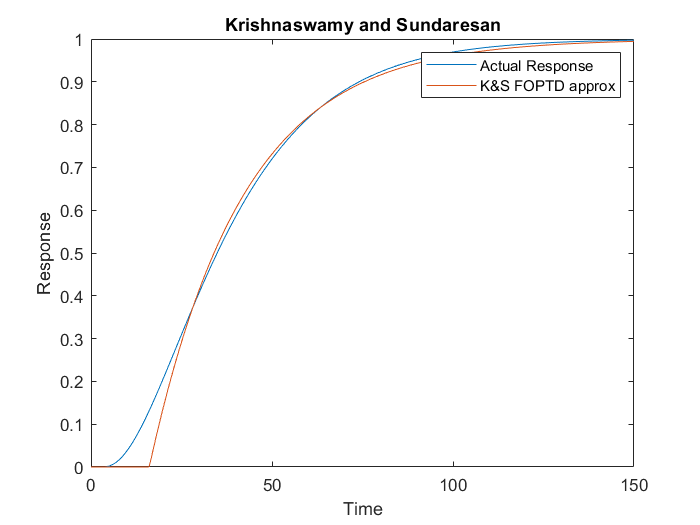
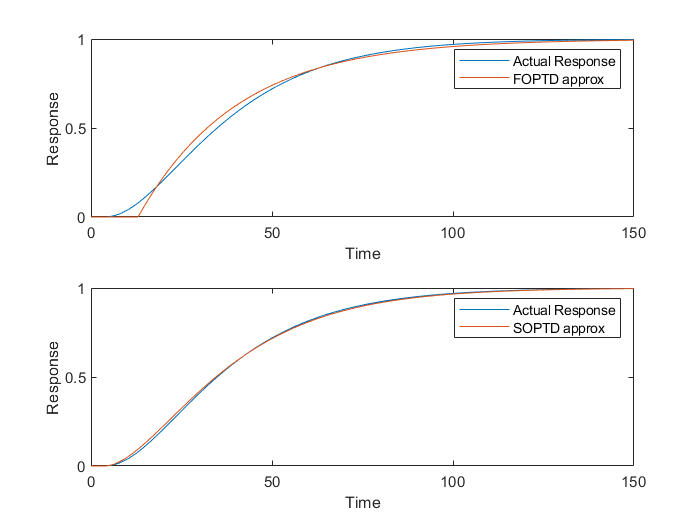
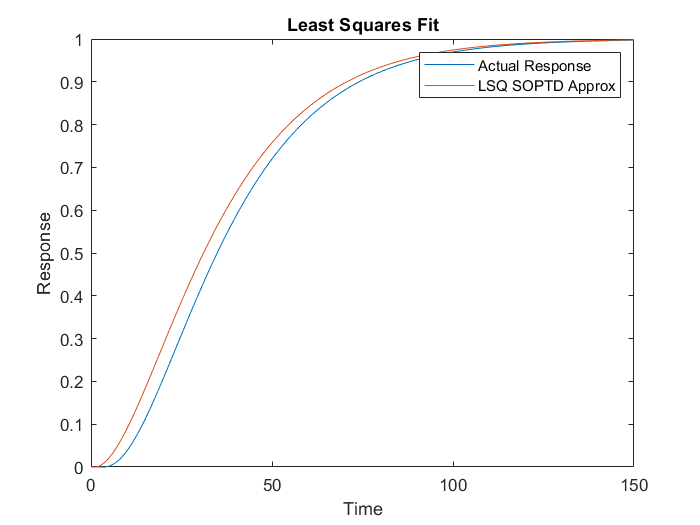
# Question 2) part d)



*Figure 2.1: Comparing actual response with Krishnaswamy and Sundaresan approximation*



*Figure 2.2: Comparing actual response with Skogestad’s half point approximations*



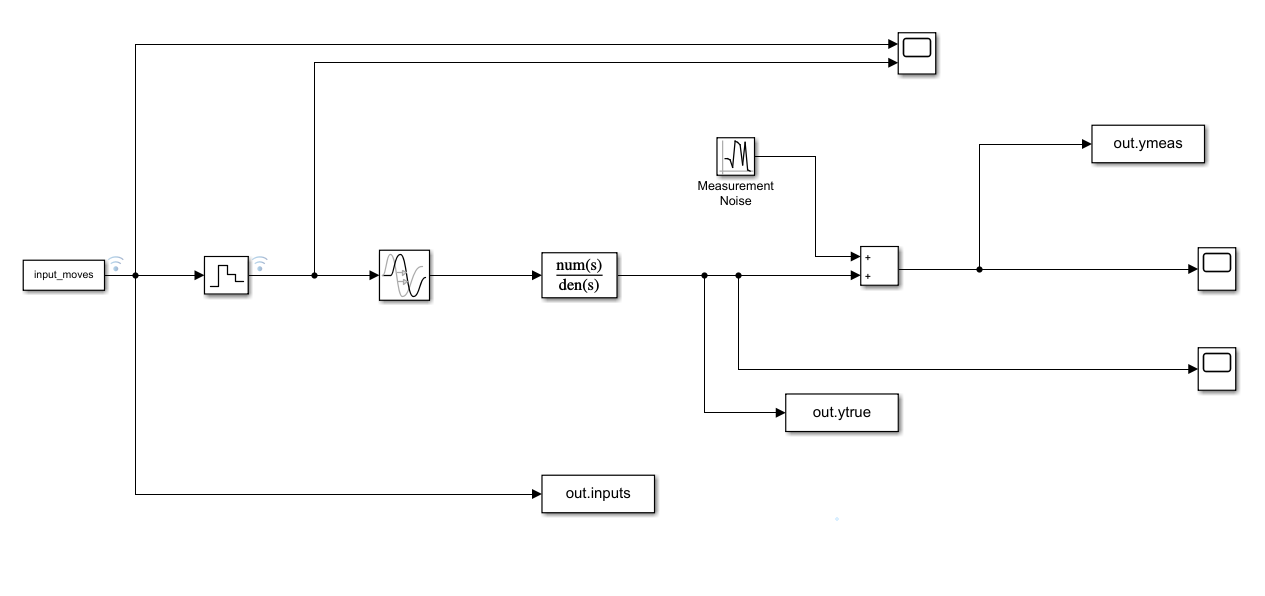
*Figure 2.3: Comparing actual response with Frequency domain Least Squares approximation*

