NAG C Library Function Document

nag_zhetrf (f07mrc)

1 Purpose

nag_zhetrf (f07mrc) computes the Bunch–Kaufman factorization of a complex Hermitian indefinite matrix.

2 Specification

```c
void nag_zhetrf (Nag_OrderType order, Nag_UploType uplo, Integer n, Complex a[],
                Integer pda, Integer ipiv[], NagError *fail)
```

3 Description

nag_zhetrf (f07mrc) factorizes a complex Hermitian matrix $A$, using the Bunch–Kaufman diagonal pivoting method. $A$ is factorized as either $A = PUDU^HPT$ if $\text{uplo} = \text{Nag Upper}$, or $A = PLDL^HPT$ if $\text{uplo} = \text{Nag Lower}$, where $P$ is a permutation matrix, $U$ (or $L$) is a unit upper (or lower) triangular matrix and $D$ is an Hermitian block diagonal matrix with 1 by 1 and 2 by 2 diagonal blocks; $U$ (or $L$) has 2 by 2 unit diagonal blocks corresponding to the 2 by 2 blocks of $D$. Row and column interchanges are performed to ensure numerical stability while keeping the matrix Hermitian.

This method is suitable for Hermitian matrices which are not known to be positive-definite. If $A$ is in fact positive-definite, no interchanges are performed and no 2 by 2 blocks occur in $D$.

4 References


5 Parameters

1: `order` – Nag_OrderType

*Input*

On entry: the `order` parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by `order = Nag_RowMajor`. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

Constraint: `order = Nag_RowMajor` or `Nag_ColMajor`.

2: `uplo` – Nag_UploType

*Input*

On entry: indicates whether the upper or lower triangular part of $A$ is stored and how $A$ has been factorized, as follows:

- if `uplo = Nag_Upper`, the upper triangular part of $A$ is stored and $A$ is factorized as $PUDU^HPT$, where $U$ is upper triangular;
- if `uplo = Nag_Lower`, the lower triangular part of $A$ is stored and $A$ is factorized as $PLDL^HPT$, where $L$ is lower triangular.

Constraint: `uplo = Nag_Upper` or `Nag_Lower`.

3: `n` – Integer

*Input*

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

Constraint: $n \geq 0$. 
4:  \( a[\text{dim}] \) – Complex
    \( \text{Input/Output} \)

    \text{Note:} \ the \ dimension, \ \text{dim}, \ of \ the \ array \ a \ must \ be \ at \ least \ \max(1, \text{pda} \times n) .

    If \ order = \text{Nag}_{-}\text{ColMajor}, \ the \ (i, j)\text{th} \ element \ of \ the \ matrix \ A \ is \ stored \ in \ a[(j-1) \times \text{pda} + i - 1] \ and \ if \ order = \text{Nag}_{-}\text{RowMajor}, \ the \ (i, j)\text{th} \ element \ of \ the \ matrix \ A \ is \ stored \ in \ a[(i-1) \times \text{pda} + j - 1] .

    \text{On entry:} \ the \ n \ by \ n \ Hermitian \ matrix \ A . \ If \ uplo = \text{Nag}_{-}\text{Upper}, \ the \ upper \ triangle \ of \ A \ must \ be \ stored \ and \ the \ elements \ of \ the \ array \ below \ the \ diagonal \ are \ not \ referenced; \ if \ uplo = \text{Nag}_{-}\text{Lower}, \ the \ lower \ triangle \ of \ A \ must \ be \ stored \ and \ the \ elements \ of \ the \ array \ above \ the \ diagonal \ are \ not \ referenced .

    \text{On exit:} \ the \ upper \ or \ lower \ triangle \ of \ A \ is \ overwritten \ by \ details \ of \ the \ block \ diagonal \ matrix \ D \ and \ the \ multipliers \ used \ to \ obtain \ the \ factor \ U \ or \ L \ as \ specified \ by \ uplo .

5:  \text{pda} – Integer
    \( \text{Input} \)

    \text{On entry:} \ the \ stride \ separating \ row \ or \ column \ elements \ (depending \ on \ the \ value \ of \ order) \ of \ the \ matrix \ A \ in \ the \ array \ a .

    Constraint: \ \text{pda} \geq \max(1, n) .

6:  \text{ipiv}[\text{dim}] – Integer
    \( \text{Output} \)

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

    \text{On exit:} \ details \ of \ the \ interchanges \ and \ the \ block \ structure \ of \ D .

    More \ precisely, \ if \ \text{ipiv}[i - 1] = k > 0, \ d_{ii} \ is \ a \ 1 \ by \ 1 \ pivot \ block \ and \ the \ i\text{th} \ row \ and \ column \ of \ A \ were \ interchanged \ with \ the \ k\text{th} \ row \ and \ column .

