Multi-Spacecraft Cooperative and Non-Cooperative Trajectory Optimization

by

Chandrakanth Venigalla

B.S., University of Virginia, 2016

M.S., University of Colorado Boulder, 2018

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Aerospace Engineering Sciences

2021

Committee Members:

Daniel J. Scheeres, Chair

Hanspeter Schaub

Marcus J. Holzinger

Zachary N. Sunberg

Xudong Chen

Venigalla, Chandrakanth (Ph.D., Aerospace Engineering Sciences)

Multi-Spacecraft Cooperative and Non-Cooperative Trajectory Optimization

Thesis directed by Prof. Daniel J. Scheeres

Multi-spacecraft systems can provide enhanced capability and robustness of space missions as compared to their single-spacecraft counterparts. This improvement, however, comes at the cost of a more complex mission design and optimization process. Spacecraft operators must also contend with space becoming increasingly congested and contested. Both cooperative and non-cooperative spacecraft interactions must be managed as an increasing number of spacecraft and spacecraft operators populate space. While these opportunities and needs exist for multi-spacecraft systems, trajectory optimization methods for them are relatively underdeveloped. Finding optimal trajectories can be critical in enabling enhanced performance of these missions.

In this dissertation, optimization theory is employed to develop methods that can simultaneously optimize the trajectories of multiple, dynamically connected spacecraft to analyze both cooperative and non-cooperative scenarios of interest. Several multi-spacecraft optimization methods are developed that cover varying applications as well as mathematical formulations. Applications explored in this work include: fuel-optimal multi-spacecraft rendezvous/deployment with unconstrained rendezvous/deployment orbits (e.g. constellation deployment), single and multiple spacecraft traveling salesman problems, cooperative and non-cooperative spacecraft collision avoidance, Pareto-optimal single spacecraft low-thrust interplanetary trajectories that are robust to missed thrust events (using a spacecraft swarm transcription), and fuel-optimal spacecraft pursuit-evasion games with terminal rendezvous. The solutions provided to these problems provide additional insight into each of these areas.

The mathematical techniques used in this work cover both single and multiple decision maker scenarios. If it can be reasonably assumed that a single decision maker decides the controls for all spacecraft, the multi-spacecraft optimization problem is formulated and solved as a mathematical programming (MPP) or optimal control problem (OCP). However, because multi-spacecraft problems are higher-dimensional than single-spacecraft optimization problems, solution methods that use MPP or OCP formulations must be carefully constructed to manage this additional complexity. When multiple decision makers must be accounted for, a differential game perspective is used to find optimal trajectories. This includes both zero-sum and general-sum games. The methods developed in this work provide additional tools to design improved spacecraft systems and trajectories.

Dedication

To amma and nanna

Acknowledgements

There are many more acknowledgements to make than can fit on this single page, but I'll make an attempt. First, I would like to thank my advisor Prof. Dan Scheeres for bringing me on as a student and guiding this work. His insights consistently guided me towards fundamentally understanding problems and asking the right questions. I'd also like to thank my committee members, Prof. Schaub, Prof. Holzinger, Prof. Sunberg, and Prof. Chen for their time and helpful perspectives on this work.

The NSTRF program has generously provided most of the funding for this work and has given me so many opportunities. Thanks to my NSTRF/JPL mentor Dr. Nathan Strange for introducing me to the mission design world, general guidance, and career advice. Thanks to my NASA GSFC mentor Dr. Jacob Englander for his general mentorship, our stimulating research collaboration, and the substantial time he spent answering my optimization questions. At JPL and GSFC I've also had the good fortune of meeting and learning from a number of excellent people who have been essential in helping me develop my optimization, astrodynamics, and general engineering skills.

My graduate student peers in CSML, CCAR, and the department have been a highlight of my time at CU. From welcoming me into grad school life, to general friendship and commiseration, conference travel, and much more, I think I've been lucky to find such a great group of people. Much of this dissertation was completed during the global COVID-19 pandemic; my gratitude to the healthcare and other essential workers for all of their efforts to keep things afloat.

Finally, thanks to my parents for all they've done to get me here, and thanks to them and my sister for their support through the years.

Contents

Chapter

Appendix

Tables

Table

Figures

Figure

Chapter 1

Introduction

There is a growing interest in spacecraft mission architectures that take advantage of two or more spacecraft. These missions, generally called “distributed space systems” [1] (DSS), include spacecraft swarms, formation flying, and constellation missions where multiple spacecraft are used in concert to accomplish a set of goals. Possible benefits to using distributed space systems include enabling increased mission robustness and providing capabilities that are not possible with a single spacecraft mission architecture. Satellite constellations are perhaps the most widely used distributed space system at present, and a current push for so-called “mega-constellations” is driving new growth in that area. Advances in SmallSat technology have also renewed interest in DSS. These technology advances have made the performance of SmallSats much more similar to more massive spacecraft than was previously possible. SmallSats can be produced at a significantly cheaper cost than more massive spacecraft, and the resulting cost savings can reduce the financial expenditure and risk required to test new and unproven DSS architectures.

Despite these advances, the widespread adoption of DSS outside of constellation missions has been relatively limited. One reason for this is that analysis and optimization of multi-spacecraft systems can be quite difficult. With the new possibilities that a DSS can provide comes additional degrees of freedom and a larger state space, both of which complicate trade space exploration and system design. These factors can make DSS more difficult to design and thus more expensive. While there has been a significant amount of work done in this area to advance our capabilities, many current approaches to formulating trajectories and control laws for multi-spacecraft systems use methods that cannot find an optimal result. Due to the immense cost per kilogram to build and launch spacecraft into Earth orbit or beyond, it can be essential to find (propellant) optimal trajectories that allow spacecraft to sufficiently extend their lifetime or to be sufficiently performant to warrant their costs. Optimality can also be especially important in SmallSats, where costs may be reduced but propellant mass and system margins might be very limited.

Thus, the primary goal of this work is to develop methods and algorithms that can give an understanding of what the optimal trajectories are for multi-spacecraft systems. These methods are primarily developed with ground-based, open-loop analysis with nonlinear dynamics in mind as opposed to being developed with on-board/online implementation in mind. This is useful for a number of reasons. First, for cases where offline solutions can be used for spacecraft control (or used as a tracking reference for a closed loop guidance law), the optimal solution computed on the ground is useful for both mission planning and for operational commanding of spacecraft. In cases where offline solutions coupled with a closed-loop tracking controller are not sufficient (e.g., there are significant uncertainties that cannot properly be accounted for), an optimal solution still gives an understanding of what the best-case system performance is. Knowledge of the optimal performance and behavior can inform the development of a more real-time applicable controller as well. If the best case performance still makes the mission unachievable given other mission constraints, alternative architectures can be sought. If the best case performance is achievable but the current best closed-loop control strategy has significantly worse performance, more work on the control strategy may then be warranted to make the mission more efficient. Whether or not the optimal solution can be operationally implemented, the development of better analysis tools may illuminate new DSS architectures that enable better or more efficient scientific data collection, surface coverage (e.g., communications, imaging), or other mission success metric. Finally, in a more abstract sense, understanding optimal trajectories for multi-spacecraft systems can give mission designers intuition into how these systems behave. Decades of work on single spacecraft trajectory optimization has given practitioners crucial intuition for solving those problems, even when solving problems adjacent to those where concrete optimality conditions have been formulated and solved. Such intuition for multi-spacecraft problems is relatively lacking, but can in part be improved through the type of optimal analysis developed in this dissertation.

Scharf et al. [2, 3] provide exhaustive overviews of spacecraft formation flying guidance and control work, though we note that in this work we do not limit ourselves to addressing only formation flying as defined in that work. Much existing work that does focus on finding multiple coupled spacecraft trajectories with consideration for optimality specifically focuses on multiple spacecraft that must achieve or maintain a certain geometric position relative to one another [2, 4, 5, 6]. This application is not strictly considered here. Other work focuses on developing control law approaches [3], which are of course necessary and crucially useful in formation flying, but are not the focus of this work which focuses more on optimality. We also contrast the optimal control approach here with a dynamical systems approach that is specifically focused on exploring and exploiting natural dynamics for single (e.g. halo orbits in the three body problem [7]) and multi-spacecraft applications (e.g. spacecraft bounded relative motion in complex dynamical environments [8]). The optimal control approach considered here is synergistic with dynamical systems approaches; both methods will illuminate different aspects of the underlying problems and in concert will enable the best solutions to be found.

The number of problems that fall under the umbrella of "multi-spacecraft trajectory optimization" is of course quite broad, and even minor changes to a single problem statement can significantly change the underlying mathematical tools used to formulate and solve the problem. This dissertation explores multiple different classes of problem in order to advance the state-of-the-art in areas that could support several different problem types. Consequently, the example scenarios found in this work are highly varied. While Section 4.1 addresses the single-spacecraft problem of robustly optimizing a low-thrust spacecraft trajectory, it is accomplished with a "virtual" multi-spacecraft method that can be extended to true multi-spacecraft problems. The perspectives, techniques, and mathematical tools summarized in Chapter 2 can be considered a (non-exhaustive) toolbox from which an analyst must judiciously select tools to apply to a given problem. To effectively formulate and solve optimal control problems the analyst must understand the strengths and weaknesses of each tool in relation to the scenario of interest.

Thesis Statement

Optimization theory is employed to develop methods that can simultaneously optimize the trajectories of multiple, dynamically connected spacecraft to analyze both cooperative and non-cooperative scenarios of interest. These methods provide analysts with greater insight and additional tools to design improved spacecraft systems and trajectories.

1.1 Organization

Chapter 2 provides a brief overview of relevant tools and techniques for solving optimization problems. This includes some discussion about different classes of optimization problems and the difficulties found in solving them, as well as different solution methods and ways to formulate the problems. This overview is by no means comprehensive, but does touch on major topics of interest in solving the types of problems explored in this dissertation.

Table 1.1: Mathematical approaches used in this dissertation for multi-spacecraft trajectory optimization

	Single decision maker	Multiple decision makers
Static	Mathematical programming	(Static) game theory
Dynamic	Optimal control theory	Dynamic (and/or differential) game theory

The main portion of this dissertation is contained in chapters 3-6, and each of these four chapters correspond to an entry in Table 1.1. The common theme in each chapter is the mathematical framework used to pose the problem, so similar applications are explored across different chapters based on the formulation. Table 1.1 is almost exactly excerpted from Table 1.1 in Basar et al. [9], and provides context into how multi-player optimization problems relate to single-player optimization problems. The notable change made here is that the term “decision maker” is used instead of the term “player.” This is to clarify that while we do explore problems where all control decisions are made by a single entity, such cases do include scenarios with multiple spacecraft trajectories that are fully controlled by a single entity.

Chapter 3 covers approaches and scenarios that fall into the category of mathematical programming. Here, mathematical programming refers to the optimization of an objective function that depends on static parameters, as opposed to optimal control which here will refer to the optimization of an objective functional which itself depends on function(s) as an input (e.g. optimize a functional which takes a control function of time as an input). In Section 3.1, a nonlinear programming problem (NLP) is formulated and solved to find total ΔV optimal cooperative rendezvous orbits for an arbitrary number of spacecraft. Each spacecraft has propulsive capabilities, and the ΔV optimal rendezvous orbit does not necessarily coincide with one of the initial orbits of the constituent spacecraft. The NLP is then to find the optimal semimajor axis (a), eccentricity (e), and inclination (i) of a rendezvous orbit that minimizes total ΔV for all spacecraft to transfer to that orbit.

Chapter 3 also explores integer linear programming problems related to multi-spacecraft systems in Section 3.2. Several variants of "traveling satellite problems" are explored, which are similar to the classic traveling salesman problem. In each of these traveling satellite problems the key question is: "in what order should a set of target orbits be visited such that the total ΔV path cost is minimized?" Such a problem is still one of optimizing static parameters, but the parameters are now integer valued and thus a different toolset must be used to solve these problems. Traveling satellite problems with both a single traveling spacecraft (one spacecraft visits all targets) and multiple traveling spacecraft (each spacecraft visits a subset of all targets) are addressed.

Finally, cooperative spacecraft collision avoidance scenarios are briefly addressed in Section 3.3. This analysis includes both linear and Keplerian dynamics, with impulsive maneuvers and two spacecraft. A method to generate the optimal trade-offs in total ΔV expended by each spacecraft (Pareto fronts) is developed, and sample Pareto fronts are shown for both types of dynamics. This type of analysis is important in understanding how to develop rules dictating how two controllable spacecraft should maneuver to avoid collision.

Chapter 4 subsequently explores multi-spacecraft optimal control problems with a single decision maker. In the optimal control approach to multi-spacecraft trajectory optimization, the states, constraints, and dynamics of each spacecraft are all combined into one problem and solved simultaneously. This is possible because a single entity dictates the control of all agents (i.e. all spacecraft are fully cooperative). While direct methods are used to solve problems in this section, the direct transcriptions used are approximating a solution to the infinite dimensional problem of finding a control function for all time that optimizes the objective under the given constraints. This is in contrast to problems in the previous chapter, in which decision variables are truly a finite set of scalars.

Section 4.1 develops a multi-spacecraft method of optimizing a low-thrust single spacecraft trajectory with consideration for robustness. A multi-spacecraft method has been found here to work well in controlling the missed thrust recovery margin (MTM) for a low-thrust spacecraft trajectory. The MTM problem is essentially that of ensuring that a low-thrust spacecraft can still reach its target even if it follows an off-nominal trajectory that is caused by an unexpected loss of thrust (missed thrust event) due to a spacecraft malfunction/safe mode event. The problem can be thought of an optimal control problem where robustness must be implemented as a path constraint, which we have effectively done by using a “virtual swarm” technique. In the virtual swarm method, the MTM of a nominal spacecraft is controlled by adding “virtual” spacecraft that represent recovery trajectories for missed thrust events that occur at different times in the nominal trajectory. The virtual spacecraft inherit the state of the nominal spacecraft at their spawn point, and have some amount of forced coasting time before they are allowed to thrust with controls independent of the nominal trajectory. The control profiles of the nominal spacecraft and each virtual/recovery trajectory are simultaneously optimized so that a given performance metric for the nominal spacecraft is optimized with the constraint that each recovery trajectory can still reach the final target within given constraints (e.g., range of permissible arrival dates at a planet). Objectives used for the virtual swarm are either the lower bound on the length of the initial shutdown time for each recovery trajectory (maximize the worst case MTM) or the lower bound on the delivered mass (fix MTM at specific points, minimize worst case propellant mass). Pareto optimal results considering robustness, time of flight, and delivered mass are also found using the virtual swarm method. The software tool developed in this work to optimize these virtual swarms, called the "N Spacecraft Trajectory Optimizer" (NSTOP), could more generally be used to optimize real spacecraft swarms as well.

Section 4.2 applies indirect methods to solving for finite time cooperative rendezvous and deployment optimal trajectories. The cooperative rendezvous problem is similar to that explored in Section 3.1, but now with consideration for maneuvering to match all orbit elements and with fewer restrictions on allowable controls. A mass-optimal rendezvous orbit is found where all spacecraft have propulsive capability, and each spacecraft can have any number of thrusting and coasting arcs. We also consider the dual problem of finding an optimal deployment orbit from which to launch several spacecraft to their final orbits. This is essentially the optimal rendezvous problem in reverse. The indirect formulation allows the potential for a lower-dimensional (fewer decision variables) representation of an optimal control problem as compared to the direct formulation in Section 4.1. However, the direct formulations can be significantly more robust to poor initial guesses, and thus enable easier convergence to a solution.

Chapter 5 begins our investigation of spacecraft differential games in this dissertation with a static game approach. When there are multiple agents with interdependent objectives ("payoffs"), dynamics, and/or constraints, the problem can be thought of as a static or dynamic game. Each agent seeks to optimize their own objective(s), but must consider the objectives and actions of other agents that would impact their own results. Static games as delineated here can be considered as those in which all possible decisions for all agents are used to find the payoffs for each agent in each scenario (see [9] or sections 9.1 and 9.2 in [10] for more detail). Frequently, two player static games with discrete decisions for each player are expressed as tables in which the payoff for each agent is compared for possible combination of decisions. The concept of a "Nash equilibrium" [11], where neither of two non-cooperative agents can unilaterally improve their outcome by altering their own strategy, is frequently expressed using these tables.

Chapter 5 analyzes a static version of a spacecraft pursuit-evasion game where a pursuing spacecraft attempts to rendezvous with an evading spacecraft that attempts to avoid rendezvous. This analysis takes a reachable set approach using ΔV optimal maneuvers to emphasize efficient maneuvers with coasting periods separating impulsive manuevers. In contrast, previous studies with actively maneuvering pursuing and evading spacecraft have solved time-optimal problems where time to intercept is the objective for each agent (the pursuer minimizes, the evader maximizes) without concern for propellant savings. We describe a method of calculating reachable sets of orbits for a pursuing and evading spacecraft, which can be used to determine whether or not it is possible for a pursuing spacecraft to capture an evading spacecraft. The static game is then relatively simple, and depends on the intersections of the two sets. However, we also explore the likely scenario where the evading spacecraft does not know the extent of the pursuing spacecraft's reachable set, and still must prudently select an evasion strategy.

Chapter 6 extends our analysis of game theoretic problems to dynamic games. The pursuit-evasion game that was explored in a more limited form in Chapter 5 is formulated in Section 6.1 as a special case of differential game called a "zero sum" game, in which the real valued objective of each agent is the negative of the other, such that the sum of their objectives is always zero. Again, the pursuit-evasion game is formulated with terminal rendezvous as the goal for the pursuing spacecraft, and the zero-sum objective is based on spacecraft fuel expenditures to enable efficient maneuvers with potential coasting periods. In this case we desire to find a Nash equilibrium solution, at which neither decision maker can improve their outcome by locally changing their strategy.

Section 6.2 explores a "general sum" game in which independently operated spacecraft both attempt to avoid an impending impact while minimizing their own control costs. Such a scenario is becoming increasingly common in space operations, especially more so-called "mega constellations" are launched into orbit. In this case we also desire to find a Nash equilibrium solution as in the zero-sum pursuit-evasion game. Such solutions could potentially be useful for independent operators to use when there is no governing entity (e.g. a space version of air traffic control) to resolve the potential conflict.

