Robot Control

Midterm Takehome Exam

Contents

Problem 1: the 2D planner robotic arm:

@ Aishwary Jagetia

Notations: For a given variable, x, dx is its time derivative, ddx is 2nd-order derivative.

```
clc
clear all;
close all;
% the following parameters for the arm
I1=10;  I2 = 10; m1=5; r1=.5; m2=5; r2=.5; l1=1; l2=1;

% we compute the parameters in the dynamic model
a = I1+I2+m1*r1^2+ m2*(l1^2+ r2^2);
b = m2*l1*r2;
d = I2+ m2*r2^2;
```

create symbolic variable for x.

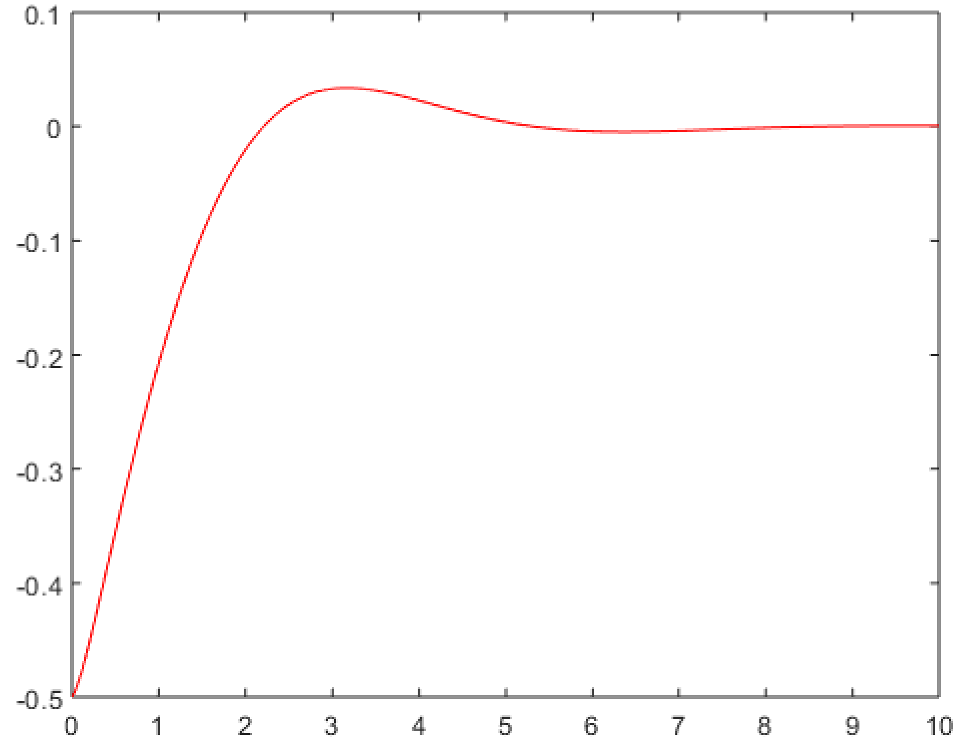
x1 - theta1 x2 - theta2

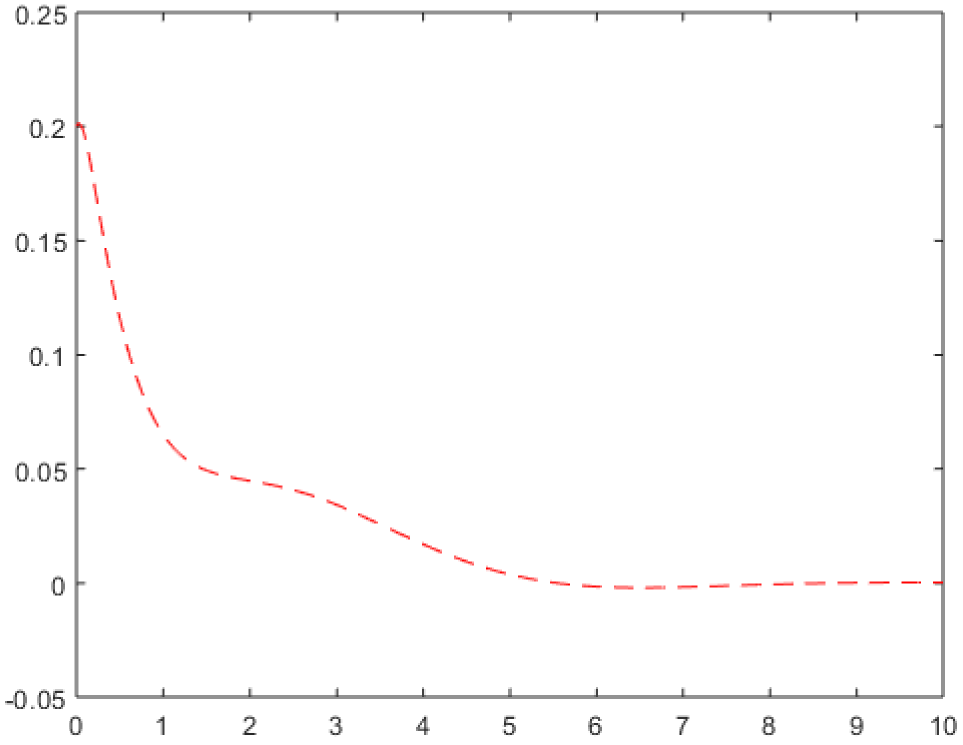
```
    symx= sym('symx',[4,1]);

    global M C
    M = [a+2*b*cos(symx(2)), d+b*cos(symx(2));
        d+b*cos(symx(2)), d];
    C = [-b*sin(symx(2))*symx(4), -b*sin(symx(2))*(symx(3)+symx(4)); b*sin(symx(2))*symx(3),0];
    invM = inv(M);
    invMC= inv(M)*C;

% the options for ode
% initial condition
x0= [-0.5,0.2,0.1,0.1];
global w
w=0.2;
tf=10;
```

