ME 494/5594 – Robotics Systems Identification

HW #4: Due Monday, September 26th, 2022

1. **Stepwise Regression:** Using the Multi-Step input data from HW #3, conduct a forward selection OLS.
   a. Show the change in R^2 vs. the number of regressors

1 regressor:	Bias	Angle	Velocity	Voltage		
R^2 :	-1.17E-19	0.0375	-8.15E-06	0.3205		
2 regressors:	Bias(1) + Angle	Angle +Velocity	Velocity +Voltage	Voltage + Bias(1)	Bias(1) +Velocity	Angle +Voltage
R^2 :	0.146	0.038	0.8807	0.3369	6.85E-07	0.3547
3 regressors:	Bias(1) + Angle +Velocity	Angle +Velocity +Voltage	Velocity +Voltage +Bias(1)	Voltage + Bias(1) +Angle		
R^2 :	0.1539	0.8817	0.8812	0.361		
4 regressors:	Bias(1) + Angle +Velocity +Voltage					
R^2 :	0.8819					

   b. Show the f-statistic for each parameter during each iteration (table format may be an easy way to represent)

		F_in = 20			
X_p	X_pj	F_0	F_0 > F_in	(R_p)^2	(R_pj)^2
[bias(1)]	[bias(1), voltage]	66.3761	yes	-1.17E-19	0.3369
[bias(1), voltage]	[bias(1), voltage, angle]	206.3047	yes	0.3369	0.361
[bias(1), voltage, angle]	[bias(1), voltage, angle, velocity]	379.2892	yes	0.361	0.8819

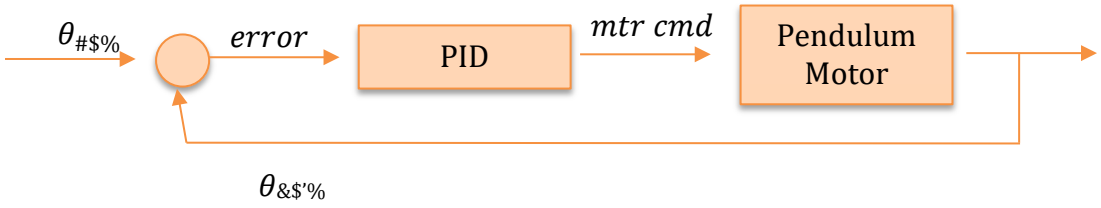
   c. Comment on your findings.

   the change in R^2 vs. the number of regressors shows that lowest average R^2 value is associated with a model with a single regressor. Models with an increasing number of regressors have an increasing average R^2 value. The table comparing the F-statistic's between models with an increasing number of regressors also shows this.

2. **Input Design:** For this problem you will use a premade simulation of a pendulum that has a motor at the hinge location. The pendulum is subject to both friction and fluid drag (linear and quadratic terms). Additionally, a Proportional-Integral-Derivative (PID) controller has been created to control the desired angle of the pendulum (theta = 0 is straight down). Your objective is to create an input design that

excites the control system such that you can effectively estimate the PID controller gains. If you are unfamiliar, a quick google search should be able to get you up to speed on the structure (or reach out and ask).

Thinking about the block diagram for the pendulum and controller. You are trying to estimate ONLY the PID block. Therefore your inputs to the PID should be error (and error_dot, and integral of error), with an output (or dependent variable, Z) of the motor command.



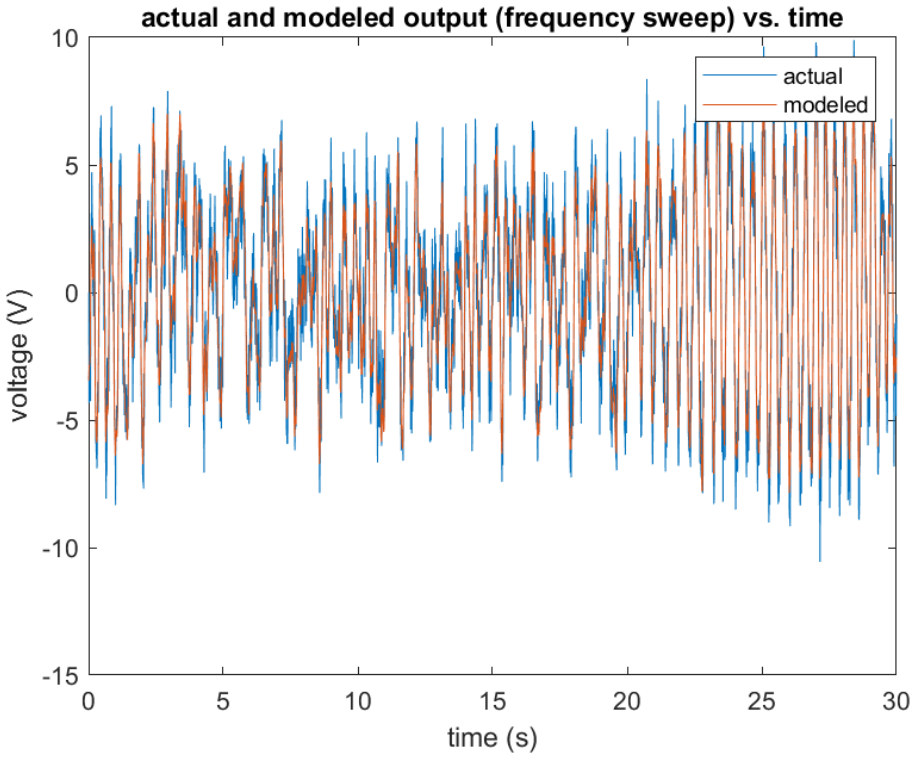
Show your input design, the model structure (with estimated coefficients), R^2, and the model output (y_hat) vs. time along with the actual motor commands vs. time. Comment on this exercise.

