NAG Library Function Document

nag_dgbcon (f07bgc)

1 Purpose
nag_dgbcon (f07bgc) estimates the condition number of a real band matrix \( A \), where \( A \) has been factorized by nag_dgbtrf (f07bdc).

2 Specification

```
#include <nag.h>
#include <nagf07.h>

void nag_dgbcon (Nag_OrderType order, Nag_NormType norm, Integer n,
                Integer kl, Integer ku, const double ab[], Integer pdab,
                const Integer ipiv[], double anorm, double *rcond, NagError *fail)
```

3 Description
nag_dgbcon (f07bgc) estimates the condition number of a real band matrix \( A \), in either the 1-norm or the \( \infty \)-norm:

\[
\kappa_1(A) = \|A\|_1 \|A^{-1}\|_1 \quad \text{or} \quad \kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty.
\]

Note that \( \kappa_\infty(A) = \kappa_1(A^T) \).

Because the condition number is infinite if \( A \) is singular, the function actually returns an estimate of the reciprocal of the condition number.

The function should be preceded by a call to nag_dgb_norm (f16rbc) to compute \( \|A\|_1 \) or \( \|A\|_\infty \), and a call to nag_dgbtrf (f07bdc) to compute the \( LU \) factorization of \( A \). The function then uses Higham’s implementation of Hager’s method (see Higham (1988)) to estimate \( \|A^{-1}\|_1 \) or \( \|A^{-1}\|_\infty \).

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 Arguments

1: order – Nag_OrderType

- **Input**
  - On entry: the order argument 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 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.
  - **Constraint:** order = Nag_RowMajor or Nag_ColMajor.

2: norm – Nag_NormType

- **Input**
  - On entry: indicates whether \( \kappa_1(A) \) or \( \kappa_\infty(A) \) is estimated.
  - \( norm = Nag_OneNorm \)
  - \( \kappa_1(A) \) is estimated.
\textbf{norm} = \text{Nag\_InfNorm}

$\kappa_\infty(A)$ is estimated.

\textit{Constraint: norm} = \text{Nag\_OneNorm} or \text{Nag\_InfNorm}.

3: \textbf{n} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} \text{n}, the order of the matrix \text{A}.

\textit{Constraint:} \text{n} \geq 0.

4: \textbf{kl} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} \text{kl}, the number of subdiagonals within the band of the matrix \text{A}.

\textit{Constraint:} \text{kl} \geq 0.

5: \textbf{ku} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} \text{ku}, the number of superdiagonals within the band of the matrix \text{A}.

\textit{Constraint:} \text{ku} \geq 0.

6: \textbf{ab}[[\text{dim}]] – const double \hspace{1cm} \textit{Input}

\textit{Note:} the dimension, \text{dim}, of the array \text{ab} must be at least \text{max}(1, pdab \times \text{n}).

\textit{On entry:} the \text{LU} factorization of \text{A}, as returned by \text{nag\_dgbtrf} (f07bdc).

7: \textbf{pdab} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} the stride separating row or column elements (depending on the value of \text{order}) of the matrix in the array \text{ab}.

\textit{Constraint:} \text{pdab} \geq 2 \times \text{kl} + \text{ku} + 1.

8: \textbf{ipiv}[[\text{dim}]] – const Integer \hspace{1cm} \textit{Input}

\textit{Note:} the dimension, \text{dim}, of the array \text{ipiv} must be at least \text{max}(1, \text{n}).

\textit{On entry:} the pivot indices, as returned by \text{nag\_dgbtrf} (f07bdc).

9: \textbf{anorm} – double \hspace{1cm} \textit{Input}

\textit{On entry:} if \text{norm} = \text{Nag\_OneNorm}, the 1-norm of the \text{original} matrix \text{A}.

If \text{norm} = \text{Nag\_InfNorm}, the $\infty$-norm of the \text{original} matrix \text{A}.

\text{anorm} may be computed by calling \text{nag\_dgb\_norm} (f16rbc) with the same value for the argument \text{norm}.

\text{anorm} must be computed either \textbf{before} calling \text{nag\_dgbtrf} (f07bdc) or else from a \textbf{copy} of the original matrix \text{A} (see Section 10).

\textit{Constraint:} \text{anorm} \geq 0.0.

10: \textbf{rcond} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} an estimate of the reciprocal of the condition number of \text{A}. \text{rcond} is set to zero if exact singularity is detected or the estimate underflows. If \text{rcond} is less than \textit{machine precision}, \text{A} is singular to working precision.

11: \textbf{fail} – NagError * \hspace{1cm} \textit{Input/Output}

The NAG error argument (see Section 3.6 in the Essential Introduction).
6 Error Indicators and Warnings

**NE_ALLOC_FAIL**
Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

**NE_BAD_PARAM**
On entry, argument \(\langle\text{value}\rangle\) had an illegal value.

**NE_INT**
On entry, \(\text{kl} = \langle\text{value}\rangle\).
Constraint: \(\text{kl} \geq 0\).

On entry, \(\text{ku} = \langle\text{value}\rangle\).
Constraint: \(\text{ku} \geq 0\).

On entry, \(\text{n} = \langle\text{value}\rangle\).
Constraint: \(\text{n} \geq 0\).

On entry, \(\text{pdab} = \langle\text{value}\rangle\).
Constraint: \(\text{pdab} > 0\).

**NE_INT_3**
On entry, \(\text{pdab} = \langle\text{value}\rangle\), \(\text{kl} = \langle\text{value}\rangle\) and \(\text{ku} = \langle\text{value}\rangle\).
Constraint: \(\text{pdab} \geq 2 \times \text{kl} + \text{ku} + 1\).

