SSY 230, System Identification
Project 1: Estimating functions from noisy data

Yuxuan Xia
yuxuan.xia@chalmers.se
Emil Staf
emil.staf@chalmers.se

April 25, 2018

1 ARX estimator

1.1 (a) arxfit

The *arxfit* fucntion can beneficially be implemented using the Linear Regression code written in project 1. The ARX model is given by

$$A(q^{-1})y(t) = B(q^{-1})u(t) + e(t) \tag{1}$$

$$\theta = (a_1, \dots a_{na}, b_1, \dots, b_{nb})^\mathsf{T} \tag{2}$$

according to (6.13a) in S & S. This can be rewritten on a Linear Regression format

$$y(t) = \phi(t)\theta + e(t) \tag{3}$$

where

$$\phi(t) = (-y(t-1), \dots, -y(t-na), u(t-1-nk), \dots, u(t-1-nk-nb))^\mathsf{T} \tag{4}$$

The function arxfit is implemented accordingly. It was verified against the function

$$y(t) = 0.2y(t-1) - 0.3y(t-2) + 0.4u(t-2) - 0.2u(t-3) \tag{5}$$

without any noise, and the correct model was obtained.

1.2 (b) id2tf

The function *id2tf* is implemented using MATLABs build in function *tf(Numerator, Denominator, -1)* where *Numerator* $= (\hat{b}_1, \dots, \hat{b}_{nb})$ and *Denominator* $= (1, \hat{a}_1, \dots, \hat{a}_{na})$. In Figure 1 the build-in MATLAB function *ltiview* was used to analyze the model found in (a) including noise.

1.3 (c) idpredict and idsimulate

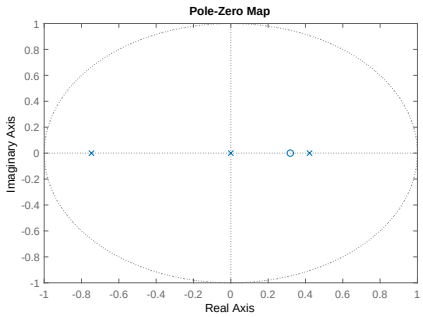
A k-step predictor for an ARX model can be fund by studying the following equations.

$$\hat{y}(t|t-1) = -a_1 y(t-1) - a_2 y(t-2) \dots \tag{6}$$

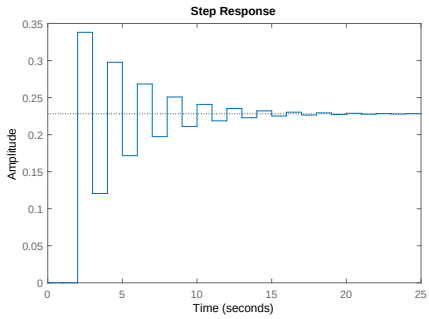
$$\hat{y}(t|t-2) = -a_1 \hat{y}(t-1|t-2) - a_2 y(t-2) \dots \tag{7}$$

$$\hat{y}(t|t-3) = -a_1 \hat{y}(t-1|t-3) - a_2 \hat{y}(t-2|t-3) \dots \tag{8}$$

It is possible to use the 1-step ahead predictor to calculate the 2-step predictor, the 1-step and 2-step predictor to calculate the 3-step predictor and so on. It is the regressor matrix $\Phi_i = [Y_i, U]$



(a) pzmap using ltiview



(b) Step response using ltiview

Figure 1: MATLABs build in function *ltiview* analyzing the found ARX model for the system in (5) including noise.

that needs to be updated in a clever way in order to achieve fast matrix calculations which is shown below. The U matrix is constant and given by

$$U = \begin{bmatrix} u(0-n_k) & u(-1-n_k) & \dots & u(1-n_b-n_k) \\ u(1-n_k) & u(0-n_k) & \dots & u(2-n_b-n_k) \\ u(2-n_k) & u(1-n_k) & \dots & u(3-n_b-n_k) \\ \vdots & \vdots & \vdots & \vdots \\ u(n-1-n_k) & u(n-2-n_k) & \dots & u(n-n_b-n_k) \end{bmatrix} \tag{9}$$

while

$$Y_1 = \begin{bmatrix} y(0) & y(-1) & \dots & y(1-n_a) \\ y(1) & y(0) & \dots & y(2-n_a) \\ y(2) & y(1) & \dots & y(3-n_a) \\ \vdots & \vdots & \vdots & \vdots \\ y(n-1) & y(n-2) & \dots & y(n-n_a) \end{bmatrix} \tag{10}$$

