An efficient algorithm for the solution of a coupled Sylvester equation appearing in descriptor systems ⋆

Amir Shahzad [a] Bryn Ll. Jones [b] Eric C. Kerrigan [a,c] George A. Constantinides [a]

[a] *Department of Electrical and Electronic Engineering, Imperial College London, SW7 2AZ, U.K.*

[b] *The Scottish Association for Marine Science, Scottish Marine Institute, Oban, Argyll, PA37 1QA, U.K.*

[c] *Department of Aeronautics, Imperial College London, SW7 2AZ, U.K.*

Abstract

Descriptor systems consisting of a large number of differential-algebraic equations (DAEs) usually arise from the discretization of partial differential-algebraic equations. This paper presents an efficient algorithm for solving the coupled Sylvester equation that arises in converting a system of linear DAEs to ordinary differential equations. A significant computational advantage is obtained by exploiting the structure of the involved matrices. The proposed algorithm removes the need to solve a standard Sylvester equation or to invert a matrix. The improved performance of this new method over existing techniques is demonstrated by comparing the number of floating-point operations and via numerical examples.

Key words: Descriptor systems; Matrix equations; Numerical algorithms; Coupled Sylvester equation; Efficient algorithms.

1 Introduction

Descriptor systems, also known as singular systems, implicit systems or generalized state-space systems, emerge in many engineering applications (Dai 1989, Kunkel & Mehrmann 2006). For example, in fluid mechanical systems, a descriptor system is produced by the discretization of Navier-Stokes equations (Jones, Kerrigan & Morrison 2009). Descriptor systems typically consist of coupled differential and algebraic equations. As a consequence, the control of descriptor systems is less well-understood than that for conventional state-space systems. However, it is often possible, via a sequence of transformations (Gerdin 2006), to completely decouple the differential and algebraic parts of a descriptor system, thus enabling the application of standard state-space control theory to this class of system.

There are three major steps in the transformation: a generalized Schur decomposition, also known as Weierstrass-Schur form; solving a coupled Sylvester equation, also called a generalized Sylvester equation; construction of appropriately-defined transformation matrices (Gerdin, Schön, Glad, Gustafsson & Ljung 2007). The first step is extensively well-studied in the field of numerical linear algebra (Kågström & Wiberg 2000, Golub & Van Loan 1996). Various existing methods for transforming a matrix into a Jordan-Schur form and a matrix pencil into a Weiestrass-Schur form are compared by Kågström & Wiberg (2000). Furthermore, these methods are extended to extracting the partial information corresponding to dominant eigenvalues from large scale matrices and matrix pencils. The solution and perturbation analysis of a coupled Sylvester equation is presented in Kågström (1994) and Kågström & Westin (1989). In Kågström & Westin (1989) the Schur method (Bartels & Stewart 1972) and the Hessenberg-Schur method (Golub, Nash & Van Loan 1979), which are used in solving a standard Sylvester equation, are extended for a coupled Sylvester equation. In Jones et al. (2009), the coupled Sylvester equation is transformed into a standard Sylvester equation and then standard techniques for solving a Sylvester equation are used.

This paper focuses on the efficient solution of the coupled Sylvester equation. The computational advantage

⋆ This paper was not presented at any IFAC meeting. Corresponding author E. C. Kerrigan. Tel. +44-(0)20-759-45139/6343. Fax +44-(0)20-7594-6282.

Email addresses: a.shahzad07@imperial.ac.uk (Amir Shahzad), bryn.jones@sams.ac.uk (Bryn Ll. Jones), e.kerrigan@imperial.ac.uk (Eric C. Kerrigan), g.constantinides@imperial.ac.uk (George A. Constantinides).

over existing methods is obtained by exploiting the special structure of the matrices involved in the transformation of a DAE. The main contribution of this paper is to present a new algorithm for the solution of the abovementioned coupled Sylvester equation, which is not only computationally more efficient than existing techniques, but also possesses the following important characteristics:

- no need to take an inverse of a matrix,
- no matrix by matrix multiplication, and
- no need to solve a standard Sylvester equation.

