NAG Toolbox

nag_sparse_complex_herm_matvec (f11xs)

1 Purpose

nag_sparse_complex_herm_matvec (f11xs) computes a matrix-vector product involving a complex sparse Hermitian matrix stored in symmetric coordinate storage format.

2 Syntax

```
[y, ifail] = nag_sparse_complex_herm_matvec(a, irow, icol, check, x, 'n', n, 'nnz', nnz)
[y, ifail] = f11xs(a, irow, icol, check, x, 'n', n, 'nnz', nnz)
```

3 Description

nag_sparse_complex_herm_matvec (f11xs) computes the matrix-vector product

$$y = Ax$$

where A is an n by n complex Hermitian sparse matrix, of arbitrary sparsity pattern, stored in symmetric coordinate storage (SCS) format (see Section 2.1.2 in the F11 Chapter Introduction). The array **a** stores all the nonzero elements in the lower triangular part of A, while arrays **irow** and **icol** store the corresponding row and column indices respectively.

4 References

None.

5 Parameters

5.1 Compulsory Input Parameters

1: **a**(**nnz**) – COMPLEX (KIND=nag_wp) array

The nonzero elements in the lower triangular part of 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 nag_sparse_complex_herm_sort (f11zp) may be used to order the elements in this way.

2: **irow**(**nnz**) – INTEGER array

3: **icol**(**nnz**) – INTEGER array

The row and column indices of the nonzero elements supplied in array **a**.

Constraints:

irow and **icol** must satisfy the following constraints (which may be imposed by a call to nag_sparse_complex_herm_sort (f11zp)):

$1 \leq \mathbf{irow}(i) \leq \mathbf{n}$ and $1 \leq \mathbf{icol}(i) \leq \mathbf{irow}(i)$, 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 $i = 2, 3, \ldots, \mathbf{nnz}$.

4: **check** – CHARACTER(1)

Specifies whether or not the SCS representation of the matrix A, values of **n**, **nnz**, **irow** and **icol** should be checked.

check = 'C'
Checks are carried out on the values of **n**, **nnz**, **irow** and **icol**.

check = 'N'
None of these checks are carried out.

Constraint: **check** = 'C' or 'N'.

5: **x**(**n**) – COMPLEX (KIND=nag_wp) array

The vector x.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the dimension of the array **x**.

n, the order of the matrix A.

Constraint: **n** ≥ 1.

2: **nz** – INTEGER

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 lower triangular part of the matrix A.

Constraint: $1 \leq$ **nnz** $\leq$ **n** $\times$ (**n** $+ 1)/2$.

5.3 Output Parameters

1: **y**(**n**) – COMPLEX (KIND=nag_wp) array

The vector y.

2: **ifail** – INTEGER

ifail = 0 unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail = 1

On entry, **check** $\neq$ 'C' or 'N'.

ifail = 2

On entry, **n** < 1,
or **nnz** < 1,
or **nnz** $>$ **n** $\times$ (**n** $+ 1)/2$.

ifail = 3

On entry, the arrays **irow** and **icol** fail to satisfy the following constraints:

$1 \leq \mathbf{irow}(i) \leq \mathbf{n}$ and $1 \leq \mathbf{icol}(i) \leq \mathbf{irow}(i)$, 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 lower triangular part of A, is out of order, or has duplicate row and column indices. Call nag_sparse_complex_herm_sort (f11zp) to reorder and sum or remove duplicates.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The computed vector y satisfies the error bound

$$\|y - Ax\|_\infty \leq c(n)\epsilon\|A\|_\infty\|x\|_\infty,$$

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 nag_sparse_complex_herm_matvec (f11xs) is proportional to **nnz**.

9 Example

This example reads in a complex sparse Hermitian positive definite matrix A and a vector x. It then calls nag_sparse_complex_herm_matvec (f11xs) to compute the matrix-vector product $y = Ax$.

9.1 Program Text

```
function f11xs_example

fprintf('f11xs example results\n\n');

% Sparse Hermitian matrix product y = Ax
a = [6 ...
    -1 + 1i  6 + 0i ...
             0 + 1i 5 + 0i  ...
                            5 + 0i ...
     2 - 2i                        4 + 0i  ...
                    1 + 1i  2 + 0i         6 + 0i ...
            -4 + 3i                 0 + 1i -1 + 0i 6 + 0i ...
                           -1 - 1i         0 - 1i        9 + 0i ...
     1 + 3i                        1 + 2i -1 + 0i        1 + 4i 9 + 0i];
irow = [nag_int(1)  2 2  3 3  4  5 5  6 6 6  7 7 7 7  8 8 8  9 9 9 9 9];
icol = [nag_int(1)  1 2  2 3  4  1 5  3 4 6  2 5 6 7  4 6 8  1 5 6 8 9];

x = [ 1 + 9i;      2 - 8i;      3 + 7i;
      4 - 6i;      5 + 5i;      6 - 4i;
      7 + 3i;      8 - 2i;      9 + 1i];

check = 'C';
```

```
[y, ifail] = f11xs( ...
                    a, irow, icol, check, x);

disp('Matrix-vector product');
disp(y);
```

9.2 Program Results

```
     f11xs example results

Matrix-vector product
   8.0000 +54.0000i
 -10.0000 -92.0000i
  25.0000 +27.0000i
  26.0000 -28.0000i
  54.0000 +12.0000i
  26.0000 -22.0000i
  47.0000 +65.0000i
  71.0000 -57.0000i
  60.0000 +70.0000i
```