NAG Toolbox for MATLAB

f04jg

1 Purpose

f04jg finds the solution of a linear least-squares problem, $Ax = b$, where A is a real m by $n(m \geq n)$ matrix and b is an m element vector. If the matrix of observations is not of full rank, then the minimal least-squares solution is returned.

2 Syntax

```
[a, b, svd, sigma, irank, work, ifail] = f04jg(a, b, tol, lwork, 'm', m,
'n', n)
```

3 Description

The minimal least-squares solution of the problem $Ax = b$ is the vector x of minimum (Euclidean) length which minimizes the length of the residual vector $r = b - Ax$.

The real m by $n(m \geq n)$ matrix A is factorized as

$$A = Q\begin{pmatrix} U \\ 0 \end{pmatrix}$$

where Q is an m by m orthogonal matrix and U is an n by n upper triangular matrix. If U is of full rank, then the least-squares solution is given by

$$x = \left(U^{-1} 0\right) Q^{\mathrm{T}} b.$$

If U is not of full rank, then the singular value decomposition of U is obtained so that U is factorized as

$$U = RDP^{\mathrm{T}},$$

where R and P are n by n orthogonal matrices and D is the n by n diagonal matrix

$$D = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n),$$

with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$, these being the singular values of A. If the singular values $\sigma_{k+1}, \ldots, \sigma_n$ are negligible, but σ_k is not negligible, relative to the data errors in A, then the rank of A is taken to be k and the minimal least-squares solution is given by

$$x = P\begin{pmatrix} S^{-1} & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} R^{\mathrm{T}} & 0 \\ 0 & I \end{pmatrix} Q^{\mathrm{T}} b,$$

where $S = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k)$.

This function obtains the factorizations by a call to f02wd.

The function also returns the value of the standard error

$$\begin{aligned} \sigma &= \sqrt{\frac{r^{\mathrm{T}} r}{m-k}}, \quad \text{if } m > k, \\ &= 0, \quad \text{if } m = k, \end{aligned}$$

$r^{\mathrm{T}} r$ being the residual sum of squares and k the rank of A.

4 References

Lawson C L and Hanson R J 1974 *Solving Least-squares Problems* Prentice–Hall

5 Parameters

5.1 Compulsory Input Parameters

1: **a(lda,n) – double array**

lda, the first dimension of the array, must be at least **m**.

The m by n matrix A.

2: **b(m) – double array**

The right-hand side vector b.

3: **tol – double scalar**

A relative tolerance to be used to determine the rank of A. **tol** should be chosen as approximately the largest relative error in the elements of A. For example, if the elements of A are correct to about 4 significant figures then **tol** should be set to about 5×10^{-4}. See Section 8 for a description of how **tol** is used to determine rank. If **tol** is outside the range $(\epsilon, 1.0)$, where ϵ is the ***machine precision***, then the value ϵ is used in place of **tol**. For most problems this is unreasonably small.

4: **lwork – int32 scalar**

Constraint: **lwork** $\geq 4 \times$ **n**.

5.2 Optional Input Parameters

1: **m – int32 scalar**

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

m, the number of rows of **a**.

Constraint: **m** $\geq$ **n**.

2: **n – int32 scalar**

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

n, the number of columns of **a**.

Constraint: $1 \leq$ **n** $\leq$ **m**.

5.3 Input Parameters Omitted from the MATLAB Interface

lda

5.4 Output Parameters

1: **a(lda,n) – double array**

If **svd** is returned as **false**, **a** contains details of the QU factorization of A (see f02wd for further details).

If **svd** is returned as **true**, the first n rows of **a** are overwritten by the right-hand singular vectors, stored by rows; and the remaining rows of the array are used as workspace.

2: **b(m) – double array**

The first n elements of **b** contain the minimal least-squares solution vector x. The remaining $m - n$ elements are used for workspace.

3: **svd – logical scalar**

Is returned as **false** if the least-squares solution has been obtained from the QU factorization of A. In this case A is of full rank. **svd** is returned as **true** if the least-squares solution has been obtained from the singular value decomposition of A.

4: **sigma – double scalar**

The standard error, i.e., the value $\sqrt{r^{\mathrm{T}}r/(m-k)}$ when $m>k$, and the value zero when $m=k$. Here r is the residual vector $b-Ax$ and k is the rank of A.

