NAG Toolbox

nag_lapack_ztzrzf (f08bv)

1 Purpose

nag_lapack_ztzrzf (f08bv) reduces the m by n ($m \leq n$) complex upper trapezoidal matrix A to upper triangular form by means of unitary transformations.

2 Syntax

```
[a, tau, info] = nag_lapack_ztzrzf(a, 'm', m, 'n', n)
[a, tau, info] = f08bv(a, 'm', m, 'n', n)
```

3 Description

The m by n ($m \leq n$) complex upper trapezoidal matrix A given by

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

where R_1 is an m by m upper triangular matrix and R_2 is an m by $(n-m)$ matrix, is factorized as

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

where R is also an m by m upper triangular matrix and Z is an n by n unitary matrix.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

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 leading m by n upper trapezoidal part of the array **a** must contain the matrix to be factorized.

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

The leading m by m upper triangular part of **a** contains the upper triangular matrix R, and elements $\mathbf{m} + 1$ to **n** of the first m rows of **a**, with the array **tau**, represent the unitary matrix Z as a product of m elementary reflectors (see Section 3.2.6 in the F08 Chapter Introduction).

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

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

The scalar factors of the elementary reflectors.

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 floating-point operations is approximately $16m^2(n - m)$.

The real analogue of this function is nag_lapack_dtzrzf (f08bh).

9 Example

This example solves the linear least squares problems

$$\min_x \|b_j - Ax_j\|_2, \quad j = 1, 2$$

for the minimum norm solutions x_1 and x_2, where b_j is the jth column of the matrix B,

$$A = \begin{pmatrix} 0.47 - 0.34i & -0.40 + 0.54i & 0.60 + 0.01i & 0.80 - 1.02i \\ -0.32 - 0.23i & -0.05 + 0.20i & -0.26 - 0.44i & -0.43 + 0.17i \\ 0.35 - 0.60i & -0.52 - 0.34i & 0.87 - 0.11i & -0.34 - 0.09i \\ 0.89 + 0.71i & -0.45 - 0.45i & -0.02 - 0.57i & 1.14 - 0.78i \\ -0.19 + 0.06i & 0.11 - 0.85i & 1.44 + 0.80i & 0.07 + 1.14i \end{pmatrix}$$

and

$$B = \begin{pmatrix} -1.08 - 2.59i & 2.22 + 2.35i \\ -2.61 - 1.49i & 1.62 - 1.48i \\ 3.13 - 3.61i & 1.65 + 3.43i \\ 7.33 - 8.01i & -0.98 + 3.08i \\ 9.12 + 7.63i & -2.84 + 2.78i \end{pmatrix}.$$

The solution is obtained by first obtaining a QR factorization with column pivoting of the matrix A, and then the RZ factorization of the leading k by k part of R is computed, where k is the estimated rank of A. A tolerance of 0.01 is used to estimate the rank of A from the upper triangular factor, R.

Note that the block size (NB) of 64 assumed in this example is not realistic for such a small problem, but should be suitable for large problems.

9.1 Program Text

```
     function f08bv_example

fprintf('f08bv example results\n\n');

% Find Least squares solution of Ax = B, m>n
m = 5;
n = 4;
a = [  0.47 - 0.34i, -0.40 + 0.54i,  0.60 + 0.01i,  0.80 - 1.02i;
      -0.32 - 0.23i, -0.05 + 0.2i,  -0.26 - 0.44i, -0.43 + 0.17i;
       0.35 - 0.60i, -0.52 - 0.34i,  0.87 - 0.11i, -0.34 - 0.09i;
       0.89 + 0.71i, -0.45 - 0.45i, -0.02 - 0.57i,  1.14 - 0.78i;
      -0.19 + 0.06i,  0.11 - 0.85i,  1.44 + 0.80i,  0.07 + 1.14i];

b = [ -1.08 - 2.59i,  2.22 + 2.35i;
      -2.61 - 1.49i,  1.62 - 1.48i;
       3.13 - 3.61i,  1.65 + 3.43i;
       7.33 - 8.01i, -0.98 + 3.08i;
       9.12 + 7.63i, -2.84 + 2.78i];

% QR factorization of A with column pivoting = Q*(R1 R2 )*(P^T)
%                                                (0  R22)
[qr, jpvt, tau, info] = f08bt( ...
        a, zeros(n,1,nag_int_name));

% QRP'X = B, => RP'X = Q^HB = C; Compute C = Q^H B
[c, info] = f08au( ...
    'Left', 'Conjugate Transpose', qr, tau, b);

% Determine the rank, k, of R relative to tol;
% Choose tol to reflect the relative accuracy of the input data
tol = 0.01;
k = find(abs(diag(qr)) <= tol*abs(qr(1,1)));
if numel(k) == 0
  k = numel(diag(qr));
else
  k = k(1)-1;
end

fprintf('\nTolerance used to estimate the rank of a\n     %11.2e\n', tol);
fprintf('Estimated rank of a\n        %d\n', k);

% Compute the RZ (TZ) factorization of the first k rows of (R1 R2)
[rz, taurz, info] = f08bv( ...
    qr(1:k,:));
```

```
% Now, (TZ)P'X = C on first k rows of C
% Let ZP'X = T^{-1}C = Y (on first k rows)
y = zeros(n, 2);
y(1:k, :) = inv(triu(rz(1:k,1:k)))*c(1:k,:);

% ZP^T X = Y => P^T X = Z^H Y = W; Form W = Z^H Y.
[w, info] = f08bx( ...
    'Left', 'ConjTrans', nag_int(n-k), rz, taurz, y);

% P^T X = W => X = PW,
x = zeros(n, 2);
for i=1:n
   x(jpvt(i), :) = w(i, :);
end
fprintf('\nLeast-squares solution(s)\n');
disp(x);

% Compute estimates of the square roots of the residual sums of
% squares (2-norm of each of the columns of C2)
rnorm = [norm(c(k+1:m,1)), norm(c(k+1:m,2))];
fprintf('Square root(s) of the residual sum(s) of squares\n');
disp(rnorm);
```

9.2 Program Results

```
     f08bv example results

Tolerance used to estimate the rank of a
        1.00e-02
Estimated rank of a
        3

Least-squares solution(s)
   1.1669 - 3.3224i  -0.5023 + 1.8323i
   1.3486 + 5.5027i  -1.4418 - 1.6465i
   4.1764 + 2.3435i   0.2908 + 1.4900i
   0.6467 + 0.0107i  -0.2453 + 0.3951i

Square root(s) of the residual sum(s) of squares
    0.2513    0.0810
```
