

ASEN 6060 - HW 2  
Spring 2025  
Jash Bhalavat

**Problem 1 - Part a**

The script starts off by calculating the collinear equilibrium points. To calculate the position of the equilibrium points, roots to this equation are computed:

$$x - \frac{(1 - \mu)(x + \mu)}{(|x + \mu|)^3} - \frac{\mu(x - 1 + \mu)}{(|x - 1 + \mu|)^3} = 0$$

But, initial guesses are necessary to find all three roots. For L1, initial guess is the midpoint of P1 and P2. Then, that initial guess is fed into Matlab's `fsolve()` function and the function tolerances are set to 1e-17 i.e. the optimality tolerance, step tolerance, and function tolerances are set to 1e-17. In Matlab function tolerance is a lower bound in the change in the step of the function, step tolerance is the lower bound on the step size of the free variable (x in this case), and optimality tolerance is the lower bound on how close x is to the first order optimal. Combined, these three tolerances assure that the root is accurate to at least the 15th decimal point.

Then, the initial guess for L2 is set to the distance between P2 and L1 and that value is added to P2's position. This value is greater than P2's x position and therefore, finds the L2 equilibrium point. Then, L2 is calculated. Lastly, for the L3 initial guess, the same distance value is used ( $x_{P2} - x_{L1}$ ), but this time, that distance is subtracted from the position of P1. Therefore, this initial guess is less than P1's x position and finds the L1 equilibrium point. Now, all collinear equilibrium positions have been found. After all the collinear equilibrium points have been identified, they are plugged back into the equation (shown above) to confirm if the result is sufficiently close to 0. Since, we're asked to report values up to 15 decimal places, a tolerance of 1e-15 is used to ensure that the equation is sufficiently close to 0. If it is, the validity flag associated with the points are set to valid, and invalid otherwise.

For L4, L5, the x position is the same -  $\frac{1}{2} - \mu$ . The y position for L4 is  $\sqrt{3}/2$  and the y position for L5 is  $-\sqrt{3}/2$ . The script is a function called `all_eq_points()` and is shown below (along with its helper functions):

```

1 function [x_Ls, validity] = all_eq_points(mu)
2     % Function that finds all equilibrium point positions
3
4     % Initial guess for L1 is midpoint of P1 and P2
5     x_L1_0 = (-mu + (1 - mu))/2;
6
7     % Call function to get position of L1 and validity of L1
8     [x_L1, x_L1_validity] = find_coll_eq_pts(x_L1_0, mu);
9
10    % Initial guess for L2 is distance between P2 and L1. That is added to P2
11    % position.
12    x_L2_0 = (1-mu)-x_L1 + (1-mu);
13
14    % Call function to get position of L2 and validity of L2
15    [x_L2, x_L2_validity] = find_coll_eq_pts(x_L2_0, mu);
16
17    % Initial guess for L3 is distance between P2 and L1. That is subtracted
18    % from P1 position.
19    x_L3_0 = -mu - ((1-mu)-x_L1);|
20
21    % Call function to get position of L2 and validity of L2
22    [x_L3, x_L3_validity] = find_coll_eq_pts(x_L3_0, mu);
23
24    % Find L4 and L5 points
25    x_L4 = 1/2 - mu;
26    y_L4 = sqrt(3)/2;
27    y_L5 = -sqrt(3)/2;
28
29    x_Ls = [x_L1, 0; x_L2, 0; x_L3, 0; x_L4, y_L4; x_L4, y_L5];
30    validity = [x_L1_validity, x_L2_validity, x_L3_validity, true, true];
31 end

```

```

1 function [x_Lx, validity] = find_coll_eq_pts(x0, mu)
2     % Function to find collinear equilibrium points
3
4     % Function for collinear equilibrium points
5     fun = @(x)collinear_eq_pts(x, mu);
6
7     % Set display to zero and thresholds to 1e-17 to get as close to double
8     % precision as possible
9     options = optimoptions('fsolve', 'Display','none', 'FunctionTolerance',1e-17, ...
10        'StepTolerance',1e-17, 'OptimalityTolerance',1e-17);|
11
12    % Use fsolve to find root of function
13    x_Lx = fsolve(fun, x0, options);
14
15    % Test if the root is sufficiently close to 0. If so, validity is true,
16    % else validity is false
17    test_Lx = collinear_eq_pts(x_Lx, mu);
18    validity = false;
19    if abs(test_Lx) < 1e-15
20        validity = true;
21    end
22 end

```

```

1 function fx = collinear_eq_pts(x, mu)
2     % Function to get roots for collinear equilibrium points
3     fx = x - ((1-mu)*(x+mu))/(abs(x+mu)^3) - (mu*(x-1+mu))/(abs(x-1+mu)^3);
4 end

```

## Part b

Since the equilibrium points are in the xy plane,  $z = 0$ . The jacobi constants are calculated by evaluating  $U^*$  at each equilibrium point and using this equation:

$$C = 2U^*$$

All these values presented are unitless.

Equilibrium Point	C	x	y
L1	<b>3.188341106545981</b>	<b>0.836915131750372</b>	<b>0</b>
L2	<b>3.172160451379589</b>	<b>1.155682160772215</b>	<b>0</b>
L3	<b>3.012147149466313</b>	<b>-1.005062645304093</b>	<b>0</b>
L4	<b>2.987997052306423</b>	<b>0.487849415605290</b>	<b>0.866025403784439</b>
L5	<b>2.987997052306423</b>	<b>0.487849415605290</b>	<b>-0.866025403784439</b>

Table from HW 1		
Equilibrium Point	C	Position (x, y, z)
L1	3.19	(0.84, 0, 0)
L2	3.17	(1.16, 0, 0)
L3	3.01	(-1.05, 0, 0)
L4	2.99	(0.48, 0.87, 0)
L5	2.99	(0.48, -0.87, 0)

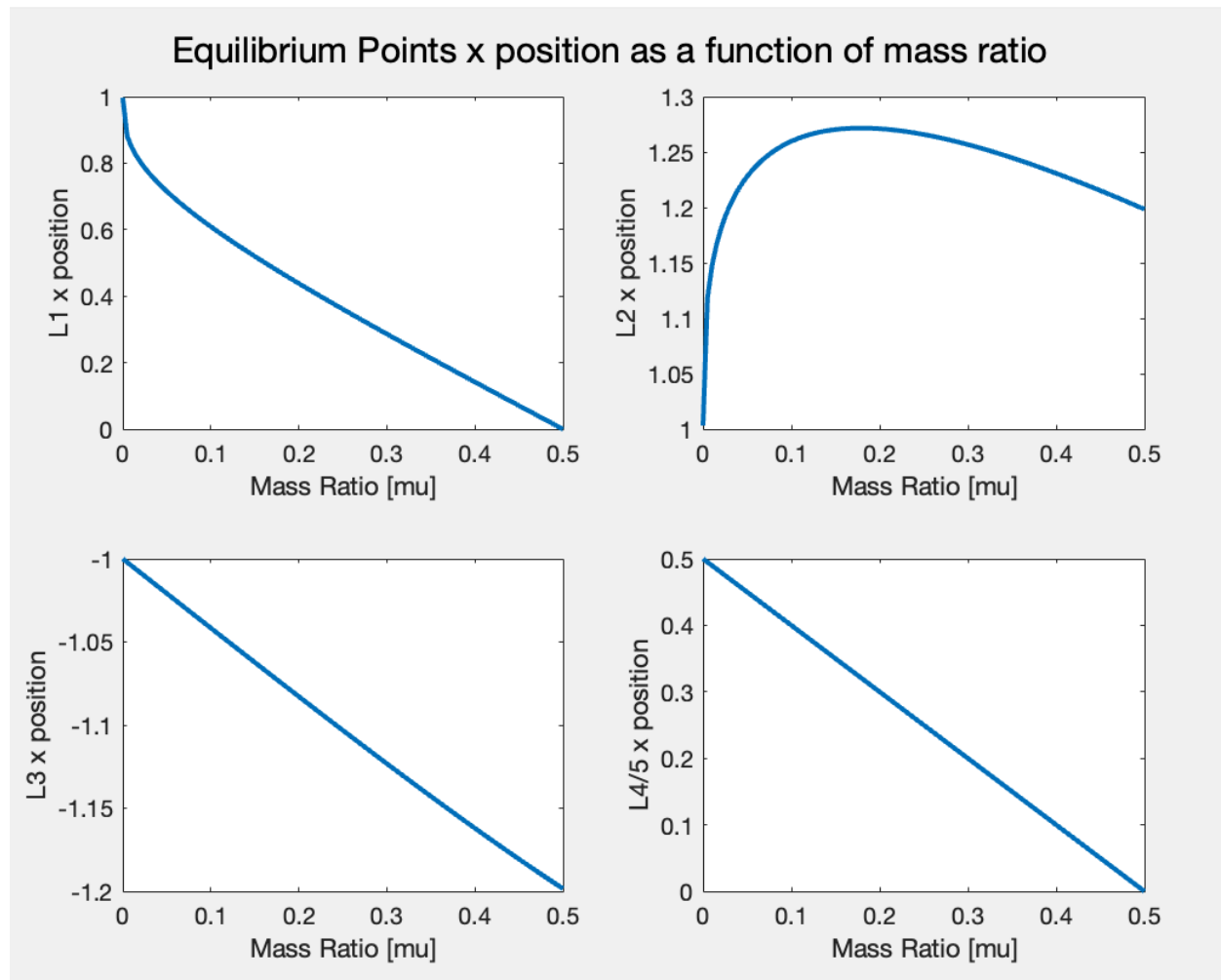
The estimates in HW 1 were fairly accurate when compared to the roots calculated.

### **Part c**

Firstly, all equilibrium points' y positions are constant. The collinear equilibrium points always have 0 y position and the triangular equilibrium points have  $\sqrt{3}/2$  and  $-\sqrt{3}/2$  y positions. So, these are not dependent on the mass ratio. So, only the x positions of the equilibrium points are dependent on the mass ratio. We'll examine those further.

Firstly, let's examine the dependence on the mass ratio based on intuition and first principles. For the x positions of the equilibrium points, it can be logically deduced that the collinear equilibrium points will move closer to the P1 body as the mass ratio is increased. Because, the mass ratio represents the mass of P2 divided by the sum of P1 and P2. This would mean that the increasing mass ratio would increase the mass of the P2 body or decrease the mass of the P1 body which would push the equilibrium points closer to P1 (because as the mass ratio increases, P1 and P2 have similar masses) and when both the bodies have the same mass ( $\mu = 0.5$ ), L1, L4/L5 x positions would be at the origin. This would also mean that the system is symmetric about the y axis which would lead to L2 and L3 being at the same distance from the origin. L4

Matlab was used to numerically calculate the equilibrium points at various mass ratios going from  $1e-7$  (really small number) to 0.5. 0.5 was picked because that's one of the assumptions of the CR3BP. Also, at 0.5, P1 and P2 have the same mass and increasing further would just flip the system (along the y axis). Here is a plot of the x positions as generated using Matlab:



This plot roughly matches with the expected positions. However, L2's x position seems like it increases until a certain mass ratio and then jumps back down. However, at 0.5 mass ratio, it's the same distance from the origin ( $\sim 1.2$ ) as L3 as expected. L4 (and similarly L5) x position linearly decreases to 0 as their x position follows this equation  $-\frac{1}{2} - \mu$ . And, lastly, L1 also slowly decreases to 0 which makes sense as when the mass ratio is  $\frac{1}{2}$ , it means that both the bodies (P1, P2) have the same mass, and the dynamic system is symmetric around the y axis, thus L1 is at the origin.