$$Y_2 = \begin{bmatrix} \hat{y}(0|-1) & y(-1) & \dots & y(1-n_a) \\ \hat{y}(1|0) & y(0) & \dots & y(2-n_a) \\ \hat{y}(2|1) & y(1) & \dots & y(3-n_a) \\ \vdots & \vdots & \vdots & \vdots \\ \hat{y}(n-1|n-2) & y(n-2) & \dots & y(n-n_a) \end{bmatrix} \tag{11}$$

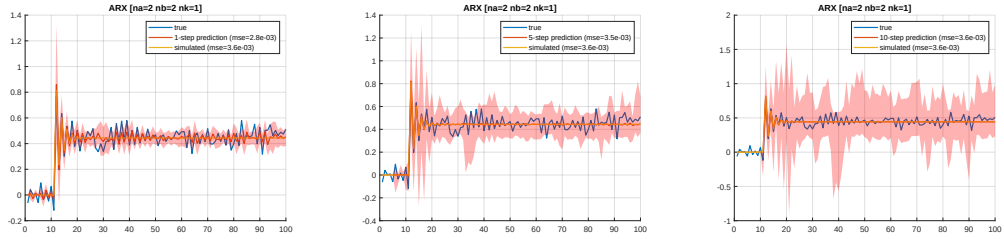
$$Y_3 = \begin{bmatrix} \hat{y}(0|-2) & \hat{y}(-1|-2) & \dots & y(1-n_a) \\ \hat{y}(1|-1) & \hat{y}(0|-1) & \dots & y(2-n_a) \\ \hat{y}(2|0) & \hat{y}(1|0) & \dots & y(3-n_a) \\ \vdots & \vdots & \vdots & \vdots \\ \hat{y}(n-1|n-3) & \hat{y}(n-2|n-1) & \dots & y(n-n_a) \end{bmatrix} \tag{12}$$

where $\hat{y}(t|t-k) = \Phi_k\hat{\theta}$ and the values for both y and u prior to $t = 1$ are initialized to zero.

Simulation can be performed by setting $k > n$, i.e. the predictor is forced not to use any of the measured outputs.

1.4 (d) idcompare

Some results for the same system as before (5) are compared using *idcompare* in Figure 2.



(a) Horizon 1 (b) Horizon 5 (c) Horizon 10

Figure 2: Some results using the implemented *idcompare* function. It is clear that the model uncertainty is increasing for increasing horizons, which means that less data is allowed in the prediction step. The mse on validation data is also increasing as the horizon is increasing.

2 OE estimator

2.1 (a) oefit

Having the Output-Error system

$$S : y(t) = \frac{B_0(q)}{A_0(q)}u(t) + e(t). \tag{13}$$

Estimating a high order ARX model $A_h(q)y(t) = B_h(q)u(t) + e(t)$ it is possible to acheive a close approximation to the system behavior, i.e. $\frac{B_h(q)}{A_h(q)} \approx \frac{B_0(q)}{A_0(q)}$. Thus by simulating the higher order ARX system it is possible to get noise free data that closely resembles the system we are trying to model

$$y_s(t) = \frac{B_h(q)}{A_h(q)} \tag{14}$$

2.2 Approximate

Now using the noise free data, an ARX model fit becomes the same as an OE model fit. Thus we obtain our OE prediction $\hat{y}(t) = \frac{\hat{B(q)}}{\hat{A}(q)}u(t) + e(t)$ from the ARX model fit $\hat{A}(q)y_s(t) = \hat{B(q)}u(t)$. The prediction error then becomes $|\frac{B_0(q)}{A_0(q)}u(t) - \frac{\hat{B(q)}}{\hat{A}(q)}u(t)|$, which is minimized for $\frac{B_h(q)}{A_h(q)} = \frac{B_0(q)}{A_0(q)}$.

2.3 Optimal

...

2.4 (b) id2tf

No changes from ARX.

