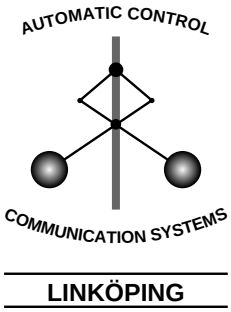
Noise Modeling, State Estimation and System Identification in Linear Differential-Algebraic Equations

Markus Gerdin, Thomas Schön

Control & Communication
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
WWW: http://www.control.isy.liu.se
E-mail: gerdin@isy.liu.se, schon@isy.liu.se

1st November 2004



Report no.: LiTH-ISY-R-2639
Submitted to CCSSE 2004

Abstract

This paper treats how parameter estimation and Kalman filtering can be performed using a Modelica model. The procedures for doing this have been developed earlier by the authors, and are here exemplified on a physical system. It is concluded that the parameter and state estimation problems can be solved using the Modelica model, and that the parameters estimation and observer construction to a large extent could be automated with relatively small changes to a Modelica environment.

Keywords: DAE, Differential-Algebraic Equations, Modelica, System Identification, Parameter Estimation, Observer, Kalman Filter.

Noise Modeling, State Estimation and System Identification in Linear Differential-Algebraic Equations

Markus Gerdin and Thomas Schön
Department of Electrical Engineering, Linköping University, Sweden
gerdin,schon@isy.liu.se

Abstract

This paper treats how parameter estimation and Kalman filtering can be performed using a Modelica model. The procedures for doing this have been developed earlier by the authors, and are here exemplified on a physical system. It is concluded that the parameter and state estimation problems can be solved using the Modelica model, and that the parameters estimation and observer construction to a large extent could be automated with relatively small changes to a Modelica environment.

Keywords: DAE, Differential-Algebraic Equations, Modelica, System Identification, Parameter Estimation, Observer, Kalman Filter.

1 Introduction

This paper is the result of a project within the PhD course Project-Oriented Study (POS), which is a part of the graduate school ECSEL. The aim of the project was to model a physical system in Modelica, estimate unknown parameters in the model using measurements from a real process, and then implement a Kalman filter to estimate unknown states. These steps are described in the paper. The work is based on theory which has been developed earlier by the authors [21, 10].

In the paper it is concluded that the estimation of unknown parameters and construction of observers (Kalman filters) could be automated to a large extent, if relatively small additions are introduced in the Modelica modeling environment. This could significantly reduce the time for modeling a system, estimating parameters, and constructing observers.

1.1 Process Description

The process we are studying in this project resembles the problem that occurs in power transfers in weak axes, such as the rear axis in trucks. This problem is studied within the area of power train control, which is an important research area for the automotive industry. The problem is that when large torques are generated by the engine the drive shaft winds itself up like a spring. In the gear shifting process when the engine torque is removed this spring will affect the gearbox, since the drive shafts winds itself back again. This is an undesirable behavior, since the gear shifting process is significantly slowed down by this fact, which will be a sever problem for automatic gearboxes. Furthermore, this torque will expose parts in the gearbox to wear. Hence, it is desirable to eliminate the elastic behavior of the drive shafts. This can be done by closing the loop using feedback control. In order to be able to construct a good controller a model of the process is needed and the internal states of this model have to be estimated. We will in this report study the modeling and state estimation problems for this process. However, since we do not have access to a real truck we will use a laboratory scale process, which contains the same phenomena. The engine is substituted with a DC-motor and a gearbox. The drive shafts and the load have been replaced by a spring and a heavy metal disc. In Figure 1, a photo of the laboratory process is given.

1.2 Object-Oriented Modeling

Simulation of dynamical systems has traditionally been performed by writing equations describing the system as ordinary differential equations (ODE), and then integrating them using an ODE solver. Tools such as SIMULINK have made this process easier by allowing graphical modeling of the ODE, but in principle the user must still transform the equations to ODE form manually. In recent years, object-oriented modeling languages have become increasingly popular. The most promising example of such a language is probably Modelica [8, 23], which is the language that will be used in this work. (More specifically, we will use the Dymola implementation of Modelica). Object-oriented languages have the advantage of being equation-based, which means that the user can enter the equations de-

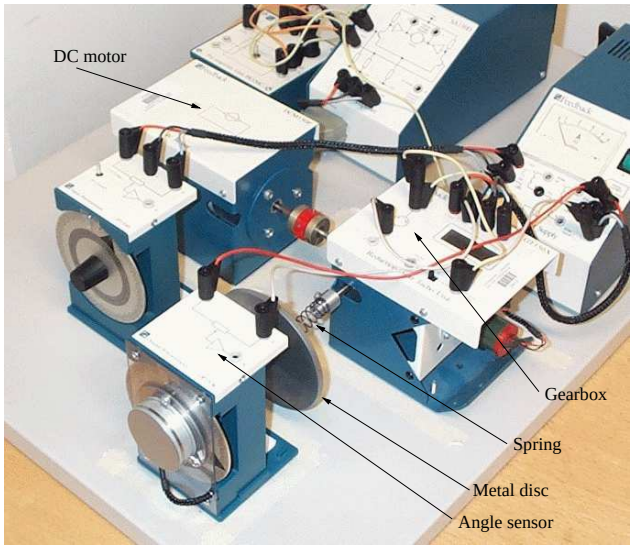


Figure 1. The examined process.

scribing the system without having to transform them into ODE form. The equations will instead be in differential-algebraic equation (DAE) form, which means that modeling environments must be able to simulate such models. The term *object-oriented* means that equations describing a commonly used system can be packaged, stored and reused. It can for example be convenient to not have to enter the equations describing a resistor every time it is used in a model. Modeling environments for object-oriented modeling languages usually allow graphical modeling, which means that different sub-models are selected from component libraries and then connected using a graphical interface.

2 System Identification

In this section we will describe how the unknown parameters in the model have been estimated from measured data. First, we will discuss some basic theory for estimation of unknown parameters in DAE:s in Section 2.1.

