0.00000    0.00000    0.00000
```

```
   0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
   1.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    1.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    1.00000
shift=? 0.0000000000000000E+000
alg05.pas -- Gauss elimination with partial pivoting
Gauss elimination complete -- determinant =  3.6288000000000000E+005

Not converged. Approximate eigenvalue= 2.5567344134720177E-001 after 100 iterations
Eigenvector
   0.29440   -0.56298    0.78208   -0.93226    1.00000   -0.97898    0.87071
  -0.68457    0.43702   -0.15018
Rayleigh quotient =  2.5567965533013154E-001   sumsquared err= 0.000000009
shift=? 1.0000000000000001E+032
order of problem (n) = 0
```

Algorithm 13

Fortran

Listing - Algorithim 13

```
      SUBROUTINE A13ESV(N,A,NA,EPS,H,ISWP,IPR,Z)
C  ALGORITHM 13 EIGENPROBLEM OF A REAL SYMMETRIC MATRIX VIA SVD
C  J.C. NASH   JULY 1978, FEBRUARY 1980, APRIL 1989
C  N      =  ORDER OF PROBLEM
C  A      =  ARRAY CONTAINING MATRIX FOR WHICH EIGENVALUES ARE TO BE
C            COMPUTED.  RETURNS EIGENVECTORS AS COLUMNS
C  NA     =  FIRST DIMENSION OF A
C  EPS    =  MACHINE PRECISION
C  H      =  A NUMBER LARGER THAN ANY POSSIBLE EIGENVALUE. CHANGED
C            DURING EXECUTION. DO NOT ENTER AS A CONSTANT
C  ISWP   =  LIMIT ON SWEEPS (INPUT). SWEEPS USED (OUTPUT).
C  IPR    =  PRINT CHANNEL. IPR.GT.0 FOR PRINTING.
C  Z      =  EIGENVALUES (OUTPUT)
C  STEP 0
      INTEGER N,NA,ISWP,IPR,LISWP,I,J,COUNT,N1,J1
      REAL A(NA,N),EPS,H,V,Z(N),P,Q,R,S,C
      LISWP=ISWP
      ISWP=0
      N1=N-1
C  STEP 1
      DO 5 I=1,N
        V=A(I,I)
        DO 3 J=1,N
          IF(J.EQ.I)GOTO 3
          V=V-ABS(A(I,J))
   3    CONTINUE
        IF(V.LT.H)H=V
   5  CONTINUE
      IF(H.LE.EPS)GOTO 6
      H=0.0
      GOTO 30
   6  H=H-SQRT(EPS)
C  STEP 2
      DO 15 I=1,N
        A(I,I)=A(I,I)-H
  15  CONTINUE
C  STEP 3
  30  COUNT=0
C  CHECK FOR ORDER 1 PROBLEMS AND SKIP WORK
      IF(N.EQ.1)GOTO 160
      ISWP=ISWP+1
      IF(ISWP.GT.LISWP)GOTO 160
C  STEP 4
      DO 140 J=1,N1
C  STEP 5
        J1=J+1
        DO 130 K=J1,N
C  STEP 6
```

```
          P=0.0
          Q=0.0
          R=0.0
          DO 65 I=1,N
            P=P+A(I,J)*A(I,K)
            Q=Q+A(I,J)**2
            R=R+A(I,K)**2
 65       CONTINUE
C  STEP 7
          IF(1.0.LT.1.0+ABS(P/SQRT(Q*R)))GOTO 80
          IF(Q.LT.R)GOTO 80
          COUNT=COUNT+1
          GOTO 130
 80       Q=Q-R
C  STEP 8
          V=SQRT(4.0*P*P+Q*Q)
          IF(V.EQ.0.0)GOTO 130
C  STEP 9
          IF(Q.LT.0.0)GOTO 110
C  STEP 10
          C=SQRT((V+Q)/(2.0*V))
          S=P/(V*C)
          GOTO 120
C  STEP 11
 110      S=SQRT((V-Q)/(2.0*V))
          IF(P.LT.0.0)S=-S
          C=P/(V*S)
C  STEP 12
 120      DO 125 I=1,N
            V=A(I,J)
            A(I,J)=V*C+A(I,K)*S
            A(I,K)=-V*S+A(I,K)*C
 125      CONTINUE
C  STEP 13
 130    CONTINUE
C  STEP 14
 140  CONTINUE
C  STEP 15
      IF(IPR.GT.0)WRITE(IPR,970)ISWP,COUNT
 970  FORMAT( 9H AT SWEEP,I4,2X,I4,18H ROTATIONS SKIPPED)
      IF(COUNT.LT.N*(N-1)/2)GOTO 30
C  STEP 16
 160  DO 168 J=1,N
        S=0.0
        DO 162 I=1,N
          S=S+A(I,J)**2
 162    CONTINUE
        S=SQRT(S)
        DO 164 I=1,N
          A(I,J)=A(I,J)/S
 164    CONTINUE
        R=S+H
        Z(J)=R
```

```
 168  CONTINUE
C  STEP 17
 170  RETURN
      END
```

Example output

```
## #!/bin/bash
gfortran ../fortran/dr13.f
mv ./a.out ../fortran/dr13.run
../fortran/dr13.run < ../fortran/dr13.in
```

```
##  ORDER N=   2
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     1 ROTATIONS SKIPPED
##  CONVERGED IN    3 SWEEPS
##  MAX. ABS. RESIDUAL=  1.47663954E-07  POSN   1   1
##  MAX. ABS. INNER PRODUCT=  0.00000000E+00  POSN   0   0
##  EIGENVALUE  1=  2.61803412E+00
##    0.52573115E+00  0.85065079E+00
##  EIGENVALUE  2=  3.81966025E-01
##   -0.85065079E+00  0.52573115E+00
##  ORDER N=   4
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     1 ROTATIONS SKIPPED
##  AT SWEEP   4     6 ROTATIONS SKIPPED
##  CONVERGED IN    4 SWEEPS
##  MAX. ABS. RESIDUAL=  6.81894505E-07  POSN   3   1
##  MAX. ABS. INNER PRODUCT=  4.60000820E-08  POSN   1   3
##  EIGENVALUE  1=  8.29086018E+00
##    0.22801343E+00  0.42852512E+00  0.57735032E+00  0.65653849E+00
##  EIGENVALUE  2=  1.00000048E+00
##   -0.57735056E+00 -0.57735002E+00  0.66493286E-07  0.57735020E+00
##  EIGENVALUE  3=  4.26022291E-01
##    0.65653813E+00 -0.22801332E+00 -0.57735056E+00  0.42852539E+00
##  EIGENVALUE  4=  2.83118486E-01
##   -0.42852521E+00  0.65653872E+00 -0.57735002E+00  0.22801307E+00
##  ORDER N=   4
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     1 ROTATIONS SKIPPED
##  AT SWEEP   4     6 ROTATIONS SKIPPED
##  CONVERGED IN    4 SWEEPS
##  MAX. ABS. RESIDUAL=  6.81894505E-07  POSN   3   1
##  MAX. ABS. INNER PRODUCT=  4.60000820E-08  POSN   1   3
##  EIGENVALUE  1=  8.29086018E+00
##    0.22801343E+00  0.42852512E+00  0.57735032E+00  0.65653849E+00
##  EIGENVALUE  2=  1.00000048E+00
##   -0.57735056E+00 -0.57735002E+00  0.66493286E-07  0.57735020E+00
##  EIGENVALUE  3=  4.26022291E-01
##    0.65653813E+00 -0.22801332E+00 -0.57735056E+00  0.42852539E+00
##  EIGENVALUE  4=  2.83118486E-01
```

