SGN – Assignment #1

Emanuele Gallo, 243222

1 Periodic orbit

Exercise 1: assignment

Consider the 3D Earth–Moon Circular Restricted Three-Body Problem with $\mu = 0.012150$. Note that the CRTBP has an integral of motion, that is, the Jacobi constant

$$J(x, y, z, v_x, x_y, v_z) \coloneqq 2\Omega(x, y, z) - v^2 = C \tag{1}$$

where $\Omega(x, y, z) = \frac{1}{2}(x^2 + y^2) + \frac{1-\mu}{r_1} + \frac{\mu}{r_2} + \frac{1}{2}\mu(1-\mu)$ and $v^2 = v_x^2 + v_y^2 + v_z^2$.

1) Find the coordinates of the five Lagrange points L_i in the rotating, adimensional reference frame with at least 10-digit accuracy and report their Jacobi constant C_i.

Solutions to the 3D CRTBP satisfy the symmetry

$$\mathcal{S} : (x, y, z, \dot{x}, \dot{y}, \dot{z}, t) \to (x, -y, z, -\dot{x}, \dot{y}, -\dot{z}, -t).$$

Thus, a trajectory that crosses perpendicularly the $y = 0$ plane twice is a periodic orbit.

2) Given the initial guess $\mathbf{x}_0 = (x_0, y_0, z_0, v_{x0}, v_{y0}, v_{z0})$, with

$$\begin{aligned} x_0 &= 1.068792441776 \\ y_0 &= 0 \\ z_0 &= 0.071093328515 \\ v_{x0} &= 0 \\ v_{y0} &= 0.319422926485 \\ v_{z0} &= 0 \end{aligned}$$

Find the periodic halo orbit having a Jacobi Constant $C = 3.09$; that is, develop the theoretical framework and implement a differential correction scheme that uses the STM, either approximated through finite differences **or** achieved by integrating the variational equation.
Hint: Consider working on $\varphi(\mathbf{x} + \Delta\mathbf{x}, t + \Delta t)$ and $J(\mathbf{x} + \Delta\mathbf{x})$ and then enforce perpendicular cross of $y = 0$ and Jacobi energy.

The periodic orbits in the CRTBP exist in families. These can be computed by 'continuing' the orbits along one coordinate or one parameter, e.g., the Jacobi energy C. The *numerical continuation* is an iterative process in which the desired variable is *gradually* varied, while the rest of the initial guess is taken from the solution of the previous iteration, thus aiding the convergence process.

3) By gradually decreasing C and using numerical continuation, compute the families of halo orbits until $C = 3.04$.

(8 points)

1.1 Lagrage Point calculation

1.1.1 Lagrange Point calculation: Method

The solution process uses the `fzero` function in MATLAB® since it is suitable in terms of accuracy and computational effort to find the root (or zero) of a well-behaved scalar function of one variable with good initial guesses. The precision requested for the function is set to 10^{-12}, both in absolute and relative value, so the solution is 2 orders of magnitude more precise than the required threshold. The initial guesses are properly set starting from physical considerations:

- The **collinear** Lagrange points lie along the line connecting the two massive bodies. These points can be determined by solving for the critical points of the effective potential function $\Omega(x, 0, 0)$, where the y- and z-coordinates are set to zero. For L_1, the initial guess is set to $[0.2, 1 - 2\mu]$, as this point lies between m_1 and m_2. In the case of L_2, the initial guess is chosen as $[1 + \mu, 2]$ because this point is located beyond m_2 towards $+\infty$. Similarly, for L_3, the initial guess is set to $[-2, -1 + \mu]$, reflecting its position beyond m_1 towards $-\infty$.
- The **triangular** Lagrange points L_4 and L_5 are located at the vertices of equilateral triangles formed with the two primary bodies. These points are determined as the intersections of two circles, each centered on m_1 and m_2, with a radius of 1 in the normalized coordinate system. Given that the distance between m_1 and m_2 is $d = |(1 - \mu) - (-\mu)| = 1$, the y-coordinates of L_4 and L_5 are $\pm \sin(60°) = \pm\sqrt{3}/2$. To find these points, the `fzero` function is employed with specific initial guesses. For L_4, the zero of the function $\Omega(x, \sqrt{3}/2, 0)$ is located using the initial guess $[0, 0.5]$. Similarly, for L_5, the zero of the function $\Omega(x, -\sqrt{3}/2, 0)$ is determined using the same initial guess $[0, 0.5]$.

The positions of the primary bodies in the normalized coordinate system are defined as m_1 (or Earth) at $[-\mu, 0, 0]$ and m_2 (or Moon) at $[1 - \mu, 0, 0]$. It is important to note that the exact positions of m_1 and m_2 cannot be used as initial guesses since the potential function at these points is singular (i.e., tends to infinity), making them unsuitable for numerical root-finding methods. The value of the Jacobi constant at each Lagrange point can be determined by substituting the corresponding coordinates and assuming zero velocity in the general expression Equation 1.

1.1.2 Lagrage Point calculation: Results and comments

The following table summarizes the computed coordinates and Jacobi constants for each Lagrange point:

	L_1	L_2	L_3	L_4	L_5
x	+0.8369180073	+1.1556799131	−1.0050624018	0.4878500000	0.4878500000
y	0	0	0	0.8660254038	−0.8660254038
C	3.2003380950	3.1841582164	3.0241489429	3	3

Table 1: Coordinates and Jacobi constants for the Lagrange points @EMB Earth-Moon rotating frame.

The results for the position confirm the previous intervals chosen. The values of the Jacobi constants decrease in the order $C_{L1} > C_{L2} > C_{L3} > C_{L4} = C_{L5}$, indicating that the energy required to maintain an object at each Lagrange point decreases from L_1 to L_5. The higher Jacobi constants at the collinear points (C_{L1}, C_{L2}, and C_{L3}) correspond to lower total energy in the rotating frame, as these points are more strongly influenced by the gravitational potential of the two primary bodies. In contrast, the lower Jacobi constants at the triangular points

($C_{L4} = C_{L5}$) reflect a higher total energy configuration, resulting from the balance between the gravitational forces of the primary bodies and the centrifugal force, which leads to a stable equilibrium in the rotating frame. Visual representation of the Lagrangian points and their respective potential is presented in Figure 1.

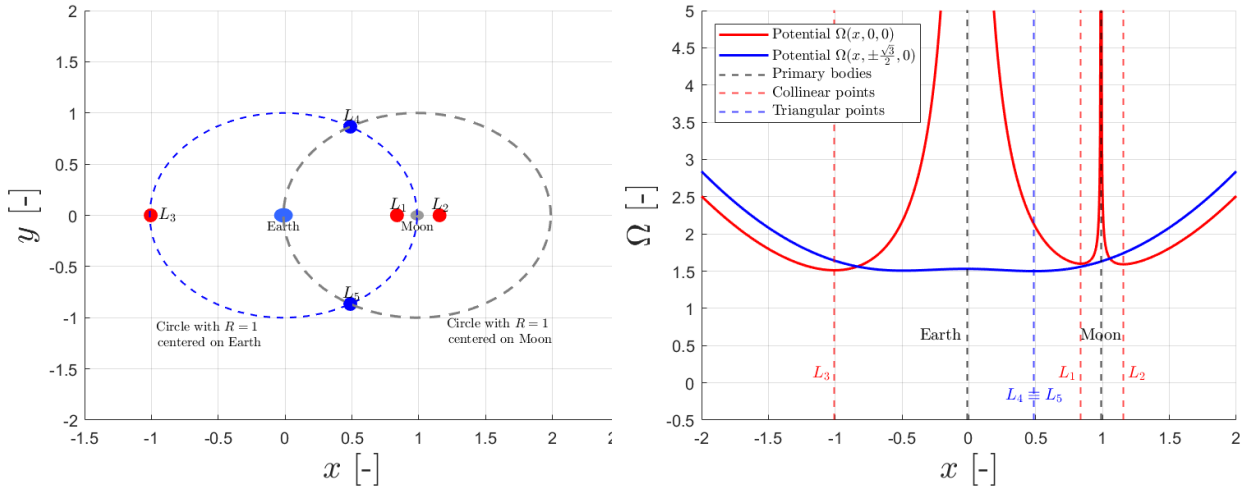


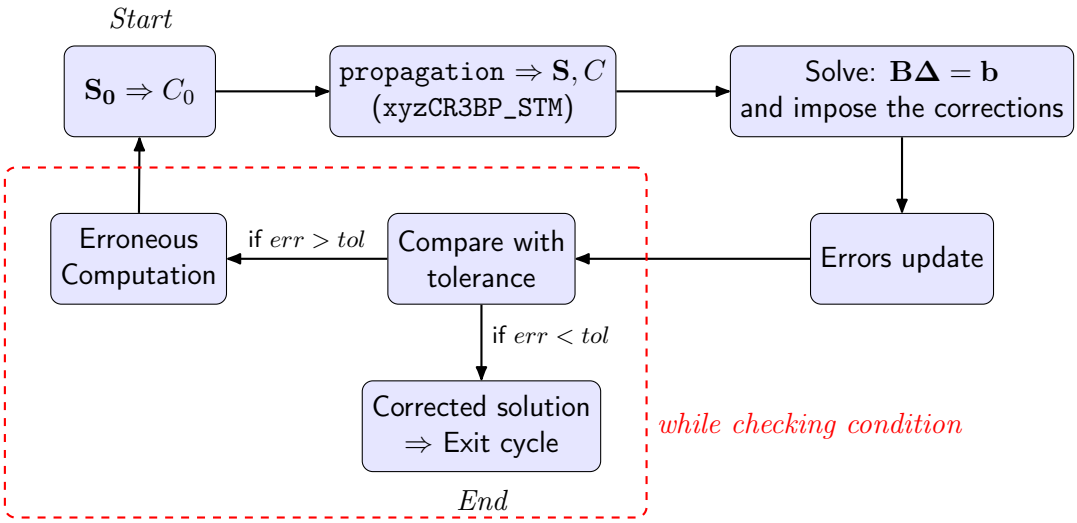
Figure 1: Representation of Lagrangian points (left) and their respective potential (right) @EMB Earth-Moon rotating frame

From this image, it is clear that the potential $\Omega(x, 0, 0)$ tends to infinite values in correspondence of the primaries, and that the Lagrangian points are critical points for their respective potential. In particular, collinear points are saddle points, so they are physically unstable from the potential point of view, while the triangular points are minima points, thus stable.

1.2 Differential Correction for Halo Orbits

1.2.1 Differential Correction: Method

The solution procedure for this point is based on the implemented `diff_corr_STM` function in MATLAB®, which basically implements the following while cycle for each iteration:



The diagram represents the step-by-step process involved in solving the system, starting from the given initial guess $\mathbf{x_0}$ and iteratively applying corrections until convergence is achieved. The following list clarifies the choices adopted:

1. **Start**: The initial state value is taken from the correction at the previous iterate, apart from the first one where $\mathbf{S_0} = \mathbf{x_0}$ is imposed. From the initial state the initial value of the Jacobi constant C_0 is calculated;

2. **Propagation**: The theoretical framework for this exercise is based on the three-dimensional Circular Restricted Three-Body Problem (3D CRTBP). To account for the dynamics of the system, a custom function, `xyzCR3BP_STM`, has been implemented. This function also computes the State Transition Matrix (STM) derived through variational equations. The whole dynamical framework is deepened in subsubsection A.1.1 in Appendix A. The goal of the method is to find a periodic orbit with a Jacobi constant $C = 3.09$ by applying a differential correction scheme. The periodicity of the orbit is ensured by leveraging a property of 3D CRTBP solutions: the trajectory must cross the $y = 0$ plane perpendicularly. This condition imposes that the velocity components in the x- and z-directions vanish at the crossing points, i.e., $\dot{x} = 0$ and $\dot{z} = 0$ at these locations. State propagation and the computation of the STM are handled by the `propagation` function. Within this function, the event of crossing the $y = 0$ plane is detected by passing the event function `xz_plane_crossing` to the integrator, which is, thus, only activated whenever the y-coordinate reaches zero. The solver chosen to propagate the dynamics is `ode78` (see subsubsection A.1.2), a high-order fixed-step solver, which is employed for its robustness and reliability in smooth, non-stiff dynamics such as spacecraft trajectory. This fixed-step approach ensures convergence properties are attributed solely to the optimization methods, isolating algorithmic behavior. Integration tolerances are tailored to balance precision and efficiency: $\text{Tol}_{\mathbf{r}} = 10^{-10}$ for position, $\text{Tol}_{\mathbf{v}} = 10^{-13}$ for velocity, and $\text{Tol}_{\mathbf{\Phi}} = 10^{-13}$ for the state transition matrix, reflecting their respective influences on the dynamics. Propagating up to the final time t, you obtain the final state $\mathbf{S_f} = [x_f, y_f, z_f, v_{xf}, v_{yf}, v_{zf}]^T$

3. **Correction calculation**: The correction procedure, detailed in subsubsection A.1.3 within Appendix A, is summarized here by highlighting its rationale and final outcomes. By linearly approximating perturbations in the state and Jacobi constant C, the flow is expanded with respect to state and time, while C is expanded solely with respect to the state. Using the chain rule and problem-specific definitions, corrections are derived by correlating non-zero parameters with those constrained to zero. Assuming linearity, this relationship is expressed as the following system:

$$\mathbf{B\Delta} = \mathbf{b}. \tag{2}$$

Where $\mathbf{B}$ is the coefficient matrix, $\mathbf{\Delta}$ is the unknown vector, and $\mathbf{b}$ is the known vector. They can be analytically expressed as:

$$\mathbf{B} = \begin{bmatrix} \Phi_{2,1} & \Phi_{2,3} & \Phi_{2,5} & \dot{y}_f \\ \Phi_{4,1} & \Phi_{4,3} & \Phi_{4,5} & \dfrac{\partial \Omega}{\partial x} + 2\dot{y} \\ \Phi_{6,1} & \Phi_{6,3} & \Phi_{6,5} & \dfrac{\partial \Omega}{\partial z} \\ \dfrac{\partial C}{\partial x} & \dfrac{\partial C}{\partial z} & \dfrac{\partial C}{\partial \dot{y}} & 0 \end{bmatrix}; \quad \mathbf{b} = \begin{bmatrix} y_{ref} - y_f \\ v_{x,ref} - \dot{x}_f \\ v_{z,ref} - \dot{z}_f \\ C_{\text{ref}} - C(\mathbf{S}_f) \end{bmatrix}; \quad \mathbf{\Delta} = \begin{bmatrix} \delta x_0 \\ \delta z_0 \\ \delta v_{y0} \\ \delta t \end{bmatrix} \tag{3}$$

Where $\Phi_{i,j}$ with $i = 2, 4, 6$ and $j = 1, 3, 5$ denote the elements of the State Transition Matrix (STM) evaluated at the final state $\mathbf{S}_f$ after propagation, $\Omega(x, y, z)$ is the potential, and $[y_{ref}, v_{x,ref}, v_{z,ref}, C_{ref}]^T$ is the reference condition.

The system defined by Equation 2 can be solved thanks to the \ command on MATLAB®, which automatically selects the most efficient way to solve the system, whose properties are not known a priori. The resulting corrections can be applied to the system by explicating

the terms in $\boldsymbol{\Delta}$:

$$\begin{cases} \delta x_0 = x_{0,\text{new}} - x_{0,\text{old}} = \boldsymbol{\Delta}_1 \\ \delta z_0 = z_{0,\text{new}} - z_{0,\text{old}} = \boldsymbol{\Delta}_2 \\ \delta v_{y0} = v_{y_{0,\text{new}}} - v_{y_{0,\text{old}}} = \boldsymbol{\Delta}_3 \\ \delta t = t_{\text{new}} - t_{\text{old}} = \boldsymbol{\Delta}_4 \end{cases} \Rightarrow \begin{cases} x_{0,\text{new}} = x_{0,\text{old}} + \boldsymbol{\Delta}_1 \\ z_{0,\text{new}} = z_{0,\text{old}} + \boldsymbol{\Delta}_2 \\ v_{y_{0,\text{new}}} = v_{y_{0,\text{old}}} + \boldsymbol{\Delta}_3 \\ t_{\text{new}} = t_{\text{old}} + \boldsymbol{\Delta}_4 \end{cases} \tag{4}$$

The other conditions relative to the initial guess $[y_0, v_{x0}, v_{z0}]^T$ are maintained constant.

