1 Purpose

nag_ztrsna (f08qyc) estimates condition numbers for specified eigenvalues and/or right eigenvectors of a complex upper triangular matrix.

2 Specification

```c
#include <nag.h>
#include <nagf08.h>

void nag_ztrsna (Nag_OrderType order, Nag_JobType job,
                Nag_HowManyType how_many, const Nag_Boolean select[],
                Integer n, const Complex t[], Integer pdt,
                const Complex vl[], Integer pdvl,
                const Complex vr[], Integer pdvr,
                double s[], double sep[], Integer mm,
                Integer *m, NagError *fail)
```

3 Description

nag_ztrsna (f08qyc) estimates condition numbers for specified eigenvalues and/or right eigenvectors of a complex upper triangular matrix $T$. These are the same as the condition numbers of the eigenvalues and right eigenvectors of an original matrix $A = ZTZ^H$ (with unitary $Z$), from which $T$ may have been derived.

nag_ztrsna (f08qyc) computes the reciprocal of the condition number of an eigenvalue $\lambda_i$ as

$$s_i = \frac{|u^H v|}{\|u\|_E \|v\|_E},$$

where $u$ and $v$ are the right and left eigenvectors of $T$, respectively, corresponding to $\lambda_i$. This reciprocal condition number always lies between zero (i.e., ill-conditioned) and one (i.e., well-conditioned).

An approximate error estimate for a computed eigenvalue $\lambda_i$ is then given by

$$\frac{\epsilon \|T\|}{s_i},$$

where $\epsilon$ is the machine precision.

To estimate the reciprocal of the condition number of the right eigenvector corresponding to $\lambda_i$, the function first calls nag_ztrexc (f08qtc) to reorder the eigenvalues so that $\lambda_i$ is in the leading position:

$$T = Q \begin{pmatrix} \lambda_i & 0 \\ 0 & T_{22} \end{pmatrix} Q^H.$$

The reciprocal condition number of the eigenvector is then estimated as $sep_i$, the smallest singular value of the matrix $(T_{22} - \lambda_i I)$. This number ranges from zero (i.e., ill-conditioned) to very large (i.e., well-conditioned).

An approximate error estimate for a computed right eigenvector $u$ corresponding to $\lambda_i$ is then given by

$$\frac{\epsilon \|T\|}{sep_i}.$$

4 References

5 Arguments

1: order – Nag_OrderType

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: job – Nag_JobType

On entry: indicates whether condition numbers are required for eigenvalues and/or eigenvectors.

job = Nag_EigVals
Condition numbers for eigenvalues only are computed.

job = Nag_EigVecs
Condition numbers for eigenvectors only are computed.

job = Nag_DoBoth
Condition numbers for both eigenvalues and eigenvectors are computed.

Constraint: job = Nag_EigVals, Nag_EigVecs or Nag_DoBoth.

3: how_many – Nag_HowManyType

On entry: indicates how many condition numbers are to be computed.

how_many = Nag_ComputeAll
Condition numbers for all eigenpairs are computed.

how_many = Nag_ComputeSelected
Condition numbers for selected eigenpairs (as specified by select) are computed.

Constraint: how_many = Nag_ComputeAll or Nag_ComputeSelected.

4: select[dim] – const Nag_Boolean

Note: the dimension, dim, of the array select must be at least
n when how_many = Nag_ComputeSelected;
otherwise select may be NULL.

On entry: specifies the eigenpairs for which condition numbers are to be computed if
how_many = Nag_ComputeSelected. To select condition numbers for the eigenpair corresponding
to the eigenvalue \( \lambda_j \), select[\( j - 1 \)] must be set to Nag_TRUE.

If how_many = Nag_ComputeAll, select is not referenced and may be NULL.

5: n – Integer

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

Constraint: \( n \geq 0 \).

6: t[dim] – const Complex

Note: the dimension, dim, of the array t must be at least pdt \( \times n \).

