NAG Toolbox for MATLAB

f04jl

1 Purpose

f04jl solves a real general Gauss–Markov linear (least-squares) model problem.

2 Syntax

```
[a, b, d, x, y, ifail] = f04jl(a, b, d, 'm', m, 'n', n, 'p', p)
```

3 Description

f04jl solves the real general Gauss–Markov linear model (GLM) problem

$$\underset{x}{\text{minimize}} \|y\|_2 \qquad \text{subject to} \qquad d = Ax + By$$

where A is an m by n matrix, B is an m by p matrix and d is an m element vector. It is assumed that $n \leq m \leq n+p$, $\text{rank}(A) = n$ and $\text{rank}(E) = m$, where $E = \begin{pmatrix} A & B \end{pmatrix}$. Under these assumptions, the problem has a unique solution x and a minimal 2-norm solution y, which is obtained using a generalized QR factorization of the matrices A and B.

In particular, if the matrix B is square and nonsingular, then the GLM problem is equivalent to the weighted linear least-squares problem

$$\underset{x}{\text{minimize}} \left\|B^{-1}(d - Ax)\right\|_2.$$

f04jl is based on the LAPACK routine SGGGLM/DGGGLM, 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

5 Parameters

5.1 Compulsory Input Parameters

1: **a(lda,$*$) – double array**

The first dimension of the array **a** must be at least $\max(1, \mathbf{m})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The m by n matrix A.

2: **b(ldb,$*$) – double array**

The first dimension of the array **b** must be at least $\max(1, \mathbf{m})$

The second dimension of the array must be at least $\max(1, \mathbf{p})$

The m by p matrix B.

3: **d**($*$) **– double array**

Note: the dimension of the array **d** must be at least $\max(1, \mathbf{m})$.

The left-hand side vector d of the GLM equation.

5.2 Optional Input Parameters

1: **m – int32 scalar**

Default: The dimension of the array **d**.

m, the number of rows of the matrices A and B.

Constraint: $\mathbf{m} \geq 0$.

2: **n – int32 scalar**

Default: The second dimension of the array **a**.

n, the number of columns of the matrix A.

Constraint: $0 \leq \mathbf{n} \leq \mathbf{m}$.

3: **p – int32 scalar**

Default: The second dimension of the array **b**.

p, the number of columns of the matrix B.

Constraint: $\mathbf{p} \geq \mathbf{m} - \mathbf{n}$.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, work, lwork

5.4 Output Parameters

1: **a**(**lda**,$*$) **– double array**

The first dimension of the array **a** must be at least $\max(1, \mathbf{m})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The array is overwritten.

2: **b**(**ldb**,$*$) **– double array**

The first dimension of the array **b** must be at least $\max(1, \mathbf{m})$

The second dimension of the array must be at least $\max(1, \mathbf{p})$

The array is overwritten.

3: **d**($*$) **– double array**

Note: the dimension of the array **d** must be at least $\max(1, \mathbf{m})$.

The array is overwritten.

4: **x**($*$) **– double array**

Note: the dimension of the array **x** must be at least $\max(1, \mathbf{n})$.

The solution vector x of the GLM problem.

5: **y(*) – double array**

Note: the dimension of the array **y** must be at least $\max(1, \mathbf{p})$.

The solution vector y of the GLM problem.

6: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, $\mathbf{m} < 0$,
or $\mathbf{n} < 0$,
or $\mathbf{n} > \mathbf{m}$,
or $\mathbf{p} < 0$,
or $\mathbf{p} < \mathbf{m} - \mathbf{n}$,
or $\mathbf{lda} < \max(1, \mathbf{m})$,
or $\mathbf{ldb} < \max(1, \mathbf{m})$,
or $\mathbf{lwork} < \max(1, \mathbf{m} + \mathbf{n} + \mathbf{p})$ and $\mathbf{lwork} \neq -1$.

7 Accuracy

For an error analysis, see Anderson *et al.* 1992.

8 Further Comments

When $p = m \geq n$, the total number of floating-point operations is approximately $\frac{2}{3}\left(2m^3 - n^3\right) + 4nm^2$; when $p = m = n$, the total number of floating-point operations is approximately $\frac{14}{3}m^3$.

9 Example

```
a = [-0.57, -1.28, -0.39;
     -1.93, 1.08, -0.31;
     2.3, 0.24, -0.4;
     -0.02, 1.03, -1.43];
b = [0.5, 0, 0, 0;
     0, 1, 0, 0;
     0, 0, 2, 0;
     0, 0, 0, 5];
d = [1.31;
     -4.01;
     5.56;
     3.22];
[aOut, bOut, dOut, x, y, ifail] = f04jl(a, b, d)
```

```
aOut =
    3.0562   -0.2694   -0.0232
    0.5322   -1.9623    0.7189
   -0.6343   -0.1120    1.3913
    0.0055    0.2893    0.5605
bOut =
   -1.2131    0.2669   -0.8691    0.6131
   -0.5707    0.7105    1.8876    1.8248
    0.1229    0.2076    2.4417    2.9277
   -0.0373   -0.1437   -0.2765   -2.3759
dOut =
```

```
    2.0000
   -1.0000
   -3.0000
   -0.0000
x =
    2.0000
   -1.0000
   -3.0000
y =
   1.0e-15 *
   -0.0127
   -0.0489
   -0.0941
    0.1534
ifail =
           0
```