NAG Toolbox

nag_linsys_complex_herm_packed_solve (f04cj)

1 Purpose

nag_linsys_complex_herm_packed_solve (f04cj) computes the solution to a complex system of linear equations $AX = B$, where A is an n by n complex Hermitian matrix, stored in packed format and X and B are n by r matrices. An estimate of the condition number of A and an error bound for the computed solution are also returned.

2 Syntax

```
[ap, ipiv, b, rcond, errbnd, ifail] = nag_linsys_complex_herm_packed_solve(uplo, ap, b, 'n', n, 'nrhs_p', nrhs_p)
[ap, ipiv, b, rcond, errbnd, ifail] = f04cj(uplo, ap, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

The diagonal pivoting method is used 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, and 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

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 the matrix A is stored.

If **uplo** = 'L', the lower triangle of the matrix 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 column-wise in a linear array. The jth column of the matrix A is stored in the array **ap** as follows:

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 $i \leq j$;

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

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 $\max(1,\mathbf{nrhs\_p})$.

The n by r matrix of right-hand sides B.

5.2 Optional Input Parameters

1: **n** – INTEGER

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

The number of linear equations n, 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**.

The number of right-hand sides r, 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 $\max(1,\mathbf{n} \times (\mathbf{n}+1)/2)$

If **ifail** ≥ 0, 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

If **ifail** ≥ 0, details of the interchanges and the block structure of D, as determined by nag_lapack_zhptrf (f07pr).

If **ipiv**$(k) > 0$, then rows and columns k and **ipiv**(k) were interchanged, and d_{kk} is a 1 by 1 diagonal block;

if **uplo** = 'U' and **ipiv**$(k) =$ **ipiv**$(k-1) < 0$, then rows and columns $k-1$ and $-$**ipiv**(k) were interchanged and $d_{k-1:k,k-1:k}$ is a 2 by 2 diagonal block;

if **uplo** = 'L' and **ipiv**$(k) =$ **ipiv**$(k+1) < 0$, then rows and columns $k+1$ and $-$**ipiv**(k) were interchanged and $d_{k:k+1,k:k+1}$ is a 2 by 2 diagonal block.

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 $\max(1,\mathbf{nrhs\_p})$.

If **ifail** $= 0$ or $\mathbf{n}+1$, the n by r solution matrix X.

4: **rcond** – REAL (KIND=nag_wp)

If no constraints are violated, an estimate of the reciprocal of the condition number of the matrix A, computed as $\mathbf{rcond} = 1/\left(\|A\|_1\left\|A^{-1}\right\|_1\right)$.

5: **errbnd** – REAL (KIND=nag_wp)

If **ifail** $= 0$ or $\mathbf{n}+1$, an estimate of the forward error bound for a computed solution $\hat{x}$, such that $\|\hat{x}-x\|_1/\|x\|_1 \leq$ **errbnd**, where $\hat{x}$ is a column of the computed solution returned in the array **b**

and x is the corresponding column of the exact solution X. If **rcond** is less than ***machine precision***, then **errbnd** is returned as unity.

6: **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 > 0 and **ifail** $\leq$ **n**

Diagonal block $\langle value \rangle$ of the block diagonal matrix is zero. The factorization has been completed, but the solution could not be computed.

ifail $=$ **n** $+ 1$ (*warning*)

A solution has been computed, but **rcond** is less than ***machine precision*** so that the matrix A is numerically singular.

ifail $= -1$

On entry, **uplo** $\neq$ 'U' or 'L'.

ifail $= -2$

Constraint: **n** ≥ 0.

ifail $= -3$

Constraint: **nrhs_p** ≥ 0.

ifail $= -7$

Constraint: $ldb \geq \max(1, \mathbf{n})$.

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.

The double allocatable memory required is ***n****, and the complex allocatable memory required is* $2 \times$ **n**. *Allocation failed before the solution could 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. nag_linsys_complex_herm_packed_solve (f04cj) uses the approximation $\|E\|_1 = \epsilon\|A\|_1$ to estimate **errbnd**. See Section 4.4 of Anderson *et al.* (1999) for further details.

8 Further Comments

The packed storage scheme is illustrated by the following example when $n = 4$ and **uplo** = 'U'. Two-dimensional storage of the Hermitian matrix A:

$$\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ & a_{22} & a_{23} & a_{24} \\ & & a_{33} & a_{34} \\ & & & a_{44} \end{array} \quad \left(a_{ij} = \bar{a}_{ji}\right).$$

Packed storage of the upper triangle of A:

$$\mathbf{ap} = \left[\, a_{11}, \quad a_{12}, \quad a_{22}, \quad a_{13}, \quad a_{23}, \quad a_{33}, \quad a_{14}, \quad a_{24}, \quad a_{34}, \quad a_{44} \,\right].$$

The total number of floating-point operations required to solve the equations $AX = B$ is proportional to $\left(\frac{1}{3}n^3 + 2n^2r\right)$. The condition number estimation typically requires between four and five solves and never more than eleven solves, following the factorization.

In practice the condition number estimator is very reliable, but it can underestimate the true condition number; see Section 15.3 of Higham (2002) for further details.

Function nag_linsys_complex_symm_packed_solve (f04dj) is for complex symmetric matrices, and the real analogue of nag_linsys_complex_herm_packed_solve (f04cj) is nag_linsys_real_symm_packed_solve (f04bj).

9 Example

This example solves the equations

$$AX = B,$$

where A is the Hermitian indefinite 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

$$B = \begin{pmatrix} 2.98 - 10.18i & 28.68 - 39.89i \\ -9.58 + 3.88i & -24.79 - 8.40i \\ -0.77 - 16.05i & 4.23 - 70.02i \\ 7.79 + 5.48i & -35.39 + 18.01i \end{pmatrix}.$$

An estimate of the condition number of A and an approximate error bound for the computed solutions are also printed.

9.1 Program Text

```
     function f04cj_example

fprintf('f04cj example results\n\n');

% Solve complex Ax = b for Hermitian packed A
% with error bound and condition number
uplo = 'U';
ap = [-1.84                                          ...
       0.11 -  0.11   -4.63 + 0i                     ...
      -1.78 -  1.18i  -1.84 + 0.03i  -8.87 + 0i      ...
       3.91 -  1.5i    2.21 + 0.21i   1.58 - 0.9i  -1.36 + 0i];
b = [ 2.98 - 10.18i,  28.68 - 39.89i;
     -9.58 +  3.88i, -24.79 -  8.4i;
     -0.77 - 16.05i,   4.23 - 70.02i;
      7.79 +  5.48i, -35.39 + 18.01i];

[ap, ipiv, x, rcond, errbnd, ifail] = ...
  f04cj(uplo, ap, b);

disp('Solution');
disp(x);
disp('Estimate of condition number');
fprintf('%10.1f\n\n',1/rcond);
disp('Estimate of error bound for computed solutions');
fprintf('%10.1e\n\n',errbnd);
```

9.2 Program Results

```
     f04cj example results

Solution
   2.0330 + 1.0218i  -8.1639 + 6.0692i
   3.0234 - 2.1166i   7.4388 - 2.0180i
  -1.0112 + 1.9975i  -1.0639 + 4.9118i
   1.0794 - 1.1045i   3.1469 - 4.2238i

Estimate of condition number
       6.7

Estimate of error bound for computed solutions
   7.5e-16
```