Model structure for the frequency sweep is:

$$Y_{hat} = -.0016 + 182.7278error + .0091error_{KI} + 8.0458error_{KD}$$

The frequency sweep R^2 value is:

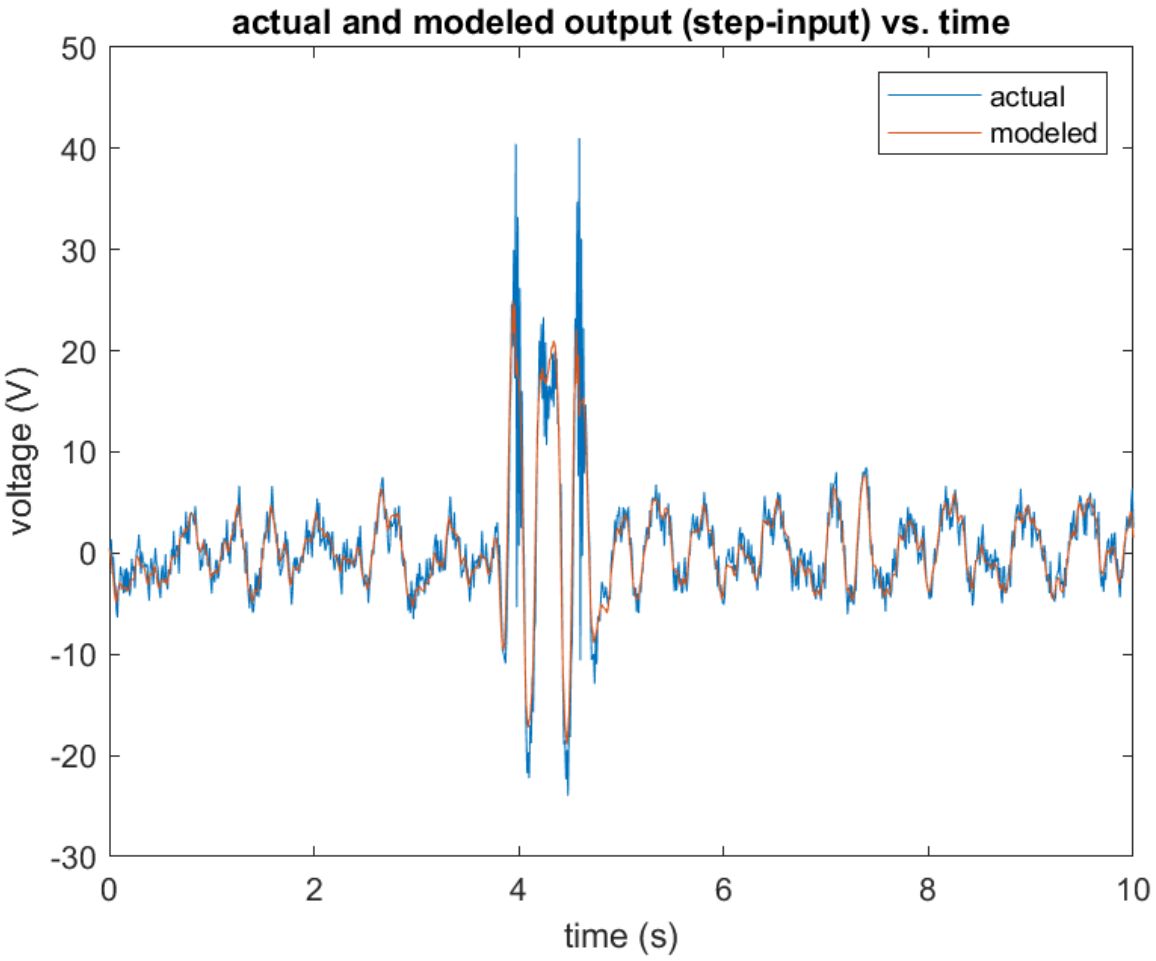
$R^2 = 0.8972$



The modeled output follows the data fairly well. The sweep max is only 3 Hz. If the sweep range is increased to approximately 10 Hz, the overshoot from the actual becomes more pronounced at the higher frequencies while the modeled amplitude remains constant.

