Table of Contents

Full Report

clears all variables from previous executions of scripts

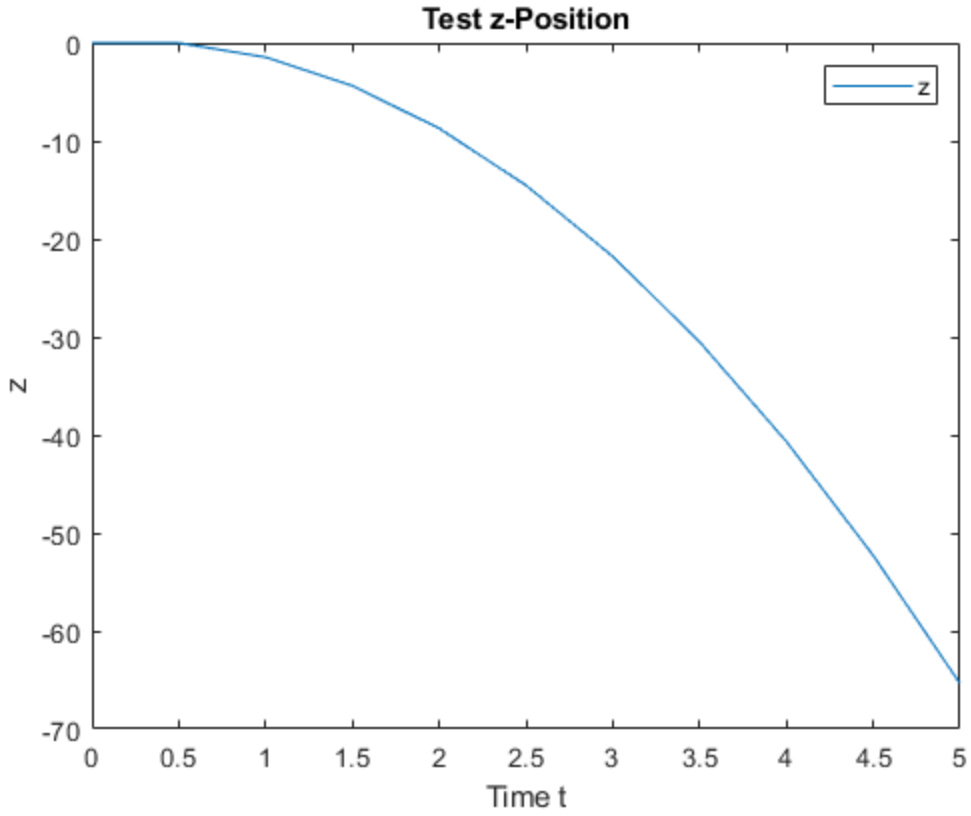
```
clear all;
close all;
```

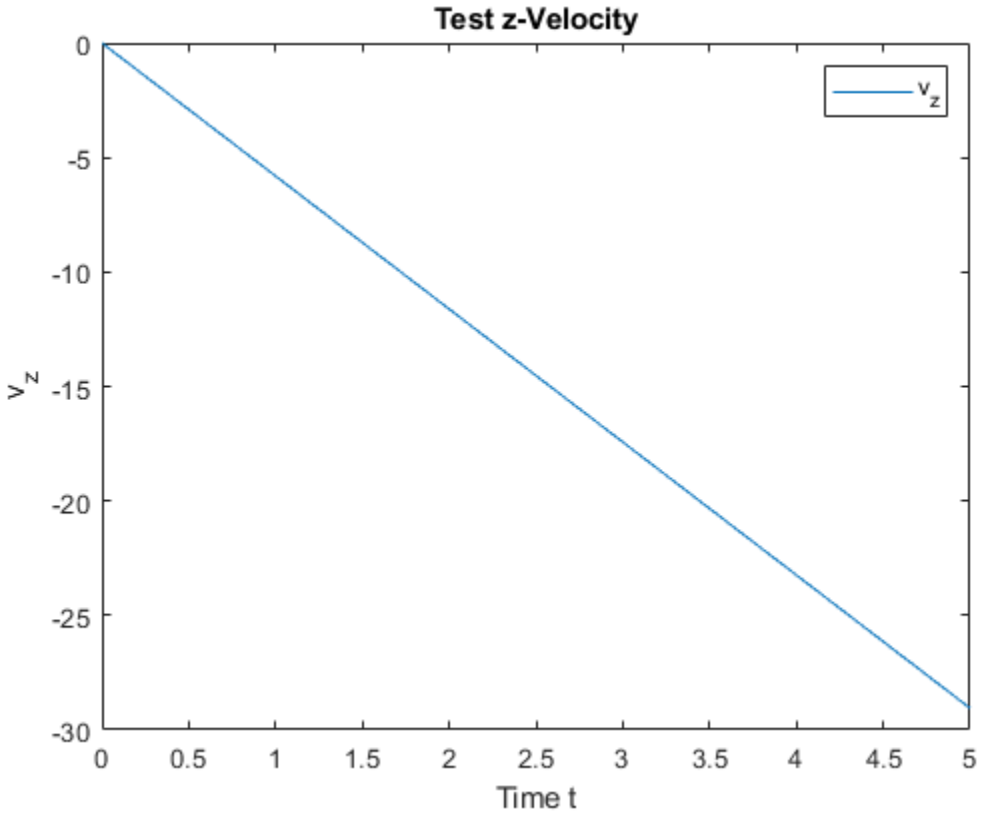
Test Case

clears all variables from previous executions of scripts

```
clear all;
close all;

% Initial time
t_0 = 0;

% Final time
t_f = 5;

% Event Horizon: [t_0 , t_f]
horizon = t_f - t_0;

% Number of time steps
N=10;

% Initial conditions: Matrix of 2 row vectors
i_c = [0; 0]; % set z(0) = v_z(0) = 0

%Time step
delta = (horizon)/N; % Time step = Horizton /number of steps

% Initial Time
t(1) = t_0;

% Initialize x = [z(t) ; v_z(t)] : Matrix of 2 row vectors
% where x is meant to represent a position matrix containing

% z positon and z velocity initial condition assignment
x(:,1) = i_c;

% Set the force
F = 1;
```

```
% For loop, sets next (t,x) values
for k=1:N

    % Updates our time t according to the step size delta
    % Used for plotting, not used in the ODE, however the manual has it
    t(k+1) = t(k) + delta; % Row vector

    % Calls the function f(t,x) = dx/dt
    % Uses Forward Euler for v_z and a_z
    x(:,k+1) = x(:,k) + delta*myODE(t(k),x(:,k), F);  % matrix of 2 rows
end

% Plots of our simulation
figure(1)
plot(t,x(1,:))
xlabel('Time t')
ylabel('z')
legend('z')
title('Test z-Position')

figure(2)
plot(t,x(2,:))
xlabel('Time t')
ylabel('v_z');
legend('v_z')
title('Test z-Velocity')
```


Test z-Position
z
0
-10
-20
-30
-40
-50
-60
-70
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
Time t


Test z-Velocity
v_z
0
-5
-10
-15
-20
-25
-30
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
Time t