2.1 Theory for Parameter Estimation in DAE:s

In the general case, a model from an object-oriented modeling tool is a nonlinear DAE, which can be written as

$$F(\dot{\xi}(t), \xi(t), u(t), \theta) = 0 \tag{1a}$$
$$y(t) = h(\xi(t), \theta). \tag{1b}$$

Here, $u(t)$ is a known input, $y(t)$ is the measured output, $\xi(t)$ is a vector of auxiliary variables used to model the system, and θ is a vector of unknown parameters. The problem is to estimate the constant values of the parameters θ using measurements of $u(t)$ and $y(t)$.

A straightforward way to solve this problem is to solve the optimization problem

$$\min_{\theta} \sum_{t} (\hat{y}(t|\theta) - y(t))^2 \tag{2}$$

where the predictor $\hat{y}(t|\theta)$ is selected as a simulated output from the model, and $y(t)$ is the measured output. In system identification terms, this corresponds to prediction error identification with an output-error (OE) model. An output-error model corresponds to the assumptions that the measurement (1b) is corrupted by noise and that the DAE (1a) holds exactly. For a more thorough discussion on noise models in system identification see, e.g., [14]. In the case when the DAE is linear,

$$E(\theta)\dot{\xi}(t) = J(\theta)\xi(t) + K(\theta)u(t) \tag{3a}$$
$$y(t) = C(\theta)\xi(t), \tag{3b}$$

other noise models than output-error can be used, which can give better estimates of the unknown parameters. With other noise models than output error (2) can still be used, but $\hat{y}(t|\theta)$ will be a predicted output, which in the linear case is calculated by the Kalman filter. To implement a Kalman filter for (3), different transformations can be used. The method used in this work is to transform the DAE into state-space form using the transformation sketched below.

$$E\dot{\xi} = J\xi + Ku \Rightarrow \tag{4}$$
$$\begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix} \begin{bmatrix} \dot{w}_1 \\ \dot{w}_2 \end{bmatrix} + \begin{bmatrix} -A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} B_1 \\ D_1 \end{bmatrix} u \Rightarrow \tag{5}$$
$$\begin{bmatrix} \dot{w}_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} Aw_1 + B_1 u \\ \sum_{i=0}^{m-1} (-N)^i D_1 u^{(i)} \end{bmatrix} \Rightarrow \tag{6}$$
$$\dot{x} = \tilde{A}x + \tilde{B}u^{(m-1)} \tag{7}$$

The first step is calculated using numerical software, and must therefore be calculated for every parameter value for which a state-space description is necessary. Note that in the last transformation, the input might be redefined as one of its derivatives. For a more thorough discussion on this see, e.g., [9] or [10].

2.2 Parameter Estimation for Laboratory Process

Since the laboratory process that was modeled in Modelica contains some unknown parameters, the technique described in the previous section was used to estimate the unknown parameters. The model is linear, so the equations can be written as

$$E(\theta)\dot{\xi}(t) = J(\theta)\xi(t) + K(\theta)u(t) \tag{8a}$$
$$y(t) = C(\theta)\xi(t). \tag{8b}$$

During the identification experiment, the input was the voltage to the motor. Two outputs were measured, the angle of the mass after the spring, and the angular velocity of the mass before the spring.

The actual identification was performed using the System identification toolbox for MATLAB. It was therefore necessary to transform the Modelica model to the format (8) in MATLAB. This was performed manually, but the procedure could quite easily be automated if it were possible to specify inputs, outputs, and unknown parameters in Modelica. This is an important subject for future work, since grey-box identification then could be performed by first modeling the system using Modelica, and then estimating unknown parameters in MATLAB without having to manipulate any equations manually.

The data from the process was collected with the sampling interval 0.01 s. However, this was too fast for this process and would have required higher order models, so the signals were resampled at 0.05 s. The input voltage (a sum of sinusoids) was multiplied by 90 since the voltage is passed through an amplifier that amplifies it 90 times. The mean values were also removed from the data, and the data set was divided into estimation and validation data. The preprocessed data that was used for the identification is shown in Figure 2–3.

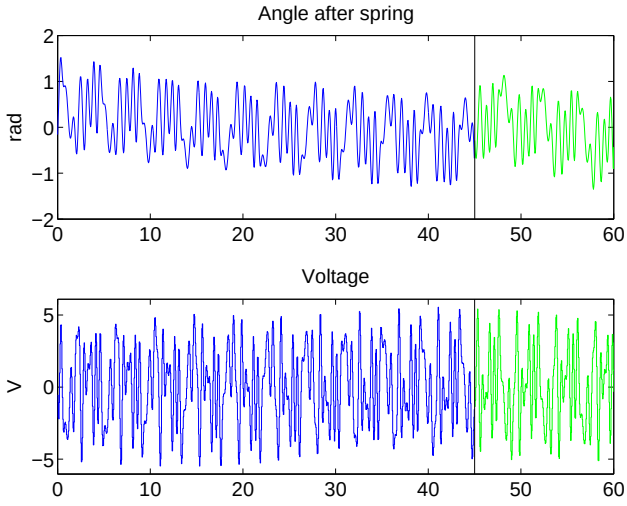


Figure 2. Input-output data from motor voltage to the angle after the spring. The data to the left of the vertical line is estimation data, and the data to the right is validation data.

To get a feeling for the system, two ARX models (from voltage to angular velocity before the spring and from voltage to angle after spring respectively) were estimated. The orders of the models were selected to be the same as the physical Modelica model of the system (three continuous-time poles from voltage to angular velocity before the spring, and four continuous-time poles to angular velocity after the spring). As can be seen in Figure 4–5, the fit is quite good.

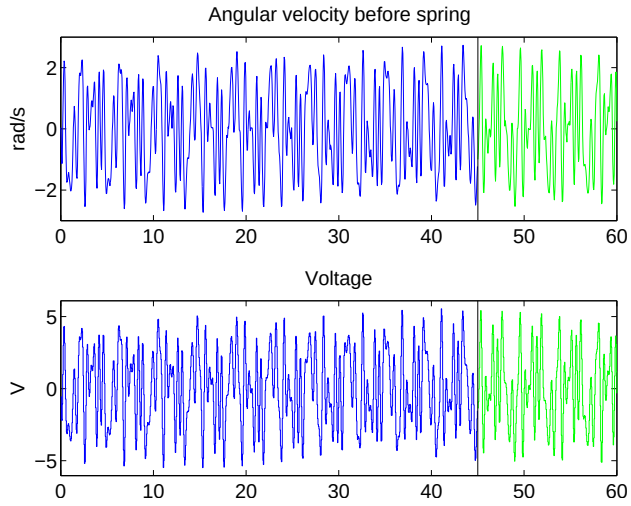


Figure 3. Input-output data from motor voltage to angular velocity before the spring. The data to the left of the vertical line is estimation data, and the data to the right is validation data.

