NAG Library Function Document

nag_matop_complex_gen_matrix_cond_log (f01kjc)

1 Purpose

nag_matop_complex_gen_matrix_cond_log (f01kjc) computes an estimate of the relative condition number $\kappa_{\log}(A)$ of the logarithm of a complex n by n matrix A, in the 1-norm. The principal matrix logarithm $\log(A)$ is also returned.

2 Specification

```
#include <nag.h>
#include <nagf01.h>
void nag_matop_complex_gen_matrix_cond_log (Integer n, Complex a[],
     Integer pda, double *condla, NagError *fail)
```

3 Description

For a matrix with no eigenvalues on the closed negative real line, the principal matrix logarithm $\log(A)$ is the unique logarithm whose spectrum lies in the strip $\{z : -\pi < \mathrm{Im}(z) < \pi\}$.

The Fréchet derivative of the matrix logarithm of A is the unique linear mapping $E \mapsto L(A, E)$ such that for any matrix E

$$\log(A + E) - \log(A) - L(A, E) = o(\|E\|).$$

The derivative describes the first order effect of perturbations in A on the logarithm $\log(A)$.

The relative condition number of the matrix logarithm can be defined by

$$\kappa_{\log}(A) = \frac{\|L(A)\|\|A\|}{\|\log(A)\|},$$

where $\|L(A)\|$ is the norm of the Fréchet derivative of the matrix logarithm at A.

To obtain the estimate of $\kappa_{\log}(A)$, nag_matop_complex_gen_matrix_cond_log (f01kjc) first estimates $\|L(A)\|$ by computing an estimate γ of a quantity $K \in \left[n^{-1}\|L(A)\|_1, n\|L(A)\|_1\right]$, such that $\gamma \leq K$.

The algorithms used to compute $\kappa_{\log}(A)$ and $\log(A)$ are based on a Schur decomposition, the inverse scaling and squaring method and Padé approximants. Further details can be found in Al–Mohy and Higham (2011) and Al–Mohy *et al.* (2012).

If A is nonsingular but has negative real eigenvalues, the principal logarithm is not defined, but nag_matop_complex_gen_matrix_cond_log (f01kjc) will return a non-principal logarithm and its condition number.

4 References

Al–Mohy A H and Higham N J (2011) Improved inverse scaling and squaring algorithms for the matrix logarithm *SIAM J. Sci. Comput.* **34(4)** C152–C169

Al–Mohy A H, Higham N J and Relton S D (2012) Computing the Fréchet derivative of the matrix logarithm and estimating the condition number *SIAM J. Sci. Comput.* **35(4)** C394–C410

Higham N J (2008) *Functions of Matrices: Theory and Computation* SIAM, Philadelphia, PA, USA

5 Arguments

1: **n** – Integer *Input*

On entry: n, the order of the matrix A.

Constraint: **n** ≥ 0.

2: **a**$[dim]$ – Complex *Input/Output*

Note: the dimension, *dim*, of the array **a** must be at least **pda** $\times$ **n**.

The (i,j)th element of the matrix A is stored in **a**$[(j-1) \times$ **pda** $+ i - 1]$.

On entry: the n by n matrix A.

On exit: the n by n principal matrix logarithm, $\log(A)$. Alternatively, if **fail.code** $=$ NE_NEGATIVE_EIGVAL, a non-principal logarithm is returned.

3: **pda** – Integer *Input*

On entry: the stride separating matrix row elements in the array **a**.

Constraint: **pda** $\geq$ **n**.

4: **condla** – double * *Output*

On exit: with **fail.code** $=$ NE_NOERROR, NE_NEGATIVE_EIGVAL or NW_SOME_PRECISION_LOSS, an estimate of the relative condition number of the matrix logarithm, $\kappa_{\log}(A)$. Alternatively, if **fail.code** $=$ NE_RCOND, contains the absolute condition number of the matrix logarithm.

5: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

On entry, **n** $=$ ⟨*value*⟩.
Constraint: **n** ≥ 0.

NE_INT_2

On entry, **pda** $=$ ⟨*value*⟩ and **n** $=$ ⟨*value*⟩.
Constraint: **pda** $\geq$ **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 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NEGATIVE_EIGVAL

A has eigenvalues on the negative real line. The principal logarithm is not defined in this case, so a non-principal logarithm was returned.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

NE_RCOND

The relative condition number is infinite. The absolute condition number was returned instead.

NE_SINGULAR

A is singular so the logarithm cannot be computed.

NW_SOME_PRECISION_LOSS

$\log(A)$ has been computed using an IEEE double precision Padé approximant, although the arithmetic precision is higher than IEEE double precision.

7 Accuracy

nag_matop_complex_gen_matrix_cond_log (f01kjc) uses the norm estimation function nag_linsys_complex_gen_norm_rcomm (f04zdc) to produce an estimate γ of a quantity $K \in \left[n^{-1}\|L(A)\|_1, n\|L(A)\|_1\right]$, such that $\gamma \leq K$. For further details on the accuracy of norm estimation, see the documentation for nag_linsys_complex_gen_norm_rcomm (f04zdc).