2 Problem formulation

Consider a linear differential-algebraic equation (DAE) of the form

$$E\dot{x}(t) = Fx(t) + Gu(t), \quad x(t_0) = x_0, \tag{1}$$

where $E, F \in \mathbb{C}^{n\times n}, G \in \mathbb{C}^{n\times m}$, $x(t)$ is the state vector and $u(t)$ is the input. Solving (1) for $x(t)$ with given initial condition x_0 and $u(\cdot)$ when E is non-singular is straightforward. In this paper, we assume that E is singular. Therefore, (1) cannot be solved by a standard linear ordinary differential equation (ODE) solver. To overcome this problem, we have adopted the procedure of Gerdin (2006), which transforms (1) into the following set of linear ODEs and a set of algebraic equations:

$$\dot{z}_1(t) = Az_1(t) + B_1u(t), \tag{2a}$$
$$z_2(t) = -\sum_{i=0}^{k-1} N^iB_2\frac{d^iu(t)}{dt^i}, \tag{2b}$$

where $A \in \mathbb{C}^{p\times p}, B_1 \in \mathbb{C}^{p\times m}, B_2 \in \mathbb{C}^{q\times m}$, $N \in \mathbb{C}^{q\times q}$ is a nilpotent matrix of index k i.e. $N^k = 0$, and $n = p+q$. Note that $z_1(t)$ and $z_2(t)$ are decoupled. Let us call (2) the *standard form* of (1). The theoretical background and the procedure to compute the matrices involved in this standard form are described next.

Definition 1 (Golub & Van Loan 1996) Let $E, F \in \mathbb{C}^{n\times n}$ be two matrices. A *matrix pencil* is a set of all matrices of the form $F-\lambda E$ with $\lambda \in \mathbb{C}$. The eigenvalues of this matrix pencil are defined by $\lambda(F, E) := \{s \in \mathbb{C} : \det(F - sE) = 0\}$.

Definition 2 (Kunkel & Mehrmann 2006) A matrix pencil $F - \lambda E$ is called *regular* if there exists an $s \in \mathbb{C}$ such that $\det(F-sE) \neq 0$, or equivalently, $\lambda(F, E) \neq \mathbb{C}$.

The regularity of a matrix pencil $F - \lambda E$ is equivalent to the existence and uniqueness of the solution for system (1) (Dai 1989).

Lemma 1 *(Gerdin 2006, Lemma* 2.1*) Consider a system (1). If the matrix pencil $F-\lambda E$ is regular, then there exist non-singular matrices P_1 and Q_1 such that*

$$P_1EQ_1 = \begin{bmatrix} E_1 & E_2 \\ 0 & E_3 \end{bmatrix} \text{ and } P_1FQ_1 = \begin{bmatrix} F_1 & F_2 \\ 0 & F_3 \end{bmatrix}, \tag{3}$$

where $E_1 \in \mathbb{C}^{p\times p}$ is non-singular, $E_3 \in \mathbb{C}^{q\times q}$ is upper triangular with all diagonal elements zero, $F_3 \in \mathbb{C}^{q\times q}$ is non-singular and upper triangular, $E_2, F_2 \in \mathbb{C}^{p\times q}$, and $F_1 \in \mathbb{C}^{p\times p}$.

The generalized Schur decomposition (3) and the subsequent reordering of the diagonal elements of E_1 can be done using MATLAB's `qz` and `ordqz` functions, respectively. These functions call LAPACK routines `zgges` and `ztgsen` for complex matrices.

Remark 1 The decomposition of the matrix pencil $F - \lambda E$ by MATLAB's `qz` function produces upper triangular matrices. Therefore, E_1 and F_1 would be upper triangular.

There are three main steps in computing the standard state-space form of (1), which are listed below:

(1) Compute the generalized Schur decomposition of the matrix pencil $F - \lambda E$ as

$$P_1(F - \lambda E)Q_1 = \begin{bmatrix} F_1 & F_2 \\ 0 & F_3 \end{bmatrix} - \lambda \begin{bmatrix} E_1 & E_2 \\ 0 & E_3 \end{bmatrix}. \tag{4}$$

(2) Solve the following coupled Sylvester equation for L and R:

$$E_1R + LE_3 = -E_2, \tag{5a}$$
$$F_1R + LF_3 = -F_2, \tag{5b}$$

where $L, R \in \mathbb{C}^{p\times q}$.

