Problem 1

```
clear all
clc;

%% Problem 1(a)
% School Linux
%run ~/Desktop/rvctools/startup_rvc.m
% Personal laptop
%run C:\Users\Dustan\Desktop\rvctools\startup_rvc.m

%define the robotics toolbox Puma 560 arm
mdl_puma560;

%set the Coulomb friction terms to zero to help with numerical simulation
p560 = p560.nofriction;

%load the torque profile and open the simulink model
load puma560_torque_profile.mat
out = sim('sl_puma_hw6.slx');

%On my Computer
q = out.get('q_sim');
qd = out.get('qd_sim');
qdd = out.get('qdd_sim');
t_sim = out.get('t_sim');

%% Problem 1(b)
%D(q)*qdd + C(q,qd)*qd + G(q) = Tau
q0 = [0; 0; 0; 0; 0; 0];
qd0 = [0; 0; 0; 0; 0; 0];
%qdd0 = p560.accel(q0, qd0, torque(1,:));
% D0 = p560.inertia(q0.'); %inertia matrix
% C0 = p560.coriolis(q0.',qd0.'); %coriolis matrix
% G0 = p560.gravload(q0.').';
[t,x] = ode45(@p560_dyn_eqns, [0, 10], [q0, qd0],[],p560,torque,time);
q_ode45 = x(:,1:6);
qd_ode45 = x(:,7:12);
for i = 1:length(t)
    qdd_ode45(i,:) = p560.accel(q_ode45(i,:),qd_ode45(i,:),...
        interp1(time,torque,t(i)));
end

%% Problem 1(c) Make Plots
figure(1)
for i = 1:6,
    subplot(3,2,i)
    hold on
    plot(t_sim,q(:,i)*180/pi)
    plot(t,q_ode45(:,i)*180/pi)
    if i == 6 || i == 5,
        xlabel('t')
    end
    ylabel(strcat('q_',num2str(i),' (deg)'))
```

```
end

figure(2)
for i = 1:6,
    subplot(3,2,i)
    hold on
    plot(t_sim,qd(:,i)*180/pi)
