Collocation Method and Model Predictive Control for accurate landing of a Mars re-entry Vehicle

Neeraj Srinivas

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science

In

Aerospace Engineering

Jonathan T. Black, Chair

Kevin Kent Schroeder

Colin Adams

February 2nd , 2021

Blacksburg, Virginia

Keywords:

Mars, Entry Descent Landing, Optimal Control, Model Predictive Control, Monte Carlo, Collocation Method

Collocation Method and Model Predictive Control for accurate landing of a Mars re-entry Vehicle

Neeraj Srinivas

ABSTRACT

This thesis aims at investigating numerical methods through which the accuracy in landing of a Mars entry-descent-landing (EDL) vehicle can be improved. The methods investigated include the collocation method and model predictive control (MPC). The primary control variable utilized in this study is the bank angle of the spacecraft, which is the angle between the lift vector and the vertical direction. Modulating this vector affects the equations of system of equations and the seven state variables, namely altitude, velocity, latitude, longitude, flight path angle, heading angle and total time taken. An optimizer is implemented which utilizes the collocation method, through which the optimal bank angle is found at every discretized state along the trajectory which are equally separated through a definite timestep, which is a function of the end time state. A 3-sigma wind disturbance model is introduced to the system, as a function of the altitude, which introduces uncertainties to the system, resulting in a final state deviating from the targeted location. The trajectory is split into two parts, for better control of the vehicle during the end stages of flight. The MPC aims at reducing the end state deviation, through the implementation of a predictor-corrector algorithm that propagates the trajectory for a certain number of timesteps, followed by running the optimizer from the current disturbed state to the desired target location. At the end of this analysis, a new set of optimal bank angle are found, which account for the wind disturbances and navigates the EDL vehicle to the desired location.

Collocation Method and Model Predictive Control for accurate landing of a Mars re-entry Vehicle

Neeraj Srinivas

GENERAL AUDIENCE ABSTRACT

Landing on Mars has always been a process of following a set of predetermined instructions by the spacecraft, in order to reach a calculated landing target. This work aims to take the first steps towards autonomy in maneuvering the spacecraft, and finding a method by which the vehicle navigates itself towards the target. This work determines the optimal control scheme a Mars re-entry vehicle must have through the atmosphere to reach the target location, and employs method through which the uncertainty in the final landing location is mitigated. A model predictive controller is employed which corrects the disturbed trajectory of the vehicle at certain timesteps, through which the previously calculated optimal control is changed so as to account for the disturbances. The control is achieved by means of changing the bank angle of the spacecraft, which in turn affects the lift and drag experienced by the vehicle. Through this work, a method has been demonstrated which reduces the uncertainty in final landing location, even with wind disturbances present.

Acknowledgements

I would like to take this opportunity to thank my parents for their unwavering support through thick and thin, and for never giving up on me or letting me feel sorry for myself.

I want to thank the incredible faculty of the Kevin T. Crofton department of aerospace and ocean engineering at Virginia Tech, without whose help I would not have been able to start this project, let alone finish it. Dr. Kevin Schroeder has been an integral part of this study, guiding me when I fumbled, which was more often than I would like to admit. I truly am grateful for all the random meetings throughout the months.

I want to thank my peers and friends, for having my back through these last few months and inspiring me to keep pushing through all your devotion to the sciences.

Lastly, I want to take this opportunity to thank the administration of Virginia Tech, who despite a global pandemic, have managed to keep the show running and took some incredible efforts to make us students feel as comfortable as possible.

Table of Contents

List of Figures and Tables

Chapter 1: Introduction

When Buzz Aldrin and Neil Armstrong were descending toward the surface of the Moon in the Lunar Module Eagle, the LM guidance computer (LGC) warned them they were miles off from the intended landing zone, and to take necessary maneuvers to avoid being too overshot from the designated landing site. Although this was later explained by a faulty bus in the LGC, this incident became one of the first time that computers helped human beings landing on another celestial body with accuracy.

The issue of landing accuracy is of paramount importance to safety and mission success, particularly on a planet or another celestial body whose surface topology has not been mapped extensively. Compounding this issue with the factor of atmospheric uncertainties on another planet, the accuracy of successfully touching down becomes a carefully choreographed act, with little to no room for error. This study aims to investigate the methods through which a process like EDL can be improved upon, computationally. This is achieved through means of computational methods, particularly the collocation method in conjunction with a constrained optimization algorithm. This thesis investigates the feasibility of an accurate and controlled landing to a designated spot, through optimal control techniques which adhere to the physical laws governing the model of the spacecraft and its limitations.

With missions like the Mars Science Laboratory (MSL) prioritizing accuracy in landing, reduction of the area of probable touchdown will be of great importance for future missions to Mars and other planets. Other celestial bodies such as comets also require a very precise set of instructions to be carried out with a very narrow margin for error, accounting for the fact that

smaller celestial objects have much weaker gravitational forces, hence a controlled and precise touchdown is imperative for mission success. In the recent OSIRIS-REx mission which successfully retrieved a sample from the near Earth asteroid 101955 Bennu relied on a series of thruster burns to accurately perform a Touch and Go (TaG) operation at one of the four selected sites on the asteroid. Errors in these thruster firings could have resulted in excess velocities while touching the surface, jeopardizing the mission safety entirely. Hence, optimal control of spacecraft is, was, and always will be the crucial factor that determines mission success.

Controlling a landing vehicle can be achieved in multiple ways. The Apollo lunar modules made use of reaction control system (RCS) thrusters, which are attached to the module in such a way as to allows for turning in every direction. Using RCS thrusters on a Mars mission has a disadvantage of added weight to the system, as storing the fuel and thrusters onboard the EDL vehicle decreases the amount of payload mass possible on board, including the scientific equipment such as rovers. Hence, another method had to be implemented to maneuver the re-entry vehicle in a Martian atmosphere. The Mars Science Laboratory (MSL) which is the most advanced EDL system to successfully touchdown on Mars, achieved control by offsetting the lift vector from the center of gravity of the vehicle. This causes a rotational moment in the vehicle, which is then controlled to achieve accurate maneuvering toward the end point of the trajectory phases. A set of ballast masses are utilized to achieve the CG offset, which are ejected from either side of the vehicle. The drawback of this system is once again, the increased total mass, which translates to a lower mass of payload in the EDL vehicle.

1.1 Thesis Objective

The bank angle of the spacecraft is the angle between the lift vector and the vertical direction. Modulating the bank angle allows for vectoring the lift vector of the vehicle, which enables control. This method is adopted in this study, investigating how the optimal bank angle can be determined for achieving a precision landing on the surface of Mars, and how this optimal trajectory will be affected by the atmospheric disturbances present on the Martian atmosphere. A model predictive controller (MPC) is implemented to overcome these atmospheric disturbances and uncertainties, and Monte Carlo analyses are done to determine the landing probability ellipses. In the trajectory optimization analysis, various objective functions are investigated, including minimizing aerothermal loads, maximizing downrange, and minimizing target error. The next section discusses the methodology implemented in previous missions to Mars, as well as various numerical methods employed to reduce the size of the landing probability ellipse.

Chapter 2: Literature Review

Mars atmospheric re-entry vehicles can adopt three main configurations: Ballistic, ballistic-lifting and lifting. [1-3]. Ballistic EDL vehicles have a lower total aerodynamic heating, shorter values of range and flight time. The peak heat flux of these vehicles is usually high. They also have uncontrolled flights, and hence the landing error ellipse is larger. [4,5] Several previous Mars missions such as the Pathfinder [6], Spirit, Opportunity [7] and Phoenix [8,9] used this configuration. For all these missions, the landing error ellipses were in the range of 100s of kilometers, but the relative masses of these EDL vehicles were low. Ballistic lifting vehicles have more control than the ballistic configurations during re-entry, as they can produce a certain angle-of-attack through offsetting the center of gravity from the center of pressure. This in turn allows for trajectory control though modulating the bank angle and the direction of the lift vector. Ballistic-lifting configurations have a lower maximum dynamic pressure, as compared to ballistic configurations. The Viking [10] and Mars Science Laboratory (MSL) [11] missions have utilized the ballistic-lifting configuration. The MSL system architecture was designed to have a controlled descent into the Martian atmosphere using ballistic masses, which when ejected augmented the lift vector of the spacecraft [12]. This reduced the size of the landing ellipse dramatically to 19 km x 6.5 km [13]. The guidance algorithm used in MSL was derived from the Apollo era command module entry guidance [14], which modulated the bank angle to control the landing. Although this configuration provides some control to the spacecraft, it does not achieve pin-point landing accuracy, as the landing ellipses are still in the range of 10s of kilometers.

Lifting configurations in EDL vehicles allow for a large lift to be produced, which in turn increases the range and maneuverability of the spacecraft. This also allows for increased accuracy for heavier payloads such as human-centric missions to Mars. [16-18]. The downsides of lifting vehicles are the high cost, low reliability, and low technological development of these configurations.

Implementation of control during re-entry has been common since after the Pathfinder and Beagle-2 missions, which attained stabilized attitudes through an initial spin imparted at the entry interface [19,20], and performed an unguided entry into the Martian atmosphere. The entry interface precision was paramount, as corrections could not be made during the descent or landing processes. Mars Exploration rovers adopted a more refined guidance and control for the descent and landing processes, thus reducing the landing errors. The Phoenix mission implemented an unguided ballistic entry in conjunction with a reaction control system (RCS) to greatly improve parachute deployment and landing accuracy. MSL adopted a ballistic-lifting entry configuration with a three-axis stabilized attitude control system similar to the Mars Smart Lander

Previous missions to Mars have had large landing ellipses, often in the 100s of kilometers. The most advanced EDL vehicle to successfully deliver it's payload to Mars, the Mars Science Laboratory (MSL), utilized a system of ballast masses to achieve a bank angle reversal, through ejecting these masses from either side of the vehicle, enabling control over the trajectory and guiding the EDL vehicle to the desired landing location. Although this method significantly reduced the size of the landing ellipse, the additional ballast masses onboard adds an extra mass of around 300 kilograms. This is an extra mass that has to be onboard with the scientific equipment, which implies a lesser total mass of the scientific equipment. An innovative method

developed to overcome this weight issue is the development of inflatable re-entry vehicles, such as the Hypersonic Inflatable Aerodynamic Decelerator (HIAD), and the Adaptable Deployable Entry Placement Technology (ADEPT), which is a deployable semi-rigid aeroshell entry system with the ability to achieve a low ballistic coefficient. Controlling these vehicles have been studied extensively, such as in Cianciolo et al. [21], who developed a Direct Force Controller (DFC) wherein an inflatable EDL vehicle with controllable flaps alongside a bank angle and angle of attack control is utilized to reduce the landing ellipse size to withing 10 meters, with a mid L/D vehicle and a shallower angle of attack assumed in conjunction with retro propulsion enabled to ensure sufficient deceleration in this human-centric mission to Mars. Balaram et al. [22] discuss the Dynamics Simulator for Entry Descent and Surface Landing (DSENDS) which is a controller that incorporates high fidelity vehicle and parachute models to a real time terrain imaging Light Detection and Ranging (Lidar) and radars. The lidar data is used in a feedback controller to enable obstacle avoidance. Davis et al. [23] have developed a Numerical Predictor Corrector (NPC) and Analytical Predictor Corrector (APC) controllers to improve the landing accuracy to within 10 meters of the target, by using the CG offset and bank angle control methods, for a mid L/D vehicle with retro propulsion on board. In this study, the CG offset methods appeared to be more feasible in terms of design limitations and fuel usage trends. Korzun et al. [24] details a concept of a 20 ton payload to within 1 km of a target landing site through hypersonic guidance, supersonic deceleration using an inflatable aerodynamic decelerator (IAD) in the second phase of entry, and terrain imaging for hazard avoidance. Up to 45 degrees of bank angle modulation is allowed in this case in the hypersonic guided entry. J. Balaram describes a Strap-on High-altitude Entry Reconnaissance and Precision Aeromaneuver (SHERPA) system that makes use of a moving mass ballast system within the EDL vehicle to

achieve a precision landing [25]. A steering device that uses CG modulation and a terrain imager are integrated into SHERPA, which is an auxiliary system that can be attached to EDL vehicles. SHERPA equipped vehicles are found to be able to land at high altitude Martian regions. In this study, the bank angle modulation is assumed to be feasible through some form on CG offset, and the SHERPA system, which is designed as a strap-on auxiliary module, seems like an ideal fit for this mission. Huneker et. al [26] describe a Shefex-3 Pseudospectral Algorithm for Re-entry Trajectory Analysis (SPARTAN) which has an exponentially quick convergence to the corresponding optimal control problem, where the OCP is transcribed into a non-linear programming (NLP) problem through a collocation method. SPARTAN is shown to have numerical advantages in scaling, stable node placement and numerical integration, and 2 EDL cases are solved which demonstrate these advantages. Desai et. al [27] describe the EDL of the Stardust entry capsule which contained a sample from the comet Wild-2, without active guidance and control. It makes use of an angular spin rate of 13.5 rpm to achieve aerodynamic stability and a supersonic drogue chute at Mach 1.37 to achieve a landing to within 10km of the target. Calhoun et. al [28] describe an EDL system controller based on the Reaction Control System used in the Apollo Lunar Excursion Module digital autopilot, which utilizes roll and pitch/yaw jets to control the vehicle. A phase-plane controller similar to the LEM is implemented, modified to account for aerodynamic cross-coupling by inclusion of a set of hysteresis switching curves which reduces RCS noise during continuous bank maneuvers. Brugarolas et. al [29] describe an RCS attitude controller which enables fast response to large bank slews through a feedforward controller while a feedback path tracks bank commands and stabilizes the angle of attack and sideslip angle. Separate Proportional Derivative (PD) controllers are engaged when attitude errors are detected as compared to the reference values.

Optimizing the trajectory can be achieved through mathematical formulation, by solving the system of equations under certain linear or nonlinear constraint and bounds. These consist of direct and indirect methods. Direct methods Darby et. al [30] discuss an hp-adaptive psuedospectral method for solving optimal control problems (OCP), which determines the locations of segment breaks and the polynomial approximation degrees required in each segment. This method is found to have been more efficient at solving numerical OCPs while having better accuracy than global psuedospectral methods. Psuedospectral methods for solving OCPs have been explored widely in the past, such as in Schultz et. al [31] where a Gauss Psuedospectral Optimal Control Software (GPOCS) is used to converge psuedospectral methods to constrained optimal reentry trajectories. To avoid the Runge phenomenon, a Chebyshev point placement is used to distribute the nodes used in the collocation strategy. This method ensures a dynamic density to the number of nodes, ensuring accurate modelling of physical phenomena in regions that require it. GPOCS is successful in generating complex trajectories, as seen in this study. Psuedospectral methods are also used in orbit simulations, as discussed in Liu et. al [32], where the method is utilized to simulate a Mars exploration orbit, with defined thrust vectors at each of the Legendre-Gauss nodes as the optimal control output vector. This study demonstrates that a with even a rough estimate of the initial and final states, the psuedospectral method converges to the optimal solution. Indirect methods are also utilized in EDL trajectory analysis, as seen in Grant et al. [33] where a continuation of fast indirect methods are used to overcome the computational costs associated with direct methods, in order to conduct a design space exploration. Initial guesses are computed through indirect methods, for unconstrained optimal trajectories. This initial guess is then used to continue the indirect methods, while introducing constraints throughout the trajectory such as g-loading and heat flux. The indirect method is

found to converge to optimal solutions even with these path constraints. Other examples of psuedospectral methods used for solving OCPs include Benson et al. [34] where a Gauss psuedospectral method is shown to solve the optimal control problem where the collocation of nodes follows Legendre-Gauss points, and Vlassenbroeck et al. [35] where a Chebyshev technique is used to solve NLP optimal control problems. Other algorithms investigated to solve the optimal control problems in EDL include a Particle Swarm Optimization (PSO) technique as described by Lavagna et al. [36], where the trajectory phases are split up based on the flight regime and optimizers are implemented to ensure the state vectors and guidance profiles preserve continuity at the interface of each phase. This work is shown to be computationally expensive. Jiang et al. [37] discuss utilizing the PSO technique in conjunction with the Gauss psuedospectral method for pin-point Mars landings. This study merges local and global optimization, with coarse optimization being utilized to provide a suitable initial guess. The bank angle is controlled at various points along the trajectory, and a good convergence is shown. Shuang et al. [38] utilize the desensitized optimal control (DOC) and direct collocation and NLP methods to solve the OCP. The DOC is used to decrease the sensitivity of the terminal state variables which are subject to atmospheric uncertainties, after which the collocation method is utilized to find the optimal control variable which in this case is the bank angle. A direct method of trajectory optimization using collocation method and NLP is discussed in Hargraves et al. [39], where cubic polynomials are used to represent state variables, linear interpolates and control variables. Collocation method is then used to satisfy the differential equations.

