Hand in Assignment 2 - System Identification

Fikri Farhan Witjaksono Oliver Wallin

October 3, 2019

# Introduction

This assignment handles application of various estimation methods.

# Part 1:

# Question 1 Unbiased estimate

We seek

# Question 2 Sample mean and maximum likelihood

We will now minimize this expression:

# Question 3 Linear least squares

## Compute the least squares estimate of θ

The least squares criteria are:

The least squares criteria are a measurement of how well the model prediction is compared to the real system.

## Is the estimate biased?

## What happens if

By looking at the equation that proves the estimate unbiased we can clearly see that if u[k] would be 0, will result in a forbidden division (1/0). **Therefore, θ can’t be estimated.**

# Question 4 Invariance property of the maximum likelihood estimate

Now we minimize:

# Part two:

# Question 1 One-step ahead predictor

## What kind of model structure is it?

Where:

Then:

## Which are the expressions for the plant model G and the noise H?

## Find the 1-step-ahead predictor

## Is it linear?

+

# The 1-step-ahead predictor is non-linear to the parameters since it is independent to the parameters in the predictor. This could be proven by replacing by expanding it. Here we could see that the parameters will not be changed after expansion. Hence, non-linear function of the parameters.

# Question 2 Prediction or simulation?

## What kind of model structure is it?

Where:

Then:

## Find the 1-step-ahead predictor.

# Computer exercise. Identification of an ARX model

First of all we start by splitting the data in estimation and validation sets by half of the input array and the output array equally.

Let

First, we need to write the predictor for the 3 different candidate ARX model as a linear regression using the estimation data input ( and output (.

Then, will be found to minimize the prediction for the given data as shown in the slide 83 of the lecture Notes.

After that, then we separate each parameter of ,,,, with respect to the which concern each different ARX model.

we need to compute the array of the 1-step-ahead predictor which is shown below.

Next, we will also need to compute the simulation

Next step would be calculating the Root Mean Square Error as a measure of quality level of estimation. The candidate with the lowest RMSE value will be the best quality one.

Let ,

The result of this calculation is shown in the table below

Table 1. Each Candidate RMSE of Predication and Simulation

|  |  |  |
| --- | --- | --- |
| Candidate |  |  |
| 1 | 1.0587 | 1.4636 |
| 2 | 0.3183 | 0.4210 |
| 3 | 0.0493 | 0.0591 |

Based on Table 1, we conclude that *candidate 3* is the best model for both the prediction as well as simulation since the RMSE value of both are the lowest compared to other candidates.

Finally, we would want to calculate the covariance of the parameters by using the formula below which is expressed in the lecture notes (pg.84).

.

The result of the covariance calculation is shown in the matrix below for each respective candidates.

Here, we could observe that the diagonal elements of covariance matrix from all candidates has the same order, that is order. On the other hand, covariance matrix of *’candidate 1’* has 3 diagonal member as it is a 3x3 matrix size whereas the other 2 candidates has 4 diagonal members as they are 4x4 matrix size.

APPENDIX A (FULL MATLAB CODES)

load input.mat

load output.mat

% 1) Model definition

% In this exercise we will work with the 3 candidate ARX defined as

% y(t)+a1\*y(t-1)+a2\*y(t-2) = b0\*u(t)+e(t)

% y(t)+a1\*y(t-1)+a2\*y(t-2) = b0\*u(t)+b1\*u(t-1)+e(t)

% y(t)+a1\*y(t-1)+a2\*y(t-2)+a3\*y(t-3) = b1\*u(t-1)+e(t)

%%

% 2. Identification using Least Squares formula

% First of all lets split the data in estimation and validation sets

% (half and half)

N = length(y);

% number of data

uest = u(1:N/2);

yest = y(1:N/2);

uval = u(N/2+1:end);

yval = y(N/2+1:end);

% Build the matrix PHI:

PHI = zeros(3,N/2); % The matrix has 3 rows (3 parameters), N/2 columns (time instants)

% PHI = zeros(4,N/2); % Switch to this PHI for the 2nd and 3rd Candidate

% Switch for each respective candidate

PHI(:,1) = [ uest(1) ; 0; 0 ]; %candidate 1

% PHI(:,1) = [ uest(1) ; uest(2); 0; 0]; %candidate 2

% PHI(:,1) = [ uest(1) ; 0; 0; 0 ]; %candidate 3

for i=3:1:N/2 % change the index to i=N/2:-1:4 for the 3rd ARX

% Switch for each respective candidate

%candidate 1

PHI(:,i) = [uest(i) ; yest(i-1); yest(i-2)] ;