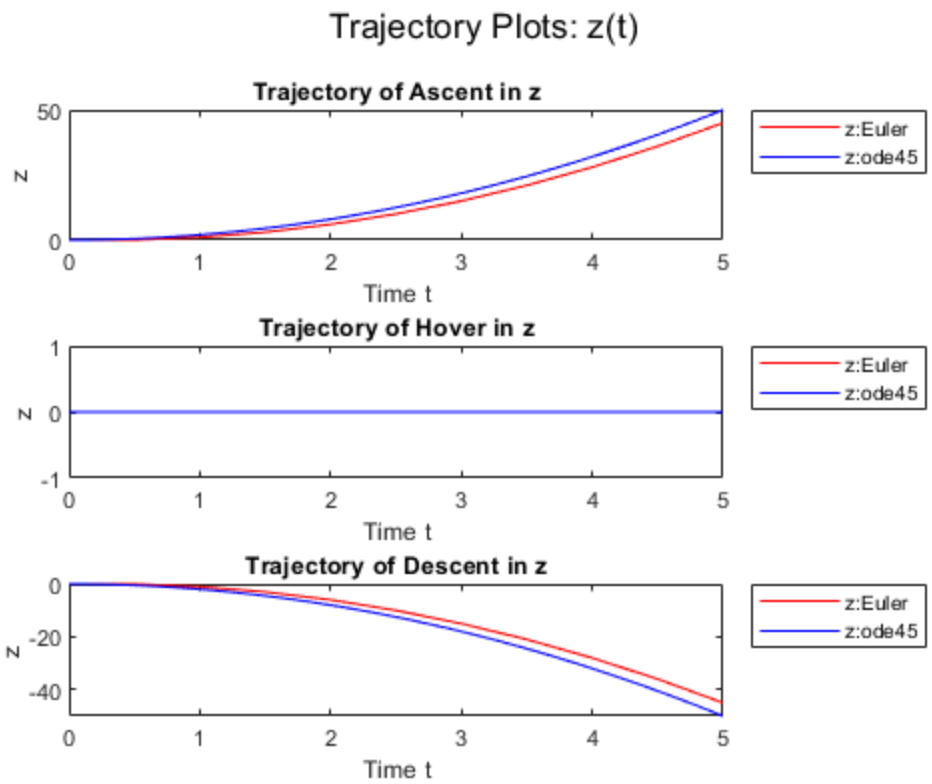
Exercise 1

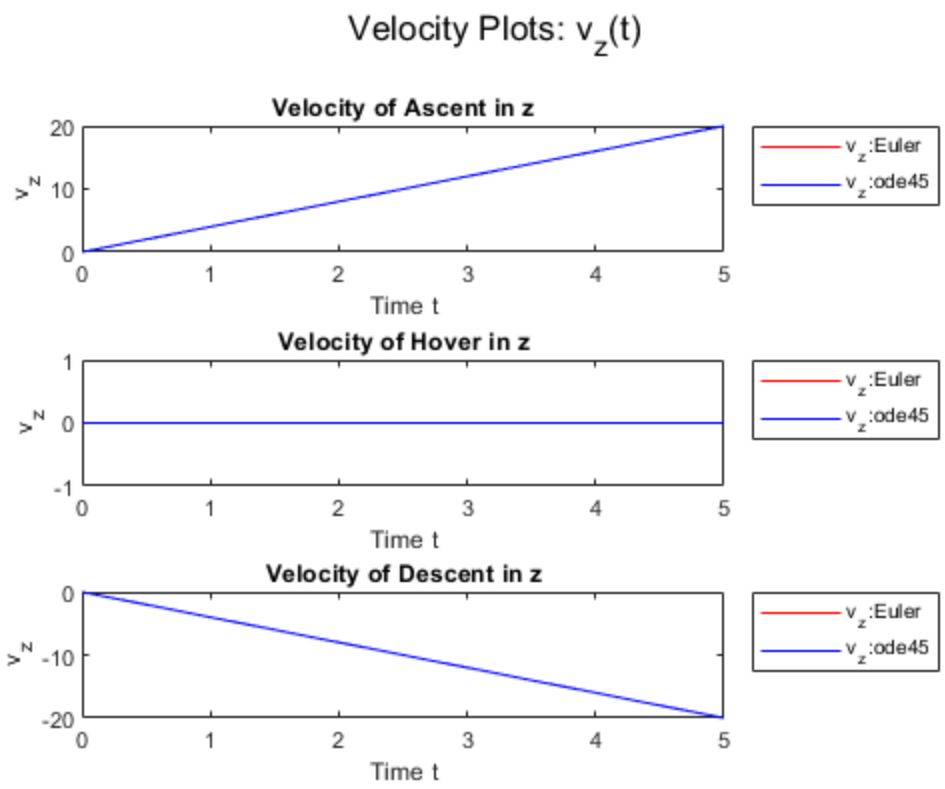
Modify myODE(): done, takes F inputs Model 1D quadrotor Submit 2 figures Position Figure: Subfigures: Hover(Euler+Ode45), Ascend(E+O), Descend(E+O) Velocity Figure: Subfigures: Hover(Euler+Ode45), Ascend(E+O), Descend(E+O) Include: Title, Legend, axis labels, submit myODE seperately Detailed Comments are in the Test Case above

```
% Best Practices
clear all;
close all;

% Initial time
t_0 = 0;

% Final time
t_f = 5;

% Event Horizon: [t_0 , t_f]
horizon = t_f - t_0;

% Number of time steps
N=10;

% Initial conditions
i_c = [0; 0];               % set z(0) = v_z(0) = 0

%Time step
delta = (horizon)/N;        % Time step = Horizton /number of steps

% Initial Time
t(1) = t_0;

% Initialize x for all 3 cases
xAe(:,1) = i_c;              % Ascend
xHe(:,1) = i_c;              % Hover
xDe(:,1) = i_c;              % Descend

% Set Gravity
g = 9.81; %m/s^2

% Set Force for all 3 cases
%   Eq: Fz = Fnet = 4F -g
FA = g/4 + 1;               % Ascend
FH = g/4;                   % Hover
FD = g/4 - 1;               % Desend

% Calculations from myODE function
% For loop: sets next (t,x) = (t,(z,vz)) values
for k=1:N
    % Time used for plotting the domain- not used in function
    t(k+1) = t(k) + delta;

    % Calls the function f(t,x) = dx/dt: used in Forward Euler (vz,az)
```

```
    % Ascend
    xAe(:,k+1) = xAe(:,k) + delta*myODE(t(k),xAe(:,k), FA);
    % Hover
    xHe(:,k+1) = xHe(:,k) + delta*myODE(t(k),xHe(:,k), FH);
    % Descend
