

## NAG Toolbox

### nag\_eigen\_complex\_triang\_svd (f02xu)

## 1 Purpose

nag\_eigen\_complex\_triang\_svd (f02xu) returns all, or part, of the singular value decomposition of a complex upper triangular matrix.

## 2 Syntax

```
[a, b, q, sv, rwork, ifail] = nag_eigen_complex_triang_svd(a, b, wantq, wantp,
'n', n, 'ncolb', ncolb)
[a, b, q, sv, rwork, ifail] = f02xu(a, b, wantq, wantp, 'n', n, 'ncolb', ncolb)
```

## 3 Description

The  $n$  by  $n$  upper triangular matrix  $R$  is factorized as

$$R = QSP^H,$$

where  $Q$  and  $P$  are  $n$  by  $n$  unitary matrices and  $S$  is an  $n$  by  $n$  diagonal matrix with real non-negative diagonal elements,  $sv_1, sv_2, \dots, sv_n$ , ordered such that

$$sv_1 \geq sv_2 \geq \dots \geq sv_n \geq 0.$$

The columns of  $Q$  are the left-hand singular vectors of  $R$ , the diagonal elements of  $S$  are the singular values of  $R$  and the columns of  $P$  are the right-hand singular vectors of  $R$ .

Either or both of  $Q$  and  $P^H$  may be requested and the matrix  $C$  given by

$$C = Q^H B,$$

where  $B$  is an  $n$  by  $ncolb$  given matrix, may also be requested.

nag\_eigen\_complex\_triang\_svd (f02xu) obtains the singular value decomposition by first reducing  $R$  to bidiagonal form by means of Givens plane rotations and then using the  $QR$  algorithm to obtain the singular value decomposition of the bidiagonal form.

Good background descriptions to the singular value decomposition are given in Dongarra *et al.* (1979), Hammarling (1985) and Wilkinson (1978).

Note that if  $K$  is any unitary diagonal matrix so that

$$KK^H = I,$$

then

$$A = (QK)S(PK)^H$$

is also a singular value decomposition of  $A$ .

## 4 References

Dongarra J J, Moler C B, Bunch J R and Stewart G W (1979) *LINPACK Users' Guide* SIAM, Philadelphia

Hammarling S (1985) The singular value decomposition in multivariate statistics *SIGNUM Newslet.* **20**(3) 2–25

Wilkinson J H (1978) Singular Value Decomposition – Basic Aspects *Numerical Software – Needs and Availability* (ed D A H Jacobs) Academic Press

## 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  $\max(1, \mathbf{n})$ .

The second dimension of the array **a** must be at least  $\max(1, \mathbf{n})$ .

The leading  $n$  by  $n$  upper triangular part of the array **a** must contain the upper triangular matrix  $R$ .

2: **b**(*ldb*, :) – COMPLEX (KIND=nag\_wp) array

The first dimension, *ldb*, of the array **b** must satisfy

if **ncolb** > 0,  $ldb \geq \max(1, \mathbf{n})$ ;  
otherwise  $ldb \geq 1$ .

The second dimension of the array **b** must be at least  $\max(1, \mathbf{ncolb})$ .

If **ncolb** > 0, the leading  $n$  by  $ncolb$  part of the array **b** must contain the matrix to be transformed.

3: **wantq** – LOGICAL

Must be *true* if the matrix  $Q$  is required.

If **wantq** = *false* then the array **q** is not referenced.

4: **wantp** – LOGICAL

Must be *true* if the matrix  $P^H$  is required, in which case  $P^H$  is returned in the array **a**, otherwise **wantp** must be *false*.

### 5.2 Optional Input Parameters

1: **n** – INTEGER

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

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

If **n** = 0, an immediate return is effected.

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

2: **ncolb** – INTEGER

*Default:* the second dimension of the array **b**.

$ncolb$ , the number of columns of the matrix  $B$ .

If **ncolb** = 0, the array **b** is not referenced.

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

### 5.3 Output Parameters

1: **a**(*lda*, :) – COMPLEX (KIND=nag\_wp) array

The first dimension of the array **a** will be  $\max(1, \mathbf{n})$ .

The second dimension of the array **a** will be  $\max(1, \mathbf{n})$ .

If **wantp** = *true*, the  $n$  by  $n$  part of **a** will contain the  $n$  by  $n$  unitary matrix  $P^H$ , otherwise the  $n$  by  $n$  upper triangular part of **a** is used as internal workspace, but the strictly lower triangular part of **a** is not referenced.

2: **b**(*ldb*,:) – COMPLEX (KIND=nag\_wp) array

The first dimension, *ldb*, of the array **b** will be

if **ncolb** > 0, *ldb* = max(1, **n**);  
otherwise *ldb* = 1.

The second dimension of the array **b** will be max(1, **ncolb**).

Stores the *n* by *ncolb* matrix  $Q^H B$ .

3: **q**(*ldq*,:) – COMPLEX (KIND=nag\_wp) array

The first dimension, *ldq*, of the array **q** will be

if **wantq** = true, *ldq* = max(1, **n**);  
otherwise *ldq* = 1.

