```
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

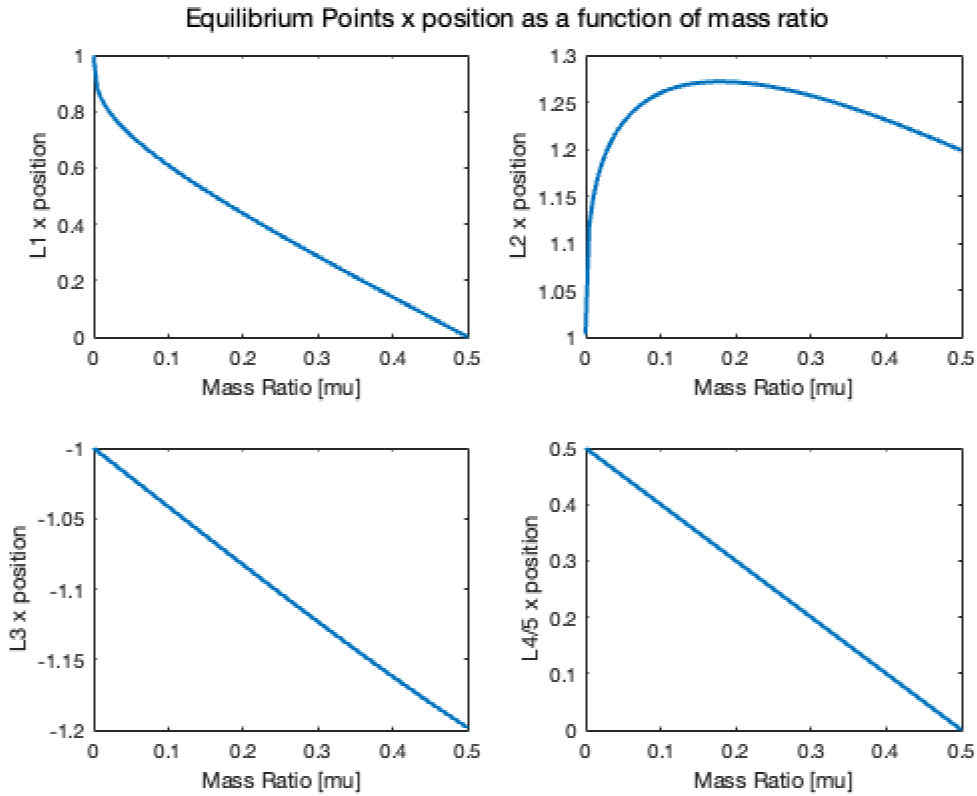
Part a, b

```
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

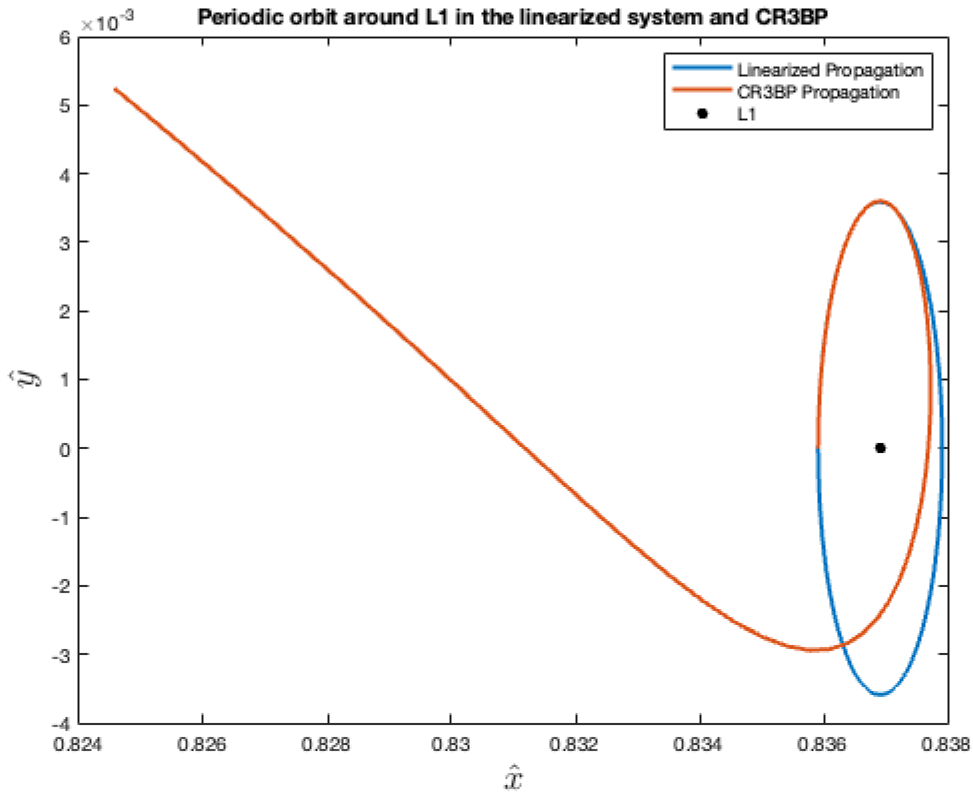
Part a

```
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;
```

