UNIVERSITY OF CALIFORNIA, RIVERSIDE

BOURNS COLLEGE OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

EE 105 Lab 2 Solution

MATLAB as an Engineer's Problem Solving Tool

LUIS FERNANDO ENRIQUEZ-CONTRERAS

Contents

List of Figures

Listings

1 Introduction

2 Pre-Lab

Given the transfer function:

$$H(s) = \frac{36}{s^2 + 3s + 36}$$

We can determine the following values:

$$\begin{aligned}
\omega_n &= 6 \\
G &= 1 \\
\zeta &= \frac{3}{12} = \frac{1}{4} \\
\sigma &= \zeta\omega_n = \frac{6}{4} = 1.5 \\
\omega_d &= \omega_n\sqrt{1-\zeta} \approx 5.2
\end{aligned}$$

The steady state response is given by:

$$\begin{aligned}
H(s) &= \frac{36}{s^2 + 3s + 36} \\
H(s)|_{s=j\omega} &= \frac{36}{36 - \omega^2 + 3j\omega} \\
|H(j\omega)|^2 &= H(j\omega)H(j\omega)^* \\
&= \left(\frac{36}{36-\omega^2+3j\omega}\right)\left(\frac{36}{36-\omega^2-3j\omega}\right) \\
|H(j\omega)| &= \frac{36}{\sqrt{\left(\left(36-\omega^2\right)^2 + 9\omega^2\right)}} \\
\angle H(j\omega) &= \left(\frac{36}{36-\omega^2+3j\omega}\right)\left(\frac{36-\omega^2-3j\omega}{36-\omega^2-3j\omega}\right) \\
&= \frac{36\left(36-\omega^2-3j\omega\right)}{\left(36-\omega^2\right)^2 - 9\omega^2}
\end{aligned}$$

The steady state response is given by the following:

$$y(t) = 0.999\sin\left(0.1t - 0.9163^\circ\right)$$

Given that $x = \begin{bmatrix} y & \dot{y} \end{bmatrix}^T$

$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \ddot{y}(t) \\
\frac{Y(s)}{U(s)} &= \frac{36}{s^2+3s+36} \\
Y(s)\left(s^2+3s+36\right) &= 36U(s) \\
s^2Y(s)+3sY(s)+36Y(s) &= 36U(s) \\
\mathscr{L}^{-1}\left[s^2Y(s)+3sY(s)+36Y(s)\right] &= \mathscr{L}^{-1}[36U(s)] \\
\ddot{y}(t)+3\dot{y}(t)+36y(t) &= 36u(t) \\
\ddot{y}(t) &= 36u(t)-3\dot{y}(t)-36y(t) \\
\ddot{y}(t) &= 36u(t)-3x_2-36x_1 \\
\dot{x}_2 &= 36u(t)-3x_2-36x_1 \\
\dot{x} &= [x_2; 36u(t)-3x_2-36x_1] \\
y &= [x_1]
\end{aligned}
$$

The system is linear, therefore we can solve for A, B, C, D

$$
A = \begin{bmatrix} 0 & 1 \\ -36 & -3 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 36 \end{bmatrix}
$$
$$
C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = [0]
$$

3 LSIM and gensig

Listing 1: lsim Simulation of the State Space Model with domain of 0-100 seconds

```
clear all;
clc;
% Define the transfer function of the system
H = tf(36,[1 3 36]);
% Define A,B,C,D
A = [0,1; -36,-3];
B = [0;36];
C = [1 0];
D = 0;
% Define state-space model
sys = ss(A,B,C,D);
% Define initial state
```

```
x0 = [0,0];
% Define the input u(t) = 1.0sin(0.1t) using sinusoidal signal
tau = 2*pi/0.1; % Period = 2*pi/0.1
% 0:Ts:Tf
Ts = 0.01; % Time step
Tf = 100; % Duration
[u,t] = gensig('sin',tau,Tf,Ts);
% Simulate the system
lsim(sys,u,t,x0);
grid on;
legend(System response);
fig = gcf; % Obtains current graphic in matlab
exportgraphics(fig, 'Fig/lsim_run_100s.pdf', 'ContentType','vector');
```

Listing 2: lsim Simulation of the State Space Model with domain of 0-4 seconds

```
clear all;
clc;
% Define the transfer function of the system
H = tf(36,[1 3 36]);
% Define A,B,C,D
A = [0,1; -36,-3];
B = [0;36];
C = [1 0];
D = 0;
% Define state-space model
sys = ss(A,B,C,D);
% Define initial state
x0 = [0,0];
% Define the input u(t) = 1.0sin(0.1t) using sinusoidal signal
tau = 2*pi/0.1; % Period = 2*pi/0.1
% 0:Ts:Tf
Ts = 0.01; % Time step
Tf = 4; % Duration
[u,t] = gensig('sin',tau,Tf,Ts);
% Simulate the system
lsim(sys,u,t,x0);
grid on;
legend(System response);
```

```
fig = gcf; % Obtains current graphic in matlab
exportgraphics(fig, 'Fig/lsim_run_4s.pdf', 'ContentType','vector');
```

