ASEN 6060 - Final Project
Spring 2025
Jash Bhalavat

I. Motivation **1**
II. Dynamical Model **2**
III. Orbit Selection **3**
IV. Initial Guess Construction **4**
Transfers 1A & 1B 6
Transfers 2A & 2B 11
V. Multiple Shooting Formulation **16**
VI. Transfer Results and Analysis **18**
Transfers 1A & 1B 18
Transfers 2A & 2B 22
Comparing the Transfers 25
VII. Recommendations for Future Analysis **26**

I. Motivation

With increasing interest in cislunar space and the interest in establishing a habitation around the moon, it is imperative that all areas of the moon be accessible to earth. So, if a moon base is established on the far-side, it is necessary to maintain communication with a spacecraft at all times. An L2 Lyapunov orbit can be used precisely for this purpose. Such an orbit would be periodic and always remain on the far-side of the moon hence providing regular communication windows. However, earth communications might be challenging if the spacecraft is always on the far side. So, another spacecraft can be added to this mission that swaps with the far-side spacecraft. The mission then has 1 spacecraft that rests in an L1 Lyapunov periodic orbit until it has to swap with the far-side spacecraft. So, in this mission, 2 spacecraft are constantly moving between an L1 Lyapunov orbit and an L2 Lyapunov orbit. This project presents transfers between L1 and L2 Lyapunov orbits using stable/unstable manifolds within the Circular Restricted 3 Body Problem (CR3BP).

II. Dynamical Model

This project utilizes a Circular Restricted 3 Body Problem (CR3BP) as the dynamical model. This is a restricted 3-Body Problem dynamical model. These are the assumptions that are fundamental to the CR3BP:

- Model only gravitational interactions between 3 bodies (earth, moon, and the spacecraft).
- Model each body with constant mass and the same gravity field as a point mass.
- Mass of P3 (spacecraft) << Masses of P1 (earth), P2 (moon)
- P1 and P2 follow circular orbits.
- P3 does not influence paths of P1 and P2.

The non-dimensional equations of motion in the rotating frame are as follows:

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

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

$$\ddot{z} = -\frac{(1-\mu)z}{r_1^3} - \frac{\mu z}{r_2^3}$$

$$r1 = \sqrt{(x+\mu)^2 + y^2 + z^2}$$

$$r2 = \sqrt{(x-1+\mu)^2 + y^2 + z^2}$$

Other relevant mathematical definitions for this project are as follows:

- μ - CR3BP mass ratio for the earth-moon system. For this project, this value is 0.012150584394710.
- For numerical integration within this model, Matlab's ODE113() function is used.
  - ODE113() has options to set relative and absolute tolerances. The relative tolerance roughly controls the number of correct digits in the solution greater than the absolute tolerance. The absolute tolerance dictates the lower bound of the solution. ODE113() computes the local error at each time step and this value has to be less than either the absolute or relative tolerances. Standard double precision is accurate up to 16 decimal places. So, 1e-16 is the lowest possible tolerance when operating with standard data types in Matlab. Therefore a tolerance of 1e-12 should suffice. Additional experimentation was done

throughout this project and this tolerance strikes a balance between precision and computation time.

- A periodic orbit is not analytically present in the CR3BP. So, a periodic orbit within some tolerance is acceptable within this model. For this project, if the spacecraft's final and initial state vectors have a difference of less than 1e-12, the orbit is known as a periodic orbit.

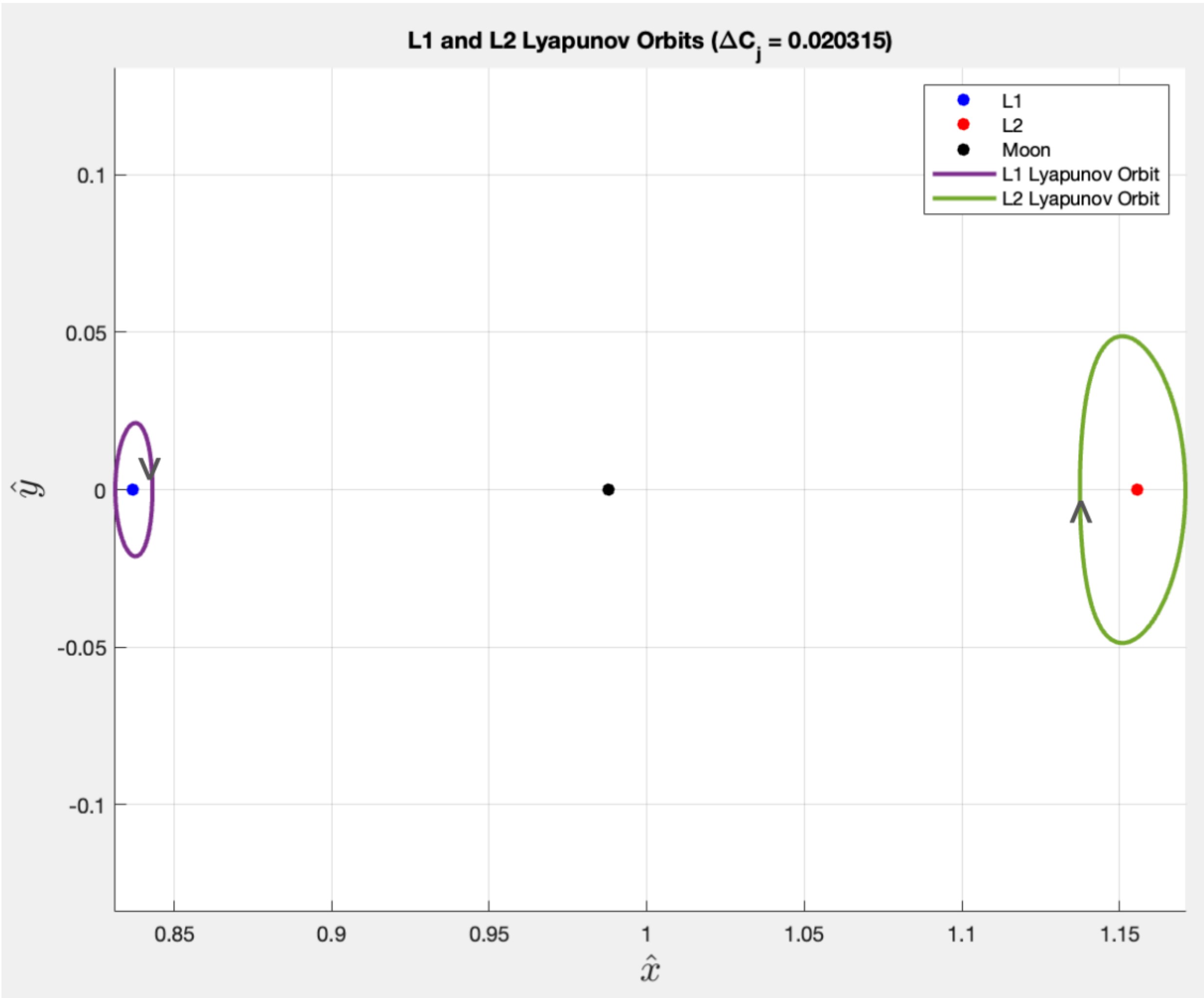
III. Orbit Selection

For this project Lyapunov orbits around L1 and L2 points (in the earth-moon system) were selected. Since periodic orbits are necessary for this project, Lyapunov orbits were one of the candidates. The motivation behind selecting these orbits is that these orbits completely lie in the earth-moon plane. Going back to the motivation behind this mission, it is necessary to communicate with the earth when the spacecrafts are on the near-side of the moon and a planar orbit would make tracking from an earth-based ground station easier.

	L1 Lyapunov	L2 Lyapunov
Jacobi Constant [-]	3.186303038920070	3.165988510858649
Orbit Period [-]	2.698788267675778	3.385307332941585
Initial State Vector [-]	[0.831330619145024, 0.0, 0.0, 0.048817317708961, 0.0]^T	[1.170871819796487, 0.0, 0.0, -0.088163404081646, 0.0]^T

The Jacobi constant difference between the two orbits is 0.020314528061422. These orbits are obtained using a combination of corrections and continuation schemes. Firstly, linearized dynamics are used to form an initial guess for the Lyapunov periodic orbit. Then, that initial guess is corrected using a corrections scheme. Then, using continuation the Lyapunov orbit family is computed. This project pulls one such orbit from the L1 and L2 Lyapunov orbit families. Throughout the rest of this project, the transfers will be separated into 1A, 1B, 2A, 2B. The distinctions are:

- 1A - Transfer from L1 Lyapunov Orbit to L2 Lyapunov Orbit with only 1 orbit around the moon.
- 1B - Transfer from L1 Lyapunov Orbit to L2 Lyapunov Orbit with 2 orbits around the moon.
- 2A - Transfer from L2 Lyapunov Orbit to L1 Lyapunov Orbit with only 1 orbit around the moon.
- 2B - Transfer from L2 Lyapunov Orbit to L1 Lyapunov Orbit with 2 orbits around the moon.



IV. Initial Guess Construction

The initial guess is constructed using Stable/Unstable manifolds from the L1 and L2 Lyapunov Orbits.

The following procedure is used to compute the manifolds for a period orbit:

- Firstly, an event function is newly added.
  - The event function sets 3 values based on state inputs
    - Value
      - The value that we want to be 0.
      - For the moon, we want the x value to be 1-μ, so we want $x - (1 - \mu) = 0$
    - isTerminal
      - This value is set to 1 to indicate to ODE that the integration needs to be terminated when value = 0.

        - Direction
            - This indicates which direction of the function at the terminal point.
                - For 1(A&B) orbits, this is set to 1 because the x-velocity is positive as the spacecraft travels from L1 to L2.
                - For 2(A&B) orbits, this is set to -1 because the x-velocity is negative as the spacecraft travels from L2 to L1.
- Then, a step size value is set - d
    - d is set to 50 km and normalized by the average semi-major axis of the earth-moon system (384,400 km from HW1 constants).
        - This value is provided from lectures and corroborated by the manifolds later in this problem.
- Then, a figure is instantiated and the Lyapunov orbit is plotted, along with the equilibrium point, earth, and moon.
- The STM matrix for the initial state is obtained from the input state - $\Phi(t_1 + T,\ t_1)$.
    - It is the STM at the last step of the input state.
- A for loop is instantiated to run for each discrete step in the periodic orbit.
    - An STM from the current state to a full period in the future is required – $\Phi(t_j + T,\ t_j)$. This is obtained from the following equation:

$$\Phi(t_j + T,\ t_j) = \Phi(t_j, t_1) * \Phi(t_1 + T,\ t_1) * \Phi(t_j, t_1)^{-1}$$

    - Eigenvalues of this STM are obtained using Matlab's eig() function.
    - Then, the trivial eigenvalues of the STM are found by subtracting each value by 1 and checking the remainder. The index of the remainder array with the minimal value is the index of the trivial eigenvalue.
    - Then, only the real eigenvalues are checked for stable and unstable modes.
        - If a real eigenvalue has a value less than 1, its index is registered as stable.
        - If a real eigenvalue has a value greater than 1, its index is registered as unstable.
    - The stable and unstable eigenvalues and corresponding eigenvectors are obtained using these indices and normalized by the position states (first 3 elements of the eigenvectors).
    - Then, the stable and unstable steps are calculated using the normalized eigenvectors using the following equations:

$$\bar{x}_s = \bar{x}_{PO} \pm d\bar{v}^s(\bar{x}_{PO})$$

$$\bar{x}_u = \bar{x}_{PO} \pm d\bar{v}^u(\bar{x}_{PO})$$

    - These are classified as moon-bound or earth-bound based by propagating using the CR3BP equations and checking the last position vector. Since, only the moon bound manifolds are required for the project, all the earth-bound manifolds are discarded.

- These stable and unstable manifolds are then numerically integrated using ODE113() and the event functions are also passed to make sure the propagation stops at the moon and the earth respectively.
- Then, these trajectories are plotted.
- This loop is repeated for each state along the periodic orbit.

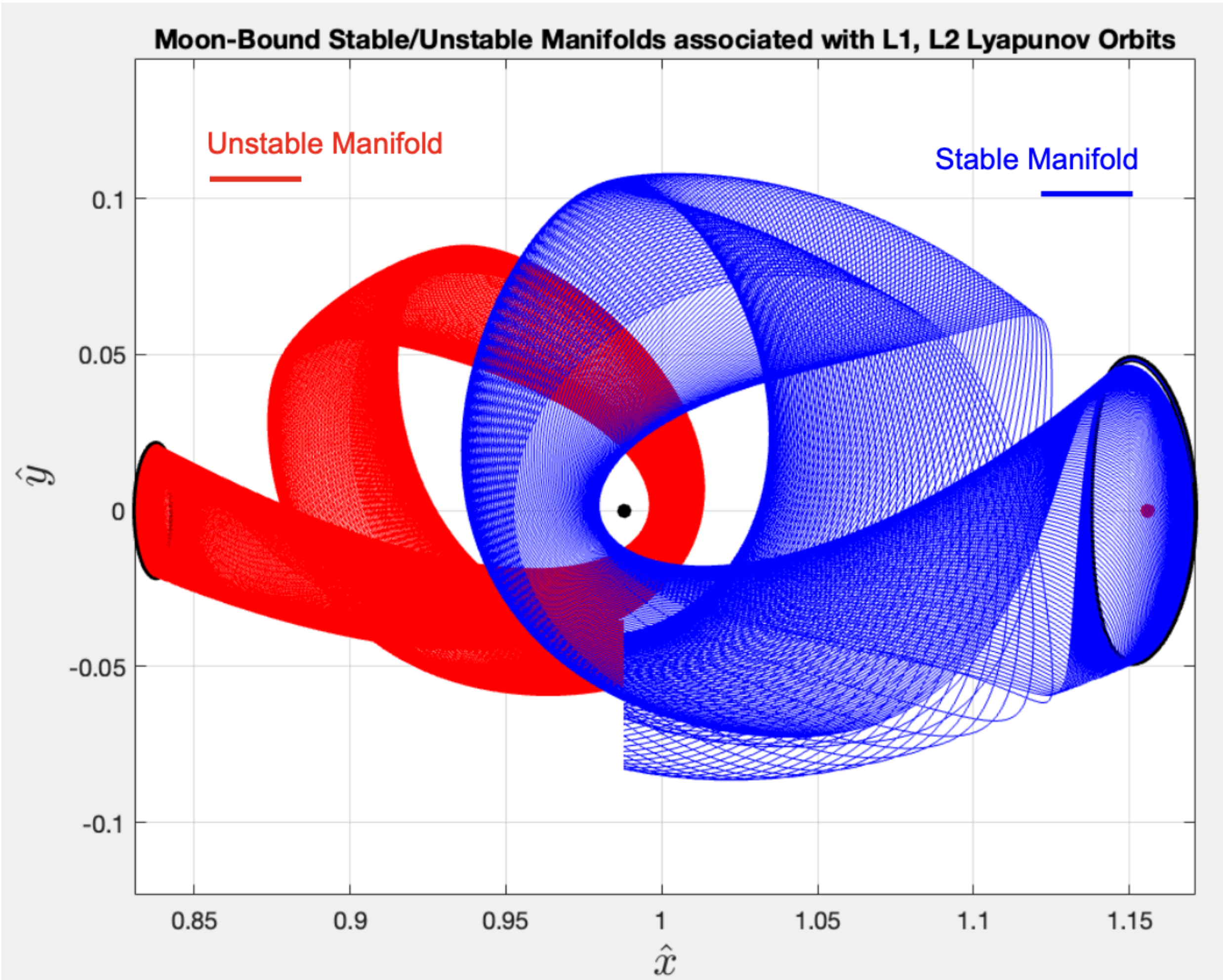
Poincaré maps are also used to map the manifolds on a 2D plane and construct initial guesses. Transfers 1 and 2 have the same surface of sections at $x = 1-\mu$ with 2 crossings, but with different velocities. The Poincaré map for transfers 1A and 1B capture positive crossings while Poincaré map for transfers 2A and 2B capture negative crossings.

