

NAG Toolbox

nag_matop_real_gen_matrix_sqrt (f01en)

1 Purpose

nag_matop_real_gen_matrix_sqrt (f01en) computes the principal matrix square root, $A^{1/2}$, of a real n by n matrix A .

2 Syntax

```
[a, ifail] = nag_matop_real_gen_matrix_sqrt(a, 'n', n)
[a, ifail] = f01en(a, 'n', n)
```

3 Description

A square root of a matrix A is a solution X to the equation $X^2 = A$. A nonsingular matrix has multiple square roots. For a matrix with no eigenvalues on the closed negative real line, the principal square root, denoted by $A^{1/2}$, is the unique square root whose eigenvalues lie in the open right half-plane.

$A^{1/2}$ is computed using the algorithm described in Higham (1987). This is a real arithmetic version of the algorithm of Björck and Hammarling (1983). In addition a blocking scheme described in Deadman *et al.* (2013) is used.

4 References

Björck D and Hammarling S (1983) A Schur method for the square root of a matrix *Linear Algebra Appl.* **52/53** 127–140

Deadman E, Higham N J and Ralha R (2013) Blocked Schur Algorithms for Computing the Matrix Square Root *Applied Parallel and Scientific Computing: 11th International Conference, (PARA 2012, Helsinki, Finland)* P. Manninen and P. Úster, Eds *Lecture Notes in Computer Science* **7782** 171–181 Springer–Verlag

Higham N J (1987) Computing real square roots of a real matrix *Linear Algebra Appl.* **88/89** 405–430

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

5 Parameters

5.1 Compulsory Input Parameters

1: **a**(lda,:) – REAL (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 .

5.2 Optional Input Parameters

1: **n** – INTEGER

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

n , the order of the matrix A .

Constraint: $n \geq 0$.

5.3 Output Parameters

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

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

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

Contains, if **ifail** = 0, the n by n principal matrix square root, $A^{1/2}$. Alternatively, if **ifail** = 1, contains an n by n non-principal square root of A .

2: **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

A has a semisimple vanishing eigenvalue. A non-principal square root is returned.

ifail = 2

A has a defective vanishing eigenvalue. The square root cannot be found in this case.

ifail = 3

A has a negative real eigenvalue. The principal square root is not defined. nag_matop_complex_gen_matrix_sqrt (f01fn) can be used to return a complex, non-principal square root.

ifail = 4

An internal error occurred. It is likely that the function was called incorrectly.

ifail = -1

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

ifail = -3

Constraint: $lda \geq \mathbf{n}$.

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

The computed square root \hat{X} satisfies $\hat{X}^2 = A + \Delta A$, where $\|\Delta A\|_F \approx O(\epsilon)n^3\|\hat{X}\|_F^2$, where ϵ is *machine precision*.

For further discussion of the condition of the matrix square root see Section 6.1 of Higham (2008).

8 Further Comments

The cost of the algorithm is $85n^3/3$ floating-point operations; see Algorithm 6.7 of Higham (2008). $O(2 \times n^2)$ of real allocatable memory is required by the function.

If condition number and residual bound estimates are required, then `nag_matop_real_gen_matrix_cond_sqrt` (f01jd) should be used.

9 Example

This example finds the principal matrix square root of the matrix

$$A = \begin{pmatrix} 507 & 622 & 300 & -202 \\ 237 & 352 & 126 & -60 \\ 751 & 950 & 440 & -286 \\ -286 & -326 & -192 & 150 \end{pmatrix}.$$

9.1 Program Text

```
function f01en_example
fprintf('f01en example results\n\n');
% Principal square root of matrix A
a = [ 507  622  300 -202;
      237  352  126  -60;
      751  950  440 -286;
      -286 -326 -192  150];
[as, ifail] = f01en(a);
disp('Square root of A:');
disp(as);
```

9.2 Program Results

```
f01en example results

Square root of A:
15.0000  14.0000   8.0000  -6.0000
 6.0000  14.0000   3.0000   0.0000
21.0000  24.0000  12.0000  -8.0000
-5.0000  -4.0000  -7.0000   8.0000
```