Finally, in Chapter 7 we present some concluding thoughts.

1.2 Contributions

The primary contributions of this work are:

- Developed methods for finding optimal cooperative rendezvous and deployment trajectories for time-free transfers as well as finite-time transfers
- Found bounding ΔV costs for orbital variant of the single and multiple traveling salesman problem
- Developed method for finding robust, Pareto-optimal low-thrust trajectories (e.g. find the minimum propellent margin required for a desired robustness to a missed thrust event)
- Studied time-free, ΔV-based spacecraft pursuit-evasion game with terminal rendezvous to understand risk posture and evasion strategies
- Formulated and numerically solved mass-optimal, continuous thrust pursuit-evasion differential game with terminal rendezvous
- Explored optimal spacecraft collision avoidance maneuvers in cooperative and non-cooperative contexts

1.3 Publications

1.3.1 Journal Papers

The following papers derived from this dissertation are currently in review or preparation:

- C. Venigalla, and D. J. Scheeres. "Cooperative and Non-Cooperative Approaches to Optimal Spacecraft Collision Avoidance" (In preparation)
- C. Venigalla, and D. J. Scheeres. "Mass-Optimal Spacecraft Pursuit-Evasion Games With Terminal Rendezvous" (In preparation)

- C. Venigalla, J. A. Englander, and D. J. Scheeres. "Multi-Objective Low-Thrust Trajectory Optimization With Robustness to Missed Thrust Events." Journal of Guidance, Control, and Dynamics. (In review)

The following peer-reviewed journal articles have already been published from this work:

- C. Venigalla and D. J. Scheeres, "Delta-V Based Analysis of Spacecraft Pursuit-Evasion Games." Journal of Guidance, Control, and Dynamics, Col. 44, No. 11, 2021, pp.1961-1971, `https://doi.org/10.2514/1.G005901`. [12]

- C. Venigalla and D. J. Scheeres, "Minimum Bounds on Multi-Spacecraft ΔV Optimal Cooperative Rendezvous." Journal of Guidance, Control, and Dynamics, Vol. 43, No. 12, 2020, pp. 2333–2348, `https://doi.org/10.2514/1.G004978`. [13]

1.3.2 Conference Papers

The following conference papers are associated with work in this dissertation:

- C. Venigalla and D. J. Scheeres, "Cooperative and Non-Cooperative Approaches to Optimal Spacecraft Collision Avoidance." 32nd AIAA/AAS Space Flight Mechanics Meeting. San Diego, California, January 3-7, 2022. [Accepted, not yet presented]

- C. Venigalla and D. J. Scheeres, "Optimal Multi-Spacecraft Cooperative Rendezvous and Constellation Deployment Trajectories." 2021 AAS/AIAA Astrodynamics Specialist Conference. Big Sky, Montana (virtual), August 9-11 2021. Paper AAS 21-718 [14]

- C. Venigalla, J. A. Englander, and D. J. Scheeres. Low-Thrust Trajectory Optimization for Maximum Missed Thrust Recovery Margin. 2020 AAS/AIAA Astrodynamics Specialist Conference. South Lake Tahoe, California (virtual), August 9-13 2020. AAS Paper 20-438. [15]

- C. Venigalla and D. J. Scheeres, "Lower Bounds on Delta-V Costs for Traveling Satellite

Problems." 10th International Workshop on Satellite Constellations and Formation Flying. Glasgow, Scotland, July 16-19 2019. Paper IWSCFF 19-33 [16]

- C. Venigalla and D. J. Scheeres, "3-Dimensional Reachable Set Applications to Multi-Spacecraft Trajectory Coordination." 29th AAS/AIAA Space Flight Mechanics Meeting. Maui, Hawaii, January 13-17 2019. Paper AAS 19-475 [17]

- C. Venigalla and D. J. Scheeres. "Numerical and Analytical Reachable Set Applications to Cooperative and Non-Cooperative Multi-Spacecraft Trajectory Coordination." 2018 International Astronautical Congress. Bremen, Germany, October 1-5 2018. Paper IAC-18,C1,5,2,x46279 [18]

- C. Venigalla and D. J. Scheeres, "Spacecraft Rendezvous and Pursuit/Evasion Analysis Using Reachable Sets." 2018 Space Flight Mechanics Meeting. Kissimmee, Florida, January 2018, `https://doi.org/10.2514/6.2018-0219`. Paper AIAA 2018-0219 [19]

Chapter 2

Optimization Problem Types and Methods

Broadly stated, the optimization problem is: how can a set of decision (control) variables be selected such that a performance index is optimized given some set of constraints. This statement encompasses a great deal of complexity that must be properly managed when using the tools of mathematics to pose and solve such problems. The "true" underlying problem of interest can be mathematically represented in many different ways, each of which has differing properties. This section gives some insight into the various techniques and perspectives on optimization that one might use when solving optimization problems. The topics covered only address a small subset of the broad field of optimization, but are targeted towards techniques found particularly useful for solving optimization problems related to those found in this dissertation. Additional detail and related citations can be found in survey papers such as [20] and [21].

2.1 Mathematical Programming

Suppose that we have a performance function $J : \mathbb{R}^n \to \mathbb{R}$ that depends on an n-dimensional finite set of decision variables $\boldsymbol{\rho} = \{\rho_1, \rho_2, ..., \rho_n\}$. Absent the mapping notation, functions may also be referred to as $f(\cdot)$ to more precisely refer to the function itself, rather than the result of the function evaluated at a certain value of its arguments, e.g. $f(t)$. In this dissertation the term "decision variable" tends to be used to describe values that can be changed in order to optimize the performance index; other terms such as "optimization variable" can also be used. There may also be some restrictions on the allowable decision variables that can be selected. These can be formulated as a function $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^m$ with $\boldsymbol{g}(\boldsymbol{\rho}) \leq 0$. Bold symbols are used to represent vector quantities or sets of values (e.g. $\boldsymbol{g}$ could be multiple functions of the set ρ).

Then, the mathematical programming problem (MPP), also called a *parameter optimization problem*, can be stated as:

$$\min_{\boldsymbol{\rho}} \quad J(\boldsymbol{\rho}) \tag{2.1}$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{\rho}) \leq 0 \tag{2.2}$$

where we usually construct the performance function for minimization. Performance functions constructed for maximization can simply be multiplied by -1 to be used for minimization. Thus, the terms *objective function* and *cost function* are also equivalent to *performance function*, though when discussing maximization problems the term *cost function* will not be used. While only inequality constraints are explicitly accounted for in Eq. (2.2), an equality constraint could be constructed by setting

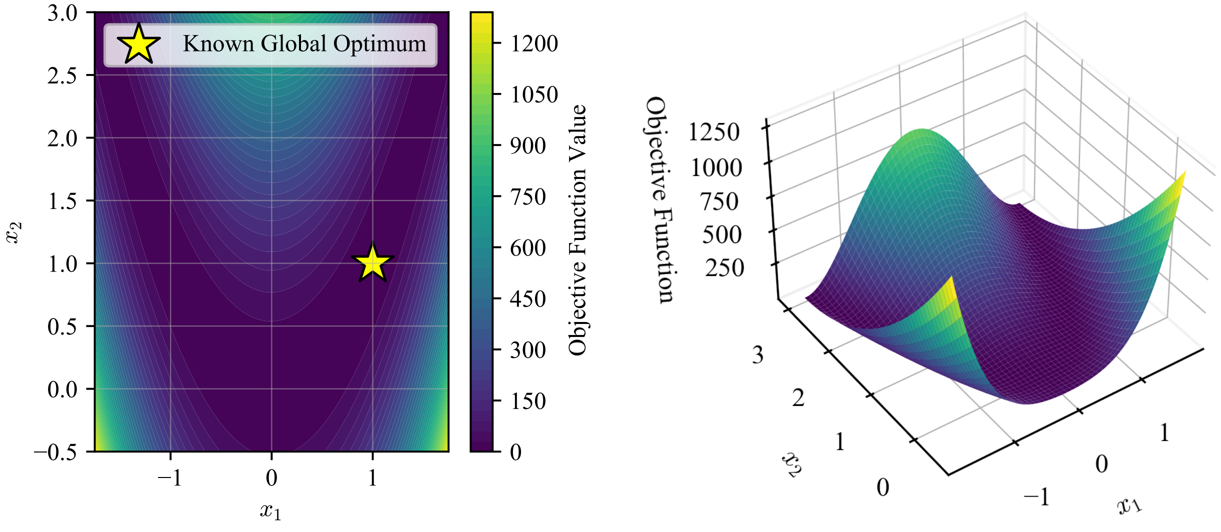
$$\boldsymbol{g}_i(\boldsymbol{\rho}) = \boldsymbol{f}(\boldsymbol{\rho}) \leq \boldsymbol{0} \tag{2.3}$$

$$\boldsymbol{g}_{i+1}(\boldsymbol{\rho}) = -\boldsymbol{f}(\boldsymbol{\rho}) \leq \boldsymbol{0} \tag{2.4}$$

Some references only explicitly include inequality constraints as in Eq. (2.2) given that equality constraints can be constructed in this manner, while others explicitly include a separate set of equality constraint functions. The set of decision variables $\boldsymbol{\rho}$ is called *feasible* if it satisfies Eq. (2.2), but a feasible solution may or may not be optimal (i.e. satisfy Eq. (2.1)).

The statement of the MPP is quite broad, and a number of of sub-classes of MPPs have been delineated and investigated. One class of MPP is *linear programming* (LP), in which the functions $J(\cdot)$ and $\boldsymbol{g}(\cdot)$ are linear. Algorithms to solve large-scale LPs (e.g., the simplex method) are readily available [22]. A number of resources treating LPs in much more detail are also available ([23, 24, 25, 22]). In Section 3.2 the “traveling satellite problems” make use of the *integer linear programming* (ILP) formulation. The ILP problem is a type of LP where decision variables must be integers. This can make finding a solution more difficult as compared to continuous LP problems, though techniques exist to manage this additional complexity ([22], see also Section 3.2). Another special class of MPPs are convex optimization problems where $J(\cdot)$ and $\boldsymbol{g}(\cdot)$ are convex. If the functions can be posed as convex, highly effective solution methods exist that can solve them [26]. The introduction of [26] also gives a useful, concise overview of various classes of optimization problems.

In this dissertation the majority of problems, however, are nonlinear and non-convex. Thus, corresponding MPPs that might be used very frequently have nonlinear, non-convex $J(\cdot)$ and/or $\boldsymbol{g}(\cdot)$, and are called *nonlinear programming* (NLP) problems. While NLP problems in this dissertation can have several hundred decision variables, it can be helpful to consider lower dimensional problems to understand challenges and pitfalls that might be encountered in attempting to find solutions.



(a) 2D view

(b) 3D view

Figure 2.1: Rosenbrock function

Consider the Rosenbrock function [27] of two decision variables x_1 and x_2:

$$f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2 \tag{2.5}$$

This function will be used as the objective function to minimize, where the minimizing values x_1^*

and x_2^* must be found as well as the optimal objective value $f(\boldsymbol{x}^*)$. For now, this is an unconstrained problem with no function(s) $\boldsymbol{g}$ to consider.

$$\min_{x_1,x_2} \quad (1-x_1)^2 + 100(x_2 - x_1^2)^2 \tag{2.6}$$

The objective function is low-dimensional and relatively simple to analyze; the function has been evaluated using a grid of $\boldsymbol{x}$ values and plotted in Fig. 2.1. This type of exhaustive search of the decision variable space is possible only because of the relative simplicity of this problem; in general, this is not always possible.

The "valley" corresponding to $x_2 = x_1^2$ is clearly a region of interest for finding the minimum value of the objective function, but the true minimum $f(1,1) = 0$ ($\boldsymbol{x}^* = [1,1]$) is harder to discern visually. Analysis of the function itself could lead an analyst to find the minimum, but this again is not generally possible with more complex functions that may be functions of many variables. Because the function is continuous and smooth, the gradient $\nabla f = \dfrac{\partial f}{\partial \boldsymbol{x}}$ can be used to find advantageous directions along which to search for a minimum from a starting point. One could imagine descending the surface in Fig. 2.1b as if a ball were released and gravity pulled it down towards the "lowest" point on the surface, a process analogous to using the gradient to select search directions for a minimum from a given starting point. When the decision variables are at the local minimum, we have $\nabla f = 0$, a necessary but not sufficient condition for a local minimum. The second order condition for a minimum, $\nabla^2 f = \dfrac{\partial^2 f}{\partial \boldsymbol{x}^2}$ is positive definite, is a sufficient condition for a local minimum if $\nabla f = 0$. Second order conditions can be quite difficult to find for many astrodynamics problems.

Suppose that a constraint is now added so that the NLP problem is:

$$\min_{x_1,x_2} \quad (1-x_1)^2 + 100(x_2 - x_1^2)^2 \tag{2.7}$$

$$\text{s.t.} \quad x_2 - \sin(5x_1) \le 0 \tag{2.8}$$

Now, the unconstrained minimum $\boldsymbol{x} = [1,1]$ is no longer a feasible point. In Fig. 2.2 the constraint line is shown with a dotted white line, with the region above the line being infeasible and the region

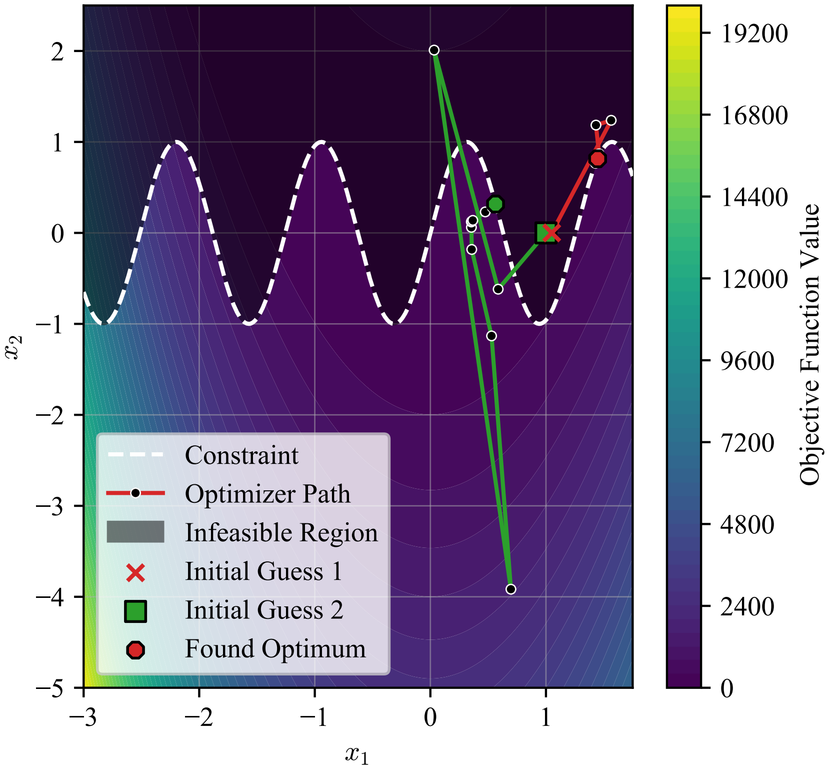


Figure 2.2: Constrained Rosenbrock optimization problem; initial guess 1 $= [1.05, 0]$ with found local optimum $J = 163.9$, initial guess 2 $= [1, 1]$ with found local optimum $J = 0.1903$

below being feasible. Also shown are two different paths a gradient-based optimizer (IPOPT [28]) took in arriving at final local minima shown with a hexagon given two different initial guesses shown with a square and an x. With just a difference of 0.05 in the x_1 initial guess, the local optimum found is significantly better starting from initial guess 2 (0.1903) as compared to the result from initial guess 1 (163.9).

The addition of the simple constraint in Eq. (2.8) has now increased the difficulty in finding a solution. While only one local minimum existed in the unconstrained problem, now multiple local minima exist. Further, the optimality criteria is now more complex; local minima can be found where $\nabla f \neq 0$. While understanding this complexity is relatively simple for this two-dimensional decision vector, the problem can quickly become very difficult to understand in this manner as the dimensionality increases and the objective function becomes more complex.

Of course, the underlying function in general may also have multiple local optima, and potentially even multiple global optima as well. Thus, in conjunction with finding local optima, the existence of other optima must also be considered. To truly solve the optimization problem, the globally minimizing $\boldsymbol{x}$ should be found, but in many cases for non-convex problems it simply cannot be known whether or not a global minimum has been found. Frequently, global search methods are coupled with local, gradient-based optimization methods to give fast local convergence as well as some ability to approximately find a global optimum. These global methods frequently have stopping conditions that are based on run time (wall time), number of iterations, or time spent searching for a solution without any improvement.

There are a multitude of available methods for solving NLP problems, including many that can handle constraints. Common gradient-based solution methods for NLP problems include sequential quadratic programming (SQP) and interior-point (IP) methods; Nocedal and Wright present a detailed discussion of numeric methods for optimization [29]. In this dissertation, the SQP-based tool SNOPT [30] and the interior-point tool IPOPT [28] are both used to solve NLP problems. Gradient based methods have the benefit of efficiently converging on local minima where the objective function is continuous and smooth. Non-gradient based methods such as genetic algorithms and particle swarm optimization can be useful for non-smooth problems and/or for finding global minima.

2.2 Optimal Control

The major difference in optimal control as compared to mathematical programming is that the decision variables are no longer a finite set. An optimal control problem (OCP) generically has an objective functional that is itself dependent on a function, e.g. $J(\boldsymbol{u}(\cdot))$ where a control function $\boldsymbol{u}(\cdot)$ must be selected to optimize the objective. A classic optimal control problem is the brachistochrone problem, in which the shape of a curve must be optimized such that a ball moves from point to point in minimum time under the influence of gravity. This is fundamentally different from mathematical programming because the shape of the curve is best described by a function, as opposed to a finite set of scalars. The optimal control problem is explored in a wide variety of texts and papers, including [10, 31, 32].

