Landing a Rocket on Mars via Fixed Flight Time Powered Descent Guidance in MATLAB

Gagandeep Thapar
California Polytechnic State University, San Luis Obispo
AERO 560 Advanced Spacecraft Controls and Dynamics
December 7, 2022

Introduction to Convex Optimization

Interplanetary travel is one of the next steps in space exploration. However, many bodies, such as the Moon or Mars, have little atmosphere and no bodies of water so existing methods of landing e.g., via parachute or water landings are difficult if not unavailable to use. Propulsive landings are another technique, however, due to constraints on thrusters, the optimization problem to land safely under various kinematic and dynamic constraints is extremely difficult. Convex Optimization is one technique to take the concave state space of viable thrust profiles and convert it into a convex space that can be used to map thrust profiles. This is called Lossless Convexification (LCvx) and is the backbone of modern day convex optimization and propulsive landings. This paper implements findings in "*Convex Optimization for Trajectory Generation: A Tutorial on Generating Dynamically Feasible Trajectories Reliably and Efficiently"*, D. Malyuta et al., to generate a feasible landing trajectory on Mars. The implementation yielded a dynamically feasible trajectory that was able to land at the specified landing point in a fixed flight time derived from total on-board propellant. Convex Optimization allows problems to be set up intelligently such that other methods can be used to generate optimal solutions.Without this step, optimal controllers could not be used as infeasible states would frequently interfere with plausible solutions. Different forms of convexification exist for different reasons. LCvx exists to fully map concave spaces to convex spaces. This is computationally expensive, however, ensures the entire original space is captured. Other methods such as Successive Convexification (SCvx) exist to save on computation costs while trading on some feasible states. While the new space may be convex, it may not capture the entirety of the original concave space, precluding some solutions from being found. This paper generally talks about both styles and implements a LCvx solution for the propulsive guidance descent in fixed flight time (pdg-fft) problem.

Controllability and Observability

In the case of the implementation, we have full observability, i.e., we are able to identify the entire state vector exactly. This is because we are able to view all properties of the rocket and feed its information into a control scheme to determine its next input. A more realistic representation of the scenario would require the use of several sensors (e.g., LIDAR or RADAR for distance and velocity) and a fuel gauge to determine how much propellant is left to guide thrusting schemes. In terms of controllability, the simulation aims to make this as realistic as possible by only allowing the change in thrust to affect the state. This is the crux of the Convex Optimization problem so accurately representing the inputs to the state is crucial.

Convex Optimization in the Field

As mentioned in the **Introduction**, Convex Optimization is an emerging field that is vital to propulsive landings and entry, descent, and landing (EDL) in general. Maturing this technology will enable improved space exploration. This paper is a huge milestone in the field as it opens the doors and lowers the fences for other organizations to independently simulate and validate results. Within spacecraft control, this paper, and this topic, is significant as it makes autonomous landing extremely feasible.

Paper Analysis

The results in the paper are a mix of theoretical and experimental results. While a majority of the paper discusses theoretical convexification and simulation setup, much of the driving theory is derived from GFOLD, a NASA-developed algorithm that helped land some of the MARS rovers. Furthermore, the overarching lab, the Autonomous Controls Laboratory at the University of Washington under Behcet Acikmese, continually implements new iterations of their Convex Optimization on hardware to test trajectory generation and navigation. The results found in the hardware testing fuels the research published in the paper.

MATLAB Implementation

The discussion in the paper was originally implemented in Julia as there is strong support for mathematical operations and optimization. However, our implementation was fully developed in MATLAB (i.e., no Simulink). The problem was set up from the beginning as an optimization problem using the MATLAB Optimization Toolbox. The objective function was to minimize the thrust. Five variables were introduced that the solver should optimize: the 3D position [m], 3D velocity [m/s], mass [kg], 3D thrust direction [~], and thrust magnitude [N]. Several constraints were set in place to mimic the dynamics and feasible thrust profile.

The initial and final conditions for position, velocity, and mass were all set such that the rocket had a feasible starting and ending point. A minimum mass of the dry mass was set such that the rocket would not consume imaginary propellant; this was not honored as seen in the **Results** section. Dynamics of the system were forced in the form of optimization constraints. Each variable was preallocated to have the size related to the fixed time of flight (derived from the total on-board propellant and minimum thrust) of the rocket. Each row would represent the state at that discrete time step. Each successive state was derived from the previous state in addition to the inputs; equations of motion were integrated manually as using the matrices described in the literature yielded infeasible trajectories. Thrusting constraints were set in place and were derived from the total consumed propellant and the absolute minimum/maximum allowable thrust (30%/80% of the engine theoretical max, in the case of this implementation). These constraints mimic the available thrust profiles available. Mass constraints were set in place based on the thrust usage and the minimum allowable mass (rocket dry mass). This helped improve the realism of the rocket mass decreasing over time as a function of thrust profile. Attitude constraints were set in place to prevent the rocket from over-rotating and would mimic how engines onboard rockets or other landers have a maximum allowable gimbal angle. The attitude constraints were set by comparing the thrust direction relative to the local vertical and the maximum allowable gimbal angle. These constraints were set in place for each time step of the system and enabled a semi-realistic landing trajectory. Initial conditions of the rocket and simulation can be found in Table 1 below.

