

## NAG Library Function Document

### nag\_dge\_norm (f16rac)

#### 1 Purpose

nag\_dge\_norm (f16rac) calculates the value of the 1-norm, the  $\infty$ -norm, the Frobenius norm, or the maximum absolute value of the elements of a real  $m$  by  $n$  matrix.

#### 2 Specification

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

void nag_dge_norm (Nag_OrderType order, Nag_NormType norm, Integer m,
                  Integer n, const double a[], Integer pda, double *r, NagError *fail)
```

#### 3 Description

Given a real  $m$  by  $n$  matrix,  $A$ , nag\_dge\_norm (f16rac) calculates one of the values given by

$$\|A\|_1 = \max_j \sum_{i=1}^m |a_{ij}|,$$

$$\|A\|_\infty = \max_i \sum_{j=1}^n |a_{ij}|,$$

$$\|A\|_F = \left( \sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2 \right)^{1/2}$$

or

$$\max_{i,j} |a_{ij}|.$$

#### 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  $\infty$ -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\_InfNorm, Nag\_FrobeniusNorm or Nag\_MaxNorm.
- 3: **m** – Integer *Input*  
*On entry:*  $m$ , the number of rows of the matrix  $A$ .  
 If  $m = 0$ , then **m** is set to zero.  
*Constraint:* **m**  $\geq 0$ .
- 4: **n** – Integer *Input*  
*On entry:*  $n$ , the number of columns of the matrix  $A$ .  
 If  $n = 0$ , then **n** is set to zero.  
*Constraint:* **n**  $\geq 0$ .
- 5: **a**[*dim*] – const double *Input*  
**Note:** the dimension, *dim*, of the array **a** must be at least  
 $\max(1, \mathbf{pda} \times \mathbf{n})$  when **order** = Nag\_ColMajor;  
 $\max(1, \mathbf{m} \times \mathbf{pda})$  when **order** = Nag\_RowMajor.  
 If **order** = 'Nag\_ColMajor',  $A_{ij}$  is stored in **a**[( $j - 1$ )  $\times$  **pda** +  $i - 1$ ].  
 If **order** = 'Nag\_RowMajor',  $A_{ij}$  is stored in **a**[( $i - 1$ )  $\times$  **pda** +  $j - 1$ ].  
*On entry:* the  $m$  by  $n$  matrix  $A$ .
- 6: **pda** – Integer *Input*  
*On entry:* the stride separating row or column elements (depending on the value of **order**) in the array **a**.  
*Constraints:*  
 if **order** = Nag\_ColMajor, **pda**  $\geq \max(1, \mathbf{m})$ ;  
 if **order** = Nag\_RowMajor, **pda**  $\geq \mathbf{n}$ .
- 7: **r** – double \* *Output*  
*On exit:* the value of the norm specified by **norm**.
- 8: **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,  $\mathbf{m} = \langle value \rangle$ .

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

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

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

### NE\_INT\_2

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

Constraint:  $\mathbf{pda} \geq \max(1, \mathbf{m})$ .

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

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

### 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.

## 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

See Section 10 in nag\_dgecon (f07age) and nag\_dtrsna (f08qlc).

---