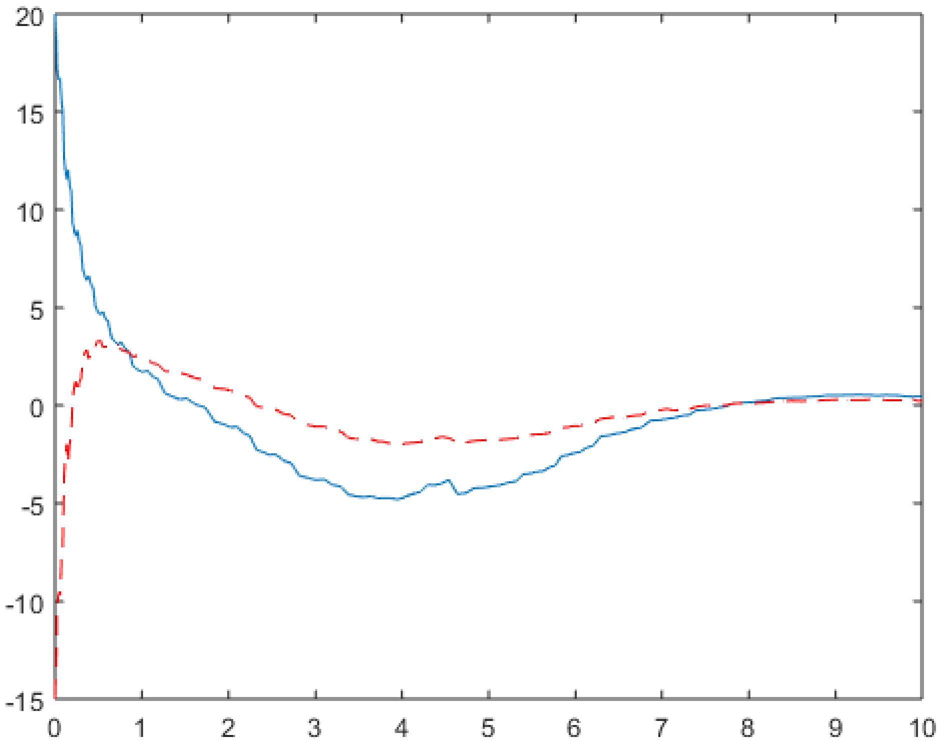
Implement the PD control for set point tracking.

```
xf = [0, 0, 0, 0];

global torque
torque = [];

options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4, 1e-4]);
[T,X] = ode45(@(t,x) PDControl(t,x),[0 tf],x0, options);
```

```
figure('Name','Theta_1 under PD SetPoint Control');
plot(T, X(:,1),'r-');
hold on

figure('Name','Theta_2 under PD SetPoint Control');
plot(T, X(:,2),'r--');
hold on

figure('Name','Input_PD control');
plot(T, torque(1,1:size(T,1)),'-' );
hold on
plot(T, torque(2,1:size(T,1)),'r--');
hold off
torque=[];
```


0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
1
2
3
4
5
6
7
8
9
10


0.25
0.2
0.15
0.1
0.05
0
-0.05
0
1
2
3
4
5
6
7
8
9
10



Implement the inverse dynamic control.

