NAG Library Function Document

nag_dtptrs (f07uec)

1 Purpose
nag_dtptrs (f07uec) solves a real triangular system of linear equations with multiple right-hand sides, \( AX = B \) or \( A^T X = B \), using packed storage.

2 Specification
#include <nag.h>
#include <nagf07.h>

void nag_dtptrs (Nag_OrderType order, Nag_UploType uplo,
                 Nag_TransType trans, Nag_DiagType diag, Integer n, Integer nrhs,
                 const double ap[], double b[], Integer pdb, NagError *fail)

3 Description
nag_dtptrs (f07uec) solves a real triangular system of linear equations \( AX = B \) or \( A^T X = B \), using packed storage.

4 References

5 Arguments
1: order – Nag_OrderType
   
   On entry: the order argument 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 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.
   
   Constraint: order = Nag_RowMajor or Nag_ColMajor.

2: uplo – Nag_UploType
   
   On entry: specifies whether \( A \) is upper or lower triangular.
   
   uplo = Nag_Upper
   \( A \) is upper triangular.
   
   uplo = Nag_Lower
   \( A \) is lower triangular.
   
   Constraint: uplo = Nag_Upper or Nag_Lower.

3: trans – Nag_TransType
   
   On entry: indicates the form of the equations.
   
   trans = Nag_NoTrans
   The equations are of the form \( AX = B \).
The equations are of the form $A^T X = B$.

**Constraint:** $\text{trans} = \text{Nag\_NoTrans}, \text{Nag\_Trans} \text{ or } \text{Nag\_ConjTrans}$.

4: $\text{diag} = \text{Nag\_DiagType}$

*Input*

*On entry:* indicates whether $A$ is a nonunit or unit triangular matrix.

- **diag** = Nag\_NonUnitDiag
  - $A$ is a nonunit triangular matrix.

- **diag** = Nag\_UnitDiag
  - $A$ is a unit triangular matrix; the diagonal elements are not referenced and are assumed to be 1.

**Constraint:** $\text{diag} = \text{Nag\_NonUnitDiag} \text{ or } \text{Nag\_UnitDiag}$.

5: $n$ – Integer

*Input*

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

**Constraint:** $n \geq 0$.

6: $\text{nrhs}$ – Integer

*Input*

*On entry:* $r$, the number of right-hand sides.

**Constraint:** $\text{nrhs} \geq 0$.

7: $\text{ap}[\text{dim}]$ – const double

*Input*

*Note:* the dimension, $\text{dim}$, of the array $\text{ap}$ must be at least $\max(1, n \times (n + 1)/2)$.

*On entry:* the $n$ by $n$ triangular matrix $A$, packed by rows or columns.

The storage of elements $A_{ij}$ depends on the $\text{order}$ and $\text{uplo}$ arguments as follows:

- if $\text{order} = \text{Nag\_ColMajor}$ and $\text{uplo} = \text{Nag\_Upper}$, $A_{ij}$ is stored in $\text{ap}[(j - 1) \times j/2 + i - 1]$, for $i \leq j$;
- if $\text{order} = \text{Nag\_ColMajor}$ and $\text{uplo} = \text{Nag\_Lower}$, $A_{ij}$ is stored in $\text{ap}[(2n - j) \times (j - 1)/2 + i - 1]$, for $i \geq j$;
- if $\text{order} = \text{Nag\_RowMajor}$ and $\text{uplo} = \text{Nag\_Upper}$, $A_{ij}$ is stored in $\text{ap}[(2n - i) \times (i - 1)/2 + j - 1]$, for $i \leq j$;
- if $\text{order} = \text{Nag\_RowMajor}$ and $\text{uplo} = \text{Nag\_Lower}$, $A_{ij}$ is stored in $\text{ap}[(i - 1) \times i/2 + j - 1]$, for $i \geq j$.

If $\text{diag} = \text{Nag\_UnitDiag}$, the diagonal elements of $A$ are assumed to be 1, and are not referenced; the same storage scheme is used whether $\text{diag} = \text{Nag\_NonUnitDiag}$ or $\text{diag} = \text{Nag\_UnitDiag}$.

8: $\text{b}[\text{dim}]$ – double

*Input/Output*

*Note:* the dimension, $\text{dim}$, of the array $\text{b}$ must be at least $\max(1, \text{pdb} \times \text{nrhs})$ when $\text{order} = \text{Nag\_ColMajor}$;

$\max(1, n \times \text{pdb})$ when $\text{order} = \text{Nag\_RowMajor}$.