The second dimension of the array **q** will be max(1, **n**) if **wantq** = true and 1 otherwise.

If **wantq** = true, the leading *n* by *n* part of the array **q** will contain the unitary matrix  $Q$ . Otherwise the array **q** is not referenced.

4: **sv(n)** – REAL (KIND=nag\_wp) array

The *n* diagonal elements of the matrix  $S$ .

5: **rwork(:)** – REAL (KIND=nag\_wp) array

The dimension of the array **rwork** will be max(1,  $2 \times (\mathbf{n} - 1)$ ) if **ncolb** = 0 and **wantq** = false and **wantp** = false, max(1,  $3 \times (\mathbf{n} - 1)$ ) if **ncolb** = 0 and **wantq** = false and **wantp** = true or **ncolb** > 0 and **wantp** = false or **wantq** = true and **wantp** = false and max(1,  $5 \times (\mathbf{n} - 1)$ ) otherwise

**rwork(n)** contains the total number of iterations taken by the  $QR$  algorithm.

The rest of the array is used as workspace.

6: **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

On entry, **n** < 0,  
or      *lda* < **n**,  
or      **ncolb** < 0,  
or      *ldb* < **n** and **ncolb** > 0,  
or      *ldq* < **n** and **wantq** = true

**ifail** > 0 (warning)

The  $QR$  algorithm has failed to converge in  $50 \times \mathbf{n}$  iterations. In this case **sv(1), sv(2), ..., sv(ifail)** may not have been found correctly and the remaining singular values may not be the smallest. The matrix  $R$  will nevertheless have been factorized as  $R = QEP^H$ , where  $E$  is a bidiagonal matrix with **sv(1), sv(2), ..., sv(n)** as the diagonal elements and **rwork(1), rwork(2), ..., rwork(n - 1)** as the superdiagonal elements.

This failure is not likely to occur.

**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 factors  $Q$ ,  $S$  and  $P$  satisfy the relation

$$QSP^H = A + E,$$

where

$$\|E\| \leq c\epsilon\|A\|,$$

$\epsilon$  is the **machine precision**,  $c$  is a modest function of  $n$  and  $\|\cdot\|$  denotes the spectral (two) norm. Note that  $\|A\| = sv_1$ .

## 8 Further Comments

For given values of **nclb**, **wantq** and **wantp**, the number of floating-point operations required is approximately proportional to  $n^3$ .

>Following the use of nag\_eigen\_complex\_triang\_svd (f02xu) the rank of  $R$  may be estimated as follows:

```
tol = eps;
irank = 1;
while (irank <= numel(sv) && sv(irank) >= tol*sv(1) )
    irank = irank + 1;
end
```

returns the value  $k$  in **irank**, where  $k$  is the smallest integer for which  $sv(k) < tol \times sv(1)$ , where **tol** is typically the **machine precision**, so that **irank** is an estimate of the rank of  $S$  and thus also of  $R$ .

## 9 Example

This example finds the singular value decomposition of the 3 by 3 upper triangular matrix

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

together with the vector  $Q^H b$  for the vector

$$b = \begin{pmatrix} 1+1i \\ -1 \\ -1+1i \end{pmatrix}.$$

### 9.1 Program Text

```
function f02xu_example

fprintf('f02xu example results\n\n');

a = [ 1,      1 + 1i,  1 + 1i;
      0 + 0i, -2 + 0i, -1 - 1i;
      0 + 0i,  0 + 0i, -3 + 0i];
```

```

b = [ 1 + 1i; -1 + 0i; -1 + 1i];

wantq = true;
wantp = true;
[a, b, q, sv, rwork, ifail] = f02xu( ...
    a, b, wantq, wantp);

fprintf('Singular value decomposition of A\n\n');

disp('Singular values');
disp(sv');
disp('Left-hand singular vectors, by column');
disp(q);
disp('Right-hand singular vectors, by column');
disp(ctranspose(a));
disp('Vector Q^H*B');
disp(transpose(b));

```

## 9.2 Program Results

f02xu example results

Singular value decomposition of A

Singular values  
 3.9263    2.0000    0.7641

Left-hand singular vectors, by column  
 -0.5005 + 0.0000i   -0.4529 + 0.0000i   0.7378 + 0.0000i  
 0.5152 - 0.1514i   0.1132 - 0.5661i   0.4190 - 0.4502i  
 0.4041 - 0.5457i   0.0000 + 0.6794i   0.2741 + 0.0468i

Right-hand singular vectors, by column  
 -0.1275 + 0.0000i   -0.2265 + 0.0000i   0.9656 + 0.0000i  
 -0.3899 + 0.2046i   -0.3397 + 0.7926i   -0.1311 + 0.2129i  
 -0.5289 + 0.7142i   -0.0000 - 0.4529i   -0.0698 - 0.0119i

Vector Q^H\*B  
 -1.9656 - 0.7935i   0.1132 - 0.3397i   0.0915 + 0.6086i

---