**Observations:**

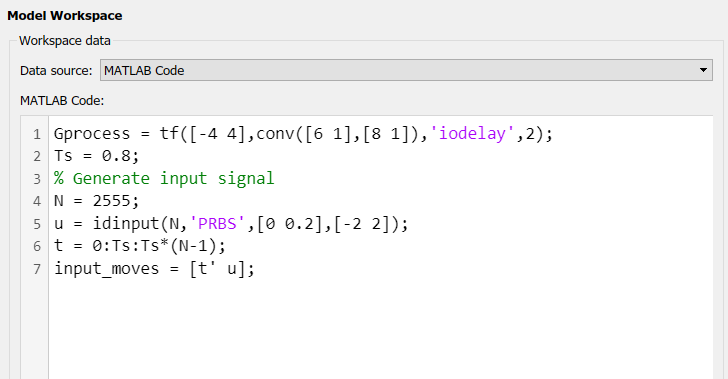
* All approximations capture the gain properly.
* As expected KS method will reach the 35% and 85% mark at the same time as the true process reaches.
* Both the FOPTD models pushed most of the higher order sluggishness as a delay, so the model response starts a long time after the true response of the process begins
* Least Squares has captured the delay most appropriately in the step response plots. This could be because we did a least squares fit to estimate delay alone.
* Comparing the step response plots, Skoegstad’s half point SOPTD model is the best model (least MSE)

# Questions 3)

## Part a) SIMULINK DIAGRAM

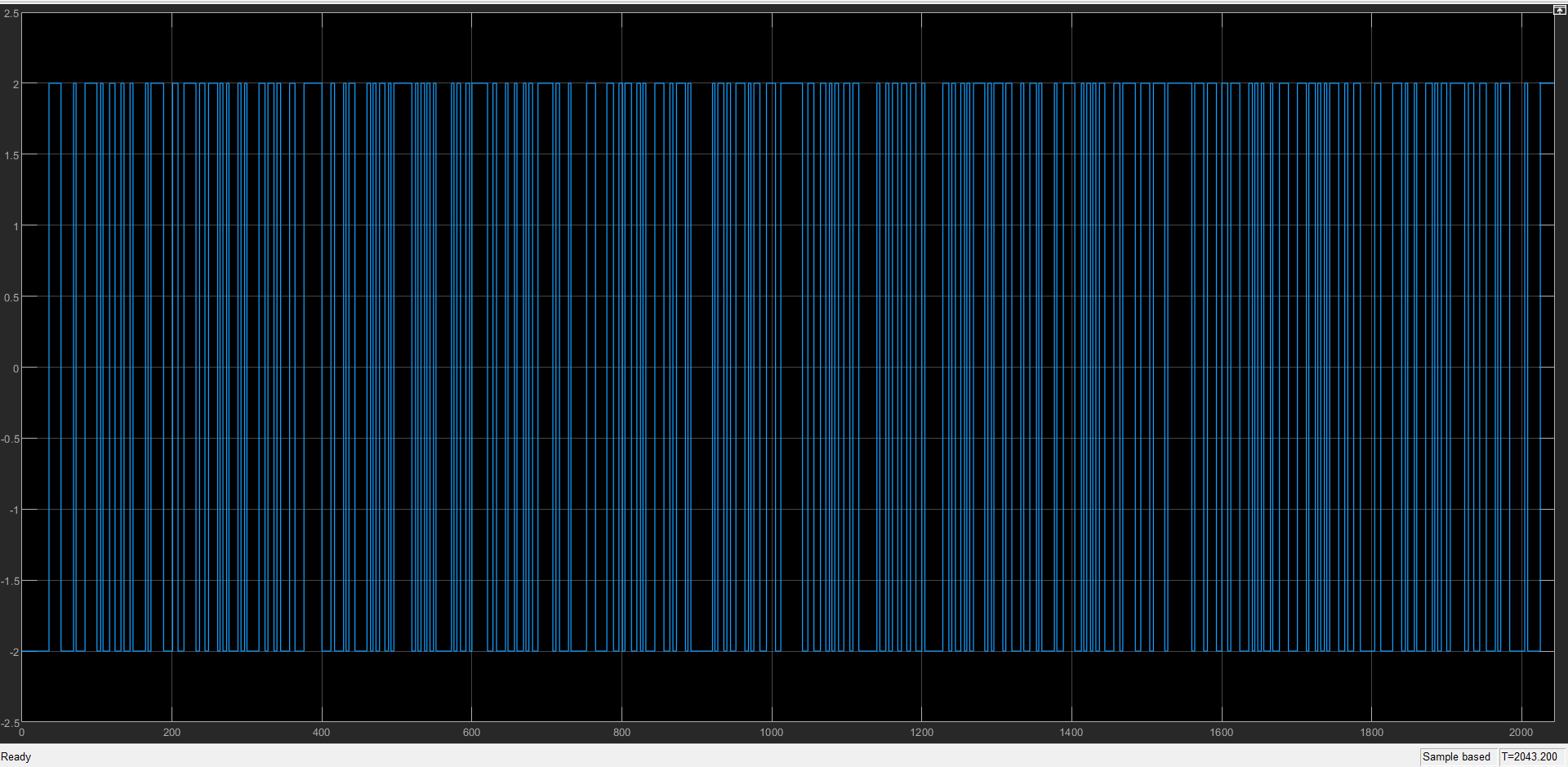


*Figure 3.1: The SIMULINK diagram. Sampler was incorporated in the ‘To Workspace’ block*

