NAG C Library Function Document

nag_dpptrf (f07gdc)

1 Purpose
nag_dpptrf (f07gdc) computes the Cholesky factorization of a real symmetric positive-definite matrix, using packed storage.

2 Specification

```c
void nag_dpptrf (Nag_OrderType order, Nag_UploType uplo, Integer n, double ap[], NagError *fail)
```

3 Description
nag_dpptrf (f07gdc) forms the Cholesky factorization of a real symmetric positive-definite matrix $A$ either as $A = U^T U$ if `uplo = Nag_Upper`, or $A = LL^T$ if `uplo = Nag_Lower`, where $U$ is an upper triangular matrix and $L$ is lower triangular, using packed storage.

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

5 Parameters
1: `order` – Nag_OrderType

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

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

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

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

3: `n` – Integer

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

Constraint: $n \geq 0$.

4: `ap[dim]` – double

Note: the dimension, $dim$, of the array `ap` must be at least max$(1, n \times (n + 1)/2)$.
On entry: the symmetric positive-definite matrix $A$, packed by rows or columns. The storage of elements $a_{ij}$ depends on the order and uplo parameters as follows:

- If order = Nag_ColMajor and uplo = Nag_Upper, $a_{ij}$ is stored in $ap[(j - 1) \times j/2 + i - 1]$, for $i \leq j$;
- If order = Nag_ColMajor and uplo = Nag_Lower, $a_{ij}$ is stored in $ap[(2n - j) \times (j - 1)/2 + i - 1]$, for $i \geq j$;
- If order = Nag_RowMajor and uplo = Nag_Upper, $a_{ij}$ is stored in $ap[(2n - i) \times (i - 1)/2 + j - 1]$, for $i \leq j$;
- If order = Nag_RowMajor and uplo = Nag_Lower, $a_{ij}$ is stored in $ap[(i - 1) \times i/2 + j - 1]$, for $i \geq j$.

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 packed storage format as described above.

5: fail – NagError *

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

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

Constraint: $n \geq 0$.

NE_POS_DEF

The leading minor of order $\langle \text{value} \rangle$ is not positive-definite and the factorization could not be completed. Hence $A$ itself is not positive-definite. This may indicate an error in forming the matrix $A$. To factorize a symmetric matrix which is not positive-definite, call nag_dsptrf (f07pdc) instead.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter $\langle \text{value} \rangle$ 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

$$|E| \leq c(n)\epsilon\|U\|\|U\|,$$

$c(n)$ is a modest linear function of $n$, and $\epsilon$ is the machine precision.

If uplo = Nag_Lower, a similar statement holds for the computed factor $L$. It follows that $|e_{ij}| \leq c(n)\epsilon\sqrt{a_{ii}a_{jj}}$.

8 Further Comments

The total number of floating-point operations is approximately $\frac{1}{3}n^3$.

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

f07gdc
nag_dpptrs (f07gec) to solve $AX = B$;
nag_dppcon (f07ggc) to estimate the condition number of $A$;
nag_dpptri (f07gjc) to compute the inverse of $A$.

The complex analogue of this function is nag_zpptrf (f07grc).

9 Example

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

$$A = \begin{pmatrix}
4.16 & -3.12 & 0.56 & -0.10 \\
-3.12 & 5.03 & -0.83 & 1.18 \\
0.56 & -0.83 & 0.76 & 0.34 \\
-0.10 & 1.18 & 0.34 & 1.18
\end{pmatrix},$$

using packed storage.

9.1 Program Text

/* nag_dpptrf (f07gdc) 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 ap_len, i, j, n;
   Integer exit_status=0;
   Nag_UploType uplo_enum;
   NagError fail;
   Nag_OrderType order;

   /* Arrays */
   char uplo[2];
   double *ap=0;

   #ifdef NAG_COLUMN_MAJOR
   #define A_UPPER(I,J) ap[J*(J-1)/2 + I - 1]
   #define A_LOWER(I,J) ap[(2*n-J)*(J-1)/2 + I - 1]
   #else
   #define A_LOWER(I,J) ap[I*(I-1)/2 + J - 1]
   #define A_UPPER(I,J) ap[(2*n-I)*(I-1)/2 + J - 1]
   #endif

   INIT_FAIL(fail);
   Vprintf("f07gdc Example Program Results\n\n");

   /* Skip heading in data file */
   Vscanf("%*[
"");
   Vscanf("%d%*[
"");
   ap_len = n*(n+1)/2;

   /* Allocate memory */
   if ( !(ap = NAG_ALLOC(ap_len, double)) )
   {
      Vprintf("Allocation failure\n");
   }

   return 0;
}
exit_status = -1;
goto END;
}
/* Read A from data file */
Vscanf("' %ls '\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;
}
if (uplo_enum == Nag_Upper)
{
    for (i = 1; i <= n; ++i)
    {
        for (j = i; j <= n; ++j)
            Vscanf("%lf", &A_UPPER(i,j));
    }
    Vscanf("%*\n ");
}
else
{
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= i; ++j)
            Vscanf("%lf", &A_LOWER(i,j));
    }
    Vscanf("%*\n ");
}
/* Factorize A */
f07gdc(order, uplo_enum, n, ap, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07gdc.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Print factor */
x04ccc(order, uplo_enum, Nag_NonUnitDiag, n, ap,
    "Factor", 0, NAGERR_DEFAULT);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from x04ccc.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
END:
if (ap) NAG_FREE(ap);
return exit_status;

9.2 Program Data
f07gdc Example Program Data
4 : Value of N
'L' : Value of UPLO
4.16
-3.12 5.03
0.56 -0.83 0.76
-0.10 1.18 0.34 1.18 : End of matrix A
## 9.3 Program Results

**f07gdc 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>2.0396</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-1.5297</td>
<td>1.6401</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.2746</td>
<td>-0.2500</td>
<td>0.7887</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-0.0490</td>
<td>0.6737</td>
<td>0.6617</td>
<td>0.5347</td>
</tr>
</tbody>
</table>