NAG C Library Function Document

nag_dpbtrf (f07hdc)

1 Purpose

nag_dpbtrf (f07hdc) computes the Cholesky factorization of a real symmetric positive-definite band matrix.

2 Specification

```c
void nag_dpbtrf (Nag_OrderType order, Nag_UploType uplo, Integer n, Integer kd,
    double ab[], Integer pdab, NagError *fail)
```

3 Description

nag_dpbtrf (f07hdc) forms the Cholesky factorization of a real symmetric positive-definite band matrix \( A \) either as \( A = U^T U \) if \( \text{uplo} = \text{Nag_Upper} \), or \( A = LL^T \) if \( \text{uplo} = \text{Nag_Lower} \), where \( U \) (or \( L \)) is an upper (or lower) triangular band matrix with the same number of super-diagonals (or sub-diagonals) as \( A \).

4 References

Demmel J W (1989) On floating-point errors in Cholesky LAPACK Working Note No. 14 University of Tennessee, Knoxville


5 Parameters

1: \textit{order} – Nag_OrderType \hspace{1cm} \textit{Input}

\textit{On entry}: the \textit{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 \textit{order} = \text{Nag_RowMajor}. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

\textit{Constraint}: \textit{order} = \text{Nag_RowMajor} or \text{Nag_ColMajor}.

2: \textit{uplo} – Nag_UploType \hspace{1cm} \textit{Input}

\textit{On entry}: indicates whether the upper or lower triangular part of \( A \) is stored and how \( A \) is factorized, as follows:

\begin{itemize}
  \item if \textit{uplo} = \text{Nag_Upper}, the upper triangular part of \( A \) is stored and \( A \) is factorized as \( U^T U \), where \( U \) is upper triangular;
  \item if \textit{uplo} = \text{Nag_Lower}, the lower triangular part of \( A \) is stored and \( A \) is factorized as \( LL^T \), where \( L \) is lower triangular.
\end{itemize}

\textit{Constraint}: \textit{uplo} = \text{Nag_Upper} or \text{Nag_Lower}.

3: \textit{n} – Integer \hspace{1cm} \textit{Input}

\textit{On entry}: \( n \), the order of the matrix \( A \).

\textit{Constraint}: \( n \geq 0 \).
4: \quad kd – Integer

*Input*

*On entry:* \( k \), the number of super-diagonals or sub-diagonals of the matrix \( A \).

*Constraint:* \( kd \geq 0 \).

5: \quad ab[dim] – double

*Input/Output*

*Note:* the dimension, \( dim \), of the array \( ab \) must be at least \( \max(1, pdab \times n) \).

*On entry:* the \( n \) by \( n \) symmetric band matrix \( A \). This is stored as a notional two-dimensional array with row elements or column elements stored contiguously. The storage of elements \( a_{ij} \) depends on the \( order \) and \( uplo \) parameters as follows:

- If \( order = \text{Nag}_\text{ColMajor} \) and \( uplo = \text{Nag}_\text{Upper} \),
  \( a_{ij} \) is stored in \( ab[k + i - j + (j - 1) \times pdab] \), for \( i = 1, \ldots, n \) and \( j = i, \ldots, \min(n, i + k) \);

- If \( order = \text{Nag}_\text{ColMajor} \) and \( uplo = \text{Nag}_\text{Lower} \),
  \( a_{ij} \) is stored in \( ab[i - j + (j - 1) \times pdab] \), for \( i = 1, \ldots, n \) and \( j = \max(1, i - k), \ldots, i \);

- If \( order = \text{Nag}_\text{RowMajor} \) and \( uplo = \text{Nag}_\text{Upper} \),
  \( a_{ij} \) is stored in \( ab[j - i + (i - 1) \times pdab] \), for \( i = 1, \ldots, n \) and \( j = i, \ldots, \min(n, i + k) \);

- If \( order = \text{Nag}_\text{RowMajor} \) and \( uplo = \text{Nag}_\text{Lower} \),
  \( a_{ij} \) is stored in \( ab[k + j - i + (i - 1) \times pdab] \), for \( i = 1, \ldots, n \) and \( j = \max(1, i - k), \ldots, i \).

*On exit:* the upper or lower triangle of \( A \) is overwritten by the Cholesky factor \( U \) or \( L \) as specified by \( uplo \), using the same storage format as described above.

6: \quad pdab – Integer

*Input*

*On entry:* the stride separating row or column elements (depending on the value of \( order \)) of the matrix \( A \) in the array \( ab \).

*Constraint:* \( pdab \geq kd + 1 \).

7: \quad fail – NagError *

*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:* \( kd = \langle \text{value} \rangle \).

*Constraint:* \( kd \geq 0 \).

*On entry:* \( pdab = \langle \text{value} \rangle \).

*Constraint:* \( pdab > 0 \).

**NE_INT_2**

*On entry:* \( pdab = \langle \text{value} \rangle, kd = \langle \text{value} \rangle \).

