nag_complex_cholesky (f01bnc)

1. Purpose

nag_complex_cholesky (f01bnc) computes a Cholesky factorization of a complex positive-definite Hermitian matrix.

2. Specification

```
#include <nag.h>
#include <nagf01.h>

void nag_complex_cholesky(Integer n, Complex a[], Integer tda,
     double p[], NagError *fail)
```

3. Description

nag_complex_cholesky computes the Cholesky factorization of a complex positive-definite Hermitian matrix $A = U^H U$, where U is a complex upper triangular matrix with real diagonal elements.

4. Parameters

n

Input: n, the order of the matrix A.
Constraint: $\mathbf{n} \geq 1$.

a[n][tda]

Input: the lower triangle of the n by n positive-definite Hermitian matrix A. The elements of the array above the diagonal need not be set.
Output: the off-diagonal elements of the upper triangular matrix U. The lower triangle of A is unchanged.

tda

Input: the second dimension of the array **a** as declared in the function from which nag_complex_cholesky is called.
Constraint: $\mathbf{tda} \geq \mathbf{n}$.

p[n]

Output: the reciprocals of the real diagonal elements of U.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_DIAG_IMAG_NON_ZERO

Matrix diagonal element **a** [⟨*value*⟩][⟨*value*⟩] has non-zero imaginary part.

NE_NOT_POS_DEF

The matrix is not positive-definite, possibly due to rounding errors.

NE_INT_ARG_LT

On entry, **n** must not be less than 1: **n** = ⟨*value*⟩.

NE_2_INT_ARG_LT

On entry, **tda** = ⟨*value*⟩ while **n** = ⟨*value*⟩. These parameters must satisfy $\mathbf{tda} \geq \mathbf{n}$.

6. Further Comments

The time taken by the function is approximately proportional to n^3.

6.1. Accuracy

The Cholesky factorization of a positive-definite matrix is known for its remarkable numerical stability. The computed matrix U satisfies the relation $U^H U = A + E$ where the 2-norms of A and E are related by

$$\|E\| \leq c\epsilon\|A\|,$$

c is a modest function of n, and ϵ is the ***machine precision***.

6.2. References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation (Vol II, Linear Algebra)* Springer-Verlag pp 9–30.

7. See Also

nag_hermitian_lin_eqn_mult_rhs (f04awc)

8. Example

To compute the Cholesky factorization of the well-conditioned positive-definite Hermitian matrix

$$\begin{pmatrix} 15 & 1-2i & 2 & -4+3i \\ 1+2i & 20 & -2+i & 3-3i \\ 2 & -2-i & 18 & -1+2i \\ -4-3i & 3+3i & -1-2i & 26 \end{pmatrix}.$$

8.1. Program Text

```
/* nag_complex_cholesky(f01bnc) Example Program
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 1, 1990.
 */

#include <nag.h>
#include <math.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagf01.h>

#define NMAX 8
#define TDA NMAX
#define COMPLEX(A) A.re, A.im

main()
{

  Integer i, j, n;
  Complex a[NMAX][TDA];
  double p[NMAX];
  static NagError fail;

  Vprintf("f01bnc Example Program Results\n");
  /* Skip heading in data file */
  Vscanf("%*[^\n]");
  Vscanf("%ld", &n);
  if (n<1 || n>NMAX)
    {
      Vprintf("\n n is out of range: n = %ld", n);
      exit(EXIT_FAILURE);
    }
  for (i=0; i<n; ++i)
    for (j=0; j<=i; ++j)
      Vscanf(" ( %lf , %lf ) ", COMPLEX(&a[i][j]));
```

```
  fail.print = TRUE;
  f01bnc(n, (Complex *)a, (Integer)TDA, p, &fail);
  if (fail.code != NE_NOERROR)
    exit(EXIT_FAILURE);
  Vprintf("\n Upper triangle of Complex matrix U by rows\n");
  for (i=0; i<n; ++i)
    {
      Vprintf("\n");
      Vprintf("   (%7.4f,%9.4f)\n", 1.0/p[i], 0.0);
      for (j=i+1; j<n; ++j)
        Vprintf("   (%7.4f,%9.4f)\n", COMPLEX(a[i][j]));
    }
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

```
f01bnc Example Program Data
  4
(15.0,  0.0)
( 1.0,  2.0)  (20.0,  0.0)
( 2.0,  0.0)  (-2.0, -1.0)  (18.0,  0.0)
(-4.0, -3.0)  ( 3.0,  3.0)  (-1.0, -2.0)  (26.0,  0.0)
```

8.3. Program Results

```
f01bnc Example Program Results

 Upper triangle of Complex matrix U by rows

   ( 3.8730,   0.0000)
   ( 0.2582,  -0.5164)
   ( 0.5164,   0.0000)
   (-1.0328,   0.7746)

   ( 4.4347,   0.0000)
   (-0.4811,   0.1654)
   ( 0.8268,  -0.6013)

   ( 4.1803,   0.0000)
   ( 0.0073,   0.3463)

   ( 4.8133,   0.0000)
```