DTDSX – a Collection of Benchmark Examples for State-Space Realizations of Discrete-Time Dynamical Systems[1]

Daniel Kreßner, Volker Mehrmann[2], Thilo Penzl[3]

November 1998

[1]This paper describes Version 1.0 of the benchmark collection DTDSX. It presents research results of the European Community BRITE-EURAM III Thematic Networks Programme NICONET (contract number BRRT-CT97-5040) and is distributed by the Working Group on Software WGS. *WGS secretariat*: Mrs. Ida Tassens, ESAT - Katholieke Universiteit Leuven, K. Mercierlaan 94, 3001-Leuven-Heverlee, BELGIUM. This report is also available by anonymous ftp from *wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN1998-10.ps.Z*

[2]Fakultät für Mathematik, TU Chemnitz, D–09107 Chemnitz, Germany. Email: `volker.mehrmann@mathematik.tu-chemnitz.de`

[3]Fakultät für Mathematik, TU Chemnitz, D–09107 Chemnitz, Germany. Email: `thilo.penzl@mathematik.tu-chemnitz.de`

Contents

1 Introduction

In the analysis of numerical methods and their implementation as numerical software it is extremely important to test the correctness of the implementation as well as the performance of the method. This validation is one of the major steps in the construction of a software library, in particular, if this library is used in practical applications.

In order to carry out such tests it is necessary to have tools that yield an evaluation of the performance of the method as well as the implementation with respect to correctness, accuracy, and speed. Similar tools are needed to compare different numerical methods, to test their robustness, and also to analyze the behaviour of the methods in extreme situations, where the limit of the possible accuracy is reached.

In many application areas benchmark collections have been created that can partially serve for this purpose. Such collections are heavily used. In order to have a fair evaluation and a comparison of methods and software, there should be a standardized set of examples, which are freely available and on which newly developed methods and their implementations can be tested. Moreover, public benchmark collections can be used by developers of algorithms and software as a reference when reporting the results of numerical experiments in publications.

In order to make such collections useful it is important that they cover a wide range of problems. Two kinds of test problems are of particular interest. First, benchmark collections should contain so-called 'real world' examples, i.e., examples reflecting current problems in applications. Second, they must contain test examples which drive numerical methods and their implementations to a limit. These are ideal test cases because errors and failures usually occur only in extreme cases and these are often not covered by standard software validation procedures.

Whereas plenty of test problems for basic linear algebra problems (e.g., systems of linear equations, eigenvalue problems) are provided in several collections, e.g., [15], benchmark examples for control problems are currently harder to find. However, in the last few years collections for control problems containing relatively few test examples have been implemented. In particular, the collection for control system design [4] by E. Davison and the collection for discrete-time Riccati equations [2] by P. Benner, A. Laub, and V. Mehrmann should be mentioned here. Since both collections have been partly included into the benchmark collection described here, a lot of credit ought to be given to their authors.

In Sections 2–5, we provide test examples of time-invariant, discrete-time *descriptor systems*

$$Ex_{k+1} = Ax_k + Bu_k \tag{1}$$

$$y_k = Cx_k + Du_k, \tag{2}$$

where $E, A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$. However, a considerable part of these examples actually represent *standard systems*, i.e., systems where $E = I_n$. Note that we call the vector sequences $\{u_k\}_{k=0}^{\infty}$, $\{x_k\}_{k=0}^{\infty}$, and $\{y_k\}_{k=0}^{\infty}$ the *input*, the *state*, and the *output* of the system, respectively.

The examples of our collection are subdivided into four different *groups*:

- Group 1 — parameter-free problems of fixed size,
- Group 2 — parameter-dependent problems of fixed size,

- Group 3 — parameter-free examples of scalable size,
- Group 4 — parameter-dependent examples of scalable size.

