227-0689-00L System Identification **28 November 2018**

Solution 8: Prediction error methods

Problem 1: Euler Approximation

Consider the continuous-time model given by

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-1}{\tau} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{\beta}{\tau} \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t),$$

where τ, $\beta > 0$.

Apply the bilinear transform approximation defined for a general continuous signal $h(t)$ as

$$\frac{dh}{dt} \approx \frac{h(k+T) - h(k)}{T} = \frac{2}{T}\frac{z-1}{z+1}\, h(k),$$

where $z \in \mathbb{C}$ and $T > 0$ is the sampling period to obtain an approximation of the discrete-time transfer function $G(e^{j\omega})$ in dependency of $\theta = \begin{bmatrix} \tau & \beta \end{bmatrix}^\top$. Assume that a constant input $u(t)$ is applied at times $kT \leq t \leq kT + T$.

Solution

The output of the system can be expressed in Laplace domain as

$$Y(s) = \left\{ C\,(sI - A(\theta))^{-1}\, B(\theta) \right\} U(s) = \frac{-\theta_1^2}{\theta_2 s (s\theta_1 + 1)} U(s)\,,$$

where we defined

$$A(\theta) = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-1}{\theta_1} \end{bmatrix},\ B(\theta) = \begin{bmatrix} 0 \\ \frac{\theta_1}{\theta_2} \end{bmatrix}.$$

We use the Euler formula to obtain an approximation for the sampled discrete-time dynamics by taking $s \approx \frac{2}{T}\frac{z-1}{z+1}$, $z = e^{j\omega}$ and the approximated discrete-time transfer function can be written as

$$G(e^{j\omega}) \approx \frac{-\theta_1^2 T^2 (e^{j\omega} + 1)^2}{2\theta_2 (e^{j\omega} - 1)\,(e^{j\omega}(T + 2\theta_1) + T - 2\theta_1)}.$$

Problem 2: Optimal predictor for a first-order ARMAX model

Consider the model

$$y(k) + ay(k-1) = bu(k-1) + e(k) + ce(k-1),$$

where $|c| < 1$ and $e(k) \sim \mathcal{N}(0, \lambda^2)$. The parameter vector is given by $\theta = \begin{bmatrix} a & b & c \end{bmatrix}^\top$.

1. Show that an optimal predictor $\hat{y}(t|t-1,\theta)$ which minimizes the prediction error variance σ is given by

$$\hat{y}(k|k-1,\theta) = -ay(k-1) + bu(k-1) + ce(k-1).$$

2. What difficulties can be encountered when implementing the optimal predictor? How can this be avoided?

3. Find a practical recursive implementation as a function of the innovation of the optimal predictor.

Solution

1. By rewritting the output at time k, we obtain

$$y(k) = [-ay(k-1) + bu(k-1) + c\,e(k-1)] + [e(k)].$$

Since $e(k)$ is a white noise, the two terms on the right-hand side are independent. If $\tilde{y}(k)$ is an arbitrary prediction of $y(k)$ (based on data up to time $k-1$), it holds for the prediction error variance σ,

$$\sigma = E[y(k) - \tilde{y}(k)]^2 = E[-ay(k-1) + bu(k-1) + c\,e(k-1) - \tilde{y}]^2 + \lambda^2 \geq \lambda^2.$$

An optimal predictor $\hat{y}(k|k-1;\theta)$, is the one that minimizes this variance, that is, the one that achieves equality and is given by

$$\hat{y}(k|k-1;\theta) = -ay(k-1) + bu(k-1) + c\,e(k-1).$$

2. The optimal predictor cannot be used in practice, due to the non-measurable signal $e(k-1)$. This problem can be alleviated, if $e(k-1)$ is reconstructed from previously collected data (corresponding to times from 0 up to $k-1$).

3. By substituting $e(k-1)$ with the system data $y(k)$ and $u(k)$, we obtain the summation

$$\hat{y}(k|k-1;\theta) = \sum_{i=0}^{k-1}(c-a)(-c)^{i-1}y(k-i) - a(-c)^{k-1}y(0) + b\sum_{i=1}^{k}(-c)^{i-1}u(k-i) - (-c)^k e(0).$$