    plot(t,qd_ode45(:,i)*180/pi)
    if i == 6 || i == 5,
        xlabel('t')
    end
    ylabel(strcat('qd_',num2str(i),' (deg)'))
end

figure(3)
for i = 1:6,
    subplot(3,2,i)
    hold on
    plot(t_sim,qdd(:,i)*180/pi)
    plot(t,qdd_ode45(:,i)*180/pi)
    if i == 6 || i == 5,
        xlabel('t')
    end
    ylabel(strcat('qdd_',num2str(i),' (deg)'))
end

% The ode45 simulation gave the same output as the simulink simulation with
% a little bit higher fidelity.

function [xp] = p560_dyn_eqns(t,x,bot,torque,time)
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here
%Dq'' + Cq' + G = Tau
%bot.accel(q',q,Tau) = q''
%x1 = q
%x2 = q'
%xp = [x1'; x2'] = [q'; q'']
x1 = x(1:6);
x2 = x(7:12);
% D = bot.inertia(x1.');
% C = bot.coriolis(x1.',x2.');
% G = bot.gravload(x1.');
% G = G.';
xp = zeros(12,1);
xp(1:6) = x2;
Tau = interp1(time,torque,t);
%Tau = Tau.';
%xp(7:12) = D\(-C*x2 - G + Tau);
xp(7:12) = bot.accel(x1.',x2.',Tau);
end
```

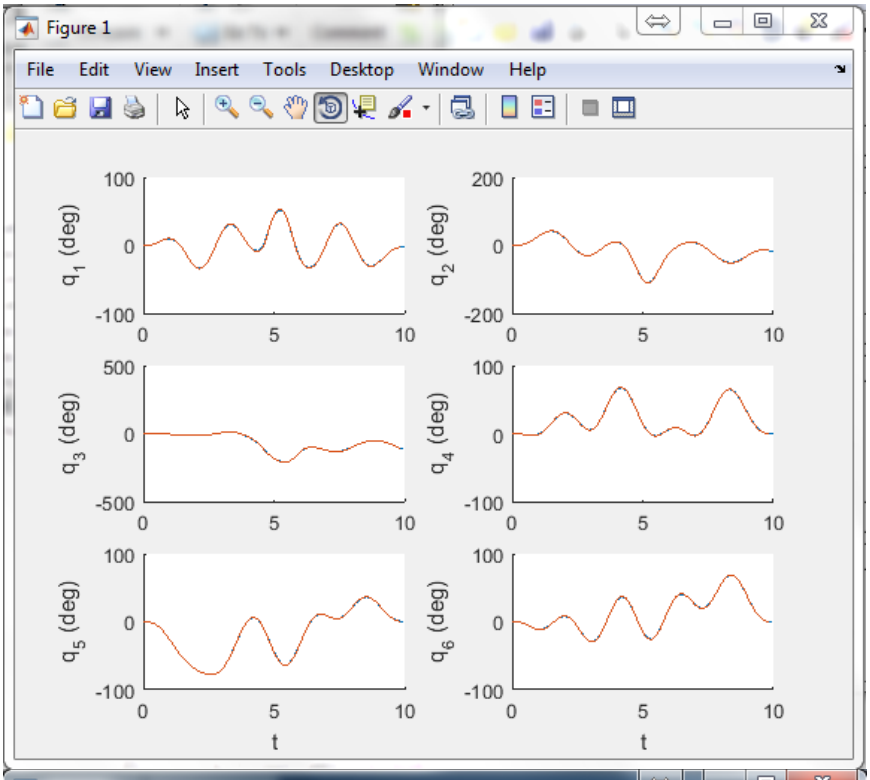


Figure 1
File Edit View Insert Tools Desktop Window Help
q1 (deg)
q2 (deg)
q3 (deg)
q4 (deg)
q5 (deg)
q6 (deg)
t
t

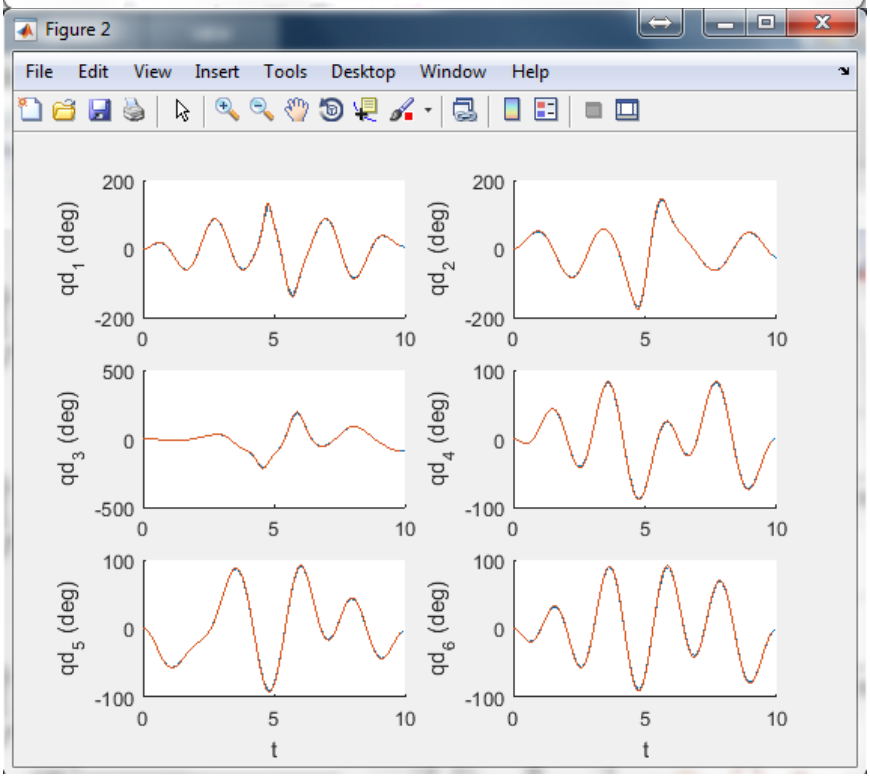
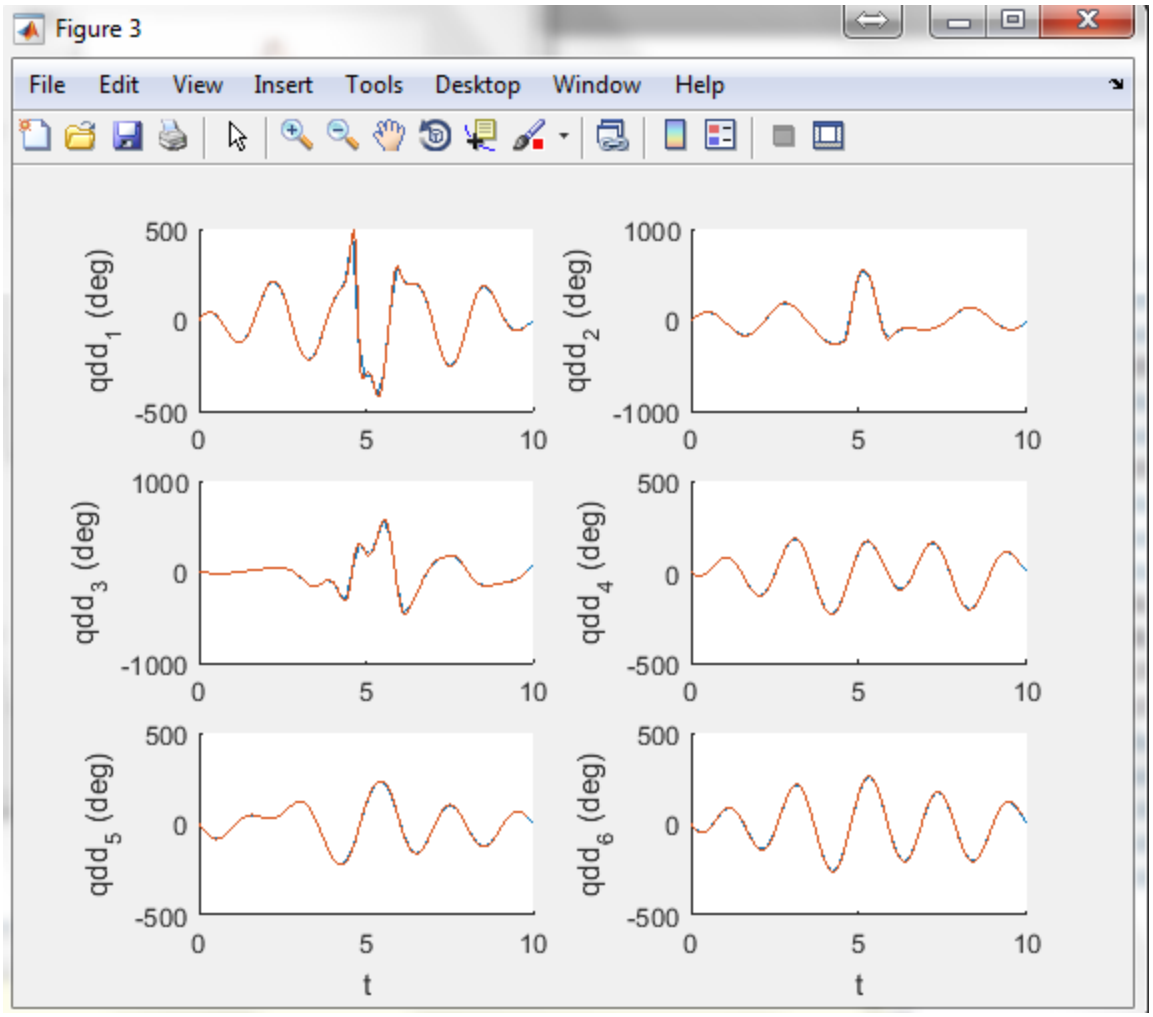


Figure 2
File Edit View Insert Tools Desktop Window Help
qd1 (deg)
qd2 (deg)
qd3 (deg)
qd4 (deg)
qd5 (deg)
qd6 (deg)
t
t



Problem 2

```
clear all
clc;
% School Linux
%run ~/Desktop/rvctools/startup_rvc.m
% Personal laptop
