nag_trans_hessenberg_controller (g13exc)

1. Purpose

nag_trans_hessenberg_controller (g13exc) reduces the matrix pair (B,A) to lower or upper controller Hessenberg form using (and optionally accumulating) the unitary state-space transformations.

2. Specification

```
#include <nag.h>
#include <nagg13.h>

void g13exc(Integer n, Integer m, Nag_ControllerForm reduceto, double a[],
            Integer tda, double b[], Integer tdb, double u[], Integer tdu,
            NagError *fail)
```

3. Description

The function computes a unitary state-space transformation U, which reduces the matrix pair (B,A) to give a compound matrix in one of the following controller Hessenberg forms:

$$\left(UB|UAU^T\right) = \overbrace{\left(\begin{array}{ccccc|cccccccc}
* & . & . & . & * & * & . & . & . & . & . & . & * \\
 & . & & & . & . & & & & & & & . \\
 & & . & & . & . & & & & & & & . \\
 & & & . & . & . & & & & & & & . \\
 & & & & * & . & & & & & & & . \\
 & & & & & * & & & & & & & . \\
 & & & & & & . & & & & & & . \\
 & & & & & & & . & & & & & . \\
 & & & & & & & & . & & & & . \\
 & & & & & & & & & * & . & . & *
\end{array}\right)}^{m \qquad\qquad n} n$$

if **reduceto** = **Nag_UH_Controller**, or

$$\left(UAU^T|UB\right) = \overbrace{\left(\begin{array}{cccccccc|ccccc}
* & . & . & * & & & & & & & & & \\
. & & & & . & & & & & & & & \\
. & & & & & . & & & & & & & \\
. & & & & & & . & & & & & & \\
. & & & & & & & * & & & & & \\
. & & & & & & & . & * & & & & \\
. & & & & & & & . & . & . & & & \\
. & & & & & & & . & . & & . & & \\
. & & & & & & & . & . & & & . & \\
* & . & . & . & . & . & . & * & * & . & . & . & *
\end{array}\right)}^{n \qquad\qquad m} n$$

if **reduceto** = **Nag_LH_Controller**.

If $m > n$, then the matrix UB is trapezoidal and if $m + 1 \geq n$ then the matrix UAU^T is full.

4. Parameters

n

Input: The actual state dimension, n, i.e., the order of the matrix A.

Constraint: $\mathbf{n} \geq 1$.

m

Input: The actual input dimension, m.

Constraint: $\mathbf{m} \geq 1$.

reduceto

Input: Indicates whether the matrix pair (B,A) is to be reduced to upper or lower controller Hessenberg form as follows:
reduceto = **Nag_UH_Controller**, (Upper controller Hessenberg form);
reduceto = **Nag_LH_Controller**, (Lower controller Hessenberg form).

a[n][tda]

Input: The leading n by n part of this array must contain the state transition matrix A to be transformed.

Output: The leading n by n part of this array contains the transformed state transition matrix UAU^T.

tda

Input: The trailing dimension of array **a** as declared in the calling program.

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

b[n][tdb]

Input: The leading n by m part of this array must contain the input matrix B to be transformed.

Output: The leading n by m part of this array contains the transformed input matrix UB.

tdb

Input: The trailing dimension of array **b** as declared in the calling program.

Constraint: $\mathbf{tdb} \geq \mathbf{m}$.

u[n][tdu]

Input: If **u** is defined, then the leading n by n part of this array must contain either a transformation matrix (e.g. from a previous call to this function) or be initialised as the identity matrix. If this information is not to be input then **u** must be set to the null pointer, i.e., (double $*$)0.

Output: If **u** is defined, then the leading n by n part of this array contains the product of the input matrix U and the state-space transformation matrix which reduces the given pair to observer Hessenberg form.

tdu

Input: The trailing dimension of array **u** as declared in the calling program.

Constraint: $\mathbf{tdu} \geq \mathbf{n}$ if **u** is defined.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_BAD_PARAM

On entry parameter **reduceto** had an illegal value.

NE_INT_ARG_LT

On entry, **n** must not be less than 1: **n** = ⟨*value*⟩.
On entry, **m** must not be less than 1: **m** = ⟨*value*⟩.

NE_2_INT_ARG_LT

On entry **tda** = ⟨*value*⟩ while **n** = ⟨*value*⟩.
These parameters must satisfy **tda** ≥ **n**.

On entry **tdb** = ⟨*value*⟩ while **m** = ⟨*value*⟩.
These parameters must satisfy **tdb** ≥ **m**.

On entry **tdu** = ⟨*value*⟩ while **n** = ⟨*value*⟩.
These parameters must satisfy **tdu** ≥ **n**.

6. Further Comments

The algorithm requires $O((n+m)n^2)$ operations (see Van Dooren and Verhaegen 1985).

6.1. Accuracy

The algorithm is backward stable.

6.2. References

Van Dooren P and Verhaegen M (1985) On the use of unitary state-space transformations. *In: Contemporary Mathematics on Linear Algebra and its Role in Systems Theory* **47** AMS, Providence.