```
##   -0.42852521E+00  0.65653872E+00 -0.57735002E+00  0.22801307E+00
##  ORDER N=  10
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     0 ROTATIONS SKIPPED
##  AT SWEEP   4    32 ROTATIONS SKIPPED
##  AT SWEEP   5    44 ROTATIONS SKIPPED
##  AT SWEEP   6    42 ROTATIONS SKIPPED
##  AT SWEEP   7    45 ROTATIONS SKIPPED
##  CONVERGED IN    7 SWEEPS
##  MAX. ABS. RESIDUAL=  2.39280362E-05  POSN   9   1
##  MAX. ABS. INNER PRODUCT=  6.23372785E-08  POSN   6  10
##  EIGENVALUE  1=  4.47660294E+01
##    0.65047376E-01  0.12864168E+00  0.18936241E+00  0.24585304E+00  0.29685175E+00
##    0.34121934E+00  0.37796459E+00  0.40626666E+00  0.42549327E+00  0.43521538E+00
##  EIGENVALUE  2=  5.04890442E+00
##   -0.18936226E+00 -0.34121895E+00 -0.42549327E+00 -0.42549345E+00 -0.34121940E+00
##   -0.18936238E+00 -0.18430426E-06  0.18936227E+00  0.34121925E+00  0.42549381E+00
##  EIGENVALUE  3=  1.87301636E+00
##    0.29685244E+00  0.43521363E+00  0.34121892E+00  0.65048993E-01 -0.24585134E+00
##   -0.42549297E+00 -0.37796640E+00 -0.12864257E+00  0.18936226E+00  0.40626761E+00
##  EIGENVALUE  4=  9.99992371E-01
##   -0.37796077E+00 -0.37796682E+00 -0.31607331E-05  0.37796402E+00  0.37796602E+00
##    0.20208058E-05 -0.37796175E+00 -0.37796679E+00 -0.16900430E-05  0.37796503E+00
##  EIGENVALUE  5=  6.43104553E-01
##    0.42549762E+00  0.18936004E+00 -0.34121791E+00 -0.34121671E+00  0.18935442E+00
##    0.42549407E+00  0.89485920E-05 -0.42549804E+00 -0.18936114E+00  0.34121749E+00
##  EIGENVALUE  6=  4.65229034E-01
##   -0.43522137E+00  0.65052554E-01  0.42549083E+00 -0.12863764E+00 -0.40626609E+00
##    0.18935713E+00  0.37796399E+00 -0.24584912E+00 -0.34122151E+00  0.29685244E+00
##  EIGENVALUE  7=  3.66199493E-01
##    0.40627682E+00 -0.29686981E+00 -0.18934317E+00  0.43520308E+00 -0.12863609E+00
##   -0.34122577E+00  0.37796903E+00  0.65047354E-01 -0.42548791E+00  0.24584727E+00
##  EIGENVALUE  8=  3.07968140E-01
##   -0.34119982E+00  0.42547908E+00 -0.18935996E+00 -0.18936360E+00  0.42549238E+00
##   -0.34120587E+00 -0.21800142E-04  0.34124032E+00 -0.42551416E+00  0.18937302E+00
##  EIGENVALUE  9=  2.73780823E-01
##    0.24583328E+00 -0.40622735E+00  0.42543337E+00 -0.29676920E+00  0.64941816E-01
##    0.18947297E+00 -0.37804705E+00  0.43525901E+00 -0.34123680E+00  0.12864587E+00
##  EIGENVALUE 10=  2.55672455E-01
##   -0.12867406E+00  0.24592136E+00 -0.34131119E+00  0.40634301E+00 -0.43523723E+00
##    0.42545170E+00 -0.37787846E+00  0.29675686E+00 -0.18929419E+00  0.65023489E-01
##  ORDER N=   1
##  CONVERGED IN    0 SWEEPS
##  MAX. ABS. RESIDUAL=  0.00000000E+00  POSN   1   1
##  EIGENVALUE  1=  1.00000000E+00
##    0.10000000E+01
##  ORDER N=   0
```

BASIC

Listing

```
10 PRINT "A13 EIGENSOLUTIONS OF A SYMMETRIC MATRIX VIA SVD"
20 LET N9=20
30 DIM A(N9,N9),V(N9,N9),B(N9,N9),D(N9),Z(N9),W(N9)
40 PRINT "ORDER OF MATRIX: ";
50 READ N
55 PRINT N
60 IF (N <= 0) THEN QUIT
70 LET M=N
80 GOSUB 1500: REM BUILD FRANK MATRIX N BY N
100 GOSUB 3000
110 FOR I=1 TO N
120   PRINT "EV(";I;")=";D(I);"    RQ=";Z(I)
125   GOSUB 500
130   FOR J=1 TO N
140      PRINT V(J,I);" ";
145   NEXT J
150   PRINT
151   PRINT "MAX(ABS(RES))=";S9;":";
152   FOR J=1 TO N
153   PRINT W(J);
154   NEXT J
155   PRINT
160 NEXT I
170 PRINT
200 GOTO 40
300 DATA  4, 0: REM 2, 4, 10, 1, 0
500 LET S9=0: REM RESIDUALS I
510 FOR J=1 TO N
520   LET T9=0
530   FOR K=1 TO N
540     LET T9=T9+B(J,K)*V(K,I)
550   NEXT K
555 rem   PRINT T9;
560   LET T9=T9-D(I)*V(J,I)
565 rem  PRINT T9;
570   IF (ABS(T9) > S9) THEN LET S9=ABS(T9)
580   LET W(J)=T9
590 NEXT J
600 RETURN
1500 REM PREPARE FRANK MATRIX IN A
1510 FOR I=1 TO M
1530 FOR J=1 TO N
1540 IF (I <= J) THEN LET A(I,J)=I ELSE LET A(I,J)=J
1545 LET B(I,J)=A(I,J): REM COPY
1550 NEXT J
1560 NEXT I
1570 RETURN
3000 PRINT "SMEV.NJ ALG 13 DEC 7 78; MINOR MOD 2021-2-11"
3002 LET E1=1E-7: REM CAN BE ADJUSTED TO MACHINE PROPERTIES
3004 IF (N > 1) THEN GOTO 3020
```

