```
% %ELEC4632 lab1
% prelab %
clc
close all
clear
% data generation
N = (100-0)/0.1;
t = linspace(0,100,N);
noise = 0.4*rand(1,N) - 0.2; % generate noise from -0.2 to 0.2
y1 = sin(0.02*pi*t)+noise;
y2 = cos(0.02*pi*t)+noise;

y1_new = y1(200:end);
y2_new = y2(200:end);
t_new = t(1:length(y1_new));
data_new = [t_new; y1_new; y2_new;]';
% original data plotting
figure(1)
subplot(2,1,1)
plot(t,y1,'r')
xlim([0 140]);
ylim([-1.5 1.5]);
xlabel({'Time (sec)';'(a)'})
ylabel('Data')
title('Original Data')
grid on
hold on
plot(t,y2,'b')
legend('sin(0.02\pit)','cos(0.02\pit)');
hold off
% cut-off data plotting
subplot(2,1,2)
plot(t_new,y1_new,'g');
xlim([0 140]);
ylim([-1.5 1.5]);
xlabel({'Time (sec)';'(b)'})
ylabel('Data')
title('Cut-off Data')
grid on
hold on
plot(t_new,y2_new,'Color',[0.6 0.7 0.8])
legend('sin(0.02\pit)','cos(0.02\pit)');
hold off
% question 1 %
clear
load SysIdenData_StudentVersion.mat
t = LogData.time;
y_act = LogData.signals(1).values(:,2);
y_actm = LogData.signals(1).values(:,1);
u_act = LogData.signals(2).values;
% find sampling time
Ts = (t(end)-t(1))/(length(t)-1);
```

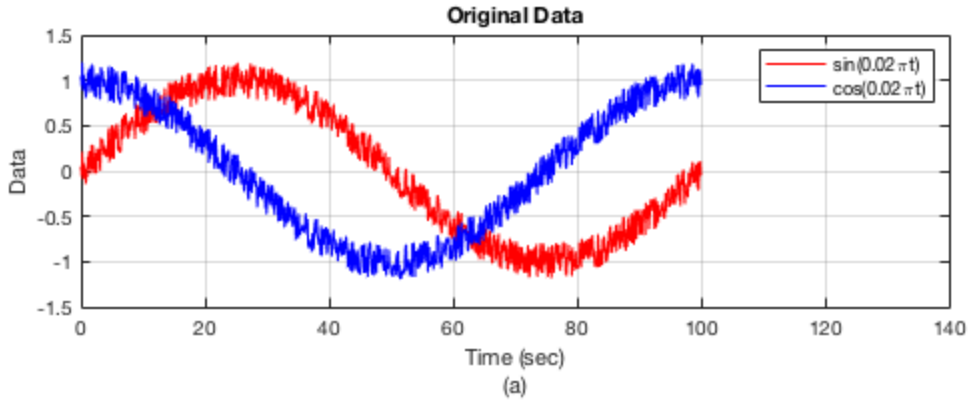
```
%Ts = t(2)-t(1);
fprintf("sampling time according to calculation is %d\n",Ts);
figure(2)
subplot(2,1,1)
hold on
plot(t,y_act,'b');
xlim([0 700]);
ylim([1 4]);
xlabel('Time (sec)');
ylabel('WaterLevel (V)');
title('Actual signal');
grid on
plot(t,y_actm,'r');
legend('Noised-Reduced Output','Measured Output');
hold off
subplot(2,1,2)
plot(t,u_act);
xlim([0 700]);
ylim([1 3]);
xlabel('Time (sec)');
ylabel('Pump voltage (V)');
title('Actual Input Signal');
legend('Actual input');
grid on

% remove input offset
u_offset = u_act(1);
u = u_act - u_offset;
figure(3)
subplot(2,1,2)
plot(t,u)
xlim([0 700]);
ylim([-0.5 0.5]);
xlabel('Time (sec)');
ylabel('Pump Voltage (V)');
title('Actual Offset-Free Input Signal');
legend('Actual Input');
grid on
% remove output offset
count = 0;
i = 1;
while(u_act(i+1) == u_act(i))
    i=i+1;
    count = count + 1;
end
%y_offset = sum(y_act(1:count))/count;
y_offset = mean(y_act(1:count));
y = y_act - y_offset;
subplot(2,1,1);
plot(t,y,'r')
xlim([0 700]);
ylim([-2 1]);
grid on
xlabel('Time (sec)');
```

```
ylabel('Water Level (V)');
title('Actual Offset-Free Output Signal');
legend('Actual Output');

N = round(length(y)/2); % first half
% start from k = 10, k should be greater than 2 otherwise index error
% will occur
k = 10;
phi = zeros(length(k:N),4);

for i = k:N
    phi(i-k+1,1) = y(i-1);
    phi(i-k+1,2) = y(i-2);
    phi(i-k+1,3) = u(i-1);
    phi(i-k+1,4) = u(i-2);
end

theta_hat = inv(phi'*phi)*phi'*y(k:N);
a1 = -theta_hat(1);
a2 = -theta_hat(2);
b1 = theta_hat(3);
b2 = theta_hat(4);
%[a1,a2,b1,b2] = second_order_regression(k,y,u);
H = tf([b1 b2],[1 a1 a2],Ts);
fprintf("Info about second order state space model is below:\n");
sys = ss(H)

figure()
%simulate second half
subplot(2,1,1)
b = [b1 b2];
a = [1 a1 a2];
y_simulate_2nd_Half = filter(b,a,u(N:end));
plot(t(N:end),y_simulate_2nd_Half,'--');
hold on
plot(t(N:end),y(N:end),'r');
xlim([t(N) 700])
ylim([-2 2]);
grid on
xlabel('Time (sec)');
ylabel('Water Level (V)');
legend('Simulated Output','Actual Output');
title('Offset-Free Model Verification (2^{nd} Half)');
hold off
% simulate entire
subplot(2,1,2)
y_simulate_entire_2nd_order = filter(b,a,u);
plot(t,y_simulate_entire_2nd_order,'--');
hold on
plot(t,y,'r')
ylim([-2 2]);
grid on
xlabel('Time (sec)');
ylabel('Water Level (V)');
```