In this dissertation we use both indirect and direct methods to solve optimal control problems. Indirect methods leverage the calculus of variations to derive necessary conditions of optimality. These conditions are based on ensuring that the first variation of the objective function is zero (i.e. stationary) while simultaneously satisfying terminal conditions (e.g. constraints on initial and final states). While second order conditions might also be derived to ensure that the stationary point is a minimum/maximum as desired, in astrodynamics problems second order conditions are frequently not addressed due to significant difficulty in deriving them. In the indirect problem, if the state vector $x \in \mathbb{R}^n$, then n additional costates (also called adjoints) are added to the problem. The dynamics and terminal conditions of these costates are found through applying the calculus of variations, and the optimal control profile is a function of both the states and costates. Solving the OCP with indirect methods then becomes that of solving the two-point boundary value problem (TPBVP) formed from the problem statement and necessary conditions for optimality. Typically, this entails finding the initial costates that satisfies the boundary conditions given the dynamics of the states and costates. This is a non-trivial problem, as the costates in general do not have a physical interpretation[1] and can be difficult to guess. Further, the state and costate dynamics can be highly sensitive to small changes in initial costate guesses, making procedures such as differential correction difficult.

Due to the need to analytically derive the necessary conditions and the numerical sensitivity of the indirect problem, many analysts prefer to use direct methods to solve optimal control problems. With a direct method, the infinite-dimensional control function is instead approximated with a finite set of decision variables. Methods of performing this transcription vary widely, and can significantly impact the ease of finding the solution as well as the fidelity of the transcribed problem. Methods for doing this include a zero-order-hold type method where decision variables dictate how control is applied during a finite number of control segments and control law methods where decision variables

[1] “adjoint control transformations” have been found to help relate physical control to values of costates

are parameters that dictate how the control law acts at each time. Once the infinite dimensional control function is made finite dimensional, the problem is then a MPP and can be solved using associated methods. Direct transcriptions especially gained popularity as computational power increased and NLP solvers that can handle many decision variables and constraints were developed. A downside of direct methods is that they can require very large numbers of decision variables, while an indirect formulation can provide a relatively compact way to represent an optimal solution with fewer decision variables to find.

2.3 Numeric Description of State Dynamics and Control

A number of methods exist to represent state dynamics when numerically solving optimization problems. In this work, single and multiple shooting are primarily used for that purpose. In single shooting, state dynamics are propagated from the initial time all the way to the final time. It is frequently likened to the iterative process of aiming a cannon by shooting a cannonball towards a final target, finding the error between the actual and desired position of the cannonball, and adjusting the aim of the cannon and repeating the process until the target is hit. More generally, there may be decision variables associated with the initial time as well as later times (e.g. controls vectors). Single shooting can be a useful, low-dimensional method to represent the state dynamics. However, in optimization we frequently use partial derivatives of objectives and constraints with respect to decision variables in order to adjust the decisions to find a feasible and optimal solution. Given the potential large nonlinearities found with single shooting, the linear approximation provided by the partial derivatives may not be practically useful, and convergence on a solution may be difficult or impossible.

Multiple shooting addresses this issue by breaking the trajectory into multiple different segments each with a shorter propagation time than the full time t_0 to t_f, and in each segment there are decision variables for what the state should be at that point (typically at the initial or final time of the segment). This requires the addition of more decision variables, and additionally more constraints to ensure that the state across each boundary between segments is continuous. This additional complexity, however, can reduce the nonlinearity of relationships between decision variables and objectives & constraints. This can be critical to being able to numerically find a solution if a good initial guess is not available and single shooting proves to be too nonlinear. A more detailed discussion of single and multiple shooting is found in Section 4.2.3, though that section is not the only portion of this dissertation using those concepts.

There are also a variety of ways to model how the spacecraft might be controlled in an optimization problem. A classic method of modeling spacecraft control is to assume that control provides instantaneous, unbounded changes to the spacecraft's velocity vector at a given time. This approximates the operation of a chemical thruster fairly well, and solutions with these impulsive controls can be used as starting points to then find exact firing time and direction for the true, finite-time burns. Low-thrust trajectories are poorly modeled by using a small number of unbounded, impulsive maneuvers. This is because when attempting to transition to finite time maneuvers, the magnitude of the thrust is not large enough to accomplish the velocity change in a short time. As a result, an optimal low-thrust trajectory can be quite different from an optimal impulsive trajectory. Instead, in the Sims-Flanagan transcription [33] (see also Section 4.1), a sequence of impulsive maneuvers are equally placed throughout the trajectory and the magnitude of each is constrained by how much ΔV a low-thrust propulsion system could provide prior to the next impulse.

A higher fidelity method, potentially using a solution to one of the bounded or unbounded impulsive maneuver problems as an initial guess, would actually integrate the equations of motion of the spacecraft with the amount of acceleration the propulsion system can provide. For certain problems, this method can make evaluating changes to desired objectives & constraints relatively slow to calculate, and thus can make it difficult to search for solutions without a good initial guess. Finally, the control profile of the spacecraft can be parameterized by a control law. With a control law approach, there is some function relating the state to what the control will be (e.g., a closed loop control law that selects the thrusting direction and magnitude based on the current state and time). This approach restricts the permissible control that can be performed by the spacecraft, and when included in an optimization problem might prevent the analyst from finding a control profile that is feasible and results in better performance than can be found with control parameterized by a control law. However, especially for highly complex problems (e.g. many revolution transfers), the use of a control law can make finding a solution much easier, and might not result in much of a decrease in performance as compared to an unrestricted control. A control law solution might also provide an initial guess for an optimization problem where the control law relation is removed and any feasible control can be selected. A popular control law option for low-thrust trajectories is the Q-law [34].

2.4 Game Theory

Game theory is a deeply complex research area that provides tools to analyze how multiple decision makers might optimally make decisions when other agents' decisions can impact their own objectives. In a game, each player generally has their own objectives that are in some way influenced by the decisions made by other players as well as each player's own decisions. The problem is then to select an "optimal" set of decisions based on what decisions other agents might themselves decide to make. Given the coupling between all agents, each agent must consider not only its own decisions but also consider what decisions other agents might make. While one might consider finding Pareto-optimal solutions across each of the agents different objective functions, in general a game has no requirement that the players cooperate with one another. For all agents to select the same Pareto-optimal solution there must be some amount of cooperation to do so. Instead, in this dissertation we focus on Nash-equilibrium solutions as our definition of optimal solutions to games. In a Nash-equilibrium solution, no agent has incentive to unilaterally change their own control because such a decision would result in a worse outcome. This type of stable solution can be a way for multiple decision makers to agree on a solution when they do not have a great deal of trust that other agents will cooperate. Multiple other definitions of "optimal" exist as well [35, 36], each resulting in different features of the solution. Also not explored in this dissertation is the impact of information structure on solutions to games. Different information structure (i.e. what

Table 2.1: Prisoner's Dilemma Game

		Player 2	
		Action x	Action y
Player 1	Action a	2 / 2	10 / 1
	Action b	1 / 10	5 / 5

information is available to each agent and when it is available) can significantly change optimal strategies for a given problem.

A sample game in matrix form is shown in Table 2.1 [35]. While in this dissertation we do not cover any games in matrix form, considering matrix games can be useful in generally understanding game theory. Table 2.1 covers the classic prisoner's dilemma, in which the two players are deciding whether or not to provide evidence of crimes committed by their partner. Actions b and y correspond to each player providing evidence against the other; if both do so, both receive 5 years in prison. This is the only equilibrium solution, because if both plan to provide evidence against the other, neither player can unilaterally decrease their prison sentence by changing their strategy alone. The solution (a, x) results in less prison time than (b, y), but at (a,x) a single player could decrease their prison sentence to 1 year by changing their mind and providing evidence against the other player. That comes at the expense of making the prison sentence of the other player 10 years. For a zero-sum game, the costs in each cell would always sum to zero.

In this dissertation, we explore zero-sum and general-sum games. Zero-sum games require that all players objective functions always sum to zero. This special property enables certain solution methods to be used. The zero-sum game explored here is the spacecraft pursuit-evasion game; pursuit evasion games were fundamental in the development of differential game theory [37]. Useful discourse on the meaning of solving differential games can also be found in [38]. We also explore a general-sum game where spacecraft objectives are not the opposite of one another (Section 6.2 collision avoidance). General-sum games present additional challenges as compared to zero-sum games, but techniques to find approximate solutions have been developed and are applied here.

Chapter 3

Mathematical Programming Approaches

This chapter covers a variety of different problems; a summary of the problems is presented in Table 3.1. While single solution methods are presented in the table, other methods are potentially viable (see corresponding sections for details).

Table 3.1: Summary of MPPs in Chapter 3

Problem	**Dynamics & Control**	**Formulation**	**Solution Method**	**Notes**
Rendezvous & Deployment (Section 3.1)	Keplerian, time-free control (two-impulse)	NLP	Grid search & NLP Solver (IPOPT)	Transfers realizable in finite time[1]
Depot Placement Problem (Section 3.2.4.2)	Keplerian, time-free control (two-impulse)	NLP	Particle Swarm Optimization	
Single Traveling Satellite Problem (Section 3.2.3)	Keplerian, time-free control (two-impulse)	ILP	Concorde	Conservative, bounding costs
Multiple Servicer Problem (Section 3.2.4)	Keplerian, time-free control (two-impulse)	ILP	ILP Solver (Gurobi)	
Cooperative Collision Avoidance (Section 3.3)	Linear & Keplerian, three-impulse	NLP	NLP Solver (IPOPT)	Provides Pareto-optimal solutions

[1] For the depot placement problem, repeat transfers to the depot and back are realizable in finite time if there is no J_2 perturbation. With J_2 perturbations, differential RAAN drift could potentially make finite time realizations trivially different or significantly different depending on the spacecraft positions. If each spacecraft only makes a single visit to its nearest depot, finite time transfers can be realized using the same method in Section 3.1

3.1 Nonlinear Programming: Minimum Bounds on Multi-Spacecraft Delta-V Optimal Cooperative Rendezvous

3.1.1 Introduction

There is a growing interest in spacecraft mission architectures that take advantage of two or more spacecraft. These missions include spacecraft swarms, formation flying, and constellation missions where multiple spacecraft have propulsive abilities. These types of missions allow for increased robustness and can provide capabilities that are not possible with single spacecraft mission architectures. However, tools available to mission planners for large numbers of spacecraft are limited, making mission design for these systems more complicated. It is difficult to understand and take advantage of the full capabilities of multi-spacecraft systems without the development of new optimization tools. Decades of work on single spacecraft trajectory design and analysis has given practitioners crucial intuition for solving those types of problems. Such intuition for multi-spacecraft problems is relatively lacking. While there has certainly been a significant amount of work completed to explore questions of how to best use and control systems of multiple spacecraft, there is much more to learn about how these systems fundamentally behave. A great deal of related past work can be found in the comprehensive survey papers of Scharf, Hadaegh, and Ploen [39, 40] that briefly describe and cite previous work done on spacecraft formation flying guidance and control. Another slightly more recent work is the textbook by Alfriend et. al [1], which aims to describe in some detail the various aspects of Earth-orbiting formation flying (it does not include discussions of deep-space or libration-point formation flying). In the interest of furthering the community's fundamental understanding of multi-spacecraft systems, we investigate the application of time- and orientation-free spacecraft orbit transfers to find lower bounds on the cost for ΔV optimal rendezvous orbits in multi-spacecraft systems.

Orbital time- and orientation-free transfers are those in which the transfer time is not specified or restricted, and the final argument of periapsis is also unspecified [41]. Perhaps the most familiar time- and orientation-free transfer in astrodynamics is the Hohmann transfer [42], the ΔV optimal two-impulse transfer between circular orbits. Many early investigations into orbit transfers were similarly based on time- and orientation-free transfers as well (see Gobetz [41] for a survey of early work). This early work on elementary optimal orbit transfer theory set bounds on the best case performance realizable in a number of different scenarios, and set the foundation for future work on the optimization of orbit transfers for more constrained problems. More recent work by Holzinger et. al. [43] has also used time- and orientation-free transfers. His work specifically focused on applying time-free ΔV optimal orbit transfers in semimajor axis (a), eccentricity (e), and inclination (i) space without specifying final orbit argument of periapsis (ω), right ascension of the ascending node (RAAN, Ω), or mean anomaly (M). Those constraints and optimal control theory were used to determine reachable sets of orbits for different amounts of available ΔV for a spacecraft at a given orbit. Reachable sets were found using level set methods to solve the Hamilton-Jacobi-Bellman partial differential equation.

Spacecraft rendezvous has been explored in a wide variety of scenarios. However, the specific case of cooperative rendezvous with two or more active spacecraft is not explored as frequently in the literature. Spacecraft rendezvous is usually treated in cases of only two spacecraft, most often with an inert or non-maneuvering target (e.g. space station, target orbit). Here, we generally define “cooperative rendezvous” to be rendezvous where both spacecraft actively maneuver, and the rendezvous orbit is not necessarily constrained to be the initial orbit of one of the constituent spacecraft. Two-spacecraft, fuel-optimal cooperative rendezvous has been explored with an optimal control approach by Prussing for finite-thrust rendezvous to match unconstrained final position and velocity [44] as well as for an impulsive terminal maneuver to match velocities of two spacecraft with already matching positions [45, 46]. Coverstone and Prussing have also investigated power-limited rendezvous between circular orbits using linearized [47] and nonlinear [48] two-body dynamics, also with an optimal control approach. Dutta and Tsiotris explored analytic solutions to planar cooperative rendezvous scenarios given fixed-time transfers, circular initial orbits, and circular rendezvous orbits [49]. Bevilacqua and Romano explored the use of differential drag with a control law for cooperative rendezvous of multiple spacecraft ($N > 2$) at a target spacecraft using linearized dynamics [50]. Thakur et. al. [51] developed a decentralized, stable control scheme that is used to bring a swarm of low-thrust spacecraft ($N > 2$) into the same orbit at different anomaly angles. A decentralized scheme is used to allow each spacecraft to decide its own control while only knowing the states of neighboring spacecraft. In that work a final orbit semimajor axis and eccentricity are prescribed, while the final orbit plane is decided on by a consensus protocol that is not optimizing factors such as time or fuel.

This section considers the question of where the ΔV optimal meeting point is for an arbitrary number of actively maneuvering spacecraft. The rendezvous orbit is unconstrained in order to enable the freedom to choose a total ΔV optimal rendezvous point or an orbit that optimizes some other ΔV based criteria. If ΔV savings can be realized by careful selection of an unconstrained rendezvous orbit, this may enable mission designers to more efficiently design a multi-spacecraft system and could perhaps enable new architectures, especially when a design is ΔV limited. Answering this question is also key to trajectory design for scenarios where there is no obvious rendezvous point such as the orbit of a space station. In scenarios without an obvious or highly constrained rendezvous orbit, the location of the final rendezvous orbit itself may be less important than minimizing the cost of rendezvous. This is especially true when the total available ΔV is limited, such as in small satellite missions. Examples of such scenarios include on-orbit construction and on-orbit placement of supply depots (e.g. placing a supply depot or servicing spacecraft at the ΔV optimal rendezvous location). The question considered here is a similar problem to the scenario addressed by Thakur et. al. [51], but we instead focus on finding the semimajor axis, eccentricity, and inclination of ΔV optimal rendezvous orbits in a centralized scenario where a central actor knows all spacecraft states and can direct the actions of all spacecraft.

In Sec. 3.1.2 we first discuss the assumptions made, and then describe how to calculate ΔV optimal orbit transfer costs under those assumptions. Notably, the orbit transfer costs found with the method of Sec. 3.1.2 only account for orbit transfers in a, e, and i space without regard for costs to achieve a certain ω, Ω, and M. However, desired ω, Ω, and M values can be achieved at no ΔV cost given infinite time and secular perturbations due to J_2. Using that method, in

Sec. 3.1.3 we formulate the cooperative ΔV optimal rendezvous orbit problem as a nonlinear programming problem (NLP) and discuss methods of solving it. The resulting solution for ΔV optimal rendezvous presents the optimal cost for full rendezvous (all six orbit elements match for all spacecraft attempting rendezvous) if an infinite amount of time is available to complete all transfers. Orbit elements a, e, and i of the rendezvous orbit are found with the solution method developed here, while the other orbit elements do not impact the time-free cost. Unless otherwise specified, the term “rendezvous” in this section will refer to this time-free rendezvous where ΔV is only expended to make each spacecraft have a matching set of a, e, and i, and other orbit elements are allowed to naturally match with no additional ΔV cost by using secular J_2 perturbations. The time-free ΔV cost found for all spacecraft to meet at the optimal rendezvous orbit is a lower bound for the ΔV cost for finite-time full rendezvous where the total transfer time cannot be infinite and for situations where there is no J_2 perturbation to cause propellant free changes in ω, Ω, and M. Section 3.1.4 explores planar scenarios (a, e) and three-dimensional scenarios (a, e, i) within this framework of finding bounding time-free ΔV costs for cooperative rendezvous. In both scenarios, instances with $N = 2$ spacecraft as well as $N > 2$ spacecraft are addressed.

In Sec. 3.1.5.1, we use an example to demonstrate that the optimal time-free ΔV cost for rendezvous found with the method of Sec. 3.1.3 can also be equal to the ΔV cost of finite-time full rendezvous if certain initial conditions are able to be selected. The same maneuvers found in the time-free case can also be used to achieve the time-free ΔV cost while achieving full rendezvous in finite time. Finally, because spacecraft deployment is similar to spacecraft rendezvous in reverse in this work, Sec. 3.1.5.2 includes a discussion of applying the ΔV optimal rendezvous solution to constellation deployment scenarios. The ΔV optimal rendezvous orbit for a set of spacecraft already in their final orbits represents the initial or deployment orbit that has the lowest total ΔV cost for all spacecraft to transfer to their final orbits.

