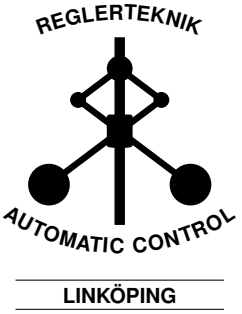
Computer Exercises in System Identification
Part 3

This version: 2023-08-28



1 Parametric Identification of State-space Models

Until now, we have worked with models

$$y(t) = G(q,\theta)u(t) + H(q,\theta)e(t), \tag{1}$$

where the transfer functions $G(q,\theta)$ and $H(q,\theta)$ are rational, and the parameters are the coefficients of the numerator and denominator polynomials. Another common model structure is state-space models

$$x(t+1) = A(\theta)x(t) + B(\theta)u(t) + K(\theta)e(t), \tag{2a}$$
$$y(t) = C(\theta)x(t) + D(\theta)u(t) + e(t), \tag{2b}$$

(for simplicity, written in innovation form, see [3, p. 99] for more information). In this description, $x(t)$ is the state vector, $A(\theta)$, $B(\theta)$, $K(\theta)$, $C(\theta)$ and $D(\theta)$ are matrices of conformable dimensions, and θ is the parameter describing unknown elements in the matrices. The state-space model (2) is just another way to write the linear model (1). By using the time shift operator q, (2) can be written in the form (1) through

$$G(q,\theta) = C(\theta)(qI - A(\theta))^{-1}B(\theta) + D(\theta), \tag{3a}$$
$$H(q,\theta) = C(\theta)(qI - A(\theta))^{-1}K(\theta) + I. \tag{3b}$$

The predictor $\hat{y}(t|\theta)$ is given by

$$\begin{aligned}\hat{x}(t+1,\theta) =& A(\theta)\hat{x}(t,\theta) + B(\theta)u(t) + \\ &+ K(\theta)\left[y(t) - C(\theta)\hat{x}(t,\theta) - D(\theta)u(t)\right],\end{aligned} \tag{4a}$$
$$\hat{y}(t|\theta) = C(\theta)\hat{x}(t,\theta) + D(\theta)u(t). \tag{4b}$$

1.1 Black-box State-space Models

State-space models are equally flexible as ARMAX models. This means that the system and noise models share poles, but have different zeros. If we fix $K = 0$, we instead get an OE model. An advantage of state-space models is that multivariate systems (systems with several inputs and outputs) are easy to model. For a state-space model with n_x states and one input and output, there are in total $n_x^2 + 3n_x + 1$ parameters (the $n_x \times n_x$ matrix $A(\theta)$,

the vectors $B(\theta), C(\theta), K(\theta)$ and the scalar $D(\theta)$). All these parameters are not required to describe a system; it is enough to use as many parameters as in an ARMAX model. There are thus many degrees of freedom in the model parameterization. For example, it is enough to put the coefficients of the denominator of G and H in the first column of $A(\theta)$, and to put the coefficients of the numerator of G and K into $B(\theta)$ and $K(\theta)$. Other elements should contain ones and zeroes (compare observable canonical form, [1]). The parameterization is handled automatically in SITB, so the only thing to be remembered is that different matrices may give the exact same input output description.

In the GUI of SITB, state-space models are estimated by clicking `Estimate` and then `State Space Models...`. For state-space models, choose the number of parameters `Model Order`. Order selection is also available by clicking `Pick best value in the range`. Under `Delay`, the input delay can be chosen by entering it in `Input Delay`, and you can also choose if you want to estimate K (equivalent to ARMAX) or set it to zero (equivalent to OE) by checking or unchecking `Include disturbance component (K)`.

To estimate state-space models, two main types of methods exist:

- Prediction error methods (PEM)
- Subspace methods (N4SID)

Prediction error methods are often most accurate, but suffer from getting stuck at local maxima in the loss function $V(\theta)$. The estimates may sometimes be very bad if the minimization is not started close enough to the right model.