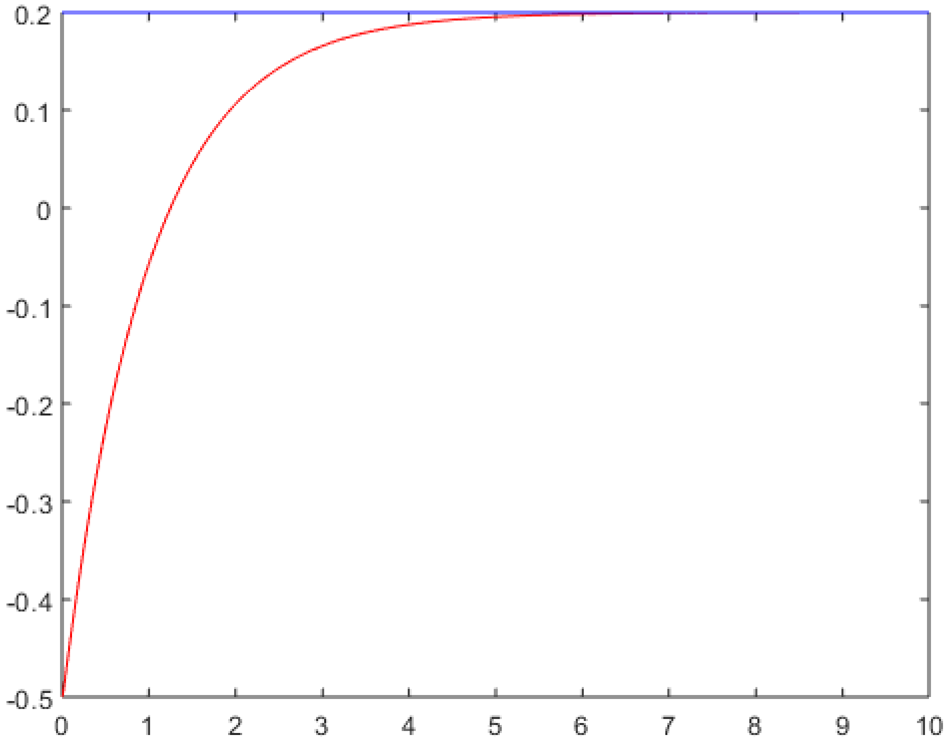
```
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4, 1e-4]);
[T,X] = ode45(@(t,x) inverseDynamicControl(t,x),[0 tf],x0, options);

figure('Name','Theta_1 under Computed Torque Control');
plot(T, X(:,1),'r-');
hold on
plot(T, w*ones(size(T,1),1),'b-');
figure('Name','Theta_2 under Computed Torque Control');
plot(T, X(:,2),'r--');
hold on
plot(T, sin(2*T),'b-');

figure('Name','Computed Torque Control');
plot(T, torque(1,1:size(T,1)),'-' );
hold on
plot(T, torque(2,1:size(T,1)),'r--');
hold off
torque=[];
```