5: **irank – int32 scalar**

k, the rank of the matrix A. It should be noted that it is possible for **irank** to be returned as n and **svd** to be returned as **true**. This means that the matrix U only just failed the test for nonsingularity.

6: **work(lwork) – double array**

If **svd** is returned as **false**, then the first n elements of **work** contain information on the QU factorization of A (see parameter **a** above and f02wd), and **work**$(n+1)$ contains the condition number $\|U\|_E\|U^{-1}\|_E$ of the upper triangular matrix U.

If **svd** is returned as **true**, then the first n elements of **work** contain the singular values of A arranged in descending order and **work**$(n+1)$ contains the total number of iterations taken by the QR algorithm. The rest of **work** is used as workspace.

7: **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, **n** <1,
or **m** $<$ **n**,
or **lda** $<$ **m**,
or **lwork** $<4\times$ **n**.

ifail $=2$

The QR algorithm has failed to converge to the singular values in $50\times$ **n** iterations. This failure can only happen when the singular value decomposition is employed, but even then it is not likely to occur.

7 Accuracy

The computed factors Q, U, R, D and P^{T} satisfy the relations

$$Q\begin{pmatrix} U \\ 0 \end{pmatrix} = A+E, Q\begin{pmatrix} R & 0 \\ 0 & I \end{pmatrix}\begin{pmatrix} D \\ 0 \end{pmatrix}P^{\mathrm{T}} = A+F,$$

where

$$\|E\|_2 \leq c_1\epsilon\|A\|_2,$$

$$\|F\|_2 \leq c_2\epsilon\|A\|_2,$$

ϵ being the ***machine precision***, and c_1 and c_2 being modest functions of m and n. Note that $\|A\|_2=\sigma_1$.

For a fuller discussion, covering the accuracy of the solution x see Lawson and Hanson 1974, especially pages 50 and 95.

8 Further Comments

If the least-squares solution is obtained from the QU factorization then the time taken by the function is approximately proportional to $n^2(3m - n)$. If the least-squares solution is obtained from the singular value decomposition then the time taken is approximately proportional to $n^2(3m + 19n)$. The approximate proportionality factor is the same in each case.

This function is column biased and so is suitable for use in paged environments.

Following the QU factorization of A the condition number

$$c(U) = \|U\|_E \|U^{-1}\|_E$$

is determined and if $c(U)$ is such that

$$c(U) \times \textbf{tol} > 1.0$$

then U is regarded as singular and the singular values of A are computed. If this test is not satisfied, U is regarded as nonsingular and the rank of A is set to n. When the singular values are computed the rank of A, say k, is returned as the largest integer such that

$$\sigma_k > \textbf{tol} \times \sigma_1,$$

unless $\sigma_1 = 0$ in which case k is returned as zero. That is, singular values which satisfy $\sigma_i \leq \textbf{tol} \times \sigma_1$ are regarded as negligible because relative perturbations of order **tol** can make such singular values zero.

9 Example

```
a = [0.05, 0.05, 0.25, -0.25;
     0.25, 0.25, 0.05, -0.05;
     0.35, 0.35, 1.75, -1.75;
     1.75, 1.75, 0.35, -0.35;
     0.3, -0.3, 0.3, 0.3;
     0.4, -0.4, 0.4, 0.4];
b = [1;
     2;
     3;
     4;
     5;
     6];
tol = 0.0005;
lwork = int32(32);
[aOut, bOut, svd, sigma, irank, work, ifail] = f04jg(a, b, tol, lwork)
```

```
aOut =
    0.5000    0.5000    0.5000   -0.5000
   -0.5000   -0.5000    0.5000   -0.5000
    0.5000   -0.5000    0.5000    0.5000
    0.5000   -0.5000   -0.5000   -0.5000
    0.1604   -0.5793   -0.0473    0.1714
    0.2138   -0.7724   -0.0630    0.2228
bOut =
    4.9667
   -2.8333
    4.5667
    3.2333
    1.1258
    0.8292
svd =
     1
sigma =
    0.9092
```

```
irank =
           3
work =
    3.0000
    2.0000
    1.0000
    0.0000
    8.0000
    1.7333
   -0.4000
    7.8000
    1.0000
    2.0000
    4.0000
    4.0000
         0
    0.0015
    0.0000
    0.0000
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
         0
ifail =
           0
```