

NAG C Library Function Document

nag_dpbcon (f07hgc)

1 Purpose

nag_dpbcon (f07hgc) estimates the condition number of a real symmetric positive-definite band matrix A , where A has been factorized by nag_dpbtrf (f07hdc).

2 Specification

```
void nag_dpbcon (Nag_OrderType order, Nag_UploType uplo, Integer n, Integer kd,
                 const double ab[], Integer pdab, double anorm, double *rcond, NagError *fail)
```

3 Description

nag_dpbcon (f07hgc) estimates the condition number (in the 1-norm) of a real symmetric positive-definite band matrix A :

$$\kappa_1(A) = \|A\|_1 \|A^{-1}\|_1.$$

Since A is symmetric, $\kappa_1(A) = \kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty$.

Because $\kappa_1(A)$ is infinite if A is singular, the function actually returns an estimate of the **reciprocal** of $\kappa_1(A)$.

The function should be preceded by a call to nag_dsb_norm (f16rec) to compute $\|A\|_1$ and a call to nag_dpbtrf (f07hdc) to compute the Cholesky factorization of A . The function then uses Higham's implementation of Hager's method (see Higham (1988)) to estimate $\|A^{-1}\|_1$.

4 References

Higham N J (1988) FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation *ACM Trans. Math. Software* **14** 381–396

5 Parameters

1: **order** – Nag_OrderType *Input*

On entry: the **order** parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = **Nag_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

Constraint: **order** = **Nag_RowMajor** or **Nag_ColMajor**.

2: **uplo** – Nag_UploType *Input*

On entry: indicates whether A has been factorized as $U^T U$ or LL^T as follows:

if **uplo** = **Nag_Upper**, $A = U^T U$, where U is upper triangular;

if **uplo** = **Nag_Lower**, $A = LL^T$, where L is lower triangular.

Constraint: **uplo** = **Nag_Upper** or **Nag_Lower**.

3: **n** – Integer *Input*

On entry: n , the order of the matrix A .

Constraint: $n \geq 0$.

- 4: **kd** – Integer *Input*
On entry: k , the number of super-diagonals or sub-diagonals of the matrix A .
Constraint: $\mathbf{kd} \geq 0$.
- 5: **ab**[*dim*] – const double *Input*
Note: the dimension, *dim*, of the array **ab** must be at least $\max(1, \mathbf{pdab} \times \mathbf{n})$.
On entry: the Cholesky factor of A , as returned by nag_dpbtrf (f07hdc).
- 6: **pdab** – Integer *Input*
On entry: the stride separating row or column elements (depending on the value of **order**) of the matrix in the array **ab**.
Constraint: $\mathbf{pdab} \geq \mathbf{kd} + 1$.
- 7: **anorm** – double *Input*
On entry: the 1-norm of the **original** matrix A , which may be computed by calling nag_dsb_norm (f16rec). **anorm** must be computed either **before** calling nag_dpbtrf (f07hdc) or else from a copy of the original matrix A .
Constraint: $\mathbf{anorm} \geq 0.0$.
- 8: **rcond** – double * *Output*
On exit: an estimate of the reciprocal of the condition number of A . **rcond** is set to zero if exact singularity is detected or the estimate underflows. If **rcond** is less than *machine precision*, A is singular to working precision.
- 9: **fail** – NagError * *Output*
The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** = $\langle value \rangle$.

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

On entry, **kd** = $\langle value \rangle$.

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

On entry, **pdab** = $\langle value \rangle$.

Constraint: $\mathbf{pdab} > 0$.

NE_INT_2

On entry, **pdab** = $\langle value \rangle$, **kd** = $\langle value \rangle$.

Constraint: $\mathbf{pdab} \geq \mathbf{kd} + 1$.

NE_REAL

On entry, **anorm** = $\langle value \rangle$.

Constraint: $\mathbf{anorm} \geq 0.0$.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter $\langle value \rangle$ had an illegal value.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

7 Accuracy

The computed estimate **rcond** is never less than the true value ρ , and in practice is nearly always less than 10ρ , although examples can be constructed where **rcond** is much larger.

8 Further Comments

A call to nag_dpbcon (f07hgc) involves solving a number of systems of linear equations of the form $Ax = b$; the number is usually 4 or 5 and never more than 11. Each solution involves approximately $4nk$ floating-point operations (assuming $n \gg k$) but takes considerably longer than a call to nag_dpbtrs (f07hec) with 1 right-hand side, because extra care is taken to avoid overflow when A is approximately singular.

