ASEN 6060 - HW 5
Spring 2025
Jash Bhalavat

Problem 1 - Part a

1,2) Here is the state vector at the initial condition:

$$\bar{x}_0 = [x_0,\ y_0,\ z_0,\ \dot{x}_0,\ \dot{y}_0,\ \dot{z}_0]^T$$

We already know that the surface of section is at $x = 1 - \mu$ and so the initial x state (x_0) has to be $1 - \mu$. Let's also select the initial z state so that the projection is only in the xy plane i.e. $z_0 = 0$. So, within this configuration, y_0 remains the only position component that can be varied. So, range of the y_0 component can be from the moon's surface to the zero velocity curve. The moon's radius is 1738 km (Vallado, Appendix D). The earth-moon system semi-major axis is 384,400 (HW 1 constants). The average distance between the earth and the moon is 384,400 km (and in the phase space, it's 1). Thus, the moon's normalized radius is the physical radius divided by the semi-major axis and it turns out to be - 0.0045213319458897. The maximum range is the edge of the zero-velocity curve (when C = 2U*).

$$2U^* = x^2 + y^2 + 2 * \frac{1-\mu}{r_1} + \frac{2\mu}{r_2} = C = 3.175$$

Since, x = 1-μ, y can be found (fsolve is used in this case) to be 0.108872496080861. (NOTE - Since, at this point the zero-velocity curve begins, the maximum is taken at 0.108). Hence, the range for y_0 is - [0.0045213319458897, 0.108] and [-0.0045213319458897, -0.108] (since this system is symmetric).

Next, the initial velocities components. We also let $\dot{z}_0$ be 0 because we want these orbits to be planar. Then, $\dot{y}_0$ is also chosen to be 0 because we want a perpendicular crossing of the surface of section. Lastly, the $\dot{x}_0$ is chosen based on the following equation:

$$v^2 = 2U^* - C$$

$$\dot{x}_{i,0} = \pm\sqrt{2U^* - C}$$

I always choose the positive velocity because the surface of section is one sided and we want positive x-velocity crossings. So, the initial condition state vector is as follows:

$$\bar{x}_0 = [1 - \mu,\ y_{i,0},\ 0,\ \dot{x}_{i,0},\ 0,\ 0]^T$$

3) The number of crossings was selected to be 300 based on the recommendation from the homework problem statement. More crossings were attempted, but it did not result in a more effective map so 300 was finalized. (Further explanation of the event function is given below)

The initial condition for loop is shown below:

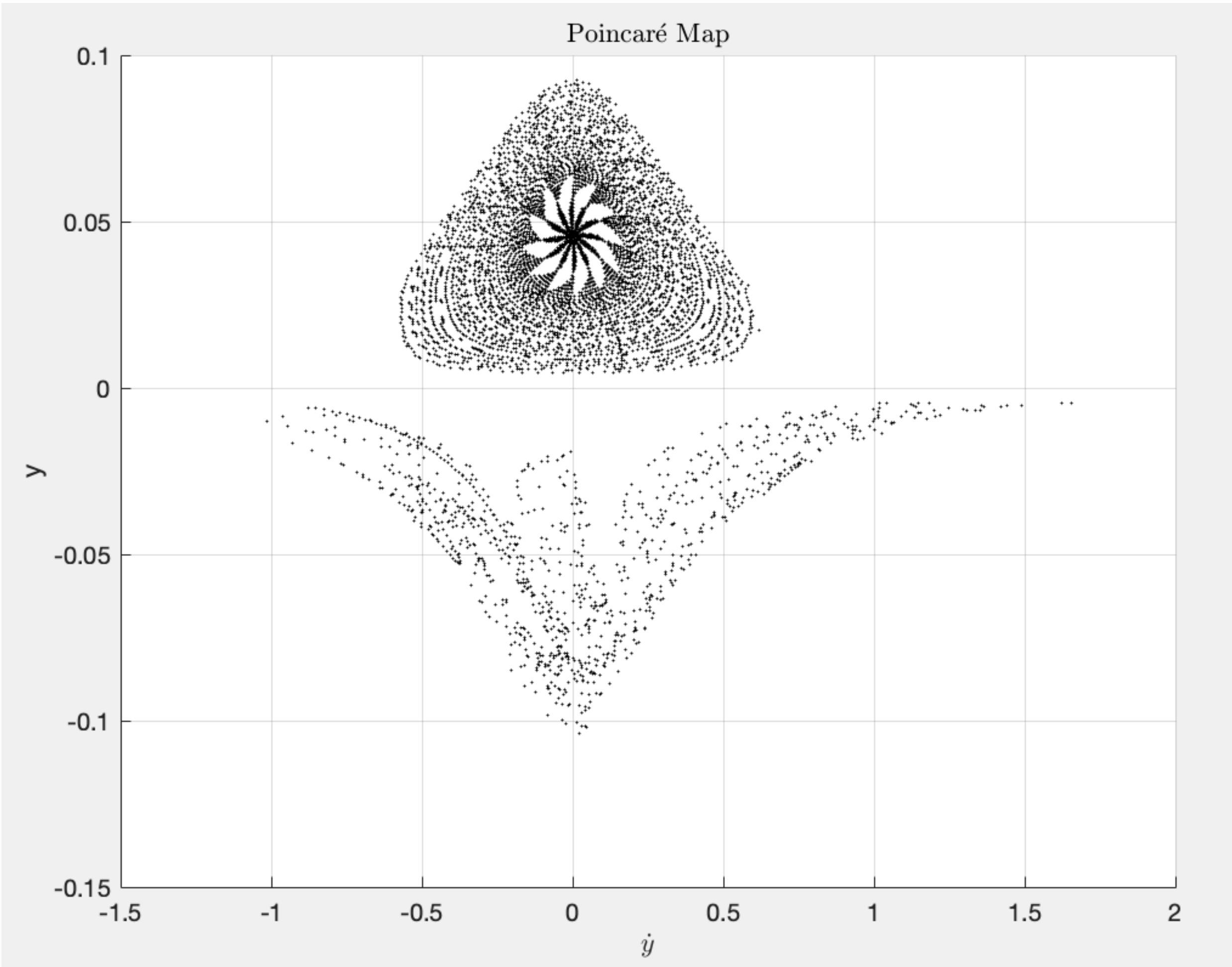
```
for i = 1:N_IC
    % global variables store the number of crossings and poincare points
    global count;
    global poincare_stored;

    % 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
```