## Problem 2

Using the position values from Problem 1, in-plane and out-of-plane eigenvalues can be calculated using the following equations:

$$\lambda = \pm i \sqrt{|U_{zz}^*|_{eq}}$$

$$\lambda = \frac{-4 + U_{xx}^* + U_{yy}^*}{2} \pm \frac{\sqrt{(4 - U_{xx}^* - U_{yy}^*)^2 - 4(U_{xx}^* U_{yy}^* - (U_{xy}^*)^2)}}{2}$$

Where the equation on the left can be used to find the out-of-plane modes and the equation on the right can be used to find the in-plane modes.

Firstly, partial derivatives of the pseudo-potential function are presented (truncated to 4 decimal points for ease of viewing):

### Earth-Moon System

Partial Derivatives of Pseudo-Potential function	
Equilibrium Point	Partial Derivatives
L1	<ul style="list-style-type: none"> <li>• <math>U_{xx}^* = 11.2959</math></li> <li>• <math>U_{yy}^* = -4.1476</math></li> <li>• <math>U_{xy}^* = 0</math></li> <li>• <math>U_{zz}^* = -5.1476</math></li> </ul>
L2	<ul style="list-style-type: none"> <li>• <math>U_{xx}^* = 7.3809</math></li> <li>• <math>U_{yy}^* = -2.1904</math></li> <li>• <math>U_{xy}^* = 0</math></li> <li>• <math>U_{zz}^* = -3.1904</math></li> </ul>
L3	<ul style="list-style-type: none"> <li>• <math>U_{xx}^* = 3.0214</math></li> <li>• <math>U_{yy}^* = -0.0107</math></li> <li>• <math>U_{xy}^* = 0</math></li> <li>• <math>U_{zz}^* = -1.0107</math></li> </ul>
L4	<ul style="list-style-type: none"> <li>• <math>U_{xx}^* = 0.75</math></li> <li>• <math>U_{yy}^* = 2.25</math></li> <li>• <math>U_{xy}^* = 1.2675</math></li> <li>• <math>U_{zz}^* = -1</math></li> </ul>
L5	<ul style="list-style-type: none"> <li>• <math>U_{xx}^* = 0.75</math></li> <li>• <math>U_{yy}^* = 2.25</math></li> <li>• <math>U_{xy}^* = -1.2675</math></li> <li>• <math>U_{zz}^* = -1</math></li> </ul>

Then, the eigenvalues are calculated by plugging into the characteristic equation:

Here are the eigenvalues for the **Earth-Moon system**:

In Plane Modes	
Equilibrium Point	Eigenvalues
L1	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 2.932055918598627 + 0i</math></li> <li>• <math>\lambda_2 = -2.932055918598627 + 0i</math></li> <li>• <math>\lambda_3 = 0.0 + 2.334385875607026i</math></li> <li>• <math>\lambda_4 = 0.0 - 2.334385875607026i</math></li> </ul>
L2	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 2.158674331407204 + 0i</math></li> <li>• <math>\lambda_2 = -2.158674331407204 + 0i</math></li> <li>• <math>\lambda_3 = 0.0 + 1.862645868650095i</math></li> <li>• <math>\lambda_4 = 0.0 - 1.862645868650095i</math></li> </ul>
L3	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 0.177875350155126 + 0i</math></li> <li>• <math>\lambda_2 = -0.177875350155126 + 0i</math></li> <li>• <math>\lambda_3 = 0.0 + 1.010419894325289i</math></li> <li>• <math>\lambda_4 = 0.0 - 1.010419894325289i</math></li> </ul>
L4	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 0.0 + 0.298208156738240i</math></li> <li>• <math>\lambda_2 = 0.0 - 0.298208156738240i</math></li> <li>• <math>\lambda_3 = 0.0 + 0.954500861840774i</math></li> <li>• <math>\lambda_4 = 0.0 - 0.954500861840774i</math></li> </ul>
L5	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 0.0 + 0.298208156738240i</math></li> <li>• <math>\lambda_2 = 0.0 - 0.298208156738240i</math></li> <li>• <math>\lambda_3 = 0.0 + 0.954500861840774i</math></li> <li>• <math>\lambda_4 = 0.0 - 0.954500861840774i</math></li> </ul>

Out of Plane Modes	
Equilibrium Point	Eigenvalues
L1	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 2.268831085285033i,</math></li> </ul>

	<ul style="list-style-type: none"> <li>• <math>\lambda_6 = 0.0 - 2.268831085285033i</math></li> </ul>
L2	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.786176149509637i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.786176149509637i</math></li> </ul>
L3	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.005331426617353i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.005331426617353i</math></li> </ul>
L4	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.0i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.0i</math></li> </ul>
L5	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.0i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.0i</math></li> </ul>

### Sun-Earth System

Firstly, the mass ratio for the Sun-Earth system is  $3.003480594542192e-06$ . Using this, using the same function presented in Problem 1, the position of the equilibrium points are computed:

The z position is 0 since the equilibrium points lie in the xy plane.

Equilibrium Point	x	y
L1	0.990026593870750	0
L2	1.010034116422210	0
L3	-1.000001251450248	0
L4	0.499996996519405	0.866025403784439
L5	0.499996996519405	-0.866025403784439

Then, again, the partial derivatives of the pseudo-potential function are calculated:

Partial Derivatives of Pseudo-Potential function	
Equilibrium Point	Partial Derivatives
L1	<ul style="list-style-type: none"> <li>• <math>U^*_{xx} = 9.1216</math></li> <li>• <math>U^*_{yy} = -3.0608</math></li> <li>• <math>U^*_{xy} = 0</math></li> </ul>



	<ul style="list-style-type: none"> <li>• <math>U^*_{zz} = -4.0608</math></li> </ul>
L2	<ul style="list-style-type: none"> <li>• <math>U^*_{xx} = 8.8815</math></li> <li>• <math>U^*_{yy} = -2.9408</math></li> <li>• <math>U^*_{xy} = 0</math></li> <li>• <math>U^*_{zz} = -3.9408</math></li> </ul>
L3	<ul style="list-style-type: none"> <li>• <math>U^*_{xx} = 3</math></li> <li>• <math>U^*_{yy} = -2.6290e-6</math></li> <li>• <math>U^*_{xy} = 0</math></li> <li>• <math>U^*_{zz} = -1</math></li> </ul>
L4	<ul style="list-style-type: none"> <li>• <math>U^*_{xx} = 0.75</math></li> <li>• <math>U^*_{yy} = 2.25</math></li> <li>• <math>U^*_{xy} = 1.2990</math></li> <li>• <math>U^*_{zz} = -1</math></li> </ul>
L5	<ul style="list-style-type: none"> <li>• <math>U^*_{xx} = 0.75</math></li> <li>• <math>U^*_{yy} = 2.25</math></li> <li>• <math>U^*_{xy} = -1.2990</math></li> <li>• <math>U^*_{zz} = -1</math></li> </ul>

Then, solving the same characteristic equation  
Here are the eigenvalues for the **Sun-Earth system**:

In Plane Modes	
Equilibrium Point	Eigenvalues
L1	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 2.532559250170969 + 0i</math></li> <li>• <math>\lambda_2 = -2.532559250170969 + 0i</math></li> <li>• <math>\lambda_3 = 0.0 + 2.086392572376518i</math></li> <li>• <math>\lambda_4 = 0.0 - 2.086392572376518i</math></li> </ul>
L2	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 2.484413408021258 + 0i</math></li> <li>• <math>\lambda_2 = -2.484413408021258 + 0i</math></li> <li>• <math>\lambda_3 = 0.0 + 2.057072933441825i</math></li> <li>• <math>\lambda_4 = 0.0 - 2.057072933441825i</math></li> </ul>
L3	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 0.002807867480545 + 0i</math></li> <li>• <math>\lambda_2 = -0.002807867480545 + 0i</math></li> <li>• <math>\lambda_3 = 0.0 + 1.000002628031872i</math></li> <li>• <math>\lambda_4 = 0.0 - 1.000002628031872i</math></li> </ul>

<b>L4</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 0.0 + 0.004502648570630i</math></li> <li>• <math>\lambda_2 = 0.0 - 0.004502648570630i</math></li> <li>• <math>\lambda_3 = 0.0 + 0.999989863026545i</math></li> <li>• <math>\lambda_4 = 0.0 - 0.999989863026545i</math></li> </ul>
<b>L5</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_1 = 0.0 + 0.004502648570630i</math></li> <li>• <math>\lambda_2 = 0.0 - 0.004502648570630i</math></li> <li>• <math>\lambda_3 = 0.0 + 0.999989863026545i</math></li> <li>• <math>\lambda_4 = 0.0 - 0.999989863026545i</math></li> </ul>

Out of Plane Modes	
Equilibrium Point	Eigenvalues
<b>L1</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 2.015148230170336i</math></li> <li>• <math>\lambda_6 = 0.0 - 2.015148230170336i</math></li> </ul>
<b>L2</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.985134989983513i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.985134989983513i</math></li> </ul>
<b>L3</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.000001314023706i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.000001314023706i</math></li> </ul>
<b>L4</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.0i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.0i</math></li> </ul>
<b>L5</b>	<ul style="list-style-type: none"> <li>• <math>\lambda_5 = 0.0 + 1.0i</math></li> <li>• <math>\lambda_6 = 0.0 - 1.0i</math></li> </ul>

### Problem 3

HW #2

Part 3 →

To find: critical value of charge in stability @  $L_4, L_5$

Assumption: CR3BP

soln:  $L_4$  pos →  $x = \frac{1}{2} - \mu, y = \frac{\sqrt{3}}{2}$

$$r_1 = \sqrt{(x+\mu)^2 + y^2} = \sqrt{\left(\frac{1}{2} - \mu + \mu\right)^2 + \left(\frac{\sqrt{3}}{2}\right)^2} = 1$$

$$r_2 = \sqrt{(x-1+\mu)^2 + y^2} = \sqrt{\left(\frac{1}{2} - \mu - 1 + \mu\right)^2 + \left(\frac{\sqrt{3}}{2}\right)^2} = 1$$

$$U_{xx}|_{L_4} = 1 - \frac{1-\mu}{1^3} - \frac{\mu}{1^3} + \frac{3(1-\mu)(1/2)^2}{1^5} + \frac{3\mu(-1/2)^2}{1^5} = 1 - \mu + \mu - \mu + \frac{3}{4} - \frac{3\mu}{4} + \frac{3\mu}{4} = 3/4$$

$$U_{yy}|_{L_4} = 1 - \frac{1-\mu}{1^3} - \frac{\mu}{1^3} + \frac{3(1-\mu)(3/4)}{1^5} + \frac{3\mu(3/4)}{1^5} = 1 - \mu + \mu - \mu + \frac{9}{4} - \frac{9\mu}{4} + \frac{9\mu}{4} = 9/4$$

$$U_{xy}|_{L_4} = 3(1-\mu)\left(\frac{1}{2} - \mu + \mu\right)\frac{\sqrt{3}}{2} + 3\mu\left(\frac{1}{2} - \mu - 1 + \mu\right)\frac{\sqrt{3}}{2} = \frac{3\sqrt{3}}{4}(1-\mu) - \frac{3\sqrt{3}}{4}\mu = \frac{3\sqrt{3}}{4}(1-2\mu)$$

$$U_{zz}|_{L_4} = -1 + \mu + \mu = -1$$

out of plane modes →

$$\lambda = \pm i \sqrt{|U_{zz}|_{L_4}} = \pm i$$

in plane modes →

$$\Lambda = \frac{-4 + U_{xx}^* + U_{yy}^* \pm \sqrt{(4 - U_{xx}^* - U_{yy}^*)^2 - 4(U_{xx}^* U_{yy}^* - U_{xy}^{*2})}}{2}$$

$$= \frac{-4 + 3/4 + 9/4 \pm \sqrt{(4 - 3/4 - 9/4)^2 - 4(3/4 \cdot 9/4 - (3\sqrt{3}/4(1-2\mu))^2)}}{2} = \frac{-1 \pm \sqrt{1 - 4(27/16 - 27/16(1-4\mu + 4\mu^2))}}{2}$$

$$= -\frac{1}{2} \pm \frac{\sqrt{1 - 27\mu + 27\mu^2}}{2}$$

When  $\Lambda$  is -ve,  $\lambda$  takes the form →  $\pm bi$ . But when  $\Lambda$  is  $a \pm bi$ ,  $\lambda$  takes the form  $a \pm bi$ . When  $\lambda$  is  $\pm bi$ , the eq. point is oscillatory. When  $\lambda$  is  $a \pm bi$ , the eq. point takes the mode of spiral in/out. So, finding when  $\mu$  goes from  $\pm bi$  to  $a \pm bi$  will lead to finding critical mass ratio.

