Table of Contents

```
clear; clc; close all;

% ASEN 6060 - HW 5, Prob 1
% Spring 2025
% Jash Bhalavat
```

Constants

```
G = 6.67408 * 10^-11; % m3/(kgs2)
G = G / (10^9); % km3/(kgs2)

% Earth
mu_earth = 398600.435507; % km3/s2
a_earth = 149598023; % km
e_earth = 0.016708617;
mass_earth = mu_earth / G; % kg

% Moon
mu_moon = 4902.800118; % km3/s2
a_moon = 384400; % km
e_moon = 0.05490;
mass_moon = mu_moon / G; % kg

% Earth-Moon system
mass_ratio_em = mass_moon / (mass_earth + mass_moon);
m_star_em = mass_earth + mass_moon;
l_star_em = a_moon;
t_star_em = sqrt(l_star_em^3/(G * m_star_em));
mu = mass_ratio_em;

global count poincare_stored
poincare_stored = [];
```

Part a

```
c_given = 3.175;
% C = x^2 + y^2 + 2*(1-mu)/r1 + mu/r2 - (x_dot^2 + y_dot^2 + z_dot^2);

n = 1001;
x = linspace(-2, 2, n);
y = linspace(-2, 2, n);
```

```
% zvc = full_zvc(c_given, mu, x, y);
% plot_zvc(zvc, x, y, c_given, mu)

N_IC = 100;
n_crossings = 300;

r_E = 6378.1363; % [km] Earth equatorial radius (Vallado, Appendix D)
r_Moon = 1738; % [km] Moon equatorial radius (Vallado, Appendix D)
r_E_normalized = r_E/a_moon;
r_Moon_normalized = r_Moon/a_moon;

y_min_pos = r_Moon_normalized;
y_min_neg = -r_Moon_normalized;

options = optimset('Display','off');
y_max_pos = fsolve(@(y)fn(y, mu, c_given), 0.108, options);
% y_max_neg = -y_max_pos;

y_max_pos = 0.108;
y_max_neg = -y_max_pos;

y_range_pos = linspace(y_min_pos, y_max_pos, N_IC);
y_range_neg = linspace(y_min_neg, y_max_neg, N_IC);

% Set options for ODE45
options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
eventFn(t, y, mu, n_crossings));

for i = 1:N_IC
    % global variables store the number of crossings and poincare points
    global count;
    global poincare_stored;

    % Need to zero out count at each iteration so that event function
    % propagates until total number of crossings
    count = 0;

    fprintf("Current IC Number - %d\n", i)

    % Calculate initial state
    U_star_times_2 = u_star_times_2(1-mu, y_range_pos(i), mu);
    x_dot_0 = +sqrt(U_star_times_2 - c_given);
    % Assuming perpendicular veloities, therefore ydot = 0
    x0 = [1-mu, y_range_pos(i), 0, x_dot_0, 0, 0];

    [tout, xout] = ode45(@(t, state)CR3BP(state, mu), [0 1000], x0, options);

    % Repeat for negative y range
    count = 0;
    x0 = [1-mu, y_range_neg(i), 0, x_dot_0, 0, 0];
    [tout, xout] = ode45(@(t, state)CR3BP(state, mu), [0 1000], x0, options);
end
```

Plot

```
figure()
scatter(poincare_stored(:,2), poincare_stored(:,1), 2, 'filled', 'black');
ylabel("y")
xlabel("$\dot{y}$", 'Interpreter','latex')
title("Poincar\'e Map", 'Interpreter','latex')
grid on
```

Functions

```
function [y_poincare, y_dot_poincare] = find_yydot(tout, xout, y_poincare,
y_dot_poincare, mu)
    for i = 1:length(tout)
        if (abs(xout(i,1) - (1-mu)) < 1e-12 && xout(i,4) > 0)
            y_poincare = [y_poincare, xout(i,2)];
            y_dot_poincare = [y_dot_poincare, xout(i,5)];
        end
    end
end

function zvc = full_zvc(c_given, mu, x, y)
    zvc = ones([length(x), length(y)]);

    for i = 1:length(x)
        for j = 1:length(y)
            c_calc = u_star_times_2(x(i), y(j), mu);
            if c_calc < c_given
                zvc(i, j) = -1;
            end
        end
    end
end

function plot_zvc(zvc, x, y, c, mu)
    figure()
    contourf(x, y, zvc')
    map = [220/255, 220/255, 220/255
        1, 1, 1];
    colormap(map)
    hold on
    scatter(-mu, 0, 200, 'filled')
    scatter(1-mu, 0, 50, 'filled')
    hold off
    legend("", "Earth", "Moon")
    xlabel("x [Non-Dimensional]")
    ylabel("y [Non-Dimensional]")
    title("Zero velocity curve for Jacobi Constant C = " + num2str(c))
end

function out = u_star_times_2(x, y, mu)
    r1 = sqrt((x + mu)^2 + y^2);
    r2 = sqrt((x - 1 + mu)^2 + y^2);
    out = (x^2 + y^2) + 2*(1 - mu)/r1 + 2*mu/r2;
end

function [value,isterminal,direction] = eventFn(t,y,mu,n_crossings)
    % Call global variables. This restores the current total crossings and
    % all the stored poincare points.
    global count;
    global poincare_stored;

    tol = 1e-12;

    % Get the normalized moon radius
    r_Moon = 1738; % [km] Moon equatorial radius (Vallado, Appendix D)
    a = 384400;
    r_Moon_normalized = r_Moon/a;

    % Moon wrt spacecraft
    p2_pos = [1-mu, 0, 0]';
    p2_minus_pos = p2_pos - y(1:3);

    if t == 0
        % Avoid initial points to be captured in the poincare map
        value = 10;
        isterminal = 0;
        direction = 0;
    elseif (norm(p2_minus_pos) < r_Moon_normalized)
        % Avoid trajectories that intersect the surface of the moon
        value = 0;
        isterminal = 1;
        direction = 0;
    elseif count < n_crossings
        % When crossing cap hasn't been met yet, find value and if the
        % value is close to 0, increase the count and store the y,ydot
        % values.
        value = y(1) - (1-mu);
        isterminal = 0;
        direction = 1;
        if (abs(value) < tol && y(4) > tol)
            count = count + 1;
            poincare_stored = [poincare_stored; y(2), y(5)];

        end
    elseif count == n_crossings
        % When crossing cap has been met, terminate integration
        value = y(1) - (1-mu); % Want x to be 1-mu
        isterminal = 1; % Halt integration when value is 0
        direction = 1; % When zero is approached from +ve i.e. x_dot > 0
        % poincare_stored = [poincare_stored; y(2), y(5)];
    end
end

function out = fn(y, mu, cj)
```

```
    out = cj - (1-mu)^2 - y^2 - (2*(1-mu))/(sqrt(1+y^2)) - (2*mu)/(sqrt(y^2));
end
```