```
3006 LET D(1)=A(1,1)
3008 LET V(1,1)=1
3010 LET Z(1)=D(1)
3012 RETURN
3014 REM END SPECIAL CASE N=1
3018 REM DIM A(N,N),V(N,N),B(N,N) (FOR RAYLEIGH QUOTIENT)
3020 LET H=1E+38: REM BIG NUMBER
3025 FOR I=1 TO N
3030   LET V1=A(I,I)
3035   FOR J=1 TO N
3040     LET B(I,J)=A(I,J)
3045     IF I=J THEN GOTO 3055
3050     LET V1=V1-ABS(A(I,J))
3055   NEXT J
3060   IF V1>=H THEN GOTO 3070
3065   LET H=V1
3070 NEXT I
3075 IF H<E1 THEN GOTO 3090
3080 LET H=0
3085 GOTO 3120
3090 LET H=H-SQR(E1)
3095 FOR I=1 TO N
3100   LET A(I,I)=A(I,I)-H
3105 NEXT I
3110 PRINT
3115 PRINT "SCALING BY SUBTRACTION OF",H
3120 LET I9=0
3125 LET N2=N*(N-1)/2
3130 LET N1=N-1
3135 LET N9=N2
3140 LET I9=I9+1
3145 LET N9=0
3150 IF I9<=30 THEN GOTO 3170
3155 PRINT
3160 PRINT "NON-",
3165 GOTO 3355
3170 FOR J=1 TO N1
3175   LET J1=J+1
3180   FOR K=J1 TO N
3185     LET P=0
3190     LET R=0
3195     LET Q=0
3200     FOR I=1 TO N
3205       LET P=P+A(I,J)*A(I,K)
3210       LET Q=Q+A(I,J)*A(I,J)
3215       LET R=R+A(I,K)*A(I,K)
3220     NEXT I
3225     IF 1+ABS(P/SQR(Q*R))>1 THEN GOTO 3235
3230     IF Q>=R THEN GOTO 3320
3235     LET Q=Q-R
3240     LET V1=SQR(4*P*P+Q*Q)
3245     IF V1=0 THEN GOTO 3320
3250     IF Q<0 THEN GOTO 3270
```

```
3255     LET C=SQR((V1+Q)/(2*V1))
3260     LET S=P/(V1*C)
3265     GOTO 3290
3270     LET S=SQR((V1-Q)/(2*V1))
3275     IF P>0 THEN GOTO 3285
3280     LET S=-S
3285     LET C=P/(V1*S)
3290     FOR I=1 TO N
3295       LET V1=A(I,J)
3300       LET A(I,J)=V1*C+A(I,K)*S
3305       LET A(I,K)=-V1*S+A(I,K)*C
3310     NEXT I
3315     GOTO 3325
3320     LET N9=N9+1
3325   NEXT K
3330 NEXT J
3335 IF N9=N2 THEN GOTO 3350
3340 PRINT "SWEEP",I9,"  ",N9,"SMALL P"
3345 GOTO 3140
3350 PRINT
3355 PRINT "CONVERGENCE AT SWEEP",I9
3360 LET V1=0
3365 LET C=0
3370 FOR J=1 TO N
3375   LET Q=0
3380   FOR I=1 TO N
3385     LET Q=Q+A(I,J)^2
3390   NEXT I
3395   LET Q=SQR(Q)
3400   FOR I=1 TO N
3405     LET V(I,J)=A(I,J)/Q
3410   NEXT I
3415   LET Q=Q+H
3420   LET D(J)=Q
3425   GOSUB 3440
3430 NEXT J
3435 RETURN
3440 LET Q=0
3445 FOR I=1 TO N
3450   FOR K=1 TO N
3455     LET Q=Q+V(I,J)*B(I,K)*V(K,J)
3460   NEXT K
3465 NEXT I
3470 LET Z(J)=Q
3475 RETURN
```

Example output

```
bwbasic ../BASIC/a13.bas  >../BASIC/a13.out
# echo "done"
```

```
Bywater BASIC Interpreter/Shell, version 2.20 patch level 2
```

```
Copyright (c) 1993, Ted A. Campbell

Copyright (c) 1995-1997, Jon B. Volkoff


A13 EIGENSOLUTIONS OF A SYMMETRIC MATRIX VIA SVD
ORDER OF MATRIX:  4
SMEV.NJ ALG 13 DEC 7 78; MINOR MOD 2021-2-11

SCALING BY SUBTRACTION OF   -3.0003162
SWEEP         1                          0             SMALL P
SWEEP         2                          0             SMALL P
SWEEP         3                          0             SMALL P
SWEEP         4                          1             SMALL P

CONVERGENCE AT SWEEP        5
EV( 1)= 8.2908594    RQ= 8.2908594
 0.2280134  0.428525  0.5773503  0.6565385
MAX(ABS(RES))= 0: -0 -0 0 0
EV( 2)= 1.0000000    RQ= 1
 -0.5773503  -0.5773503  0  0.5773503
MAX(ABS(RES))= 0: -0 -0 -0 0
EV( 3)= 0.426022    RQ= 0.426022
 0.6565385  -0.2280134  -0.5773503  0.428525
MAX(ABS(RES))= 0: 0 -0 -0 0
EV( 4)= 0.2831186    RQ= 0.2831186
 -0.428525  0.6565385  -0.5773503  0.2280134
MAX(ABS(RES))= 0: -0 -0 0 0

ORDER OF MATRIX:  0
```

Pascal

Listing – Algorithm 13

```
Procedure evsvd(n: integer; var A,V : rmatrix; initev: boolean;
             W : wmatrix; var Z: rvector);

var
  i, j: integer;
  shift, t : real ;

begin
  writeln('alg13.pas -- symmetric matrix eigensolutions via svd');
  shift:=0.0;
  for i:=1 to n do
  begin
    t:=A[i,i];
    for j:=1 to n do
      if i<>j then t:=t-abs(A[i,j]);
    if t<shift then shift:=t;
  end;
  shift:=-shift;
```

```
  if shift<0.0 then shift:=0.0;
  writeln('Adding a shift of ',shift,' to diagonal of matrix.');
  for i:=1 to n do
  begin
    for j:=1 to n do
    begin
      W[i,j]:=A[i,j];
      if i=j then W[i,i]:=A[i,i]+shift;
      if initev then
      begin
        if i=j then W[i+n,i]:=0.0
        else
        begin
          W[i+n,j]:=0.0;
        end;
      end;
    end;
  end;
  if (n > 1) then
     NashSVD(n, n, W, Z)
  else
  begin { order 1 matrix }
     Z[1] := A[1,1]*A[1,1];
     W[2,1]:= 1.0; {Eigenvector!}
  end;
  for i:=1 to n do
  begin
    Z[i]:=sqrt(Z[i])-shift;
    for j:=1 to n do
      V[i,j]:=W[n+i,j];
  end;
end;
```

Example output

```
fpc ../Pascal2021/dr13.pas
# copy to run file
mv ../Pascal2021/dr13 ../Pascal2021/dr13.run
../Pascal2021/dr13.run <../Pascal2021/dr13p.in >../Pascal2021/dr13p.out
```

```
## Free Pascal Compiler version 3.0.4+dfsg-23 [2019/11/25] for x86_64
## Copyright (c) 1993-2017 by Florian Klaempfl and others
## Target OS: Linux for x86-64
## Compiling ../Pascal2021/dr13.pas
## Linking ../Pascal2021/dr13
## /usr/bin/ld.bfd: warning: link.res contains output sections; did you forget -T?
## 326 lines compiled, 0.2 sec
```

```
Order of problem (n): Initial matrix of order 4
   1.00000    1.00000    1.00000    1.00000
   1.00000    2.00000    2.00000    2.00000
   1.00000    2.00000    3.00000    3.00000
   1.00000    2.00000    3.00000    4.00000
Calling evsvd
```