Chapter 3: Nonlinear Programming

Calculating the minima or maxima of a given problem given some objective or cost function, subject to some constraints on its variables or subject to satisfaction of some equalities and/or inequalities are known as optimization problems. An optimization program with nonlinearities in its objective function or constraints are known as nonlinear programs. Nonlinearities in objective functions or constraints are crucial in certain application, to properly represent the application as a mathematical program.

A general optimization problem is aimed at selecting *n* separate decision variables $x_1, x_2, x_3 \ldots x_n$ from a provided feasible region of values in such a way as to optimize a given cost function. The goal of the optimization may be to minimize or maximize the objection function

$$f(x_1, x_2, x_3, \ldots, x_n)$$

This problem is called a nonlinear programming problem (NLP) if the objective or cost function is nonlinear or if the feasible region given to the system is determined by some nonlinear constraints. Hence, a general nonlinear optimization problem may be described as:

$$Minimize\ f(x_1, x_2, x_3, \ldots, x_n) \tag{1}$$

Subject to the constraints:

$$g_1(x_1, x_2, x_3, \ldots, x_n) \leq b_1 \tag{2}$$

$$g_2(x_1, x_2, x_3, \ldots, x_n) \leq b_2 \tag{3}$$

:

:

$$g_m(x_1, x_2, x_3, \ldots, x_n) \leq b_m \tag{4}$$

Each of the constraint functions g_1 through g_m are provided to the system.

Some other constraints provided to the system may be in the form of limits for the decision variables, specifying the bounds within which each variable must lie

$$(x_1, x_2, x_3, \dots, x_n) \leq K \tag{5}$$

Where *K* is some numerical value. Adding these bounds can affect the regions of feasibility of the solution, where the local or global optimum is found. Another set of constraints that may be added to the NLP are the equality and inequality constraints. These state that constraints or variables added to the system must satisfy some equality or inequality criteria for the solution to be feasible. These constraints may also take nonlinear forms, thereby complicating the NLP and restricting the feasible region of solutions further. These are expressed as:

$$h_1(x_1) = c_1 \tag{6}$$

$$h_2(x_2) \leq c_1 \tag{7}$$

Here, h_1 is an equality constraint, which is a function of the variable x_1 specifying that the function should equal some value c_1. Similarly, h_2 is an inequality constraint in the variable x_1, specifying that the function should be less than some value c_2 for the NLP to be converged to an optimal solution.

Nonlinear programs are searching for local or global maxima or minima, known as the optima of the function. Geometrically, nonlinear programs can behave very differently from their linear counterparts, even when the constraints applied to the problem are linear. As such, the optimal solution of the problem may occur at an interior point of the feasible region, on the boundary of the feasible region, which is not an extreme point, or at an extreme point of the feasible region.

Extrema's are points at the beginning or end of the function, which are usually rarely where the optima lie. A global optimum is a solution to the overall optimization problem, and its objective value is as good as any other point in the specified feasible region. A local optimum, however, is the optimal solution only with respect to the points close to it, where the feasible solutions lie. Points far removed from the local optimum play no significant role in its definition and may be preferred to the local optimum. Mathematically, this is represented as

Let $x = (x_1, x_2, x_3, \dots, x_n)$ be one feasible solution to a maximization function whose objective function is $f(x)$ is a:

1. Global maximum, if $f(x) \geq f(y)$ for every feasible point $y = (y_1, y_2, y_3, \dots, y_n)$ which lie within the prescribed bounds;
2. A local maximum, if $f(x) \geq f(y)$ for every feasible point $y = (y_1, y_2, y_3, \dots, y_n)$ which lie sufficiently close to *x.* Simply put, if there are a small number of points so that, whenever each variable y_j is within a small limit of x, and y_j is a feasible point within the prescribed bounds, then $f(x) \geq f(y)$.

Generally, most general-purpose nonlinear programming procedures are near-sighted, as they seek out the local optima as opposed to the global optima. This is mainly due to computational costs associated with searching the entire feasible region for the global optima, which is complicated and computationally expensive given the nature of nonlinear constraints and objective functions present. However, since every global optimum is also a local optimum, the overall optimization problem may be stated as seeking the best local optima. Under certain conditions, the local maxima and minima are known to be global. A local minimum will be a global minimum when a function is *"convex".* Graphically, a function is convex when it tends to curve upwards. On the other hand, a local maximum will be the global maximum when the

function is" concave" or tends to curve downwards. For this reason, convex functions are usually minimized, and concave functions are maximized.

In the context of this EDL problem, the constraints applied to the optimization problem as well as the various objective functions studied are nonlinear, making this a nonlinear programming problem. The equality and inequality constraints, as well as the objective functions used are mentioned in detail later on in this text**.** The re-entry vehicle's trajectory is constrained so as to follow the trapezoidal integration rule for the system dynamics, and the various state and control variables are bound so as to keep the results in the physically feasible region. Nonlinear constraints are also applied to the problem to prevent the control variable from having large abrupt changes from one node to the next, where the system of equations is solved.

3.1 Trajectory Optimization

The optimization of trajectory of a spacecraft has existed since the early Apollo era of spacecrafts. In a nutshell, trajectory optimization is defined as finding the optimal path a body takes to travel from a source to a destination. This path would usually include the state of the body, i.e. the position, velocity, flight path angle, etc. and the control of the body, i.e. the bank angle, thrust, etc. The state and control of the body are predominantly functions of time, and this will be discussed in length later in the text. Trajectory optimization aims at calculating the best path for the system from a given initial state to a later state in time, while following certain constraints and parameters. The end value of state may or may not be known to the system, depending upon the type of analysis conducted. Some studies, particularly ones devised for calculating the operational envelope of the system, often do not define all the end state values, instead incorporating conditions in the optimization algorithm such that the system itself

converges to the most feasible solution within these operating constraints. The control variables in an optimization problem are often computed by the algorithm, to satisfy the initial and final conditions, while following the system dynamics model. The trajectory optimization problem can be modelled as a collection of a number of phases *N*, with the commonly used independent variable of time *t* for each phase *j* defined between the bounds $t_0^{(j)} \leq t \leq t_f^{(j)}$. In each of the phases *j* the dynamics of the system can be described using a set of dynamic variables $\boldsymbol{y}^{(j)}(t)$ and $\boldsymbol{u}^{(j)}(t)$, which represent the state variables and the control variables, respectively. The number of state and control variables are a function of the algorithm, denoted by $n_y^{(j)}$ and $n_u^{(j)}$. In addition to the state and control variables, various other parameters $\boldsymbol{p}^{(j)}$ may be incorporated, which are independent of *t.*

Generally, the system dynamics are represented by a set of ordinary differential equations written explicitly, referred to as the state equations or the system equations:

$$\dot{\boldsymbol{y}} = \boldsymbol{f}[\boldsymbol{y}(t), \boldsymbol{u}(t), \boldsymbol{p}, t] \tag{8}$$

Here, $\boldsymbol{y}(t)$ is the state vector, consisting of n_y states. At the time t_0, the initial conditions are given as

$$\Psi[\boldsymbol{y}(t_0), \boldsymbol{u}(t_0), \boldsymbol{p}, t_0] = \Psi_0 \tag{9}$$

Where Ψ is the combination of the state variable, control variable, and the time at the current state. Similarly, the final conditions are represented as

$$\Psi[\boldsymbol{y}(t_f), \boldsymbol{u}(t_f), \boldsymbol{p}, t_f] = \Psi_f \tag{10}$$

Often, the initial and final conditions as well as the states in between them have upper and lower bounds for their numerical values, depending upon system constraints or physical limits. These bounds can be represented as

$$\boldsymbol{y}_{lower} \leq \boldsymbol{y}(t) \leq \boldsymbol{y}_{upper} \tag{11}$$

The control variables may also be bounded by limits, similar to the state variables

$$\boldsymbol{u}_{lower} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{upper} \tag{12}$$

Optimal control problems are in essence determining the values of the *n* number of control vectors $\boldsymbol{u}^{(j)}(t)$ and the parameters $\boldsymbol{p}^{(j)}(t)$ in order to satisfy the objective function of the problem, which include minimizing or maximizing the performance index of the system represented as

$$J = \Phi[\boldsymbol{y}\left(t_0^{(1)}\right), t_0^{(1)}, \boldsymbol{y}\left(t_f^{(1)}\right), \boldsymbol{p}^{(1)}, t_f^{(1)}, \ldots$$

$$\ldots \boldsymbol{y}\left(t_0^{(N)}\right), t_0^{(N)}, \boldsymbol{y}\left(t_f^{N}\right), \boldsymbol{p}^{(N)}, t_f^{(N)}] \tag{13}$$

Where *N* represents the number of nodes or elements over which the optimization algorithm is implemented. Optimization techniques frequently involve the discretization of the trajectory into nodes, where the system dynamics are solved at each node, and each node is separated by a finite timestep. The size of the timestep remains constant throughout the trajectory, often calculated using the simple formula

$$\Delta t = \frac{Total\ time\ taken}{Number\ of\ nodes - 1} \tag{14}$$

The optimization algorithm solves the system of equations associated with the body at each node, forming the state vector for that particular node. The control variable is then calculated

while taking into consideration the state at each node, the constraints (if any) have to satisfied, and the values of each state and control vector lies within the prescribed bounds for the system.

Optimization algorithms are also often divided into two or more phases. Also referred to as an arc, a phase is a division of the time domain. Within a phase, the system's defining equations cannot change, but the change is possible when moving from one phase to another. The major reason for different phases in these analyses is to accommodate for changes in the system dynamics, such as separation of various stages of a rocket, from when it achieves lift-off to achieving a stable orbit. As the system moves to a lower gravitational environment, the equations of motion change are affected, which can be accommodated in the analyses through means of implementing a phase change, where the system dynamics are updated. The boundaries of phases are called junction points, or events. An event criterion is a boundary condition that defines the end of a unique phase.

3.2 Collocation Method

The collocation method is a numerical technique employed in the solving of ordinary differential equations, partial differential equations, integral equations, or a system of each type. A finite-dimensional space is utilized in this method, which contains the candidate solutions and several points in the domain. The optimal solution is selected which satisfies the given equations at the given nodes in the domain. These nodes are known as collocation points.

Collocation methods determine the approximate solution to a given integral equation, ODE or PDE at the given set of collocation points, but discretizing the integral into a number of individual functions within the variable bounds described by the equation or provided by the

user. This is a form of nonlinear programming, which will be discussed in detail further on in this study.

To find the approximate solution to a Given an integral equation

$$u(t) = \int_a^b K\big(t, s, u(s)\big)ds + f(t) \tag{15}$$

Through the collocation method, a certain *n*-parameter family of functions is chosen, represented by $\phi(t, c_1, c_2, c_3, \dots c_n)$ at the collocation points, $t_1, t_2, \dots t_n$ with a constant or variable value of timestep, depending upon the algorithm. The conditions that the equation is subject to is represented by

$$u_n(t_i) = \int_a^b K\big(t_i, s, u_n(s)\big)ds + f(t), \; i = 1 \dots n \tag{16}$$

Which represent a system of *n* equations in the unknowns $c_1 \dots c_n$. If the initial equation to be solved is linear and the solution approximation is a linear combination represented as $u_{n(t)} = c_1 \varphi_1(t) + \dots c_n \varphi_n(t)$ of the given function $\varphi_1 \dots \varphi_n$ then the system of equations in the unknows $c_1 \dots c_n$ will also be a linear set.

Collocation methods are also subject to constraints and bounds imposed by the algorithm. Examples of constraints include path constraint, equality constraints, and inequality constraints. Path constraints define the limits of mobility for the given agent in the simulation. Restricting a self-driven automobile to a certain lane in a highway can be assumed as a form of a path constraint. The algorithm tries to minimize deviations from this path, while satisfying the physical model and equations by which the system is defined. These constraints are generally computationally expensive, and thus make the numerical problem more tedious to solve. These are generally hard constraints, meaning they must be satisfied for the solution to converge.

Equality constraints are stated such that the equality must be satisfied for the convergence of the solution, these can be for several variables. Similarly, inequality constraints state the values of certain variables must be greater than or lesser than a provided numerical value. A generalized constrained minimization problem can be described as:

$$Minimize\ f(x)$$

$$Subject\ to:\ \ g_i(x) = c_i\ \ for\ i = 1 \ldots n\ Equality\ Constraints$$

$$h_i(x) \geq\ d_j\ \ for\ j = 1 \ldots n\ Inequality\ Constraints$$

where $g_i(x)$ and $h_j(x)$ are the hard equality and inequality constraints, respectively. The algorithm must satisfy these constraints along with the function *f(x)* for the solution to converge. The equality and inequality constraints used in the analysis will be discussed in the Procedure section below. Figure shows an example of the collocation method, where a continuous polynomial $u(t)$ is split into a set of linearly piecewise functions between timesteps t_0 and t_1, with constraints c_1 through c_3 applied.

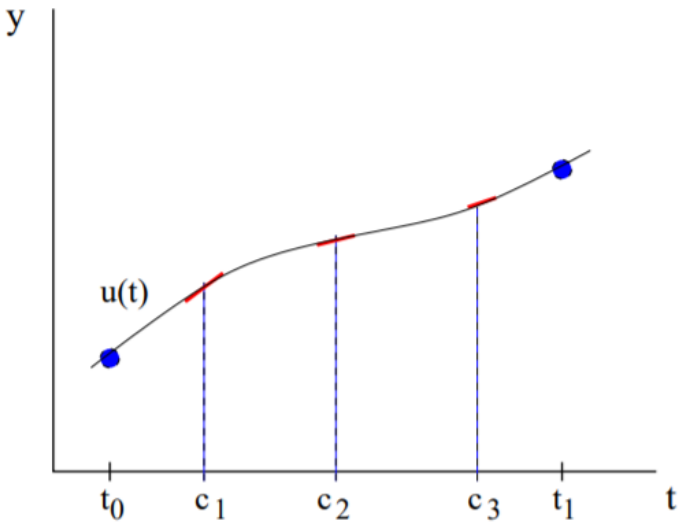

Figure 1: Collocation Method

In this analysis, the continuous function split into 50 nodes along the trajectory through the collocation method, and sequential quadratic programming is utilized to solve the constrained optimization problem. Collocation method is chosen as a numerical method for this study, due to it's advantages of being effective when both initial value and boundary value problems are present, while being able to accommodate nonlinear constraints within the function.