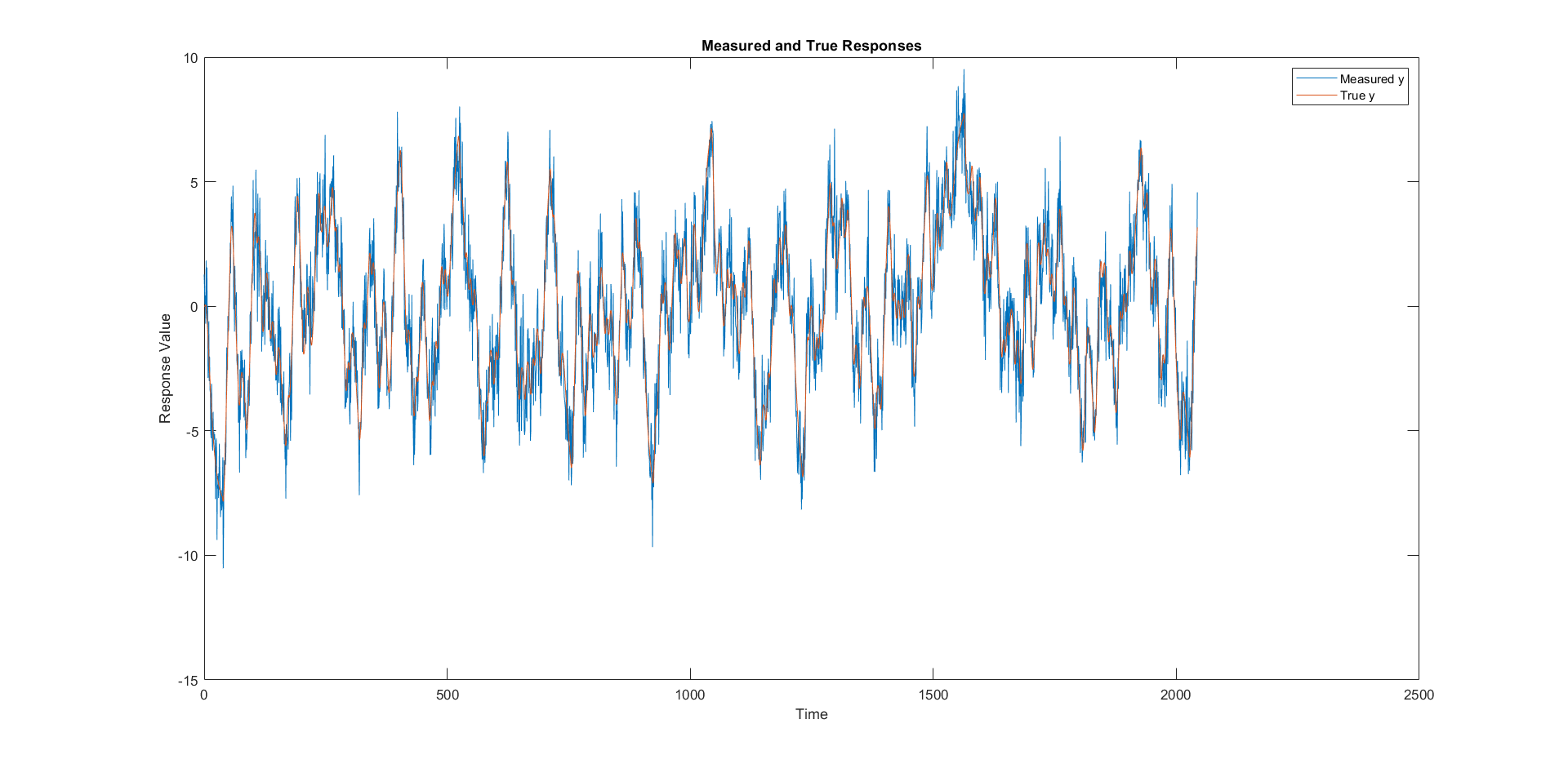


*Figure 3.2: Workspace variables*

## Part b) Designing input and getting output



*Figure 3.3: PRBS inputs after ZOH*

**

*Figure 3.4: Measured and True responses*

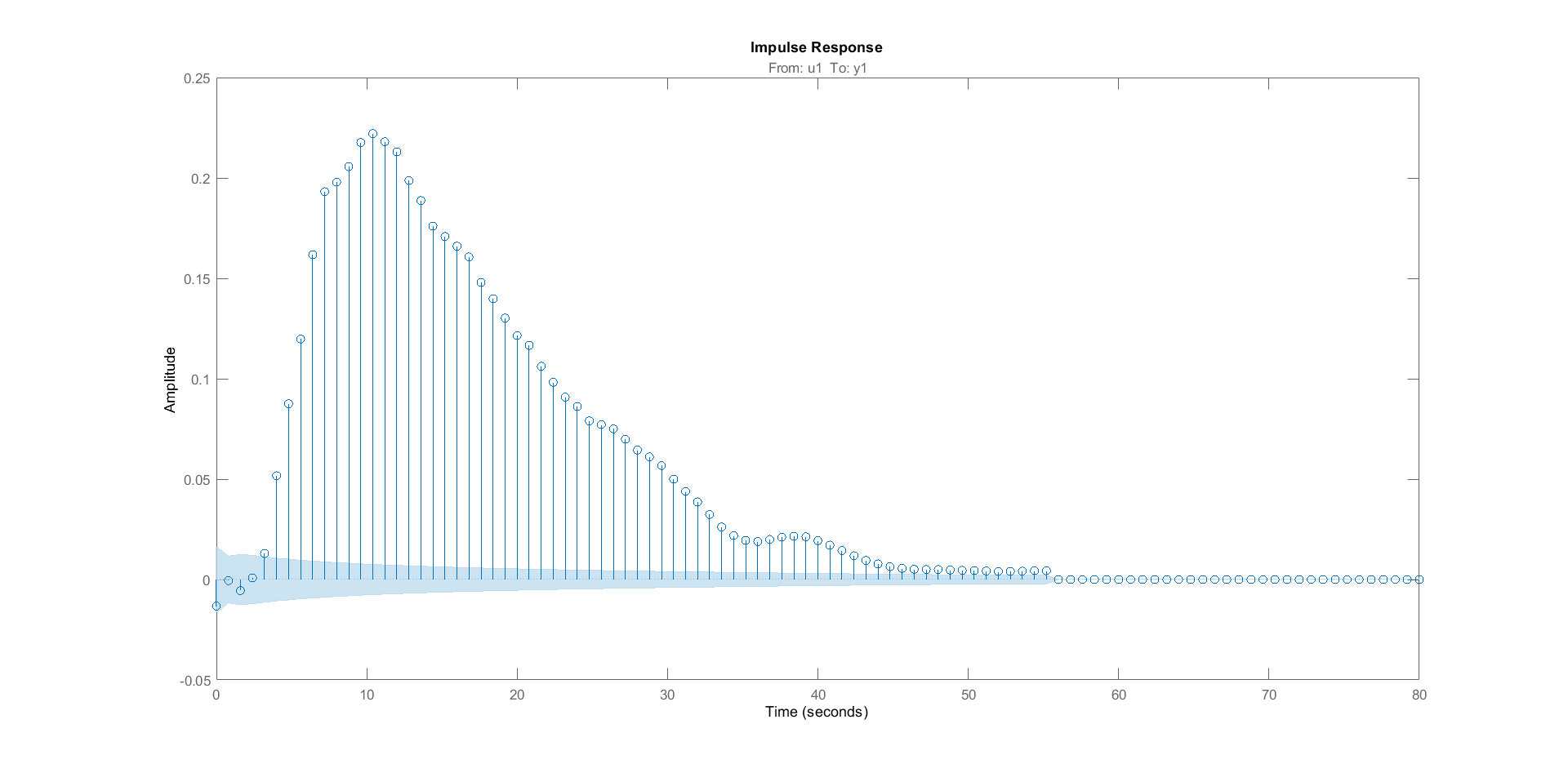
## Part c) Response based model

Finite impulse response is built as depicted in live class session. The *‘impulseest’* routine was used for that purpose.

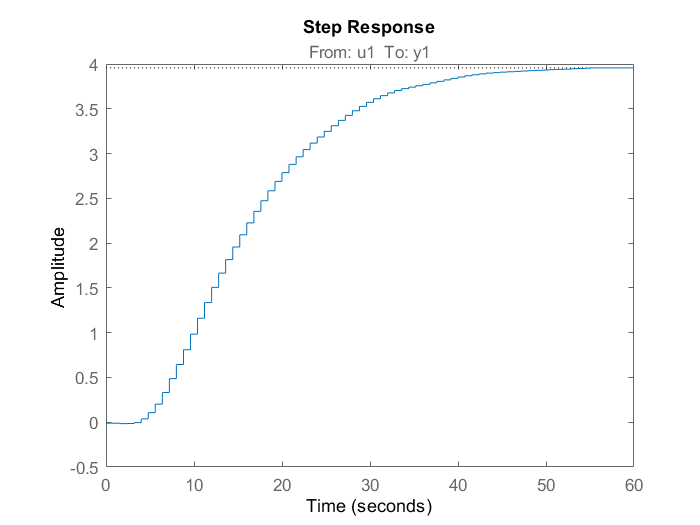
From figure 3.5 (Impulse response) we can see that there only the response at the fifth time instant is significant. So there is a delay of **4 steps**. (In time units, it is 4\*0.8 = 3.2s)