Transfers 1A & 1B

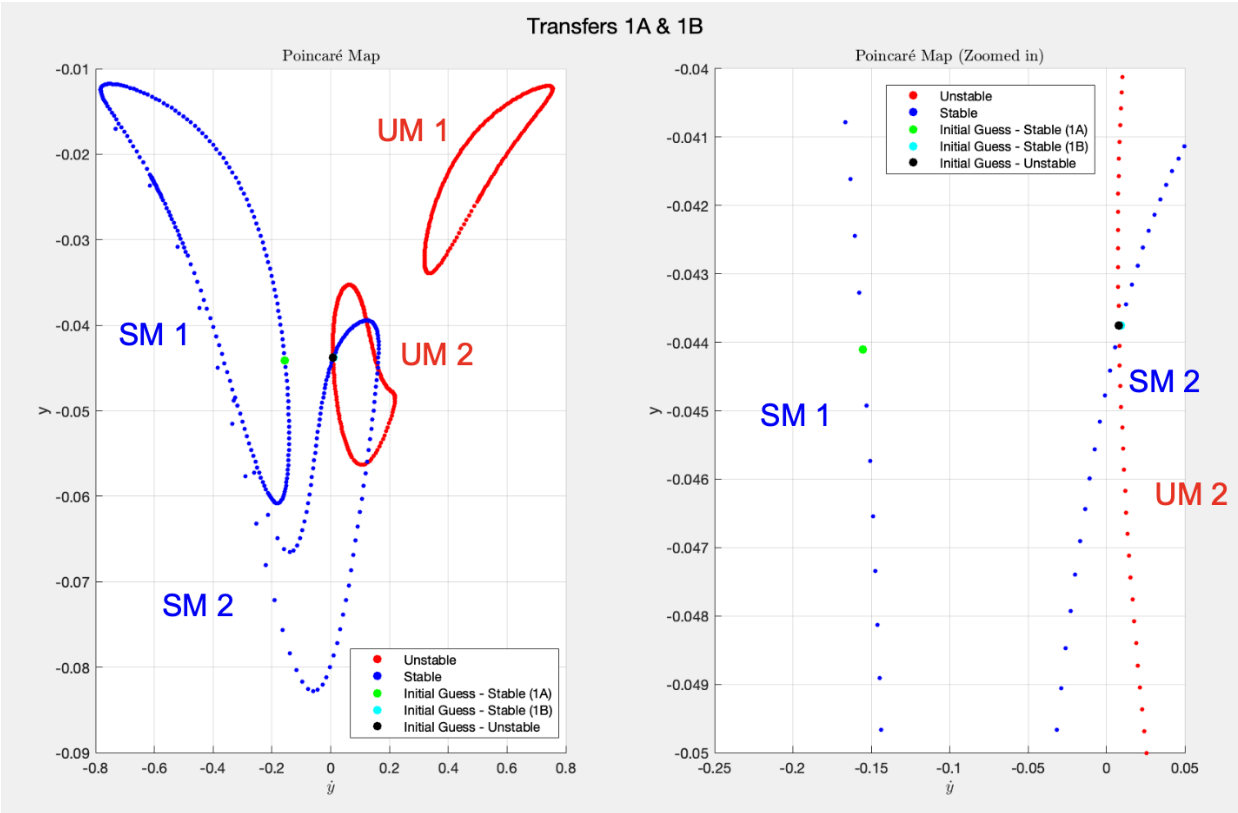
Let's discuss maneuvers 1A and 1B. These transfers are divided into 4 arcs:

1. L1 Lyapunov orbit initial position to L1 Lyapunov orbit position that's closest to the L1 unstable manifold.
2. L1 unstable manifold to L2 stable manifold.
3. L2 stable manifold to point that's closest to the L2 Lyapunov orbit.
4. Point on the L2 Lyapunov orbit that's closest to the L2 stable manifold to the initial state vector given in section III.

The stable/unstable manifolds for 1(A&B) transfers are shown below. Since this transfer aims to go from L1 to L2, an unstable manifold from L1 and a stable manifold to L2 are required.



Then, the initial guesses can be constructed using the Poincaré map and the initial state vectors. Transfer 1A is chosen such that y positions are closest between UM 2 (unstable manifold from L1 with 2 crossing) and SM 1 (stable manifold from L2 with 1 crossings). The 2B transfer is chosen such that the SM 2 (stable manifold from L1 with 2 crossings) and UM 2 are closest to each other. This means that 1A will have 2 crossings of the (1-μ) xz plane before transferring from the unstable to stable manifold and the stable manifold has 1 crossing of the (1-μ) xz plane which means that the transfer will have one orbit around the moon before reaching the L2 Lyapunov orbit. Similarly, 1B will have 2 crossings of the (1-μ) xz plane before transferring from the unstable to stable manifold and the stable manifold has 2 crossings of the (1-μ) xz plane which means that the transfer will have two orbits around the moon before reaching the L2 Lyapunov orbit.



The initial guesses are then found from the manifolds using the y position and y velocities from the Poincaré maps. The points on the map represent phase space locations with the same x and z values, so the maps help in forming initial guesses. The initial guesses methodology is as follows:

1. Initial state for Arc 1 is simply the initial state vector for the L1 Lyapunov orbit.
2. 2 points from the Poincaré map are obtained - Unstable and Stable manifold points (UM & SM).
    a. Each manifold calculated from the Lyapunov orbit is done by perturbing from a discrete time step on the periodic orbit. Such discrete steps have a state vector and time associated with them. These are used to calculate the initial state vectors for the arcs.
3. Since, each point on the L1 Lyapunov orbit is used to calculate a manifold, the UM point also corresponds to a discrete time step on the Lyapunov Orbit. That time step is the flight time for Arc 1.
4. Initial state for Arc 2 is the initial state for the UM. The flight time for Arc 2 is how long the unstable manifold is integrated to meet 2 positive crossings.
5. Since Arc 3 is on a stable manifold which is integrated using negative time, the initial state for Arc 3 is the final state of the SM. The flight time for Arc 3 is how long the stable manifold is integrated to meet 1(for 1A) or 2 (for 1B) negative crossings.
    a. When the stable manifold is computed, it is propagated in backwards time. So, the time has to be manually negated.
6. Going back to the discrete points on the periodic orbit, the initial state and flight time for

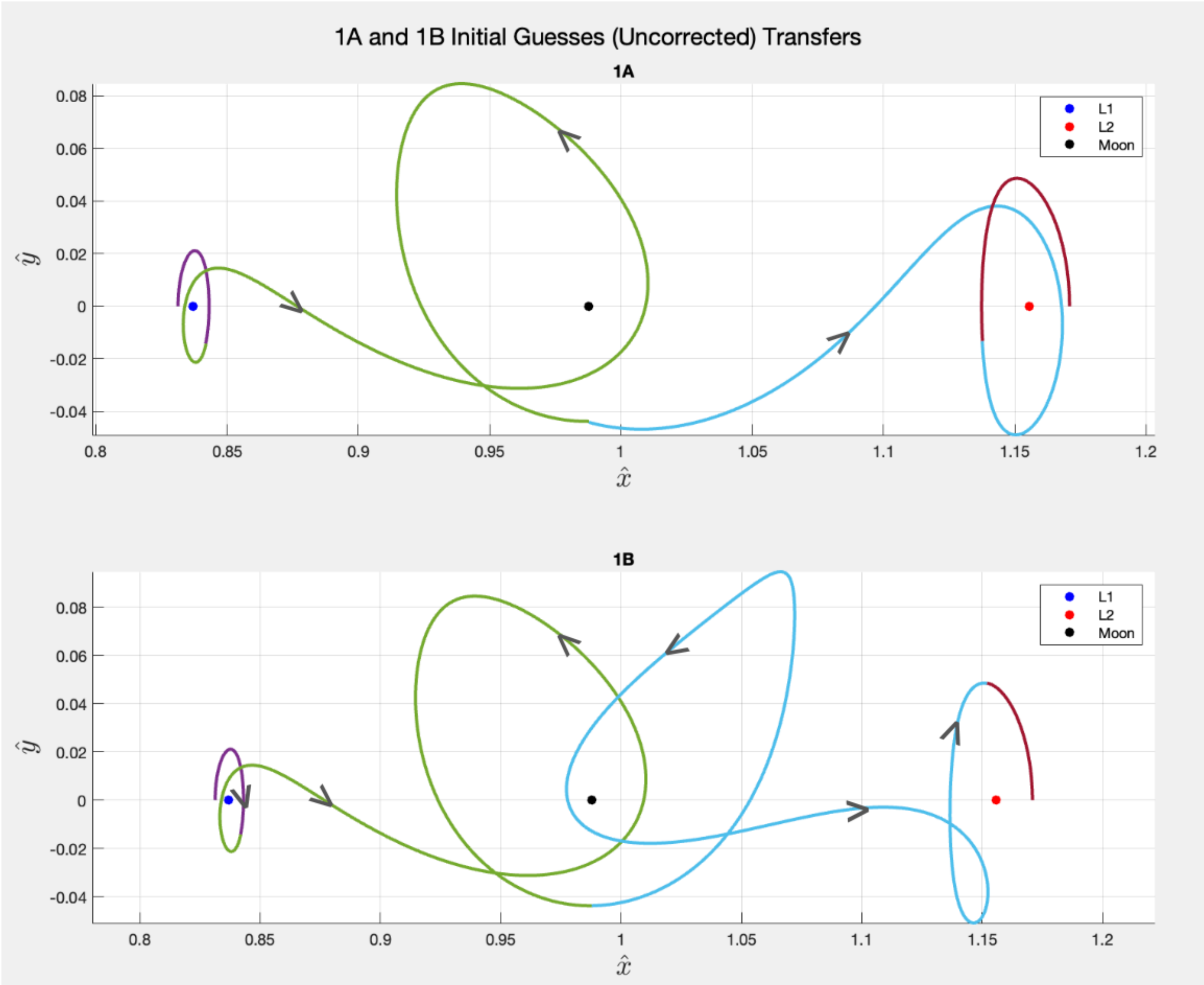
Arc 4 is given by the point that is used to calculate the stable manifold.

On the Poincaré map, points are chosen such that the y-velocity difference between them is minimal to reduce the ΔV required. The initial guesses using the above methodology are presented below:

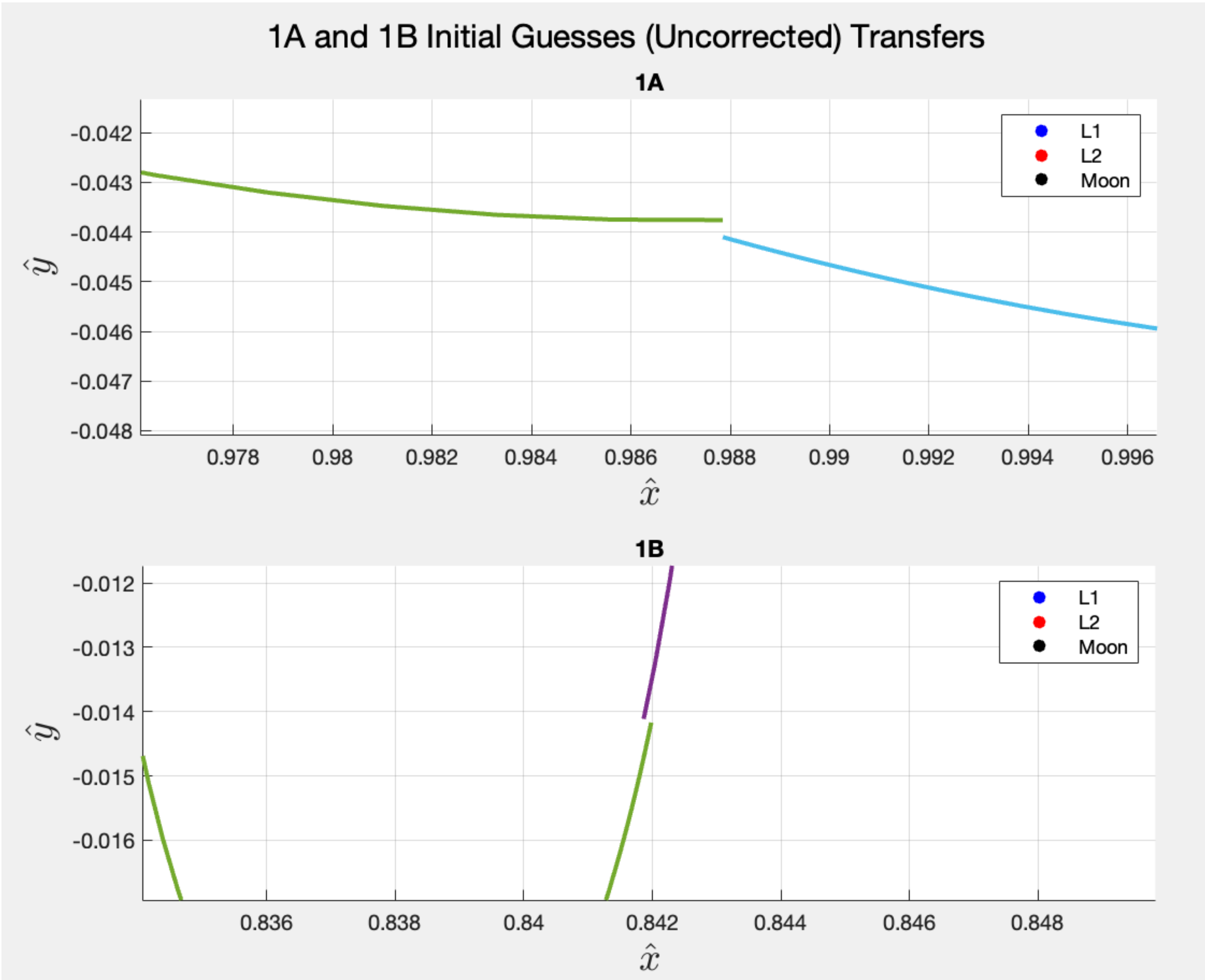
Initial Guesses (1A)	Arc #1	Arc #2	Arc #3	Arc #4
Initial State Vector	[0.83133061914502, 0.0, 0.0, 0.0, 0.04881731770896, 0.0]^T	[0.84198244217627, -0.01417021372350, 0.0, -0.00768086394308, -0.03710639328882, 0.0]^T	[0.98784941560529, -0.04410141472571, 0.0, 0.55905682176621, -0.15513846689630, 0.0]^T	[1.13773474237718, -0.01321881520923, 0.0, -0.00504204847613, 0.09034825289998, 0.0]^T
Flight Time	1.659824080408740	3.277504942680836	3.171173148313691	1.834993546267586
Flight Time [Days]	7.207752678560378	14.232499280155810	13.770755602188935	7.968422560175283

Initial Guesses (1B)	Arc #1	Arc #2	Arc #3	Arc #4
Initial State Vector	[0.83133061914502, 0.0, 0.0, 0.0, 0.04881731770896, 0.0]^T	[0.84198244217627, -0.01417021372350, 0.0, -0.00768086394308, -0.03710639328882, 0.0]^T	[0.98784941560529, -0.04375202050035, 0.0, 0.58388974130869, 0.00922785592719, 0.0]^T	[1.15205047884728, 0.04855878824120, 0.0, 0.03226365848907, -0.00614942014713, 0.0]^T
Flight Time	1.659824080408740	3.277504942680836	4.379508950241148	0.825846665593065
Flight Time [Days]	7.207752678560378	14.232499280155810	19.017929514015965	3.586222531813575

The initial guess transfer is plotted below.