3.3 Sequential Quadratic Programming

Sequential quadratic programming (SQP) is one of the most effective and widely utilized methods for solving nonlinear constrained optimization problems. SQP is widely used in commercial solvers, due to its robust ability to tackle small- or large-scale problems. The method generates steps by solving quadratic subproblems, in the feasible region specific to the problem. SQP can be described as a generalization of the Newton's method for solving quadratic functions, utilized in unconstrained optimization in that it finds a step away from the current point being analyzed by minimizing a quadratic model of the problem. In essence, the SQP algorithm replaces the objective function with a quadratic function of the form

$$q_k(d) = \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 xx\, L(x_k, \lambda_k) d \tag{17}$$

SQP also replaces the constraint functions with linear approximations. In the general form of a NLP optimization problem

$$\text{minimize } f(x)$$

$$\text{subject to} \qquad c_i(x) = 0, \quad \forall i \in \boldsymbol{\varepsilon} \tag{18}$$

$$c_j(x) \leq 0, \quad \forall i \in \mathrm{I} \tag{19}$$

Where $c_i(x)$ are the equality and inequality constraints.

The Lagrangian estimates for this problem is defined as

$$L(x, \lambda, \sigma) = f(x) - \lambda c_i(x) - \sigma c_j(x) \quad (20)$$

At every step d_k, the quadratic subprogram developed is solved:

$$min\{q_k(d): c_i(x_k) + \nabla c_i(x_k)^T d \leq 0\,, i \in I\; c_i(x_k) + \; + \nabla c_i(x_k)^T d = 0, i \in \; \boldsymbol{\varepsilon}\} \quad (21)$$

If the starting point of the estimation, x_0, is sufficiently close to the optimum x^*, and the Lagrange multiplier estimates λ_k remains close to the optimum λ^*, then the sequence generated by setting the step $x_{k+1} = \; x_k + \; d_k$ converges to the optimum x^* at a rate with an order 2. The Lagrange multipliers estimates are obtained by using the optimal multipliers for the quadratic subproblem at the previous iteration, or through solving a new sub-problem.

The convergence of SQP can be improved by implementing a line search algorithm. The choice on how much distance to move along the direction generated by the sub-problem is clear in the unconstrained case, where a step length is chosen that approximately minimizes the function along the direction of search. However, for a constrained optimization, we would like the next iteration to not only minimize the function value, but also the adhere to the constraints and strive at satisfying them. Often, a weighted function is used to adjust the relative importance of these two objectives, and a merit system is implemented to determine if one point on the line is better as compared to another. The merit function is as follows:

$$p_1(x; \vartheta) = f(x) = \sum_{i \in \xi} \vartheta_i |c_i(x)| + \sum_{i \in \xi} \vartheta_i \, max(c_i(x), 0\,) \quad (22)$$

Where $\vartheta_i > 0$ are the penalty parameters. This approach is used in the MATLAB function *fmincon.* The optimizer outputs the trajectory that satisfies the provided nonlinear constraints.

Disturbances are added to this trajectory, to account for atmospheric wind in the Martian atmosphere. A model predictive controller is then implemented to overcome these disturbances and provide a more accurate landing.

3.4 Model Predictive Control

Model predictive control (MPC) is a form of prediction-correction algorithm, that optimizes a function subject to constraints and an objective function (also called cost function) that is subject to minimization. Traditional MPC models have a finite horizon implementation. This translates to the controller being able to predict the system dynamics for a prescribed interval of time in the future – known as the horizon- and optimize the control variable such that the disturbances and uncertainties occurring in the predicted future are minimized.

The MPC approach computes the current control action on-line rather than using a prescribed or pre-defined set of control variables. This makes the approach ideal for real-time scenarios, where unknown disturbances are present. This robustness of MPC makes it ideal to use for this specific EDL problem, where atmospheric disturbances in the form of winds cause position and velocity uncertainties in the system. MPC is a control algorithm in multiple variables and has the following key components:

- System dynamics model, detailing the behavior of the system through its governing equations
- A cost function **J** which is considered over a finite amount of time horizon
- An optimization algorithm that modulates the control variable **u** to minimize the said cost function **J.**

MPC models that optimize nonlinear systems or are subject to nonlinear constraints are known as nonlinear MPC or NMPC, the main difference being that the problems may not be convex. NMPC optimal control solutions are usually found through direct optimal control methods such as direct collocation, which is implemented in this study. The NMPC implemented in this study allows for uncertainties at the entry interface, as well as wind disturbances throughout the trajectory.

Considering a continuous time system described by the following non-linear differential equation

$$\boldsymbol{x}(t) = \boldsymbol{f}\big(\boldsymbol{x}(t), \dot{\boldsymbol{u}}(t)\big), \;\; \boldsymbol{x}(0) = \boldsymbol{x_0} \tag{23}$$

This system is subject to the following state constraints and input constraints

$$\boldsymbol{u}(t) \in \dot{U}, \;\; \forall t \geq 0 \tag{24}$$

$$\boldsymbol{x}(t) \in \dot{X}, \;\; \forall t \geq 0 \tag{25}$$

Where $\boldsymbol{u}(t)$ and $\boldsymbol{x}(t)$ represent the control and state vectors respectively. These vectors may be subject to limits of their own, where the optimizer attempts to converge to a feasible solution within these bounds. These limits may be represented as $\boldsymbol{u_{min}}$, $\boldsymbol{u_{max}}$, $\boldsymbol{x_{min}}$ and $\boldsymbol{x_{max}}$.

NMPC inputs are usually applied to the system based on the open-loop optimal control problem, with a finite time horizon. At every re-calculation instant, this problem is solved again.

The cost function $\boldsymbol{J}\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big)$ is subject to minimization, with the optimal control vector denoted as $\overline{\boldsymbol{u}}(t)$. This is represented as

$$\boldsymbol{J}\left(\boldsymbol{x}(t), \overline{\boldsymbol{u}}(k)\right) = \int_{t}^{t+T_P} F(\,\overline{\boldsymbol{x}}(\tau), \overline{\boldsymbol{u}}(\tau))d\tau \tag{26}$$

Subject to

$$\dot{\bar{\boldsymbol{x}}}(\tau) = f\left(\bar{x}(\tau), \bar{u}(\tau)\right),\ \bar{x}(t) = x(t) \tag{27}$$

$$\bar{u}(\tau) \in U,\ \forall\tau \in [t, t + T_c] \tag{28}$$

$$\overline{\boldsymbol{u}}(\tau) = \overline{\boldsymbol{u}}(t + T_c)\ \ \forall\tau \in [t + T_c, t + T_P] \tag{29}$$

$$\bar{x}(t) \in X,\ \forall\tau \in \left[t, t + T_p\right] \tag{30}$$

Here, T_p and T_c represents the prediction and correction horizons. In most cases, T_p will be greater than T_c. The internal controller variables are denoted by the bar above the variables, and $\dot{\bar{\boldsymbol{x}}}(k)$ denotes the solution of equation 28 for the input specified as $\dot{\bar{\boldsymbol{u}}}(k)$ which lies within the above mentioned limits in **U**. The predicted values in the controller may not always equal the actual system variables, hence a distinction is provided. This difference between the predicted and real system values are due to the determination of the applied specific input through a re-optimization at every instant of recalculation, through a moving finite horizon specified as T_c.

The cost function provided to the controller is defined in terms of the cost at each stage of the analysis, denoted by $\boldsymbol{F}$, which specifies the performance of the MPC. A quadratic stage cost $\boldsymbol{F}$ can be denotes as

$$F(\boldsymbol{x}, \boldsymbol{u}) = (\boldsymbol{x} - \boldsymbol{x_s})^T Q(\boldsymbol{x} - \boldsymbol{x_s}) + (\boldsymbol{u} - \boldsymbol{u_s})^T R(\boldsymbol{u} - \boldsymbol{u_s}) \tag{31}$$

Where the reference trajectory is denote using the state variable $\boldsymbol{x_s}$ and its control variable $\boldsymbol{u_s}$**.** As the real trajectory with disturbances differs from the reference trajectory, the

magnitude of deviation is denoted by the definite positive matrices Q and R. For example, in the case of a stabilization problem where the system reference is returning to the origin, $\boldsymbol{x_s}$ and $\boldsymbol{u_s}$ may be denoted as (0,0), the steady state.

In NMPC, the behavior of the system in the future is predicted using the system model initialized by the actual system state. This is denoted in equation 28**,** where the initial condition is input to the system at the moment of recalculation. For the prediction calculation, all the state information is necessary. Hence The full state must be either estimated or measured. Equation 30 fixes the input beyond the specified control horizon denoted as $\overline{\boldsymbol{u}}(t + T_c)$.

Optimal solutions of the minimization of the cost function are denoted as $\overline{\boldsymbol{u}}^*\big(k; x(t)\big)$ in the range [t, $t + T_p$] which belong to **U**. The optimal open-loop control problem is repeatedly solved at the instants of recalculation $t_j = j\delta,\ \ j = 0,1$ ... and the applied input to the system is the optimal solutions found through minimization of the cost function, denoted as

$$\boldsymbol{u}(t) \coloneqq \overline{\boldsymbol{u}}^* \left(t; \boldsymbol{x}\big(t_j\big)\right) \tag{32}$$

The closest recalculation instant to time t is denoted as t_j. Hence, the closed loop system is represented as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \overline{\boldsymbol{u}}^* \left(t; x\big(t_j\big)\right)\right) \tag{33}$$

The optimal minimization cost function as a function of the state of the system is known as the value function

$$\boldsymbol{V}(x) = J\big(\boldsymbol{x}, \overline{\boldsymbol{u}}^*(k; x)\big) \tag{34}$$

V(x) is often considered as the Lyapunov function and plays a vital role in stability analysis.

In an ideal scenario, the prediction and control horizon will be set to infinity, to minimize the total cost from start to end state. Due to computational costs however, this method is infeasible.

Implementation of the NMPC consists of two distinct procedures:

- Propagating the re-entry vehicle through the Martian atmosphere with the previously calculated optimal control, while introducing disturbances to the system. This acts as the prediction horizon and is dependent on the specified timestep of the system. The EDL vehicle trajectory can be propagated for any specific number of nodes at one time, by altering the timestep that the uncontrolled dynamics are solved for. As a result of the propagation, the system is not at the optimal trajectory points calculated through the collocation method previously, due to the disturbances introduced and the lack of optimal control to overcome these disturbances.
- The new position after a specified number of nodes of uncontrolled trajectory is used as the starting location for the optimizer. From this location, the optimizer calculates the trajectory to the specified end state, through optimizing the bank angle. Due to the altered initial state, the control variable at the starting node of this optimization solution will be altered as compared to the baseline solution. Once the optimal control to the final state has been calculated, the uncontrolled propagation is again continued from the node for a specified number of nodes. This process is repeated till the optimizer has computed the new optimal bank angle for all nodes from the entry interface to the end state. Detailed explanation of this NMPC is provided in the later Procedure sections.

Chapter 4: Procedure

4.1 System Dynamics

The re-entry vehicle dynamics model chosen for this analysis is a 6 degree of freedom point mass model, in a planet-fixed coordinate frame. This model is a mid L/D configuration, considering the unpowered flight. The control variable chosen is the bank angle of the spacecraft.,

Similar to the Apollo spacecraft and the Mars Science Laboratory (MSL) re-entry vehicle, the bank angle of the vehicle is chosen as the control variable for this analysis. The bank angle has proven to be an effective method of controlling the EDL vehicle during its hypersonic phase, by offsetting the lift vector of the vehicle from its center of gravity. Through this offset, the magnitude of lift produced on the vehicle varies and can be vectored directionally, hence enabling the controlling of the EDL vehicle. Traditionally, re-entry vehicles that utilize bank angle control have an open loop design, which decreases landing accuracy. In this analysis, a closed loop model predictive controller is utilized to gain feedback and correct the changes in the state variables that occur due to disturbances along the trajectory. This enables a far more accurate end state. The bank angle is calculated at every node along the discretized trajectory to optimize the trajectory.

The equations of motion used are as follows:

$$\frac{d\theta}{dt} = \frac{V}{r}\frac{\cos(\gamma)\cos(\psi)}{\cos(\phi)} \tag{36}$$

$$\frac{d\phi}{dt} = \frac{V}{r}\cos(\gamma)\sin(\psi) \tag{37}$$

$$\frac{dr}{dt} = Vsin(\gamma) \tag{38}$$

$$\frac{dV}{dt} = -D - gsin(\gamma) \tag{39}$$

$$\frac{d\gamma}{dt} = \frac{1}{V}\left[Lcos(\sigma) - \left(g - \frac{V^2}{r}\right)\cos(\gamma)\right] \tag{40}$$

$$\frac{d\psi}{dt} = -\frac{1}{Vcos(\gamma)}\left[Lsin(\sigma) + \frac{V^2}{r}\cos^2(\gamma)\cos(\psi)\tan(\phi)\right] \tag{41}$$

Equations (36) through (41) model the trajectory of the system using changes the longitude, latitude, altitude, velocity, flight path angle and flight heading angles, respectively. σ is the bank angle, defined as the angle from the vertical direction to the lift vector. A positive bank angle corresponds to a right-side bank and a negative bank angle corresponds to a left side bank. A bank angle value of zero represents the lift vector directed upward in the longitudinal plane. D is the drag force experienced by the vehicle, represented by equation (42). In order to simplify the flight model, the lift produced is maintained as a value dependent on drag, represented by equation (43). This assumes the vehicle is at a constant angle of attack throughout its trajectory which in this case is set at 20 degrees.

$$D = \frac{1}{2}\rho V^2 SCd \tag{42}$$

$$\frac{L}{D} = 0.22 \tag{43}$$

The density model chosen for this study is an exponential function as described in equation (44).

$$\rho = \rho_0 \, exp^{\frac{r-r0}{h_s}} \tag{44}$$

The function is dependent on the density at ground level (ρ_0), radial distance from the center of Mars (r), and the atmospheric scale height (h_s). This model has been used extensively in both 3 degrees of freedom as well as 6 degrees of freedom models and has proven to be a reliable yet computationally inexpensive atmospheric reference model.

Throughout the trajectory, the EDL vehicle will experience atmospheric disturbances in the form of wind, density, and temperature fluctuations. In order to improve the fidelity of this simulation, a wind disturbance model has been incorporated into the trajectory. The model is a combination of the Mars GRAM (Global Reference Atmospheric Model) and data obtained from the Mars orbiter sounder data.

Figure 3 depicts the 3-sigma limits of these values, and the mean value as a function of the altitude. North-South winds are the disturbance, which have a maximum magnitude at around 60 kilometers in altitude.