%candidate 2

% PHI(:,i) = [uest(i) ; uest(i-1); yest(i-1); yest(i-2)];

%candidate 3

% PHI(:,i) = [uest(i-1) ; yest(i-1); yest(i-2); yest(i-3)];

end

% Write the expression in code as well, so we get our estimate!

th = (PHI\*PHI')\PHI\*yest;

% th should be be a vector with two elements, estimated value for a and

% estimate value for b. Pick this two elements separetely:

% Switch for each respective candidate

%First ARX Parameters

bhat\_0 = th(1)

ahat\_1 = -th(2)

ahat\_2 = -th(3)

%Second ARX Parameters

% bhat\_0 = th(1)

% bhat\_1 = th(2)

% ahat\_1 = -th(3)

% ahat\_2 = -th(4)

%Third ARX Parameters

% bhat\_1 = th(1)

% ahat\_1 = -th(2)

% ahat\_2 = -th(3)

% ahat\_3 = -th(4)

%% 3. ARX Prediction & Simulation

N = length(y);

NN = N/2;

% Redefine the data variables so it will be easy to switch from

% validation to estimation sets.

yn = yval;

un = uval;

% yn = yest;

% un = uest;

% Write your code for 1-step-ahead predictor

ypred = zeros(NN,1); % the vector where we will store the predicted output

for i=NN:-1:3 % For the 3rd ARX model, we switch the index to i=NN:-1:4

%Switch between the 1st ARX, 2nd ARX and 3rd ARX for the predictor

%candidate 1

ypred(i) = -ahat\_1\*yn(i-1)-ahat\_2\*yn(i-2)+ bhat\_0\*un(i);

%candidate 2

% ypred(i) = -ahat\_1\*yn(i-1)-ahat\_2\*yn(i-2) + bhat\_0\*un(i) +bhat\_1\*un(i-1);

%candidate 3

% ypred(i)= -ahat\_1\*yn(i-1)-ahat\_2\*yn(i-2)-ahat\_3\*yn(i-3) + bhat\_1\*un(i);

end

% Write the code for simulation

ysim = zeros(NN,1); % the vector where we will store the simulated output

for i=NN:-1:3 %For the 3rd ARX model, we switch the index to i=NN:-1:4

%Switch between the 1st ARX, 2nd ARX and 3rd ARX for the simulation

%candidate 1

ysim(i) = -ahat\_1\*ysim(i-1)-ahat\_2\*ysim(i-2)+ bhat\_0\*un(i);

%candidate 2

% ysim(i) = -ahat\_1\*ysim(i-1)-ahat\_2\*ysim(i-2) + bhat\_0\*un(i) +bhat\_1\*un(i-1);

%candidate 3

% ysim(i)= -ahat\_1\*ysim(i-1)-ahat\_2\*ysim(i-2)-ahat\_3\*ysim(i-3) + bhat\_1\*un(i-1);

end

% compare with real data and compute RMSE

predERROR = yn-ypred;

predRMSE = rms(predERROR)

simERROR = yn-ysim;

simRMSE = rms(simERROR)

% 4. Covariance Analysis of the ARX models

%ASK

%Gaussian White Noise

noise\_std = 0.01;

covariance = inv(PHI\*PHI.')\*(1/NN)\*(noise\_std^2)

% plot Predicted DATA vs MODEL prediction and

figure (1)

subplot(2,1,1)

plot(yn)

hold on

plot(ypred)

legend(' DATA','Model prediction')

title('Output')

xlabel('Samples')

ylabel('output')

subplot(2,1,2)

plot(predERROR)

legend('prediction error')

xlabel('Samples')

ylabel('error')

figure (2)

subplot(2,1,1)

plot(yn)

hold on

plot(ysim)

legend('DATA','Model simulation')

title('Output')

xlabel('Samples')

ylabel('output')

subplot(2,1,2)

plot(simERROR)

legend('Simulation error')

xlabel('Samples')

ylabel('error')