$$\text{Find: } \sqrt{1 - 27\mu(1-\mu)} = 0 \rightarrow 27\mu^2 - 27\mu + 1 = 0, \mu = \frac{+27 \pm \sqrt{27^2 - 4(27)}}{2}$$

$$\text{Picking root } < 0.5 \rightarrow \mu = 0.038520896504551 = \mu_{crit, L_4}$$

@  $\mu_{crit, L_4}$ , stability @  $L_4$  point switches from oscillatory to spiral in/out

$L_5$  →

$$x = \frac{1}{2} - \mu, y = -\frac{\sqrt{3}}{2}, r_1 = 1, r_2 = 1, U_{xx}|_{L_5} = 3/4, U_{yy}|_{L_5} = -1, U_{xy}|_{L_5} = 9/4$$

$$U_{xx}|_{L_5} = \frac{3(1-\mu)\left(\frac{1}{2} - \mu + \mu\right)\left(-\frac{\sqrt{3}}{2}\right)}{1^5} + \frac{3\mu\left(\frac{1}{2} - \mu - 1 + \mu\right)\left(-\frac{\sqrt{3}}{2}\right)}{1^5} = -\frac{3\sqrt{3}}{4}(1-\mu) + \frac{3\sqrt{3}}{4}\mu = -\frac{3\sqrt{3}}{4}(1-2\mu)$$

out of plane modes →

$$\lambda = \pm i \sqrt{|U_{zz}|_{L_5}} = \pm i$$

in plane modes →

$$\Lambda \text{ is same as } L_4: (U_{xx}^*|_{L_4})^2 = (U_{xx}^*|_{L_5})^2 \therefore \mu_{crit, L_5} = \mu_{crit, L_4}$$

$$= 0.038520896504551$$



## Problem 4

Problem 4 → Assume: CR3BP

a) 
$$\begin{bmatrix} \dot{\xi}_0 \\ \dot{\xi}_0 \\ \dot{\eta}_0 \\ \dot{\eta}_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \lambda_1 & -\lambda_1 & \lambda_3 & -\lambda_3 \\ \alpha_1 & -\alpha_1 & \alpha_3 & -\alpha_3 \\ \alpha_1 \lambda_1 & \alpha_1 \lambda_1 & \alpha_3 \lambda_3 & \alpha_3 \lambda_3 \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix}, \quad \alpha_i = \frac{\lambda_i^2 - U_{xx}^*}{2\lambda_i}$$

Using Matlab to inverse matrix to find  $A_i$

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} = \begin{bmatrix} \frac{-\lambda_3^2 + U_{xx}^*}{2(\lambda_1^2 - \lambda_3^2)} & \frac{-\lambda_1 \lambda_3^2 + \lambda_1 U_{xx}^*}{2U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{\lambda_1 \lambda_3^2}{U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{1}{\lambda_1^2 - \lambda_3^2} \\ \frac{-\lambda_3^2 + U_{xx}^*}{2(\lambda_1^2 - \lambda_3^2)} & \frac{-(-\lambda_1 \lambda_3^2 + \lambda_1 U_{xx}^*)}{2U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{-\lambda_1 \lambda_3^2}{U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{1}{\lambda_1^2 - \lambda_3^2} \\ \frac{-(-\lambda_1^2 + U_{xx}^*)}{2(\lambda_1^2 - \lambda_3^2)} & \frac{-\lambda_3 (-\lambda_1^2 + U_{xx}^*)}{2U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{-\lambda_3^2 \lambda_1}{U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{-1}{\lambda_1^2 - \lambda_3^2} \\ \frac{-(-\lambda_1^2 + U_{xx}^*)}{2(\lambda_1^2 - \lambda_3^2)} & \frac{\lambda_3 (-\lambda_1^2 + U_{xx}^*)}{2U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{\lambda_3^2 \lambda_1}{U_{xx}^* (\lambda_1^2 - \lambda_3^2)} & \frac{-1}{\lambda_1^2 - \lambda_3^2} \end{bmatrix} \begin{bmatrix} \dot{\xi}_0 \\ \dot{\xi}_0 \\ \dot{\eta}_0 \\ \dot{\eta}_0 \end{bmatrix}$$

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{\lambda_1^2 - \lambda_3^2} \left[ -\dot{\xi}_0 \alpha_3 \lambda_3 - \frac{\alpha_3 \lambda_1 \lambda_3}{U_{xx}^*} \dot{\xi}_0 + \frac{\lambda_1 \lambda_3^2}{U_{xx}^*} \dot{\eta}_0 + \dot{\eta}_0 \right] \\ \frac{1}{\lambda_1^2 - \lambda_3^2} \left[ -\dot{\xi}_0 \alpha_3 \lambda_3 + \frac{\alpha_3 \lambda_1 \lambda_3}{U_{xx}^*} \dot{\xi}_0 - \frac{\lambda_1 \lambda_3^2}{U_{xx}^*} \dot{\eta}_0 + \dot{\eta}_0 \right] \\ \frac{1}{\lambda_1^2 - \lambda_3^2} \left[ \dot{\xi}_0 \alpha_1 \lambda_1 + \frac{\alpha_1 \lambda_1 \lambda_1}{U_{xx}^*} \dot{\xi}_0 - \frac{\lambda_1^2 \lambda_3}{U_{xx}^*} \dot{\eta}_0 - \dot{\eta}_0 \right] \\ \frac{1}{\lambda_1^2 - \lambda_3^2} \left[ \dot{\xi}_0 \alpha_1 \lambda_1 - \frac{\alpha_1 \lambda_1 \lambda_3 \lambda_1}{U_{xx}^*} \dot{\xi}_0 + \frac{\lambda_1^2 \lambda_3}{U_{xx}^*} \dot{\eta}_0 - \dot{\eta}_0 \right] \end{bmatrix}$$
 where  $\alpha_i = \frac{\lambda_i^2 - U_{xx}^*}{2\lambda_i}$

b) To locate oscillatory modes,  $A_1 = A_2 = 0$

$$0 = \frac{1}{\lambda_1^2 - \lambda_3^2} \left[ -\dot{\xi}_0 \alpha_3 \lambda_3 - \frac{\alpha_3 \lambda_1 \lambda_3}{U_{xx}^*} \dot{\xi}_0 + \frac{\lambda_1 \lambda_3^2}{U_{xx}^*} \dot{\eta}_0 + \dot{\eta}_0 \right] \quad (1)$$

$$0 = \frac{1}{\lambda_1^2 - \lambda_3^2} \left[ -\dot{\xi}_0 \alpha_3 \lambda_3 + \frac{\alpha_3 \lambda_1 \lambda_3}{U_{xx}^*} \dot{\xi}_0 - \frac{\lambda_1 \lambda_3^2}{U_{xx}^*} \dot{\eta}_0 + \dot{\eta}_0 \right] \quad (2)$$

Add (1) (2) →  $0 = -2\dot{\xi}_0 \alpha_3 \lambda_3 + 2\dot{\eta}_0 \rightarrow \dot{\eta}_0 = \alpha_3 \lambda_3 \dot{\xi}_0 \rightarrow \text{Sub. into (1)}$

$$0 = -\dot{\xi}_0 \alpha_3 \lambda_3 - \frac{\alpha_3 \lambda_1 \lambda_3}{U_{xx}^*} \dot{\xi}_0 + \frac{\lambda_1 \lambda_3^2}{U_{xx}^*} \dot{\eta}_0 + \alpha_3 \lambda_3 \dot{\xi}_0 \rightarrow \dot{\xi}_0 = \lambda_3 \dot{\eta}_0 / \alpha_3$$

## Part c

Given:  $\xi_0 = -0.001$  and  $\eta_0 = 0$ . Using equations derived in part b (both values are unitless):

- $\dot{\xi}_0 = 0$
- $\dot{\eta}_0 = 0.008372273201672$
- So, initial variation is → [-0.001, 0, 0, 0.008372273201672]**

Using all the initial variations for the initial state vector for the CR3BP:

- For L1, the position vector in the rotating frame (from Problem 1) is →  $x = 0.836915131750372$ ,  $y = 0$ ,  $z = 0$ , and all velocity components are 0 (since, L1 is an equilibrium point).
- $\bar{x}_0 = [0.835915131750372, 0, 0, 0, 0.008372273201672, 0]$  (unitless)

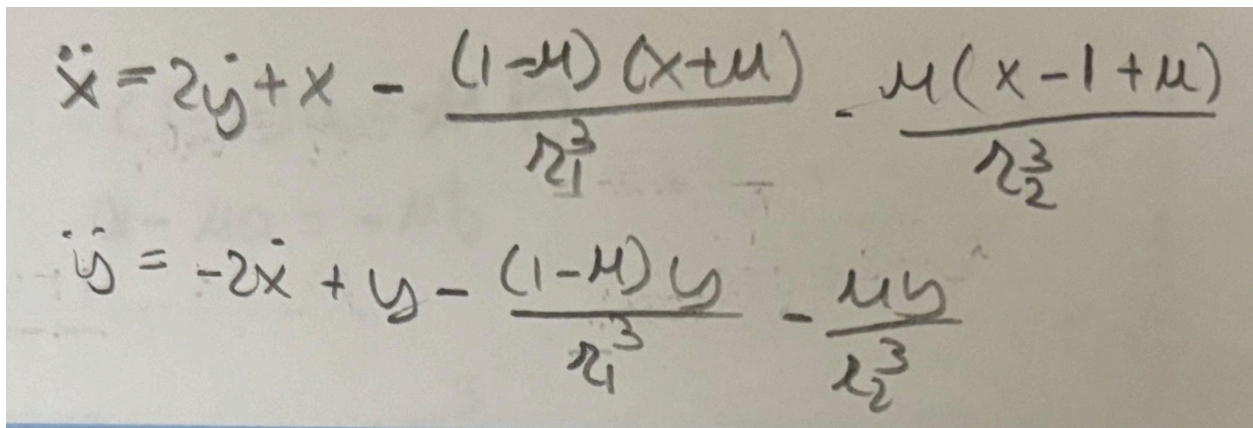
The linear propagation is carried out using the following equations:

- $\xi(t) = A_3 e^{\lambda_3 t} + A_4 e^{-\lambda_3 t}$  (since  $A_1, A_2 = 0$  to excite oscillatory modes or periodic motion)
- $\eta(t) = A_3 \alpha_3 e^{\lambda_3 t} - A_4 \alpha_3 e^{-\lambda_3 t}$  (since  $A_1, A_2 = 0$  to excite oscillatory modes or periodic motion)

Where  $\lambda_1 = 2.932055918598627 + 0i$ ,  $\lambda_3 = 0.0 + 2.334385875607026i$ , from problem 2. Using these, the  $\alpha$  values can be calculated -  $\alpha_1 = -0.460127151772006$  and  $\alpha_3 = 0.0 + 3.586499254111257i$ . Using these,  $A_3$  and  $A_4$  can be calculated:

$A_3 = -5.0 \times 10^{-4}$  and  $A_4 = -5.0 \times 10^{-4}$  using equations from part a.