```
alg13.pas -- symmetric matrix eigensolutions via svd
Adding a shift of  3.0000000000000000E+000 to diagonal of matrix.
alg01.pas -- NashSVD
End of Sweep #1-  no. of rotations performed =6
End of Sweep #2-  no. of rotations performed =6
End of Sweep #3-  no. of rotations performed =6
End of Sweep #4-  no. of rotations performed =4
End of Sweep #5-  no. of rotations performed =0

Eigenvalue 1 =  8.2908593693815913E+000
 0.2280134  0.4285251  0.5773503  0.6565385
Residuals
-1.22E-015  2.22E-016 -1.55E-015 -2.22E-015
Sum of squared residuals = 8.8870111353804611E-030

Eigenvalue 2 =  1.0000000000000009E+000
-0.5773503 -0.5773503  0.0000000  0.5773503
Residuals
 6.66E-016  4.44E-016 -2.22E-016 -4.44E-016
Sum of squared residuals = 8.8746851837363828E-031

Eigenvalue 3 =  4.2602204776046149E-001
 0.6565385 -0.2280134 -0.5773503  0.4285251
Residuals
 2.78E-016 -1.11E-016  2.22E-016  6.66E-016
Sum of squared residuals = 5.8240121518270012E-031

Eigenvalue 4 =  2.8311858285794766E-001
-0.4285251  0.6565385 -0.5773503  0.2280134
Residuals
-3.89E-016  8.88E-016 -1.11E-016  5.55E-016
Sum of squared residuals = 1.2603285556070071E-030
Order of problem (n):
```

Algorithm 14 – Jacobi symmetric matrix eigensolutions

Fortran

Listing – Algorithm 14

```
      SUBROUTINE A14JE(N,A,NA,V,NV,ISWP,IPR,SETV,COMV)
C  ALGORITHM 14  JACOBI EIGENVALUES AND EIGENVECTORS OF REAL SYMMETRIC
C     MATRIX
C  J.C. NASH   JULY 1978, FEBRUARY 1980, APRIL 1989
C  N     = ORDER OF PROBLEM
C  A     = ARRAY CONTAINING MATRIX  -- MUST BE SYMMETRIC
C  NA    = FIRST DIMENSION OF A
C  V     = ARRAY INTO WHICH VECTORS COMPUTED
C  NV    = FIRST DIMENSION OF V
C  ISWP  =  SWEEP LIMIT (INPUT). SWEEP COUNT (OUTPUT)
C  IPR   = PRINT CHANNEL.  PRINTING ONLY IF IPR.GT.0
C  SETV  = LOGICAL SWITCH TO SET V INITIALLY TO IDENTITY OF ORDER N.
C  COMV  = LOGICAL SWITCH. IF .TRUE. THEN VECTORS TO BE CALCULATED.
C  STEP 0
      LOGICAL SETV,COMV
      INTEGER N,NA,NV,IPR,ISWP,LISWP,M,I,J,K,N1,I1
      REAL A(NA,N),V(NV,N),P,Q,T,S,C,FACT
C  FACTOR USED IN TEST AT STEP 7
      FACT=100.0
      N1=N-1
      LISWP=ISWP
      ISWP=0
C   EIGENVALUES LEFT IN DIAGONAL ELEMENTS OF A.
      IF(.NOT.COMV)GOTO 10
      IF(.NOT.SETV)GOTO 10
      DO 5 I=1,N
        DO 3 J=1,N
          V(I,J)=0.0
   3    CONTINUE
        V(I,I)=1.0
   5  CONTINUE
C  STEP 1
  10  ISWP=ISWP+1
      IF(ISWP.GT.LISWP)RETURN
      M=0
C  STEP 2
      IF(N.EQ.1)RETURN
      DO 160 I=1,N1
C  STEP 3
      I1=I+1
        DO 150 J=I1,N
C  STEP 4
          P=0.5*(A(I,J)+A(J,I))
          Q=A(I,I)-A(J,J)
          T=SQRT(4.0*P*P+Q*Q)
C  STEP 5
          IF(T.EQ.0.0)GOTO 110
C  STEP 6
```

```
          IF(Q.LT.0.0)GOTO 90
C  STEP 7
          IF(ABS(A(I,I)).LT.ABS(A(I,I))+FACT*ABS(P))GOTO 80
          IF(ABS(A(J,J)).EQ.ABS(A(J,J))+FACT*ABS(P))GOTO 110
C  STEP 8
  80      C=SQRT((T+Q)/(2.0*T))
          S=P/(T*C)
          GOTO 100
C  STEP 9
  90      S=SQRT((T-Q)/(2.0*T))
          IF(P.LT.0.0)S=-S
          C=P/(T*S)
C  STEP 10
 100      IF(1.0.LT.1.0+ABS(S))GOTO 120
C  STEP 11
 110      M=M+1
          GOTO 150
C  STEP 12
 120      DO 125 K=1,N
            Q=A(I,K)
            A(I,K)=C*Q+S*A(J,K)
            A(J,K)=-S*Q+C*A(J,K)
 125      CONTINUE
C  STEP 13
          DO 135 K=1,N
            Q=A(K,I)
            A(K,I)=C*Q+S*A(K,J)
            A(K,J)=-S*Q+C*A(K,J)
 135      CONTINUE
          IF(.NOT.COMV)GOTO 150
C  STEP 14
          DO 145 K=1,N
            Q=V(K,I)
            V(K,I)=C*Q+S*V(K,J)
            V(K,J)=-S*Q+C*V(K,J)
 145      CONTINUE
C  STEP 15
 150    CONTINUE
C  STEP 16
 160  CONTINUE
C  STEP 17
      IF(IPR.GT.0)WRITE(IPR,970)ISWP,M
 970  FORMAT( 9H AT SWEEP,I4,2X,I4,18H ROTATIONS SKIPPED)
      IF(M.LT.N*(N-1)/2)GOTO 10
      RETURN
      END
```

Example output

We use a Frank matrix of order 4 for our example.

```
## #!/bin/bash
gfortran ../fortran/dr14x.f
mv ./a.out ../fortran/dr14x.run
```

```
../fortran/dr14x.run < ../fortran/dr14xf.in
```

```
##  ORDER N=   4
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     1 ROTATIONS SKIPPED
##  AT SWEEP   4     6 ROTATIONS SKIPPED
##  MAX. ABS. RESIDUAL=  9.11166524E-07  POSN   4   1
##  MAX. ABS. INNER PRODUCT=  1.11389284E-07  POSN   2   4
##  EIGENVALUE  1=  8.29085827E+00
##    0.22801341E+00  0.42852503E+00  0.57735026E+00  0.65653843E+00
##  EIGENVALUE  2=  9.99999702E-01
##   -0.57735038E+00 -0.57735026E+00 -0.66010671E-07  0.57735032E+00
##  EIGENVALUE  3=  4.26021874E-01
##    0.65653849E+00 -0.22801340E+00 -0.57735032E+00  0.42852509E+00
##  EIGENVALUE  4=  2.83118576E-01
##   -0.42852506E+00  0.65653843E+00 -0.57735014E+00  0.22801338E+00
##  ORDER N=   0
```

BASIC

The two-sided Jacobi eigensolution routine (Algorithm 14) follows.

Listing

