# DECUS
## PROGRAM LIBRARY

| | |
|---|---|
| DECUS NO. | 8-317 |
| TITLE | EIG (Compute Eigenvalues and Eigenvectors) |
| AUTHOR | J. N. R. Jeffers |
| COMPANY | Submitted by: A. J. P. Gore<br>The Nature Conservancy<br>Lancashire, England |
| DATE | September 25, 1969 |
| SOURCE LANGUAGE | FORTRAN D |

# EIG (Compute Eigenvalues and Eigenvectors)

DECUS Program Library Write-up

DECUS No. 8-317

## ABSTRACT

The extraction of eigenvalues and eigenvectors is carried out by means of a group of programs, the intermediate stages of the computations being stored on the disk. The eigenvalues, and the associated eigenvectors, are extracted one by one, starting with the largest eigenvalue, and the process can be terminated as soon as sufficient eigenvalues have been extracted from the basic data matrix. The programs were designed to enable principal component and canonical variate analyses to be carried out.

## TAPES REQUIRED

1. Form of program tapes - All program tapes are written in the PDP-8 FORTRAN-D language, and are in the source language. There are four programs in the set, as follows:

(a) SMOD - program to store data matrix on disk.

(b) EIG1 - program to compute the highest latent root and vector.

(c) EIG2 - program to extract and print the highest latent root and vector.

(d) AITK - program to square the current data matrix, so as to increase the speed of convergence in the process of extracting the latent root.

2. Form of data tape - The data tape should consist of the matrix of correlation coefficients or of corrected sums of squares and products. The program requires the complete matrix, including the principal diagonal.

## OPERATING INSTRUCTIONS

Operation of this suite of programs is simplified if all of the programs are compiled onto the disk before the data are read, i. e.

```
.FORT
*OUT-S:SMOD
*
*IN-R:
* ↑
*.FORT
*OUT-S:EIG1
*
*IN-R:
* ↑
```

```
*.FORT
*OUT-S:EIG2
*
*IN-R:
* ↑
*.FORT
*OUT-S:AITK
*
*IN-R:
* ↑
*
```

The data tape should now be placed in the high-speed reader, and read onto the disk as follows:

```
.FOSL
*IN-S:SMOD
*
*OPT-S:
*OUT-S:DATA
*IN-
* ↑
*READY
↑
```

The program will pause for the entry of the number of variables, i. e. the dimension of the matrix, and this should be entered on the teletype and terminated by "Return." The program will type ! on completion.

The first latent root and vector can now be computed, as follows:

```
.FOSL
*IN-S:EIG1
*
*OPT-S:
*OUT-
*IN-S:DATA
* ↑
*READY
↑
```

The program will pause for the entry of the dimension of the matrix, and this should be entered on the teletype and terminated by "Return." The high-speed punch should also be switched on as soon as this has been done. The program will output a vector on the high-speed punch and then type ! on completion. The speed of convergence of the computation depends partly on the dimension of the matrix and partly on the closeness of the latent roots. If the calculation does not end after five minutes or so, it is advisable to restart the computation after squaring the matrix (see operating instructions for the AITK program).

The latent root and vector can be printed, and extracted from the data matrix, as follows:

```
.FOSL
*IN-S:EIG2
*
*OPT-S:
*OUT-S:DATA
*IN-S:DATA
* ↑                    Output from EIG1 in high-speed
*READY                          reader
↑
```

The program will pause for the entry of the dimension of the matrix, and this should be entered on the teletype and terminated by "Return." The program will print the eigenvalue and the eigenvector, and then store the residual data matrix on the disk, typing ! on completion. If further latent roots and vectors are required, return to program EIG1 above.

Convergence of the iterative procedure used in calculating the latent root and vector may, in certain cases, be slow. In these cases, the data matrix can be modified to improve the convergence as follows:

```
.FOSL
*IN-S:AITK
*
*OPT-S:
*OUT-S:DATA
*IN-S:DATA
* ↑
*READY
↑
```

The program will pause for the entry of the dimension of the matrix, and this should be entered on the teletype, and terminated by "Return." This procedure can be repeated three or four times. In general, the speed of convergence will be increased by a factor of 2 for every time the procedure is used, but it is not advisable to apply the procedure too often because of possible loss of accuracy.

In difficult cases, convergence may not be reached even by applying the Aitken process. In such cases, the condition for convergence may be relaxed by altering the FORTRAN statement following the labelled statement 9 to:

```
IF (TEST-0.001) 6,6,7
```

OUTPUT

The only printed output is obtained from the EIG2 program, and consists of the eigenvalue and the associated eigenvector.