*Constraint:* \( pdab \geq kd + 1 \).

**NE_POS_DEF**

The matrix \( A \) is not positive definite.
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 factor $U$ is the exact factor of a perturbed matrix $A + E$, where

$$|E| \leq c(k+1)\varepsilon|U^T||U|,$$

$c(k+1)$ is a modest linear function of $k+1$, and $\varepsilon$ is the machine precision.

If uplo = Nag_Lower, a similar statement holds for the computed factor $L$. It follows that

$$|e_{ij}| \leq c(k+1)\varepsilon\sqrt{a_{ii}a_{jj}}.$$

8 Further Comments
The total number of floating-point operations is approximately $n(k+1)^2$, assuming $n \gg k$.

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

- nag_dpbtrs (f07hec) to solve $AX = B$;
- nag_dpbcon (f07hgc) to estimate the condition number of $A$.

The complex analogue of this function is nag_zpbtrf (f07hrc).

9 Example
To compute the Cholesky factorization of the matrix $A$, where

$$A = \begin{pmatrix} 5.49 & 2.68 & 0.00 & 0.00 \\ 2.68 & 5.63 & -2.39 & 0.00 \\ 0.00 & -2.39 & 2.60 & -2.22 \\ 0.00 & 0.00 & -2.22 & 5.17 \end{pmatrix}. $$

9.1 Program Text
/* nag_dpbtrf (f07hdc) Example Program. */
/* Copyright 2001 Numerical Algorithms Group. */
/* Mark 7, 2001. */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf07.h>
#include <nagx04.h>

int main(void)
{
    /* Scalars */
    Integer i, j, k, kd, n, pdab;
    Integer exit_status=0;
    Nag_UploType uplo_enum;
NagError fail;
Nag_OrderType order;
/* Arrays */
char uplo[2];
double *ab=0;

#ifdef NAG_COLUMN_MAJOR
#define AB_UPPER(I,J) ab[(J-1)*pdab + k + I - J - 1]
#define AB_LOWER(I,J) ab[(J-1)*pdab + I - J]
#else
#define AB_UPPER(I,J) ab[(I-1)*pdab + J - I]
#define AB_LOWER(I,J) ab[(I-1)*pdab + k + J - I - 1]
#endif
order = Nag_ColMajor;
#endif
INIT_FAIL(fail);
Vprintf("f07hdc Example Program Results\n\n");
/* Skip heading in data file */
Vscanf("%*[^n\n] ");
Vscanf("%ld%ld%*[^n\n] ", &n, &kd);
pdag = kd + 1;
/* Allocate memory */
if ( !(ab = NAG_ALLOC((kd+1) * n, double)) )
{
    Vprintf("Allocation failure\n");
    exit_status = -1;
    goto END;
}
/* Read A from data file */
Vscanf("' %1s ' %*[^n\n] ", uplo);
if (*(unsigned char *)uplo == 'L')
    uplo_enum = Nag_Lower;
else if (*(unsigned char *)uplo == 'U')
    uplo_enum = Nag_Upper;
else
{
    Vprintf("Unrecognised character for Nag_UploType type\n");
    exit_status = -1;
    goto END;
}
k = kd + 1;
if (uplo_enum == Nag_Upper)
{
    for (i = 1; i <= n; ++i)
    {
        for (j = i; j <= MIN(i+kd,n); ++j)
            Vscanf("%lf", &AB_UPPER(i,j));
    }
    Vscanf("%*[^n\n] ");
}
else
{
    for (i = 1; i <= n; ++i)
    {
        for (j = MAX(1,i-kd); j <= i; ++j)
            Vscanf("%lf", &AB_LOWER(i,j));
    }
    Vscanf("%*[^n\n] ");
}
/* Factorize A */
f07hdc(order, uplo_enum, n, kd, ab, pdab, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07hdc.\nn%s\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Print details of factorization */
if (uplo_enum == Nag_Upper)
  x04cec(order, n, n, 0, kd, ab, pdab, "Factor", 0, &fail);
else
  x04cec(order, n, n, kd, 0, ab, pdab, "Factor", 0, &fail);
if (fail.code != NE_NOERROR)
{
  Vprintf("Error from x04cec.\n%s\n", fail.message);
  exit_status = 1;
  goto END;
}
END:
if (ab) NAG_FREE(ab);
return exit_status;

9.2 Program Data

f07hdc Example Program Data

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Values of N and KD</th>
<th>Value of UPLO</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>5.49</td>
<td>'L'</td>
</tr>
<tr>
<td>2.68</td>
<td>5.63</td>
<td>-2.39</td>
<td>2.60</td>
</tr>
<tr>
<td>-2.22</td>
<td>5.17</td>
<td>:End of matrix A</td>
<td></td>
</tr>
</tbody>
</table>

9.3 Program Results

f07hdc Example Program Results

<table>
<thead>
<tr>
<th>Factor</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.3431</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.1438 2.0789</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-1.1497 1.1306</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-1.9635 1.1465</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>