```
3000 PRINT "SMEV.JC ALG 14 DEC 7 78"
3005 REM DIM A(N,N),V(N,N)
3010 FOR I=1 TO N
3015   FOR J=1 TO N
3020     LET V(I,J)=0
3025   NEXT J
3030   LET V(I,I)=1
3035 NEXT I
3040 LET I9=0
3045 LET I9=I9+1
3050 IF I9>30 THEN GOTO 3240
3055 LET N8=0
3060 FOR I=1 TO N-1
3065   FOR J=I+1 TO N
3070     LET P=A(I,J)+A(J,I)
3075     LET Q=A(I,I)-A(J,J)
3080     LET T=SQRT(P*P+Q*Q)
3085     IF T=0 THEN GOTO 3145
3090     IF Q<0 THEN GOTO 3120
3095     IF ABS(A(I,I))<ABS(A(I,I))+50*ABS(P) THEN GOTO 3105
3100     IF ABS(A(J,J))=ABS(A(J,J))+50*ABS(P) THEN GOTO 3145
3105     LET C=SQRT((T+Q)/(2*T))
3110     LET S=.5*P/(T*C)
3115     GOTO 3140
3120     LET S=SQRT((T-Q)/(2*T))
3125     IF P>0 THEN GOTO 3135
3130     LET S=-S
3135     LET C=.5*P/(T*S)
3140     IF 1<1+ABS(S) THEN GOTO 3155
```

```
3145     LET N8=N8+1
3150     GOTO 3220
3155     FOR K=1 TO N
3160       LET Q=A(I,K)
3165       LET A(I,K)=C*Q+S*A(J,K)
3170       LET A(J,K)=-S*Q+C*A(J,K)
3175     NEXT K
3180     FOR K=1 TO N
3185       LET Q=A(K,I)
3190       LET A(K,I)=C*Q+S*A(K,J)
3195       LET A(K,J)=-S*Q+C*A(K,J)
3200       LET Q=V(K,I)
3205       LET V(K,I)=C*Q+S*V(K,J)
3210       LET V(K,J)=-S*Q+C*V(K,J)
3215     NEXT K
3220   NEXT J
3225 NEXT I
3230 PRINT N8," SMALL ROTNS -- SWEEP",I9
3235 IF N8<N*(N-1)/2 THEN GOTO 3045
3240 PRINT "CONVERGED"
3245 FOR I=1 TO N
3250   LET D(I)=A(I,I)
3255 NEXT I
3260 RETURN
```

Example output

The following example substitutes the BASIC code for Algorithm 14 for that of Algorithm 13 in the BASIC example above that uses the Frank matrix.

?? include Rayleigh quotient. ?? residuals??

```
bwbasic ../BASIC/dr14.bas  >../BASIC/dr14b.out
# echo "done"
```

```
Bywater BASIC Interpreter/Shell, version 2.20 patch level 2

Copyright (c) 1993, Ted A. Campbell

Copyright (c) 1995-1997, Jon B. Volkoff


A14 JACOBI EIGENSOLUTIONS OF A SYMMETRIC MATRIX VIA SVD
ORDER OF MATRIX:  3
A and B:
 1  1  1
 1  2  2
 1  2  3
 0
SMEV.JC ALG 14 DEC 7 78
 0             SMALL ROTNS -- SWEEP        1
 0             SMALL ROTNS -- SWEEP        2
 0             SMALL ROTNS -- SWEEP        3
 3             SMALL ROTNS -- SWEEP        4
CONVERGED
```

```
D, V:
 5.0489173: 0.3279853 -0.7369762 0.591009
 0.6431041: 0.591009 -0.3279853 -0.7369762
 0.3079785: 0.7369762 0.591009 0.3279853
compute residual and RQ for soln          1
 1 1 1.6559706
 1 2 2.9839558
 1 3 3.720932
EV( 1)= 5.0489173    RQ= 5.0489173
 0.3279853  0.591009  0.7369762  MAX ABS RESIDUAL= 3.720932
compute residual and RQ for soln          2
 2 1 -0.4739525
 2 2 -0.2109287
 2 3 0.3800804
EV( 2)= 0.6431041    RQ= 0.6431041
 -0.7369762  -0.3279853  0.591009  MAX ABS RESIDUAL= 3.720932
compute residual and RQ for soln          3
 3 1 0.182018
 3 2 -0.2269729
 3 3 0.1010124
EV( 3)= 0.3079785    RQ= 0.3079785
 0.591009  -0.7369762  0.3279853  MAX ABS RESIDUAL= 3.720932

ORDER OF MATRIX:  0
```

Pascal

Listing – Algorithm 14

```
Procedure evJacobi(n: integer;
               var A,V : rmatrix;
                   initev: boolean);
var
  count, i, j, k, limit, skipped : integer;
  c, p, q, s, t : real;
  oki, okj, rotn : boolean;

begin
  writeln('alg14.pas -- eigensolutions of a real symmetric');
  writeln('               matrix via a Jacobi method');
  if initev then
  begin
    for i := 1 to n do
    begin
      for j := 1 to n do V[i,j] := 0.0;
      V[i,i] := 1.0;
    end;
  end;
  count := 0;
  limit := 30;
  skipped := 0;

  while (count<=limit) and (skipped<((n*(n-1)) div 2) ) do
```

```
begin
  count := count+1;
  write('sweep ',count,' ');
  skipped := 0;
  for i := 1 to (n-1) do
  begin
    for j := (i+1) to n do
    begin
      rotn := true;
      p := 0.5*(A[i,j]+A[j,i]);
      q := A[i,i]-A[j,j];
      t := sqrt(4.0*p*p+q*q);
      if t=0.0 then
      begin
        rotn := false;
      end
      else
      begin
        if q>=0.0 then
        begin

          oki := (abs(A[i,i])=abs(A[i,i])+100.0*abs(p));
          okj := (abs(A[j,j])=abs(A[j,j])+100.0*abs(p));
          if oki and okj then rotn := false
                         else rotn := true;

          if rotn then
          begin
             c := sqrt((t+q)/(2.0*t)); s := p/(t*c);
          end;
        end
        else
        begin
          rotn := true;
          s := sqrt((t-q)/(2.0*t));
          if p<0.0 then s := -s;
          c := p/(t*s);
        end;
        if 1.0=(1.0+abs(s)) then rotn := false;
      end;
      if rotn then
      begin
        for k := 1 to n do
        begin
          q := A[i,k]; A[i,k] := c*q+s*A[j,k]; A[j,k] := -s*q+c*A[j,k];
        end;

        for k := 1 to n do
        begin
          q := A[k,i]; A[k,i] := c*q+s*A[k,j]; A[k,j] := -s*q+c*A[k,j];

          q := V[k,i]; V[k,i] := c*q+s*V[k,j]; V[k,j] := -s*q+c*V[k,j];
        end;
```

```
        end
        else

           skipped := skipped+1;
      end;
    end;
    writeln('  ',skipped,' / ',n*(n-1) div 2,'  rotations skipped');
  end;
end;
```

Example output

