Science & Technology
Facilities Council

YM01

PACKAGE SPECIFICATION HSL ARCHIVE

1 SUMMARY

This subroutine **generates** an *m* by *n* **random sparse matrix** with user-specified options such as structural nonsingularity and bandedness. The matrix is held in a packed form in a standard sparse matrix format.

ATTRIBUTES — Version: 1.0.0. **Types:** `YM01A, YM01AD.` **Calls:** `FA04.` **Original date:** February 1987. **Origin:** I.S.Duff, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Argument list

The single precision version

```
CALL YM01A(M,N,NZIN,NZOUT,IBW,ITYPE,NONSIN,IRN,A,JCOLST,IW)
```

The double precision version

```
CALL YM01AD(M,N,NZIN,NZOUT,IBW,ITYPE,NONSIN,IRN,A,JCOLST,IW)
```

`M` is an `INTEGER` variable that must be set by the user to *m* the number of rows required in the matrix. This argument is not altered by the subroutine.

`N` is an `INTEGER` variable that must be set by the user to *n* the number of columns required in the matrix. This argument is not altered by the subroutine.

`NZIN` is an `INTEGER` variable that must be set by the user to the total number of entries required in the matrix. This argument is not altered.

`NZOUT` is an `INTEGER` variable that need not be set by the user. It will be set by the subroutine to the total number of entries in the matrix produced. This will usually have the same value as `NZIN.` It will be different only if `NZIN` is larger than the maximum number of entries that can be accommodated in the requested structure, or if `NZIN` is less than min(*m*,*n*) and structural nonsingularity is requested.

`IBW` is an `INTEGER` variable that must be set by the user to indicate whether matrix entries should be clustered round the diagonal. If `IBW` is negative, no clustering is performed. Otherwise, all entries a_{ij} will be such that $|i-j| \leq$ `IBW`. If the requested number of entries (`NZIN`) exceeds the number of entries in the band, then a full band matrix of semibandwidth `IBW` will be produced.

`ITYPE` is an `INTEGER` variable that must be set by the user to specify the symmetry of the matrix and whether the values of the entries are wanted, according to the following:

1 for an asymmetric structure, with values

–1 for an asymmetric structure, without values

2 for symmetric matrix, with values

–2 for a symmetric matrix, without values

Values of 2 and –2 will produce the lower triangular part only. Values greater than 2 will be treated as +1 and values 0 or less than –2 as –1. This argument is not altered by the subroutine.

`NONSIN` is a `LOGICAL` variable. If it is set `.TRUE.`, then the matrix is guaranteed structurally nonsingular. In the unsymmetric non-band case, there will be a set of min(*m*,*n*) distinct entries no two of which lie in the same row or column. In the band and symmetric cases, all of the diagonal will consist of entries. This argument is not altered by the subroutine.

IRN is an INTEGER array that need not be set by the user. The first NZOUT entries will be set by the subroutine to the row indices of the entries, ordered so that the entries of a single column are contiguous. The entries of column J precede those of column J+1 (J=1, ..., N-1), and there is no wasted space between columns. Row indices are not ordered within columns, except that in the banded or symmetric structurally nonsingular case the diagonal entry comes first. For structurally nonsingular unsymmetric non-banded systems, steps are taken to ensure that the set of distinct entries do not lie in the same relative position in each column.

A is a REAL (DOUBLE PRECISION in the D version) array that need not be set by the user. If ITYPE is positive, A(K) will be set by the subroutine to the value of the entry in position K of IRN, K=1, ..., NZOUT. If ITYPE is negative, the array is not accessed by the subroutine.

JCOLST is an INTEGER array of length n+1 that need not be set by the user. It will be set by the subroutine so that JCOLST(J) is the position in arrays IRN and A of the first entry in column J (J=1, ..., N). JCOLST(N+1) will be equal to NZOUT+1.

IW is an INTEGER array of length m+n that will be used as workspace.

3 GENERAL INFORMATION

Workspace: Array IW is used as workspace (see §2.1).

Use of common: None.

Other routines called directly: Within the package: YM01B/BD. Outside the package: FA04B/BD and FA04A/AD.

Input/output: None.

Restrictions: None.

4 METHOD

The number of entries in each column is calculated as follows. Counts are initialized to zero unless structural nonsingularity has been requested, in which case min(m,n) of the counts are initialized to 1. A random number j from the uniform distribution [1,n] is generated and the count for column j is augmented by one if it is less than the maximum permitted number in the column. To avoid a bias towards shorter columns in the symmetric and banded cases, the acceptance of the randomly generated value is weighted by the length of the column and by the number of entries already in the column. This is repeated until the sum of the column counts equals the required number of entries.