The four arcs seem to be continuous, but that is deceptive because when zoomed in, the discontinuities can be visible. The image below is the same uncorrected trajectory as the image above, just zoomed in to highlight the discontinuities.



This formulation requires 3 impulsive maneuvers:

1. Transfer from L1 Lyapunov orbit to L1 unstable manifold.
    a. Needed because without a maneuver, the orbit would just be periodic.
2. Transfer from L1 unstable manifold to L2 stable manifold.
    a. Needed because without a maneuver, the orbit would continue to be on an L1 unstable manifold.
3. Transfer from L2 stable manifold to L2 Lyapunov orbit.
    a. Needed to insert the spacecraft into an L2 Lyapunov periodic orbit.

The placement of these maneuvers aligns with the unsmooth parts of the arcs where a maneuver is necessary to change the velocity to complete the transfer. This applies to Transfers 2(A&B) as well.

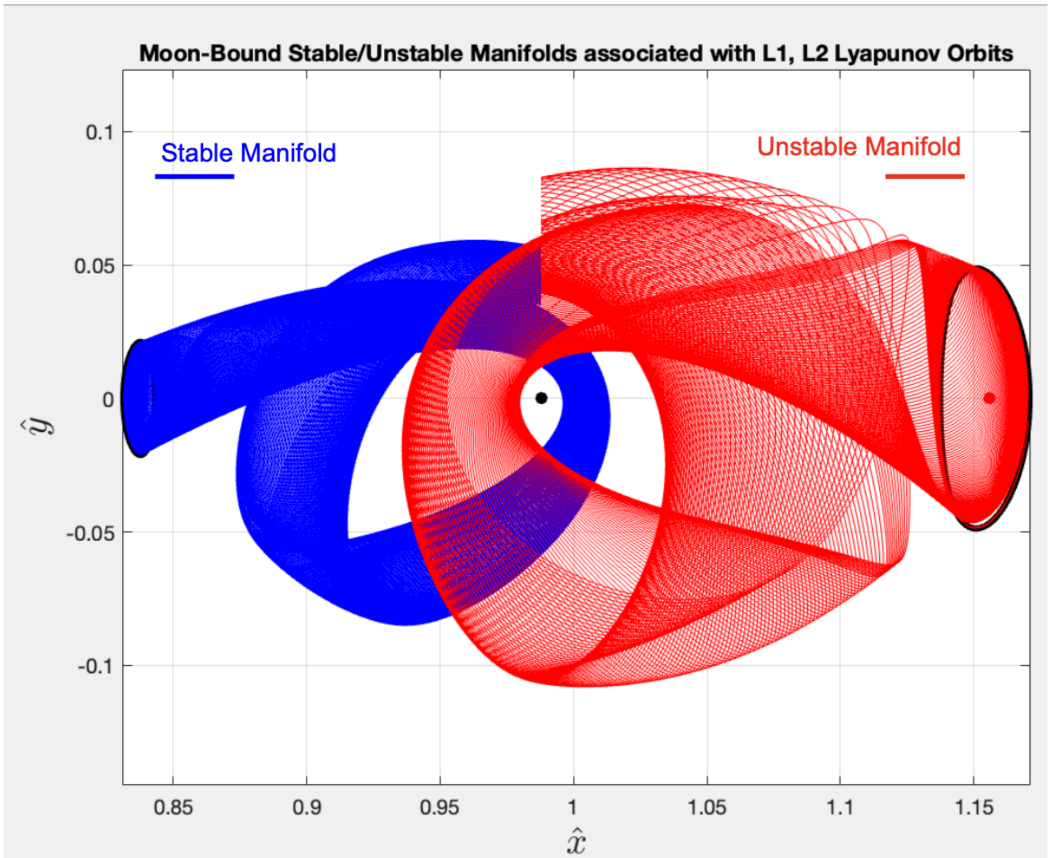
Transfers 2A & 2B

Let's discuss transfer 2A and 2B. These transfers are divided into 4 arcs:

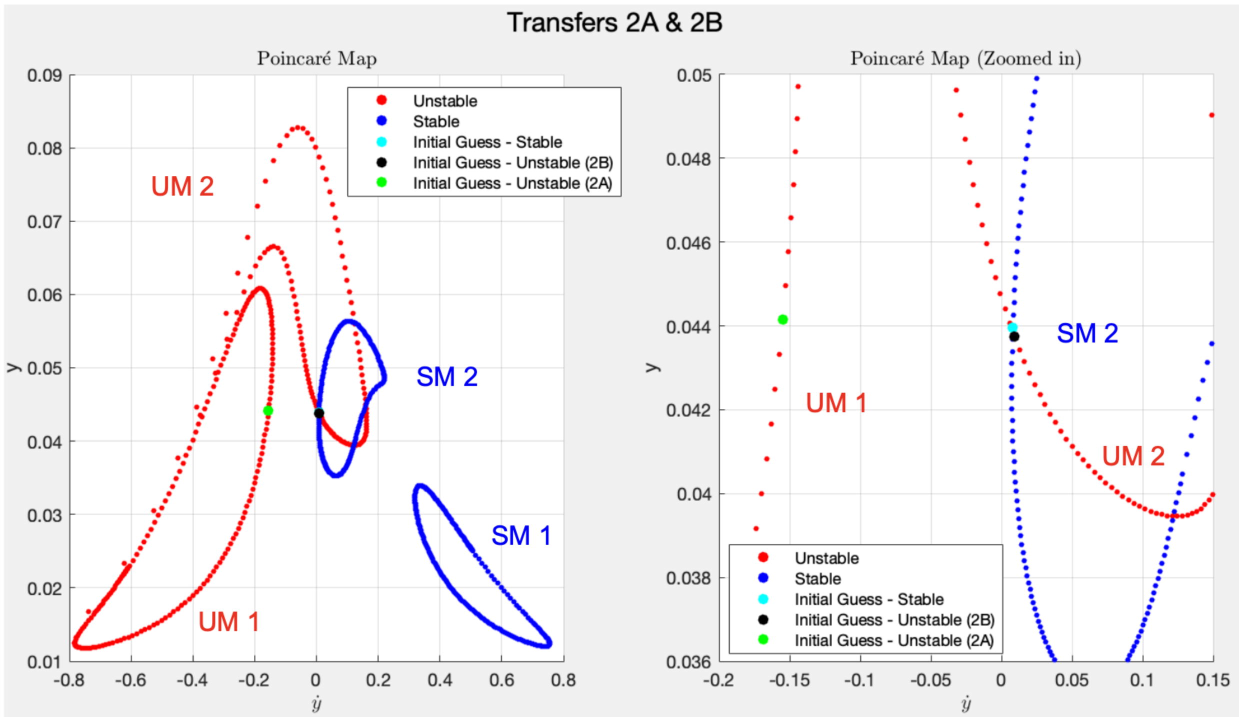
1. L2 Lyapunov orbit initial position to L2 Lyapunov orbit position that's closest to the L2

unstable manifold.
2. L2 unstable manifold to L1 stable manifold.
3. L1 stable manifold to point that's closest to the L1 Lyapunov orbit.
4. Point on the L1 Lyapunov orbit that's closest to the L1 stable manifold to the initial state vector given in section III.

The stable/unstable manifolds for 2(A&B) transfers are shown below. Since this transfer aims to go from L2 to L1, an unstable manifold from L2 and a stable manifold to L1 are required.



Then, the initial guesses can be constructed using the Poincaré map and the initial state vectors. Transfer 2A is chosen such that y positions are closest between UM 1 (unstable manifold from L2 with 1 crossing) and SM 2 (stable manifold from L1 with 2 crossings). The 2B transfer is chosen such that the SM 2 and UM 2 (unstable manifold from L2 with 2 crossings) are closest to each other. This means that 2A will have 1 crossing of the (1-μ) xz plane before transferring from the unstable to stable manifold and the stable manifold has 2 crossings of the (1-μ) xz plane which means that the transfer will have one orbit around the moon before reaching the L1 Lyapunov orbit. Similarly, 2B will have 2 crossings of the (1-μ) xz plane before transferring from the unstable to stable manifold and the stable manifold has 2 crossings of the (1-μ) xz plane which means that the transfer will have two orbits around the moon before reaching the L1 Lyapunov orbit.



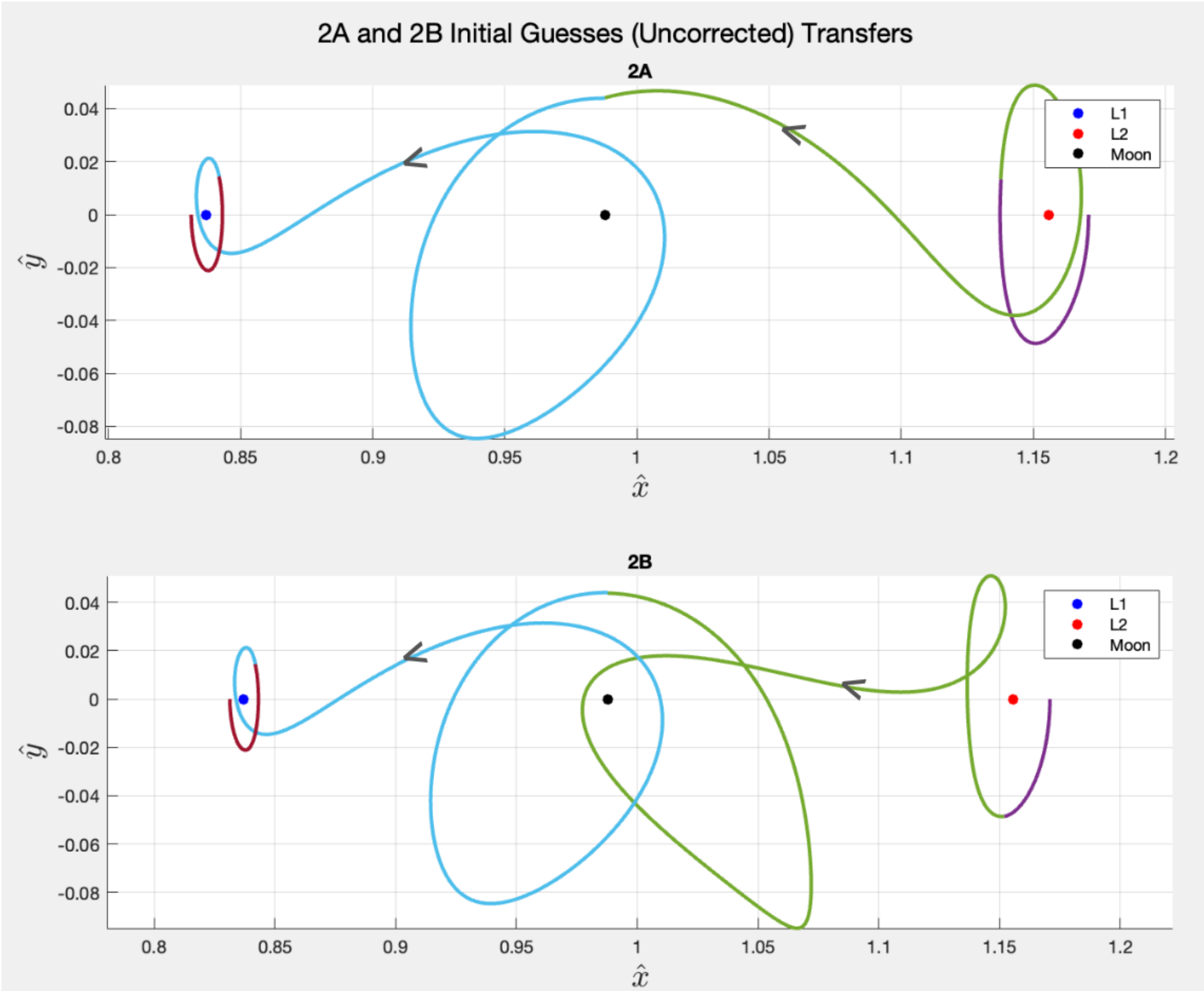
The initial guesses are then found from the manifolds using the y position and y velocities from the Poincaré maps. The initial guesses methodology is as follows:

1. Initial state for Arc 1 is simply the initial state vector for the L2 Lyapunov orbit.
2. 2 points from the Poincaré map are obtained - Unstable and Stable manifold points (UM & SM).
    a. Each manifold calculated from the Lyapunov orbit is done by perturbing from a discrete time step on the periodic orbit. Such discrete steps have a state vector and time associated with them. These are used to calculate the initial state vectors for the arcs.
3. Since, each point on the L2 Lyapunov orbit is used to calculate a manifold, the UM point also corresponds to a discrete time step on the Lyapunov Orbit. That time step is the flight time for Arc 1.
4. Initial state for Arc 2 is the initial state for the UM. The flight time for Arc 2 is how long the unstable manifold is integrated to meet the number of negative crossings needed (1 or 2).
5. Since Arc 3 is on a stable manifold which is integrated using negative time, the initial state for Arc 3 is the final state for the SM. The flight time for Arc 3 is how long the stable manifold is integrated to meet 2 negative crossings.
    a. When the stable manifold is computed, it is propagated in backwards time. So, the time has to be manually negated.
6. The initial state and flight time for Arc 4 is associated with the stable manifold on the periodic orbit.