3.1.2 Time- and Orientation-Free Optimal Transfers in Three-Dimensions

The original motivation of this work was to apply the reachable set methods of Holzinger [43] to multi-spacecraft scenarios. While doing so is possible [19], we found that the level set methods used to calculate the reachable sets were in practice cumbersome to work with; calculating sets at high accuracies took a considerable amount of computation time and only gave results at discrete locations. For multi-spacecraft scenarios, such limitations make analysis especially difficult because the points of interest are those where the reachable sets of different spacecraft intersect. In this work we instead focus on using known optimal transfer sequences between orbits to address multi-spacecraft scenarios. Calculating the optimal ΔV from the fixed initial condition to a final orbit is equivalent to solving for a single point on a full reachable set surface. This has the advantage of allowing the exact, analytic calculation of transfer ΔV costs between orbits in a much faster manner than solving for a full reachable set of orbits.

In this work we make similar assumptions to those made in the work of Holzinger [43]. Maneuvers are assumed to be impulsive, there is no constraint on transfer time, and the final angles ω, Ω, and M are not specified for the final orbit when transferring from one orbit to another. For planar transfers, the final orbit inclination is also unspecified. Most of this work assumes Keplerian dynamics, while Keplerian dynamics with secular effects from J_2 are used in Sec. 3.1.5.1. Note that there is no secular effect from J_2 on a, e, and i. Orbit transfers here go from an initial orbit $[a_i \quad e_i \quad i_0]$ to a final orbit $[a_f \quad e_f \quad i_f]$. For planar results, orbit transfers go from an initial orbit $[a_i \quad e_i]$ to a final orbit $[a_f \quad e_f]$. Throughout this work, the planar orbit shape $[a \quad e]$ will be treated as equivalent to an orbit parameterization $[q \quad Q]$ with periapsis radius $q = a(1-e)$ and apoapsis radius $Q = a(1+e)$. ΔV optimal transfers in the three-dimensional and planar cases will form the basis of the rendezvous analyses.

While the time- and orientation-free transfers used here do not specifically target ω, Ω, and M, perturbations from the J_2 spherical harmonic term will cause a secular drift in these parameters (the two-body drift rate in mean anomaly is modified by the J_2 perturbation). Thus, given a gravity field with a J_2 perturbation and no restriction on transfer time, any desired ω, Ω, and M can be achieved at a single instant in time at no ΔV cost by waiting until the secular perturbations cause the orbit to precess into the proper orientation and anomaly angle. Note that for this to be true, no two angular rates ($\frac{d\bar{\omega}}{dt}$, $\frac{d\bar{\Omega}}{dt}$, $\frac{d\bar{M}}{dt}$; the bar denotes orbit-averaged quantities) can be equal to one another or commensurate with one another. This is usually the case in Earth orbit, but the amount of time necessary to reach the desired values of ω, Ω, and M can potentially be infinite or impractically large for a given set of initial conditions and desired final conditions. If the initial values of ω, Ω, and M can be selected, any final values of those orbit elements can be achieved at no cost by selecting initial values such that they drift into the final desired values over the known transfer time (see Sec. 3.1.5.1 for details). Holzinger [43] makes a similar case about zero ΔV cost changes to ω and Ω when using optimal control theory to generate reachable sets of orbits in $a - e - i$ space. Without considering J_2 effects, the results will still give a lower-bound on the ΔV cost for a transfer that does have additional constraints on ω, Ω, and/or M.

We also limit the number of impulses to two, in part to simplify the analysis. While it has been shown that for the orientation-free ellipse-to-ellipse transfer (equivalent to the case where both the initial and final ellipse have the same argument of periapsis) there are three-impulse transfers through infinity that can cost less ΔV than two-impulse transfers in some cases [41, 52], such transfers are quite extreme. Transfers through infinity use a parabolic trajectory to reach a distance infinitely far from the central body, at which point one or more maneuvers are performed (e.g. a bi-parabolic transfer). Reachable sets generated with the two-impulse transfers used here look qualitatively similar to those generated by Holzinger, though quantitative comparisons are difficult due to the accuracies of the non-analytically generated reachable sets. We also show in Sec. 3.1.4.1 that we find the same ΔV optimal rendezvous orbits as in the case where the non-analytically generated reachable sets are used.

3.1.2.1 Planar Orbit Transfers

The planar transfer case is that of finding an optimal transfer between two coplanar, elliptical orbits. The ΔV optimal transfer sequence for this scenario is well understood and has been proven in a number of different ways; the work of Mease and Rao [53] is one such explanation of the optimal maneuvers. To summarize, the optimal transfer pathways are along lines of constant periapsis and apoapsis, so throughout this work orbits are parameterized with radius of periapsis (q) and radius of apoapsis (Q).

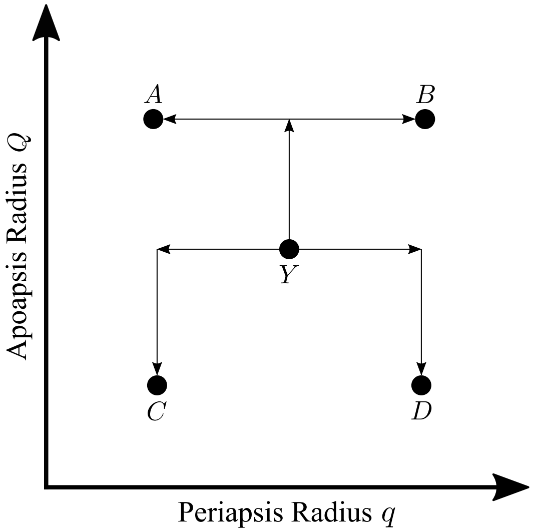


Figure 3.1: Optimal Transfer Pathways

The optimal transfer sequence from an initial orbit Y to several possible final orbits is shown in Fig. 3.1. For a given initial orbit, the optimal transfer sequence is dictated by the target orbit's apoapsis radius. If the target orbit has a larger apoapsis radius than the initial orbit, then the optimal transfer sequence is to first change the apoapsis radius to match the target apoapsis, and then change the periapsis radius to match the target. This is equivalent to following the path from orbit Y to orbit A or B in Fig. 3.1. If the target orbit has a smaller apoapsis radius, then the optimal transfer sequence is to first change the the periapsis radius to match the target periapsis, and then change the apoapsis radius to match the target. This is equivalent to following the path from orbit Y to orbit C or D in Fig. 3.1.

For two-impulse transfers, these optimal sequences are accomplished with a combination of two tangential maneuvers at periapsis and apoapsis. For example, to move from orbit Y to orbit A, the spacecraft will first apply a ΔV tangentially at periapsis to raise apoapsis to match A, then it will apply a ΔV tangentially at apoapsis to lower periapsis to match A. To achieve the optimal ΔV cost, however, the periapsis and apoapsis maneuvers need not occur all in one impulse. That is, the spacecraft could split the apoapsis raise maneuver into multiple maneuvers happening at successive passes of periapsis, and then after finally achieving the desired final apoapsis radius it could split the periapsis changing maneuver into multiple impulses all occurring at different passes of apoapsis. Splitting a periapsis or apoapsis change into multiple impulses will cost the same amount of ΔV as a single maneuver because each sub-maneuver can be linearly combined to get the same cost. Importantly, all apoapsis change maneuvers must be completed before changing periapsis when the target has a larger apoapsis, or vice versa when the target has a smaller apoapsis. These maneuvers are also consistent with the allowed ΔV optimal maneuvers in Holzinger's work on generating reachable sets [43].

Note the symmetry present here; the optimal transfer sequence to move to a target orbit is simply reversed if moving from the target orbit to the initial orbit. Degenerate cases where the target orbit has the same radius of periapsis or apoapsis as the initial orbit simply result in one of the two impulsive maneuvers having a magnitude of 0.

Given that the optimal transfer pathways are known, an expression for the optimal ΔV from some initial orbit to any final orbit is

$$\Delta V'_{total} = \Delta V'_1 + \Delta V'_2 \tag{3.1}$$

where

$$\Delta V'_1 = |v_{1t} - v_0| \tag{3.2}$$

$$\Delta V'_2 = |v_f - v_{2t}| \tag{3.3}$$

and subscript "t" is for the transfer orbit. v_0 is the velocity on the initial orbit immediately prior to the first impulse, and v_f is the velocity on the final orbit immediately after the second impulse. The optimal two-impulse transfer sequence and ΔV cost relies on the values of the initial and final radius of apoapsis Q_0 and Q_f, where q_0 and q_f are the initial and final periapsis radii. For $Q_f > Q_0$,

$$v_0 = \sqrt{\frac{2\mu}{q_0 + Q_0}\frac{Q_0}{q_0}} \tag{3.4}$$

$$v_{1t} = \sqrt{\frac{2\mu}{q_0 + Q_f}\frac{Q_f}{q_0}} \tag{3.5}$$

$$v_{2t} = \sqrt{\frac{2\mu}{q_0 + Q_f}\frac{q_0}{Q_f}} \tag{3.6}$$

$$v_f = \sqrt{\frac{2\mu}{q_f + Q_f}\frac{q_f}{Q_f}} \tag{3.7}$$

For $Q_f \leq Q_0$,

$$v_0 = \sqrt{\frac{2\mu}{q_0 + Q_0}\frac{q_0}{Q_0}} \tag{3.8}$$

$$v_{1t} = \sqrt{\frac{2\mu}{q_f + Q_0}\frac{q_f}{Q_0}} \tag{3.9}$$

$$v_{2t} = \sqrt{\frac{2\mu}{q_f + Q_0}\frac{Q_0}{q_f}} \tag{3.10}$$

$$v_f = \sqrt{\frac{2\mu}{q_f + Q_f}\frac{Q_f}{q_f}} \tag{3.11}$$

Equations (3.2–3.3) can be rewritten without absolute value operators with knowledge of the initial and final states. This is easier to see if Eqs. (3.4–3.11) are rewritten in a different form. For example, Eq. (3.4) can be rewritten as

$$v_0 = \sqrt{2\mu\left(\frac{1}{q_0} - \frac{1}{q_0 + Q_0}\right)}, \tag{3.12}$$

which enables clearer magnitude comparisons with other velocities. One important note is that the optimal ΔV cost will not have continuous derivatives as the final orbit parameters q_f and Q_f are changed. Discontinuities in partial derivatives of Eqs. (3.2–3.3) with respect to the final orbit ($\partial/\partial q_f$, $\partial/\partial Q_f$) arise for two reasons. First, because the optimal transfer sequence changes depending on whether Q_f is greater than or less than Q_0, the derivatives will be discontinuous across the line where $Q_0 = Q_f$. Second, the derivatives of Eqs. (3.2–3.3) are discontinuous because of the absolute value operator. Even if the expressions are re-written to be positive based on known values of q_0, Q_0, q_f, and Q_f, the derivative is only continuous for limited regions where the difference in certain values has a constant sign.

3.1.2.2 Three-Dimensional Transfers

The ΔV cost of an optimal two-impulse transfer can also be found for the case where an inclination change is included with the orbit shape change. Chobotov[54] has given the optimal two-impulse transfer to perform the orbit transfer of interest here. This transfer has been called a "Mod-2 Hohmann" transfer as well as a "dogleg" maneuver, and while it has not been rigorously proven in [54] to be the optimal transfer policy for this case, it is the best known transfer for minimizing ΔV. This optimal transfer entails using the same two-impulse sequence as discussed for the planar transfer, but both impulses occur at a node and have added out-of-plane components that sum to give the total inclination change needed. Thus, the ΔV cost of each impulse can be calculated by applying the law of cosines to a velocity triangle. This velocity triangle has initial and final velocities defined by the ellipse-to-ellipse Hohmann transfer discussed above, and the angle between the initial and final velocities is some fraction of the total inclination change. Given that the angle between the two velocity vectors is some direct fraction of the total inclination change, the maneuvers are assumed to occur at the ascending or descending node of the orbit. The two velocity triangles used to calculate the cost of each impulse are shown in Fig. 3.2. Additional information and analysis on the "Mod-2 Hohmann" transfer can be found in previous work [55, 56]. These maneuvers are consistent with Holzinger's enumeration of valid candidates for optimal impulsive maneuvers (see Table 1 in [43]), but when selecting a minimal set of optimal maneuvers he discards inclination changes at periapsis because those maneuvers do not locally maximize the Hamiltonian. Here, we use inclination changes at periapsis because we are selecting each maneuver with the knowledge of what the next maneuver and inclination change will be. In contrast, in Holzinger's approach there is no a priori knowledge of what future maneuvers will be, so the locally optimal maneuver is selected to maximize the growth of the reachable set.

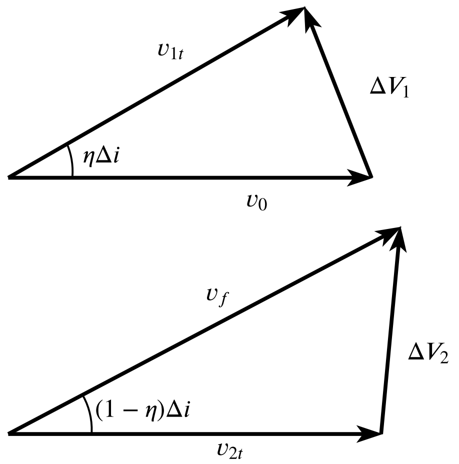


Figure 3.2: Velocity triangles used to calculate $\boldsymbol{\Delta V_1}$ and $\boldsymbol{\Delta V_2}$

The general expression for the optimal ΔV cost to transfer from one elliptical orbit to another elliptical orbit in a different plane using a two-impulse sequence is

$$J(q_0, Q_0, i_0, q_f, Q_f, i_f) = \Delta V_{total} = \Delta V_1 + \Delta V_2 = \sqrt{v_{1t}^2 + v_0^2 - 2v_{1t}v_0\cos(\eta\Delta i)} + \sqrt{v_f^2 + v_{2t}^2 - 2v_{2t}v_f\cos((1-\eta)\Delta i)} \quad (3.13)$$

This equation combines the planar and out-of-plane maneuvers using the law of cosines. Fractional inclination changes at each of the two impulses are dictated by the parameter η, which is defined in the range $0 \leq \eta \leq 1$. Also note that

$$\Delta i = |i_f - i_0|. \quad (3.14)$$

The η^* that minimizes ΔV_{total} can be found by taking the partial derivative of Eq. (3.13) with respect to η and equating it to 0. The result,

$$F = \frac{\partial \Delta V_{total}}{\partial \eta} = \frac{\Delta i v_0 v_{1t}\sin(\eta^*\Delta i)}{\Delta V_1} - \frac{\Delta i v_f v_{2t}\sin((1-\eta^*)\Delta i)}{\Delta V_2} = 0 \quad (3.15)$$

cannot be explicitly solved for η. The implicit function can, however, be solved using a variety of techniques; in this work a bisection method is used to find the correct value for each individual transfer. The bisection method has the benefit of being able to constrain the solution to within the correct bounds for η, which are $0 \leq \eta \leq 1$. Potentially faster methods such as a Newton-Raphson solver are not as robust and can converge to solutions that exist outside of the allowed domain of η. In degenerate cases where $v_0 = v_{1t}$ and/or $v_f = v_{2t}$, Eq. (3.15) can be simplified to avoid undefined behavior.

Partial derivatives of the cost in equation (3.13) will be important for a number of reasons discussed throughout this work. Details of the calculations, however, are included in Appendix A.

3.1.3 Calculation of Time-Free Delta-V Costs for Rendezvous

The analytic expressions for quick computation of optimal transfer costs from one orbit to another can be used to find total ΔV optimal rendezvous orbits. However, an important caveat is that for the multi-spacecraft case, minimizing ΔV is no longer equivalent to minimizing propellant mass, because in general multiple spacecraft do not have equivalent masses and characteristic velocities. Thus, the ΔV optimal results here are closest to mass optimal results when all spacecraft have the same mass and I_{sp}, and are less linked to mass optimal results as the spacecraft masses and I_{sp} values diverge. Rendezvous here is defined as having all spacecraft attain the same $[a_f \quad e_f \quad i_f]$, or the same $[a_f \quad e_f]$ for the planar case, again because we are finding time-free optimal ΔV costs for rendezvous. For time-free rendezvous, the angles ω, Ω, and M are assumed to match for all spacecraft at some time in the future given secular precession of these angles due to J_2. This comes at no additional ΔV cost, so the cost found here is the optimal cost for time-free rendezvous. The single minimum ΔV rendezvous orbit for a system of N spacecraft gives the optimal cost

$$J_{total}^* = \min_{q_f, Q_f, i_f} J_{total} \tag{3.16}$$

where

$$J_{total}(q_f, Q_f, i_f) = \sum_{j=1}^{N} J_j \tag{3.17}$$

$$J_j = J(q_{0,j}, Q_{0,j}, i_{0,j}, q_f, Q_f, i_f) \tag{3.18}$$

and J is defined by Eq. (3.13). J_{total} is simply the sum of the optimal costs for each spacecraft to transfer to a specific final orbit. The problem of minimizing Eq. (3.16) is a variant of the well-known Weber problem. The Weber problem, as stated in Drezner [57], is the problem of finding a point "which minimizes the sum of weighted Euclidean distances from itself to n fixed points." In the spacecraft rendezvous case as formulated here, the point to be found in the Weber problem is the rendezvous orbit, and the weighted distances are the ΔV costs for each spacecraft to transfer to the rendezvous orbit. While there is a long history of work done on the Weber problem, including some solutions to simpler instances of the problem, none of the past methods of solution for the Weber problem appear to give more insight than the techniques discussed below.

The problem of minimizing Eq. (3.16) by choosing q_f, Q_f, i_f is more generally a parameter optimization/nonlinear programming problem. There are a wide number of well-developed techniques that can be used to solve this class of problem, including gradient based methods, genetic algorithms, and grid searches.