(3) According to Lemma 1, if the matrix pencil $F-\lambda E$ in (1) is regular, there exist nonsingular matrices P and Q such that the transformation:

$$PEQQ^{-1}\dot{x}(t) = PFQQ^{-1}x(t) + PGu(t) \tag{6}$$

gives the system in standard form (2), where

$$P := \begin{bmatrix} E_1^{-1} & 0 \\ 0 & F_3^{-1} \end{bmatrix} \begin{bmatrix} I & L \\ 0 & I \end{bmatrix} P_1, \tag{7a}$$

$$Q := Q_1 \begin{bmatrix} I & R \\ 0 & I \end{bmatrix}, N := F_3^{-1}E_3, A := E_1^{-1}F_1, \tag{7b}$$

$$\begin{bmatrix} B_1 \\ B_2 \end{bmatrix} := PG, \; x(t) = Q \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix}. \tag{7c}$$

Algorithm 1 Solution of a coupled Sylvester equation

Input: $E_1, E_2, E_3, F_1, F_2, F_3$
Output: R, L.
Algorithm:

1: **for** $i = 1$ to q **do**
2: Solve $E_1 r_i = -e_i^2 - \sum_{k=1}^{i-1} e_{ki}^3 l_k$ for r_i using backward substitution.
3: Compute $l_i = -\frac{1}{f_{ii}^3}\left(f_i^2 + F_1 r_i + \sum_{k=1}^{i-1} f_{ki}^3 l_k\right)$
4: **end for**

3 Solution of the coupled Sylvester equation

In this section, we propose a new and efficient algorithm for the solution of the coupled Sylvester equation (5) for R and L. This is done by exploiting the structure of the given matrices. From (5), we get

$$E_1 R + L \begin{bmatrix} 0 & e_{12}^3 & e_{13}^3 & \cdots & e_{1,q}^3 \\ 0 & 0 & e_{22}^3 & \cdots & e_{2,q}^3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & e_{q-1,q}^3 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} = -E_2, \tag{8a}$$

$$F_1 R + L \begin{bmatrix} f_{11}^3 & f_{12}^3 & \cdots & f_{1,q}^3 \\ 0 & f_{22}^3 & \cdots & f_{2,q}^3 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_{q,q}^3 \end{bmatrix} = -F_2, \tag{8b}$$

where e_{ij}^k denotes the $(i,j)^{\text{th}}$ element of matrix E_k. The i^{th} column of R, L, E_2, and F_2 is denoted by r_i, l_i, e_i^2, and f_i^2 respectively. By comparing the first column of both sides of (8a), we get $E_1 r_1 = -e_1^2$. Since E_1 is upper triangular, the above equation can be solved for r_1 using backward substitution. By comparing the first column of both sides of (8b), we get $l_1 = -\frac{1}{f_{11}^3}\left(f_1^2 + F_1 r_1\right)$ if $f_{11}^3 \neq 0$. Since F_3 is non-singular and upper-triangular, $f_{ii}^3 \neq 0$ for each i. Similarly, by comparing the i^{th} column of (8a), we get $E_1 r_i = -e_i^2 - \sum_{k=1}^{i-1} e_{ki}^3 l_k$ and by comparing the i^{th} column of (8b) we get $l_i = -\frac{1}{f_{ii}^3}\left(f_i^2 + F_1 r_i + \sum_{k=1}^{i-1} f_{ki}^3 l_k\right)$ if $f_{ii}^3 \neq 0$. The complete algorithm is described in Algorithm 1. Algorithm 1 is well-defined for $q \geq 1$, which is proven next.

Proposition 1 *Consider Lemma 1. If rank$(E) = n - r$ for $r \geq 1$, then $q \geq 1$.*

Proof. Let $\tilde{E} := P_1 E Q_1$, where P_1 and Q_1 are non-singular matrices defined in Lemma 1, hence $\text{rank}(P_1) = \text{rank}(Q_1) = n$.

$$\begin{aligned}\text{rank}(\tilde{E}) &= \text{rank}(P_1 E Q_1) \\ &\leq \min\{\text{rank}(P_1), \text{rank}(E), \text{rank}(Q_1)\} = n - r\end{aligned}$$