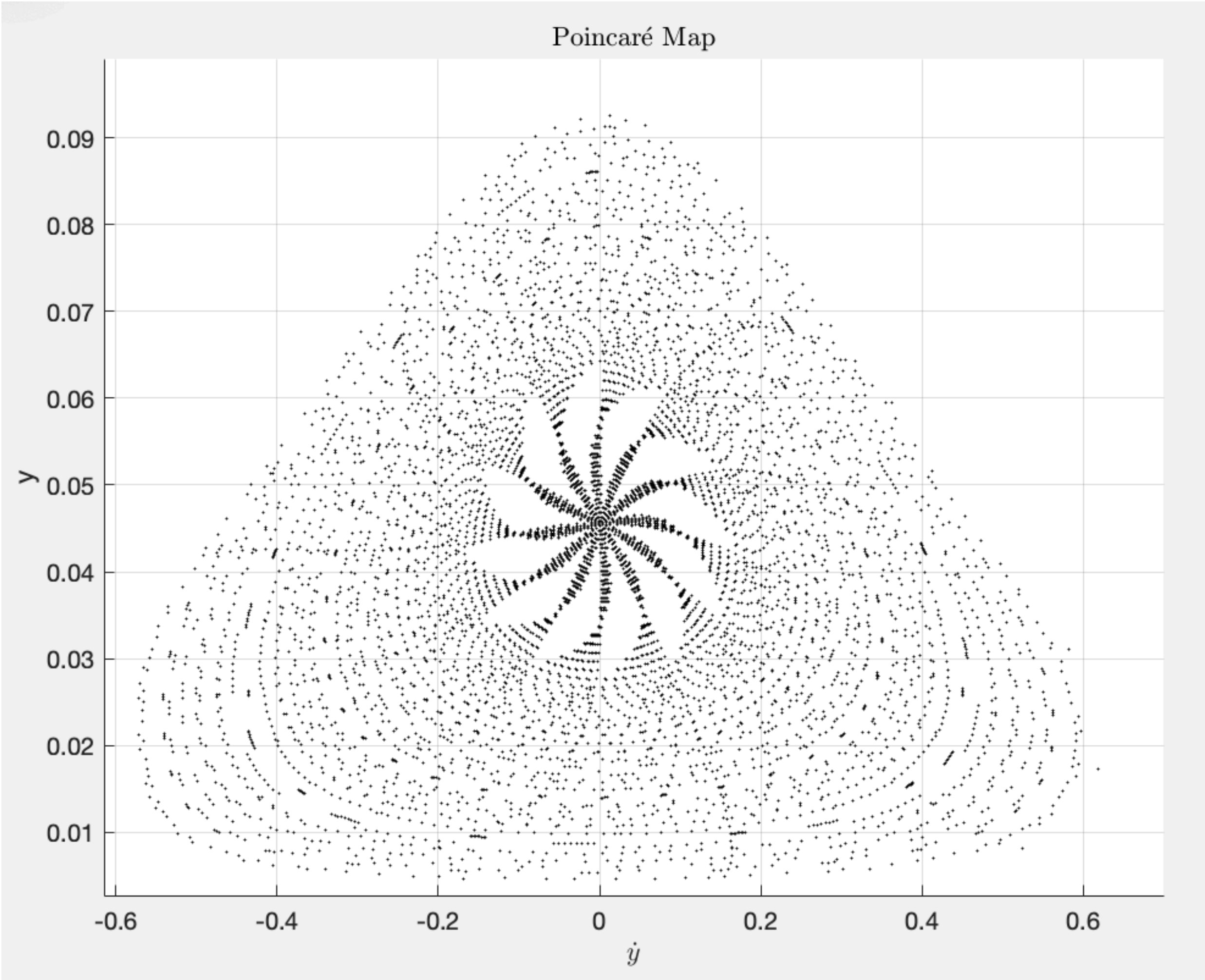
(the y range discussed above is stored as a 1d array)

Additionally, the Poincaré map event function is shown below. The event function has 3 outputs - value, isterminal, and direction. The event function has a global variable count that is iterated every time the trajectory crosses the surface of section with a positive x-component velocity. When that happens, the y and y-component velocity are stored to be plotted later onto the Poincaré map. Note that the propagation is terminated if the trajectory is hitting the moon's surface. And, an additional if statement is added to check if the event function isn't triggered due to the initial condition.

```matlab
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
```

Problem 1 - Part b





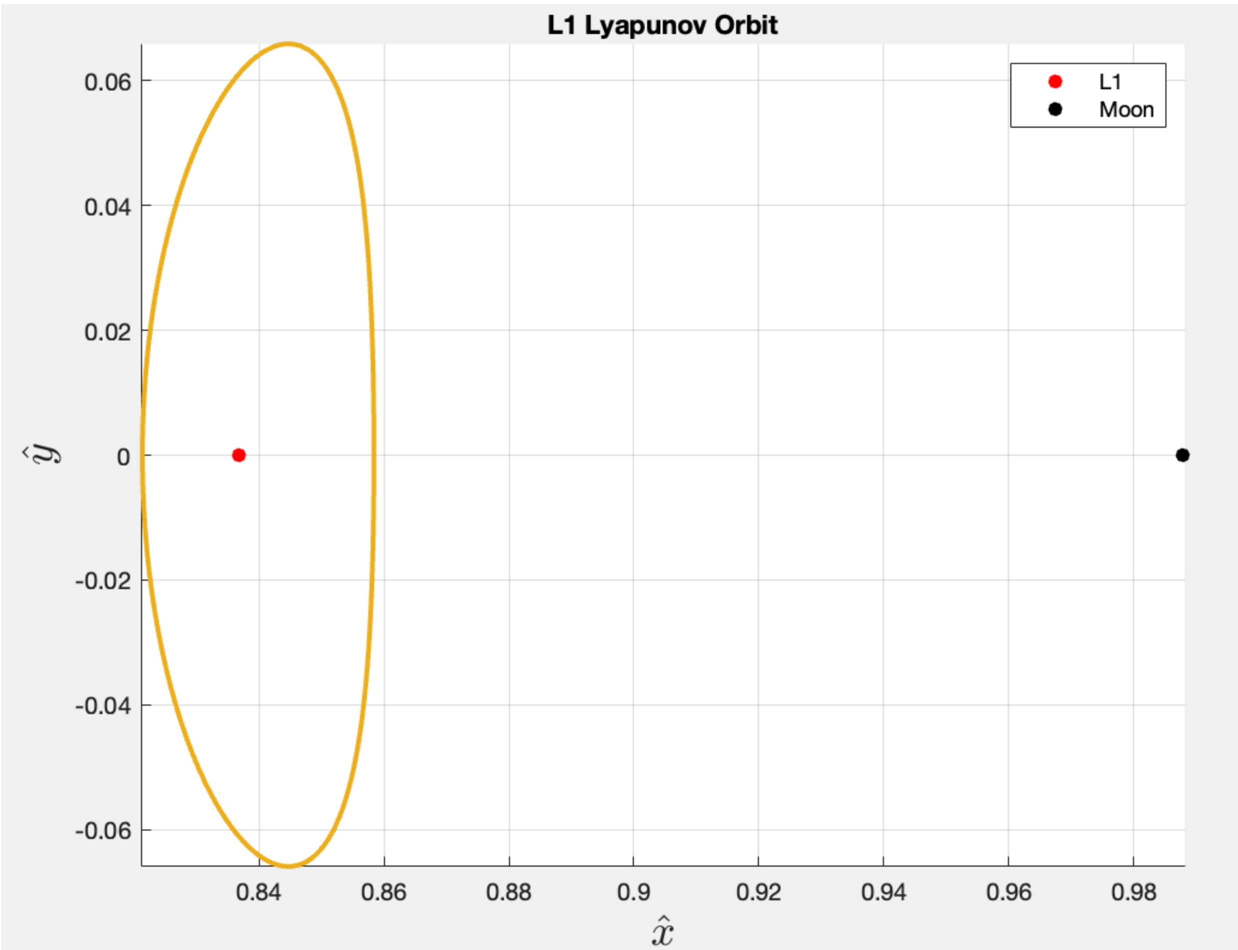
(Zooming into only the top half of the map only for visualization purposes)