Subspace methods consist of two steps. The first one is to estimate the state vector $x(t)$ (or the observability matrix) by projections into certain subspaces formed from the data. When this state vector is estimated, the matrices in the state description can be computed by solving a linear least squares problem (like ARX). The first step is also a least squares problem, and thus there are no problems with local minima. Subspace methods are based on approximations, however, since in reality an infinite amount of data is needed to estimate the state vector (or the observability matrix) accurately. The best combination is to start with a subspace method and then improve

the model through prediction error minimization (this is done automatically when estimating a state-space model using PEM in SITB).

In `Estimation Options`, more settings can be changed. In particular the parameter `N4Horizon` may need to be changed to get better estimates with N4SID. The parameter sets the prediction horizon (and the number of old y and u which is used for prediction) in the first stage of the subspace method. Sometimes (in particular for oscillatory systems), this parameter might need to be increased in order to get good estimates. The exact value depends on the problem, but try something between 10 and 200.

Exercises for Section 1.1

1. Let us return to exercise 3 from the nonparametric identification session: the vibration analysis data. The file `vibrationdata.mat` (run `load vibrationdata`) contains the `iddata` objects `zh` (impulse hammer) and `zs` (shaker). Furthermore, the true system `Gd` is included, which can be used in order to verify the estimated models.

   (a) Import the data set `zs`, preprocess it, and split it into estimation and validation data.

   (b) First estimate an ARX model, for example by `Order Selection`. First look at `Model output`. Also compare the frequency response `Frequency resp` with the best spectral model from before (you can also import the true system `Gd` through `Import models` for comparison).

   (c) How many poles are required to describe every resonance in the system?

   (d) Now try to estimate a more advanced model to improve model fit. For instance, choose an OE model (which in fact gives an accurate input-output description even if the noise is not white) with orders chosen from (c). For simplicity set $n_f = n_b$ and $n_k = 1$. Is the OE model an improvement?

   (e) Now try to estimate a state-space model with order chosen from (c). For a state-space OE model, uncheck `Include disturbance component (K)` in `Model Structure`. Try changing `N4Horizon` in `Estimation Options` until you get a good estimate.

   (f) Model order can also be studied using `Pick best value in the`

`range`. Try this with `N4Horizon = 200`. Is this consistent with what you found in (c)?

Hint: The meaning of `Log of Singular Values` in the Y-axis of the model order plot can be seen as a measure of how much the additional state affects the input-output description. A small value means that the state can be removed with small effect on the input-output description.

Note

- The system in exercise 1 is highly oscillatory. This is very demanding for identification methods.
- While OE and state-space with $K = 0$ yield the same input-output description, the results may differ because of the different estimation methods for the models.
- ARX and N4SID do not suffer from getting stuck at local minima for the loss function $V(\theta)$, but other methods require good starting guesses of the parameters to avoid local minima. This is especially sensitive for oscillatory systems (a resonance peak at the wrong frequency can yield a worse model than no resonance peak at all).

2. (a) Let us return to the electric motor with load from the previous sessions. Load the data set from the file `elmotor.mat`, with the vector u and y which contain 1000 data points of the input and output signals, which have been collected with sample time 0.3 s. Preprocess and split the data into estimation and validation data (for instance by using `Quick start`, but make sure to check that the output is reasonable!).

   (b) Estimate a few state-space models using `Pick best value in the range`. Compare the result with the models you got from earlier sessions. Try both with and without a noise model (check or uncheck `Include disturbance component (K)`) and compare.

3. Bonus: Do exercises 9.13 (chemical reactor) and 9.14 (evaporator) in exercise book to try estimating state-space models for multi-input and multi-output signals.

1.2 Physically Parameterized State-space Models

In many cases, it is interesting or even necessary to estimate models parameterized using physical parameters ("grey-box models").

Physically parameterized continuous-time state-space models have both advantages and disadvantages in comparison to black-box models. This section shows how to estimate a physically parameterized model from data and compares this approach with traditional black-box models.