This means that $\tilde{E}$ is rank deficient and at least r eigenvalues of $\tilde{E}$ would be zero. Since $E_1 \in \mathbb{C}^{p\times p}$ is non-singular and upper triangular and $E_3 \in \mathbb{C}^{q\times q}$ is upper triangular with all diagonal elements zero, it follows from the above definition of $\tilde{E}$ with (3) that $\tilde{E}$ has p non-zero eigenvalues and q zero eigenvalues. Hence $q \geq r \geq 1$. □

4 Computational Complexity Analysis

In this section we will first present an overview of existing techniques used to compute the solution of a coupled Sylvester equation. A computational complexity analysis of these methods, followed by our proposed method, is then presented. The existing methods used in the literature for the solution of coupled Sylvester equation are listed below:

(1) Solving (5) is equivalent to solving the following linear system of equations (Kågström 1994):

$$\underbrace{\begin{bmatrix} I_q \otimes E_1 & E_3^T \otimes I_p \\ I_q \otimes F_1 & F_3^T \otimes I_p \end{bmatrix}}_{H} \begin{bmatrix} \text{vec}(R) \\ \text{vec}(L) \end{bmatrix} = \begin{bmatrix} -\text{vec}(E_2) \\ -\text{vec}(F_2) \end{bmatrix}, \tag{9}$$

where $\otimes$ denotes the Kronecker product, and vec(L) denotes an ordered stack of the columns of the matrix L from left to right, starting with the first column. Note that E_3^T denotes the transpose of E_3, but not the complex conjugate transpose.

(2) Solve the following discrete-time Sylvester equation for L (Jones et al. 2009):

$$A_s L B_s - L + C_s = 0, \tag{10}$$

where $A_s := F_1 E_1^{-1}, B_s := E_3 F_3^{-1}$, and $C_s := -(F_2 - F_1 E_1^{-1} E_2) F_3^{-1}$. Substitute L in the following equation to get R:

$$R = -E_1^{-1}(E_2 - L E_3). \tag{11}$$

An apparent drawback in the arguments A_s, B_s and C_s of the discrete-time Sylvester equation (10) is the inclusion of the inverses of E_1 and F_3. This can be eradicated by using an alternative form of these arguments, which is

$$A_s = \left((E_1^T)^{-1} F_1^T\right)^T, B_s = \left((F_3^T)^{-1} E_3^T\right)^T, \tag{12a}$$

$$C_s = -\left((F_3^T)^{-1}(F_2 - A_s E_2)^T\right)^T. \tag{12b}$$

Table 1
Computational cost of solving a coupled Sylvester equation

Method	**Flops**	**Flops** $p = q = \frac{n}{2}$
LU (Golub & Van Loan 1996)	$\frac{16}{3}p^3q^3$	$0.08n^6$
Sylvester (Jones et al. 2009)	$\frac{8}{3}p^3 + 11q^3 + 6p^2q + \frac{3}{2}pq^2$	$2.65n^3$
New	$2p^2q + 2pq^2$	$0.5n^3$

Since E_1 and F_3 are upper-triangular matrices, the matrices in (12) can be computed by forward substitution.

Computational complexity of each method will be measured in terms of floating-point operations (flops). A flop is defined as one addition, subtraction, multiplication or division of two floating-point numbers (Golub & Van Loan 1996). For simplicity of presentation, all matrices are considered to be real, and only higher order terms that contribute the most towards the computational cost of an algorithm are presented in this section.

The computational cost of calculating the generalized Schur decomposition (4) by the `qz` routine is $66n^3$ flops (Golub & Van Loan 1996). The computational cost of calculating all matrices in (7) is $3mn^2 + \frac{1}{3}(p^3 + q^3)$ flops.

LU factorization can be used to solve (9), and its computational cost is given in Table 1. The computational cost of forming the Sylvester equation is $p^3 + p^2q + q^3 - pq^2$, and solving this Sylvester equation by the Hessenberg-Schur algorithm (Golub et al. 1979) is $\frac{5}{3}p^3 + 10q^3 + 5p^2q + \frac{5}{2}pq^2$. The total cost of this method is given in Table 1 and named Sylvester. The computational cost of solving a coupled Sylvester equation by generalized Schur method is $15p^3 + 15q^3 + 5p^2q + 5pq^2 + 97/4pq$ and by generalized Hessenburg-Schur method is $(5 + 35/12q)p^3 + 15q^3 + 19/2p^2q + 6pq^2$ (Kågström & Westin 1989), which is not better than the Sylvester method. The computational cost of our proposed method is named New and given in Table 1.