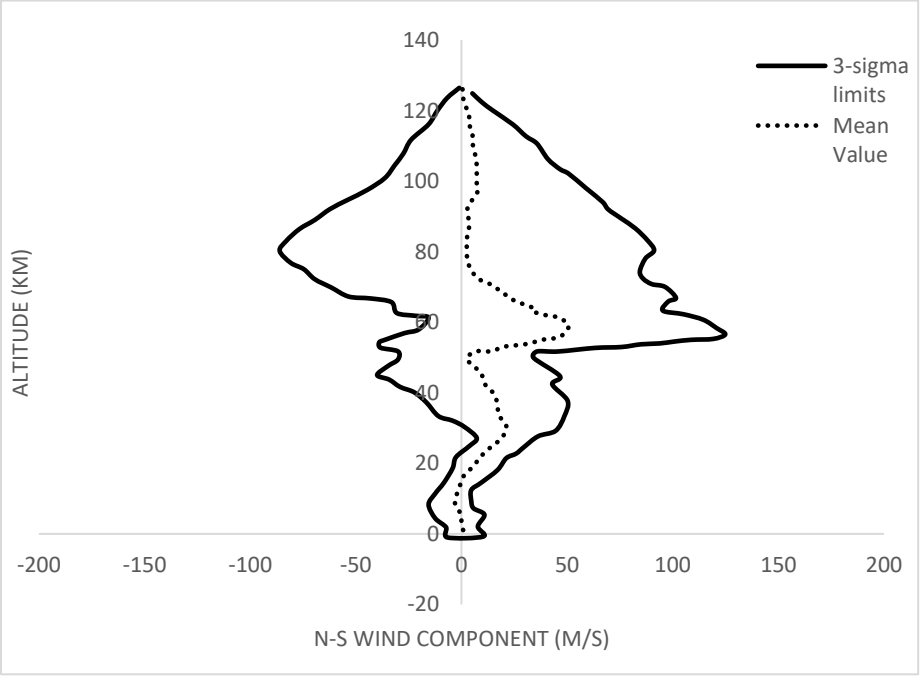

Fig. 2: Wind disturbance model

4.2 Optimization

The collocation method utilized in this analysis discretizes the trajectory from the entry interface to the point of touchdown into 50 nodes, each of which are separated by a constant timestep Δt. The number of nodes is fixed at 50 as it requires a comparatively lower computational cost while being able to capture the dynamics of the system well. The timestep is calculated by the solver, using equation (14). At each node, the equations of motion of the system are solved, while satisfying the equality and inequality constraints and staying within the bounds specified for each variable. The entire trajectory is solved simultaneously, which enables the calculation of the control variable at each node, which is the bank angle of the vehicle in this analysis. Thus, the optimal control problem is solved at 50 nodes simultaneously. Through the collocation method, a greater resolution can be achieved while studying the trajectory of the EDL vehicle, as it allows for a better understanding of the state of the vehicle at each node along the trajectory.

The discretization of the trajectory begins through defining the initial conditions and final landing conditions of the EDL vehicle, specifying the latitude and longitude where the touchdown must occur. A Chebyshev-Gauss-Lobatto quadrature formula is then utilized to create a set of initial states for the vehicle. The state vectors for this system include the altitude above the surface, velocity, longitude, latitude, flight path angle, heading angle and the time taken for the trajectory. Time is specified as a state variable to avoid restricting the feasible ranges of the solution. If time were specified as an initial and final condition, the system would be restricted to converging the solution to results that always satisfy the given value of end time, which restricts

its freedom in the broader sense. Aside from these state variables, the bank angle is the control variable in this analysis, which is optimized by the system.

The 6 state variables of the system are calculated along every node in the discretized trajectory, which adds up to 300 state values for the entire trajectory. The time state is only calculated at the end state, hence accounts for only one state value. The timestep which governs the distance between each node is then calculated using this final time. The initial values of the control variable σ are input as 50 values of 0 degrees. The initial 351 values of state and control variables are used as an initial guess by the optimizer, which utilizes the initial guess and tweaks the variables to satisfy the bounds and constraints specified. The state values in the initial guess vector is calculated through equation (45)

$$state(i+1) = slope * timestep(i) + state(1) \tag{45}$$

The timestep used in the above equation is found using the Chebyshev-Gauss-Lobatto quadrature formula which enables for optimal weighted spacing of the timesteps in each of the states, for a total of 50 nodes. The CGL algorithm assigns weights to each of the states to determine the optimal timestep. The weights for the nodes except the first and last nodes are defined as

$$Weight = \frac{\pi}{no.of\ nodes-1} \tag{46}$$

The first and last node are assigned half the weight as compared to the rest of the nodes, defined as

$$Weight' = 0.5 * \frac{\pi}{no.of\ nodes-1} \tag{47}$$

The initial guess value for time is provided at 500 seconds, to facilitate for longer simulation runtimes. The state and control vectors have lower and upper bounds which are hard boundaries

for the numeric values these variables can have. These limits are based off physical constraints such as in the case of latitude and longitude, and other factors such as maximum altitude, minimum velocity, and the equations of motion. These are discussed in length later in this text.

The initial guess vector is defined as *x0*, consisting of 351 elements. MATLAB 2015a is utilized to run this optimization, using the *fmincon* optimizer with sequential quadratic programming (SQP) as the iterative method. SQP has numerous advantages including a robust algorithm capable of handling non-linear optimization problems with even a non-linear constraint.

The drawbacks of this algorithm are the computational costs required in computing the divergence matrix. This makes the algorithm laborious to solve analytically, but it is suitable for computational applications with adequate processing power. SQP in essence is an algorithm which converts large, nonlinear problems into a sequence of smaller quadratic problems, and hence is suitable for this trajectory optimization solution.

4.2.1 Equality and inequality constraints

The constraints employed in this optimization study can be classified into 2 types: Equality constraints and inequality constraints. As discussed previously, these are numerical functions that have to be satisfied for the solution to be converged and feasible. In this trajectory optimization study, the equality constraint is the condition that the system of equations that define the EDL vehicle must satisfy. The previously discussed equations of motion of the system are solved simultaneously by the optimizer. These correspond to the state variables of r, v, θ, ϕ, γ, and ψ. Between one node and the next, these equations can be approximated as definite integrals, with the upper and lower limits being the current node and the next node, respectively. To solve these definite integrals, we deploy an integration technique which averages the left and

right definite integral approximations which are again finite sums – the trapezoidal method. The equality constraint binds the changes in the state values from one node to the next to follow the trapezoidal integration method for explicit time steps, as described in equation (48).

$$\Delta n = x_{n+1} - x_n - \frac{f(x_{n+1})+f(x_n)}{2}\Delta t = 0 \quad (48)$$

Where $\Delta t = \frac{t_{final}}{No.\ of\ nodes}$ is the timestep between consecutive nodes and *x* represents each of the state vectors. In moving from a specific node to the next, the system of equations must follow the trapezoidal integration rule, whereby consecutive values of states are dependent on both the current and next states, and the state variables at both these states. Trapezoidal integration is robust and valid in this case as the spacing is equal between nodes, ensured through the equal timesteps between consecutive nodes. The equality constraint ensures that the system dynamics do not fluctuate to extreme values between consecutive nodes and follows a valid trajectory from entry interface till touchdown. In the absence of this constraint, the function may still converge to a solution, but this would likely not follow a feasible path, violating the governing equations and having extreme values for the state variables. The vector of 350 equality constraints must be within tolerance limits of 0 for the solution to converge, and the system adjusts the control variable such that this constraint is satisfied.

The inequality constraint provided to the algorithm quantifies the amount of change in the bank angle between consecutive nodes. During the reentry procedure, excessive banking movement of the vehicle in the extreme aerothermal environment may lead to exponentially growing disturbances in the trajectory, in addition to a higher heat flux through the vehicle, which poses a risk of structural damage. Hence the vehicle is restrained in its movement throughout the EDL process, which also provides a greater level of fidelity to the simulation. The inequality

constraint limits the change in the absolute value of the control variable between consecutive nodes to no more than 5 degrees and is represented in equation (49).

$$\left|\left|\sigma(i+1) - \ \sigma(i)\right|\right| - 5 \ < \ 0 \tag{49}$$

The inequality constraint is only applied on the control variables, hence numerically the absolute value of 50 control variables should not change by more than 5 degrees between consecutive nodes. The values of 5 degrees was chosen as this best represented the real world EDL scenario, where in a timestep of ~12 seconds, the vehicle would be restricted from banking excessively to avoid overturning and/or structural damage. This constraint also ensures the bank angle does not have sharp fluctuations of magnitude during the trajectory, which are not ideal in the hypersonic regime. Hence this constraint improves the fidelity of the simulation, and a study can be conducted on the effects of increasing the allowable change in bank angle between consecutive nodes. The equality and inequality constraints are provided the *fmincon* optimizer as an input parameter.

4.2.2 Objective functions

In nonlinear programming and optimization, the objective function (also referred to as the cost function) is the function that is subject to minimization or maximization by the algorithm. It can be a linear or nonlinear function, subject to the optimization problem at hand. A feasible solution that maximizes or minimizes the provided objective function to within tolerance limits of the algorithm is called an optimal solution. *Fmincon* utilizes a minimization algorithm, hence the objective function is subject to minimization as the end goal of the algorithm. A local minimum is said to be reached when there exists some region around the local minimum where all the

function values are greater than or equal to the value of the element at that point. Mathematically this can be expressed as:

some $\delta > 0$ such that $\forall\, x \in A$ where $||x - x^*|| <= \delta$, the expression $f(x^*) <= f(x)$ holds true.

Local maxima are also defined similarly, but with the function value being higher at that point as compared to the nearby points.

Most commercial solvers and optimization algorithms aim at finding the local optimum, rather than the global optimum. The global maxima or minima is the optimum point as compared to all the available points, regardless of the region of points around it. Generally, the objective function by nature will have multiple local minima or maxima. If the function is convex, the local minimum in the interior of the function is also the global minimum, as long as the local minimum is not on the edge of the set of feasible points. If the function is nonconvex, the problem might have more than one local optimum not all of which need be global optima.

In this study, various objective functions are utilized to investigate the optimum combination of functions required for an accurate landing on the surface of Mars, while being subject to nonlinear equality and inequality constraints. The following section discusses each of these objective functions in detail.

A. Minimize deceleration

This objective function aims at minimizing the absolute value of deceleration experienced during the trajectory. Deceleration also corresponds to the aerothermal loads experienced by the vehicle, hence this objective function also acts as a minimizer for the potential structural damage experienced by the EDL vehicle. The optimization algorithm modifies the bank angle – which is the control variable – and controls the trajectory such that the deceleration is minimal.

The deceleration is calculated as the change in velocity between consecutive nodes divided by the time between 2 nodes, which is the timestep. For 50 nodes, there exists a deceleration vector of 25 elements. The absolute maximum of these values is calculated and set as the minimization function.

$$\max\left(abs\left(\frac{v(i+1)-v(i)}{\Delta t}\right)\right) \tag{50}$$

B. Maximize downrange

Another case being studied is to maximize the downrange location of the vehicle. This translates to the performance bounds of the EDL vehicle, shedding light on the farthest location from the entry interface where the vehicle can successfully touchdown or deploy the parachute.

The downrange is calculated from the initial and final latitude and longitude points, through the haversine formula. The haversine formula calculates the great-circle distance between the two points, that is, the shortest distance over the planet's surface. This formula does not account for the variations in elevation between the two points, which may increase the measured distance on the ground between the two points. The direction along which this distance is calculated is based on the heading angle at the entry interface of the EDL process, as the haversine formula is only dependent on the latitudes and longitudes. Equation (56) is the expression utilized to find the downrange distance:

$$\Delta\, lat = Initial\ latitude - Final\ latitude \tag{51}$$

$$\Delta\, long = Initial\ longitude - Final\ longitude \tag{52}$$

$$Hav(lat) = \frac{1-\cos(\Delta\, lat)}{2} \tag{53}$$

$$Hav(long) = \frac{1-\cos(\Delta\ long)}{2} \quad (54)$$

$$c\ = 1 - 2 * (Hav(lat) + \left(\cos(lat_i) * \cos\left(lat_f\right)\right) * Hav(long)) \quad (55)$$

$$Downrange = R_{Mars} * c \quad (56)$$

Here, $Hav(lat)$ and $Hav(long)$ represent the Haversine angles of latitude and longitude respectively. R_{Mars} is the radius of Mars, and lat_i and lat_f represent the initial and final values of latitude, respectively.

C. Minimize error at final state

This objective function focuses on minimizing the error between the end latitude and longitude of the optimized trajectory, and a specified landing coordinate. The optimizer works at reducing this error to within the tolerances of the solver used in *fmincon* while satisfying the equality constraint of landing at a specific altitude.

The optimization algorithm calculates the trajectory such that the end latitude and longitude are close to a specified coordinate, and adjusts the bank angle which affects the rest of the state variables, in order to achieve this final position.

4.3 Bounds

The collocation method used in this study is subject to bounds for its state variables, namely altitude, velocity, latitude, longitude, flight path angle, and heading angle. The control variable bank angle is also bounded in this analysis. Bounds are applied to optimization problems to include specific limits between which the solution must stay. These might be to ensure that the solution to a particular problem reaches the desired state within some desired values of state

variables. Bounds are also applied to control the feasibility of the solution, and to ensure that the state variables and system dynamics remain real values, following physical principles. Mathematically, bounds can be expressed as

$$t_{lower} \leq t_0 < t_{final} \leq t_{upper}, \text{ bounds on initial and final time,}$$

$$x0_{lower} \leq x(t0) \leq x0_{upper}, \text{ bound on initial state,}$$

$$xF_{lower} \leq x(t_f) \leq xF_{upper}, \text{ bounds on the final state values.}$$

Table 1 displays the bounds used for the state and control variables:

	Altitude above surface (m)	Velocity (m/s)	Longitude (deg)	Latitude (deg)	Flight path angle (deg)	Heading (deg)	Time(s)	Bank Angle (deg)
Lower Bound	0	100	-180	-90	-90	-90	100	-90
Upper Bound	150,000	7000	180	90	90	90	2000	90

Table 1. Bounds for state and control variables

4.4 Model Predictive Control Implementation

The model predictive controller (MPC) implemented in this study works at mitigating atmospheric wind disturbances and optimizing the trajectory towards to end state while correcting these disturbances. The predictor-corrector algorithm propagates the trajectory for a defined number of nodes while subject to atmospheric disturbances. At the end of this specified propagation, the current state will differ from the optimal state required at that specific node. The optimization algorithm is then implemented from the offset state which calculates again calculates the optimal trajectory to the target location. The bank angle vector is hence optimized

multiple times, and the optimal bank angle at the current state is then stored. At the end of the MPC process, the new optimized vector of bank angles will differ from the collocation result without disturbances, at all the nodes the MPC is run at.

The optimizer utilized at each of these offset nodes use the collocation method to split the trajectory to the end state into 50 nodes, thus preserving the resolution of the original collocation method used to determine the optimal bank angle vector.

4.4.1 Trajectory Propagation

For the trajectory propagation, the system of equations is solved using the *ode45* solver in MATLAB, which utilizes the Runge-Kutta 4^{th} order methodology to solve a set of ordinary differential equations over a specified timestep, which in this case is calculated using equation (14). The system of equations provided to the ODE solver is also provided the previously calculated optimal bank angles at each corresponding node. However due to the NS-wind component, the end state is offset at the end of each ODE solver run. The wind component affects the position and velocity values throughout the propagation, which in turn affects the other state variables, such as the flight path angle and the heading angle.

4.4.2 Optimizer

After the trajectory has been propagated, the optimizer *fmincon* is utilized once again, to calculate the optimal trajectory from the current node to the end state. Depending upon the objective function and the constraints applied, a new optimal bank angle is calculated, which in essence takes allows the vehicle to overcome the disturbances present and still achieve a desired end state. The collocation method discussed in the previous chapter is once again implemented, at each of the nodes selected to perform the MPC. Figure 3 shows the MPC flowchart.

