NAG Toolbox

nag_lapack_zpbequ (f07ht)

1 Purpose

nag_lapack_zpbequ (f07ht) computes a diagonal scaling matrix S intended to equilibrate a complex n by n Hermitian positive definite band matrix A, with bandwidth $(2k_d + 1)$, and reduce its condition number.

2 Syntax

```
[s, scond, amax, info] = nag_lapack_zpbequ(uplo, kd, ab, 'n', n)
[s, scond, amax, info] = f07ht(uplo, kd, ab, 'n', n)
```

3 Description

nag_lapack_zpbequ (f07ht) 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: **uplo** – CHARACTER(1)

Indicates whether the upper or lower triangular part of A is stored in the array **ab**, as follows:

uplo = 'U'
The upper triangle of A is stored.

uplo = 'L'
The lower triangle of A is stored.

Constraint: **uplo** = 'U' or 'L'.

2: **kd** – INTEGER

k_d, the number of superdiagonals of the matrix A if **uplo** = 'U', or the number of subdiagonals if **uplo** = 'L'.

Constraint: **kd** ≥ 0.

3: **ab**$(ldab, :)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **ab** must be at least **kd** $+ 1$.

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

The upper or lower triangle of the Hermitian positive definite band matrix A whose scaling factors are to be computed.

The matrix is stored in rows 1 to $k_d + 1$, more precisely,

if **uplo** = 'U', the elements of the upper triangle of A within the band must be stored with element A_{ij} in $\mathbf{ab}(k_d + 1 + i - j, j)$ for $\max(1, j - k_d) \leq i \leq j$;

if **uplo** = 'L', the elements of the lower triangle of A within the band must be stored with element A_{ij} in $\mathbf{ab}(1 + i - j, j)$ for $j \leq i \leq \min(n, j + k_d)$.

Only the elements of the array **ab** corresponding to the diagonal elements of A are referenced. (Row $(k_d + 1)$ of **ab** when **uplo** = 'U', row 1 of **ab** when **uplo** = 'L'.)

5.2 Optional Input Parameters

1: **n** – INTEGER

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

n, the order of the matrix A.

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

5.3 Output Parameters

1: **s**(**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 $\langle$*value*$\rangle$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_dpbequ (f07hf).

9 Example

This example equilibrates the Hermitian positive definite matrix A given by

$$A = \begin{pmatrix} 9.39 & 1.08 - 1.73i & 0 & 0 \\ 1.08 + 1.73i & 1.69 & (-0.04 + 0.29i) \times 10^{10} & 0 \\ 0 & (-0.04 - 0.29i) \times 10^{10} & 2.65 \times 10^{20} & (-0.33 + 2.24i) \times 10^{10} \\ 0 & 0 & (-0.33 - 2.24i) \times 10^{10} & 2.17 \end{pmatrix}.$$

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

9.1 Program Text

```
     function f07ht_example

fprintf('f07ht example results\n\n');

% Symmetric banded A
uplo = 'U';
kd = nag_int(1);
n  = nag_int(4);
ab = [0,          1.08 - 1.73i,  -0.04e10 + 0.29e10i,  -0.33e10 + 2.24e10i;
      9.39 + 0i,  1.69 + 0i,      2.65e20 + 0i,         2.17    + 0i];

% Scale A
[s, scond, amax, info] = f07ht( ...
                                uplo, kd, ab);

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

% Apply scalings
asp = ab*diag(s);
for i = 1:n
  for j = 0:min(kd,n-i)
    asp(kd+1-j,i+j) = s(i)*asp(kd+1-j,i+j);
  end
end

kl = nag_int(0);
[ifail] = x04de( ...
                 n, n, kl, kd, asp, 'Scaled matrix');
```

9.2 Program Results

```
     f07ht example results

scond =  8.0e-11, amax =  2.6e+20

Diagonal scaling factors
   3.3e-01   7.7e-01   6.1e-11   6.8e-01

 Scaled matrix
             1          2          3          4
 1      1.0000     0.2711
        0.0000    -0.4343

 2                 1.0000    -0.0189
                   0.0000     0.1370

 3                            1.0000    -0.1376
                              0.0000     0.9341

 4                                       1.0000
                                         0.0000
```