NAG Toolbox

nag_lapack_zgeqrf (f08as)

1 Purpose

nag_lapack_zgeqrf (f08as) computes the QR factorization of a complex m by n matrix.

2 Syntax

```
[a, tau, info] = nag_lapack_zgeqrf(a, 'm', m, 'n', n)
[a, tau, info] = f08as(a, 'm', m, 'n', n)
```

3 Description

nag_lapack_zgeqrf (f08as) forms the QR factorization of an arbitrary rectangular complex m by n matrix. No pivoting is performed.

If $m \geq n$, the factorization is given by:

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix},$$

where R is an n by n upper triangular matrix (with real diagonal elements) and Q is an m by m unitary matrix. It is sometimes more convenient to write the factorization as

$$A = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}\begin{pmatrix} R \\ 0 \end{pmatrix},$$

which reduces to

$$A = Q_1 R,$$

where Q_1 consists of the first n columns of Q, and Q_2 the remaining $m - n$ columns.

If $m < n$, R is trapezoidal, and the factorization can be written

$$A = Q\begin{pmatrix} R_1 & R_2 \end{pmatrix},$$

where R_1 is upper triangular and R_2 is rectangular.

The matrix Q is not formed explicitly but is represented as a product of $\min(m, n)$ elementary reflectors (see the F08 Chapter Introduction for details). Functions are provided to work with Q in this representation (see Section 9).

Note also that for any $k < n$, the information returned in the first k columns of the array **a** represents a QR factorization of the first k columns of the original matrix A.

4 References

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

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 $\max(1, \mathbf{m})$.

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

The m by n matrix A.

5.2 Optional Input Parameters

1: **m** – INTEGER

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

m, the number of rows of the matrix A.

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

2: **n** – INTEGER

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

n, the number of columns of the matrix A.

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

5.3 Output Parameters

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

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

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

If $m \geq n$, the elements below the diagonal store details of the unitary matrix Q and the upper triangle stores the corresponding elements of the n by n upper triangular matrix R.

If $m < n$, the strictly lower triangular part stores details of the unitary matrix Q and the remaining elements store the corresponding elements of the m by n upper trapezoidal matrix R.

The diagonal elements of R are real.

2: **tau**$(:)$ – COMPLEX (KIND=nag_wp) array

The dimension of the array **tau** will be $\max(1, \min(\mathbf{m}, \mathbf{n}))$

Further details of the unitary matrix Q.

3: **info** – INTEGER

info $= 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

info $= -i$

If **info** $= -i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **m**, 2: **n**, 3: **a**, 4: **lda**, 5: **tau**, 6: **work**, 7: **lwork**, 8: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

7 Accuracy

The computed factorization is the exact factorization of a nearby matrix $(A + E)$, where

$$\|E\|_2 = O(\epsilon)\|A\|_2,$$

and ϵ is the ***machine precision***.

8 Further Comments

The total number of real floating-point operations is approximately $\frac{8}{3}n^2(3m-n)$ if $m \geq n$ or $\frac{8}{3}m^2(3n-m)$ if $m < n$.

To form the unitary matrix Q nag_lapack_zgeqrf (f08as) may be followed by a call to nag_lapack_zungqr (f08at):

```
[a, info] = f08at(a, tau);
```

but note that the second dimension of the array **a** must be at least **m**, which may be larger than was required by nag_lapack_zgeqrf (f08as).

When $m \geq n$, it is often only the first n columns of Q that are required, and they may be formed by the call:

```
[a, info] = f08at(a(:,1:n), tau);
```

To apply Q to an arbitrary complex rectangular matrix C, nag_lapack_zgeqrf (f08as) may be followed by a call to nag_lapack_zunmqr (f08au). For example,

```
[c, info] = f08au('Left', 'Conjugate Transpose', a, tau, c);
```

forms $C = Q^{\mathrm{H}}C$, where C is m by p.