For a normal matrix A (for which $A^{\mathrm{H}}A = AA^{\mathrm{H}}$), the Schur decomposition is diagonal and the computation of the matrix logarithm reduces to evaluating the logarithm of the eigenvalues of A and then constructing $\log(A)$ using the Schur vectors. This should give a very accurate result. In general, however, no error bounds are available for the algorithm. The sensitivity of the computation of $\log(A)$ is worst when A has an eigenvalue of very small modulus or has a complex conjugate pair of eigenvalues lying close to the negative real axis. See Al–Mohy and Higham (2011) and Section 11.2 of Higham (2008) for details and further discussion.

8 Parallelism and Performance

nag_matop_complex_gen_matrix_cond_log (f01kjc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_matop_complex_gen_matrix_cond_log (f01kjc) 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

nag_matop_complex_gen_matrix_cond_std (f01kac) uses a similar algorithm to nag_matop_complex_gen_matrix_cond_log (f01kjc) to compute an estimate of the *absolute* condition number (which is related to the relative condition number by a factor of $\|A\|/\|\log(A)\|$). However, the required Fréchet derivatives are computed in a more efficient and stable manner by nag_matop_complex_gen_matrix_cond_log (f01kjc) and so its use is recommended over nag_matop_complex_gen_matrix_cond_std (f01kac).

The amount of complex allocatable memory required by the algorithm is typically of the order $10n^2$.

The cost of the algorithm is $O\left(n^3\right)$ floating-point operations; see Al–Mohy *et al.* (2012).

If the matrix logarithm alone is required, without an estimate of the condition number, then nag_matop_complex_gen_matrix_log (f01fjc) should be used. If the Fréchet derivative of the matrix logarithm is required then nag_matop_complex_gen_matrix_frcht_log (f01kkc) should be used. The real analogue of this function is nag_matop_real_gen_matrix_cond_log (f01jjc).

10 Example

This example estimates the relative condition number of the matrix logarithm $\log(A)$, where

$$A = \begin{pmatrix} 3+2i & 1 & 1 & 1+2i \\ 0+2i & -4 & 0 & 0 \\ 1 & -2 & 3+2i & 0+i \\ 1 & i & 1 & 2+3i \end{pmatrix}.$$

10.1 Program Text

```
/* nag_matop_complex_gen_matrix_cond_log (f01kjc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf01.h>
#include <nagx04.h>

#define A(I,J) a[J*pda + I]

int main(void)
{
  /* Scalars */
  Integer exit_status = 0;
  Integer pda;
  Integer i, j, n;
  double condla;
  /* Arrays */
  Complex *a = 0;
  /* Nag Types */
  Nag_OrderType order = Nag_ColMajor;
  NagError fail;

  INIT_FAIL(fail);

  printf("nag_matop_complex_gen_matrix_cond_log (f01kjc) ");
  printf("Example Program Results\n\n");
  fflush(stdout);

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

  /* Read in the problem size */
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%*[^\n]", &n);
#else
  scanf("%" NAG_IFMT "%*[^\n]", &n);
#endif

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

  /* Read in the matrix A from data file */
```

```
  for (i = 0; i < n; i++)
    for (j = 0; 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

  /* Find log(A) and the condition number using
   * nag_matop_complex_gen_matrix_cond_log (f01kjc)
   * Condition number for complex matrix logarithm
   */
  nag_matop_complex_gen_matrix_cond_log(n, a, pda, &condla, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_matop_complex_gen_matrix_cond_log (f01kjc)\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
  }

  /* Print matrix log(A) using nag_gen_cmplx_mat_print (x04dac)
   * Print complex general matrix (easy-to-use)
   */
  nag_gen_complx_mat_print(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, n,
                           a, pda, "log(A)", NULL, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_gen_cmplx_mat_print (x04dac)\n%s\n", fail.message);
    exit_status = 2;
    goto END;
  }

  /* Print relative condition number estimate */
  printf("Estimated relative condition number is: %7.2f\n", condla);

END:

  NAG_FREE(a);

  return exit_status;
}
```

10.2 Program Data

```
nag_matop_complex_gen_matrix_cond_log (f01kjc) Example Program Data

   4                                                :Value of n

  (3.0, 2.0) ( 1.0, 0.0) (1.0, 0.0) (1.0, 2.0)
  (0.0, 2.0) (-4.0, 0.0) (0.0, 0.0) (0.0, 0.0)
  (1.0, 0.0) (-2.0, 0.0) (3.0, 2.0) (0.0, 1.0)
  (1.0, 0.0) ( 0.0, 1.0) (1.0, 0.0) (2.0, 3.0)  :End of matrix a
```

10.3 Program Results

```
nag_matop_complex_gen_matrix_cond_log (f01kjc) Example Program Results

 log(A)
              1           2           3           4
 1       1.4498      0.3665      0.1358      0.4890
         0.5154      0.6955     -0.1097      0.1622

 2      -0.9351      1.2908      0.1010      0.3128
         0.2859     -2.8365     -0.0672      0.2538

 3      -0.1399     -0.3208      1.2738      0.2658
        -0.1083     -0.8912      0.5775      0.3127
```

```
 4      0.3049     -0.4858      0.1797      1.1843
       -0.0019      0.3215     -0.1922      0.9427

Estimated relative condition number is:    2.25
```