

# SSY 230, System Identification

## Project 1: Estimating functions from noisy data

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

May 8, 2018

### 1 ARX estimator

#### 1.1 (a) arxfit

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

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

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

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

$$y(t) = \phi(t)\theta + e(t) \quad (3)$$

where

$$\phi(t) = (-y(t-1), \dots, -y(t-na), u(t-1-nk), \dots, u(t-1-nk-nb))^T \quad (4)$$

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

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

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, 'Variable', 'q^{-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 found by studying the following equations.

$$\hat{y}(t|t-1) = -a_1y(t-1) - a_2y(t-2) \dots \quad (6)$$

$$\hat{y}(t|t-2) = -a_1\hat{y}(t-1|t-2) - a_2y(t-2) \dots \quad (7)$$

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

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  $\Phi_i = [Y_i, U]$

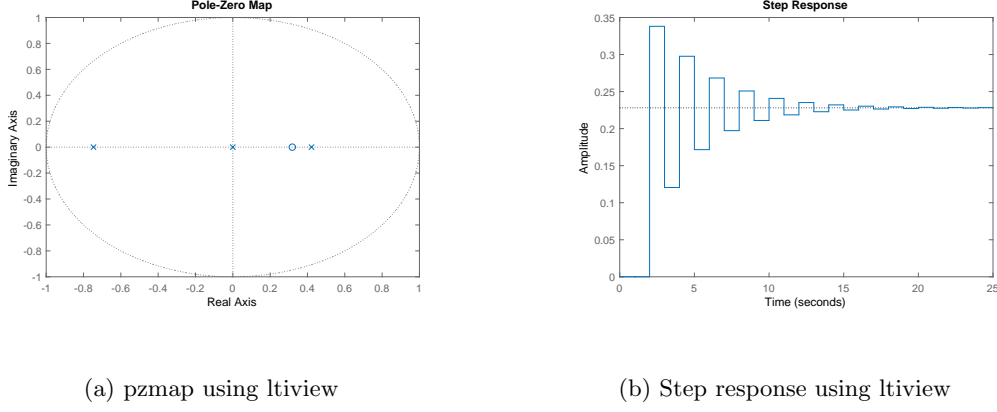


Figure 1: MATLABs build in function *ltiview* analyzing the found ARX model for the system in (5) including noise.

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

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

while

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

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

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

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*, along with their prediction/simulation uncertainties, see Figure 2.

When we estimate the model parameters from data, we also obtain their estimation covariance, with the help of which, we can visualize how accurate our estimations are in a con

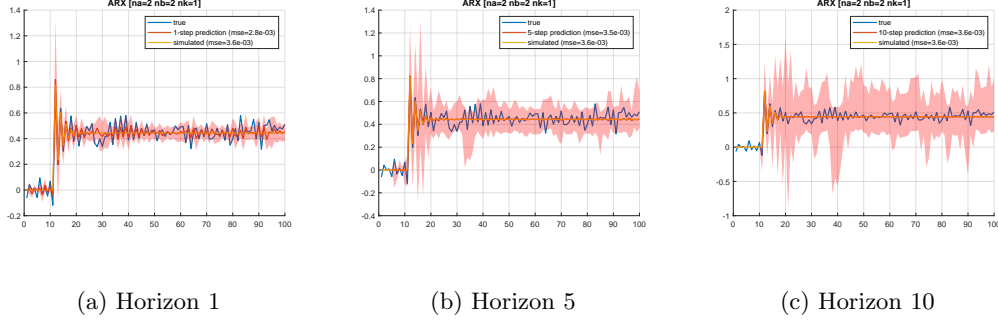


Figure 2: Some results using the implemented *idcompare* function. It is clear that the model uncertainty is increasing for increasing horizons, which means that less data is allowed in the prediction step. The mse on validation data is also increasing as the horizon is increasing.

dence region. The magnitude of the uncertainties provide a measure of the reliability of the model. With the help of MATLAB System Identification toolbox, we can compute and visualize the effect of parameter uncertainties on the model response in time and frequency domains using pole-zero maps, Bode plot, and step response plots, etc.

