

NAG C Library Function Document

nag_dgetri (f07ajc)

1 Purpose

nag_dgetri (f07ajc) computes the inverse of a real matrix A , where A has been factorized by nag_dgetrf (f07adc).

2 Specification

```
void nag_dgetri (Nag_OrderType order, Integer n, double a[], Integer pda,
                 const Integer ipiv[], NagError *fail)
```

3 Description

To compute the inverse of a real matrix A , the function must be preceded by a call to nag_dgetrf (f07adc), which computes the LU factorization of A as $A = PLU$. The inverse of A is computed by forming U^{-1} and then solving the equation $XPL = U^{-1}$ for X .

4 References

Du Croz J J and Higham N J (1992) Stability of methods for matrix inversion *IMA J. Numer. Anal.* **12** 1–19

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: **n** – Integer *Input*
On entry: n , the order of the matrix A .
Constraint: $n \geq 0$.
- 3: **a**[dim] – double *Input/Output*
Note: the dimension, dim , of the array **a** must be at least $\max(1, pda \times n)$.
 If **order** = **Nag_ColMajor**, the (i, j) th element of the matrix A is stored in **a**[($j - 1$) $\times$ **pda** + $i - 1$] and if **order** = **Nag_RowMajor**, the (i, j) th element of the matrix A is stored in **a**[($i - 1$) $\times$ **pda** + $j - 1$].
On entry: the LU factorization of A , as returned by nag_dgetrf (f07adc).
On exit: the factorization is overwritten by the n by n matrix A^{-1} .
- 4: **pda** – Integer *Input*
On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **a**.
Constraint: **pda** $\geq \max(1, n)$.

- 5: **ipiv**[*dim*] – const Integer *Input*
Note: the dimension, *dim*, of the array **ipiv** must be at least $\max(1, \mathbf{n})$.
On entry: the pivot indices, as returned by nag_dgetrf (f07adc).
- 6: **fail** – NagError * *Output*
The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** = $\langle value \rangle$.

Constraint: **n** ≥ 0 .

On entry, **pda** = $\langle value \rangle$.

Constraint: **pda** > 0 .

NE_INT_2

On entry, **pda** = $\langle value \rangle$, **n** = $\langle value \rangle$.

Constraint: **pda** $\geq \max(1, \mathbf{n})$.

NE_SINGULAR

Element $\langle value \rangle$ of the diagonal is zero. *U* is singular, and the inverse of *A* cannot be computed.

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

The computed inverse *X* satisfies a bound of the form:

$$|XA - I| \leq c(n)\epsilon|X|P|L||U|,$$

where $c(n)$ is a modest linear function of *n*, and ϵ is the *machine precision*.

Note that a similar bound for $|AX - I|$ cannot be guaranteed, although it is almost always satisfied. See Du Croz and Higham (1992).

8 Further Comments

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

The complex analogue of this function is nag_zgetri (f07awc).

9 Example

To compute the inverse of the matrix A , where

$$A = \begin{pmatrix} 1.80 & 2.88 & 2.05 & -0.89 \\ 5.25 & -2.95 & -0.95 & -3.80 \\ 1.58 & -2.69 & -2.90 & -1.04 \\ -1.11 & -0.66 & -0.59 & 0.80 \end{pmatrix}.$$

Here A is nonsymmetric and must first be factorized by nag_dgetrf (f07adc).

9.1 Program Text

```
/* nag_dgetri (f07ajc) 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, ipiv_len, j, n, pda;
    Integer exit_status=0;
    NagError fail;
    Nag_OrderType order;
    /* Arrays */
    double *a=0;
    Integer *ipiv=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("f07ajc Example Program Results\n\n");

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

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

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

    /* Read A from data file */
```

```

for (i = 1; i <= n; ++i)
{
    for (j = 1; j <= n; ++j)
        Vscanf("%lf", &A(i,j));
}
Vscanf("%*[^\\n] ");

/* Factorize A */
f07adc(order, n, n, a, pda, ipiv, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07adc.%s\\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Compute inverse of A */
f07ajc(order, n, a, pda, ipiv, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07ajc.%s\\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print inverse */
x04cac(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, n, a, pda,
        "Inverse", 0, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from x04cac.%s\\n", fail.message);
    exit_status = 1;
    goto END;
}
END:
if (a) NAG_FREE(a);
if (ipiv) NAG_FREE(ipiv);

return exit_status;
}

```

9.2 Program Data

```

f07ajc Example Program Data
4                               :Value of N
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 of matrix A

```

9.3 Program Results

f07ajc Example Program Results

Inverse	1	2	3	4
1	1.7720	0.5757	0.0843	4.8155
2	-0.1175	-0.4456	0.4114	-1.7126
3	0.1799	0.4527	-0.6676	1.4824
4	2.4944	0.7650	-0.0360	7.6119
