Math 438 Project: Optimal Spacecraft Control

Sam Cochran, Jacob Murri, Caleb Wilson

April 12, 2021

Abstract

Space exploration has become an essential mechanism for scientific discovery, and many expect the importance of space exploration to extend to other fields in the future. As spacecraft-designing capability grows, the development of methods to find optimal spacecraft trajectories between two points becomes more important as well. However, this problem is a nontrivial one because the equations of motion for n bodies under gravitational interaction are notoriously hard to solve, and it is even harder to solve optimal control formulations which include spacecraft thrust. In this paper we present code which uses scipy's `solve_bvp` to solve a formulation of the spacecraft optimal control problem obtained through Pontryagin's maximum principle, under the assumptions of the Tsiolkovsky rocket equation. While somewhat sensitive to initial guesses, our method is able to find optimal solutions for the problem in many different configurations of planetary orbits.

Keywords: Pontryagin's Maximum Principle, n-body problem, spacecraft control, slingshot effect

1 Background

One of the most interesting control problems is that of controlling a spacecraft. Because spacecraft set out with a limited amount of fuel, and the amount of thrust the fuel provides changes with the amount of fuel that has been used up (due to the change in mass of the spacecraft), the problem of arriving at a destination in the minimal amount of time is nontrivial. The problem is further complicated by the presence of other moving bodies in the system.

For this project, we will solve an optical control problem related to spacecraft trajectories and the gravitational slingshot effect. We will start with a simple model, and work our way up to a more complicated slingshot model, incorporating ideas of control theory to the trajectories we produce.

We will implement a cost functional that enforces the various conditions we want in our problem. Specifically, we want to punish high fuel use, long travel times and acceleration values that are too high. We will also implement constraints to ensure that we actually arrive at our final destination, and that we arrive with the desired final velocity.

The end goal is to develop a realistic model of how to get a spacecraft from point A to point B in the solar system using a set amount of fuel, and in the fastest time possible given that amount of fuel.

1.1 Model Assumptions

In this model we assume that both the spacecraft and the bodies that gravitationally influence it can be well-approximated by point masses. We furthermore assume that the spacecraft can change the amount or direction of its acceleration instantaneously (i.e. we place no constraint on jerk). These are both unrealistic assumptions but we believe that they represent good approximations, and enable our solution to be computationally feasible. We also assume that the mass of the spacecraft is too small to gravitationally affect the other bodies in the system.

2 Mathematical Representation

2.1 State Equations

Let $\mathbf{x} : \mathbf{R} \to \mathbf{R}^2$ represent the parametrized path of a spacecraft over time. We impose the initial condition that $\mathbf{x}(0) = \mathbf{x}_0$ and the final condition that $\mathbf{x}(T) \approx \mathbf{x}_f$, where T is an unknown final time, and $\mathbf{x}_0$ and $\mathbf{x}_f$ are initial and desired final positions, respectively. We also an initial velocity condition $\mathbf{x}'(0) = \mathbf{v}_0$, and explore adding a Bolza cost term for requiring the final velocity $\mathbf{x}'(T) \approx \mathbf{v}_f$. Our spacecraft carries some fuel to provide thrust which we write as a Euclidean vector control $\mathbf{u}(t) = (u_1(t), u_2(t))^T$. We denote our rate of fuel use by the magnitude of this vector; $u(t) = \|\mathbf{u}\|$. Then, using the Tsiolkovsky rocket equation (or the conservation of momentum), the acceleration $\mathbf{a}(t)$ of our spacecraft due to our thrust control is given by

$$\mathbf{a}(t) = \frac{v_e \mathbf{u}(t)}{m(t)}$$

where $m(t)$ is a state variable representing the mass of our spacecraft over time, and v_e is the velocity of the fuel as it exits the spacecraft. If m_s is the total mass of our spacecraft at time zero (including the mass of fuel we start with), then $m(t)$ evolves according to the differential equation

$$\begin{cases} m'(t) = -u(t) = -\|\mathbf{u}(t)\|, & t \in [0, T] \\ m(0) = m_s \end{cases}$$