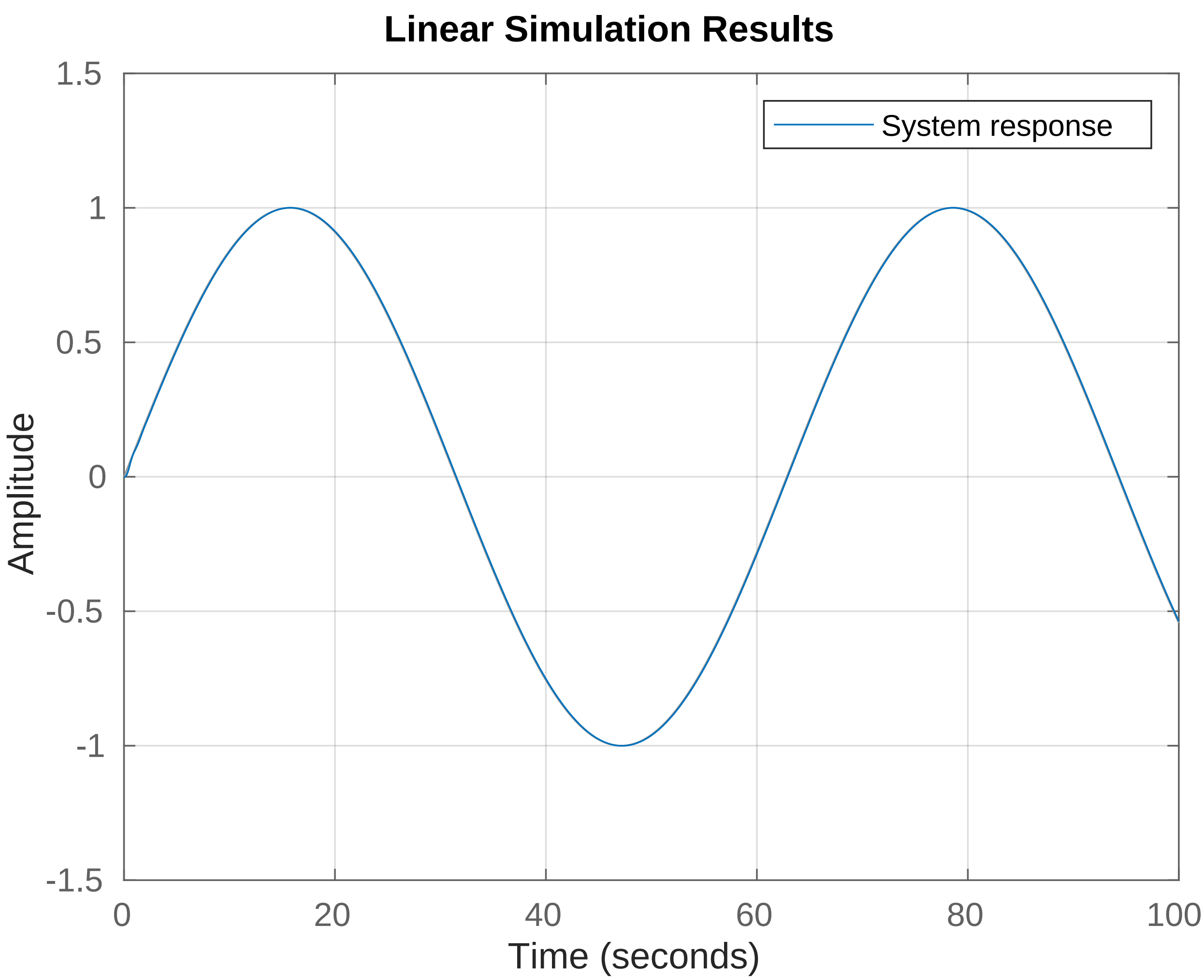

Figure 1: lsim Simulation of the State Space Model with a domain of 0-100 seconds

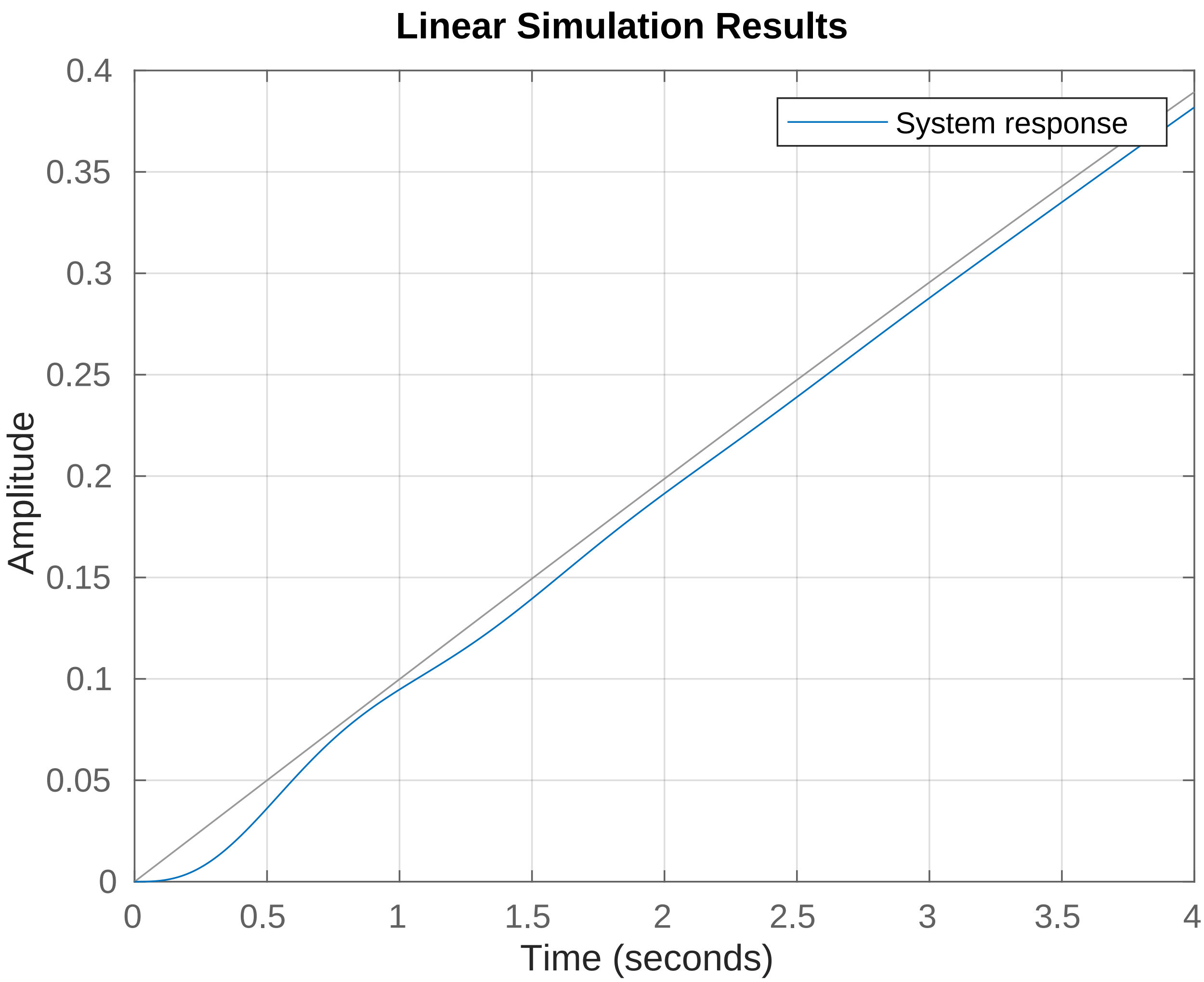

Figure 2: lsim Simulation of the State Space Model with a domain of 0-4 seconds

4 Varying Step-Size of 1.0 sin(0.1)

Listing 3: Function for the State Space of the representation of the system

```
function [dx] = f(x,u)
% x -- [2xn] column vector
% u -- [1xn] vector
A = [0,1; -36,-3];
B = [0;36];
```

```
dx = A*x + B*u;
end
```

