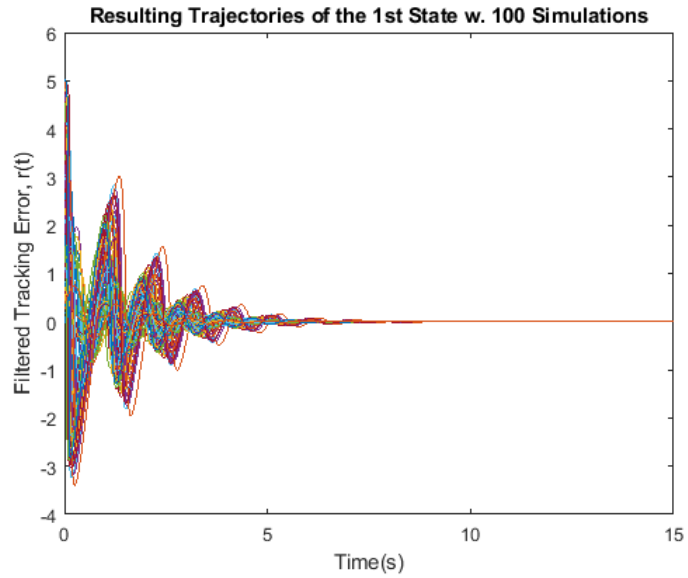
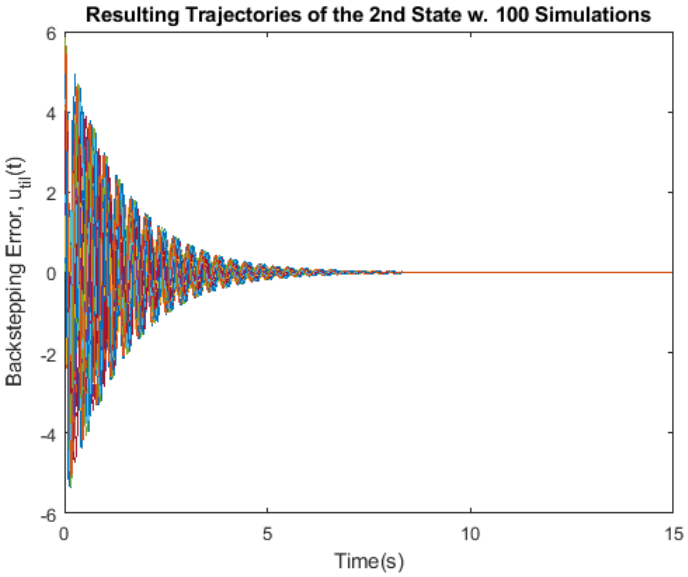
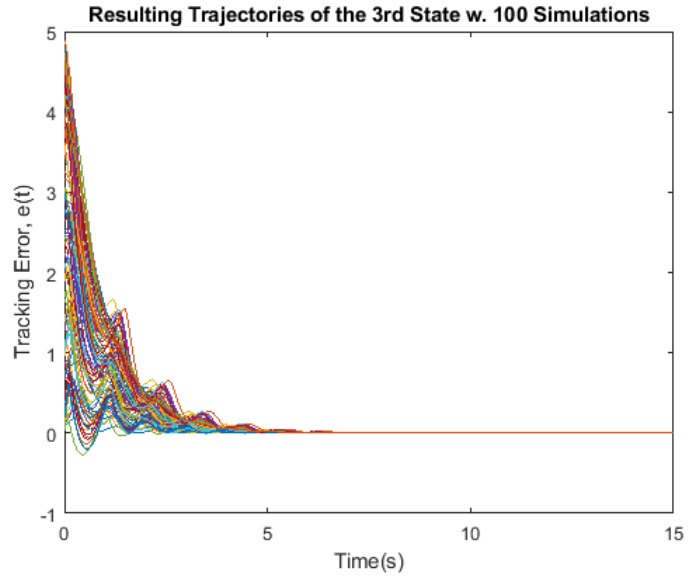
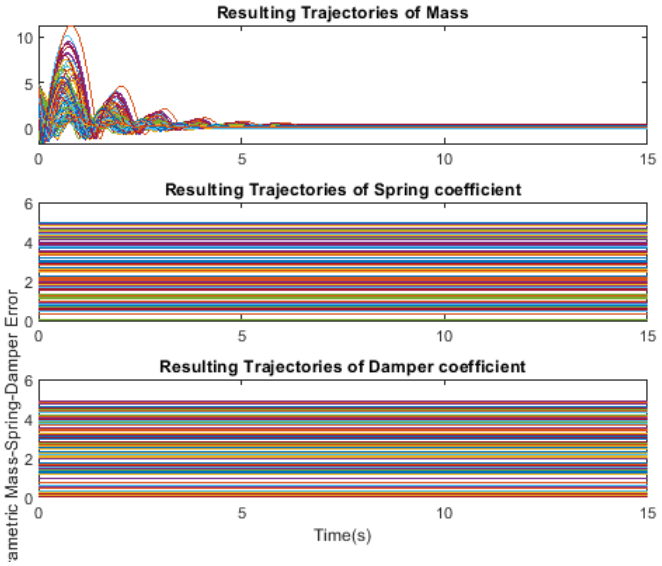
```
clc;clear;

%Define the constants and their conditions
g     = 9.81;           %gravity
a_hat = 5*rand + 0.01; %estimated input coefficient
m_hat = 5*rand + 0.01; %estimated mass
c_hat = 5*rand + 0.01; %estimated damper coefficient
k_hat = 5*rand + 0.01; %estimated spring coefficient
phi   = -5 + (5+5)*rand; %tilt angle has no restriction, could be negative
alpha = 5*rand + 0.01; %design variable, (must be greater than zero)
beta  = 5*rand + 0.01; %design variable, (must be greater than zero)
xd    = -1 + (1+1)*rand;%desired location of the mass, positive constant
theta_hat = [m_hat; c_hat; k_hat];
lambda = 5*rand + 0.01; %design variable, (must be greater than zero)
gamma_a = 5*rand + 0.01;
tspan = [0 15];

%for-loop for 100 simulations with 100 random initial states
for ii = 1:100
    r_0         = 5*rand + 0.01; %random initial condition for 1st state
    u_til_0     = 5*rand + 0.01; %random initial condition for 2nd state
    e_0         = 5*rand + 0.01; %radnom initial condition for 3rd state
    m_til_0     = 5*rand + 0.01;
    c_til_0     = 5*rand + 0.01;
    k_til_0     = 5*rand + 0.01;
    a_til_0     = 5*rand + 0.01; %radnom initial condition for 5th state
    x_dot_0     = 0;
    y_0    = [r_0, u_til_0, e_0, m_til_0, c_til_0, k_til_0, a_til_0, x_dot_0];
    [t,y]   = ode45(@(t,y) sys4(t,y,a_hat, m_hat, c_hat, k_hat, theta_hat,g,alpha, phi, xd, beta,lambda,gamma_a), tspan, y_0); %use ode to solve the system
    %linear plot for each state