For the non-linear propagation, EOMs from the CR3BP are used as shown below:



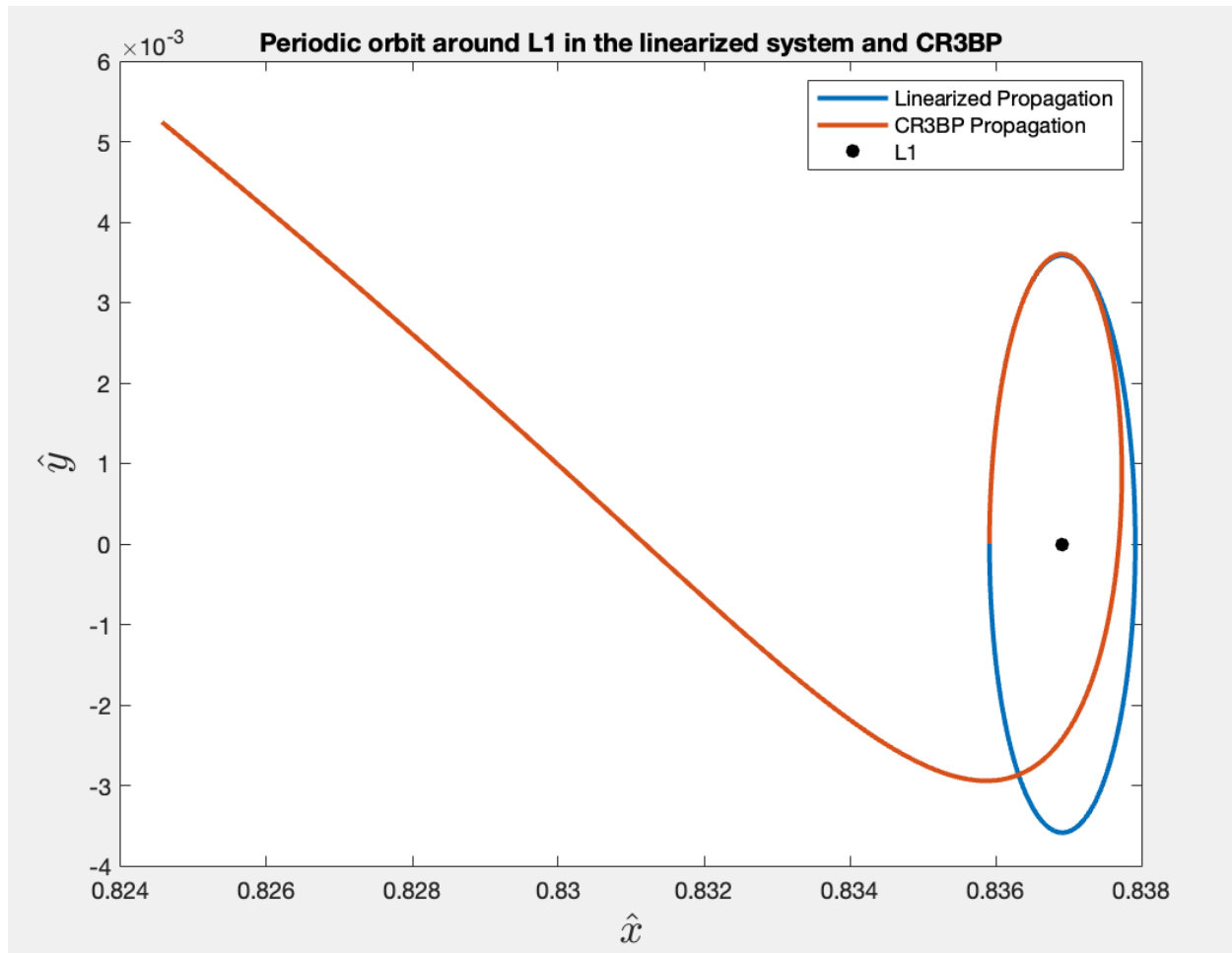
$$\ddot{x} = 2\dot{y} + x - \frac{(1-\mu)(x+\mu)}{r_1^3} - \frac{\mu(x-1+\mu)}{r_2^3}$$

$$\ddot{y} = -2\dot{x} + y - \frac{(1-\mu)y}{r_1^3} - \frac{\mu y}{r_2^3}$$

(The EOM for acceleration in  $z$  is zero as this motion is only restricted to the  $xy$  plane)

These EOMs are then numerically integrated through Matlab's ODE113() integrator to get the non-linear trajectory.

Below are the results for linear vs CR3BP propagation using the initial variation:

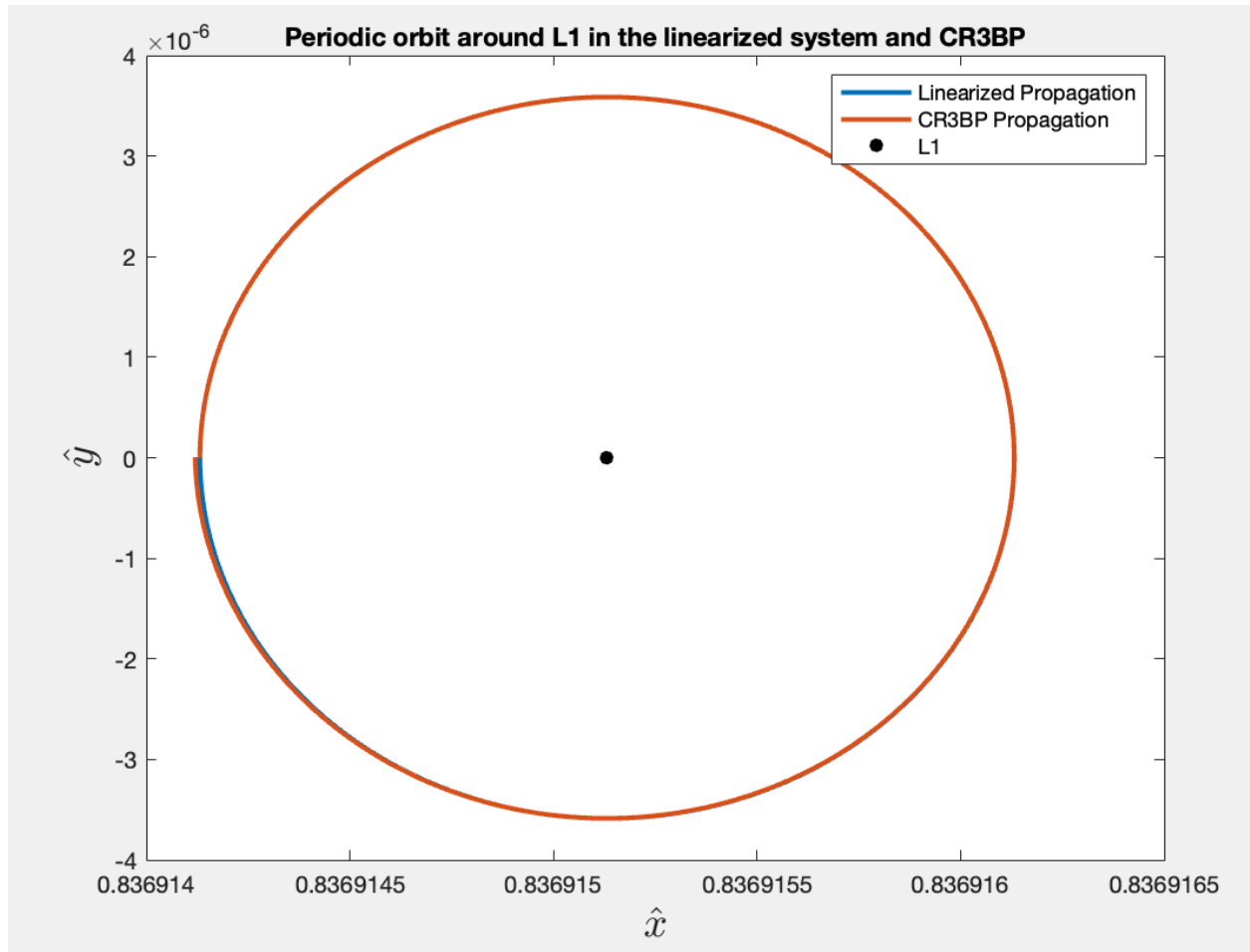


As expected, the linearized propagation is periodic and follows an elliptical path while the non-linear (CR3BP) propagation starts by following the linear path, but veers off after some time (about 1/4 of the period) i.e. the non-linear trajectory is not periodic (because it does not return back to its original position). That's the biggest difference between the two trajectories.

### Part d

No, the trajectory generated in Problem 4c clearly isn't periodic around the L1 equilibrium point. So, it won't be a good initial guess for a planar periodic orbit around L1. In order to get a better initial guess using this linear approximation, the magnitude of the linearized initial variations must be reduced. For example, when  $\xi_0$  is reduced to -0.000001, the following trajectories are generated. As seen below, the linear and non-linear systems have a much better alignment in

this case compared to 4c.



This also isn't completely periodic but it is much closer to a periodic orbit than part c.

### Part e

Eigenvectors for  $\lambda_3, \lambda_4$  can be calculated by generating the A2D matrix and using Matlab's eig() function. The A2D matrix for L1 can be generated using the following form:

$$\begin{bmatrix} \dot{\xi} \\ \dot{\eta} \\ \ddot{\xi} \\ \ddot{\eta} \end{bmatrix} = \begin{bmatrix} 0_{2 \times 2} & I_{2 \times 2} \\ U_{XX}^* & \Omega \end{bmatrix} \begin{bmatrix} \xi \\ \eta \\ \dot{\xi} \\ \dot{\eta} \end{bmatrix} \longrightarrow \delta \dot{\bar{x}}_{2D} = A_{2D} \delta \bar{x}_{2D}$$

$$A_{2D} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 11.295188987111322 & 0 & 0 & 2 \\ 0 & -4.147594493555661 & 2 & 0 \end{bmatrix}$$

$A_{2D} =$

Using Matlab to find the eigenvectors:

- Eigenvector for  $\lambda_3$ :
  - $[-0.105758072598292 + 0i, 0.0 - 0.379301248490016i, 0.0 - 0.246880150904874i, 0.885435477075205 + 0i]$
  - Real components
    - $[-0.105758072598292, 0.0, 0.0, 0.885435477075205]$
  - Imaginary components (i is assumed)
    - $[0.0, -0.379301248490016, -0.246880150904874, 0.0]$
- Eigenvector for  $\lambda_4$ :
  - $[-0.105758072598292 + 0i, 0.0 + 0.379301248490016i, 0.0 + 0.246880150904874i, 0.885435477075205 + 0i]$

The real and imaginary components of the complex conjugate pair:

- Real components
  - $u = [-0.105758072598292, 0.0, 0.0, 0.885435477075205]$
- Imaginary components
  - $v = [0.0, 0.379301248490016, 0.246880150904874, 0.0]$

$u$  and  $v$  form a set of real basis vectors to find initial guesses for the L1 Lyapunov orbit. A linear combination of  $u$ ,  $v$  can be used to find initial guesses for a periodic path around L1 (let's call this  $dx_0$ ):

$$dx_0 = n*u + m*v \text{ (where } m \text{ and } n \text{ are real numbers)}$$

For this problem let  $m = 0$ , and  $n = 0.004727771485579$ . So:

$$dx_0 = [-0.001, 0, 0, 0.008372273201672]$$

which is the same as the initial variation from part c.

Hence, the vector formed by linearly combining  $u$  and  $v$  can serve as an initial guess for a periodic path around L1 in this linear system.



## Problem 5

### Part a

NOTE: For this entire problem, eigenvectors (and initial variations) follow this convention -  $[\xi_0, \eta_0, \dot{\xi}_0, \dot{\eta}_0]$ .

