Load Models constants for the PZA and Motor

This code sets the parameters for both actuators.

It runs a Step-response analysis in case it is the only exercise required.

It calls the simulation (DualFdbk_PZAwH_MotorDC.slx) and has option to use the data from the simulation on workspace.

```
close all
clear
```

Parameter for PZA hysteresis

```
a=0.38;
beta=0.0335;
gamma=0.0295;
```

Plate constants

```
mp = 0.057; % mass
bp = 17.1912; % damper
```

Constants for the PZA model P235.40

```
ma = 0.94; % kg
ka = 380e-6; %N/m
ba = 0;
kp = 20000000; % N/m
d = 60e-6/1000 ;  % Nominal Travel, 0V to 1000V, 60um
```

Create PZA plant and controller

In case a Simulink license is not available.

```
s = tf('s');

m = ma + mp ;
k = ka + kp ;
b = ba + bp ;

num = [1];
den = [m b k];

P_act = k*d/(m*s^2 + b*s +k);

Kp = 62.6;
Ki = 80;
Kd = 0.03;

P_con = (1/d)*pid(Kp,Ki,Kd);
```

Constants for the NEMA 23 motor

In SI units, the motor torque and back emf constants are equal, that is, K = K_t = K_e.

(Ke) electromotive force constant 0.04 V/rad/sec

(Kt) motor torque constant 0.04 Nm/Amp

```
J = 8.07E-6; % (J)     moment of inertia of the rotor  kg.m^2
b = 3.51E-6; %(b)     motor viscous friction constant   Nms
K = 0.4;
R = 0.49; %  (R)     electric resistance  ohm
L = 2.5E-3; % (L)  electric inductance   H

Kp = 36.95; %7.28;
Ki = 671.02; %62.31;
Kd = -0.07; %0;
N = 521.98; %0;
```

Create Motor plant and controller

```
P_motor = 1/(s*((J*s+b)*(L*s+R)+K^2));
M_con = pid(Kp, Ki, Kd, 1/N);
```

Run Simulation

```
DualFdbk_PZAwH_MotorDC
sim('DualFdbk_PZAwH_MotorDC')
```

Closed-loop TF for PZA, motor and Dual architecture

```
t = 0:0.001:1;

motor_cl = feedback(M_con*P_motor,1);

pzo_cl = feedback(P_con*P_act, +1);

dual_cl = minreal(motor_cl + pzo_cl + motor_cl*pzo_cl) / ((1 + motor_cl)*(1 + pzo_cl));
```

Plots

Step response plots with the characteristic info available for each and display the info.

```
figure(5); clf
h = stepplot(pzo_cl);
grid on
h.showCharacteristic('PeakResponse')
h.showCharacteristic('RiseTime')
h.showCharacteristic('SettlingTime')
h.showCharacteristic('TransientTime')
h.showCharacteristic('SteadyState')
set(findall(gcf,'type','line'),'linewidth',1.5);
title('Closed-loop Step-response - PZA PI235.40')
```

```
stepinfo(pzo_cl)

figure(6); clf
h = stepplot(motor_cl);
grid on
h.showCharacteristic('PeakResponse')
h.showCharacteristic('RiseTime')
h.showCharacteristic('SettlingTime')
h.showCharacteristic('TransientTime')
h.showCharacteristic('SteadyState')
set(findall(gcf,'type','line'),'linewidth',1.5);
title('Closed-loop Step-response - DC motor')
stepinfo(motor_cl)

figure(10); clf
h = stepplot(dual_cl);
grid on
h.showCharacteristic('PeakResponse')
h.showCharacteristic('RiseTime')
h.showCharacteristic('SettlingTime')
h.showCharacteristic('TransientTime')
h.showCharacteristic('SteadyState')
set(findall(gcf,'type','line'),'linewidth',1.5);
title('Step-response - Dual system architecture')
stepinfo(dual_cl)
```

Run after the Simulation ONLY

Section to use the data sent from Simulink to Workspace after the simulation - if needed.

```
% % After running the simulation
%
%
% Ref = out.simout.Data(:,2);
% Err = out.simout.Data(:,1);
% Pos = out.simout.Data(:,3);
%
% figure(15); clf
% plot(out.simout.Time, Pos, 'b')
% hold on
% plot(out.simout.Time, Ref, 'k--')
% plot(out.simout.Time, Err, 'r')
% grid on;
% legend('Pos','Reference', 'error')
% title("Simulation data")
```