## 2 Write an OE estimator

### 2.1 (a) oefit

Having the Output-Error system

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

Estimating a high order ARX model  $A_h(q)y(t) = B_h(q)u(t) + e(t)$  it is possible to achieve 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

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

Adding white noise to the output signal given that the number of data is high, we can re-obtain the true parameters using an OE model; however, this is not the case for ARX model. This difference is a direct consequence of the different ways modeling the system disturbances in the OE and the ARX model. In the OE model, system disturbances are directly added to the output signal, whereas in the ARX model, system disturbances are part of the system dynamics.

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  $\frac{B_h(q)}{A_h(q)} = \frac{B_0(q)}{A_0(q)}$ .

We have also successfully implemented the OE model using this optimal method specified in the problem description, i.e., filtering the simulated and input data using a FIR filter with parameter  $A_h$  before estimating using the ARX model with original order, see attached MATLAB file.

### 2.2 (b) id2tf

No changes from ARX.

## 2.3 (c) idpredict

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

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

which for an OE-model simplifies to

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

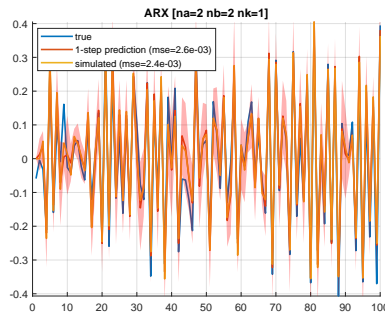
since  $H^{-1}(q) = 1$  and  $\bar{H}_k^{-1}(q) = 1$  for all  $k > 1$ . Therefore prediction is the same as simulation for the OE-model and the *idpredict* function calls the *idsimulate* function in the case of an OE-model.

## 2.4 (d) idsimulate

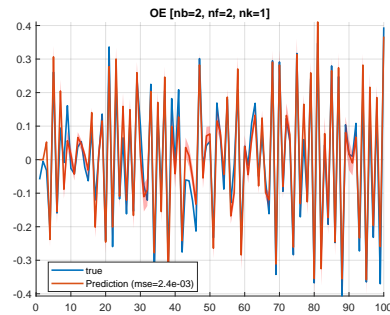
Using the MATLAB build in function *lsim* after creating the OE transfer function by calling *id2tf*.

## 2.5 (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

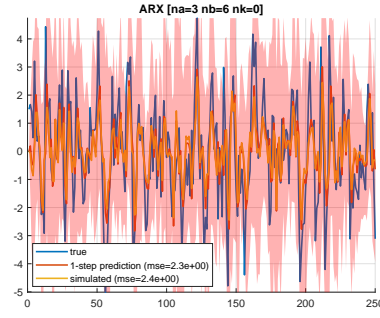
Figure 3: OE-model vs ARX-model on OE system of order  $nb = 2$ ,  $nf = 2$ ,  $nk = 1$ .

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

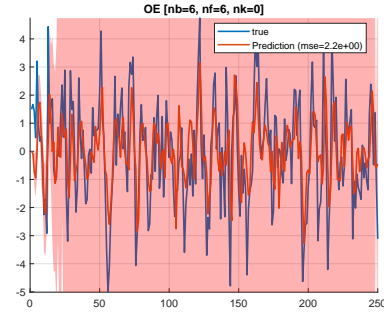
## 3 Identify two systems

### 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. It is therefore possible to use either of the models to model the stable system generating exercise1 data. We choose the model according to the Parsimony Principle and that is the ARX model which has a total of 9 parameters while the OE model has a total of 12 parameters.