From figure 3.6 (Step response), gain was obtained to be 3.96 units.

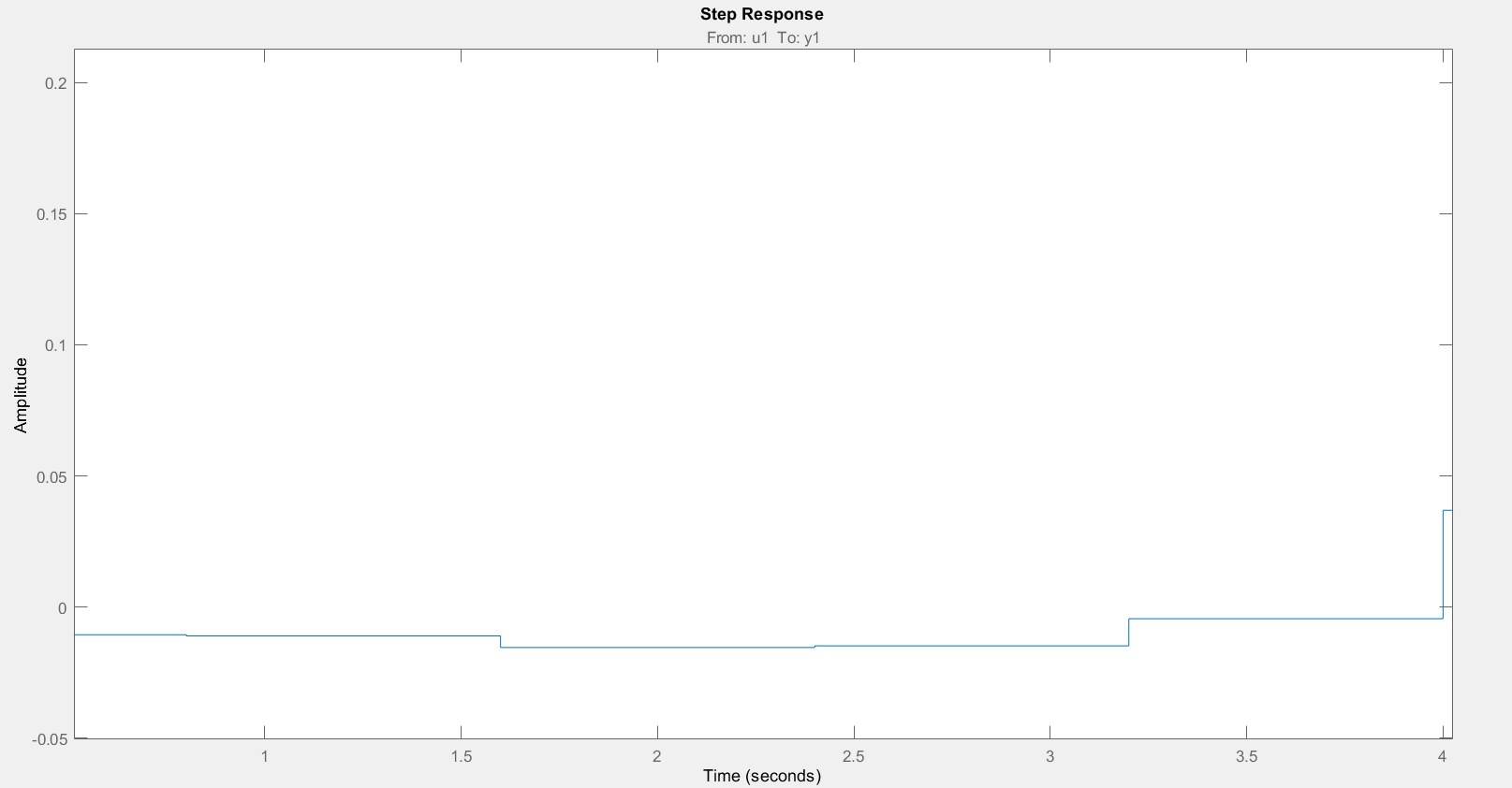
Initially we see an inverse response (figure 3.5, and figure 3.7), this is a characteristic of a **zero**. There are no maxima/minima/oscillations in the step response, neither there is much sluggishness (only a delay is there). So for simplicity we can assume that the denominator is either a first order or second order polynomial. Therefore, it is a **first** or **second** order process with **a** **zero** and a **delay of 4 units**. Note that a delay of 4 units implies inputs only upto u[k-5] will affect the output at kth instant!