To compute a QR factorization with column pivoting, use nag_lapack_zgeqpf (f08bs).

The real analogue of this function is nag_lapack_dgeqrf (f08ae).

9 Example

This example solves the linear least squares problems

$$\text{minimize } \|Ax_i - b_i\|_2, \quad i = 1, 2$$

where b_1 and b_2 are the columns of the matrix B,

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.03 + 0.96i & -0.91 + 2.06i & -0.05 + 0.41i \\ -0.98 + 1.98i & -1.20 + 0.19i & -0.66 + 0.42i & -0.81 + 0.56i \\ 0.62 - 0.46i & 1.01 + 0.02i & 0.63 - 0.17i & -1.11 + 0.60i \\ -0.37 + 0.38i & 0.19 - 0.54i & -0.98 - 0.36i & 0.22 - 0.20i \\ 0.83 + 0.51i & 0.20 + 0.01i & -0.17 - 0.46i & 1.47 + 1.59i \\ 1.08 - 0.28i & 0.20 - 0.12i & -0.07 + 1.23i & 0.26 + 0.26i \end{pmatrix}$$

and

$$B = \begin{pmatrix} -1.54 + 0.76i & 3.17 - 2.09i \\ 0.12 - 1.92i & -6.53 + 4.18i \\ -9.08 - 4.31i & 7.28 + 0.73i \\ 7.49 + 3.65i & 0.91 - 3.97i \\ -5.63 - 2.12i & -5.46 - 1.64i \\ 2.37 + 8.03i & -2.84 - 5.86i \end{pmatrix}.$$

9.1 Program Text

```
function f08as_example

fprintf('f08as example results\n\n');

a = [ 0.96 - 0.81i, -0.03 + 0.96i, -0.91 + 2.06i, -0.05 + 0.41i;
     -0.98 + 1.98i, -1.20 + 0.19i, -0.66 + 0.42i, -0.81 + 0.56i;
      0.62 - 0.46i,  1.01 + 0.02i,  0.63 - 0.17i, -1.11 + 0.60i;
     -0.37 + 0.38i,  0.19 - 0.54i, -0.98 - 0.36i,  0.22 - 0.20i;
      0.83 + 0.51i,  0.20 + 0.01i, -0.17 - 0.46i,  1.47 + 1.59i;
      1.08 - 0.28i,  0.20 - 0.12i, -0.07 + 1.23i,  0.26 + 0.26i];
b = [-2.09 + 1.93i,  3.26 - 2.70i;
      3.34 - 3.53i, -6.22 + 1.16i;
     -4.94 - 2.04i,  7.94 - 3.13i;
```

```
      0.17 + 4.23i,  1.04 - 4.26i;
     -5.19 + 3.63i, -2.31 - 2.12i;
      0.98 + 2.53i, -1.39 - 4.05i];
% Compute the QR factorization of A
[qr, tau, info] = f08as(a);

% Compute C = [c1;C2] = (Q^H)*B
[c, info] = f08au(...
                  'Left', 'Conjugate transpose', qr, tau, b);

% Compute least-squares solutions by backsubstitution in R*x = C1
[x, info] = f07ts(...
                  'Upper', 'No transpose', 'Non-Unit', qr(1:4,:), c(1:4,:));

fprintf('Least-squares solution(s)\n');
disp(x);

fprintf('Square root(s) of the residual sum(s) of squares\n');
for i=1:2
  fprintf('%8.3f ',norm(c(5:6,i)));
end
fprintf('\n');
```

9.2 Program Results

```
     f08as example results

Least-squares solution(s)
  -0.5044 - 1.2179i   0.7629 + 1.4529i
  -2.4281 + 2.8574i   5.1570 - 3.6089i
   1.4872 - 2.1955i  -2.6518 + 2.1203i
   0.4537 + 2.6904i  -2.7606 + 0.3318i

Square root(s) of the residual sum(s) of squares
   0.069    0.187
```