The $(i, j)$th element of the matrix $B$ is stored in

- $\text{b}[(j - 1) \times \text{pdb} + i - 1]$ when $\text{order} = \text{Nag\_ColMajor}$;
- $\text{b}[(i - 1) \times \text{pdb} + j - 1]$ when $\text{order} = \text{Nag\_RowMajor}$.

*On entry:* the $n$ by $r$ right-hand side matrix $B$.

*On exit:* the $n$ by $r$ solution matrix $X$. 
9: \[ \texttt{pdb} - \text{Integer} \]

\[\text{Input}\]

On entry: the stride separating row or column elements (depending on the value of \texttt{order}) in the array \texttt{b}.

Constraints:

\[
\begin{align*}
&\text{if } \texttt{order} = \text{Nag\_ColMajor}, \texttt{pdb} \geq \max(1, \texttt{n}); \\
&\text{if } \texttt{order} = \text{Nag\_RowMajor}, \texttt{pdb} \geq \max(1, \texttt{nrhs}).
\end{align*}
\]

10: \[ \texttt{fail} - \text{NagError}^* \]

\[\text{Input/Output}\]

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

6 \hspace{1em} \textbf{Error Indicators and Warnings}

\textbf{NE\_ALLOC\_FAIL}

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

\textbf{NE\_BAD\_PARAM}

On entry, argument \(\langle\text{value}\rangle\) had an illegal value.

\textbf{NE\_INT}

On entry, \(\texttt{n} = \langle\text{value}\rangle\).
Constraint: \(\texttt{n} \geq 0\).

On entry, \(\texttt{nrhs} = \langle\text{value}\rangle\).
Constraint: \(\texttt{nrhs} \geq 0\).

On entry, \(\texttt{pdb} = \langle\text{value}\rangle\).
Constraint: \(\texttt{pdb} > 0\).

\textbf{NE\_INT\_2}

On entry, \(\texttt{pdb} = \langle\text{value}\rangle\) and \(\texttt{n} = \langle\text{value}\rangle\).
Constraint: \(\texttt{pdb} \geq \max(1, \texttt{n})\).

On entry, \(\texttt{pdb} = \langle\text{value}\rangle\) and \(\texttt{nrhs} = \langle\text{value}\rangle\).
Constraint: \(\texttt{pdb} \geq \max(1, \texttt{nrhs})\).

\textbf{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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

\textbf{NE\_NO\_LICENCE}

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

\textbf{NE\_SINGULAR}

Element \(\langle\text{value}\rangle\) of the diagonal is exactly zero. \(A\) is singular and the solution has not been computed.
7 Accuracy

The solutions of triangular systems of equations are usually computed to high accuracy. See Higham (1989).

For each right-hand side vector \( b \), the computed solution \( x \) is the exact solution of a perturbed system of equations \( (A + E)x = b \), where

\[
|E| \leq c(n)\epsilon|A|,
\]

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

If \( \hat{x} \) is the true solution, then the computed solution \( x \) satisfies a forward error bound of the form

\[
\frac{\|x - \hat{x}\|_\infty}{\|x\|_\infty} \leq c(n)\text{cond}(A, x)\epsilon, \quad \text{provided } c(n)\text{cond}(A, x)\epsilon < 1,
\]

where \( \text{cond}(A, x) = \|[A^{-1}||A||x]\|_\infty /\|x\|_\infty \).

Note that \( \text{cond}(A, x) \leq \text{cond}(A) = \|[A^{-1}||A]\|_\infty \leq \kappa_\infty(A) \); \( \text{cond}(A, x) \) can be much smaller than \( \text{cond}(A) \) and it is also possible for \( \text{cond}(A^T) \) to be much larger (or smaller) than \( \text{cond}(A) \).

Forward and backward error bounds can be computed by calling nag_dtpfors (f07uhc), and an estimate for \( \kappa_\infty(A) \) can be obtained by calling nag.dtpcon (f07ugc) with \( \text{norm} = \text{Nag\_InfNorm} \).

8 Parallelism and Performance

\text{nag\_dtptrs} (f07uec) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

\text{nag\_dtptrs} (f07uec) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.

9 Further Comments

The total number of floating-point operations is approximately \( n^2r \).

The complex analogue of this function is \text{nag\_ztptrs} (f07usec).

10 Example

This example solves the system of equations \( AX = B \), where

\[
A = \begin{pmatrix}
4.30 & 0.00 & 0.00 & 0.00 \\
-3.96 & -4.87 & 0.00 & 0.00 \\
0.40 & 0.31 & -8.02 & 0.00 \\
-0.27 & 0.07 & -5.95 & 0.12
\end{pmatrix}
\quad \text{and} \quad
B = \begin{pmatrix}
-12.90 & -21.50 \\
16.75 & 14.93 \\
-17.55 & 6.33 \\
-11.04 & 8.09
\end{pmatrix},
\]

using packed storage for \( A \).

