NAG Library Function Document

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

```c
#include <nag.h>
#include <nagg13.h>
void nag_trans_hessenberg_controller (Integer n, Integer m,
   Nag_ControllerForm reduceto, double a[], Integer tda, double b[],
   Integer tdb, double u[], Integer tdu, NagError *fail)
```

3 Description

nag_trans_hessenberg_controller (g13exc) 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:

\[
(UB | UA^T) = \begin{pmatrix}
* & \cdots & * & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & * \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
* & \cdots & * & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & * \\
\end{pmatrix}
\]

If \(\text{reduceto} = \text{Nag\_UH\_Controller}\), or

\[
(UA^T | UB) = \begin{pmatrix}
* & \cdots & * \\
\vdots & \ddots & \ddots \\
\vdots & \ddots & \ddots \\
\vdots & \ddots & \ddots \\
\vdots & \ddots & \ddots \\
\vdots & \ddots & \ddots \\
* & \cdots & * & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & * \\
\end{pmatrix}
\]

If \(\text{reduceto} = \text{Nag\_LH\_Controller}\). If \(m > n\), then the matrix \(UB\) is trapezoidal and if \(m + 1 \geq n\) then the matrix \(UA^T\) is full.
4 References

5 Arguments
1: \( n \) – Integer \(^{1}\)\(\text{Input} \)
   On entry: the actual state dimension, \( n \), i.e., the order of the matrix \( A \).
   Constraint: \( n \geq 1 \).

2: \( m \) – Integer \(^{1}\)\(\text{Input} \)
   On entry: the actual input dimension, \( m \).
   Constraint: \( m \geq 1 \).

3: reduceto – Nag_ControllerForm \(^{1}\)\(\text{Input} \)
   On entry: 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).
   Constraint: reduceto = Nag_UH_Controller or Nag_LH_Controller.

4: \( a[n \times tda] \) – double \(^{1}\)\(\text{Input/Output} \)
   Note: the \((i, j)\)th element of the matrix \( A \) is stored in \( a[(i - 1) \times tda + j - 1] \).
   On entry: the leading \( n \) by \( n \) part of this array must contain the state transition matrix \( A \) to be transformed.
   On exit: the leading \( n \) by \( n \) part of this array contains the transformed state transition matrix \( UAU^T \).

5: \( tda \) – Integer \(^{1}\)\(\text{Input} \)
   On entry: the stride separating matrix column elements in the array \( a \).
   Constraint: \( tda \geq n \).

6: \( b[n \times tdb] \) – double \(^{1}\)\(\text{Input/Output} \)
   Note: the \((i, j)\)th element of the matrix \( B \) is stored in \( b[(i - 1) \times tdb + j - 1] \).
   On entry: the leading \( n \) by \( m \) part of this array must contain the input matrix \( B \) to be transformed.
   On exit: the leading \( n \) by \( m \) part of this array contains the transformed input matrix \( UB \).

7: \( tdb \) – Integer \(^{1}\)\(\text{Input} \)
   On entry: the stride separating matrix column elements in the array \( b \).
   Constraint: \( tdb \geq m \).

8: \( u[n \times tdu] \) – double \(^{1}\)\(\text{Input/Output} \)
   Note: the \((i, j)\)th element of the matrix \( U \) is stored in \( u[(i - 1) \times tdu + j - 1] \).
On entry: if $u$ is not NULL, 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 initialized as the identity matrix. If this information is not to be input then $u$ must be set to NULL.

On exit: if $u$ is not NULL, 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.

9: $\text{tdu} – \text{Integer}$

On entry: the stride separating matrix column elements in the array $u$.

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

10: $\text{fail} – \text{NagError *}$

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_2_INT_ARG_LT

On entry, $\text{tda} = \langle \text{value} \rangle$ while $n = \langle \text{value} \rangle$. These arguments must satisfy $\text{tda} \geq n$. On entry $\text{tdb} = \langle \text{value} \rangle$ while $m = \langle \text{value} \rangle$. These arguments must satisfy $\text{tdb} \geq m$. On entry $\text{tdu} = \langle \text{value} \rangle$ while $n = \langle \text{value} \rangle$. These arguments must satisfy $\text{tdu} \geq n$.