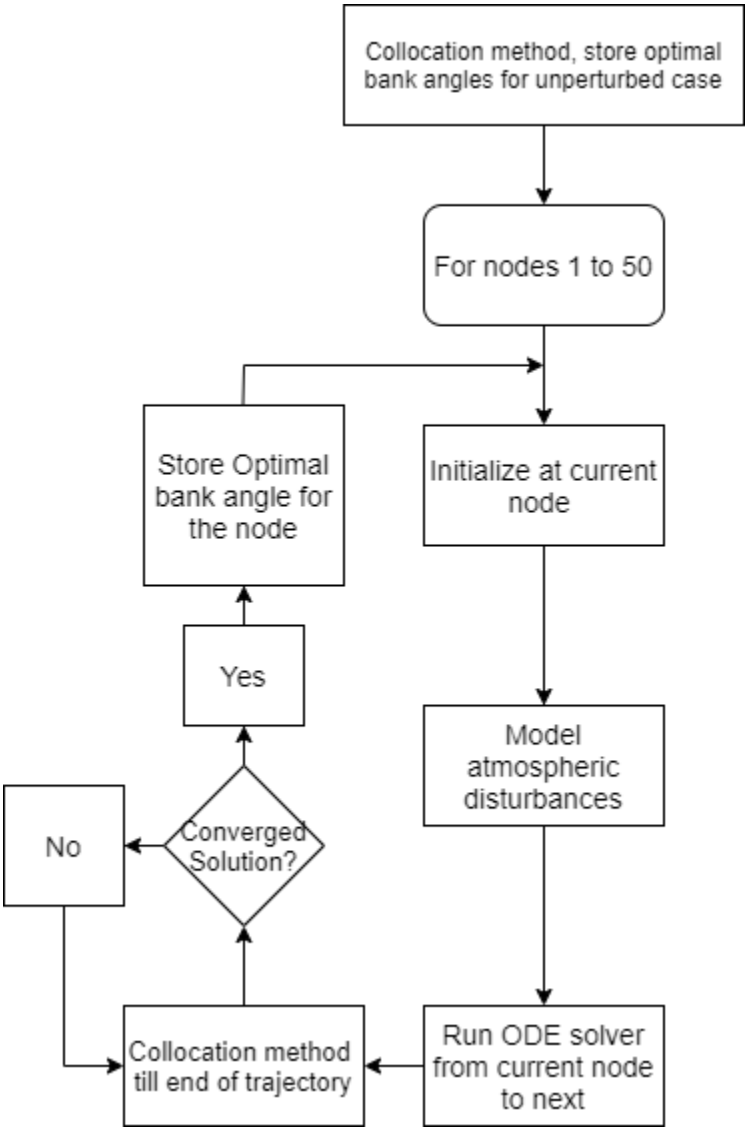

Figure 3: Model predictive controller

Chapter 5: Results

A number of analyses have been done incorporating the collocation method and model predictive controller. These include single-trajectory and multi-trajectory studies, with Monte Carlo analyses for each. All the analyses have been done on MATLAB^{R} 2015a, on a single PC running on an Intel Core i5-8520U CPU at 1.8Ghz.

5.1 Single Objective Function

For establishing a baseline for the collocation method, a trajectory is run from the entry interface to the final state, employing a single objective function of minimizing error at the final state. Table 2 describes the initial and final target states. The initial states are provided to the optimizer as equality constraints, hence these have to be satisfied for a converged solution.

	Altitude above surface (m)	Velocity (m/s)	Longitude (deg)	Latitude (deg)	Flight path angle (deg)	Heading (deg)	Time(s)	Bank Angle (deg)
Initial State	120,000	5,505	15^{o} E	30^{o} N	-16	90	0	0
Final State	1,500	30	26.17^{o} E	60.03^{o} N	90	90	600	90

Table 2. Initial and Final states

Here, the final values of flight path angle, heading angle, time, and bank angle are not hard constraints. i.e., the optimizer determines the best values for these end states, as is computes the optimal trajectory. Since time is a state variable itself, the end time state is determined to suit the best timestep for the trajectory, as seen in equation (14). The bank angle being the control

variable for this analysis, is determined by the optimizer, while satisfying the inequality constraint as outlined in equation(49), which limits the change between consecutive bank angles to below 5 degrees between timesteps. The following are the results of the analysis:

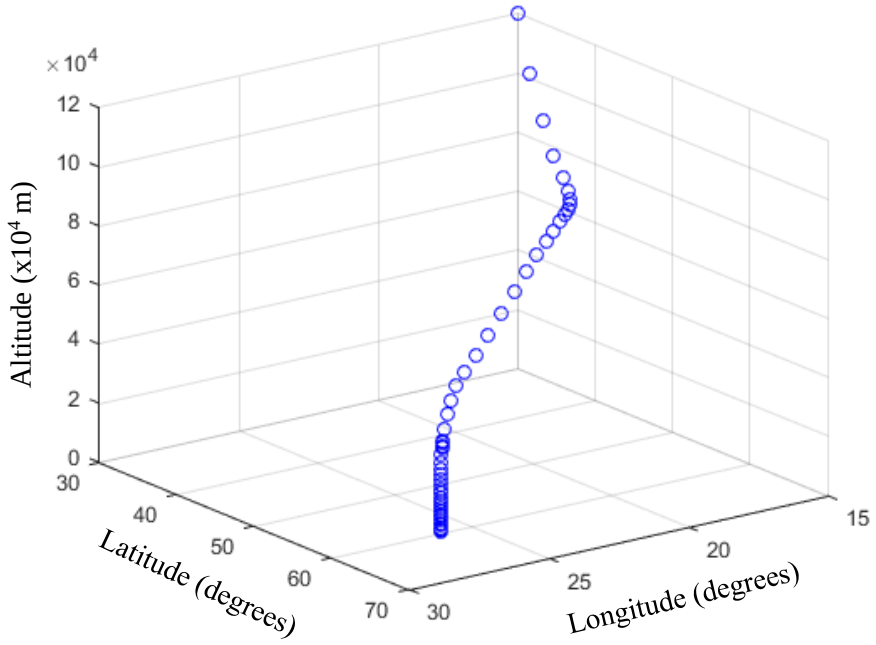

Figure 4: Baseline Trajectory

This trajectory is used as a baseline for further analysis. A few features of the baseline trajectory can be highlighted to demonstrate the performance of the system. Around node 10 which is 80 km above the surface, the EDL vehicle modelled starts to lift up, as seen in the figure(4). This is due to a few factors, such as the exponential increase in density at lower altitudes, and the flight path angle being shallow, which affects the system dynamics model. Around node 10 is also the point of peak deceleration, as seen in figure 12. This point is also where maximum lift occurs, and this contributes to the lifting behavior of the system.

To get a better understanding of the performance limits of this trajectory, disturbances have been added at each node corresponding to the NS wind component, as seen in figure (3). With the addition of these disturbances, slight uncertainties also occur at the entry interface of the EDL vehicle, and a 100 element Monte Carlo analysis has been conducted to find a distribution of the most probable points of the end state.

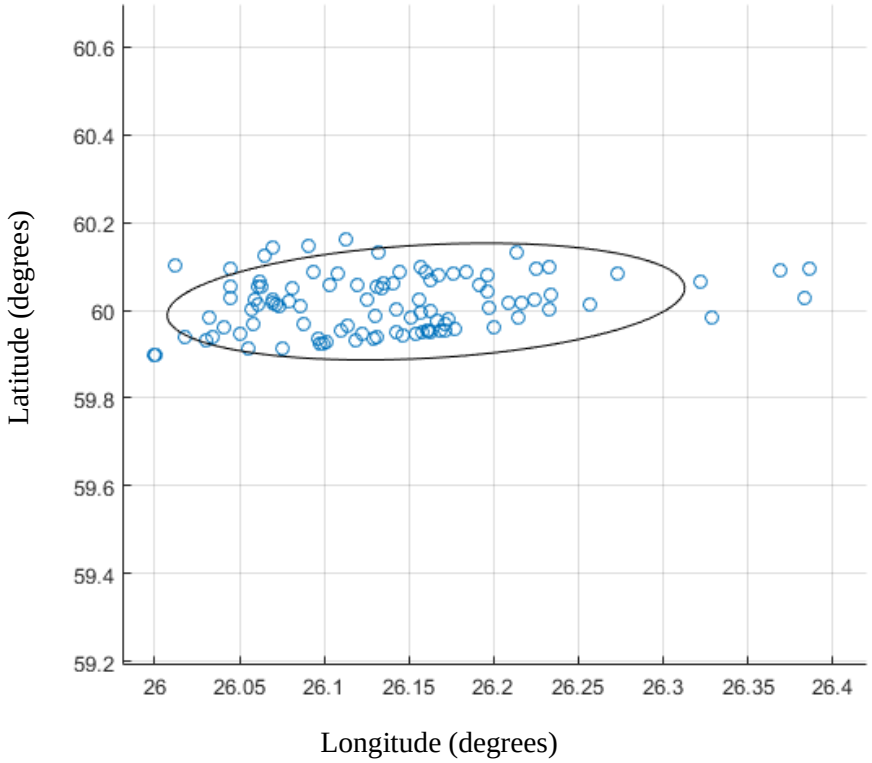

Figure 5: Monte Carlo analysis of baseline trajectory

As seen in the Monte Carlo analysis, the end states of the trajectory are dispersed from the target location of 26.17^{o} E 60.03^{o} N, with the objective function being minimize error to target. This and further Monte Carlo analyses done are of the 1-sigma variation, as opposed to 3-sigma Monte Carlo analyses done with higher computational power. For the purposes of this study, the 1-sigma analysis was deemed satisfactory.

As the vehicle travels from the entry interface to the final state, the system has to satisfy the equality and inequality constraints, while minimizing the error to the target. Due to this, the final

location is off by a magnitude of tens of kilometers. In order to mitigate this large error ellipse, further steps have been taken to increase the controllability of the spacecraft during the final stages of descent.

5.2 Multiple Objective Functions

In order to obtain better control during the final phases of the trajectory, the EDL process has been split into two distinct trajectories. Through imparting different objective functions, different goals can be met for each of the trajectories. For the first objective function, the main objective is to reduce the aerothermal loading experienced by the vehicle. Since the entry interface velocity is around 5,500 m/s, the vehicle experiences large aerodynamic forces and heat loads during this first entry and descent phase. Therefore, the objective function chosen for this case is to minimize the deceleration experienced by the vehicle. As discussed in the previous section, the minimize deceleration objective function calculates the absolute maximum value of deceleration experienced during the trajectory and tries to minimize this value. The initial conditions and entry interface are the same as described in Table 2. The following are the details of the first trajectory.