    If \ uplo = \text{Nag}_{-}\text{Upper} \ and \ \text{ipiv}[i - 2] = \text{ipiv}[i - 1] = -l < 0, \ \begin{pmatrix} d_{i-1,j-1} & d_{i,j-1} \\ d_{i,j-1} & d_{ii} \end{pmatrix} \ \text{is \ a \ 2 \ by \ 2 \ pivot} \ \text{block \ and \ the \ (i-1)\text{th} \ row \ and \ column \ of \ A \ were \ interchanged \ with \ the \ l\text{th} \ row \ and \ column} .

    If \ uplo = \text{Nag}_{-}\text{Lower} \ and \ \text{ipiv}[i - 1] = \text{ipiv}[i] = -m < 0, \ \begin{pmatrix} d_{ii} & d_{i+1,i} \\ d_{i+1,i} & d_{i+1,i+1} \end{pmatrix} \ \text{is \ a \ 2 \ by \ 2 \ pivot} \ \text{block \ and \ the \ (i+1)\text{th} \ row \ and \ column \ of \ A \ were \ interchanged \ with \ the \ m\text{th} \ row \ and \ column} .

7:  \text{fail} – \text{NagError} *
    \( \text{Output} \)

    The \ NAG \ error \ parameter \ (see \ the \ Essential \ Introduction) .

6 \ Error \ Indicators \ and \ Warnings

NE\_INT

On entry, \( n = \langle \text{value} \rangle . \)

Constraint: \( n \geq 0 . \)

On entry, \( \text{pda} = \langle \text{value} \rangle . \)

Constraint: \( \text{pda} > 0 . \)

NE\_INT\_2

On entry, \( \text{pda} = \langle \text{value} \rangle, \ n = \langle \text{value} \rangle . \)

Constraint: \( \text{pda} \geq \max(1, n) . \)

NE\_SINGULAR

The \ block \ diagonal \ matrix \ D \ is \ exactly \ singular.

NE\_ALLOC\_FAIL

Memory allocation failed.
NE_BAD_PARAM
On entry, parameter (value) had an illegal value.

NE_INTERNAL_ERROR
An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

7 Accuracy
If uplo = Nag_Upper, the computed factors U and D are the exact factors of a perturbed matrix A + E, where

\[ |E| \leq c(n)\epsilon |U||D||U^H|P^T, \]

\[ c(n) \text{ is a modest linear function of } n, \text{ and } \epsilon \text{ is the machine precision. } \]

If uplo = Nag_Lower, a similar statement holds for the computed factors L and D.

8 Further Comments
The elements of D overwrite the corresponding elements of A; if D has 2 by 2 blocks, only the upper or lower triangle is stored, as specified by uplo.

The unit diagonal elements of U or L and the 2 by 2 unit diagonal blocks are not stored. The remaining elements of U or L are stored in the corresponding columns of the array a, but additional row interchanges must be applied to recover U or L explicitly (this is seldom necessary). If \( |\text{ipiv}[i-1]| = i \), for \( i = 1, 2, \ldots, n \) (as is the case when A is positive-definite), then U or L is stored explicitly (except for its unit diagonal elements which are equal to 1).

The total number of real floating-point operations is approximately \( \frac{4}{3}n^3 \).

A call to this function may be followed by calls to the functions:

- nag_zhetrs (f07msc) to solve \( AX = B \);
- nag_zhecon (f07muc) to estimate the condition number of A;
- nag_zhetri (f07mwc) to compute the inverse of A.

The real analogue of this function is nag_dsytrf (f07mdc).

9 Example
To compute the Bunch–Kaufman factorization of the matrix A, where

\[
A = \begin{pmatrix}
-1.36 + 0.00i & 1.58 + 0.90i & 2.21 - 0.21i & 3.91 + 1.50i \\
1.58 - 0.90i & -8.87 + 0.00i & -1.84 - 0.03i & -1.78 + 1.18i \\
2.21 + 0.21i & -1.84 + 0.03i & -4.63 + 0.00i & 0.11 + 0.11i \\
3.91 - 1.50i & -1.78 - 1.18i & 0.11 - 0.11i & -1.84 + 0.00i 
\end{pmatrix}.
\]

9.1 Program Text

/* nag_zhetrf (f07mrc) Example Program. *
 * Copyright 2001 Numerical Algorithms Group. *
 * Mark 7, 2001. */

#include <stdio.h>
#include <nag.h>
#include <nagf07.h>
#include <nagx04.h>
int main(void) {

