NAG C Library Function Document

nag_ztrsv (f16sjc)

1 Purpose

nag_ztrsv (f16sjc) solves one of the systems of equations

\[ Tx = \alpha b \quad \text{or} \quad T^T x = \alpha b, \]

where \( T \) is a complex triangular matrix.

2 Specification

```c
void nag_ztrsv (Nag_OrderType order, Nag_UploType uplo, Nag_TransType trans,
                Nag_DiagType diag, Integer n, Complex alpha, const Complex t[],
                Integer pdt, Complex x[], Integer incx, NagError *fail)
```

3 Description

nag_ztrsv (f16sjc) performs one of the matrix-vector operations

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

where \( T \) is an \( n \) by \( n \) complex triangular matrix, \( x \) is an \( n \) element complex vector 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

\( \text{Input} \)

\( \text{On entry:} \) 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.

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

2: \( \text{uplo} \) – Nag_UploType

\( \text{Input} \)

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

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

\( \text{Constraint:} \ \text{uplo} = \text{Nag_Upper} \) or \( \text{Nag_Lower} \).

3: \( \text{trans} \) – Nag_TransType

\( \text{Input} \)

\( \text{On entry:} \) specifies the operation to be performed as follows:

- if \( \text{trans} = \text{Nag_NoTrans} \), \( x \leftarrow T^{-1} x \);
- if \( \text{trans} = \text{Nag_Trans} \), \( x \leftarrow T^{-T} x \);
- if \( \text{trans} = \text{Nag_ConjTrans} \), \( x \leftarrow T^{-H} x \).

\( \text{Constraint:} \ \text{trans} = \text{Nag_NoTrans} \), \( \text{Nag_Trans} \) or \( \text{Nag_ConjTrans} \).
4: \( \textbf{diag} \) – Nag_DiagType 

\textit{Input}

\textit{On entry:} specifies whether \( A \) has non-unit or unit diagonal elements, as follows:

- if \( \text{diag} = \text{Nag\_NonUnitDiag} \), the diagonal elements are stored explicitly;
- if \( \text{diag} = \text{Nag\_UnitDiag} \), the diagonal elements are assumed to be 1, and are not referenced.

\textit{Constraint:} \( \text{diag} = \text{Nag\_NonUnitDiag} \) or \( \text{Nag\_UnitDiag} \).

5: \( n \) – Integer 

\textit{Input}

\textit{On entry:} \( n \), the order of the matrix \( T \).

\textit{Constraint:} \( n \geq 0 \).

6: \( \alpha \) – Complex

\textit{Input}

\textit{On entry:} the scalar \( \alpha \).

7: \( t[dim] \) – const Complex

\textit{Input}

\textit{Note:} the dimension, \( dim \), of the array \( t \) must be at least \( \max(1, \text{pdt} \times n) \).

\textit{On entry:} the \( n \) by \( n \) triangular matrix \( T \). If \( \text{uplo} = \text{Nag\_Upper} \), \( T \) is upper triangular and the elements of the array below the diagonal are not referenced; if \( \text{uplo} = \text{Nag\_Lower} \), \( T \) is lower triangular and the elements of the array above the diagonal are not referenced. If \( \text{diag} = \text{Nag\_UnitDiag} \), the diagonal elements of \( T \) are not referenced, but are assumed to be 1.

8: \( \text{pdt} \) – Integer

\textit{Input}

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

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

9: \( x[dim] \) – Complex

\textit{Input/Output}

\textit{Note:} the dimension, \( dim \), of the array \( x \) must be at least \( 1 + (n - 1) \times \text{incx} \).

\textit{On entry:} the right hand side vector \( b \).

\textit{On exit:} the solution vector \( x \).

10: \( \text{incx} \) – Integer

\textit{Input}

\textit{On entry:} the increment in the subscripts of \( x \) between successive elements of \( x \).

\textit{Constraint:} \( \text{incx} \neq 0 \).

11: \( \text{fail} \) – NagError * 

\textit{Input/Output}

The NAG error parameter (see the Essential Introduction).

\section{6 Error Indicators and Warnings}

\textbf{NE\_INT}

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

\textit{Constraint:} \( n \geq 0 \).

On entry, \( \text{incx} = (\text{value}) \).

\textit{Constraint:} \( \text{incx} \neq 0 \).

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

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

No test for singularity or near-singularity of $T$ is included in this routine. Such tests must be performed before calling this routine.

9 Example

None.