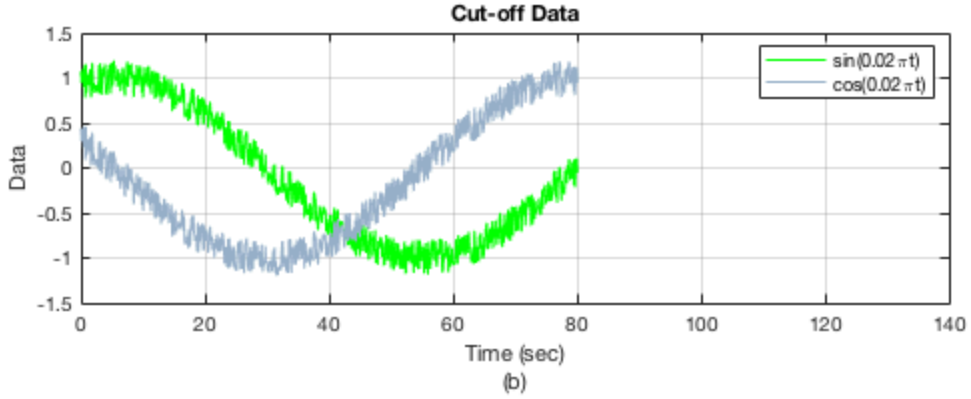
```
legend('Simulated Output','Actual Output');
title('Offset-Free Model Verification (Entire)');
% move on to first order %
% same as before, start from k=10. k should be greater than 2
 otherwise
% index error will occur
k = 10;
phi1 = zeros(length(k:N),2);
%build the phi matrix for first order model
for i = k:N
    phi1(i-k+1,1) = y(i-1);
    phi1(i-k+1,2) = u(i-1);
end
theta_hat1 = inv(phi1'*phi1)*phi1'*y(k:N);
a1 = -theta_hat1(1);
b1 = theta_hat1(2);
H1 = tf(b1,[1 a1],Ts);
fprintf("Info about the first order state space model is below:\n");
sys1 = ss(H1)
y_simulate_firstOrder = filter(b1,[1 a1],u);
figure()
plot(t,y,'r');
hold on
plot(t,y_simulate_firstOrder,'g.');
plot(t,y_simulate_entire_2nd_order,'--b');
xlabel('Time (sec)');
ylabel('Water Level (V)');
xlim([0 700]);
ylim([-1.5 1]);
grid on
title('Comparison of Different Offset-Free Order Models');
legend('Actual Output','1^{st} Order Model Response','2^{nd} Order
 Model Resonse');
% first_order_mse = immse(y_simulate_firstOrder,y);
% second_order_mse = immse(y_simulate_entire_2nd_order,y);
first_order_mse = myMSE(y_simulate_firstOrder,y);
second_order_mse = myMSE(y_simulate_entire_2nd_order,y);
str = {'MSE1 = ',first_order_mse,'MSE2 = ',second_order_mse};
text(5,0.5,str)
```

```
sampling time according to calculation is 7.500000e-01
Info about second order state space model is below:

sys =

  A =
           x1       x2
   x1    1.378  -0.7854
   x2      0.5        0

  B =
         u1
   x1  0.25
   x2     0
```