3. **Verification:** Create a new data file with a new input design for the pendulum. Your input design needs to include a step input that goes from 0 - ¹" radians, and then back to 0 radians with a step "on" time of 0.5 seconds. Show the model fit compared with the actual motor commands vs. time.

   Based on your results, if you had to stick your model in as the controller (say it needed new gains but the original code was lost), would your model be good enough to stick in? Comment.



The controller does sufficiently produce an output signal. The .5 second pulse width is likely over kill as the pendulum motor will not likely be operating at this speed. If the pendulum does need to operate that fast then the current lowpass filter will not be sufficient.

Code:

```
#1
  load qube_data_multistep
time2 = data(1,:)';
volt2 = data(2,:)';
ang2 = data(3,:)';
vel2 = data(4,:)';
acc2 = deriv(vel2,.01);
N = length(time2);
bias = ones(N,1);

% x2 = [bias];
% x2 = [volt2];
% x2 = [ang2];
% x2 = [vel2];
% x2 = [bias, volt2];
% x2 = [bias, ang2];
% x2 = [bias, vel2];
% x2 = [ang2, vel2];
% x2 = [volt2, vel2];
% x2 = [ang2, volt2];
% x2 = [bias, volt2, ang2];
% x2 = [bias, volt2, vel2];
% x2 = [bias, ang2, vel2];
% x2 = [volt2, ang2, vel2];
% x2 = [volt2, ang2, vel2, bias];

x2 = [ones(N,1), volt2, ang2];
```

```
T_hat2 =(x2'*x2)\x2'*acc2;

Y_hat2 = x2*T_hat2;

disp('The p R squared value is:')

v2 = acc2 - Y_hat2;

R_sq2 = (T_hat2'*x2'*acc2 - length(acc2)*mean(acc2)^2) / (acc2'*acc2 - length(acc2)
* mean(acc2)^2)


x2j = [ones(N,1), volt2, ang2, vel2];

T_hat2j =(x2j'*x2j)\x2j'*acc2;

Y_hat2j = x2j*T_hat2j;

disp('The pj R squared value is:')

v2j = acc2 - Y_hat2j;

R_sq2j = (T_hat2j'*x2j'*acc2 - length(acc2)*mean(acc2)^2) / (acc2'*acc2 -
length(acc2) * mean(acc2)^2)

ssr = 0;

for i = 1:N

   ssr = ((Y_hat2(i) - mean(acc2))^2) + ssr;

end

ssrj = 0;

for i = 1:N

   ssrj = ((Y_hat2j(i) - mean(acc2))^2) + ssrj;

end

s_sq = sum(v2.^2)/(length(v2) -length(T_hat2));


disp('The f-statistic is:')

f = (ssr + ssrj) / s_sq
```

```
#2

fmin = 0.1; % hz

fmax = 3.0;   % hz

dt = .01; % s

T = 30; % s

amp = 3*3.14/180; % rad


% frequency sweep input/output %

[u,t,pf,f] = mksswp(amp,fmin,fmax,dt,T);

N = length(t);

[y,yd, ydd, u_cmd, t] = pend(u, dt, T);


% error %

e_kp = u-y;

e_ki = cumtrapz(T,e_kp);

e_kd = deriv(e_kp,dt);


% model %

x = [ones(N,1), e_kp, e_ki, e_kd];

T_hat = (x'*x)\x'*u_cmd

Y_hat = x*T_hat;


% coefficient of determination %

disp('The step input model validation using frequency sweep data R squared value
is:')

R_sq = (T_hat'*x'*u_cmd - N*mean(u_cmd)^2) / (u_cmd'*u_cmd -
N*mean(u_cmd)^2)
```

```
% Plot %
plot(t,u_cmd,t,Y_hat)
title('actual and modeled output (frequency sweep) vs. time')
xlabel('time (s)')
ylabel('voltage (V)')
legend('actual','modeled')

#3
amp = 3.14/2;
tpulse = .5;
npulse = 1;
tdelay = 4.0;
dt = .01;
T = 10;

% Step-Input input/output %
[u2,t] = mksqw(amp,tpulse,npulse,tdelay,dt,T);
N = length(t);
u = lowpass(u2, .01);
[y, yd, ydd, u_cmd, t] = pend(u , dt, T);

% error %
e_kp = u - y;
e_ki = cumtrapz(T,e_kp);
e_kd = deriv(e_kp,dt);

% model %
x = [ones(N,1), e_kp, e_ki, e_kd];
```

```
T_hat = (x'*x)\x'*u_cmd

Y_hat = x*T_hat;


% coefficient of determination %

disp('The step input model validation using frequency sweep data R squared value is:')

R_sq = (T_hat'*x'*u_cmd - N*mean(u_cmd)^2) / (u_cmd'*u_cmd - N*mean(u_cmd)^2)


% Plot %


% plot(t, u2, t, u) % filtered and unfiltered step

plot(t, u_cmd, t, Y_hat)

title('actual and modeled output (step-input) vs. time')

xlabel('time (s)')

ylabel('voltage (V)')

legend('actual','modeled')
```