because as fuel is burned, the rocket gets lighter. We can leave the final mass $m(T)$ free as long as we put a cost on the acceleration, because as $m(t) \to 0, a(t) \to \infty$ (assuming we are still burning fuel). If we don't burn any more fuel, then the mass stays the same, hence our cost can prevent our mass from going to 0. Now that we have written our acceleration in terms of our control

parameters u_1 and u_1, we can write the full state equation of our system. We assume that the spacecraft interacts with bodies with masses $M_1, M_2, \ldots, M_n$ with positions $\mathbf{r}_1, \ldots, \mathbf{r}_n$. Then using Newton's laws, our state equation for the spacecraft position is

$$\begin{aligned}\mathbf{x}''(t) &= \mathbf{a}(t) + \sum_{i=1}^{n} GM_i \frac{\mathbf{r}_i - \mathbf{x}}{\|\mathbf{r}_i - \mathbf{x}\|^3} \\ &= \frac{v_e \mathbf{u}(t)}{m(t)} + \sum_{i=1}^{n} GM_i \frac{\mathbf{r}_i - \mathbf{x}}{\|\mathbf{r}_i - \mathbf{x}\|^3}\end{aligned}$$

(where G is the universal gravitational constant), subject to the initial conditions $\mathbf{x}(0) = \mathbf{x}_0, \mathbf{x}'(0) = \mathbf{v}_0$.

2.2 Cost Functional

Because we want our spacecraft to minimize the amount of time it takes to reach its destination, we include a constant term in the cost functional that will penalize taking more time. However, we do not want to accelerate too much or change direction too fast, so as to avoid disturbing the contents of the spacecraft (whether cargo or people). Hence we add the cost term $\alpha a(t)^2 = \alpha \|\mathbf{a}(t)\|^2$ to our cost functional, where $\alpha > 0$ is a weight that we can choose later. We also want to minimize our fuel use, so we add the term $\beta u(t)^2 = \beta \|\mathbf{u}(t)\|^2 = \beta(u_1(t))^2 + (u_2(t))^2$ to the cost functional, where $\beta > 0$ is another weight that we can choose later. We also want to arrive at our destination $\mathbf{x}_f$ with final velocity $\mathbf{v}_f$ at time T, so we add a term outside of the integral to put our functional in Bolza form. We also add the weight coefficients δ_1 and δ_2 to implement these final constraints. This yields the cost functional

$$J[\mathbf{u}] = \int_0^T 1 + \alpha \|\mathbf{a}(t)\|^2 + \beta \|\mathbf{u}(t)\|^2 \, dt + \delta_1 \|\mathbf{x}(T) - \mathbf{x}_f\|^2 + \delta_2 \|\mathbf{x}' - \mathbf{v}_f\|^2,$$

where $\|\mathbf{a}(t)\|$ is the magnitude of the acceleration (as defined above) which depends only on $\mathbf{u}(t)$, and $\mathbf{u}(t)$ is the rate of fuel use.

In summary, our cost functional and constraint will minimize the amount of time it takes to approximately reach our destination at approximately the desired velocity subject to our fuel constraints. The cost functional will also penalize large accelerations and fuel usage. It is important to note that these constraints only apply to our control parameters, so changes in direction and acceleration due to gravity are not subject to these constraints.

3 Solution

3.1 Derivation

Since we are dealing with the very nonlinear gravity equations we cannot use LQR to solve this problem. We will use Pontryagin's maximum principle, noting

that the Hamiltonian nature of the N-body problem upon which we base our state equations lends itself nicely to this method.

We first need to write our state equation as a first order equation in the form $\mathbf{y}' = f(\mathbf{y}, u, \theta)$. We first write the state equation as a first order equation:

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{x}' \\ m \end{bmatrix}' = f\left(\begin{bmatrix} \mathbf{x} \\ \mathbf{x}' \\ m \end{bmatrix}, \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \right) = \begin{bmatrix} \mathbf{x}' \\ \frac{v_e \mathbf{u}(t)}{m(t)} + \sum_{i=1}^{n} GM_i \frac{\mathbf{r}_i - \mathbf{x}}{\|\mathbf{r}_i - \mathbf{x}\|^3} \\ -\|\mathbf{u}\| \end{bmatrix}$$