By noting that the transfer functions from voltage to angular velocity before the spring from the ARX model and from the physical model respectively should be the same, it is possible to quite easily estimate the value of k.

$$G_{\text{ARX}} = \frac{13.68s^2 + 621.9s + 6128}{s^3 + 24.95s^2 + 1394s + 11180} \tag{9}$$

$$G_{\text{physical}} = -30k\frac{J_2 s^2 + ds + c}{900RJ_1J_2s^3 + \dots} \frac{\dots}{(900J_2k^2 + 900RJ_1d + J_2Rd)s^2 + \dots} \frac{\dots}{(900dk^2 + 900RJ_1c + J_2Rc)s + 900ck^2} \tag{10}$$

We see that an approximate value of k is given by

$$k \approx \frac{11180}{6128 \cdot (-30)} \approx -0.06. \tag{11}$$

This value will later be used as an initial value for k in the parameter estimation.

Since the ARX models give such good fits, they might be all that is necessary for some applications. However, we are interested in the actual parameter values and to have a physical model where the connection between the different outputs is clear, so we proceed with parameter estimation for the physical model.

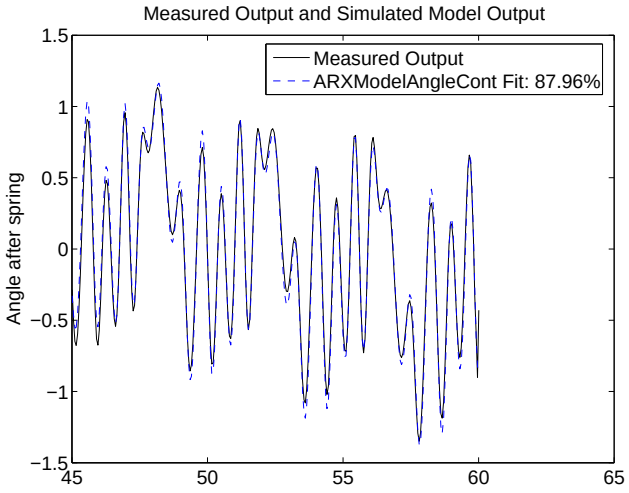


Figure 4. Validation of ARX model from voltage to angle after spring using simulated output.

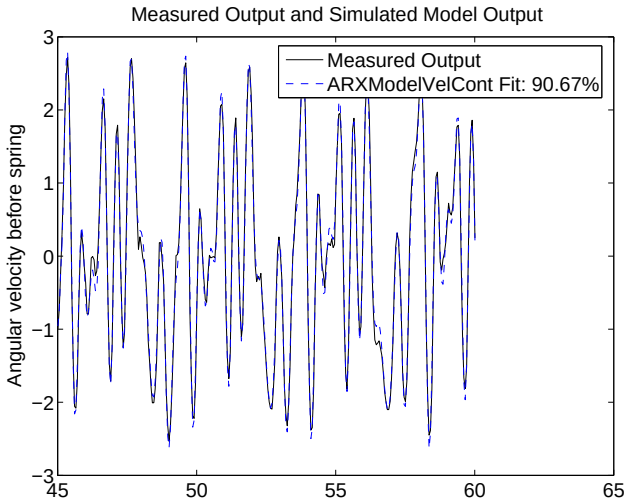


Figure 5. Validation of ARX model from voltage to angular velocity using spring with simulated output.

The actual parameter estimation is performed using the `idgrey` command in MATLAB. For each parameter value that `idgrey` needs a state-space model, the transformation that was briefly described in Section 2.1 is performed using numerical software. The selected initial values and estimated values for the unknown parameters are shown in Table 1. The initial values where selected from physical insight, except for k which was estimated from the ARX models, see (11), and R which was measured.

Parameter	Initial value	Estimated value
d (Nms/rad)	$5.43 \cdot 10^{-4}$	$5.87 \cdot 10^{-4}$
c (Nm/rad)	0.0362	0.0341
J_1 (kg m^2)	$3.0 \cdot 10^{-6}$	$6.30 \cdot 10^{-6}$
J_2 (kg m^2)	$3.66 \cdot 10^{-4}$	$4.17 \cdot 10^{-4}$
R (Ω)	4.50	4.60
k (Nm/A)	-0.060	-0.0599

Table 1. Initial and estimated values for the unknown parameters.

A comparison between the initial model and the validation data and between the estimation model and the validation data is shown in Figure 6 and 7 respectively. As can be seen, already the initial parameters describe the angular velocity quite well, while the estimation process improves the fit for the angle considerably. Compared to the ARX models discussed earlier, the estimated grey box model has a somewhat worse fit for the angular velocity, and a comparable fit for the angle. The grey box model has the advantage of describing the whole system in one model.

3 State Estimation

The state estimation problem is about finding the best estimate of the system state using the information available in the measurements from the system. In our case we need the state estimates in order to design a controller, which removes as much of the oscillations in the spring as possible. From a general control theoretical perspective the use of the state estimates for controller design is quite common, and many of the modern control algorithms rely on a model of some kind.

The linear state estimation problem will be discussed in quite general terms in Section 3.1. Section 3.2 describes and solves the problems with introducing white noise in linear differential-algebraic equations. When the white noise has been introduced it is quite straightforward to derive the Kalman filter for linear DAE:s, which is done in Section 3.4. Finally the results from the experiments are given.