    xDe(:,k+1) = xDe(:,k) + delta*myODE(t(k),xDe(:,k), FD);
end

% Data from ode45 file
% Load Variables from ode_sim_ode45.m
load("ode45Data.mat","xA","xH","xD","t45")

% % Plots-------------------------------------
% Position x(1,:)
figure(1);

subplot(3,1,1)
plot(t,xAe(1,:), Color='r');
hold on
plot(t45,xA(:,1),Color ='b');
xlabel('Time t');
ylabel('z');
title('Trajectory of Ascent in z')
legend('z:Euler','z:ode45','Location','bestoutside')

subplot(3,1,2)
plot(t,xHe(1,:), Color='r');
hold on
plot(t45,xH(:,1),Color ='b');
xlabel('Time t');
ylabel('z');
title('Trajectory of Hover in z')
legend('z:Euler','z:ode45','Location','bestoutside')

subplot(3,1,3)
plot(t,xDe(1,:), Color='r');
hold on
plot(t45,xD(:,1),Color ='b');
xlabel('Time t');
ylabel('z');
title('Trajectory of Descent in z')
legend('z:Euler','z:ode45','Location','bestoutside')
sgtitle('Trajectory Plots: z(t)')

% Velocity x(2,:)
figure(2);

subplot(3,1,1)
plot(t,xAe(2,:),Color='r');
hold on
plot(t45,xA(:,2),Color= 'b')
xlabel('Time t');
ylabel('v_z');
title('Velocity of Ascent in z')
```

```
legend('v_z:Euler','v_z:ode45','Location','bestoutside')

subplot(3,1,2)
plot(t,xHe(2,:),Color='r');
hold on
plot(t45,xH(:,2),Color= 'b')
xlabel('Time t');
ylabel('v_z');
title('Velocity of Hover in z')
legend('v_z:Euler','v_z:ode45','Location','bestoutside')

subplot(3,1,3)
plot(t,xDe(2,:),Color='r');
hold on
plot(t45,xD(:,2),Color= 'b')
xlabel('Time t');
ylabel('v_z');
title('Velocity of Descent in z')
legend('v_z:Euler','v_z:ode45','Location','bestoutside')

sgtitle('Velocity Plots: v_z(t)')
```