The complex analogue of this function is nag_zpbcon (f07huc).

9 Example

To estimate the condition number in the 1-norm (or infinity-norm) of the matrix A , where

$$A = \begin{pmatrix} 5.49 & 2.68 & 0.00 & 0.00 \\ 2.68 & 5.63 & -2.39 & 0.00 \\ 0.00 & -2.39 & 2.60 & -2.22 \\ 0.00 & 0.00 & -2.22 & 5.17 \end{pmatrix}.$$

Here A is symmetric and positive-definite, and is treated as a band matrix, which must first be factorized by nag_dpbtrf (f07hdc). The true condition number in the 1-norm is 74.15.

9.1 Program Text

```
/* nag_dpbcon (f07hgc) Example Program.
 *
 * Copyright 2001 Numerical Algorithms Group.
 *
 * Mark 7, 2001.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf07.h>
#include <nagf16.h>
#include <nagx02.h>

int main(void)
{
    /* Scalars */
    Integer i, j, k, kd, n, pdab;
    Integer exit_status=0;
    double anorm, rcond;
    NagError fail;
    Nag_UploType uplo_enum;
    Nag_OrderType order;

    /* Arrays */
    char uplo[2];
    double *ab=0;
```

```

#ifdef NAG_COLUMN_MAJOR
#define AB_UPPER(I,J) ab[(J-1)*pdab + k + I - J - 1]
#define AB_LOWER(I,J) ab[(J-1)*pdab + I - J]
    order = Nag_ColMajor;
#else
#define AB_UPPER(I,J) ab[(I-1)*pdab + J - I]
#define AB_LOWER(I,J) ab[(I-1)*pdab + k + J - I - 1]
    order = Nag_RowMajor;
#endif

INIT_FAIL(fail);
Vprintf("f07hgc Example Program Results\n\n");

/* Skip heading in data file */
Vscanf("%*[^\\n] ");
Vscanf("%ld%ld%*[^\\n] ", &n, &kd);
pdab = kd + 1;

/* Allocate memory */
if ( !(ab = NAG_ALLOC((kd+1) * n, double)) )
{
    Vprintf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

/* Read A from data file */
Vscanf(" ' %ls '%*[^\\n] ", uplo);
if (*(unsigned char *)uplo == 'L')
    uplo_enum = Nag_Lower;
else if (*(unsigned char *)uplo == 'U')
    uplo_enum = Nag_Upper;
else
{
    Vprintf("Unrecognised character for Nag_UploType type\n");
    exit_status = -1;
    goto END;
}
k = kd + 1;
if (uplo_enum == Nag_Upper)
{
    for (i = 1; i <= n; ++i)
    {
        for (j = i; j <= MIN(i+kd,n); ++j)
            Vscanf("%lf", &AB_UPPER(i,j));
    }
    Vscanf("%*[^\\n] ");
}
else
{
    for (i = 1; i <= n; ++i)
    {
        for (j = MAX(1,i-kd); j <= i; ++j)
            Vscanf("%lf", &AB_LOWER(i,j));
    }
    Vscanf("%*[^\\n] ");
}

/* Compute norm of A */
f16rec(order, Nag_OneNorm, uplo_enum, n, kd, ab, pdab, &anorm, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f16rec.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Factorize A */
f07hdc(order, uplo_enum, n, kd, ab, pdab, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from f07hdc.\n%s\n", fail.message);
}

```

```

        exit_status = 1;
        goto END;
    }
    /* Estimate condition number */
    f07hgc(order, uplo_enum, n, kd, ab, pdab, anorm, &rcond, &fail);
    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from f07hgc.\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }
    if (rcond >= X02AJC)
        Vprintf("Estimate of condition number =%10.2e\n\n", 1.0/rcond);
    else
        Vprintf("A is singular to working precision\n");
END:
    if (ab) NAG_FREE(ab);
    return exit_status;
}

```

9.2 Program Data

f07hgc Example Program Data

```

4  1                      :Values of N and KD
'L'                      :Value of UPLO
5.49
2.68    5.63
        -2.39    2.60
                -2.22    5.17    :End of matrix A

```

9.3 Program Results

f07hgc Example Program Results

Estimate of condition number = 7.42e+01