*Figure 3.5: The finite impulse response model*



*Figure 3.6: Step response obtained from the FIR model*

**

*Figure 3.7: Initial inverted response*

## Part d) Parameterized model

### First try: d = 5, m = 5, n =1

Fit to estimation data: 54.21%

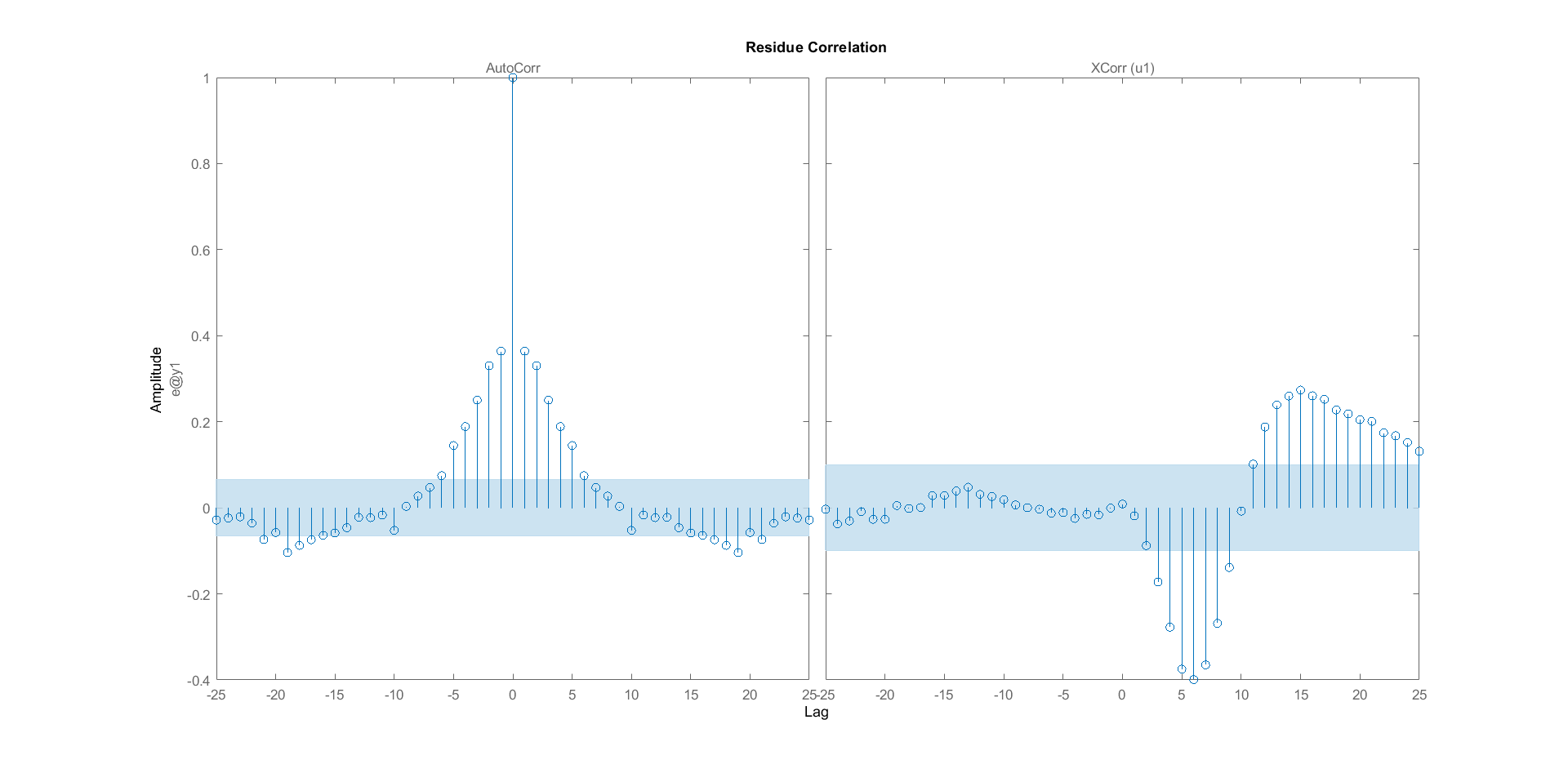
FPE: 2.108, MSE: 2.102

We see that the fit % is not very good.

## Part e) Testing for underfit and over fit

To test for underfit we examine the residuals. Residuals should not be correlated among themselves (auto correlation) or correlated to the input (cross correlation). (Note that correlated to the input will take care of correlation to the output, because output itself is correlated to input.)

### First try: d = 5, m = 5, n =1



*Figure 3.7 Residual correlations. (obtained using ‘resid’ routine)*

We see that the auto correlations as well as the cross correlations are significant at multiple lags. So this model is an underfit!

### Second try: d = 5, m = 1, n =2

Let’s fit a model following second order dynamics.

Fit to estimation data: 64.54%

FPE: 1.266, MSE: 1.261

Both MSE and fit % have improved!