Caution must be used when applying gradient-based methods to solving this problem. In addition to the usual concern about only finding local minima, this problem specifically has several boundaries across which the cost function derivatives are not continuous. As discussed in Appendix A, the boundaries occur because the optimal orbit transfer sequence can change depending on the relative values of the initial and final orbits. Thus, as the values of q_f, Q_f, i_f are changed across certain boundaries, the optimal transfer sequence to calculate $J(q_{0,j}, Q_{0,j}, i_{0,j}, q_f, Q_f, i_f)$ will also change, leading to discontinuous derivatives of $J_{total}(q_f, Q_f, i_f)$. Consequently, gradient-based searches would need to be performed separately in different areas, bounded in regions where the derivatives are continuous. Further, as the number of spacecraft in the system increases, the number of these separate regions with differing cost function derivatives also increases. However, gradient-based methods are useful for local optimization, and the partial derivatives provided in Appendix A can be applied to the larger problem of minimizing Eq. (3.16).

In much of this work, a grid search is instead used to explore the structure of solutions to this problem for differing initial conditions. The grid search approach gives insight into the structure of the solutions to the scenarios explored here, whereas other approaches result in single optima without necessarily giving an understanding of the overall solution space. The grid search solution, however, comes at the expense of computation time. Minima found using a grid search can subsequently be refined using a NLP solver such as IPOPT.

To perform the grid search, Eq. (3.13) is used to calculate the optimal two-impulse ΔV costs for a spacecraft with given initial conditions $(q_{0,j}, Q_{0,j}, i_{0,j})$ to transfer to a discrete grid of final orbits in q-Q-i space. For a single spacecraft, this grid allows the computation of the reachable set of orbits for a given amount of available ΔV. For systems of multiple spacecraft, grids from each spacecraft can be summed together to give the cost for all spacecraft to meet at any given orbit in the grid. Exploring this summed grid to find minimum total ΔV rendezvous orbits is equivalent to performing a grid search for the optimal q_f, Q_f, i_f.

The grid search method used here is summarized in Algorithm 1. For a system of N spacecraft, the algorithm finds the minimum rendezvous orbit cost J^*_{total} using a grid of n_q values of q_f, n_Q values of Q_f, and n_i values of i_f.

Algorithm 1: J^*_{total} computation algorithm

```
Result: Optimal cost $J^*_{total}$
for $k = 1, 2, ..., n_q$ do
    for $l = 1, 2, ..., n_Q$ do
        for $m = 1, 2, ..., n_i$ do
            $J_{total}(q_{f,k}, Q_{f,l}, i_{f,m}) \leftarrow 0$ ;
            for $j = 1, 2, ..., N$ do
                $\eta^* \leftarrow$ Eq. (3.15) solution ;
                $J_j \leftarrow J^*(q_{0,j}, Q_{0,j}, i_{0,j}, q_{f,k}, Q_{f,l}, i_{f,m}, \eta^*)$ (Eq. (3.13)) ;
                $J_{total}(q_{f,k}, Q_{f,l}, i_{f,m}) \leftarrow J_{total}(q_{f,k}, Q_{f,l}, i_{f,m}) + J_j$ ;
            end
        end
    end
end
$J^*_{total} \leftarrow \min[J_{total}(q_{f,1:n_q}, Q_{f,1:n_Q}, i_{f,1:n_i})]$
```

Computing dense grids is relatively fast because calculating the ΔV optimal transfer costs relies on a mostly analytic expression. The root-finding algorithm necessary to find an optimal value of η does, however, make the computation slower. Still, computing a grid of 1 million costs J_{total} for a system of four spacecraft takes only 7 minutes on a laptop with a four-core processor. Further, this method lends itself to parallelization on larger scale computing platforms (e.g. supercomputers) because each computation of Eq. (3.13) is independent.

Alternative, derivative-free solution methods (genetic algorithms and particle swarm optimization) have given very similar results as the grid search method, but with a vastly improved runtime. However, without a grid-search solution as a point of comparison, the level of confidence in the solution may or may not be high enough for a given scenario or use case. For operational purposes, the problem may best be solved with a hybrid evolutionary algorithm and NLP solver method.

3.1.4 Time-Free Delta-V Optimal Rendezvous Solutions

3.1.4.1 Two Spacecraft, Planar Rendezvous

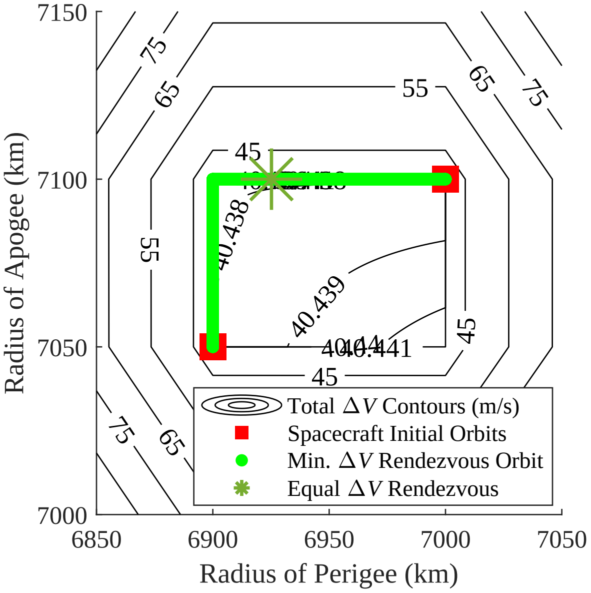


Figure 3.3: ΔV optimal planar rendezvous orbits for a two-spacecraft system

In the planar, two spacecraft case, there is no single, unique ΔV optimal rendezvous orbit. Rather, there is a continuous set of optimal rendezvous orbits that occurs along the transfer path between both spacecraft. The transfer path is dictated by the relative orbital positions of the two spacecraft (see Fig. 3.1), and any point along that path may be selected as the rendezvous orbit to achieve a minimum total ΔV rendezvous. Because the total ΔV for rendezvous is the same at any point along this path, the path is called the "invariant curve." A free parameter here is the relative ΔV expended by each spacecraft to reach the rendezvous orbit. Different rendezvous orbits along the invariant curve will require different relative ΔV expenditures by each spacecraft. This result arises because transfers along the invariant curve can be split into an infinite number of impulses, as discussed in Sec. 3.1.2.1. Thus, both spacecraft can incrementally move along the optimal transfer pathways (while maneuvering only at periapsis or apoapsis) until they meet without making the overall transfer sub-optimal.

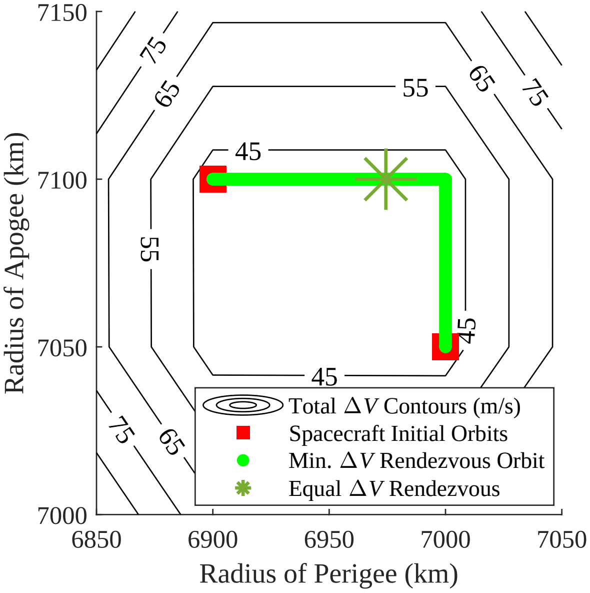


Figure 3.4: ΔV optimal planar rendezvous orbits for a two-spacecraft system

This result can be seen through example grid search results for two-spacecraft planar rendezvous in low Earth orbit (LEO) in Figs. 3.3 – 3.4 ($\mu = 3.986\,004\,415 \times 10^5\,\text{km}^3/\text{s}^2$). Because the grid search is parameterized in radius of periapsis and radius of apoapsis space, the invariant curve can clearly be seen as a set of discrete points all with minimum total ΔV for rendezvous. The minimum total ΔV rendezvous orbits, shown with green circles, match exactly the orbits that are along the optimal transfer pathway between the two spacecraft initial orbits. Fig. 3.4 shows a simplified view with total rendezvous ΔV contour lines, and red squares to show the initial spacecraft orbits. Fig. 3.3 shows a more detailed view of a similar case with additional contour lines detailing the interior structure of the total ΔV in the region where there is relatively little change in total ΔV compared to the outer contour lines. A finer grid search and smaller changes in ΔV between contour lines are used in this interior region for the curved contours. Contour lines of 40.438 m/s, 40.439 m/s, 40.440 m/s, and 40.441 m/s are included, and all narrowly encompass the west and north sides of the invariant curve minima (with cost of 40.4379 m/s).

The invariant curve found here is the same result that was found in [19], which used the reachable set methodology of Holzinger [43, 58] to explore two-spacecraft rendezvous. Thus, the transfers used in this work appear to be consistent with the optimal transfer conditions used to calculate those reachable sets.

The point along the invariant curve where both spacecraft would expend the same amount of ΔV is also included. Due to the nonlinear relationships involved, the point of equal ΔV expenditure does not occur halfway along the invariant curve in $q - Q$ space. To calculate the equal ΔV expenditure point a simple off-the-shelf root finding algorithm is used to search along the invariant curve.

3.1.4.2 Two Spacecraft, Three-Dimensional Rendezvous

When the initial orbits of two spacecraft are not in the same plane, the previously observed invariant curve no longer exists. This result is expected, because the three-dimensional optimal transfers used here cannot equivalently be split into an infinite number of impulses. Instead, for two spacecraft in the three-dimensional case, two types of rendezvous solutions have been found. In the first type, there are two equivalent ΔV optimal rendezvous orbits, each coincident with the initial orbit of one of the spacecraft. That is, the minimum total ΔV redezvous solution is for one of the spacecraft to transfer to the initial orbit of the other spacecraft. Fig. 3.5 shows a sample scenario with two non-coplanar spacecraft where the minimum total ΔV rendezvous orbits are labeled as

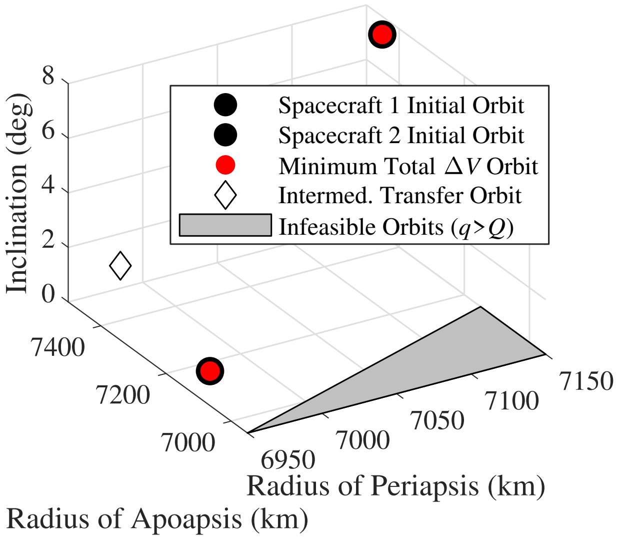


Figure 3.5: Minimum total ΔV rendezvous orbits - two spacecraft, type 1

red dots. The partial derivatives of the total cost are discontinuous at each rendezvous solution, but are positive in each direction that the rendezvous orbit elements can be perturbed. That is,

$$\frac{\partial J_{total}}{\partial q_f^+} > 0 \qquad \frac{\partial J_{total}}{\partial q_f^-} > 0 \tag{3.19}$$

$$\frac{\partial J_{total}}{\partial Q_f^+} > 0 \qquad \frac{\partial J_{total}}{\partial Q_f^-} > 0 \tag{3.20}$$

$$\frac{\partial J_{total}}{\partial i_f^+} > 0 \qquad \frac{\partial J_{total}}{\partial i_f^-} > 0 \tag{3.21}$$

While there are an infinite number of directions that the Gateaux (directional) derivative can be taken in this three-dimensional parameter space, to first order each derivative will be a linear combination of these positive basis derivatives. Because each of these basis derivatives are all positive, all Gateux derivatives must also be positive. This shows that these points satisfy the necessary and sufficient conditions for local minima. This has been confirmed with both analytic derivatives and finite differencing approximations of the partial derivatives of the cost.

To illustrate how a transfer would occur in Fig. 3.5, the intermediate transfer orbit has been included with a diamond marker. For either spacecraft to complete a transfer to the other spacecraft, it would first impulsively maneuver onto the transfer orbit, then impulsively maneuver to reach the initial orbit of the other spacecraft. In this case, $\eta = 0.2$ for the spacecraft with 0 inclination, and $\eta = 0.8$ for the spacecraft with an inclination of 7.4°. Thus, the transfer orbit has an inclination of 1.5°, which is 20% of the total inclination difference between the two spacecraft. Periapsis and apoapsis changes are made as in the planar case (see Fig. 3.1), but without the option of splitting changes to a single apsis into multiple impulses. Consequently, the transfer orbit has a periapsis radius equal to that of the 0 inclination spacecraft (6973 km) and apoapsis radius equal to that of the spacecraft with an inclination of 7.4° (7446 km).

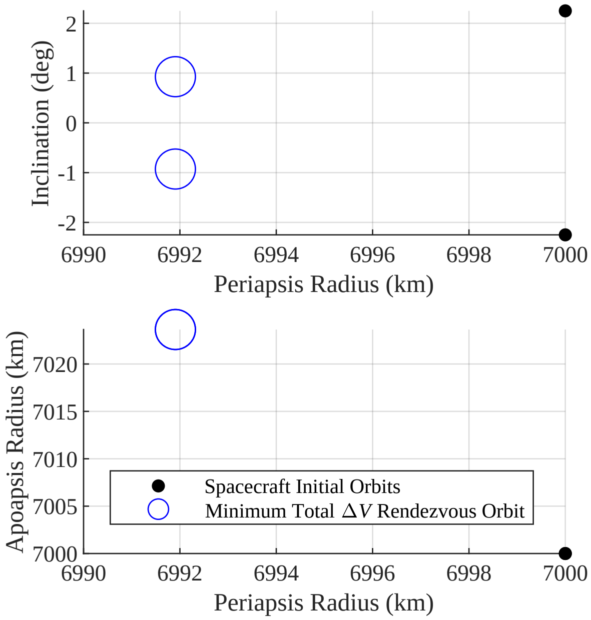
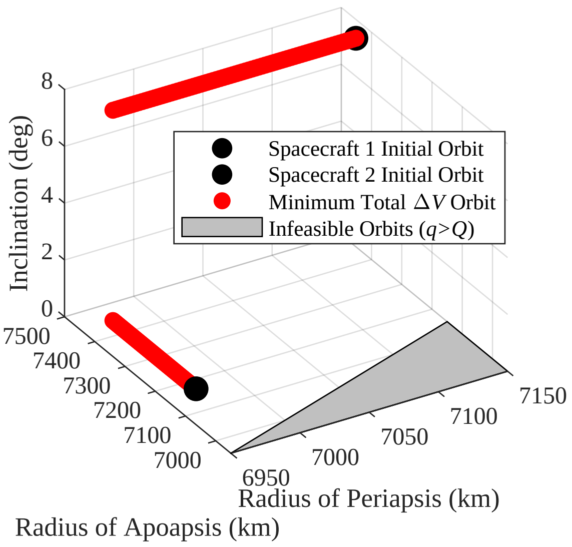


Figure 3.6: Minimum total ΔV rendezvous orbits - two spacecraft, type 2

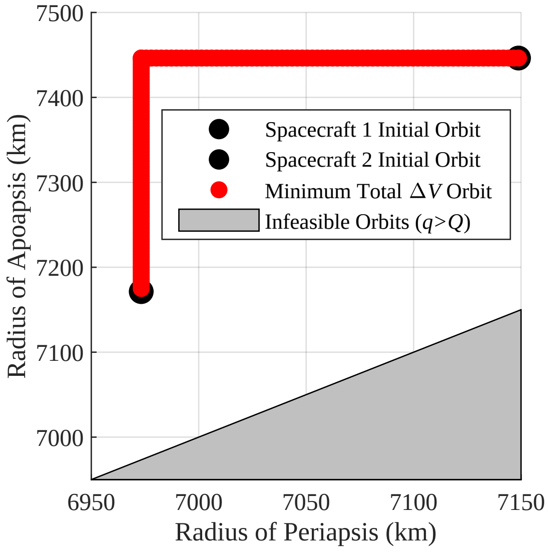
The second type of two spacecraft rendezvous orbit found is separate from the initial orbits of the two spacecraft. That is, in certain cases we find that either one or two total ΔV optimal rendezvous orbits are found in locations not coincident with the initial two spacecraft orbits. A simple example is shown in Fig. 3.6, where optimal rendezvous orbits for two non-coplanar spacecraft in circular orbits of equal radius are shown. In this case, it is actually advantageous for both spacecraft to rendezvous at one of two elliptical orbits of intermediate inclination where the periapsis radius is lower than the circular orbit radius and the apoapsis radius is larger than the circular orbit radius. In contrast to a pure plane change maneuver required for one spacecraft to match the initial orbit of the other, this rendezvous orbit allows the spacecraft to use more efficient plane changes at slower speeds, as well as the dogleg maneuver than can further minimize total ΔV. For both of the equivalent ΔV optimal rendezvous orbits in Fig. 3.6, both spacecraft would perform more than half of the total inclination change at apoapsis (i.e. $\eta < 0.5$, and in this case $\eta_1 = 0.18$ and $\eta_2 = 0.44$).

Two equivalent optimal rendezvous orbits are found due to the symmetries of the two spacecraft initial orbits; they have equal and opposite inclinations as well as equal circular orbit altitudes. In test cases with this solution type but without such symmetries, single "separate" optimal rendezvous orbit are found. The ΔV cost improvement found in Fig. 3.6 is minor, however, at about 0.07% less total ΔV (0.4 m/s) than a simple transfer from one spacecraft initial orbit to the other's orbit. The ΔV savings found in other cases was also of similar magnitude. The magnitude of this cost improvement likely makes the discovery of these types of solutions more difficult. It is possible that this type of solution is always the true global optimum for the two spacecraft case, and the first type of coincident solution is only found due to a failure to find the non-coincident solution. In the case of Fig. 3.6 and other similar scenarios, a fine grid search combined with local gradient-based optimization with IPOPT [59] was used to find those solutions.