Table 1. Initial Conditions of the Rocket and Simulation

Acceleration due to gravity [m/s^2]:	3.711
Dry Mass [kg]:	1505
Wet Mass [kg]:	1905
Specific Impulse [sec]:	225
Max Engine Gimbal Angle [deg]:	27
Minimum Allowable Thrust [N]:	30% of 3100
Maximum Allowable Thrust [N]:	80% of 3100
Initial Position [m]:	[2000, 0, 1500]
Initial Velocity [m/s]:	[80 30 -75]
Simulation Time Step [sec]:	1
Number of States [~]:	7 (Position, Velocity Mass)
Number of Inputs [~]:	4 (Thrust Direction, Thrust Magnitude)

MATLAB Results

As a result of the hundreds of intelligently set constraints, a feasible landing trajectory was developed as seen below.

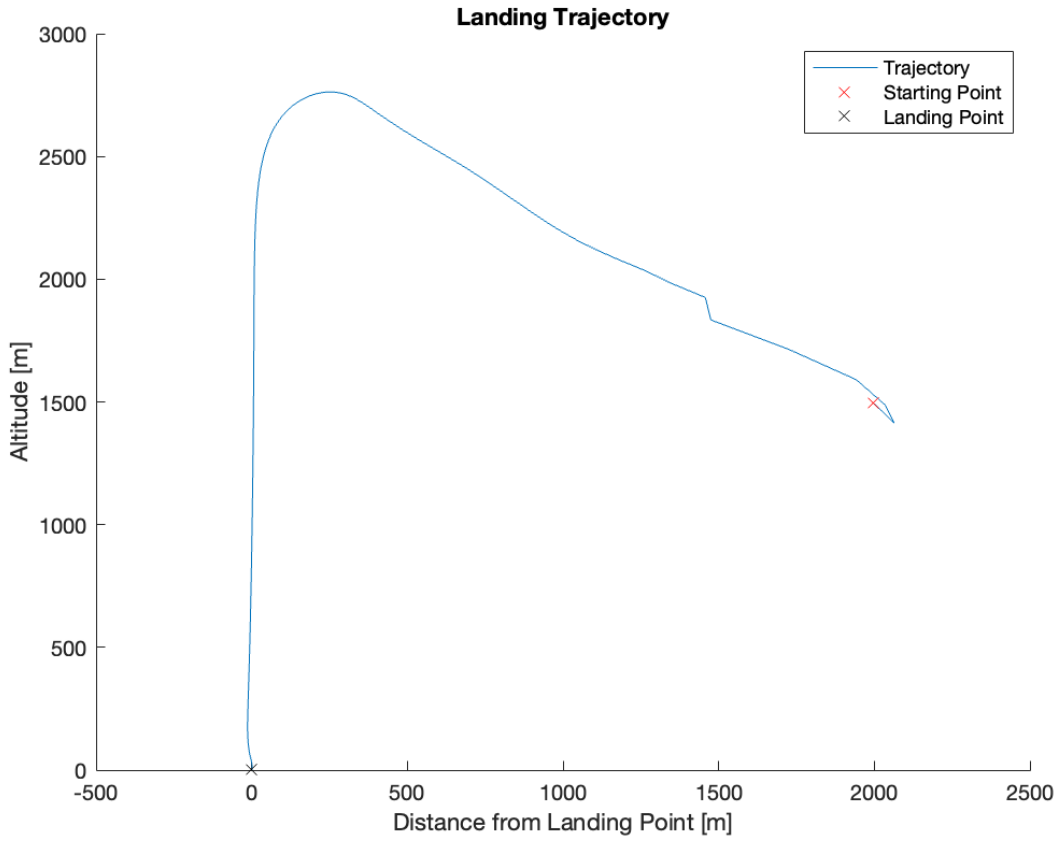

Figure 1. Landing Trajectory of the rocket, visualizing lateral and altitude differences over time