    figure(1)
    plot(t,y(:,1));
    title('Resulting Trajectories of the 1st State w. 100 Simulations')
    xlabel('Time(s)')
    ylabel('Filtered Tracking Error, r(t)')
    hold on
    figure(2)
    plot(t,y(:,2));
    title('Resulting Trajectories of the 2nd State w. 100 Simulations')
    xlabel('Time(s)')
    ylabel('Backstepping Error, u_{til}(t)')
    hold on
    figure(3)
    plot(t,y(:,3));
    title('Resulting Trajectories of the 3rd State w. 100 Simulations')
    xlabel('Time(s)')
    ylabel('Tracking Error, e(t)')
    hold on


    figure(4)
    subplot(3,1,1)
    plot(t,y(:,4));
    title('Resulting Trajectories of Mass')
    hold on
    subplot(3,1,2)
    plot(t,y(:,5));
    title('Resulting Trajectories of Spring coefficient')
    hold on
    subplot(3,1,3);
    plot(t,y(:,6));
    title('Resulting Trajectories of Damper coefficient')
    xlabel('Time(s)')
    ylabel('Parametric Mass-Spring-Damper Error')
    hold on

        figure(5)
    plot(t,y(:,7));
    title('Resulting Trajectories of the 5th State w. 100 Simulations')
    xlabel('Time(s)')
    ylabel('Parametric Input Coefficient Error, a_{til}(t)')
    hold on
end
 hold off

 %Define the modified system with 3 states
function dydt = sys4(t, y, a_hat, m_hat, c_hat, k_hat, theta_hat, g, alpha, phi, xd, beta,lambda,gamma_a)
    dydt = zeros(8,1); %create an empty matrix of 3 X 1
    r         = y(1); %Filtered tracking error
    u_til     = y(2); %backstepping error
    e         = y(3); %tracking error of position of the mass
    m_til     = y(4);
    c_til     = y(5);
    k_til     = y(6);
    a_til     = y(7);
```