Listing 4: Function to run Euler recursion

```
function sim_t = Euler(t,x0,h,u, filename)
% For the given initial condition x0 and step size
% h this function uses Euler integration to
% numerically solve the differential equation
% of the transfer function.
% Function output: sim_t, Euler Simulation time cost
tic; % start the clock
N = length(u); % The iteration steps based on the length of input signal
% Initialize x
x = zeros(length(x0),N); %The dimension of x in terms of dimension of x0
x(:,1) = x0; % IC
for i=1:N
    [dx] = f(x(:,i),u(i));
    x(:,i+1) = x(:,i) + dx*h;
end
sim_t = toc; % end the clock
sim_t = sim_t*1000;
display(append('Euler Simulation Took = ', string(sim_t) ,'ms'));
figure
subplot(2, 1, 1);
plot(t,x(1,1:i));
hold on;
plot(t,u);
legend('System response', 'Input signal');
plot_title = {[append('Euler Simulation: for step-size h = ', string(h))] [
    append('Simulation took ', string(sim_t), ' ms')] ['$\bf y$ vs. Time']};
grid on;
title(plot_title, 'Interpreter', 'latex');
xlabel('Time (s)');
ylabel('$ \bf y$', 'Interpreter', 'latex');
subplot(2, 1, 2);
plot (t,x(2,1:i));
title ('y prime vs. Time', 'Interpreter', 'latex')
xlabel('Time (S)');
ylabel('y prime', 'Interpreter', 'latex');
grid on;
```

```
fig = gcf; % Obtains current graphic in matlab
exportgraphics(fig, filename,'ContentType','vector');
end
```

Listing 5: Run the Euler function for different step sizes h

```
% Define the input u(t) = 1.0sin(0.1t) using sinusoidal signal
tau = 2*pi/0.1; % Period = 2*pi/0.1
% 0:Ts:Tf
Ts = 1; % Time step
Tf = 10; % Duration
% Define initial state
x0 = [3,0];
% Define step-size
% count = 1;
% h_str = ['1', '01', '005', '001', '0001'];
for h = [1, 0.1, 0.05, 0.01, 0.001]
    %[u,t] = gensig('sin',tau,Tf,h);
    N = (Tf/h);
    t = h*(0:N-1);
    u = sin(0.1*t);
    filename = append('Fig/Euler_plot_h_', string(h),'.pdf');
    sim_t = Euler(t,x0,h,u, filename);
    count = count + 1;
end
```