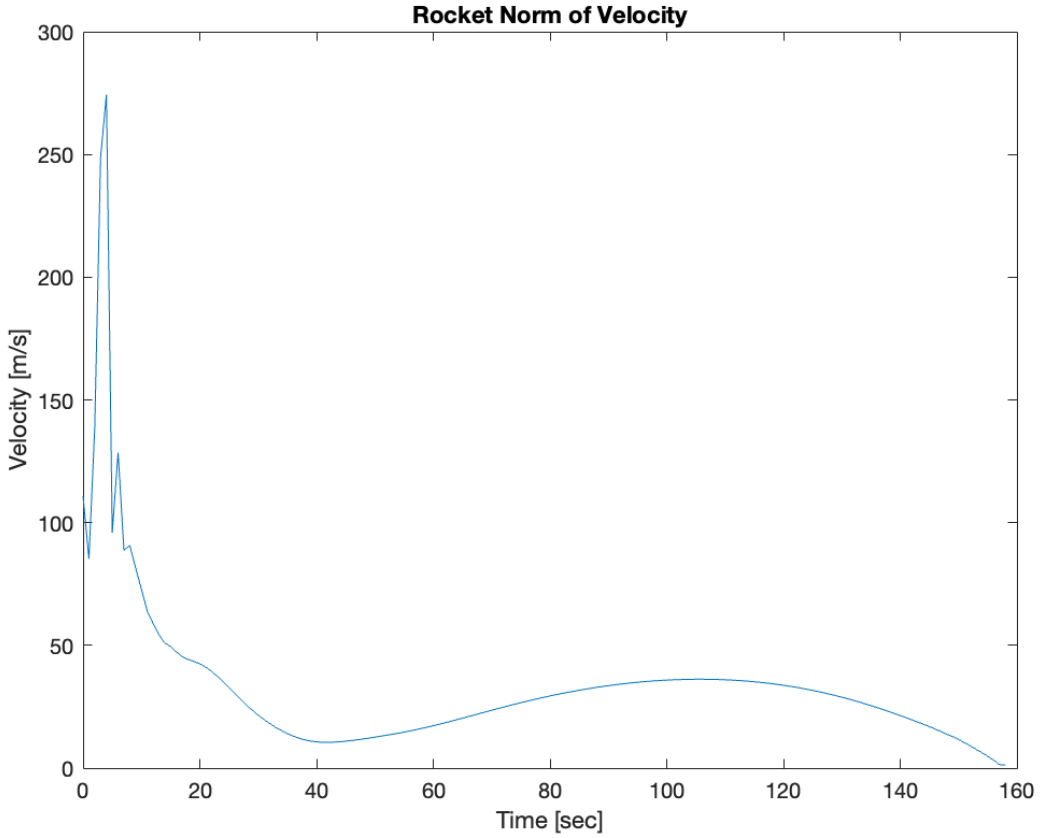

Figure 2. Norm of the rocket velocity over time

As seen in Figures 1 and 2, the rocket, given the initial position and velocity, was able to navigate towards the landing point and gently land as shown by the 0 final velocity and gentle slope in Figure 2. This was exciting to see as much of the optimization itself was a black box whose results could not be verified until the end. What was interesting to see was the rocket mass being less than the constrained dry mass as seen in Figure 3.