Using Matlab's eig() function to calculate the eigenvalues and eigenvectors of the A2D matrix:

Name (eval)	Eigenvalue	Name (evec)	Eigenvector
$\lambda_1$	<b>0.0 + 0.95450086184077 4i</b>	$v_1$	<b>[-0.585677575109078 + 0i, 0.234834498587212 - 0.353696321201434i, 0.0 - 559029750202430i, 0.337603443416680 + 0.224149731291441i]</b>
$-\lambda_1 = \lambda_2$	<b>0.0 - 0.95450086184077 4i</b>	$v_2 = -v_1$	<b>[-0.585677575109078 + 0i, 0.234834498587212 + 0.353696321201434i, 0.0 + 559029750202430i, 0.337603443416680 - 0.224149731291441i]</b>
$\lambda_3$	<b>0.0 + 0.29820815673824 0i</b>	$v_3$	<b>[0.822132300708100 + 0i, -0.445515188489824 + 0.209640097523081i, 0.0000000000000001 + 245166557989131i, -0.062516387060783 - 0.132856263158440i]</b>
$-\lambda_3 = \lambda_4$	<b>0.0 - 0.29820815673824 0i</b>	$v_4 = -v_3$	<b>[0.822132300708100 + 0i, -0.445515188489824 - 0.209640097523081i, 0.0000000000000001 - 245166557989131i, -0.062516387060783 + 0.132856263158440i]</b>

The short and long periods are  $2\pi/\omega$  where  $\omega$  is the imaginary component of the eigenvalues:

- **Short period - 21.069796936154265 unitless**
- **Long period - 6.582692125664860 unitless**

### Part b

For short period initial guesses,  $\lambda_1$  shall be used because it has the higher frequency of the two eigenvalue pairs ( $0.954500861840774 > 0.298208156738240$ ). And, for the long period initial guesses  $\lambda_3$  shall be used because it has the lower frequency of the two eigenvalue pairs. For the initial variation, only the real components of the short and long eigenvectors are used i.e.  $\text{Re}(v_1)$  for short period and  $\text{Re}(v_3)$  for long period:

Period	Basis Vector (real components of eigenvector)	Basis Vector Normalized s.t. Pos components norm is 0.02 - <b>Initial Variation</b> - $[\xi_0, \eta_0, \dot{\xi}_0, \dot{\eta}_0]$
--------	---	--

Short Period Re( $v_1$ )	[-0.585677575109078, 0.234834498587212, 0.0, 0.337603443416680]	[0.018563369007067, 0.007443207044511, 0.0, 0.010700524596717]
Long Period Re( $v_3$ )	[0.822132300708100, -0.445515188489824, 0.0000000000000001, -0.062516387060783]	[0.017584105573543, -0.009528863058020, 0.0, -0.001337126335027]

### Part c

Using all the initial variations for the initial state vector for the CR3BP for the **short period**:

- For L4, position vector in rotating frame is  $\rightarrow x = 0.487849415605290$ ,  $y = 0.866025403784439$ ,  $z = 0$ , and all velocity components are 0 (since, L1 is an equilibrium point).
- $\bar{x}_0 = [0.469286046598223, 0.873468610828950, 0, 0, 0.010700524596717, 0]$   
(unitless)

For linear propagation, the following equation is used:

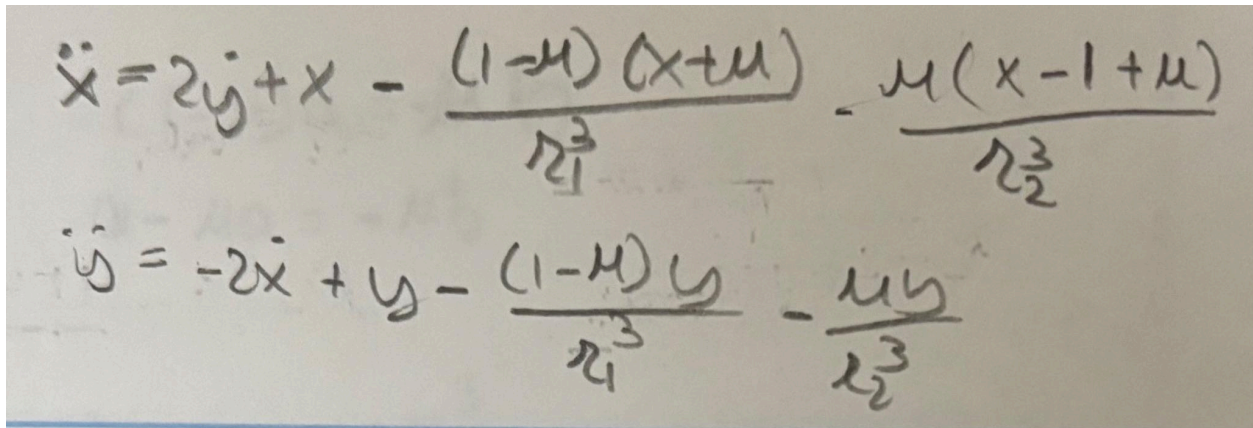
$$\begin{bmatrix} \dot{\xi} \\ \dot{\eta} \\ \ddot{\xi} \\ \ddot{\eta} \end{bmatrix} = \begin{bmatrix} 0_{2 \times 2} & I_{2 \times 2} \\ U_{XX}^* & \Omega \end{bmatrix} \begin{bmatrix} \xi \\ \eta \\ \dot{\xi} \\ \dot{\eta} \end{bmatrix} \longrightarrow \delta \dot{x}_{2D} = A_{2D} \delta \bar{x}_{2D}$$

Since the initial variations are known, this EOM is plugged into ODE45() to numerically integrate the trajectory.  $A_{2D}$  is constant (and using partial derivative functions at L4 from Problem 2) :

$$A_{2D} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{3}{4} & 1.267469961406722 & 0 & 2 \\ 1.267469961406722 & 2.25 & -2 & 0 \end{bmatrix}$$

The output of the numerical integrator is used to plot the linear trajectory.

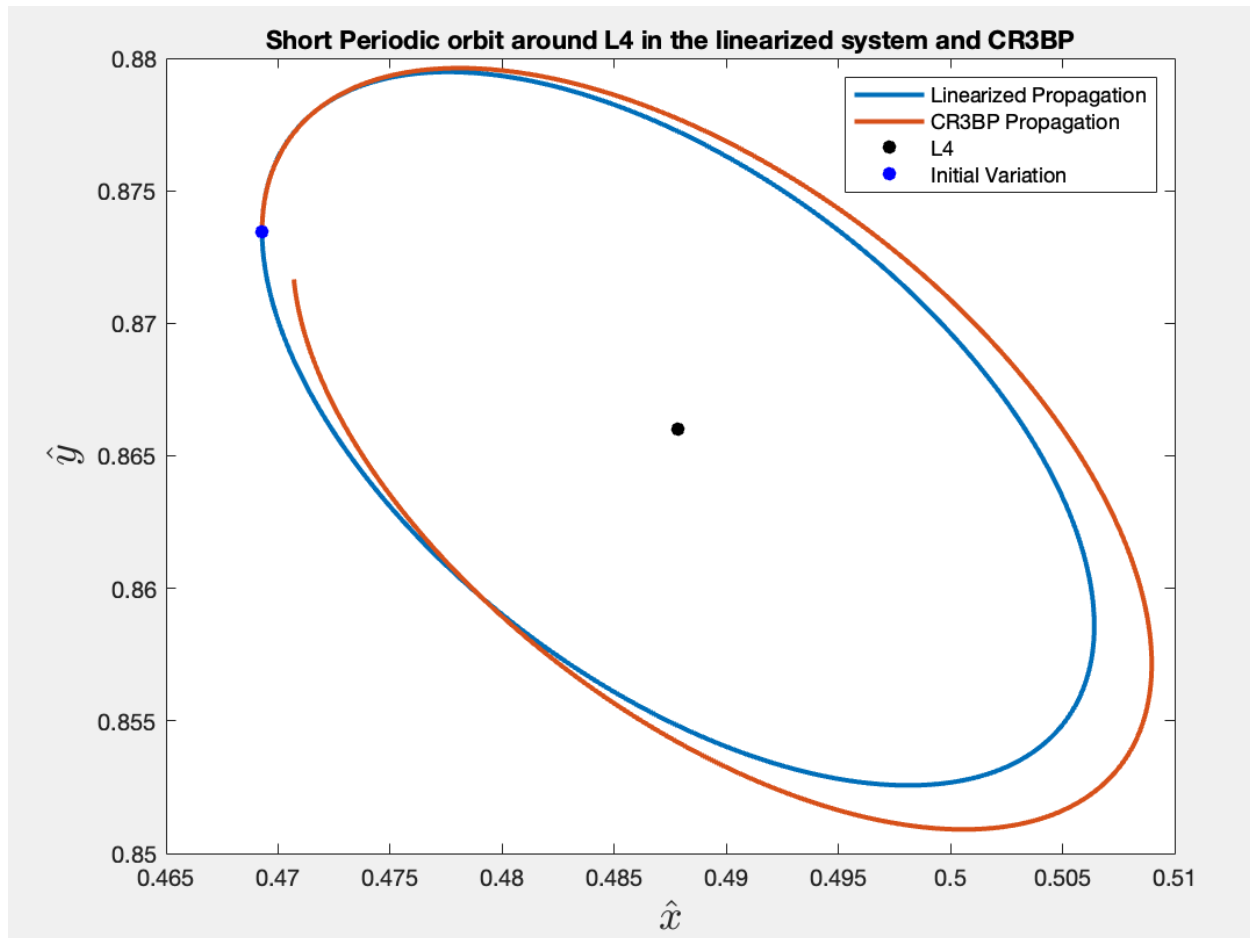
For the non-linear propagation, EOMs from the CR3BP are used as shown below:


$$\ddot{x} = 2\dot{y} + x - \frac{(1-\mu)(x+\mu)}{r_1^3} - \frac{\mu(x-1+\mu)}{r_2^3}$$
$$\ddot{y} = -2\dot{x} + y - \frac{(1-\mu)y}{r_1^3} - \frac{\mu y}{r_2^3}$$

(The EOM for acceleration in z is zero as this motion is only restricted to the xy plane)

These EOMs are then numerically integrated through Matlab's ODE113() integrator to get the non-linear trajectory.

