THE UNIVERSITY OF TEXAS AT ARLINGTON, TEXAS
DEPARTMENT OF ELECTRICAL ENGINEERING

EE 5321 - 001

OPTIMAL CONTROL

HW # 4
ASSIGNMENT

by

SOUTRIK MAITI

1001569883

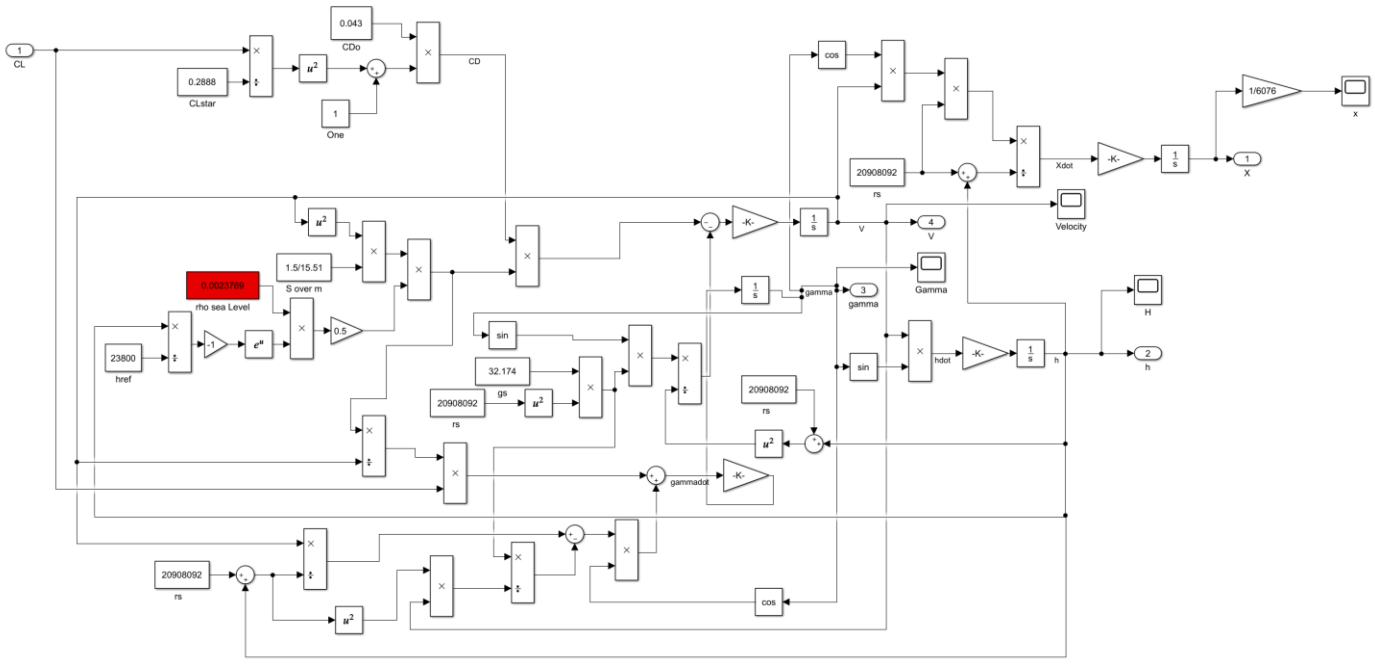
Presented to

Prof. Michael Niestroy

March 29,2018

Problem 1:

Simulink diagram of SRAM



a) Sram_main MATLAB code

```
tfinal = 40 / 100;
tau = 0:0.02:1;

CL=ones(length(tau),1)*(-0.2);
CL(end+1) = tfinal;

lb=ones(length(tau),1)*(-pi);
ub=ones(length(tau),1)*pi;
lb(end+1)=0.01;
ub(end+1)=1;

options = optimset('Display','iter','TolCon',1e-3,'Algorithm','interior-
point','PlotFcns','optimplotx','MaxFunEvals',2500);

[CL_final, cost] = fmincon('sram_cost',
CL,[],[],[],[],lb,ub,'sram_constraint',options);

tfinal=CL_final(end);
[tout,yout]=sim('SRAM',1,[],[tau' CL_final(1:end-1)]);
figure
plot(yout(:,1)/6076,yout(:,2));grid;xlabel('X, nm');ylabel('h');
figure
plot(tau*CL_final(end)*100,CL_final(1:end-1));xlabel('Time,
sec');ylabel('Final CL');grid
figure
```

```
plot(tau*CL_final(end)*100,yout(:,4));xlabel('Time,
sec');ylabel('Velocity, ft/sec');grid
```

sram_cost MATLAB file

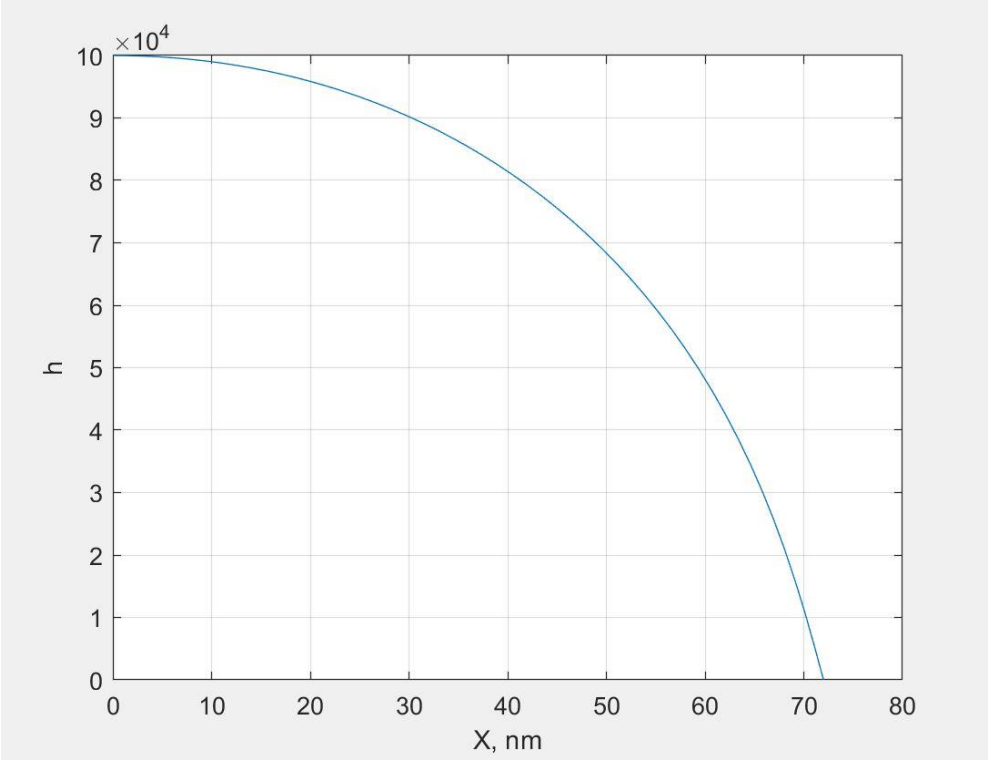
```
function y = sram_cost(p)

assignin('base', 'tfinal', p(end));

tau=[0:0.02:1]';
u=[p(1:end-1)];

[tout,yout]=sim('SRAM',1,[],[tau u]);

y = 0.0 - yout(end,4) / 11000; % maximum final velocity
end
```

