NAG Toolbox for MATLAB

f11md

1 Purpose

f11md computes a column permutation suitable for *LU* factorization (by f11me) of a real sparse matrix in compressed column (Harwell–Boeing) format and applies it to the matrix. This function must be called prior to f11me.

2 Syntax

```
[iprm, ifail] = f11md(spec, n, icolzp, irowix, iprm)
```

3 Description

Given a sparse matrix in compressed column (Harwell–Boeing) format A and a choice of column permutation schemes, the function computes those data structures that will be needed by the *LU* factorization function f11me and associated functions f11mm, f11mf and f11mh. The column permutation choices are:

original order (that is, no permutation);

user-supplied permutation;

a permutation, computed by the function, designed to minimize fill-in during the *LU* factorization.

The algorithm for this computed permutation is based on the approximate minimum degree column ordering algorithm COLAMD. The computed permutation is not sensitive to the magnitude of the nonzero values of A.

4 References

Amestoy P R, Davis T A and Duff I S 1996 An approximate minimum degree ordering algorithm *SIAM J. Matrix Anal. Appl.* **17** 886–905

Gilbert J R and Larimore S I 2004 A column approximate minimum degree ordering algorithm *ACM Trans. Math. Software* **30,3** 353–376

Gilbert J R, Larimore S I and Ng E G 2004 Algorithm 836: COLAMD, an approximate minimum degree ordering algorithm *ACM Trans. Math. Software* **30, 3** 377–380

5 Parameters

5.1 Compulsory Input Parameters

1: **spec – string**

Indicates the permutation to be applied.

spec = 'N'

The identity permutation is used (i.e., the columns are not permuted).

spec = 'U'

The permutation in the **iprm** array is used, as supplied by you.

spec = 'M'

The permutation computed by the COLAMD algorithm is used

Constraint: **spec** = 'N', 'U' or 'M'.

2: **n – int32 scalar**

n, the order of the matrix A.

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

3: **icolzp**$(*)$ **– int32 array**

Note: the dimension of the array **icolzp** must be at least $\mathbf{n} + 1$.

icolzp(i) contains the index in A of the start of a new column. See Section 2.1.3 in the F11 Chapter Introduction.

4: **irowix**$(*)$ **– int32 array**

Note: the dimension of the array **irowix** must be at least **icolzp**$(\mathbf{n} + 1) - 1$, the number of nonzeros of the sparse matrix A.

irowix(i) contains the row index in A for element $A(i)$. See Section 2.1.3 in the F11 Chapter Introduction.

5: **iprm**$(7 \times \mathbf{n})$ **– int32 array**

The first **n** entries contain the column permutation supplied by you. This will be used if **spec** = 'U', and ignored otherwise. If used, it must consist of a permutation of all the integers in the range $[0, (\mathbf{n} - 1)]$, the leftmost column of the matrix A denoted by 0 and the rightmost by $\mathbf{n} - 1$. Labelling columns in this way, **iprm**$(i) = j$ means that column $i - 1$ of A is in position j in AP_c, where $P_r AP_c = LU$ expresses the factorization to be performed.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **iprm**$(7 \times \mathbf{n})$ **– int32 array**

A new permutation is returned in the first **n** entries. The rest of the array contains data structures that will be used by other functions. The function computes the column elimination tree for A and a post-order permutation on the tree. It then compounds the **iprm** permutation given or computed by the COLAMD algorthm with the post-order permutation. This array is needed by the LU factorization function f11me and associated functions f11mm, f11mf and f11mh and should be passed to them unchanged.

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, **spec** $\neq$ 'N', 'U' or 'M',
or **n** < 0.

ifail $= 2$

On entry, **spec** = 'U', but **iprm** does not represent a valid permutation of the integers in $[0, (\mathbf{n}-1)]$. An input value of **iprm** is either out of range or repeated.

ifail $= 3$

Unspecified failure of the COLAMD algorithm. This should not happen and should be reported to NAG.

ifail $= 4$

On entry, the array **icolzp** failed to satisfy the following constraints:

$\mathbf{icolzp}(1) = 1;$

$\mathbf{icolzp}(i-1) \leq \mathbf{icolzp}(i),$ for $i = 2, 3, \ldots, \mathbf{n}+1;$

$\mathbf{icolzp}(i) \leq \mathbf{n} \times \mathbf{n} + 1,$ for $i = 1, 2, \ldots, \mathbf{n}+1.$

ifail $= 5$

On entry, the array **irowix** failed to satisfy the following constraints:

$1 \leq \mathbf{irowix}(i) \leq \mathbf{n}$ for $i = 1, 2, \ldots, \mathbf{icolzp}(\mathbf{n}+1);$

for each column $i = 1 \ldots \mathbf{n}$, the row indices $\mathbf{irowix}(j)$, where $j = \mathbf{icolzp}(i) \ldots \mathbf{icolzp}(i+1) - 1$, do not repeat.

ifail $= 301$

Unable to allocate required internal workspace.

7 Accuracy

Not applicable. This computation does not use floating-point numbers.

8 Further Comments

We recommend calling this function with **spec** = 'M' before calling f11me. The COLAMD algorithm computes a sparsity-preserving permutation P_c solely from the pattern of A such that the LU factorization $P_r A P_c = LU$ remains as sparse as possible, regardless of the subsequent choice of P_r. The algorithm takes advantage of the existence of super-columns (columns with the same sparsity pattern) to reduce running time.

9 Example

```
spec = 'M';
n = int32(5);
icolzp = [int32(1);
     int32(3);
     int32(5);
     int32(7);
     int32(9);
     int32(12)];
irowix = [int32(1);
     int32(3);
     int32(1);
     int32(5);
     int32(2);
     int32(3);
     int32(2);
     int32(4);
     int32(3);
```

```
     int32(4);
     int32(5)];
iprm = [int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0);
     int32(0)];
[iprmOut, ifail] = f11md(spec, n, icolzp, irowix, iprm)
```

```
iprmOut =
           1
           0
           4
           3
           2
           0
           0
           0
           0
           0
           2
           0
           8
           6
           4
           4
           2
          11
           8
           6
           1
           2
           3
           4
           5
           2
           2
           2
           2
           1
```

```
            1
            1
            1
            2
            0
ifail =
            0
```