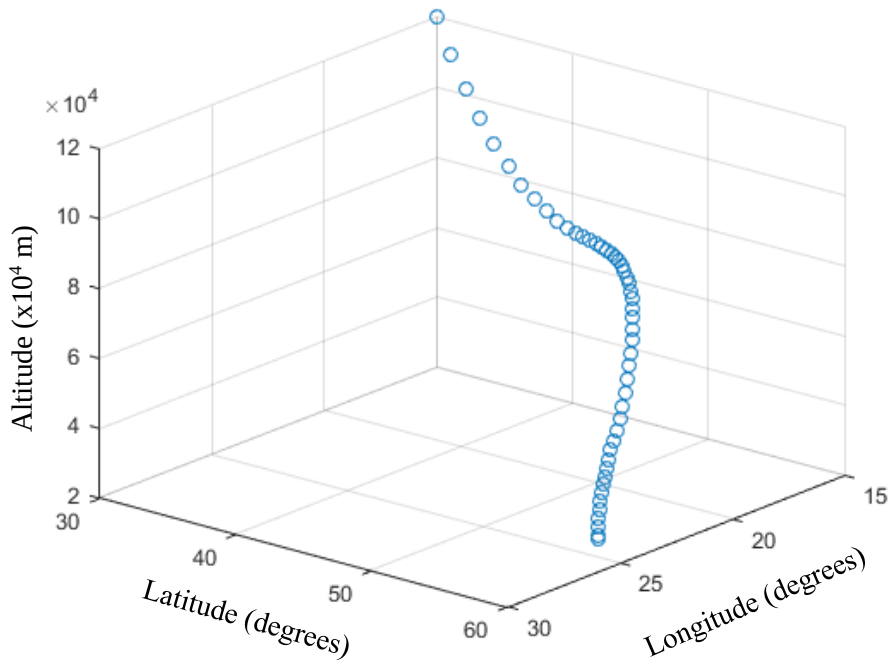

Figure 6: Trajectory 1

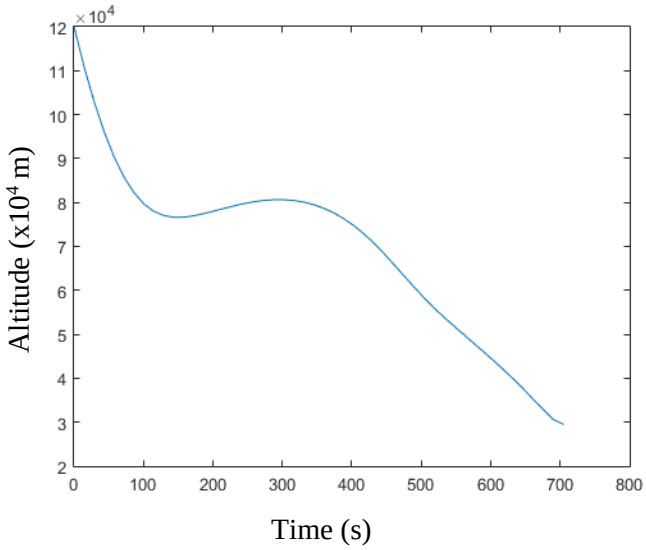

Figure 7: Altitude vs. Time for Trajectory 1

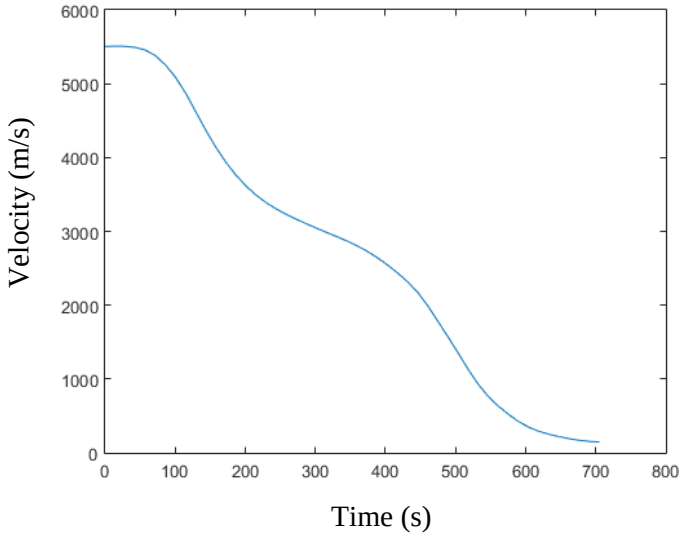

Figure 8: Velocity vs. Time for Trajectory 1

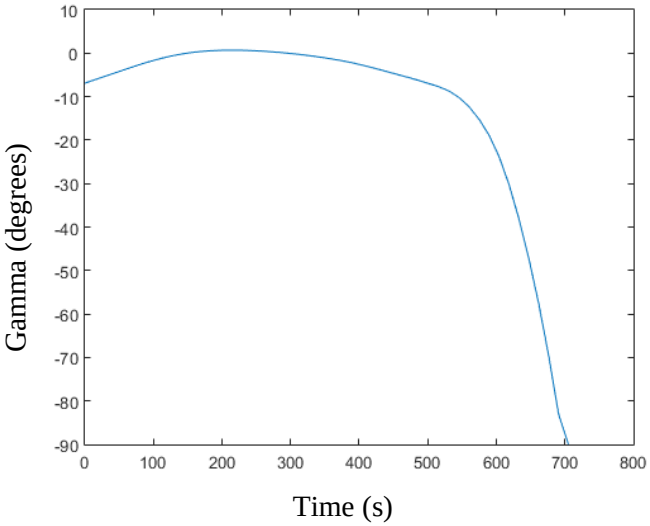

Figure 9: Flight Path Angle vs. Time for Trajectory 1

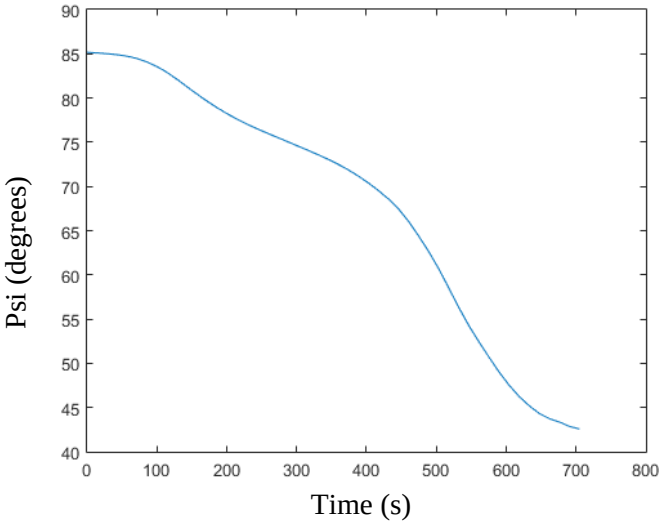

Figure 10: Heading Angle vs. Time for Trajectory 1

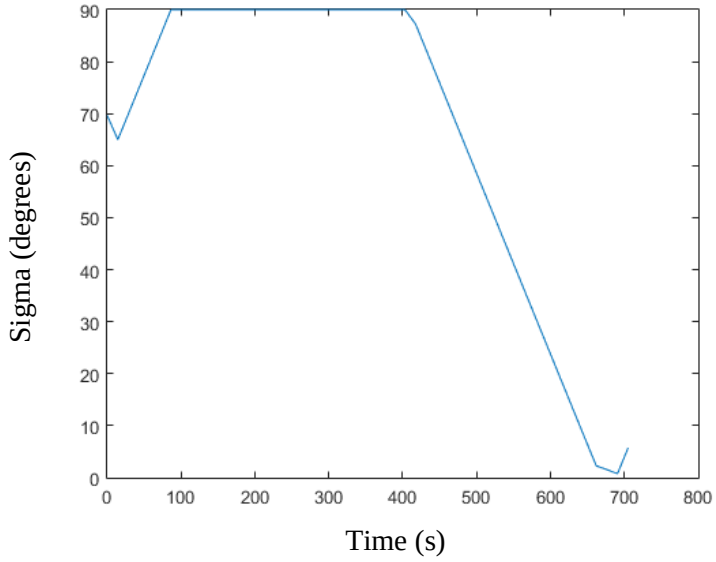

Figure 11: Bank angle vs. Time for Trajectory 1

Utilizing a minimize max deceleration objective function as compared to the single trajectory is depicted in figure (12), which shows a lower value of maximum deceleration for trajectory 1 (plotted above). This ensures that the vehicle stays withing it's aerodynamic loading limits, and this may be further investigated if a specific material design of the EDL vehicle is implemented.

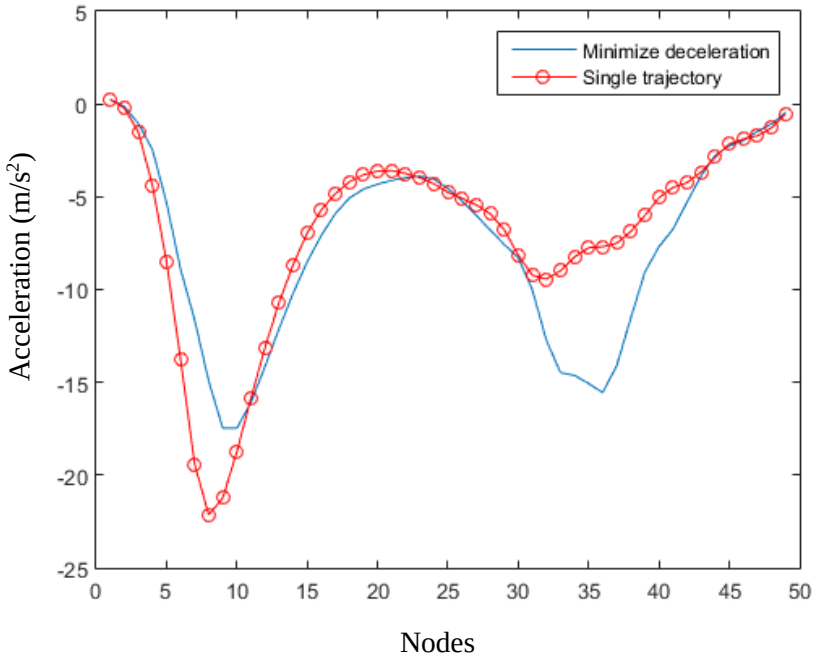

Figure 12: Deceleration comparison

The final time state is determined by the optimizer and is chosen in such a way as to equally distribute the nodes along the trajectory. At the point where the first trajectory ends and the second begins, the state values of altitude, velocity, latitude, longitude, and heading angle are continued from 1 trajectory to the next. These are stated as equality constraints in the second trajectory, and therefore need to be satisfied for the solution to converge. This ensures that the continuity of the trajectory is preserved. Where trajectory 1 ends is determined by its equality constraints. A latitude and longitude close to the final location is provided as an equality constraint to the optimizer, which enables the optimizer to reach near the final location with

accuracy. The altitude at this tradeoff point is not specified to the optimizer, and is left as a degree of freedom at this point.

For the second trajectory, the objective function is changed, to minimizing final state error. This allows for 50 nodes which are all optimized to get the system as close as possible to the final target. The results of the 2nd trajectory are as follows:

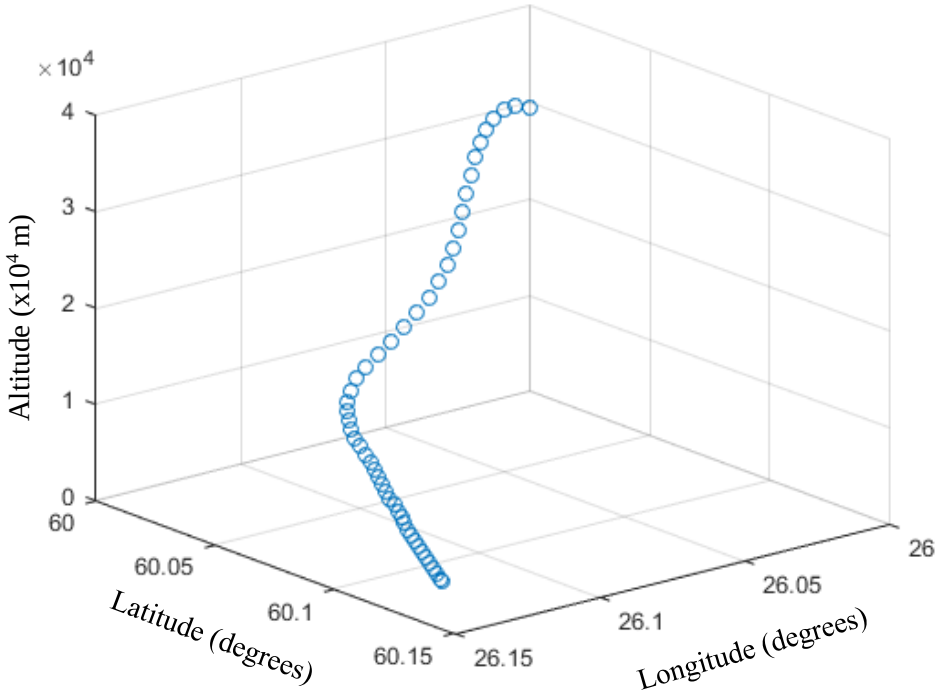

Figure 13: Trajectory 2

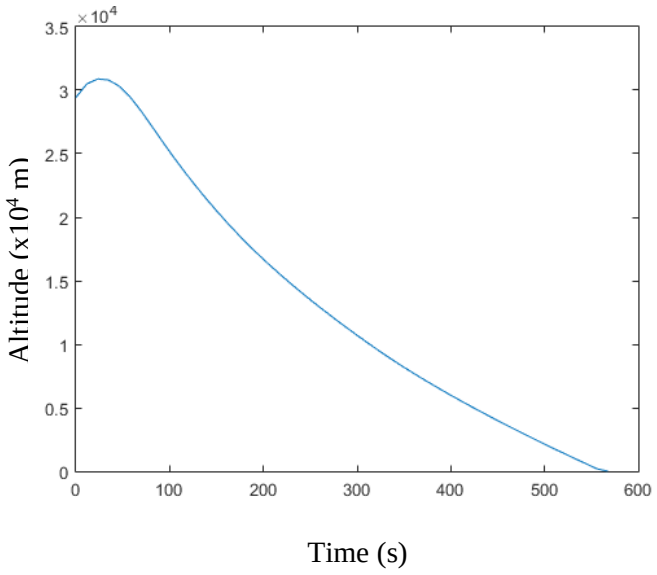

Figure 14: Altitude vs. Time for Trajectory 2

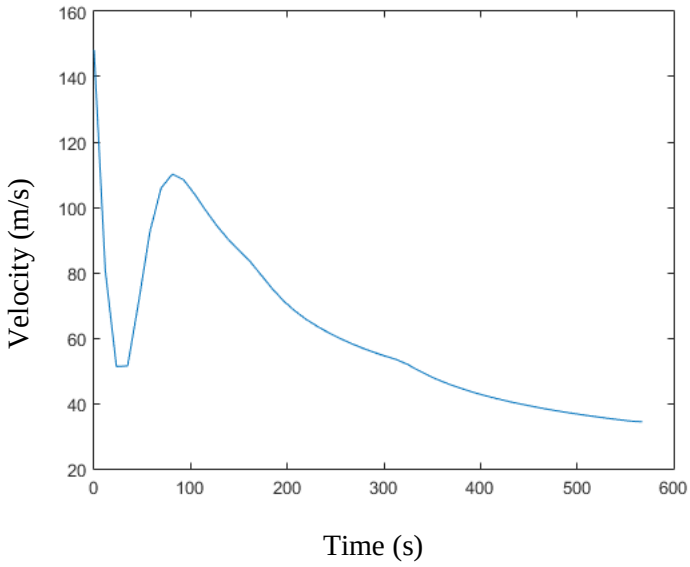

Figure 15: Velocity vs. Time for Trajectory 2

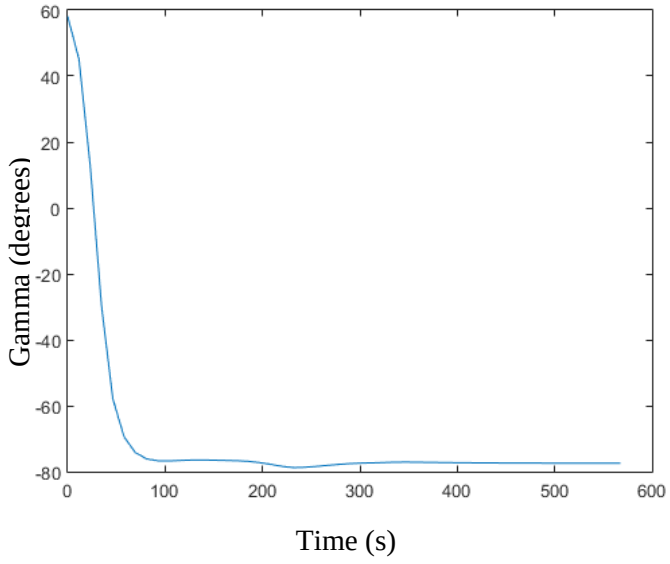

Figure 16: Flight Path Angle vs. Time for Trajectory 2

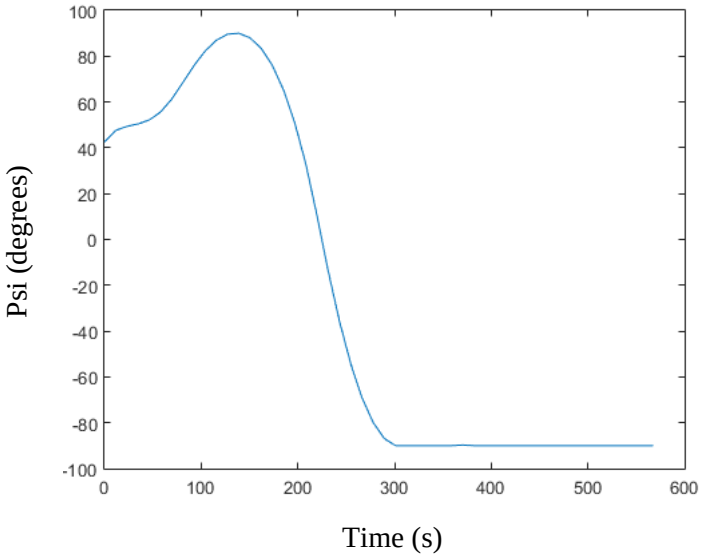

Figure 17: Heading vs. Time for Trajectory 2

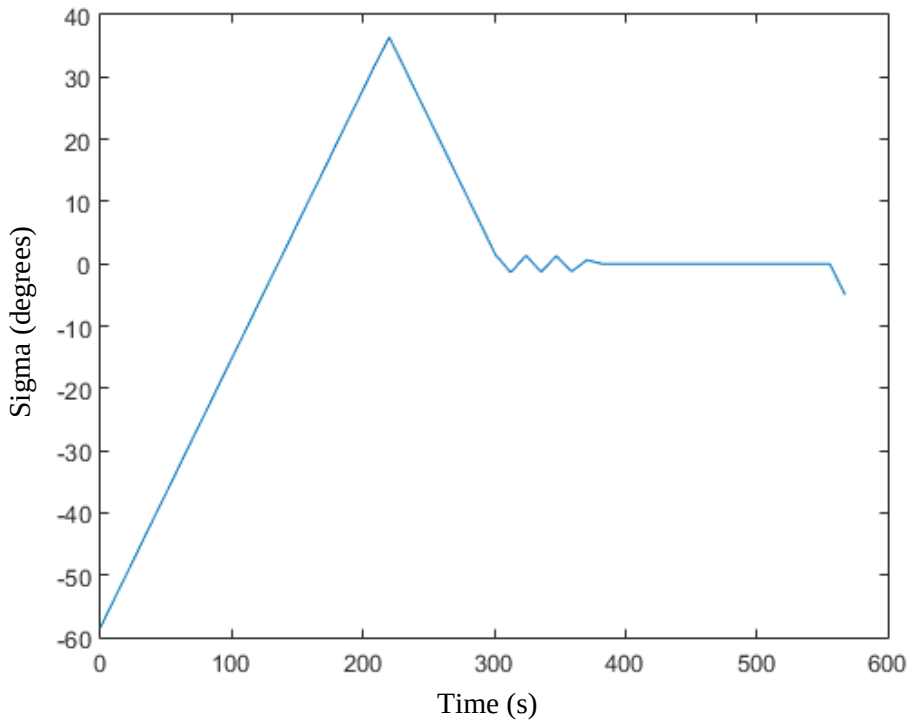

Figure 18: Bank Angle vs. Time for Trajectory 2

A Monte Carlo analysis is done on the second trajectory, to incorporate the wind disturbances as well as the uncertainty at its first node. i.e., the uncertainty at the point where the objective functions switch from minimizing deceleration to minimizing final state error.