sram_constraint MATLAB file

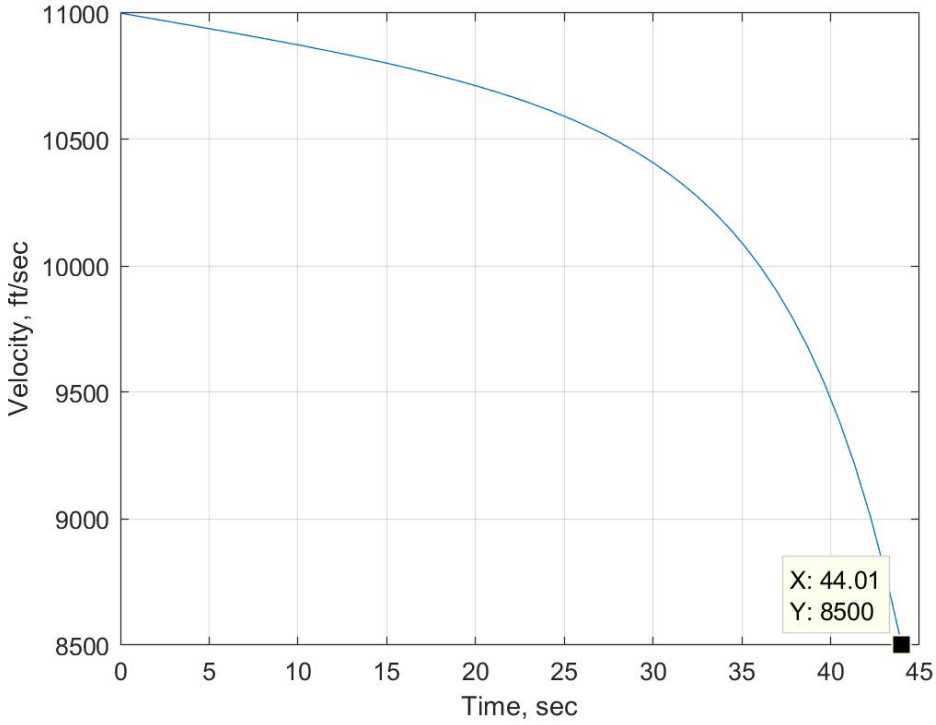
```
function [cineq, ceq] = sram_constraint(p)

cineq = [];
assignin('base', 'tfinal', p(end));

tau=[0:0.02:1]';
u=[p(1:end-1)];

[tout,yout]=sim('SRAM',1,[],[tau u]);

ceq(1) = (yout(end,1) - 72*6076) / 500000;
ceq(2) = yout(end,2) / 100000;

end
```

The plots are as follows:

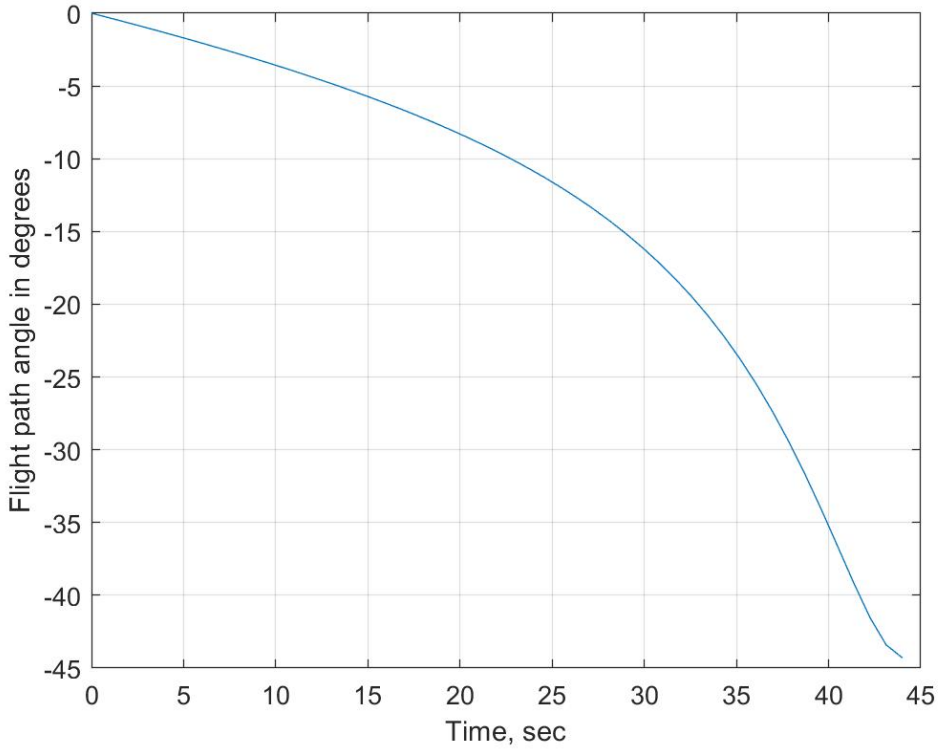
Height vs distance



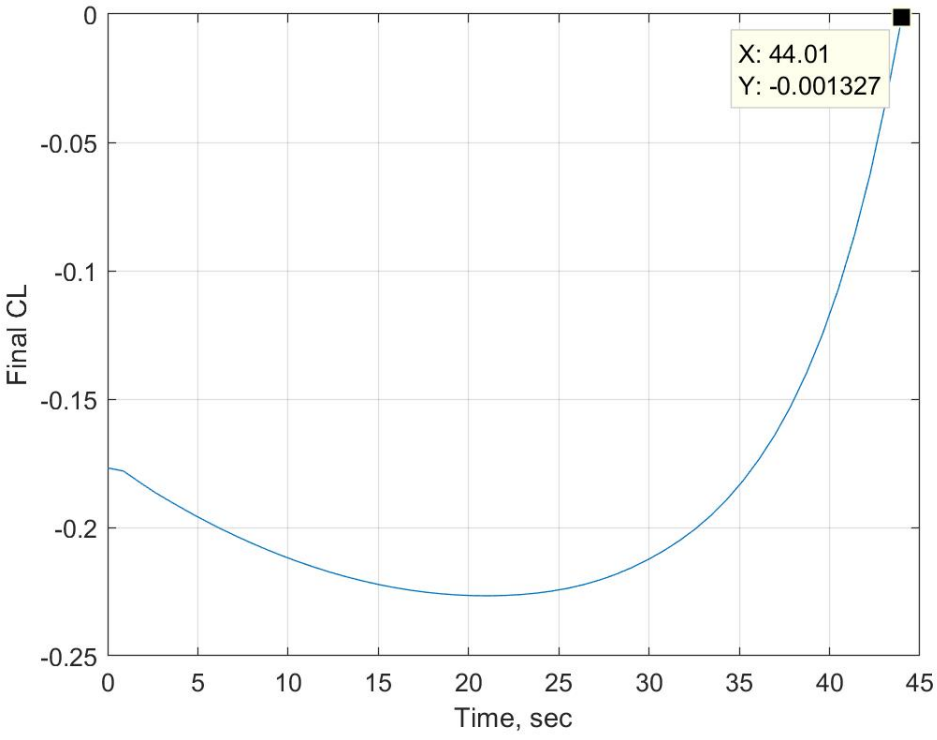
Velocity as a function of time



Flight path angle in degrees



Flight control vs time



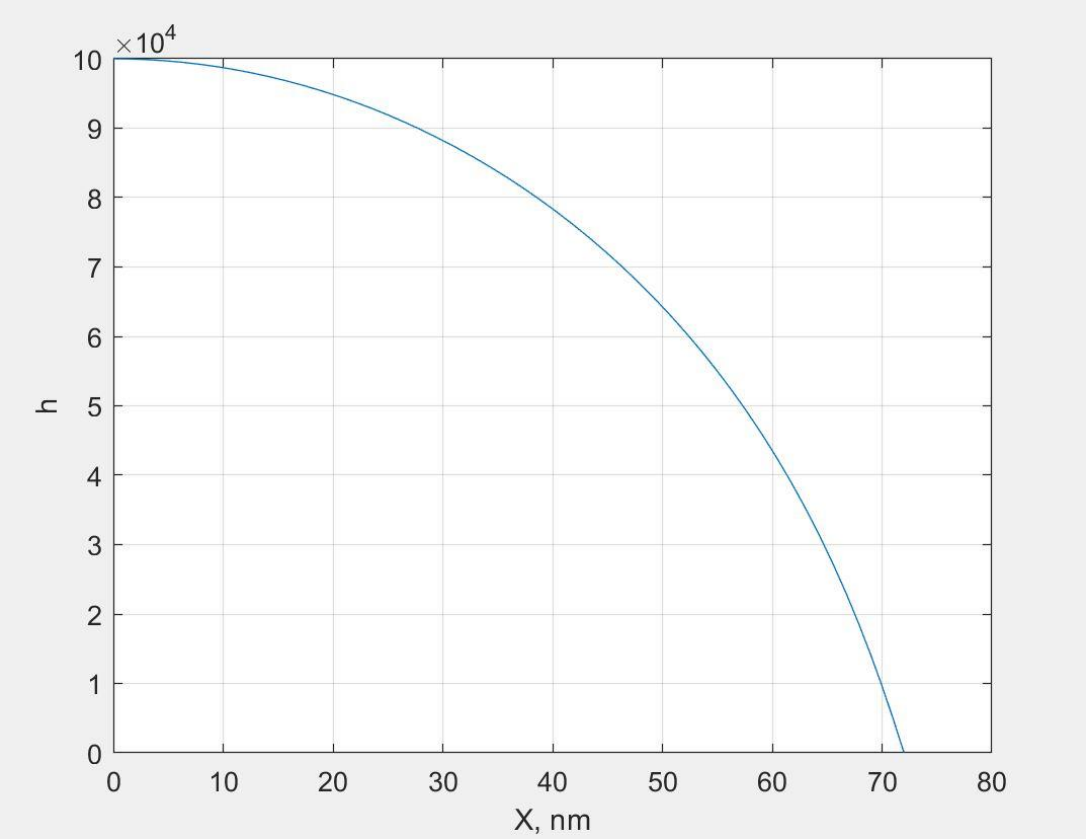
b) The only change that occurs for this part is the cost function which is as follows:

sram_cost MATLAB file

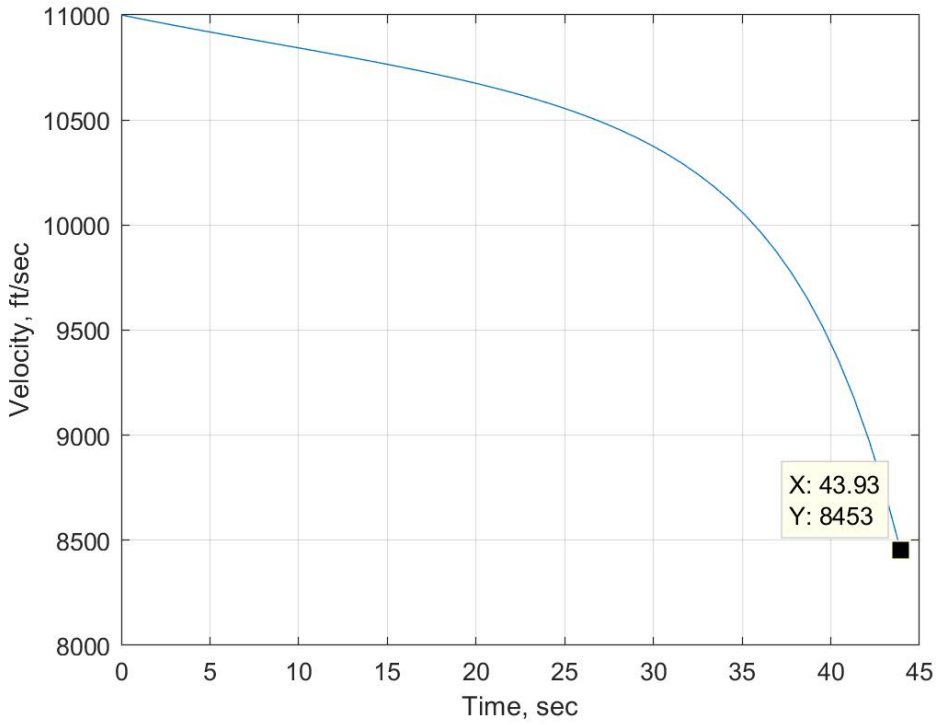
```
function y = sram_cost(p)

assignin('base', 'tfinal', p(end));

tau=[0:0.02:1]';
u=[p(1:end-1)];

[tout,yout]=sim('SRAM',1,[],[tau u]);

y = p(end); % minimum final time
end
```

The plots are as follows:

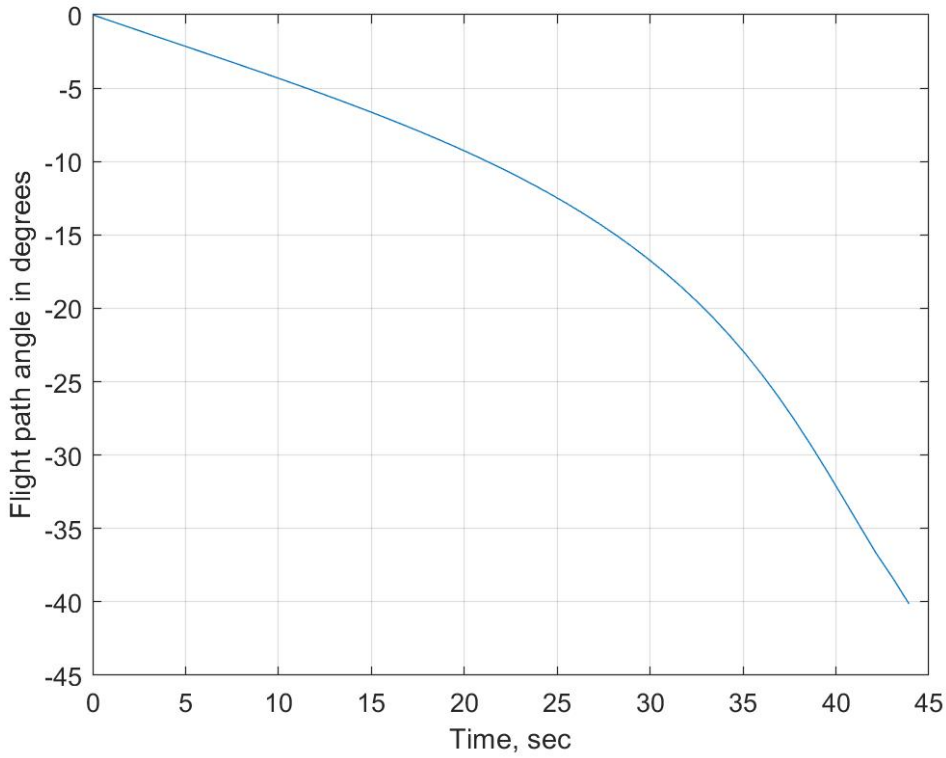
Height vs distance



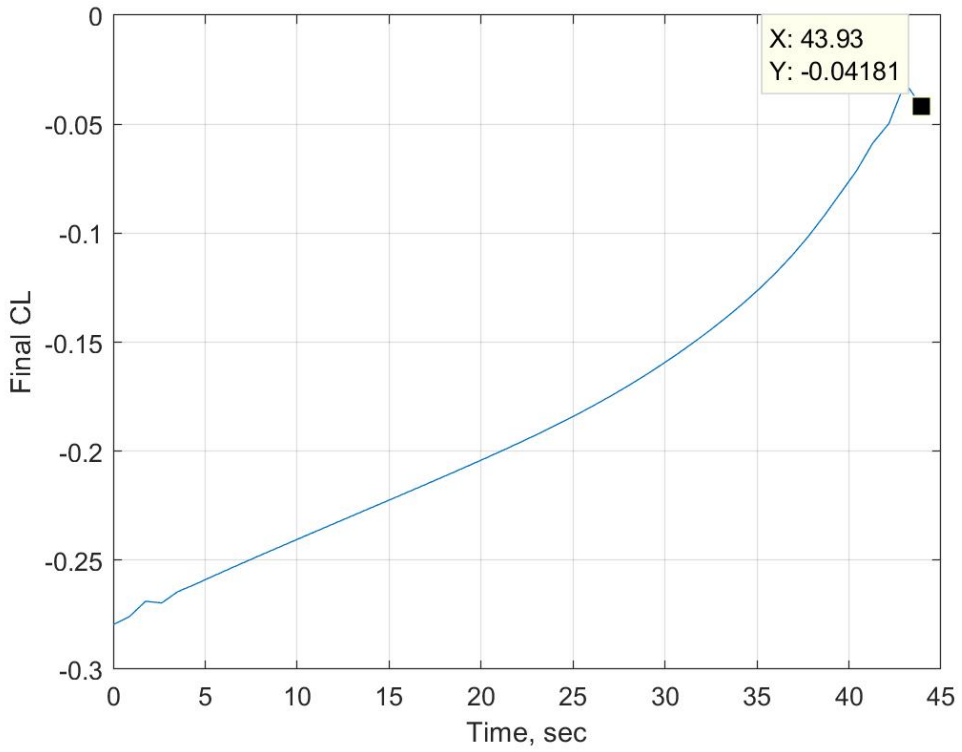
Velocity as a function of time



Flight path angle in degrees



Flight control vs time



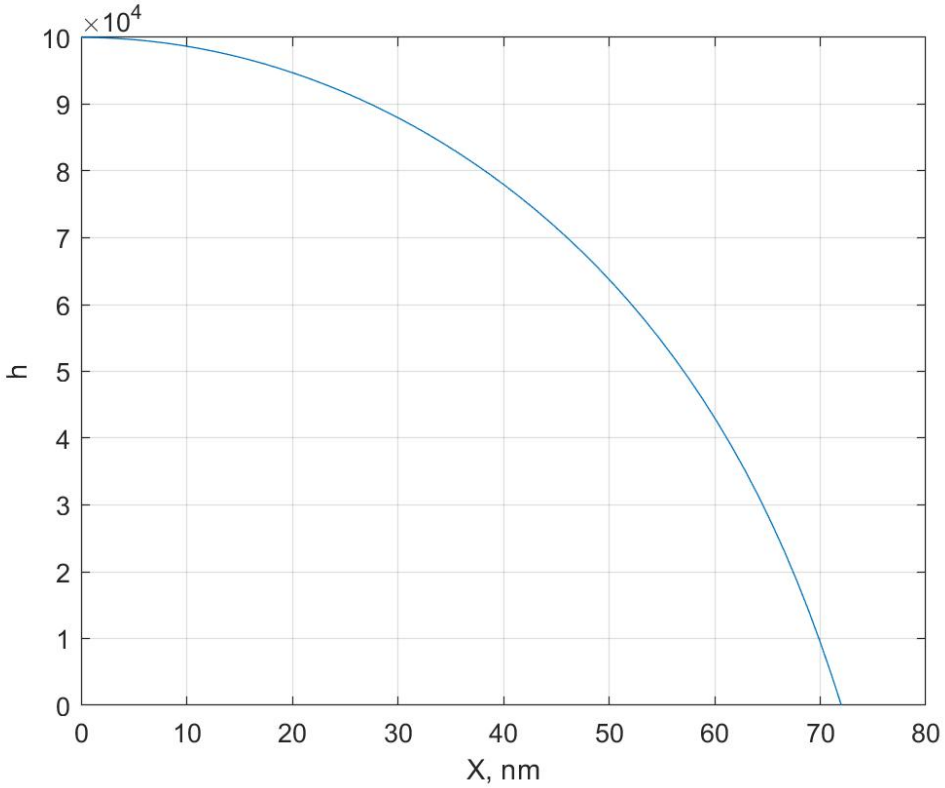
c)

	Final velocity	Final time
Part a)	8500(*Maximum velocity*)	44.01
Part b)	8453	43.93 (*Minimum time*)

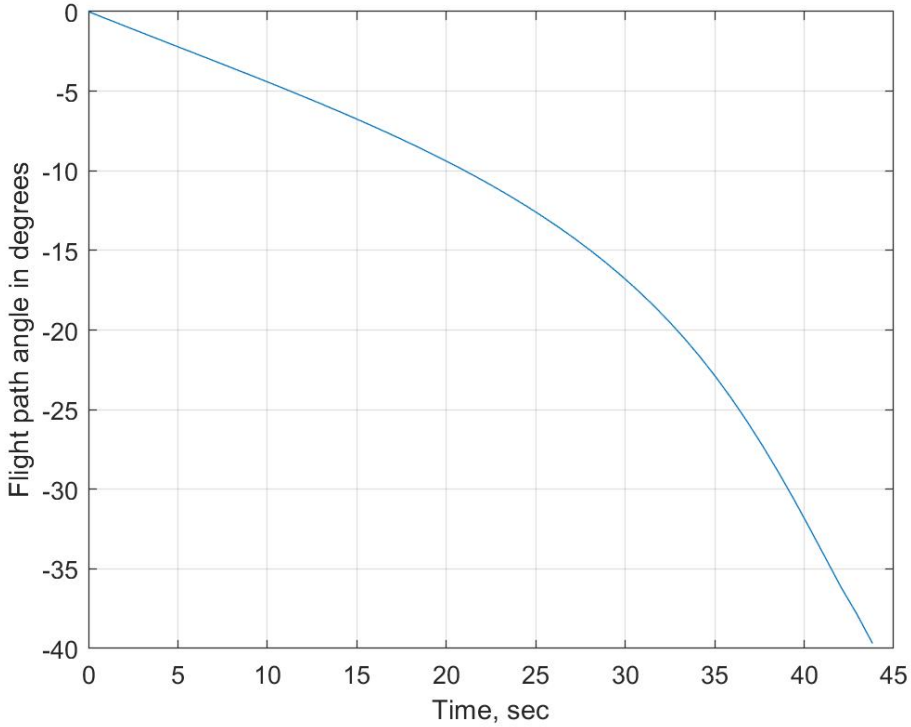
d)

case 1-When the atmospheric density is changed to *0.0023769*0.9* then we get the following plots:

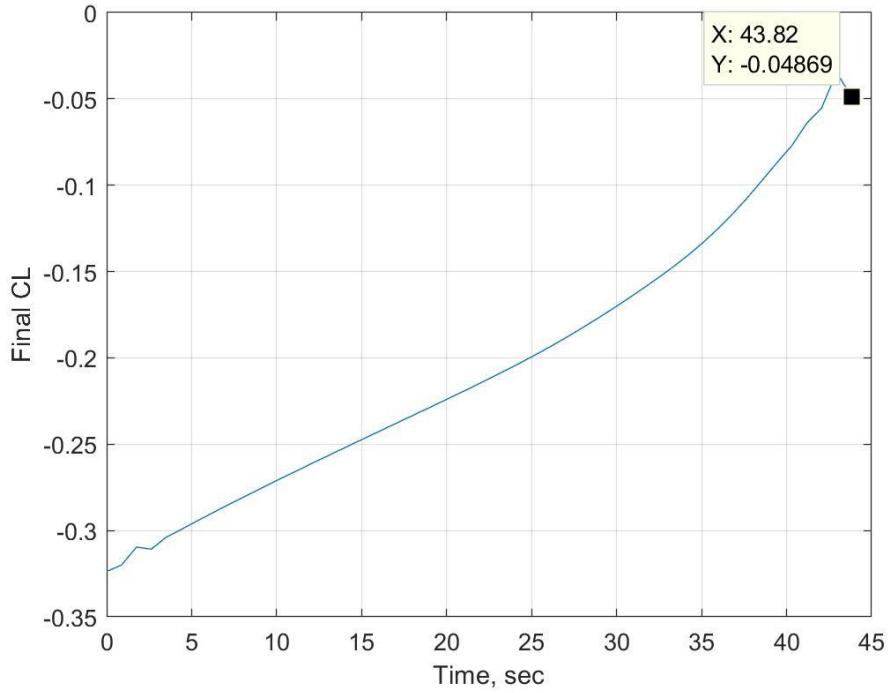
Height vs distance



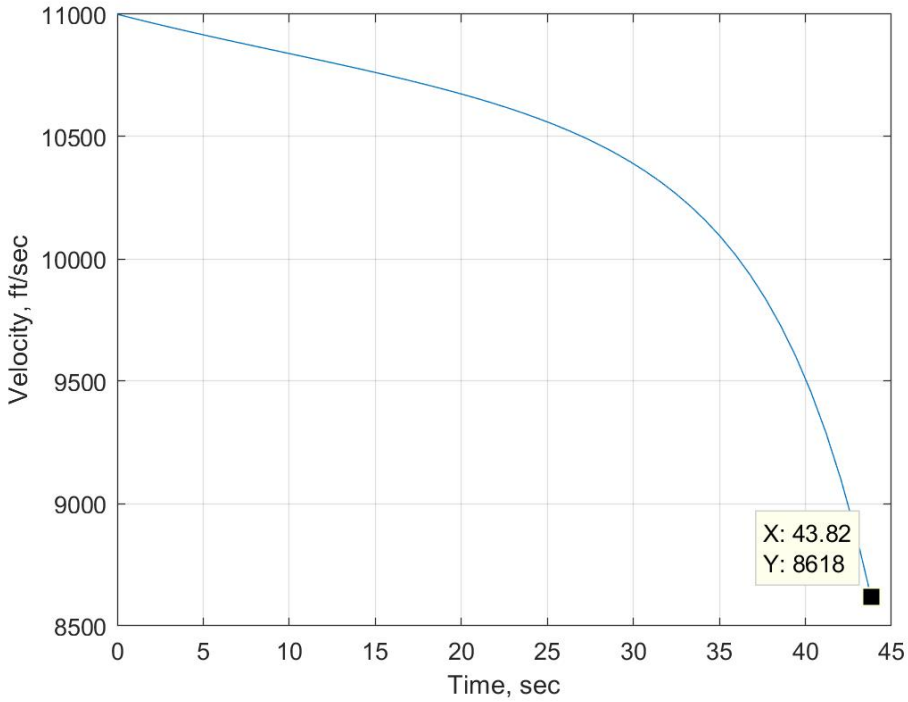
Flight path angle in degrees



Flight control vs time

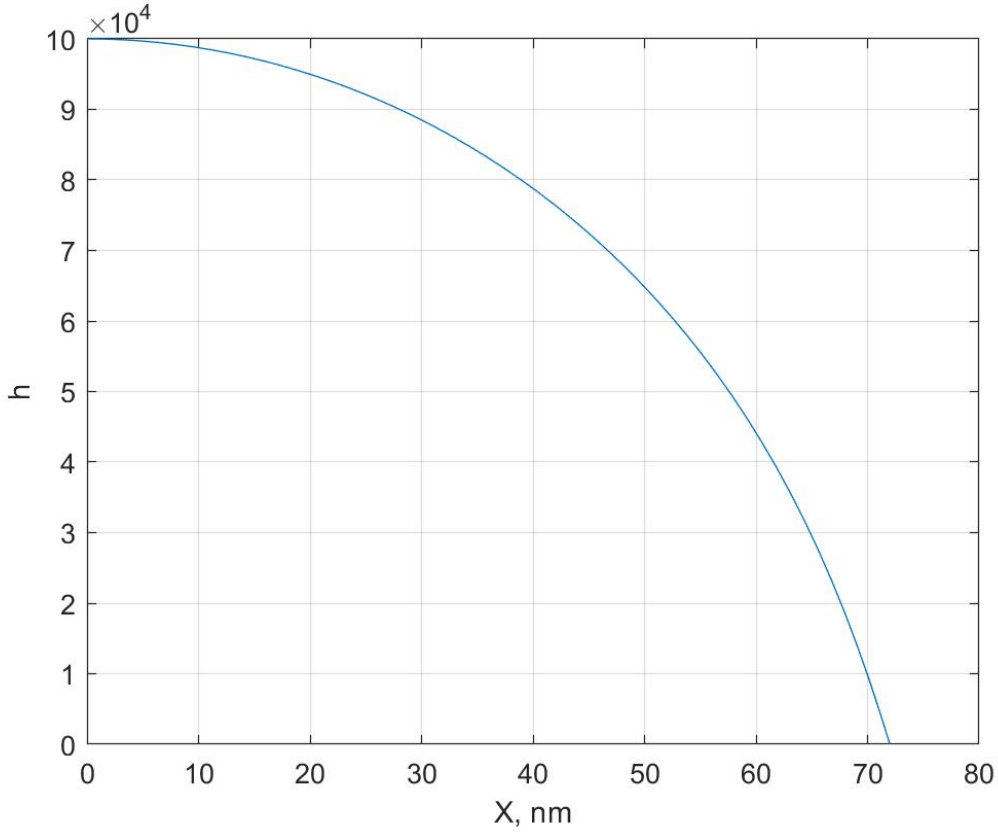


Velocity as a function of time

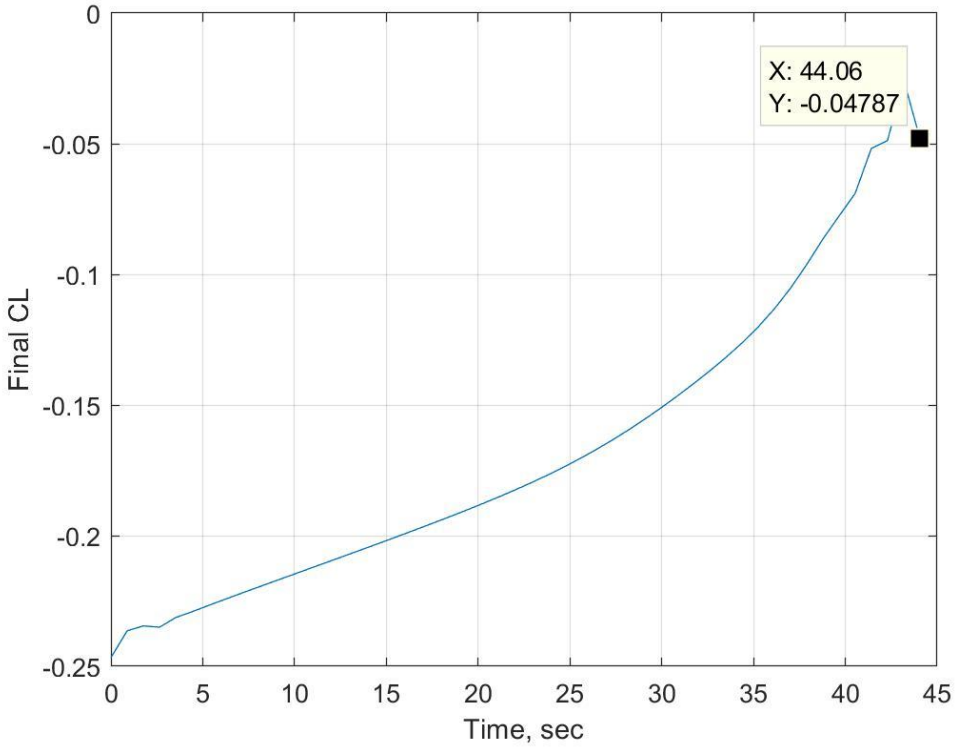


case 2-When the atmospheric density is changed to *0.0023769*1.1* then we get the following plots:

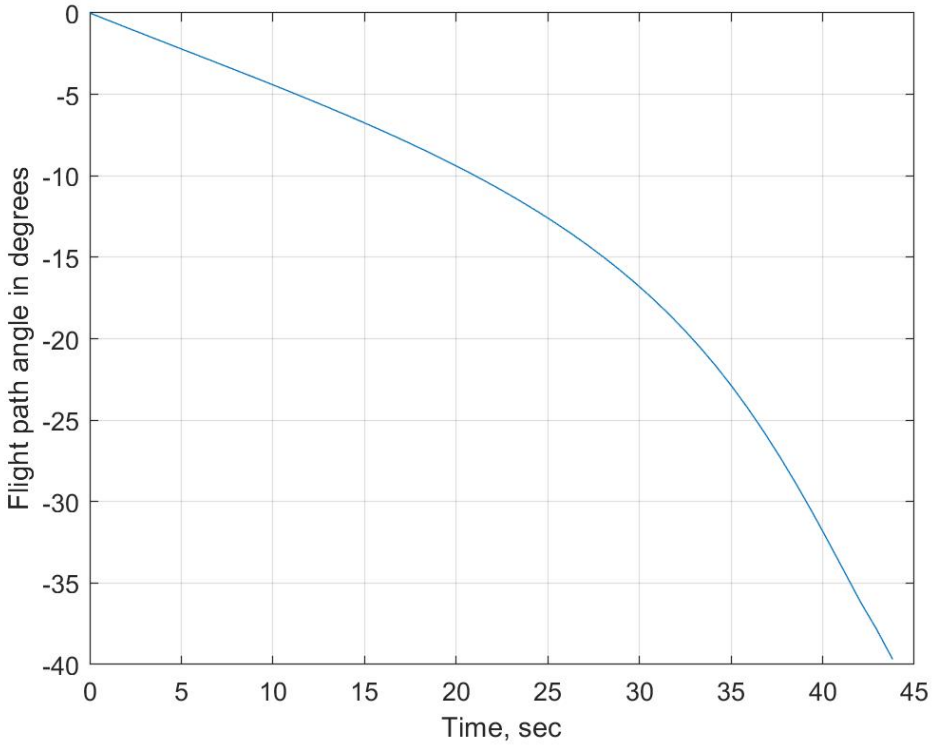
Height vs distance



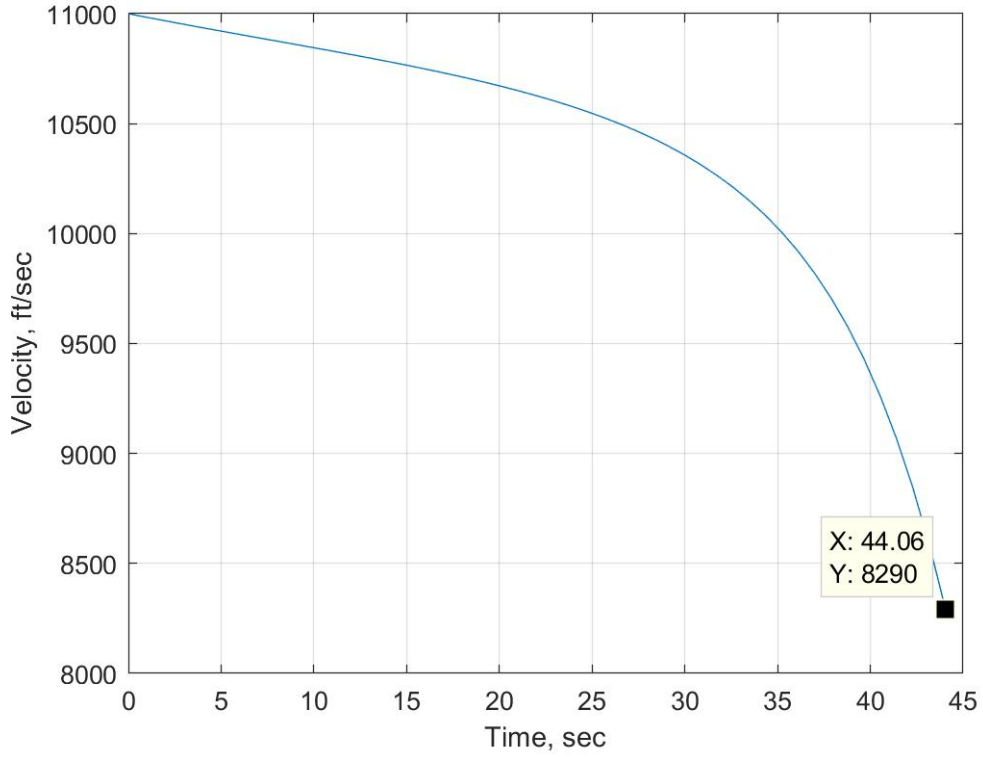
Flight control vs time



Flight path angle in degrees



Velocity as a function of time

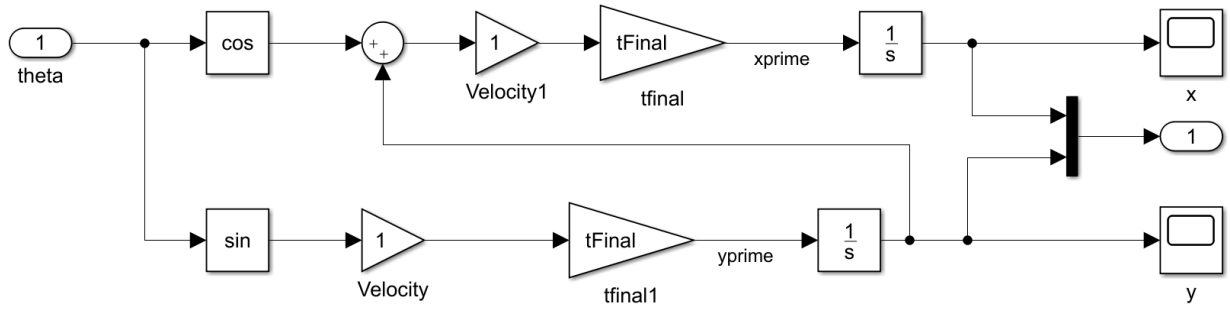


Atmospheric Density	Velocity	Optimal final time
0.0023769*0.9	8616	43.82
0.0023769*1.1	8290	44.06

Although there is no significant changes in the flight angle but we can see a significant drop of *326 ft/s* in velocity and rise in optimal final time of *0.24s* when the atmospheric density increases.

Problem 2:

a) Simulink diagram



zermeloMain MATLAB code:

```
%% initial guess at tFinal
tFinal = 2.75;
%% tau goes from 0 to 1
tau = 0:0.02:1;

%% initial x
x0 = 4.9;

%% initial y
y0 = 1.66;

%% initial guess at control time history
theta0 = ones(length(tau),1)*(255/57.3);

%% initial guess at final time
theta0(end+1) = tFinal;

%% lower bound on control
lB = ones(length(tau),1)*(-2*pi);

%% upper bound on control
uB = ones(length(tau),1)*pi*2;

%% lower bound on final time
lB(end+1) = 1;

%% upper bound on final time
uB(end+1) = 30;

options = optimset('Display','iter','TolCon',1e-4,'Algorithm','interior-
