NAG Toolbox

nag_linsys_complex_tridiag_solve (f04cc)

1 Purpose

nag_linsys_complex_tridiag_solve (f04cc) computes the solution to a complex system of linear equations $AX = B$, where A is an n by n tridiagonal matrix 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

```
[dl, d, du, du2, ipiv, b, rcond, errbnd, ifail] =
nag_linsys_complex_tridiag_solve(dl, d, du, b, 'n', n, 'nrhs_p', nrhs_p)
[dl, d, du, du2, ipiv, b, rcond, errbnd, ifail] = f04cc(dl, d, du, b, 'n', n,
'nrhs_p', nrhs_p)
```

3 Description

The LU decomposition with partial pivoting and row interchanges is used to factor A as $A = PLU$, where P is a permutation matrix, L is unit lower triangular with at most one nonzero subdiagonal element, and U is an upper triangular band matrix with two superdiagonals. The factored form of A is then used to solve the system of equations $AX = B$.

Note that the equations $A^{\mathrm{T}}X = B$ may be solved by interchanging the order of the arguments **du** and **dl**.

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: **dl**(:) – COMPLEX (KIND=nag_wp) array

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

Must contain the $(n - 1)$ subdiagonal elements of the matrix A.

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

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

Must contain the n diagonal elements of the matrix A.

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

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

Must contain the $(n - 1)$ superdiagonal elements of the matrix A

4: **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** and the dimension of the array **d**.

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: **dl**(:) – COMPLEX (KIND=nag_wp) array

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

If $\mathbf{ifail} \geq 0$, **dl** stores the $(n-1)$ multipliers that define the matrix L from the LU factorization of A.

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

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

If $\mathbf{ifail} \geq 0$, **d** stores the n diagonal elements of the upper triangular matrix U from the LU factorization of A.

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

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

If $\mathbf{ifail} \geq 0$, **du** stores the $(n-1)$ elements of the first superdiagonal of U.

4: **du2**$(\mathbf{n} - \mathbf{2})$ – COMPLEX (KIND=nag_wp) array

If $\mathbf{ifail} \geq 0$, **du2** returns the $(n-2)$ elements of the second superdiagonal of U.

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

If $\mathbf{ifail} \geq 0$, the pivot indices that define the permutation matrix P; at the ith step row i of the matrix was interchanged with row $\mathbf{ipiv}(i)$. $\mathbf{ipiv}(i)$ will always be either i or $(i+1)$; $\mathbf{ipiv}(i) = i$ indicates a row interchange was not required.

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

7: **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 $\mathbf{rcond} = 1/\left(\|A\|_1 \|A^{-1}\|_1\right)$.

8: **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.

9: **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 element $\langle$*value*$\rangle$ of the upper triangular factor 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 $= -1$

Constraint: **n** ≥ 0.

ifail $= -2$

Constraint: **nrhs_p** ≥ 0.

ifail $= -9$

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 complex allocatable memory required is $2 \times$ **n**. *In this case the factorization and the solution* X *have been computed, but* ***rcond*** *and* ***errbnd*** *have not been 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_tridiag_solve (f04cc) 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 total number of floating-point operations required to solve the equations $AX = B$ is proportional to nr. 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.

The real analogue of nag_linsys_complex_tridiag_solve (f04cc) is nag_linsys_real_tridiag_solve (f04bc).

9 Example

This example solves the equations

$$AX = B,$$

where A is the tridiagonal matrix

$$A = \begin{pmatrix} -1.3+1.3i & 2.0-1.0i & 0 & 0 & 0 \\ 1.0-2.0i & -1.3+1.3i & 2.0+1.0i & 0 & 0 \\ 0 & 1.0+1.0i & -1.3+3.3i & -1.0+1.0i & 0 \\ 0 & 0 & 2.0-3.0i & -0.3+4.3i & 1.0-1.0i \\ 0 & 0 & 0 & 1.0+1.0i & -3.3+1.3i \end{pmatrix}$$

and

$$B = \begin{pmatrix} 2.4-5.0i & 2.7+6.9i \\ 3.4+18.2i & -6.9-5.3i \\ -14.7+9.7i & -6.0-0.6i \\ 31.9-7.7i & -3.9+9.3i \\ -1.0+1.6i & -3.0+12.2i \end{pmatrix}.$$

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 f04cc_example

fprintf('f04cc example results\n\n');

du = [                2.0 -  1.0i;  2.0 + 1.0i; -1.0 + 1.0i;  1.0 - 1.0i];
d  = [ -1.3 +  1.3i; -1.3 +  1.3i; -1.3 + 3.3i; -0.3 + 4.3i; -3.3 + 1.3i];
dl = [  1.0 -  2.0i;  1.0 +  1.0i;  2.0 - 3.0i;  1.0 + 1.0i             ];

b = [   2.4 -  5.0i,  2.7 +  6.9i;
        3.4 + 18.2i, -6.9 -  5.3i;
      -14.7 +  9.7i, -6.0 -  0.6i;
       31.9 -  7.7i, -3.9 +  9.3i;
       -1.0 +  1.6i, -3.0 + 12.2i];

[dl, d, du, du2, ipiv, x, rcond, errbnd, ifail] = ...
  f04cc(dl, d, du, b);
```

```
fprintf('Solution is x:\n');
disp(x);
fprintf('\nApproximate condition number = %8.3f\n',1/rcond);
fprintf('Error bound on solution      = %11.3e\n',errbnd);
```

9.2 Program Results

```
     f04cc example results

Solution is x:
   1.0000 + 1.0000i   2.0000 - 1.0000i
   3.0000 - 1.0000i   1.0000 + 2.0000i
   4.0000 + 5.0000i  -1.0000 + 1.0000i
  -1.0000 - 2.0000i   2.0000 + 1.0000i
   1.0000 - 1.0000i   2.0000 - 2.0000i

Approximate condition number =  183.970
Error bound on solution      =   2.042e-14
```