NE_BAD_PARAM

On entry, argument reduceto had an illegal value.

NE_INT_ARG_LT

On entry, $m = \langle \text{value} \rangle$.

Constraint: $m \geq 1$.

On entry, $n = \langle \text{value} \rangle$.

Constraint: $n \geq 1$.

7 Accuracy

The algorithm is backward stable.

8 Parallelism and Performance

Not applicable.

9 Further Comments

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

10 Example

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

Mark 25
10.1 Program Text

/* nag_trans_hessenberg_controller (g13exc) Example Program.  *
 * Copyright 2014 Numerical Algorithms Group  *
 * Mark 3, 1993  
 * Mark 8 revised, 2004.  *
*/

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg13.h>
#define A(I, J) a[(I) *tda + J]
#define B(I, J) b[(I) *tdb + J]
#define U(I, J) u[(I) *tdu + J]

int main(void)
{
    Integer exit_status = 0, i, j, m, n, tda, tdb, tdu;
    double *a = 0, *b = 0, one = 1.0, *u = 0, zero = 0.0;
    Nag_ControllerForm reduceto;
    NagError fail;

    INIT_FAIL(fail);

    printf(  
        "nag_trans_hessenberg_controller (g13exc) Example Program Results\n"
    );

    /* Skip the heading in the data file and read the data. */
    #ifdef _WIN32
    scanf_s("%*[\n]");
    #else
    scanf("%*[\n]");
    #endif
    #ifdef _WIN32
    scanf("%"NAG_IFMT"%"NAG_IFMT", &n, &m);
    #else
    scanf("%"NAG_IFMT"%"NAG_IFMT", &n, &m);
    #endif
    if (n >= 1 || m >= 1)
    {
        if (!(a = NAG_ALLOC(n*n, double)) ||  
            !(b = NAG_ALLOC(n*m, double)) ||  
            !(u = NAG_ALLOC(n*n, double)))
        {
            printf("Allocation failure\n");
            exit_status = -1;
            goto END;
        }
        tda = n;
        tdb = m;
        tdu = n;
    }
    else
    {
        printf("Invalid n or m.\n");
        exit_status = 1;
        return exit_status;
    }
    reduceto = Nag_UH_Controller;

    for (j = 0; j < n; ++j)
        for (i = 0; i < n; ++i)
            #ifdef _WIN32
            scanf_s("%lf", &A(i, j));
            #else
            scanf("%lf", &A(i, j));
            #endif
for (j = 0; j < m; ++j)
for (i = 0; i < n; ++i)
#ifdef _WIN32
    scanf_s("%lf", &B(i, j));
#else
    scanf("%lf", &B(i, j));
#endif
#endif

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. */
#ifndef _WIN32
    nag_trans_hessenberg_controller(n, m, reduceto, a, tda, b, tdb, u, tdu,
#else
    nag_trans_hessenberg_controller(n, m, reduceto, a, tda, b, tdb, u, tdu,
#endif
    &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_trans_hessenberg_controller (g13exc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

printf("The transformed state transition matrix is\n\n");
for (i = 0; i < n; ++i)
{
    for (j = 0; j < n; ++j)
        printf("%8.4f ", A(i, j));
    printf("\n");
}

printf("The transformed input matrix is\n\n");
for (i = 0; i < n; ++i)
{
    for (j = 0; j < m; ++j)
        printf("%8.4f ", B(i, j));
    printf("\n");
}
if (u)
{
    printf("The matrix that reduces (B,A) to ");
    printf("controller Hessenberg form is\n\n");
    for (i = 0; i < n; ++i)
    {
        for (j = 0; j < n; ++j)
            printf("%8.4f ", U(i, j));
        printf("\n");
    }
}
END:
NAG_FREE(a);
NAG_FREE(b);
NAG_FREE(u);
return exit_status;
10.2 Program Data