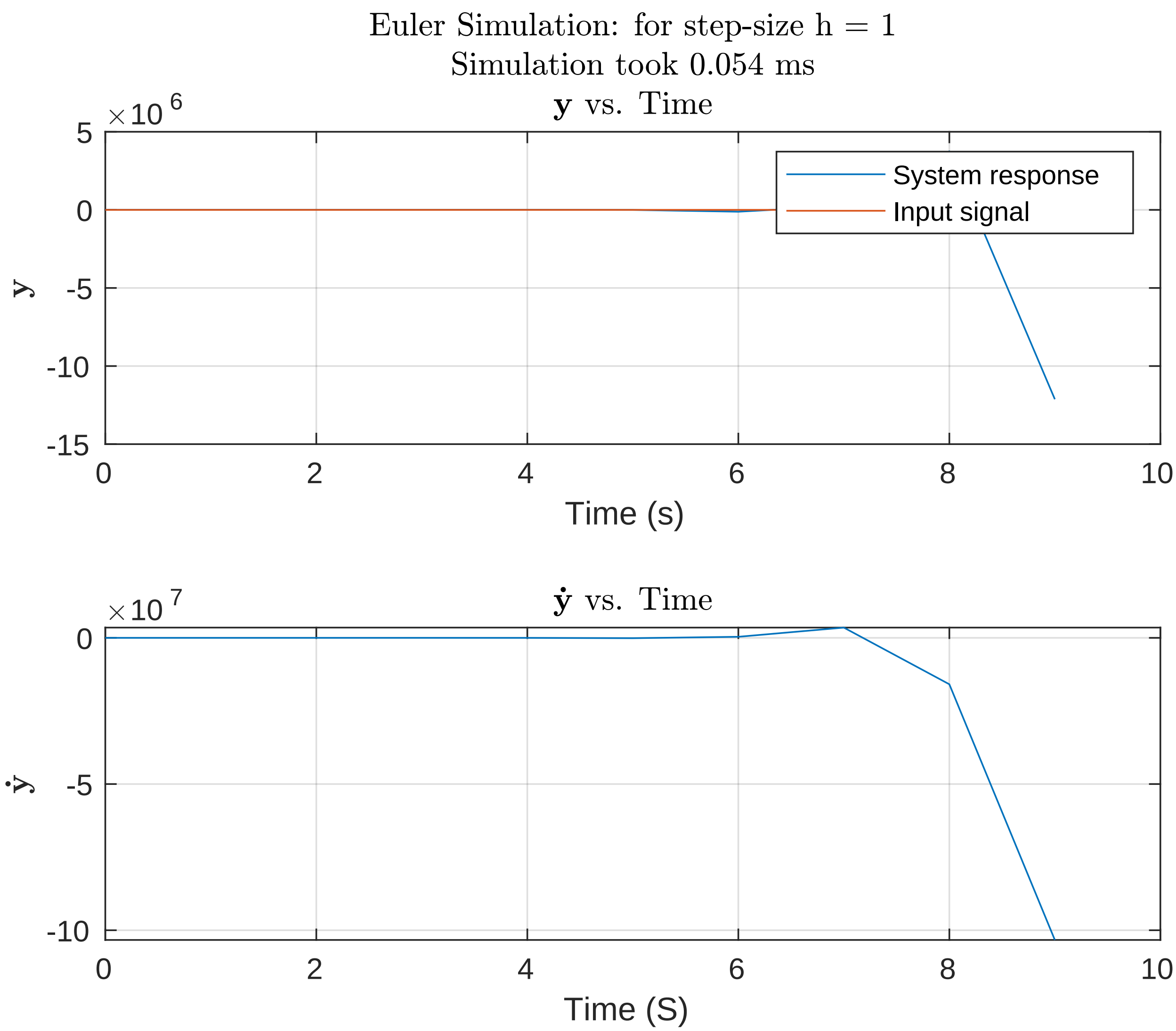

Figure 3: Euler Plot for $h = 1$

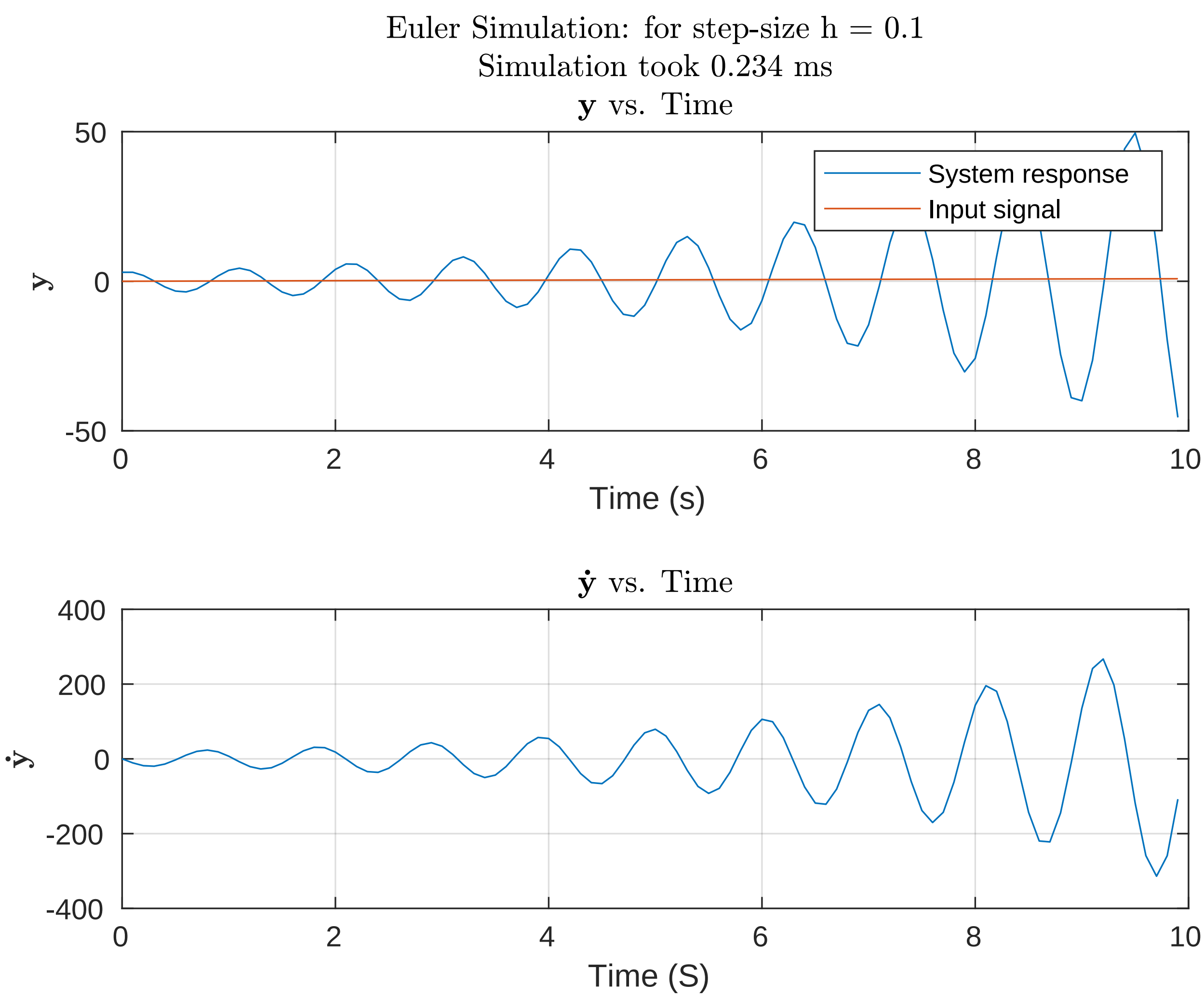

Figure 4: Euler Plot for $h = 0.1$

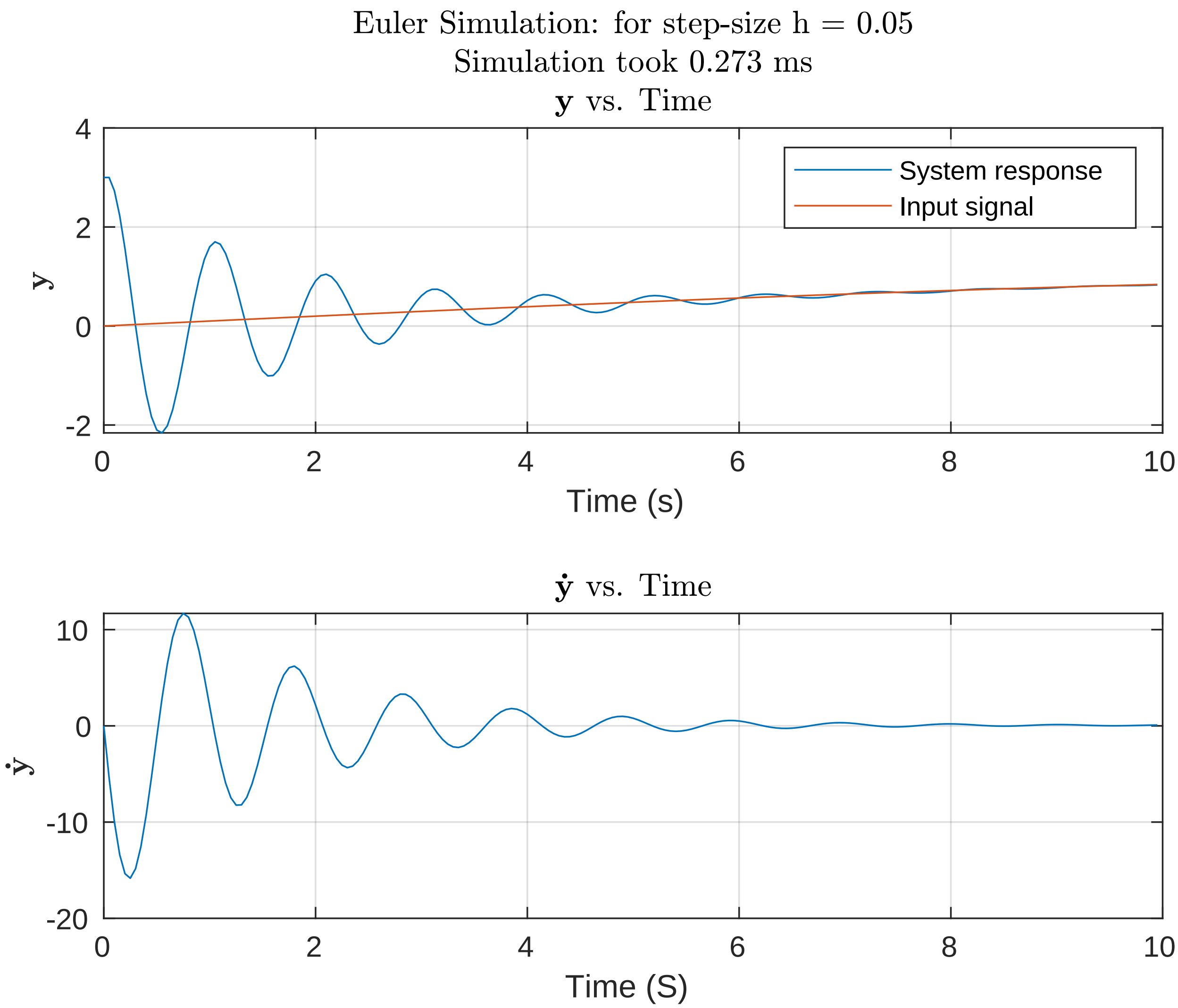

Figure 5: Euler Plot for $h = 0.05$

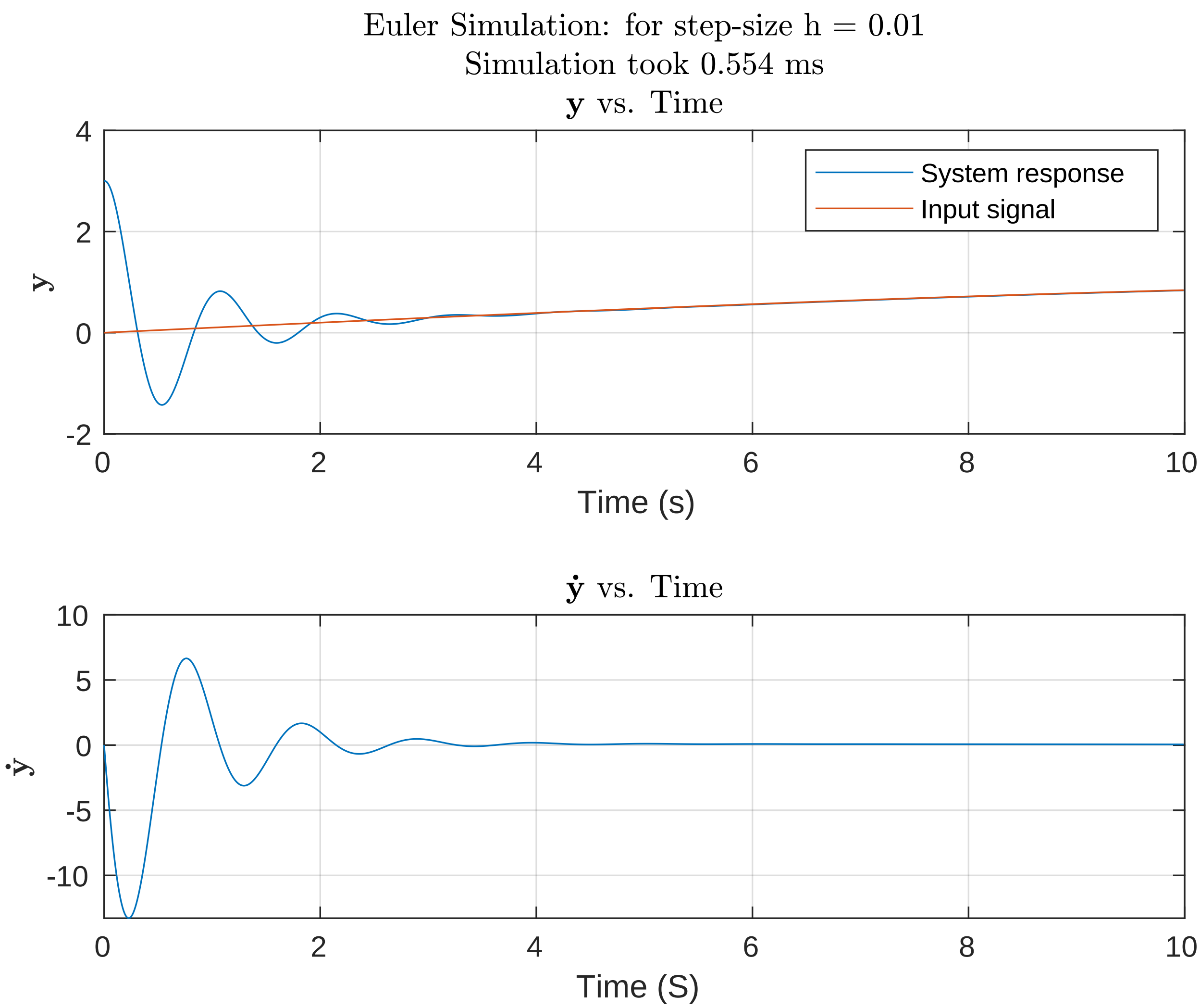

Figure 6: Euler Plot for $h = 0.01$

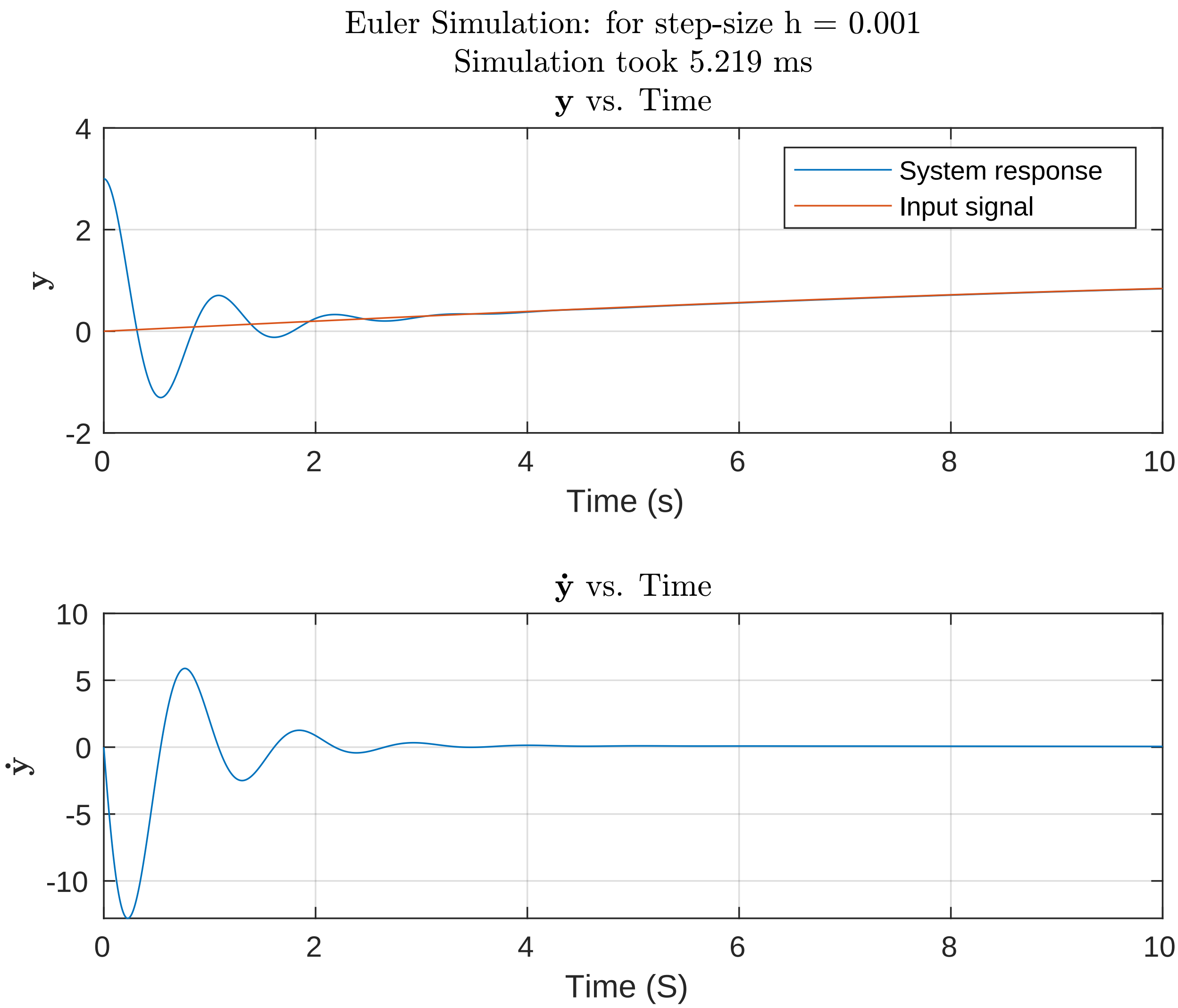

Figure 7: Euler Plot for $h = 0.001$

5 Varying ω in 1.0 sin(ωt)

Listing 6: Function for the State Space of the representation of the system for ODE23

```
function [dx, u] = f(t,x,omega)
% x -- [2xn] column vector