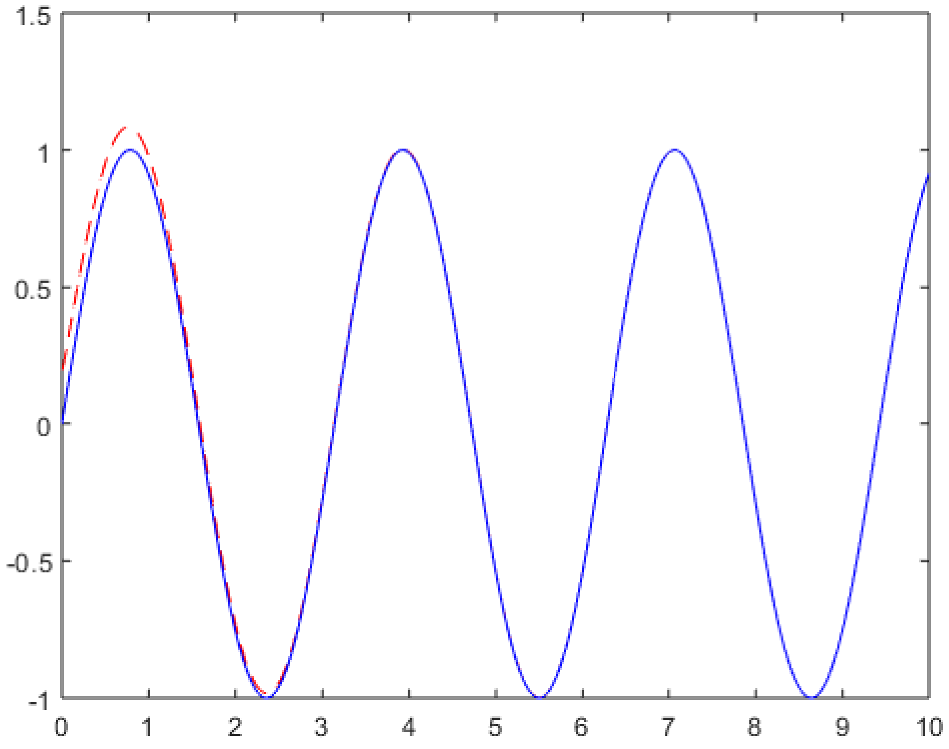
PD Control

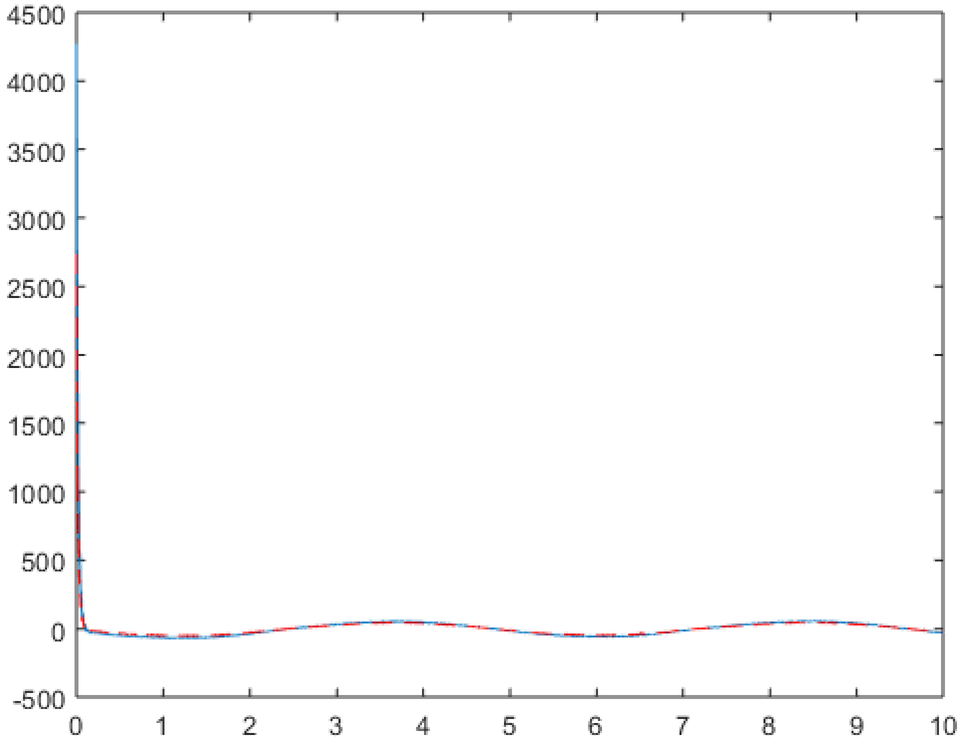
```
function dx = PDControl(t,x)
    theta_d=[0;0]; % [x1d;x2d]
    dtheta_d=[0;0]; % [x1d_dot;x2d_dot]
    theta=x(1:2,1); % [x1;x2]=[x(1);x(2)]
    dtheta=x(3:4,1); % [x1_dot;x2_dot]=[x(3);x(4)]

    global M C
    symx= sym('symx',[4,1]);
    M = subs(M, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    C = subs(C, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    invM = inv(M);
    invMC= inv(M)*C;
    tau = Controler1(theta_d,dtheta_d,theta,dtheta); % Control Law

    global torque
    torque = [torque, tau];

    dx = zeros(4,1);
    dx(1)= x(3); %dtheta1
    dx(2)= x(4); %dtheta2
    dx(3:4) = -invMC* x(3:4) + invM*tau;
end

function tau = Controler1(theta_d,dtheta_d,theta,dtheta)
   P_e = theta_d - theta;
   V_e = dtheta_d - dtheta;
   Kp = 50*eye(2);
   Kv = 50*eye(2);
   tau = Kp*P_e + Kv*V_e;
end
```

Inverse Dynamic Control

```
function dx = inverseDynamicControl(t,x)
    w=0.2;
    theta_d=[w;sin(2*t)]; % [x1d;x2d] Desired trajectory
    dtheta_d=[0;2*cos(2*t)]; % [x1d_dot;x2d_dot]
    ddtheta_d=[0;-4*sin(2*t)]; % [x1d_ddot;x2d_ddot]
    theta=x(1:2,1); % [x1;x2]=[x(1);x(2)]
    dtheta=x(3:4,1); % [x1_dot;x2_dot]=[x(3);x(4)]

    global M C
    symx= sym('symx',[4,1]);
    M = subs(M, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    C = subs(C, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    invM = inv(M);
    invMC= inv(M)*C;

    tau = Controler2(theta_d,dtheta_d,ddtheta_d,theta,dtheta);

    global torque
    torque = [torque, tau];

    dx = zeros(4,1);
    dx(1)= x(3); %dtheta1
    dx(2)= x(4); %dtheta2
    dx(3:4) = -invMC* x(3:4) + invM*tau;
end

function tau = Controler2(theta_d,dtheta_d,ddtheta_d,theta,dtheta)
   P_e = theta_d - theta;
   V_e = dtheta_d - dtheta;
   Kp = 100*eye(2);
   Kv = 100*eye(2);

   global M C
   tau = M*(Kp*P_e + Kv*V_e) + C*dtheta + M*ddtheta_d;
end
```


0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
1
2
3
4
5
6
7
8
9
10


1.5
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8
9
10



Published with MATLAB® R2017b

With Initial Error

Contents

Problem 1: the 2D planner robotic arm:

@ Aishwary Jagetia

Notations: For a given variable, x, dx is its time derivative, ddx is 2nd-order derivative.

```
clc
clear all;
close all;
% the following parameters for the arm
I1=10;  I2 = 10; m1=5; r1=.5; m2=5; r2=.5; l1=1; l2=1;

% we compute the parameters in the dynamic model
a = I1+I2+m1*r1^2+ m2*(l1^2+ r2^2);
b = m2*l1*r2;
d = I2+ m2*r2^2;
```

create symbolic variable for x.

