NAG Library Function Document
nag_zhptri (f07pwc)

1 Purpose
nag_zhptri (f07pwc) computes the inverse of a complex Hermitian indefinite matrix A, where A has been factorized by nag_zhptrf (f07prc), using packed storage.

2 Specification
#include <nag.h>
#include <nagf07.h>
void nag_zhptri (Nag_OrderType order, Nag_UploType uplo, Integer n,
Complex ap[], const Integer ipiv[], NagError *fail)

3 Description
nag_zhptri (f07pwc) is used to compute the inverse of a complex Hermitian indefinite matrix A, the function must be preceded by a call to nag_zhptrf (f07prc), which computes the Bunch–Kaufman factorization of A, using packed storage.

If uplo = Nag_Upper, \( A = PUDU^HPT \) and \( A^{-1} \) is computed by solving \( U^HPTXPU = D^{-1} \) for \( X \).

If uplo = Nag_Lower, \( A = PLDL^HPT \) and \( A^{-1} \) is computed by solving \( L^HP^TXPL = D^{-1} \) 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:  uplo – Nag_UploType

   On entry: specifies how A has been factorized.

   uplo = Nag_Upper
   \( A = PUDU^HPT \), where U is upper triangular.

   uplo = Nag_Lower
   \( A = PLDL^HPT \), 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 ≥ 0.
4: \( \text{ap}[\text{dim}] - \text{Complex} \)  

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

\textit{On entry:} the factorization of \( A \) stored in packed form, as returned by nag_zhptrf (f07prc).

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

The storage of elements \( A_{ij} \) depends on the \textit{order} and \textit{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 \( i \geq j \).

5: \( \text{ipiv}[\text{dim}] - \text{const Integer} \)  

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

\textit{On entry:} details of the interchanges and the block structure of \( D \), as returned by nag_zhptrf (f07prc).

6: \( \text{fail} - \text{NagError * } \)  

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

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

\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. \( D \) is singular and the inverse of \( A \) cannot be computed.
7 Accuracy

The computed inverse $X$ satisfies a bound of the form

$$ |D^TU^T P^TXPU - I| \leq c(n)\epsilon( |D||U^T|P^T|X|P|U| + |D||D^{-1}|) $$

if $\text{uplo} = \text{Nag}._{\text{Upper}}$, $|D^LP^TXPL - I| \leq c(n)\epsilon( |D||L^T|P^T|X|P|L| + |D||D^{-1}|)$,

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

8 Parallelism and Performance

nag_zhptri (f07pwc) is not threaded by NAG in any implementation.

nag_zhptri (f07pwc) 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 $\frac{8}{3}n^3$.

The real analogue of this function is nag_dsptri (f07pjc).

10 Example

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

$$ A = \begin{pmatrix} -1.36 + 0.00i & 1.58 + 0.90i & 2.21 - 0.21i & 3.91 + 1.50i \\ 1.58 - 0.90i & -8.87 + 0.00i & -1.84 - 0.03i & -1.78 + 1.18i \\ 2.21 + 0.21i & -1.84 + 0.03i & -4.63 + 0.00i & 0.11 + 0.11i \\ 3.91 - 1.50i & -1.78 - 1.18i & 0.11 - 0.11i & -1.84 + 0.00i \end{pmatrix}. $$

Here $A$ is Hermitian indefinite, stored in packed form, and must first be factorized by nag_zhptrf (f07prc).

10.1 Program Text

/* nag_zhptri (f07pwc) 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;
    Integer exit_status = 0;
    NagError fail;
    Nag_UploType uplo;
    Nag_OrderType order;
    /* Arrays */
    Integer *ipiv = 0;
    char *nag_enum_arg[40];

    ...
Complex *ap = 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]
    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]
    order = Nag_RowMajor;
#endif

INIT_FAIL(fail);
printf("nag_zhptri (f07pwc) Example Program Results\n\n");

/* Skip heading in data file */
#ifndef _WIN32
    scanf_s("%*[\n] ");
#else
    scanf("%*[\n] ");
#endif

#ifndef _WIN32
    scanf("%"NAG_IFMT"%*[\n] ", &n);
#else
    scanf("%"NAG_IFMT"%*[\n] ", &n);
#endif

ap_len = n * (n + 1)/2;