The \((i, j)\)th element of the matrix \( T \) is stored in

\[ t[(j - 1) \times pdt + i - 1] \text{ when order = Nag_ColMajor}; \]
\[ t[(i - 1) \times pdt + j - 1] \text{ when order = Nag_RowMajor}. \]

On entry: the \( n \) by \( n \) upper triangular matrix \( T \), as returned by nag_zhseqr (f08psc).
7: \texttt{pdt} – Integer \hfill \textit{Input}

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

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

8: \texttt{vl}[\textit{dim}] – const Complex \hfill \textit{Input}

\textit{Note:} the dimension, \textit{dim}, of the array \texttt{vl} must be at least

- \texttt{pdvl} \times \texttt{mm} when \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth} and \texttt{order} = \texttt{Nag\_ColMajor};
- \texttt{n} \times \texttt{pdvl} when \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth} and \texttt{order} = \texttt{Nag\_RowMajor};
- otherwise \texttt{vl} may be \texttt{NULL}.

The \((i, j)\)th element of the matrix is stored in

- \texttt{vl}[(j - 1) \times \texttt{pdvl} + i - 1] when \texttt{order} = \texttt{Nag\_ColMajor};
- \texttt{vl}[(i - 1) \times \texttt{pdvl} + j - 1] when \texttt{order} = \texttt{Nag\_RowMajor}.

\textit{On entry:} if \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth}, \texttt{vl} must contain the left eigenvectors of \(T\) (or of any matrix \(QTQ^H\) with \(Q\) unitary) corresponding to the eigenpairs specified by \texttt{how\_many} and \texttt{select}. The eigenvectors must be stored in consecutive rows or columns (depending on the value of \texttt{order}) of \texttt{vl}, as returned by \texttt{nag\_zhsein (f08pxc)} or \texttt{nag\_ztrevc (f08qxc)}.

If \texttt{job} = \texttt{Nag\_EigVecs}, \texttt{vl} is not referenced and may be \texttt{NULL}.

9: \texttt{pdvl} – Integer \hfill \textit{Input}

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

\textit{Constraints:}

- if \texttt{order} = \texttt{Nag\_ColMajor},
  - if \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth}, \texttt{pdvl} \geq \texttt{n};
  - if \texttt{job} = \texttt{Nag\_EigVecs}, \texttt{vl} may be \texttt{NULL};
- if \texttt{order} = \texttt{Nag\_RowMajor},
  - if \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth}, \texttt{pdvl} \geq \texttt{mm};
  - if \texttt{job} = \texttt{Nag\_EigVecs}, \texttt{vl} may be \texttt{NULL}.

The \((i, j)\)th element of the matrix is stored in

- \texttt{vl}[(j - 1) \times \texttt{pdvl} + i - 1] when \texttt{order} = \texttt{Nag\_ColMajor};
- \texttt{vl}[(i - 1) \times \texttt{pdvl} + j - 1] when \texttt{order} = \texttt{Nag\_RowMajor}.

\textit{On entry:} if \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth}, \texttt{vr} must contain the right eigenvectors of \(T\) (or of any matrix \(QTQ^H\) with \(Q\) unitary) corresponding to the eigenpairs specified by \texttt{how\_many} and \texttt{select}. The eigenvectors must be stored in consecutive rows or columns (depending on the value of \texttt{order}) of \texttt{vr}, as returned by \texttt{nag\_zhsein (f08pxc)} or \texttt{nag\_ztrevc (f08qxc)}.

If \texttt{job} = \texttt{Nag\_EigVecs}, \texttt{vr} is not referenced and may be \texttt{NULL}.

10: \texttt{vr}[\textit{dim}] – const Complex \hfill \textit{Input}

\textit{Note:} the dimension, \textit{dim}, of the array \texttt{vr} must be at least

- \texttt{pdvr} \times \texttt{mm} when \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth} and \texttt{order} = \texttt{Nag\_ColMajor};
- \texttt{n} \times \texttt{pdvr} when \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth} and \texttt{order} = \texttt{Nag\_RowMajor};
- otherwise \texttt{vr} may be \texttt{NULL}.

The \((i, j)\)th element of the matrix is stored in

- \texttt{vr}[(j - 1) \times \texttt{pdvr} + i - 1] when \texttt{order} = \texttt{Nag\_ColMajor};
- \texttt{vr}[(i - 1) \times \texttt{pdvr} + j - 1] when \texttt{order} = \texttt{Nag\_RowMajor}.