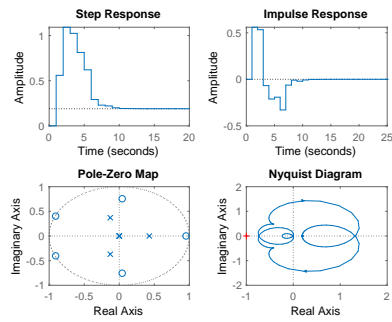


(a) ARX

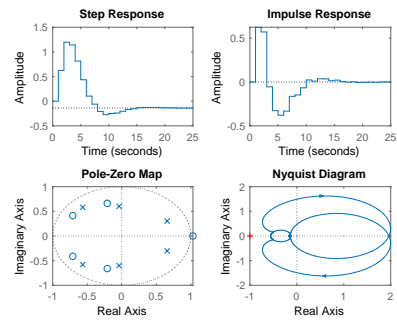


(b) OE

Figure 4: Dataset exercise1



(a) ARX



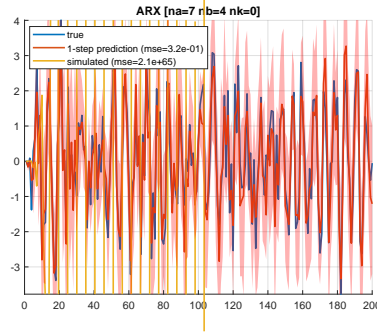
(b) OE

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

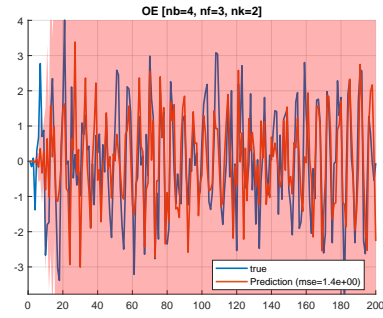
### 3.2 Results on data set exercise2

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.

It is obvious from Figure 8 that the system modelled is unstable due to the large MSE for the simulated ARX model. Due to the fact that the simulated output from an ARX model is used to find the parameters for the OE model, it does not make sense using it to model an unstable system by making use of the simulated output which does not match the true system at all. Thus, the ARX model is the preferred model for exercise2 data.

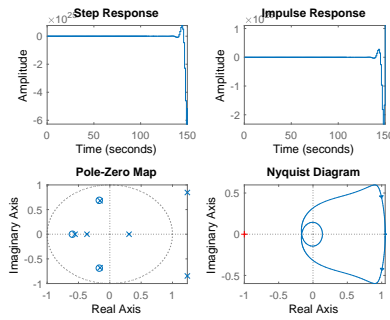


(a) ARX

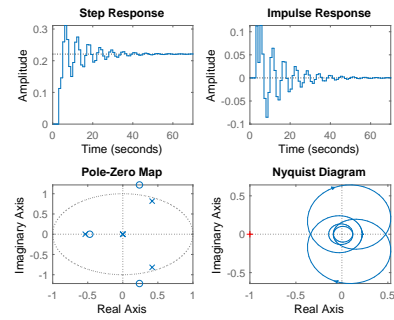


(b) OE

Figure 6: Dataset exercise2

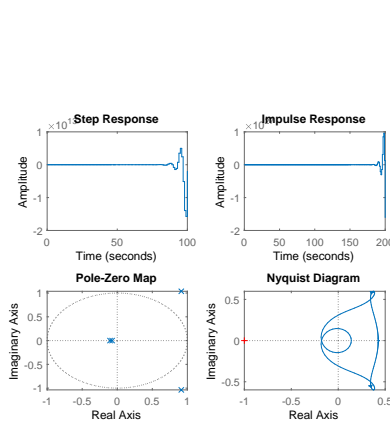


(a) ARX

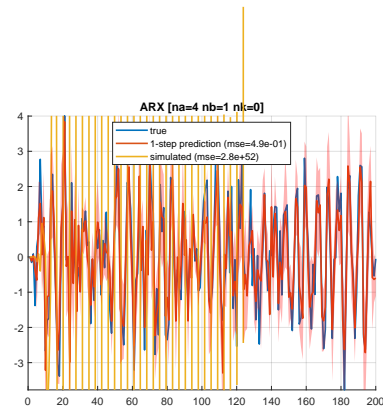


(b) OE

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.