4. **Errors update**: The errors are updated between the final state and the reference position, velocity, and Jacobi constant;

5. **While checking condition**: The while loop ends by comparing updated errors with imposed tolerances. If $err > tol$, the cycle repeats until the tolerances are met or the maximum iterations ($N_{max} = 100$) are reached. Otherwise, the corrected solution is obtained. Tolerances are set to 10^{-10} for position, 10^{-13} for velocity, and 10^{-13} for the Jacobi constant to balance computational efficiency with precision. The stricter tolerance for C ensures accuracy, as it is a critical parameter, but smaller tolerances are unnecessary due to the non-chaotic, short-term propagation.

1.2.2 Differential Correction: Results and comments

The corrected initial state $\mathbf{x}_0$ obtained is reported in Table 2: The resulting trajectory is

x	y	z	v_x	v_x	v_x
1.0590402077	0	0.0739277378	0	0.3469245709	0

Table 2: Corrected initial state of the halo orbit with $C = 3.09$ Earth-Moon rotating frame centred @EMB

visualized by propagating the corrected state only backwards in time since the forward propagation is already performed within the `propagation` function. The orbit propagated with the corrected initial condition is reported in Figure 2:

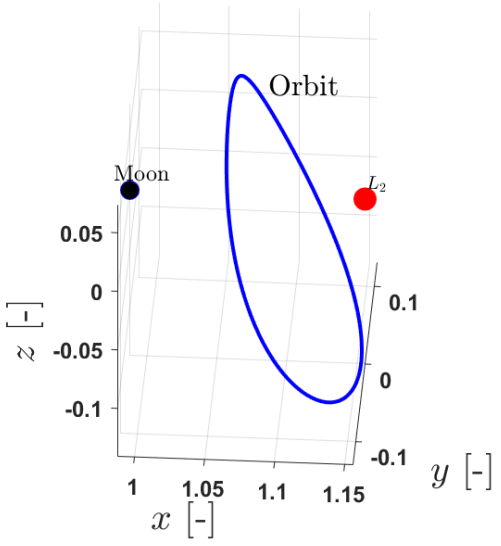


Figure 2: Propagated orbit with corrected initial condition @EMB Earth-Moon rotating frame

The results are validated by verifying the Jacobi constant and the perpendicular crossings, confirming the periodic nature of the obtained halo orbit.

1.3 Numerical Continuation for Halo Orbits

1.3.1 Numerical Continuation for Halo Orbits: Method

A numerical continuation technique was employed to explore a family of halo orbits by gradually varying the Jacobi constant from $C_{\text{ref}} = 3.09$ to $C_f = 3.04$. This is performed thanks to a `for` cycle, in which the `diff_corr_STM` is run iteratively for varying values of C, which are all contained within the vector $\mathbf{C}$. This continuation process enables the tracing of the halo orbit evolution as a function of the Jacobi constant, providing insights into the geometry of the orbits near the L_2 point. Furthermore, it facilitates a more efficient solution to the problem. Specifically, the gradual variation allows you to rely on the previous solution to find the subsequent one, thereby reducing the computational load at each iteration. It is important to note that the number of $\mathbf{C}$-values analyzed must be carefully chosen to avoid too low values, which would render the matrix singular and prevent the solution of the linear system. Conversely, an excessively large number of values should be avoided to prevent unnecessary computations. Through a process of trial and error, a total of 20 elements were chosen for the $\mathbf{C}$-vector analyzed.

1.3.2 Numerical Continuation for Halo Orbits: Results

The following Table 3 summarizes the corrected initial state of the halo orbit with $C = 3.04$:

x	y	z	v_x	v_y	v_x
1.0125655235	0.000000000	0.0672339583	0.000000000	0.5103251959	0.000000000

Table 3: Corrected initial state of the halo orbit with $C = 3.04$ (Earth-Moon rotating frame centred @EMB)

The (sampled) resulting family of halo orbit from $C_{ref} = 3.09$ to $C_{target} = 3.04$ is depicted in Figure 3:

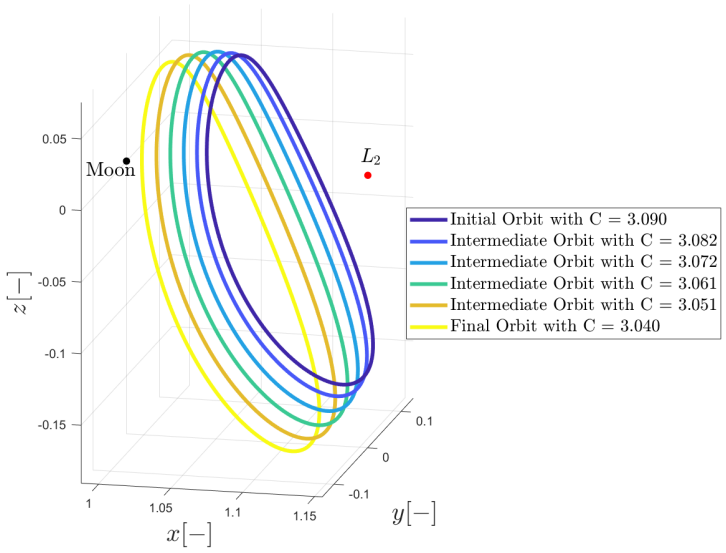


Figure 3: Halo orbit family from $C_{ref} = 3.09$ to $C_{target} = 3.04$ (Earth-Moon rotating frame centred @EMB)

Also in this case, the results are validated by verifying the Jacobi constant, which is equal to the desired one, and the perpendicular crossings, confirming the periodic nature of the reached halo orbit.

2 Impulsive guidance

Exercise 2

Consider the two-impulse transfer problem stated in Section 3.1 (Topputo, 2013)*.

1) Using the procedure in Section 3.2, produce a first guess solution using $\alpha = 0.2\pi$, $\beta = 1.41$, $\delta = 4$, and $t_i = 2$. Plot the solution in both the rotating frame and Earth-centered inertial frame (see Appendix 1 in (Topputo, 2013)). Consider the parameters listed in Table 4 and extract the radius and gravitational parameters of the Earth and Moon from the provided kernels and use the latter to compute the parameter μ.

Symbol	Value	Units	Meaning
m_s	3.28900541×10^5	-	Scaled mass of the Sun
ρ	3.88811143×10^2	-	Scaled Sun-(Earth+Moon) distance
ω_s	$-9.25195985 \times 10^{-1}$	-	Scaled angular velocity of the Sun
ω_{em}	$2.66186135 \times 10^{-1}$	s^{-1}	Earth-Moon angular velocity
l_{em}	3.84405×10^8	m	Earth-Moon distance
h_i	167	km	Altitude of departure orbit
h_f	100	km	Altitude of arrival orbit
DU	3.84405000×10^5	km	Distance Unit
TU	4.34256461	days	Time Unit
VU	1.02454018	km/s	Velocity Unit

Table 4: Constants to be considered to solve the PBRFBP. The units of distance, time, and velocity are used to map scaled quantities into physical units.

2) Considering the first guess in 1) and using $\{\mathbf{x}_i, t_i, t_f\}$ as variables, solve the problem in Section 3.1 with simple shooting in the following cases

   a) without providing any derivative to the solver, and

   b) by providing the derivatives and by estimating the state transition matrix with variational equations.

3) Considering the first guess solution in 1) and the procedure in Section 3.3, solve the problem with multiple shooting taking $N = 4$ and using the variational equation to compute the Jacobian of the nonlinear equality constraints.

4) Perform an n-body propagation using the solution $\{\mathbf{x}_i, t_i, t_f\}$ obtained in point 2), transformed in Earth-centered inertial frame and into physical units. To move from 2-D to 3-D, assume that the position and velocity components in inertial frame are $r_z(t_i) = 0$ and $v_z(t_i) = 0$. To perform the propagation it is necessary to identify the epoch t_i. This can be done by mapping the relative position of the Earth, Moon and Sun in the PCRTBP to a similar condition in the real world:

   a) Consider the definition of $\theta(t)$ provided in Section 2.2 to compute the angle $\theta_i = \theta(t_i)$. Note that this angle corresponds to the angle between the rotating frame x-axis, aligned to the position vector from the Earth-Moon System Barycenter (EMB) to the Moon, and the Sun direction.

   b) The angle θ ranges between $[0, 2\pi]$ and it covers this domain in approximately the revolution period of the Moon around the Earth.

*F. Topputo, "On optimal two-impulse Earth–Moon transfers in a four-body model", Celestial Mechanics and Dynamical Astronomy, Vol. 117, pp. 279–313, 2013, DOI: 10.1007/s10569-013-9513-8.

c) Solve a zero-finding problem to determine the epoch at which the angle Moon-EMB-Sun is equal to θ_i, considering as starting epoch `2024 Sep 28 00:00:00.000 TDB`. **Hints**: Exploit the SPK kernels to define the orientation of the rotating frame axes in the inertial frame for an epoch t. Consider only the projection of the EMB-Sun position vector onto the so-defined x-y plane to compute the angle (planar motion).

Plot the propagated orbit and compare it to the previously found solutions.

(11 points)

2.1 First guess solution

The first guess solution is retrieved referring to the following formulas extracted from the reference [3]. That is, take $\{\alpha,\ \beta,\ t_i,\ \delta\}$ from the given values, and let the initial orbit radius be $r_0 = r_i$ and the initial velocity be $v_0 = \beta\sqrt{(1-\mu)/r_0}$. Thus, the initial transfer state is:

$$x_0 = r_0\cos\alpha - \mu, \quad y_0 = r_0\sin\alpha, \quad \dot{x}_0 = -(v_0 - r_0)\sin\alpha, \quad \dot{y}_0 = (v_0 - r_0)\cos\alpha.$$

The results are reported in Table 5:

$r_{x,0}$ $[DU]$	$r_{y,0}$ $[DU]$	$v_{x,0}$ $[VU]$	$v_{y,0}$ $[VU]$
0.001624	0.010008	-6.302738	8.674975

Table 5: Initial guess (Earth-Moon rotating frame centred @EMB).

The final time is calculated as $t_f = t_i + \delta$, so the initial state vector and the augmented state vector (which is useful to the code) can be defined as follows:

$$\mathbf{S_0} = [x_0, y_0, \dot{x}_0, \dot{y}_0]^T; \quad \mathbf{AS_0} = \left[\mathbf{S_0}^T, t_i, t_f\right]^T \tag{5}$$

Once the first guess solution is computed, the trajectory is propagated using the `propagation` function in MATLAB® with the 2D PBRFBP (Planar Bicircular Restricted Four Body Model) dynamics (fully described in in subsubsection A.2.1 in Appendix A taken from [3]) and `ode78` integrator (see subsubsection A.1.2). The equations of motion are expressed in a rotating reference and are modelled thanks to the function `xyPBRFBP_STM_ROT`. The State Transition Matrix is derived through variational equations. The results are then transformed to the Earth-centered inertial (ECI) frame using the `ROT2ECI` function (whose equations are taken from [3] and reported in subsubsection A.2.2), which converts the state vectors from the rotating frame to an inertial frame centred on the Earth.
The trajectory resulting from the propagation is plotted in both the rotating reference frame and the Earth-Centred Inertial (ECI) reference frame, as shown below:

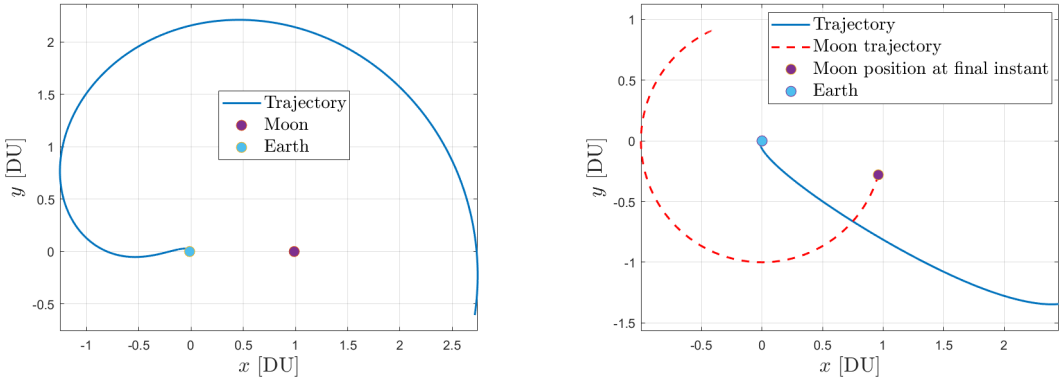


Figure 4: Trajectory shown both in @EMB Earth-Moon rotating frame (left) and ECI (right)

As a result, you can see that the final position of the spacecraft around the Moon is not matched and no circular orbit around the Moon is achieved. This means that the mission is not achieved.

2.2 Simple shooting problem

2.2.1 Simple shooting problem: definition and statement

The spacecraft is initially in a circular Earth orbit with a scaled radius $r_i = (R_E + h_i)/\text{DU}$, where R_E is the Earth radius and h_i is the altitude of the departure orbit. At t_i^- its state is $\mathbf{S_i} = [x_i, y_i, \dot{x}_i, \dot{y}_i]^T$. At t_i^+ a velocity impulse Δv_i aligns with the local circular velocity $\sqrt{(1-\mu)/r_i}$, initiating a transfer trajectory. At t_f^- the satellite state is $\mathbf{S_f} = [x_f, y_f, \dot{x}_f, \dot{y}_f]^T$. At t_f^+ a second impulse Δv_f aligns with the local circular velocity $\sqrt{\mu/r_f}$, inserting the spacecraft into a circular lunar orbit of scaled radius $r_f = (R_M + h_f)/\text{DU}$. The final expression for the (scalar) objective function is:

$$f(\mathbf{S_f}) = \Delta v_i(\mathbf{S_i}) + \Delta v_f(\mathbf{S_f}) \tag{6}$$

Where:

- $\Delta v_i = \sqrt{(\dot{x}_i - y_i)^2 + (\dot{y}_i + x_i + \mu)^2} - \sqrt{\dfrac{1-\mu}{r_i}}$ is a scalar, nonlinear function of $\mathbf{x}_i$, that is $\Delta v_i(\mathbf{S_i})$.
- $\Delta v_f = \sqrt{(\dot{x}_f - y_f)^2 + (\dot{y}_f + x_f + \mu - 1)^2} - \sqrt{\dfrac{\mu}{r_f}}$ is a scalar, nonlinear function of the final state; i.e., $\Delta v_f(\mathbf{S_f})$.

