

F11ZAF – 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

Sorts the non-zero elements of a real sparse nonsymmetric matrix, represented in coordinate storage format.

2 Specification

```

SUBROUTINE F11ZAF(N, NNZ, A, IROW, ICOL, DUP, ZERO, ISTR, IWORK,
1              IFAIL)
  INTEGER      N, NNZ, IROW(*), ICOL(*), ISTR(N+1), IWORK(N),
1              IFAIL
  real        A(*)
  CHARACTER*1  DUP, ZERO

```

3 Description

F11ZAF takes a coordinate storage (CS) representation (see Section 2.1.1 of the Chapter Introduction) of a real n by n sparse nonsymmetric matrix A , and reorders the non-zero elements by increasing row index and increasing column index within each row. Entries with duplicate row and column indices may be removed, or the values may be summed. Any entries with zero values may optionally be removed.

F11ZAF also returns a pointer ISTR to the starting address of each row in A .

4 References

None.

5 Parameters

- 1:** N — INTEGER *Input*
On entry: n , the order of the matrix A .
Constraint: $N \geq 1$.
- 2:** NNZ — INTEGER *Input/Output*
On entry: the number of non-zero elements in the matrix A .
Constraint: $NNZ \geq 0$.
On exit: the number of non-zero elements with unique row and column indices.
- 3:** A(*) — *real* array *Input/Output*
Note: the dimension of the array A must be at least $\max(1, NNZ)$.
On entry: the non-zero elements of the matrix A . These may be in any order and there may be multiple non-zero elements with the same row and column indices.
On exit: the non-zero elements ordered by increasing row index, and by increasing column index within each row. Each non-zero element has a unique row and column index.

- 4:** IROW(*) — INTEGER array *Input/Output*
Note: the dimension of the array IROW must be at least $\max(1, \text{NNZ})$.
On entry: the row indices corresponding to the non-zero elements supplied in the array A.
Constraint: $1 \leq \text{IROW}(i) \leq N$, for $i = 1, 2, \dots, \text{NNZ}$.
On exit: the first NNZ elements contain the row indices corresponding to the non-zero elements returned in the array A.
- 5:** ICOL(*) — INTEGER array *Input/Output*
Note: the dimension of the array ICOL must be at least $\max(1, \text{NNZ})$.
On entry: the column indices corresponding to the non-zero elements supplied in the array A.
Constraint: $1 \leq \text{ICOL}(i) \leq N$, for $i = 1, 2, \dots, \text{NNZ}$.
On exit: the first NNZ elements contain the row indices corresponding to the non-zero elements returned in the array A.
- 6:** DUP — CHARACTER*1 *Input*
On entry: indicates how any non-zero elements with duplicate row and column indices are to be treated:
 if DUP = 'R' the entries are removed;
 if DUP = 'S' the relevant values in A are summed;
 if DUP = 'F' the routine fails on detecting a duplicate, with IFAIL = 3.
Constraint: DUP = 'R', 'S', or 'F'.
- 7:** ZERO — CHARACTER*1 *Input*
On entry: indicates how any elements with zero values in A are to be treated:
 if ZERO = 'R' the entries are removed;
 if ZERO = 'K' the entries are kept;
 if ZERO = 'F' the routine fails on detecting a zero, with IFAIL = 4.
Constraint: ZERO = 'R', 'K', or 'F'.
- 8:** ISTR(N+1) — INTEGER array *Output*
On exit: ISTR(i), for $i = 1, 2, \dots, N$, contains the index of arrays A, IROW and ICOL where row i of the matrix A starts. ISTR(N+1) contains the index + 1 of the last non-zero element in A .
- 9:** IWORK(N) — INTEGER array *Workspace*
- 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, `N < 1`,
 or `NNZ < 0`,
 or `DUP ≠ 'R', 'S', or 'F'`,
 or `ZERO ≠ 'R', 'K', or 'F'`.

`IFAIL = 2`

On entry, a non-zero element has been supplied which does not lie within the matrix A , i.e., one or more of the following constraints has been violated:

$$1 \leq \text{IROW}(i) \leq N,$$

$$1 \leq \text{ICOL}(i) \leq N,$$

for $i = 1, 2, \dots, \text{NNZ}$.

