NAG Library Routine Document

F04AMF

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

F04AMF calculates the accurate least squares solution of a set of m linear equations in n unknowns, $m \geq n$ and rank $= n$, with multiple right-hand sides, $AX = B$, using a QR factorization and iterative refinement.

2 Specification

```
SUBROUTINE F04AMF (A, LDA, X, LDX, B, LDB, M, N, IR, EPS, QR, LDQR, ALPHA,      &
                   E, Y, Z, R, IPIV, IFAIL)

INTEGER            LDA, LDX, LDB, M, N, IR, LDQR, IPIV(N), IFAIL
REAL (KIND=nag_wp) A(LDA,N), X(LDX,IR), B(LDB,IR), EPS, QR(LDQR,N),            &
                   ALPHA(N), E(N), Y(N), Z(N), R(M)
```

3 Description

To compute the least squares solution to a set of m linear equations in n unknowns $(m \geq n)$ $AX = B$, F04AMF first computes a QR factorization of A with column pivoting, $AP = QR$, where R is upper triangular, Q is an m by m orthogonal matrix, and P is a permutation matrix. Q^{T} is applied to the m by r right-hand side matrix B to give $C = Q^{\mathrm{T}}B$, and the n by r solution matrix X is calculated, to a first approximation, by back-substitution in $RX = C$. The residual matrix $S = B - AX$ is calculated using ***additional precision***, and a correction D to X is computed as the least squares solution to $AD = S$. X is replaced by $X + D$ and this iterative refinement of the solution is repeated until full machine accuracy has been obtained.

4 References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation II, Linear Algebra* Springer–Verlag

5 Parameters

1: A(LDA,N) – REAL (KIND=nag_wp) array *Input*

On entry: the m by n matrix A.

2: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F04AMF is called.

Constraint: LDA $\geq$ M.

3: X(LDX,IR) – REAL (KIND=nag_wp) array *Output*

On exit: the n by r solution matrix X.

4: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F04AMF is called.

Constraint: LDX $\geq$ N.

5: B(LDB,IR) – REAL (KIND=nag_wp) array *Input*

On entry: the m by r right-hand side matrix B.

6: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F04AMF is called.

Constraint: LDB $\geq$ M.

7: M – INTEGER *Input*

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

Constraint: M ≥ 1.

8: N – INTEGER *Input*

On entry: n, the number of columns of the matrix A, i.e., the number of unknowns.

Constraint: $0 \leq$ N $\leq$ M.

9: IR – INTEGER *Input*

On entry: r, the number of right-hand sides.

10: EPS – REAL (KIND=nag_wp) *Input*

On entry: must be set to the value of the ***machine precision***.

11: QR(LDQR,N) – REAL (KIND=nag_wp) array *Output*

On exit: details of the QR factorization.

12: LDQR – INTEGER *Input*

On entry: the first dimension of the array QR as declared in the (sub)program from which F04AMF is called.

Constraint: LDQR $\geq$ M.

13: ALPHA(N) – REAL (KIND=nag_wp) array *Output*

On exit: the diagonal elements of the upper triangular matrix R.

14: E(N) – REAL (KIND=nag_wp) array *Workspace*

15: Y(N) – REAL (KIND=nag_wp) array *Workspace*

16: Z(N) – REAL (KIND=nag_wp) array *Workspace*

17: R(M) – REAL (KIND=nag_wp) array *Workspace*

18: IPIV(N) – INTEGER array *Output*

On exit: details of the column interchanges.

19: 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.

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.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

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 $= 1$

The rank of A is less than n; the problem does not have a unique solution.

IFAIL $= 2$

The iterative refinement fails to converge, i.e., the matrix A is too ill-conditioned.

7 Accuracy

Although the correction process is continued until the solution has converged to full machine accuracy, all the figures in the final solution may not be correct since the correction D to X is itself the solution to a linear least squares problem. For a detailed error analysis see page 116 of Wilkinson and Reinsch (1971).

8 Further Comments

The time taken by F04AMF is approximately proportional to $n^2(3m - n)$, provided r is small compared with n.

9 Example

This example calculates the accurate least squares solution of the equations

$$\begin{aligned} 1.1x_1 + 0.9x_2 &= 2.2 \\ 1.2x_1 + 1.0x_2 &= 2.3 \\ 1.0x_1 + 1.0x_2 &= 2.1 \end{aligned}$$

9.1 Program Text

```
    Program f04amfe

!     F04AMF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f04amf, nag_wp, x02ajf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
```

```
      Real (Kind=nag_wp)              :: eps
      Integer                         :: i, ifail, ir, lda, ldb, ldqr, ldx,   &
                                         m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), alpha(:), b(:,:), e(:),      &
                                         qr(:,:), r(:), x(:,:), y(:), z(:)
      Integer, Allocatable            :: ipiv(:)
!     .. Executable Statements ..
      Write (nout,*) 'F04AMF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n
      ir = 1
      lda = m
      ldb = m
      ldqr = m
      ldx = n
      Allocate (a(lda,n),alpha(n),b(ldb,ir),e(n),qr(ldqr,n),r(m),x(ldx,ir), &
        y(n),z(n),ipiv(n))
      Read (nin,*)(a(i,1:n),b(i,1:ir),i=1,m)
      eps = x02ajf()

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f04amf(a,lda,x,ldx,b,ldb,m,n,ir,eps,qr,ldqr,alpha,e,y,z,r,ipiv, &
        ifail)

      Write (nout,*) ' Solution'
      Do i = 1, n
        Write (nout,99999) x(i,1:ir)
      End Do

99999 Format (1X,8F9.4)
    End Program f04amfe
```

9.2 Program Data

```
F04AMF Example Program Data
  3  2                 : m, n
  1.1  0.9    2.2
  1.2  1.0    2.3
  1.0  1.0    2.1      : matrices A and B
```

9.3 Program Results

```
 F04AMF Example Program Results

  Solution
    1.3010
    0.7935
```