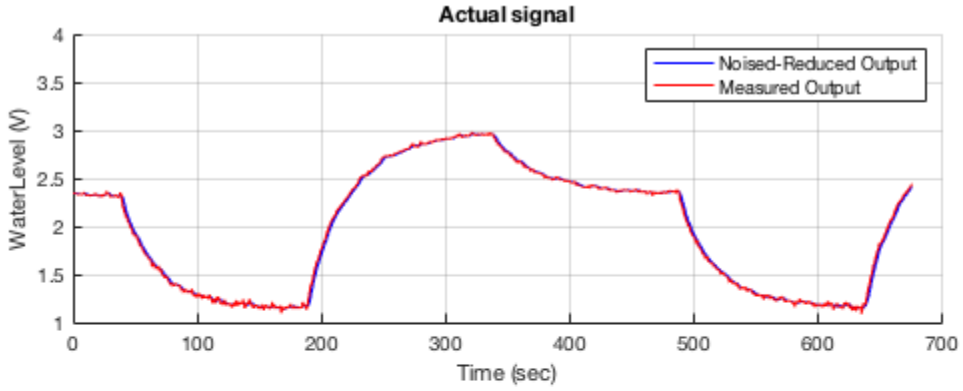
```
  C =
           x1        x2
   y1  -0.02662    0.4109

  D =
       u1
   y1   0

Sample time: 0.75 seconds
Discrete-time state-space model.

Info about the first order state space model is below:

sys1 =

  A =
          x1
   x1  0.9776

  B =
         u1
   x1  0.25

  C =
          x1
   y1  0.2724

  D =
       u1
   y1   0

Sample time: 0.75 seconds
Discrete-time state-space model.
```

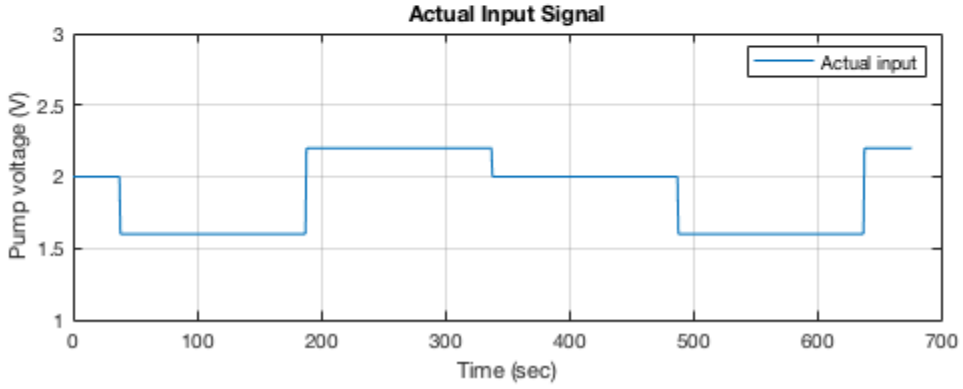

Original Data
sin(0.02πt)
cos(0.02πt)
Data
Time (sec)