Exercise 2

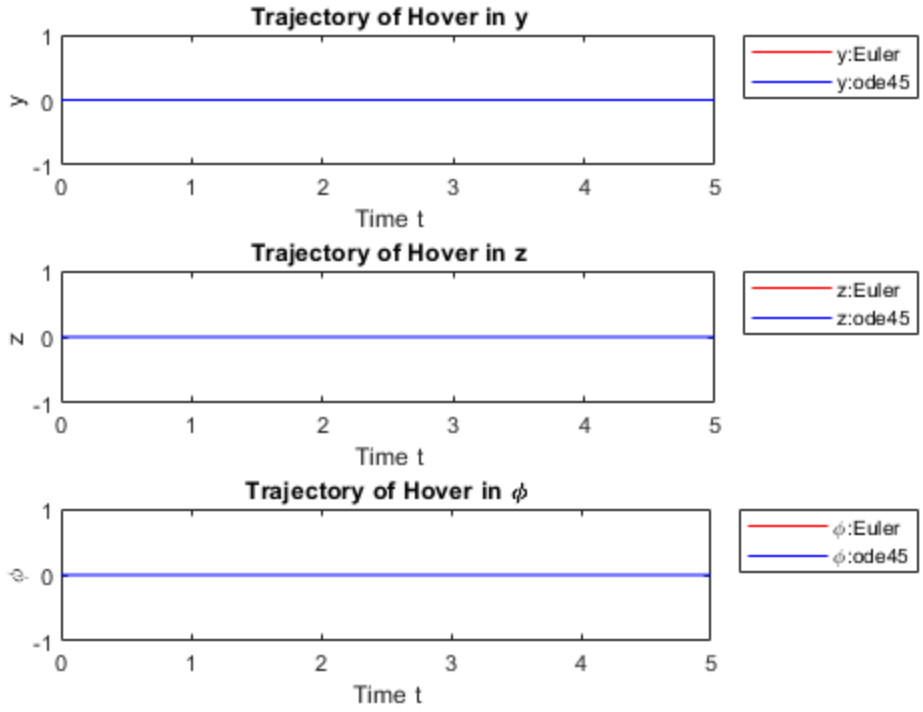
simulate 2D in the Hover state Create a new function myODE2D

```
% Best Practices ------------------------------------------------
clear all;
close all;

% Initial time
t_0 = 0;

% Final time
t_f = 5;

% Event Horizon: [t_0 , t_f]
horizon = t_f - t_0;

% Number of time steps
N=10;

% Initial conditions
%   set: y(0) = vy(0) = z(0) = vz(0) = phi(0) = vphi(0) = 0
i_c = [0; 0; 0; 0; 0; 0;]; % 6x1

%Time step
delta = (horizon)/N;        % Time step = Horizton /number of steps
```