We can also demonstrate why the invariant curve does not exist for three-dimensional transfers by examining a scenario where sub-optimal transfers are used to calculate the cost J_{total}. To calculate the suboptimal transfer, the cost J_{total} was calculated by assuming that the total inclination change was performed at the point in the Hohmann transfer where the spacecraft's speed was the lowest, giving a minimum of three impulses for the total transfer (2 maneuvers for the orbit shape changes, 1 maneuver for the inclination change). These sub-optimal transfers allow the total transfer to be broken up into many impulses, so a type of invariant curve is again found, as seen in Fig. 3.7. In this case, there is a seemingly continuous number of options for final rendezvous orbits that cost the same total ΔV for both spacecraft, but with an inclination jump dividing the options. That is, all ΔV optimal rendezvous orbits share the same inclination as one of the spacecraft initial orbits.



(a) 3D view



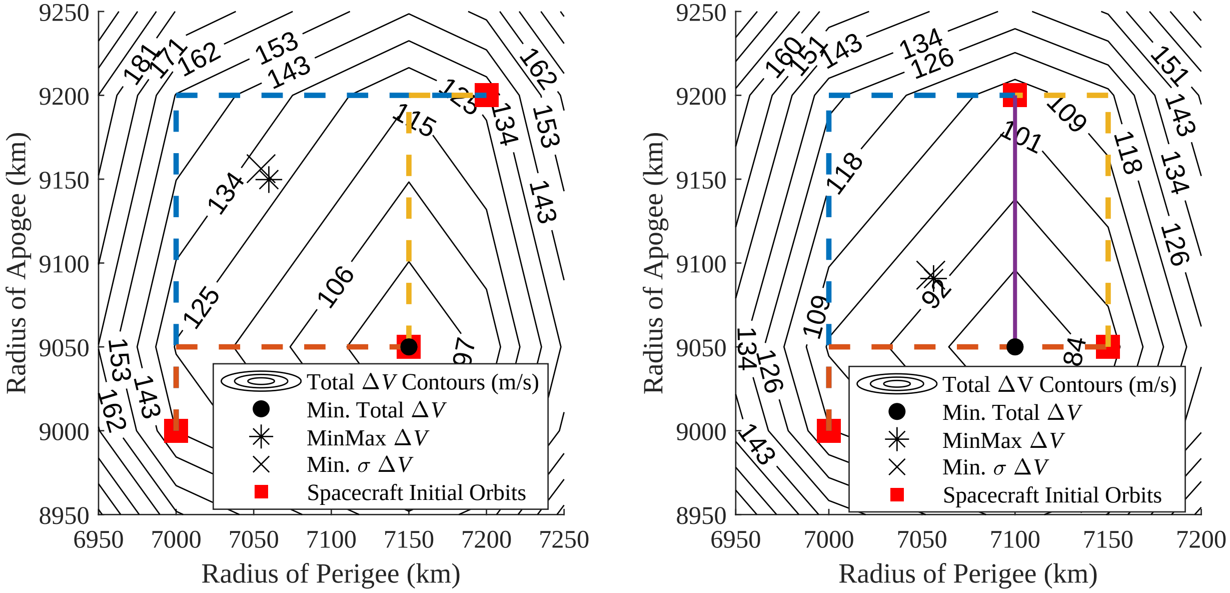
(b) 2D view

Figure 3.7: Minimum total ΔV rendezvous orbits without use of dogleg maneuvers

3.1.4.3 More Than Two Spacecraft, Planar Rendezvous

The grid search process can be extended to systems of more than 2 spacecraft. This results in total ΔV contour plots such as those in Figs. 3.8 (three spacecraft) and 3.9 (four spacecraft). The overall minimum ΔV orbit is now a single point, as opposed to the invariant curve observed for two-spacecraft systems. In Fig. 3.8a, this minimum ΔV point happens to be coincident with the initial orbit of one of the spacecraft. But this is not always true, as seen in Fig. 3.8b. Also note that the minimum total ΔV rendezvous orbit does not lie on an intersection of all optimal transfer pathways between each pair of spacecraft in the system. The optimal transfer pathways between each pair of spacecraft are indicated with uniquely colored dotted lines in Figs. 3.8–3.9 and 3.15–3.16. In Fig. 3.8b the optimal transfer path from the spacecraft with the largest Q to the minimum total ΔV rendezvous orbit is indicated with a purple solid line, because the rendezvous orbit does not lie on any of the inter-spacecraft transfer pathways connected to that spacecraft.

To explore the impact on the minimum ΔV rendezvous orbit when adding a third spacecraft to a two-spacecraft system, the orbit space is divided into 9 sections as seen in Fig. 3.10. The



(a) Scenario 1

(b) Scenario 2

Figure 3.8: Optimal rendezvous orbits for three-spacecraft systems. See Sec. 3.1.4.5 for detail on MinMax ΔV and Min. σ ΔV orbits.

grid is used to explore where the minimum total ΔV rendezvous orbit will lie if a third spacecraft is added to an existing system of two-spacecraft in red. The interior box section defined by the two initial spacecraft is not considered, because if a third spacecraft were placed in that section, it could be considered one of the original two, with one of the other spacecraft being the third added spacecraft.

When a third spacecraft is added to the system, the minimum total ΔV rendezvous point will lie on the box whose corners are defined by the orbits of the initial two spacecraft. Further, the minimum total ΔV rendezvous orbit will be the point on the box that costs the least amount of ΔV for the third spacecraft to transfer to. The control policy shown in Fig. 3.1 also dictates the point on the box that will cost the least amount of ΔV. While a static set of two spacecraft defines a box in Fig. 3.10, any two spacecraft for the three-spacecraft case can be selected to define the "box," and the minimum total ΔV rendezvous orbit will be located on the closest point on the box to the third spacecraft. With this in mind, the impact of adding a fourth spacecraft to a system of

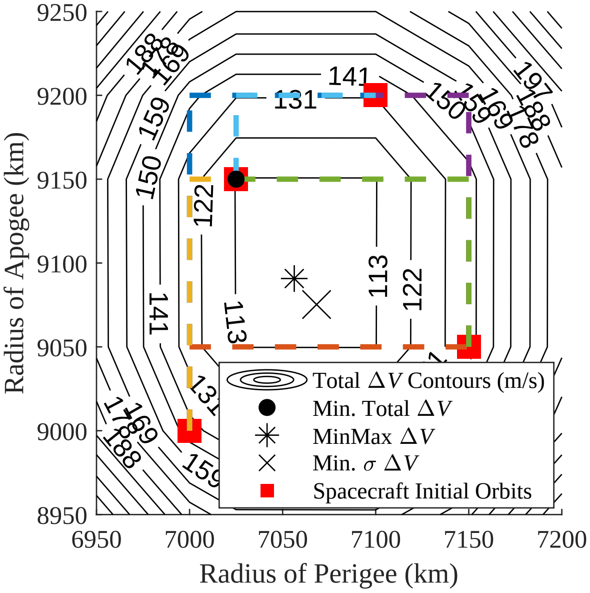


Figure 3.9: Optimal rendezvous orbits for a four-spacecraft system

three spacecraft is considered.

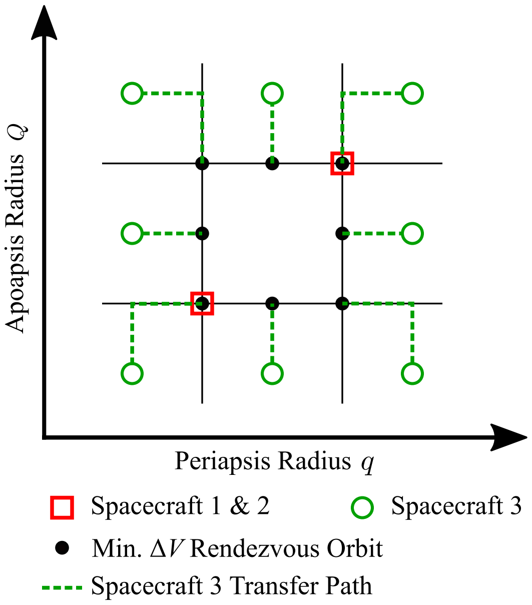


Figure 3.10: Minimum ΔV rendezvous orbit location when adding third spacecraft

In Fig. 3.11, there is a similar set of 9 sections as in the three spacecraft case in Fig. 3.10. Now, however, the interior section is the box defined by the initial three spacecraft, and the addition of a fourth spacecraft in different relative positions with respect to this interior box is explored. The locations of the minimum ΔV rendezvous orbit have some similarities to the three spacecraft case, but two important differences. First, if a fourth spacecraft is added to the bottom middle section (at positions A or B), the minimum total ΔV rendezvous orbit lies within the box defined by the initial three spacecraft, as opposed to lying strictly on the box as in the three spacecraft case. Second, while many minimum ΔV rendezvous orbits lie on the box defined by the three initial spacecraft, these points on the box are no longer strictly the closest point on the box to the orbit of the added spacecraft. For example, if a fourth spacecraft is added to the top left section at position C, the minimum total ΔV rendezvous orbit is not at the top left corner of the interior box as is the case in the three spacecraft scenario. This implies that there is not a recursive relationship with respect to the location of minimum total ΔV orbit locations as more and more spacecraft are added to the system.

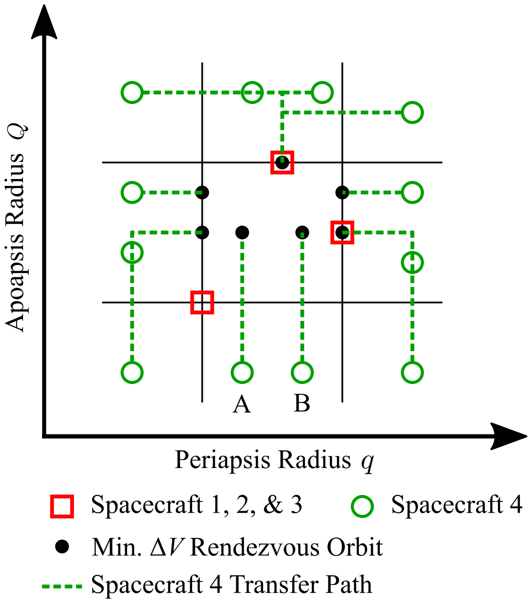
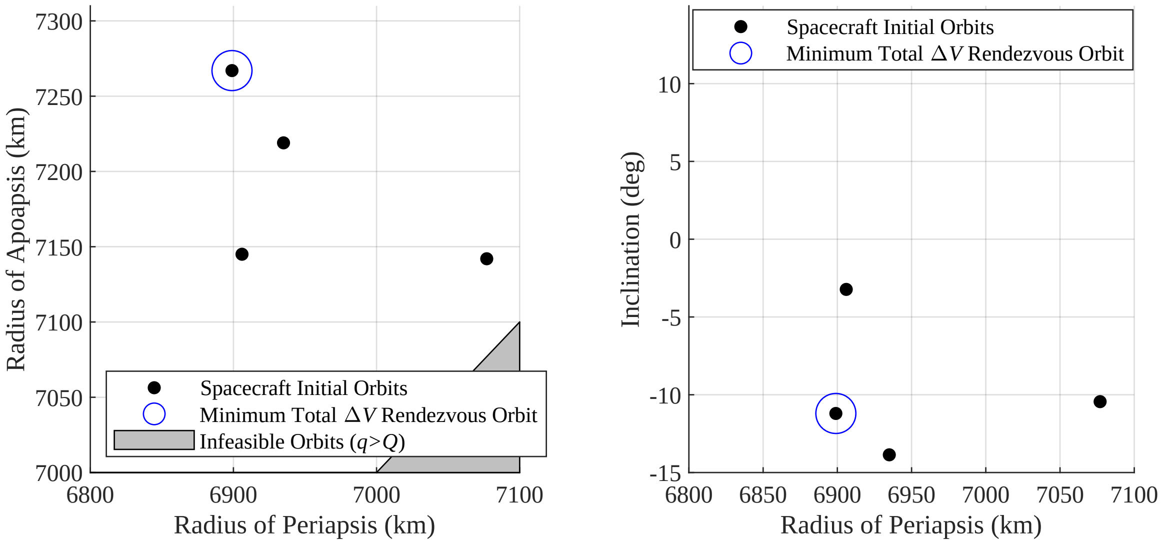


Figure 3.11: Minimum ΔV rendezvous orbit location when adding fourth spacecraft

3.1.4.4 More Than Two Spacecraft, Three-Dimensional Rendezvous

In the three-dimensional case with $N > 2$ where orbits exist in $q-Q-i$ space, there are three types of solutions for minimum total ΔV rendezvous orbits that can be found. First, minimum ΔV orbits can be found to be coincident with the initial orbit of one of the spacecraft, as seen in Fig. 3.12. In this case, it is ΔV optimal for all spacecraft to meet at the orbit of a single spacecraft. Like the two-spacecraft case, the partial derivatives of the cost function with respect to the rendezvous orbit elements are non-continuous, but the cost is found to be increasing in each direction the orbit elements are varied.



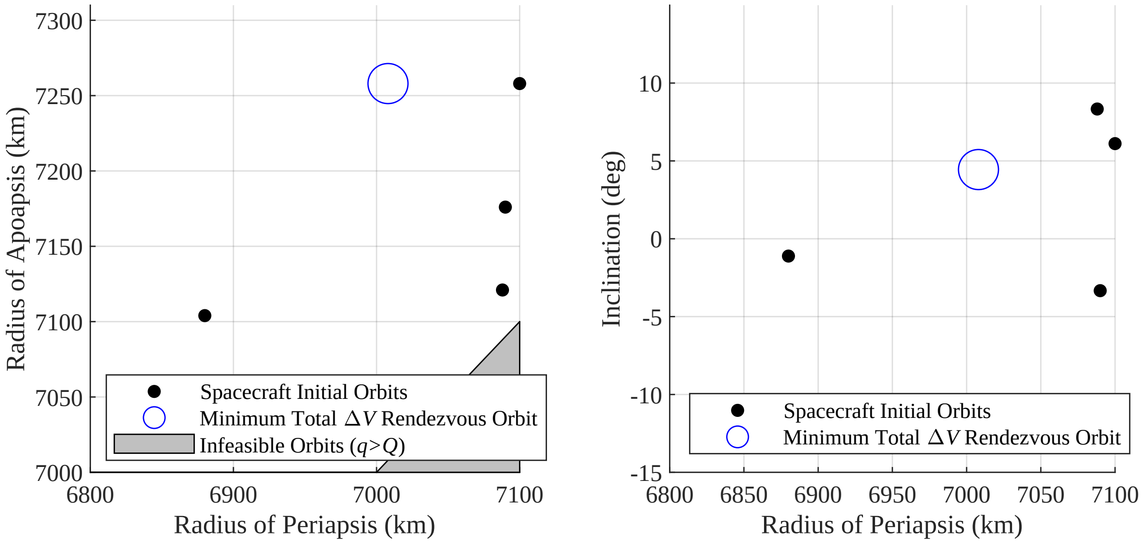
(a) 2D view - periapsis and apoapsis

(b) 2D view - periapsis and inclination

Figure 3.12: Minimum total ΔV rendezvous orbit coincident with initial spacecraft orbit

Importantly, this is not the only type of minimum ΔV rendezvous orbit that is found. Some optimal orbits are also found to be interior to the initial conditions of the spacecraft. That is, the optimal rendezvous orbit lies within the range of initial spacecraft orbits, as in Fig. 3.13. In all test cases with interior solutions we found that the optimal rendezvous orbit shares at least one orbit element with one of the initial spacecraft orbits, but we cannot conclusively state that this is true in all cases of interior solutions. The interior region is perhaps the most natural range to search;

one might expect that the optimal rendezvous orbit does not have a larger q, Q, or i than the highest spacecraft, or a smaller q, Q, or i than the lowest spacecraft. However, as seen in Fig. 3.14, sometimes the optimal rendezvous orbit will lie exterior to the range of spacecraft initial orbits. The optimal rendezvous orbit in Fig. 3.14 has a larger radius of apoapsis than any of the other spacecraft initial orbits. Solutions such as this one are in a region where the partial derivatives are continuous, so it is relatively simple to use a NLP solver to find the local minimum that satisfies the necessary conditions when starting from a more coarse approximation of the minimum provided by a grid search or evolutionary algorithm. Thus, when using this technique to optimize a given scenario, optimal solutions may be missed if a sufficiently large search space is not explored.



(a) 2D view - periapsis and apoapsis

(b) 2D view - periapsis and inclination

Figure 3.13: Minimum total ΔV rendezvous orbit interior to spacecraft initial conditions

3.1.4.5 Alternative Optimization Criteria

The grid search method of finding total ΔV optimal rendezvous orbits also enables the use of alternative optimization criteria because the method involves calculating the ΔV cost for each individual spacecraft to transfer the rendezvous orbit. Of potential interest is minimizing the largest

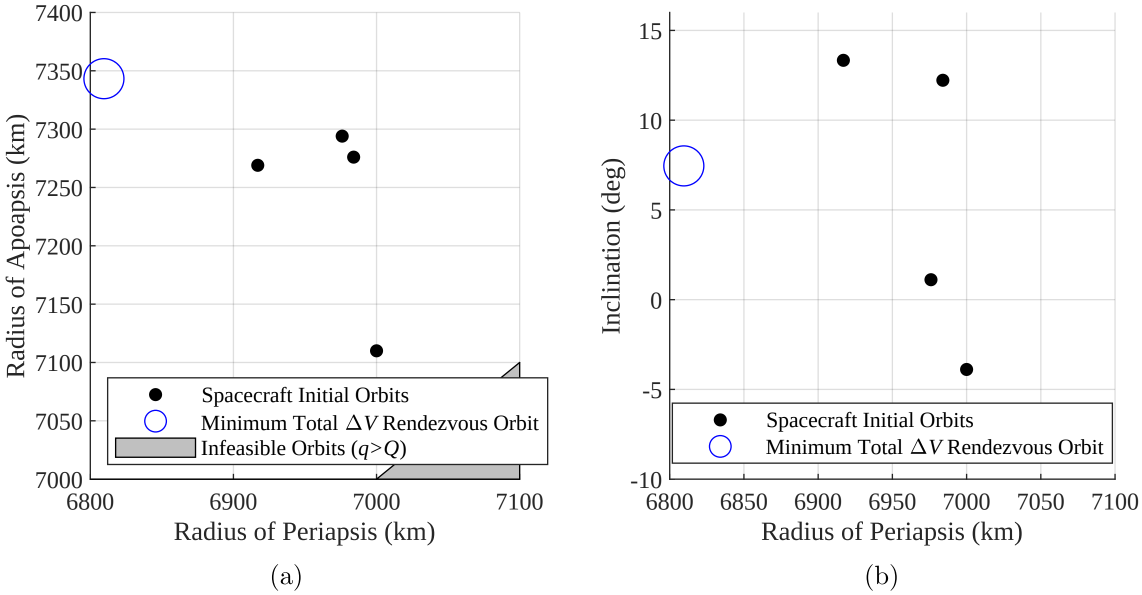


Figure 3.14: Minimum total ΔV rendezvous orbit exterior to spacecraft initial conditions

single ΔV expended by a spacecraft for a given rendezvous orbit. Fig. 3.15 shows a contour plot of the maximum single ΔV expended by a spacecraft for a planar four spacecraft system, with the smallest maximum indicated by a star.