The examples in the Groups 2 and 4 depend on several parameters, which have a direct impact on the algebraic properties of the descriptor system. The Groups 3 and 4 contain examples which are freely scalable by a so-called scaling parameter. Of course, this parameter can have an (indirect) influence on the algebraic system properties, too. In some of the examples, the parameters are restricted to certain ranges. These are indicated in the description. Moreover, default values are provided for each parameter.

Note that the current version of DTDSX does not contain examples corresponding to Group 4. However, examples of this group as well as further examples of the other groups can be obtained from the benchmark collection for continuous-time state-space systems CTDSX [10]. By means of the Caylay transformation (e.g., [16]), continuous-time systems can be transformed into their discrete-time counterparts and vice versa. See also Example 1.11.

The benchmark collection DTDSX has been implemented in FORTRAN and MATLAB. The FORTRAN codes conform to the implementation and documentation standards of the software library SLICOT [21].

We intend to augment the benchmark collection in the future. Contributions to forthcoming releases are highly appreciated.

2 Parameter-free problems of fixed size (Group 1)

The first few examples present 'academic' test problems of very small scale.

Example 1.1 [12, Example 2] (see also [2, Example 1])

n	m	p
2	1	1

This is an example of stabilizable-detectable, but uncontrollable-unobservable data. We have the following system matrices:

$$E = I_2, \qquad A = \begin{bmatrix} 4 & 3 \\ -\frac{9}{2} & -\frac{7}{2} \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \qquad C = \begin{bmatrix} 3 & 2 \end{bmatrix}, \qquad D = 0_{2\times 1}.$$

Example 1.2 [12, Example 3], [11, Example 6.15] (see also [2, Example 2])

n	m	p
2	2	2

The coefficient matrices are

$$E = I_2, \qquad A = \begin{bmatrix} 0.9512 & 0 \\ 0 & 0.9048 \end{bmatrix}, \qquad B = \begin{bmatrix} 4.877 & 4.877 \\ -1.1895 & 3.569 \end{bmatrix},$$

$$C = I_2, \qquad D = 0_{2\times 2}.$$

Example 1.3 [22, Example II] (see also [2, Example 3])

n	m	p
2	1	1

This example was used in [22] to demonstrate a compression technique for the extended pencil in context with a linear-quadratic optimal control problem. The data are given by

$$E = I_2, \qquad A = \begin{bmatrix} 2 & -1 \\ 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad C = [\,0 \;\; 1\,], \qquad D = 0.$$

Example 1.4 [8] (see also [2, Example 4])

n	m	p
2	2	2

The data are

$$E = I_2, \qquad A = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}, \qquad C = I_2, \qquad D = 0_{2\times 2}.$$

Example 1.5 [9], [16, Example 2] (see also [2, Example 5])

n	m	p
2	1	2

$$E = I_2, \qquad A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad C = I_2, \qquad D = 0_{2\times 1}.$$

Example 1.6 [1] (see also [2, Example 6])

n	m	p
4	2	4

The data of this example represent a simple control problem for a satellite. The system is given by equations describing the small angle altitude variations about the roll and yaw axes of a satellite in circular orbit. These equations originally form a second-order differential equation. A first-order realization of this model and sampling every 0.1 seconds yields the system matrices

$$E = I_4, \qquad A = \begin{bmatrix} 0.998 & 0.067 & 0 & 0 \\ -0.067 & 0.998 & 0 & 0 \\ 0 & 0 & 0.998 & 0.153 \\ 0 & 0 & -0.153 & 0.998 \end{bmatrix}, \qquad B = \begin{bmatrix} 0.0033 & 0.02 \\ 0.1 & -0.0007 \\ 0.04 & 0.0073 \\ -0.0028 & 0.1 \end{bmatrix},$$

$$C = I_4, \qquad D = 0_{4\times 2}.$$

Example 1.7 [13] (see also [2, Example 7])

n	m	p
4	2	4

This is a simple example of a control system having slow and fast modes.