2.5 (c) idpredict

According to the lecture notes a k-step predictor can be written on the form

$$\hat{y}(t|t-k) = \bar{H}_k(q)H^{-1}(q)G(q)u(t) + (1 - \bar{H}_k(q)H^{-1}(q))y(t) \tag{15}$$

which for an OE-model simplifies to

$$\hat{y}(t|t-k) = G(q)u(t) \tag{16}$$

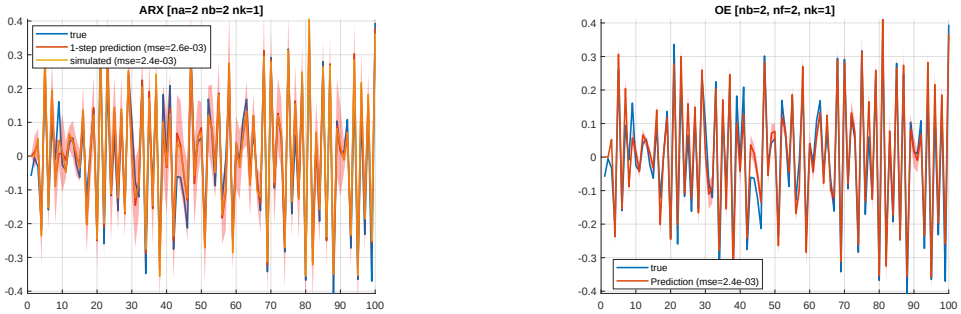
since $H^{-1}(q) = 1$ and $\bar{H}_k^{-1}(q) = 1$ for all $k > 1$. Therefore prediction is the same as simulation for the OE-model and the *idpredict* function calls the *idsimulate* function in the case of an OE-model.

2.6 (d) idsimulate

Using the MATLAB build in function *lsim* after creating the OE transfer function by calling *id2tf*.

2.7 (e) idcompare

No specific changes from ARX except that the prediction and simulation are the same so one of them were removed in the plots. Showing only the prediction since simulation is the same as stated above. This can be seen in the Figure 3, where OE-data was created and an ARX-model is compared to an OE-model.



(a) ARX-model

(b) OE-model

Figure 3: OE-model vs ARX-model on OE system of order nb = 2, nf = 2, nk = 1.

As can be seen in Figure 3 the model uncertainty is substantially smaller for the OE-model in comparison to the ARX model.

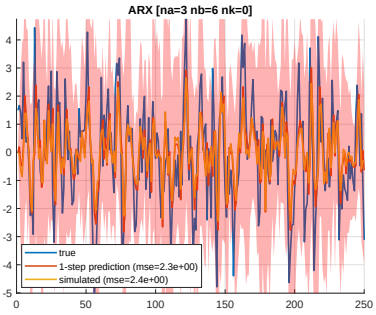
3 Identify two systems

3.1 Results on data set exercise1

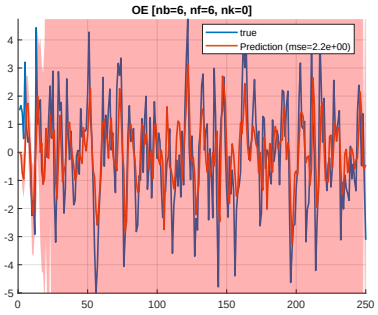
The results on exercise1 data shows that both the ARX and OE system are stable and that they produce prediction results of similar accuracy. Therefore we choose the model according to the Parcimony Principle and that is the ARX model which has a total of 9 parameters while the OE model has a total of 12 parameters.

3.2 Results on data set exercise2

In Figure 7a there are three zeros and poles that are close and should cancel each other out. This means that we probably have chosen a too high model order and are likely to overfit the data. After reducing the model order the result in figure 8 is obtained.