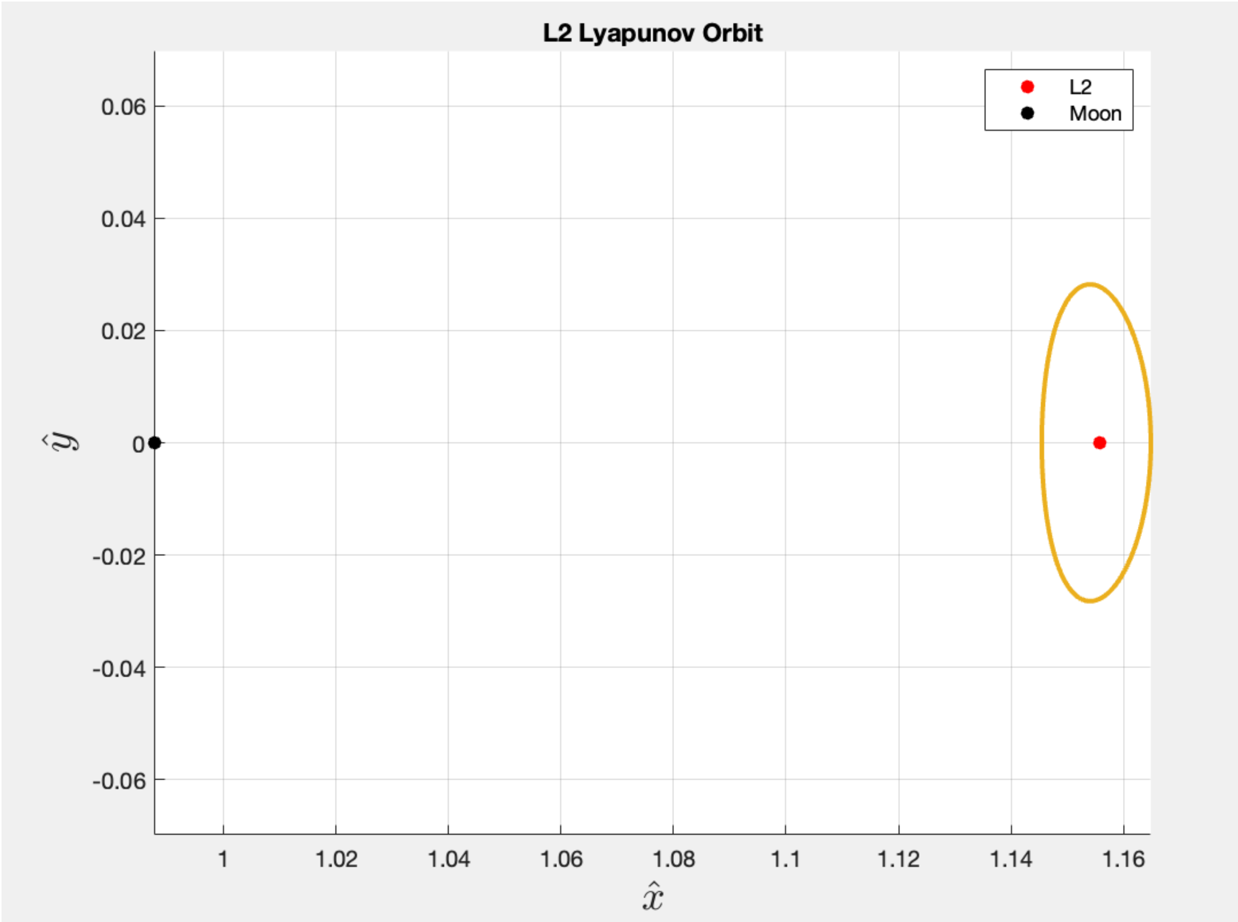
Problem 1 - Part c

The Poincaré map above shows a stable periodic orbit near the center of the map. This can be deduced from the following facts: A periodic orbit on the Poincaré map shows up as only one point. The center of the map shows one such orbit. A quasi-periodic orbit shows up as a closed curve. Since the periodic orbit is surrounded by a number of quasi-periodic orbits, it can be concluded that the periodic orbit is stable. All of the above observations are true for the region above the moon. When the negative y region is looked at, a lot of chaotic behavior is observed. This indicates that the positive y region has a much better chance of a periodic/quasi-periodic orbit than the negative y region. That makes sense because the positive y region is prograde motion for the Earth-Moon system.

Problem 2 - Part a

Using the initial conditions provided and correcting using the method from HW 3, these are the L1 and L2 Lyapunov orbits computed:





The state vector at one point along the orbit: Corrected initial state -

L1 Lyapunov Orbit - $\bar{x}$ = [0.821384937691756, 0, 0, 0, 0.147514346876754, 0], T = 2.76329895885651

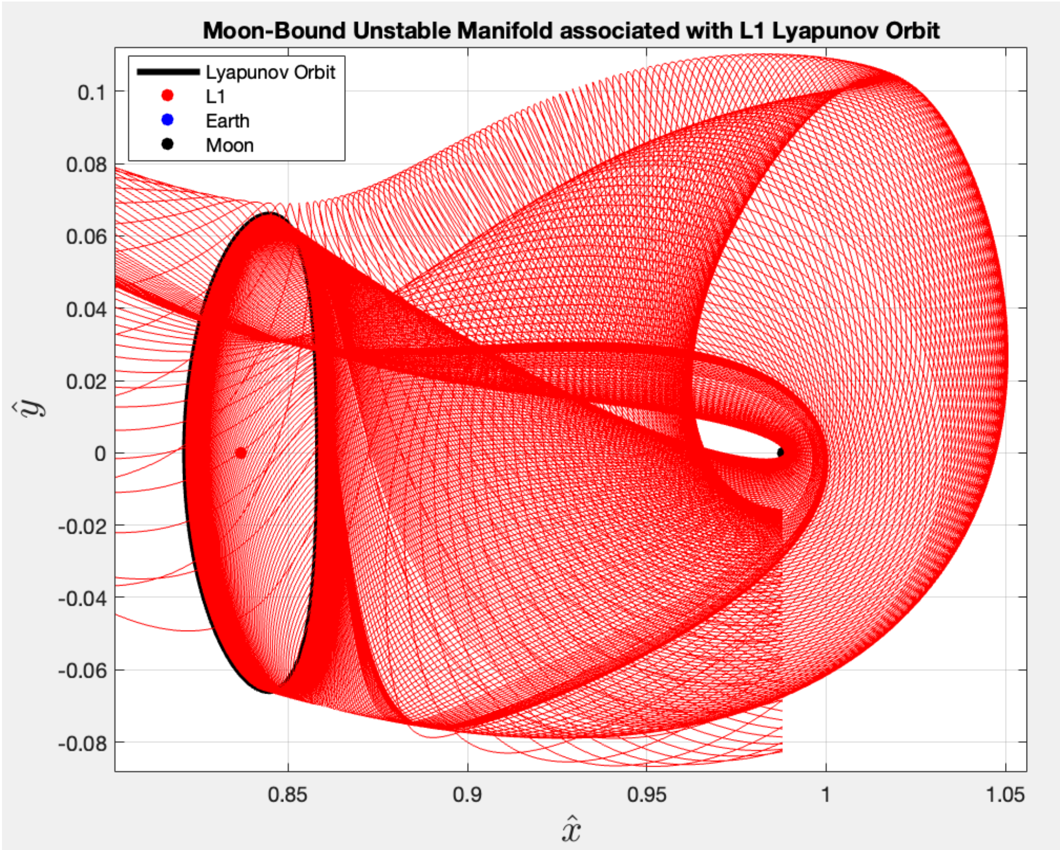
The Jacobi constant for this orbit is 3.16916035284557.