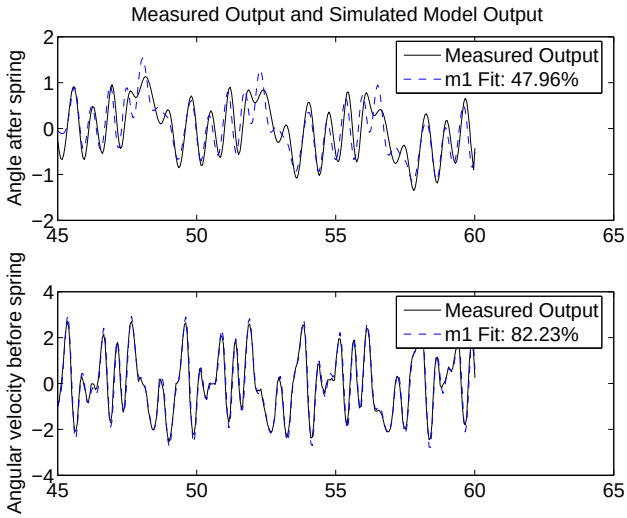


Figure 6. Validation of grey box model with initial parameter values using simulated output.

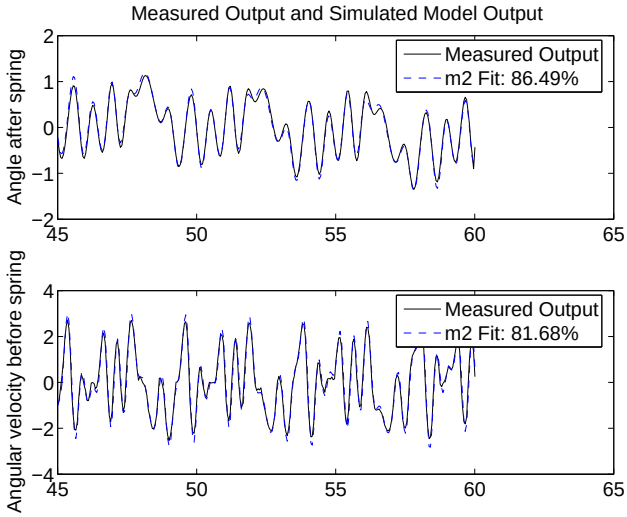


Figure 7. Validation of grey box with estimated parameter values model using simulated output.

3.1 Linear State Estimation

Since the system we are studying in this project can be modeled using linear dynamics we are only concerned with linear state estimation. The problem we are faced with in linear state estimation is to estimate the states, x, given measurements of the input, u, and output, y, signals in the following model

$$\dot{x} = Ax + Bu + w, \tag{12a}$$
$$y = Cx + e, \tag{12b}$$

where the matrices A, B, and C are given. Furthermore, the process noise w and the measurement noise e are white, Gaussian, with

$$\mathrm{E}[w(t)w^T(s)] = R_1(t)\delta(t-s), \tag{12c}$$
$$\mathrm{E}[e(t)e^T(s)] = R_2(t)\delta(t-s), \tag{12d}$$
$$\mathrm{E}[w(t)e^T(s)] = R_{12}(t)\delta(t-s). \tag{12e}$$

This problem was solved by Kalman and Luenberger in [13, 16, 17]. The idea is to use an observer which simulates the dynamics and adapts the state according to its ability to describe the output according to

$$\dot{\hat{x}} = A\hat{x} + Bu + K(y - C\hat{x}), \tag{13a}$$
$$\hat{x}(0) = \hat{x}_0. \tag{13b}$$

Hence, the problem boils down to finding the K-matrix that gives the best estimates. By forming the error dynamics ($\tilde{x} = x - \hat{x}$)

$$\dot{\tilde{x}} = (A - KC)\tilde{x} + w - Ke \tag{14}$$

we can conclude that the choice of the K-matrix constitutes a compromise between convergence speed and sensitivity to disturbances. The speed of convergence is given by the locations of the eigenvalues of the matrix $(A - KC)$. If system (12) is observable we can chose arbitrary locations for these eigenvalues, by assigning a certain K-matrix. Of course we would like the convergence to be as fast as possible, which implies large entries in the K-matrix. However, the downside of having such a K-matrix is that the measurement noise is amplified with the K-matrix. Hence, a K-matrix with large entries implies that the measurement noise will give a significant influence on the estimate. This is of course undesirable, since we want to minimize the influence of the measurement noise on the state estimate. Hence, it is not obvious how to chose the observer gain K. Either we can use trial and error by placing the eigenvalues for the error dynamics and checking the influence of the measurement noise, or we can formulate an optimization problem of some kind. In this way the estimates will be optimal in the sense of the optimization problem. One possible choice

is to find the estimates that minimizes the variance of the estimation error. This results in the following K-matrix

$$K = PC^T R_2^{-1}, \tag{15}$$

where P is given by the following Ricatti equation

$$\dot{P} = AP + PA^T + R_1 - PC^T R_2^{-1} CP \tag{16a}$$
$$P(0) = P_0 \tag{16b}$$

An observer using the K-matrix given in (15) is referred to as a Kalman filter [13]. The Kalman filter can also be motivated from a purely deterministic point of view, as the solution to a certain weighted least-squares problem [22, 20].

The assumption that the observer has the form (13a) might seem ad hoc. However, it can in fact be proven that this structure arises from the deterministic weighted least-squares formulation [18], from the maximum a posteriori formulation [20], etc. Furthermore, it can be proven that if the system is linear, subject to non-Gaussian noise the Kalman filter provides the best linear unbiased estimator (BLUE) [1].

The discussion has until now been purely in continuous time, but since the measurements from the real world are inherently discrete we have to consider the discrete time Kalman filter.

The problem is that the model obtained from Modelica is not in the form (12), but rather it is in the form

$$E\dot{x} = Ax + Bu, \tag{17}$$

where the E-matrix can be singular. A model of this type is referred to as a linear differential-algebraic equation (DAE). Other common names for this type of equation are implicit systems, descriptor systems, semi-state systems, generalized systems, and differential equations on a manifold [4]. If the singularity assumption of the E-matrix is dropped we can multiply with E^{-1} from the left and obtain an ordinary differential equation (ODE), and standard Kalman filtering theory applies.

Our aim is to be able to estimate the internal variables, x, from (17) directly, without transforming the equation into a standard state-space form (12). The reason is that we want a theory that is directly applicable to the models generated by Modelica, so that as much as possible can be done automatically. Furthermore, this allows us to add noise where it makes physical sense, simply by inserting it at appropriate places in the model. In order to accomplish this Modelica has to be extended with some kind of interaction mode and the possibility of generating random variables (noise). With this capabilities we could model and simulate stochastic, as well as, deterministic models in Modelica.

