F11XNF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

Computes a matrix-vector or conjugate transposed matrix-vector product involving a complex sparse non-Hermitian matrix stored in coordinate storage format.

2 Specification

```
SUBROUTINE F11XNF(TRANS, N, NNZ, A, IROW, ICOL, CHECK, X, Y, IFAIL)
INTEGER            N, NNZ, IROW(NNZ), ICOL(NNZ), IFAIL
complex            A(NNZ), X(N), Y(N)
CHARACTER*1        TRANS, CHECK
```

3 Description

F11XNF computes either the matrix-vector product $y = Ax$, or the conjugate transposed matrix-vector product $y = A^H x$, according to the value of the argument TRANS, where A is a complex n by n sparse non-Hermitian matrix, of arbitrary sparsity pattern. The matrix A is stored in coordinate storage (CS) format (see Section 2.1.1 of the Chapter Introduction). The array A stores all the non-zero elements of A, while arrays IROW and ICOL store the corresponding row and column indices respectively.

It is envisaged that a common use of F11XNF will be to compute the matrix-vector product required in the application of F11BSF to sparse complex linear systems. This is illustrated in Section 9 of the document for F11DRF.

4 References

None.

5 Parameters

1: TRANS — CHARACTER*1 *Input*

On entry: specifies whether or not the matrix A is conjugate transposed:

if TRANS = 'N', then $y = Ax$ is computed;
if TRANS = 'T', then $y = A^H x$ is computed.

Constraint: TRANS = 'N' or 'T'.

2: N — INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: N ≥ 1.

3: NNZ — INTEGER *Input*

On entry: the number of non-zero elements in the matrix A.

Constraint: $1 \leq$ NNZ $\leq$ N^2.

4: A(NNZ) — ***complex*** array *Input*

On entry: the non-zero 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 routine F11ZNF may be used to order the elements in this way.

5: IROW(NNZ) — INTEGER array *Input*

6: ICOL(NNZ) — INTEGER array *Input*

On entry: the row and column indices of the non-zero elements supplied in A.

Constraint: IROW and ICOL must satisfy the following constraints (which may be imposed by a call to F11ZNF):

$1 \leq \text{IROW}(i) \leq \text{N}$ and $1 \leq \text{ICOL}(i) \leq \text{N}$, for $i = 1, 2, \ldots,$NNZ.
IROW($i - 1$) < IROW(i), or
IROW($i - 1$) = IROW(i) and ICOL($i - 1$) < ICOL(i), for $i = 2, 3, \ldots,$NNZ.

7: CHECK — CHARACTER*1 *Input*

On entry: specifies whether or not the CS representation of the matrix A should be checked:

if CHECK = 'C', checks are carried on the values of N, NNZ, IROW and ICOL;
if CHECK = 'N', none of these checks are carried out.

See also Section 8.2.

Constraint: CHECK = 'C' or 'N'.

8: X(N) — ***complex*** array *Input*

On entry: the vector x.

9: Y(N) — ***complex*** array *Output*

On exit: the vector y.

10: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, TRANS ≠ 'N' or 'T',
or CHECK ≠ 'C' or 'N'.

IFAIL = 2

On entry, N < 1,
or NNZ < 1,
or NNZ > N^2.

IFAIL = 3

On entry, the arrays IROW and ICOL fail to satisfy the following constraints:

$1 \leq \text{IROW}(i) \leq \text{N}$ and $1 \leq \text{ICOL}(i) \leq \text{N}$, for $i = 1, 2, \ldots,$NNZ.
IROW($i - 1$) < IROW(i), or
IROW($i - 1$) = IROW(i) and ICOL($i - 1$) < ICOL(i), for $i = 2, 3, \ldots,$NNZ.

Therefore a non-zero element has been supplied which does not lie within the matrix A, is out of order, or has duplicate row and column indices. Call F11ZNF to reorder and sum or remove duplicates.