`IFAIL = 3`

On entry, `DUP = 'F'`, and non-zero elements have been supplied which have duplicate row and column indices.

`IFAIL = 4`

On entry, `ZERO = 'F'`, and at least one matrix element has been supplied with a zero coefficient value.

7 Accuracy

Not applicable.

8 Further Comments

The time taken for a call to `F11ZAF` is proportional to `NNZ`.

Note that the resulting matrix may have either rows or columns with no entries. If row i has no entries then `ISTR(i) = ISTR(i + 1)`.

9 Example

This example program reads the CS representation of a real sparse matrix A , calls `F11ZAF` to reorder the non-zero elements, and outputs the original and the reordered representations.

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.

```
*      F11ZAF Example Program Text
*      Mark 19 Revised. 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      DUP, ZERO
*
* .. Local Arrays ..
real          A(LA)
INTEGER        ICOL(LA), IROW(LA), ISTR(NMAX+1), IWORK(NMAX)
*
* .. External Subroutines ..
EXTERNAL       F11ZAF
*
* .. Executable Statements ..
WRITE (NOUT,*) 'F11ZAF Example Program Results'
WRITE (NOUT,*)
*
* 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 and output the original non-zero elements
*
*   DO 20 I = 1, NNZ
*     READ (NIN,*) A(I), IROW(I), ICOL(I)
20  CONTINUE
*   WRITE (NOUT,*) 'Original elements'
*   WRITE (NOUT,'(A,I4)') ' NNZ = ', NNZ
*   DO 40 I = 1, NNZ
*     WRITE (NOUT,'(I8,D16.4,2I8)') I, A(I), IROW(I), ICOL(I)
40  CONTINUE
*
*   Reorder, sum duplicates and remove zeros
*
*   DUP = 'S'
*   ZERO = 'R'
*   IFAIL = 0
*
*   CALL F11ZAF(N,NNZ,A,IROW,ICOL,DUP,ZERO,ISTR,IWORK,IFAIL)
*
*   Output results
*
*   WRITE (NOUT,*) 'Reordered elements'
*   WRITE (NOUT,'(A,I4)') ' NNZ = ', NNZ
*   DO 60 I = 1, NNZ
*     WRITE (NOUT,'(I8,D16.4,2I8)') I, A(I), IROW(I), ICOL(I)
60  CONTINUE
*
* END IF
* STOP
* END

```

9.2 Program Data

F11ZAF Example Program Data

5			N
15			NNZ
4.	3	1	
-2.	5	2	
1.	4	4	
-2	4	2	
-3	5	5	
1.	1	2	
0.	1	5	
1.	3	5	
-1.	2	4	
6.	5	5	
2.	1	1	
2.	4	2	
1.	2	3	
1.	3	3	
2.	4	5	A(I), IROW(I), ICOL(I), I=1,...,NNZ

9.3 Program Results

F11ZAF Example Program Results

Original elements

NNZ =	15		
1	0.4000E+01	3	1
2	-0.2000E+01	5	2
3	0.1000E+01	4	4
4	-0.2000E+01	4	2
5	-0.3000E+01	5	5
6	0.1000E+01	1	2
7	0.0000E+00	1	5
8	0.1000E+01	3	5
9	-0.1000E+01	2	4
10	0.6000E+01	5	5
11	0.2000E+01	1	1
12	0.2000E+01	4	2
13	0.1000E+01	2	3
14	0.1000E+01	3	3
15	0.2000E+01	4	5

Reordered elements

NNZ =	11		
1	0.2000E+01	1	1
2	0.1000E+01	1	2
3	0.1000E+01	2	3
4	-0.1000E+01	2	4
5	0.4000E+01	3	1
6	0.1000E+01	3	3
7	0.1000E+01	3	5
8	0.1000E+01	4	4
9	0.2000E+01	4	5
10	-0.2000E+01	5	2
11	0.3000E+01	5	5