\textit{On entry:} if \texttt{job} = \texttt{Nag\_EigVals} or \texttt{Nag\_DoBoth}, \texttt{vr} must contain the right eigenvectors of \(T\) (or of any matrix \(QTQ^H\) with \(Q\) unitary) corresponding to the eigenpairs specified by \texttt{how\_many} and \texttt{select}. The eigenvectors must be stored in consecutive rows or columns (depending on the value of \texttt{order}) of \texttt{vr}, as returned by \texttt{nag\_zhsein (f08pxc)} or \texttt{nag\_ztrevc (f08qxc)}.

If \texttt{job} = \texttt{Nag\_EigVecs}, \texttt{vr} is not referenced and may be \texttt{NULL}.

11: \texttt{pdvr} – Integer \hfill \textit{Input}

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

Mark 25
Constraints:

if order = Nag_ColMajor,
    if job = Nag_EigVals or Nag_DoBoth, pdv \geq n;
    if job = Nag_EigVecs, vr may be NULL;
if order = Nag_RowMajor,
    if job = Nag_EigVals or Nag_DoBoth, pdv / C21
    mm;
    if job = Nag_EigVecs, vr may be NULL.

12: \( \text{s}[\text{dim}] \) – double

Output

Note: the dimension, \( \text{dim} \), of the array \( \text{s} \) must be at least

\( \text{mm} \) when \( \text{job} = \text{Nag} \_\text{EigVals} \) or \( \text{Nag} \_\text{DoBoth} \);
otherwise \( \text{s} \) may be NULL.

On exit: the reciprocal condition numbers of the selected eigenvalues if \( \text{job} = \text{Nag} \_\text{EigVals} \) or \( \text{Nag} \_\text{DoBoth} \), stored in consecutive elements of the array. Thus \( s[j - 1], \text{sep}[j - 1] \) and the \( j \)th rows or columns of \( \text{vl} \) and \( \text{vr} \) all correspond to the same eigenpair (but not in general the \( j \)th eigenpair unless all eigenpairs have been selected).

If \( \text{job} = \text{Nag} \_\text{EigVecs} \), \( \text{s} \) is not referenced and may be NULL.

13: \( \text{sep}[\text{dim}] \) – double

Output

Note: the dimension, \( \text{dim} \), of the array \( \text{sep} \) must be at least

\( \text{mm} \) when \( \text{job} = \text{Nag} \_\text{EigVecs} \) or \( \text{Nag} \_\text{DoBoth} \);
otherwise \( \text{sep} \) may be NULL.

On exit: the estimated reciprocal condition numbers of the selected right eigenvectors if \( \text{job} = \text{Nag} \_\text{EigVecs} \) or \( \text{Nag} \_\text{DoBoth} \), stored in consecutive elements of the array.

If \( \text{job} = \text{Nag} \_\text{EigVecs} \), \( \text{sep} \) is not referenced and may be NULL.

14: \( \text{mm} \) – Integer

Input

On entry: the number of elements in the arrays \( \text{s} \) and \( \text{sep} \), and the number of rows or columns (depending on the value of order) in the arrays \( \text{vl} \) and \( \text{vr} \) (if used). The precise number required, \( \text{required} \_\text{owcol} \), is \( n \) if \( \text{how} \_\text{many} = \text{Nag} \_\text{ComputeAll} \); if \( \text{how} \_\text{many} = \text{Nag} \_\text{ComputeSelected} \), \( \text{required} \_\text{owcol} \) is the number of selected eigenpairs (see \( \text{select} \)), in which case \( 0 \leq \text{required} \_\text{owcol} \leq n \).

Constraints:

if \( \text{how} \_\text{many} = \text{Nag} \_\text{ComputeAll} \), \( \text{mm} \geq n \);
otherwise \( \text{mm} \geq \text{required} \_\text{owcol} \).

15: \( \text{m} \) – Integer *

Output

On exit: \( \text{required} \_\text{owcol} \), the number of selected eigenpairs. If \( \text{how} \_\text{many} = \text{Nag} \_\text{ComputeAll} \), \( \text{m} \) is set to \( n \).

16: \( \text{fail} \) – 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.
See Section 3.2.1.2 in the Essential Introduction for further information.
NE_BAD_PARAM
On entry, argument \textit{value} had an illegal value.

NE_ENUM_INT_2
On entry, \textit{how many} = \textit{value}, \textit{mm} = \textit{value} and \textit{n} = \textit{value}.
Constraint: if \textit{how many} = Nag_ComputeAll, \textit{mm} \geq \textit{n};
otherwise \textit{mm} \geq \text{required, oucol}.