    /* Scalars */
    Integer i, j, n, pda;
    Integer exit_status=0;
    Nag_UploType uplo_enum;
    Nag_MatrixType matrix;
    NagError fail;
    Nag_OrderType order;

    /* Arrays */
    Integer *ipiv=0;
    char uplo[2];
    Complex *a=0;

    #ifdef NAG_COLUMN_MAJOR
    #define A(I,J) a[(J-1)*pda+I-1]
    order = Nag_ColMajor;
    #else
    #define A(I,J) a[(I-1)*pda+J-1]
    order = Nag_RowMajor;
    #endif

    INIT_FAIL(fail);
    Vprintf("f07mrc Example Program Results\n\n");

    /* Skip heading in data file */
    Vscanf("%*[^
] ");
    Vscanf("%ld%*[^
] ", &n);

    #ifdef NAG_COLUMN_MAJOR
    pda = n;
    #else
    pda = n;
    #endif

    /* Allocate memory */
    if ( !(ipiv = NAG_ALLOC(n, Integer)) ||
        !(a = NAG_ALLOC(n* n, Complex)) )
    {
        Vprintf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

    /* Read A from data file */
    Vscanf("\' %1s \'%*[\n] ", uplo);
    if (*(unsigned char *)uplo == 'L')
    {
        uplo_enum = Nag_Lower;
        matrix = Nag_LowerMatrix;
    }
    else if (*(unsigned char *)uplo == 'U')
    {
        uplo_enum = Nag_Upper;
        matrix = Nag_UpperMatrix;
    }
    else
    {
        Vprintf("Unrecognised character for Nag_UploType type\n");
        exit_status = -1;
        goto END;
    }

    if (uplo_enum == Nag_Upper)
    {
        for (i = 1; i <= n; ++i)
        {
            for (j = i; j <= n; ++j)
                Vscanf(" ( %lf , %lf )", &A(i,j).re, &A(i,j).im);
        }
    }
}

f07mrc.4 [NP3645/7]
else
{
  for (i = 1; i <= n; ++i)
  {
    for (j = 1; j <= i; ++j)
      Vscanf(" ( %lf , %lf )", &A(i,j).re, &A(i,j).im);
  }
  Vscanf("%*[\n ]");
}

/* Factorize A */
f07mrc(order, uplo_enum, n, a, pda, ipiv, &fail);
if (fail.code != NE_NOERROR)
  {
    Vprintf("Error from f07mrc.
    %s
    ", fail.message);
    exit_status = 1;
    goto END;
  }
/* Print factor */
x04dbc(order, matrix, Nag_NonUnitDiag, n, n, a, pda, Nag_BracketForm,
  "%7.4f", "Details of Factorization", Nag_IntegerLabels, 0,
  Nag_IntegerLabels, 0, 80, 0, 0, &fail);
if (fail.code != NE_NOERROR)
  {
    Vprintf("Error from x04dbc.
    %s
    ", fail.message);
    exit_status = 1;
    goto END;
  }
/* Print pivot indices */
Vprintf("%ld", ipiv[i-1], i%7==0 ?"\n":"");
Vprintf("\n");
END:
if (ipiv) NAG_FREE(ipiv);
if (a) NAG_FREE(a);
return exit_status;

9.2 Program Data

f07mrc Example Program Data

4 :Value of N
'U' :Value of UPLO
(-1.36, 0.00) ( 1.58, 0.90) ( 2.21,-0.21) ( 3.91, 1.50)
(-8.87, 0.00) (-1.84,-0.03) (-1.78, 1.18)
(-4.63, 0.00) ( 0.11, 0.11)
(-1.84, 0.00) :End of matrix A

9.3 Program Results

f07mrc Example Program Results

Details of Factorization

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(-1.3600, 0.0000)</td>
<td>( 3.9100, 1.5000)</td>
<td>( 0.3100,-0.0433)</td>
<td>(-0.1518,-0.3743)</td>
</tr>
<tr>
<td>2</td>
<td>(-1.8400, 0.0000)</td>
<td>( 0.5637,-0.2850)</td>
<td>( 0.3397,-0.0303)</td>
<td>(-0.2997,-0.1578)</td>
</tr>
<tr>
<td>3</td>
<td>(-5.4176, 0.0000)</td>
<td>( 0.2997,-0.1578)</td>
<td>( 0.3397,-0.0303)</td>
<td>(-7.1028, 0.0000)</td>
</tr>
<tr>
<td>4</td>
<td>(-5.4176, 0.0000)</td>
<td>( 0.2997,-0.1578)</td>
<td>( 0.3397,-0.0303)</td>
<td>(-7.1028, 0.0000)</td>
</tr>
</tbody>
</table>

IPIV
-4 -4 3 4