NAG Fortran Library Routine Document

F07GFF (DPPEQU)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F07GFF (DPPEQU) computes a diagonal scaling matrix S intended to equilibrate a real n by n symmetric positive-definite matrix A, stored in packed format, and reduce its condition number.

2 Specification

```
SUBROUTINE F07GFF (UPLO, N, AP, S, SCOND, AMAX, INFO)

INTEGER             N, INFO
double precision    AP(*), S(*), SCOND, AMAX
CHARACTER*1         UPLO
```

The routine may be called by its LAPACK name ***dppequ***.

3 Description

F07GFF (DPPEQU) computes a diagonal scaling matrix S chosen so that

$$s_j = 1/\sqrt{a_{jj}}.$$

This means that the matrix B given by

$$B = SAS,$$

has diagonal elements equal to unity. This in turn means that the condition number of B, $\kappa_2(B)$, is within a factor n of the matrix of smallest possible condition number over all possible choices of diagonal scalings (see Corollary 7.6 of Higham (2002)).

4 References

Higham N J (2002) *Accuracy and Stability of Numerical Algorithms* (2nd Edition) SIAM, Philadelphia

5 Parameters

1: UPLO – CHARACTER*1 *Input*

On entry: indicates whether the upper or lower triangular part of A is stored in the array AP, as follows:

UPLO = 'U'

The upper triangle of A is stored.

UPLO = 'L'

The lower triangle of A is stored.

Constraint: UPLO = 'U' or 'L'.

2: N – INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: N $\geq$ 0.

3: AP($*$) – ***double precision*** array *Input*

Note: the dimension of the array AP must be at least $\max(1, \mathrm{N} \times \mathrm{N}/2)$.

On entry: the upper or lower triangle of the symmetric matrix A whose scaling factors are to be computed, packed column-wise in a linear array. The jth column of A is stored in the array AP as follows:

if UPLO = 'U', $\mathrm{AP}(i + (j-1)j/2) = a_{ij}$ for $1 \leq i \leq j$;
if UPLO = 'L', $\mathrm{AP}(i + (j-1)(2n-j)/2) = a_{ij}$ for $j \leq i \leq n$.

Only the elements of AP corresponding to the diagonal elements A are referenced.

4: S($*$) – ***double precision*** array *Output*

Note: the dimension of the array S must be at least $\max(1, \mathrm{N})$.

On exit: if INFO $= 0$ on exit, S contains the diagonal elements of the scaling matrix S.

5: SCOND – ***double precision*** *Output*

On exit: if INFO $= 0$ on exit, SCOND contains the ratio of the smallest value of $\mathrm{S}(i)$ to the largest value of $\mathrm{S}(i)$. If SCOND ≥ 0.1 and AMAX is neither too large nor too small, it is not worth scaling by S.

6: AMAX – ***double precision*** *Output*

On exit: $\max |a_{ij}|$. If AMAX is very close to overflow or underflow, the matrix A should be scaled.

7: INFO – INTEGER *Output*

On exit: INFO $= 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, the ith argument had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0

If INFO $= i$, the ith diagonal element of A is not positive (and hence A cannot be positive-definite).

7 Accuracy

The computed scale factors will be close to the exact scale factors.

8 Further Comments

The complex analogue of this routine is F07GTF (ZPPEQU).

9 Example

To equilibrate the symmetric positive-definite matrix A given by

$$A = \begin{pmatrix} 4.16 & -3.12 \times 10^{5} & 0.56 & -0.10 \\ -3.12 \times 10^{5} & 5.03 \times 10^{10} & -0.83 \times 10^{5} & 1.18 \times 10^{5} \\ 0.56 & -0.83 \times 10^{5} & 0.76 & 0.34 \\ -0.10 & 1.18 \times 10^{5} & 0.34 & 1.18 \end{pmatrix}.$$

Details of the scaling factors and the scaled matrix are output.