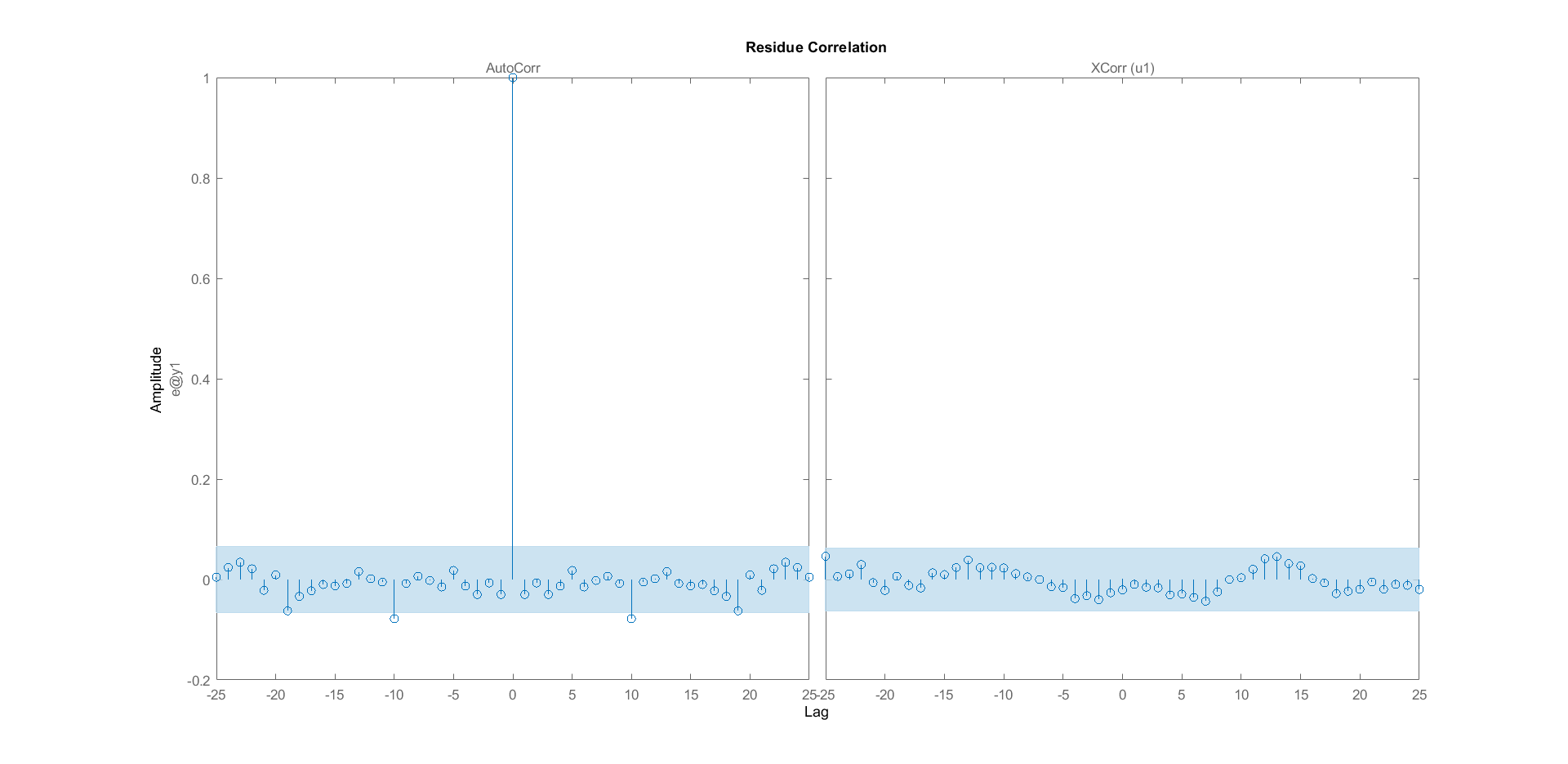
Let’s examine the auto-correlations. From figure 3.8 we see that the cross and auto-correlation of residuals at all non-zero lags are zero!

This means that the new model is **not an underfit**!

Parameter estimates:

None of the confident intervals contain zero in them,

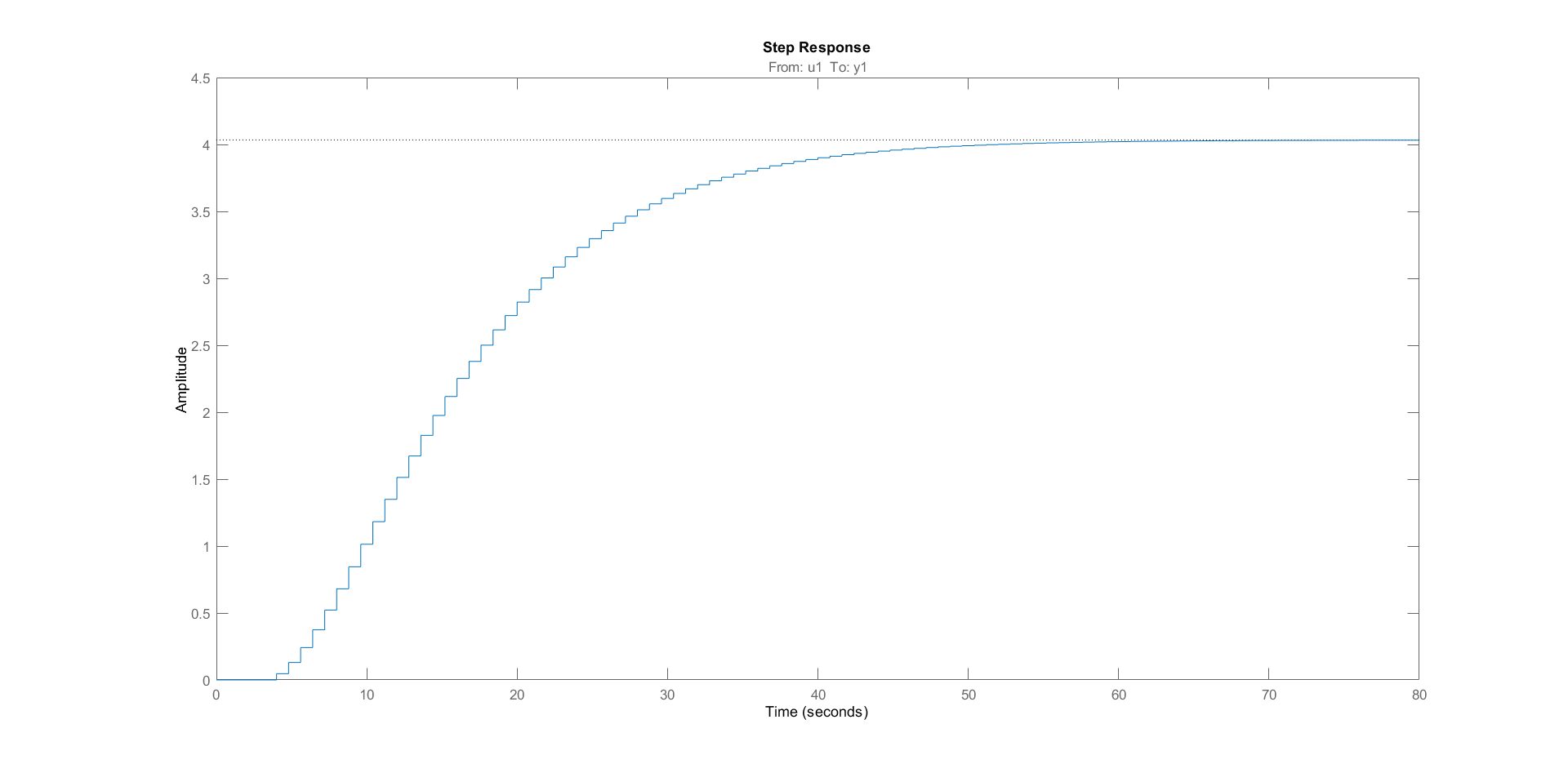
=> All parameters of the model are significant and hence the model is **not an overfit**!



