NAG Library Function Document

nag_zgetri (f07awc)

1 Purpose

nag_zgetri (f07awc) computes the inverse of a complex matrix \( A \), where \( A \) has been factorized by nag_zgetrf (f07arc).

2 Specification

```c
#include <nag.h>
#include <nagf07.h>

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

3 Description

nag_zgetri (f07awc) is used to compute the inverse of a complex matrix \( A \), the function must be preceded by a call to nag_zgetrf (f07arc), 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


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: n – Integer

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

   Constraint: \( n \geq 0 \).

3: a[dim] – Complex

   On entry: the \( (i,j) \)th element of the matrix \( A \) is stored in
   \[ a[(j - 1) \times \text{pda} + i - 1] \] when order = Nag_ColMajor;
   \[ a[(i - 1) \times \text{pda} + j - 1] \] when order = Nag_RowMajor.

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

   On exit: the \( LU \) factorization of \( A \), as returned by nag_zgetrf (f07arc).

   On exit: the factorization is overwritten by the \( n \) by \( n \) matrix \( A^{-1} \).
4:   pda – Integer

    On entry: the stride separating row or column elements (depending on the value of order) in the array a.

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

5:   ipiv[\text{dim}] – const Integer

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

    On entry: the pivot indices, as returned by \text{nag_zgetrf} (f07arc).

6:   fail – NagError*

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

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM

On entry, argument \langle value\rangle had an illegal value.

NE_INT

On entry, n = \langle value\rangle.

    Constraint: n \geq 0.

On entry, pda = \langle value\rangle.

    Constraint: pda > 0.

NE_INT_2

On entry, pda = \langle value\rangle and n = \langle value\rangle.

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

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.

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.

NE_SINGULAR

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

7 Accuracy

The computed inverse X satisfies a bound of the form:

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

where c(n) is a modest linear function of n, and \varepsilon is the \text{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 Parallelism and Performance

nag_zgetri (f07awc) is not threaded by NAG in any implementation.

nag_zgetri (f07awc) 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 real floating-point operations is approximately $\frac{16}{3}n^3$.

The real analogue of this function is nag_dgetri (f07ajc).

## 10 Example

This example computes the inverse of the matrix $A$, where

$$
A = \begin{pmatrix}
-1.34 + 2.55i & 0.28 + 3.17i & -6.39 - 2.20i & 0.72 - 0.92i \\
-0.17 - 1.41i & 3.31 - 0.15i & -0.15 + 1.34i & 1.29 + 1.38i \\
-3.29 - 2.39i & -1.91 + 4.42i & -0.14 - 1.35i & 1.72 + 1.35i \\
2.41 + 0.39i & -0.56 + 1.47i & -0.83 - 0.69i & -1.96 + 0.67i
\end{pmatrix}.
$$

Here $A$ is nonsymmetric and must first be factorized by nag_zgetrf (f07arc).

### 10.1 Program Text

```c
/* nag_zgetri (f07awc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 7, 2001. */

#include <stdio.h>
#include <nag.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 */
    Complex *a = 0;
    Integer *ipiv = 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(I, J) a[(J-1)*pda + I - 1]
    order = Nag_ColMajor;
    #endif

    a[0] = -1.34 + 2.55i;
    a[1] = 0.28 + 3.17i;
    a[2] = -6.39 - 2.20i;
    a[3] = 0.72 - 0.92i;
    a[4] = -0.17 - 1.41i;
    a[5] = 3.31 - 0.15i;
    a[6] = -0.15 + 1.34i;
    a[7] = 1.29 + 1.38i;
    a[8] = -3.29 - 2.39i;
    a[9] = -1.91 + 4.42i;
    a[10] = -0.14 - 1.35i;
    a[11] = 1.72 + 1.35i;
    a[12] = 2.41 + 0.39i;
    a[13] = -0.56 + 1.47i;
    a[14] = -0.83 - 0.69i;
    a[15] = -1.96 + 0.67i;

    // ... (rest of the program) ...