/* Allocate memory */
if (! (ipiv = NAG_ALLOC(n, Integer)) || (! (ap = NAG_ALLOC(ap_len, Complex)))
    { printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

/* Read A from data file */
#ifndef _WIN32
    scanf_s("%39s%*[\n] ", nag_enum_arg, _countof(nag_enum_arg));
#else
    scanf("%39s%*[\n] ", nag_enum_arg);
#endif

#ifndef _WIN32
    * (nag_enum_name_to_value (x04nac)). Converts NAG enum member name to value */
#else
    scanf("%39s%*[\n] ", nag_enum_arg);
#endif

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)
            #ifdef _WIN32
                scanf_s("( %lf , %lf )", &A_UPPER(i, j).re,
                        &A_UPPER(i, j).im);
            #else
                scanf("( %lf , %lf )", &A_UPPER(i, j).re,
                        &A_UPPER(i, j).im);
            #endif
        }      #ifdef _WIN32
            scanf_s("%*[\n] ");
        #else
            scanf("%*[\n] ");
        }
#endif
} else {
  for (i = 1; i <= n; ++i) {
    for (j = 1; j <= i; ++j)
      #ifdef _WIN32
        scanf_s(" ( %lf , %lf )", &A_LOWER(i, j).re,
                &A_LOWER(i, j).im);
      #else
        scanf(" ( %lf , %lf )", &A_LOWER(i, j).re,
              &A_LOWER(i, j).im);
      #endif
  }
  #ifdef _WIN32
    scanf_s("%*[^

] ");
  #else
    scanf("%*[^

] ");
  #endif

  /* Factorize A */
  /* nag_zhptrf (f07prc). 
     * Bunch-Kaufman factorization of complex Hermitian 
     * indefinite matrix, packed storage 
     */
  nag_zhptrf(order, uplo, n, ap, ipiv, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_zhptrf (f07prc).
            %s
", fail.message);
    exit_status = 1;
    goto END;
  }
  /* Compute inverse of A */
  /* nag_zhptri (f07pwc). 
     * Inverse of complex Hermitian indefinite matrix, matrix 
     * already factorized by nag_zhptrf (f07prc), packed storage 
     */
  nag_zhptri(order, uplo, n, ap, ipiv, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_zhptri (f07pwc).
            %s
", fail.message);
    exit_status = 1;
    goto END;
  }
  /* Print inverse */
  /* nag_pack_complx_mat_print_comp (x04ddc). 
     * Print complex packed triangular matrix (comprehensive) 
     */
  fflush(stdout);
  nag_pack_complx_mat_print_comp(order, uplo, Nag_NonUnitDiag, n, ap,
                                  Nag_BracketForm, "%7.4f", "Inverse",
                                  Nag_IntegerLabels, 0, Nag_IntegerLabels, 0,
                                  80, 0, 0, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_pack_complx_mat_print_comp (x04ddc).
            %s
", fail.message);
    exit_status = 1;
    goto END;
  }
END:
NAG_FREE(ipiv);
NAG_FREE(ap);
return exit_status;
10.2 Program Data

\texttt{nag_zhptri (f07pwc) Example Program Data}

\begin{verbatim}
4 :Value of n
Nag_Lower :Value of uplo
(-1.36, 0.00) :End of matrix A
( 1.58,-0.90) (-8.87, 0.00)
( 2.21, 0.21) (-1.84, 0.03) (-4.63, 0.00)
( 3.91,-1.50) (-1.78,-1.18) ( 0.11,-0.11) (-1.84, 0.00)
\end{verbatim}

10.3 Program Results

\texttt{nag_zhptri (f07pwc) Example Program Results}

\begin{verbatim}
Inverse

\begin{tabular}{cccc}
1 & 2 & 3 & 4 \\
0.0826 & 0.0000 & & \\
-0.0335 & 0.0440 & -0.1408 & 0.0000 \\
0.0603 & -0.0105 & 0.0422 & -0.0222 & -0.2007 & -0.0000 \\
0.2391 & -0.0926 & 0.0304 & 0.0203 & 0.0982 & -0.0635 & 0.0073 & 0.0000 \\
\end{tabular}
\end{verbatim}