1 Purpose

nag_zhb_norm (f16uec) calculates the value of the 1-norm, the ∞-norm, the Frobenius norm or the maximum absolute value of the elements of a complex $n$ by $n$ Hermitian band matrix.

2 Specification

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

void nag_zhb_norm (Nag_OrderType order, Nag_NormType norm, Nag_UploType uplo, Integer n, Integer k, const Complex ab[], Integer pdab, double *r, NagError *fail)

3 Description

Given a complex $n$ by $n$ Hermitian band matrix, $A$, nag_zhb_norm (f16uec) calculates one of the values given by

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

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

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

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

Note that, since $A$ is symmetric, $\|A\|_1 = \|A\|_\infty$.

4 References


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 \( \max_{ij} |a_{ij}| \) (not a norm).

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

3: uplo – Nag_UploType
   Input
   On entry: specifies whether the upper or lower triangular part of \( A \) is stored.

   uplo = Nag_Upper
       The upper triangular part of \( A \) is stored.

   uplo = Nag_Lower
       The lower triangular part of \( A \) is stored.

Constraint: uplo = Nag_Upper or Nag_Lower.

4: n – Integer
   Input
   On entry: \( n \), the order of the matrix \( A \).
   If \( n = 0 \), then \( n \) is set to zero.

Constraint: \( n \geq 0 \).

5: k – Integer
   Input
   On entry: \( k \), the number of subdiagonals or superdiagonals of the matrix \( A \).

Constraint: \( k \geq 0 \).

6: ab[dim] – const Complex
   Input
   Note: the dimension, \( dim \), of the array \( ab \) must be at least \( \max(1,\text{pdab} \times n) \).
   On entry: the \( n \) by \( n \) Hermitian band matrix \( A \).

This is stored as a notional two-dimensional array with row elements or column elements stored contiguously. The storage of elements of \( A_{ij} \), depends on the order and uplo arguments as follows:

if order = Nag_ColMajor and uplo = Nag_Upper,
   \( A_{ij} \) is stored in \( ab[k + i - j + (j - 1) \times \text{pdab}] \), for \( j = 1, \ldots, n \) and \( i = \max(1, j-k), \ldots, j \);

if order = Nag_ColMajor and uplo = Nag_Lower,
   \( A_{ij} \) is stored in \( ab[i - j + (j - 1) \times \text{pdab}] \), for \( j = 1, \ldots, n \) and \( i = j, \ldots, \min(n, j+k) \);
if order = Nag_RowMajor and uplo = Nag_Upper, 
\[ A_{ij} \text{ is stored in } ab[j - i + (i - 1) \times pdab], \text{ for } i = 1, \ldots, n \text{ and } j = i, \ldots, \min(n, i + k); \]
if order = Nag_RowMajor and uplo = Nag_Lower, 
\[ A_{ij} \text{ is stored in } ab[k + j - i + (i - 1) \times pdab], \text{ for } i = 1, \ldots, n \text{ and } j = \max(1, i - k), \ldots, i. \]

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

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

\textit{Constraint:} \( \text{pdab} \geq k + 1 \).

8: \textbf{r} – double * \hspace{1cm} \textbf{Output}

\textit{On exit:} the value of the norm specified by \textbf{norm}.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

\textbf{NE_ALLOC_FAIL}

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

\textbf{NE_BAD_PARAM}

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

\textbf{NE_INT}

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

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

\textbf{NE_INT_2}

On entry, \( \text{pdab} = \langle \text{value} \rangle, k = \langle \text{value} \rangle \).
Constraint: \( \text{pdab} \geq k + 1 \).

\textbf{NE_INTERNAL_ERROR}

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

\textbf{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.

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_zpbcon (f07huc).