Now, we let $\mathbf{y}$ as a vector be the concatenation of the position $\mathbf{x}$, the velocity $\mathbf{x}'$, and the mass m, so that

$$\mathbf{y}' = \begin{bmatrix} \mathbf{y}_1' \\ \mathbf{y}_2' \\ m' \end{bmatrix} = f\left(\mathbf{y}, \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \right) = \begin{bmatrix} \mathbf{y}_2 \\ \frac{v_e \mathbf{u}(t)}{m(t)} + \sum_{i=1}^{n} GM_i \frac{\mathbf{r}_i - \mathbf{y}_1}{\|\mathbf{r}_i - \mathbf{y}_1\|^3} \\ -\|\mathbf{u}\| \end{bmatrix}$$

where $\mathbf{y}_1 = \mathbf{x}$ and $\mathbf{y}_2 = \mathbf{x}'$ as explained above. Here we note that we have assumed that the spacecraft is too small to significantly influence the other bodies, so the positions $\mathbf{r}_i$ of the bodies can be solved for beforehand as an initial value problem (see Section 3.2), and this equation is really just an equation in $\mathbf{y} = [\mathbf{x}, \mathbf{x}', m]^T$.

We now write the Hamiltonian of our system:

$$H = \mathbf{p} \cdot f\left(\mathbf{y}, \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \right) - \mathcal{L}\left(\mathbf{y}, \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \right)$$

where

$$\mathcal{L}\left(\mathbf{y}, \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \right) = 1 + \alpha \frac{v_e}{m(t)} \|\mathbf{u}(t)\|^2 + \beta \|\mathbf{u}(t)\|^2.$$

so

$$H = \mathbf{p} \cdot \begin{bmatrix} \mathbf{y}_2 \\ \frac{v_e \mathbf{u}(t)}{m(t)} + \sum_{i=1}^{n} GM_i \frac{\mathbf{r}_i - \mathbf{y}_1}{\|\mathbf{r}_i - \mathbf{y}_1\|^3} \\ -\|\mathbf{u}(t)\| \end{bmatrix} - 1 - \alpha \frac{v_e}{m(t)} \|\mathbf{u}(t)\|^2 - \beta \|\mathbf{u}(t)\|^2$$

We can use Pontryagin's maximum principle to find the optimal control and optimal trajectory. We have already defined our state equation, and we define $\mathbf{p} = (p_1, p_2, p_3, p_4, p_5)^T$ to satisfy our costate equation:

$$\mathbf{p}' = -\frac{DH}{D\mathbf{y}} = -\mathbf{p} \cdot \frac{Df}{D\mathbf{y}} + \frac{D\mathcal{L}}{D\mathbf{y}} = -\begin{bmatrix} \mathbf{0} & I & \mathbf{0} \\ \frac{d}{d\mathbf{y}_1} \sum_{i=1}^{n} GM_i \frac{\mathbf{r}_i - \mathbf{y}_1}{\|\mathbf{r}_i - \mathbf{y}_1\|^3} & \mathbf{0} & \frac{-v_e \mathbf{u}(t)}{(m(t))^2} \\ 0 & \mathbf{0}^T & 0 \end{bmatrix} \mathbf{p}$$

Where

$$\frac{d}{d\mathbf{y}_1}\sum_{i=1}^{n} GM_i\frac{\mathbf{r}_i-\mathbf{y}_1}{\|\mathbf{r}_i-\mathbf{y}_1\|^3} = \sum_{i=1}^{n}[GM_i\frac{-I}{\|\mathbf{r}_i-\mathbf{y}_1\|^3} + GM_i\frac{d}{d\mathbf{y}_1}(\frac{1}{\|\mathbf{r}_i-\mathbf{y}_1\|^3})(\mathbf{r}_i-\mathbf{y}_1)^T]$$
$$= \sum_{i=1}^{n}[GM_i\frac{-I}{\|\mathbf{r}_i-\mathbf{y}_1\|^3} + GM_i\frac{3(\mathbf{r}_i-\mathbf{y}_1)}{\|\mathbf{r}_i-\mathbf{y}_1\|^5}(\mathbf{r}_i-\mathbf{y}_1)^T]$$
$$= \sum_{i=1}^{n}\frac{GM_i}{\|\mathbf{r}_i-\mathbf{y}_1\|^3}(-I+\frac{3(\mathbf{r}_i-\mathbf{y}_1)(\mathbf{r}_i-\mathbf{y}_1)^T}{\|\mathbf{r}_i-\mathbf{y}_1\|^2}).$$