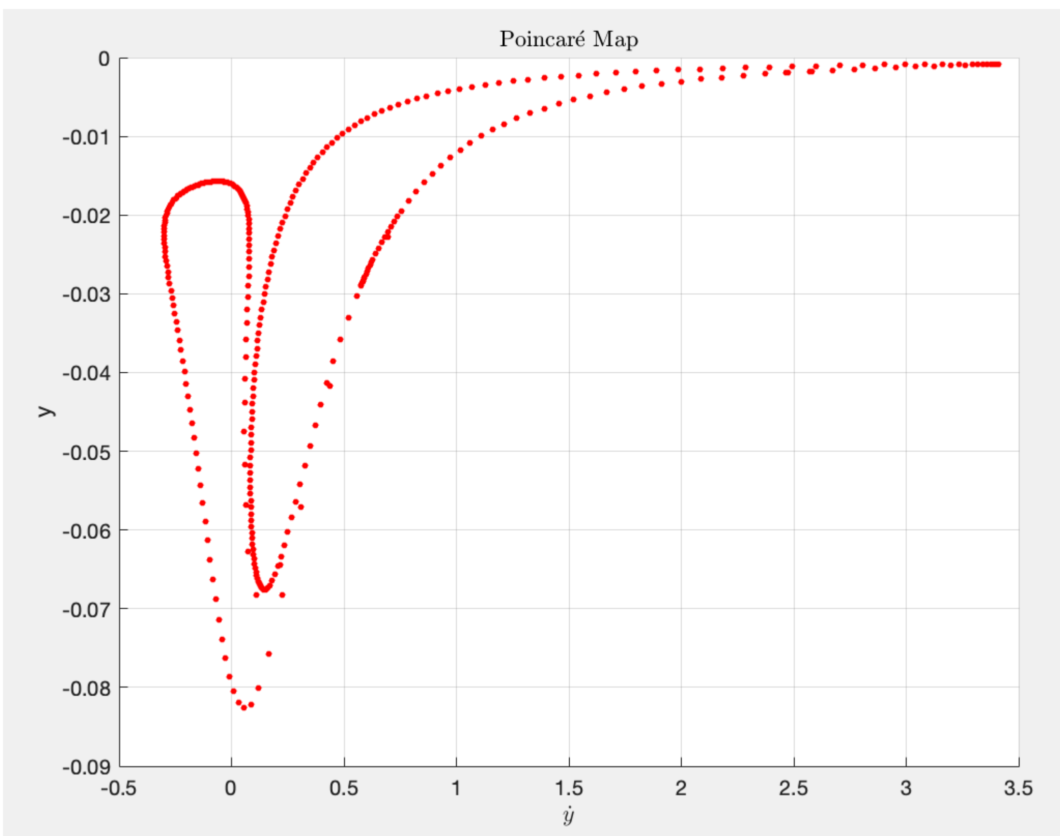
L2 Lyapunov Orbit - $\bar{x}$ = [1.16485510668702, 0, 0, 0, -0.0516671532607056, 0], T = 3.37721354575301

The Jacobi constant for this orbit is 3.17008877477016.

Problem 2 - Part b

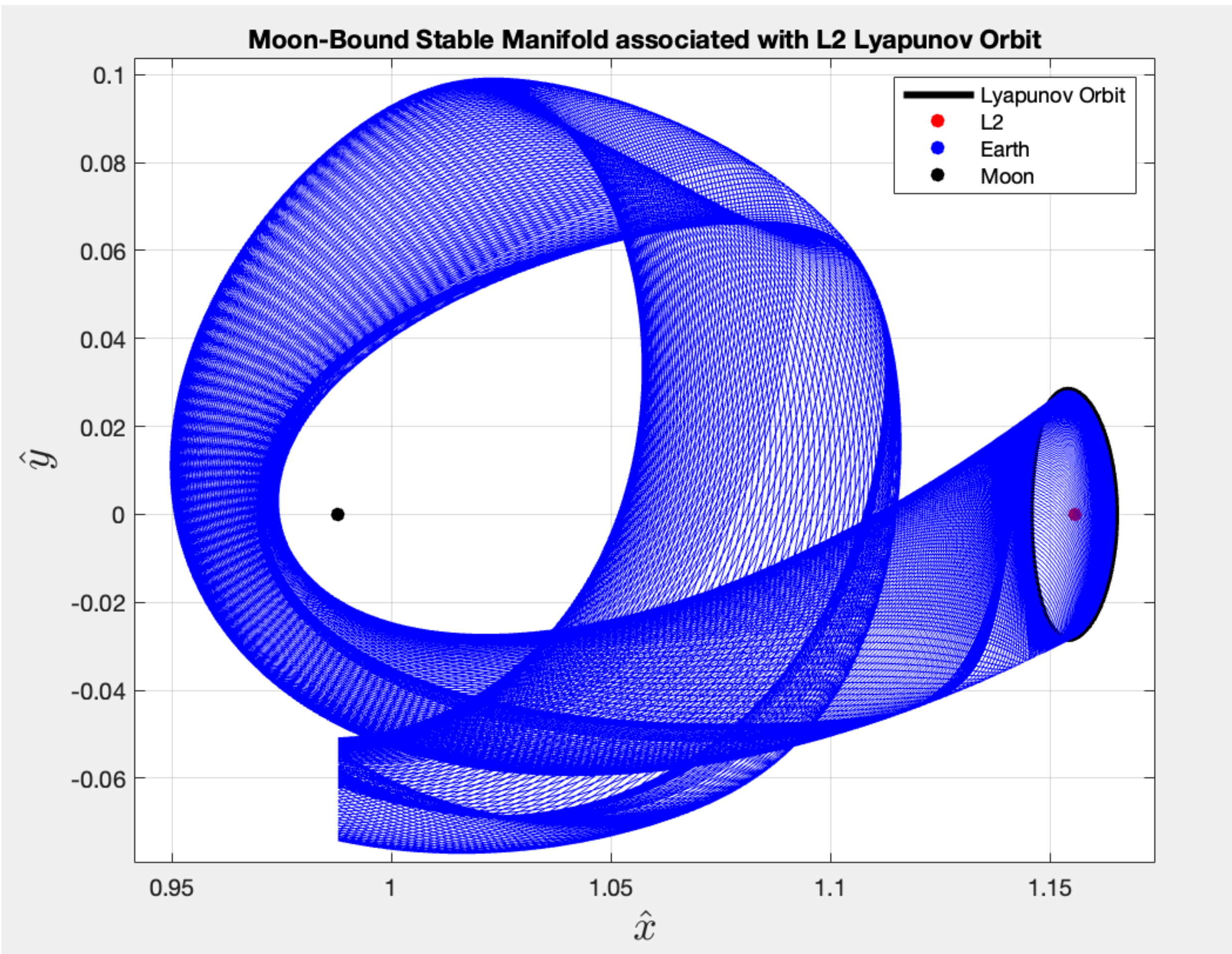
Using the approach from HW 4 to generate the unstable half-manifold towards the moon from the L1 Lyapunov orbit. The moon-bound unstable half manifold from L1 Lyapunov orbit is shown below (the terminating condition is a surface of section at x = 1-μ with 2 positive crossings) along with the Poincaré map:


Moon-Bound Unstable Manifold associated with L1 Lyapunov Orbit
Lyapunov Orbit
L1
Earth
Moon
0.1
0.08
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
-0.08
0.85
0.9
0.95
1
1.05
$\hat{x}$
$\hat{y}$

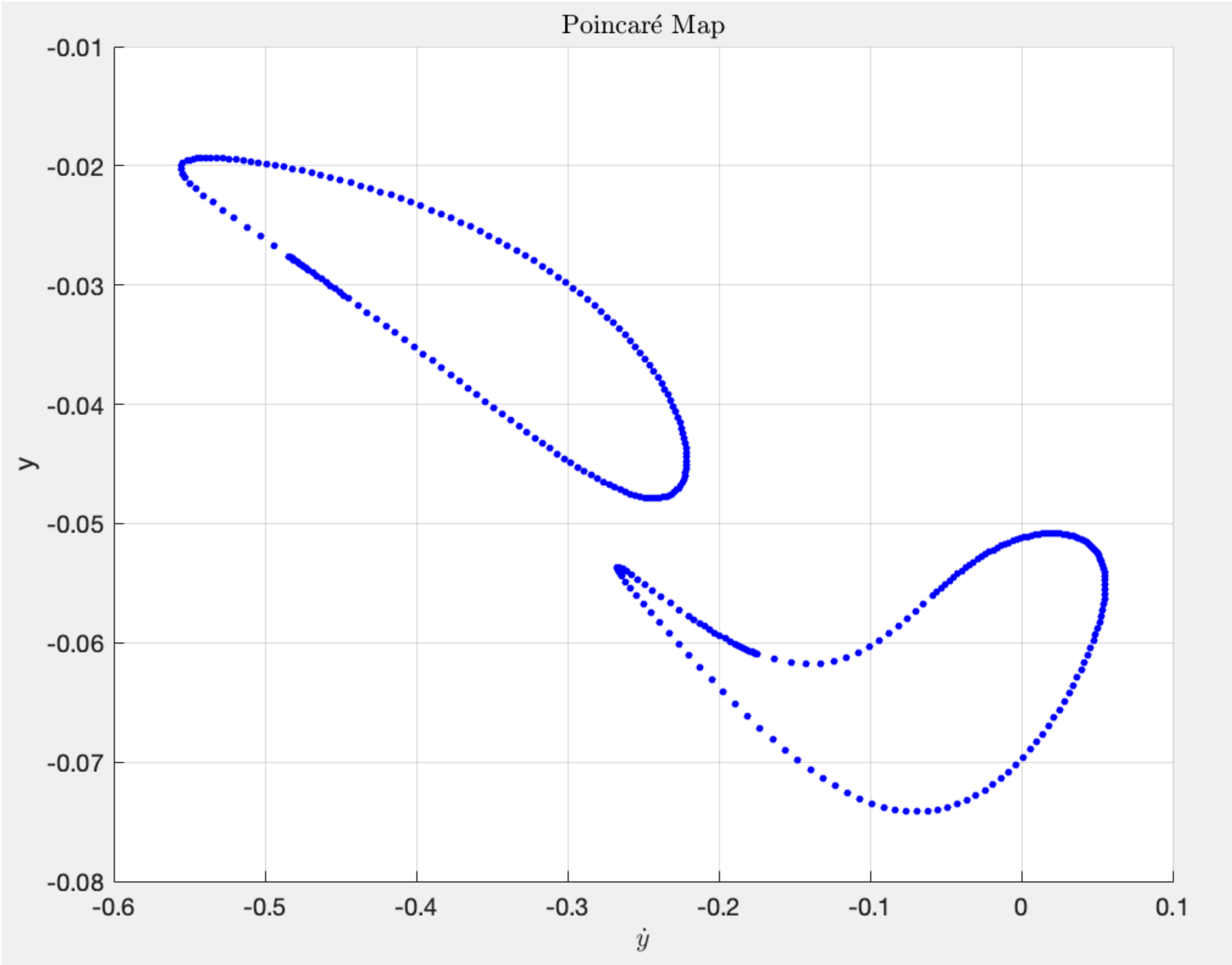

Poincaré Map
0
-0.01
-0.02
-0.03
-0.04
-0.05
-0.06
-0.07
-0.08
-0.09
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
$\dot{y}$
y

Problem 2 - Part c

Using the same approach as part b, the moon-bound stable half manifold from the L2 Lyapunov orbit is shown below (the terminating condition is a surface of section at x = 1-μ with 2 positive crossings) along with the Poincaré map:

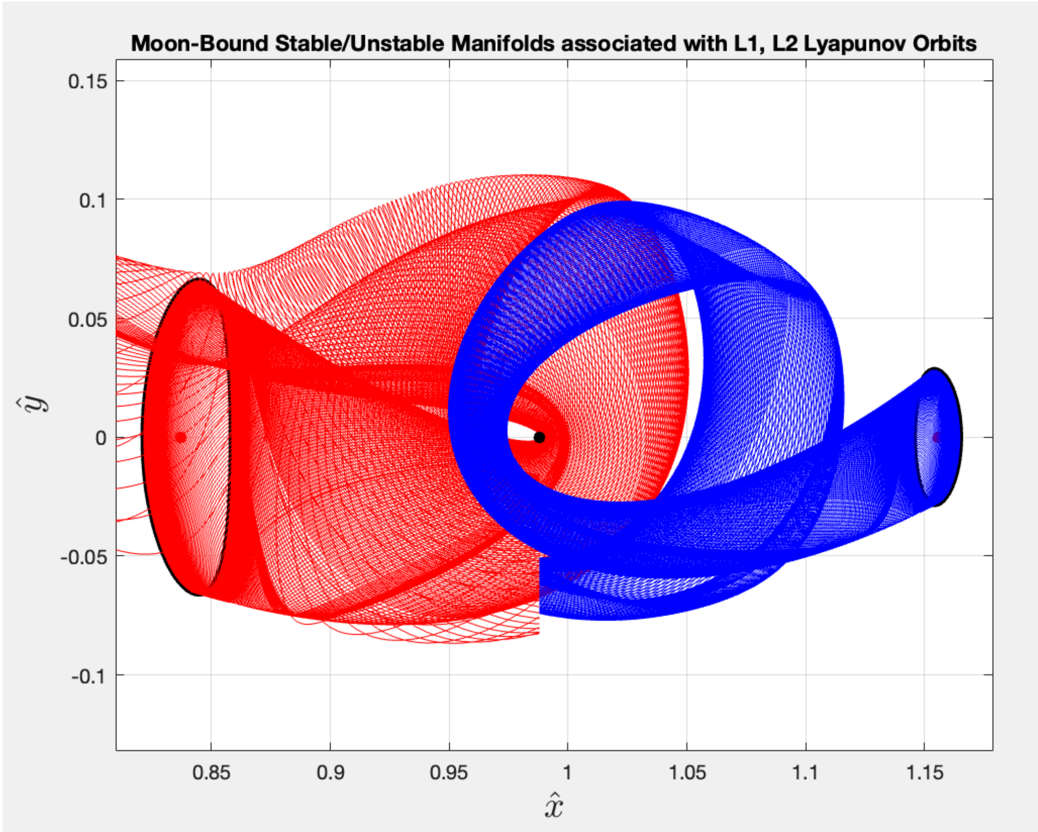


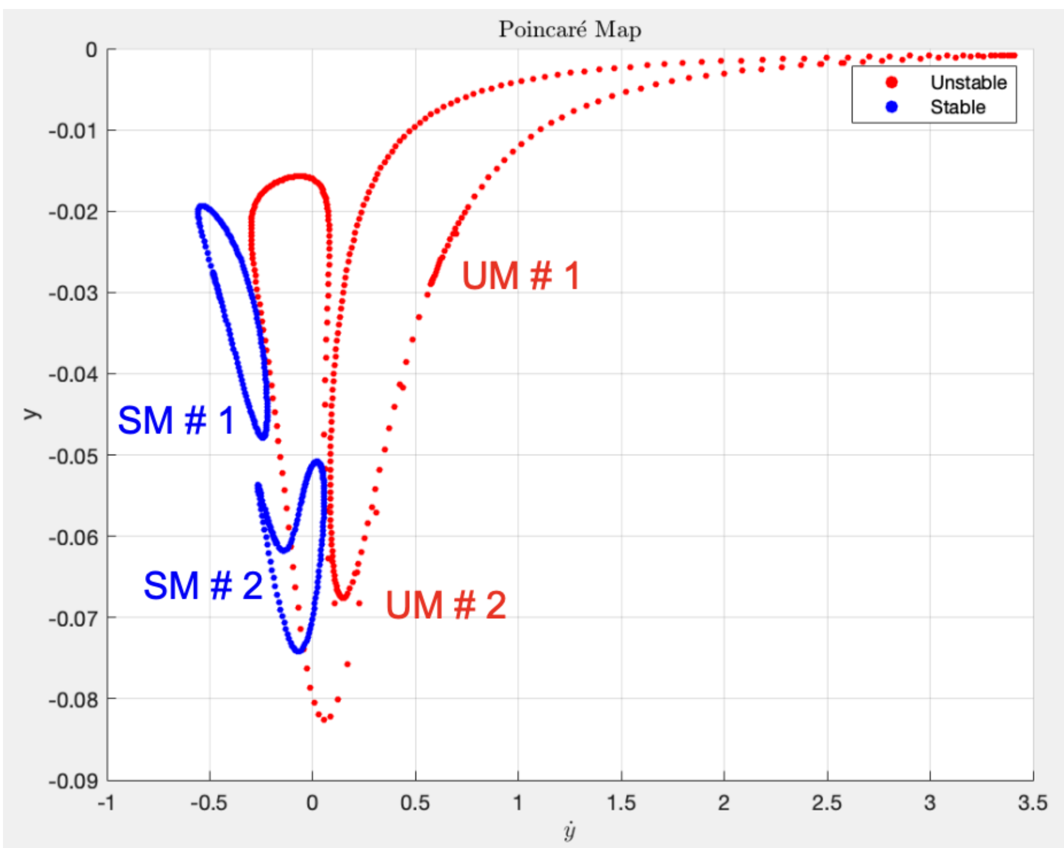
Then, using a surface of section at x = 1-μ and 2 positive crossings, the Poincaré map is shown below:



Problem 2 - Part d

Firstly, plotting the L1 unstable and L2 stable manifolds from parts b, c together.


Moon-Bound Stable/Unstable Manifolds associated with L1, L2 Lyapunov Orbits
0.15
0.1
0.05
0
-0.05
-0.1
0.85
0.9
0.95
1
1.05
1.1
1.15
$\hat{y}$
$\hat{x}$


Poincaré Map
0
-0.01
-0.02
-0.03
-0.04
-0.05
-0.06
-0.07
-0.08
-0.09
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
y
$\dot{y}$
Unstable
Stable
UM # 1
UM # 2
SM # 1
SM # 2

In order to go from from L1 Lyapunov orbit to L2 Lyapunov orbit using a natural trajectory, the SM #2 and UM #2 points on the Poincaré map can be looked at. On the Poincaré map, two points where SM #2 and UM #2 intersect, are the same states. Since, y and $\dot{y}$ are the same at that point (as seen on the map), and x is the same (because all these points are captured at the same surface of section), z and $\dot{z}$ are the same (planar trajectories). If all of these are the same and since the Jacobi constant is the same for both these manifolds (3.17), the $\dot{x}$ component also has to be the same, so the states are the same. Hence, when SM #2 and UM #2 intersect, the trajectory naturally goes from L1 Lyapunov orbit to L2 Lyapunov orbit. These intersections are good places for initial guesses for the transfer.

In order to get distinct geometries, the crossing identifier needs to be looked at. If a particular transfer goes from UM #1 to SM #1, then the transfer trajectory does not orbit around the moon at all. If the transfer goes from UM #1 to SM #2, then the transfer trajectory orbits around the moon twice.

In an impulsive maneuver, the position of the P3 body does not change and only the velocity component changes. In this map, if a y position is picked and the $\dot{y}$ is changed, that can signify an impulsive maneuver. For example, for the L1 Lyapunov orbit to L2 Lyapunov orbit transfer, a point from the UM #1 closed curve can be picked and changing the $\dot{y}$ can bring the P3 body to the SM #1 closed curve which goes to the L2 Lyapunov orbit (shown below). Similarly, this can be applied to other parts of this map to get a similar transfer.