Each column is then generated in turn, by generating row indices from a uniform distribution in the appropriate range. For unsymmetric structures this is [1,m], for banded systems [max(1,j–ibw), min(m,j+ibw)] for column j, and for symmetric structures this is [j,m] for column j.

If structural nonsingularity without band clustering or symmetry is requested, indices corresponding to a random selection of min(m,n) integers from [1,m] are placed in min(m,n) columns selected at random from [1,n]. If structural nonsingularity with band clustering or symmetry is requested, the diagonal is included in the structure..

In all cases where values are requested, these are generated in a final pass from a uniform distribution in the range [−1,1].

5 EXAMPLE OF USE

Two 20×20 random matrices may be generated by the following code

```
      PARAMETER (MAXN=20,MAXNZ=160)
      INTEGER IRN(MAXNZ),IW(MAXN,2)
      INTEGER JCOLST(MAXN+1)
      LOGICAL NONSIN
      REAL A(1)

      NONSIN = .TRUE.
      DO 50 KASE = 1,2
        READ (5,FMT=100) M,N,NZIN,IBW,ITYPE
        WRITE (6,FMT=110) M,N

  100   FORMAT (5I4)
  110   FORMAT (/'  The ',I4,' by ',I4,' matrix pattern')

        CALL YM01A(M,N,NZIN,NZOUT,IBW,ITYPE,NONSIN,IRN,A,JCOLST,IW)
        CALL PRMAT(M,N,NZOUT,IRN,JCOLST)

   50 CONTINUE
      STOP

      END
      SUBROUTINE PRMAT(M,N,NZ,IRN,JCOLST)
      CHARACTER MAT(40000)
      INTEGER  IRN(NZ)
      INTEGER JCOLST(N)
      MN=M*N
      DO 17 J=1,MN
        MAT(J)='.'
  17  CONTINUE
      DO 6 J=1,N
        K=NZ
        IF (J .LT. N) K=JCOLST(J+1)-1
        DO 7 I=JCOLST(J),K
          IROW=IRN(I)
          III=(IROW-1)*N+J
          MAT(III)='X'
  7     CONTINUE
  6   CONTINUE
      WRITE (6,6001)
 6001 FORMAT(' ')
      DO 20 I=1,MN,N
        WRITE(6,6000)(MAT(J),J=I,I+N-1)
 6000   FORMAT(' ',60A2)
  20  CONTINUE
      RETURN
      END
```

run on the data

```
     20  20  60   5  -2
     10  20  60  -1  -1
```

This would generate two random matrices such as these

```
x . . . . . . . . . . . . . . . . . . .
. x . . . . . . . . . . . . . . . . . .
. x x . . . . . . . . . . . . . . . . .
. . . x . . . . . . . . . . . . . . . .
x . . . x . . . . . . . . . . . . . . .
x . . x . x . . . . . . . . . . . . . .
. x . x x x x . . . . . . . . . . . . .
. . . . x x . x . . . . . . . . . . . .
. . . . x x x . x . . . . . . . . . . .
. . . . . x x . . x . . . . . . . . . .
. . . . . x x . . . x . . . . . . . . .
. . . . . . x . . . x x . . . . . . . .
. . . . . . . . . x x x x . . . . . . .
. . . . . . . . x x x x . x . . . . . .
. . . . . . . . . x x x . . x . . . . .
. . . . . . . . . . x . . x . x . . . .
. . . . . . . . . . . x . . x . x . . .
. . . . . . . . . . . . x x . . x x . .
. . . . . . . . . . . . . . . . x x x .
. . . . . . . . . . . . . . . . x . x x
```

and

```
. . . . x x . . x . . . x x . x x . . x
. . x x . x . . . . . x . . . . . . x .
x . . . x . . x . . . . . x . x x . . x
x . x . x . . . . . . . . x . x . . . .
. x x x x x . . x . . . x . . . . . . x
. x . . x x . . . . . . x x . . x . . .
. x . x x . . x . . . . . x . x . . . .
. x . x . x . x . . . . x . x . x . . .
. . . . x . x . . . . . . . . x . . . .
. x . x x . . x . x . . . . . . . . . .
```

For the purposes of clarity, entries are represented by an “x” and positions not in the structure by a “.”.