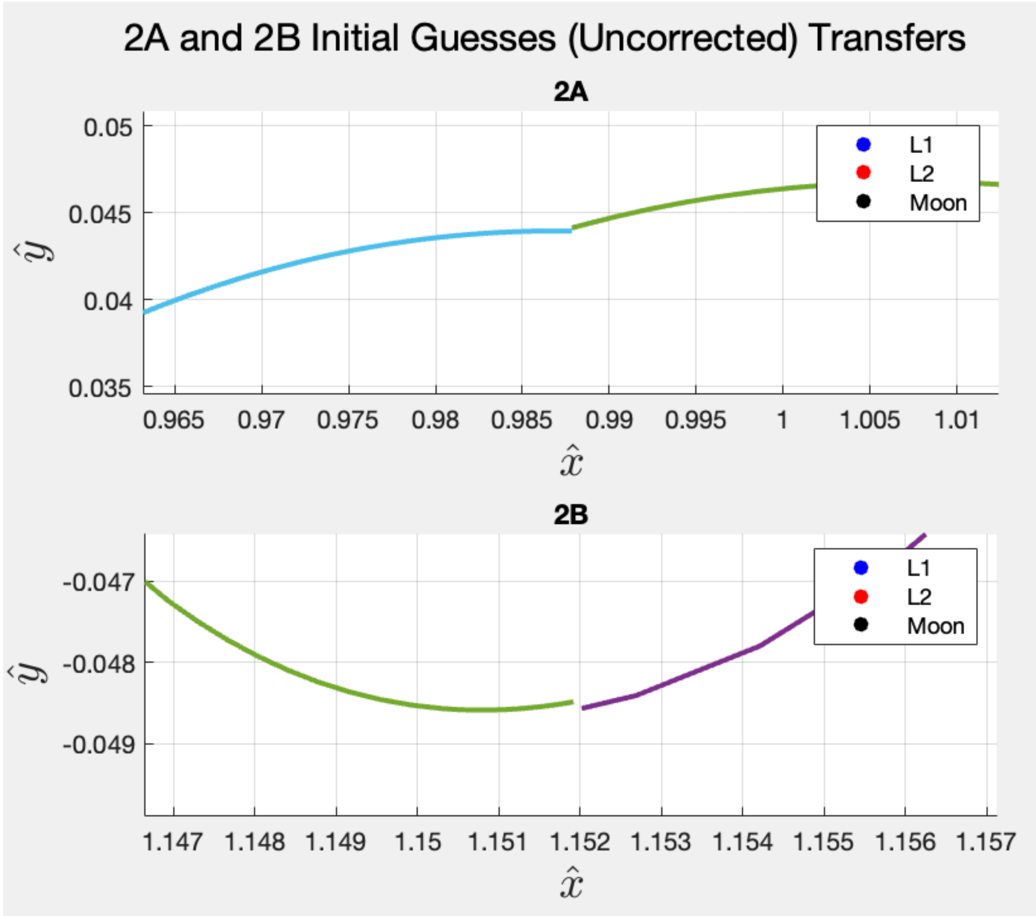
Initial Guesses (2A)	Arc #1	Arc #2	Arc #3	Arc #4
Initial State Vector	[1.170871819796487, 0.0, 0.0, -0.088163404081646, 0.0]^T	[1.137636542098959, 0.013382193862626, 0.0, 0.004821792624014, 0.090439654149302, 0.0]^T	[0.987849415605290, 0.043961784601220, 0.0, -0.563898270092335, 0.008164149911985, 0.0]^T	[0.84180164680475, 0.01439971347086, 0.0, 0.00821814113189, -0.03634592965672, 0.0]^T
Flight Time	1.835921394412072	3.171180556950323	3.279038108843100	1.667601880842165
Flight Time [Days]	7.972451722077230	13.770787774044468	14.239157023372799	7.241527620476825

Initial Guesses (2B)	Arc #1	Arc #2	Arc #3	Arc #4
Initial State Vector	[1.170871819796487, 0.0, 0.0, -0.088163404081646, 0.0]^T	[1.15191858390934, -0.04848366506154, 0.0, -0.03244927496284, -0.00579211488534, 0.0]^T	[0.98784941560529, 0.04396178460122, 0.0, -0.56389827009234, 0.00816414991198, 0.0]^T	[0.84180164680475, 0.01439971347086, 0.0, 0.00821814113189, -0.03634592965672, 0.0]^T
Flight Time	0.826774513737551	4.378959723601197	3.279038108843100	1.667601880842165
Flight Time [Days]	3.590251693715521	19.015544508380746	14.239157023372799	7.241527620476825

The initial guess transfer is plotted below:



The four arcs seem to be continuous, but that is deceptive because when zoomed in, the discontinuities can be visible. The image below is the same uncorrected trajectory as the image above, just zoomed in to highlight the discontinuities.



V. Multiple Shooting Formulation

The free variable vector ($\bar{V}$) is 28 x 1, and the constraint vector ($\bar{F}(\bar{V})$) is 12 x 1:

$$\underset{(28\times 1)}{\bar{V}} = \begin{array}{c} \left[\begin{array}{c} \vec{X}_{1,0} \\ \Delta t_1 \\ \vec{X}_{2,0} \\ \Delta t_2 \\ \vec{X}_{3,0} \\ \Delta t_3 \\ \vec{X}_{4,0} \\ \Delta t_4 \end{array}\right] \end{array} \begin{array}{l} \left.\right] \bar{V}_1 \\ \\ \left.\right] \bar{V}_2 \\ \\ \left.\right] \bar{V}_3 \\ \\ \left.\right] \bar{V}_4 \end{array},$$

$$\underset{(12\times 1)}{\bar{F}(\bar{V})} = \begin{bmatrix} \vec{\ell}_1 \\ \vec{\ell}_2 \\ \vec{\ell}_3 \\ \vec{r}_{4,f} - \vec{r}_{des,f} \end{bmatrix} = \begin{array}{c} \begin{bmatrix} \vec{r}_{1,f} - \vec{r}_{2,0} \\ \vec{r}_{2,f} - \vec{r}_{3,0} \\ \vec{r}_{3,f} - \vec{r}_{4,0} \\ \vec{r}_{4,f} - \vec{r}_{des,f} \end{bmatrix} \end{array} \begin{array}{l} - \bar{F}_1 \\ - \bar{F}_2 \\ - \bar{F}_3 \\ - \bar{F}_4 \end{array}$$

As seen above, the free variable vector is a collection of the initial states and flight times of each arc in the maneuver where the subscript (for the state vectors) x,0 signifies arc #x and initial state. Since, there are 4 arcs and each arc has a state vector along with flight time (7 elements), the free variable vector has a size of 28x1. The constraint vector is a collection of the position errors between arcs and the error between the final position and the desired position after the maneuver. The last constraint is added to ensure that the final state of the spacecraft ends up at the desired periodic orbit. Since, all these errors are position errors, they have 3 elements and for 4 arcs that makes the size 12x1.

This project assumes that an impulsive maneuver is used to maneuver the spacecraft between L1 and L2. This means that the spacecraft's position before and after the maneuver remains the same. So, the spacecraft can change its velocities as long as it's at the same position. There are no limits on the total change in velocity that the spacecraft can achieve.

Since this project assumes impulsive maneuvers, only the position vectors need to be constrained between the arcs. The maneuvers will meet any discrepancy in the velocities between on the edges of the arcs. The final desired position is also added to ensure that the maneuver accurately places the spacecraft at the desired final position. The Jacobian matrix has a size of mxn where m is the length of the constraint vector and n is the length of the free variable vector. Hence, the Jacobian matrix must have a size of 12x28. The Jacobian matrix can be computed as such:

$$\underset{(12\times 28)}{D\bar{F}(\bar{V})} = \begin{bmatrix} \partial(\vec{r}_{1,f}-\vec{r}_{2,0})/\partial\vec{x}_{1,0} & \partial(\vec{r}_{1,f}-\vec{r}_{2,0})/\partial\Delta t_1 & \partial(\vec{r}_{1,f}-\vec{r}_{2,0})/\partial\vec{x}_{2,0} & \cdots & \partial(\vec{r}_{1,f}-\vec{r}_{2,0})/\partial\Delta t_4 \\ \partial(\vec{r}_{2,f}-\vec{r}_{3,0})/\partial\vec{x}_{1,0} & \partial(\vec{r}_{2,f}-\vec{r}_{3,0})/\partial\Delta t_1 & \partial(\vec{r}_{2,f}-\vec{r}_{3,0})/\partial\vec{x}_{2,0} & \cdots & \partial(\vec{r}_{2,f}-\vec{r}_{3,0})/\partial\Delta t_4 \\ \partial(\vec{r}_{3,f}-\vec{r}_{4,0})/\partial\vec{x}_{1,0} & \partial(\vec{r}_{3,f}-\vec{r}_{4,0})/\partial\Delta t_1 & \partial(\vec{r}_{3,f}-\vec{r}_{4,0})/\partial\vec{x}_{2,0} & \cdots & \partial(\vec{r}_{3,f}-\vec{r}_{4,0})/\partial\Delta t_4 \\ \partial(\vec{r}_{4,f}-\vec{r}_{des,f})/\partial\vec{x}_{1,0} & \partial(\vec{r}_{4,f}-\vec{r}_{des,f})/\partial\Delta t_1 & \partial(\vec{r}_{4,f}-\vec{r}_{des,f})/\partial\vec{x}_{2,0} & \cdots & \partial(\vec{r}_{4,f}-\vec{r}_{des,f})/\partial\Delta t_4 \end{bmatrix}$$

$$= \begin{bmatrix} \Phi_1(t_{0,1},t_{f,1}) & \dot{\vec{r}}_{1,f} & -I_{3\times6} & 0_{3\times1} & 0_{3\times6} & 0_{3\times1} & 0_{3\times6} & 0_{3\times1} \\ 0_{3\times6} & 0_{3\times1} & \Phi_2(t_{0,2},t_{f,2}) & \dot{\vec{r}}_{2,f} & -I_{3\times6} & 0_{3\times1} & 0_{3\times6} & 0_{3\times1} \\ 0_{3\times6} & 0_{3\times1} & 0_{3\times6} & 0_{3\times1} & \Phi_3(t_{0,3},t_{f,3}) & \dot{\vec{r}}_{3,f} & -I_{3\times6} & 0_{3\times1} \\ 0_{3\times6} & 0_{3\times1} & 0_{3\times6} & 0_{3\times1} & 0_{3\times6} & 0_{3\times1} & \Phi_4(t_{0,4},t_{f,4}) & \dot{\vec{r}}_{4,f} \end{bmatrix}$$

(where I_{3x6} means an identity matrix with 3 rows and 6 columns, 0_{3x6} means a zero matrix with 3 rows and 6 columns, $\Phi_i(t_{0,i},\ t_{f,i})$ means the state transition matrix for arc i from the initial to final integration/flight time, $\dot{\bar{r}}_i$ means the velocity vector at the final point for arc i)

After these are calculated, the update to the free variable vector is made using this equation:

$$\bar{V}_{i+1} = \bar{V}_i - D\bar{F}(\bar{V}_i)^T[D\bar{F}(\bar{V}_i) * D\bar{F}(\bar{V}_i)^T]^{-1}\bar{F}(\bar{V}_i)$$

This is updated and the norm of the constraint vector is calculated at each time step. The stopping conditions are either:

- Norm of the constraint vector is less than 1e-10.

Or

- More than 50 iterations have occurred.

In Matlab, a while loop is used to perform the multiple shooting corrections scheme. The while loop's flow can be summarized as follows:

- The while loop checks if the norm of the current constraint vector is within the threshold or if the counter has exceeded the maximum possible value (50). If either of this is true, the while loop stops.
- The four arcs are propagated (until the current flight time) and the final states are recovered.
- All of these are fed into the constraint vector and its norm is saved.
- The DF matrix is then calculated.
- Then the next free variable vector is found using the following equation:

$$\bar{V}_{i+1} = \bar{V}_i - D\bar{F}(\bar{V})^T * (D\bar{F}(\bar{V}) * D\bar{F}(\bar{V})^T)^{-1} * \bar{F}$$

- The norm of the constraint vector is then calculated and stored and the counter iteration goes up by 1. Then the loop ends.

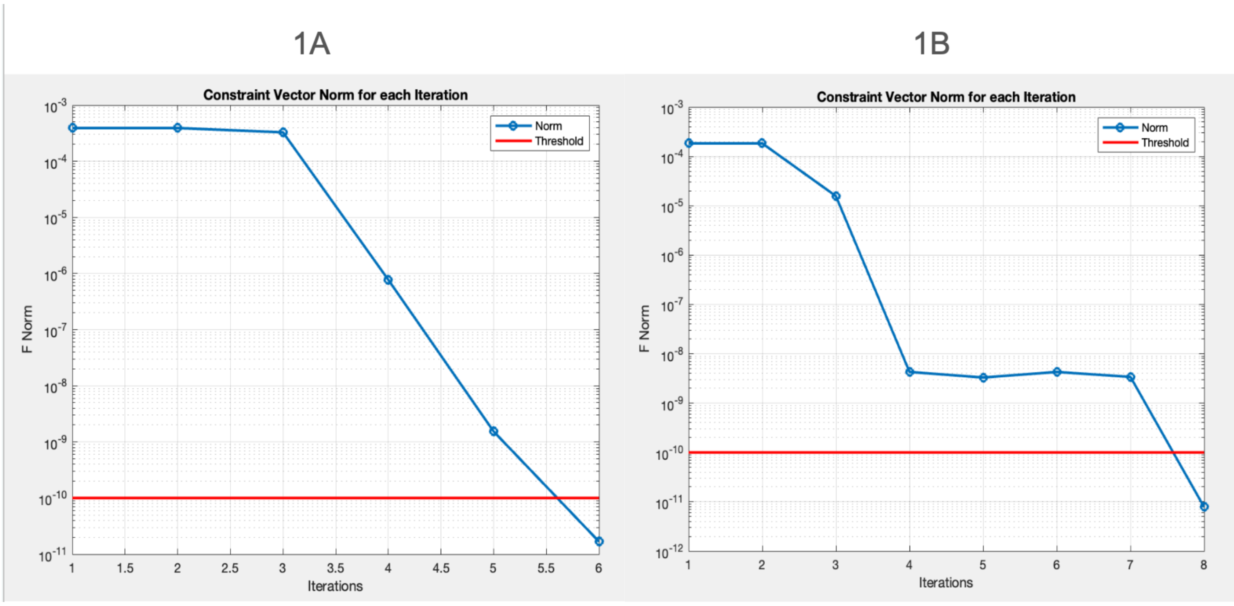
VI. Transfer Results and Analysis

Transfers 1A & 1B

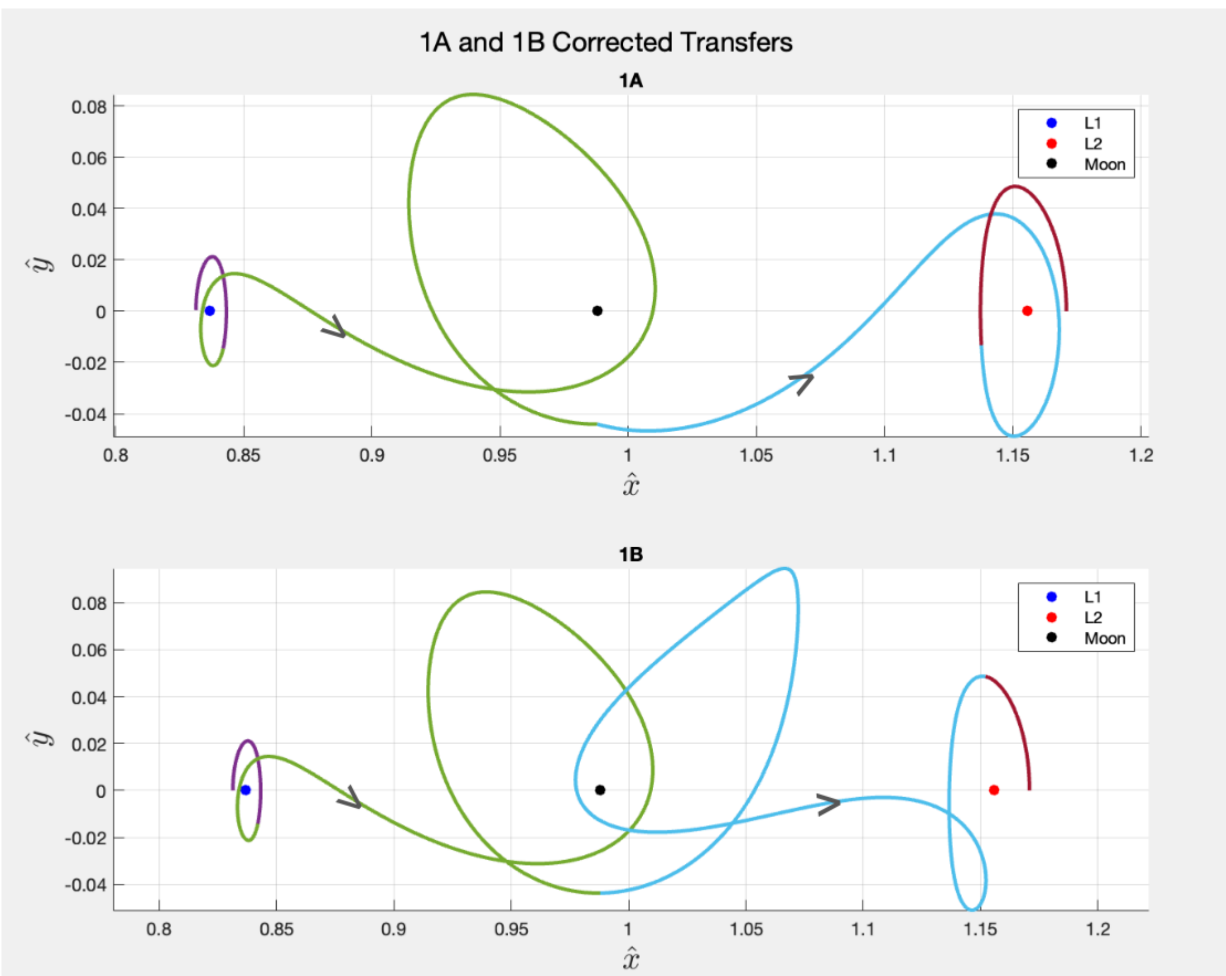
There are 3 evidence to the fact that the multiple shooting scheme works in correcting the position vectors of all 4 arcs such that all 4 arcs are one continuous maneuver and the initial and final orbits are Lyapunov L1, L2 orbits (respectively).

