NAG Toolbox for MATLAB

f11dd

1 Purpose

f11dd solves a system of linear equations involving the preconditioning matrix corresponding to SSOR applied to a real sparse nonsymmetric matrix, represented in co-ordinate storage format.

2 Syntax

```
[x, ifail] = f11dd(trans, a, irow, icol, rdiag, omega, check, y, 'n', n, 'nnz', nnz)
```

3 Description

f11dd solves a system of linear equations

$$Mx = y, \quad \text{or} \quad M^{\mathrm{T}}x = y,$$

according to the value of the parameter **trans**, where the matrix

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

corresponds to symmetric successive-over-relaxation (SSOR) (see Young 1971) applied to a linear system $Ax = b$, where A is a real sparse nonsymmetric matrix stored in co-ordinate storage (CS) format (see Section 2.1.1 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, U is the strictly upper triangular part of A, and ω is a user-defined relaxation parameter.

It is envisaged that a common use of f11dd will be to carry out the preconditioning step required in the application of f11be to sparse linear systems. For an illustration of this use of f11dd see the example program given in Section 9. f11dd is also used for this purpose by the Black Box function f11de.

4 References

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

5 Parameters

5.1 Compulsory Input Parameters

1: **trans – string**

Specifies whether or not the matrix M is transposed.

trans = 'N'

$Mx = y$ is solved.

trans = 'T'

$M^{\mathrm{T}}x = y$ is solved.

Constraint: **trans** = 'N' or 'T'.

2: **a(nnz) – double array**

The nonzero elements in 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 f11za may be used to order the elements in this way.

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

4: **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 $1 \leq \mathbf{icol}(i) \leq \mathbf{n}$, 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 f11za):

5: **rdiag**(**n**) – **double array**

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

6: **omega** – **double scalar**

The relaxation parameter ω.

Constraint: $0.0 < \mathbf{omega} < 2.0$.

7: **check** – **string**

Specifies whether or not the CS representation of the matrix M should be checked.

check = 'C'

Checks are carried 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'.

8: **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: $\mathbf{n} \geq 1$.

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 matrix A.

Constraint: $1 \leq \mathbf{nnz} \leq \mathbf{n}^2$.

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, **trans** $\neq$ 'N' or 'T',
or **check** $\neq$ 'C' or 'N'.

ifail $= 2$

On entry, $\mathbf{n} < 1$,
or $\mathbf{nnz} < 1$,
or $\mathbf{nnz} > \mathbf{n}^2$,
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 $1 \leq \mathbf{icol}(i) \leq \mathbf{n}$, 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 matrix A, is out of order, or has duplicate row and column indices. Call f11za to reorder and sum or remove duplicates.

ifail $= 4$

On entry, the matrix A has a zero diagonal element. The SSOR preconditioner is not appropriate for this problem.

7 Accuracy

If **trans** $=$ 'N' the computed solution x is the exact solution of a perturbed system of equations $(M + \delta M)x = y$, where

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

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***. An equivalent result holds when **trans** $=$ 'T'.

8 Further Comments

8.1 Timing

The time taken for a call to f11dd is proportional to **nnz**.

8.2 Use of check

It is expected that a common use of f11dd will be to carry out the preconditioning step required in the application of f11be to sparse linear systems. In this situation f11dd 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

```
trans = 'N';
a = [2;
     1;
     -1;
     -3;
     -2;
     1;
     1;
     5;
     3;
     1;
     -2;
     -3;
     -1;
     4;
     -2;
     -6];
irow = [int32(1);
     int32(1);
     int32(1);
     int32(2);
     int32(2);
     int32(2);
     int32(3);
     int32(3);
     int32(3);
     int32(3);
     int32(4);
     int32(4);
     int32(4);
     int32(5);
     int32(5);
     int32(5)];
icol = [int32(1);
     int32(2);
     int32(4);
     int32(2);
     int32(3);
     int32(5);
     int32(1);
     int32(3);
     int32(4);
     int32(5);
     int32(1);
     int32(4);
     int32(5);
     int32(2);
     int32(3);
     int32(5)];
rdiag = [0.5;
     -0.3333333333333333;
     0.2;
     -0.3333333333333333;
     -0.1666666666666667];
omega = 1.1;
check = 'C';
y = [0;
     -7;
     33;
     -19;
     -28];
[x, ifail] = f11dd(trans, a, irow, icol, rdiag, omega, check, y)
```

```
x =
    1.5987
    1.9265
    2.4820
    4.8333
    3.9182
ifail =
           0
```