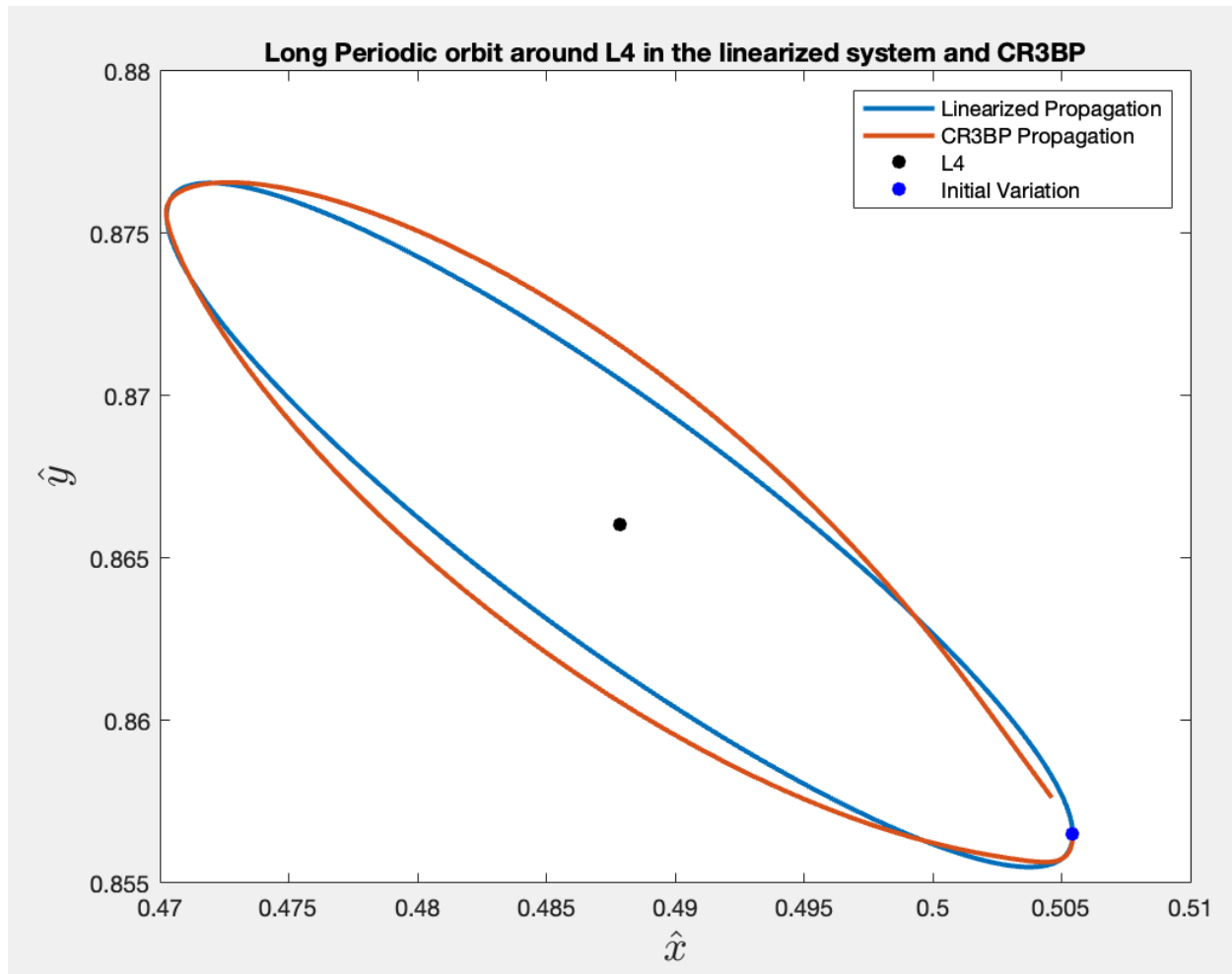
Below are the results for linear vs CR3BP propagation using the initial variation for the short period:



Using all the initial variations for the initial state vector for the CR3BP for the **long period**:

- For L4, position vector in rotating frame is  $\rightarrow x = 0.487849415605290$ ,  $y = 0.866025403784439$ ,  $z = 0$ , and all velocity components are 0 (since, L1 is an equilibrium point).
- $\bar{x}_0 = [0.505433521178833, 0.856496540726419, 0, 0, -0.001337126335027, 0]$  (unitless)

The linear and non-linear propagations are done the same way as the short period trajectories.



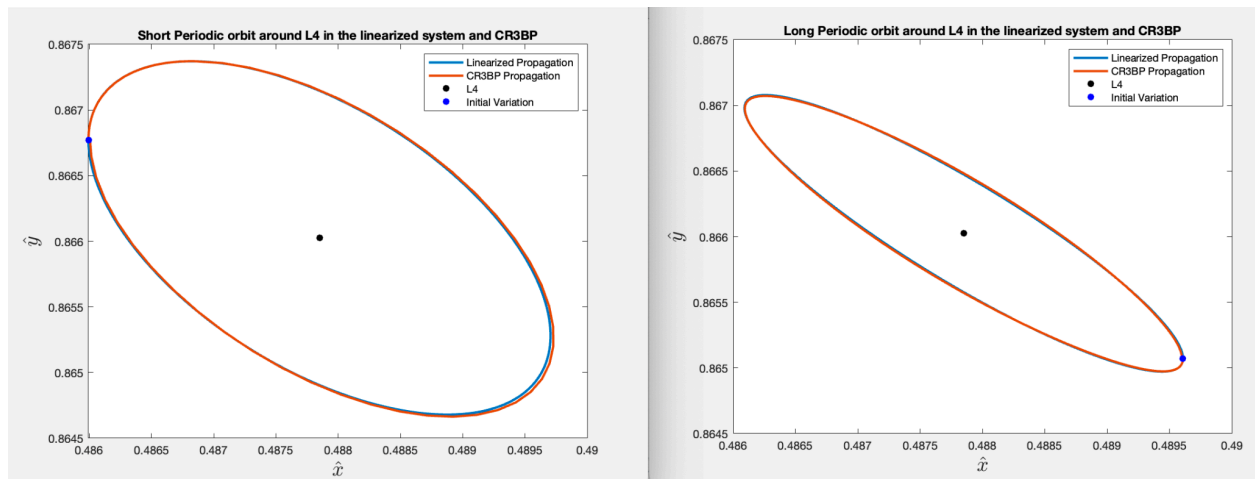
As expected, the linearized propagation is periodic and follows an elliptical path while the non-linear (CR3BP) propagation follows the linear trajectory closely, but is not periodic within the time specified i.e. it does not return to the initial variation within the time specified. Visually, both the trajectories don't match completely, but for a linear approximation, this does a much better job of getting a periodic path around L4

### Part d

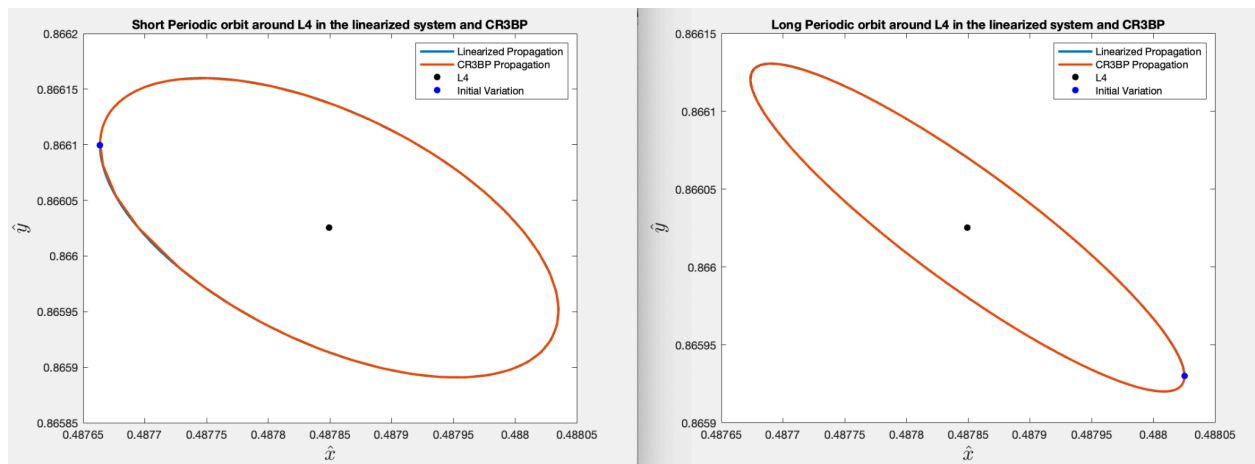
The non-linear trajectory generated clearly isn't periodic around the L4 equilibrium point. But, it is a much better approximation than Problem 4. For a first pass, I would argue that these supply a "good enough" approximation for a planar periodic orbit around L4. However, this can be greatly improved.

When the magnitude of the linearized initial variations is reduced, the linear approximation follows the non-linear path increasingly well. Below are a few examples:

When the position components magnitude is decreased to 0.002, here are the trajectories:



Similarly, when the magnitude is decreased to 0.0002, here are the trajectories:



As seen, decreasing the magnitude improves the linear approximation. The above trajectories are very good linear approximations for the planar periodic orbits around L4 in the nonlinear system.

---

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

## Constants

## 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

% Sun
mu_sun = 132712440041.279419; % km3/s2
mass_sun = mu_sun / 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));

% Sun-Earth system
mass_ratio_se = mass_earth / (mass_earth + mass_sun);
m_star_se = mass_earth + mass_sun;
l_star_se = a_earth;
t_star_se = sqrt(l_star_se^3/(G * m_star_se));
```

## ASEN 6060 - HW 2, Problem 1

### Part a, b

```
mu = mass_ratio_em;

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

% Jacobi constant at each eq point
for i = 1:5
    x = em_eq_pts(i, 1);
```

---

```
y = em_eq_pts(i, 2);
r1 = sqrt((x + mu)^2 + y^2);
r2 = sqrt((x - 1 + mu)^2 + y^2);
c_at_eq(i) = (x^2 + y^2) + 2*(1 - mu)/r1 + 2*mu/r2;
end
```

## Part c

```
mu_range = linspace(1e-7, 0.5, 100);

for i = 1:length(mu_range)
    [eq_pts(:, :, i), eq_validity(:, :, i)] = all_eq_points(mu_range(i));
end

figure()
subplot(2,2,1)
plot(mu_range, squeeze(eq_pts(1,1,:)), 'LineWidth', 2)
xlabel("Mass Ratio [mu]")
ylabel("L1 x position")

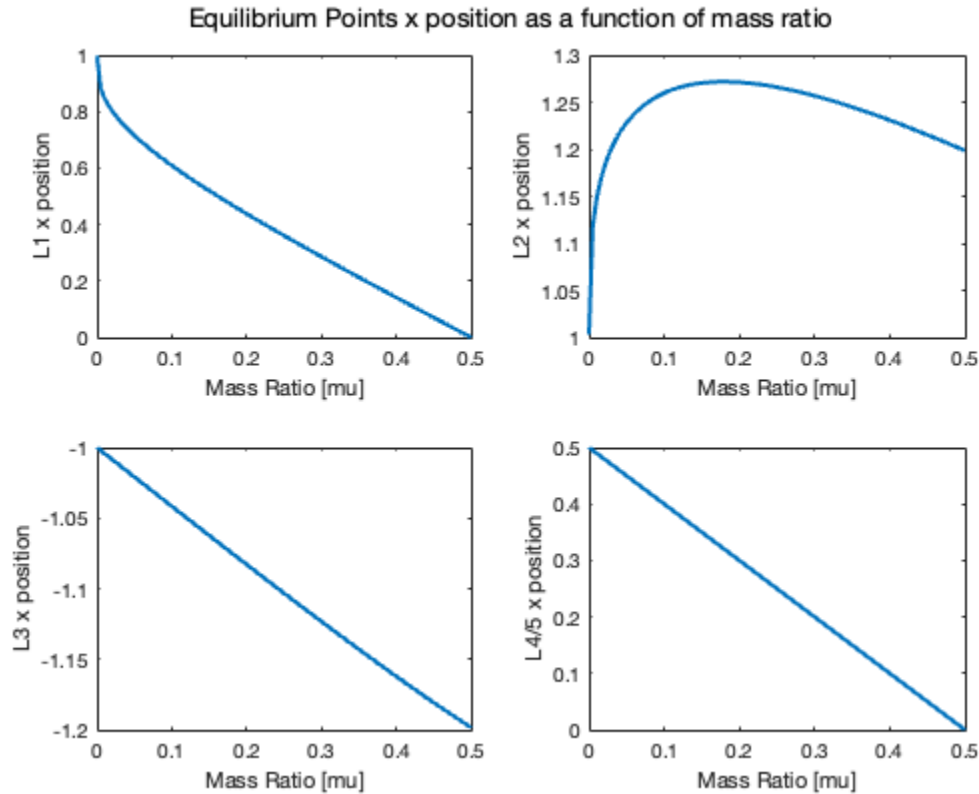
subplot(2,2,2)
plot(mu_range, squeeze(eq_pts(2,1,:)), 'LineWidth', 2)
xlabel("Mass Ratio [mu]")
ylabel("L2 x position")

subplot(2,2,3)
plot(mu_range, squeeze(eq_pts(3,1,:)), 'LineWidth', 2)
xlabel("Mass Ratio [mu]")
ylabel("L3 x position")

subplot(2,2,4)
plot(mu_range, squeeze(eq_pts(4,1,:)), 'LineWidth', 2)
xlabel("Mass Ratio [mu]")
ylabel("L4/5 x position")

sgtitle("Equilibrium Points x position as a function of mass ratio")
```





## ASEN 6060 - HW 2, Problem 2

### Evals for EM system