%run C:\Users\Dustan\Desktop\rvctools\startup_rvc.m

%Define Robot
L = 0.4; %link length
m = 1; %link mass
Izz = 0.01; %rotational inertia
grav = [0; 0; 9.81];
I = [Izz, 0, 0; 0, Izz, 0; 0, 0, Izz];

Lk1 = Link('revolute', 'offset', 0, 'd', 0, 'a', L, 'alpha', 0);

Lk2(1) = Link('revolute', 'offset', 0, 'd', 0, 'a', L, 'alpha', 0);
Lk2(2) = Link('revolute', 'offset', 0, 'd', 0, 'a', L, 'alpha', 0);
```

```
Lk3(1) = Link('revolute', 'offset', 0, 'd', 0, 'a', L, 'alpha', 0);
Lk3(2) = Link('revolute', 'offset', 0, 'd', 0, 'a', L, 'alpha', 0);
Lk3(3) = Link('revolute', 'offset', 0, 'd', 0, 'a', L, 'alpha', 0);

Lk3(1).m = m;
Lk3(1).I = I;
Lk3(1).r = [L/2, 0, 0];

Lk3(2).m = m;
Lk3(2).I = I;
Lk3(2).r = [L/2, 0, 0];

Lk3(3).m = m;
Lk3(3).I = I;
Lk3(3).r = [L/2, 0, 0];

bot1 = SerialLink(Lk1, 'name', '1_link');
bot2 = SerialLink(Lk2, 'name', '2_link');
bot3 = SerialLink(Lk3, 'name', '3_link');

bot = [bot1, bot2, bot3];

bot3.nofriction('all');

q = [pi/4; pi/4; pi/4];
qd = [pi/6; -pi/4; pi/3];
qdd = [-pi/6; pi/3; pi/6];
bot3.plot(q.')
view([0 90])

%Find w,alpha, and a
w_iminus1 = [0; 0; 0]; %initial omega
al_iminus1 = [0; 0; 0]; %inital alpha
ae_iminus1 = [0; 0; 0]; %initial a_e
z_iminus1_0 = [0; 0; 1];

w = zeros(3);
al = zeros(3);
ac = zeros(3);
for i = 1:3,
    T = bot(i).fkine(q(1:i));
    R_0_i = T(1:3,1:3)'; %rotation from frame 0 to i
    R_iminus1_i = [...
        cos(q(i))   sin(q(i))   0;...