We now illustrate how physical parameters can be estimated in SITB by an example. A model for a DC motor [1, p. 18] can be described by the transfer function

$$G(s) = \frac{k}{s(\tau s + 1)}. \tag{5}$$

If we instead write the model in state-space form, we get

$$\begin{aligned} \dot{x}(t) &= \begin{pmatrix} 0 & 1 \\ 0 & -1/\tau \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ k/\tau \end{pmatrix} u(t), \\ y(t) &= \begin{pmatrix} 1 & 0 \end{pmatrix} x(t). \end{aligned} \tag{6}$$

To estimate the system we introduce the unknown parameters

$$\theta_1 = \tau, \quad \theta_2 = k, \quad \theta_3 = x_1(0).$$

We make an initial guess for these parameters as

$$\theta_1 = 0.1, \quad \theta_2 = 1, \quad \theta_3 = 0.$$

The structure of the model is defined in an m-file (in this example, `dcmotor.m`) as

```
function [A,B,C,D,K,x0]=dcmotor(par,T,aux)
  A=[0 1; 0 -1/par(1)];
  B=[0; par(2)/par(1)];
  C=[1 0];
  D=0;
  K=[0; 0];
  x0=[par(3); 0];
end
```

where `T` is the sample time and `aux` are other auxiliary arguments. In the simplest case with a continuous-time state-space model, these arguments need not be used (write `help idgrey` for more information). In the identification, this function is used to compute the system matrices `A`, `B`, `C`, `D` and `K` and the initial state`x0` for a given value of the parameter vector `par`. To identify the model, we must now create a model object as

```
m0=idgrey('dcmotor',[0.1 1 0],'c');
```

The second argument `[0.1 1 0]` is the initial guess of the parameter vector `par` $= \theta$, and `'c'` means that the state-space model is continuous. The model parameters can now be estimated in the GUI by clicking `Estimate` and then `Refine Existing Models`. This yields the dialogue shown in Figure 1. The model object `m0` is entered in `Initial Model`. The model can then be estimated by clicking `Estimate`.

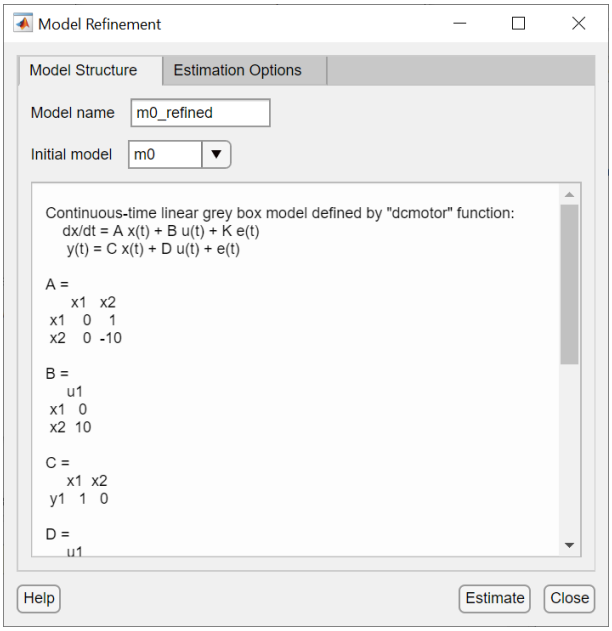


Figure 1: Prompt for estimation of physical models.

Note

- You don't have to introduce parameters for the initial state `x0` (where `par(3)` is used in the example) if you don't want to. Instead, you can choose `Estimate` in `Initial conditions` under `Estimation Options` in Figure 1. If the initial states are left fixed, the parameter estimates may be poor.
- The choice of initial parameter guess can be tricky and usually requires some physical knowledge of the system to be modeled. This is a disadvantage of physically parameterized models. One way to get a reasonable initial guess can be to try some values and compare with a spectral model or a black-box model. If there are resonance peaks in the system, the parameters should have values such that at least some resonance peak appears in the interval between 0 and $\omega_s/2$ (the Nyquist frequency). When parameters are changed, the Bode diagram should also change in this interval; otherwise the parameters will not be identifiable.