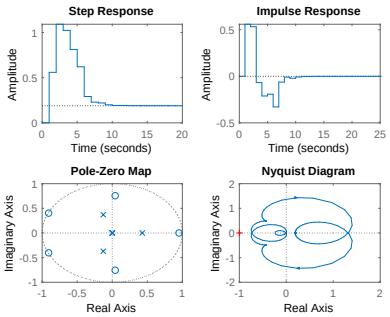


(a) ARX

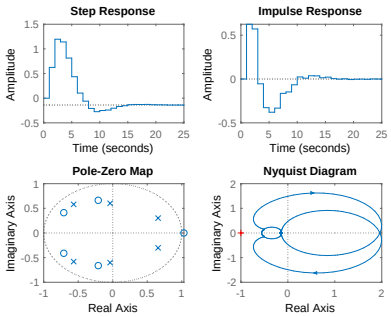


(b) OE

Figure 4: Dataset exercise1



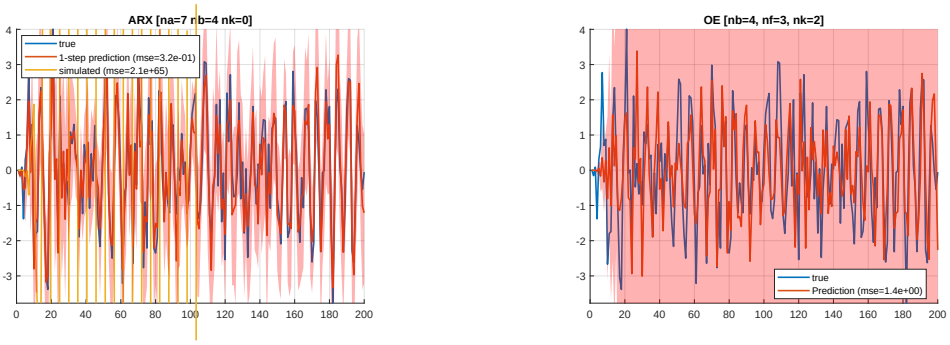
(a) ARX



(b) OE

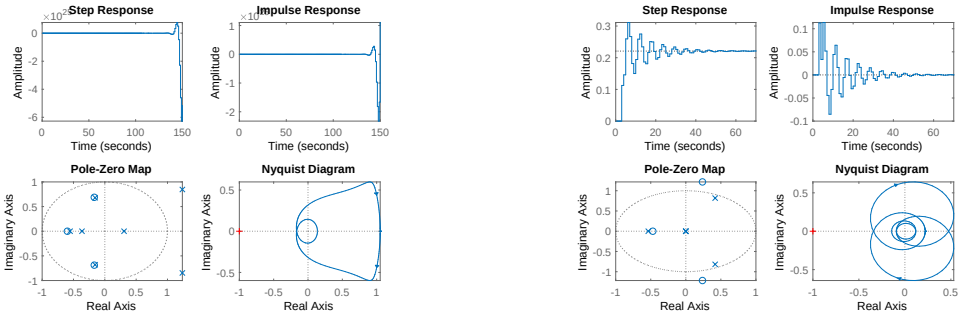
Figure 5: Dataset exercise1, using build-in MATLAB plotting functionality.

The ARX model is still unstable for the data set exercise2, while the OE model is stable. Thus the OE model is probably the correct model to use for the data set exercise2. As the pole-zero map shows in Figure 7 it could be worth trying out an OE model of order $n_F = 2$, $n_B = 3$ and $n_K = 2$.



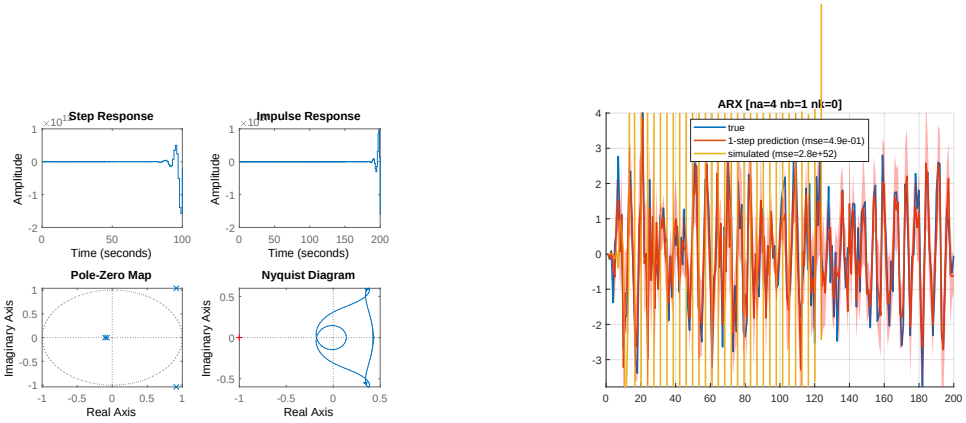
(a) ARX (b) OE

Figure 6: Dataset exercise2



(a) ARX (b) OE

Figure 7: Dataset exercise2, using build-in MATLAB plotting functionality.



(a) *ltiview* (b) *idcompare*

Figure 8: ARX result after canceling out some zeros and poled from dataset exercise2.