A multi-trajectory study has been conducted to better understand the performance boundaries of the second trajectory. To find out the maximum downrange distance that can be travelled by the vehicle from the start of the second trajectory, the objective function has been switched to that of maximizing downrange distance, as seen in equation (56). The remaining trajectories are

specified various other points that are further from the target location, and a trail-and-error analysis is done to determine the performance limits of this trajectory.

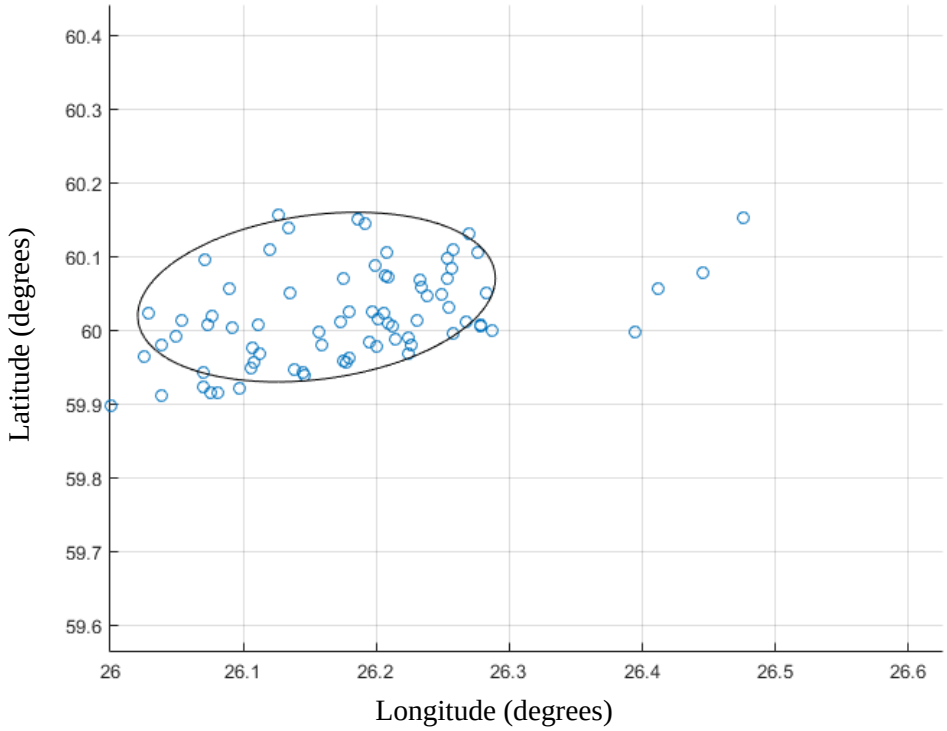

Figure 19: Monte Carlo analysis for Trajectory 2

Comparing the 2 Monte Carlo Analysis done, it is evident that splitting up the trajectory decreases the error from the target landing location, as seen in figure (20).This is a result of better controllability provided to the vehicle during the final stages of landing, as it has a new set of 50 nodes along which the optimal bank angle is calculated. Providing this increased control to the vehicle results in a closer grouping of landing points, as compared to the baseline trajectory.

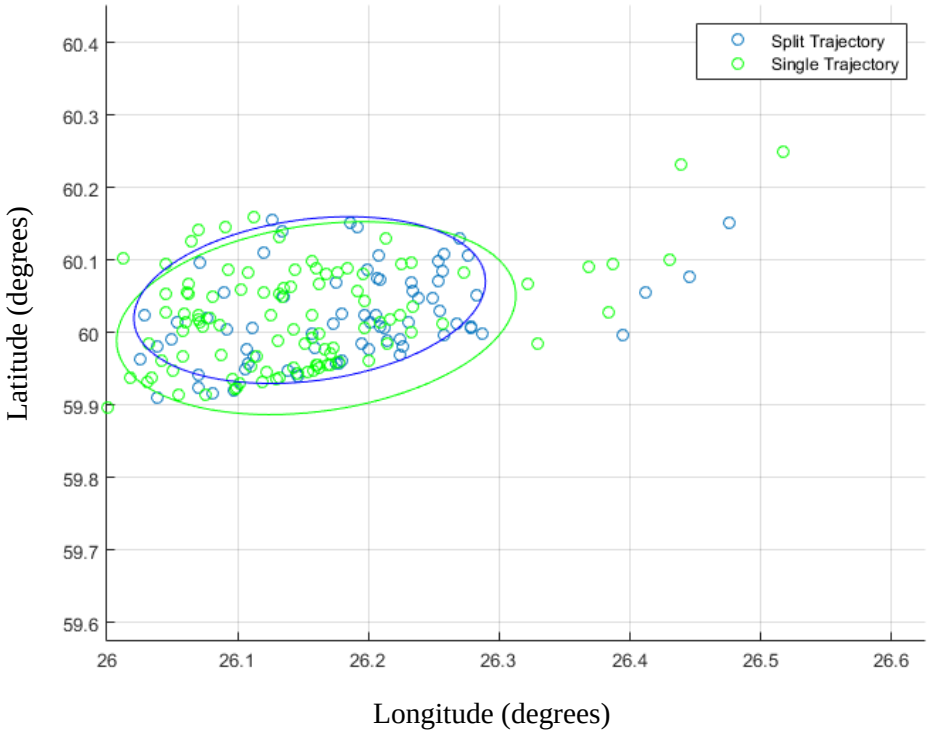

Figure 20: Monte Carlo Analysis Comparison

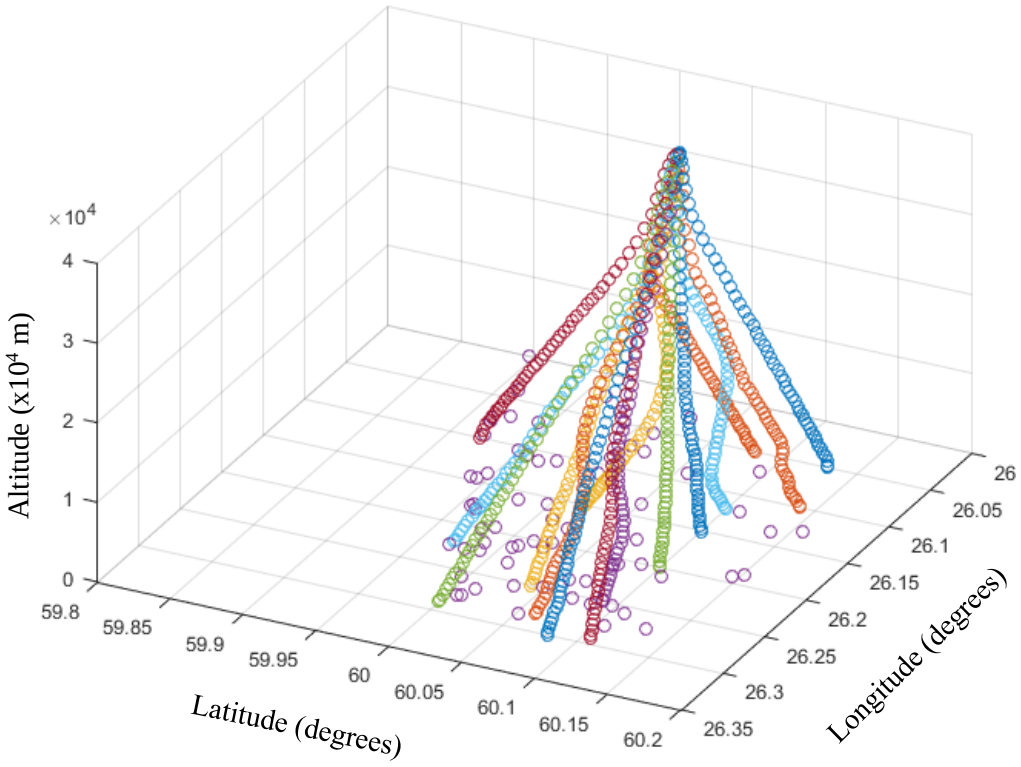

Figure 21: Multi-trajectory performance analysis for Trajectory 2

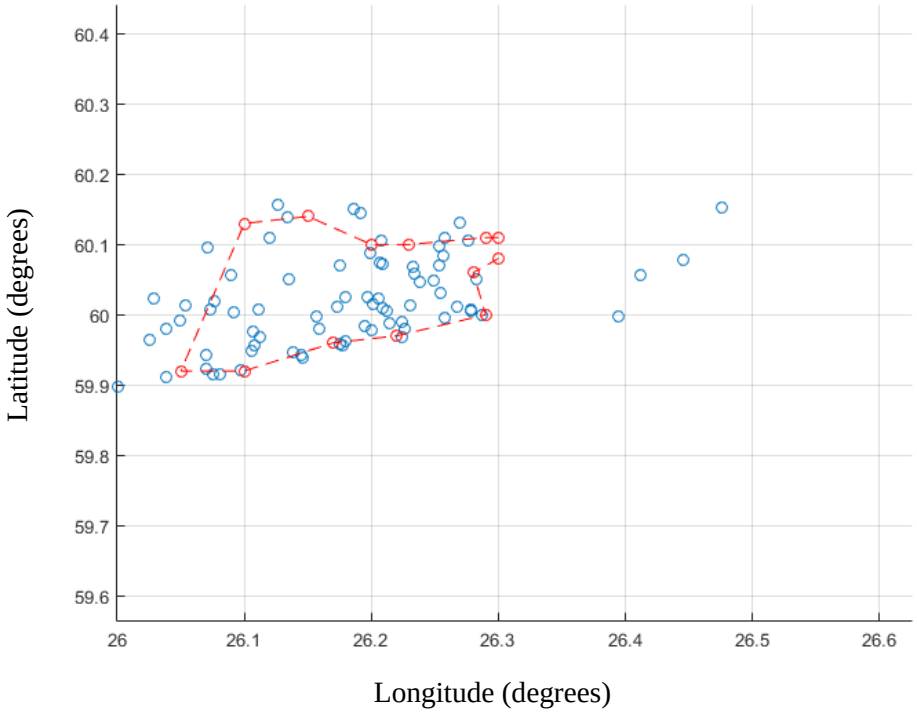

Figure 22: Performance limits for Trajectory 2

From the Monte Carlo analysis, it is evident that the disturbances cause the final location of the lander to be off from the target location of 26.17° E 60.03° N, although having a closer distribution to the target, as compared to figure (5). Although some points appear outside the performance boundaries of the 2nd trajectory, these are few, and are a result of the perturbed initial state of the 2nd trajectory, which may have been at a higher altitude, hence accounting for further end states. The majority of the end states from the Monte Carlo analysis lie within the performance bounds derived above.

In order to overcome the error causing disturbances, the model predictive controller is now implemented to the system. As discussed previously, the MPC models the perturbed trajectory,

and optimizes from the end node of the perturbed trajectory to the required end state. Due to the excessive computational costs of implementing the MPC at every timestep, it is instead run at every other node, i.e. at 25 nodes in total. This does not sacrifice a lot in terms of error mitigation, as each timestep is around 12 seconds, which implies the MPC will be run every 24 seconds along the trajectory. The MPC is implemented to both trajectories 1 and 2, to investigate the effectiveness of the error correction. A 20 element Monte Carlo analysis is done of trajectory 2, to study the landing probability ellipse with the model predictive controller enabled.

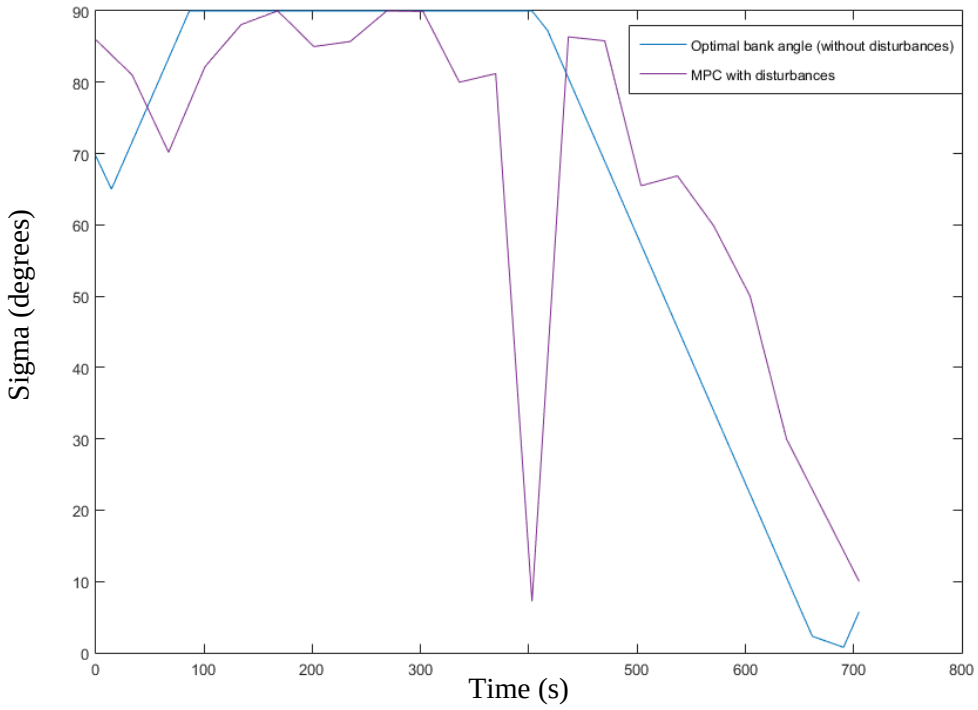

Figure 23:Example of bank angles with and without Model Predictive Control and disturbances vs. time for trajectory 1

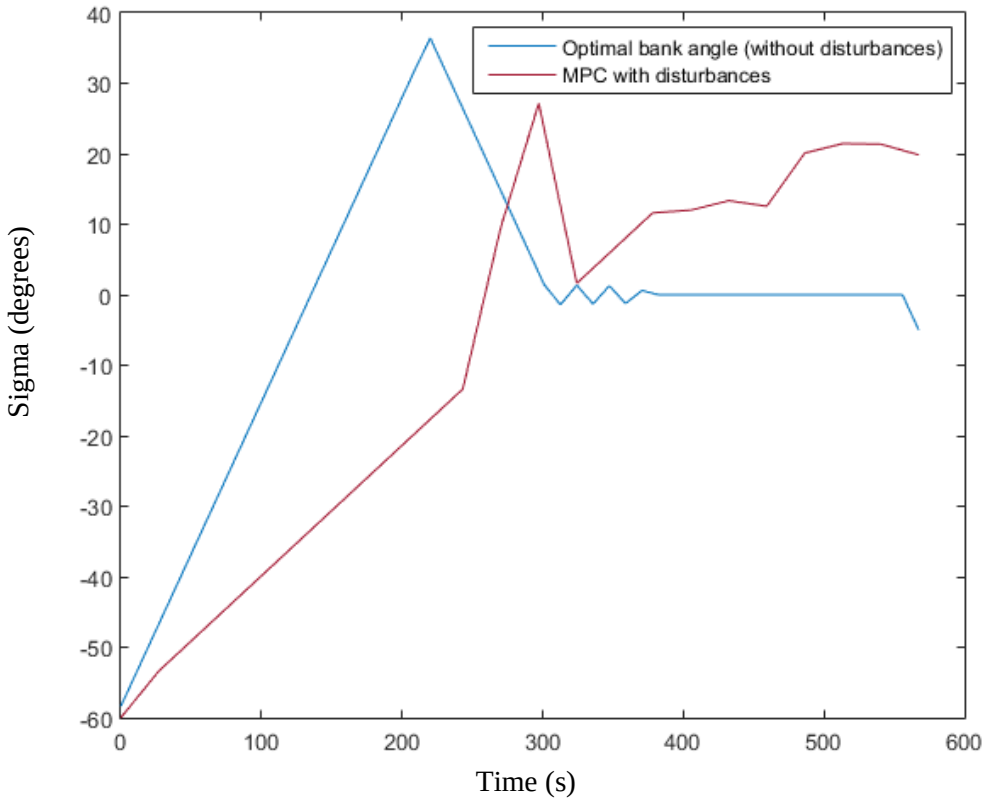

Figure 24: Example of bank angles with and without Model Predictive Control and disturbances vs. time for trajectory 2

The new set of optimal bank angles from the model predictive controller differ from the control vector without MPC, as the disturbances experienced at each 2nd node being taken into account and the trajectory is then optimized again till the final state. This results in variations in the bank angle at these nodes, as depicted in the above figures. Due to the random nature of wind disturbances modelled into the system, no two MPC implementations result in exactly similar solutions. The above figure is an example of one implementation of the MPC.