1. Constraint vector norm goes below a low threshold proving that the error in the position vectors after corrections reduces to an acceptable value.
2. The corrected trajectory looks continuous when the plots are zoomed in.
3. When the initial and final state vectors are propagated, Lyapunov orbits emerge with the same period as shown in Section III.

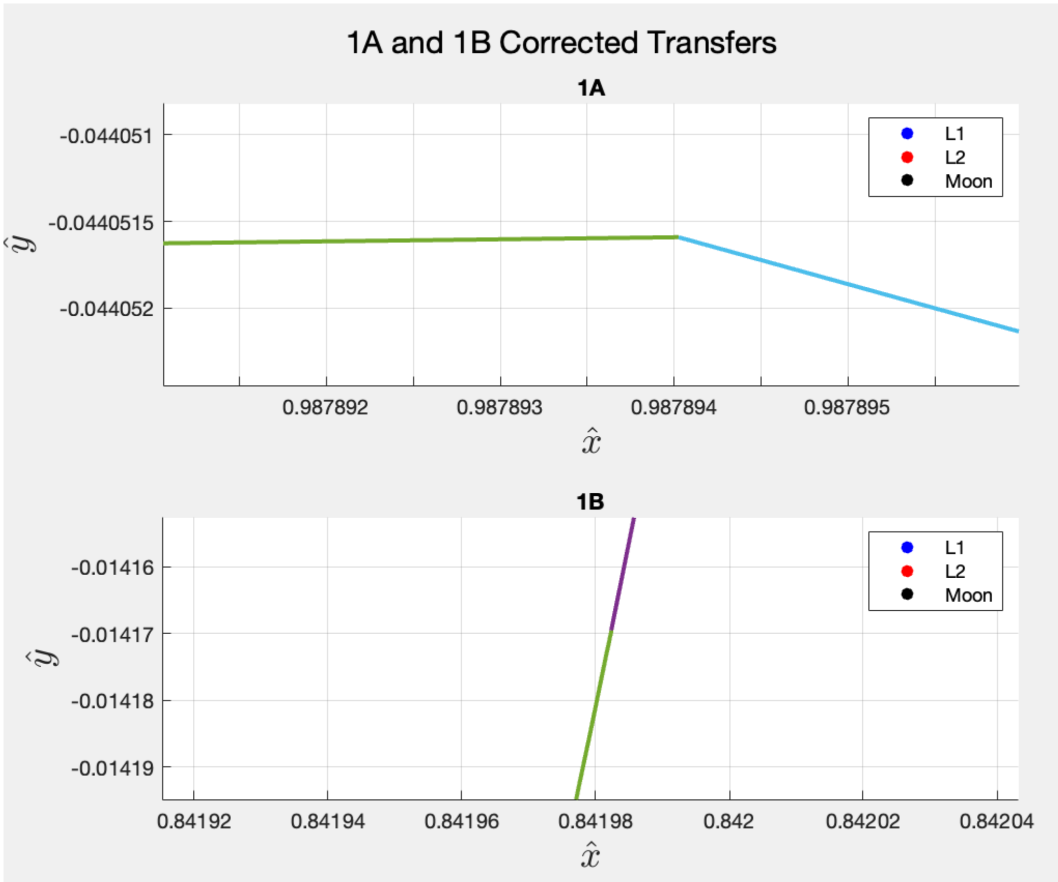
Here is a plot of the constraint vector norm as it goes through the multiple shooting corrections scheme:



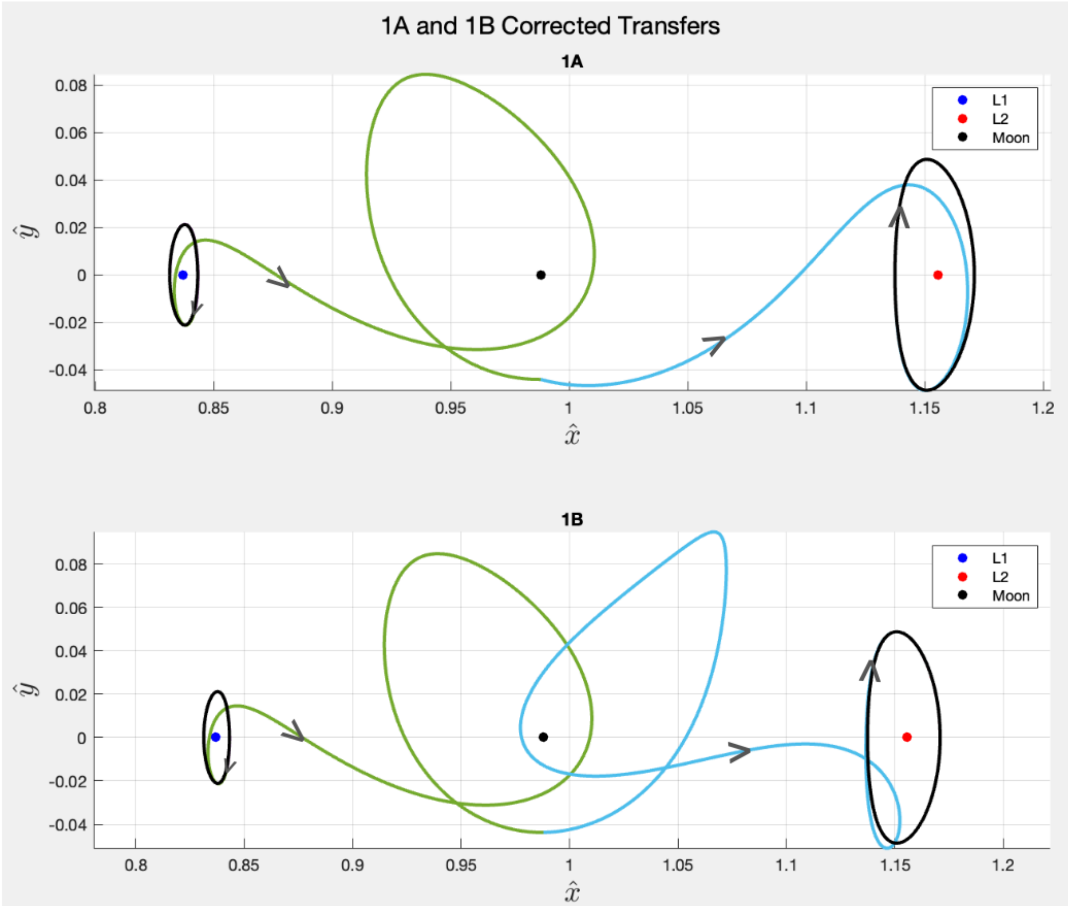
As seen, the norm of the constraint vector decreases and is close to its minimum tolerance (1e-10). Then, the plot of the continuous transfers are shown below:



As seen, the transfers retain very similar geometric properties as the initial uncorrected trajectories except for the fact that it is one continuous transfer now instead of 4 discontinuous arcs. Zooming in to actually confirm that the arcs are continuous (only a couple arcs are shown, but all the arcs are continuous):



Transfer 1B is more smooth when compared to 1A but transfer 1A has a shorter flight time. Lastly, propagating the initial and final states leads to Lyapunov periodic orbits:



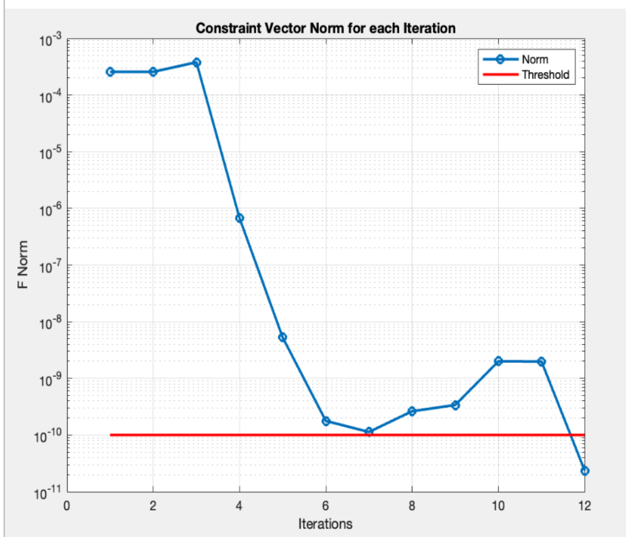
The total ΔV required for these maneuvers is as follows along with the flight times:

	1A	1B
ΔV_{tot} [km/s]	0.167961452690682	0.018862497786300
Flight time [-]	9.943640007134274	10.142683475377211
Flight time [days]	43.180056695172645	44.044398951868118

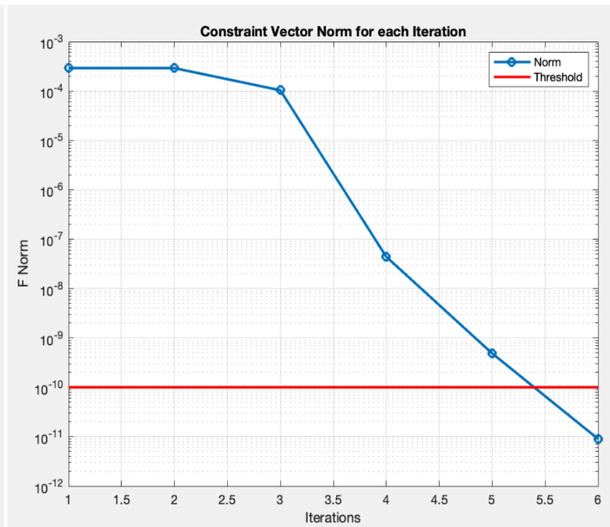
Transfers 2A & 2B

Similar to 1(A&B), the same three pieces of evidence will be presented to verify that the multiple shooting scheme has been implemented successfully and an L2 to L1 transfer is achieved. Here is a plot of the constraint vector norm as it goes through the multiple shooting corrections scheme:

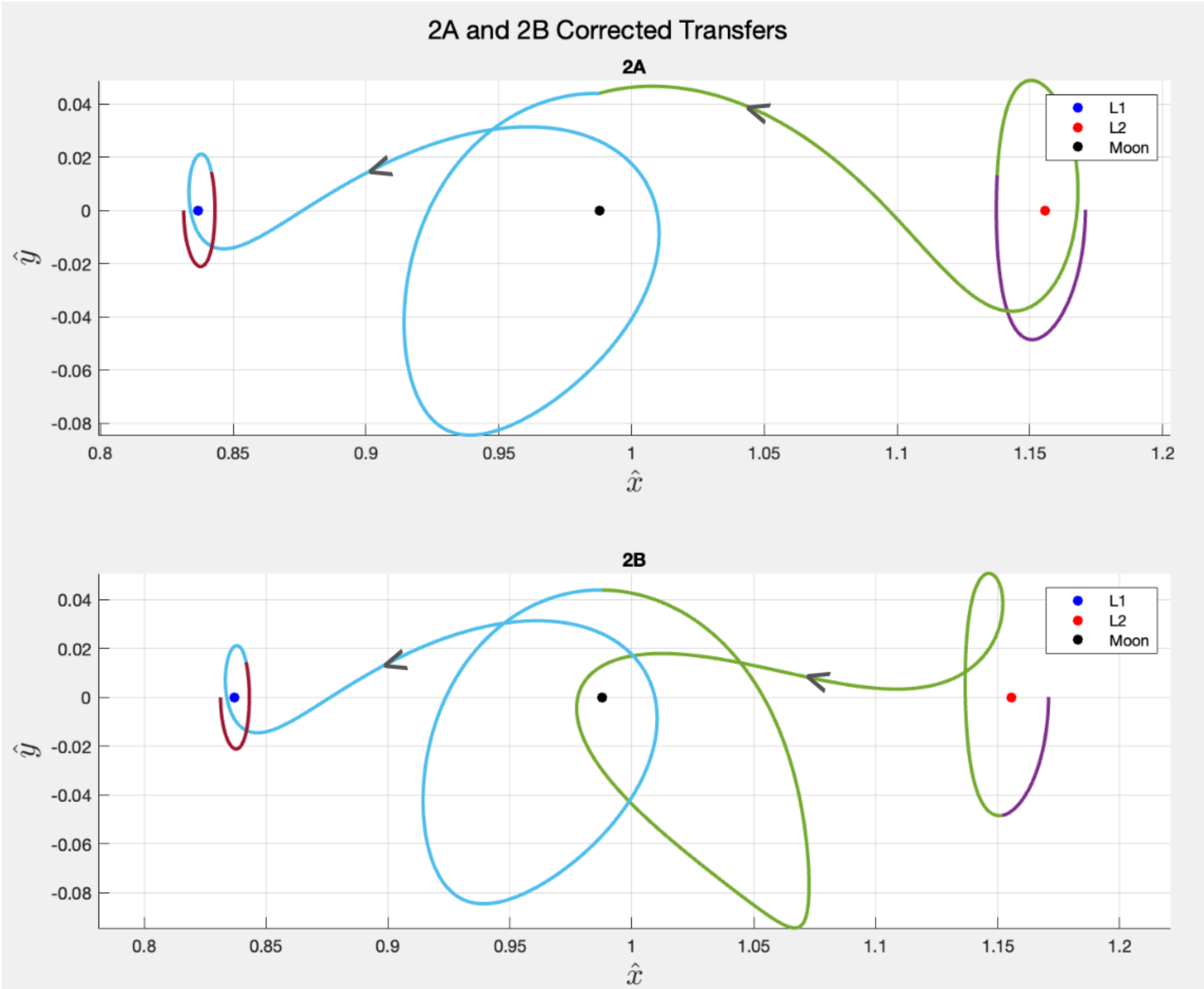
2A



2B

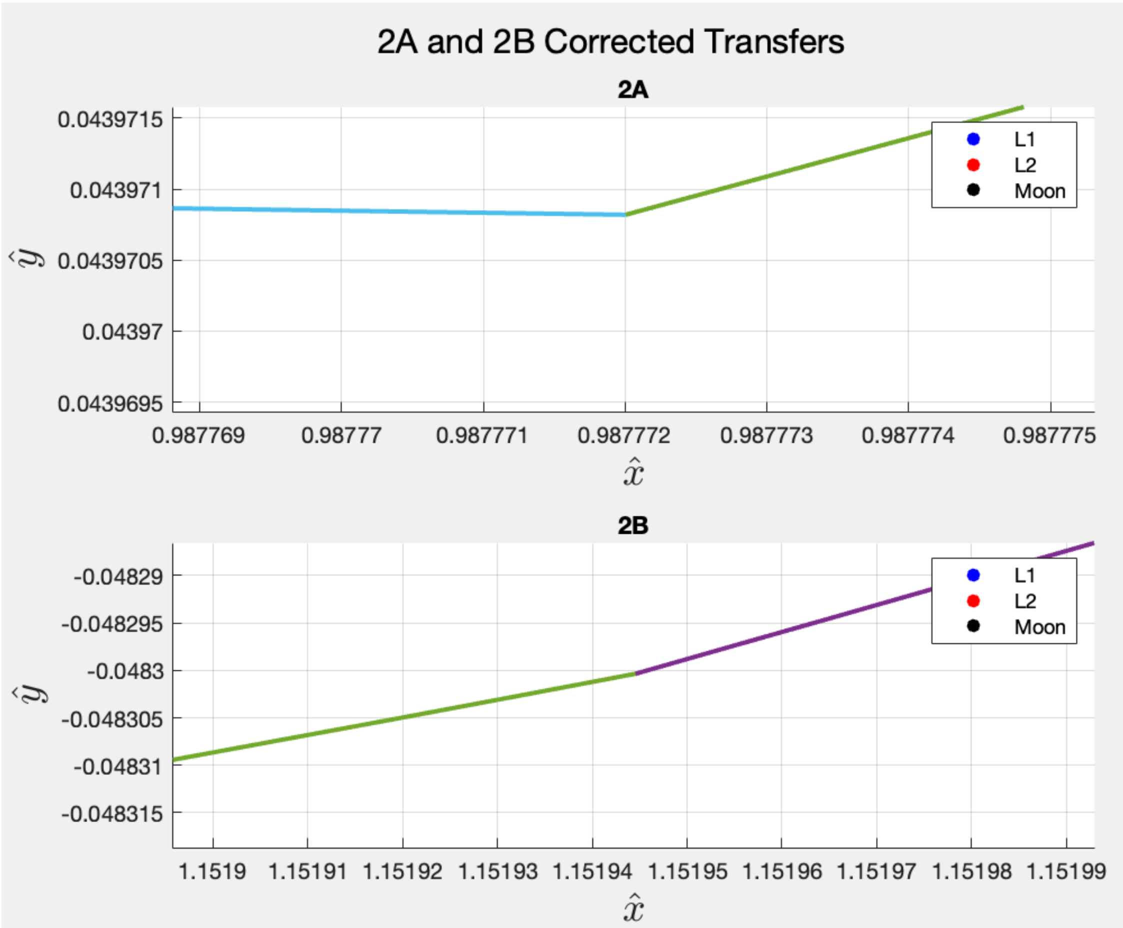


As seen, the norm of the constraint vector decreases and is below 1e-10. Then, the plot of the continuous transfers are shown below:

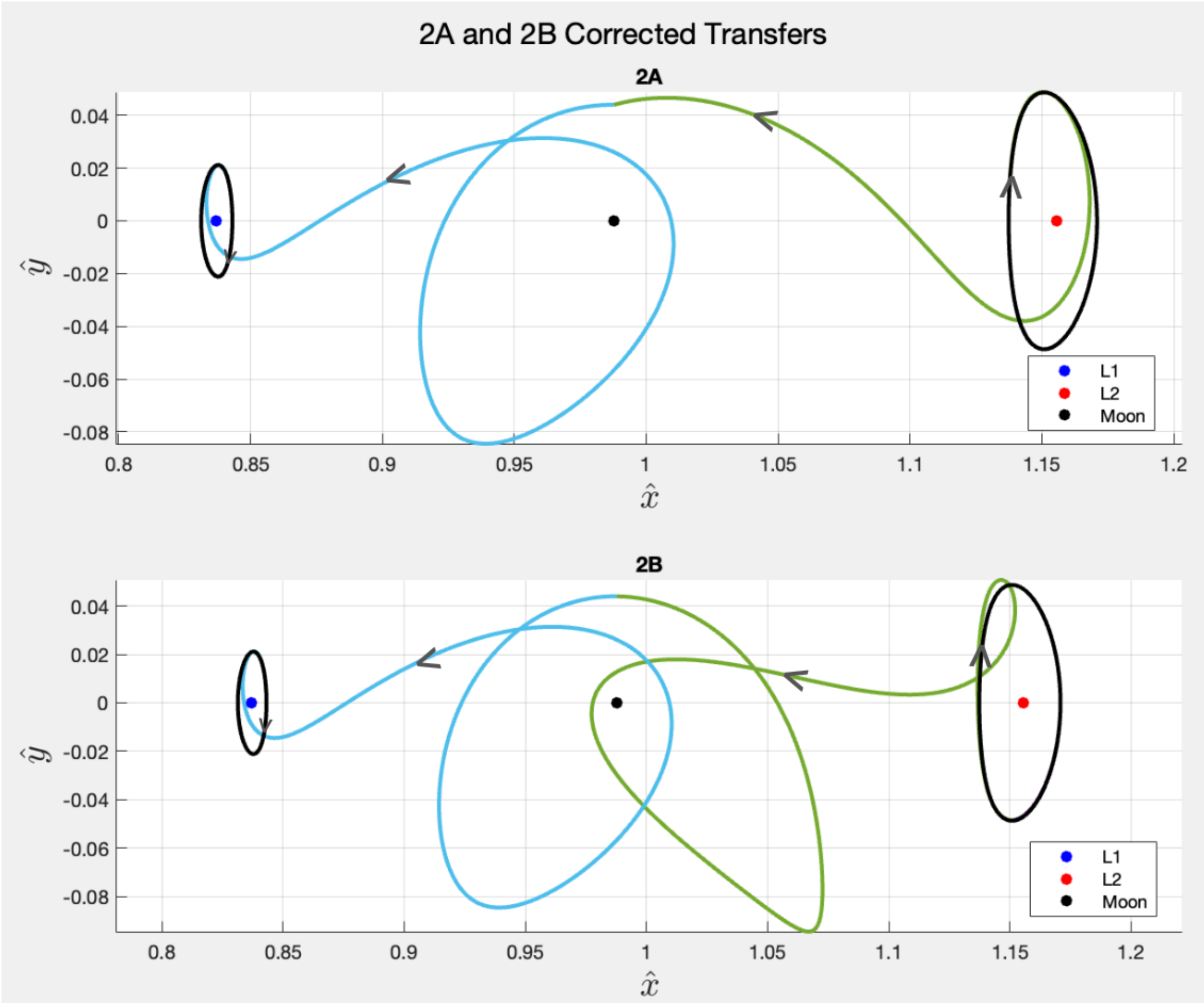


As seen, the transfers retain very similar geometric properties to the uncorrected transfer except for the fact that it is one continuous transfer now instead of 4 discontinuous arcs.

Zooming in to actually confirm that the arcs are continuous (only a couple arcs are shown, but all the arcs are continuous):



Transfer 1B is more smooth when compared to 1A but transfer 1A has a shorter flight time. Lastly, propagating the initial and final states leads to Lyapunov periodic orbits:



The total ΔV required for these maneuvers is as follows along with the flight times:

	2A	2B
ΔV_{tot} [km/s]	0.156791754226861	0.021373320991330
Flight Time [-]	9.953653509863834	10.152307351170817
Flight time [days]	43.223540129334516	44.086190438898726

Comparing the Transfers

It can be seen that 1B is the cheapest maneuver (when comparing ΔV) and 1A is the most expensive maneuver. When looking at the flight times, 1A is the fastest and 2B is slowest. To draw a conclusion, it is fair to assume that L1 → L2 transfers take very similar amounts of time and maneuvering capabilities as L2 → L1 transfers. Both these transfers spend different amounts of time at different regions of the moon. So, if science or other objectives need to be met, the transfers could be distinguished. Otherwise, this project makes a fairly good case that these transfers are reasonably similar.

VII. Recommendations for Future Analysis

For future analysis, it could be really interesting to try Poincaré maps with higher number of crossings and see if even smaller maneuvers can be used for these transfers or if other geometries could be unlocked. Additionally, other L1/L2 periodic orbits (such as Halo or Axial) could also be tested to see if they produce transfers with lower total ΔV. Additionally, maneuvers with more arcs could also be tested for better maneuver/corrections performance.

While the CR3BP is a really good first approximation of the dynamical system, it has its limitations. This project is a very good first order model of how a spacecraft can travel between L1 and L2 Lyapunov orbits. However, in order to do a deeper dive into this, higher fidelity models such as an Elliptic Restricted 3 Body Problem (ER3BP), Bicircular 4 Body Problem (B4BP), etc. can be used. Additionally, ephemeris based simulations such as ones available in STK/GMAT can be used to perform high fidelity simulations.

Table of Contents

.... 1
Constants 1
1A & 1B Orbits 2
Manifolds 3
Poincare Map 4
Multiple Shooting Initial Guess - 1 5
Delta V Calc 6
Uncorrected Trajectory 6
Corrected Trajectory 7
Multiple Shooting Initial Guess - 2 8
Delta V Calc 8
Uncorrected Trajectory 9
Corrected Trajectory 9
Functions 10
Constants 16
2A & 2B Orbits 17
Manifolds 18
Poincare Map 19
Multiple Shooting Initial Guess - 1 20
Delta V Calc 21
Uncorrected Trajectory 21
Corrected Trajectory 22
Multiple Shooting Initial Guess - 2 22
Uncorrected Trajectory 23
Corrected Trajectory 24


```
clear; clc; close all;

% ASEN 6060 - Final Project
% 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];

% 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];

TOL = 5e-14;

% Set options for ode113
options = odeset('RelTol', TOL, 'AbsTol', TOL);

global count poincare_stored iteration
```

1A & 1B Orbits

```
% L1 orbit is a Lyapunov orbit
load("V_family_L1_Lyapunov.mat")
V_family_L1_Lyapunov = V_family;
l1_orbit_idx = 50;

% L2 orbit is a Lyapunov orbit
load("V_family_L2_Lyapunov.mat")
V_family_L2_Lyapunov = V_family_a1;
l2_orbit_idx = 10;

figure(1)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

[tout_L1, xout_L1] = ode113(@(t,state)CR3BP_full(state,
mu), [0, V_family_L1_Lyapunov(7,l1_orbit_idx)],
[V_family_L1_Lyapunov(1:6,l1_orbit_idx); reshape(eye(6), [36,1])], options);
plot(xout_L1(:,1), xout_L1(:,2), 'LineWidth',2)

[tout_L2, xout_L2] = ode113(@(t,state)CR3BP_full(state,
mu), [0, V_family_L2_Lyapunov(7,l2_orbit_idx)],
[V_family_L2_Lyapunov(1:6,l2_orbit_idx); reshape(eye(6), [36,1])], options);
plot(xout_L2(:,1), xout_L2(:,2), 'LineWidth',2)
```

```
cj_L1 = jacobi_constant(V_family_L1_Lyapunov(:,l1_orbit_idx), mu);
cj_L2 = jacobi_constant(V_family_L2_Lyapunov(:,l2_orbit_idx), mu);
delta_cj = cj_L2 - cj_L1;

hold off
legend("L1", "L2", "Moon", "L1 Lyapunov Orbit", "L2 Lyapunov Orbit")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("L1 and L2 Lyapunov Orbits (\DeltaC_j = " + abs(delta_cj) + ")")
```

Manifolds

```
n_crossings = 2;
L1_manifold_time = 10;
L2_manifold_time = 6;

L1_manifolds = unstable_manifolds(tout_L1, xout_L1, mu, l1_pos,
L1_manifold_time, n_crossings);
L2_manifolds = stable_manifolds(tout_L2, xout_L2, mu, l2_pos,
L2_manifold_time, n_crossings);

poincare_stored = [];
stable_options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
stable_event(t, y, mu, n_crossings));
options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
options_event = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
eventFn(t, y, mu));
unstable_options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
unstable_event(t, y, mu, n_crossings));

figure(2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 3)
hold on
plot(xout_L2(:,1), xout_L2(:,2), 'black', 'LineWidth', 3)
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', ' black')

iteration = 1;

for k = 1:length(L1_manifolds)
    count = 0;
    [moon_unstable_t, moon_unstable_x] = ode113(@(t, state)CR3BP(state, mu),
[0, L1_manifold_time], L1_manifolds(:,k), unstable_options);
    unstable_end_times(k) = moon_unstable_t(end);
    unstable_end_states(k, :) = moon_unstable_x(end, :);
    plot(moon_unstable_x(:,1), moon_unstable_x(:,2), 'red')
    iteration = iteration + 1;
end
grid on
axis equal
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
```

```
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)

poincare_unstable = poincare_stored;

poincare_stored = [];

iteration = 1;

for k = 1:length(L2_manifolds)
    count = 0;
    [moon_stable_t, moon_stable_x] = ode113(@(t, state)CR3BP(state, mu), [0,
-L2_manifold_time], L2_manifolds(:,k), stable_options);
    stable_end_times(k) = moon_stable_t(end);
    stable_end_states(k, :) = moon_stable_x(end, :);
    if (abs(stable_end_states(k, 1) - (1-mu)) < 1e-6 && stable_end_states(k,
2) < 0)
        plot(moon_stable_x(:,1), moon_stable_x(:,2), 'blue')
    end
    iteration = iteration + 1;
end
poincare_stable = poincare_stored;

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

Poincare Map

```
figure(3)
subplot(1,2,1)
scatter(poincare_unstable(:,2), poincare_unstable(:,1), 10, 'filled', 'red');
hold on
scatter(poincare_stable(:,2), poincare_stable(:,1), 10, 'filled', 'blue');
xlabel("$\dot{y}$", 'Interpreter','latex')
ylabel("y")
title("Poincar\'e Map", 'Interpreter','latex')
grid on

stable_y_value = -0.043753;
stable_ydot_value = 0.00922752;
[pc_stable_min, pc_stable_min_ind] = min(abs(poincare_stable(:,1) -
stable_y_value));
stable_ig = poincare_stable(pc_stable_min_ind, 3);


unstable_y_value = -0.0437569;
unstable_ydot_value = 0.00797096;
[pc_unstable_min, pc_unstable_min_ind] = min(abs(poincare_unstable(:,1) -
unstable_y_value));
unstable_ig = poincare_unstable(pc_unstable_min_ind, 3);

stable_y_value_2 = -0.0441014;
stable_ydot_value_2 = -0.155138;
```

```
[pc_stable_min_2, pc_stable_min_ind_2] = min(abs(poincare_stable(:,1) -
stable_y_value_2));
stable_ig_2 = poincare_stable(pc_stable_min_ind_2, 3);

scatter(poincare_stable(pc_stable_min_ind_2, 2),
poincare_stable(pc_stable_min_ind_2, 1), 'filled', 'green')
scatter(poincare_stable(pc_stable_min_ind, 2),
poincare_stable(pc_stable_min_ind, 1), 'filled', 'cyan')
scatter(poincare_unstable(pc_unstable_min_ind, 2),
poincare_unstable(pc_unstable_min_ind, 1), 'filled', 'black')
legend("Unstable", "Stable", "Initial Guess - Stable (1A)", "Initial Guess -
Stable (1B)", "Initial Guess - Unstable")
hold off

subplot(1,2,2)
scatter(poincare_unstable(:,2), poincare_unstable(:,1), 10, 'filled', 'red');
hold on
scatter(poincare_stable(:,2), poincare_stable(:,1), 10, 'filled', 'blue');
xlabel("$\dot{y}$", 'Interpreter','latex')
ylabel("y")
title("Poincar\'e Map (Zoomed in)", 'Interpreter','latex')
grid on
scatter(poincare_stable(pc_stable_min_ind_2, 2),
poincare_stable(pc_stable_min_ind_2, 1), 'filled', 'green')
scatter(poincare_stable(pc_stable_min_ind, 2),
poincare_stable(pc_stable_min_ind, 1), 'filled', 'cyan')
scatter(poincare_unstable(pc_unstable_min_ind, 2),
poincare_unstable(pc_unstable_min_ind, 1), 'filled', 'black')
legend("Unstable", "Stable", "Initial Guess - Stable (1A)", "Initial Guess -
Stable (1B)", "Initial Guess - Unstable")
hold off
xlim([-0.25, 0.05])
ylim([-0.05, -0.04])

sgtitle("Transfers 1A & 1B")
```

Multiple Shooting Initial Guess - 1

```
% Initial guess
x_1_0 = xout_L1(1, 1:6)';
delta_T1 = tout_L1(unstable_ig);
[tout, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T1], x_1_0);
x_1_f = xout_L1(unstable_ig, 1:6)';

x_2_0 = L1_manifolds(:, unstable_ig);
delta_T2 = unstable_end_times(unstable_ig);
[tout, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T2], x_2_0);
x_2_f = unstable_end_states(unstable_ig, :)';

x_3_f = L2_manifolds(:, stable_ig);
delta_T3 = -stable_end_times(stable_ig);
x_3_0 = stable_end_states(stable_ig, :)';
```

