

## nag trans hessenberg observer (g13ewc)

## 5 Arguments

- 1:    **n** – Integer *Input*  
*On entry:* the actual state dimension,  $n$ , i.e., the order of the matrix  $A$ .  
*Constraint:*  $\mathbf{n} \geq 1$ .
  
- 2:    **p** – Integer *Input*  
*On entry:* the actual output dimension,  $p$ .  
*Constraint:*  $\mathbf{p} \geq 1$ .
  
- 3:    **reduceto** – Nag\_ObserverForm *Input*  
*On entry:* indicates whether the matrix pair  $(A, C)$  is to be reduced to upper or lower observer Hessenberg form  
**reduceto** = Nag\_UH\_Observer  
           Upper observer Hessenberg form).  
**reduceto** = Nag\_LH\_Observer  
           Lower observer Hessenberg form).  
*Constraint:* **reduceto** = Nag\_UH\_Observer or Nag\_LH\_Observer.
  
- 4:    **a[n × tda]** – double *Input/Output*  
**Note:** the  $(i, j)$ th element of the matrix  $A$  is stored in  $\mathbf{a}[(i - 1) \times \mathbf{tda} + j - 1]$ .  
*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 *Input*  
*On entry:* the stride separating matrix column elements in the array **a**.  
*Constraint:*  $\mathbf{tda} \geq \mathbf{n}$ .
  
- 6:    **c[p × tdc]** – double *Input/Output*  
**Note:** the  $(i, j)$ th element of the matrix  $C$  is stored in  $\mathbf{c}[(i - 1) \times \mathbf{tdc} + j - 1]$ .  
*On entry:* the leading  $p$  by  $n$  part of this array must contain the output matrix  $C$  to be transformed.  
*On exit:* the leading  $p$  by  $n$  part of this array contains the transformed output matrix  $CU^T$ .
  
- 7:    **tdc** – Integer *Input*  
*On entry:* the stride separating matrix column elements in the array **c**.  
*Constraint:*  $\mathbf{tdc} \geq \mathbf{n}$ .
  
- 8:    **u[n × tdu]** – double *Input/Output*  
**Note:** the  $(i, j)$ th element of the matrix  $U$  is stored in  $\mathbf{u}[(i - 1) \times \mathbf{tdu} + j - 1]$ .  
*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: **tdu** – Integer *Input*

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

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

10: **fail** – NagError \* *Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

### NE\_2\_INT\_ARG\_LT

On entry, **tda** =  $\langle value \rangle$  while **n** =  $\langle value \rangle$ . These arguments must satisfy **tda**  $\geq$  **n**.

On entry **tdc** =  $\langle value \rangle$  while **n** =  $\langle value \rangle$ . These arguments must satisfy **tdc**  $\geq$  **n**.

On entry **tdu** =  $\langle value \rangle$  while **n** =  $\langle value \rangle$ . These arguments must satisfy **tdu**  $\geq$  **n**.

### NE\_BAD\_PARAM

On entry, argument **reduceto** had an illegal value.

### NE\_INT\_ARG\_LT

On entry, **n** =  $\langle value \rangle$ .

Constraint: **n**  $\geq$  1.

On entry, **p** =  $\langle value \rangle$ .

Constraint: **p**  $\geq$  1.

## 7 Accuracy

The algorithm is backward stable.

## 8 Parallelism and Performance

nag\_trans\_hessenberg\_observer (g13ewc) is not threaded in any implementation.

## 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  $(A, C)$  to upper observer Hessenberg form.

### 10.1 Program Text

```
/* nag_trans_hessenberg_observer (g13ewc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group
 *
 * Mark 26, 2016.
 */
```

```

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <naggl3.h>

#define A(I, J) a[(I) *tda + J]
#define C(I, J) c[(I) *tdc + J]
#define U(I, J) u[(I) *tdu + J]
int main(void)
{
    Integer exit_status = 0, i, j, n, p, tda, tdc, tdu;
    double *a = 0, *c = 0, one = 1.0, *u = 0, zero = 0.0;
    Nag_ObserverForm reduceto;
    NagError fail;

    INIT_FAIL(fail);

    printf("nag_trans_hessenberg_observer (g13ewc) 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_s("%" NAG_IFMT "%" NAG_IFMT "", &n, &p);
#else
    scanf("%" NAG_IFMT "%" NAG_IFMT "", &n, &p);
#endif
    if (n >= 1 || p >= 1) {
        if (!(a = NAG_ALLOC(n * n, double)) ||
            !(c = NAG_ALLOC(p * n, double)) || !(u = NAG_ALLOC(n * n, double)))
        {
            printf("Allocation failure\n");
            exit_status = -1;
            goto END;
        }
        tda = n;
        tdc = n;
        tdu = n;
    }
    else {
        printf("Invalid n or p.\n");
        exit_status = 1;
        return exit_status;
    }

    reduceto = Nag_UH_Observer;

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

    if (u) /* Initialize 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 (A,C) to reduced observer Hessenberg form. */
/* nag_trans_hessenberg_observer (g13ewc).
 * Unitary state-space transformation to reduce (AC) to
 * lower or upper observer Hessenberg form
 */
nag_trans_hessenberg_observer(n, p, reduceto, a, tda, c, tdc, u, tdu,
                             &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_trans_hessenberg_observer (g13ewc).\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
}

printf("\nThe 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("\nThe transformed input matrix is \n\n");
for (i = 0; i < p; ++i) {
    for (j = 0; j < n; ++j)
        printf("%8.4f ", C(i, j));
    printf("\n");
}
if (u) {
    printf("\nThe transformation matrix that reduces (A,C) "
           "to observer 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(c);
NAG_FREE(u);
return exit_status;
}

```

## 10.2 Program Data

nag\_trans\_hessenberg\_observer (g13ewc) Example Program Data

```

5      3
15.0  21.0  -3.0   3.0   9.0
20.0   1.0   2.0   8.0   9.0
 4.0   1.0   7.0  13.0  14.0
 5.0   6.0  12.0  13.0  -6.0
 5.0  11.0  17.0  -7.0  -1.0
 7.0  -1.0   3.0  -6.0  -3.0
 4.0   5.0   6.0  -2.0  -3.0
 9.0   8.0   5.0   2.0   1.0

```

## 10.3 Program Results

nag\_trans\_hessenberg\_observer (g13ewc) Example Program Results

The transformed state transition matrix is

```

 7.1637  -0.9691 -16.5046   0.2869   0.9205
-2.3285  11.5431  -8.7471   3.4122  -3.7118
-10.5440 -7.6032  -0.3215   3.6571  -0.4335
-3.6845   5.6449   0.5906 -15.6996  17.4267
 0.0000  -6.4260   1.5591  14.4317  32.3143

```

The transformed input matrix is

0.0000	0.0000	7.6585	5.2973	-4.1576
0.0000	0.0000	0.0000	5.8305	-7.4837
0.0000	0.0000	0.0000	0.0000	-13.2288

The transformation matrix that reduces (A,C) to observer Hessenberg form is

0.1863	-0.4823	0.2645	0.6648	-0.4698
-0.1137	-0.3601	0.6748	-0.0512	0.6320
0.6742	-0.5151	-0.1897	-0.4940	-0.0097
-0.1872	0.0813	0.5439	-0.5371	-0.6116
-0.6803	-0.6047	-0.3780	-0.1512	-0.0756

---