The problem is that in the general case we cannot add noise to all equations in (17). The reason is that derivatives of white noise might occur, which do not exist. This issue will be discussed in more detail later. Hence, we have to find a suitable subspace where the noise can live. More specifically we have to find all the possible B_w-matrices in

$$E\dot{x} = Ax + Bu + B_w w_t, \tag{18}$$

where w_t is a white noise sequence (i.e., B_w-matrices that assures that white noise is not differentiated). The two main reasons for why we want to introduce white noise in (18) are:

- There are un-modeled dynamics and disturbances acting on the system, that can only be introduced as an unknown stochastic term.
- There is a practical need for tuning in the filter in order to make the trade-off between tracking ability and sensor noise attenuation. This is in the Kalman filter accomplished by keeping the sensor noise covariance matrix constant and tuning the process noise covariance matrix, or the other way around. Often, it is easier to describe the sensor noise in a stochastic setting, and then it is more natural to tune the process noise.

The problem of finding the subspace where the noise can live was solved in [21] and is briefly discussed in the subsequent section.

3.2 Noise Modeling

We will omit the deterministic input in this derivation for notational convenience, so the continuous time linear invariant differential-algebraic equations considered is

$$E\dot{x}(t) + Fx(t) = Bw(t) \tag{19a}$$
$$y(t) = Cx(t) + e(t) \tag{19b}$$

The reader is referred to [10] for details on how the non-causality with respect to the input signal, $u(t)$, can be handled. For the purpose of this discussion we will assume that w and e are continuous time white noises. (See e.g., [2] for a thorough treatment of continuous time white noise). If $\det(Es + F)$ is not identically zero as a function of $s \in \mathbb{R}$, (19) can always be transformed into the *standard form* (21) (see [3, 5]). Note that if $\det(Es + F)$ is identically zero, then $x(t)$ is not uniquely determined by $w(t)$ and the initial value $x(0)$. This can be realized by Laplace transforming (19). Therefore it is a reasonable assumption that $\det(Es + F)$ is not identically zero.

First, a transformation to the standard form is needed. This is done by finding a suitable change of variables $x = Qz$ and a matrix P to multiply (19a) from the left. Both P and Q are non-singular matrices. By doing this we get

$$PEQ\dot{z}(t) + PFQz(t) = PBw(t), \tag{20}$$

which for suitably chosen P- and Q-matrices can be written in the following standard form:

$$\begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix} \begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} + \begin{bmatrix} -A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} w(t), \tag{21}$$

where the N-matrix is *nilpotent*, i.e., $N^k = 0$ for some k. The matrices P and Q can be calculated using, e.g., ideas from [24] involving the generalized real Schur form and the generalized Sylvester equation, the details can be found in [9]. We can also write (21) on the form (22), see e.g., [6] or [15].

$$\dot{z}_1(t) = Az_1(t) + G_1 w(t), \tag{22a}$$

$$z_2(t) = \sum_{i=0}^{k-1} (-N)^i G_2 \frac{d^i w(t)}{dt^i}. \tag{22b}$$

From a theoretical point of view G_1 can be chosen arbitrarily, since it describes how white noise should enter an ordinary differential equation. However, constraints on G_1 can of course be imposed by the physics of the system that is modeled. When it comes to G_2, the situation is different, here we have to find a suitable parameterization. The problem is now that white noise cannot be differentiated, so we proceed to find a condition on the B-matrix in (19a) under which there does not occur any derivatives in (22b), i.e., $N^i G_2 = 0$ for all $i \geq 1$. This is equivalent to

$$NG_2 = 0. \tag{23}$$

The result is given in Theorem 3.1. To formulate the theorem, we need to consider the transformation (20) with matrices P and Q which gives a system in the form (21). Let the matrix N have the singular value decomposition

$$N = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} V^T = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^T, \tag{24}$$

where V_2 contains the last k columns of V having zero singular values. Finally, define the matrix M as

$$M = P^{-1} \begin{bmatrix} I & 0 \\ 0 & V_2 \end{bmatrix}. \tag{25}$$

It is now possible to derive a condition on B.

Theorem 3.1 *The condition* (23) *is equivalent to*

$$B \in \mathcal{R}(M) \tag{26}$$

where B is defined in (19) *and M is defined in* (25)*.*

The expression (26) means that B is in the *range* of M, that is the columns of B are linear combinations of the columns of M.

Proof 3.1 *From the discussion above we know that there exist matrices P and Q such that*

$$PEQQ^{-1}\dot{\xi}(t) = PJQQ^{-1}\xi(t) + PBw(t) \tag{27}$$

gives the canonical form

$$\begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix} Q^{-1}\dot{\xi}(t) + \begin{bmatrix} -A & 0 \\ 0 & I \end{bmatrix} Q^{-1}\xi(t) = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} w(t). \tag{28}$$

Note that B can be written as

$$B = P^{-1} \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}. \tag{29}$$

Let the matrix N have the singular value decomposition

$$N = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} V^T \tag{30}$$

where Σ is a diagonal matrix with nonzero elements. Since N is nilpotent it is also singular, so k singular values are zero. Partition V as

$$V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}, \tag{31}$$

where V_2 contains the last k columns of V having zero singular values. Then $NV_2 = 0$.

We first prove the implication (26) $\Rightarrow$ (23)*: Assume that* (26) *is fulfilled. B can then be written as*

$$B = M \begin{bmatrix} S \\ T \end{bmatrix} = P^{-1} \begin{bmatrix} I & 0 \\ 0 & V_2 \end{bmatrix} \begin{bmatrix} S \\ T \end{bmatrix} = P^{-1} \begin{bmatrix} S \\ V_2 T \end{bmatrix} \tag{32}$$

for some matrices S and T. Comparing with (29)*, we see that $G_1 = S$ and $G_2 = V_2T$. This gives*

$$NG_2 = NV_2T = 0 \tag{33}$$

so (23) *is fulfilled.*

Now the implication (23) $\Rightarrow$ (26) *is proved: Assume that* (23) *is fulfilled. We then get*

$$0 = NG_2 = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} G_2 = U \begin{bmatrix} \Sigma V_1^T G_2 \\ 0 \end{bmatrix}. \tag{34}$$