        -sin(q(i))  cos(q(i))   0;...
        0           0           1]; %rotation from frame i-1 to i
    w_i = R_iminus1_i*w_iminus1 + R_0_i*z_iminus1_0*qd(i);
    w(i,:) = w_i.';
    al_i = R_iminus1_i*al_iminus1 + R_0_i*z_iminus1_0*qdd(i) +...
        cross(w_i, R_0_i*z_iminus1_0*qd(i));
    al(i,:) = al_i.';
    ae_i = R_iminus1_i*ae_iminus1 + cross(al_i,[L;0;0]) +...
        cross(w_i,cross(w_i,[L;0;0]));
    ac_i = R_iminus1_i*ae_iminus1 + cross(al_i,[L/2;0;0]) +...
```

```
        cross(w_i,cross(w_i,[L/2;0;0]));
    ac(i,:) = ac_i';
    w_iminus1 = w_i;
    al_iminus1 = al_i;
    ae_iminus1 = ae_i;
end

f_i_pl_1 = [0; 0; 0];
tau_i_pl_1 = [0; 0; 0];
j = 3;

while j>0,
    R_i_pl_1_i = [...
        cos(q(j))   sin(q(j))   0;...
        -sin(q(j))  cos(q(j))   0;...
        0           0           1].'; %rotation from frame i+1 to i
    f_i = R_i_pl_1_i*f_i_pl_1 + m*ac(j,:).' - m*grav;
    tau_i = R_i_pl_1_i*tau_i_pl_1 - cross(f_i,[L/2;0;0]) +...
        cross(R_i_pl_1_i*f_i_pl_1,[-L/2;0;0]) + I*al(j,:).' + ...
        cross(w(j,:).',I*w(j,:).');
    j = j - 1;
end
tau_i
tau_toolbox = bot3.rne(q.', qd.', qdd.')

%They don't match and I spent many hours on this. I'm excited to see the
solution and learn what I messed up. Because I couldn't get them to match, I
wasn't able to do part (c), but to do it I would implement what we learned in
class about using Newton Euler to find D(q)
```