The conditions relative to circular orbit and alignment with local circular velocity for the initial and final conditions can be grouped in the equality constraint vector $\mathbf{c_{eq}}$, defined as follows:

$$\mathbf{c_{eq}} = \begin{bmatrix} \boldsymbol{\Psi_i}(\mathbf{S_i}) \\ \boldsymbol{\Psi_f}(\mathbf{S_f}) \end{bmatrix} = \begin{bmatrix} (x_i+\mu)^2 + y_i^2 - r_i^2 \\ (x_i+\mu)(v_{x_i} - y_i) + y_i(v_{y_i} + x_i + \mu) \\ (x_f+\mu-1)^2 + y_f^2 - r_f^2 \\ (x_f+\mu-1)(v_{x_f} - y_f) + y_f(v_{y_f} + x_f + \mu - 1) \end{bmatrix} = \mathbf{0} \tag{7}$$

The trajectory optimization must avoid solutions which impact either the Earth or the Moon, this is imposed indirectly thanks to the equality constraints, since the initial and final orbits are imposed. Moreover, $t > t_i$ can be imposed as an inequality constraint which is linear on the optimization variables. Thus, it can imposed by defining the linear inequality constraints coefficient matrix, which indeed reduces to a vector, $\mathbf{A} = [\mathbf{0_{(1\times 4)}}, 1, -1]$, and the linear inequality constraints known vector $\mathbf{b}$, which reduces to $b = 0$ scalar.
The **NLP statement** of the simple shooting of Earth-Moon transfers problem so-defined is the following:
Let $\mathbf{x}(t) = \varphi(\mathbf{S_i}, t_i; t)$ be a solution of the 2D PBRFBP for the transfer problem defined, integrated from $(\mathbf{S_i}, t_i)$ to $t \geq t_i$, where $\mathbf{x_i} = [x_i, y_i, v_{xi}, v_{yi}]^T$. The optimization problem for simple shooting Earth-Moon transfers for $\{\mathbf{S_i}, t_i, t_f\}$ with $t_f > t_i$ consists of finding:

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{such that:} \begin{cases} \mathbf{Ax} \leq \mathbf{b}, \\ \mathbf{c_{eq}}(\mathbf{x}) = \mathbf{0}. \end{cases}$$

2.2.2 Simple shooting problem: optimisation without gradient

The optimisation variables are the initial state $\mathbf{S_i}$, the initial time t_i and the final time t_f. The optimisation process is performed thanks to the `fmincon` built-in function in MATLAB®. This is fed with the following inputs:

- **Objective function**: the objective function is imposed thanks to the function `ObjFun` used with 'simple' setting, after an initial propagation under the 2D PBRFBP framework;
- **Initial guess**: the initial guess chosen for this algorithm is the one derived in subsection 2.1, comprising also initial and final time.
- **Solution Boundaries**: The optimization boundaries are defined based on physical constraints to ensure that the spacecraft's trajectory remains within realistic bounds. The spacecraft's x- **and** y**-coordinates** are constrained to the range $[-r_i, +r_i]$, where r_i represents the distance from the Earth's centre to the initial parking orbit. This range is selected to encompass all possible positions within the initial parking orbit. The spacecraft's **velocity components** are bounded with a margin relative to the local circular velocity at the initial Earth orbit. The margin, $v_{\text{margin}} = 1.5\sqrt{\mu_E/r_i}$, defines the lower and upper bounds for the velocity components as $-v_{\text{margin}}$ and $+v_{\text{margin}}$, respectively. For the **time variables**, the initial time, t_i, and the final time, t_f, are lower-bounded by zero. The upper bound on the initial time is set to $-2\pi/\omega_s$, corresponding to the Sun's (dimensionless) orbital period, while the upper bound on the final time is $-2\pi/\omega_s + 28/TU$ (dimensionless), accounting for the approximate dimensionless Moon's orbital period;
- **Constraints**: the non-linear equality constraints are imposed thanks to the implemented function `constraints` used with 'simple' setting. While the linear inequality constraints are imposed passing as proper inputs the matrix $\mathbf{A}$, and the vector $\mathbf{b}$ to the `fmincon` function;
- **Optimization options**: here only the relevant optimization options are discussed. In particular, the built-in algorithm chosen is `'active-set'`. This is because it is well-suited for this spacecraft optimization problem due to its ability to handle nonlinear constraints efficiently. It is particularly effective when constraints are sparse and structured, which matches the problem's setup (position, velocity, and time boundaries). The method also provides numerical stability in problems with nonlinear dynamics, and its flexibility in handling active constraints ensures it converges efficiently to a local minimum. The tolerance on the function and on the constraints is set to 10^{-12} and 10^{-10}, respectively, to obtain an accurate solution. For this stage, no analytical form of the gradient is passed to the function neither from the objective function nor related to the constraint.

2.2.3 Simple shooting problem: optimisation with gradient

The optimisation in this case is performed with the same options as in the previous case, apart from the fact that the analytical form of objective function and constraints gradients is provided to the `fmincon` function. The analytical expressions for the simple shooting problem are extensively treated in subsubsection A.2.3 in Appendix A. In Equation 8 and Equation 9, for fluency reasons, only the final expressions are reported:

$$\boldsymbol{\nabla f} = \left[\frac{\partial(\Delta v_i)}{\partial \mathbf{S_i}} + \frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}}\boldsymbol{\Phi}(t_i,t_f) \quad -\frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}}\boldsymbol{\Phi}(t_i,t_f)\,\mathbf{f_4}(\mathbf{S_i},t_i) \quad \frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}}\mathbf{f_4}(\mathbf{S_f},t_f)\right] \in \Re^{(1\times 6)} \tag{8}$$

$$\boldsymbol{\nabla c_{eq}} = \begin{bmatrix} \dfrac{\partial \boldsymbol{\Psi_i}}{\partial \mathbf{S_i}} & \mathbf{0}_{(2\times 1)} & \mathbf{0}_{(2\times 1)} \\ \dfrac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}}\boldsymbol{\Phi}(\boldsymbol{t_i},\boldsymbol{t_f}) & -\dfrac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}}\boldsymbol{\Phi}(t_i,t_f)\,\mathbf{f_4}(\mathbf{S_i},t_i) & \dfrac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}}\mathbf{f_4}(\mathbf{S_f},t_f) \end{bmatrix} \in \Re^{(4\times 6)} \tag{9}$$

Where $\boldsymbol{\Phi}(t_i,t_f)$ is the STM from time t_i to time t_f, as defined in Equation 41 in subsubsection A.2.1. The term $\mathbf{f_4}(x,t)$ is the vector representing the dynamics, as stated in Equation 40 in subsubsection A.2.1. Both $\boldsymbol{\Phi}(t_i,t_f)$ and $\mathbf{f_4}(x,t)$ can be retrieved as outputs of the dynamics. All the other terms can be computed analytically or symbolically. Those gradients had been computationally tested thanks to the `'CheckGradient'` option, which detected a negligible maximum error. This option, due to excessively stringent tolerance has been removed, to allow the algorithm to run.

2.2.4 Simple shooting problem: results

In both cases, a solution has been found, it satisfies all mission constraints while minimizing the objective function. In particular, the results obtained are:

Gradients	$r_{x,0}$ [DU]	$r_{y,0}$ [DU]	$v_{x,0}$ [VU]	$v_{y,0}$ [VU]	t_i [TU]	t_f [TU]	$tof[TU]$	**Cost** [VU]
False	0.001558	0.010098	-6.344350	8.612645	2.068	6.081	4.013	4.0073
True	0.001620	0.010013	-6.290741	8.651415	2.198	6.203	4.005	4.0068

Table 6: Solutions obtained through simple shooting (@EMB Earth-Moon rotating frame).

The results indicate that the solutions obtained with and without the inclusion of analytical gradients are similar. This close agreement confirms the correctness and reliability of the implemented analytical gradients. Additionally, both the cost function value and the time of flight achieved when using analytical gradients are slightly lower than that obtained without them, highlighting the advantages of incorporating such gradients into the optimization process. Moreover, using analytical gradients facilitates faster convergence of the solver, further demonstrating their beneficial impact on optimization efficiency.
The similarity between the solutions extends to the transfer trajectories, which are depicted in the following figures:

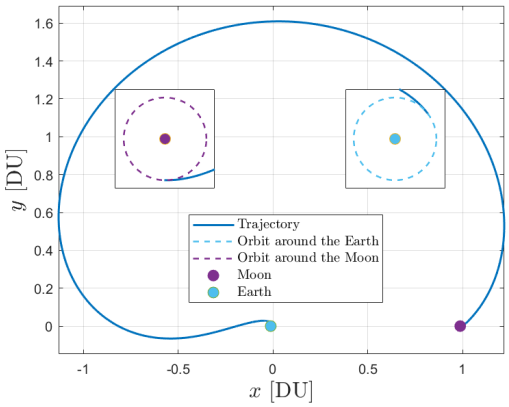


Figure 5: Trajectory optimised without gradient (@EMB Earth-Moon rotating frame)

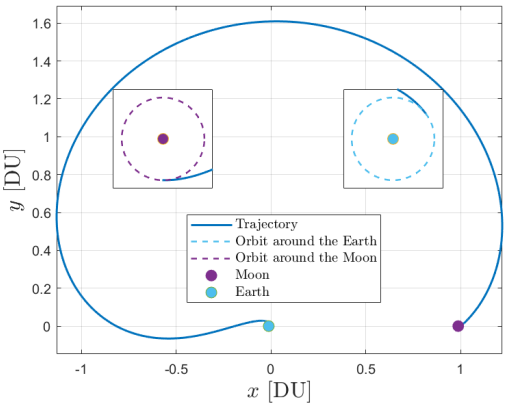


Figure 6: Trajectory optimised with gradient (@EMB Earth-Moon rotating frame)

The trajectories, as shown, are nearly indistinguishable, further supporting the consistency and reliability of the implemented optimization method. Moreover, that is a clear proof that the wanted orbits are reached, as can be seen in the zoom-ins.
It is possible to characterize the final orbit (i.e. around the Moon), obtained by propagating those initial conditions, thanks to the implemented function `characterise_orbits`. It checks the final values of the angular momentum h_2 and Kepler energy H_2, whose definitions are taken from reference [1]. In both cases, since $sign(h_2) < 0$ the final orbit is a retrograde capture solution, while since $sign(H_2) > 0$, the capture is non-ballistic.

2.3 Multiple shooting problem

2.3.1 Multiple shooting problem: definition and statement

This subsection provides the multiple shooting problem definition and statement, which are directly taken from [3].
The equations describing the 2D PBRFBP system, as already done in subsubsection A.2.1, can be written in the first-order form $\dot{x} = \mathbf{f}(\mathbf{x}, t)$, where $\mathbf{x}(t)$ represents the solution. These equations are discretized over a uniform time grid, defined by $N = 4$ points such that $t_1 = t_0 + n\Delta t$ and

$$t_j = t_1 + \frac{j-1}{N-1}(t_N - t_1), \quad j = 1, \ldots, N. \tag{10}$$

Let $x_j = (y_j, \dot{y}_j, z_j, \dot{z}_j)$ denote the state at the j-th grid point, such that $x_j = x(t_j)$ for $j = 1, \ldots, N = 4$. Using the multiple shooting method, the dynamics are integrated over the intervals $[t_j, t_{j+1}]$, with x_j as the initial condition for each interval. Continuity of position and velocity requires that the defects

$$\boldsymbol{\zeta_j} = \varphi(\mathbf{x}_j, t_j; t_{j+1}) - \mathbf{x_{j+1}}, \quad j = 1, \ldots, N-1,$$

must vanish. The decision variables for the nonlinear programming (NLP) problem are organized into a vector

$$\mathbf{y} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}, t_1, t_N\}, \tag{11}$$

where the initial and final times, t_1 and t_N, are left free to vary in order to find an optimal solution. Defining $\boldsymbol{\Psi_1} = \{\Psi_i\}(\mathbf{x_1})$ and $\boldsymbol{\Psi} = \varphi(\mathbf{x_N})$, the equality constraints can be expressed as:

$$\mathbf{c_{eq,4}}(\mathbf{y}) = \{\boldsymbol{\Psi}_1, \boldsymbol{\zeta}_2, \ldots, \boldsymbol{\zeta}_{N-1}, \boldsymbol{\Psi}_N\} \tag{12}$$

To ensure the trajectory avoids collision with the Earth or Moon, constraints are imposed at each grid point:

$$\left(\frac{R_E}{DU}\right)^2 - (x_j + \mu)^2 + y_j^2 > 0; \quad \left(\frac{R_M}{DU}\right)^2 - (x_j + \mu - 1)^2 + y_j^2 > 0, \quad j = 1, \ldots, N.$$

Additionally, the transfer duration must be strictly positive, meaning $t_N - t_1 < 0$. Combining these, the inequality constraints are represented as:

$$\mathbf{c_4}(\mathbf{y}) = \{\eta_1, \eta_2, \ldots, \eta_N, \tau\} \tag{13}$$

where $\boldsymbol{\eta_j = \eta(x_j)}$ is derived from the left-hand sides of the previous constraints, and $\tau = t_1 - t_N$. The objective function to be minimized is

$$f(\mathbf{y}) = \Delta v_1 + \Delta v_N, \tag{14}$$

where Δv_1 and Δv_N are computed using the initial and final states, as done in Equation 6.
The dimensionality of the variables and constraints can be summarized as follows: the decision vector $\mathbf{y}$ has a size of $4N + 2$, the equality constraints $\mathbf{c_4}$ form a vector of size $4N$, and the inequality constraints $\mathbf{g}$ form a vector of size $2N+1$. With the constraints satisfied, the problem includes $4N + 2$ variables and $4N$ constraints.
With the problem so defined, the **NLP statement** of the multiple shooting of Earth-Moon transfers problem is the following:
Being Equation 11 the optimisation variable, the optimization problem for multiple shooting Earth-Moon transfers for $\{\mathbf{Si}, t_i, t_f\}$ *with* $t_f > t_i$ *involves finding:*

$$\min_{\mathbf{y}} f(\mathbf{y}) \quad \text{such that:} \begin{cases} \mathbf{c_4}(\mathbf{y}) \leq \mathbf{0}, \\ \mathbf{c_{4,eq}}(\mathbf{y}) = \mathbf{0}. \end{cases}$$

Analytical form of the gradients for the multiple shooting problem

This subsection provides analytical derivatives for the objective function and constraints with respect to the optimization variables of the NLP multiple shooting problem. Those considerations are taken from the reference [1].
The derivative of the objective function f with respect to the optimization variables $\mathbf{y}$ is expressed as:

$$\frac{\partial f}{\partial \mathbf{y}} = \begin{bmatrix} \mathbf{P_1} & \mathbf{0}_{(1\times 8)} & \mathbf{P_4} & \mathbf{0}_{(1\times 2)} \end{bmatrix} \in \Re^{(1\times 18)} \tag{15}$$

where the components $\mathbf{P_1}$ and $\mathbf{P_4}$ are given by:

$$\mathbf{P_1} = \frac{\partial f}{\partial \mathbf{x_1}} = \frac{1}{\sqrt{(\dot{x_1} - \dot{y_1})^2 + (\dot{y_1} + x_1 + \mu)^2}} \begin{bmatrix} \dot{y_1} + x_1 + \mu, & \dot{y_1}, & -\dot{x_1}, & -\dot{y_1} + x_1 + \mu \end{bmatrix}, \in \Re^{(1\times 4)}$$

$$\mathbf{P_4} = \frac{\partial f}{\partial \mathbf{x_4}} = \frac{1}{\sqrt{(\dot{x_4} - \dot{y_4})^2 + (\dot{y_4} + x_4 + \mu - 1)^2}} \begin{bmatrix} \dot{y_4} + x_4 + \mu - 1, & \dot{y_4}, & -\dot{x_4}, & -\dot{y_4} + x_4 + \mu - 1 \end{bmatrix} \in \Re^{(1\times 4)}$$

For the equality constraints $\mathbf{c_{eq,4}} = \{\mathbf{\Psi_1}, \boldsymbol{\zeta_j}, \mathbf{\Psi_4}\}$, their derivative with respect to $\mathbf{y}$ is written as:

$$\frac{\partial \mathbf{c_{eq,4}}}{\partial \mathbf{y}} = \begin{bmatrix} \mathbf{R_1} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 1)} & \mathbf{0}_{(2\times 1)} \\ \mathbf{\Phi}(t_1, t_2) & -\mathbf{I_4}_{(4\times 4)} & \mathbf{0}_{(4\times 4)} & \mathbf{0}_{(4\times 4)} & \mathbf{Q_1^1} & \mathbf{Q_1^4} \\ \mathbf{0}_{(4\times 4)} & \mathbf{\Phi}(t_2, t_3) & -\mathbf{I_4}_{(4\times 4)} & \mathbf{0}_{(4\times 4)} & \mathbf{Q_2^1} & \mathbf{Q_2^4} \\ \mathbf{0}_{(4\times 4)} & \mathbf{0}_{(4\times 4)} & \mathbf{\Phi}(t_3, t_4) & -\mathbf{I_4}_{(4\times 4)} & \mathbf{Q_3^1} & \mathbf{Q_3^4} \\ \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{R_4} & \mathbf{0}_{(2\times 1)} & \mathbf{0}_{(2\times 1)} \end{bmatrix} \in \Re^{(16\times 18)} \tag{16}$$