*Figure 3.8 Residual correlations of the new model. (obtained using ‘resid’ routine)*

Gain comparison with non-parametric model

In part c) we obtained a gain of 3.96. In this case we obtain a gain of 4.03 (refer figure 3.9). Both are close to the actual value of 4.



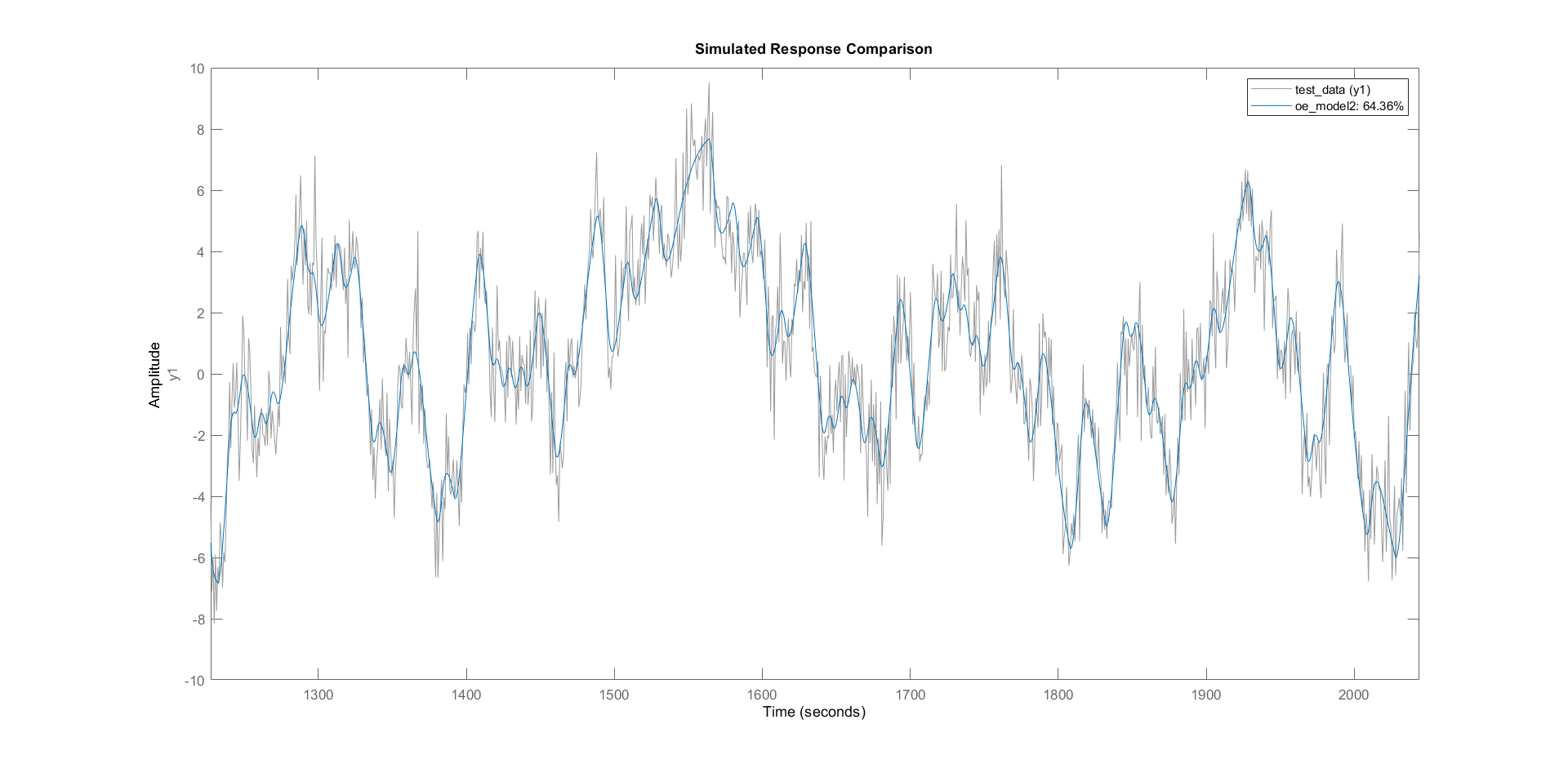
*Figure 3.9: Step response the parametric model*

## Part f) Cross validation

Cross validation was performed using the compare routine (on the test data). The model had a Percentage Fit of about 65%.