To see the computational cost in terms of the number of states n, we define $p := \alpha n$ for $0 < \alpha < 1$. Since $n = p+q$, this implies that $q = (1 - \alpha)n$. For a typical case when $\alpha = 0.5$, the computational cost of these methods is shown in the last column of Table 1.

5 Effect of rounding errors

In this section, we briefly describe the effect of finite precision arithmetic in the solution of a coupled Sylvester equation. We know from linear system sensitivity theory that if H is ill-conditioned, then small changes in E_1, E_3, F_1 and F_3 can induce large changes in the solution of (9). From (10), we have

$$\underbrace{(B_s^T \otimes A_s - I_{pq})}_{H_s} \mathrm{vec}(L) = -\mathrm{vec}(C_s). \qquad (13)$$

The condition number is generally used as a measure of sensitivity; for large condition numbers we may expect large rounding errors due to finite precision effects compared to small condition numbers. To compare the effect of rounding errors of our proposed method over existing methods, we compare the condition number of E_1 with H and H_s. For a detailed error analysis of the solution to a coupled Sylvester equation, see Kågström & Westin (1989) and Kågström (1994), for the solution to a Sylvester equation see Golub et al. (1979) and Higham (2002, Chapter 16), and for the solution to a triangular system see Higham (2002, Chapter 8).

6 Numerical Results

To compare the performance of our proposed method (Algorithm 1) with existing methods (9) and (10), we applied these algorithms on two examples.

6.1 Example 1

Consider an example from fluid mechanics, namely plane Poiseuille flow. A number of linear DAEs of the form (1) of different state dimensions are obtained by the discretization of linearized Navier-Stokes equations by changing the number of grid points (Jones et al. 2009).

All three algorithms to solve the coupled Sylvester equation described in the previous section have been implemented in MATLAB [1]. Sparse LU factorization is used to solve (9) via MATLAB's `mldivide` routine. (10) is a discrete-time Sylvester equation, and its solution is computed using the `dlyap` routine, which implements SLICOT routine `SB04QD`. The computational time taken by each method is measured by MATLAB's `tic` and `toc` functions and is shown in Fig. 1(a). It is obvious that our proposed method is better in terms of computational time and conditioning than the sparse LU factorization and Sylvester method. The error in the solution of the coupled Sylvester equation is defined as

$$\epsilon := \|E_1R + LE_3 + E_2\|_2 + \|F_1R + LF_3 + F_2\|_2, \quad (14)$$

and is shown in Fig. 1(b). It is evident from Fig. 1(b) that the error in the solution of our proposed scheme is comparable to other techniques. The second important factor in the solution of the coupled Sylvester equation

[1] All computations are performed on a 2.83GHz Quad Core Intel CPU machine in MATLAB 7.6.0 (R2008a) using IEEE double precision.

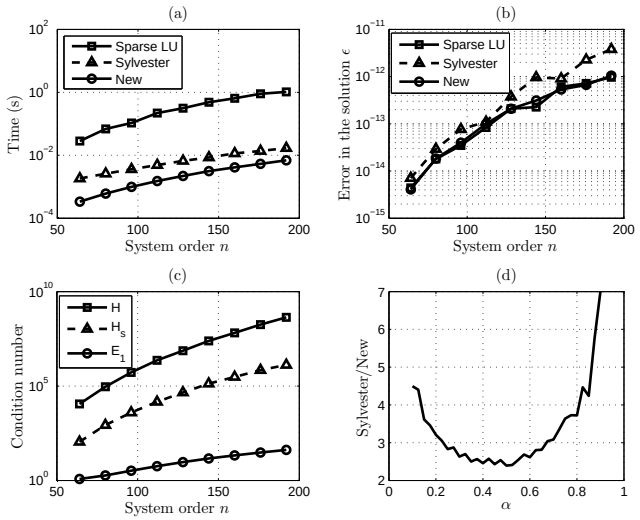

Fig. 1. (a) Time to compute the solution of a coupled Sylvester equation with variable system order n, (b) Error in the solution of a coupled Sylvester equation with variable system order n, (c) Condition number of H, H_s and E_1 with variable system order n, (d) Ratio of the time taken by the Sylvester method to our proposed method when solving a coupled Sylvester equation, where $\alpha = p/(p+q)$. The simulation results are computed with fixed number of states $n = 200$.