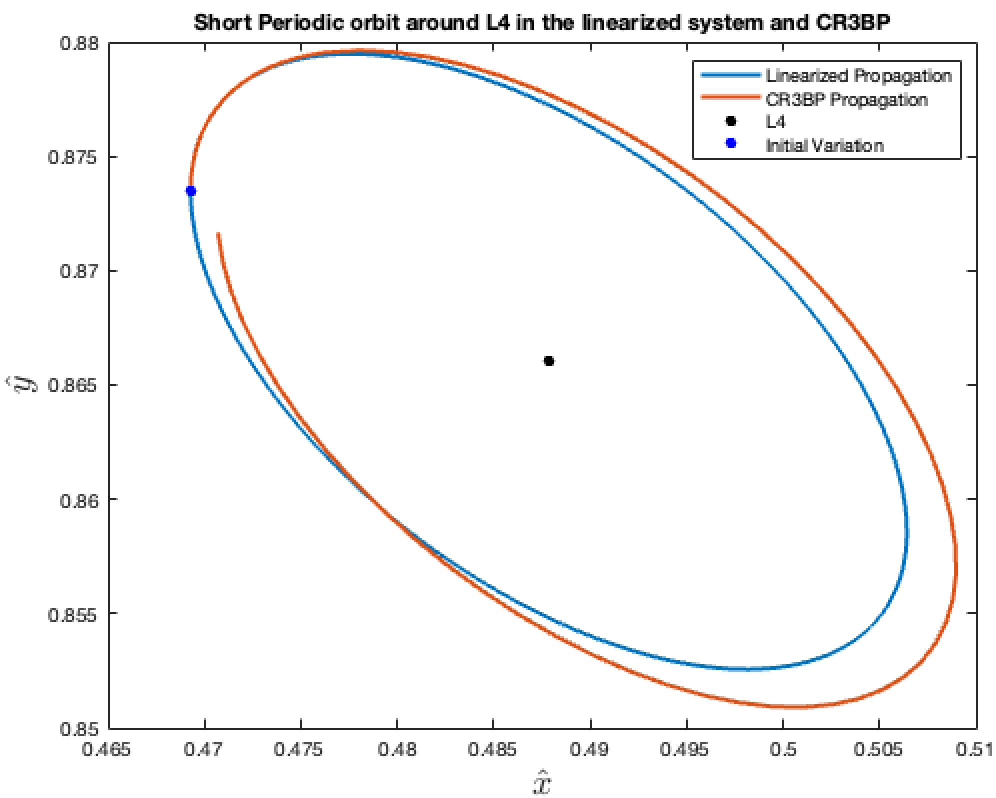
ASEN 6060 - HW 2, Problem 5

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
```

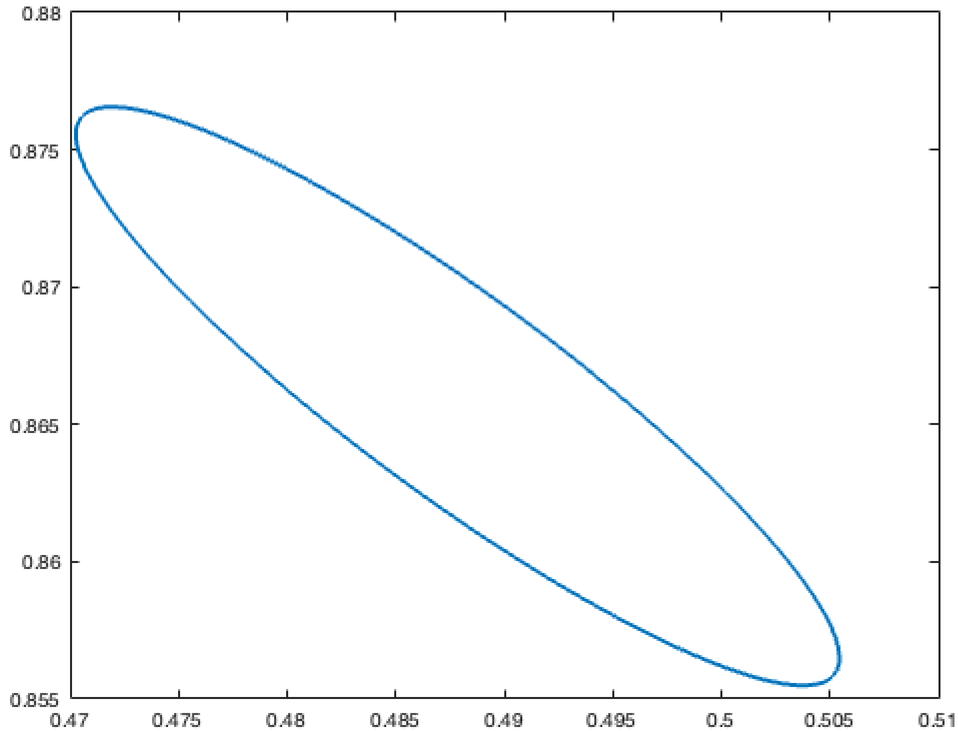
```
x0 = [l4_pos(1) + xi_0; l4_pos(2) + eta_0; 0; xi_dot_0; eta_dot_0; 0];

% 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