```
mu = mass_ratio_em;  
  
% Earth Moon system equilibrium points  
[em_eq_pts, em_eq_validity] = all_eq_points(mu);  
  
% Calculate out of plane modes for all 5 eq points  
for i = 1:5  
    em_eq_pts_out_of_plane_modes(i,:) = out_of_plane_modes(mu,  
em_eq_pts(i,:));  
    em_eq_pts_in_plane_modes(i,:) = in_plane_modes(mu, em_eq_pts(i,:));  
end
```

### Evals for SE system

```
mu = mass_ratio_se;  
  
% Earth Moon system equilibrium points  
[se_eq_pts, se_eq_validity] = all_eq_points(mu);  
  
% Calculate out of plane modes for all 5 eq points
```

---

```
for i = 1:5
    se_eq_pts_out_of_plane_modes(i,:) = out_of_plane_modes(mu,
se_eq_pts(i,:));
    se_eq_pts_in_plane_modes(i,:) = in_plane_modes(mu, se_eq_pts(i,:));
end
```

## ASEN 6060 - HW 2, Prob 4

### Part a

```
syms lambda_1 lambda_3 uxx

alpha_1 = (lambda_1^2 - uxx)/(2*lambda_1);
alpha_3 = (lambda_3^2 - uxx)/(2*lambda_3);

mat = [1 1 1 1;
        lambda_1 -lambda_1 lambda_3 -lambda_3;
        alpha_1 -alpha_1 alpha_3 -alpha_3;
        alpha_1*lambda_1 alpha_1*lambda_1 alpha_3*lambda_3 alpha_3*lambda_3];

mat_inv = inv(mat);
```

### Part c

```
mu = mass_ratio_em;

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

% Calculate out of plane modes for all 5 eq points
for i = 1:5
    em_eq_pts_out_of_plane_modes(i,:) = out_of_plane_modes(mu,
em_eq_pts(i,:));
    em_eq_pts_in_plane_modes(i,:) = in_plane_modes(mu, em_eq_pts(i,:));
end

% Only looking at L1 eq point planar oscillatory modes
l1_pos = em_eq_pts(1,:);
lambda_1 = em_eq_pts_in_plane_modes(1,1);
lambda_3 = em_eq_pts_in_plane_modes(1,3);
uxx_l1 = u_xx(mu, l1_pos);
uxy_l1 = u_xy(mu, l1_pos);
uyy_l1 = u_yy(mu, l1_pos);
alpha_1 = (lambda_1^2 - uxx_l1)/(2*lambda_1);
alpha_3 = (lambda_3^2 - uxx_l1)/(2*lambda_3);

xi_0 = -0.001;
% xi_0 = -0.000001;
eta_0 = 0.0;
xi_dot_0 = lambda_3 * eta_0 / alpha_3;
eta_dot_0 = alpha_3 * lambda_3 * xi_0;
init_var = [xi_0, xi_dot_0, eta_0, eta_dot_0];
```

---

```

% Time is one period
t = linspace(0, 2*pi/imag(lambda_3), 1000);

A3 = 1/(lambda_1^2 - lambda_3^2) * (xi_0*alpha_1*lambda_1 +
(alpha_1*lambda_3*lambda_1*xi_dot_0)/uxx_l1 - (lambda_1^2*lambda_3*eta_0)/
uxx_l1 - eta_dot_0);
A4 = 1/(lambda_1^2 - lambda_3^2) * (xi_0*alpha_1*lambda_1 -
(alpha_1*lambda_3*lambda_1*xi_dot_0)/uxx_l1 + (lambda_1^2*lambda_3*eta_0)/
uxx_l1 - eta_dot_0);

for i = 1:length(t)
    xi_t(i) = A3*exp(lambda_3*t(i)) + A4*exp(-lambda_3*t(i));
    eta_t(i) = A3*alpha_3*exp(lambda_3*t(i)) - A4*alpha_3*exp(-lambda_3*t(i));
end

figure()
plot(xi_t+l1_pos(1), eta_t, 'LineWidth',2)
hold on

x0 = [l1_pos(1) + xi_0; l1_pos(2) + eta_0; 0; xi_dot_0; eta_dot_0; 0];

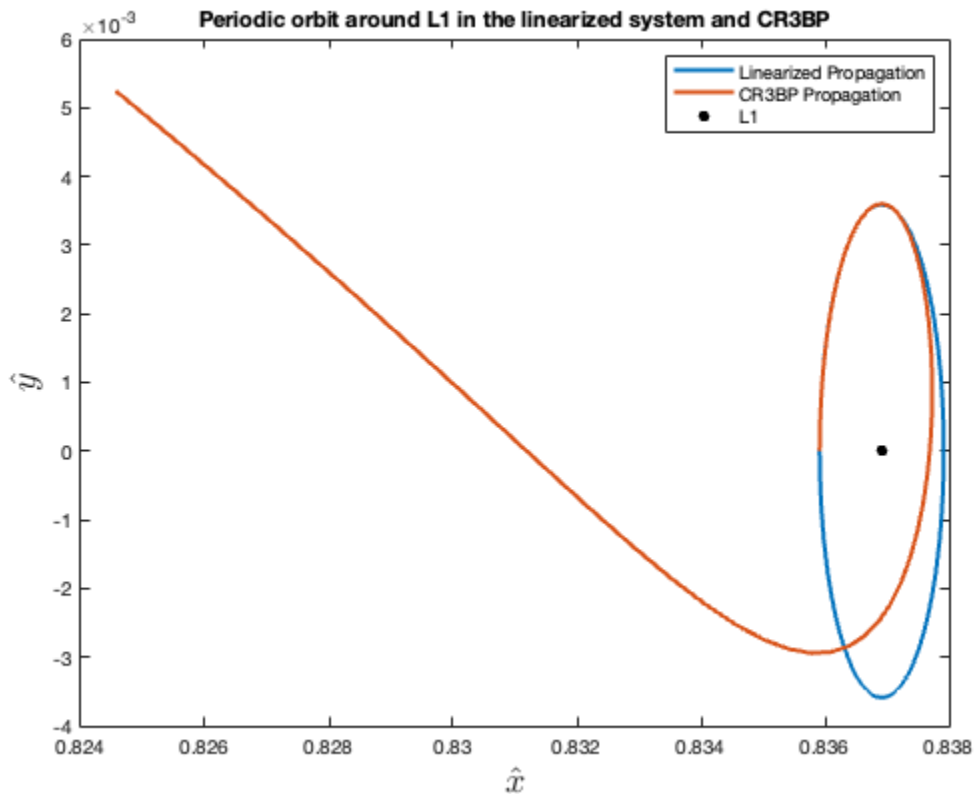
% Set options for ode113
options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);

% Call ode113 function
[tout, xout] = ode113(@(t, state)CR3BP(state, mu), [0 t(end)], x0, options);

plot(xout(:,1), xout(:,2), 'LineWidth',2)
scatter(l1_pos(1), l1_pos(2), 'black', 'filled')
hold off
legend("Linearized Propagation", "CR3BP Propagation", "L1")
xlabel('$$\hat{x}$$', 'Interpreter', 'Latex', 'FontSize', 18)
ylabel('$$\hat{y}$$', 'Interpreter', 'Latex', 'FontSize', 18)
title("Periodic orbit around L1 in the linearized system and CR3BP")

```

---



## Part e

```
U_star_XX = [uxx_l1, uxy_l1; uxy_l1, uyy_l1];
Omega = [0 2; -2 0];

A2D = [zeros(2), eye(2); U_star_XX, Omega];

[V, D] = eig(A2D);

evec_3 = V(:,3);
evec_4 = V(:,4);

basis = evec_3 + evec_4;
```

## ASEN 6060 - HW 2, Problem 5

### Part a

```
mu = mass_ratio_em;

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

% Only looking at L4 eq point planar oscillatory modes
```

---

```

l4_pos = em_eq_pts(4,:);

l4_in_plane_modes = in_plane_modes(mu, l4_pos);

lambda_1 = l4_in_plane_modes(1);
lambda_3 = l4_in_plane_modes(3);
uxx_l4 = u_xx(mu, l4_pos);
uxy_l4 = u_xy(mu, l4_pos);
uyy_l4 = u_yy(mu, l4_pos);
U_star_XX = [uxx_l4, uxy_l4; uxy_l4, uyy_l4];

Omega = [0 2; -2 0];

A2D = [zeros(2), eye(2); U_star_XX, Omega];

[V, D] = eig(A2D);

sp_evec = real(V(:,1));
lp_evec = real(V(:,3));

sp_pos_mag = norm([sp_evec(1), sp_evec(2)]);
lp_pos_mag = norm([lp_evec(1), lp_evec(2)]);

pos_mag_req = 0.02;
% pos_mag_req = 0.002;
% pos_mag_req = 0.0002;

sp_mag_factor = pos_mag_req / sp_pos_mag;
lp_mag_factor = pos_mag_req / lp_pos_mag;

sp_ic = sp_evec .* sp_mag_factor;
lp_ic = lp_evec .* lp_mag_factor;

% Short Period Linear Prop
xi_0 = sp_ic(1);
xi_dot_0 = sp_ic(3);
eta_0 = sp_ic(2);
eta_dot_0 = sp_ic(4);
dx0 = [xi_0; eta_0; xi_dot_0; eta_dot_0];

% Time is one period
t = linspace(0, 2*pi/imag(lambda_3), 1000);

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

[t_out, dxs] = ode45(@(t,dx)A2D*dx, t, dx0, options);

xi_t = dxs(:,1);
etai_t = dxs(:,2);

figure()
plot(xi_t+l4_pos(1), etai_t+l4_pos(2), 'LineWidth',2)
hold on

```

---

---

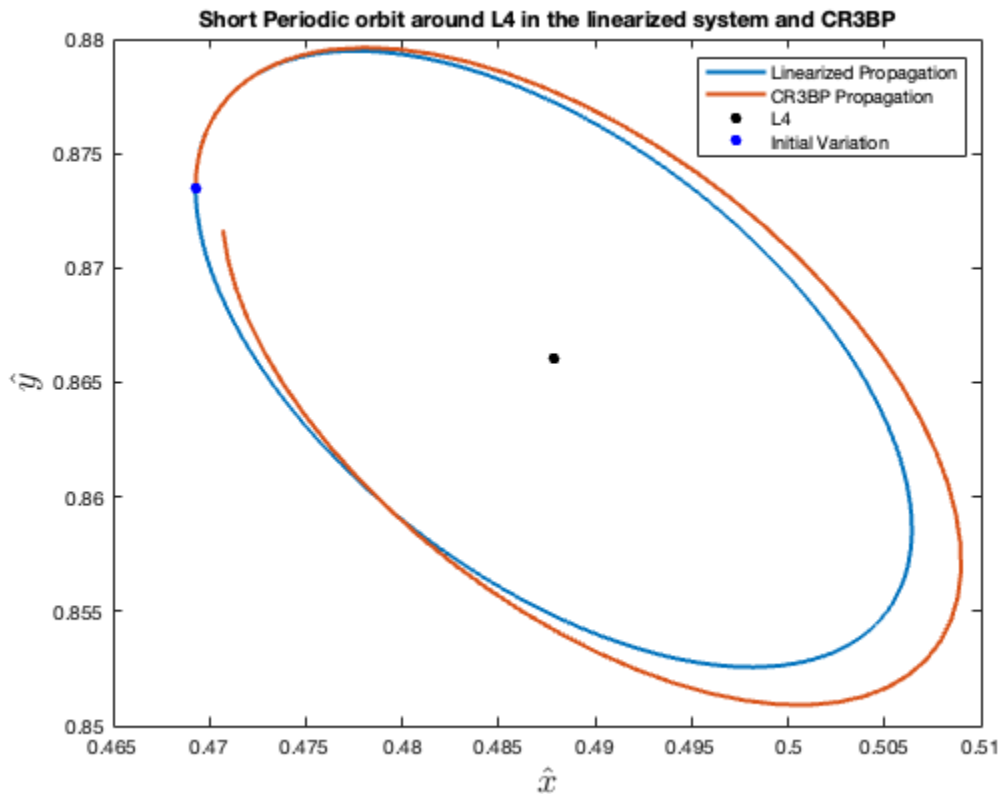
```

x0 = [l4_pos(1) + xi_0; l4_pos(2) + eta_0; 0; xi_dot_0; eta_dot_0; 0];