On entry, \textit{job} = \textit{value}, \textit{pdvl} = \textit{value}, \textit{mm} = \textit{value}.
Constraint: if \textit{job} = Nag_EigVals or Nag_DoBoth, \textit{pdvl} \geq \textit{mm}.

On entry, \textit{job} = \textit{value}, \textit{pdvl} = \textit{value}, \textit{n} = \textit{value}.
Constraint: if \textit{job} = Nag_EigVals or Nag_DoBoth, \textit{pdvl} \geq \textit{n}.

On entry, \textit{job} = \textit{value}, \textit{pdvr} = \textit{value}, \textit{mm} = \textit{value}.
Constraint: if \textit{job} = Nag_EigVals or Nag_DoBoth, \textit{pdvr} \geq \textit{mm}.

On entry, \textit{job} = \textit{value}, \textit{pdvr} = \textit{value}, \textit{n} = \textit{value}.
Constraint: if \textit{job} = Nag_EigVals or Nag_DoBoth, \textit{pdvr} \geq \textit{n}.

NE_INT
On entry, \textit{n} = \textit{value}.
Constraint: \textit{n} \geq 0.

On entry, \textit{pdt} = \textit{value}.
Constraint: \textit{pdt} > 0.

On entry, \textit{pdvl} = \textit{value}.
Constraint: \textit{pdvl} > 0.

On entry, \textit{pdvr} = \textit{value}.
Constraint: \textit{pdvr} > 0.

NE_INT_2
On entry, \textit{pdt} = \textit{value} and \textit{n} = \textit{value}.
Constraint: \textit{pdt} \geq \text{max}(1, \textit{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.

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

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 computed values \textit{sep}, may over estimate the true value, but seldom by a factor of more than 3.

8 Parallelism and Performance
\texttt{nag_ztrsna (f08qyc)} is not threaded by NAG in any implementation.

\texttt{nag_ztrsna (f08qyc)} makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.
Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.

9 Further Comments

The real analogue of this function is nag_dtrsna (f08qlc).

10 Example

This example computes approximate error estimates for all the eigenvalues and right eigenvectors of the matrix $T$, where

$$
T = \begin{pmatrix}
-6.0004 - 6.9999i & 0.3637 - 0.3656i & -0.1880 + 0.4787i & 0.8785 - 0.2539i \\
0.0000 + 0.0000i & -5.0000 + 2.0060i & -0.0307 - 0.7217i & -0.2290 + 0.1313i \\
0.0000 + 0.0000i & 0.0000 + 0.0000i & 7.9982 - 0.9964i & 0.9357 + 0.5359i \\
0.0000 + 0.0000i & 0.0000 + 0.0000i & 0.0000 + 0.0000i & 3.0023 - 3.9998i
\end{pmatrix}
$$

10.1 Program Text

```c
/* nag_ztrsna (f08qyc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 7, 2001. */
*/
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <naga02.h>
#include <nagf08.h>
#include <nagf16.h>
#include <nagx02.h>

int main(void)
{
    /* Scalars */
    Integer i, j, m, n, pdt, pdvl, pdvr;
    Integer s_len;
    Integer exit_status = 0;
    double eps, tnorm;
    NagError fail;
    Nag_OrderType order;
    /* Arrays */
    double *s = 0, *sep = 0;
    Complex *t = 0, *vl = 0, *vr = 0;

    #ifdef NAG_COLUMN_MAJOR
    define T(I, J) t[(J-1)*pdt +I-1]
    order = Nag_ColMajor;
    #else
    define T(I, J) t[(I-1)*pdt +J-1]
    order = Nag_RowMajor;
    #endif

    INIT_FAIL(fail);

    printf("nag_ztrsna (f08qyc) Example Program Results\n");

    /* Skip heading in data file */
    #ifdef _WIN32
    scanf_s("%*[\n ");
    #else
    scanf("%*[\n ");
    #endif
    #ifdef _WIN32
```
scanf_s("%"NAG_IFMT"%*[\n ]", &n);
#else
    scanf("%"NAG_IFMT"%*[\n ]", &n);
#endif
#else
    scanf("%"NAG_IFMT"%*[\n ]", &n);
#endif
    ifdef NAG_COLUMN_MAJOR
    pdt = n;
    pdvl = n;
    pdvr = n;
#else
    pdt = n;
    pdvl = n;
    pdvr = n;
#endif
    s_len = n;