This gives that

$$V_1^T G_2 = 0, \tag{35}$$

so the columns of G_2 are orthogonal to the columns of V_1, and G_2 can be written as

$$G_2 = V_2 T. \tag{36}$$

Equation (29) *now gives*

$$\begin{aligned} B &= P^{-1} \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = P^{-1} \begin{bmatrix} G_1 \\ V_2 T \end{bmatrix} = P^{-1} \begin{bmatrix} I & 0 \\ 0 & V_2 \end{bmatrix} \begin{bmatrix} G_1 \\ T \end{bmatrix} \\ &= M \begin{bmatrix} G_1 \\ T \end{bmatrix} \in \mathcal{R}(M). \end{aligned} \tag{37}$$

(26) *is fulfilled.*

The B-matrices satisfying (29) will thus allow us to incorporate white noise without having a problem with differentiation of white noise. The design parameters to be specified are G_1 and T defined in the proof above. Also note that the requirement that the white noise should not be differentiated is related to the concept of *impulse controllability* discussed in [6].

3.2.1 Constructing the Process Noise Subspace

The proof for Theorem 3.1 is constructive. Hence, it can be used to derive the process noise subspace for the laboratory example treated in this work. Forming the singular value decomposition for the 50×50 matrix N in (21) we obtain two nonzero singular values (1.41 and 1.00). Hence, $V_2 \in \mathbb{R}^{50\times 48}$. We can parameterize all matrices G_2, satisfying

$$NG_2 = 0, \tag{38}$$

using an arbitrary matrix $T \in \mathbb{R}^{48\times 48}$, according to $G_2 = V_2 T$. Furthermore we can, using the result (25), write the M-matrix according to

$$M = P^{-1}\begin{bmatrix} I_4 & 0 \\ 0 & V_2 \end{bmatrix}, \tag{39}$$

where I_4 is a four dimensional identity matrix. The subspace where the noise can live is now given by $\mathcal{R}(M)$ according to Theorem 3.1. Ultimately we want the user to be able to, directly in Modelica, specify where in the model the noise should enter. For instance, if a certain part of the model corresponds to a significant simplification of the true system we would like to add noise there in order to mathematically account for the fact that we are uncertain about the dynamics at this part. Another example might be that we are uncertain about a parameter value, and model this uncertainty by adding noise to the corresponding equation. If the user were allowed to add noise to all equations, some of this noise would inevitably end up in the forbidden noise subspace, i.e., the subspace where $B \notin \mathcal{R}(M)$. Hence, we want to find the equations to which we cannot add noise. We will do this by studying a vector $b \in \mathbb{R}^{54}$, with only one nonzero element. Think of the case where we only have one noise source, $B = b$. It is not a restriction to limit the treatment to this type of B-matrices, since generally we cannot physically motivate the use of the same noise at two places in the model. That would correspond to using exactly the same disturbance at for instance the resistor and the spring in our model. If we want noise at both these places we would add two separate noise sources, which of course could have the same statistical properties.

For Theorem 3.1 to be valid the b-vectors constituting the B-matrix cannot have any contribution in $\mathcal{N}(M^T)$, which is the orthogonal complement to the range space of M, $\mathcal{R}(M)$. By projecting a b-vector onto $\mathcal{N}(M^T)$ using the following projection matrix

$$I - MM^\dagger \tag{40}$$

we can check whether the corresponding vector can be part of the B-matrix or not. Hence, b-vectors with only one nonzero entry cannot be part of the B-matrix if

$$(I - MM^\dagger)b \neq 0. \tag{41}$$

Applying (41) to our model implies that we cannot add noise to 6 of the 54 equations. These 6 equations involve angles of various kinds. It is quite natural that we are not allowed to add white noise to angles, since that would correspond to an infinite jump in the corresponding derivative, which we cannot have. Instead the noise have to be added to the derivative and then integrated to affect the angle.

3.3 Discrete Model

There are several reasons to derive a discrete model, such as

- The measurements from systems are obtained in discrete time.
- A common use of the estimated states are for model based control. These controllers are implemented using computers, which implies that the control signal will inherently be discrete.

If the noise enters the system according to a B-matrix satisfying Theorem 3.1 the original system (19) can be written as

$$\dot{z}_1(t) = Az_1(t) + G_1 w(t), \tag{42a}$$
$$z_2(t) = G_2 w(t), \tag{42b}$$
$$y(t) = CQz(t) + e(t). \tag{42c}$$

where $x = Qz$. Furthermore $w(t)$ and $e(t)$ are both assumed to be Gaussian white noise signals with covariances R_1 and R_2 respectively, and zero cross-covariance (the case of nonzero cross-covariance can be handled as well, the only difference is that the expressions are more involved).

The system (42) can be discretized using standard techniques from linear systems theory, see e.g., [19]. If we assume that $w(t)$ remains constant during one sample interval[1], we have (here it is assumed that sampling interval is one to simplify the notation)

$$w(t) = w[k], \qquad k \leq t < (k+1) \tag{43}$$

[1] See e.g., [11] for a discussion on other possible assumptions on the stochastic process $w(t)$ when it comes to discretization.

we obtain

$$z_1[k+1] = \tilde{A}z_1[k] + \tilde{G}_1 w[k], \tag{44a}$$
$$z_2[k] = G_2 w[k], \tag{44b}$$
$$y[k] = CQz[k] + e[k] \tag{44c}$$

where

$$\tilde{A} = e^A \qquad \tilde{G}_1 = \int_0^1 e^{A\tau} d\tau G_1. \tag{45}$$

Hence, (44) and (45) constitutes a discrete time model of (19).