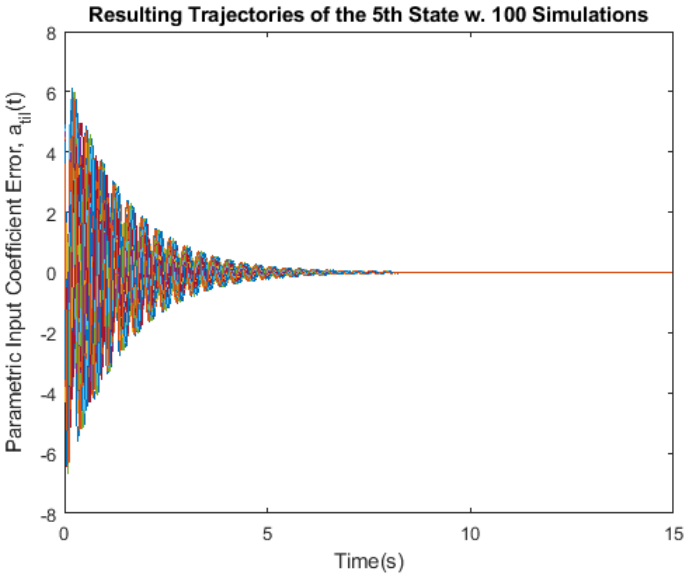
```
    x_dot     = y(8);
    %DESIGNS:(Ud & mu)
    x         = xd - e;   %position of the mass
    e_dot     = -x_dot;
    e_ddot    = dydt(1)-alpha*dydt(3);
    x_ddot    = -e_ddot;

    Y = [alpha*e_dot-g*sin(phi);x_dot;x];
    Y_dot = [alpha*e_ddot; x_ddot; x_dot];
    Gamma = eye(3);

    Ud        = Y.'*theta_hat + u_til + e + beta*r; %backstepping force
    Ud_dot    = Y_dot.'*theta_hat +dydt(2) +dydt(3)+beta*dydt(1); %rate of change of the force

    mu        = Ud_dot + a_hat*Ud -a_hat*u_til + lambda*u_til; %controlled input to the actuator


    m = m_hat + m_til;
    c = c_hat + c_til;
    k = k_hat + k_til;
    a = a_hat + a_til;

    %define the differential equations of the modified dynamic system
    r_dot     = (c*x_dot + k*x - m*g*sin(phi) - Ud + u_til + m*alpha*dydt(3))/m;
    u_til_dot = Ud_dot + a*Ud -a*u_til - mu;
    e_dot     = r - alpha*e;
    theta_til_dot = - Gamma .* Y' .* r;
    m_til_dot = theta_til_dot(1);
    c_til_dot = theta_til_dot(2);
    k_til_dot = theta_til_dot(3);
    a_hat_dot = - gamma_a*(Ud*u_til - u_til^2);
    dydt      = [r_dot; u_til_dot; e_dot; m_til_dot; c_til_dot; k_til_dot; a_hat_dot; x_ddot];
end
```


Resulting Trajectories of the 1st State w. 100 Simulations
Filtered Tracking Error, r(t)
Time(s)


Resulting Trajectories of the 2nd State w. 100 Simulations
Backstepping Error, u_til(t)
Time(s)


Resulting Trajectories of the 3rd State w. 100 Simulations
Tracking Error, e(t)
Time(s)





Published with MATLAB® R2021a