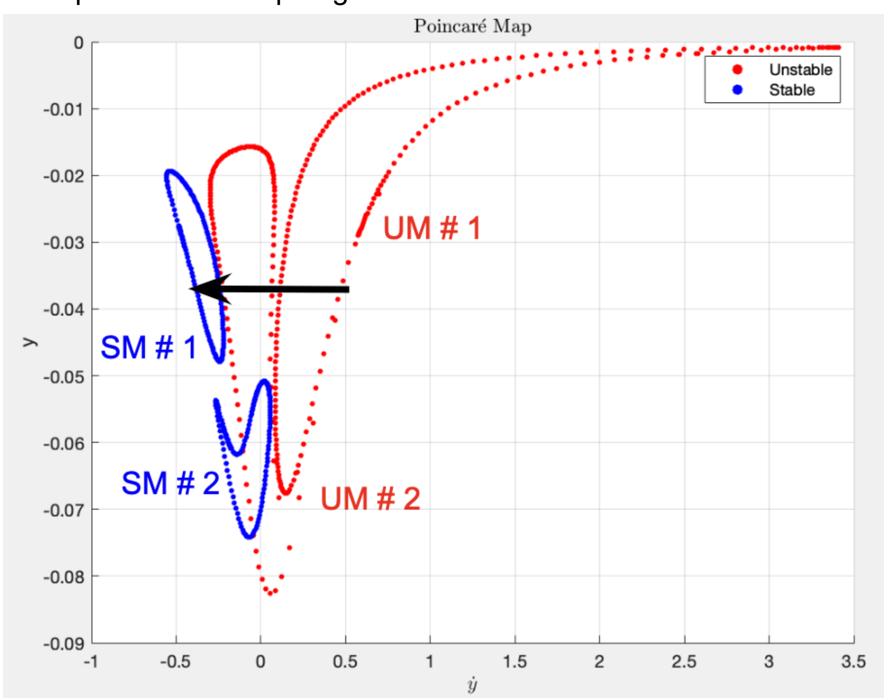


Table of Contents


```
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);
```

```matlab
% 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
```

Published with MATLAB® R2024a

Table of Contents


```
clear; clc; close all;

% ASEN 6060 - HW 5, Prob 2a
% 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;

p1_pos = [-mu, 0, 0];
p2_pos = [1-mu, 0, 0];

global count poincare_stored
```

Part a

```
TOL = 1e-12;
% Set options for ode113
options = odeset('RelTol', TOL, 'AbsTol', TOL);
```

```
% Get L2 Point
% Earth Moon system equilibrium points
[em_eq_pts, em_eq_validity] = all_eq_points(mu);

% Only looking at L2 eq point planar oscillatory modes
l1_pos = [em_eq_pts(1,:), 0];
l2_pos = [em_eq_pts(2,:), 0];

x0_1 = [0.8213849,0,0,0,0.1475143,0];
x0_2 = [1.164855,0,0,0, -0.0516671,0];
T1 = 2.763299;
T2 = 3.377214;

V0_1 = [x0_1, T1]';
V0_2 = [x0_2, T2]';

L1_periodic = gen_3d_periodic_orbit_single_shooting(V0_1, mu, false);
[L1_tout, L1_xout] = ode113(@(t, state)CR3BP_full(state, mu), [0,
L1_periodic(end)], [L1_periodic(1:6); reshape(eye(6), [36,1])], options);

L2_periodic = gen_3d_periodic_orbit_single_shooting(V0_2, mu, false);
[L2_tout, L2_xout] = ode113(@(t, state)CR3BP_full(state, mu), [0,
L2_periodic(end)], [L2_periodic(1:6); reshape(eye(6), [36,1])], options);
```

Part b, c

```
n_crossings = 2;

part_b(L1_tout, L1_xout, mu, l1_pos, 10, n_crossings);
poincare_unstable = poincare_stored;

part_c(L2_tout, L2_xout, mu, l2_pos, 6, n_crossings);

title("Moon-Bound Stable/Unstable Manifolds associated with L1, L2 Lyapunov
Orbits")

figure(2)
scatter(poincare_unstable(:,2), poincare_unstable(:,1), 10, 'filled', 'red');
hold on
scatter(poincare_stored(:,2), poincare_stored(:,1), 10, 'filled', 'blue');
xlabel("$\dot{y}$", 'Interpreter','latex')
ylabel("y")
title("Poincar\'e Map", 'Interpreter','latex')
grid on
legend("Unstable", "Stable")
```