$$E = I_4, \qquad A = 10^{-3} \times \begin{bmatrix} 984.75 & -79.903 & 0.9054 & -1.0765 \\ 41.588 & 998.99 & -35.855 & 12.684 \\ -546.62 & 44.916 & -329.91 & 193.18 \\ 2662.4 & -100.45 & -924.55 & -263.25 \end{bmatrix},$$

$$B = 10^{-4} \times \begin{bmatrix} 37.112 & 7.361 \\ -870.51 & 0.093411 \\ -11984.0 & -4.1378 \\ -31927.0 & 9.2535 \end{bmatrix}, \qquad C = I_4, \qquad D = 0_{4\times 2}.$$

Example 1.8 [14, Example 4.3] (see also [2, Example 8])

n	m	p
4	4	4

Here, the matrices of the system are constructed as follows. Given

$$A_0 = \begin{bmatrix} 0.4 & 0 & 0 & 0 \\ 1 & 0.6 & 0 & 0 \\ 0 & 1 & 0.8 & 0 \\ 0 & 0 & 0 & -0.999982 \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} 1 & -1 & -1 & -1 \\ 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

the system matrices are

$$E = I_4, \qquad A = VA_0V^{-1} = \begin{bmatrix} -0.6 & -2.2 & -3.6 & -5.400018 \\ 1 & 0.6 & 0.8 & 3.399982 \\ 0 & 1 & 1.8 & 3.799982 \\ 0 & 0 & 0 & -0.999982 \end{bmatrix}, \qquad B = V,$$

$$C = V^{-1} = \begin{bmatrix} 1 & 1 & 2 & 4 \\ 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad D = 0_{4\times 4}.$$

Example 1.9 [6, Section 2.7.4] (see also [2, Example 9])

n	m	p
5	2	5

The fifth-order linearized state-space model of a chemical plant presented in [7, 19] is discretized by sampling every 0.5 seconds, yielding a discrete-time system defined by

$$E = I_5, \qquad A = 10^{-4} \times \begin{bmatrix} 9540.70 & 196.43 & 35.97 & 6.73 & 1.90 \\ 4084.90 & 4131.70 & 1608.40 & 446.79 & 119.71 \\ 1221.70 & 2632.60 & 3614.90 & 1593.00 & 1238.30 \\ 411.18 & 1285.80 & 2720.90 & 2144.20 & 4097.60 \\ 13.05 & 58.08 & 187.50 & 361.62 & 9428.00 \end{bmatrix}$$

$$B = 10^{-4} \times \begin{bmatrix} 4.34 & -1.22 \\ 266.06 & -104.53 \\ 375.30 & -551.00 \\ 360.76 & -660.00 \\ 46.17 & -91.48 \end{bmatrix}, \qquad C = I_5, \qquad D = 0_{5\times 2}.$$

Example 1.10 [5] (see also [2, Example 10])

n	m	p
6	2	2

This is an example where the matrix D is not zero. The matrices of the linear system are given by:

$$E = I_6, \qquad A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \end{bmatrix}, \qquad D = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}.$$

Example 1.11 [18] (see also [2, Example 11])

n	m	p
9	3	2

This is the data for a 9th-order discrete state-space model of a tubular ammonia reactor. It should be noted that the underlying model includes a disturbance term which is neglected here. The continuous state-space model of this problem is provided in the benchmark collection CTDSX [10, Example 1.5]. Sampling every 30 seconds yields the following system matrices for the discrete model:

$$E = I_9, \qquad A = 10^{-2}\times$$

$$\begin{bmatrix} 87.01 & 13.50 & 1.159 & 0.05014 & -3.722 & 0.03484 & 0 & 0.4242 & 0.7249 \\ 7.655 & 89.74 & 1.272 & 0.05504 & -4.016 & 0.03743 & 0 & 0.4530 & 0.7499 \\ -12.72 & 35.75 & 81.70 & 0.1455 & -10.28 & 0.0987 & 0 & 1.185 & 1.872 \\ -36.35 & 63.39 & 7.491 & 79.66 & -27.35 & 0.2653 & 0 & 3.172 & 4.882 \\ -96.00 & 164.59 & -12.89 & -0.5597 & 7.142 & 0.7108 & 0 & 8.452 & 12.59 \\ -66.44 & 11.296 & -8.889 & -0.3854 & 8.447 & 1.36 & 0 & 14.43 & 10.16 \\ -41.02 & 69.30 & -5.471 & -0.2371 & 6.649 & 1.249 & 0.01063 & 9.997 & 6.967 \\ -17.99 & 30.17 & -2.393 & -0.1035 & 6.059 & 2.216 & 0 & 21.39 & 3.554 \\ -34.51 & 58.04 & -4.596 & -0.1989 & 10.56 & 1.986 & 0 & 21.91 & 21.52 \end{bmatrix},$$

$$B = 10^{-4}\times$$

$$\begin{bmatrix} 4.76 & 0.879 & 1.482 & 3.892 & 10.34 & 7.203 & 4.454 & 1.971 & 3.773 \\ -0.5701 & -4.773 & -13.12 & -35.13 & -92.75 & -61.59 & -36.83 & -15.54 & -30.28 \\ -83.68 & -2.73 & 8.876 & 24.80 & 66.80 & 38.34 & 20.29 & 6.937 & 14.69 \end{bmatrix}^T.$$

In the discrete model, only the first and fifth state variables are used as outputs

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad D = 0_{2\times 3}.$$

Example 1.12 [20] (see also [4, Example 4])

n	m	p
10	3	5

This problem deals with a two-stand cold rolling mill. The process dynamics are dominated by the interstand time delay. A discrete 10th order model is given. The model is multivariable and strongly interacting. The original problem [20] has disturbances which are ignored here.

$$E = I_{10}, \qquad A = \begin{bmatrix} 0_{1\times 9} & 0.112 \\ I_9 & 0_{9\times 1} \end{bmatrix}, \qquad B = \begin{bmatrix} 2.76 \quad -1.35 \quad -0.46 \\ 0_{9\times 3} \end{bmatrix},$$

$$C = \begin{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} & 0_{5\times 8} & \begin{bmatrix} 0 \\ 0.894 \\ -16.930 \\ 0.070 \\ 0.398 \end{bmatrix} \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 & 0 \\ -0.223 & 1.850 & -0.542 \\ 28.300 & 204.000 & 68.700 \\ -5.210 & -0.843 & -0.285 \\ -0.101 & -6.750 & -0.246 \end{bmatrix}$$

3 Parameter-dependent problems of fixed size (Group 2)

Example 2.1 [3], [17] (see also [2, Example 14])

n	m	p	parameter	default value
4	1	1	τ	10^8
			δ	1.0
			K	1.0

The following system describes a very simple process control of a paper machine. The continuous-time model with a time delay is sampled at intervals of length δ which yields a singular transition matrix A. The time delay is equal to the length of three sampling intervals. The other parameters defining the system are a first-order time constant τ and the steady-state gain K. The system is then given by

$$E = I_4, \qquad A = \begin{bmatrix} 1-\delta/\tau & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} K\delta/\tau \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}, \qquad D = 0.$$

4 Parameter-free examples of scalable size (Group 3)

Example 3.1 [17, Example 3] (see also [2, Example 15])

n	m	p	scaling parameter	default value
n	1	n	n	100

Here, the system is defined by

$$E = I_n, \qquad A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ & & & & 0 \\ 0 & & & 0 & 1 \\ 0 & \dots & & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix},$$

$$C = I_n, \qquad D = 0_{n\times 1}.$$

5 Parameter-dependent examples of scalable size (Group 4)

Version 1.0 of DTDSX does not contain parameter-dependent examples of scalable size.

Acknowledgements

The authors wish to thank P. Benner and E. Davison (IFAC), who considerably supported the development of the benchmark collection DTDSX.