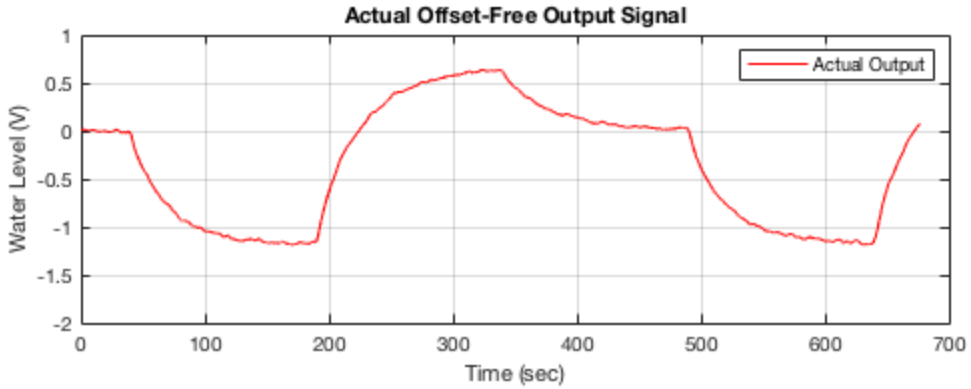
(a)

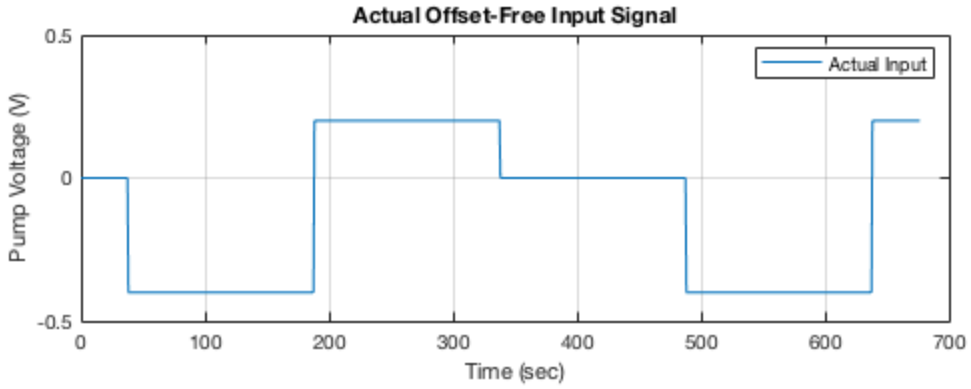

Cut-off Data
sin(0.02πt)
cos(0.02πt)
Data
Time (sec)

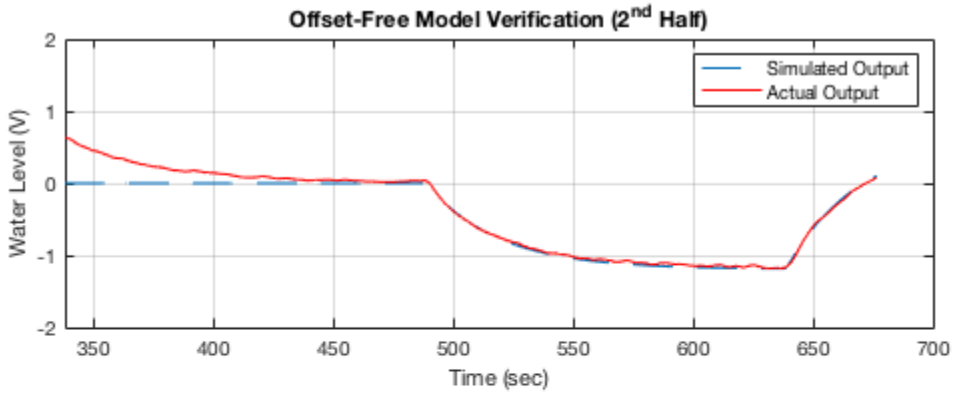
(b)