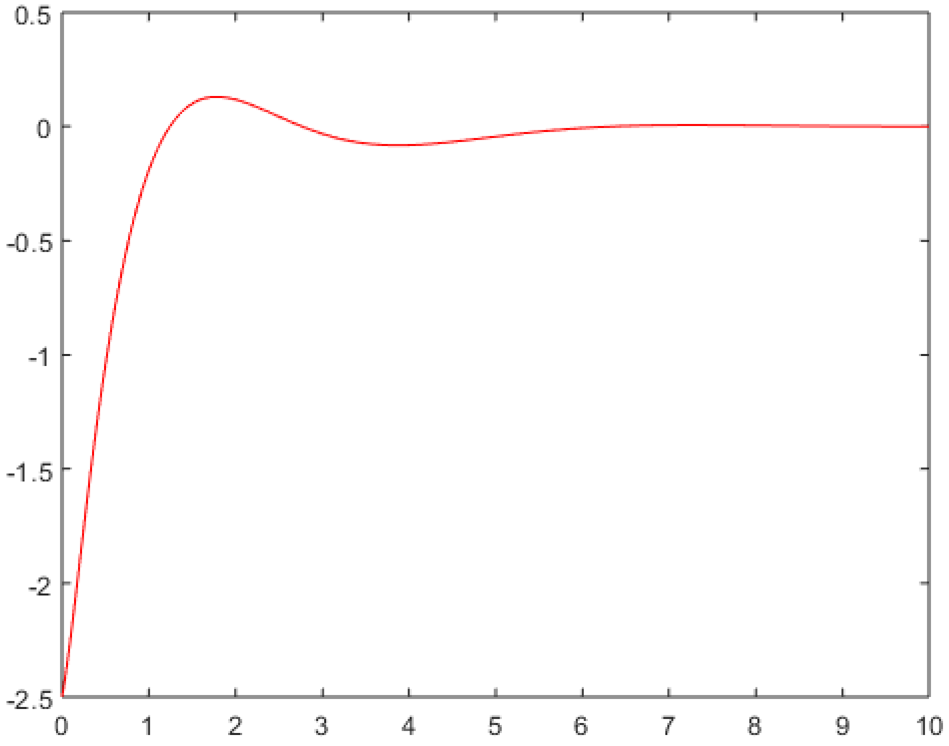
x1 - theta1 x2 - theta2

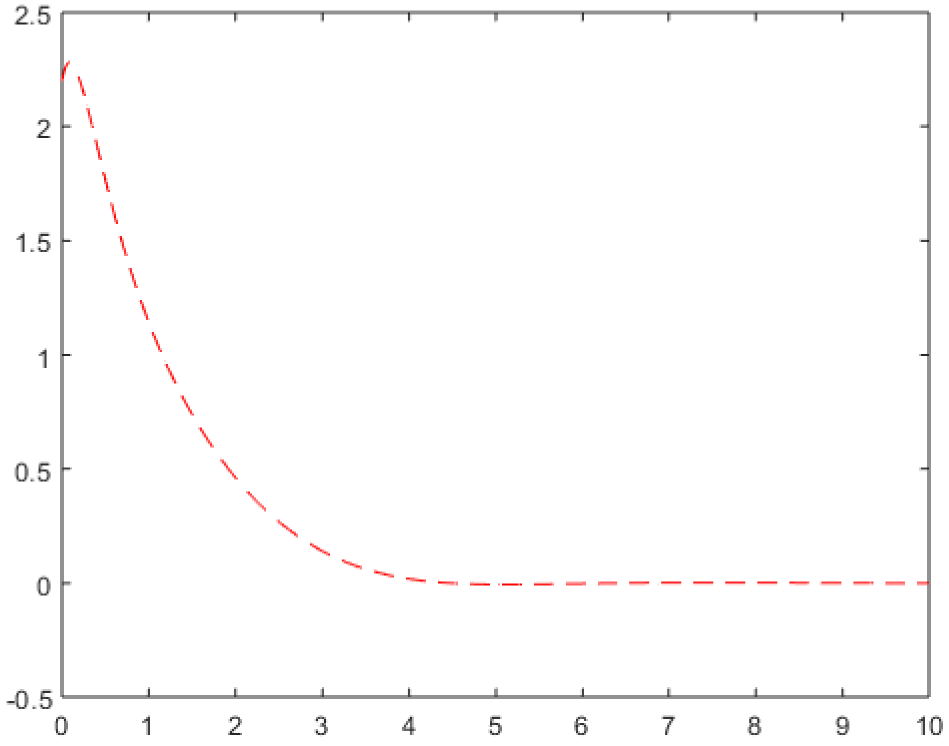
```
    symx= sym('symx',[4,1]);

    global M C
    M = [a+2*b*cos(symx(2)), d+b*cos(symx(2));
        d+b*cos(symx(2)), d];
    C = [-b*sin(symx(2))*symx(4), -b*sin(symx(2))*(symx(3)+symx(4)); b*sin(symx(2))*symx(3),0];
    invM = inv(M);
    invMC= inv(M)*C;

% the options for ode
% initial condition
x0= [-2.5,2.2,2.1,2.1];
global w
w=0.2;
tf=10;
```

