NAG Toolbox

nag_lapack_zhpsv (f07pn)

1 Purpose

nag_lapack_zhpsv (f07pn) computes the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n Hermitian matrix stored in packed format and X and B are n by r matrices.

2 Syntax

```
[ap, ipiv, b, info] = nag_lapack_zhpsv(uplo, ap, b, 'n', n, 'nrhs_p', nrhs_p)
[ap, ipiv, b, info] = f07pn(uplo, ap, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

nag_lapack_zhpsv (f07pn) uses the diagonal pivoting method to factor A as $A = UDU^{\mathrm{H}}$ if **uplo** = 'U' or $A = LDL^{\mathrm{H}}$ if **uplo** = 'L', where U (or L) is a product of permutation and unit upper (lower) triangular matrices, D is Hermitian and block diagonal with 1 by 1 and 2 by 2 diagonal blocks. The factored form of A is then used to solve the system of equations $AX = B$.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

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

Higham N J (2002) *Accuracy and Stability of Numerical Algorithms* (2nd Edition) SIAM, Philadelphia

5 Parameters

5.1 Compulsory Input Parameters

1: **uplo** – CHARACTER(1)

If **uplo** = 'U', the upper triangle of A is stored.

If **uplo** = 'L', the lower triangle of A is stored.

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

2: **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 Hermitian matrix A, packed by columns.

More precisely,

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

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

3: **b**$(ldb,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **b** must be at least $\max(1,\mathbf{n})$.

The second dimension of the array **b** must be at least $\max(1,\mathbf{nrhs\_p})$.

Note: to solve the equations $Ax = b$, where b is a single right-hand side, **b** may be supplied as a one-dimensional array with length $ldb = \max(1,\mathbf{n})$.

The n by r right-hand side matrix B.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the first dimension of the array **b**.

n, the number of linear equations, i.e., the order of the matrix A.

Constraint: $\mathbf{n} \geq 0$.

2: **nrhs_p** – INTEGER

Default: the second dimension of the array **b**.

r, the number of right-hand sides, i.e., the number of columns of the matrix B.

Constraint: $\mathbf{nrhs\_p} \geq 0$.

5.3 Output Parameters

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

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

The block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = UDU^{\mathrm{H}}$ or $A = LDL^{\mathrm{H}}$ as computed by nag_lapack_zhptrf (f07pr), stored as a packed triangular matrix in the same storage format as A.

2: **ipiv**$(\mathbf{n})$ – INTEGER array

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

if $\mathbf{ipiv}(i) = k > 0$, 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 $\mathbf{uplo} = \text{'U'}$ and $\mathbf{ipiv}(i-1) = \mathbf{ipiv}(i) = -l < 0$, $\begin{pmatrix} d_{i-1,i-1} & \bar{d}_{i,i-1} \\ \bar{d}_{i,i-1} & d_{ii} \end{pmatrix}$ 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 $\mathbf{uplo} = \text{'L'}$ and $\mathbf{ipiv}(i) = \mathbf{ipiv}(i+1) = -m < 0$, $\begin{pmatrix} d_{ii} & d_{i+1,i} \\ d_{i+1,i} & d_{i+1,i+1} \end{pmatrix}$ 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: **b**$(ldb,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **b** will be $\max(1,\mathbf{n})$.

The second dimension of the array **b** will be $\max(1,\mathbf{nrhs\_p})$.

Note: to solve the equations $Ax = b$, where b is a single right-hand side, **b** may be supplied as a one-dimensional array with length $ldb = \max(1,\mathbf{n})$.

If $\mathbf{info} = 0$, the n by r solution matrix X.

4: **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 $\langle value \rangle$ of the diagonal is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution could not be computed.

7 Accuracy

The computed solution for a single right-hand side, $\hat{x}$, satisfies an equation of the form

$$(A + E)\hat{x} = b,$$

where

$$\|E\|_1 = O(\epsilon)\|A\|_1$$

and ϵ is the ***machine precision***. An approximate error bound for the computed solution is given by

$$\frac{\|\hat{x} - x\|_1}{\|x\|_1} \leq \kappa(A)\frac{\|E\|_1}{\|A\|_1},$$

where $\kappa(A) = \left\|A^{-1}\right\|_1 \|A\|_1$, the condition number of A with respect to the solution of the linear equations. See Section 4.4 of Anderson *et al.* (1999) and Chapter 11 of Higham (2002) for further details.

nag_lapack_zhpsvx (f07pp) is a comprehensive LAPACK driver that returns forward and backward error bounds and an estimate of the condition number. Alternatively, nag_linsys_complex_herm_packed_solve (f04cj) solves $AX = B$ and returns a forward error bound and condition estimate. nag_linsys_complex_herm_packed_solve (f04cj) calls nag_lapack_zhpsv (f07pn) to solve the equations.

8 Further Comments

The total number of floating-point operations is approximately $\frac{4}{3}n^3 + 8n^2r$, where r is the number of right-hand sides.

The real analogue of this function is nag_lapack_dspsv (f07pa). The complex symmetric analogue of this function is nag_lapack_zspsv (f07qn).

9 Example

This example solves the equations

$$Ax = b,$$

where A is the Hermitian matrix

$$A = \begin{pmatrix} -1.84 & 0.11 - 0.11i & -1.78 - 1.18i & 3.91 - 1.50i \\ 0.11 + 0.11i & -4.63 & -1.84 + 0.03i & 2.21 + 0.21i \\ -1.78 + 1.18i & -1.84 - 0.03i & -8.87 & 1.58 - 0.90i \\ 3.91 + 1.50i & 2.21 - 0.21i & 1.58 + 0.90i & -1.36 \end{pmatrix}$$

and

$$b = \begin{pmatrix} 2.98 - 10.18i \\ -9.58 + 3.88i \\ -0.77 - 16.05i \\ 7.79 + 5.48i \end{pmatrix}.$$

Details of the factorization of A are also output.

9.1 Program Text

```
     function f07pn_example

fprintf('f07pn example results\n\n');

% Hermitian indefinite matrix A (Upper triangular part stored in packed format)
uplo = 'Upper';
n  = nag_int(4);
ap = [-1.84;
       0.11 - 0.11i;  -4.63 + 0i;
      -1.78 - 1.18i;  -1.84 + 0.03i;  -8.87 + 0i;
       3.91 - 1.50i;   2.21 + 0.21i;   1.58 - 0.90i;  -1.36 + 0i];

% RHS
b = [ 2.98 - 10.18i;
     -9.58 +  3.88i;
     -0.77 - 16.05i;
      7.79 +  5.48i];

% Solve
[apf, ipiv, x, info] = f07pn( ...
                              uplo, ap, b);

disp('Solution');
disp(x);

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

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

9.2 Program Results

```
     f07pn example results

Solution
   2.0000 + 1.0000i
   3.0000 - 2.0000i
  -1.0000 + 2.0000i
   1.0000 - 1.0000i

 Details of factorization
            1          2          3          4
 1     -7.1028     0.2997     0.3397    -0.1518
        0.0000     0.1578     0.0303     0.3743

 2                -5.4176     0.5637     0.3100
                   0.0000     0.2850     0.0433

 3                           -1.8400     3.9100
                             0.0000    -1.5000
```

```
 4                                      -1.3600
                                         0.0000

Pivot indices
             1           2          -1          -1
```