NAG Toolbox

nag_lapack_zptsvx (f07jp)

1 Purpose

nag_lapack_zptsvx (f07jp) uses the factorization

$$A = LDL^{\mathrm{H}}$$

to compute the solution to a complex system of linear equations

$$AX = B,$$

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

2 Syntax

```
[df, ef, x, rcond, ferr, berr, info] = nag_lapack_zptsvx(fact, d, e, df, ef, b, 'n', n, 'nrhs_p', nrhs_p)
[df, ef, x, rcond, ferr, berr, info] = f07jp(fact, d, e, df, ef, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

nag_lapack_zptsvx (f07jp) performs the following steps:

1. If **fact** = 'N', the matrix A is factorized as $A = LDL^{\mathrm{H}}$, where L is a unit lower bidiagonal matrix and D is diagonal. The factorization can also be regarded as having the form $A = U^{\mathrm{H}}DU$.
2. If the leading i by i principal minor is not positive definite, 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'
df and **ef** contain the factorized form of the matrix A. **df** and **ef** will not be modified.

fact = 'N'
The matrix A will be copied to **df** and **ef** and factorized.

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

2: **d**(:) – REAL (KIND=nag_wp) array

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

The n diagonal elements of the tridiagonal matrix A.

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

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

The $(n - 1)$ subdiagonal elements of the tridiagonal matrix A.

4: **df**(:) – REAL (KIND=nag_wp) array

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

If **fact** = 'F', **df** must contain the n diagonal elements of the diagonal matrix D from the LDL^{H} factorization of A.

5: **ef**(:) – COMPLEX (KIND=nag_wp) array

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

If **fact** = 'F', **ef** must contain the $(n - 1)$ subdiagonal elements of the unit bidiagonal factor L from the LDL^{H} factorization of A.

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 array **b** and the dimension of the arrays **d**, **df**.

n, 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: **df**(:) – REAL (KIND=nag_wp) array

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

If **fact** = 'N', **df** contains the n diagonal elements of the diagonal matrix D from the LDL^{H} factorization of A.

2: **ef**(:) – COMPLEX (KIND=nag_wp) array

The dimension of the array **ef** will be $\max(1, \mathbf{n} - 1)$

If **fact** = 'N', **ef** contains the $(n-1)$ subdiagonal elements of the unit bidiagonal factor L from the LDL^{H} factorization of A.

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 **n** + 1, the n by r solution matrix X.

4: **rcond** – REAL (KIND=nag_wp)

The reciprocal condition number of the matrix A. If **rcond** is less than the ***machine precision*** (in particular, if **rcond** = 0.0), the matrix is singular to working precision. This condition is indicated by a return code of **info** $\geq$ **n** + 1.

5: **ferr**(**nrhs_p**) – REAL (KIND=nag_wp) array

The forward error bound for each solution vector $\hat{x}_j$ (the jth column of the solution matrix X). If x_j is the true solution corresponding to $\hat{x}_j$, **ferr**(j) is an estimated upper bound for the magnitude of the largest element in $(\hat{x}_j - x_j)$ divided by the magnitude of the largest element in $\hat{x}_j$.

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

The component-wise relative backward error of each 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** $\leq$ **n**

The leading minor of order ⟨*value*⟩ of A is not positive definite, so the factorization could not be completed, and the solution has not been 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

$$|E| \leq c(n)\epsilon \left|R^{\mathrm{T}}\right||R|, \text{ where } R = D^{\frac{1}{2}}U,$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***. See Section 10.1 of Higham (2002) for further details.

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

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

where $\operatorname{cond}(A,\hat{x},b) = \left\|\left|A^{-1}\right|(|A||\hat{x}|+|b|)\right\|_\infty / \|\hat{x}\|_\infty \leq \operatorname{cond}(A) = \left\|\left|A^{-1}\right||A|\right\|_\infty \leq \kappa_\infty(A)$. If $\hat{x}$ is the jth 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 number of floating-point operations required for the factorization, and for the estimation of the condition number of A is proportional to n. The number of floating-point operations required for the solution of the equations, and for the estimation of the forward and backward error is proportional to nr, where r is the number of right-hand sides.

The condition estimation is based upon Equation (15.11) of Higham (2002). For further details of the error estimation, see Section 4.4 of Anderson *et al.* (1999).

The real analogue of this function is nag_lapack_dptsvx (f07jb).

9 Example

This example solves the equations

$$AX = B,$$

where A is the Hermitian positive definite tridiagonal matrix

$$A = \begin{pmatrix} 16.0 & 16.0-16.0i & 0 & 0 \\ 16.0+16.0i & 41.0 & 18.0+9.0i & 0 \\ 0 & 18.0-9.0i & 46.0 & 1.0+4.0i \\ 0 & 0 & 1.0-4.0i & 21.0 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 64.0+16.0i & -16.0-32.0i \\ 93.0+62.0i & 61.0-66.0i \\ 78.0-80.0i & 71.0-74.0i \\ 14.0-27.0i & 35.0+15.0i \end{pmatrix}.$$

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

9.1 Program Text

```
function f07jp_example

fprintf('f07jp example results\n\n');

% Hermitian tridiagonal A stored as two diagonals
d = [ 16            41          46            21];
e = [ 16 + 16i      18 - 9i      1 - 4i         ];

%RHS
```

```
b = [ 64 + 16i, -16 - 32i;
      93 + 62i,  61 - 66i;
      78 - 80i,  71 - 74i;
      14 - 27i,  35 + 15i];

% Input parameters
n    = numel(d);
fact = 'Not factored';
df   = zeros(n, 1);
ef   = complex(zeros(n-1, 1));

%Solve
[df, ef, x, rcond, ferr, berr, info] = ...
  f07jp( ...
         fact, d, e, df, ef, 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

```
     f07jp example results

Solution(s)
   2.0000 + 1.0000i  -3.0000 - 2.0000i
   1.0000 + 1.0000i   1.0000 + 1.0000i
   1.0000 - 2.0000i   1.0000 - 2.0000i
   1.0000 - 1.0000i   2.0000 + 1.0000i

Backward errors (machine-dependent)
   0.0e+00   0.0e+00
Estimated forward error bounds (machine-dependent)
   9.0e-12   6.1e-12

Estimate of reciprocal condition number
   1.1e-04
```