NAG Library Function Document

nag_zpocon (f07fuc)

1 Purpose
nag_zpocon (f07fuc) estimates the condition number of a complex Hermitian positive definite matrix $A$, where $A$ has been factorized by nag_zpotrf (f07frc).

2 Specification
#include <nag.h>
#include <nagf07.h>
void nag_zpocon (Nag_OrderType order, Nag_UploType uplo, Integer n,
const Complex a[], Integer pda, double anorm, double *rcond,
NagError *fail)

3 Description
nag_zpocon (f07fuc) estimates the condition number (in the 1-norm) of a complex Hermitian positive definite matrix $A$:

$$\kappa_1(A) = \|A\|_1 \|A^{-1}\|_1.$$ 

Since $A$ is Hermitian, $\kappa_1(A) = \kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty$. 

Because $\kappa_1(A)$ is infinite if $A$ is singular, the function actually returns an estimate of the reciprocal of $\kappa_1(A)$.

The function should be preceded by a call to nag_zhe_norm (f16ucc) to compute $\|A\|_1$ and a call to nag_zpotrf (f07frc) to compute the Cholesky factorization of $A$. The function then uses Higham’s implementation of Hager’s method (see Higham (1988)) to estimate $\|A^{-1}\|_1$.

4 References
Higham N J (1988) FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation ACM Trans. Math. Software 14 381–396

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
$A = U^H U$, where $U$ is upper triangular.
uplo = Nag_Lower
    A = LL^H, where L is lower triangular.

Constraint: uplo = Nag_Upper or Nag_Lower.

3:  n – Integer  
    Input
    On entry: n, the order of the matrix A.
    Constraint: n ≥ 0.

4:  a[dim] – const Complex  
    Input
    Note: the dimension, dim, of the array a must be at least max(1, pda × n).
    On entry: the Cholesky factor of A, as returned by nag_zpotrf (f07frc).

5:  pda – Integer  
    Input
    On entry: the stride separating row or column elements (depending on the value of order) of the matrix in the array a.
    Constraint: pda ≥ max(1, n).

6:  anorm – double  
    Input
    On entry: the 1-norm of the original matrix A, which may be computed by calling nag_zhe_norm (f16ucc) with its argument norm = Nag_OneNorm. anorm must be computed either before calling nag_zpotrf (f07frc) or else from a copy of the original matrix A.
    Constraint: anorm ≥ 0.0.

7:  rcond – double *  
    Output
    On exit: an estimate of the reciprocal of the condition number of A. rcond is set to zero if exact singularity is detected or the estimate underflows. If rcond is less than machine precision, A is singular to working precision.

8:  fail – NagError *  
    Input/Output
    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 ⟨value⟩ had an illegal value.

NE_INT
    On entry, n = ⟨value⟩.
    Constraint: n ≥ 0.
    On entry, pda = ⟨value⟩.
    Constraint: pda > 0.

NE_INT_2
    On entry, pda = ⟨value⟩ and n = ⟨value⟩.
    Constraint: pda ≥ max(1, n).
7 Accuracy
The computed estimate $\text{rcond}$ is never less than the true value $\rho$, and in practice is nearly always less than $10\rho$, although examples can be constructed where $\text{rcond}$ is much larger.

8 Parallelism and Performance
nag_zpocon (f07fuc) is not threaded by NAG in any implementation.
nag_zpocon (f07fuc) 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
A call to nag_zpocon (f07fuc) involves solving a number of systems of linear equations of the form $Ax = b$; the number is usually 5 and never more than 11. Each solution involves approximately $8n^2$ real floating-point operations but takes considerably longer than a call to nag_zpotrs (f07fsc) with one right-hand side, because extra care is taken to avoid overflow when $A$ is approximately singular.

The real analogue of this function is nag_dpocon (f07fgc).

10 Example
This example estimates the condition number in the 1-norm (or $\infty$-norm) of the matrix $A$, where

$$A = \begin{bmatrix}
3.23 + 0.00i & 1.51 - 1.92i & 1.90 + 0.84i & 0.42 + 2.50i \\
1.51 + 1.92i & 3.58 + 0.00i & -0.23 + 1.11i & -1.18 + 1.37i \\
1.90 - 0.84i & -0.23 - 1.11i & 4.09 + 0.00i & 2.33 - 0.14i \\
0.42 - 2.50i & -1.18 - 1.37i & 2.33 + 0.14i & 4.29 + 0.00i
\end{bmatrix}.$$ 

Here $A$ is Hermitian positive definite and must first be factorized by nag_zpotrf (f07frc). The true condition number in the 1-norm is 201.92.
10.1 Program Text

/* nag_zpocon (f07fuc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 7, 2001. */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <naga02.h>
#include <nagf07.h>
#include <nagf16.h>
#include <nagx02.h>

int main(void)
{
    /* Scalars */
    double anorm, rcond;
    Integer i, j, n, pda;
    Integer exit_status = 0;
    Nag_UploType uplo;
    NagError fail;
    Nag_OrderType order;
    /* Arrays */
    char nag_enum_arg[40];
    Complex *a = 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);

    printf("nag_zpocon (f07fuc) 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"%*[\n] ", &n);
    #else
    scanf("%"NAG_IFMT"%*[\n] ", &n);
    #endif
    
    #ifdef NAG_COLUMN_MAJOR
    pda = n;
    #else
    pda = n;
    #endif

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

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

/* Compute norm of A */
/* nag_zhe_norm (f16ucc).
 * l1-norm, infinity-norm, Frobenius norm, largest absolute
 * element, complex Hermitian matrix
 */
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_zhe_norm (f16ucc).\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Factorize A */
/* nag_zpotrf (f07frc).
 * Cholesky factorization of complex Hermitian
 * positive-definite matrix
 */
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_zpotrf (f07frc).\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Estimate condition number */
/* nag_zpocon (f07fuc).
 * Estimate condition number of complex Hermitian

* positive-definite matrix, matrix already factorized by
  * nag_zpotrf (f07frc)
*/

nag_zpocon(order, uplo, n, a, pda, anorm, &rcond, &fail);
if (fail.code != NE_NOERROR)
{
  printf("Error from nag_zpocon (f07fuc).\n%s\n", fail.message);
  exit_status = 1;
  goto END;
}

/* nag_machine_precision (x02ajc).
* The machine precision
*/
if (rcond >= nag_machine_precision)
  printf("Estimate of condition number =%11.2e\n", 1.0/rcond);
else
  printf("A is singular to working precision\n");

END:
  NAG_FREE(a);
  return exit_status;
}

10.2 Program Data

nag_zpocon (f07fuc) Example Program Data
  4 :Value of n
      Nag_Lower :Value of uplo
    (3.23, 0.00)
    (1.51, 1.92) ( 3.58, 0.00)
    (1.90,-0.84) (-0.23,-1.11) ( 4.09, 0.00)
    (0.42,-2.50) (-1.18,-1.37) ( 2.33, 0.14) ( 4.29, 0.00) :End of matrix A

10.3 Program Results

nag_zpocon (f07fuc) Example Program Results

Estimate of condition number = 1.51e+02