References

[1] G. Ackerson and K. Fu. On the state estimation in switching environments. *IEEE Trans. Automat. Control*, AC-15:10–17, 1970.

[2] P. Benner, A. Laub, and V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations II: Discrete-time case. Technical Report SPC 95_23, Fak. f. Mathematik, TU Chemnitz–Zwickau, 09107 Chemnitz, FRG, 1995.

[3] W.L. Bialkowski. Application of steady-state Kalman filters–Theory with field results. In *Proc. Joint Automat. Cont. Conf.*, Philadelphia, PA, 1978.

[4] E.J. Davison. Benchmark problems in control system design. Report of the IFAC theory committee, International Federation of Automatic Control, May 1990.

[5] E.J. Davison and S.H. Wang. Properties and calculation of transmission zeros of linear multivariable systems. *Automatica*, 10:643–658, 1974.

[6] Z. Gajić and X. Shen. *Parallel Algorithms for Optimal Control of Large Scale Linear Systems*. Springer-Verlag, London, 1993.

[7] K. Gomathi, S. Prabhu, and M. Pai. A suboptimal controller for minimum sensitivity of closed-loop eigenvalues to parameter variations. *IEEE Trans. Automat. Control*, AC-25:587–588, 1980.

[8] V. Ionescu and M. Weiss. On computing the stabilizing solution of the discrete-time Riccati equation. *Linear Algebra Appl.*, 174:229–238, 1992.

[9] E. Jonckheere. On the existence of a negative semidefinite antistabilizing solution to the discrete algebraic Riccati equation. *IEEE Trans. Automat. Control*, AC-26:707–712, 1981.

[10] D. Kreßner, V. Mehrmann, and T. Penzl. CTDSX – a collection of benchmark examples for state-space realizations of continuous-time dynamical systems. SLICOT working note 1998-9.

[11] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems*. Wiley-Interscience, New York, 1972.

[12] A.J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Automat. Control*, AC-24:913–921, 1979. (see also *Proc. 1978 CDC (Jan. 1979)*, pp. 60-65).

[13] B. Litkouhi. *Sampled-Data Control of Systems with Slow and Fast Modes.* PhD thesis, Michigan State University, 1983.

[14] L.Z. Lu and W.W. Lin. An iterative algorithm for the solution of the discrete time algebraic Riccati equation. *Linear Algebra Appl.*, 188/189:465–488, 1993.

[15] Matrix Market. URL: `http://math.nist.gov/MatrixMarket/`.

[16] V. Mehrmann. A step towards a unified treatment of continuous and discrete time control problems. *Linear Algebra Appl.*, 241–243:749–779, 1996.

[17] T. Pappas, A.J. Laub, and N.R. Sandell. On the numerical solution of the discrete-time algebraic Riccati equation. *IEEE Trans. Automat. Control*, AC-25:631–641, 1980.

[18] L.M. Patnaik, N. Viswanadham, and I.G. Sarma. Computer control algorithms for a tubular ammonia reactor. *IEEE Trans. Automat. Control*, AC-25(4):642–651, 1980.

[19] V.G. Shankar and K. Ramar. Pole assignment with minimum eigenvalue sensitivity to parameter variations. *Internat. J. Control*, 23:493–504, 1976.

[20] H.W. Smith. Dynamic control of a two-stand cold mill. *Automatica*, 5:109–115, 1969.

[21] SLICOT Implementation and Documentation Standards. URL: `http://www.win.tue.nl/wgs/stand.html`.

[22] P. Van Dooren. A generalized eigenvalue approach for solving Riccati equations. *SIAM J. Sci. Statist. Comput.*, 2:121–135, 1981.