The terms $\mathbf{Q_j^1}$, $\mathbf{Q_j^3}$, $\mathbf{R_1}$, and $\mathbf{R_3}$ are defined as follows:

$$\mathbf{Q_j^1} = \frac{\partial \boldsymbol{\zeta_j}}{\partial t_1} = \frac{4-j}{4-1}\mathbf{\Phi}(t_j, t_{j+1})\mathbf{f}(\mathbf{x}, t_j) + \frac{4-j-1}{4-1}\mathbf{f}(\mathbf{\Phi}(\mathbf{x}, t_j, t_{j+1}), t_{j+1}), \in \Re^{(4\times 1)}$$

$$\mathbf{Q_j^4} = \frac{\partial \boldsymbol{\zeta_j}}{\partial t_4} = \frac{j}{4-1}\mathbf{\Phi}(t_j, t_{j+1})\mathbf{f}(\mathbf{x}, t_j) + \frac{j-1}{4-1}\mathbf{f}(\mathbf{\Phi}(\mathbf{x}, t_j, t_{j+1}), t_{j+1}), \in \Re^{(4\times 1)}$$

$$\mathbf{R_1} = \frac{\partial \mathbf{\Psi_1}}{\partial \mathbf{x_1}} = \begin{bmatrix} 2(x_1 + \mu) & 2y_1 & 0 & 0 \\ \dot{x}_1 & \dot{y}_1 & x_1 + \mu & y_1 \end{bmatrix} \in \Re^{(2\times 4)}$$

$$\mathbf{R_4} = \frac{\partial \mathbf{\Psi_4}}{\partial \mathbf{x_4}} = \begin{bmatrix} \in \Re^{(2)} 2(x_N + \mu - 1) & 2y_4 & 0 & 0 \\ \dot{x}_4 & \dot{y}_4 & x_4 + \mu - 1 & y_4 \end{bmatrix} \in \Re^{(2\times 4)}$$

The inequality constraints $\mathbf{c_4}(\mathbf{y}) = \{\boldsymbol{\eta_j}, \tau\} < 0$ have a derivative with respect to $\mathbf{y}$ expressed as:

$$\frac{\partial \mathbf{c_4}}{\partial \mathbf{y}} = \begin{bmatrix} \mathbf{S}_1 & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 2)} \\ \mathbf{0}_{(2\times 4)} & \mathbf{S}_2 & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 2)} \\ \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{S}_3 & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 2)} \\ \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{0}_{(2\times 4)} & \mathbf{S}_4 & \mathbf{0}_{(2\times 2)} \\ \mathbf{0}_{(1\times 4)} & \mathbf{0}_{(1\times 4)} & \mathbf{0}_{(1\times 4)} & \mathbf{0}_{(1\times 4)} & \mathbf{S}_\tau \end{bmatrix} \in \Re^{(9\times 18)} \tag{17}$$

The components S_j (for $j = 1, \ldots, 4$) and S_τ are written as:

$$\mathbf{S_j} = \frac{\partial \boldsymbol{\eta_j}}{\partial \mathbf{x_j}} = \begin{bmatrix} -2(x_j + \mu) & -2y_j & 0 & 0 \\ -2(x_j + \mu - 1) & -2y_j & 0 & 0 \end{bmatrix}; \in \Re^{(2\times 4)} \quad \mathbf{S}_\tau = \begin{bmatrix} \frac{\partial \tau}{\partial t_1} & \frac{\partial \tau}{\partial t_4} \end{bmatrix} = \begin{bmatrix} 1 & -1 \end{bmatrix} \in \Re^{(1\times 2)}$$

2.3.2 Multiple Shooting Problem: implementation

The optimization variables are contained in the vector $\mathbf{y}$. As in the previous case, the optimization process employs the MATLAB® built-in function `fmincon`. The same functions and same choices (algorithm, gradient check, ...) of simple case are used also in multiple case, but with 'multiple_4' setting. Only the modified inputs compared to the simple shooting are detailed here:

- **Initial guess**: The initial guess for the multiple shooting method is derived from the trajectory previously optimized using a gradient-based approach (see Table 6). The trajectory is divided into four segments at the time instants specified in Equation 10. At each node, the initial state matches the final state of the preceding segment, forming the state component of the initial guess. Each node contributes four elements to the state vector, resulting in a total of 16 state-related components. Additionally, the initial and final times, taken directly from Equation 10 constitue two more components. This construction ensures segment continuity and provides a consistent basis for optimization.

- **Solution Boundaries**: The lower and upper boundaries for the initial state (`lb_m` and `ub_m`) are defined to balance flexibility and consistency with the physical behavior of the problem. For the first node ($i = 1$), the boundaries are set as

$$\mathbf{lb}_m = [-4, -4, -\infty, -\infty]; \qquad \mathbf{ub}_m = [4, 4, \infty, \infty],$$

providing a broad range for the initial guess. These constraints ensure sufficient freedom for the optimization algorithm while limiting the position variables (x, y) to a physically reasonable domain based on prior problem knowledge. For subsequent segments ($i > 1$), the boundaries are adjusted relative to the initial guesses **AS0_m_na**, computed as:

$$\mathbf{lb}_m = \mathbf{AS0\_m\_na} - [4, 4, 5v_{\text{margin}}, 5v_{\text{margin}}]; \quad \mathbf{ub}_m = \mathbf{AS0\_m\_na} + [4, 4, 5v_{\text{margin}}, 5v_{\text{margin}}]$$

Here, v_{margin}, defined in subsubsection 2.2.2, accounts for potential variations in velocity, ensuring additional flexibility to accommodate trajectory corrections. This adaptive strategy maintains consistency with the propagated solution while allowing deviations necessary for optimization.

2.3.3 Multiple shooting problem: results

The main results of the described algorithm are reported in Table 7:

$r_{x,0}$ $[DU]$	$r_{y,0}$ $[DU]$	$v_{x,0}$ $[VU]$	$v_{y,0}$ $[VU]$	t_i $[TU]$	t_f $[TU]$	tof $[TU]$	Cost $[VU]$
0.004105	0.005065	-3.181423	10.210316	2.430	6.088	3.658	3.9973

Table 7: Multiple shooting solution (Earth-Moon rotating frame centred @EMB).

A comparison of the results with those in Table 6 clearly demonstrates the advantages of the multiple shooting technique. It provides a solution with a lower cost and shorter time of flight, while the primary drawback is the higher number of optimization iterations. However, these iterations are computationally efficient, and do not significantly affect the overall performance. The trajectory, along with the corresponding zoom-in views, is shown in the rotating reference frame:

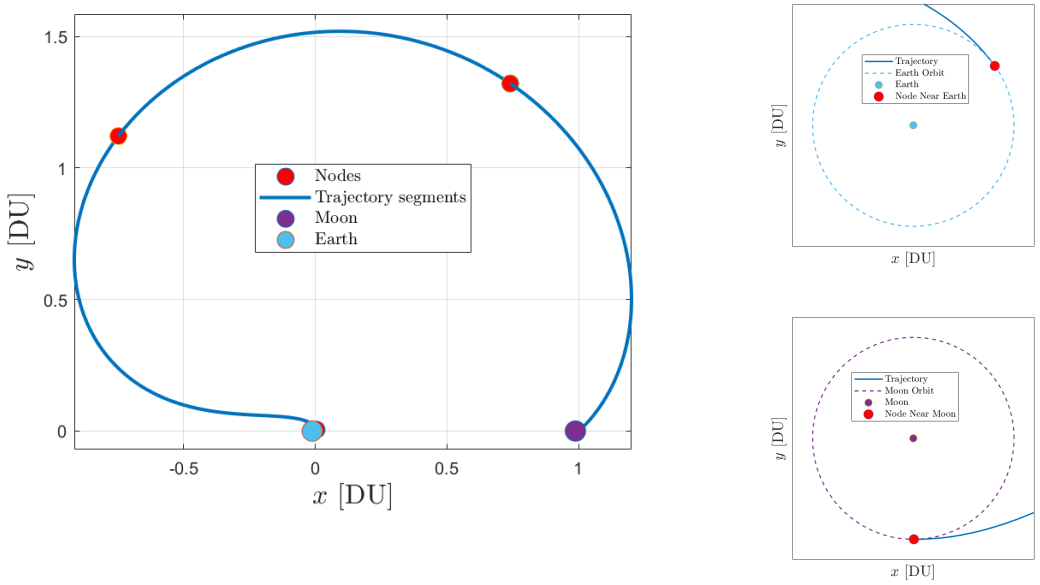


Figure 7: Transfer orbit for multiple shooting (left) together with relative zoomed-in views near Earth (top-right) and Moon (bottom-right). The reference frame is @EMB Earth-Moon rotating.

These figures illustrate the effectiveness of the multiple shooting method in achieving a trajectory that closely matches the prescribed initial and final orbits, as required by the problem setup.
Additionally, using the same orbit characterization algorithm `characterise_orbits`, the angular momentum is found to be negative, and the Kepler energy positive. This confirms the retrograde nature of the orbit and non-ballistic capture. The solutions obtained via both simple shooting and multiple shooting methods are consistent in this regard.

2.4 Real world analysis

2.4.1 Epoch finding

The solution begins by determining the initial Moon-EMB-Sun angle, calculated as $\theta_i = \omega_s t_i = 243.4764°$ (refer to [3]). The time t_i is taken from the optimization results in Table 6, derived using gradient-based methods for improved accuracy. The `theta_find` function is implemented to determine the system's initial configuration epoch. The steps followed by this function are:

- The Sun's position and the Moon's state are retrieved from SPK kernels, relative to the Earth-Moon Barycenter (EMB) in the ECLIPJ2000 or ECI reference frame.
- The orbital axes of the Moon $\mathbf{x_{Moon}}, \mathbf{y_{Moon}}$, and $\mathbf{z_{Moon}}$ are computed. These vectors define the rotation matrix from ECLIPJ2000 to the Moon's orbital frame. Explicit expressions can be found in Appendix A.
- The Sun's position vector is transformed to the Moon's orbital frame, and the angle θ between the transformed vector and $\mathbf{x}_{\text{Moon}}$ is computed.
- The angle $\theta - \theta_i$ is calculated. This is done thanks to angdiff function, to account properly angle periodicity. The absolute value is not considered to ensure proper root-finding behavior for the `fzero` algorithm).

The evolution of θ over the period *28 Sep 2024* to *28 Dec 2024* (see Figure 8) shows a nearly periodic 30-day evolution, consistent with expectations. Notably, $\theta - \theta_i$ changes sign for the first time after *28 Sep 2024*, between *8 Oct 2024* and *16 Oct 2024*. These bounds are passed to the `fzero` algorithm to locate the root and determine t_i. Since the transfer matches that in subsubsection 2.2.4, the same time of flight (tof) applies, allowing determination of the final epoch t_f. Results are summarized below:

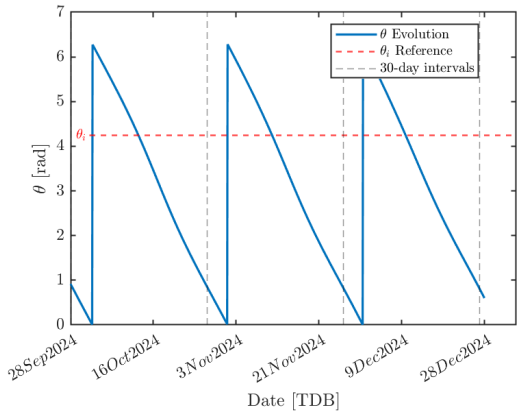


Figure 8: θ evolution over time.

Symbol	**Calendar epoch** (UTC)
t_i	2024-10-12 21:25:33.697
t_f	2024-10-30 06:50:19.858

Table 8: Initial and final epochs in the Earth-centered inertial frame.

2.4.2 Propagation in n-body and PBRFBP

The initial conditions obtained in subsection 2.2 are transformed from the Earth-Moon frame centred at EMB to ECI, by using the implemented `ROT2ECI` MATLAB® function (see subsubsection A.2.2 in Appendix A). Moreover, the z-coordinates are added both in position and velocity (both assumed null), and the resulting conditions are dimensionalised. The results are reported in Table 9.

$r_{x,0}$ $[km]$	$r_{y,0}$ $[km]$	$r_{z,0}$ $[km]$	$v_{x,0}$ $[km/s]$	$v_{y,0}$ $[km/s]$	$v_{z,0}$ $[km/s]$
-6223.62808715	2026.14573648	0.0	-3.39800047	-10.43749755	0.0

Table 9: Initial state in Earth-centered inertial frame

After that, the retrieved initial conditions are propagated in an n-body framework, accounting for the Sun, Moon, all solar system planets (or, equivalently their barycentre), and Pluto, obtaining the (violet) trajectory in Figure 9. The same transformation of the solution (i.e. rotation to ECI frame, z-coordinates adding in position and velocity, and dimensionalisation) is performed for the complete trajectory retrieved by optimising with simple shooting with gradient. The results are shown (in light blue) in the Figure 9, together with the Moon trajectory in reality (extracted from kernels in @Earth ECLIPJ2000 frame) and as predicted by PBRFBP. The approach assumes that the Earth-Centered Inertial (ECI) frame aligns with the ECLIPJ2000 frame. In reality, their differences are minimal when assuming that the Moon's orbit lies within the ecliptic plane. For a more precise comparison of the two trajectories, an alternative method is employed. The trajectory in the rotating frame is mapped to the ECLIPJ2000 frame using an appropriate rotation matrix, as detailed in subsubsection A.2.4 in Appendix A. The resulting trajectory is illustrated in Figure 10.

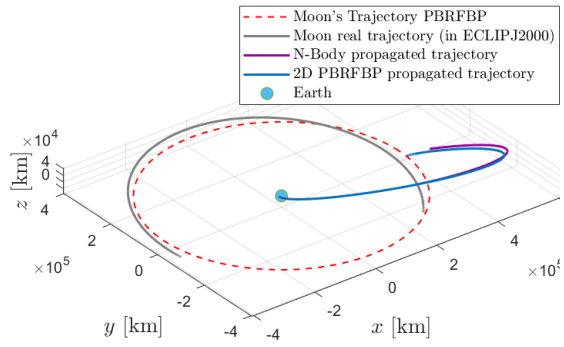


Figure 9: N-body, PBRFBP, and Moon trajectories in Earth inertial reference frame

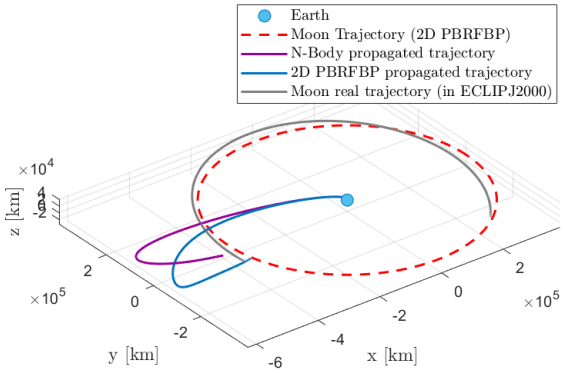


Figure 10: N-body, PBRFBP, and Moon trajectories in Earth ECLIPJ2000 frame.

