NAG Toolbox

nag_lapack_zsptrf (f07qr)

1 Purpose

nag_lapack_zsptrf (f07qr) computes the Bunch–Kaufman factorization of a complex symmetric matrix, using packed storage.

2 Syntax

```
[ap, ipiv, info] = nag_lapack_zsptrf(uplo, n, ap)
[ap, ipiv, info] = f07qr(uplo, n, ap)
```

3 Description

nag_lapack_zsptrf (f07qr) factorizes a complex symmetric matrix A, using the Bunch–Kaufman diagonal pivoting method and packed storage. A is factorized as either $A = PUDU^{\mathrm{T}}P^{\mathrm{T}}$ if **uplo** $=$ 'U' or $A = PLDL^{\mathrm{T}}P^{\mathrm{T}}$ if **uplo** $=$ 'L', where P is a permutation matrix, U (or L) is a unit upper (or lower) triangular matrix and D is a symmetric block diagonal matrix with 1 by 1 and 2 by 2 diagonal blocks; U (or L) has 2 by 2 unit diagonal blocks corresponding to the 2 by 2 blocks of D. Row and column interchanges are performed to ensure numerical stability while preserving symmetry.

4 References

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

5.1 Compulsory Input Parameters

1: **uplo** – CHARACTER(1)

Specifies whether the upper or lower triangular part of A is stored and how A is to be factorized.

uplo $=$ 'U'
: The upper triangular part of A is stored and A is factorized as $PUDU^{\mathrm{T}}P^{\mathrm{T}}$, where U is upper triangular.

uplo $=$ 'L'
: The lower triangular part of A is stored and A is factorized as $PLDL^{\mathrm{T}}P^{\mathrm{T}}$, where L is lower triangular.

Constraint: **uplo** $=$ 'U' or 'L'.

2: **n** – INTEGER

n, the order of the matrix A.

Constraint: **n** ≥ 0.

3: **ap**(:) – COMPLEX (KIND=nag_wp) array

The dimension of the array **ap** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$

The n by n symmetric matrix A, packed by columns.

More precisely,

if **uplo** = 'U', the upper triangle of A must be stored with element A_{ij} in **ap**$(i + j(j-1)/2)$ for $i \le j$;

if **uplo** = 'L', the lower triangle of A must be stored with element A_{ij} in **ap**$(i + (2n - j)(j-1)/2)$ for $i \ge j$.

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **ap**(:) – COMPLEX (KIND=nag_wp) array

The dimension of the array **ap** will be $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$

A stores details of the block diagonal matrix D and the multipliers used to obtain the factor U or L as specified by **uplo**.

2: **ipiv**(**n**) – INTEGER array

Details of the interchanges and the block structure of D. More precisely,

if **ipiv**$(i) = k > 0$, d_{ii} is a 1 by 1 pivot block and the ith row and column of A were interchanged with the kth row and column;

if **uplo** = 'U' and **ipiv**$(i-1)$ = **ipiv**$(i) = -l < 0$, $\begin{pmatrix} d_{i-1,i-1} & \bar{d}_{i,i-1} \\ \bar{d}_{i,i-1} & d_{ii} \end{pmatrix}$ is a 2 by 2 pivot block and the $(i-1)$th row and column of A were interchanged with the lth row and column;

if **uplo** = 'L' and **ipiv**(i) = **ipiv**$(i+1) = -m < 0$, $\begin{pmatrix} d_{ii} & d_{i+1,i} \\ d_{i+1,i} & d_{i+1,i+1} \end{pmatrix}$ is a 2 by 2 pivot block and the $(i+1)$th row and column of A were interchanged with the mth row and column.

3: **info** – INTEGER

info $= 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

info < 0

If **info** $= -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

info > 0 (*warning*)

Element ⟨*value*⟩ of the diagonal is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, and division by zero will occur if it is used to solve a system of equations.

7 Accuracy

If **uplo** = 'U', the computed factors U and D are the exact factors of a perturbed matrix $A + E$, where

$$|E| \le c(n)\epsilon P|U||D|\left|U^{\mathrm{T}}\right|P^{\mathrm{T}},$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***.

If **uplo** = 'L', a similar statement holds for the computed factors L and D.

8 Further Comments

The elements of D overwrite the corresponding elements of A; if D has 2 by 2 blocks, only the upper or lower triangle is stored, as specified by **uplo**.

The unit diagonal elements of U or L and the 2 by 2 unit diagonal blocks are not stored. The remaining elements of U or L overwrite elements in the corresponding columns of A, but additional row interchanges must be applied to recover U or L explicitly (this is seldom necessary). If **ipiv**$(i) = i$, for $i = 1, 2, \ldots, n$, then U or L are stored explicitly in packed form (except for their unit diagonal elements which are equal to 1).

The total number of real floating-point operations is approximately $\frac{4}{3}n^3$.

A call to nag_lapack_zsptrf (f07qr) may be followed by calls to the functions:

nag_lapack_zsptrs (f07qs) to solve $AX = B$;

nag_lapack_zspcon (f07qu) to estimate the condition number of A;

nag_lapack_zsptri (f07qw) to compute the inverse of A.

The real analogue of this function is nag_lapack_dsptrf (f07pd).

9 Example

This example computes the Bunch–Kaufman factorization of the matrix A, where

$$A = \begin{pmatrix} -0.39 - 0.71i & 5.14 - 0.64i & -7.86 - 2.96i & 3.80 + 0.92i \\ 5.14 - 0.64i & 8.86 + 1.81i & -3.52 + 0.58i & 5.32 - 1.59i \\ -7.86 - 2.96i & -3.52 + 0.58i & -2.83 - 0.03i & -1.54 - 2.86i \\ 3.80 + 0.92i & 5.32 - 1.59i & -1.54 - 2.86i & -0.56 + 0.12i \end{pmatrix},$$

using packed storage.

9.1 Program Text

```
function f07qr_example

fprintf('f07qr example results\n\n');

% complex symmetric matrix, upper triangle stored in packed format
uplo = 'U';
n    = nag_int(4);
ap = [-0.56 + 0.12i;
      -1.54 - 2.86i; -2.83 - 0.03i;
       5.32 - 1.59i; -3.52 + 0.58i;  8.86 + 1.81i;
       3.80 + 0.92i; -7.86 - 2.96i;  5.14 - 0.64i; -0.39 - 0.71i];

[apf, ipiv, info] = f07qr( ...
                           uplo, n, ap);

[ifail] = x04dc( ...
                 uplo, 'Non-unit', n, apf, 'Details of factorization');

fprintf('\nPivot indices\n    ');
fprintf('%11d', ipiv);
fprintf('\n');
```

9.2 Program Results

```
f07qr example results

 Details of factorization
             1          2          3          4
 1     -2.0954    -0.1071    -0.4823     0.4426
       -2.2011    -0.3157     0.0150     0.1936

 2                 4.4079    -0.6078     0.5279
```

```
                      5.3991      0.2811     -0.3715

 3                               -2.8300     -7.8600
                                 -0.0300     -2.9600

 4                                           -0.3900
                                             -0.7100

Pivot indices
              1           2          -2          -2
```