A The FORTRAN 77 subroutine BD02AD

The FORTRAN 77 subroutine BD02AD has the following calling sequence and in-line documentation.

```
      SUBROUTINE BD02AD( DEF, NR, DPAR, IPAR, VEC, N, M, P, E, LDE, A,
     1                   LDA, B, LDB, C, LDC, D, LDD, NOTE, DWORK,
     2                   LDWORK, INFO )
C
C     RELEASE 3.0, WGS COPYRIGHT 1999.
C
C     PURPOSE
C
C     This routine is an implementation of the benchmark library
C     DTDSX (Version 1.0) described in [1].
C
C     It generates benchmark examples for time-invariant,
C     discrete-time dynamical systems
C
C       E x_k+1 = A x_k + B u_k
C
C           y_k = C x_k + D u_k
C
C     E, A are real N-by-N matrices, B is N-by-M, C is P-by-N, and
C     D is P-by-M. In many examples, E is the identity matrix and D is
C     the zero matrix.
C
C
C     ARGUMENTS
C
C     Mode Parameters
C
C     DEF     CHARACTER*1
C             Specifies the kind of values used as parameters when
C             generating parameter-dependent and scalable examples
C             (i.e., examples with NR(1) = 2, 3, or 4):
C             DEF = 'D' or 'd': Default values defined in [1] are used.
C             DEF = 'N' or 'n': Values set in DPAR and IPAR are used.
C             This parameter is not referenced if NR(1) = 1.
C             Note that the scaling parameter of examples with
C             NR(1) = 3 or 4 is considered as a regular parameter in
C             this context.
C
C     Input/Output Parameters
C
```

```
C     NR       (input) INTEGER array, dimension 2
C              Specifies the index of the desired example according
C              to [1].
C              NR(1) defines the group:
C                    1 : parameter-free problems of fixed size
C                    2 : parameter-dependent problems of fixed size
C                    3 : parameter-free problems of scalable size
C                    4 : parameter-dependent problems of scalable size
C              NR(2) defines the number of the benchmark example
C              within a certain group according to [1].
C
C     DPAR     (input/output) DOUBLE PRECISION array, dimension 7
C              On entry, if DEF = 'N' or 'n' and the desired example
C              depends on real parameters, then the array DPAR must
C              contain the values for these parameters.
C              For an explanation of the parameters see [1].
C              For Example 2.1, DPAR(1), ..., DPAR(3) define the
C              parameters 'tau', 'delta', 'K', respectively.
C              On exit, if DEF = 'D' or 'd' and the desired example
C              depends on real parameters, then the array DPAR is
C              overwritten by the default values given in [1].
C
C     IPAR     (input/output) INTEGER array of DIMENSION at least 1
C              On entry, if DEF = 'N' or 'n' and the desired example
C              depends on integer parameters, then the array IPAR must
C              contain the values for these parameters.
C              For an explanation of the parameters see [1].
C              For Example 3.1, IPAR(1) defines the parameter 'n'.
C              On exit, if DEF = 'D' or 'd' and the desired example
C              depends on integer parameters, then the array IPAR is
C              overwritten by the default values given in [1].
C
C     VEC      (output) LOGICAL array, dimension 8
C              Flag vector which displays the availabilty of the output
C              data:
C              VEC(1), ..., VEC(3) refer to N, M, and P, respectively,
C              and are always .TRUE.
C              VEC(4) is .TRUE. iff E is NOT the identity matrix.
C              VEC(5), ..., VEC(7) refer to A, B, and C, respectively,
C              and are always .TRUE.
C              VEC(8) is .TRUE. iff D is NOT the zero matrix.
C
C     N        (output) INTEGER
C              The actual state dimension, i.e., the order of the
C              matrices E and A.
```

```
C
C     M       (output) INTEGER
C             The number of columns in the matrices B and D.
C
C     P       (output) INTEGER
C             The number of rows in the matrices C and D.
C
C     E       (output) DOUBLE PRECISION array, dimension (LDE,N)
C             The leading N-by-N part of this array contains the
C             matrix E.
C             NOTE that this array is overwritten (by the identity
C             matrix), if VEC(4) = .FALSE.
C
C     LDE     INTEGER
C             The leading dimension of array E. LDE >= N
C
C     A       (output) DOUBLE PRECISION array, dimension (LDA,N)
C             The leading N-by-N part of this array contains the
C             matrix A.
C
C     LDA     INTEGER
C             The leading dimension of array A. LDA >= N
C
C     B       (output) DOUBLE PRECISION array, dimension (LDB,M)
C             The leading N-by-M part of this array contains the
C             matrix B.
C
C     LDB     INTEGER
C             The leading dimension of array B. LDB >= N
C
C     C       (output) DOUBLE PRECISION array, dimension (LDC,N)
C             The leading P-by-N part of this array contains the
C             matrix C.
C
C     LDC     INTEGER
C             The leading dimension of array C. LDC >= P
C
C     D       (output) DOUBLE PRECISION array, dimension (LDD,M)
C             The leading P-by-M part of this array contains the
C             matrix D.
C             NOTE that this array is overwritten (by the zero
C             matrix), if VEC(8) = .FALSE.
C
C     LDD     INTEGER
C             The leading dimension of array D. LDD >= P
```

```
C
C     NOTE    (output) CHARACTER*70
C             String containing short information about the chosen
C             example.
C
C     Workspace
C
C     DWORK   DOUBLE PRECISION array, dimension (LDWORK)
C             NOTE that DWORK is not used in the current version
C             of DTDSX.
C
C     LDWORK  INTEGER
C             LDWORK >= 1.
C
C     Error Indicator
C
C     INFO    INTEGER
C             = 0:  successful exit;
C             < 0:  if INFO = -i, the i-th argument had an illegal
C                   value; in particular, INFO = -3 or -4 indicates
C                   that at least one of the parameters in DPAR or
C                   IPAR, respectively, has an illegal value;
C             = 1:  data file can not be opened or has wrong format.
C
C     REFERENCES
C
C     [1]  D. Kressner, V. Mehrmann, and T. Penzl.
C          DTDSX - a Collection of Benchmark Examples for State-Space
C          Realizations of Discrete-Time Dynamical Systems.
C          SLICOT working note 1998-10. 1998.
```