Exercises for Section 1.2

4. Again consider the electric motor (`load elmotor`). A simplified physical model for the electric motor is given by

$$\begin{aligned} u &= Ri + k_m\omega_1 \\ \dot{\phi} &= \omega_1 - \omega_2 \\ J_1\dot{\omega}_1 &= k_m i - k\phi - d(\omega_1 - \omega_2) \\ J_2\dot{\omega}_2 &= k\phi + d(\omega_1 - \omega_2) \end{aligned} \tag{7}$$

   where u input voltage, k_m and R are a motor constant and resistance (the inductance is neglected), ϕ is the difference of angles between the motor and the load, ω_1 and ω_2 are the angular velocities of the motor and load, J_1 and J_2 are inertia for the motor and load, and k and d is angular spring and damper coefficients respectively.

   (a) Create a state-space model with u as input, ω_2 as output, and the state-space vector as $x = \begin{bmatrix}\phi & \omega_1 & \omega_2\end{bmatrix}^T$.

   (b) Assume that the motor parameters have known values $J_1 = 1$, $k_m = 0.468$, $R = 0.21$, but that the load is unknown. Estimate the physical parameters J_2, k, d by first creating an m-file defining the state-space model (7), and then creating an `idgrey` object like the previous example. Estimate the initial state by selecting `Estimate` in `Initial States`. As an initial guess of the parameters, choose for instance $J_2 = 1$, $k = 1$, $d = 0$.

Hint: The estimated parameters can be viewed by dragging the model icon to the area `To Workspace` and then writing `m0c.ParameterVector` in the command prompt (if the model is `m0c`). Enter `get(m0c)` to see what properties can be studied. Among others, `m0c.CovarianceMatrix` may be interesting.

(c) Compare the physical model with the earlier parametric and non-parametric models (for instance an SPA model and a state-space model). Can you see any similarities or differences?

(d) Physical models usually have fewer unknown parameters, which yields lower estimation variance. Study this by plotting confidence intervals in the various plots (choose `Show 99% confidence intervals` in the menu `Options` in every plot). Study in particular the frequency response and the poles and zeros.

(e) What are the disadvantages of physical models?

Solutions with Discussion

The solutions below are not necessarily the only correct solution. Depending on choices during data processing and estimation, your results may be perfectly correct even if they do not exactly agree with the discussion below.

1. (b) ARX with `Order selection` yields the order `[10 5 4]` as the best fit. This model is almost useless: it gives a fit to validation data of 3.89%! In the frequency domain (`Frequency resp` and `Noise spectrum`) we can see that the model mostly fits the noise spectrum. (Compare `SPAFDR` with 5000 logarithmically spaced frequencies.)

   (c) For each resonance peak we need a complex pole pair. There are 4 clearly visible resonances (and maybe another one around 45 rad/s) so we need 8 (or maybe 10) poles.

   (d) An OE model with order `[8 8 1]` has a model fit of 49.1%, which is a clear improvement. However, in the frequency response we can see that the model only has captured the resonance at 6.5 rad/s. There is more work to do.

   (e) A state-space model of order 8 (estimated using `N4SID`) gives an unstable model (with noise model) or a model fit of -0.16% (with

no noise model, $K = 0$). This doesn't appear to work at all. However, by increasing the prediction horizon `N4Horizon`, more resonance peaks are captured. With `N4Horizon`=200 and $K = 0$ we get a model fit of 98.6% and all the first 4 resonances are captured.

(f) We can see that the singular values (roughly, the input-output gain which the additional state adds) drops off after order 8. This indicates that a higher model order will not affect the input-output description much. This order was the one we found in exercise (c).

