NAG Toolbox for MATLAB

f11jd

1 Purpose

f11jd solves a system of linear equations involving the preconditioning matrix corresponding to SSOR applied to a real sparse symmetric matrix, represented in symmetric co-ordinate storage format.

2 Syntax

```
[x, ifail] = f11jd(a, irow, icol, rdiag, omega, check, y, 'n', n, 'nnz', nnz)
```

3 Description

f11jd solves a system of equations

$$Mx = y$$

involving the preconditioning matrix

$$M = \frac{1}{\omega(2-\omega)}(D+\omega L)D^{-1}(D+\omega L)^{\mathrm{T}}$$

corresponding to symmetric successive-over-relaxation (SSOR) (see Young 1971) on a linear system $Ax = b$, where A is a sparse symmetric matrix stored in symmetric co-ordinate storage (SCS) format (see Section 2.1.2 in the F11 Chapter Introduction).

In the definition of M given above D is the diagonal part of A, L is the strictly lower triangular part of A, and ω is a user-defined relaxation parameter.

It is envisaged that a common use of f11jd will be to carry out the preconditioning step required in the application of f11ge to sparse linear systems. For an illustration of this use of f11jd see the example program given in Section 9. f11jd is also used for this purpose by the Black Box function f11je.

4 References

Young D 1971 *Iterative Solution of Large Linear Systems* Academic Press, New York

5 Parameters

5.1 Compulsory Input Parameters

1: **a**(**nnz**) – **double array**

The nonzero elements in the lower triangular part of the matrix A, ordered by increasing row index, and by increasing column index within each row. Multiple entries for the same row and column indices are not permitted. The function f11zb may be used to order the elements in this way.

2: **irow**(**nnz**) – **int32 array**

3: **icol**(**nnz**) – **int32 array**

The row and column indices of the nonzero elements supplied in **a**.

Constraints:

$1 \leq \mathbf{irow}(i) \leq \mathbf{n}$ and $1 \leq \mathbf{icol}(i) \leq \mathbf{irow}(i)$, for $i = 1, 2, \ldots, \mathbf{nnz}$;
$\mathbf{irow}(i-1) < \mathbf{irow}(i)$ or $\mathbf{irow}(i-1) = \mathbf{irow}(i)$ and $\mathbf{icol}(i-1) < \mathbf{icol}(i)$, for $i = 2, 3, \ldots, \mathbf{nnz}$.

irow and **icol** must satisfy the following constraints (which may be imposed by a call to f11zb):

4: **rdiag(n) – double array**

The elements of the diagonal matrix D^{-1}, where D is the diagonal part of A.

5: **omega – double scalar**

The relaxation parameter ω.

Constraint: $0.0 \leq$ **omega** ≤ 2.0.

6: **check – string**

Specifies whether or not the input data should be checked.

check = 'C'

Checks are carried out on the values of **n**, **nnz**, **irow**, **icol** and **omega**.

check = 'N'

None of these checks are carried out.

See also Section 8.2.

Constraint: **check** = 'C' or 'N'.

7: **y(n) – double array**

The right-hand side vector y.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the arrays **rdiag**, **y**, **x**. (An error is raised if these dimensions are not equal.)

n, the order of the matrix A.

Constraint: **n** ≥ 1.

2: **nnz – int32 scalar**

Default: The dimension of the arrays **a**, **irow**, **icol**. (An error is raised if these dimensions are not equal.)

the number of nonzero elements in the lower triangular part of A.

Constraint: $1 \leq$ **nnz** $\leq$ **n** $\times$ (**n** $+ 1$)/2.

5.3 Input Parameters Omitted from the MATLAB Interface

iwork

5.4 Output Parameters

1: **x(n) – double array**

The solution vector x.

2: **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, **check** $\neq$ 'C' or 'N'.

ifail $= 2$

On entry, $\mathbf{n} < 1$,
or $\mathbf{nnz} < 1$,
or $\mathbf{nnz} > \mathbf{n} \times (\mathbf{n}+1)/2$,
or **omega** lies outside the interval $[0.0, 2.0]$,

ifail $= 3$

On entry, the arrays **irow** and **icol** fail to satisfy the following constraints:

$1 \leq \mathbf{irow}(i) \leq \mathbf{n}$ and $1 \leq \mathbf{icol}(i) \leq \mathbf{irow}(i)$, for $i = 1, 2, \ldots, \mathbf{nnz}$;

$\mathbf{irow}(i-1) < \mathbf{irow}(i)$ or $\mathbf{irow}(i-1) = \mathbf{irow}(i)$ and $\mathbf{icol}(i-1) < \mathbf{icol}(i)$, for $i = 2, 3, \ldots, \mathbf{nnz}$.

Therefore a nonzero element has been supplied which does not lie in the lower triangular part of A, is out of order, or has duplicate row and column indices. Call f11zb to reorder and sum or remove duplicates.

7 Accuracy

The computed solution x is the exact solution of a perturbed system of equations $(M + \delta M)x = y$, where

$$|\delta M| \leq c(n)\epsilon |D + \omega L| \left|D^{-1}\right| \left|(D + \omega L)^{\mathrm{T}}\right|,$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***.

8 Further Comments

8.1 Timing

The time taken for a call to f11jd is proportional to **nnz**.

8.2 Use of check

It is expected that a common use of f11jd will be to carry out the preconditioning step required in the application of f11ge to sparse symmetric linear systems. In this situation f11jd is likely to be called many times with the same matrix M. In the interests of both reliability and efficiency, you are recommended to set **check** to 'C' for the first of such calls, and to 'N' for all subsequent calls.

9 Example

```
a = [4;
     1;
     5;
     2;
     2;
     3;
     -1;
     1;
     4;
     1;
     -2;
```

```
     3;
     2;
     -1;
     -2;
     5];
irow = [int32(1);
     int32(2);
     int32(2);
     int32(3);
     int32(4);
     int32(4);
     int32(5);
     int32(5);
     int32(5);
     int32(6);
     int32(6);
     int32(6);
     int32(7);
     int32(7);
     int32(7);
     int32(7)];
icol = [int32(1);
     int32(1);
     int32(2);
     int32(3);
     int32(2);
     int32(4);
     int32(1);
     int32(4);
     int32(5);
     int32(2);
     int32(5);
     int32(6);
     int32(1);
     int32(2);
     int32(3);
     int32(7)];
rdiag = [0.25;
     0.2;
     0.5;
     0.3333333333333333;
     0.25;
     0.3333333333333333;
     0.2];
omega = 1;
check = 'C';
y = [15;
     18;
     -8;
     21;
     11;
     10;
     29];
[x, ifail] = f11jd(a, irow, icol, rdiag, omega, check, y)
```

```
x =
    2.9037
    1.2534
   -0.7300
    3.6306
    4.4083
    3.9917
    3.2700
ifail =
           0
```