In both representations, the final positions of the spacecraft and Moon trajectories within the 2D PBRFBP model align with the desired final orbit, consistent with the problem's requirements (i.e., the spacecraft reaches the Moon). However, as time progresses, significant divergence between the 2D PBRFBP and n-body trajectories occurs, primarily due to the perturbative effects of additional planetary bodies in the n-body model. Aside from the apparent deviation in the x- and y-coordinates, these perturbations induce a noticeable z-coordinate variation, which is not accounted for in the PBRFBP model. In Figure 10, the xy-plane represents the ecliptic plane, with the PBRFBP trajectory confined to the Moon's orbital plane. The spacecraft intersects the true Moon orbit shortly before the trajectory concludes, ending at the position of the circular Moon. This mismatch arises because the PBRFBP model neglects the actual lunar orbit. In contrast, the n-body model causes the trajectory to deviate from the lunar plane, resulting in a discrepancy with the expected final conditions.

3 Continuous guidance

Exercise 3

A low-thrust option is being considered to perform an orbit raising maneuver using a low-thrust propulsion system in Earth orbit. The spacecraft is released on a circular orbit on the equatorial plane at an altitude of 800 km and has to reach an orbit inclined by 0.75 deg on the equatorial plane at 1000 km. This orbital regime is characterized by a large population of resident space objects and debris, whose spatial density q can be expressed as:

$$q(\rho) = \frac{k_1}{k_2 + \left(\dfrac{\rho - \rho_0}{DU}\right)^2}$$

where ρ is the distance from the Earth center. The objective is to design an optimal orbit raising that minimizes the risk of impact, that is to minimize the following objective function

$$F(t) = \int_{t_0}^{t_f} q(\rho(t))\,\mathrm{d}t.$$

The parameters and reference Distance Unit to be considered are provided in Table 10.

Symbol	Value	Units	Meaning
h_i	800	km	Altitude of departure orbit
h_f	1000	km	Altitude of arrival orbit
Δi	0.75	deg	Inclination change
R_e	6378.1366	km	Earth radius
μ	398600.435	$\mathrm{km}^3/\mathrm{s}^2$	Earth gravitational parameter
ρ_0	$750 + R_e$	km	Reference radius for debris flux
k_1	1×10^{-5}	DU^{-1}	Debris spatial density constant 1
k_2	1×10^{-4}	DU^2	Debris spatial density constant 2
m_0	1000	kg	Initial mass
$T_{\max}$	3.000	N	Maximum thrust
I_{sp}	3120	s	Specific impulse
DU	7178.1366	km	Distance Unit
MU	m_0	kg	Mass Unit

Table 10: Problem parameters and constants. The units of time TU and velocity VU can be computed imposing that the scaled gravitational parameter $\overline{\mu} = 1$.

1) Plot the debris spatial density $q(\rho) \in [h_i - 100;\, h_f + 100]$ km and compute the initial state and target orbital state, knowing that: i) the initial and final state are located on the x-axis of the equatorial J2000 reference frame; ii) the rotation of the angle Δi is performed around the x-axis of the equatorial J2000 reference frame (RAAN = 0).

2) Adimensionalize the problem using as reference length $DU = \rho_i = h_i + R_e$ and reference mass $MU = m_0$, imposing that $\mu = 1$. Report all the adimensionalized parameters.

3) Using the PMP, write down the spacecraft equations of motion, the costate dynamics, and the zero-finding problem for the unknowns $\{\boldsymbol{\lambda}_0, t_f\}$ with the appropriate transversality condition. **Hint**: the spacecraft has to reach the target state computed in point 1).

4) Solve the problem considering the data provided in Table 10. To obtain an initial guess for the costate, generate random numbers such that $\lambda_{0,i} \in [-250; +250]$ while $t_f \approx 20\pi$.

Report the obtained solution in terms of $\{\boldsymbol{\lambda}_0, t_f\}$ and the error with respect to the target. Assess your results exploiting the properties of the Hamiltonian in problems that are not time-dependent and time-optimal solution. Plot the evolution of the components of the primer vector $\boldsymbol{\alpha}$ in a NTW reference frame[†].

5) Solve the problem for a lower thrust level $T_{\max} = 2.860$ N. Compare the new solution with the one obtained in the previous point. **Hint**: exploit numerical continuation.

(11 points)

3.1 Spatial debris function and initial and final state computation

The result of the plot of the assigned spatial debris function in the prescribed interval is:

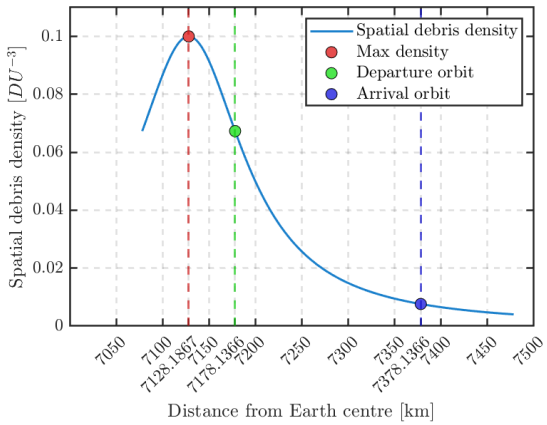


Figure 11: Spatial debris density as a function of distance from Earth centre

The peak density of approximately $0.1\,\mathrm{DU}^{-3}$ occurs at 7128.1867 km. The region with the highest collision risk for a spacecraft should be either avoided or, if the S/C forced to pass there, be crossed as fast as possible. A trade-off between collision risk and fuel consumption must be performed to obtain a successful mission analysis. Moving away from the peak, increasing or decreasing the distance from Earth's centre, the spatial debris density tends to decrease, making those regions safer to work on. The initial and final radii for the orbits are also highlighted and it can be seen that the transfer is advantageous to minimise the debris collision risk.
The initial and final state can be retrieved by imposing the circularity of both the initial and final orbit and exploiting the information given. The results are the following Table 11:

$r_{x,i}$ $[km]$	$r_{y,i}$ $[km]$	$r_{z,i}$ $[km]$	$v_{x,i}$ $[km/s]$	$v_{y,i}$ $[km/s]$	$v_{z,i}$ $[km/s]$
7178.136600	0.000000	0.000000	0.000000	7.45183148	0.000000
$r_{x,f}$ $[km]$	$r_{y,f}$ $[km]$	$r_{z,f}$ $[km]$	$v_{x,f}$ $[km/s]$	$v_{y,f}$ $[km/s]$	$v_{z,f}$ $[km/s]$
7378.136600	0.000000	0.000000	0.000000	7.34950906	0.09621034

Table 11: Initial and target state in Earth-centered equatorial J2000 inertial frame.

3.2 Adimensionalization

The adimensionalization of the parameters is performed to help the convergence of the problem. The results of this process are reported in the following Table 12:

[†]The T-axis is aligned with the velocity, the N-axis is normal to the angular momentum, while the W-axis is pointing inwards, i.e., towards the Earth.

$h_{i,a}$	$h_{f,a}$	Δi_a	$R_{e,a}$	μ_a	$\rho_{0,a}$	k_1	k_2	$m_{0,a}$	$T_{\max,\mathrm{a}}$	$I_{\mathrm{sp,\ a}}$	$g_{0,a}$
0.1114	0.1393	0.0131	0.8886	1	0.9930	$1 \cdot 10^{-5}$	$1 \cdot 10^{-4}$	1	0.0003878	3.2390	1.2681

Table 12: Adimensionalised parameters for the orbit raising problem

The adimensionalized final and initial states are $\mathbf{S_{f,a}} = [1.02786238, 0, 0, 0, 0.98626882, 0.01291097]^T$ and $\mathbf{S_{i,a}} = [1, 0, 0, 0, 1, 0]^T$, respectively.

3.3 PMP application to low-thrust guidance problem

3.3.1 Preliminary definitions

Before delving deeper into the subject, the following quantities are defined for clarity. The augmented state vector is given by $\mathbf{x} = [\mathbf{r}, \mathbf{v}, m]^T$, where $\mathbf{r}$ represents the position vector, $\mathbf{v}$ the velocity vector, and m the mass. The norms of $\mathbf{r}$ and $\mathbf{v}$ are r and v, respectively. All these quantities are nondimensional due to the adimensionalization process applied. The costate vector is defined as $\boldsymbol{\lambda} = [\boldsymbol{\lambda_r}, \boldsymbol{\lambda_v}, \lambda_m]^T$, where $\boldsymbol{\lambda_r}$ is the costate associated with the position vector, $\boldsymbol{\lambda_v}$ with the velocity vector, and λ_m with the mass. The norms of $\boldsymbol{\lambda_r}$ and $\boldsymbol{\lambda_v}$ are given by λ_r and λ_v, respectively. The full state vector is $\mathbf{y} = [\mathbf{x}, \boldsymbol{\lambda}]^T$, representing the combined state and costate vectors. Finally, the base vector is $\mathbf{x_b} = [\mathbf{S_{i,a}}, m_{0,a}]^T = [\mathbf{r_0}, \mathbf{v_0}, m_{0,a}]^T$, which includes the initial position $\mathbf{r_0}$, initial velocity $\mathbf{v_0}$, and initial mass $m_{0,a}$.

3.3.2 PMP application, state and costate dynamics

The required **minimization problem** for the case analysed is stated as follows:

$$\min_{\mathbf{u}\in U} F(t) := \int_{t_0}^{t_f} q(\rho(t))\, dt \qquad \text{subject to: } \begin{cases} \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t), \\ \mathbf{x}(t_0) = \mathbf{x}_0, \\ x_i(t_f) = x_{f,i}, \quad i = 1, \ldots, I < n, \end{cases} \tag{18}$$

Where $\mathbf{u} = u\hat{\boldsymbol{\alpha}}_{\boldsymbol{u}}$ is the thrust vector, defined by the control magnitude u and the thrust-pointing unitary vector $\hat{\boldsymbol{\alpha}}_{\boldsymbol{u}}$. U is the admissible control set, that for this specific case can be defined as $U := \{(u, \hat{\boldsymbol{\alpha}}_{\boldsymbol{u}}) \mid u \in [0, 1], \|\hat{\boldsymbol{\alpha}}_{\boldsymbol{u}}\| = 1\}$. The Lagrange function $l(\mathbf{x}, \mathbf{u}, t)$ is equal to the spatial debris density $q(\rho)$ (see subsection 3.1). Thus, the Hamiltonian function can be defined as follows:

$$\mathcal{H} = q + \boldsymbol{\lambda} \cdot \mathbf{f} = q + \boldsymbol{\lambda_r} \cdot \mathbf{v} - \frac{\mu}{r^3}\boldsymbol{\lambda_v} \cdot r + \frac{uT_{max}}{m}\hat{\boldsymbol{\alpha}}_{\boldsymbol{u}} \cdot \boldsymbol{\lambda_v} - \lambda_m \left(\frac{uT_{max}}{I_{sp}g_0}\right) \tag{19}$$

The following specifications must be done:

- The **system dynamics** $\dot{\mathbf{x}} = \partial\mathcal{H}/\partial\boldsymbol{\lambda} = f(\mathbf{x}, \mathbf{u}, t)$ in this context is defined as:

$$\begin{cases} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\dfrac{\mu_a}{\|\mathbf{r}\|^3}\mathbf{r} + u\dfrac{T_{\max,a}}{m_a}\hat{\boldsymbol{\alpha}}_{\boldsymbol{u}} \\ \dot{m} &= -u\dfrac{T_{\max,a}}{I_{\mathrm{sp,\ a}}g_{0,a}} \end{cases} \Rightarrow f(\mathbf{x}, \mathbf{u}, t) = \begin{bmatrix} \mathbf{v} \\ -\dfrac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} + u\dfrac{T_{\max,a}}{m_a} \\ -u\dfrac{T_{\max,a}}{I_{sp,a}g_{0,a}} \end{bmatrix} = \mathbf{f} \tag{20}$$

  where the parameters considered are the adimensional ones, thus defined as in subsection 3.2. This choice is adapted to aid the convergence properties of the algorithm used to solve the problem;

- The boundary conditions (BCs) for a fixed initial and final state, and initial mass are defined as follows:

$$\mathbf{x}(t_0) = \mathbf{x_b} = \begin{cases} \mathbf{r}(t_0) = \mathbf{r_0} \\ \mathbf{v}(t_0) = \mathbf{v_0} \\ m(t_0) = m_{0,a} \end{cases} \quad ; \quad \mathbf{x_i}(t_f) = \mathbf{x_{f,i}} = \begin{cases} \mathbf{r}(t_f) = \mathbf{r_f} \\ \mathbf{v}(t_f) = \mathbf{v_f} \\ \lambda_m(t_f) = 0 \end{cases} \tag{21}$$

Where the condition $\lambda_m(t_f) = 0$ is imposed since the mass at the final instant is a free parameter;

- The **costate dynamics** $\dot{\boldsymbol{\lambda}} = -\partial\mathcal{H}/\partial\mathbf{x} = \mathbf{g}$ can be written as follows:

$$\begin{cases} \dot{\boldsymbol{\lambda}}_{\boldsymbol{r}} &= \dfrac{-3\mu_a}{r^5}\left(\mathbf{r}\cdot\boldsymbol{\lambda}_{\boldsymbol{v}}\right)\mathbf{r} + \dfrac{\mu_a}{r^3}\boldsymbol{\lambda}_{\boldsymbol{v}} - \dfrac{\partial q}{\partial \mathbf{r}} \\ \dot{\boldsymbol{\lambda}}_{\boldsymbol{v}} &= -\boldsymbol{\lambda}_{\boldsymbol{r}} \\ \dot{\lambda}_m &= -u\dfrac{\lambda_v T_{\max,a}}{m_a^2} \end{cases} \Rightarrow g(\mathbf{x},\mathbf{u},t) = \begin{bmatrix} \dfrac{-3\mu_a}{r^5}\left(\mathbf{r}\cdot\boldsymbol{\lambda}_{\boldsymbol{v}}\right)\mathbf{r} + \dfrac{\mu_a}{r^3}\boldsymbol{\lambda}_{\boldsymbol{v}} - \dfrac{\partial q}{\partial \mathbf{r}} \\ -\boldsymbol{\lambda}_{\boldsymbol{r}} \\ -u\dfrac{\lambda_v T_{\max,a}}{m_a^2} \end{bmatrix} = \mathbf{g} \tag{22}$$

Where the gradient of the objective function with respect to the position can be calculated as follows:

$$\frac{\partial q}{\partial \mathbf{r}} = -\frac{2k_1\left(r-\rho_{0,a}\right)}{\left(k_2+\left(r-\rho_{0,a}\right)^2\right)^2 r}\mathbf{r}; \tag{23}$$

- By applying the Pontryagin Maximum Principle (PMP), the optimal thrust direction direction is the so-defined primer vector $\hat{\boldsymbol{\alpha}}^*_{\boldsymbol{u}} = -\boldsymbol{\lambda}_{\boldsymbol{v}}/\lambda_v$. By substituting this into Equation 19, the switching function and the related thrust-throttle policy are:

$$S_t = -\frac{\lambda_v}{m_a} I_{sp,a} g_{0,a} - \lambda_m, \qquad u^* := \begin{cases} 0 & \text{if } S_t > 0 \\ 1 & \text{if } S_t < 0 \\ \in (0,\,1) & \text{if } S_t = 0 \end{cases}$$

From Equation 22, it is evident that $\dot{\lambda_m} < 0$, and, from Equation 21, $\lambda_m(t_f) = 0$. This means $\lambda_m(t) \geq 0$. In turn, this means that $S_t(t) < 0$, and therefore $u^*(t) = 1$.