The final latitude and longitudes from the Monte Carlo analysis done for both the trajectories with MPC enabled is depicted below.

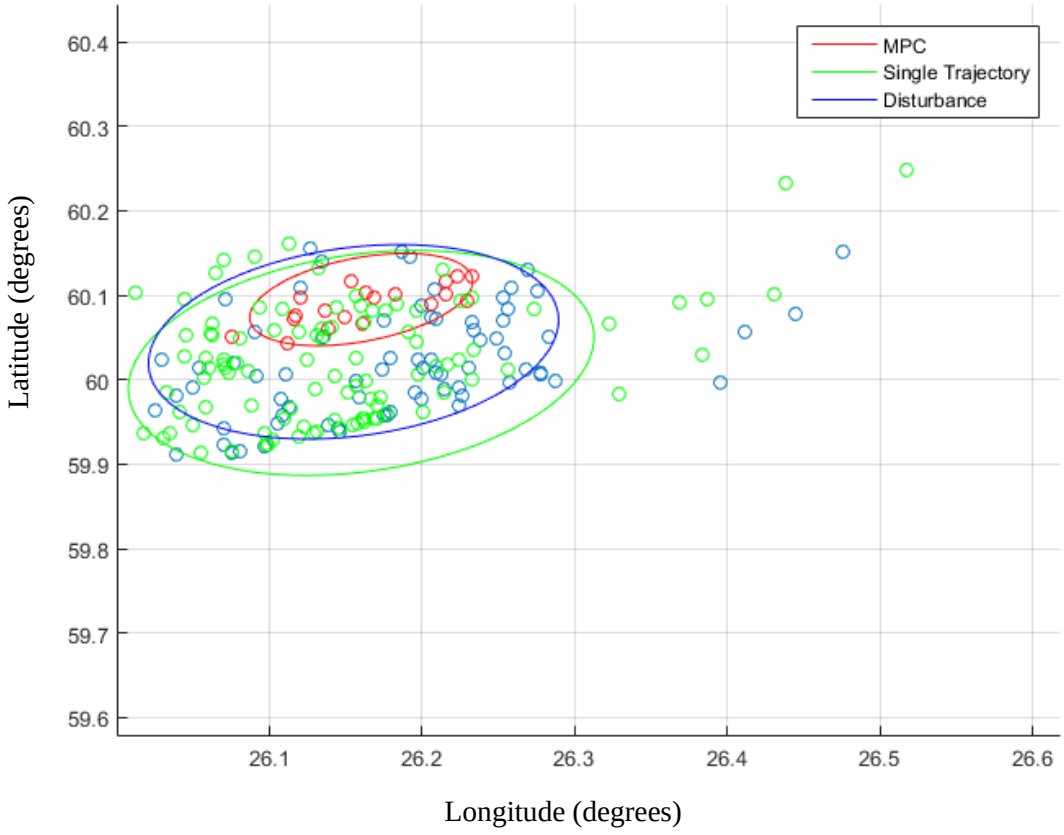

Figure 25: Monte Carlo Analysis comparison

As evident from the landing probability ellipse in figure (25), the vehicle has achieved a final value of latitude and longitude closer to the targeted values of 26.17^{o} E and 60.03^{o} N when the model predictive controller is applied to the system. Although the Monte Carlo analysis consists on 20 elements, a multivariate random number analysis can be conducted on these values to determine the most probable landing ellipse for an analysis with a higher population. This technique is used mathematically to populate lower element sets of data, with the covariance of the population being known. The Monte Carlo analysis conducted in this study does not look at all potential disturbances that may be experienced by the vehicle, only at the discrepancies in states caused by a wind component.

Chapter 6: Conclusion

This study investigated the effect of an optimization algorithm coupled with a model predictive controller on the accuracy of landing on the surface of Mars. From this analysis, it is evident that a model predictive controller helps improve the accuracy of the end state of the EDL vehicle, as the optimization algorithms are subject to uncertainties due to the atmospheric winds on the planet. The objective function employed by the optimizer also plays a key role in improving the accuracy of the final state, as evident from the results. Splitting up the entire entry-descent-landing process into 2 separate trajectories for better control and analysis plays a vital role in increasing the accuracy of the final state, as the 2^{nd} trajectory is devoted to reducing the final state error. Through this study, a robust model predictive controller algorithm has been developed which is implemented in conjunction with an optimizer, and this system of non-linear programming methods combine to give a landing probability ellipse that is smaller and closer to the target location, as opposed to without them. Although some calculated assumptions were made in this analysis, the level of fidelity is given priority, through the incorporation of wind disturbances and position uncertainties.

Computational costs are a drawback of this type of analysis, with a 20 element Monte Carlo analysis performed with the model predictive controller run over 25 nodes along the two trajectories requiring 1000 optimizer runs, which is incredibly time consuming and computationally expensive without powerful hardware. A psuedospectral analysis is another method through which a similar study can be conducted, which analyses multiple trajectories at once, determining the best possible solution which satisfies given path and dynamics constraints and is not as computationally expensive compared to direct collocation.

Further improvements can be done to this study through the inclusion of an angle of attack control parameter, in conjunction with a better model for the coefficient of drag experienced by the vehicle throughout the trajectory. A 3-dimensional model of a re-entry vehicle with surfaces where the coefficient of pressure is calculated through the Newtonian flow technique would improve the level of fidelity of this study substantially. As a long term goal, this study may be viewed as a stepping stone towards complete autonomy of an EDL vehicle, through which a landing site can be selected by the agent itself based on the current state of the vehicle, the atmospheric disturbances experienced, and the scientific output potential of potential landing sites in range.

This study demonstrates a set of numerical methods through which an investigation is conducted in reducing the probability of landing ellipse size, under certain educated assumptions. The collocation method is a powerful tool by which trajectory analysis is conducted to find the optimal control vector for a specific targeted landing, and the model predictive controller has been proven to be capable of mitigating disturbances experienced by the spacecraft, while conforming to the physical laws followed by the trajectory.

References

[1] Benito Manrique J. “Advances in spacecraft atmospheric entry guidance”. *PhD dissertation, Irvine: University of California,* 2010.

[2] Dierlam TA.” Entry vehicle performance analysis and atmospheric guidance algorithm for precision landing on Mars”. *PhD dissertation, Massachusetts Institute of Technology,* 1990

[3] Wingrove RC. “Survey of atmosphere re-entry guidance and control methods”. *AIAA Journal* 1963;1(9):2019–29.

[4] French JR. “Trends in unmanned planetary entry systems.” In: *AIAA 16th Thermophysics conference*, AIAA-81-1125; 1981

[5] Seiff A. “Developments in entry vehicle technology.” In*: 1st AIAA annual meeting, AIAA* 64-528; 1964

[6] Thurman SW, Pollmeier VM. “Guidance and navigation for the Mars Pathfinder mission.” *National Aeronautics and Space Administration (NASA)* 1994 (IAA-L-0604P)

[7] Steltzner A, Desai P, Lee W, Bruno R. “The Mars exploration rovers entry descent and landing and the use of aerodynamic decelerators.” In: *17th AIAA aerodynamic decelerator systems technology conference and seminar,* AIAA- 2003-2125; 2003.

[8] Grover MR, Cichy BD, Desai PN. “Overview of the Phoenix entry, descent, and landing system architecture*.” Journal of Spacecraft and Rockets* 2011;48(5):706–12.

[9] Dyakonov AA, Glass CE, Desai PN, Van Norman JW. “Analysis of effectiveness of Phoenix entry reaction control system*.” Journal of Spacecraft and Rockets* 2011;48 (5):746–55

[10] Ingoldby RN. “Guidance and control system design of the Viking planetary lander.” *Journal of Guidance and Control 1978; Vol. 1(3),* pp. 189–96

[11] Lockwood MK. “Introduction: Mars Science Laboratory: the next generation of Mars landers.” *Journal of Spacecraft Rockets* 2006;43(2):257

[12] Way DW, Davisy JL, Shidner JD. “Assessment of the Mars science laboratory entry, descent, and landing simulation.” In: *23rd AAS/AIAA space flight mechanics meeting,* AAS 13-420; 2013.

[13] Way DW, Powell RW, Chen A, Steltzner AD, Martin AMS, Burkhart PD, et al. “Mars Science Laboratory: entry, descent, and landing system performance.” In: *2007 IEEE aerospace conference*; 2007

[14] Garcıa-Llama E. “Apollo-derived terminal control for bank-modulated Mars entries with altitude maximization.” In: *AIAA guidance, navigation and control conference and exhibit,* AIAA 2008-6819; 2008

[15] Murray JE, Tartabini PV. “Development of a Mars airplane entry, descent, and flight trajectory.” In: *39th AIAA aerospace sciences meeting and exhibit,* AIAA 2001-0839; 2001

[16] Braun RD, Wright HS, Croom MA, Levine JS, Spencer DA.” The Mars airplane: a credible science platform.” In: *2004 IEEE aerospace conference*; 2004

[17] Braun RD, Wright HS, Croom MA, Levine JS, Spencer DA. “Design of the ARES Mars airplane and mission architecture.” *Journal of Spacecraft and Rockets* 2006;43 (5):1026–34

[18] Shimoyama K, Oyama A, Fujii K. “Multi-objective six sigma approach applied to robust airfoil design for Mars airplane.” In: *48th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference,* AIAA 2007-1966; 2007

[19] Spencer DA, Braun RD. “Mars Pathfinder atmospheric entry: trajectory design and dispersion analysis.” *Journal of Spacecraft and Rockets* 1996;33(5):670–6

[20] Schmidt R. “Mars Express—ESA's first mission to planet Mars.” *Acta Astronautica* 2003;52(2):197–202

[21] Cianciolo, A. D., Powell, R. W., “Entry Descent and Landing Guidance and Control Approaches to Satisfy Mars Human Mission Landing Criteria.” *AIAA Space Flight Mechanics Meeting*, February 2017

[22] Balaram, J., Austin, R., Banerjee, P.,Bentley, T., Henriquez, D., Martin, B., McMahon, E., Sohl, G. “ DSENDS - A high-fidelity dynamics and spacecraft simulator for entry, descent and surface landing”, *IEEE Aerospace Conference Proceedings,* 2002, *Vol. 7, pp. 3389-3352*

[23] Davis, Jody L., Cianciolo, Alicia Dwyer, Powell, Richard W., Shidner, Jeremy D., García-Llama, Eduardo, “Guidance and control algorithms for the mars entry, descent and landing systems analysis”, *AIAA/AAS Astrodynamics Specialist Conference 2010*

[24] Korzun, Ashley M., Dubos, Gregory F., Iwata, Curtis K., Stahl, Benjamin A., Quicksall, John J., “A concept for the entry, descent, and landing of high-mass payloads at Mars”, *Acta Astronautica,* 2010, Vol. 66 Issue 7-8. Doi: 10.1016/j.actaastro.2009.10.003

[25] Balaram, J., “Sherpa Moving Mass Entry Descent Landing System”, *Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2005.*

[26] Huneker, L., Sagliano, Marco, Arslantas, Yunus E., “SPARTAN: An Improved Global Pseudospectral Algorithm for High-Fidelity Entry-Descent-Landing Guidance Analysis”, *30th ISTS Conference, Kobe, Japan,* 2015.

[27] Desai, Prasun N., Lyons, Dan T., Tooley, Jeff, Kangas, Julie, "Entry, descent, and landing operations analysis for the stardust entry capsule", *Journal of Spacecraft and Rockets*, 2008. Doi: 10.2514/1.37090

[28] Calhoun, Philip C, Queen, Eric M., "Entry vehicle control system design for the mars smart lander", *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2002, Doi: 10.2514/6.2002-4504

[29] Brugarolas, Paul B., San Martin, A. Miguel, Wong, Edward C., "Entry attitude controller for the mars science laboratory", *IEEE Aerospace Conference Proceedings*, 2007. Doi: 10.1109/AERO.2007.352824

[30] Darby, Christopher L., Hager, William W., Rao, Anil V., "An hp-adaptive pseudospectral method for solving optimal control problems", *Optimal Control Applications and Methods, Vol. 32, Issue 4*, 2011. Doi: 10.1002/oca.957

[31] Jorris, Timothy R., Schulz, Christopher S., Friedl, Franklin R., Rao, Anil V., "Constrained trajectory optimization using pseudospectral methods", *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2008. Doi: 10.2514/6.2008-6218

[32] Liu, Yue, Qian, Yingjing, Li, Jianqing, Gao, Changsheng, "Mars exploring trajectory optimization using Gauss pseudo-spectral method", *2012 IEEE International Conference on Mechatronics and Automation, ICMA 2012, pp 2371-2377*, Doi: 10.1109/ICMA.2012.6285716

[33] Grant, Michael J., Clarky, Ian G., Braunz, Robert D., "Rapid Design Space Exploration for Conceptual Design of Hypersonic Missions", *AIAA Atmospheric Flight Mechanics Conference 2011*, 2011. Doi: 10.2514/6.2011-6576

[34] Benson, David A. Huntington, Geoffrey T. Thorvaldsen, Tom P., Rao, Anil V., "Direct trajectory optimization and costate estimation via an orthogonal collocation method", *Journal of Guidance, Control, and Dynamics, Vol. 29, Issue 6*, 2009. Doi: 10.2514/1.20478

[35] Vlassenbroeck, J., and Van Doreen, R., "A Chebyshev Technique for Solving Nonlinear Optimal Control Problems," *IEEE Transactions on Automatic Control, Vol. 33, No. 4*, 1988, pp. 333–340

[36] M.Lavagna, C.Parigini, R. Armellin, "PLANETARY ATMOSPHERE ENTRY VEHICLES: MULTIOBJECTIVE OPTIMIZATION PSO ALGORITHM APPLIED TO A MULTI-BODY MULTIPLE FLIGHT REGIME MODELLING", *IAC 2006.*

[37] Jiang, Xiuqiang,Li, Shuang, "Mars atmospheric entry trajectory optimization via particle swarm optimization and Gauss pseudo-spectral method", *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, Vol. 230, issue 12*, 2016, Doi: 10.1177/0954410015622230

[38] Li, Shuang, Peng, Yuming, “Mars entry trajectory optimization using DOC and DCNLP”, *Advances in Space Research, Vol. 47, Issue 3,* Doi: 10.1016/j.asr.2010.09.005

[39] Beiner, L., Paris, S. W., “Direct trajectory optimization using nonlinear programming and collocation”, *Journal of Guidance, Control, and Dynamics, Vol. 10, Issue 4.* Doi: 10.2514/3.20223