

NAG Toolbox

nag_lapack_zhesvx (f07mp)

1 Purpose

nag_lapack_zhesvx (f07mp) uses the diagonal pivoting factorization to compute the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n Hermitian matrix and X and B are n by r matrices. Error bounds on the solution and a condition estimate are also provided.

2 Syntax

```
[af, ipiv, x, rcond, ferr, berr, info] = nag_lapack_zhesvx(fact, uplo, a, af,
ipiv, b, 'n', n, 'nrhs_p', nrhs_p)

[af, ipiv, x, rcond, ferr, berr, info] = f07mp(fact, uplo, a, af, ipiv, b, 'n',
n, 'nrhs_p', nrhs_p)
```

3 Description

nag_lapack_zhesvx (f07mp) performs the following steps:

1. If **fact** = 'N', the diagonal pivoting method is used to factor A . The form of the factorization is $A = UDU^H$ if **uplo** = 'U' or $A = LDL^H$ 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.
2. If some $d_{ii} = 0$, so that D is exactly singular, then the function returns with **info** = i . Otherwise, the factored form of A is used to estimate the condition number of the matrix A . If the reciprocal of the condition number is less than *machine precision*, **info** $\geq n + 1$ is returned as a warning, but the function still goes on to solve for X and compute error bounds as described below.
3. The system of equations is solved for X using the factored form of A .
4. Iterative refinement is applied to improve the computed solution matrix and to calculate error bounds and backward error estimates for it.

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: **fact** – CHARACTER(1)

Specifies whether or not the factorized form of the matrix A has been supplied.

fact = 'F'

af and **ipiv** contain the factorized form of the matrix A . **af** and **ipiv** will not be modified.

fact = 'N'

The matrix A will be copied to **af** and factorized.

Constraint: **fact** = 'F' or 'N'.

2: **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'.

3: **a**(*lda*,:) – COMPLEX (KIND=nag_wp) array

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

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

The n by n Hermitian matrix A .

If **uplo** = 'U', the upper triangular part of a must be stored and the elements of the array below the diagonal are not referenced.

If **uplo** = 'L', the lower triangular part of a must be stored and the elements of the array above the diagonal are not referenced.

4: **af**(*ldaf*,:) – COMPLEX (KIND=nag_wp) array

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

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

If **fact** = 'F', **af** contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $\mathbf{a} = UDU^H$ or $\mathbf{a} = LDL^H$ as computed by nag_lapack_zhetrf (f07mr).

5: **ipiv**(:) – INTEGER array

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

If **fact** = 'F', **ipiv** contains details of the interchanges and the block structure of D , as determined by nag_lapack_zhetrf (f07mr).

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

- 6: **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 right-hand side matrix B .

5.2 Optional Input Parameters

- 1: **n** – INTEGER
Default: the first dimension of the arrays **a**, **af**, **b** and the second dimension of the arrays **a**, **af**, **ipiv**.
 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: **af**(*ldaf*,:) – COMPLEX (KIND=nag_wp) array
 The first dimension of the array **af** will be $\max(1, \mathbf{n})$.
 The second dimension of the array **af** will be $\max(1, \mathbf{n})$.
 If **fact** = 'N', **af** returns the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $\mathbf{a} = UDU^H$ or $\mathbf{a} = LDL^H$.
- 2: **ipiv**(:) – INTEGER array
 The dimension of the array **ipiv** will be $\max(1, \mathbf{n})$.
 If **fact** = 'N', **ipiv** contains details of the interchanges and the block structure of D , as determined by nag_lapack_zhetrf (f07mr), as described above.
- 3: **x**(*ldx*,:) – COMPLEX (KIND=nag_wp) array
 The first dimension of the array **x** will be $\max(1, \mathbf{n})$.
 The second dimension of the array **x** will be $\max(1, \mathbf{nrhs_p})$.
 If **info** = 0 or $\mathbf{n} + 1$, the n by r solution matrix X .
- 4: **rcond** – REAL (KIND=nag_wp)
 The estimate of the reciprocal condition number of the matrix A . If **rcond** = 0.0, the matrix may be exactly singular. This condition is indicated by **info** > 0 and **info** ≤ \mathbf{n} . Otherwise, if **rcond** is less than the *machine precision*, the matrix is singular to working precision. This condition is indicated by **info** ≥ $\mathbf{n} + 1$.
- 5: **ferr**(:) – REAL (KIND=nag_wp) array
 The dimension of the array **ferr** will be $\max(1, \mathbf{nrhs_p})$.
 If **info** = 0 or $\mathbf{n} + 1$, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \mathbf{ferr}(j)$ where \hat{x}_j is the j th column of the computed solution returned in the array **x** and x_j is the corresponding column of the exact solution X . The estimate

is as reliable as the estimate for **rcond**, and is almost always a slight overestimate of the true error.