.

is the condition number of the matrices that need to be inverted. Fig. 1(c) shows how the condition number of matrix H, which appears in the LU factorization, H_s associated with (13), and E_1 increases with the system order n. In Algorithm 1 at line 2 the solution of a linear system with upper triangular matrix E_1 is determined with back substitution only, therefore the condition number of E_1 can be considered as the condition number of our method. This shows that our proposed method is better in terms of conditioning than LU factorization of (9) and the Sylvester method.

6.2 Example 2

In this example a number of DAEs of the form (1) are generated randomly by varying the rank of E and keeping the number of states fixed. The $(2,2)$ block matrix of E of size $q \times q$ is taken as zero while F and other block matrices of E are generated randomly by MATLAB command `sprandsym`. This command gives us a sparse symmetric random matrix whose elements are normally distributed with mean zero and variance one. The generalized Schur decomposition of each matrix pencil $F - \lambda E$ gives us a coupled Sylvester equation with different dimension variables p and q. This set of coupled Sylvester equations is solved by the method used by Jones et al. (2009), called the Sylvester method, and our proposed one in MATLAB. To see the gain of our method over the Sylvester method, the ratio of the computational time of the Sylvester method to our method against $\alpha = p/(p+q)$ is plotted in Fig. 1(d). It shows that our method is at least 2.5 times faster than the Sylvester method. The gain is even more at lower and higher values of α, which is consistent with the results of Table 1.

7 Conclusions

This paper was motivated by the fact that the discretization of partial differential-algebraic equations leads to a large number of differential-algebraic equations, and the solution time of DAEs grows with the number of states. An attempt has been made in this technical note to reduce the computational time of the coupled Sylvester equation, which is one part in computing the solution of a differential-algebraic system. It was shown that the proposed algorithm is not only computationally more efficient than existing techniques, but also numerically better conditioned. It is shown numerically that our algorithm is at least 2.5 times faster than the method used in Jones et al. (2009).

References

Bartels, R. H. & Stewart, G. W. (1972), 'Solution of the matrix equation $AX + XB = C$', *Communications of the ACM* **15**(9), 820–826.

Dai, L. (1989), *Singular control systems*, Lecture Notes in Control and Information Sciences, Springer-Verlag, Berlin Heidelberg, New York.

Gerdin, M. (2006), Identification and Estimation for Models Described by Differential-Algebraic Equations, PhD thesis, Linköping University, Sweden.

Gerdin, M., Schön, T. B., Glad, T., Gustafsson, F. & Ljung, L. (2007), 'On parameter and state estimation for linear differential-algebraic equations', *Automatica* **43**(3), 416–425.

Golub, G. A. & Van Loan, C. F. (1996), *Matrix Computations*, The John Hopkins University Press.

Golub, G. H., Nash, S. & Van Loan, C. F. (1979), 'A Hessenberg-Schur method for the problem $AX + XB = C$', *IEEE Transactions on Automatic Control* **AC-24**(6), 909–913.

Higham, N. J. (2002), *Accuracy and Stability of Numerical Algorithms*, second edn, SIAM.

Jones, B. L., Kerrigan, E. C. & Morrison, J. F. (2009), A modeling and filtering framework for the semi-discretised Navier-Stokes equations, *in* 'Proc. European Control Conference', Budapest, Hungary, pp. 138–143.

Kågström, B. (1994), 'A perturbation analysis of the generalized Sylvester equation', *SIAM Journal on Matrix Analysis and Applications* **15**(4), 1045–1060.

Kågström, B. & Westin, L. (1989), 'Generalized Schur methods with condition estimators for solving the generlized Sylvester equation', *IEEE Transactions on Automatic Control* **34**(7), 745–751.

Kågström, B. & Wiberg, P. (2000), 'Extracting partial canonical structure for large scale eigenvalue problems', *Numerical algorithms* **24**(3), 195–237.

Kunkel, P. & Mehrmann, V. (2006), *Differential-Algebraic Equations: Analysis and Numerical Solution*, European Mathematical Society.