By imposing $\delta F = 0$, the following **necessary conditions** must be verified to effectively solve the problem:

$$\begin{cases} \dot{\mathbf{x}} = \dfrac{\partial\mathcal{H}}{\partial\boldsymbol{\lambda}} = \mathbf{f}, & \mathbf{x}(t_0) = \mathbf{x_0} \\ \dot{\boldsymbol{\lambda}} = -\dfrac{\partial\mathcal{H}}{\partial\mathbf{x}} = \mathbf{g}, & \mathbf{x}(t_f) = \boldsymbol{\Psi} = \mathbf{x_{f,i}} \\ (\mathbf{u} := \text{argmin}\mathcal{H}(\mathbf{x},\boldsymbol{\lambda},\mathbf{u})) & \\ \mathcal{H}(t_f) = 0 & \end{cases} \tag{24}$$

The first three equations of Equation 24 are the Euler-Lagrange equations. In particular, the third is in between brackets since it is automatically satisfied by imposing $\hat{\boldsymbol{\alpha}}^*_{\boldsymbol{u}} = -\boldsymbol{\lambda}_{\boldsymbol{v}}/\lambda_v$ and $u^* = 1$. The fourth one is the **transversality condition**. In general, the transversality condition is defined as $\mathcal{H}(t_f) - \boldsymbol{\lambda}(t_f)\cdot\dot{\boldsymbol{\Psi}}(t_f) = 0$. However, in the case analysed $\dot{\boldsymbol{\Psi}}(t_f) = 0$, since $\boldsymbol{\Psi} = \mathbf{x_{f,i}}$ is constant (see Equation 21).

3.3.3 Zero-finding problem

Finally, the **zero-finding problem** can be formulated as follows:
Starting from the initial condition $\mathbf{x}(t_0) = \mathbf{x_0}$*, find* $\{\boldsymbol{\lambda_0}, t_f\}$ *such that* $\varphi\left(\left[\mathbf{x_0},\boldsymbol{\lambda_0}\right]^T, t_0, t_f\right)$ *produces a solution* $\mathbf{x}(t)$ *and* $\boldsymbol{\lambda}(t)$ *satisfying the following conditions at* t_f*:*

$$\begin{cases} \mathbf{r}(t_f) - \mathbf{r_f} = 0, \\ \mathbf{v}(t_f) - \mathbf{v_f} = 0, \\ \lambda_m(t_f) = 0, \\ \mathcal{H}(t_f) = 0. \end{cases} \tag{25}$$

3.4 PMP solution

3.4.1 PMP solution: method

To solve the zero-finding problem in subsubsection 3.3.3 the following logical steps, within a `while` cycle, are followed:

1. Randomized initial guesses for $\boldsymbol{\lambda}_0$ are generated within the range $[-250, 250]$ for each component. This is not valid for the λ_m component, which is only randomly generated in the range $[0, 250]$, since $\lambda_m(t) \geq 0$, as already demonstrated. The initial time guess is set as $t_f = 20\pi$ with a perturbation range of $\pm\pi$;

2. The solver `fsolve` is run, since the optimisation is multidimensional. The condition in Equation 25 are imposed thanks to `zerofun`, in which the propagation from t_0 to t_f is performed using `ode78`. The dynamics is imposed thanks to `H_dynamics` function, which accounts both for the state and costate dynamics;

If the optimization result is satisfying, i.e. the convergence is reached, the cycle is stopped, otherwise it restarts.
For solving nonlinear least-squares problems under random and broad initial conditions, the 'Levenberg-Marquardt' algorithm is preferred. It combines rapid convergence of Gauss-Newton and robustness of gradient descent, making it versatile, especially for low-thrust guidance problems with steep gradients. It balances stability and speed, outperforming trust-region methods in this case. High iterations and evaluations are needed, with a strong emphasis on tolerances for function and variables to reach an optimal solution.

3.4.2 PMP solution: results

The solving final mass and time together with the correct lambdas are hereinafter reported:

t_f [mins]	m_f $[kg]$
1035.1974	993.9120

λ_{0,r_x}	λ_{0,r_y}	λ_{0,r_z}	λ_{0,v_x}	λ_{0,v_y}	λ_{0,v_z}	$\lambda_{0,m}$
−214.9812	−10.3659	0.8856	−10.3929	−214.6105	−112.9454	2.5964

Table 13: Optimal orbit raising transfer solution ($T_{\max} = 3.000$ N).

The final state is characterised by the following errors on position and velocity:

Error	**Value**	**Units**
$\|\|\mathbf{r}(t_f) - \mathbf{r}_f\|\|$	$5.9141 \cdot 10^{-8}$	km
$\|\|\mathbf{v}(t_f) - \mathbf{v}_f\|\|$	$5.8315 \cdot 10^{-8}$	m/s

Table 14: Final state error with respect to target position and velocity ($T_{\max} = 3.000$ N).

The errors seen can be safely neglected, and it can be said that the final state is reached.
The problem analyzed is time-independent, as the Lagrangian lacks explicit time dependence. This property imposes constancy conditions on the Hamiltonian. From the transversality condition, $\mathcal{H}(t_f) = 0$. Consequently, the Hamiltonian must be uniformly null throughout the optimal solution, as confirmed by numerical verification through Figure 12. The trend shown validates the results obtained.
The evolution of the components of the primer vector $\hat{\boldsymbol{\alpha}}_{\boldsymbol{u}}^*$ in NTW reference frame in this case is shown in Figure 13.

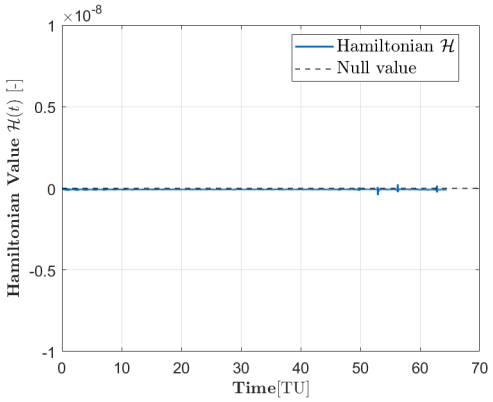


Figure 12: Hamiltonian function evolution

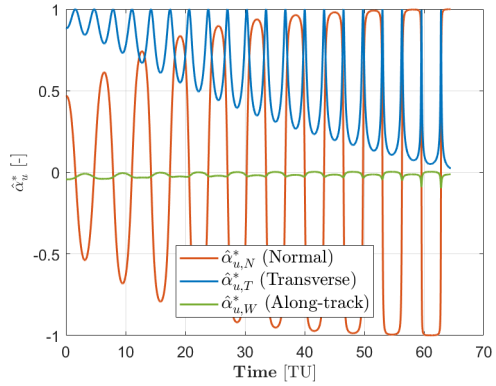


Figure 13: Time-evolution of $\hat{\boldsymbol{\alpha}}^*_{\boldsymbol{u}}$ in NTW reference frame

This plot in Figure 13 is essential for analyzing the temporal evolution of the spacecraft's thrust vector, which provides critical insight into the control strategy. A key characteristic that emerges is the periodic nature of the components. This behavior is consistent with the inherent orbital motion of the spacecraft, to which the low-thrust guidance problem must adapt to successfully perform optimal adjustments. It is noteworthy that the normal component of the thrust vector is the dominant one. This indicates that the primary effort of the control strategy is focused on altering the spacecraft's inclination, a type of maneuver that is typically the most costly in terms of energy expenditure. Furthermore, the transverse component becomes increasingly dominant, suggesting a moderate yet growing change in the spacecraft's velocity and, consequently, its energy. Finally, the along-track component remains relatively small but shows a gradual increase. This implies that the control strategy's least effort, although still growing, is directed towards increasing the spacecraft's distance from the central attracting body.
The obtained trajectory can be represented in two ways: first, with unscaled z-coordinates to preserve the real-world perspective, and second, with scaled z-coordinates to enhance the clarity and visualization of the trajectory, which is actually characterised by small variations in inclination.

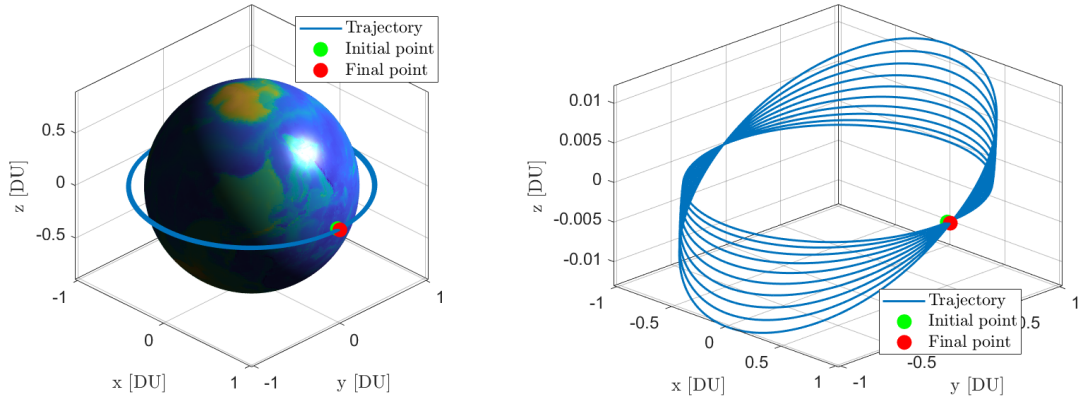


Figure 14: Non-scaled (on the left) and properly scaled (on the right) optimised trajectories in Earth-centred J2000

3.5 Numerical continuation: solution for reduced thrust

3.5.1 Numerical continuation: method

Once the problem with $T_{\max} = 3\,\mathrm{N}$ has been solved, the case with $T_{\max} = 2.860\,\mathrm{N}$ is analyzed while keeping the other parameters constant. To address this, numerical continuation is em-

ployed. The thrust is gradually reduced to the desired value, with the solution being found iteratively for each intermediate thrust value, which is then used as the initial guess for the subsequent iteration to facilitate convergence. A total of 10 distinct thrust values are considered, determined through a trial-and-error approach to optimize both numerical efficiency and the performance of the `fsolve` algorithm. This selection strikes a balance between avoiding excessive computation with similar thrust values and preventing algorithm overload by avoiding overly divergent values. In this context, the 'trust-region-dogleg' method is selected, as it outperforms the previously used algorithm, particularly in the more reliable region of the solution space.

3.5.2 Numerical continuation: results

In the following tables the results found for the requested thrust level are reported in terms of final time, final mass, the related costate vector to find the solution and the norm of the errors for the position and velocity on the final state:

t_f [mins]	m_f $[kg]$
1030.9760	994.2198

λ_{0,r_x}	λ_{0,r_y}	λ_{0,r_z}	λ_{0,v_x}	λ_{0,v_y}	λ_{0,v_z}	$\lambda_{0,m}$
-593.1529	-11.5904	2.2973	-11.9293	-592.7890	-920.3698	17.3814

Table 15: Optimal orbit raising transfer solution ($T_{\max} = 2.860$ N).

Error	**Value**	**Units**
$\|\|\mathbf{r}(t_f) - \mathbf{r}_f\|\|$	$1.4208 \cdot 10^{-7}$	km
$\|\|\mathbf{v}(t_f) - \mathbf{v}_f\|\|$	$1.3942 \cdot 10^{-7}$	m/s

Table 16: Final state error with respect to target position and velocity ($T_{\max} = 2.860$ N).

Under the assumption of unchanged other parameters and almost the same mission, reducing thrust in this Earth-Moon transfer increases final mass while extending transfer time. Lower thrust slows propellant consumption, leaving more propellant unburned and increasing final mass. Simultaneously, reduced thrust enables to better exploit the natural dynamics of the problem, allowing the spacecraft to use them effectively rather than opposing them with high thrust. Thus, in this case, reducing the thrust allows for a more efficient transfer in terms of fuel consumption and transfer time. Moreover, the errors asses at lower values compared to the previous case.
The Hamiltonian function, having the same problem as in the previous case, must respect the same properties, i.e. remaining constantly null. This is verified thanks to the Figure 15.
As for the primer vector components, their evolution across the transfer can be depicted as in Figure 16.

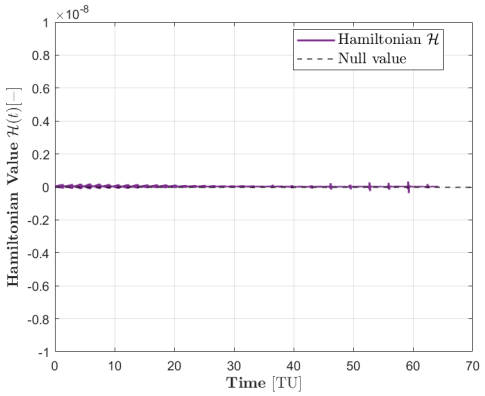


Figure 15: Hamiltonian function time evolution

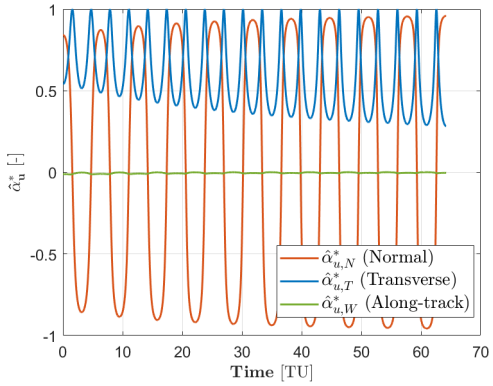


Figure 16: Time-evolution of $\hat{\boldsymbol{\alpha}}^*_{\boldsymbol{u}}$ in NTW reference frame

As in the previous case, the control components exhibit a quasi-periodic behavior, reflecting the symmetry of the orbits and the dynamics of the problem. The normal component starts at higher values due to the reduced maximum thrust, requiring larger initial corrections to reach the final orbit from the same starting conditions. Toward the end of the transfer, the effort decreases as fewer corrections are needed. The tangential component remains the second most significant control direction, with an initially higher value that decreases progressively as the final orbit is approached. The along-track component, however, maintains the lowest values throughout the transfer.

The total control effort is higher in the reduced-thrust case ($T = 2.86\,\text{N}$). This has the main consequences of reducing the transfer time (as already observed in subsubsection 3.5.2), but making the higher-thrust case ($T = 3\,\text{N}$) preferable when feasible. Additionally, this observation aligns with Figure 18, where the higher thrust maintains lower spatial debris density values along the transfer, thus, it fulfils the mission goals more effectively.

The control differences between the $T = 3\,\text{N}$ and $T = 2.86\,\text{N}$ cases naturally influence the transfer orbit, resulting in coherent adjustments. These trajectories (both real and scaled ones), which are nearly identical to those of the previous case, are detailed in subsubsection A.2.6 within the Appendix A, while the orbit comparison is shown in Figure 17.