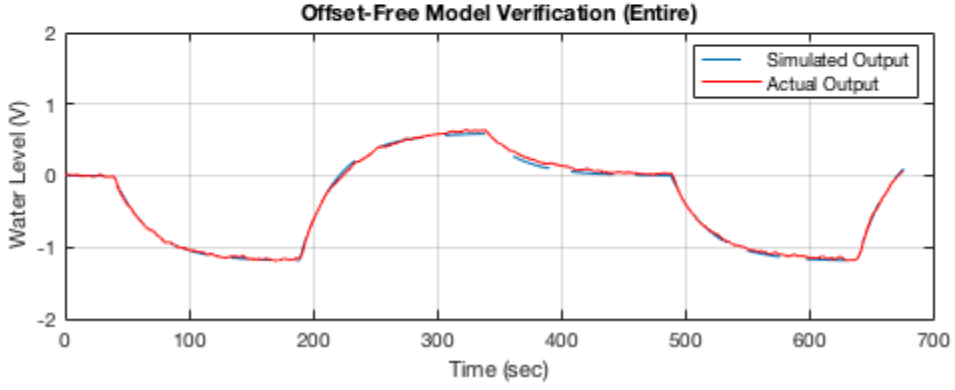

Actual signal
Noised-Reduced Output
Measured Output
WaterLevel (V)
Time (sec)

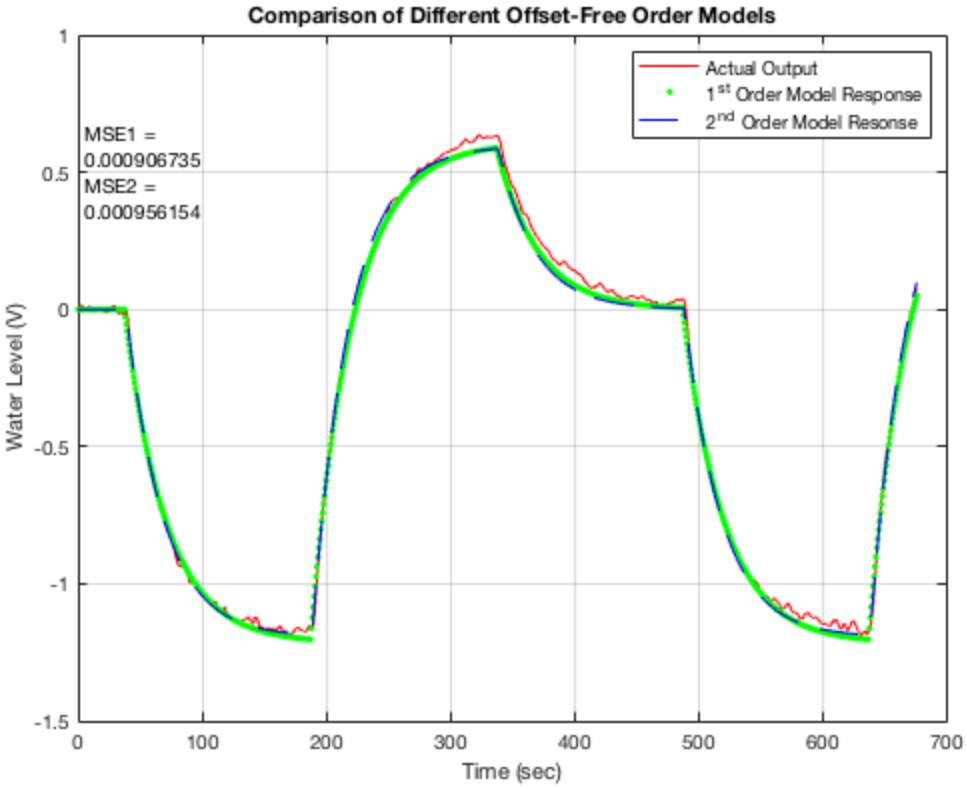

Actual Input Signal
Actual input
Pump voltage (V)
Time (sec)


Actual Offset-Free Output Signal
Actual Output
Water Level (V)
Time (sec)


Actual Offset-Free Input Signal
Actual Input
Pump Voltage (V)
Time (sec)


Offset-Free Model Verification (2nd Half)
Simulated Output
Actual Output
Water Level (V)
Time (sec)


Offset-Free Model Verification (Entire)
Simulated Output
Actual Output
Water Level (V)
Time (sec)



Published with MATLAB® R2018a