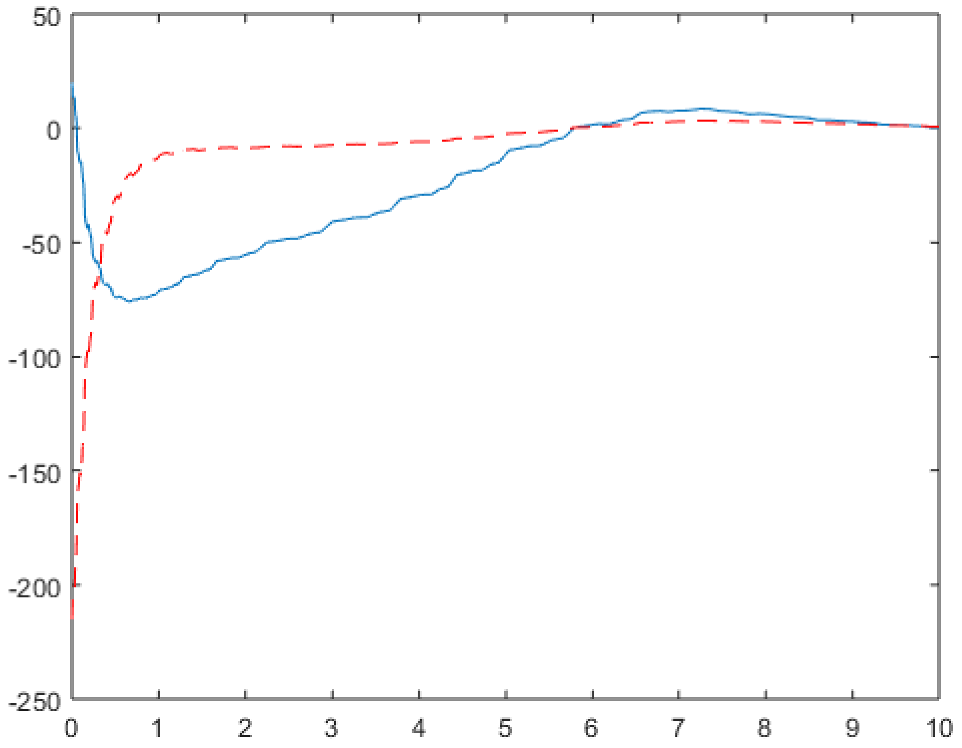
Implement the PD control for set point tracking.

```
xf = [0, 0, 0, 0];

global torque
torque = [];

options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4, 1e-4]);
[T,X] = ode45(@(t,x) PDControl(t,x),[0 tf],x0, options);
```

```
figure('Name','Theta_1 under PD SetPoint Control');
plot(T, X(:,1),'r-');
hold on

figure('Name','Theta_2 under PD SetPoint Control');
plot(T, X(:,2),'r--');
hold on

figure('Name','Input_PD control');
plot(T, torque(1,1:size(T,1)),'-' );
hold on
plot(T, torque(2,1:size(T,1)),'r--');
hold off
torque=[];
```


0.5
0
-0.5
-1
-1.5
-2
-2.5
0
1
2
3
4
5
6
7
8
9
10


2.5
2
1.5
1
0.5
0
-0.5
0
1
2
3
4
5
6
7
8
9
10



Implement the inverse dynamic control.