% Call ode113 function
[tout, xout] = ode113(@(t, state)CR3BP(state, mu), [0 t(end)], x0, options);

plot(xout(:,1), xout(:,2), 'LineWidth',2)
scatter(l4_pos(1), l4_pos(2), 'filled', 'black')
scatter(xi_0 + l4_pos(1), eta_0 + l4_pos(2), 'filled', 'blue')
hold off
legend("Linearized Propagation", "CR3BP Propagation", "L4", "Initial Variation")
xlabel('$$\hat{x}$$','Interpreter','Latex','FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex','FontSize',18)
title("Short Periodic orbit around L4 in the linearized system and CR3BP")

```



## Long Period Linear Prop

```

xi_0 = lp_ic(1);
xi_dot_0 = lp_ic(3);
eta_0 = lp_ic(2);
eta_dot_0 = lp_ic(4);
dx0 = [xi_0; eta_0; xi_dot_0; eta_dot_0];

% Time is one period
t = linspace(0, 2*pi/imag(lambda_1), 1000);

```

---

```

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

[t_out, dxs] = ode45(@(t,dx)A2D*dx, t, dx0, options);

xi_t = dxs(:,1);
etai_t = dxs(:,2);

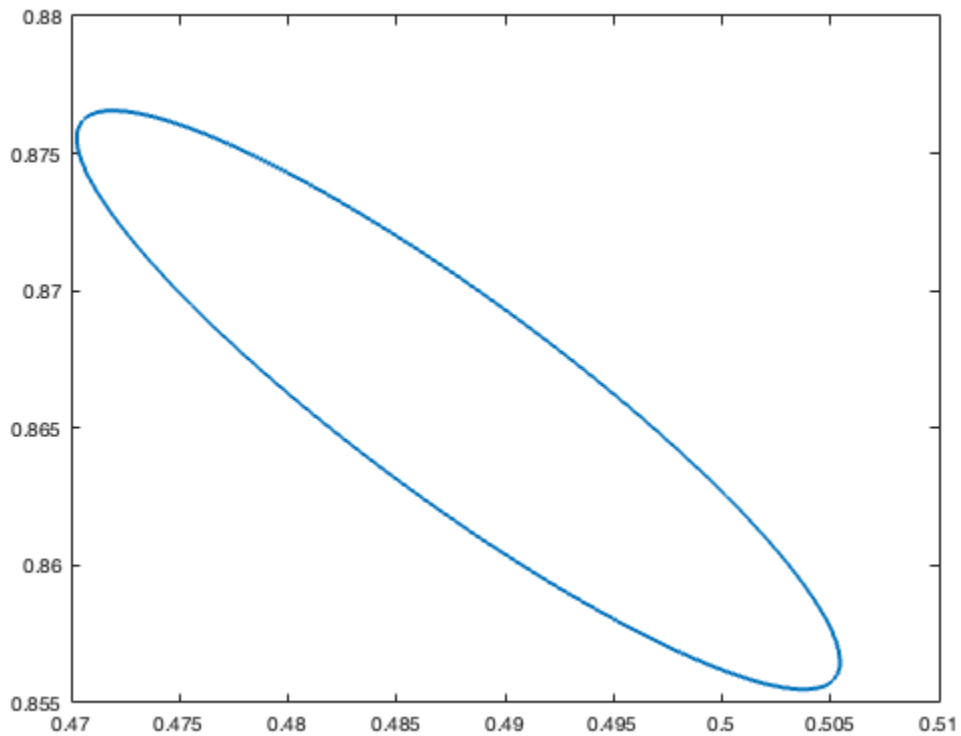
figure()
plot(xi_t+l4_pos(1), etai_t+l4_pos(2), 'LineWidth',2)
hold on

x0 = [l4_pos(1) + xi_0; l4_pos(2) + eta_0; 0; xi_dot_0; eta_dot_0; 0];

% Call ode113 function
[tout, xout] = ode113(@(t, state)CR3BP(state, mu), [0 t(end)], x0, options);

plot(xout(:,1), xout(:,2), 'LineWidth',2)
scatter(l4_pos(1), l4_pos(2), 'filled', 'black')
scatter(xi_0 + l4_pos(1), eta_0 + l4_pos(2), 'filled', 'blue')
hold off
legend("Linearized Propagation", "CR3BP Propagation", "L4", "Initial
Variation")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
title("Long Periodic orbit around L4 in the linearized system and CR3BP")

```



## Functions

```
function state_dot = CR3BP(state, mu)
    % Circular Restricted 3 Body Problem non-dimensional EOMs
    x = state(1);
    y = state(2);
    z = state(3);
    xdot = state(4);
    ydot = state(5);
    zdot = state(6);

    r1 = sqrt((x + mu)^2 + (y)^2 + (z)^2);
    r2 = sqrt((x - 1 + mu)^2 + (y)^2 + (z)^2);

    state_dot(1, 1) = xdot;
    state_dot(2, 1) = ydot;
    state_dot(3, 1) = zdot;

    state_dot(4, 1) = 2*ydot + x - (1 - mu)*(x + mu)/(r1^3) - mu * (x - 1 +
mu)/(r2^3);
    state_dot(5, 1) = -2*xdot + y - (1 - mu)*y/(r1^3) - mu*y/(r2^3);
    state_dot(6, 1) = - (1 - mu)*z/(r1^3) - mu*z/(r2^3);
end

function [x_Ls, validity] = all_eq_points(mu)
```



---

```

% Function that finds all equilibrium point positions

% Initial guess for L1 is midpoint of P1 and P2
x_L1_0 = (-mu + (1 - mu))/2;

% Call function to get position of L1 and validity of L1
[x_L1, x_L1_validity] = find_coll_eq_pts(x_L1_0, mu);

% Initial guess for L2 is distance between P2 and L1. That is added to P2
% position.
x_L2_0 = (1-mu)-x_L1 + (1-mu);

% Call function to get position of L2 and validity of L2
[x_L2, x_L2_validity] = find_coll_eq_pts(x_L2_0, mu);

% Initial guess for L3 is distance between P2 and L1. That is subtracted
% from P1 position.
x_L3_0 = -mu - ((1-mu)-x_L1);

% Call function to get position of L2 and validity of L2
[x_L3, x_L3_validity] = find_coll_eq_pts(x_L3_0, mu);

% Find L4 and L5 points
x_L4 = 1/2 - mu;
y_L4 = sqrt(3)/2;
y_L5 = -sqrt(3)/2;

x_Ls = [x_L1, 0; x_L2, 0; x_L3, 0; x_L4, y_L4; x_L4, y_L5];
validity = [x_L1_validity, x_L2_validity, x_L3_validity, true, true];
end

function fx = collinear_eq_pts(x, mu)
    % Function to get roots for collinear equilibrium points
    fx = x - ((1-mu)*(x+mu))/(abs(x+mu)^3) - (mu*(x-1+mu))/(abs(x-1+mu)^3);
end

function [x_Lx, validity] = find_coll_eq_pts(x0, mu)
    % Function to find collinear equilibrium points

    % Function for collinear equilibrium points
    fun = @(x)collinear_eq_pts(x, mu);

    % Set display to zero and thresholds to 1e-17 to get as close to double
    % precision as possible
    options = optimoptions('fsolve', 'Display','none',
'FunctionTolerance',1e-17, ...
    'StepTolerance',1e-17, 'OptimalityTolerance',1e-17);

    % Use fsolve to find root of function
    x_Lx = fsolve(fun, x0, options);

    % Test if the root is sufficiently close to 0. If so, validity is true,
    % else validity is false
    test_Lx = collinear_eq_pts(x_Lx, mu);

```

---

---

```

    validity = false;
    if abs(test_Lx) < 1e-15
        validity = true;
    end
end

function out = in_plane_modes(mu, x_eq)
    % Calculate four in plane modes for eq points
    uxx = u_xx(mu, x_eq);
    uyy = u_yy(mu, x_eq);
    uzz = u_zz(mu, x_eq);
    uxy = u_xy(mu, x_eq);
    Lambda_1 = (-4+uxx+uyy)/2 + (sqrt((4-uxx-uyy)^2 - 4*(uxx*uyy - uxy^2)))/2;
    Lambda_2 = (-4+uxx+uyy)/2 - (sqrt((4-uxx-uyy)^2 - 4*(uxx*uyy - uxy^2)))/2;
    lambda_1 = sqrt(Lambda_1);
    lambda_2 = -sqrt(Lambda_1);
    lambda_3 = sqrt(Lambda_2);
    lambda_4 = -sqrt(Lambda_2);
    out = [lambda_1, lambda_2, lambda_3, lambda_4];
end

function out = out_of_plane_modes(mu, x_eq)
    % Calculate two out of plane modes for eq points
    lambda_pos = sqrt(u_zz(mu, x_eq));
    lambda_neg = -sqrt(u_zz(mu, x_eq));
    out = [lambda_pos, lambda_neg];
end

function out = u_xx(mu, x_eq)
    % Pseudo potential function partial derivative wrt x, x at eq point
    % Assuming z = 0
    x = x_eq(1);
    y = x_eq(2);
    r1 = sqrt((x + mu)^2 + y^2);
    r2 = sqrt((x - 1 + mu)^2 + y^2);
    out = 1 - (1-mu)/(r1^3) - mu/(r2^3) + (3*(1-mu)*(x+mu)^2)/(r1^5) +
    (3*mu*(x-1+mu)^2)/(r2^5);
end

function out = u_xy(mu, x_eq)
    % Pseudo potential function partial derivative wrt x, y at eq point
    % Assuming z = 0
    x = x_eq(1);
    y = x_eq(2);
    r1 = sqrt((x + mu)^2 + y^2);
    r2 = sqrt((x - 1 + mu)^2 + y^2);
    out = (3*(1-mu)*(x+mu)*y)/(r1^5) + (3*mu*y*(x-1+mu))/(r2^5);
end

function out = u_yy(mu, x_eq)
    % Pseudo potential function partial derivative wrt y, y at eq point
    % Assuming z = 0
    x = x_eq(1);
    y = x_eq(2);

```

---

---

```

    r1 = sqrt((x + mu)^2 + y^2);
    r2 = sqrt((x - 1 + mu)^2 + y^2);
    out = 1 - (1-mu)/(r1^3) - mu/(r2^3) + (3*(1-mu)*y^2)/(r1^5) + (3*mu*y^2)/
(r2^5);
end

function out = u_zz(mu, x_eq)
    % Pseudo potential function partial derivative wrt z, z at eq point
    % Assuming z = 0
    x = x_eq(1);
    y = x_eq(2);
    r1 = sqrt((x + mu)^2 + y^2);
    r2 = sqrt((x - 1 + mu)^2 + y^2);
    out = -(1-mu)/(r1^3) - mu/(r2^3);
end

```

*Published with MATLAB® R2024a*