% u -- [1xn] vector
A = [0,1; -36,-3];
```

```
B = [0;36];
u = sin(omega*t);
dx = A*x + B*u;
end
```

Listing 7: Run ODE23 for sin(0.1t)

```
tspan = [0 10]; % Interval of integration
x0 = [3 0]; % Initial condition
omega = 0.1;
tic
[t_out,y] = ode23(@(t,x) f_ode(t,x,omega), tspan, x0);
tm = toc;
tm = tm*1000;
h = length(t_out) \ (tspan(2) - tspan(1));
t = tspan(1):h:tspan(2);
u = sin(omega*t);
display(append('ODE23 Simulation Took ', string(tm), 'ms'));
figure;
subplot(2, 1, 1);
plot(t_out, y(:,1));
hold on;
plot(t,u);
legend('System response', 'Input signal');
plot_title = {[append('ODE23 Simulation: for omega = ', string(omega))] [
    append('Simulation took ', string(sim_t), ' ms')] ['$ \bf y$ vs. Time']};
title(plot_title, 'Interpreter', 'latex');
xlabel('Time, t, seconds');
ylabel('$ \bf y$', 'Interpreter', 'latex');
grid on;
subplot(2, 1, 2);
plot(t_out, y(:,2));
title ('y prime vs. Time', 'Interpreter', 'latex');
xlabel('Time, t, seconds');
ylabel('y prime', 'Interpreter', 'latex');
grid on;
fig = gcf; % Obtains current graphic in matlab
exportgraphics(fig, 'Fig/ode_sin_input_01.pdf','ContentType','vector');
```