Note that by Pontryagin's maximum principle, allowing our final mass $m(T)$ to be free induces the endpoint condition that $p_5(T)=0$. In the costate equation we have that $p_5'=0$, meaning that $p_5(t)=0$ for all $t\in[0,T]$. As we will see, this free final mass assumption and the resulting fact that p_5 identically vanishes allows us to express the optimal control $\tilde{\mathbf{u}}$ as a linear function of the costate $\mathbf{p}$. This is because Pontryagin's maximum principle implies that $\tilde{\mathbf{u}}=[\tilde{u}_1,\tilde{u}_2]^T$ must satisfy $\frac{DH}{D\mathbf{u}}=0$, so, writing our costate as $\mathbf{p}=(p_1,p_2,p_3,p_4,p_5)^T$, we get the following condition:

$$\frac{DH}{D\mathbf{u}} = \left(\frac{Df}{D\mathbf{u}}\right)^T\mathbf{p}+\frac{D\mathcal{L}}{D\mathbf{u}} = \begin{bmatrix}\mathbf{0}\\ \frac{v_e}{m(t)}I\\ -\frac{\mathbf{u(t)}^T}{\|\mathbf{u(t)}\|}\end{bmatrix}^T\mathbf{p}+\frac{2\alpha v_e}{m(t)}\mathbf{u}(t)+2\beta\mathbf{u}(t)=\mathbf{0}$$

Because (as we noted earlier) $p_5(t)=0$ for all t, our equation simplifies to

$$v_e\begin{bmatrix}p_3\\p_4\end{bmatrix}+(2\alpha v_e+2\beta m(t))\,\mathbf{u}(t)=\mathbf{0}$$

which means that the optimal control is given by a linear function of $\mathbf{p}$ as

$$\tilde{\mathbf{u}}(t)=-\frac{1}{2\alpha+\frac{2\beta}{v_e}m(t)}\begin{bmatrix}p_3\\p_4\end{bmatrix}$$

Since we are optimizing over the final time T, we have one more boundary condition:

$$H(T,\mathbf{x}(T),\mathbf{u}(T),\mathbf{p}(T))=\frac{\partial\phi}{\partial t}(T)=\delta_1\frac{\partial\|\mathbf{y}_1-\mathbf{x}_f\|^2}{\partial t}(T)+\delta_2\frac{\partial\|\mathbf{y}_2-\mathbf{v}_f\|^2}{\partial t}(T)=0$$

Given an initial point $\mathbf{y}(t_0)=\mathbf{y}_0$, we can implement the derived quantities above to simultaneously solve the following coupled equations for the state and costate as a boundary value problem:

$$\tilde{\mathbf{y}}'=\frac{DH}{D\mathbf{p}}=f\left(\mathbf{y},\begin{bmatrix}u_1(t)\\u_2(t)\end{bmatrix}\right),$$

$$\tilde{\mathbf{y}}(0) = \mathbf{x}_f,$$

$$\tilde{\mathbf{p}}' = -\frac{DH}{D\mathbf{y}},$$

$$\tilde{\mathbf{p}}(T) = -\frac{D\phi}{D\mathbf{y}(T)} = \begin{bmatrix} -\delta_1 \frac{D\|\mathbf{y}_1(T)-\mathbf{x_f}\|^2}{D\mathbf{y}_1(T)} \\ -\delta_2 \frac{D\|\mathbf{y}_2(T)-\mathbf{v_f}\|^2}{D\mathbf{y}_2(T)} \\ 0 \end{bmatrix} = \begin{bmatrix} -2\delta_1(\mathbf{y}_1(T)-\mathbf{x_f}) \\ -2\delta_2(\mathbf{y}_2(T)-\mathbf{v_f}) \\ 0 \end{bmatrix}$$

3.2 Implementation

In the mathematical derivation above, we used dimensional physics equations for the acceleration due to gravity in our system. However, in order to make our system easier to analyze numerically, we can rescale the state such that any quantities in terms of space, time, and mass are dimensionless, and therefore closer to 1 (so they are more numerically) tractable. The primary result of this is that the gravitational constant G is set to 1 and the mass of the sun is also set to 1. This rescaling also made it easier to code up our solution. Because the transformation is only a rescaling, if we needed the solution in dimensional coordinates we can simply reverse the rescaling.

