NAG Library Routine Document

F04JMF

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

F04JMF solves a real linear equality-constrained least-squares problem.

2 Specification

```
SUBROUTINE F04JMF(M, N, P, A, LDA, B, LDB, C, D, X, WORK, LWORK, IFAIL)
INTEGER            M, N, P, LDA, LDB, LWORK, IFAIL
double precision   A(LDA,*), B(LDB,*), C(*), D(*), X(*),
1                  WORK(max(1,LWORK))
```

3 Description

F04JMF solves the real linear equality-constrained least-squares (LSE) problem

$$\underset{x}{\text{minimize}} \|c - Ax\|_2 \qquad \text{subject to} \qquad Bx = d$$

where A is an m by n matrix, B is a p by n matrix, c is an m element vector and d is a p element vector. It is assumed that $p \leq n \leq m + p$, $\text{rank}(B) = p$ and $\text{rank}(E) = n$, where $E = \begin{pmatrix} A \\ B \end{pmatrix}$. These conditions ensure that the LSE problem has a unique solution, which is obtained using a generalized RQ factorization of the matrices B and A.

F04JMF is based on the LAPACK routine SGGLSE/DGGLSE, see Anderson *et al.* (1999).

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia

Anderson E, Bai Z and Dongarra J (1992) Generalized *QR* factorization and its applications *Linear Algebra Appl. (Volume 162–164)* 243–271

Eldèn L (1980) Perturbation theory for the least-squares problem with linear equality constraints *SIAM J. Numer. Anal.* **17** 338–350

5 Parameters

1: M – INTEGER *Input*

On entry: m, the number of rows of the matrix A.

Constraint: $\text{M} \geq 0$.

2: N – INTEGER *Input*

On entry: n, the number of columns of the matrices A and B.

Constraint: $\text{N} \geq 0$.

3: P – INTEGER *Input*

On entry: p, the number of rows of the matrix B.

Constraint: $0 \leq \mathrm{P} \leq \mathrm{N} \leq \mathrm{M} + \mathrm{P}$.

4: A(LDA,$*$) – ***double precision*** array *Input/Output*

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

On entry: the m by n matrix A.

On exit: A is overwritten.

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F04JMF is called.

Constraint: $\mathrm{LDA} \geq \max(1, \mathrm{M})$.

6: B(LDB,$*$) – ***double precision*** array *Input/Output*

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

On entry: the p by n matrix B.

On exit: B is overwritten.

7: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F04JMF is called.

Constraint: $\mathrm{LDB} \geq \max(1, \mathrm{P})$.

8: C($*$) – ***double precision*** array *Input/Output*

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

On entry: the right-hand side vector c for the least-squares part of the LSE problem.

On exit: the residual sum of squares for the solution vector x is given by the sum of squares of elements $\mathrm{C}(\mathrm{N} - \mathrm{P} + 1), \mathrm{C}(\mathrm{N} - \mathrm{P} + 2), \ldots, \mathrm{C}(\mathrm{M})$, provided $m + p > n$; the remaining elements are overwritten.

9: D($*$) – ***double precision*** array *Input/Output*

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

On entry: the right-hand side vector d for the equality constraints.

On exit: D is overwritten.

10: X($*$) – ***double precision*** array *Output*

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

On exit: the solution vector x of the LSE problem.

11: WORK($\max(1, \mathrm{LWORK})$) – ***double precision*** array *Workspace*

On exit: if $\mathrm{IFAIL} = 0$, WORK(1) contains the minimum value of LWORK required for optimum performance.

12: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the subprogram from which F04JMF is called unless $\mathrm{LWORK} = -1$, in which case a workspace query is assumed and the routine only calculates the optimal dimension of WORK (using the formula given below).

Suggested value: for optimum performance LWORK should be at least $\mathrm{P} + \min(\mathrm{M}, \mathrm{N}) + \max(\mathrm{M}, \mathrm{N}) \times nb$, where nb is the ***blocksize***.

Constraint: $\mathrm{LWORK} \geq \max(1, \mathrm{M} + \mathrm{N} + \mathrm{P})$ or $\mathrm{LWORK} = -1$.

13: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

On exit: $\mathrm{IFAIL} = 0$ unless the routine detects an error (see Section 6).

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{M} < 0$,
or $\mathrm{N} < 0$,
or $\mathrm{P} < 0$,
or $\mathrm{P} > \mathrm{N}$,
or $\mathrm{P} < \mathrm{N} - \mathrm{M}$,
or $\mathrm{LDA} < \max(1, \mathrm{M})$,
or $\mathrm{LDB} < \max(1, \mathrm{P})$,
or $\mathrm{LWORK} < \max(1, \mathrm{M} + \mathrm{N} + \mathrm{P})$ and $\mathrm{LWORK} \neq -1$.

