NAG Toolbox for MATLAB

f04dh

1 Purpose

f04dh computes the solution to a complex system of linear equations $AX = B$, where A is an n by n complex symmetric 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

```
[a, ipiv, b, rcond, errbnd, ifail] = f04dh(uplo, a, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

The diagonal pivoting method is used to factor A as $A = UDU^{\mathrm{T}}$, if **uplo** = 'U', or $A = LDL^{\mathrm{T}}$, if **uplo** = 'L', where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric 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 URL: 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 – string**

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: **a**(**lda**,∗) **– complex array**

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

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

The n by n complex symmetric matrix A.

If **uplo** = 'U', the leading **n** by **n** upper triangular part of the array **a** contains the upper triangular part of the matrix A, and the strictly lower triangular part of **a** is not referenced.

If **uplo** = 'L', the leading **n** by **n** lower triangular part of the array **a** contains the lower triangular part of the matrix A, and the strictly upper triangular part of **a** is not referenced.

3: **b**(**ldb**,∗) **– complex array**

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

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

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

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The second dimension of the array **a**.

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

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

2: **nrhs_p – int32 scalar**

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 Input Parameters Omitted from the MATLAB Interface

lda, ldb

5.4 Output Parameters

1: **a(lda,$*$) – complex array**

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

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

If $\mathbf{ifail} \geq 0$, the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = UDU^{\mathrm{T}}$ or $A = LDL^{\mathrm{T}}$ as computed by f07nr.

2: **ipiv($*$) – int32 array**

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

If $\mathbf{ifail} \geq 0$, details of the interchanges and the block structure of D, as determined by f07nr.

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

3: **b(ldb,$*$) – complex array**

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

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

If $\mathbf{ifail} = 0$ or $Np1$, the n by r solution matrix X.

4: **rcond – double scalar**

If $\mathbf{ifail} \geq 0$, 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 – double scalar**

If **ifail** $= 0$ or $Np1$, 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 – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail < 0 and **ifail** $\neq -999$

If **ifail** $= -i$, the ith argument had an illegal value.

ifail $= -999$

Allocation of memory failed. The double allocatable memory required is **n**, and the ***complex*16*** allocatable memory required is $\max(2 \times \mathbf{n}, \mathbf{lwork})$, where **lwork** is the optimum workspace required by f07nn. If this failure occurs it may be possible to solve the equations by calling the packed storage version of f04dh, f04dj, or by calling f07nn directly with less than the optimum workspace (see Chapter F07).

ifail > 0 and **ifail** $\leq N$

If **ifail** $= i$, d_{ii} is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution could not be computed.

ifail $= N + 1$

rcond is less than ***machine precision***, so that the matrix A is numerically singular. A solution to the equations $AX = B$ has nevertheless 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. f04dh 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 $\left(\frac{1}{3}n^3 + 2n^2r\right)$. 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 f04ch is for complex Hermitian matrices, and the real analogue of f04dh is f04bh.

9 Example

```
uplo = 'Upper';
a = [complex(-0.56, +0.12), complex(-1.54, -2.86), complex(5.32, -1.59),
complex(3.8, +0.92);
         complex(0,  0),  complex(-2.83,  -0.03),  complex(-3.52,  +0.58),
complex(-7.86, -2.96);
     complex(0, 0), complex(0, +0), complex(8.86, +1.81), complex(5.14, -
0.64);
     complex(0, 0), complex(0, 0), complex(0, 0), complex(-0.39, -0.71)];
b = [complex(-6.43, +19.24), complex(-4.59, -35.53);
     complex(-0.49, -1.47), complex(6.95, +20.49);
     complex(-48.18, +66), complex(-12.08, -27.02);
     complex(-55.64, +41.22), complex(-19.09, -35.97)];
[aOut, ipiv, bOut, rcond, errbnd, ifail] = f04dh(uplo, a, b)
```

```
aOut =
   -2.0954 - 2.2011i   -0.1071 - 0.3157i   -0.4823 + 0.0150i    0.4426 +
0.1936i
         0               4.4079 + 5.3991i   -0.6078 + 0.2811i    0.5279 -
0.3715i
         0                    0             -2.8300 - 0.0300i   -7.8600 -
2.9600i
         0                    0                   0             -0.3900 -
0.7100i
ipiv =
          1
          2
         -2
         -2
bOut =
  -4.0000 + 3.0000i  -1.0000 + 1.0000i
   3.0000 - 2.0000i   3.0000 + 2.0000i
  -2.0000 + 5.0000i   1.0000 - 3.0000i
   1.0000 - 1.0000i  -2.0000 - 1.0000i
rcond =
    0.0486
errbnd =
   2.2884e-15
ifail =
          0
```