```
fpc ../Pascal2021/dr14x.pas
# copy to run file
mv ../Pascal2021/dr14x ../Pascal2021/dr14x.run
../Pascal2021/dr14x.run <../Pascal2021/dr14xp.in >../Pascal2021/dr14xp.out
```

```
## Free Pascal Compiler version 3.0.4+dfsg-23 [2019/11/25] for x86_64
## Copyright (c) 1993-2017 by Florian Klaempfl and others
## Target OS: Linux for x86-64
## Compiling ../Pascal2021/dr14x.pas
## Linking ../Pascal2021/dr14x
## /usr/bin/ld.bfd: warning: link.res contains output sections; did you forget -T?
## 475 lines compiled, 0.2 sec
```

```
Order of problem (n) = 5
Matrixin.pas -- generate or input a real matrix 5 by 5
Enter type to generate 3
Type: 3) Moler
   1.00000   -1.00000   -1.00000   -1.00000   -1.00000
  -1.00000    2.00000    0.00000    0.00000    0.00000
  -1.00000    0.00000    3.00000    1.00000    1.00000
  -1.00000    0.00000    1.00000    4.00000    2.00000
  -1.00000    0.00000    1.00000    2.00000    5.00000
alg14.pas -- eigensolutions of a real symmetric
             matrix via a Jacobi method
sweep 1   0 / 10  rotations skipped
sweep 2   0 / 10  rotations skipped
sweep 3   0 / 10  rotations skipped
sweep 4   0 / 10  rotations skipped
sweep 5   7 / 10  rotations skipped
sweep 6   10 / 10  rotations skipped
Eigenvalue 1 =  7.4874999307049812E+000
 0.2556536 -0.0465883 -0.3403404 -0.5720713 -0.6995524
Residuals
 2.00E-015 -2.08E-016 -1.11E-015 -3.77E-015 -4.44E-016
Sum of squared residuals = 1.9715552247697695E-029

Eigenvalue 2 =  2.8289347824203293E+000
-0.3290435  0.3969474  0.6440739  0.1142105 -0.5534327
Residuals
-8.88E-016  1.33E-015  1.11E-015  8.88E-016 -4.44E-016
Sum of squared residuals = 4.7824692379023841E-030
```

```
Eigenvalue 3 =  2.3964685319861365E+000
-0.2547217  0.6424765 -0.0808234 -0.6002650  0.3943227
Residuals
-4.44E-016  6.66E-016  1.11E-016 -1.11E-015  0.00E+000
Sum of squared residuals = 1.8858706015439813E-030

Eigenvalue 4 =  2.2784504826045859E+000
-0.1364558  0.4900540 -0.6433686  0.5337402 -0.2059737
Residuals
-6.11E-016  1.11E-016  2.78E-016  9.99E-016 -2.22E-016
Sum of squared residuals = 1.5099290763995929E-030

Eigenvalue 5 =  8.6462722839592467E-003
-0.8618981 -0.4328202 -0.2210920 -0.1203898 -0.0801439
Residuals
-2.36E-016  2.22E-016  4.16E-017  2.50E-016  2.22E-016
Sum of squared residuals = 2.1840045569351255E-031

Order of problem (n) = 0
```

Algorithm 15 - Generalized symmetric eigenproblem

We aim to solve the generalized symmetric eigenproblem

$$Ax = eBx$$

for x and e, where symmetric matrices A and B and B is positive definite.

Fortran

Listing – Algorithm 15

```
      SUBROUTINE A15GSE(N,A,NA,B,NB,V,NV,FAIL,ISWP,IPR)
C  ALGORITHM 15 GENERALISED SYMMETRIC EIGENPROBLEM BY 2 APPLICATIONS
C    OF JACOBI ALGORITHM 14
C  J.C. NASH   JULY 1978, FEBRUARY 1980, APRIL 1989
C  N     =   ORDER OF PROBLEM
C  A     =   A  MATRIX OF EIGENPROBLEM. DIAGONAL ELEMENTS BECOME
C  NA    =   FIRST DIMENSION OF A
C  B     =   B  MATRIX OF EIGENPROBLEM, MUST BE POSITIVE DEFINITE
C  NB    =   FIRST DIMENSION OF B
C  V     =   VECTOR MATRIX. ON OUTPUT COLUMNS ARE EIGENVECTORS
C            EIGENVALUES
C  FAIL  =   LOGICAL FLAG SET .TRUE. IF B NOT COMPUTATIONALLY
C              POSITIVE DEFINITE OR IF EITHER APPLICATION OF
C              ALGORITHM 14 TAKES MORE THAN ISWP SWEEPS
C  NV    =   FIRST DIMENSION OF V
C  ISWP  =   BOUND ON SWEEPS IN ALG. 14.
C  NA,NB,NV ALL .GE. N
C  IPR   =   PRINT CHANNEL.  IPR.GT.0 FOR PRINTING
C  STEP 0
      LOGICAL FAIL,COMV,SETV
```

```
      INTEGER N,NA,NB,NV,STAGE,ISWP,LISWP,I,J,K,M,IPR
      REAL A(NA,N),B(NB,N),V(NV,N),S
      FAIL=.FALSE.
      STAGE=1
      SETV=.TRUE.
      COMV=.TRUE.
C  STEP 1  INTERCHANGE - NOT GENERALLY EFFICIENT
  10  DO 16 I=1,N
        DO 14 J=1,N
          S=A(I,J)
          A(I,J)=B(I,J)
          B(I,J)=S
  14    CONTINUE
  16  CONTINUE
C  STEP 2
      LISWP=ISWP
      IF(IPR.GT.0)WRITE(IPR,964)STAGE
 964  FORMAT( 6H STAGE,I3)
      CALL A14JE(N,A,NA,V,NV,LISWP,IPR,SETV,COMV)
      IF(LISWP.GE.ISWP)FAIL=.TRUE.
      IF(FAIL)RETURN
C  STEP 3
      IF(STAGE.EQ.2)GOTO 80
C  STEP 4
      STAGE=2
      SETV=.FALSE.
      DO 46 I=1,N
        IF(A(I,I).LE.0.0)FAIL=.TRUE.
        IF(FAIL)RETURN
        S=1.0/SQRT(A(I,I))
        DO 44 J=1,N
          V(J,I)=S*V(J,I)
  44    CONTINUE
  46  CONTINUE
C  STEP 5
      DO 56 I=1,N
        DO 54 J=1,N
          A(I,J)=B(I,J)
  54    CONTINUE
  56  CONTINUE
C  STEP 6
      DO 68 I=1,N
        DO 66 J=I,N
          S=0.0
          DO 64 K=1,N
            DO 62 M=1,N
              S=S+V(K,I)*A(K,M)*V(M,J)
  62        CONTINUE
  64      CONTINUE
          B(I,J)=S
          B(J,I)=S
  66    CONTINUE
  68  CONTINUE
```

```
C  STEP 7
      GOTO 10
  80  ISWP=0
      RETURN
      END
```

Example output

This example uses the Frank matrix as B and a Unit matrix as matrix A. Essentially we get the eigensolutions of the inverse of the Frank matrix.