3.4 The Kalman Filter

In order to apply the Kalman filter to the discrete time model (44) we start out by rewriting (44c) as

$$\begin{aligned} y[k] &= CQz[k] + e[k] = [\tilde{C}_1 \tilde{C}_2] \begin{bmatrix} z_1[k] \\ z_2[k] \end{bmatrix} + e[k] \\ &= \tilde{C}_1 z_1[k] + \tilde{C}_2 z_2[k] + e[k] \\ &= \tilde{C}_1 z_1[k] + \underbrace{\tilde{C}_2 G_2 w[k] + e[k]}_{\tilde{e}[k]} \end{aligned} \tag{46}$$

Combining (44a) and (46) we obtain

$$z_1[k+1] = \tilde{A}z_1[k] + \tilde{G}_1 w[k] \tag{47a}$$
$$y[k] = \tilde{C}_1 z_1[k] + \tilde{e}[k] \tag{47b}$$

Note that the measurement noise, $\tilde{e}[k]$, and the process noise, $w[k]$, are correlated, if $\tilde{C}_2 G_2 \neq 0$. Now, the Kalman filter can be applied to (47) in order to estimate the internal variables $z_1[k]$ and the process noise $w[k]$. Finally an estimate of the internal variables $z_2[k]$ can be found using the estimated process noise, since $z_2[k] = G_2 w[k]$, according to (44b). Finally the internal variables, $x[k]$, are obtained by $x[k] = Q^{-1}z[k]$. For details regarding the Kalman filter see e.g., [12, 11, 1]. Since we did not find any good references on how to derive the estimate of the process noise we include a derivation of this in the subsequent section.

3.4.1 Estimating the Process Noise

The problem is how to estimate the process noise w_t given the innovations, ν_i, $i \leq t$. We will use the following theorem from [12, p. 81]:

Theorem 3.2 *Given two zero mean random variables x and y, the linear least-mean-squares estimator $\hat{x} = K_0 y$ of x given y is given by any solution to the so-called normal equations*

$$K_0 R_y = R_{xy} \tag{48}$$

where $R_y = \mathrm{E}(yy^T)$ and $R_{xy} = \mathrm{E}(xy^T)$.

Now back to our problem. Estimating w_t given ν_i, $i \leq t$ is equivalent to estimating w_t given ν_t since both processes are white. According to Theorem 3.2, our estimator $\hat{w}_{t|t} = K_0 \nu_t$ is given by the solution to

$$K_0 R_{\nu_t} = R_{w_t \nu_t} \tag{49}$$

From the Kalman filter equations, we have

$$R_{\nu_t} = C_t P_{t|t-1} C_t^T + R_t. \tag{50}$$

Furthermore,

$$R_{w_t \nu_t} = E(w_t \nu_t^T) = E(w_t(y_t - C_t \hat{x}_{t|t-1})). \tag{51}$$

Since w_t is independent of $\hat{x}_{t|t-1}$ we get

$$R_{w_t \nu_t} = E(w_t y_t^T) = R_{12} \tag{52}$$

Finally, this gives

$$K_0 = R_{12}(C_t P_{t|t-1} C_t^T + R_t)^{-1} \tag{53}$$

and thus

$$\hat{w}_{t|t} = R_{12}(C_t P_{t|t-1} C_t^T + R_t)^{-1} \nu_t. \tag{54}$$

3.4.2 Experiments

The theory derived in the previous sections will now be tested using real data from the spring servo system studied throughout this work. Both the time-varying and the time-invariant (stationary) Kalman filter will be used to estimate the states. We have access to two output signals from our process, the angular velocity of the motor and the angular position of the mass after the spring. In order to validate our theory we will use the angular position as output signal and use the information available in this signal to estimate the other states. The estimated motor angular velocity will be compared with the true (measured) motor angular velocity in order to validate the estimation algorithm.

The following noise covariances were used

$$R_1 = 0.1 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R_2 = 0.1 \tag{55}$$

In Figure 8 the estimation performance is showed using the stationary and the time-varying Kalman filter with correct initial conditions (zero). From this figure is it clear that there is nothing to gain in using the time-varying Kalman filter in this case, which is due to the fact that the time-varying Kalman filter will coincide with the stationary Kalman filter directly. However, it we does not have access to the correct initial conditions, which we typically do not have, the result will be different (see Figure 9). In the transient phase the time-varying Kalman filter will be superior to the stationary Kalman filter. When the transient has disappeared the time-varying and the stationary Kalman filter both yield the same result, see Figure 10.

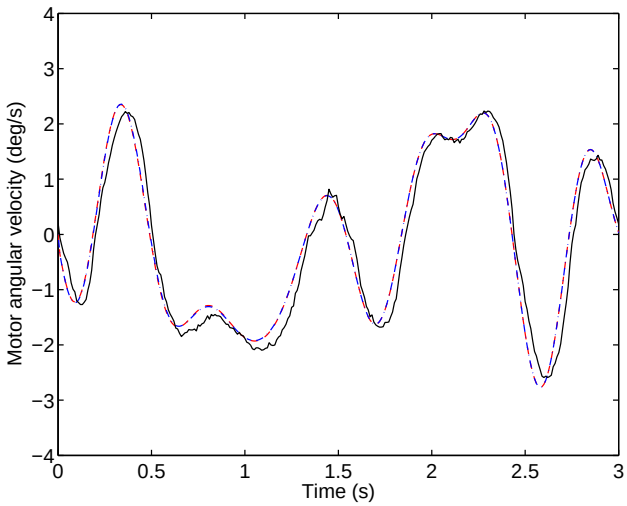


Figure 8. Transient behavior of the motor velocity estimates with zero initial states. The red (dashed), and the blue (dash-dotted) curve shows the estimates from the time-invariant and the time-varying Kalman filters respectively. The measured motor velocity is the black (solid) curve.

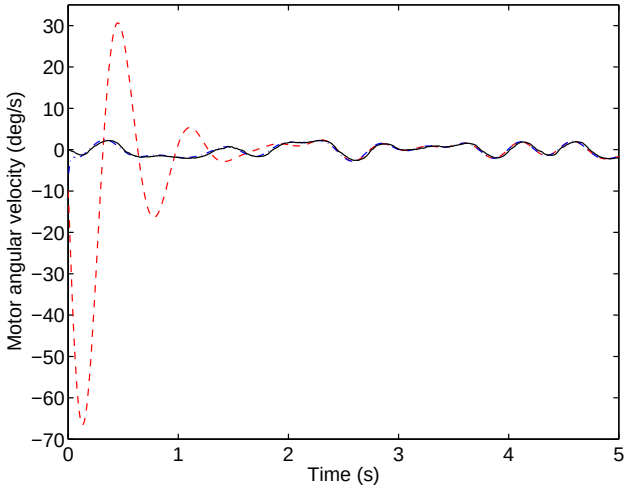


Figure 9. Transient behavior of the motor velocity estimates with nonzero initial states. The red (dashed), and the blue (dash-dotted) curve shows the estimates from the time-invariant and the time-varying Kalman filters respectively. The measured motor velocity is the black (solid) curve.