Listing 8: Run ODE23 for various sin(ωt)

```
for i = [0.001, 0.09, 1]
    tspan = [0 10]; % Interval of integration
    x0 = [3 0]; % Initial condition
    omega = i;
    tic
    [t_out,y] = ode23(@(t,x) f_ode(t,x,omega), tspan, x0);
    tm = toc;
    tm = tm*1000;
    h = length(t_out) \ (tspan(2) - tspan(1));
    t = tspan(1):h:tspan(2);
    u = sin(omega*t);
    display(append('ODE23 Simulation Took ', string(tm), 'ms'));
    figure;
    subplot(2, 1, 1);
    plot(t_out, y(:,1));
    hold on;
    plot(t,u);
    legend('System response', 'Input signal');
    plot_title = {[append('ODE23 Simulation: for omega = ', string(omega))] [append('Simulation took ', string(sim_t), ' ms')] ['$ \bf y$ vs. Time']};
    title(plot_title, 'Interpreter', 'latex');
    xlabel('Time, t, seconds');
    ylabel('$ \bf y$', 'Interpreter', 'latex');
    grid on;
    subplot(2, 1, 2)
    plot(t_out, y(:,2))
    title ('y prime vs. Time', 'Interpreter', 'latex');
    xlabel('Time, t, seconds');
    ylabel('y prime', 'Interpreter', 'latex');
    grid on;
    filename = append('Fig/ode_sin_input_', string(omega), '.pdf');
    fig = gcf; % Obtains current graphic in matlab
    exportgraphics(fig, filename,'ContentType','vector');
end
```

