NAG C Library Function Document

nag_ztrsm (f16zjc)

1 Purpose

nag_ztrsm (f16zjc) solves one of the matrix equations

\[ TX = \alpha B, \quad T^T X = \alpha B, \quad T^H X = \alpha B, \quad XT = \alpha B, \quad XT^T = \alpha B \text{ or } XT^H = \alpha B, \]

where \( X \) and \( B \) are \( m \) by \( n \) complex matrices and \( T \) is a complex triangular matrix.

2 Specification

```c
void nag_ztrsm (Nag_OrderType order, Nag_SideType side, Nag_UploType uplo,
               Nag_TransType transt, Nag_DiagType diag, Integer m, Integer n,
               Complex alpha, const Complex t[], Integer pdt, Complex b[],
               Integer pdb, NagError *fail)
```

3 Description

nag_ztrsm (f16zjc) performs one of the matrix-matrix operations

\[ B \leftarrow \alpha T^{-1} B, \quad B \leftarrow \alpha T^{-T} B, \quad B \leftarrow \alpha T^{-H} B, \]
\[ B \leftarrow \alpha BT^{-1}, \quad B \leftarrow \alpha BT^{-T} \text{ or } B \leftarrow \alpha BT^{-H}, \]

where \( T \) is a complex triangular matrix, \( B \) is an \( m \) by \( n \) complex matrix, and \( \alpha \) is a complex scalar. \( T^{-T} \) denotes \( (T^T)^{-1} \) or equivalently \( (T^{-1})^T \); \( T^{-H} \) denotes \( (T^H)^{-1} \) or equivalently \( (T^{-1})^H \).

4 References


5 Parameters

1: \( \text{order} \) – Nag_OrderType \hspace{1cm} \text{Input}

\( \text{On entry:} \) the \( \text{order} \) 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 \( \text{order} = \text{Nag_RowMajor}. \) See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

\( \text{Constraint:} \ \text{order} = \text{Nag_RowMajor} \text{ or } \text{Nag_ColMajor}. \)

2: \( \text{side} \) – Nag_SideType \hspace{1cm} \text{Input}

\( \text{On entry:} \) specifies whether \( B \) is operated on from the left or the right, as follows:

\( \text{if} \ \text{side} = \text{Nag_LeftSide}, \) \( B \) is pre-multiplied from the left;

\( \text{if} \ \text{side} = \text{Nag_RightSide}, \) \( B \) is post-multiplied from the right.

\( \text{Constraint:} \ \text{side} = \text{Nag_LeftSide} \text{ or } \text{Nag_RightSide}. \)

3: \( \text{uplo} \) – Nag_UploType \hspace{1cm} \text{Input}

\( \text{On entry:} \) specifies whether \( T \) is upper or lower triangular as follows:

\( \text{if} \ \text{uplo} = \text{Nag_Upper}, \) \( T \) is upper triangular;

\( \text{if} \ \text{uplo} = \text{Nag_Lower}, \) \( T \) is lower triangular.

\( \text{Constraint:} \ \text{uplo} = \text{Nag_Upper} \text{ or } \text{Nag_Lower}. \)
4: transt – Nag_TransType  
On entry: specifies the operation to be performed as follows:
   if side = Nag_LeftSide and transt = Nag_Trans, \( B \leftarrow \alpha T^{-T}B \);
   if side = Nag_LeftSide and transt = Nag_NoTrans, \( b \leftarrow \alpha T^{-1}B \);
   if side = Nag_LeftSide and transt = Nag_ConjTrans, \( B \leftarrow \alpha T^{-H}B \);
   if side = Nag_RightSide and transt = Nag_Trans, \( B \leftarrow \alpha BT^{-T} \);
   if side = Nag_RightSide and transt = Nag_NoTrans, \( B \leftarrow \alpha BT^{-1} \).
   if side = Nag_RightSide and transt = Nag_ConjTrans, \( B \leftarrow \alpha BT^{-H} \).