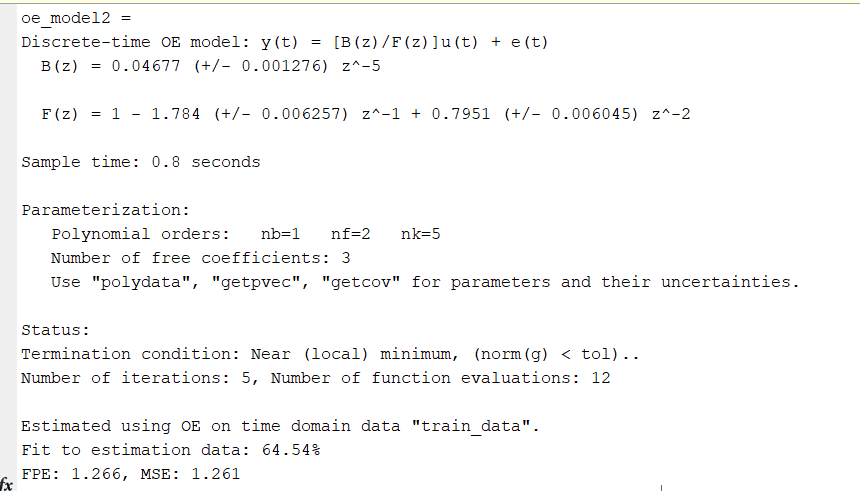
Also, note that data has a certain amount of measurement noise, so we don’t want a model that exactly mimics the data (Because it will end up including the effects of noise too.).

The model captures the trend of the data successfully!



*Figure 3.10: Cross validation using the test data*

**Final model:**

****

*Figure 3.11: Final model details obtained using ‘present’ routine.*

# MATLAB codes

## Question-2

clear; close all;

%% Simulating the actual process

G = tf([2 1],conv(conv(conv([20,1],[15,1]),[4 1]),[0.5 1]),'InputDelay',3);

t = 0:0.5:150;

[Yactual,Tactual] = step(G,t);

yfinal = 1;

%% Krishnaswamy and Sundaresan's method

[val,loc]= min(abs(Yactual-yfinal\*0.3531));

t1 = Tactual(loc);

[val2,loc2] = min(abs(Yactual-yfinal\*0.8531));

t2 = Tactual(loc2);

Dks = 1.3\*t1 - 0.29\*t2;

tauks = 0.67\*(t2-t1);

G\_ks = tf(1,[tauks 1],'InputDelay',Dks);

[Yks,Tks] = step(G\_ks,t);

%% Skogestad's half rule method

% FOPTD

tau\_s = 27.5; D\_s = 13;

G\_sk1 = tf(1,[tau\_s 1],'InputDelay',D\_s);

[Ysk1,Tsk1] = step(G\_sk1,t);

% SOPTD

tau\_1= 20; tau\_2 = 17; D\_s2 = 3.5;

G\_sk2 = tf(1,conv([tau\_1 1],[tau\_2 1]),'InputDelay',D\_s2);

[Ysk2,Tsk2] = step(G\_sk2,t);

%% Least Squares (Frequency Domain)

[MAG,PHASE,W] = bode(G);

mpar = lsqcurvefit(@(mpar,wdata) magpred(mpar,wdata),[1 1 1]',W,squeeze(MAG));

Kp = mpar(1); tau1 = mpar(2); tau2 = mpar(3);

Delay = lsqcurvefit(@(D,wdata) phasepred(D,wdata,Kp,tau1,tau2),1,W,cos(squeeze(PHASE)));

Glsq = tf(Kp,conv([tau1 1],[tau2 1]),'InputDelay',Delay);

[Ylsq,Tlsq] = step(Glsq,t);

%% Compare step responses

% K&S

figure();

plot(Tactual,Yactual,Tks,Yks); xlabel('Time'); ylabel('Response');

title('Krishnaswamy and Sundaresan');

legend('Actual Response','K&S FOPTD approx');

% Half Rule

figure();

title("Skogestad's Half Rule");

subplot(2,1,1);

plot(Tactual,Yactual,Tsk1,Ysk1); xlabel('Time'); ylabel('Response');

legend('Actual Response','FOPTD approx');

subplot(2,1,2);

plot(Tactual,Yactual,Tsk2,Ysk2); xlabel('Time'); ylabel('Response');

legend('Actual Response','SOPTD approx');

% Least Squares in Frequency Domain

figure();

plot(Tactual,Yactual,Tlsq,Ylsq); xlabel('Time'); ylabel('Response');

title('Least Squares Fit');

legend('Actual Response','LSQ SOPTD Approx');

%% Functions for least squares fit

function Mag = magpred(mpar,wdata)

Kp = mpar(1); tau1 = mpar(2);tau2 = mpar(3);

Mag = Kp./sqrt(1+(tau1.\*wdata).^2)./sqrt(1+(tau2.\*wdata).^2);

end

function Phase = phasepred(D,wdata,Kp,tau1,tau2)

Gw = Kp./(1+tau1\*(1j\*wdata))./(1+tau2\*(1j\*wdata));

Phase = cos(phase(Gw)-D\*wdata);

end

## Question-5

clear; close all;

Ts = 0.8;

N = 2555;

%% Getting the data and splitting to train and test

open\_system('model');

out = sim('model');

% Get the input and output

tk = out.ymeas.time;

uk = out.inputs.Data;

yk = out.ymeas.Data;

% Splitting the data: 60% train, rest 40% test

% tf is always in terms of deviation variables so we dont need to do any

% additional processing

ntrain = 0.6\*N;

dataset = iddata(yk,uk,Ts);

train\_data = dataset(1:ntrain);

test\_data = dataset(ntrain+1:end);

%% Non parametric estimation

options = impulseestOptions;

options.Advanced.AROrder = 0;

fir\_model = impulseest(train\_data ,options);

[Y,T] = impulse(fir\_model);

figure;

impulse(fir\_model);

figure;

step(fir\_model);

%% Parametric Estimation

d = 4;

% One zero because of inverse response

nb = 1;

% Looks overdamped so second order but can also be first. Trying both

nf1 = 1; nf2 = 2;

% First 4 responses are negligible

nk = 5;

oe\_model1 = oe(train\_data,[nb nf1 nk]);

figure;

resid(oe\_model1,train\_data);

oe\_model2 = oe(train\_data,[nb nf2 nk]);

figure;

resid(oe\_model2,train\_data);

% nf = 2 is not an underfit

% Check for overfit

present(oe\_model2);

%% Cross Validation

figure;

compare(oe\_model2,test\_data);

figure;

step(oe\_model2);