We solved our boundary value problem for the optimal control using Scipy's `solve_bvp` function. In our solution code, we do not include the Bolza cost on the final velocity, instead allowing it to be free. This is equivalent to setting $\delta_2 = 0$ in our formulation above, and corresponds to seeking only a fast flyby of our target destination before continuing on into deep space.

The BVP solver is very sensitive to initial conditions, and it took a lot of work to find initial conditions that allow it to converge. We had trouble with randomly initialized guesses because they violated the state evolution of the system so much that the bvp solver could not converge to an optimal solution. To overcome this complication, we first solved the following rescaled initial value problem, which uses the state equations in the absence of control thrusts to provide an initial trajectory guess. Using this formulation, $\mathbf{r}_{n+1}$ will be the initial guess for our spacecraft trajectory, and we set the spacecraft mass relative to the mass of the primaries to be $M_{n+1} = 0$ in our guess.

$$\mathbf{r}_i''(t) = \sum_{j=1, j\neq i}^{n+1} \frac{M_j(\mathbf{r}_j(t) - \mathbf{r}_i(t))}{\|\mathbf{r}_j(t) - \mathbf{r}_i(t)\|^3}, \qquad 1 \leq i \leq n+1$$

$$\mathbf{r}_i(0) = \mathbf{r}_{i,0}, \mathbf{r}_i'(0) = \mathbf{v}_{i,0}$$

The IVP above not only provides us with the initial guess trajectory of the spacecraft, but also with the positions of the primary bodies of the system needed to calculate the gravitational acceleration in our BVP solution for the optimal control. In practice we calculate the IVP using a significantly longer final time than our initial guess final time for the optimal solution, so that the positions of the primary masses will always be known. We note that our code can handle any n but we only investigated solutions where $n = 2$.

Below is a small portion of the code we use to solve the boundary value problem. The state_guess variable is the solution to the no-control scenario for the spacecraft. We determined our costate guess from the state guess, the costate evolution equation, and other properties derived above. We then solved our coupled boundary value problem to determine the optimal control thrust values and optimal trajectory. Our full code is found on our GitHub.

```
#when the control u=0, p1 and p2 (the integrals of p3 and p4)
#are constant and given by the Bolza cost endpoint condition
#p_12(t_f) = -Dphi/D_x(t_f) = -2delta(x(t_f) - x_f)
p12 = (-delta*2*(state_guess[3*(n-1): 3*(n-1)+2, -1] - xf))\
    .reshape(2,1)
#Shapes: (2,1)*(1,t_steps) = (2,t_steps)
p12_guess = p12*np.ones((1, t_steps))

guess = np.concatenate([
    #Guess positions from nbody solution with no control
    state_guess[3*(n-1): 3*(n-1)+2, :],
    #Guess velocities from nbody solution with no control
    state_guess[3*n + 3*(n-1): 3*n + 3*(n-1)+2, :],
    #Guess mass as not changing due to u = 0
    m0*np.ones((1,t_steps)),
    p12_guess,
    #p5 = 0 for all t. p3 and p4 are 0 when the control u = 0
    np.zeros((3,t_steps))
], axis=0)
```

4 Interpretation

We used our code for several different target destinations to evaluate the efficacy of our implementation. We also varied the initial conditions used for each destination. We used our solar system in each of the examples below, setting our two primary masses to be the Sun and Jupiter. The solutions below demonstrate the validity of our optimal control solutions. We note that we have plotted the paths of both primaries, along with the spacecraft's controlled trajectory with thrust and the natural trajectory that would be achieved with zero thrust. For each instance, we have also plotted the evolution of the mass over time as fuel is used, along with the magnitude of the thrust in the x and y directions.

