# NAG Toolbox

# nag_matop_complex_gen_matrix_cond_usd (f01kc)

## 1 Purpose

nag_matop_complex_gen_matrix_cond_usd (f01kc) computes an estimate of the absolute condition number of a matrix function $f$ of a complex $n$ by $n$ matrix $A$ in the 1-norm, using analytical derivatives of $f$ you have supplied.

## 2 Syntax

```
[a, user, iflag, conda, norma, normfa, ifail] =
nag_matop_complex_gen_matrix_cond_usd(a, f, 'n', n, 'user', user)
```

```
[a, user, iflag, conda, norma, normfa, ifail] = f01kc(a, f, 'n', n, 'user',
user)
```

## 3 Description

The absolute condition number of $f$ at $A$, $\mathrm{cond}_{\mathrm{abs}}(f, A)$ is given by the norm of the Fréchet derivative of $f$, $L(A)$, which is defined by

$$\|L(X)\| := \max_{E \neq 0} \frac{\|L(X, E)\|}{\|E\|},$$

where $L(X, E)$ is the Fréchet derivative in the direction $E$. $L(X, E)$ is linear in $E$ and can therefore be written as

$$\mathrm{vec}(L(X, E)) = K(X)\mathrm{vec}(E),$$

where the vec operator stacks the columns of a matrix into one vector, so that $K(X)$ is $n^2 \times n^2$. nag_matop_complex_gen_matrix_cond_usd (f01kc) computes an estimate $\gamma$ such that $\gamma \leq \|K(X)\|_1$, where $\|K(X)\|_1 \in \left[n^{-1}\|L(X)\|_1, n\|L(X)\|_1\right]$. The relative condition number can then be computed via

$$\mathrm{cond}_{\mathrm{rel}}(f, A) = \frac{\mathrm{cond}_{\mathrm{abs}}(f, A)\|A\|_1}{\|f(A)\|_1}.$$

The algorithm used to find $\gamma$ is detailed in Section 3.4 of Higham (2008).

## 4 References

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

## 5 Parameters

### 5.1 Compulsory Input Parameters

1: **a**$(lda, :)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **a** must be at least **n**.

The second dimension of the array **a** must be at least **n**.

The $n$ by $n$ matrix $A$.

2: **f** – SUBROUTINE, supplied by the user.

Given an integer $m$, the function **f** evaluates $f^{(m)}(z_i)$ at a number of points $z_i$.

```
[iflag, fz, user] = f(m, iflag, nz, z, user)
```

**Input Parameters**

1: **m** – INTEGER

The order, $m$, of the derivative required.

If $\mathbf{m} = 0$, $f(z_i)$ should be returned. For $\mathbf{m} > 0$, $f^{(m)}(z_i)$ should be returned.

2: **iflag** – INTEGER

**iflag** will be zero.

3: **nz** – INTEGER

$n_z$, the number of function or derivative values required.

4: **z**(**nz**) – COMPLEX (KIND=nag_wp) array

The $n_z$ points $z_1, z_2, \ldots, z_{n_z}$ at which the function $f$ is to be evaluated.

5: **user** – INTEGER array

**f** is called from nag_matop_complex_gen_matrix_cond_usd (f01kc) with the object supplied to nag_matop_complex_gen_matrix_cond_usd (f01kc).

**Output Parameters**

1: **iflag** – INTEGER

**iflag** should either be unchanged from its entry value of zero, or may be set nonzero to indicate that there is a problem in evaluating the function $f(z)$; for instance $f(z)$ may not be defined. If **iflag** is returned as nonzero then nag_matop_complex_gen_matrix_cond_usd (f01kc) will terminate the computation, with $\mathbf{ifail} = 3$.

2: **fz**(**nz**) – COMPLEX (KIND=nag_wp) array

The $n_z$ function or derivative values. $\mathbf{fz}(i)$ should return the value $f^{(m)}(z_i)$, for $i = 1, 2, \ldots, n_z$.

3: **user** – INTEGER array

## 5.2 Optional Input Parameters

1: **n** – INTEGER

*Default*: the first dimension of the array **a**.

$n$, the order of the matrix $A$.

*Constraint*: $\mathbf{n} \geq 0$.

2: **user** – INTEGER array

**user** is not used by nag_matop_complex_gen_matrix_cond_usd (f01kc), but is passed to **f**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

### 5.3 Output Parameters

1: **a**($lda$, :) – COMPLEX (KIND=nag_wp) array

The first dimension of the array **a** will be **n**.

The second dimension of the array **a** will be **n**.

The $n$ by $n$ matrix, $f(A)$.