10.1 Program Text

/* \text{nag\_dtptrs} (f07uec) Example Program. */
/* Copyright 2014 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, nrhs, pdb;
    Integer exit_status = 0;
    Nag_UploType uplo;
    NagError fail;
    Nag_OrderType order;
    /* Arrays */
    char nag_enum_arg[40];
    double *ap = 0, *b = 0;

    #ifdef NAG_LOAD_FP
    /* The following line is needed to force the Microsoft linker
     * to load floating point support */
    float force_loading_of_ms_float_support = 0;
    #endif /* NAG_LOAD_FP */
    #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]
    #define B(I, J) b[(J-1)*pdb + I - 1]
    order = Nag_ColMajor;
    #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]
    #define B(I, J) b[(I-1)*pdb + J - 1]
    order = Nag_RowMajor;
    #endif
    INIT_FAIL(fail);
    printf("nag_dtptrs (f07uec) Example Program Results\n\n");
    /* Skip heading in data file */
    #ifdef _WIN32
    scanf_s("%*[^\n] ");
    #else
    scanf("%*[^\n] ");
    #endif
    #ifdef _WIN32
    scanf_s("%"NAG_IFMT"%"NAG_IFMT"%[^\n] ", &n, &nrhs);
    #else
    scanf("%"NAG_IFMT"%"NAG_IFMT"%[^\n] ", &n, &nrhs);
    #endif
    ap_len = n*(n+1)/2;
    #ifdef NAG_COLUMN_MAJOR
    pdb = n;
    #else
    pdb = nrhs;
    #endif
    /* Allocate memory */
    if (!(ap = NAG_ALLOC(ap_len, double)) ||
        !(b = NAG_ALLOC(n * nrhs, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }
    /* Read A and B from data file */
    #ifdef _WIN32
    scanf_s(" %39s %*[^\n] ", nag_enum_arg, _countof(nag_enum_arg));
    #else
    scanf(" %39s [%\n] ", nag_enum_arg);
    /* Read A and B from data file */
    #ifdef _WIN32
    scanf_s(" %39s [%\n] ", nag_enum_arg, _countof(nag_enum_arg));
    #else
    scanf(" %39s [%\n] ", nag_enum_arg);
    Mark 25
    "}
/* nag_enum_name_to_value (x04nac).
* Converts NAG enum member name to value
*/
uplo = (Nag_UploType) nag_enum_name_to_value(nag_enum_arg);
if (uplo == Nag_Upper)
{
    for (i = 1; i <= n; ++i)
    {
        for (j = i; j <= n; ++j)
            if (_WIN32)
                scanf_s("%lf", &A_UPPER(i, j));
            else
                scanf("%lf", &A_UPPER(i, j));
    }
    /* Compute solution */
    /* nag_dtptrs (f07uec).
    * Solution of real triangular system of linear equations,
    * multiple right-hand sides, packed storage
    */
    nag_dtptrs(order, uplo, Nag_NoTrans, Nag_NoUnitDiag, n, nrhs, ap, b, pdb, &fail);
    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_dtptrs (f07uec).\n", fail.message);
        exit_status = 1;
        goto END;
    }
    /* Print solution */
    /* nag_gen_real_mat_print (x04cac).
    * Print real general matrix (easy-to-use)
fflush(stdout);

nag_gen_real_mat_print(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, nrhs, b, pdb, "Solution(s)", 0, &fail);

if (fail.code != NE_NOERROR)
{
    printf("Error from nag_gen_real_mat_print (x04cac).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

END:

NAG_FREE(ap);
NAG_FREE(b);

return exit_status;

10.2 Program Data

nag_dtptrs (f07uec) Example Program Data

4 2 :Values of n and nrhs
Nag_Lower :Value of uplo
4.30
-3.96 -4.87
0.40 0.31 -8.02
-0.27 0.07 -5.95 0.12 :End of matrix A
-12.90 -21.50
16.75 14.93
-17.55 6.33
-11.04 8.09 :End of matrix B

10.3 Program Results

nag_dtptrs (f07uec) Example Program Results

Solution(s)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-3.0000</td>
</tr>
<tr>
<td>2</td>
<td>-1.0000</td>
</tr>
<tr>
<td>3</td>
<td>2.0000</td>
</tr>
<tr>
<td>4</td>
<td>1.0000</td>
</tr>
</tbody>
</table>