7 Accuracy

The computed vector y satisfies the error bound:

$\|y - Ax\|_\infty \le c(n)\epsilon\|A\|_\infty\|x\|_\infty$, if TRANS = ’N’, or
$\|y - A^Tx\|_\infty \le c(n)\epsilon\|A^T\|_\infty\|x\|_\infty$, if TRANS = ’T’,

where $c(n)$ 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 F11XNF is proportional to NNZ.

8.2 Use of CHECK

It is expected that a common use of F11XNF will be to compute the matrix-vector product required in the application of F11BSF to sparse complex linear systems. In this situation F11XNF is likely to be called many times with the same matrix A. 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

This example program reads in a complex sparse matrix A and a vector x. It then calls F11XNF to compute the matrix-vector product $y = Ax$ and the conjugate transposed matrix-vector product $y = A^Hx$.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users’ Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F11XNF Example Program Text.
*     Mark 19 Release. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          LA, NMAX
      PARAMETER        (LA=10000,NMAX=1000)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, N, NNZ
      CHARACTER        CHECK, TRANS
*     .. Local Arrays ..
      complex          A(LA), X(NMAX), Y(NMAX)
      INTEGER          ICOL(LA), IROW(LA)
*     .. External Subroutines ..
      EXTERNAL         F11XNF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F11XNF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*
*     Read order of matrix and number of non-zero entries
*
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
         READ (NIN,*) NNZ
*
```

```
*        Read the matrix A
*
         DO 20 I = 1, NNZ
            READ (NIN,*) A(I), IROW(I), ICOL(I)
   20    CONTINUE
*
*        Read the vector x
*
         READ (NIN,*) (X(I),I=1,N)
*
*        Calculate matrix-vector product
*
         TRANS = 'Not transposed'
         CHECK = 'C'
         IFAIL = 0
         CALL F11XNF(TRANS,N,NNZ,A,IROW,ICOL,CHECK,X,Y,IFAIL)
*
*        Output results
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Matrix-vector product'
         DO 40 I = 1, N
            WRITE (NOUT,'(1X,''('',D16.4,'','',D16.4,'')'')') Y(I)
   40    CONTINUE
*
*        Calculate conjugate transposed matrix-vector product
*
         TRANS = 'Transposed'
         CHECK = 'N'
         IFAIL = 0
         CALL F11XNF(TRANS,N,NNZ,A,IROW,ICOL,CHECK,X,Y,IFAIL)
*
*        Output results
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Conjugate transposed matrix-vector product'
         DO 60 I = 1, N
            WRITE (NOUT,'(1X,''('',D16.4,'','',D16.4,'')'')') Y(I)
   60    CONTINUE
*
      END IF
      STOP
      END
```

9.2 Program Data

```
F11XNF Example Program Data
  5                     N
 11                     NNZ
  ( 2., 3.)   1    1
  ( 1.,-4.)   1    2
  ( 1., 0.)   2    3
  (-1.,-2.)   2    4
  ( 4., 1.)   3    1
  ( 0., 1.)   3    3
  ( 1., 3.)   3    5
  ( 0.,-1.)   4    4
  ( 2.,-6.)   4    5
  (-2., 0.)   5    2
  ( 3., 1.)   5    5     A(I), IROW(I), ICOL(I), I=1,...,NNZ
 ( 0.70, 0.21)
 ( 0.16,-0.43)
 ( 0.52, 0.97)
 ( 0.77, 0.00)
 ( 0.28,-0.64)          X(I), I=1,...,N
```

9.3 Program Results

```
F11XNF Example Program Results

 Matrix-vector product
(     -0.7900E+00,      0.1450E+01)
(     -0.2500E+00,     -0.5700E+00)
(      0.3820E+01,      0.2260E+01)
(     -0.3280E+01,     -0.3730E+01)
(      0.1160E+01,     -0.7800E+00)

 Conjugate transposed matrix-vector product
(      0.5080E+01,      0.1680E+01)
(     -0.7000E+00,      0.4290E+01)
(      0.1130E+01,     -0.9500E+00)
(      0.7000E+00,      0.1520E+01)
(      0.5170E+01,      0.1830E+01)
```