4. (a) With states $x = \begin{bmatrix} \phi & \omega_1 & \omega_2 \end{bmatrix}^T$, the matrices are:

$$A = \begin{bmatrix} 0 & 1 & -1 \\ -k/J_1 & -(d + k_m^2/R)/J_1 & d/J_1 \\ k/J_2 & d/J_2 & -d/J_2 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ k_m/(RJ_1) \\ 0 \end{bmatrix},$$

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

(b) First write an m file, for instance:

```
function[A,B,C,D,K,X0] = elmotorgrey(par,T,aux)

J1=1;  km=0.468;  R=0.21;
J2=par(1); k=par(2); d=par(3);

A = [0 1 -1; -k/J1 -(d+km^2/R)/J1 d/J1; k/J2 d/J2 -d/J2];
B = [0 km/R/J1 0]';
C = [0 0 1];
D = 0;
K = [0 0 0]';
X0 = [0 0 0]';

end
```

This function is saved in the file `elmotorgrey.m` (same name as the function). After this, create an `idgrey` object in MATLAB through

```
m0=idgrey('elmotorgrey',[1 1 0],'c');
```

The model is estimated by selecting this model in `Initial model` and clicking `Estimate`.

The initial guess model can be imported through `Import models`. This model has a model fit of 64.83%, which is fairly good. After parameter estimation, we get a better model with fit of 92.16%. The parameters are

$$\begin{bmatrix} J_2 \\ k \\ d \end{bmatrix} = \begin{bmatrix} 1.4723 \\ 0.7014 \\ -0.0496 \end{bmatrix} \pm \begin{bmatrix} 0.0125 \\ 0.0041 \\ 0.0041 \end{bmatrix}$$

where also the standard deviation is shown (`sqrt(diag(m0c.Cov))`). Note that the damper coefficient is negative. This is not physically realistic. However, the damper only marginally affects the model performance. If the damper coefficient is set to 0 in the model, a slightly reduced model fit of 91.32% is achieved.

(c) The model is compared with `SPAFDR` with 100 logarithmically spaced frequencies, and a state-space model of order 3 (the same number of states as the physically parameterized model, which also is what order selection recommends). The state-space model has a fit of 92.19% (estimated with PEM) or 91.61% (estimated using N4SID). The physical model thus has a similar result as a black-box model with the same order. The difference can be seen in the noise model (`Noise spectrum`), where the physical model fails to accurately describe the noise. This can be solved by also parameterizing K in the physical model.

(d) In the frequency response we can see that the uncertainty is larger in the black-box models. In particular, at higher frequencies the relative uncertainty is large (because the frequency response amplitude decreases with higher frequency but the uncertainty the same, compare (12.105) in [2]). For physical models, the uncertainty is smaller due to fewer parameters. It doesn't increase for higher frequencies, which may be because the frequency response of the physical model at this frequency band doesn't depend much on the estimated parameters.

For poles and zeroes, remember the difference between continuous-time and discrete-time models (unit circle or imaginary axle are stability limits, continuous-time pole at s corresponds to discrete-time pole at $z = e^{isT}$). Despite this difference, we can see that the uncertainty is significantly larger for the black-box model. In particular the black-box model zeros are uncertain.

(e) Physical models contain prior knowledge of the system in the model. Fewer parameters must then be estimated from data. This

yields lower uncertainty/variance for the estimated parameters. An obvious disadvantage is if this prior knowledge does not agree with reality. We will then get low variance, but a bias in the estimate caused by the inaccurate model assumptions. Another problem is that it may be difficult to find a suitable initial guess of the parameters. If the model is overparameterized, identifiability may also be a problem, such as when you try to estimate both α and β when only the product $\alpha \cdot \beta$ occurs in the equations.

References

[1] T. Glad and L. Ljung. *Reglerteknik. Grundläggande teori.* Studentlitteratur, 2006.

[2] L. Ljung, T. Glad and A. Hansson. *Modeling and identification of Dynamic Systems.* Studentlitteratur, 2021.

[3] L. Ljung. *System identification: Theory for the User.* Prentice Hall, Upper Saddle River, New Jersey, USA, 2nd edition, 1999.