NAG C Library Function Document

nag_ztr_copy (f16tec)

1 Purpose

nag_ztr_copy (f16tec) copies a complex triangular matrix.

2 Specification

```
void nag_ztr_copy (Nag_OrderType order, Nag_UploType uplo, Nag_TransType trans,
     Nag_DiagType diag, Integer n, const Complex a[], Integer pda, Complex b[],
     Integer pdb, NagError *fail)
```

3 Description

nag_ztr_copy (f16tec) performs the triangular matrix copy operations

$$B \leftarrow A, \quad B \leftarrow A^T \quad \text{or} \quad B \leftarrow A^H.$$

where A and B are n by n complex triangular matrices.

4 References

The BLAS Technical Forum Standard (2001) *www.netlib.org/blas/blast-forum*

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: specifies whether the upper or lower triangular part of A is stored as follows:

if **uplo** = **Nag_Upper**, the upper triangular part of A is stored;

if **uplo** = **Nag_Lower**, the lower triangular part of A is stored.

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

3: **trans** – Nag_TransType *Input*

On entry: specifies the operation to be performed as follows:

if **trans** = **Nag_NoTrans**, $B \leftarrow A$;

if **trans** = **Nag_Trans**, $B \leftarrow A^T$;

if **trans** = **Nag_ConjTrans**, $B \leftarrow A^H$.

Constraint: **trans** = **Nag_NoTrans**, **Nag_Trans** or **Nag_ConjTrans**.

4: **diag** – Nag_DiagType *Input*

On entry: specifies whether A has non-unit or unit diagonal elements, as follows:

if **diag** = **Nag_NonUnitDiag**, the diagonal elements are stored explicitly;

if **diag** = **Nag_UnitDiag**, the diagonal elements are assumed to be 1, and are not referenced.

Constraint: **diag** = **Nag_NonUnitDiag** or **Nag_UnitDiag**.

5: **n** – Integer *Input*

On entry: n, the order of the matrices A and B.

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

6: **a**[dim] – const Complex *Input*

Note: the dimension, dim, of the array **a** must be at least $\max(1, \mathbf{pda} \times \mathbf{n})$.

If **order** = **Nag_ColMajor**, the (i,j)th element of the matrix A is stored in **a**$[(j-1) \times \mathbf{pda} + i - 1]$ and if **order** = **Nag_RowMajor**, the (i,j)th element of the matrix A is stored in **a**$[(i-1) \times \mathbf{pda} + j - 1]$.

On entry: the n by n triangular matrix A. If **uplo** = **Nag_Upper**, A is upper triangular and the elements of the array below the diagonal are not referenced; if **uplo** = **Nag_Lower**, A is lower triangular and the elements of the array above the diagonal are not referenced.

7: **pda** – Integer *Input*

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

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

8: **b**[dim] – Complex *Output*

Note: the dimension, dim, of the array **b** must be at least $\max(1, \mathbf{pdb} \times \mathbf{n})$.

If **order** = **Nag_ColMajor**, the (i,j)th element of the matrix B is stored in **b**$[(j-1) \times \mathbf{pdb} + i - 1]$ and if **order** = **Nag_RowMajor**, the (i,j)th element of the matrix B is stored in **b**$[(i-1) \times \mathbf{pdb} + j - 1]$.

On exit: the n by n triangular matrix B. If **uplo** = **Nag_Upper** B is upper triangular and the elements of the array below the diagonal are not set; if **uplo** = **Nag_Lower** B is lower triangular and the elements of the array above the diagonal are not set.

9: **pdb** – Integer *Input*

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

Constraint: $\mathbf{pdb} \geq \max(1, \mathbf{n})$.

10: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: $\mathbf{n} \geq 0$.

On entry, **pda** = ⟨*value*⟩.
Constraint: $\mathbf{pda} \geq \max(1, \mathbf{n})$.

On entry, **pdb** = ⟨*value*⟩.
Constraint: $\mathbf{pdb} \geq \max(1, \mathbf{n})$.

NE_BAD_PARAM

On entry, parameter ⟨*value*⟩ had an illegal value.

7 Accuracy

The BLAS standard requires accurate implementations which avoid unnecessary over/underflow (see section 2.7 of The BLAS Technical Forum Standard (2001)).

8 Further Comments

None.

9 Example

None.