7 Accuracy

For an error analysis, see Anderson *et al.* (1992) and Eldèn (1980).

8 Further Comments

When $m \geq n = p$, the total number of floating-point operations is approximately $\frac{2}{3}n^2(6m + n)$; if $p \ll n$, the number reduces to approximately $\frac{2}{3}n^2(3m - n)$.

E04NCF/E04NCA may also be used to solve LSE problems. It differs from F04JMF in that it uses an iterative (rather than direct) method, and that it allows general upper and lower bounds to be specified for the variables x and the linear constraints Bx.

9 Example

This example solves the least-squares problem

$$\underset{x}{\mathrm{minimize}} \, \|c - Ax\|_2 \qquad \text{subject to} \qquad x_1 = x_3 \qquad \text{and} \qquad x_2 = x_4$$

where

$$c = \begin{pmatrix} -3.15 \\ -0.11 \\ 1.99 \\ -2.70 \\ 0.26 \\ 4.50 \end{pmatrix}$$

and

$$A = \begin{pmatrix} -0.57 & -1.28 & -0.39 & 0.25 \\ -1.93 & 1.08 & -0.31 & -2.14 \\ 2.30 & 0.24 & 0.40 & -0.35 \\ -1.93 & 0.64 & -0.66 & 0.08 \\ 0.15 & 0.30 & 0.15 & -2.13 \\ -0.02 & 1.03 & -1.43 & 0.50 \end{pmatrix};$$

the equality constraints are formulated by setting

$$B = \begin{pmatrix} 1.0 & 0.0 & -1.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & -1.0 \end{pmatrix}$$

and

$$d = \begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}.$$

9.1 Program Text

```
*     F04JMF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, NMAX, PMAX, LDA, LDB, LWORK
      PARAMETER        (MMAX=10,NMAX=10,PMAX=10,LDA=MMAX,LDB=PMAX,
     +                 LWORK=PMAX+NMAX+64*(MMAX+NMAX))
*     .. Local Scalars ..
      DOUBLE PRECISION RSS
      INTEGER          I, IFAIL, J, M, N, P
*     .. Local Arrays ..
      DOUBLE PRECISION A(LDA,NMAX), B(LDB,NMAX), C(MMAX), D(PMAX),
     +                 WORK(LWORK), X(NMAX)
*     .. External Functions ..
      DOUBLE PRECISION DDOT
      EXTERNAL         DDOT
*     .. External Subroutines ..
      EXTERNAL         F04JMF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F04JMF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) M, N, P
      IF (M.LE.MMAX .AND. N.LE.NMAX .AND. P.LE.PMAX) THEN
*
*        Read A, B, C and D from data file
*
         READ (NIN,*) ((A(I,J),J=1,N),I=1,M)
         READ (NIN,*) ((B(I,J),J=1,N),I=1,P)
         READ (NIN,*) (C(I),I=1,M)
         READ (NIN,*) (D(I),I=1,P)
*
*        Solve the equality-constrained least-squares problem
*
*        minimize ||C - A*X|| (in the 2-norm) subject to B*X = D
*
         IFAIL = 1
*
```

```
         CALL F04JMF(M,N,P,A,LDA,B,LDB,C,D,X,WORK,LWORK,IFAIL)
*
*        Print least-squares solution
*
         IF (IFAIL.EQ.0) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Constrained least-squares solution'
            WRITE (NOUT,99999) (X(I),I=1,N)
*
*           Compute the residual sum of squares
*
            WRITE (NOUT,*)
            RSS = DDOT(M-N+P,C(N-P+1),1,C(N-P+1),1)
            WRITE (NOUT,99998) 'Residual sum of squares = ', RSS
         ELSE
            WRITE (NOUT,99997) IFAIL
         END IF
      END IF
*
99999 FORMAT (1X,8F9.4)
99998 FORMAT (1X,A,1P,E10.2)
99997 FORMAT (1X,/1X,' ** F04JMF returned with IFAIL = ',I5)
      END
```

9.2 Program Data

```
F04JMF Example Program Data
  6  4  2                      :Values of M, N and P
 -0.57  -1.28  -0.39   0.25
 -1.93   1.08  -0.31  -2.14
  2.30   0.24   0.40  -0.35
 -1.93   0.64  -0.66   0.08
  0.15   0.30   0.15  -2.13
 -0.02   1.03  -1.43   0.50 :End of matrix A
  1.00   0.00  -1.00   0.00
  0.00   1.00   0.00  -1.00 :End of matrix B
 -3.15
 -0.11
  1.99
 -2.70
  0.26
  4.50                      :End of C
  0.00
  0.00                      :End of D
```

9.3 Program Results

```
 F04JMF Example Program Results

 Constrained least-squares solution
    0.4857   0.9956   0.4857   0.9956

 Residual sum of squares =   2.95E+01
```