2: **user** – INTEGER array

3: **iflag** – INTEGER

**iflag** $= 0$, unless **iflag** has been set nonzero inside **f**, in which case **iflag** will be the value set and **ifail** will be set to **ifail** $= 3$.

4: **conda** – REAL (KIND=nag_wp)

An estimate of the absolute condition number of $f$ at $A$.

5: **norma** – REAL (KIND=nag_wp)

The 1-norm of $A$.

6: **normfa** – REAL (KIND=nag_wp)

The 1-norm of $f(A)$.

7: **ifail** – INTEGER

**ifail** $= 0$ unless the function detects an error (see Section 5).

## 6 Error Indicators and Warnings

Errors or warnings detected by the function:

**ifail** $= 1$

An internal error occurred when estimating the norm of the Fréchet derivative of $f$ at $A$. Please contact NAG.

**ifail** $= 2$

An internal error occurred when evaluating the matrix function $f(A)$. You can investigate further by calling nag_matop_complex_gen_matrix_fun_usd (f01fm) with the matrix $A$ and the function $f$.

**ifail** $= 3$

**iflag** has been set nonzero by the user-supplied function.

**ifail** $= -1$

On entry, **n** $< 0$.

**ifail** $= -3$

On entry, argument $lda$ is invalid.
Constraint: $lda \geq$ **n**.

**ifail** $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

**ifail** $= -399$

Your licence key may have expired or may not have been installed correctly.

**ifail** $= -999$

Dynamic memory allocation failed.

## 7 Accuracy

nag_matop_complex_gen_matrix_cond_usd (f01kc) uses the norm estimation function nag_linsys_complex_gen_norm_rcomm (f04zd) to estimate a quantity $\gamma$, where $\gamma \leq \|K(X)\|_1$ and $\|K(X)\|_1 \in \left[n^{-1}\|L(X)\|_1, n\|L(X)\|_1\right]$. For further details on the accuracy of norm estimation, see the documentation for nag_linsys_complex_gen_norm_rcomm (f04zd).

## 8 Further Comments

Approximately $6n^2$ of complex allocatable memory is required by the routine, in addition to the memory used by the underlying matrix function routine nag_matop_complex_gen_matrix_fun_usd (f01fm).

nag_matop_complex_gen_matrix_cond_usd (f01kc) returns the matrix function $f(A)$. This is computed using nag_matop_complex_gen_matrix_fun_usd (f01fm). If only $f(A)$ is required, without an estimate of the condition number, then it is far more efficient to use nag_matop_complex_gen_matrix_fun_usd (f01fm) directly.

The real analogue of this function is nag_matop_real_gen_matrix_cond_usd (f01jc).

## 9 Example

This example estimates the absolute and relative condition numbers of the matrix function $e^{3A}$ where

$$A = \begin{pmatrix} 1.0 + 1.0i & 0.0 + 1.0i & 1.0 + 0.0i & 2.0 + 0.0i \\ 0.0 + 0.0i & 2.0 + 0.0i & 0.0 + 2.0i & 1.0 + 0.0i \\ 0.0 + 1.0i & 0.0 + 1.0i & 0.0 + 0.0i & 2.0 + 0.0i \\ 1.0 + 0.0i & 0.0 + 1.0i & 1.0 + 0.0i & 0.0 + 1.0i \end{pmatrix}.$$

### 9.1 Program Text

```
     function f01kc_example

fprintf('f01kc example results\n\n');

a = [1+1i, 0+1i, 1+0i, 2+0i;
     0+0i, 2+0i, 0+2i, 1+0i;
     0+1i, 0+1i, 0+0i, 2+0i;
     1+0i, 0+1i, 1+0i, 0+1i];

% Find absolute condition number estimate
[a, user, iflag, conda, norma, normfa, ifail] = ...
f01kc(a, @fexp3);

fprintf('\nf(A) = exp(3A)\n');
fprintf('Estimated absolute condition number is: %7.2f\n', conda);

%  Find relative condition number estimate
eps = x02aj;
if normfa > eps
   cond_rel = conda*norma/normfa;
   fprintf('Estimated relative condition number is: %7.2f\n', cond_rel);
else
  fprintf('The estimated norm of f(A) is effectively zero;\n');
```

```
  fprintf('the relative condition number is therefore undefined.\n');
end

function [iflag, fz, user] = fexp3(m, iflag, nz, z, user)
  fz = 3^double(m)*exp(3*z);
```

## 9.2 Program Results

```
     f01kc example results

f(A) = exp(3A)
Estimated absolute condition number is: 9474.43
Estimated relative condition number is:   13.74
```