Another criteria of interest is minimizing the standard deviation of the ΔV expended by each spacecraft to reach the rendezvous orbit. This is another method of preventing certain spacecraft from expending too much ΔV relative to its peers in a formation or swarm. Fig. 3.16 shows contour plots of the standard deviation of all ΔV expenditures needed at each rendezvous orbit. While the minimum σ orbit is close to the min/max point in both examples shown here, this is not always the case.

Note that these examples are planar for the purposes of visualization, but the same alternative optimization criteria can be applied in the three-dimensional case as well. The flexibility of this method further allows analysis of other cases, such as cases where a single spacecraft has limited fuel. Given the reachable set of a single limited spacecraft, options within that set can be explored with respect to the implications for the full formation.

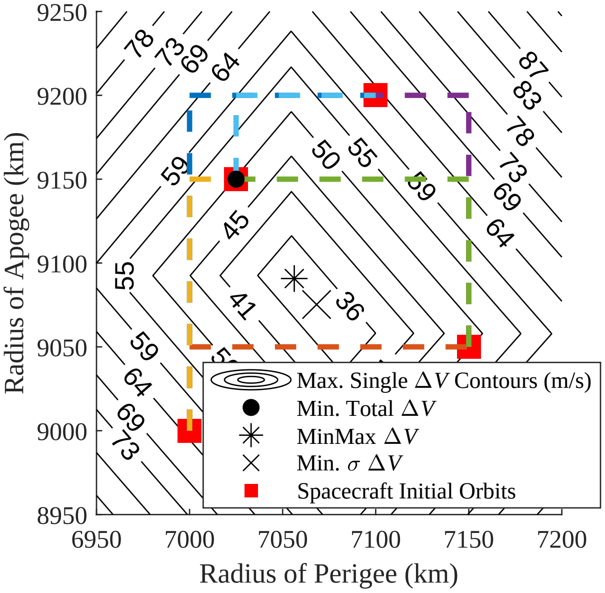


Figure 3.15: Contour plot of maximum ΔV expended by a single spacecraft for each possible rendezvous orbit

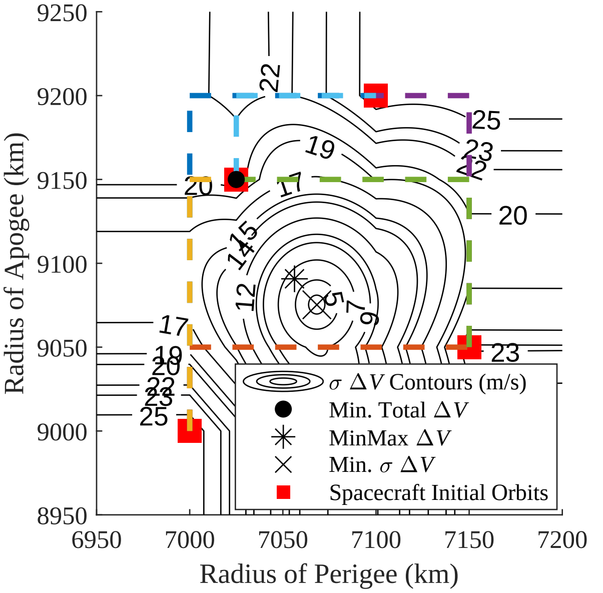


Figure 3.16: Contour plot of the standard deviation of ΔV for all spacecraft

3.1.5 Examples

3.1.5.1 Finite Time Implementation for Full Rendezvous

Because the current method described thus far is time-free, it only describes a method of calculating optimal ΔV costs for a set of spacecraft to achieve the same q, Q, and i while J_2 perturbations and an indeterminate amount of time (possibly infinite) are used to allow the other three orbit elements to match for rendezvous. However, it is possible to realize in finite time these optimal ΔV costs that are found with the time-free method while using the same maneuvers found with that method. To demonstrate this, we begin with a set of N spacecraft with given initial values of $q_{0,k}, Q_{0,k}$, and $i_{0,k}$ where $k = 1, 2, ..., N$. Using the optimal rendezvous orbit a_f, e_f, i_f found with the time-free method, we discuss how to find initial conditions for all spacecraft such that the time-free optimal ΔV cost is not exceeded for the spacecraft to further achieve the same argument of periapsis (ω_f), right ascension of the ascending node (RAAN, Ω_f), and mean anomaly (M_f) at the rendezvous orbit in addition to achieving the same a_f, e_f, and i_f. Thus, we are investigating how to find $\Omega_{0,k}$, $\omega_{0,k}$, and $M_{0,k}$ at time t_0, and how to find ω_f, Ω_f, and $M_{0,f}$ of the rendezvous orbit such that the optimal ΔV value can be realized.

First, note that the optimal transfer sequence dictates that each of the two impulsive maneuvers for each spacecraft's transfer must be performed at periapsis or apoapsis. The spacecraft will also be restricted to an argument of periapsis $\omega_{i,k} = 0$ to ensure that maneuvers performed at periapsis or apoapsis also occur at an ascending or descending node. This is necessary given the velocity triangles used to find the equations for optimal ΔV costs; the velocity triangles used here assume that inclination changes occur at a node. Under these assumptions, no impulsive maneuver can change the orbit RAAN or ω, so these values need not be considered when targeting a set of final orbit elements. That is, $\omega_f = \omega_{i,k} = 0$, and $\Omega_{i,1} = \Omega_{i,2} = ... = \Omega_{i,N}$. This restricts all spacecraft to having the same initial RAAN, which will also be the same for the rendezvous orbit.

Further, the anomaly angle where the spacecraft inserts itself into the final orbit is dictated by the Hohmann transfer, and will occur at periapsis or apoapsis of the final orbit. Given these restrictions, a choice must be made as far as when each spacecraft should arrive at the final rendezvous orbit. All spacecraft can only simultaneously arrive at the rendezvous orbit if the optimal transfer sequence dictates that they insert into the final orbit at the same true anomaly. This occurs when all spacecraft have a larger initial apoapsis radius than the rendezvous orbit (insert at rendezvous orbit periapsis) or if all spacecraft have a smaller initial apoapsis radius than the rendezvous orbit (insert at rendezvous orbit apoapsis). If neither is the case, then one set of spacecraft can insert at periapsis while the other inserts at apoapsis. Of course, this ignores operational concerns with having several spacecraft meet at the same point in space at the same time. Alternatively, each spacecraft can be assigned a unique time of periapsis or apoapsis passage on the rendezvous orbit for insertion.

As an example, we'll find $M_{0,k}$ for all spacecraft in the scenario in Fig. 3.14, which has a rendezvous orbit with a larger apoapsis radius than all spacecraft in the system. In this case, all spacecraft maneuver first at their initial periapsis, and maneuver second at their final apoapsis. We'll further assume that all spacecraft will meet at the same time at apoapsis of the final rendezvous orbit, though it will become apparent that the spacecraft initial conditions can be selected such that the spacecraft can insert into the rendezvous orbit at different apoapsis passage times.

The first step is to find the total transfer time from each spacecraft to the final rendezvous orbit. The subscript tr is used to indicate that a value corresponds to the transfer orbit being used to connect the initial and final orbits. The transfer time $t_{tr,k}$ is simply one half of the period of the transfer orbit connecting the initial and final orbits.

$$t_{tr,k} = \frac{T_{tr,k}}{2} = \pi\sqrt{\frac{a_{tr,k}^3}{\mu}}, \quad k = 1, 2, ..., N \tag{3.22}$$

Further, a time of rendezvous t_f is set where the final rendezvous orbit has a mean anomaly $M_f = \pi$. Then, each spacecraft must have $M_{i,k} = 0$ at time $t_{i,k}$ when each spacecraft makes its first maneuver. With the given information, it is known that

$$t_{i,k} = t_f - t_{tr,k} \tag{3.23}$$

For spacecraft j that has the largest time $t_{tr,j}$, we set $t_f = t_{tr,j}$, so that $t_{i,j} = t_0 = 0$. We then simply need to use the times $t_{i,k}$ to find each $M_{0,k}$ from which after $t_{i,k}$ seconds $M_{i,k} = 0$.

If each spacecraft has an initial orbit mean motion of $n_{i,k}$, where

$$a_{i,k} = \frac{q_{i,k} + Q_{i,k}}{2} \tag{3.24}$$

$$n_{i,k} = \sqrt{\frac{\mu}{a_{i,k}^3}} \tag{3.25}$$

then

$$M_{0,k} = M_{i,k} - n_{i,k} t_{i,k} = -n_{i,k} t_{i,k} \tag{3.26}$$

The mean anomaly of the rendezvous orbit at time t_0, $M_{0,f}$, is

$$M_{0,f} = M_f - n_f t_f = \pi - n_f t_f \tag{3.27}$$

Table 3.2: Constants used for calculations

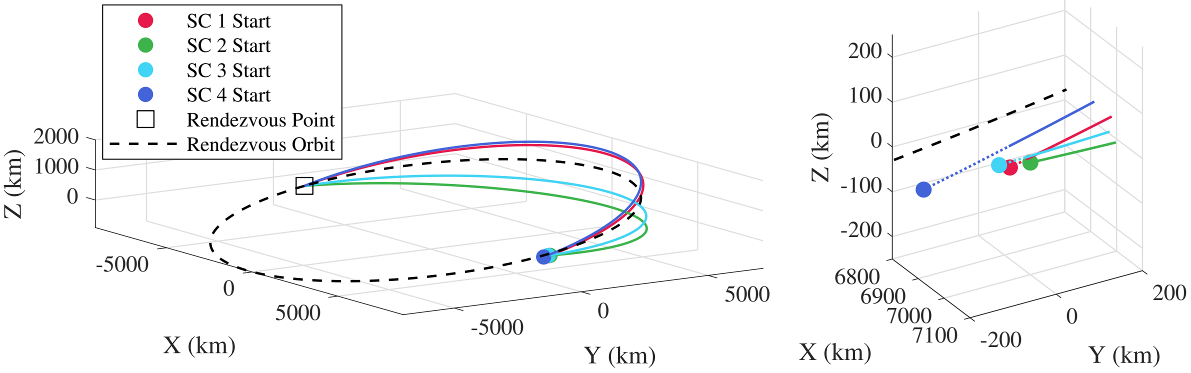
Constant	Value
Radius of Earth (r_c)	6378 km
J_2	0.0010826269
Gravitational Parameter of Earth (μ)	$3.986\,004\,415 \times 10^5\ \mathrm{km}^3/\mathrm{s}^2$

Table 3.3: Initial conditions for system of four spacecraft and the corresponding optimal rendezvous orbit

Value	Spacecraft 1	Spacecraft 2	Spacecraft 3	Spacecraft 4	Rendezvous
q_0 (km)	6984	7000	6976	6917	6809.5
Q_0 (km)	7276	7110	7294	7269	7343.2
i_0 (deg)	12.2	-3.9	1.1	13.3	7.452

Thus, given the constants in Table 3.2, and for a system of four spacecraft with initial conditions listed in Table 3.3, we have found the remaining initial conditions shown in Table 3.4 that allow the spacecraft to rendezvous for the same ΔV cost found with the time-free ΔV optimal method. Note that initial and final ω and Ω have been set to 0 in this example; ω must be 0, but Ω can equivalently be any value that remains the same in the initial and rendezvous orbits.

This rendezvous scenario is illustrated in Fig. 3.17, where the transfer orbits are shown as solid lines, the final rendezvous orbit is shown with a black dotted line, and the initial spacecraft



(a) Wide View

(b) Detail view of start points

Figure 3.17: Cartesian visualization of full rendezvous scenario

trajectories prior to the first maneuver are shown as colored dotted lines in Fig. 3.17b. Note that spacecraft 2 spends no time on its initial orbit, because in this simulation its first maneuver occurs at t_0. Figure 3.18 shows how q, Q, and i of each spacecraft varies over time. All spacecraft perform their final maneuver at the same time at the same rendezvous point in space, while the first maneuver time varies. The x-axis time scale of the Q plot is modified to show detail in the initial maneuver time; apoapsis remains unchanged after the first maneuver for each spacecraft.

The preceding method can be modified in a number of ways to adapt to different scenarios. For example, the spacecraft initial conditions can be found such that they insert into the rendezvous orbit at different times. This method can also be modified to work in scenarios where spacecraft must insert into different parts of the rendezvous orbit.

Finally, the secular orbit perturbation due to the J_2 term of a non-spherical central body can also be included. This perturbation secularly affects an orbit's Ω, ω, and the mean motion n. To account for these secular effects, the orbital period T in Eq. (3.22) should be replaced with

$$\overline{T} = \frac{2\pi}{\overline{n}} \tag{3.28}$$

where $\overline{n}$ is the secular rate of mean anomaly given a J_2 perturbation. $\overline{n}$ should also replace n in

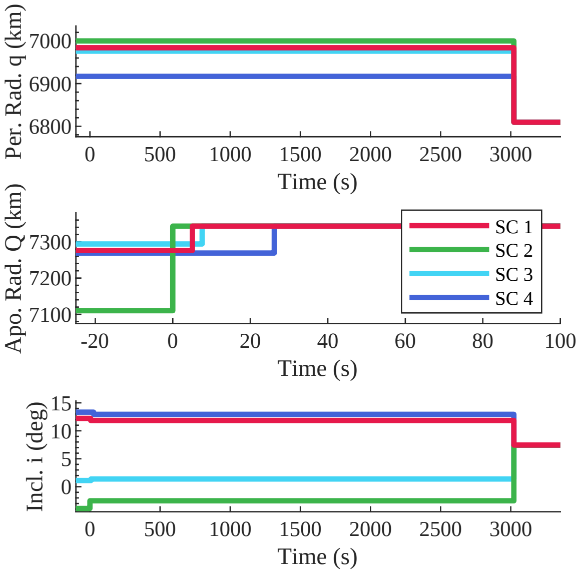


Figure 3.18: Orbit elements over time for full rendezvous scenario

Eqs. (3.26–3.27), and is given in [60] as

$$\overline{n} = n\left[1 + \frac{3C_{20}\sqrt{1-e^2}}{2p^2}\left(\frac{3}{2}\sin^2 i - 1\right)\right] \tag{3.29}$$

The secular rates of Ω and ω are also given in [60] as

$$\frac{d\overline{\omega}}{dt} = \frac{3nJ_2r_c^2}{2p^2}\left(\frac{5}{2}\sin^2 i - 2\right) \tag{3.30}$$

$$\frac{d\overline{\Omega}}{dt} = \frac{3nJ_2r_c^2}{2p^2}\cos i \tag{3.31}$$

where r_c is the radius of the central body. An important caveat here is that in the J_2 perturbed case, ω can no longer be held constant at 0 due to the secular drift $\frac{d\overline{\omega}}{dt}$. Thus, the maneuvers at periapsis and apoapsis will no longer occur at an ascending or descending node, and the velocity vector rotations in Fig. 3.2 (Eq. (3.13)) no longer directly translate to inclination changes. However, as a practical matter for Earth orbiting spacecraft specifically, the secular drift rate $\frac{d\overline{\omega}}{dt}$ is on the order of 10^{-6} radians/sec. Therefore, for practical, short term considerations, ω can be considered to be essentially constant, and can be set to be roughly 0.

Working backwards from a certain desired rendezvous orbit Ω_f ($M_{0,f}$ is found in the same

Table 3.4: Remaining initial conditions to realize optimal rendezvous cost

Orbit	M (deg)
Spacecraft 1 (M_0)	359.7
Spacecraft 2 (M_0)	0
Spacecraft 3 (M_0)	359.5
Spacecraft 4 (M_0)	358.4
Rendezvous Orbit ($M_{0,f}$)	356.4

manner as before but with modified mean motion, ω is considered to be essentially constant), the changes in Ω as it is perturbed by J_2 throughout the transfer process must be accounted for. Because we take $\omega \approx 0$, and because each maneuver is only tangential at periapsis or apoapsis, each maneuver still will not impulsively change Ω, ω, or M. This can be observed, for example, by examining Gauss' variational equations [61]. Thus, to find the initial conditions $\Omega_{0,k}$, one can account for all changes in Ω with

$$\Omega_{0,k} = \Omega_f - t_{tr,k} \frac{d\overline{\Omega}}{dt} \tag{3.32}$$

Consequently, for the case where ω is set to be roughly 0, the optimal transfer results discussed here can be matched fairly closely in the J_2 perturbed case. The precession of Ω alone does not impact the theoretical ability to match the time- and orientation-free transfer cost. As ω deviates more from 0, the ΔV cost relative to these results will increase.