Functions

```
function part_b(tout, xout, mu, l1_pos, manifold_time, n_crossings)
    % Set options for ode113()
    % Part b
    options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
eventFn(t, y, mu));
```

```
a = 384400; % [kg] EM average SMA
d = 50 / a; % [-] Unitless, normalized by a

period = tout(end);

p1_pos = [-mu, 0, 0];
p2_pos = [1-mu, 0, 0];

figure()
plot(xout(:,1), xout(:,2), 'black', 'LineWidth', 3)
hold on
scatter(l1_pos(1), l1_pos(2), 'filled', 'red')
scatter(p1_pos(1), p1_pos(2), 'filled', 'blue')
scatter(p2_pos(1), p2_pos(2), 'filled', ' black')

% Compute STM - phi(t1+T, t1)
phi_t1T_t1 = reshape(xout(end,7:42), [6,6])';

moon_unstable_cnt = 0;

% Begin for loop
for i = 1:length(tout)

    % Compute STM - phi(tj+T, tj)
    phi_tj_t1 = reshape(xout(i, 7:42), [6,6])';
    phi_tjT_tj = phi_tj_t1 * phi_t1T_t1 * inv(phi_tj_t1);

    % Get evals, evecs
    [V, D] = eig(phi_tjT_tj);

    % Get evals as an array
    for j = 1:6
        evals(j) = D(j,j);
    end

    % Subtract evals by 1 and get 2 minimum indices. These are trivial
    % indices
    evals_minus_1 = evals - 1;
    [min_evals, trivial_index] = mink(abs(evals_minus_1), 2);

    % If eval is real and not trivial, assign stable and unstable
    % indices
    for j = 1:2
        if (isreal(evals(j)) && isnotin(trivial_index, j))
            if evals(j) < 1
                stable_index = j;
            elseif evals(j) > 1
                unstable_index = j;
            end
        end
    end

    % Get unstable evec and normalize eigenvector by 1st 3 terms
```

```
        unstable_eval = D(unstable_index, unstable_index);
        unstable_evec = V(:, unstable_index);
        unstable_pos_norm = norm(unstable_evec(1:3));
        unstable_evec = unstable_evec/unstable_pos_norm;

        % ONLY FOR L1
        % If x-velocity is positive, moon-bound
        % If x-velocity if negative, earth-bound
        x_manifold_u_p = xout(i,1:6)' + d * unstable_evec;
        x_manifold_u_n = xout(i,1:6)' - d * unstable_evec;
        if (x_manifold_u_p(4) > 0)
            moon_unstable = x_manifold_u_p;
            earth_unstable = x_manifold_u_n;
        else
            moon_unstable = x_manifold_u_n;
            earth_unstable = x_manifold_u_p;
        end

        % Propagate using the event functions
        [moon_unstable_t, moon_unstable_x] = ode113(@(t, state)CR3BP(state,
mu), [0, manifold_time], moon_unstable, options);
        [earth_unstable_t, earth_unstable_x] = ode113(@(t, state)CR3BP(state,
mu), [0, manifold_time], earth_unstable, options);

        % plot(moon_unstable_x(:,1), moon_unstable_x(:,2), 'red')
        % plot(earth_unstable_x(:,1), earth_unstable_x(:,2), 'red')

        if abs(moon_unstable_x(end,1) - (1-mu)) < 1e-6
            moon_unstable_cnt = moon_unstable_cnt + 1;
            moon_bound_unstable(:,moon_unstable_cnt) = moon_unstable;
        elseif abs(earth_unstable_x(end,1) - (1-mu)) < 1e-6
            moon_unstable_cnt = moon_unstable_cnt + 1;
            moon_bound_unstable(:,moon_unstable_cnt) = earth_unstable;
        end

    end

    global count;
    global poincare_stored;
    poincare_stored = [];
    for k = 1:moon_unstable_cnt
        count = 0;
        options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
b_eventFn(t, y, mu, n_crossings));
        [moon_unstable_t, moon_unstable_x] = ode113(@(t, state)CR3BP(state,
mu), [0, manifold_time], moon_bound_unstable(:,k), options);
        plot(moon_unstable_x(:,1), moon_unstable_x(:,2), 'red')
    end
    % hold off
    % legend("Lyapunov Orbit", "L1", "Earth", "Moon")
    grid on
    axis equal
    xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
    ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
```

```
end



function part_c(tout, xout, mu, l2_pos, manifold_time, n_crossings)
    % Set options for ode113()
    % Part c
    options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
eventFn(t, y, mu));

    a = 384400; % [kg] EM average SMA
    d = 50 / a; % [-] Unitless, normalized by a

    period = tout(end);

    p1_pos = [-mu, 0, 0];
    p2_pos = [1-mu, 0, 0];

    % figure()
    plot(xout(:,1), xout(:,2), 'black', 'LineWidth', 3)
    % hold on
    scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
    scatter(p1_pos(1), p1_pos(2), 'filled', 'blue')
    scatter(p2_pos(1), p2_pos(2), 'filled', ' black')

    % Compute STM - phi(t1+T, t1)
    phi_t1T_t1 = reshape(xout(end,7:42), [6,6])';

    moon_stable_cnt = 0;

    % Begin for loop
    for i = 1:length(tout)

        % Compute STM - phi(tj+T, tj)
        phi_tj_t1 = reshape(xout(i, 7:42), [6,6])';
        phi_tjT_tj = phi_tj_t1 * phi_t1T_t1 * inv(phi_tj_t1);

        % Get evals, evecs
        [V, D] = eig(phi_tjT_tj);

        % Get evals as an array
        for j = 1:6
            evals(j) = D(j,j);
        end

        % Subtract evals by 1 and get 2 minimum indices. These are trivial
        % indices
        evals_minus_1 = evals - 1;
        [min_evals, trivial_index] = mink(abs(evals_minus_1), 2);

        % If eval is real and not trivial, assign stable and unstable
        % indices
        for j = 1:6
            if (isreal(evals(j)) && isnotin(trivial_index, j))
```

```
                if evals(j) < 1
                    stable_index = j;
                elseif evals(j) > 1
                    unstable_index = j;
                end
            end
        end

        % Get stable/unstable evec and normalize eigenvector by 1st 3 terms
        stable_eval = D(stable_index, stable_index);
        stable_evec = V(:, stable_index);
        stable_pos_norm = norm(stable_evec(1:3));
        stable_evec = stable_evec/stable_pos_norm;
        % stable_evec(4:6) = -stable_evec(4:6);

        % Step into manifold
        x_manifold_s_p = xout(i,1:6)' + d * stable_evec;
        x_manifold_s_n = xout(i,1:6)' - d * stable_evec;

        % If x-velocity is positive, moon-bound
        % If x-velocity if negative, earth-bound
        if (x_manifold_s_p(4) > 0)
            moon_stable = x_manifold_s_p;
            earth_stable = x_manifold_s_n;
        else
            moon_stable = x_manifold_s_n;
            earth_stable = x_manifold_s_p;
        end

        % Propagate using the event functions
        [moon_stable_t, moon_stable_x] = ode113(@(t, state)CR3BP(state, mu),
[0, -manifold_time], moon_stable, options);
        [earth_stable_t, earth_stable_x] = ode113(@(t, state)CR3BP(state,
mu), [0, -manifold_time], earth_stable, options);

        % plot(moon_stable_x(:,1), moon_stable_x(:,2), 'blue')
        % plot(earth_stable_x(:,1), earth_stable_x(:,2), 'red')

        if (abs(moon_stable_x(end,1) - (1-mu)) < 1e-6 && moon_stable_x(end,2)
< 0)
            moon_stable_cnt = moon_stable_cnt + 1;
            moon_bound_stable(:,moon_stable_cnt) = moon_stable;
        else
            moon_stable_cnt = moon_stable_cnt + 1;
            moon_bound_stable(:,moon_stable_cnt) = earth_stable;
        end
    end

    global count;
    global poincare_stored;
    poincare_stored = [];
    for k = 1:moon_stable_cnt
        count = 0;
        options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
```

```
c_eventFn(t, y, mu, n_crossings));
        [moon_stable_t, moon_stable_x] = ode113(@(t, state)CR3BP(state, mu),
[0, -manifold_time], moon_bound_stable(:,k), options);
        plot(moon_stable_x(:,1), moon_stable_x(:,2), 'blue')
    end
    hold off
    % legend("Lyapunov Orbit", "L1", "Earth", "Moon")
    grid on
    axis equal
    xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
    ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
end

function [value,isterminal,direction] = c_eventFn(t,y,mu,n_crossings)
    global count;
    global poincare_stored;
    if count < n_crossings
        value = y(1) - (1-mu);
        isterminal = 0;
        direction = -1;
        if (abs(value) < 1e-12 && y(4) > 0)
            count = count + 1;
            poincare_stored = [poincare_stored; y(2), y(5)];

        end
    elseif count == n_crossings
        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
        if (abs(value) < 1e-12 && y(4) > 0)
            poincare_stored = [poincare_stored; y(2), y(5)];

        end
    end
end


function [value,isterminal,direction] = b_eventFn(t,y,mu,n_crossings)
    global count;
    global poincare_stored;
    if count < n_crossings
        value = y(1) - (1-mu);
        isterminal = 0;
        direction = 1;
        if (abs(value) < 1e-12 && y(4) > 0)
            count = count + 1;
            poincare_stored = [poincare_stored; y(2), y(5)];

        end
    elseif count == n_crossings
        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
        if (abs(value) < 1e-12 && y(4) > 0)
```

```
            poincare_stored = [poincare_stored; y(2), y(5)];

        end
    end
end

function [value,isterminal,direction] = eventFn(t,y, mu)
    value = [1-mu-y(1), y(1)-(-mu)];
    isterminal = [1, 1]; % Halt integration when value is 0
    direction = [0, 0]; % When zero is approached from either side
end



function out = isnotin(array, val)
    out = true;
    for el = 1:length(array)
        if val == array(el)
            out = false;
        end
    end
end
```

Published with MATLAB® R2024a