NAG C Library Function Document

nag_dgetrs (f07aec)

1 Purpose

nag_dgetrs (f07aec) solves a real system of linear equations with multiple right-hand sides, $AX = B$ or $A^TX = B$, where $A$ has been factorized by nag_dgetrf (f07adc).

2 Specification

```c
void nag_dgetrs (Nag_OrderType order, Nag_TransType trans, Integer n, Integer nrhs,
const double a[], Integer pda, const Integer ipiv[], double b[], Integer pdb,
    NagError *fail)
```

3 Description

To solve a real system of linear equations $AX = B$ or $A^TX = B$, this function must be preceded by a call to nag_dgetrf (f07adc) which computes the $LU$ factorization of $A$ as $A = PLU$. The solution is computed by forward and backward substitution.

If $trans = \text{Nag\_NoTrans}$, the solution is computed by solving $PLY = B$ and then $UX = Y$.

If $trans = \text{Nag\_Trans}$ or $\text{Nag\_ConjTrans}$, the solution is computed by solving $U^TY = B$ and then $L^TP^TX = Y$.

4 References


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: trans – Nag_TransType  
   On entry: indicates the form of the equations as follows:
   - if $trans = \text{Nag\_NoTrans}$, $AX = B$ is solved for $X$;
   - if $trans = \text{Nag\_Trans}$ or $\text{Nag\_ConjTrans}$, $A^TX = B$ is solved for $X$.
   Constraint: $trans = \text{Nag\_NoTrans}$, Nag_Trans or Nag_ConjTrans.

3: n – Integer  
   On entry: $n$, the order of the matrix $A$.
   Constraint: $n \geq 0$. 
4: \textbf{nrhs} – Integer 
\textbf{Input}

On entry: \( r \), the number of right-hand sides.

\textit{Constraint: nrhs} \( \geq 0 \).

5: \textbf{a} \([\text{dim}]\) – const double 
\textbf{Input}

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

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

On entry: the \( LU \) factorization of \( A \), as returned by nag\_dgetrf (f07adc).

6: \textbf{pda} – Integer 
\textbf{Input}

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

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

7: \textbf{ipiv} \([\text{dim}]\) – const Integer 
\textbf{Input}

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

On entry: the pivot indices, as returned by nag\_dgetrf (f07adc).

8: \textbf{b} \([\text{dim}]\) – double 
\textbf{Input/Output}

\textit{Note:} the dimension, \textit{dim}, of the array \textbf{b} must be at least \( \max(1, \text{pdb} \times \text{nrhs}) \) when \textit{order} = \textit{Nag\_ColMajor} and at least \( \max(1, \text{pdb} \times n) \) when \textit{order} = \textit{Nag\_RowMajor}.

If \textit{order} = \textit{Nag\_ColMajor}, the \((i,j)\)th element of the matrix \( B \) is stored in \( b[(j - 1) \times \text{pdb} + i - 1] \) and if \textit{order} = \textit{Nag\_RowMajor}, the \((i,j)\)th element of the matrix \( B \) is stored in \( b[(i - 1) \times \text{pdb} + j - 1] \).

On entry: the \( n \) by \( r \) right-hand side matrix \( B \).

On exit: the \( n \) by \( r \) solution matrix \( X \).

9: \textbf{pdb} – Integer 
\textbf{Input}

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

\textit{Constraints:}

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

10: \textbf{fail} – NagError * 
\textbf{Output}

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

\textbf{NE\_INT}

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

\textit{Constraint: n} \( \geq 0 \).

On entry, \textbf{nrhs} = \langle \text{value} \rangle.

\textit{Constraint: nrhs} \( \geq 0 \).

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

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

On entry, \textbf{pdb} = \langle \text{value} \rangle.

\textit{Constraint: pdb} \( > 0 \).
NE_INT_2

On entry, pda = \langle value \rangle, n = \langle value \rangle.
Constraint: pda \geq \max(1, n).

On entry, pdb = \langle value \rangle, n = \langle value \rangle.
Constraint: pdb \geq \max(1, n).

On entry, pdb = \langle value \rangle, nrhs = \langle value \rangle.
Constraint: pdb / C21 max(1, nrhs).