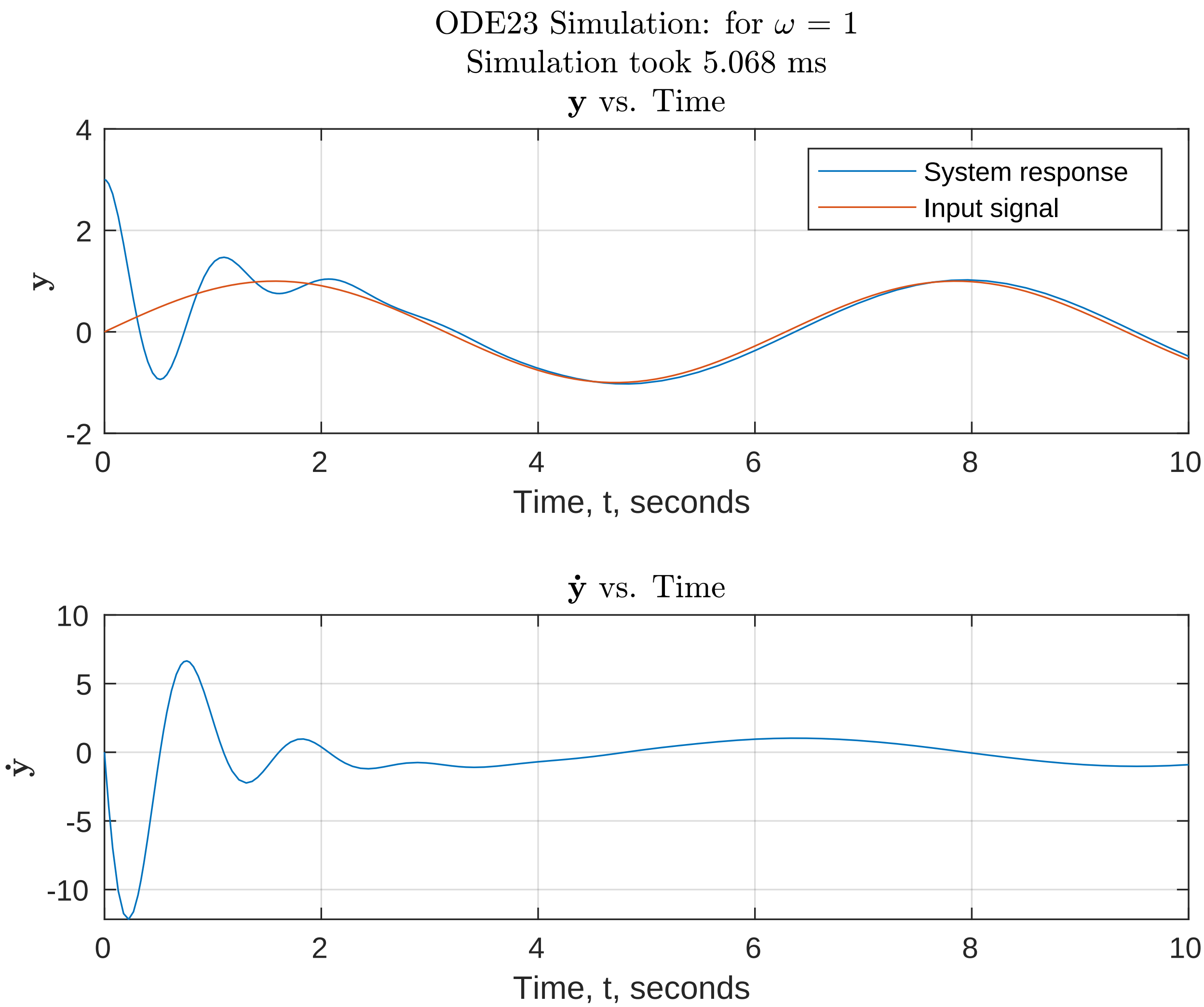

Figure 8: Ode Plot for $\omega = 1$

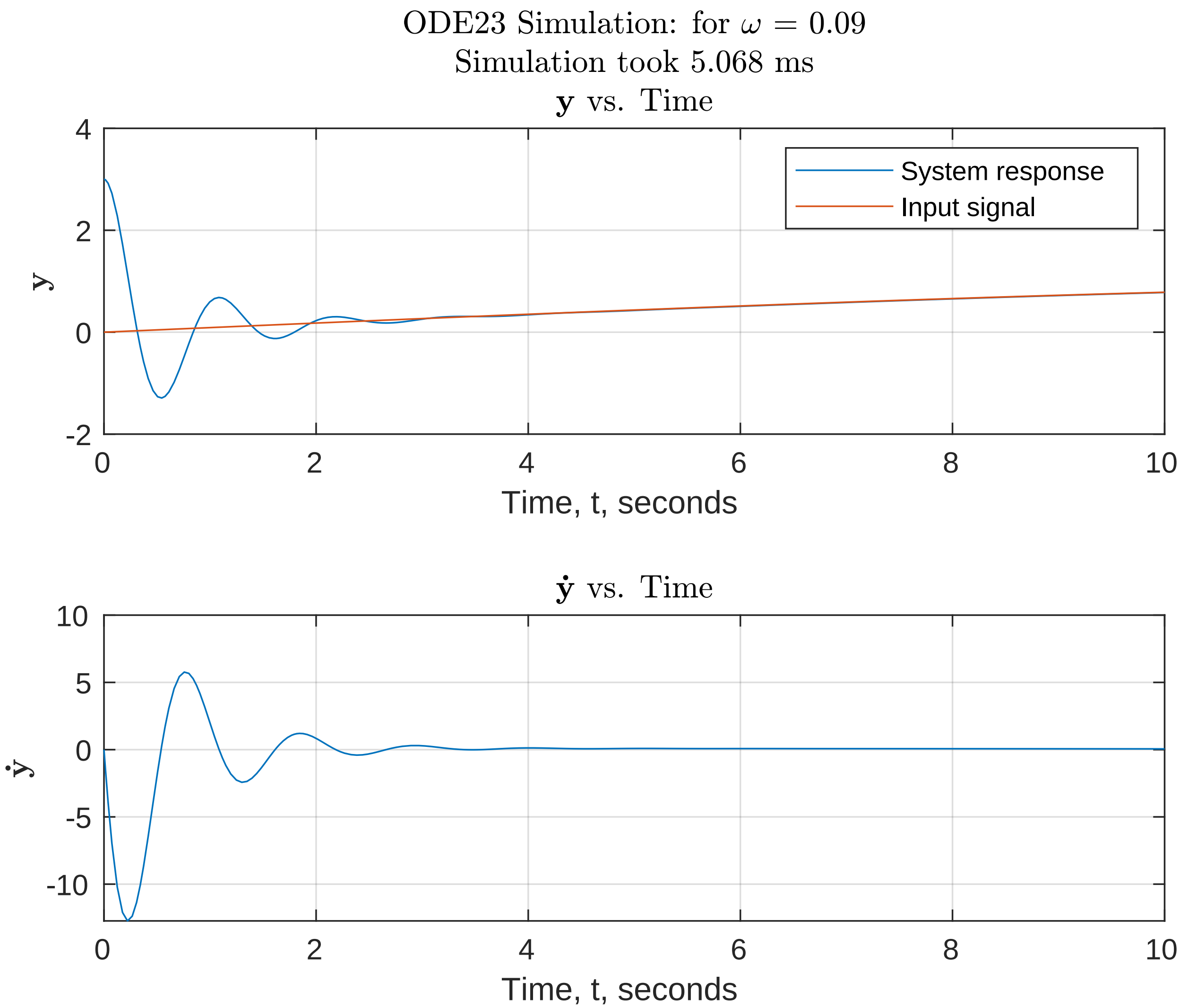

Figure 9: Ode Plot for $\omega = 0.09$

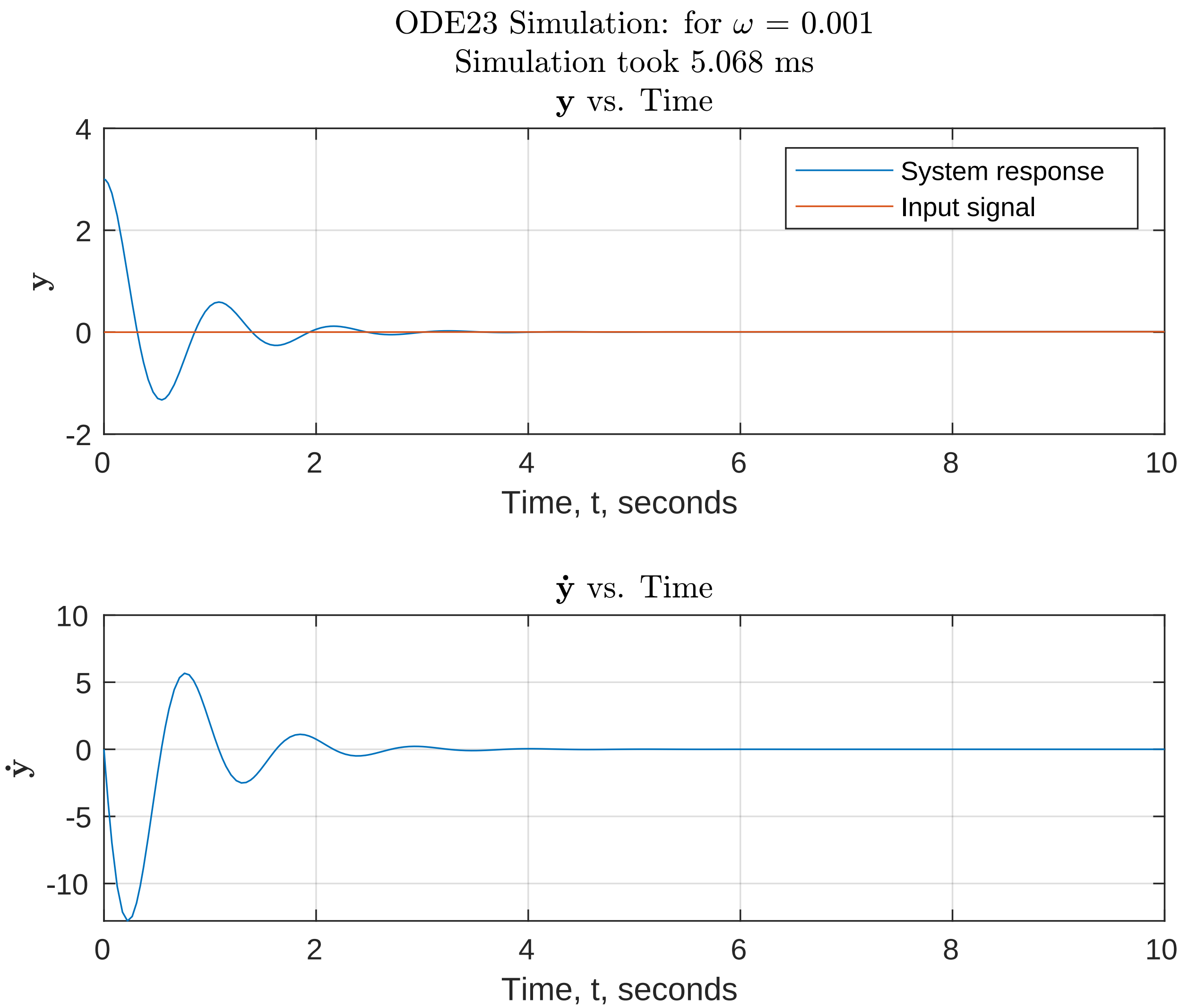

Figure 10: Ode Plot for $\omega = 0.001$

6 Conclusion