Under the assumption $|c| < 1$, the last term has only a decaying transient effect and can be neglected for large k. This implies that

$$\hat{y}(k|k-1;\theta) + c\hat{y}(k-1|k-2;\theta) = (c-a)y(k-1) + bu(k-1).$$

With this simple recursion, we can compute the optimal predictor. Finally by defining $\epsilon(k) = y(k) - \hat{y}(k|k-1;\theta)$, it follows that,

$$\epsilon(k) + c\,\epsilon(k-1;\theta) = y(k) + ay(k-1) - bu(k-1).$$

MATLAB **Exercise:**

Consider the ARMAX model

$$A(z)y(k) = B(z)u(k) + C(z)e(k),\ k = 1, \dots, N$$

with the matrices defined by

$$A(z) = 1 - 1.5z^{-1} + 0.7z^{-2}$$
$$B(z) = 1.0z^{-1} + 0.5z^{-2}$$
$$C(z) = 1 - 1.0z^{-1} + 0.2z^{-2},$$

where $e(k) \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1)$. Fix an input sequence $u(k)$ from the following ARMAX process

$$u(k) = 0.1u(k-1) + 0.12\,u(k-2) + e_u(k-1) + 0.2\,e_u(k-2).$$

where $e_u \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1)$.

Assuming that the transfer function $C(z)$ is exactly known:

1. Obtain least-squares (LS) estimates $\hat{A}_{LS}(z)$ and $\hat{B}_{LS}(z)$ for $A(z)$ and $B(z)$ respectively.
2. Plot the predicted values along with the true response $y(k)$ for validation set.
3. Repeat (1) for a different set of realizations of $e(k)$ and plot the histograms of the parameters.

Solution hints:

The first thing that should be done is to form the discrete time systems that will be used in order to generate relevant system data. These are the transfer functions given by $\frac{B(z)}{A(z)}$ and $\frac{C(z)}{A(z)}$ and the transfer function needed to generate $u(k) = L(z)e_u(k)$. These transfer functions can be generated by using the *tf* command. Random signals e and v can be created by using the *randn* command. The signal u can then be generated from the signal v and the output signal y can be generated from u and e by using the *lsim* command.

1. The output can be written in the following form

$$y(k) = \frac{B(z)}{A(z)}u(k) + \frac{C(z)}{A(z)}e(k)$$

Since $C(z)$ is exactly known by assumption, it can be verified that it is stably invertible and we can write the following

$$C^{-1}(z)y(k) = \frac{B(z)}{A(z)}C^{-1}(z)u(k) + \frac{1}{A(z)}e(k).$$

By bringing the ARMAX model into the ARX model form (see slide 10.39), we can proceed with the Least Square Estimate after redefining the input as $u_F(k) = C^{-1}(z)u(k)$ and the output $y_F(k) = C^{-1}(z)y(k)$. We first write the regressor form,

$$y_F(k) = \varphi(k)^T\theta,$$

where

$$\varphi(k) = \begin{bmatrix} -y_F(k-1) \\ -y_F(k-2) \\ u_F(k-1) \\ u_F(k-2) \end{bmatrix}, \theta = \begin{bmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{bmatrix}.$$

The least squares estimate of θ can then be calculated in MATLAB by the comand $\theta = \Phi\backslash Y_F$, where $Y_F = [y_F(1), \dots, y_F(N)]^T$ and,

$$\Phi = \begin{bmatrix} \varphi(1)^T \\ \vdots \\ \varphi(N)^\top \end{bmatrix}.$$

Note that if the transfer function $C(z)$ was not known and to be estimated along with the matrices $A(z)$ and $B(z)$, then the function `fmincon` can be used to identify $C(z)$.

2. In order to compare the measurements with the predictions, the appropriate transfer function $\frac{B_{LS}(z)}{A_{LS}(z)}$ needs to be formed from the parameters in $\hat{\theta}$ by using the *tf* command. Then by using the *lsim* command on the same input and noise sequence but with real and estimated transfer functions, one can obtain the actual and predicted outputs. The comparison is shown in Fig. 8.1. An alternative method to calculate the residual at each time step k is to take the difference between the true output $y(k)$ and the estimated output $\hat{y}(k) = C\hat{y}_F(k), \hat{y}_F(k) = \varphi(k)\hat{\theta}$, where $\hat{\theta}$ is the vector containing the estimated parameters.

3. By repeating the procedure described in point 1. for different realizations of $e(k)$, we get a sequence of estimated values for each of the parameters a_1, a_2, b_1 and b_2. The histograms of the obtained values can be plotted by using the *histogram* command. Fig. 8.2 shows the obtained histograms.