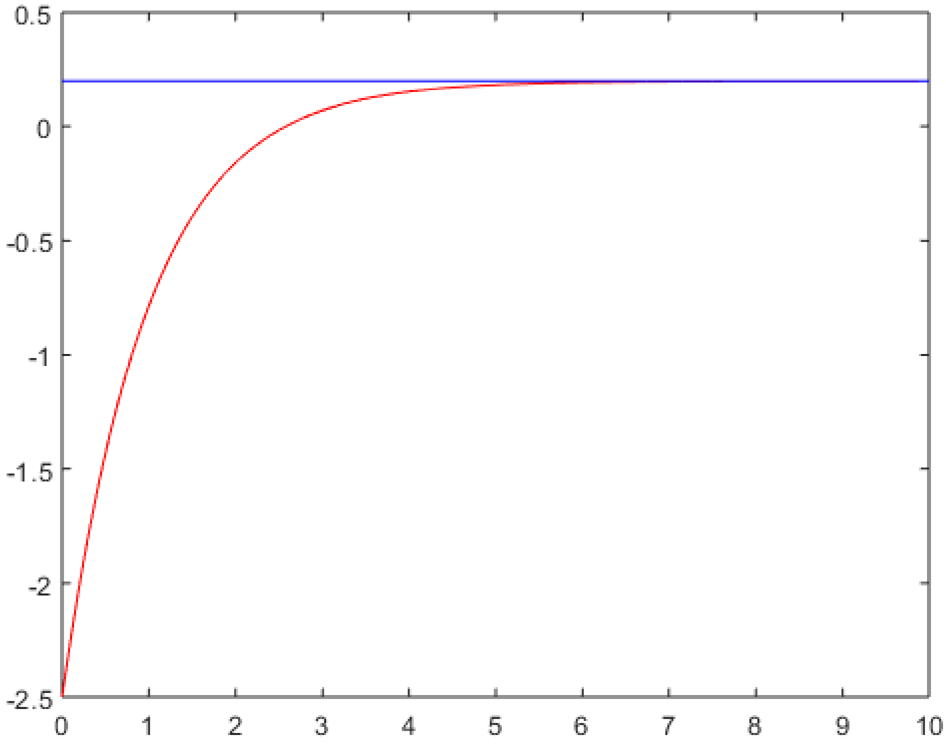
```
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4, 1e-4]);
[T,X] = ode45(@(t,x) inverseDynamicControl(t,x),[0 tf],x0, options);

figure('Name','Theta_1 under Computed Torque Control');
plot(T, X(:,1),'r-');
hold on
plot(T, w*ones(size(T,1),1),'b-');
figure('Name','Theta_2 under Computed Torque Control');
plot(T, X(:,2),'r--');
hold on
plot(T, sin(2*T),'b-');

figure('Name','Computed Torque Control');
plot(T, torque(1,1:size(T,1)),'-' );
hold on
plot(T, torque(2,1:size(T,1)),'r--');
hold off
torque=[];
```