Constraint: side = Nag_LeftSide or Nag_RightSide; transt = Nag_NoTrans or Nag_Trans.

5: diag – Nag_DiagType  
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.

6: m – Integer  
On entry: \( m \), the number of rows of the matrix \( B \); the order of \( T \) if side = Nag_LeftSide.

Constraint: \( m \geq 0 \).

7: n – Integer  
On entry: \( n \), the number of columns of the matrix \( B \); the order of \( T \) if side = Nag_RightSide.

Constraint: \( n \geq 0 \).

8: alpha – Complex  
On entry: the scalar \( \alpha \).

9: t[dim] – const Complex  
Note: the dimension, \( dim \), of the array \( t \) must be at least \( \max(1, pdt \times m) \) when side = Nag_LeftSide and at least \( \max(1, pdt \times n) \) when side = Nag_RightSide.

If order = Nag_ColMajor, the \((i, j)\)th element of the matrix \( T \) is stored in \( t[(j - 1) \times pdt + i - 1] \) and if order = Nag_RowMajor, the \((i, j)\)th element of the matrix \( T \) is stored in \( t[(i - 1) \times pdt + j - 1] \).

On entry: the \( m \) by \( m \) triangular matrix \( T \) if side = Nag_LeftSide or \( n \) by \( n \) triangular matrix \( T \) if side = Nag_RightSide. If uplo = Nag_Upper, \( T \) is upper triangular and the elements of the array below the diagonal are not referenced; if uplo = Nag_Lower, \( T \) is lower triangular and the elements of the array above the diagonal are not referenced. If diag = Nag_UnitDiag, the diagonal elements of \( T \) are not referenced, but are assumed to be 1.

10: pdt – Integer  
On entry: the stride separating row or column elements (depending on the value of order) of the matrix \( T \) in the array \( t \).

Constraints:
   if side = Nag_LeftSide, \( pdt \geq \max(1,m) \);
   if side = Nag_RightSide, \( pdt \geq \max(1,n) \).
11: \( \mathbf{b}[\text{dim}] \) – Complex

*Input/Output*

**Note:** the dimension, \( \text{dim} \), of the array \( \mathbf{b} \) must be at least \( \max(1, \text{pdb} \times n) \) when \( \text{order} = \text{Nag\_ColMajor} \) and at least \( \max(1, \text{pdb} \times m) \) when \( \text{order} = \text{Nag\_RowMajor} \).

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

*On entry:* the \( m \) by \( n \) matrix \( \mathbf{B} \). If \( \text{alpha} = 0 \), \( \mathbf{b} \) need not be set.

*On exit:* the updated matrix \( \mathbf{B} \).

12: \( \text{pdb} \) – Integer

*Input*

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

**Constraints:**

- if \( \text{order} = \text{Nag\_ColMajor} \), \( \text{pdb} \geq \max(1, m) \);
- if \( \text{order} = \text{Nag\_RowMajor} \), \( \text{pdb} \geq \max(1, n) \).

13: \( \text{fail} \) – NagError *

*Input/Output*

The NAG error parameter (see the Essential Introduction).

### 6 Error Indicators and Warnings

**NE\_INT**

*On entry,* \( m = \langle \text{value} \rangle \).

Constraint: \( m \geq 0 \).

*On entry,* \( n = \langle \text{value} \rangle \).

Constraint: \( n \geq 0 \).

*On entry,* \( \text{pdt} = \langle \text{value} \rangle \).

Constraint: \( \text{pdt} \geq \max(1, n) \).

*On entry,* \( \text{pdb} = \langle \text{value} \rangle \).

Constraint: \( \text{pdb} \geq \max(1, m) \).

*On entry,* \( \text{pdb} = \langle \text{value} \rangle \).

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

**NE\_BAD\_PARAM**

*On entry,* parameter \( \langle \text{value} \rangle \) 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

No test for singularity or near-singularity of \( \mathbf{T} \) is included in this routine. Such tests must be performed before calling this routine.

### 9 Example

None.