

NAG Library Function Document

nag_dgb_norm (f16rbc)

1 Purpose

nag_dgb_norm (f16rbc) calculates the value of the 1-norm, the ∞ -norm, the Frobenius norm or the maximum absolute value of the elements of a real m by n band matrix, stored in banded form.

2 Specification

```
#include <nag.h>
#include <nagf16.h>

void nag_dgb_norm (Nag_OrderType order, Nag_NormType norm, Integer m,
                  Integer n, Integer kl, Integer ku, const double ab[], Integer pdab,
                  double *r, NagError *fail)
```

3 Description

Given a real m by n banded matrix, A , nag_dgb_norm (f16rbc) calculates one of the values given by

$$\|A\|_1 = \max_j \sum_{i=1}^m |a_{ij}| \quad (\text{the 1-norm of } A),$$

$$\|A\|_\infty = \max_i \sum_{j=1}^n |a_{ij}| \quad (\text{the } \infty\text{-norm of } A),$$

$$\|A\|_F = \left(\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2 \right)^{1/2} \quad (\text{the Frobenius norm of } A), \quad \text{or}$$

$$\max_{i,j} |a_{ij}| \quad (\text{the maximum absolute element value of } A).$$

4 References

Basic Linear Algebra Subprograms Technical (BLAST) Forum (2001) *Basic Linear Algebra Subprograms Technical (BLAST) Forum Standard* University of Tennessee, Knoxville, Tennessee <http://www.netlib.org/blas/blast-forum/blas-report.pdf>

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: specifies the value to be returned.

norm = Nag_OneNorm
The 1-norm.

norm = Nag_InfNorm
The ∞ -norm.

norm = Nag_FrobeniusNorm
The Frobenius (or Euclidean) norm.

norm = Nag_MaxNorm
The value $\max_{i,j} |a_{ij}|$ (not a norm).

Constraint: **norm** = Nag_OneNorm, Nag_TwoNorm, Nag_FrobeniusNorm, Nag_InfNorm or Nag_MaxNorm.

3: **m** – Integer *Input*

On entry: m , the number of rows of the matrix A .

Constraint: $m \geq 0$.

4: **n** – Integer *Input*

On entry: n , the number of columns of the matrix A .

Constraint: $n \geq 0$.

5: **kl** – Integer *Input*

On entry: k_l , the number of subdiagonals within the band of A .

Constraint: $kl \geq 0$.

6: **ku** – Integer *Input*

On entry: k_u , the number of superdiagonals within the band of A .

Constraint: $ku \geq 0$.

7: **ab**[*dim*] – const double *Input*

Note: the dimension, *dim*, of the array **ab** must be at least

$\max(1, \mathbf{pdab} \times \mathbf{n})$ when **order** = Nag_ColMajor;
 $\max(1, \mathbf{m} \times \mathbf{pdab})$ when **order** = Nag_RowMajor.

On entry: the m by n band matrix A .

This is stored as a notional two-dimensional array with row elements or column elements stored contiguously. The storage of elements A_{ij} , for row $i = 1, \dots, m$ and column $j = \max(1, i - k_l), \dots, \min(n, i + k_u)$, depends on the **order** argument as follows:

if **order** = 'Nag_ColMajor', A_{ij} is stored as **ab**[($j - 1$) \times **pdab** + **ku** + $i - j$];

if **order** = 'Nag_RowMajor', A_{ij} is stored as **ab**[($i - 1$) \times **pdab** + **kl** + $j - i$].

8: **pdab** – Integer *Input*

On entry: the stride separating row or column elements (depending on the value of **order**) of the matrix A in the array **ab**.

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

9: **r** – double * *Output*

On exit: the value of the norm specified by **norm**.

10: **fail** – NagError * *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.

NE_BAD_PARAM

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

NE_INT

On entry, $\mathbf{kl} = \langle value \rangle$.

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

On entry, $\mathbf{ku} = \langle value \rangle$.

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

On entry, $\mathbf{m} = \langle value \rangle$.

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

On entry, $\mathbf{n} = \langle value \rangle$.