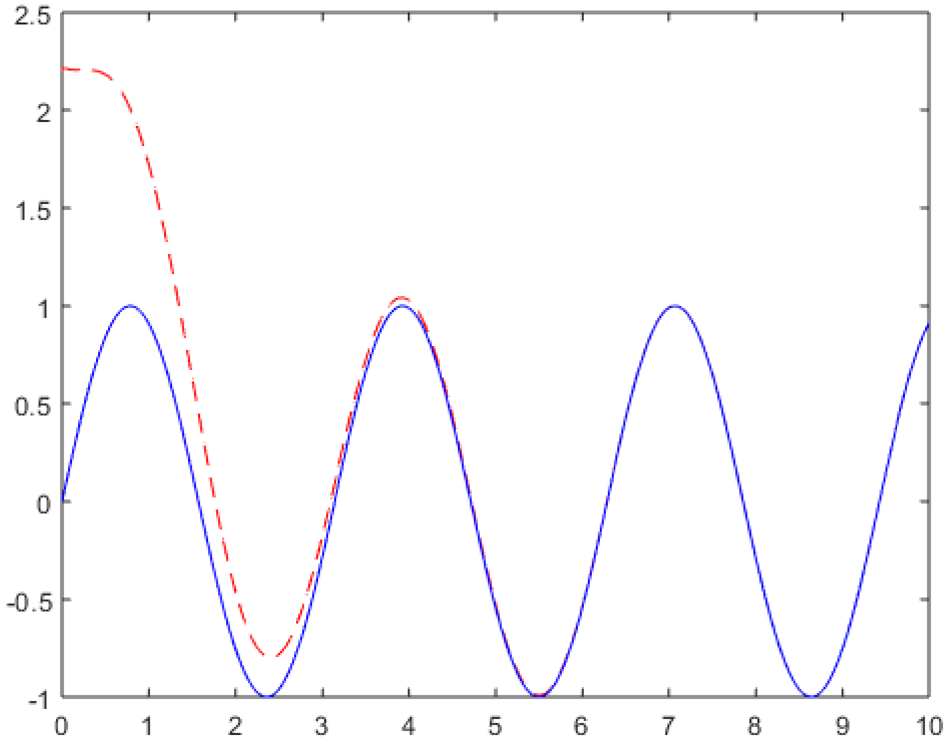
PD Control

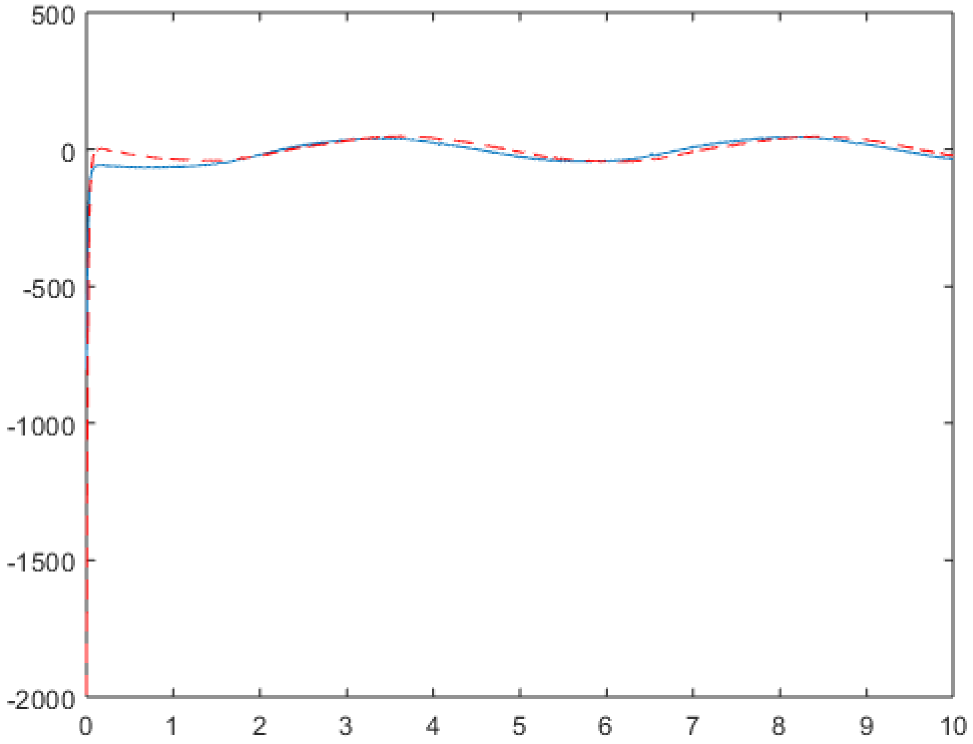
```
function dx = PDControl(t,x)
    theta_d=[0;0]; % [x1d;x2d]
    dtheta_d=[0;0]; % [x1d_dot;x2d_dot]
    theta=x(1:2,1); % [x1;x2]=[x(1);x(2)]
    dtheta=x(3:4,1); % [x1_dot;x2_dot]=[x(3);x(4)]

    global M C
    symx= sym('symx',[4,1]);
    M = subs(M, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    C = subs(C, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    invM = inv(M);
    invMC= inv(M)*C;
    tau = Controler1(theta_d,dtheta_d,theta,dtheta); % Control Law

    global torque
    torque = [torque, tau];

    dx = zeros(4,1);
    dx(1)= x(3); %dtheta1
    dx(2)= x(4); %dtheta2
    dx(3:4) = -invMC* x(3:4) + invM*tau;
end

function tau = Controler1(theta_d,dtheta_d,theta,dtheta)
   P_e = theta_d - theta;
   V_e = dtheta_d - dtheta;
   Kp = 50*eye(2);
   Kv = 50*eye(2);
   tau = Kp*P_e + Kv*V_e;
end
```

Inverse Dynamic Control

```
function dx = inverseDynamicControl(t,x)
    w=0.2;
    theta_d=[w;sin(2*t)]; % [x1d;x2d] Desired trajectory
    dtheta_d=[0;2*cos(2*t)]; % [x1d_dot;x2d_dot]
    ddtheta_d=[0;-4*sin(2*t)]; % [x1d_ddot;x2d_ddot]
    theta=x(1:2,1); % [x1;x2]=[x(1);x(2)]
    dtheta=x(3:4,1); % [x1_dot;x2_dot]=[x(3);x(4)]

    global M C
    symx= sym('symx',[4,1]);
    M = subs(M, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    C = subs(C, [symx(1);symx(2);symx(3);symx(4)], [x(1);x(2);x(3);x(4)]);
    invM = inv(M);
    invMC= inv(M)*C;

    tau = Controler2(theta_d,dtheta_d,ddtheta_d,theta,dtheta);

    global torque
    torque = [torque, tau];

    dx = zeros(4,1);
    dx(1)= x(3); %dtheta1
    dx(2)= x(4); %dtheta2
    dx(3:4) = -invMC* x(3:4) + invM*tau;
end

function tau = Controler2(theta_d,dtheta_d,ddtheta_d,theta,dtheta)
   P_e = theta_d - theta;
   V_e = dtheta_d - dtheta;
   Kp = 100*eye(2);
   Kv = 100*eye(2);

   global M C
   tau = M*(Kp*P_e + Kv*V_e) + C*dtheta + M*ddtheta_d;
end
```


0.5
0
-0.5
-1
-1.5
-2
-2.5
0
1
2
3
4
5
6
7
8
9
10


2.5
2
1.5
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8
9
10



Published with MATLAB® R2017b