9.1 Program Text

```
*     F07GFF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=8)
      CHARACTER        UPLO
      PARAMETER        (UPLO='U')
*     .. Local Scalars ..
      DOUBLE PRECISION AMAX, BIG, SCOND, SJ, SMALL
      INTEGER          I, IFAIL, INFO, J, JJ, N
*     .. Local Arrays ..
      DOUBLE PRECISION AP((NMAX*(NMAX+1))/2), S(NMAX)
*     .. External Functions ..
      DOUBLE PRECISION X02AJF, X02AMF
      INTEGER          X02BHF
      EXTERNAL         X02AJF, X02AMF, X02BHF
*     .. External Subroutines ..
      EXTERNAL         DPPEQU, X04CCF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F07GFF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
*
*        Read the upper or lower triangular part of the matrix A from
*        data file
*
         IF (UPLO.EQ.'U') THEN
            READ (NIN,*) ((AP(I+(J*(J-1))/2),J=I,N),I=1,N)
         ELSE IF (UPLO.EQ.'L') THEN
            READ (NIN,*) ((AP(I+((2*N-J)*(J-1))/2),J=1,I),I=1,N)
         END IF
*
*        Print the matrix A
*
         IFAIL = 0
         CALL X04CCF(UPLO,'Non-unit diagonal',N,AP,'Matrix A',IFAIL)
         WRITE (NOUT,*)
*
*        Compute diagonal scaling factors
*
         CALL DPPEQU(UPLO,N,AP,S,SCOND,AMAX,INFO)
*
         IF (INFO.GT.0) THEN
            WRITE (NOUT,99999) 'Diagonal element', INFO,
     +        ' of A is non positive'
         ELSE
*
*           Print SCOND, AMAX and the scale factors
*
            WRITE (NOUT,99998) 'SCOND = ', SCOND, ', AMAX = ', AMAX
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Diagonal scaling factors'
            WRITE (NOUT,99997) (S(I),I=1,N)
            WRITE (NOUT,*)
*
*           Compute values close to underflow and overflow
*
            SMALL = X02AMF()/(X02AJF()*X02BHF())
            BIG = 1.0D0/SMALL
            IF ((SCOND.LT.0.1D0) .OR. (AMAX.LT.SMALL) .OR. (AMAX.GT.BIG)
     +          ) THEN
*
*              Scale A
*
```

```
               IF (UPLO.EQ.'U') THEN
                  DO 40 J = 1, N
                     SJ = S(J)
                     JJ = (J*(J-1))/2
                     DO 20 I = 1, J
                        AP(I+JJ) = S(I)*AP(I+JJ)*SJ
   20                CONTINUE
   40             CONTINUE
               ELSE IF (UPLO.EQ.'L') THEN
                  DO 80 J = 1, N
                     SJ = S(J)
                     JJ = ((2*N-J)*(J-1))/2
                     DO 60 I = J, N
                        AP(I+JJ) = S(I)*AP(I+JJ)*SJ
   60                CONTINUE
   80             CONTINUE
               END IF
*
*              Print the scaled matrix
*
               IFAIL = 0
               CALL X04CCF(UPLO,'Non-unit diagonal',N,AP,
     +                     'Scaled matrix',IFAIL)
*
            END IF
         END IF
      ELSE
         WRITE (NOUT,*) 'NMAX too small'
      END IF
      STOP
*
99999 FORMAT (1X,A,I4,A)
99998 FORMAT (1X,2(A,1P,E7.1))
99997 FORMAT ((1X,1P,7E11.1))
      END
```

9.2 Program Data

```
F07GFF Example Program Data
  4                                             :Value of N
  4.16D+00  -3.12D+05   0.56D+00  -0.10D+00
             5.03D+10  -0.83D+05   1.18D+05
                        0.76D+00   0.34D+00
                                   1.18D+00 :End of matrix A
```

9.3 Program Results

```
 F07GFF Example Program Results

 Matrix A
             1            2            3            4
 1    4.1600E+00  -3.1200E+05   5.6000E-01  -1.0000E-01
 2                 5.0300E+10  -8.3000E+04   1.1800E+05
 3                              7.6000E-01   3.4000E-01
 4                                           1.1800E+00

 SCOND = 3.9E-06, AMAX = 5.0E+10

 Diagonal scaling factors
     4.9E-01    4.5E-06    1.1E+00    9.2E-01

 Scaled matrix
           1          2          3          4
 1      1.0000    -0.6821     0.3149    -0.0451
 2                 1.0000    -0.4245     0.4843
 3                            1.0000     0.3590
 4                                       1.0000
```