**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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

**NE_NO_LICENCE**
Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

**NE_REAL**
On entry, \(\text{anorm} = \langle\text{value}\rangle\).
Constraint: \(\text{anorm} \geq 0.0\).

7 Accuracy
The computed estimate \(\text{rcond}\) is never less than the true value \(\rho\), and in practice is nearly always less than \(10^\rho\), although examples can be constructed where \(\text{rcond}\) is much larger.

8 Parallelism and Performance
nag_dgbecon (f07bgc) is not threaded by NAG in any implementation.

nag_dgbecon (f07bgc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.
9 Further Comments

A call to nag_dgbcon (f07bgc) involves solving a number of systems of linear equations of the form \(Ax = b\) or \(A^T x = b\); the number is usually 4 or 5 and never more than 11. Each solution involves approximately \(2n(2k_l + k_u)\) floating-point operations (assuming \(n \gg k_l\) and \(n \gg k_u\)) but takes considerably longer than a call to nag_dgbtrs (f07bec) with one right-hand side, because extra care is taken to avoid overflow when \(A\) is approximately singular.

The complex analogue of this function is nag_zgbcon (f07buc).

10 Example

This example estimates the condition number in the 1-norm of the matrix \(A\), where

\[
A = \begin{pmatrix}
-0.23 & 2.54 & -3.66 & 0.00 \\
-6.98 & 2.46 & -2.73 & -2.13 \\
0.00 & 2.56 & 2.46 & 4.07 \\
0.00 & 0.00 & -4.78 & -3.82 \\
\end{pmatrix}
\]

Here \(A\) is nonsymmetric and is treated as a band matrix, which must first be factorized by nag_dgbtrf (f07bdc). The true condition number in the 1-norm is 56.40.

10.1 Program Text

/* nag_dgbcon (f07bgc) Example Program. */
* Copyright 2014 Numerical Algorithms Group.
*/
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf07.h>
#include <nagx02.h>
int main(void)
{
    /* Scalars */
    Integer i, ipiv_len, j, kl, ku, n, pdab;
    Integer exit_status = 0;
    double anorm, rcond, sum;
    NagError fail;
    Nag_OrderType order;

    /* Arrays */
    double *ab = 0;
    Integer *ipiv = 0;

    /* Scalars */
    ifdef NAG_COLUMN_MAJOR
    #define AB(I, J) ab[(J-1)*pdab + kl + ku + I - J]
    order = Nag_ColMajor;
    #else
    #define AB(I, J) ab[(I-1)*pdab + kl + J - I]
    order = Nag_RowMajor;
    endif

    INIT_FAIL(fail);

    printf("nag_dgbcon (f07bgc) Example Program Results\n\n");

    /* Skip heading in data file */
    ifndef _WIN32
    scanf_s("%*[\n \n ");
    else
    scanf("%*[\n \n ");

f07bgc.4  Mark 25

NAG Library Manual
#ifdef _WIN32
    scanf_s("%NAG_IFMT"%NAG_IFMT"%NAG_IFMT"%[\n] ", &n, &kl, &ku);
#else
    scanf("%NAG_IFMT"%NAG_IFMT"%NAG_IFMT"%[\n] ", &n, &kl, &ku);
#endif

ipiv_len = n;
pdab = 2*kl + ku + 1;

/* Allocate memory */
if (!(ab = NAG_ALLOC((2*kl+ku+1) * n, double)) ||
    !(ipiv = NAG_ALLOC(ipiv_len, Integer)))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

/* Read AB from data file */
for (i = 1; i <= n; ++i)
{
    for (j = MAX(i-kl, 1); j <= MIN(i+ku, n); ++j)
    #ifdef _WIN32
        scanf_s("%lf", &AB(i, j));
    #else
        scanf("%lf", &AB(i, j));
    #endif
    }
# ifdef _WIN32
    scanf_s("%*[\n] ");
#else
    scanf("%*[\n] ");
#endif
# endif

/* Compute norm of A */
anorm = 0.0;
for (j = 1; j <= n; ++j)
{
    sum = 0.0;
    for (i = MAX(j-ku, 1); i <= MIN(j+k1, n); ++i)
        sum = sum + ABS(AB(i, j));
anorm = MAX(anorm, sum);
}

/* Factorize A */
/* nag_dgbtrf (f07bdc).
 * LU factorization of real m by n band matrix
 */
nag_dgbtrf(order, n, n, kl, ku, ab, pdab, ipiv, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_dgbtrf (f07bdc).\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Estimate condition number */
/* nag_dgbcon (f07bgc).
 * Estimate condition number of real band matrix, matrix
 * already factorized by nag_dgbtrf (f07bdc)
 */
nag_dgbcon(order, Nag_OneNorm, n, kl, ku, ab, pdab, ipiv, anorm, &rcond, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_dgbcon (f07bgc).\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print condition number */
/* nag_machine_precision (x02ajc).
 * The machine precision
 */
if (rcond > nag_machine_precision)

Mark 25

f07bgc.5
printf("Estimate of condition number = %11.2e\n", 1.0/rcond);
else
printf("A is singular to working precision\n");
END:
NAG_FREE(ab);
NAG_FREE(ipiv);
return exit_status;
}

10.2 Program Data

nag_dgbcon (f07bgc) Example Program Data
4 1 2 :Values of N, KL and KU
-0.23  2.54  -3.66
-6.98  2.46  -2.73  -2.13
  2.56  2.46  4.07
   -4.78 -3.82 :End of matrix A

10.3 Program Results

nag_dgbcon (f07bgc) Example Program Results

Estimate of condition number = 5.64e+01