```
## #!/bin/bash
gfortran ../fortran/dr1415.f
mv ./a.out ../fortran/dr1415.run
../fortran/dr1415.run < ../fortran/dr1415f.in
```

```
##  ORDER N=   5
##  A = UNIT MATRIX
##  B = FRANK MATRIX
##  STAGE  1
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     0 ROTATIONS SKIPPED
##  AT SWEEP   4     9 ROTATIONS SKIPPED
##  AT SWEEP   5    10 ROTATIONS SKIPPED
##  STAGE  2
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     9 ROTATIONS SKIPPED
##  AT SWEEP   3    10 ROTATIONS SKIPPED
##  MAX. ABS. RESIDUAL=  2.65677636E-06  POSN   4   1
##  MAX. ABS. INNER PRODUCT=  1.49453115E-07  POSN   2   3
##  EIGENVALUE  1=  3.68250632E+00
##   -0.62562543E+00  0.10526186E+01 -0.11454134E+01  0.87454730E+00 -0.32601866E+00
##  EIGENVALUE  2=  2.83082914E+00
##    0.92290354E+00 -0.76677585E+00 -0.28584322E+00  0.10042626E+01 -0.54852855E+00
##  EIGENVALUE  3=  1.71537042E+00
##    0.78175271E+00  0.22251040E+00 -0.71841979E+00 -0.42699385E+00  0.59688485E+00
##  EIGENVALUE  4=  6.90278590E-01
##   -0.37863755E+00 -0.49590981E+00 -0.27086613E+00  0.14115055E+00  0.45573431E+00
##  EIGENVALUE  5=  8.10140595E-02
##    0.48356060E-01  0.92794605E-01  0.12971547E+00  0.15612756E+00  0.16989112E+00
##  ORDER N=  10
##  A = UNIT MATRIX
##  B = FRANK MATRIX
##  STAGE  1
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2     0 ROTATIONS SKIPPED
##  AT SWEEP   3     0 ROTATIONS SKIPPED
##  AT SWEEP   4    23 ROTATIONS SKIPPED
##  AT SWEEP   5    45 ROTATIONS SKIPPED
##  STAGE  2
##  AT SWEEP   1     0 ROTATIONS SKIPPED
##  AT SWEEP   2    35 ROTATIONS SKIPPED
##  AT SWEEP   3    45 ROTATIONS SKIPPED
##  MAX. ABS. RESIDUAL=  7.95809774E-06  POSN  10   1
```

```
## MAX. ABS. INNER PRODUCT=  1.70029864E-07  POSN   1   3
## EIGENVALUE  1=  3.91114664E+00
##    0.25440717E+00 -0.48620966E+00  0.67481190E+00 -0.80345494E+00  0.86070871E+00
##   -0.84148449E+00  0.74749005E+00 -0.58707643E+00  0.37449750E+00 -0.12864262E+00
## EIGENVALUE  2=  3.65247846E+00
##   -0.46986169E+00  0.77643681E+00 -0.81318295E+00  0.56733066E+00 -0.12432010E+00
##   -0.36189401E+00  0.72234160E+00 -0.83175814E+00  0.65211862E+00 -0.24585268E+00
## EIGENVALUE  3=  3.24698114E+00
##   -0.61485553E+00  0.76671326E+00 -0.34122097E+00 -0.34121776E+00  0.76671290E+00
##   -0.61485648E+00  0.70596684E-09  0.61485595E+00 -0.76671231E+00  0.34121889E+00
## EIGENVALUE  4=  2.73068285E+00
##   -0.67134637E+00  0.49054128E+00  0.31291714E+00 -0.71918464E+00  0.21257788E+00
##    0.56385678E+00 -0.62457776E+00 -0.10748890E+00  0.70311815E+00 -0.40626636E+00
## EIGENVALUE  5=  2.14946055E+00
##    0.63807106E+00 -0.95366970E-01 -0.62381732E+00  0.18860267E+00  0.59562886E+00
##   -0.27762464E+00 -0.55413532E+00  0.36044526E+00  0.50026351E+00 -0.43521550E+00
## EIGENVALUE  6=  1.55495822E+00
##   -0.53058165E+00 -0.23613074E+00  0.42549348E+00  0.42549339E+00 -0.23613124E+00
##   -0.53058165E+00  0.14518602E-06  0.53058153E+00  0.23613124E+00 -0.42549354E+00
## EIGENVALUE  7=  1.00000012E+00
##   -0.37796488E+00 -0.37796432E+00  0.30240781E-06  0.37796441E+00  0.37796423E+00
##   -0.16945556E-08 -0.37796435E+00 -0.37796468E+00  0.11743472E-06  0.37796459E+00
## EIGENVALUE  8=  5.33896327E-01
##   -0.21690433E+00 -0.31800413E+00 -0.24932273E+00 -0.47528878E-01  0.17964047E+00
##    0.31090042E+00  0.27617189E+00  0.93996122E-01 -0.13836364E+00 -0.29685175E+00
## EIGENVALUE  9=  1.98062271E-01
##    0.84274180E-01  0.15185684E+00  0.18936239E+00  0.18936241E+00  0.15185687E+00
##    0.84274203E-01 -0.56293572E-07 -0.84274210E-01 -0.15185684E+00 -0.18936238E+00
## EIGENVALUE 10=  2.23383494E-02
##    0.97219953E-02  0.19226812E-01  0.28302142E-01  0.36745246E-01  0.44367522E-01
##    0.50998691E-01  0.56490630E-01  0.60720690E-01  0.63594334E-01  0.65047383E-01
## ORDER N=   4
## A = UNIT MATRIX
## B = FRANK MATRIX
## STAGE  1
## AT SWEEP   1     0 ROTATIONS SKIPPED
## AT SWEEP   2     0 ROTATIONS SKIPPED
## AT SWEEP   3     1 ROTATIONS SKIPPED
## AT SWEEP   4     6 ROTATIONS SKIPPED
## STAGE  2
## AT SWEEP   1     0 ROTATIONS SKIPPED
## AT SWEEP   2     5 ROTATIONS SKIPPED
## AT SWEEP   3     6 ROTATIONS SKIPPED
## MAX. ABS. RESIDUAL=  9.88243642E-07  POSN   3   1
## MAX. ABS. INNER PRODUCT=  5.05645765E-08  POSN   1   3
## EIGENVALUE  1=  3.53208756E+00
##   -0.80536371E+00  0.12338886E+01 -0.10850633E+01  0.42852503E+00
## EIGENVALUE  2=  2.34729719E+00
##   -0.10058755E+01  0.34933683E+00  0.88455212E+00 -0.65653861E+00
## EIGENVALUE  3=  1.00000060E+00
##   -0.57735038E+00 -0.57735050E+00 -0.25529275E-07  0.57735038E+00
## EIGENVALUE  4=  1.20614767E-01
##   -0.79188228E-01 -0.14882518E+00 -0.20051166E+00 -0.22801344E+00
## ORDER N=   0
```

```
## Note: The following floating-point exceptions are signalling: IEEE_UNDERFLOW_FLAG IEEE_DENORMAL
```

Pascal

Listing