nag_trans_hessenberg_controller (g13exc) Example Program Data

<table>
<thead>
<tr>
<th>6</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>35.0</td>
<td>1.0</td>
</tr>
<tr>
<td>6.0</td>
<td>26.0</td>
</tr>
<tr>
<td>19.0</td>
<td>24.0</td>
</tr>
<tr>
<td>3.0</td>
<td>32.0</td>
</tr>
<tr>
<td>7.0</td>
<td>21.0</td>
</tr>
<tr>
<td>23.0</td>
<td>25.0</td>
</tr>
<tr>
<td>31.0</td>
<td>9.0</td>
</tr>
<tr>
<td>2.0</td>
<td>22.0</td>
</tr>
<tr>
<td>17.0</td>
<td>10.0</td>
</tr>
<tr>
<td>15.0</td>
<td>30.0</td>
</tr>
<tr>
<td>5.0</td>
<td>34.0</td>
</tr>
<tr>
<td>12.0</td>
<td>14.0</td>
</tr>
<tr>
<td>16.0</td>
<td>4.0</td>
</tr>
<tr>
<td>36.0</td>
<td>29.0</td>
</tr>
<tr>
<td>13.0</td>
<td>18.0</td>
</tr>
<tr>
<td>11.0</td>
<td>8.0</td>
</tr>
<tr>
<td>28.0</td>
<td>33.0</td>
</tr>
<tr>
<td>17.0</td>
<td>10.0</td>
</tr>
<tr>
<td>5.0</td>
<td>30.0</td>
</tr>
<tr>
<td>12.0</td>
<td>14.0</td>
</tr>
<tr>
<td>16.0</td>
<td>4.0</td>
</tr>
<tr>
<td>36.0</td>
<td>29.0</td>
</tr>
<tr>
<td>13.0</td>
<td>18.0</td>
</tr>
<tr>
<td>11.0</td>
<td>8.0</td>
</tr>
<tr>
<td>28.0</td>
<td>33.0</td>
</tr>
<tr>
<td>17.0</td>
<td>10.0</td>
</tr>
<tr>
<td>5.0</td>
<td>30.0</td>
</tr>
<tr>
<td>12.0</td>
<td>14.0</td>
</tr>
<tr>
<td>16.0</td>
<td>4.0</td>
</tr>
<tr>
<td>36.0</td>
<td>29.0</td>
</tr>
<tr>
<td>13.0</td>
<td>18.0</td>
</tr>
<tr>
<td>11.0</td>
<td>8.0</td>
</tr>
<tr>
<td>28.0</td>
<td>33.0</td>
</tr>
<tr>
<td>17.0</td>
<td>10.0</td>
</tr>
<tr>
<td>5.0</td>
<td>30.0</td>
</tr>
<tr>
<td>12.0</td>
<td>14.0</td>
</tr>
<tr>
<td>16.0</td>
<td>4.0</td>
</tr>
<tr>
<td>36.0</td>
<td>29.0</td>
</tr>
<tr>
<td>13.0</td>
<td>18.0</td>
</tr>
<tr>
<td>11.0</td>
<td>8.0</td>
</tr>
<tr>
<td>28.0</td>
<td>33.0</td>
</tr>
<tr>
<td>17.0</td>
<td>10.0</td>
</tr>
<tr>
<td>5.0</td>
<td>30.0</td>
</tr>
<tr>
<td>12.0</td>
<td>14.0</td>
</tr>
<tr>
<td>16.0</td>
<td>4.0</td>
</tr>
<tr>
<td>36.0</td>
<td>29.0</td>
</tr>
<tr>
<td>13.0</td>
<td>18.0</td>
</tr>
<tr>
<td>11.0</td>
<td>8.0</td>
</tr>
<tr>
<td>28.0</td>
<td>33.0</td>
</tr>
<tr>
<td>17.0</td>
<td>10.0</td>
</tr>
<tr>
<td>5.0</td>
<td>30.0</td>
</tr>
<tr>
<td>12.0</td>
<td>14.0</td>
</tr>
<tr>
<td>16.0</td>
<td></td>
</tr>
</tbody>
</table>

10.3 Program Results

nag_trans_hessenberg_controller (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 |