7. See Also

nag_kalman_sqrt_filt_info_invar (g13edc)

8. Example

To reduce the matrix pair (B,A) to upper controller Hessenberg form, and return the unitary state-space transformation matrix U.

8.1. Program Text

```
/* nag_trans_hessenberg_controller(g13exc)  Example Program
 *
 * Copyright 1994 Numerical Algorithms Group
 *
 * Mark 3, 1994.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg13.h>

#define NMAX 20
#define MMAX 20
```

```
#define TDA NMAX
#define TDB MMAX
#define TDU NMAX

main()

{
  double a[NMAX][TDA];
  double b[NMAX][TDB];
  double u[NMAX][TDU];
  Integer i, j, m, n;
  Nag_ControllerForm reduceto;
  double zero = 0.0, one = 1.0;
  Integer nmax, mmax;

  Vprintf("g13exc Example Program Results\n");

  /* Skip the heading in the data file and read the data. */
  Vscanf("%*[^\n]");

  nmax = NMAX;
  mmax = MMAX;

  Vscanf("%ld%ld",&n,&m);
  if (n<=0 || m<=0 ||
      n>nmax || m>mmax)
    {
      Vfprintf(stderr, "One of n or m is out of range \
n = %ld, m = %ld\n", n, m);
      exit(EXIT_FAILURE);
    }

  reduceto = Nag_UH_Controller;

  for (j=0; j<n; ++j)
    for (i=0; i<n; ++i)
      Vscanf("%lf",&a[i][j]);
  for (j=0; j<m; ++j)
    for (i=0; i<n; ++i)
      Vscanf("%lf",&b[i][j]);

  if (u) /* Initialise U as the identity matrix. */
    for (i=0; i<n; ++i)
      {
        for (j=0; j<n; ++j)
          u[i][j] = zero;
        u[i][i] = one;
      }

  /* Reduce the pair (B,A) to reduceto controller Hessenberg form. */
  g13exc(n, m,  reduceto, (double *)a, (Integer)TDA, (double *)b, (Integer)TDB,
         (double *)u, (Integer)TDU, NAGERR_DEFAULT);

  Vprintf("\nThe transformed state transition matrix is\n\n");
  for (i=0; i<n; ++i)
    {
      for (j=0; j<n; ++j)
        Vprintf ("%8.4f ",a[i][j]);
      Vprintf("\n");
    }

  Vprintf("\nThe transformed input matrix is\n\n");
  for (i=0; i<n; ++i)
    {
      for (j=0; j<m; ++j)
        Vprintf("%8.4f ", b[i][j]);
      Vprintf("\n");
    }
  if (u)
```

```
    {
      Vprintf("\nThe matrix that reduces (B,A) to ");
      Vprintf("controller Hessenberg form is\n\n");
      for (i=0; i<n; ++i)
        {
          for (j=0; j<n; ++j)
            Vprintf("%8.4f ", u[i][j]);
          Vprintf("\n");
        }
    }
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

```
g13exc Example Program Data
  6     3
  35.0   1.0   6.0  26.0  19.0  24.0
   3.0  32.0   7.0  21.0  23.0  25.0
  31.0   9.0   2.0  22.0  27.0  20.0
   8.0  28.0  33.0  17.0  10.0  15.0
  30.0   5.0  34.0  12.0  14.0  16.0
   4.0  36.0  29.0  13.0  18.0  11.0
   1.0   5.0  11.0
  -1.0   4.0  11.0
  -5.0   1.0   9.0
 -11.0  -4.0   5.0
 -19.0 -11.0  -1.0
 -29.0 -20.0  -9.0
```

8.3. Program Results

```
g13exc Example Program Results

The transformed state transition matrix is

 60.3649  58.8853   5.0480  -5.4406   2.1382  -7.3870
 54.5832  33.1865  36.5234   6.3272  -3.1377   8.8154
 17.6406  21.4501 -13.5942   0.5417   1.6926   0.0786
 -9.0567  10.7202   0.3531   1.5444  -1.2846  24.6407
  0.0000   6.8796 -20.1372  -2.6440   2.4983 -21.8071
  0.0000   0.0000   0.0000   0.0000   0.0000  27.0000

The transformed input matrix is

-16.8819  -8.8260  13.9202
  0.0000  13.8240  39.9205
  0.0000   0.0000   4.1928
  0.0000   0.0000   0.0000
  0.0000   0.0000   0.0000
  0.0000   0.0000   0.0000

The matrix that reduces (B,A) to controller Hessenberg form is

 -0.0592  -0.2962  -0.6516   0.0592  -0.2369  -0.6516
 -0.3995  -0.1168   0.2350  -0.7579  -0.4406  -0.0543
 -0.5311  -0.5286  -0.3131   0.1029   0.2119   0.5339
 -0.2594   0.5309  -0.3641  -0.3950   0.5927  -0.1051
  0.6357  -0.0637  -0.4542  -0.4149  -0.1423   0.4394
 -0.2887   0.5774  -0.2887   0.2887  -0.5774   0.2887
```