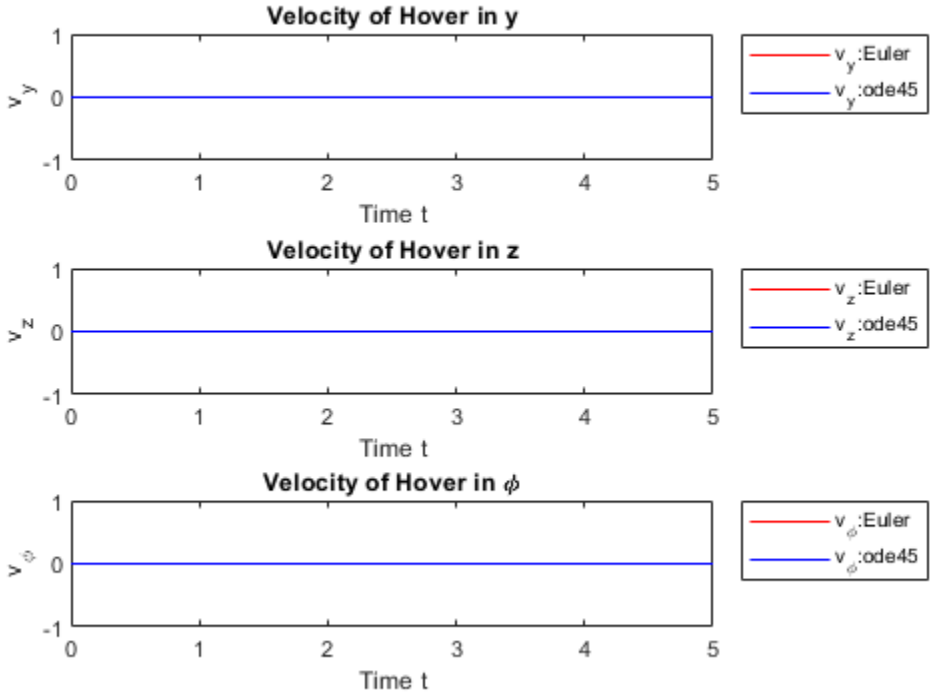
```
% Initial Time
t(1) = t_0;

% Initialize x for HOVER
xHe(:,1) = i_c;              % Hover

% Realized all the values are zero, so I simplified it

% Set Gravity and mass
%g = 9.81; %m/s^2
%m = 1;

% Set Force for HOVER in 2D % Fz- needs to be iterated
%Fy = 0;
%M  = 0;


% For loop: sets next (t,x) = (t,(z,vz)) values
for k=1:N
    % Time used for plotting the domain- not used in function
    t(k+1) = t(k) + delta;

    % Calls the function f(t,x) = dx/dt: used in Forward Euler
    % Hover
    xHe(:,k+1) = xHe(:,k) + delta*myODE2D(t(k),xHe(:,k));

end

% Load Variables from ode_sim_ode45.m
load("ode45Data2D.mat","xH","t45")

% % Plots-------------------------------------
% Position x(1,:)
% Mind the dimension
figure(1);
subplot(3,1,1)
plot(t,xHe(1,:), Color='r');
hold on
plot(t45,xH(:,1),Color ='b');
xlabel('Time t');
ylabel('y');
title('Trajectory of Hover in y')
legend('y:Euler','y:ode45','Location','bestoutside')

subplot(3,1,2)
plot(t,xHe(3,:), Color='r');
hold on
plot(t45,xH(:,3),Color ='b');
xlabel('Time t');
ylabel('z');
title('Trajectory of Hover in z')
legend('z:Euler','z:ode45','Location','bestoutside')
```

```
subplot(3,1,3)
plot(t,xHe(5,:), Color='r');
hold on
plot(t45,xH(:,5),Color ='b');
xlabel('Time t');
ylabel('\phi');
title('Trajectory of Hover in \phi')
legend('\phi:Euler','\phi:ode45','Location','bestoutside')
sgtitle('Trajectory Hover Plots: y(t), z(t),\phi(t)')

% Velocity x(2,:)
figure(2);
subplot(3,1,1)
plot(t,xHe(2,:),Color='r');
hold on
plot(t45,xH(:,2),Color= 'b')
xlabel('Time t');
ylabel('v_y');
title('Velocity of Hover in y')
legend('v_y:Euler','v_y:ode45','Location','bestoutside')

subplot(3,1,2)
plot(t,xHe(4,:), Color='r');
hold on
plot(t45,xH(:,4),Color ='b');
xlabel('Time t');
ylabel('v_z');
title('Velocity of Hover in z')
legend('v_z:Euler','v_z:ode45','Location','bestoutside')

subplot(3,1,3)
plot(t,xHe(6,:), Color='r');
hold on
plot(t45,xH(:,6),Color ='b');
xlabel('Time t');
ylabel('v_\phi');
title('Velocity of Hover in \phi')
legend('v_\phi:Euler','v_\phi:ode45','Location','bestoutside')

sgtitle('Velocity Hover Plots: v_y(t), v_z(t),v_\phi(t)')
```

Trajectory Hover Plots: y(t), z(t),ϕ(t)