    /* Allocate memory */
    if (!(t = NAG_ALLOC(n * n, Complex)) ||
        !(vl = NAG_ALLOC(n * n, Complex)) ||
        !(vr = NAG_ALLOC(n * n, Complex)) ||
        !(s = NAG_ALLOC(s_len, double)) ||
        !(sep = NAG_ALLOC(s_len, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

    /* Read T from data file */
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= n; ++j)
        {
            #ifdef _WIN32
                scanf_s(" ( %lf , %lf ) ", &T(i, j).re, &T(i, j).im);
            #else
                scanf(" ( %lf , %lf ) ", &T(i, j).re, &T(i, j).im);
            #endif
        }
    }
    #ifdef _WIN32
    scanf_s("%*[\n ]");
    #else
    scanf("%*[\n ]");
    #endif

    /* Calculate right and left eigenvectors of T */
    /* nag_ztrevc (f08qxc).
       * Left and right eigenvectors of complex upper triangular
       * matrix
       */
    nag_ztrevc(order, Nag_BothSides, Nag_ComputeAll, NULL, n, t, pdt,
               vl, pdvl, vr, pdvr, n, &m, &fail);
    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_ztrevc (f08qxc).\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }

    /* Estimate condition numbers for all the eigenvalues and */
    /* right eigenvectors of T */
    /* nag_ztrsna (f08qyc).
       * Estimates of sensitivities of selected eigenvalues and
       * eigenvectors of complex upper triangular matrix
       */
    nag_ztrsna(order, Nag_DoBoth, Nag_ComputeAll, NULL, n, t, pdt,
               vl, pdvl, vr, pdvr, s, sep, n, &m, &fail);
    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_ztrsna (f08qyc).\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }
/* Print condition numbers of eigenvalues and right eigenvectors */
printf("\nS\n");
for (i = 0; i < n; ++i)
    printf("%11.1e", s[i]);
printf("\n\nSep\n");
for (i = 0; i < n; ++i)
    printf("%11.1e", sep[i]);
printf("\n");
/* Calculate approximate error estimates (using the 1-norm) */
/* nag_zge_norm (f16uac). */
/* 1-norm, infinity-norm, Frobenius norm, largest absolute */
/* element, complex general matrix */
/* nag_zge_norm(order, Nag_OneNorm, n, n, t, pdt, &tnorm, &fail); */
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_zge_norm (f16uac). \n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
/* nag_machine_precision (x02ajc). */
/* The machine precision */
eps = nag_machine_precision;
printf("\nApproximate error estimates for eigenvalues" "of T (machine dependent)\n");
for (i = 0; i < m; ++i)
    printf("%11.1e", eps*tnorm/s[i]);
printf("\n\nApproximate error estimates for right eigenvectors" "of T (machine dependent)\n");
for (i = 0; i < m; ++i)
    printf("%11.1e", eps*tnorm/sep[i]);
printf("\n");
END:
NAG_FREE(t);
NAG_FREE(s);
NAG_FREE(sep);
NAG_FREE(vl);
NAG_FREE(vr);
return exit_status;
}

10.2 Program Data
nag_ztrsna (f08qyc) Example Program Data
4 :Value of N
(-6.0004,-6.9999) ( 0.3637,-0.3656) (-0.1880, 0.4787) ( 0.8785,-0.2539)
( 0.0000, 0.0000) (-5.0000, 2.0060) (-0.0307,-0.7217) (-0.2290, 0.1313)
( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 7.9982,-0.9964) ( 0.9357, 0.5359)
( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 3.0023,-3.9998)
:End of matrix T

10.3 Program Results
nag_ztrsna (f08qyc) Example Program Results
S
  9.9e-01  1.0e+00  9.8e-01  9.8e-01
Sep
  8.4e+00  8.0e+00  5.8e+00  5.8e+00
Approximate error estimates for eigenvalues of $T$ (machine dependent)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0e-15</td>
<td>1.0e-15</td>
<td>1.1e-15</td>
<td>1.1e-15</td>
</tr>
</tbody>
</table>

Approximate error estimates for right eigenvectors of $T$ (machine dependent)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2e-16</td>
<td>1.3e-16</td>
<td>1.8e-16</td>
<td>1.8e-16</td>
</tr>
</tbody>
</table>