point','PlotFcns','optimplotx','MaxFunEvals',4500);
```

```
[thetaFinal, cost] = fmincon('zermeloCost',
theta0,[],[],[],[],[],[],'zermeloConstraint',options);

%% optimized final time
tFinal = thetaFinal(end);

%% Optimal time history with optimal control and final time
[tOut,yOut] = sim('zermelo',1,[],[tau' thetaFinal(1:end-1)]);

%% phase plane plot
figure
plot(yOut(:,1),yOut(:,2));
title('phase plane plot');
grid;
xlabel('x');
ylabel('y');

%% plot of optimal control as a function of actual time
figure
plot(tau*tFinal, thetaFinal(1:end-1)*57.3);
xlabel('Final Time');
ylabel('Final Theta in deg');
grid;
s = sprintf('Final Time = %5.3f seconds', tFinal);
title(s)
```

zermeloCost MATLAB code:

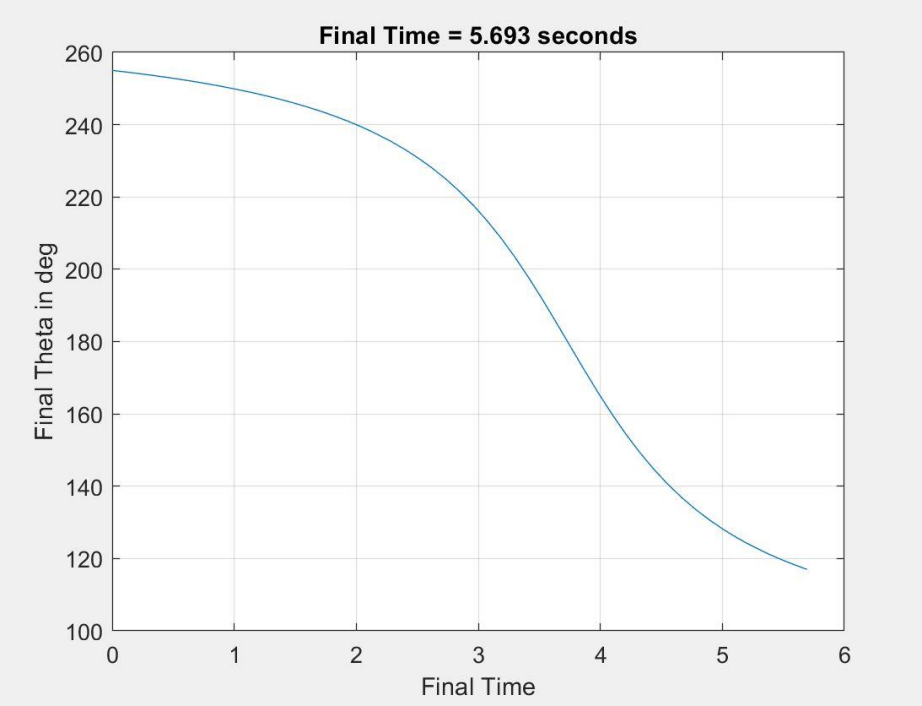
```
function y = zermeloCost(p)
y = p(end);
end
```

zermeloConstraint MATLAB code:

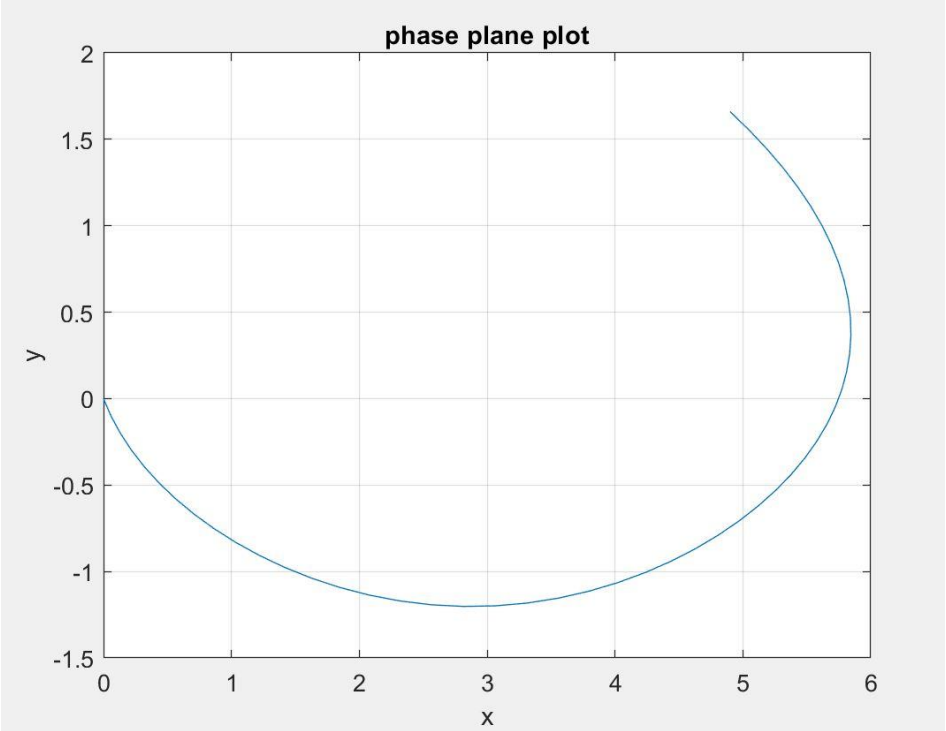
```
function [cineq, ceq] = zermeloConstraint(p)

cineq = [];
assignin('base','tFinal', p(end));
tau = [0:0.02:1]';
u = [p(1:end-1)];

[tOut,yOut] = sim('zermelo',1,[],[tau u]);

ceq(1) = yOut(end,1);
ceq(2) = yOut(end,2);
end
```

