

NAG Library Routine Document

F04KMF

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

F04KMF solves a complex linear equality-constrained least-squares problem.

2 Specification

```
SUBROUTINE F04KMF(M, N, P, A, LDA, B, LDB, C, D, X, WORK, LWORK, IFAIL)
INTEGER          M, N, P, LDA, LDB, LWORK, IFAIL
complex*16      A(LDA,*), B(LDB,*), C(*), D(*), X(*),
1               WORK(max(1,LWORK))
```

3 Description

F04KMF solves the complex linear equality-constrained least-squares (LSE) problem

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

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 .

F04KMF is based on the LAPACK routine CGGLSE/ZGGLSE, 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 | <i>Input</i> |
| | <i>On entry:</i> m , the number of rows of the matrix A . | |
| | <i>Constraint:</i> $M \geq 0$. | |
| 2: | N – INTEGER | <i>Input</i> |
| | <i>On entry:</i> n , the number of columns of the matrices A and B . | |
| | <i>Constraint:</i> $N \geq 0$. | |

- 3: P – INTEGER *Input*
On entry: p , the number of rows of the matrix B .
Constraint: $0 \leq P \leq N \leq M + P$.
- 4: A(LDA,*) – **complex*16** array *Input/Output*
Note: the second dimension of the array A must be at least $\max(1, N)$.
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 F04KMF is called.
Constraint: $LDA \geq \max(1, M)$.
- 6: B(LDB,*) – **complex*16** array *Input/Output*
Note: the second dimension of the array B must be at least $\max(1, N)$.
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 F04KMF is called.
Constraint: $LDB \geq \max(1, P)$.
- 8: C(*) – **complex*16** array *Input/Output*
Note: the dimension of the array C must be at least $\max(1, M)$.
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 $C(N - P + 1), C(N - P + 2), \dots, C(M)$, provided $m + p > n$; the remaining elements are overwritten.
- 9: D(*) – **complex*16** array *Input/Output*
Note: the dimension of the array D must be at least $\max(1, P)$.
On entry: the right-hand side vector d for the equality constraints.
On exit: D is overwritten.
- 10: X(*) – **complex*16** array *Output*
Note: the dimension of the array X must be at least $\max(1, N)$.
On exit: the solution vector x of the LSE problem.
- 11: WORK(max(1, LWORK)) – **complex*16** array *Workspace*
On exit: if 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 F04KMF is called unless 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 $P + \min(M, N) + \max(M, N) \times nb$, where *nb* is the **blocksize**.

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

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: 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 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:

IFAIL = 1

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

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 real floating-point operations is approximately $\frac{8}{3}n^2(6m + n)$; if $p \ll n$, the number reduces to approximately $\frac{8}{3}n^2(3m - n)$.

9 Example

This example solves the least-squares problem

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

where

$$c = \begin{pmatrix} -1.54 + 0.76i \\ 0.12 - 1.92i \\ -9.08 - 4.31i \\ 7.49 + 3.65i \\ -5.63 - 2.12i \\ 2.37 + 8.03i \end{pmatrix}$$

and

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.03 + 0.96i & -0.91 + 2.06i & -0.05 + 0.41i \\ -0.98 + 1.98i & -1.20 + 0.19i & -0.66 + 0.42i & -0.81 + 0.56i \\ 0.62 - 0.46i & 1.01 + 0.02i & 0.63 - 0.17i & -1.11 + 0.60i \\ 0.37 + 0.38i & 0.19 - 0.54i & -0.98 - 0.36i & 0.22 - 0.20i \\ 0.83 + 0.51i & 0.20 + 0.01i & -0.17 - 0.46i & 1.47 + 1.59i \\ 1.08 - 0.28i & 0.20 - 0.12i & -0.07 + 1.23i & 0.26 + 0.26i \end{pmatrix};$$

the equality constraints are formulated by setting

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

and

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

9.1 Program Text

```
*      F04KMF 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 ..
      COMPLEX *16      A(LDA,NMAX), B(LDB,NMAX), C(MMAX), D(PMAX),
+                      WORK(LWORK), X(NMAX)
*      .. External Functions ..
      COMPLEX *16      ZDOTC
      EXTERNAL         ZDOTC
*      .. External Subroutines ..
      EXTERNAL         F04KMF
*      .. Intrinsic Functions ..
      INTRINSIC        AIMAG, DBLE
*      .. Executable Statements ..
      WRITE (NOUT,*) 'F04KMF 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 F04KMF(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) (' (' ,DBLE(X(I)),',',',',AIMAG(X(I)),')',I=1,
```

```

      +      N)
*
*      Compute the residual sum of squares
*
      WRITE (NOUT,*)
      RSS = DBLE(ZDOTC(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
END IF
*
99999 FORMAT ((3X,4(A,F7.4,A,F7.4,A,:)))
99998 FORMAT (1X,A,1P,E10.2)
99997 FORMAT (1X,/1X,' ** F04KMF returned with IFAIL = ',I5)
END

```

9.2 Program Data

F04KMF Example Program Data

```

  6  4  2                                     :Values of M, N and P
( 0.96,-0.81) (-0.03, 0.96) (-0.91, 2.06) (-0.05, 0.41)
(-0.98, 1.98) (-1.20, 0.19) (-0.66, 0.42) (-0.81, 0.56)
( 0.62,-0.46) ( 1.01, 0.02) ( 0.63,-0.17) (-1.11, 0.60)
( 0.37, 0.38) ( 0.19,-0.54) (-0.98,-0.36) ( 0.22,-0.20)
( 0.83, 0.51) ( 0.20, 0.01) (-0.17,-0.46) ( 1.47, 1.59)
( 1.08,-0.28) ( 0.20,-0.12) (-0.07, 1.23) ( 0.26, 0.26) :End of matrix A
( 1.00, 0.00) ( 0.00, 0.00) (-1.00, 0.00) ( 0.00, 0.00)
( 0.00, 0.00) ( 1.00, 0.00) ( 0.00, 0.00) (-1.00, 0.00) :End of matrix B
(-1.54, 0.76)
( 0.12,-1.92)
(-9.08,-4.31)
( 7.49, 3.65)
(-5.63,-2.12)
( 2.37, 8.03)                                     :End of C
( 0.00, 0.00)
( 0.00, 0.00)                                     :End of D

```

9.3 Program Results

F04KMF Example Program Results

Constrained least-squares solution

```
( 1.0789,-1.9523) (-0.7581, 3.7203) ( 1.0789,-1.9523) (-0.7581, 3.7203)
```

Residual sum of squares = 1.75E+02
