NAG Toolbox

nag_lapack_zpoequ (f07ft)

1 Purpose

nag_lapack_zpoequ (f07ft) computes a diagonal scaling matrix S intended to equilibrate a complex n by n Hermitian positive definite matrix A and reduce its condition number.

2 Syntax

```
[s, scond, amax, info] = nag_lapack_zpoequ(a, 'n', n)
[s, scond, amax, info] = f07ft(a, 'n', n)
```

3 Description

nag_lapack_zpoequ (f07ft) computes a diagonal scaling matrix S chosen so that

$$s_j = 1/\sqrt{a_{jj}}.$$

This means that the matrix B given by

$$B = SAS,$$

has diagonal elements equal to unity. This in turn means that the condition number of B, $\kappa_2(B)$, is within a factor n of the matrix of smallest possible condition number over all possible choices of diagonal scalings (see Corollary 7.6 of Higham (2002)).

4 References

Higham N J (2002) *Accuracy and Stability of Numerical Algorithms* (2nd Edition) SIAM, Philadelphia

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 matrix A whose scaling factors are to be computed. Only the diagonal elements of the array **a** are referenced.

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 A.

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

5.3 Output Parameters

1: **s**$(\mathbf{n})$ – REAL (KIND=nag_wp) array

If **info** $= 0$, **s** contains the diagonal elements of the scaling matrix S.

2: **scond** – REAL (KIND=nag_wp)

If **info** $= 0$, **scond** contains the ratio of the smallest value of **s** to the largest value of **s**. If **scond** ≥ 0.1 and **amax** is neither too large nor too small, it is not worth scaling by S.

3: **amax** – REAL (KIND=nag_wp)

$\max|a_{ij}|$. If **amax** is very close to overflow or underflow, the matrix A should be scaled.

4: **info** – INTEGER

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

6 Error Indicators and Warnings

info < 0

If **info** $= -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

info > 0

The ⟨*value*⟩th diagonal element of A is not positive (and hence A cannot be positive definite).

7 Accuracy

The computed scale factors will be close to the exact scale factors.

8 Further Comments

The real analogue of this function is nag_lapack_dpoequ (f07ff).

9 Example

This example equilibrates the Hermitian positive definite matrix A given by

$$A = \begin{pmatrix} 3.23 & 1.51 - 1.92i & (1.90 + 0.84i) \times 10^5 & 0.42 + 2.50i \\ 1.51 + 1.92i & 3.58 & (-0.23 + 1.11i) \times 10^5 & -1.18 + 1.37i \\ (1.90 - 0.84i) \times 10^5 & (-0.23 - 1.11i) \times 10^5 & 4.09 \times 10^{10} & (2.33 - 0.14i) \times 10^5 \\ 0.42 - 2.50i & -1.18 - 1.37i & (2.33 + 0.14i) \times 10^5 & 4.29 \end{pmatrix}.$$

Details of the scaling factors and the scaled matrix are output.

9.1 Program Text

```
function f07ft_example

fprintf('f07ft example results\n\n');

a = [3.23 + 0i    1.51 - 1.92i  1.90e+05 + 8.40e+04i   0.42     + 2.50i;
     0    + 0i    3.58 + 0i    -2.30e+04 + 1.11e+05i  -1.18     + 1.37i;
     0    + 0i    0    + 0i     4.09e+10 + 0i          2.33e+05 - 1.40e+04i;
     0    + 0i    0    + 0i     0        + 0i          4.29     + 0i];

% Scale A
[s, scond, amax, info] = f07ft(a);

fprintf('scond = %8.1e, amax = %8.1e\n\n', scond, amax);
disp('Diagonal scaling factors');
fprintf('%10.1e',s);
fprintf('\n\n');
```

```
% Scaled matrix
as = diag(s)*a*diag(s);

[ifail] = x04da( ...
                 'Upper', 'Non-unit', as, 'Scaled matrix');
```

9.2 Program Results

```
     f07ft example results

scond =  8.9e-06, amax =  4.1e+10

Diagonal scaling factors
   5.6e-01   5.3e-01   4.9e-06   4.8e-01

 Scaled matrix
          1       2       3       4
 1   1.0000  0.4441  0.5227  0.1128
     0.0000 -0.5646  0.2311  0.6716

 2           1.0000 -0.0601 -0.3011
             0.0000  0.2901  0.3496

 3                   1.0000  0.5562
                     0.0000 -0.0334

 4                           1.0000
                             0.0000
```