The plots are as follows:

The control input vs time



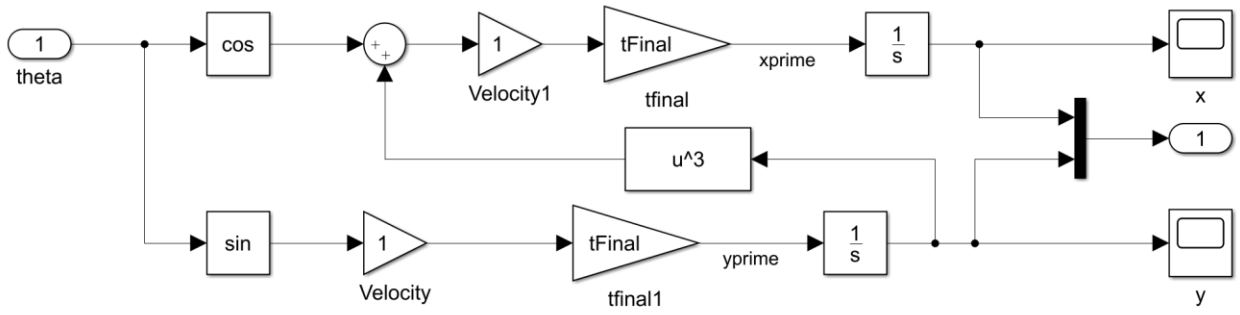
The trajectory of the boat



b) Now we need to find the optimization with the y term cubed as shown below:

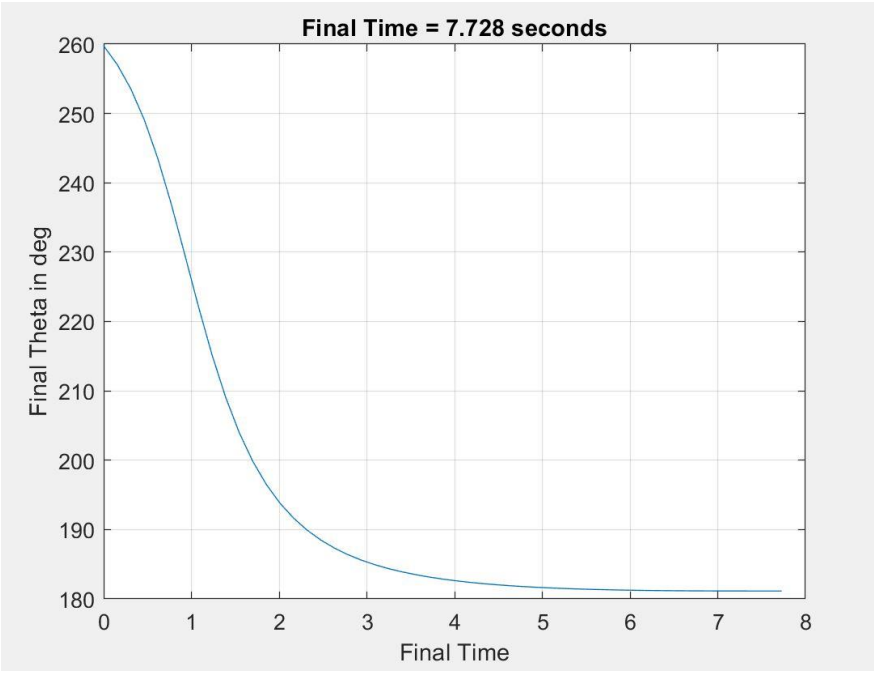
$$\dot{x} = V\cos\theta + Vy^3$$

There are no changes in the MATLAB code but there is only a minor change in the Simulink diagram as follows:

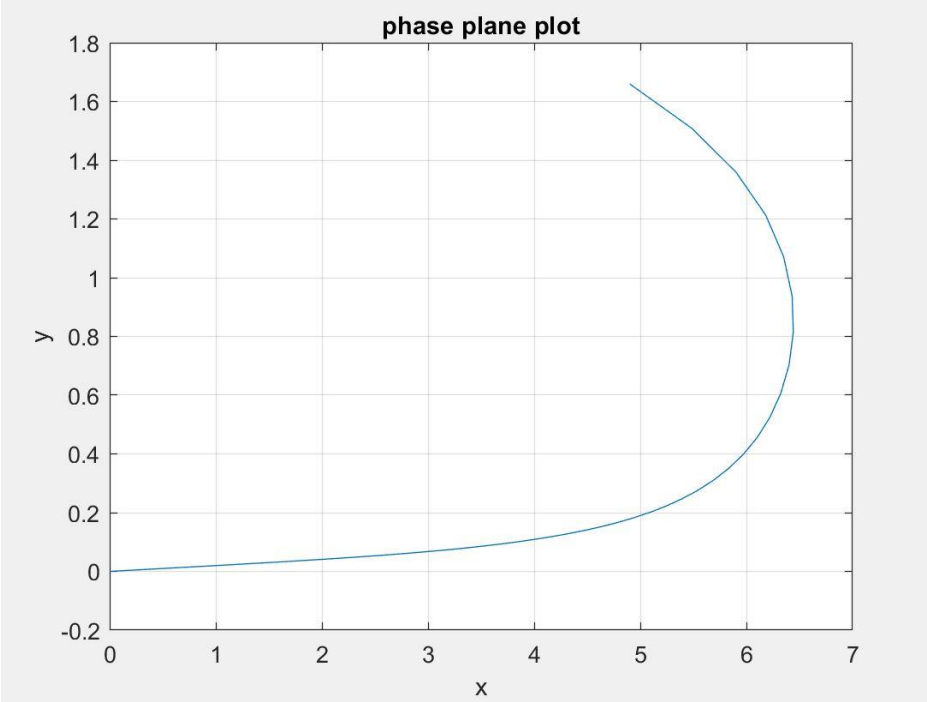


The plots generated will be as follows:

Control input vs time



The trajectory of the boat



Conclusion:

The effect of y cubed term is that the optimization results in a different final time, the state and control time histories converge quickly as compared to part a. The trajectory of the boat is different as compared to part a because it never crosses the y = 0. The control time history smoothly settles to the final theta value as compared to case a. The final time after optimization is greater than the one obtained in the case a.