```
procedure genevJac( n : integer;
                var A, B, V : rmatrix);

var
 i,j,k,m : integer;
 s : real;
 initev : boolean;

begin
  writeln('alg15.pas -- generalized symmetric matrix eigenproblem');
  initev := true;
  writeln('Eigensolutions of metric B');
  evJacobi(n, B, V, initev);



  for i := 1 to n do
  begin
    if B[i,i]<=0.0 then halt;
    s := 1.0/sqrt(B[i,i]);
    for j := 1 to n do V[j,i] := s * V[j,i];
  end;


  for i := 1 to n do
  begin
    for j := i to n do
    begin
      s := 0.0;
      for k := 1 to n do
        for m := 1 to n do
          s := s+V[k,i]*A[k,m]*V[m,j];
      B[i,j] := s; B[j,i] := s;
    end;
  end;

  initev := false;
  writeln('Eigensolutions of general problem');
  evJacobi( n, B, V, initev);
end;
```

Example output

```
fpc ../Pascal2021/dr15x.pas
# copy to run file
mv ../Pascal2021/dr15x ../Pascal2021/dr15x.run
../Pascal2021/dr15x.run <../Pascal2021/dr15xp.in >../Pascal2021/dr15xp.out
```

```
## Free Pascal Compiler version 3.0.4+dfsg-23 [2019/11/25] for x86_64
## Copyright (c) 1993-2017 by Florian Klaempfl and others
## Target OS: Linux for x86-64
## Compiling ../Pascal2021/dr15x.pas
## dr15x.pas(369,3) Note: Local variable "ch" not used
## dr15x.pas(490,7) Note: Local variable "k" not used
## dr15x.pas(491,1) Note: Local variable "s" not used
## dr15x.pas(491,4) Note: Local variable "s2" is assigned but never used
## dr15x.pas(493,4) Note: Local variable "Y" not used
## dr15x.pas(495,1) Note: Local variable "ch" not used
## Linking ../Pascal2021/dr15x
## /usr/bin/ld.bfd: warning: link.res contains output sections; did you forget -T?
## 550 lines compiled, 0.2 sec
## 6 note(s) issued
```

```
Order of matrices = 5
Provide matrix A
Matrixin.pas -- generate or input a real matrix 5 by 5
Enter type to generate 10
Type: 10) Unit
   1.00000    0.00000    0.00000    0.00000    0.00000
   0.00000    1.00000    0.00000    0.00000    0.00000
   0.00000    0.00000    1.00000    0.00000    0.00000
   0.00000    0.00000    0.00000    1.00000    0.00000
   0.00000    0.00000    0.00000    0.00000    1.00000
Provide matrix B
Matrixin.pas -- generate or input a real matrix 5 by 5
Enter type to generate 4
Type: 4) Frank symmetric
   1.00000    1.00000    1.00000    1.00000    1.00000
   1.00000    2.00000    2.00000    2.00000    2.00000
   1.00000    2.00000    3.00000    3.00000    3.00000
   1.00000    2.00000    3.00000    4.00000    4.00000
   1.00000    2.00000    3.00000    4.00000    5.00000
alg15.pas -- generalized symmetric matrix eigenproblem
Eigensolutions of metric B
alg14.pas -- eigensolutions of a real symmetric
             matrix via a Jacobi method
sweep 1   0 / 10  rotations skipped
sweep 2   0 / 10  rotations skipped
sweep 3   0 / 10  rotations skipped
sweep 4   2 / 10  rotations skipped
sweep 5   10 / 10  rotations skipped
Eigensolutions of general problem
alg14.pas -- eigensolutions of a real symmetric
             matrix via a Jacobi method
sweep 1   0 / 10  rotations skipped
sweep 2   10 / 10  rotations skipped

Eigenvalue 1 =  3.68250706566235 93E+000
-0.6256253  1.0526189 -1.1454135  0.8745474 -0.3260187
Generalized matrix eigensolution residuals
-4.44E-016  1.78E-015 -2.22E-015 -1.78E-015  0.00E+000
Sum of squared residuals = 1.1438483125704671E-029
```

```
Rayleigh quotient =  3.6825070656623629E+000
Generalized matrix eigensolution residuals
-2.22E-016 -4.44E-016 -1.33E-015 -1.78E-015  8.88E-016
Sum of squared residuals = 5.9657605957339018E-030

Eigenvalue 2 =  2.8308300260037713E+000
 0.9229035 -0.7667759 -0.2858430  1.0042629 -0.5485287
Generalized matrix eigensolution residuals
-8.88E-016 -3.55E-015 -2.66E-015 -8.88E-016 -2.66E-015
Sum of squared residuals = 2.8398992587956425E-029
Rayleigh quotient =  2.8308300260037744E+000
Generalized matrix eigensolution residuals
-1.55E-015 -2.66E-015 -1.78E-015 -3.55E-015 -1.78E-015
Sum of squared residuals = 2.8448296394532738E-029

Eigenvalue 3 =  1.7153703234534294E+000
-0.7817528 -0.2225101  0.7184199  0.4269937 -0.5968848
Generalized matrix eigensolution residuals
-4.44E-016 -4.44E-016  0.00E+000  0.00E+000 -8.88E-016
Sum of squared residuals = 1.1832913578315177E-030
Rayleigh quotient =  1.7153703234534299E+000
Generalized matrix eigensolution residuals
-2.22E-016 -4.44E-016  0.00E+000 -8.88E-016  0.00E+000
Sum of squared residuals = 1.0353799381025780E-030

Eigenvalue 4 =  6.9027853210942958E-001
 0.3786376  0.4959098  0.2708661 -0.1411506 -0.4557341
Generalized matrix eigensolution residuals
-1.11E-016 -2.22E-016 -4.44E-016 -1.11E-015 -8.88E-016
Sum of squared residuals = 2.2803010541544873E-030
Rayleigh quotient =  6.9027853210942991E-001
Generalized matrix eigensolution residuals
-2.78E-016 -4.44E-016 -6.66E-016 -1.11E-015 -6.66E-016
Sum of squared residuals = 2.3943161068622116E-030

Eigenvalue 5 =  8.1014052771005290E-002
 0.0483561  0.0927946  0.1297155  0.1561276  0.1698911
Generalized matrix eigensolution residuals
-5.03E-017 -1.56E-016 -1.11E-016 -1.18E-016 -1.25E-016
Sum of squared residuals = 6.8746671218010005E-032
Rayleigh quotient =  8.1014052771005221E-002
Generalized matrix eigensolution residuals
-1.04E-017 -7.63E-017 -6.94E-018  2.08E-017  2.78E-017
Sum of squared residuals = 7.1861261049948738E-033
```

Cleanup of working files

The following script is included to remove files created during compilation or execution of the examples.

```
## remove object and run files
cd ../fortran/
echo `pwd`
rm  *.o
```

```
rm  *.run
rm *.out
cd ../Pascal2021/
echo `pwd`
rm  *.o
rm  *.run
rm *.out
cd ../BASIC
echo `pwd`
rm *.out
cd ../Documentation
## ?? others
```

```
## /versioned/Nash-Compact-Numerical-Methods/fortran
## rm: cannot remove '*.o': No such file or directory
## /versioned/Nash-Compact-Numerical-Methods/Pascal2021
## /versioned/Nash-Compact-Numerical-Methods/BASIC
```

References

Nash, John C. 1979. *Compact Numerical Methods for Computers : Linear Algebra and Function Minimisation.* Book. Hilger: Bristol.