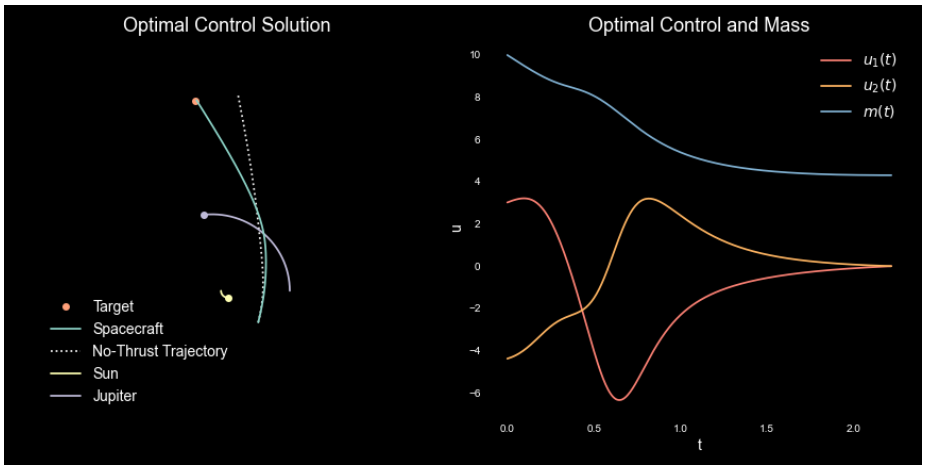

Figure 1: Our first control attempt. Interestingly, the control trajectory crosses the no-thrust trajectory twice here, since the spacecraft first thrusts in one direction, and then in the opposite direction to avoid getting caught by Jupiter's gravitational field. We see this in the plot on the right, where the magnitudes for the thrust in the x and y directions cross about a quarter of the way through the flight. Note the continued and varied fuel use throughout the flight.

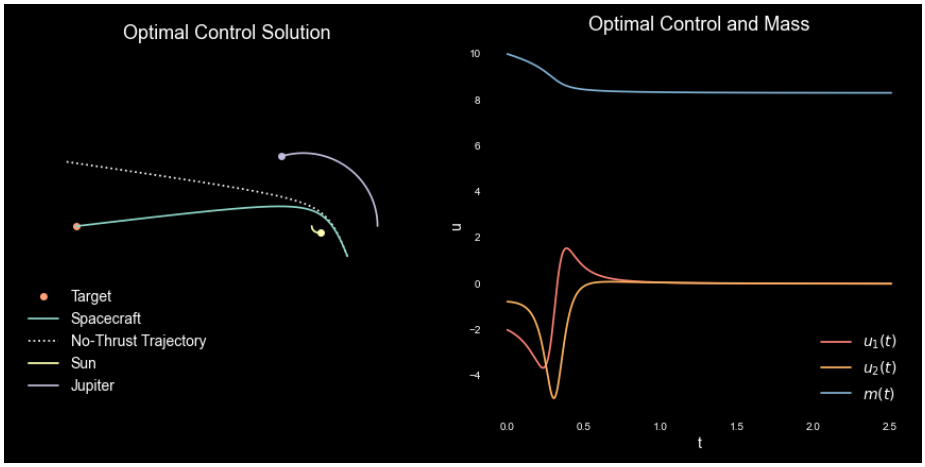

Figure 2: Our second control attempt. Here, we note that since the target destination is far below where the no-thrust trajectory would carry the spacecraft, the control solution has the craft thrust strongly downward at the beginning before cutting off the thrust for the remainder of the flight. It also seems to slingshot around the sun. This solution fits with our intuition that it is best to spend the beginning of the flight using fuel to maneuver the craft into the desired trajectory.