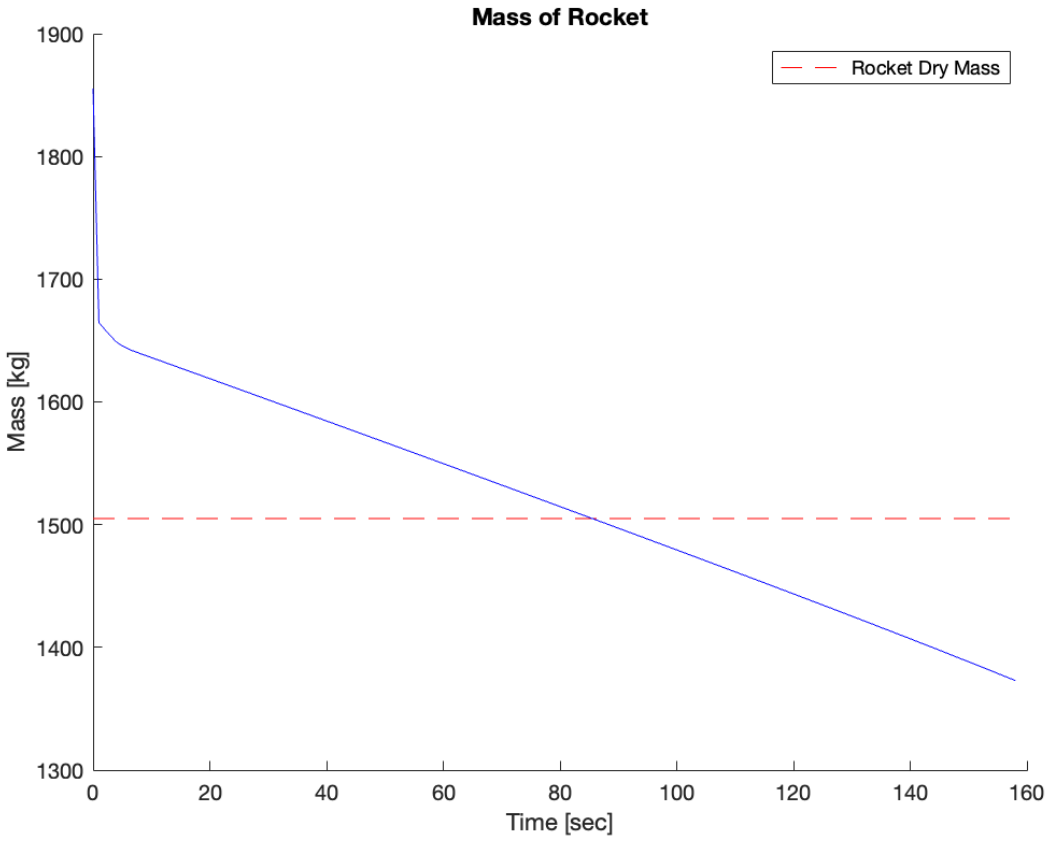

Figure 3. Mass of the rocket over time

This mass discrepancy is likely due to a misrepresentation of how the mass changes with respect to the thrusting scheme. Figure 4 offers another perspective into the landing trajectory. The lateral distances are seen slowly approaching 0 from their initial conditions and the altitude is shown first increasing before coming back down, likely to cover more lateral distance. This was interesting as the idea of a rocket first moving up to slow down before moving laterally to come back down is expected.

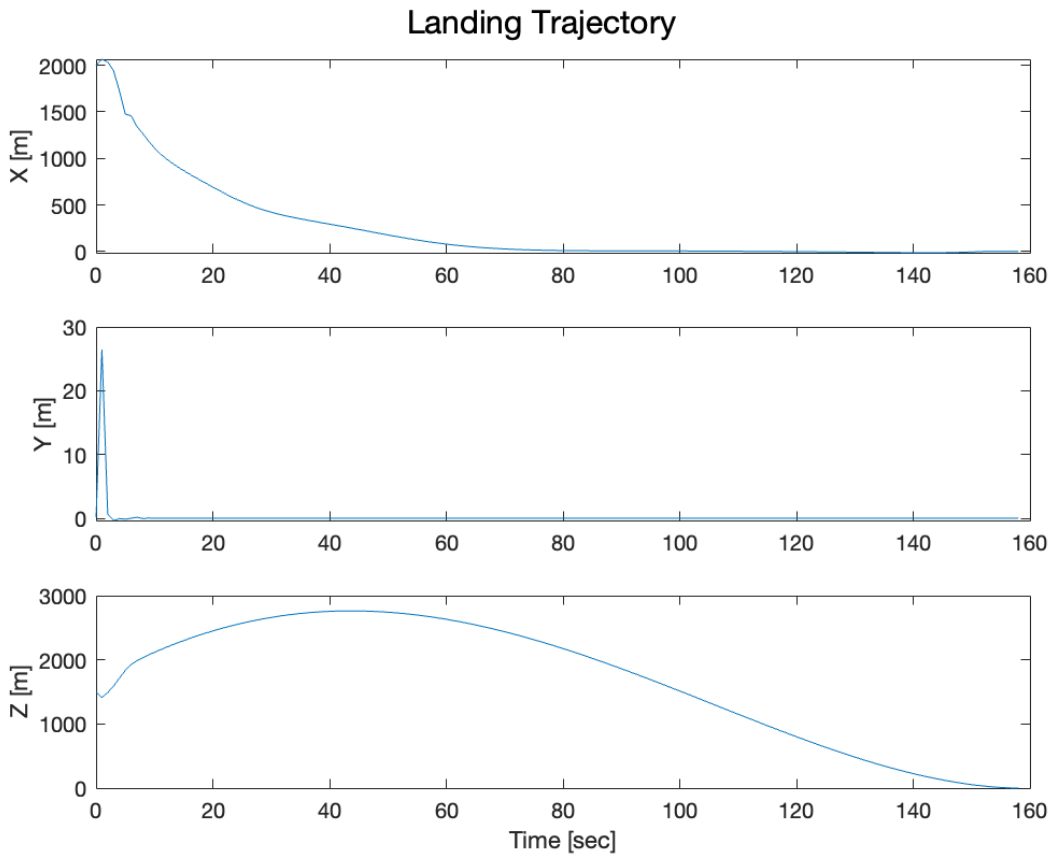

Figure 4. Landing trajectory of the rocket in each dimension: X (top), Y (middle), and Z (bottom). The initial position of the rocket was set to [2000, 0, 1500] km.