%Second ARX Parameters

% bhat\_0 = th(1)

% bhat\_1 = th(2)

% ahat\_1 = -th(3)

% ahat\_2 = -th(4)

%Third ARX Parameters

% bhat\_1 = th(1)

% ahat\_1 = -th(2)

% ahat\_2 = -th(3)

% ahat\_3 = -th(4)

%% 3. ARX Prediction & Simulation

N = length(y);

NN = N/2;

% Redefine the data variables so it will be easy to switch from

% validation to estimation sets.

yn = yval;

un = uval;

% Write your code for 1-step-ahead predictor

ypred = zeros(NN,1); % the vector where we will store the predicted output

for i=3:1:NN % For the 3rd ARX model, we switch the index to i=NN:-1:4

%Switch between the 1st ARX, 2nd ARX and 3rd ARX for the predictor

%candidate 1

ypred(i) = -ahat\_1\*yn(i-1)-ahat\_2\*yn(i-2)+ bhat\_0\*un(i);

%candidate 2

% ypred(i) = -ahat\_1\*yn(i-1)-ahat\_2\*yn(i-2) + bhat\_0\*un(i) +bhat\_1\*un(i-1);

%candidate 3

% ypred(i)= -ahat\_1\*yn(i-1)-ahat\_2\*yn(i-2)-ahat\_3\*yn(i-3) + bhat\_1\*un(i-1);

end

% Write the code for simulation

ysim = zeros(NN,1); % the vector where we will store the simulated output

for i=3:1:NN%For the 3rd ARX model, we switch the index to i=NN:-1:4

%Switch between the 1st ARX, 2nd ARX and 3rd ARX for the simulation

%candidate 1

ysim(i) = -ahat\_1\*ysim(i-1)-ahat\_2\*ysim(i-2)+ bhat\_0\*un(i);

%candidate 2

% ysim(i) = -ahat\_1\*ysim(i-1)-ahat\_2\*ysim(i-2) + bhat\_0\*un(i) +bhat\_1\*un(i-1);

%candidate 3

% ysim(i)= -ahat\_1\*ysim(i-1)-ahat\_2\*ysim(i-2)-ahat\_3\*ysim(i-3) + bhat\_1\*un(i-1);

end

% compare with real data and compute RMSE

predERROR = yn-ypred;

predRMSE = rms(predERROR)

simERROR = yn-ysim;

simRMSE = rms(simERROR)

% 4. Covariance Analysis of the ARX models

%ASK

%Gaussian White Noise

noise\_std = 0.01;

covariance = inv(PHI\*PHI.')\*(1/NN)\*(noise\_std^2)

% plot Predicted DATA vs MODEL prediction and

figure (1)

subplot(2,1,1)

plot(yn)

hold on

plot(ypred)

legend(' DATA','Model prediction')

title('Output')

xlabel('Samples')

ylabel('output')

subplot(2,1,2)

plot(predERROR)

legend('prediction error')

xlabel('Samples')

ylabel('error')

figure (2)

subplot(2,1,1)

plot(yn)

hold on

plot(ysim)

legend('DATA','Model simulation')

title('Output')

xlabel('Samples')

ylabel('output')

subplot(2,1,2)

plot(simERROR)

legend('Simulation error')

xlabel('Samples')

ylabel('error')

% compare with real data and compute RMSE

predERROR = yn-ypred;

predRMSE = rms(predERROR)

simERROR = yn-ysim;

simRMSE = rms(simERROR)

% 4. Covariance Analysis of the ARX models

%Gaussian White Noise

noise\_std = 0.01;

covariance = inv(PHI\*PHI.')\*(noise\_std^2)

% plot Predicted DATA vs MODEL prediction and

figure (1)

subplot(2,1,1)

plot(yn)

hold on

plot(ypred)

legend(' DATA','Model prediction')

title('Output')

xlabel('Samples')

ylabel('output')

subplot(2,1,2)

plot(predERROR)

legend('prediction error')

xlabel('Samples')

ylabel('error')

figure (2)

subplot(2,1,1)

plot(yn)

hold on

plot(ysim)

legend('DATA','Model simulation')

title('Output')

xlabel('Samples')

ylabel('output')

subplot(2,1,2)

plot(simERROR)

legend('Simulation error')

xlabel('Samples')

ylabel('error')