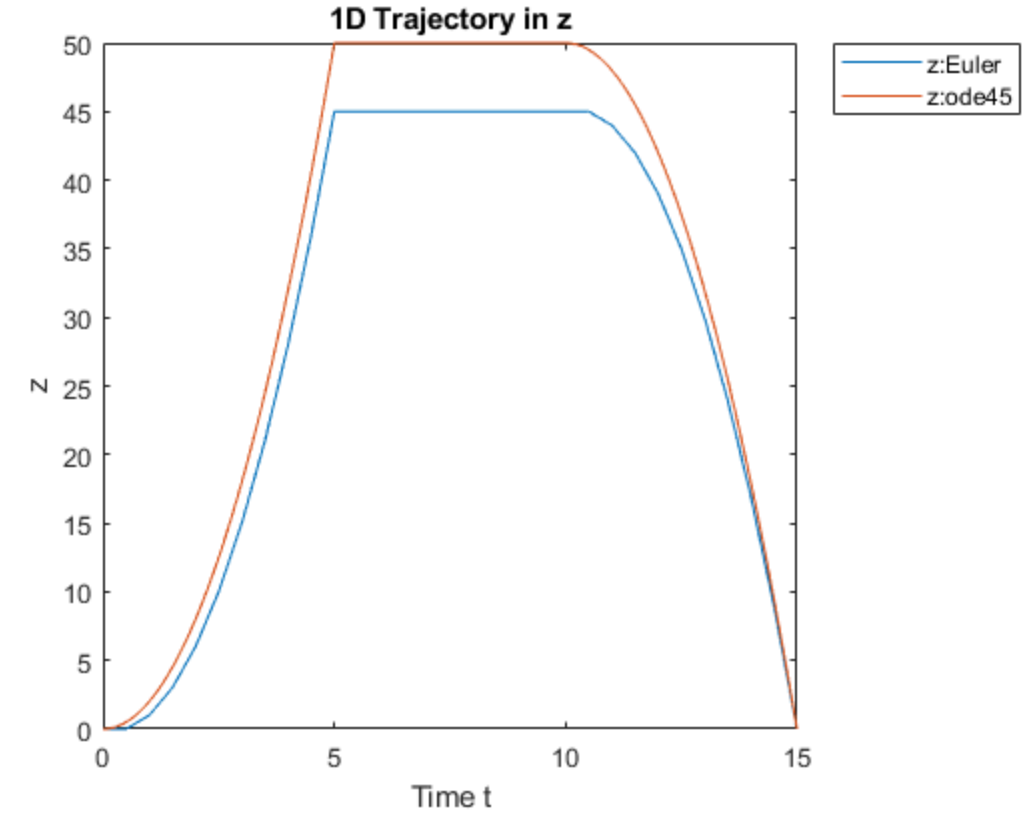
Velocity Hover Plots: v_y(t), v_z(t),v_ϕ(t)