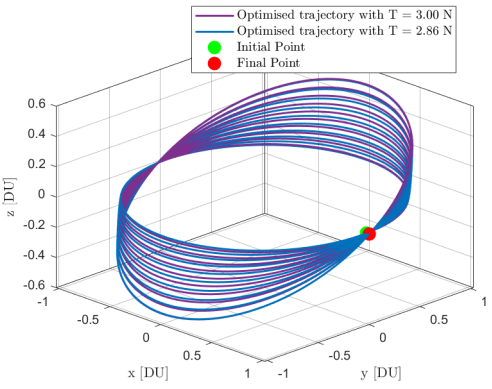


Figure 17: Comparison between the (scaled) optimised trajectories for $T = 3N$ (in violet) and $T = 2.86N$ (in blue) in J2000 centred @Earth

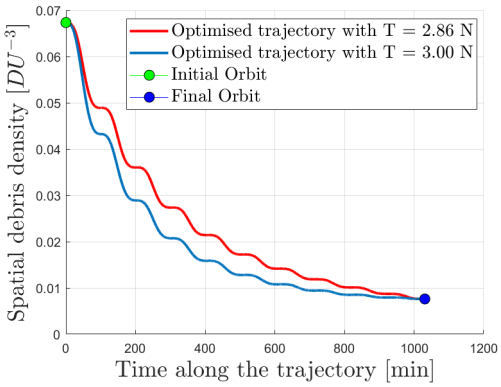


Figure 18: Spatial debris density along the optimised trajectories

A Appendix

A.1 Appendix: Exercise 1

A.1.1 3D CRTBP dynamics

This paragraph deepens the model developed for the three-dimensional Circular Restricted Three Body Problem (3D CRTBP) used in .
The 3D CRTBP dynamics implemented can be written as follows (presented both as a system of equations and in matrix form):

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{z} = v_z \\ \dot{v_x} = \dfrac{\partial \Omega}{\partial x} + 2v_y \\ \dot{v_y} = \dfrac{\partial \Omega}{\partial y} - 2v_x \\ \dot{v_z} = \dfrac{\partial \Omega}{\partial z} \end{cases} \quad \text{with initial conditions:} \quad \begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \\ z(t_0) = z_0 \\ v_x(t_0) = v_{x0} \\ v_y(t_0) = v_{y0} \\ v_z(t_0) = v_{z0} \end{cases} \tag{26}$$

Rewriting the system in matrix form:

$$\dot{S} = \left[v_x, v_y, v_z, \frac{\partial \Omega}{\partial x} + 2v_y, \frac{\partial \Omega}{\partial y} - 2v_x, \frac{\partial \Omega}{\partial z} \right]^T = \mathbf{f}, \tag{27}$$

Where $\dot{\mathbf{S}}$ is the time variation of the state $\mathbf{S} = [x, y, z, v_x, v_y, v_z]^T = [\mathbf{r}, \mathbf{v}]^T$, $\mathbf{r}$ is the position, $\mathbf{v}$ is the velocity, $\mathbf{f}$ is the vector condensing the dynamics of the system, and $\Omega(x, y, z)$ is the (scalar) potential function relative to the Earth-Moon 3D CRTBP.
The STM from initial time t_0 to final time t can be found by solving the following 36 scalar ODEs:

$$\begin{cases} \text{Variational equation :} & \dot{\mathbf{\Phi}}(t_0, t) = \mathbf{A}(t)\mathbf{\Phi}(t_0, t) \\ \text{Initial conditions :} & \mathbf{\Phi}(t_0, t_0) = \mathbf{I} \end{cases} \tag{28}$$

Where $\mathbf{I} \in \Re^{6x6}$ is the identity matrix, and the matrix $\mathbf{A}(t)$ can be retrieved symbolically by computing the Jacobian matrix of the dynamics over the state, that is $\mathbf{A}(t) = \dfrac{d\mathbf{f}}{d\mathbf{S}}$. The result is reported here:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \Omega_{xx} & \Omega_{xy} & \Omega_{xz} & 0 & 2 & 0 \\ \Omega_{yx} & \Omega_{yy} & \Omega_{yz} & -2 & 0 & 0 \\ \Omega_{zx} & \Omega_{zy} & \Omega_{zz} & 0 & 0 & 0 \end{bmatrix} \tag{29}$$

Where:

$$\Omega_{xx} = \frac{\mu-1}{r_1^3} - \frac{\mu}{r_2^3} + \frac{3\mu(\mu+x-1)(\mu+x-1)}{r_2^5} - \frac{3(\mu+x)(\mu+x)(\mu-1)}{r_1^5} + 1;$$
$$\Omega_{xy} = \frac{3\mu y(\mu+x-1)}{r_2^5} - \frac{3y(\mu+x)(\mu-1)}{r_1^5} = \Omega_{yx};$$
$$\Omega_{xz} = \frac{3\mu z(\mu+x-1)}{r_2^5} - \frac{3z(\mu+x)(\mu-1)}{r_1^5} = \Omega_{zx};$$
$$\Omega_{yy} = \frac{\mu-1}{r_1^3} - \frac{\mu}{r_2^3} - \frac{3y^2(\mu-1)}{r_1^5} + \frac{3\mu y^2}{r_2^5} + 1;$$
$$\Omega_{yz} = \frac{3\mu yz}{r_2^5} - \frac{3yz(\mu-1)}{r_1^5} = \Omega_{zy};$$
$$\Omega_{zz} = \frac{\mu-1}{r_1^3} - \frac{\mu}{r_2^3} - \frac{3z^2(\mu-1)}{r_1^5} + \frac{3\mu z^2}{r_2^5}.$$

While $r_1 = \sqrt{(x+\mu)^2+y^2+z^2}$ and $r_2 = \sqrt{(x+\mu-1)^2+y^2+z^2}$.

A.1.2 `ode78` and relative tolerances

The integrator employed for this task is `ode78`, a high-order solver with fixed step sizes. This solver is particularly well-suited for long-duration integrations due to its robustness, stability, and numerical reliability in scenarios characterized by smooth and predictable dynamics. Designed for non-stiff problems, `ode78` is ideal for applications such as spacecraft trajectory guidance, where the dynamics are not subject to rapid or abrupt changes. The choice of this fixed-step algorithm ensures that the convergence behavior is solely attributed to the optimization methods under investigation, not enhanced by the ode solver used. This choice, thereby, isolates and validates the inherent properties of the algorithms being analyzed. The integration is performed with Relative Tolerance set to 10^{-12}, and specific absolute tolerances for different quantities to balance computational precision and efficiency. For the position $\mathbf{r}$, the tolerance is set to $\text{Tol}_{\mathbf{r}} = 10^{-10}$, providing sufficient accuracy without imposing excessive computational demands. The velocity $\mathbf{v}$ requires higher precision due to its direct influence on the dynamics, and thus its tolerance is set to $\text{Tol}_{\mathbf{v}} = 10^{-13}$. For the components of the state transition matrix (STM) $\mathbf{\Phi}$, the tolerance is chosen as $\text{Tol}_{\mathbf{\Phi}} = 10^{-13}$. While the STM is crucial for understanding the trajectory's evolution, extreme precision is deemed unnecessary because the perturbations it experiences over time are relatively small. These tolerances are carefully selected to ensure computational resources are allocated effectively and efficiently.

A.1.3 Differential Correction

This paragraph deepens the model developed to reach the differential correction used for the problem subsubsection 1.2.1.
In the context of the final state $\mathbf{S} = [x, y, z, v_x, v_y, v_z]^T = [\mathbf{r}, \mathbf{v}]^T$, the following equation holds:

$$\mathbf{S} = \varphi(\mathbf{S_0}, t_0; t),$$

where $\varphi(\mathbf{S_0}, t_0; t)$ represents the flow, a function of the initial state $\mathbf{S_0} = [x_0, y_0, z_0, v_{x0}, v_{y0}, v_{z0}]^T = [\mathbf{r_0}, \mathbf{v_0}]^T$, the initial time t_0, and the final time t. The Jacobi constant at the final time, $C = C(\mathbf{S})$, is a function of the state $\mathbf{S}$ alone.
By linearly approximating the evolution of small perturbations in the state and the Jacobi constant, we expand the flow both in terms of state and time, and the Jacobi constant only in terms of the state. This leads to the following system:

$$\begin{cases} \mathbf{S}(\varphi(\mathbf{S_0}+\Delta\mathbf{S_0}, t+\Delta t)): & \delta\mathbf{S} = \dfrac{\partial\varphi}{\partial\mathbf{S_0}}\delta\mathbf{S_0} + \dfrac{\partial\varphi}{\partial t}\delta t, \\ C(\mathbf{S_0}+\Delta\mathbf{S_0}): & \delta C = \dfrac{\partial C}{\partial\mathbf{S_0}}\delta\mathbf{S_0}. \end{cases} \tag{30}$$

Note that $\dfrac{\partial\varphi}{\partial\mathbf{S_0}} = \mathbf{\Phi}(t_0, t)$ is the State Transition Matrix (STM), which links the state at t_0 to the state at t, and $\dfrac{\partial\varphi}{\partial t} = \mathbf{f}^T$, where $\mathbf{f}$ corresponds to the dynamics evaluated at the final time. Furthermore, the partial derivative of the Jacobi constant with respect to the initial state is given by:

$$\frac{\partial C}{\partial\mathbf{S_0}} = \left[2\frac{\partial\Omega}{\partial x}(x,y,z), 2\frac{\partial\Omega}{\partial y}(x,y,z), 2\frac{\partial\Omega}{\partial z}(x,y,z), -2v_x, -2v_y, -2v_z\right], \tag{31}$$

which can be computed directly from the definition of the Jacobi constant Equation 1 and the 3D CRTBP potential $\Omega(x, y, z)$.
By defining $\delta\mathbf{S}$ and δC as the errors in the final state $\mathbf{S}$ and the corresponding Jacobi constant C relative to a reference condition $\mathbf{S_{ref}}$ and C_{ref}, we have the following relations:

$$\begin{bmatrix}\partial\mathbf{S}\\ \partial C\end{bmatrix} = \begin{bmatrix}\mathbf{S_{ref}}\\ C_{ref}\end{bmatrix} - \begin{bmatrix}\mathbf{S}\\ C\end{bmatrix} \Rightarrow \begin{bmatrix}\partial\mathbf{S}\\ \partial C\end{bmatrix} = \begin{bmatrix} x_{\text{ref}} - x \\ y_{\text{ref}} - y \\ z_{\text{ref}} - z \\ v_{x_{\text{ref}}} - v_x \\ v_{y_{\text{ref}}} - v_y \\ v_{z_{\text{ref}}} - v_z \\ C_{\text{ref}} - C \end{bmatrix} \tag{32}$$

Here, $\delta\mathbf{S_0}$ and δt represent the corrections to be applied to the initial state and final time to minimize the error in the final state for a reference condition. Thus, the following correction equations hold:

$$\begin{bmatrix}\delta\mathbf{S_0}\\ \delta t\end{bmatrix} = \begin{bmatrix} \mathbf{\Phi}_{(6\times6)} & \dfrac{\partial\varphi}{\partial t}_{(1\times6)} \\ \dfrac{\partial C}{\partial\mathbf{S}}_{(6\times1)} & 0_{(1\times1)} \end{bmatrix} \begin{bmatrix} \delta x_0 \\ \delta y_0 \\ \delta z_0 \\ \delta v_{x0} \\ \delta v_{y0} \\ \delta v_{z0} \\ \delta t \end{bmatrix} = \begin{bmatrix} x_{0,\text{new}} - x_{0,\text{old}} \\ y_{0,\text{new}} - y_{0,\text{old}} \\ z_{0,\text{new}} - z_{0,\text{old}} \\ v_{x_{0,\text{new}}} - v_{x_{0,\text{old}}} \\ v_{y_{0,\text{new}}} - v_{y_{0,\text{old}}} \\ v_{z_{0,\text{new}}} - v_{z_{0,\text{old}}} \\ t_{\text{new}} - t_{\text{old}} \end{bmatrix} \tag{33}$$

By substituting Equation 31, Equation 32, and Equation 33 into the original system, we can rewrite the Equation 35 in matrix form as follows:

$$\begin{bmatrix} \mathbf{\Phi}_{(6\times6)} & \dfrac{\partial\varphi}{\partial t}_{(1\times6)} \\ \dfrac{\partial C}{\partial\mathbf{S}}_{(6\times1)} & 0_{(1\times1)} \end{bmatrix} \begin{bmatrix} x_{0,\text{new}} - x_{0,\text{old}} \\ y_{0,\text{new}} - y_{0,\text{old}} \\ z_{0,\text{new}} - z_{0,\text{old}} \\ v_{x_{0,\text{new}}} - v_{x_{0,\text{old}}} \\ v_{y_{0,\text{new}}} - v_{y_{0,\text{old}}} \\ v_{z_{0,\text{new}}} - v_{z_{0,\text{old}}} \\ t_{\text{new}} - t_{\text{old}} \end{bmatrix} = \begin{bmatrix} x_{\text{ref}} - x \\ y_{\text{ref}} - y \\ z_{\text{ref}} - z \\ v_{x_{\text{ref}}} - v_x \\ v_{y_{\text{ref}}} - v_y \\ v_{z_{\text{ref}}} - v_z \\ C_{\text{ref}} - C \end{bmatrix} \Rightarrow \tag{34}$$

This system can be simplified by recalling that there are only conditions on $\mathbf{\Delta} = [\delta x_0, \delta z_0, \delta v_{y0}, \delta t]^T$ elements of the unknown vector, referred to the $\mathbf{b} = [y_{\text{ref}} - y_0, v_{x_{\text{ref}}} - v_{x0}, v_{z_{\text{ref}}} - v_{z0}, C_{\text{ref}} - C]$ known quantities. Moreover, the perpendicular crossing of the $y = 0$ coordinate means that y_{ref}, $v_{x_{\text{ref}}}$, and $v_{z_{\text{ref}}}$ must all be null quantities, while $C_{\text{ref}} = 3.09$. So, the system to be solved is:

$$\mathbf{B}\boldsymbol{\Delta} = \mathbf{b} \tag{35}$$

Where:

$$\mathbf{B} = \begin{bmatrix} \Phi_{2,1} & \Phi_{2,3} & \Phi_{2,5} & \dot{y}_f \\ \Phi_{4,1} & \Phi_{4,3} & \Phi_{4,5} & \dfrac{\partial \Omega}{\partial x} + 2\dot{y} \\ \Phi_{6,1} & \Phi_{6,3} & \Phi_{6,5} & \dfrac{\partial \Omega}{\partial z} \\ \dfrac{\partial C}{\partial x} & \dfrac{\partial C}{\partial z} & \dfrac{\partial C}{\partial \dot{y}} & 0 \end{bmatrix}; \qquad \mathbf{b} = \begin{bmatrix} 0 - y_f \\ 0 - \dot{x}_f \\ 0 - \dot{z}_f \\ C_{\text{ref}} - C(\mathbf{S}_f) \end{bmatrix}, \tag{36}$$

Where $\Phi_{i,j}$ with $i = 2, 4, 6$ and $j = 1, 3, 5$ denote the elements of the State Transition Matrix (STM) evaluated at the final state $\mathbf{S}$ after propagation.