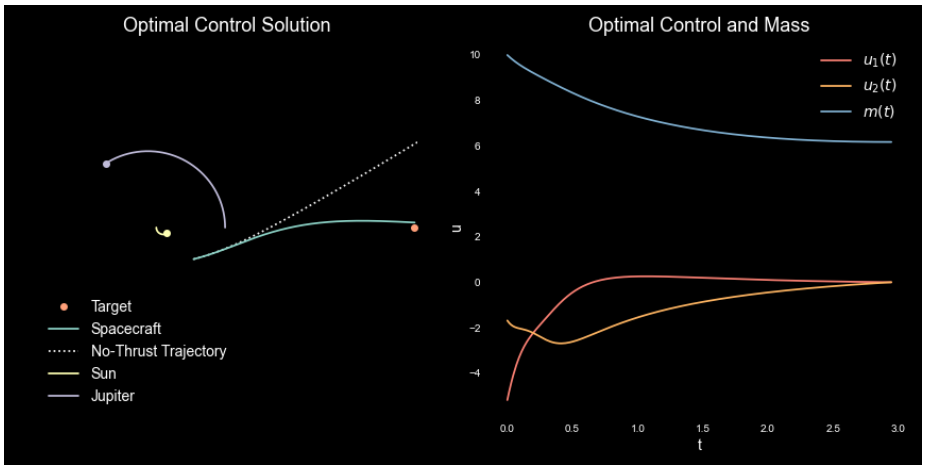

Figure 3: Our third control attempt. In this scenario, our destination is again far below where the no-thrust trajectory would take the spacecraft, so we have to thrust downward. But in this case, there is no slingshot maneuver, and the dynamics of the system are such that a continuous burn is needed through most of the flight. We see in the plot on the right that the mass continues to drop off (at a decreasing rate) throughout the flight, and the thrust vectors only approach zero at the very end.

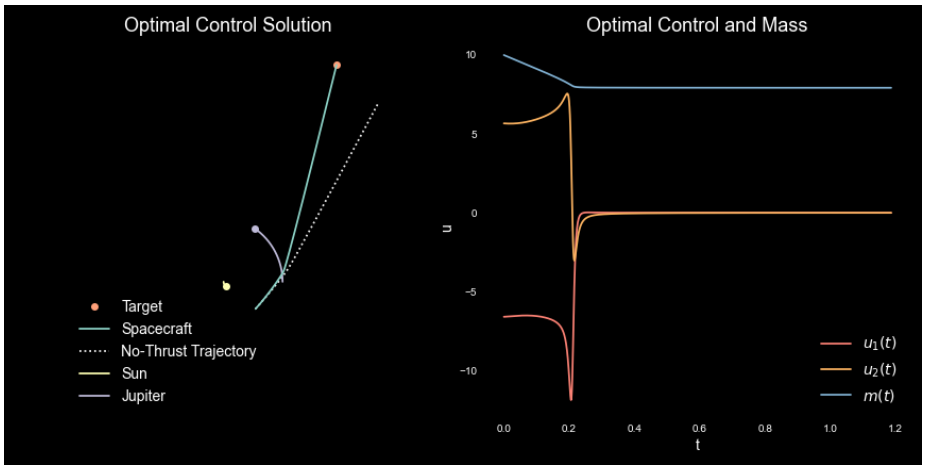

Figure 4: A successful slingshot maneuver with control. Interestingly, we see that the majority of the thrust for this flight takes place at the very beginning, and cuts off as soon as the slingshot maneuver is complete. This matches well with our intuition that it is best to thrust hard through the slingshot maneuver while close to the planet, and then to allow the craft to continue without thrust towards the destination for the remainder of the flight. Similarly to the scenario depicted in Figure 2, all of the maneuvering to put the craft onto a trajectory leading to the destination takes place at the very beginning of the flight. We also emphasize how fast the control cuts off after the slingshot is completed in this example. It almost looks like a piecewise function, and it is pretty amazing that our model achieved this using only soft constraints on the control.

5 Conclusion

We have managed to formulate and solve a very interesting and complicated control problem. As noted at the beginning of our paper, finding the optimal magnitude and direction in which to thrust each given point in the trajectory is a highly non-trivial problem. Add to that the fact that the primary bodies continue to move as well, and the problem is a pretty difficult one.

The solutions shown above can be difficult to fully understand as stationary plots, and the trajectories are better seen as animations. For all of the figures shown above, we have corresponding animations on our GitHub.