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

NE_INT_3

On entry, $\mathbf{pdab} = \langle value \rangle$, $\mathbf{kl} = \langle value \rangle$, $\mathbf{ku} = \langle value \rangle$.

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

7 Accuracy

The BLAS standard requires accurate implementations which avoid unnecessary over/underflow (see Section 2.7 of Basic Linear Algebra Subprograms Technical (BLAST) Forum (2001)).

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

Calculates the various norms of a 6 by 4 banded matrix with two subdiagonals and one superdiagonal.

10.1 Program Text

```
/* nag_dgb_norm (f16rbc) Example Program.
 *
 * Copyright 2005 Numerical Algorithms Group.
 *
 * Mark 8, 2005.
 */

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

int main(void)
{
    /* Scalars */
```

```

double      r_one, r_inf, r_f, r_max;
Integer     ab_size, exit_status, i, j, kl, ku;
Integer     m, n, pdab;

/* Arrays */
double      *ab = 0;

/* Nag Types */
NagError    fail;
Nag_OrderType order;

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

exit_status = 0;
INIT_FAIL(fail);

printf("nag_dgb_norm (f16rbc) Example Program Results\n\n");

/* Skip heading in data file */
scanf("%*[\n] ");

/* Read the problem dimensions */
scanf("%ld%ld%ld%ld%*[\n] ",
      &m, &n, &kl, &ku);

pdab = kl + ku + 1;
#ifdef NAG_COLUMN_MAJOR
    ab_size = pdab*n;
#else
    ab_size = pdab*m;
#endif

if (m > 0 && n > 0)
{
    /* Allocate memory */
    if (!(ab = NAG_ALLOC(ab_size, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }
}
else
{
    printf("Invalid m or n\n");
    exit_status = 1;
    return exit_status;
}

/* Input matrix A. */

for (i = 1; i <= m; ++i)
{
    for (j = MAX(1, i-kl); j <= MIN(n, i+ku); ++j)
        scanf("%lf", &AB(i, j));
    scanf("%*[\n] ");
}

/* nag_dgb_norm (f16rbc).
 * calculates norm of real valued general band matrix.
 */
nag_dgb_norm(order, Nag_OneNorm, m, n, kl, ku, ab, pdab, &r_one, &fail);

if (fail.code != NE_NOERROR) goto GB_FAIL;

```

```

nag_dgb_norm(order, Nag_InfNorm, m, n, kl, ku, ab, pdab, &r_inf, &fail);
if (fail.code != NE_NOERROR) goto GB_FAIL;

nag_dgb_norm(order, Nag_FrobeniusNorm, m, n, kl, ku, ab, pdab, &r_f, &fail);
if (fail.code != NE_NOERROR) goto GB_FAIL;

nag_dgb_norm(order, Nag_MaxNorm, m, n, kl, ku, ab, pdab, &r_max, &fail);
if (fail.code != NE_NOERROR) goto GB_FAIL;

/* Print norms of A. */
printf(" Norms of banded matrix A:\n\n");
printf(" One norm          = %7.4f\n", r_one);
printf(" Infinity norm     = %7.4f\n", r_inf);
printf(" Frobenius norm     = %7.4f\n", r_f);
printf(" Maximum norm      = %7.4f\n", r_max);
goto END;

GB_FAIL:
printf("Error from nag_dgb_norm.\n%s\n", fail.message);
exit_status = 1;

END:
NAG_FREE(ab);

return exit_status;
}

```

10.2 Program Data

```

nag_dgb_norm (f16rbc) Example Program Data
  6 4 2 1          :Values of m, n, kl, ku
  1.0  1.0
  2.0  2.0  2.0
  3.0  3.0  3.0  3.0
    4.0  4.0  4.0
      5.0  5.0
        6.0 : the end of matrix A

```

10.3 Program Results

```

nag_dgb_norm (f16rbc) Example Program Results

Norms of banded matrix A:

One norm          = 18.0000
Infinity norm     = 12.0000
Frobenius norm    = 13.5647
Maximum norm      =  6.0000

```