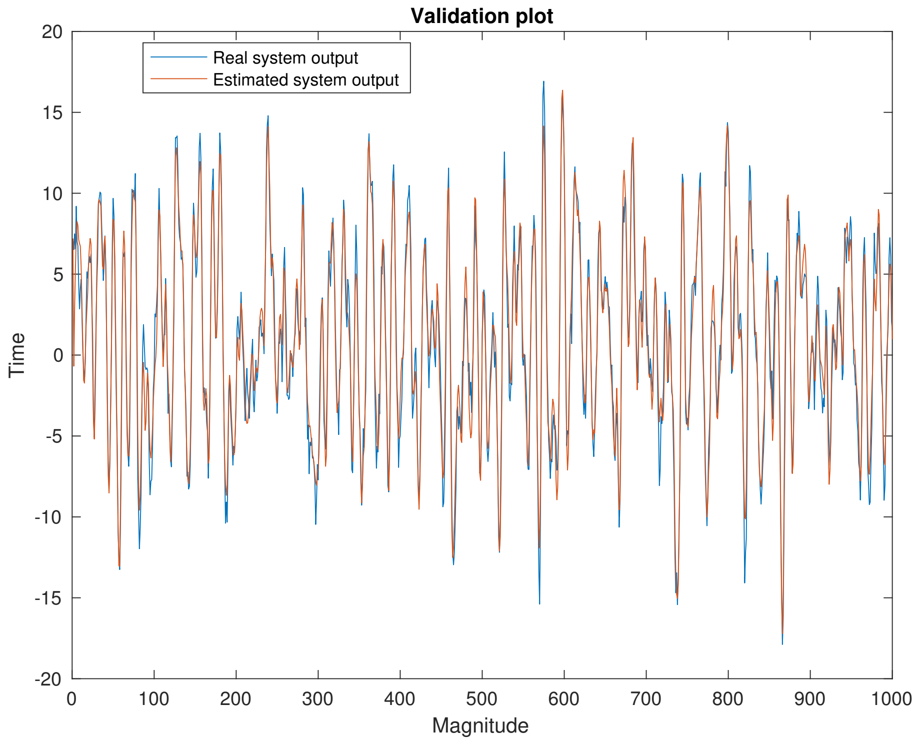

Figure 8.1: Comparison of the true and predicted values of $y(k)$.

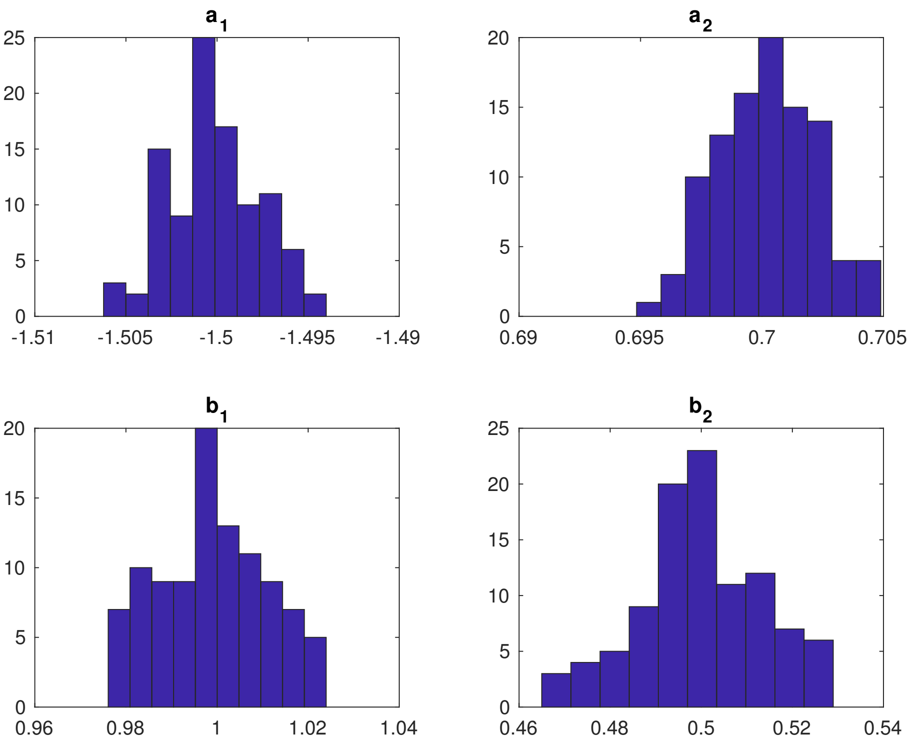

Figure 8.2: Histograms of the estimated parameters a_1, a_2, b_1, b_2.