STORAGE

Normal for FORTRAN-D
For the 4K version, the dimension of the matrix is limited to 12.

METHOD

See "A Short Course In Multivariate Analysis," M. G. Kendall, Griffin, 1956.

SMOD

```
L
C           PROGRAM TO STORE MATRIX ON DISK
            DEFINE DISK
            DIMENSION R(144)
            ACCEPT 1ØØ, N
            N=N*N
            DO 1 I=1, N
            READ 2, 1Ø1, R(I)
1           CONTINUE
            DO 2 I=1, N
            WRITE 3, 1Ø1, R(I)
2           CONTINUE
1ØØ         FORMAT (I)
1Ø1         FORMAT (/, E)
            END

*
```

```
L
C       PROGRAM TO COMPUTE LATENT ROOT AND VECTOR
        DEFINE DISK
        DIMENSION R(144),A(12),B(12)
        TYPE 1ØØ
1ØØ     FORMAT (/,"ENTER NO OF VARIABLES",/,/)
        ACCEPT 1Ø1,N
1Ø1     FORMAT (I)
        NN=N*N
        DO  1 I=1,NN
        READ 3,1Ø2,R(I)
1       CONTINUE
1Ø2     FORMAT (/,E)
        DO 2 I=1,N
        A(I)=Ø.Ø
        DO 2 J=1,N
        K=I+N*(J-1)
        A(I)=A(I)+R(K)
2       CONTINUE
        V=A(1)
        DO 3 I=1,N
        A(I)=A(I)/V
3       CONTINUE
7       TEST=Ø.Ø
        K=1
        DO 4 I=1,N
        B(I)=Ø.Ø
4       CONTINUE
        DO 5 I=1,N
        DO 5 J=1,N
        B(J)=B(J) +R(K)*A(I)
        K=K+1
5       CONTINUE
        V=B(1)
        DO 9 I=1,N
        B(I)=B(I)/V
        TEST=TEST+ABSF(A(I)-B(I) )
        A(I)=B(I)
9       CONTINUE
        IF (TEST-Ø.ØØØ1)6,6,7
6       DO 8 I=1,N
        WRITE 2,1Ø2,A(I)
8       CONTINUE
        END

*
```

```
L
C       PROGRAM TO PRINT LATENT ROOT AND VECTOR
        DEFINE DISK
        DIMENSION R(144),A(12)
        ACCEPT 1ØØ,N
1ØØ     FORMAT (I)
        DO 1 I=1,N
        READ 2,1Ø1,A(I)
1       CONTINUE
1Ø1     FORMAT (/,E)
        NN=N*N
        DO 2 I=1,NN
        READ 3, 1Ø1,R(I)
2       CONTINUE
        ROOT=Ø.Ø
        DO 3 I=1,N
        ROOT=ROOT+A(I)*A(I)
3       CONTINUE
        ROOT=1.Ø /SQTF(ROOT)
        EIG=Ø.Ø
        DO 4 I=1,N
        EIG=EIG+A(I)*R(I)
4       CONTINUE
        TYPE 1Ø1,EIG
        DO 5 I=1,N
        A(I)=ROOT*A(I)
        TYPE 1Ø1,A(I)
5       CONTINUE
        DO 6 I=1,N
        DO 6 J=1,N
        K=I+N*(J-1)
        R(K)=R(K)-A(I)*A(J)*EIG
6       CONTINUE
        DO 7 I=1,NN
        WRITE 3, 1Ø1,R(I)
7       CONTINUE
        END

*
```

```
L?
*L
C       PROGRAM TO SQUARE MATRIX
        DEFINE DISK
        DIMENSION R(144)
        ACCEPT 1ØØ, N
1ØØ     FORMAT (I)
        NN=N*N
        DO 1 I=1, NN
        READ 3, 1Ø1, R(I)
1       CONTINUE
        IR=Ø
        IK=-N
        DO 1Ø K=1, N
        IK=IK+N
        DO 1Ø  J=1, N
        IR=IR+1
        JI=J-N
        IB=IK
        RNEW=Ø.Ø
        DO 2Ø I=1, N
        JI=JI+N
        IB=IB+1
        RNEW=RNEW+R(JI)*R(IB)
2Ø      CONTINUE
        WRITE 3, 1Ø1, RNEW
1Ø      CONTINUE
1Ø1     FORMAT (E)
        END

*
```