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter \langle 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

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
\[
|E| \leq c(n)\epsilon|P| |L| |U|,
\]
\( c(n) \) 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
\[
\frac{\|x - \hat{x}\|_\infty}{\|x\|_\infty} \leq c(n) \text{cond}(A, x)\epsilon
\]
where \( \text{cond}(A, x) = ||A^{-1}|| |A||x||_\infty / ||x||_\infty \leq \text{cond}(A) = ||A^{-1}|| |A||_\infty \leq \kappa_\infty(A) \). Note that \( \text{cond}(A, x) \) can be much smaller than \( \text{cond}(A) \), and \( \text{cond}(A^T) \) can be much larger (or smaller) than \( \text{cond}(A) \).
Forward and backward error bounds can be computed by calling \( \text{nag_dgerfs (f07ahc)} \), and an estimate for \( \kappa_\infty(A) \) can be obtained by calling \( \text{nag_dgecon (f07agc)} \) with \( \text{norm = Nag_InfNorm} \).

8 Further Comments

The total number of floating-point operations is approximately \( 2n^2r \).
This function may be followed by a call to \( \text{nag_dgerfs (f07ahc)} \) to refine the solution and return an error estimate.

The complex analogue of this function is \( \text{nag_zgetrs (f07asc)} \).

9 Example

To solve the system of equations \( AX = B \), where
\[
A = \begin{pmatrix}
  1.80 & 2.88 & 2.05 & -0.89 \\
  5.25 & -2.95 & -0.95 & -3.80 \\
  1.58 & -2.69 & -2.90 & -1.04 \\
 -1.11 & -0.66 & -0.59 & 0.80
\end{pmatrix}
\quad \text{and} \quad
B = \begin{pmatrix}
  9.52 & 18.47 \\
  24.35 & 2.25 \\
  0.77 & -13.28 \\
 -6.22 & -6.21
\end{pmatrix}.
\]
Here \( A \) is nonsymmetric and must first be factorized by \( \text{nag_dgetrf (f07adc)} \).
9.1 Program Text

/* nag_dgetrs (f07aec) 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, n, nrhs, pda, pdb;
    Integer exit_status=0;
    NagError fail;
    Nag_OrderType order;

    /* Arrays */
    double *a=0, *b=0;
    Integer *ipiv=0;

    /* Skip heading in data file */
    Vprintf("f07aec Example Program Results\n\n");

    /* Read A and B from data file */
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= n; ++j)
            Vscanf("%lf", &A(i,j));
    }
    Vscanf("%*[^\n] ");
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= n; ++j)
            Vscanf("%lf", &A(i,j));
    }
}
for (j = 1; j <= nrhs; ++j)
    Vscanf("%lf", &B(i,j));

Vscanf("%*[\n]");

/* Factorize A */
f07adc(order, n, n, a, pda, ipiv, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07adc.
%s
", fail.message);
    exit_status = 1;
    goto END;
}

/* Compute solution */
f07aec(order, Nag_NoTrans, n, nrhs, a, pda, ipiv, b, pdb, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07aec.
%s
", fail.message);
    exit_status = 1;
    goto END;
}

/* Print solution */
x04cac(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, nrhs, b, pdb, "Solution(s)", 0, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from x04cac.
%s
", fail.message);
    exit_status = 1;
    goto END;
}

END:
if (a) NAG_FREE(a);
if (b) NAG_FREE(b);
if (ipiv) NAG_FREE(ipiv);
return exit_status;

9.2 Program Data
f07aec Example Program Data

<table>
<thead>
<tr>
<th>4</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.80</td>
<td>2.88</td>
</tr>
<tr>
<td>5.25</td>
<td>-2.95</td>
</tr>
<tr>
<td>1.58</td>
<td>-2.69</td>
</tr>
<tr>
<td>-1.11</td>
<td>-0.66</td>
</tr>
<tr>
<td>9.52</td>
<td>18.47</td>
</tr>
<tr>
<td>24.35</td>
<td>2.25</td>
</tr>
<tr>
<td>0.77</td>
<td>-13.28</td>
</tr>
<tr>
<td>-6.22</td>
<td>-6.21</td>
</tr>
</tbody>
</table>

:Values of N and NRHS

| :End of matrix A |
| :End of matrix B |

9.3 Program Results
f07aec Example Program Results

Solution(s)

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
</tr>
<tr>
<td>2</td>
<td>-1.0000</td>
</tr>
<tr>
<td>3</td>
<td>3.0000</td>
</tr>
<tr>
<td>4</td>
<td>-5.0000</td>
</tr>
</tbody>
</table>