```
x_4_0 = xout_L2(stable_ig, 1:6)';
delta_T4 = V_family_L2_Lyapunov(7, l2_orbit_idx) - tout_L2(stable_ig);
[tout, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T4], x_4_0);
x_4_f = x_4(end, 1:6)';

r_des_4 = xout_L2(1, 1:3)';

V0 = [x_1_0; delta_T1; x_2_0; delta_T2; x_3_0; delta_T3; x_4_0; delta_T4];

e1 = x_1_f - x_1_0;
e2 = x_2_f - x_2_0;
e3 = x_3_f - x_3_0;

V_soln = multiple_shooting(V0, mu, true, r_des_4);

flight_time_1B = V_soln(7) + V_soln(14) + V_soln(21) + V_soln(28);
```

Delta V Calc

```
v_1_f = x_1_f(4:6);
v_2_0 = x_2_0(4:6);
v_2_f = x_2_f(4:6);
v_3_0 = x_3_0(4:6);
v_3_f = x_3_f(4:6);
v_4_0 = x_4_0(4:6);

function out = calc_dv(v1, v2)
    out = sqrt((v1(1) - v2(1))^2 + (v1(2) - v2(2))^2 + (v1(3) - v2(3))^2);
end

dv1 = calc_dv(v_1_f, v_2_0);
dv2 = calc_dv(v_2_f, v_3_0);
dv3 = calc_dv(v_3_f, v_4_0);

dv_tot_1B = dv1 + dv2 + dv3;
dv_tot_dim_1B = dv_tot_1B * l_star_em / t_star_em;

% x_1_0 = xout_L1(unstable_ig, 1:6)';
% ig_um = L1_manifolds(:, unstable_ig);
% ig_sm = L2_manifolds(:, stable_ig);
% ig_L2_PO = xout_L2(stable_ig, 1:6)';
```

Uncorrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V0(7)], V0(1:6), options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V0(14)], V0(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V0(21)], V0(15:20),
options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V0(28)], V0(22:27),
options);

figure(5)
```

```
subplot(2, 1, 2)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 2)
plot(xout_L2(:,1), xout_L2(:,2), 'black',  'LineWidth', 2)

legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("1B")

sgtitle("1A and 1B Initial Guesses (Uncorrected) Transfers")
```

Corrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(7)], V_soln(1:6),
options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(14)], V_soln(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(21)],
V_soln(15:20), options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(28)],
V_soln(22:27), options);

x1f = x_1(end,:);

figure(6)
subplot(2,1,2)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 2)
plot(xout_L2(:,1), xout_L2(:,2), 'black',  'LineWidth', 2)

hold off
legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
```

```
grid on
axis equal
title("1B")

sgtitle("1A and 1B Corrected Transfers")
```

Multiple Shooting Initial Guess - 2

```
% Initial guess
x_1_0 = xout_L1(1, 1:6)';
delta_T1 = tout_L1(unstable_ig);
[tout, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T1], x_1_0);
x_1_f = xout_L1(unstable_ig, 1:6)';

x_2_0 = L1_manifolds(:, unstable_ig);
delta_T2 = unstable_end_times(unstable_ig);
[tout, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T2], x_2_0);
x_2_f = unstable_end_states(unstable_ig, :)';

x_3_f = L2_manifolds(:, stable_ig_2);
[tout, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, -L2_manifold_time],
x_3_f, options_event);
% delta_T3 = -stable_end_times(stable_ig_2);
delta_T3 = -tout(end);
x_3_0 = x_3(end,:)';
% x_3_0 = stable_end_states(stable_ig_2, :)';


x_4_0 = xout_L2(stable_ig_2, 1:6)';
delta_T4 = V_family_L2_Lyapunov(7, l2_orbit_idx) - tout_L2(stable_ig_2);
[tout, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T4], x_4_0);
x_4_f = x_4(end, 1:6)';

r_des_4 = xout_L2(1, 1:3)';

V0 = [x_1_0; delta_T1; x_2_0; delta_T2; x_3_0; delta_T3; x_4_0; delta_T4];

e1 = x_1_f - x_1_0;
e2 = x_2_f - x_2_0;
e3 = x_3_f - x_3_0;

V_soln = multiple_shooting(V0, mu, true, r_des_4);

flight_time_1A = V_soln(7) + V_soln(14) + V_soln(21) + V_soln(28);
```

Delta V Calc

```
v_1_f = x_1_f(4:6);
v_2_0 = x_2_0(4:6);
v_2_f = x_2_f(4:6);
v_3_0 = x_3_0(4:6);
v_3_f = x_3_f(4:6);
v_4_0 = x_4_0(4:6);
```

```
dv1 = calc_dv(v_1_f, v_2_0);
dv2 = calc_dv(v_2_f, v_3_0);
dv3 = calc_dv(v_3_f, v_4_0);

dv_tot_1A = dv1 + dv2 + dv3;
dv_tot_dim_1A = dv_tot_1A * l_star_em / t_star_em;


% x_1_0 = xout_L1(unstable_ig, 1:6)';
% ig_um = L1_manifolds(:, unstable_ig);
% ig_sm = L2_manifolds(:, stable_ig);
% ig_L2_PO = xout_L2(stable_ig, 1:6)';
```

Uncorrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V0(7)], V0(1:6), options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V0(14)], V0(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V0(21)], V0(15:20),
options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V0(28)], V0(22:27),
options);

figure(5)
subplot(2, 1, 1)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)

hold off
legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("1A")
```

Corrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(7)], V_soln(1:6),
options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(14)], V_soln(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(21)],
V_soln(15:20), options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(28)],
```

```
V_soln(22:27), options);

x1f = x_1(end,:);
figure(6)
subplot(2,1,1)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 2)
plot(xout_L2(:,1), xout_L2(:,2), 'black',  'LineWidth', 2)

hold off
legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("1A")
```

Functions

```
function moon_bound_stable = stable_manifolds(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_stable_cnt = 0;
```

```
% Begin for loop
for i = 1:length(tout)
    print_out = sprintf('Discrete Time Step  - %d', i);
    disp(print_out)

    % 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
    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;

    % ONLY FOR L1
    % If x-velocity is positive, moon-bound
    % If x-velocity if negative, earth-bound
    x_manifold_s_p = xout(i,1:6)' + d * stable_evec;
    x_manifold_s_n = xout(i,1:6)' - d * stable_evec;
    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_unstable_x(:,1), moon_unstable_x(:,2), 'red')
        % plot(earth_unstable_x(:,1), earth_unstable_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;
        elseif abs(earth_stable_x(end,1) - (1-mu)) < 1e-6
            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_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 moon_bound_stable = unstable_manifolds(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_unstable_cnt = 0;

% Begin for loop
for i = 1:length(tout)
    print_out = sprintf('Discrete Time Step  - %d', i);
    disp(print_out)

    % 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
    [~, unstable_index] = max(real(evals_minus_1));

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

```
        % Step into manifold
        x_manifold_u_p = xout(i,1:6)' + d * unstable_evec;
        x_manifold_u_n = xout(i,1:6)' - d * unstable_evec;

        % If x-velocity is positive, moon-bound
        % If x-velocity if negative, earth-bound
        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_stable_x(:,1), moon_stable_x(:,2), 'blue')
        % plot(earth_stable_x(:,1), earth_stable_x(:,2), 'red')

        if (abs(moon_unstable_x(end,1) - (1-mu)) < 1e-6 &&
moon_unstable_x(end,2) < 0)
            moon_unstable_cnt = moon_unstable_cnt + 1;
            moon_bound_stable(:,moon_unstable_cnt) = moon_unstable;
        else
            moon_unstable_cnt = moon_unstable_cnt + 1;
            moon_bound_stable(:,moon_unstable_cnt) = earth_unstable;
        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] = unstable_event(t,y,mu,n_crossings)
    global count;
```

```
    global poincare_stored;
    global iteration;
    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), iteration];
            % counted_once = true;
        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), iteration];
            % counted_twice = true;
        end
    end
end

function [value,isterminal,direction] = stable_event(t,y,mu,n_crossings)
    global count;
    global poincare_stored;
    global iteration;
    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), iteration];
            % counted_once = true;
        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), iteration];
            % counted_twice = true;
        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

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

function C = jacobi_constant(x, mu)
    C = u_star_times_2(x(1), x(2), x(3), mu) - x(4)^2 - x(5)^2 - x(6)^2;
end

clear; clc; close all;

% ASEN 6060 - Final Project, 2A and 2B
% 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];

% 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];

TOL = 5e-14;

% Set options for ode113
options = odeset('RelTol', TOL, 'AbsTol', TOL);

global count poincare_stored iteration
```

2A & 2B Orbits

```
% L1 orbit is a Lyapunov orbit
load("V_family_L1_Lyapunov.mat")
V_family_L1_Lyapunov = V_family;
l1_orbit_idx = 50;

% L2 orbit is a Lyapunov orbit
load("V_family_L2_Lyapunov.mat")
V_family_L2_Lyapunov = V_family_a1;
l2_orbit_idx = 10;

figure(1)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

[tout_L1, xout_L1] = ode113(@(t,state)CR3BP_full(state,
mu), [0, V_family_L1_Lyapunov(7,l1_orbit_idx)],
[V_family_L1_Lyapunov(1:6,l1_orbit_idx); reshape(eye(6), [36,1])], options);
plot(xout_L1(:,1), xout_L1(:,2), 'LineWidth',2)

[tout_L2, xout_L2] = ode113(@(t,state)CR3BP_full(state,
mu), [0, V_family_L2_Lyapunov(7,l2_orbit_idx)],
[V_family_L2_Lyapunov(1:6,l2_orbit_idx); reshape(eye(6), [36,1])], options);
plot(xout_L2(:,1), xout_L2(:,2), 'LineWidth',2)

cj_L1 = jacobi_constant(V_family_L1_Lyapunov(:,l1_orbit_idx), mu);
cj_L2 = jacobi_constant(V_family_L2_Lyapunov(:,l2_orbit_idx), mu);
delta_cj = cj_L2 - cj_L1;

hold off
legend("L1", "L2", "Moon", "L1 Lyapunov Orbit", "L2 Lyapunov Orbit")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("L1 and L2 Lyapunov Orbits (\DeltaC_j = " + abs(delta_cj) + ")")
```

Manifolds

```
n_crossings = 2;
L1_manifold_time = 10;
L2_manifold_time = 6;

L1_manifolds = stable_manifolds(tout_L1, xout_L1, mu, l1_pos,
L1_manifold_time, n_crossings);
L2_manifolds = unstable_manifolds(tout_L2, xout_L2, mu, l2_pos,
L2_manifold_time, n_crossings);

poincare_stored = [];
stable_options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
stable_event(t, y, mu, n_crossings));
options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
options_event = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
eventFn(t, y, mu));
unstable_options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12, 'Events', @(t,y)
unstable_event(t, y, mu, n_crossings));

figure(2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 3)
hold on
plot(xout_L2(:,1), xout_L2(:,2), 'black', 'LineWidth', 3)
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', ' black')

iteration = 1;

for k = 1:length(L1_manifolds)
    count = 0;
    [moon_stable_t, moon_stable_x] = ode113(@(t, state)CR3BP(state, mu), [0,
-L1_manifold_time], L1_manifolds(:,k), stable_options);
    stable_end_times(k) = moon_stable_t(end);
    stable_end_states(k, :) = moon_stable_x(end, :);
    plot(moon_stable_x(:,1), moon_stable_x(:,2), 'blue')
    iteration = iteration + 1;
end
grid on
axis equal
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)

poincare_stable = poincare_stored;

poincare_stored = [];

iteration = 1;

for k = 1:length(L2_manifolds)
    count = 0;
    [moon_unstable_t, moon_unstable_x] = ode113(@(t, state)CR3BP(state, mu),
[0, L2_manifold_time], L2_manifolds(:,k), unstable_options);
```

```
    unstable_end_times(k) = moon_unstable_t(end);
    unstable_end_states(k, :) = moon_unstable_x(end, :);
    if (abs(unstable_end_states(k, 1) - (1-mu)) < 1e-6 &&
unstable_end_states(k, 2) > 0)
        plot(moon_unstable_x(:,1), moon_unstable_x(:,2), 'red')
    end
    iteration = iteration + 1;
end
poincare_unstable = poincare_stored;

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

Poincare Map

```
figure(3)
subplot(1,2,1)
scatter(poincare_unstable(:,2), poincare_unstable(:,1), 10, 'filled', 'red');
hold on
scatter(poincare_stable(:,2), poincare_stable(:,1), 10, 'filled', 'blue');
xlabel("$\dot{y}$", 'Interpreter','latex')
ylabel("y")
title("Poincar\'e Map", 'Interpreter','latex')
grid on

stable_y_value = 0.0439618;
stable_ydot_value = 0.00816415;
[pc_stable_min, pc_stable_min_ind] = min(abs(poincare_stable(:,1) -
stable_y_value));
stable_ig = poincare_stable(pc_stable_min_ind, 3);

unstable_y_value = 0.0437366;
unstable_ydot_value = 0.00939577;
[pc_unstable_min, pc_unstable_min_ind] = min(abs(poincare_unstable(:,1) -
unstable_y_value));
unstable_ig = poincare_unstable(pc_unstable_min_ind, 3);

unstable_y_value_2 = 0.0441415;
unstable_ydot_value_2 = -0.155022;
[pc_unstable_min_2, pc_unstable_min_ind_2] = min(abs(poincare_unstable(:,1) -
unstable_y_value_2));
unstable_ig_2 = poincare_unstable(pc_unstable_min_ind_2, 3);

scatter(poincare_stable(pc_stable_min_ind, 2),
poincare_stable(pc_stable_min_ind, 1), 'filled', 'cyan')
scatter(poincare_unstable(pc_unstable_min_ind, 2),
poincare_unstable(pc_unstable_min_ind, 1), 'filled', 'black')
scatter(poincare_unstable(pc_unstable_min_ind_2, 2),
poincare_unstable(pc_unstable_min_ind_2, 1), 'filled', 'green')
legend("Unstable", "Stable", "Initial Guess - Stable", "Initial Guess -
Unstable (2B)", "Initial Guess - Unstable (2A)")
hold off
```