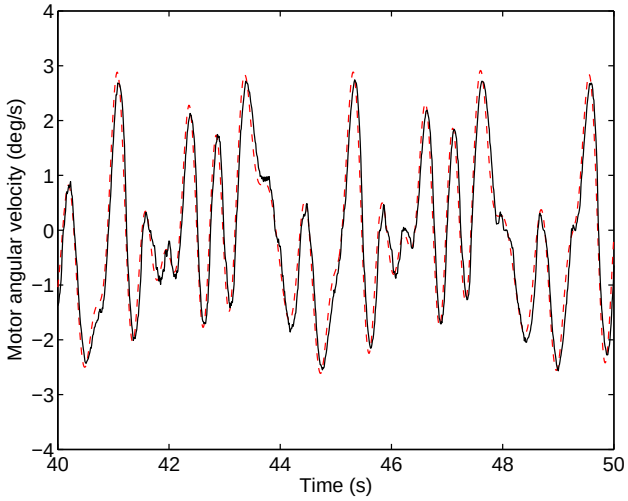


Figure 10. Stationary performance. The measured motor velocity is the black (solid) and the red (dashed) curve is the estimate from the Kalman filter.

4 Concluding Remarks

In this work we have applied theory previously developed by the authors [21, 10] to perform system identification and state estimation using differential-algebraic equations (DAE:s). The system was modeled in Modelica, the resulting equations were then transfered to MATLAB where the unknown parameters were estimated. Using this estimated model the states could then be successfully estimated.

4.1 Further Work

There are several ideas for further work, some of the most important directions are:

- Automatic translation of the Modelica model into the form

$$E(\theta)\dot{\xi}(t) = J(\theta)\xi(t) + K(\theta)e(t), \tag{56a}$$

$$y(t) = C(\theta)\xi(t), \tag{56b}$$

  in MATLAB. This should be fairly straightforward if it was possible to specify inputs, outputs, and unknown parameters in Modelica. If this translation existed we could perform grey-box system identification by first modeling the system in Modelica and then translate the model into the form (56), where the matrices are readily imported in MATLAB. The unknown parameters can then be estimated without having to manually manipulate the equations.

- Simulation of noise in Modelica. In this way it would be possible to model disturbances as well. It would be necessary to include some kind of interface in Modelica where we can check where it is possible to include the noise using Theorem 3.1.
- Investigate to what extent these results could be extended to nonlinear systems. The particle filter [7] can probably be useful in the process of estimating the states in a nonlinear DAE.

References

[1] B. Anderson and J. Moore. *Optimal Filtering*. Information and system science series. Prentice Hall, Englewood Cliffs, New Jersey, 1979.

[2] K. Åström. *Introduction to Stochastic Control Theory*. Mathematics in Science and Engineering. Academic Press, New York and London, 1970.

[3] K. Brenan, S. Campbell, and L. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. SIAM's Classics in Applied Methematics. SIAM, New York, 1996.

[4] S. Campbell. Descriptor systems in the 90's. In *proceedings of the 29th Conference on Decision and control*, pages 442–447, Honolulu, Hawaii, USA, December 1990.

[5] L. Dai. Filtering and LQG problems for discrete-time stochastic singular systems. *IEEE Transactions on Automatic Control*, 34(10):1105–1108, Oct. 1989.

[6] L. Dai. *Singular Control Systems*. Lecture Notes in Control and Information Sciences. Springer-Verlag, Berlin, New York, 1989.

[7] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer Verlag, 2001.

[8] P. Fritzson. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. Wiley-IEEE, New York, 2004.

[9] M. Gerdin. Parameter estimation in linear descriptor systems. Licentiate Thesis No 1085, Linköping University, 2004.

[10] M. Gerdin, T. Glad, and L. Ljung. Parameter estimation in linear differential-algebraic equations. In *proceedings of the 13th IFAC Symposium on System Identification*, Rotterdam, The Netherlands, Aug. 2003. IFAC.

[11] F. Gustafsson. *Adaptive Filtering and Change Detection*. John Wiley & Sons, 2000.

[12] T. Kailath, A. Sayed, and B. Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall, Upper Saddle River, New Jersey, 2000.

[13] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. AMSE, J. Basic Engineering*, 82:35–45, 1960.

[14] L. Ljung. *System Identification - Theory for the User*. Information and System Sciences Series. Prentice Hall PTR, Upper Saddle River, N.J., 2. edition, 1999.

[15] L. Ljung and T. Glad. *Modellygge och simulering*. Studentlitteratur, 2003. In Swedish.

[16] D. Luenberger. Observers for multivariable systems. *IEEE Transactions on Automatic Control*, AC-11(2):190–197, Apr. 1966.

[17] D. Luenberger. An introduction to observers. *IEEE Transactions on Automatic Control*, AC-16(6):596–602, Dec. 1971.

[18] C. Rao. *Moving Horizon Strategies for the Constrained Monitoring and Control of Nonlinear Discrete-Time Systems*. PhD thesis, University of Wisconsin Madison, 2000.

[19] W. Rugh. *Linear System Theory*. Information and system sciences series. Prentice Hall, Upper Saddle River, New Jersey, 2 edition, 1996.

[20] T. Schön. *On Computational Methods for Nonlinear Estimation*. Licentiate thesis, Linköping University, Oct. 2003. Thesis No. 1047.

[21] T. Schön, M. Gerdin, T. Glad, and F. Gustafsson. A modeling and filtering framework for linear differential-algebraic equations. In *proceedings of the 42nd Conference on Decision and Control*, Maui, Hawaii, USA, Dec. 2003.

[22] H. Sorenson. Least-squares estimation: from Gauss to Kalman. *IEEE Spectrum*, 7:63–68, July 1970.

[23] M. Tiller. *Introduction to Physical Modeling with Modelica*. Kluwer, Boston, Mass., 2001.

[24] A. Varga. Numerical algorithms and software tools for analysis and modelling of descriptor systems. In *proceedings of 2nd IFAC Workshop on System Structure and Control, Prague, Czechoslovakia*, pages 392–395, 1992.