    return exit_status;
}
```
#define A(I, J) a[(I-1)*pda +J - 1]
order = Nag_RowMajor;
#endif

INIT_FAIL(fail);

/* Skip heading in data file */
#endif_WIN32
scanf_s("%*[\n ]");
#else
scanf("%*[\n ]");
#endif_WIN32
scanf_s("%NAG_IFMT%*[\n ]", &n);
#else
scanf("%NAG_IFMT%*[\n ]", &n);
#endif_WIN32
#define NAG_COLUMN_MAJOR
pda = n;
#else
pda = n;
#endif_WIN32
ipiv_len = n;

/* Allocate memory */
if (!(a = NAG_ALLOC(n * n, Complex)) ||
   !(ipiv = NAG_ALLOC(ipiv_len, Integer)))
{
    printf("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)
        #ifdef _WIN32
            scanf_s(" ( %lf , %lf )", &A(i, j).re, &A(i, j).im);
        #else
            scanf(" ( %lf , %lf )", &A(i, j).re, &A(i, j).im);
        #endif
    #ifdef _WIN32
        scanf_s("%*[\n ]");
    #else
        scanf("%*[\n ]");
    #endif
}
/* Factorize A */
#endif.Win32
nag_zgetrf(order, n, n, a, pda, ipiv, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_zgetrf (f07arc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Compute inverse of A */
#endif.Win32
nag_zgetri(order, n, a, pda, ipiv, &fail);
if (fail.code != NE_NOERROR)
printf("Error from nag_zgetri (f07awc)\n", fail.message);
exit_status = 1;
goto END;
}

/* Print inverse */
/* nag_gen_complx_mat_print_comp (x04dbc).
* Print complex general matrix (comprehensive)
*/
fflush(stdout);
nag_gen_complx_mat_print_comp(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n,
n, a, pda, NagBracketForm, "%7.4f", "Inverse",
Nag_IntegerLabels, 0, Nag_IntegerLabels,
0, 80, 0, 0, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_gen_complx_mat_print_comp (x04dbc)\n", fail.message);
    exit_status = 1;
goto END;
}
END:
NAG_FREE(a);
NAG_FREE(ipiv);
return exit_status;

10.2 Program Data

nag_zgetri (f07awc) Example Program Data

<table>
<thead>
<tr>
<th>Value of N</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-1.34, 2.55) ( 0.28, 3.17) (-6.39,-2.20) ( 0.72,-0.92)</td>
</tr>
<tr>
<td>(-0.17,-1.41) ( 3.31,-0.15) ( 0.15, 1.34) ( 1.29, 1.38)</td>
</tr>
<tr>
<td>(-3.29,-2.39) (-1.91, 4.42) (-0.14,-1.35) ( 1.72, 1.35)</td>
</tr>
<tr>
<td>( 2.41, 0.39) (-0.56, 1.47) (-0.83,-0.69) (-1.96, 0.67)</td>
</tr>
</tbody>
</table>

10.3 Program Results

nag_zgetri (f07awc) Example Program Results

<table>
<thead>
<tr>
<th>Inverse</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( 0.0757,-0.4324)</td>
<td>( 1.6512,-3.1342)</td>
<td>( 1.2663, 0.0418)</td>
<td>( 3.8181, 1.1195)</td>
</tr>
<tr>
<td>2</td>
<td>(-0.1942, 0.0798)</td>
<td>(-1.1900,-0.1426)</td>
<td>(-0.2401,-0.5889)</td>
<td>(-0.0101,-1.4969)</td>
</tr>
<tr>
<td>3</td>
<td>(-0.0957,-0.0491)</td>
<td>( 0.7371,-0.4290)</td>
<td>( 0.3224, 0.0776)</td>
<td>( 0.6887, 0.7891)</td>
</tr>
<tr>
<td>4</td>
<td>( 0.3702,-0.5040)</td>
<td>( 3.7253,-3.1813)</td>
<td>( 1.7014, 0.7267)</td>
<td>( 3.9367, 3.3255)</td>
</tr>
</tbody>
</table>