NAG Library Function Document

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 (f01bnc) 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 References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation II, Linear Algebra* Springer–Verlag

5 Arguments

1: **n** – Integer *Input*

On entry: n, the order of the matrix A.

Constraint: $\mathbf{n} \geq 1$.

2: **a**[**n** × **tda**] – Complex *Input/Output*

On entry: 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.

On exit: the off-diagonal elements of the upper triangular matrix U. The lower triangle of A is unchanged.

3: **tda** – Integer *Input*

On entry: the stride separating matrix column elements in the array **a**.

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

4: **p**[**n**] – double *Output*

On exit: the reciprocals of the real diagonal elements of U.

5: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_2_INT_ARG_LT

On entry, **tda** $= \langle value \rangle$ while **n** $= \langle value \rangle$. These arguments must satisfy **tda** $\geq$ **n**.

NE_DIAG_IMAG_NON_ZERO

Matrix diagonal element **a**$[(\langle value \rangle) \times$ **tda** $+ \langle value \rangle]$ has nonzero imaginary part.

NE_INT_ARG_LT

On entry, **n** $= \langle value \rangle$.
Constraint: **n** ≥ 1.

NE_NOT_POS_DEF

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

7 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***.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by nag_complex_cholesky (f01bnc) is approximately proportional to n^3.

10 Example

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

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

10.1 Program Text

```
/* nag_complex_cholesky (f01bnc) Example Program.
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 1, 1990.
 * Mark 8 revised, 2004.
 */

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

#define COMPLEX(A) A.re, A.im
```

```
#define A(I, J)    a[(I) *tda + J]
int main(void)
{

  Complex  *a = 0;
  Integer  exit_status = 0, i, j, n, tda;
  NagError fail;
  double   *p = 0;

  INIT_FAIL(fail);

  printf("nag_complex_cholesky (f01bnc) Example Program Results\n");
  /* Skip heading in data file */
  scanf("%*[^\n]");
  scanf("%ld", &n);
  if (n >= 1)
    {
      if (!(p = NAG_ALLOC(n, double)) ||
          !(a = NAG_ALLOC(n*n, Complex)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
      tda = n;
    }
  else
    {
      printf("Invalid n.\n");
      exit_status = 1;
      return exit_status;
    }
  for (i = 0; i < n; ++i)
    for (j = 0; j <= i; ++j)
      scanf(" ( %lf,  %lf ) ", COMPLEX(&A(i, j)));
  /* nag_complex_cholesky (f01bnc).
   * UU^H factorization of complex Hermitian positive-definite
   * matrix
   */
  nag_complex_cholesky(n, a, tda, p, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_complex_cholesky (f01bnc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }
  printf("\n Upper triangle of Complex matrix U by rows\n");
  for (i = 0; i < n; ++i)
    {
      printf("\n");
      printf("   (%7.4f,%9.4f)\n", 1.0/p[i], 0.0);
      for (j = i+1; j < n; ++j)
        printf("   (%7.4f,%9.4f)\n", COMPLEX(A(i, j)));
    }
 END:
  NAG_FREE(p);
  NAG_FREE(a);
  return exit_status;
}
```

10.2 Program Data

```
nag_complex_cholesky (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)
```

10.3 Program Results

```
nag_complex_cholesky (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)
```