6: **berr**(:) – REAL (KIND=nag_wp) array

The dimension of the array **berr** will be $\max(1, \text{nrhs.p})$

If **info** = 0 or **n** + 1, an estimate of the component-wise relative backward error of each computed solution vector \hat{x}_j (i.e., the smallest relative change in any element of A or B that makes \hat{x}_j an exact solution).

7: **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 and **info** ≤ **n** (*warning*)

Element $\langle \text{value} \rangle$ of the diagonal is exactly zero. The factorization has been completed, but the factor D is exactly singular, so the solution and error bounds could not be computed. **rcond** = 0.0 is returned.

info = **n** + 1 (*warning*)

D is nonsingular, but **rcond** is less than *machine precision*, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of **rcond** would suggest.

7 Accuracy

For each right-hand side vector b , the computed solution \hat{x} is the exact solution of a perturbed system of equations $(A + E)\hat{x} = b$, where

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

where ϵ is the *machine precision*. See Chapter 11 of Higham (2002) for further details.

If \hat{x} is the true solution, then the computed solution x satisfies a forward error bound of the form

$$\frac{\|x - \hat{x}\|_\infty}{\|\hat{x}\|_\infty} \leq w_c \text{cond}(A, \hat{x}, b)$$

where $\text{cond}(A, \hat{x}, b) = \frac{\| |A^{-1}|(|A|\hat{x} + |b|) \|_\infty}{\|\hat{x}\|_\infty} \leq \text{cond}(A) = \| |A^{-1}| |A| \|_\infty \leq \kappa_\infty(A)$. If \hat{x} is the j th column of X , then w_c is returned in **berr**(j) and a bound on $\|x - \hat{x}\|_\infty / \|\hat{x}\|_\infty$ is returned in **ferr**(j). See Section 4.4 of Anderson *et al.* (1999) for further details.

8 Further Comments

The factorization of A requires approximately $\frac{4}{3}n^3$ floating-point operations.

For each right-hand side, computation of the backward error involves a minimum of $16n^2$ floating-point operations. Each step of iterative refinement involves an additional $24n^2$ operations. At most five steps of iterative refinement are performed, but usually only one or two steps are required. Estimating the forward error involves solving a number of systems of equations of the form $Ax = b$; the number is usually 4 or 5 and never more than 11. Each solution involves approximately $8n^2$ operations.

The real analogue of this function is `nag_lapack_dsylvx` (f07mb). The complex symmetric analogue of this function is `nag_lapack_zsylvx` (f07np).

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

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

Error estimates for the solutions, and an estimate of the reciprocal of the condition number of the matrix A are also output.

9.1 Program Text

```
function f07mp_example

fprintf('f07mp example results\n\n');

% Hermitian indefinite matrix A (Upper triangular part stored)
uplo = 'Upper';
a = [-1.84 + 0i, 0.11 - 0.11i, -1.78 - 1.18i, 3.91 - 1.5i;
      0 + 0i, -4.63 + 0i, -1.84 + 0.03i, 2.21 + 0.21i;
      0 + 0i, 0 + 0i, -8.87 + 0i, 1.58 - 0.9i;
      0 + 0i, 0 + 0i, 0 + 0i, -1.36 + 0i];

% RHS
b = [ 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];

% Input parameters
fact = 'Not factored';
af = a;
ipiv = zeros(size(b,1), 1, nag_int_name);

% Solve
[af, ipiv, x, rcond, ferr, berr, info] = ...
    f07mp( ...
        fact, uplo, a, af, ipiv, b);

disp('Solution(s)');
disp(x);
disp('Backward errors (machine-dependent)');
fprintf('%10.1e',berr);
fprintf('\n');
disp('Estimated forward error bounds (machine-dependent)');
fprintf('%10.1e',ferr);
fprintf('\n\n');
disp('Estimate of reciprocal condition number');
fprintf('%10.1e\n\n',rcond);
```

9.2 Program Results

f07mp example results

Solution(s)

```
2.0000 + 1.0000i  -8.0000 + 6.0000i
3.0000 - 2.0000i   7.0000 - 2.0000i
-1.0000 + 2.0000i -1.0000 + 5.0000i
1.0000 - 1.0000i   3.0000 - 4.0000i
```

Backward errors (machine-dependent)

```
7.3e-17  8.1e-17
```

Estimated forward error bounds (machine-dependent)

```
2.6e-15  3.1e-15
```

Estimate of reciprocal condition number

```
1.5e-01
```