B The MATLAB function DTDSX

The MATLAB function `dtdsx` has the following calling sequence and in-line documentation.

```
function [E,A,B,C,D] = dtdsx(nr,parin)
%DTDSX
%
% Usage:  [E,A,B,C,D] = dtdsx(nr,parin)
%         [E,A,B,C,D] = dtdsx(nr)
%
% Main routine of the benchmark library DTDSX (Version 1.0) described
% in [1]. It generates benchmark examples for time-invariant,
% discrete-time, dynamical systems
%
%       E x_k+1 = A x_k + B u_k
%                                                                  (1)
%           y_k = C x_k + D u_k
%
% E, A are real n-by-n matrices, B is n-by-m, C is p-by-n, and
% D is p-by-m.
%
% Input:
%  - nr    : index of the desired example according to [1];
%            nr is a 1-by-2 matrix;
%            nr(1) defines the group:
%             = 1 : parameter-free problems of fixed size
%             = 2 : parameter-dependent problems of fixed size
%             = 3 : parameter-free problems of scalable size
%             = 4 : parameter-dependent problems of scalable size
%            nr(2) defines the number of the benchmark example within
%            a certain group.
%  - parin : parameters of the chosen example;
%            referring to [1], the entries in parin have the following
%            meaning:
%            Ex. 2.1 : parin(1:3) = [tau delta K]
%            Ex. 3.1 : parin(1)   = n
%            parin is optional; default values as defined in [1] are
%            used as example parameters if 'parin' is omitted. Note that
%            parin is not referenced if nr(1) = 1.
%
% Output:
%  - E, A, B, C, D :  matrices of the dynamical system (1).
%
% References:
%
```

```
% [1] D. Kressner, V. Mehrmann, and T. Penzl.
%     DTDSX - a Collection of Benchmark Examples for State-Space
%     Realizations of Discrete-Time Dynamical Systems.
%     SLICOT working note 1998-10. 1998.
```