```
subplot(1,2,2)
scatter(poincare_unstable(:,2), poincare_unstable(:,1), 10, 'filled', 'red');
hold on
scatter(poincare_stable(:,2), poincare_stable(:,1), 10, 'filled', 'blue');
xlabel("$\dot{y}$", 'Interpreter','latex')
ylabel("y")
title("Poincar\'e Map (Zoomed in)", 'Interpreter','latex')
grid on
scatter(poincare_stable(pc_stable_min_ind, 2),
poincare_stable(pc_stable_min_ind, 1), 'filled', 'cyan')
scatter(poincare_unstable(pc_unstable_min_ind, 2),
poincare_unstable(pc_unstable_min_ind, 1), 'filled', 'black')
scatter(poincare_unstable(pc_unstable_min_ind_2, 2),
poincare_unstable(pc_unstable_min_ind_2, 1), 'filled', 'green')
legend("Unstable", "Stable", "Initial Guess - Stable", "Initial Guess -
Unstable (2B)", "Initial Guess - Unstable (2A)")
hold off
xlim([-0.2, 0.15])
ylim([0.036, 0.05])

sgtitle("Transfers 2A & 2B")
```

Multiple Shooting Initial Guess - 1

```
% Initial guess
x_1_0 = xout_L2(1, 1:6)';
delta_T1 = tout_L2(unstable_ig);
[tout, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T1], x_1_0);
x_1_f = xout_L2(unstable_ig, 1:6)';

x_2_0 = L2_manifolds(:, unstable_ig);
delta_T2 = unstable_end_times(unstable_ig);
[tout, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T2], x_2_0);
x_2_f = unstable_end_states(unstable_ig, :)';

x_3_f = L1_manifolds(:, stable_ig);
delta_T3 = -stable_end_times(stable_ig);
x_3_0 = stable_end_states(stable_ig, :)';

x_4_0 = xout_L1(stable_ig, 1:6)';
delta_T4 = V_family_L1_Lyapunov(7, l1_orbit_idx) - tout_L1(stable_ig);
[tout, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T4], x_4_0);
x_4_f = x_4(end, 1:6)';

r_des_4 = xout_L1(1, 1:3)';

V0 = [x_1_0; delta_T1; x_2_0; delta_T2; x_3_0; delta_T3; x_4_0; delta_T4];

e1 = x_1_f - x_1_0;
e2 = x_2_f - x_2_0;
e3 = x_3_f - x_3_0;
```

```
V_soln = multiple_shooting(V0, mu, true, r_des_4);

flight_time_2B = V_soln(7) + V_soln(14) + V_soln(21) + V_soln(28);

% x_1_0 = xout_L1(unstable_ig, 1:6)';
% ig_um = L1_manifolds(:, unstable_ig);
% ig_sm = L2_manifolds(:, stable_ig);
% ig_L2_PO = xout_L2(stable_ig, 1:6)';
```

Delta V Calc

```
v_1_f = x_1_f(4:6);
v_2_0 = x_2_0(4:6);
v_2_f = x_2_f(4:6);
v_3_0 = x_3_0(4:6);
v_3_f = x_3_f(4:6);
v_4_0 = x_4_0(4:6);

dv1 = calc_dv(v_1_f, v_2_0);
dv2 = calc_dv(v_2_f, v_3_0);
dv3 = calc_dv(v_3_f, v_4_0);

dv_tot_2B = dv1 + dv2 + dv3;
dv_tot_dim_2B = dv_tot_2B * l_star_em / t_star_em;
```

Uncorrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V0(7)], V0(1:6), options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V0(14)], V0(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V0(21)], V0(15:20),
options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V0(28)], V0(22:27),
options);

figure(5)
subplot(2, 1, 2)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)

legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("2B")
```

```
sgtitle("2A and 2B Initial Guesses (Uncorrected) Transfers")
```

Corrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(7)], V_soln(1:6),
options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(14)], V_soln(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(21)],
V_soln(15:20), options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(28)],
V_soln(22:27), options);

x1f = x_1(end,:);

figure(6)
subplot(2,1,2)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 2)
plot(xout_L2(:,1), xout_L2(:,2), 'black', 'LineWidth', 2)

hold off
legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("2B")

sgtitle("2A and 2B Corrected Transfers")
```

Multiple Shooting Initial Guess - 2

```
% Initial guess
x_1_0 = xout_L2(1, 1:6)';
delta_T1 = tout_L2(unstable_ig_2);
[tout, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T1], x_1_0);
x_1_f = xout_L2(unstable_ig_2, 1:6)';

x_2_0 = L2_manifolds(:, unstable_ig_2);
delta_T2 = unstable_end_times(unstable_ig);
[tout, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T2], x_2_0,
options_event);
delta_T2 = tout(end);
```

```
x_2_f = unstable_end_states(unstable_ig_2, :)';

x_3_f = L1_manifolds(:, stable_ig);
delta_T3 = -stable_end_times(stable_ig);
x_3_0 = stable_end_states(stable_ig, :)';

x_4_0 = xout_L1(stable_ig, 1:6)';
delta_T4 = V_family_L1_Lyapunov(7, l1_orbit_idx) - tout_L1(stable_ig);
[tout, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, delta_T4], x_4_0);
x_4_f = x_4(end, 1:6)';

r_des_4 = xout_L1(1, 1:3)';

V0 = [x_1_0; delta_T1; x_2_0; delta_T2; x_3_0; delta_T3; x_4_0; delta_T4];

e1 = x_1_f - x_1_0;
e2 = x_2_f - x_2_0;
e3 = x_3_f - x_3_0;

V_soln = multiple_shooting(V0, mu, true, r_des_4);

flight_time_2A = V_soln(7) + V_soln(14) + V_soln(21) + V_soln(28);

v_1_f = x_1_f(4:6);
v_2_0 = x_2_0(4:6);
v_2_f = x_2_f(4:6);
v_3_0 = x_3_0(4:6);
v_3_f = x_3_f(4:6);
v_4_0 = x_4_0(4:6);

dv1 = calc_dv(v_1_f, v_2_0);
dv2 = calc_dv(v_2_f, v_3_0);
dv3 = calc_dv(v_3_f, v_4_0);

dv_tot_2A = dv1 + dv2 + dv3;
dv_tot_dim_2A = dv_tot_2A * l_star_em / t_star_em;

% x_1_0 = xout_L1(unstable_ig, 1:6)';
% ig_um = L1_manifolds(:, unstable_ig);
% ig_sm = L2_manifolds(:, stable_ig);
% ig_L2_PO = xout_L2(stable_ig, 1:6)';
```

Uncorrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V0(7)], V0(1:6), options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V0(14)], V0(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V0(21)], V0(15:20),
options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V0(28)], V0(22:27),
options);
```

```
figure(5)
subplot(2, 1, 1)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)

hold off
legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("2A")
```

Corrected Trajectory

```
[~, x_1] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(7)], V_soln(1:6),
options);
[~, x_2] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(14)], V_soln(8:13),
options);
[~, x_3] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(21)],
V_soln(15:20), options);
[~, x_4] = ode113(@(t, state)CR3BP(state, mu), [0, V_soln(28)],
V_soln(22:27), options);

x1f = x_1(end,:);
figure(6)
subplot(2,1,1)
scatter(l1_pos(1), l1_pos(2), 'filled', 'blue')
hold on
scatter(l2_pos(1), l2_pos(2), 'filled', 'red')
scatter(p2_pos(1), p2_pos(2), 'filled', 'black')

plot(x_1(:,1), x_1(:,2), 'LineWidth', 2)
plot(x_2(:,1), x_2(:,2), 'LineWidth', 2)
plot(x_3(:,1), x_3(:,2), 'LineWidth', 2)
plot(x_4(:,1), x_4(:,2), 'LineWidth', 2)
plot(xout_L1(:,1), xout_L1(:,2), 'black', 'LineWidth', 2)
plot(xout_L2(:,1), xout_L2(:,2), 'black', 'LineWidth', 2)

hold off
legend("L1", "L2", "Moon")
xlabel('$$\hat{x}$$','Interpreter','Latex', 'FontSize',18)
ylabel('$$\hat{y}$$','Interpreter','Latex', 'FontSize',18)
grid on
axis equal
title("2A")
```

```matlab
function V_soln = multiple_shooting(V0, system_params, plot_input, r_des_4)
    % Script to compute a general three-dimensional periodic orbit via
multiple shooting
    % Inputs
    % V0 - initial guess for a free variable vector
    % statef_V0 - final state when V0 is used as initial guess using CR3BRP
    % EOMs
    % system_params - system parameters
    % r_des_4 - Desired position vector after 4th arc
    %
    % Output
    % V_soln - free variable vector corresponding to a solution

    % Get mass ratio of system
    mu = system_params(1);

    % Set tolerance for numerical integrator and constraint vector
    TOL = 1e-12;

    % Set options for ode113
    options = odeset('RelTol', TOL, 'AbsTol', TOL);

    % Propagate V0 non-linear CR3BP EOMs
    [tout, x1] = ode113(@(t, state)CR3BP(state, mu), [0 V0(7)], V0(1:6),
options);
    [tout, x2] = ode113(@(t, state)CR3BP(state, mu), [0 V0(14)], V0(8:13),
options);
    [tout, x3] = ode113(@(t, state)CR3BP(state, mu), [0 V0(21)], V0(15:20),
options);
    [tout, x4] = ode113(@(t, state)CR3BP(state, mu), [0 V0(28)], V0(22:27),
options);

    % Final final variables using V0
    % statef_V0 = xout(end,:);
    statef_x1 = x1(end,:)';
    statef_x2 = x2(end,:)';
    statef_x3 = x3(end,:)';
    statef_x4 = x4(end,:)';
    statef_V0 = [statef_x1; V0(7); statef_x2; V0(14); statef_x3; V0(21);
statef_x4; V0(28)];

    % Period is a free variable
    % T = V0(end);

    % Initialize constraint vector norm
    F_norm(1) = norm(F(V0, statef_V0, r_des_4));

    % Matrix of all free variable vectors
    V(:,1) = V0;

    % While loop params
    counter = 1;
    counter_max = 50;
```

```
    phi0 = reshape(eye(6), [36, 1]); % Initial phi is identity

    TOL = 1e-10;

    % While loop to reduce F_norm
    while ((F_norm(counter) > TOL) && (counter < counter_max))
        % x1_0 = [V(1:6,counter); phi0];
        % x2_0 = [V(8:13,counter); phi0];
        % x3_0 = [V(15:20,counter); phi0];
        % x4_0 = [V(22:27,counter); phi0];

        % Propagate full state and STM
        [~, x1_out] = ode113(@(t, state)CR3BP(state, mu), [0 V(7,counter)],
V(1:6,counter), options);
        [~, x2_out] = ode113(@(t,state)CR3BP(state, mu), [0 V(14,counter)],
V(8:13,counter), options);
        [~, x3_out] = ode113(@(t,state)CR3BP(state, mu), [0 V(21,counter)],
V(15:20,counter), options);
        [~, x4_out] = ode113(@(t,state)CR3BP(state, mu), [0 V(28,counter)],
V(22:27,counter), options);

        x1_f = x1_out(end, :)';
        x2_f = x2_out(end, :)';
        x3_f = x3_out(end, :)';
        x4_f = x4_out(end, :)';
        statef = [x1_f; V(7,counter); x2_f; V(14,counter); x3_f;
V(21,counter); x4_f; V(28,counter)];

        F_i = F(V(:,counter), statef, r_des_4);
        DF_i = DF_mat(V(:,counter), options, mu);

        % Find V_i+1
        V(:,counter+1) = V(:,counter) - DF_i' * inv(DF_i * DF_i') * F_i;

        % Calculate F_norm and update counter
        F_norm(counter+1) = norm(F_i);
        counter = counter + 1;
    end

    if plot_input
        figure()
        plot([1:counter], F_norm, '-o', 'LineWidth', 2)
        yscale log
        grid on
        xlabel("Iterations")
        ylabel("F Norm")
        title("Constraint Vector Norm for each Iteration")
        hold on
        tol_yline = ones([counter,1])*TOL;

        plot([1:counter], tol_yline, 'red', 'LineWidth', 2)
        hold off
        legend("Norm", "Threshold")
```

```
    end

    V_soln = V(:,end);

end

function out = DF_mat(V, options, mu)
    % Modified constraint DF matrix

    phi0 = reshape(eye(6), [36, 1]); % Initial phi is identity

    x1_0 = [V(1:6); phi0];
    x2_0 = [V(8:13); phi0];
    x3_0 = [V(15:20); phi0];
    x4_0 = [V(22:27); phi0];

    [~, x1_out] = ode113(@(t,state)CR3BP_full(state, mu), [0 V(7)], x1_0,
options);
    [~, x2_out] = ode113(@(t,state)CR3BP_full(state, mu), [0 V(14)], x2_0,
options);
    [~, x3_out] = ode113(@(t,state)CR3BP_full(state, mu), [0 V(21)], x3_0,
options);
    [~, x4_out] = ode113(@(t,state)CR3BP_full(state, mu), [0 V(28)], x4_0,
options);

    x1_f = x1_out(end, :)';
    x2_f = x2_out(end, :)';
    x3_f = x3_out(end, :)';
    x4_f = x4_out(end, :)';

    phi_row_1 = x1_f(7:end);
    phi_mat_1 = reshape(phi_row_1, [6,6])';
    phi_row_2 = x2_f(7:end);
    phi_mat_2 = reshape(phi_row_2, [6,6])';
    phi_row_3 = x3_f(7:end);
    phi_mat_3 = reshape(phi_row_3, [6,6])';
    phi_row_4 = x4_f(7:end);
    phi_mat_4 = reshape(phi_row_4, [6,6])';

    out = [phi_mat_1(1:3,:), x1_f(4:6), -eye([3,6]), zeros([3,15]);
            zeros([3,7]), phi_mat_2(1:3,:), x2_f(4:6), -eye([3,6]),
zeros([3,8]);
            zeros([3,14]), phi_mat_3(1:3,:), x3_f(4:6), -eye([3,6]),
zeros([3,1]);
            zeros([3,21]), phi_mat_4(1:3,:), x4_f(4:6)];

end

function out = F(state0, statef, r_des_4)
    % Modified Constraint Vector
    out = [statef(1:3) - state0(8:10);
            statef(8:10) - state0(15:17);
            statef(15:17) - state0(22:24);
            statef(22:24) - r_des_4];
```

```
end

function state_phi_dot = CR3BP_full(state_phi, mu)
    % Full state vector and state transition matrix differential equation
    % Inputs:
    % state_phi - Augmented state vector and STM [42x1]. The state vector -
    % [x0, y0, z0, x0_dot, y0_dot, z0_dot]. The STM - is 6x6 with each
    % element described as - phi_ij = dxi(tf)/dxj(t0). The phi matrix is
    % reshaped such that all the rows are concatenated vertically. For
    % example -
    % phi_mat = [phi11, phi12, phi13, ..., phi16;
    %            [phi21, phi22, phi23, ..., phi26;
    %            ...
    %            [phi61, phi62, phi63, ..., phi66]
    % becomes
    % phi_row = [phi11, phi12, ..., phi16, phi21, phi22, ..., phi66]'
    %
    % mu - system mass ratio [-]
    %
    % Output
    % state_phi_dot - Augmented state vector dot and STM_dot [42x1]. The
    % augmentation and reshaping scheme remains the same as the input.

    x = state_phi(1);
    y = state_phi(2);
    z = state_phi(3);
    xdot = state_phi(4);
    ydot = state_phi(5);
    zdot = state_phi(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);

    % Calc pseudo-potentials
    uxx = u_xx(mu, [x, y, z]);
    uyy = u_yy(mu, [x, y, z]);
    uxy = u_xy(mu, [x, y, z]);
    uzz = u_zz(mu, [x, y, z]);
    uxz = u_xz(mu, [x, y, z]);
    uyz = u_yz(mu, [x, y, z]);

    U_mat = [uxx, uxy uxz; uxy, uyy uyz; uxz uyz uzz];
    Omega = [0 2 0; -2 0 0; 0 0 0];
    A = [zeros(3), eye(3);
        U_mat, Omega];
```

```
    % Get only the phi elements into a row
    phi_row = state_phi(7:end);

    % Converting phi to matrix
    phi_mat = reshape(phi_row, [6,6])';

    % Get phi_dot
    phi_dot_mat = A * phi_mat;

    % Convert back to row
    phi_dot_row = reshape(phi_dot_mat', [36,1]);

    % Augment state and phi (in row form)
    state_phi_dot = [state_dot; phi_dot_row];

end
```

Published with MATLAB® R2024a