A.2 Appendix: Exercise 2

A.2.1 2D PBRFBP dynamics in rotating frame

This paragraph deepens the two-dimensional Planar Bicircular Restricted Four Body Problem (2D PBRFBP) in the rotating frame for the problem exposed in section 2.
The 2D equations of motion for the negligible mass in the rotating frame are:

$$\ddot{x} = \frac{\partial \Omega_4}{\partial x} + 2\dot{y}, \quad \ddot{y} = \frac{\partial \Omega_4}{\partial y} - 2\dot{x},$$

where x and y denote the particle's coordinates in the rotating frame. The effective potential of the four-body system, $\Omega_4(x, y, t)$, is defined as:

$$\Omega_4(x, y, t) = \Omega_3(x, y) + \frac{m_s}{r_3(t)} - \frac{m_s}{\rho^2}\left(x\cos(\omega_s t) + y\sin(\omega_s t)\right), \tag{37}$$

where Ω_3 represents the potential of the CR3BP.
In this model, t is the time, m_s is the Sun's mass, ρ is the distance between the Sun and the barycenter of the Earth–Moon system, and ω_s is the Sun's angular velocity in the rotating reference frame. The Sun's phase angle is given by $\theta(t) = \omega_s t$, and its location at time t is $(\rho\cos(\omega_s t), \rho\sin(\omega_s t))$. Using this position, the distance between the Sun and the particle is:

$$r_3(t) = \sqrt{(x - \rho\cos(\omega_s t))^2 + (y - \rho\sin(\omega_s t))^2}. \tag{38}$$

It is important to highlight that in this framework, the Sun is not stationary within the chosen reference frame. As a result, the equations of motion depend explicitly on time, rendering the Planar Bicircular Restricted Four-Body Problem (PBRFBP) a non-autonomous system. A table summarizing the system's constants and their units (e.g., distance, time, and velocity) can be found in Table 4.
The full treatment is in [3]. Here, only the final expression of the dynamics of the 2D PBRFBP is reported:

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{v_x} = \dfrac{\partial \Omega_4}{\partial x} + 2v_y \\ \dot{v_y} = \dfrac{\partial \Omega_4}{\partial y} - 2v_x \end{cases} \quad \text{with initial conditions:} \quad \begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \\ v_x(t_0) = v_{x0} \\ v_y(t_0) = v_{y0} \end{cases} \tag{39}$$

Rewriting the system in matrix form:

$$\dot{\mathbf{S}} = \left[v_x, v_y, \frac{\partial \Omega_4(x,y,t)}{\partial x} + 2v_y, \frac{\partial \Omega_4(x,y,t)}{\partial y} - 2v_x \right]^T = \mathbf{f_4}(\mathbf{S}, t), \tag{40}$$

Where $\mathbf{S} = [x, y, v_x, v_y]^T$ is the state vector, and $\mathbf{f_4}(\mathbf{S}, t)$ is the vector of the dynamics of the 2D PBRFBP.
The STM from initial time t_0 to final time t can be found by solving the following 16 scalar ODEs:

$$\begin{cases} \text{Variational equation :} & \dot{\mathbf{\Phi}}(t_0, t) = \mathbf{A_4}(t)\mathbf{\Phi}(t_0, t) \\ \text{Initial conditions :} & \mathbf{\Phi}(t_0, t_0) = \mathbf{I} \end{cases} \tag{41}$$

Where $\mathbf{I} \in \Re^{4\times4}$ is the identity matrix, and the matrix $\mathbf{A}(t)$ can be retrieved symbolically by computing the Jacobian matrix of the dynamics over the state, that is $\mathbf{A}(t) = \dfrac{d\mathbf{f}}{d\mathbf{S}}$. The result is reported here:

$$\mathbf{A_4}(t) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \Omega_{4,xx} & \Omega_{4,xy} & 0 & 2 \\ \Omega_{4,yx} & \Omega_{4,yy} & -2 & 0 \end{bmatrix},$$

where:

$$\Omega_{4,xx} = \frac{\mu-1}{r_1^3} - \frac{m_s}{r_3(t)^3} - \frac{\mu}{r_2^3} - \frac{3(\mu+x)^2(\mu-1)}{r_1^5} + \frac{3\mu(\mu+x-1)^2}{r_2^5} + \frac{3m_s(x-\rho\cos(\omega_s t))^2}{r_3(t)^5} + 1;$$
$$\Omega_{4,xy} = \frac{3\mu y(\mu+x-1)}{r_2^5} + \frac{3m_s(x-\rho\cos(\omega_s t))(y-\rho\sin(\omega_s t))}{r_3(t)^5} - \frac{3y(\mu+x)(\mu-1)}{r_1^5} = \Omega_{4,yx};$$
$$\Omega_{4,yy} = \frac{\mu-1}{r_1^3} - \frac{m_s}{r_3(t)^3} - \frac{\mu}{r_2^3} - \frac{3y^2(\mu-1)}{r_1^5} + \frac{3m_s(y-\rho\sin(\omega_s t))^2}{r_3(t)^5} + \frac{3\mu y^2}{r_2^5} + 1.$$

Where $r_1 = \sqrt{(x+\mu)^2 + y^2}$ and $r_2 = \sqrt{(x+\mu-1)^2 + y^2}$.

A.2.2 Model of `ROT2ECI` function

The equations used to convert from the rotating reference frame to the Earth-Centred Inertial (ECI) reference frame are the ones in [3], whose mathematical expression is here reported. An orbit described in the rotating frame, $\mathbf{x(t)} = \{x(t), y(t), \dot{x}(t), \dot{y}(t)\}$, can be transformed into an orbit represented in the P_1-centred (Earth-centered) inertial frame, $\mathbf{X_1(t)} = \{X_1(t), Y_1(t), \dot{X}_1(t), \dot{Y}_1(t)\}$, using the following relationships:

$$\begin{cases} X_1(t) = \big(x(t)+\mu\big)\cos t - y(t)\sin t, \\ Y_1(t) = \big(x(t)+\mu\big)\sin t + y(t)\cos t, \\ \dot{X}_1(t) = \big(\dot{x}(t) - y(t)\big)\cos t - \big(\dot{y}(t) + x(t) + \mu\big)\sin t, \\ \dot{Y}_1(t) = \big(\dot{x}(t) - y(t)\big)\sin t + \big(\dot{y}(t) + x(t) + \mu\big)\cos t. \end{cases} \tag{42}$$

Here, t refers to the current scaled time. To perform the transformation into the P_2-centred (Moon-centered) inertial frame, equation (34) is used with the substitution $\mu \to \mu - 1$.

A.2.3 Analytical form of the gradients for the simple shooting problem

In this paragraph the analytic derivatives of the objective function and the constraints with respect to the NLP variables for the simple shooting in the optimization problem in subsubsection 2.2.1 are treated.

The **objective function gradient** with respect to the augmented state vector $\mathbf{AS_0} = [x_i, y_i, v_{xi}, v_{yi}, t_i, t_f]^T = [\mathbf{S_i}, t_i, t_f]^T$ is retrieved as:

$$\mathbf{G_f} = \frac{\partial f}{\partial \mathbf{AS_0}} = \left[\frac{\partial f}{\partial x_i}, \frac{\partial f}{\partial y_i}, \frac{\partial f}{\partial v_{xi}}, \frac{\partial f}{\partial v_{yi}}, \frac{\partial f}{\partial t_i}, \frac{\partial f}{\partial t_f}\right] = \left[\frac{\partial f}{\partial \mathbf{S_i}}, \frac{\partial f}{\partial t_i}, \frac{\partial f}{\partial t_f}\right]^T$$

Where f is defined as in Equation 6, and its derivatives can be computed as:

$$\frac{\partial f}{\partial \mathbf{S_i}} = \frac{\partial(\Delta v_i)}{\partial \mathbf{S_i}} + \frac{\partial(\Delta v_f)}{\partial \mathbf{S_i}} = \frac{\partial(\Delta v_i)}{\partial \mathbf{S_i}} + \frac{\partial(\Delta v_f)}{\partial \boldsymbol{\varphi}} \cdot \frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{S_i}} = \frac{\partial(\Delta v_i)}{\partial \mathbf{S_i}} + \frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}} \boldsymbol{\Phi}(t_i, t_f)$$

$$\frac{\partial f}{\partial t_i} = \frac{\partial(\Delta v_i)}{\partial t_i} + \frac{\partial(\Delta v_f)}{\partial t_f} = \frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}} \cdot \frac{\partial \varphi}{\partial t_i} = -\frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}} \boldsymbol{\Phi}(t_i, t_f)\, \mathbf{f_4}(\mathbf{S_i}, t_i)$$

$$\frac{\partial f}{\partial t_f} = \frac{\partial(\Delta v_i)}{\partial t_f} + \frac{\partial(\Delta v_f)}{\partial t_f} = \frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}} \cdot \frac{\partial \varphi}{\partial t_f} = \frac{\partial(\Delta v_f)}{\partial \mathbf{S_f}} \mathbf{f_4}(\mathbf{S_f}, t_f)$$

Where the red means that those terms can be computed analytically (or symbolically), while the blue means that those terms are available from previous computations. Additionally, from the statement of the problem, since Δv_i does not depend on t_i and t_f, the following equations hold:

$$\frac{\partial \Delta v_i}{\partial t_i} = \frac{\partial \Delta v_i}{\partial t_f} = 0$$

$\boldsymbol{\Phi}(t_i, t_f)$ is the STM from time t_i to time t_f, as defined in Equation 41 in subsubsection A.2.1. The term $\mathbf{f_4}(x, t)$ is the vector representing the dynamics, as stated in Equation 40 in subsubsection A.2.1.

The **equality constraint gradient** with respect to the augmented state vector $\mathbf{AS_0}$ can be defined as follows:

$$\mathbf{G_{c_{eq}}} = \frac{\partial \mathbf{c_{eq}}}{\partial \mathbf{AS}} = \left[\frac{\partial \mathbf{c_{eq}}}{\partial \mathbf{S_i}}, \frac{\partial \mathbf{c_{eq}}}{\partial t_i}, \frac{\partial \mathbf{c_{eq}}}{\partial t_f}\right]^T \tag{43}$$

Recall that $\mathbf{c_{eq}} = [\boldsymbol{\Psi_i}, \boldsymbol{\Psi_f}]^T$. Therefore:

$$\frac{\partial \mathbf{c_{eq}}}{\partial \mathbf{S_i}} = \begin{bmatrix} \frac{\partial \boldsymbol{\Psi_i}}{\partial \mathbf{S_i}} \\ \frac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}} \frac{\partial \varphi}{\partial \mathbf{S_i}} \end{bmatrix} = \begin{bmatrix} \frac{\partial \boldsymbol{\Psi_i}}{\partial \mathbf{S_i}} \\ \frac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}} \boldsymbol{\Phi}(t_i, t_f) \end{bmatrix},$$

$$\frac{\partial \mathbf{c_{eq}}}{\partial t_i} = \begin{bmatrix} \frac{\partial \boldsymbol{\Psi_i}}{\partial t_i} \\ \frac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}} \frac{\partial \varphi}{\partial t_i} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -\frac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}} \boldsymbol{\Phi}(t_i, t_f)\, \mathbf{f_4}(\mathbf{S_i}, t_i) \end{bmatrix}$$

$$\frac{\partial \mathbf{c_{eq}}}{\partial t_f} = \begin{bmatrix} \mathbf{0} \\ \frac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}} \frac{\partial \varphi}{\partial t_i} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \frac{\partial \boldsymbol{\Psi_f}}{\partial \mathbf{S_f}} \mathbf{f_4}(\mathbf{S_f}, t_f) \end{bmatrix}$$

Where $\frac{\partial \boldsymbol{\Psi_i}}{\partial t_i} = \frac{\partial \boldsymbol{\Psi_i}}{\partial t_f} = \mathbf{0}$, since, as already stated, $\boldsymbol{\Psi_i} = \boldsymbol{\Psi_i}(\mathbf{S_i})$, and the color code is the same as in the previous case.

A.2.4 Rotation matrix from Earth centre ECLIPJ2000 to Moon's orbital frame

$$\mathbf{x}_{\text{Moon}} = \frac{\mathbf{r}}{\|\mathbf{r}\|} = \begin{bmatrix} x_{\text{Moon},x} \\ x_{\text{Moon},y} \\ x_{\text{Moon},z} \end{bmatrix}, \quad \text{unit vector along the } x\text{-axis.}$$

$$\mathbf{z}_{\text{Moon}} = \frac{\mathbf{r} \times \mathbf{v}}{\|\mathbf{r} \times \mathbf{v}\|} = \begin{bmatrix} z_{\text{Moon},x} \\ z_{\text{Moon},y} \\ z_{\text{Moon},z} \end{bmatrix}, \quad \text{unit vector along the } z\text{-axis.}$$

$$\mathbf{y}_{\text{Moon}} = \frac{\mathbf{x} \times (-\mathbf{z})}{\|\mathbf{x} \times (-\mathbf{z})\|} = \begin{bmatrix} y_{\text{Moon},x} \\ y_{\text{Moon},y} \\ y_{\text{Moon},z} \end{bmatrix}, \quad \text{unit vector along the } y\text{-axis.}$$

The rotation matrix is the following:

$$T_{ECLIP2000 \to Moon} = \begin{bmatrix} x_{\text{Moon},x} & x_{\text{Moon},y} & x_{\text{Moon},z} \\ y_{\text{Moon},x} & y_{\text{Moon},y} & y_{\text{Moon},z} \\ z_{\text{Moon},x} & z_{\text{Moon},y} & z_{\text{Moon},z} \end{bmatrix}$$

A.2.5 Rotation Matrix: Earth-Moon Rotating Frame to ECLIPJ2000

This paragraph is about deriving the transformation matrix from the Earth-Moon Rotating Reference Frame to the ECLIPJ2000 frame. This is taken from [2].
First, the PBRFBP trajectory, originally expressed in the rotating frame, is normalized and centered on the Earth. An additional spatial component is appended to represent the trajectory in three dimensions:

$$\mathbf{Y} = [(x + \mu), y, z, \dot{x} - VU, \dot{y}, \dot{z}]. \tag{44}$$

Time in the trajectory, expressed in TU, is converted to ephemeris time to synchronize with celestial mechanics:

$$t = (t_t - t_0) \cdot \text{TU} + t_{\text{start}}. \tag{45}$$

Using ephemeris data, the Moon's position and velocity relative to Earth are obtained to define the rotating frame axes:

- $\hat{X}$: aligned with the Moon's position vector,
- $\hat{Y}$: in-plane and orthogonal to $\hat{X}$,
- $\hat{Z}$: perpendicular to the plane defined by $\hat{X}$ and $\hat{Y}$.

The transformation matrix $\mathbf{R}$, mapping coordinates from the rotating frame to the inertial frame, is constructed as:

$$\mathbf{R} = \begin{bmatrix} \hat{X} & \hat{Y} & \hat{Z} \end{bmatrix}. \tag{46}$$

The position in the ECLIPJ2000 frame is then given by:

$$\mathbf{Y}_{\text{ECLIPJ2000}} = \mathbf{R} \cdot \mathbf{Y}_{\text{rot}}. \tag{47}$$

To determine the velocity in the inertial frame, the instantaneous angular velocity of the rotating frame is calculated as:

$$\boldsymbol{\omega} = \frac{\mathbf{V}_{\text{Moon}} \times \mathbf{X}_{\text{Moon}}}{\|\mathbf{X}_{\text{Moon}}\|^2}. \tag{48}$$

Finally, the inertial velocity is computed as:

$$\mathbf{V}_{\text{ECLIPJ2000}} = \mathbf{R} \cdot \mathbf{V}_{\text{rot}} + \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \cdot \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \cdot \mathbf{Y}_{\text{rot}}. \tag{49}$$

For each trajectory step, the state vector and velocity are transformed into the inertial ECLIPJ2000 frame. This transformation initializes the n-body propagation with initial states expressed in the inertial frame.

A.2.6 Real and scaled trajectory for continuous guidance with reduced thrust level

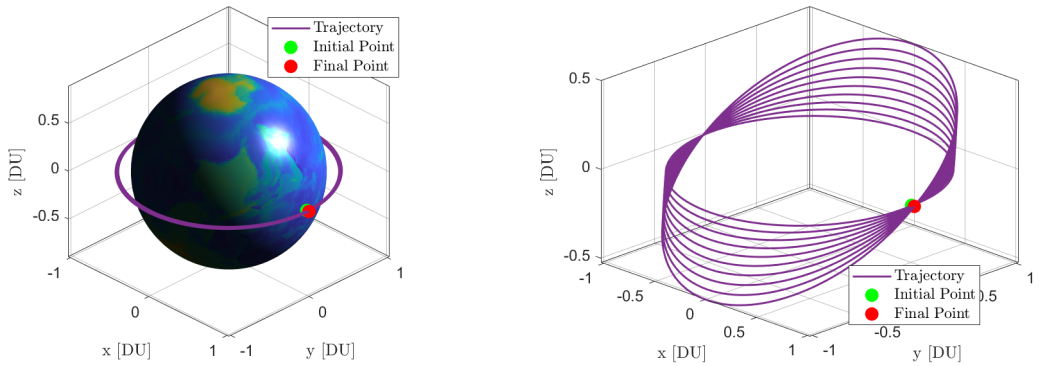


Figure 19: Non-scaled (on the left) and properly scaled (on the right) optimised trajectories in Eart-centred J2000

References

[1] K. Oshima, F. Topputo, and T. Yanao. "Low-energy transfers to the Moon with long transfer time". In: *Celestial Mechanics and Dynamical Astronomy* 131.1 (Jan. 2019). DOI: 10.1007/s10569-019-9883-7 (cit. on pp. 11, 12).

[2] T. A. Pavlak. "Trajectory design and orbit maintenance strategies in multi-body dynamical regimes". In: *PhDT* (Jan. 2013) (cit. on p. 31).

[3] F. Topputo. "On optimal two-impulse Earth-Moon transfers in a four-body model". In: *Celestial Mechanics and Dynamical Astronomy* 117 (2013), pp. 279–313. DOI: 10.1007/s10569-013-9513-8 (cit. on pp. 8, 11, 15, 28, 29).