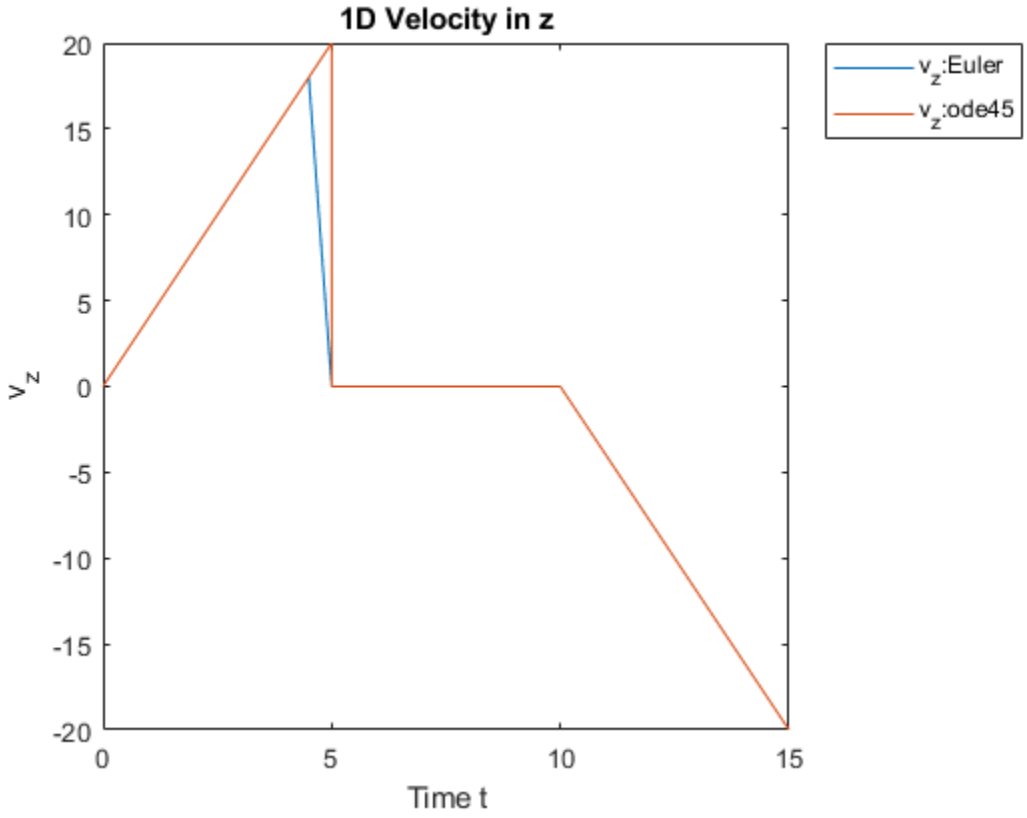


Exercise 3

```
% 1D sim
% Ascend for 5s, Hover for 5s, descend for 5s
% Too tired to figure out the correct way, we are going to brute for this

% Best Practices ---------------------------------------------
clear all;
close all;

% Initial time
t_0 = 0;

% Final time
t_f = 15;

% Event Horizon: [t_0 , t_f]
horizon1 = 5 - t_0;
horizon2 = 10 - 5;
horizon3 = t_f - 10;

% Number of time steps
N=10;

% Initial conditions
i_c = [0; 0];               % set z(0) = v_z(0) = 0

%Time step
% Delta is the same for all 3 intervals so just use one
delta = (horizon1)/N;        % Time step = Horizton /number of steps

% Ascend -----------------------------------
% Initial Time
t(1) = t_0;

% Initialize x
xe(:,1) = i_c;              % Ascend

% Set Gravity
g = 9.81; %m/s^2

% Set Force:
%   Eq: Fz = Fnet = 4F -g
FA = g/4 + 1;               % Ascend
FH = g/4;                   % Hover
FD = g/4 - 1;               % Desend

% For loop
for k=1:N
    % Time used for plotting the domain- not used in function
    t(k+1) = t(k) + delta;

    % Calls the function f(t,x) = dx/dt: used in Forward Euler (vz,az)
```

```
    % Ascend
    xe(:,k+1) = xe(:,k) + delta*myODE(t(k),xe(:,k), FA);
end

% New IC for Hover
xe(2,end) = 0;

% Append the matrix xe for each section
for k=length(xe):N+length(xe)-1
    % Time used for plotting the domain- not used in function
    t(k+1) = t(k) + delta;

    % Calls the function f(t,x) = dx/dt: used in Forward Euler (vz,az)
    % Ascend
    xe(:,k+1) = xe(:,k) + delta*myODE(t(k),xe(:,k), FH);
end

% IC is fine to take the values of xe
for k=length(xe):N+length(xe)-1
    % Time used for plotting the domain- not used in function
    t(k+1) = t(k) + delta;

    % Calls the function f(t,x) = dx/dt: used in Forward Euler (vz,az)
    % Ascend
    xe(:,k+1) = xe(:,k) + delta*myODE(t(k),xe(:,k), FD);
end

% Load Variables from ode_sim_ode45.m
load("ode45AHD.mat","x","t45AHD")

% Plots
figure(1)
box on
plot(t,xe(1,:))
hold on
plot(t45AHD,x(:,1))
xlabel('Time t');
ylabel('z');
title('1D Trajectory in z')
legend('z:Euler','z:ode45','Location','bestoutside')

figure(2)
box on
plot(t,xe(2,:))
hold on
plot(t45AHD,x(:,2))
xlabel('Time t');
ylabel('v_z');
title('1D Velocity in z')
legend('v_z:Euler','v_z:ode45','Location','bestoutside')
```


1D Trajectory in z
z:Euler
z:ode45
z
Time t
0
5
10
15
0
5
10
15
20
25
30
35
40
45
50


1D Velocity in z
v_z:Euler
v_z:ode45
v_z
Time t
0
5
10
15
-20
-15
-10
-5
0
5
10
15
20

Published with MATLAB® R2022b