3.1.5.2 Constellation Deployment

One interesting application of this method is to find the optimal orbit from which to launch a constellation of several spacecraft. The deployment problem in this case is essentially rendezvous in reverse; the desired final orbits for spacecraft in the constellation can be fixed as the "initial conditions," and the minimum ΔV rendezvous orbit found in the same manner as above represents the optimal orbit from which to launch the spacecraft to their final orbits.

For a constellation of four spacecraft in circular, 7000 km orbits, with inclinations of $-9°$, $-5.25°$, $5.25°$, and $9°$ (symmetric), the optimal deployment orbit is not actually a circular, 7000 km orbit. Rather, the optimal deployment orbit has 0° inclination, with $q = 6901$ km and $Q = 7252$ km

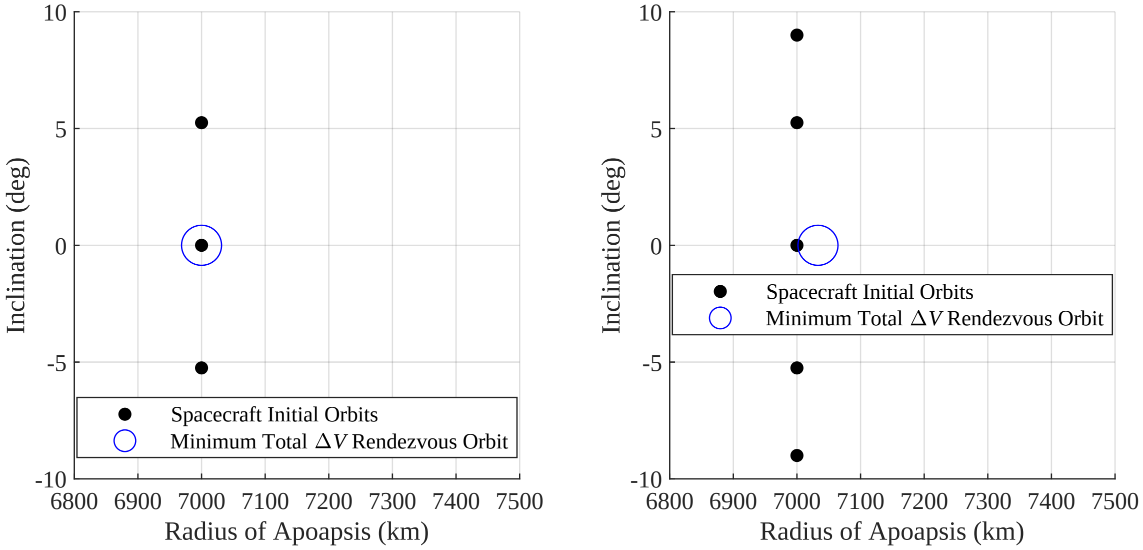
Table 3.5: ΔV savings for different deployment scenarios

Final Orbit Inclinations (deg)	ΔV Cost Savings Over Nominal Deployment Orbit (m/s)	Optimal Deployment Orbit [q,Q,i] (km, km, deg)
[0, ±5.25, ±9]	2.7	[7000.0, 7033.1, 0]
[0, ±5.25, ±12]	8.4	[7000.0, 7064.6, 0]
[0, ±5.25, ±9, ±12]	26.9	[7000.0, 7148.8, 0]
[0, ±5.25, ±9, ±14]	37.9	[7000.0, 7184.9, 0]
[±5.25, ±9, ±12]	101.1	[6862.1, 7377.4, 0]
[±5.25, ±9, ±14]	126.1	[6853.4, 7445.5, 0]

(exterior type solution). However, for a constellation of three spacecraft in circular, 7000 km orbits, with inclinations of $-5.25°$, $5.25°$, and $0°$ (Fig. 3.19a), the optimal deployment orbit is a 7000 km circular orbit with $0°$ inclination (coincident type solution). If two spacecraft of inclinations $-9°$ and $9°$ are added to the previous case (Fig. 3.19b), the optimal deployment orbit shifts to $i = 0°$, $q = 7000$ km, and $Q = 7033$ km (exterior type solution). If the central spacecraft with final inclination of $0°$ in Fig. 3.19b is moved above or below $0°$ of inclination, the deployment orbit follows suit. For example, if the central spacecraft with inclination of $0°$ is moved to an inclination of $1°$, the optimal deployment orbit will also have an inclination of $1°$. This is generally found to be the case in "unbalanced" distributions; the ΔV optimal deployment orbit matches the inclination of the central spacecraft.

The ΔV savings for the scenario in Fig. 3.19b is only about 2.7 m/s less total ΔV than if all spacecraft were to deploy from a nominal, circular deployment orbit with $q = 7000$, $Q = 7000$, and $i = 0$. However, as shown in Table 3.5, the cost savings over the nominal, circular deployment orbit grows as the destination orbits increase their final inclination values. The savings further increase if there is no spacecraft to be deployed at zero inclination. Also note that the optimal deployment orbits move farther and farther away from the 7000 km circular orbit as the ΔV savings increases.

This type of analysis can be useful when performing integrated system wide optimization to determine what launch options, orbit planes, and satellite configurations are most cost effective for a given set of constellation objectives. The notable result that some intermediate deployment orbit may provide relatively low total-ΔV-cost access to several different orbital inclinations may enable certain system architectures. The analysis here can be used as a starting point to explore the trade-offs for which inclination orbits are best serviced by a single launcher, the amount of ΔV needed for each satellite to insert into its initial orbit, and the most economical number of satellites to place on a single launcher. Further, questions of ground coverage, overall satellite lifetime, and more can also be considered simultaneously along with those factors.



(a) Three spacecraft

(b) Five spacecraft

Figure 3.19: Optimal constellation deployment orbits

3.1.6 Conclusions

Locations and lower bound costs of ΔV optimal cooperative rendezvous orbits for a system of many spacecraft can be found using time- and orientation-free ΔV optimal transfers. For two spacecraft in the same orbit plane, there are an infinite number of total ΔV optimal rendezvous orbits. As the number of spacecraft is increased and/or if the spacecraft are initially placed in different planes, the number of optimal rendezvous orbits is reduced to one. Importantly, the ΔV optimal rendezvous orbit is not necessarily at one of the initial orbits of one of the spacecraft. While the methods used in this section give lower bound ΔV costs for rendezvous where fuel is used only to change a, e, i, it has also been shown that this lower bound cost for rendezvous is achievable in finite time with all spacecraft matching all orbit elements. Therefore, this lower bound is not over-conservative and can be used in mission design scenarios or as a reference value. Notably, the method described here could be used to provide an initial guess for a multi-spacecraft finite time transfer where there is less freedom to choose the initial angles. Section 4.2 explores the finite time full rendezvous/deployment optimal control problem.

3.2 Integer Linear Programming: Lower Bounds on Delta-V Costs for Multi-Target Space Missions

3.2.1 Introduction

Spacecraft missions where one or more spacecraft each visit multiple targets are attractive from the standpoint of efficiently accomplishing a number of goals, but trajectories for these missions can be very difficult to optimize. The introduction of multiple targets and multiple spacecraft can rapidly increase the size of the design space such that efficiently exploring the space for optimal mission architectures quickly becomes difficult. Applications of these types of missions include space-debris removal missions where one or more spacecraft de-orbits multiple debris objects [62, 63, 64], tours of multiple asteroids with one or more spacecraft [65, 66, 67], and using a single spacecraft to refuel multiple on-orbit satellites [68, 69]. Further, there is also renewed interest in paradigms with on-orbit supply depots, such as having an orbiting spacecraft serve as a refueling station [70].

These types of problems frequently can be thought of as variants of the well known traveling salesmen problem (TSP). The TSP, simply stated, is the problem of finding the lowest cost route that travels through each node (or “city”) in a graph exactly once, starting and ending at the same node. We refer to the general problem of finding an optimal spacecraft path or paths through a number of targets as a “traveling satellite problem,” and several variants of these problems are explored in this section. Importantly, we also use the term "traveling satellite problem" to refer to scenarios different from the classic TSP. While optimal solutions to some traveling satellite problems can be calculated and verified for relatively small numbers of targets in a tractable amount of time, larger and more complex systems can require the use of techniques that do not guarantee a globally optimal solution. Added complexities found in variants of the TSP may include having multiple "salesmen" and having time varying costs to travel between "cities."

There are a significant number of previous studies that explore variants of the TSP for spacecraft path planning problems. [66, 62] have explored the use of an exhaustive tree search (small number of targets), ant colony optimization (ACO), and auctioning methods for the target selection and ordering process (with more typical low-thrust trajectory optimization or impulsive ΔV estimates used for transfer cost calculation). [71] also explored a low-thrust single active spacecraft multi-target scenario, where the path planning for a TSP variant was solved with a Physarum heuristic algorithm. [67, 64] have explored the use of a two-phase optimization approach to solve TSP variants with both a single and multiple active-spacecraft visiting the given targets. The first phase in their solution method creates a database of locally optimal target-target transfers across all allowable times, and the second phase then uses those component transfers to pose and solve an integer linear programming (ILP) problem that finds an optimal path through the targets. [63] use a "greedy" type heuristic method for target sequence selection. [65] discuss several genetic algorithm variants and evolutionary neurocontrol as options for the global optimization problem. [69] also use a heuristic method to address the sequencing problem, though in their case they apply it to the problem of servicing several spacecraft that are spaced throughout the same orbit.

Each of the preceeding works addresses different instances related to multi-target rendezvous scenarios, and thus each has differences in dynamics and propulsion models, target types, and constraints. Thus, comparing between studies is difficult, but a common theme is that the high dimensionality of the problems addressed necessitates the use of solution methods that cannot guarantee that a global optimum is found. Further, it can be difficult to asses how far from a global optimum a given solution is. Thus, in this work we address the problem of finding bounding optimal ΔV values to aid searches for global optima in more constrained problems. Further, our method of finding ΔV costs for optimal tours results in realizable tours if given the flexibility to choose certain spacecraft and target orbital parameters. This makes the method potentially useful for scenarios such as constellation design with consideration for ΔV optimal on-orbit servicing.

In this section we explore different formulations and solution methods for various traveling satellite problems and multiple traveling satellite problems. The simplified dynamics and transfers used here enable us to find exact bounding values in the single spacecraft case, and still give insight into higher dimensional multiple spacecraft cases as well. This provides an important bounding reference value that gives more context to the optimization results for more complex systems. We first explore the problem of selecting a sequence of target orbits such that the path through all the target orbits has a minimum ΔV cost. Different solution methods as well as some sample results are explored to give an idea of the structure of the solutions. We then explore multiple satellite tours, including the "multiple servicer" problem and the "depot placement" problem.

3.2.2 Dynamics and Transfers

In this work we explore fundamental minimum ΔV paths for spacecraft visiting multiple targets in a time-free sense. That is, a spacecraft "visit" or rendezvous with a target orbit occurs when the two entities have matching orbit semi-major axis (a), eccentricity (e), and inclination (i). This rendezvous is "time-free" in the sense that in Earth orbit with secular orbit perturbations due to the oblateness of the Earth (J_2), the perturbations can provide desired changes to orbit right ascension of the ascending node (RAAN) and argument of periapsis (ω) at no ΔV cost, though in many cases with the considerable cost of time. Further, given an infinite amount of time, and ignoring resonant objects, the two objects will at some point have the proper relative phasing at the time of transfer in order to match anomaly angles at rendezvous. However, if the initial and final RAAN, ω, and anomaly angles are able to be selected, then the transfers can be realized in finite time for the same cost as the time-free transfers (see Section 3.1.2 for more detail). If spacecraft transfers are further limited to being at most two-impulse, the optimal cost to transfer from one orbit to another can be expressed analytically as

$$J^*(q_0, Q_0, i_0, q_f, Q_f, i_f) = J^*(0, f) = \Delta V_{total} = \Delta V_1 + \Delta V_2 =$$
$$\sqrt{v_{1t}^2 + v_0^2 - 2v_{1t}v_0 \cos(\alpha \Delta i)} + \sqrt{v_f^2 + v_{2t}^2 - 2v_{2t}v_f \cos((1-\alpha)\Delta i)} \quad (3.33)$$

(same as Eq. (3.13)) with α, the fraction of inclination change performed in the first maneuver, being implicitly defined by the equation

$$F = \frac{\partial \Delta V_{total}}{\partial \alpha} = \frac{\Delta i v_0 v_{1t} \sin(\alpha^* \Delta i)}{\Delta V_1} - \frac{\Delta i v_f v_{2t} \sin((1-\alpha^*)\Delta i)}{\Delta V_2} = 0 \quad (3.34)$$

(same as Eq. (3.15)) such that it minimizes the cost in Eq. (3.33). This transfer is essentially an elliptical hohmann transfer that additionally uses a "dogleg" maneuver to distribute the total inclination change between both impulses. The variable $q = a(1-e)$ represents the periapsis radius, variable $Q = a(1+e)$ represents the apoapsis radius, the 0 subscript denotes the initial orbit parameters, and the f subscript denotes the final orbit parameters. The ΔV cost equation has been derived using a combination of an elliptical hohmann transfer and velocity triangles to account for inclination changes. For full details on evaluating Eq. (3.33), see Section 3.1.2. Note that these transfers are symmetric in that the cost to go from any one orbit to another is the same if the transfer is reversed.

Of course, actual space missions do not have the luxury of infinite time (and aren't necessarily in Earth orbit), but the advantage of the time-free result is that it gives bounding values for optimal transfer costs. Even in cases outside of Earth orbit where J_2 cannot theoretically give "free" orbit changes, the transfer cost is still a theoretical lower bound for the true, full cost that includes ΔV costs to match all orbit elements. A theoretical bounding minimum ΔV for a given spacecraft tour can be used to quickly evaluate if a potential new solution cannot possibly have a lower ΔV cost than a reference solution. This can enable a more judicious and efficient search for globally optimal solutions when the search space is quite large by quickly ruling out candidate solutions that are necessarily higher cost than a reference solution before spending additional time optimizing the candidate to understand its cost. The lower bound ΔV is also a point of comparison when applying inexact solution methods to find tours that do take into account factors such as phasing costs. Without a point of reference, there is no way to determine if a solution found with an inexact method is very close to globally optimal, or if it is quite far from globally optimal and potentially has the opportunity to be improved significantly. Further, exploring ΔV optimal spacecraft tours where rendezvous only depends on non-time varying orbit parameters (a, e, i) enables the analysis of how distributions of a, e, and i affect solutions, independent of other orbit elements. When addressing tour design for the full dynamics of a system, including time varying transfer costs, the results become more specific to the given problem being considered.

Alternatively, approximations for the ΔV cost to match all orbit elements can be used (e.g. as was done in [62]) in place of the computation of transfer ΔV cost with Eq. (3.33) for each of the scenarios posed within this work. Such an approach would have the advantage of finding optimal tours that also consider costs to match RAAN, ω, and phasing costs in addition to the costs to match a, e, and i, but the component transfers in each tour may or may not be truly realizable for the estimated ΔV cost of each transfer.

3.2.3 Single Satellite Tours

The single satellite traveling satellite problem here is formulated as follows. Given a single spacecraft with propulsive capabilities and a set of N target orbits (or N "cities") to rendezvous with, select the ordered sequence of target orbits to visit such that the total ΔV cost for the spacecraft to visit each target is minimized. This problem has two variants that are considered. In the first variant, referred to as a "closed loop" path, the spacecraft begins and ends at the same target orbit. Each target orbit will be visited exactly once, except for the starting point which is visited twice. In this formulation, any target orbit can equivalently selected as the start and end point for a closed loop path. In the second variant, referred to as an "open loop" path, the spacecraft begins at one of the target orbits and does not make any more transfers once it achieves rendezvous with the last target orbit in the tour. In the open loop problem, the start and endpoints are interchangeable because paths are equivalent when reversed (transfer costs are symmetric), but the start and end points are not arbitrary as in the case of the closed loop problem. The open loop problem can be solved in the same manner as the classic traveling salesman problem (which is closed loop) by adding an additional "city" that has 0 cost to travel to or from any other city [22]. The cities adjacent to this "virtual" city in the solution are the starting and ending cities.

In mathematical terms, an integer linear programming (ILP) formulation of the TSP is to minimize the cost

$$J = \sum_{(i,j)\in E} d_{ij}x_{ij} \tag{3.35}$$

subject to

$$\sum_{j\in V} x_{ij} = 2, \forall i \in V \tag{3.36}$$

$$x_{ij} \in \{0, 1\} \tag{3.37}$$

$$x_{ij} = 0, \quad i = j \tag{3.38}$$

$$\sum_{i,j\in S_V} x_{ij} \leq |S_V| - 1 \quad (\forall S_V \subset V; 2 \leq |S_V| \leq N - 2) \tag{3.39}$$

[72, 73, 74] where x_{ij} is a decision variable set to 1 if the path from city i to city j is used in the tour, d_{ij} is the cost from city i to city j, V is the set of all cities to visit, E is the set of all edges (paths) that connect two cities, N is the total number of cities to visit, and $|S_V|$ refers to the cardinality (number of elements) of the set S_V. Equation (3.36) ensures that only two paths connect a single city to other cities, and Eq. (3.39) is a constraint that prevents "subtours" from forming. A subtour is one where some proper subset S_V of the full set of cities V is connected with a tour but has no member that connects with any of the remaining cities in V but outside of S_V. That is, there is a closed loop tour that only connects a subset of all target orbits. Other equivalent mathematical statements of the problem also exist. Note that because a symmetric problem is being considered, the cost $d_{ij} = d_{ji}$, and tours "forward" or "backward" though a set of targets are equivalent. Thus, the constraint $x_{ij} = x_{ji}$ is implicitly enforced through the selection of the full set of edges E.

The closed loop path problem addresses scenarios such as a constellation servicing or fueling mission, where a single spacecraft travels from a supply depot to a full set of spacecraft in a constellation, and then returns to the supply depot. Supply depot placement is not considered for this type of tour, because under these assumptions the optimal placement of the depot should be coincident with the orbit of one of the targets (placing the supply depot at one of the target orbits adds 0 ΔV cost to the path). The open loop path problem addresses scenarios such as multi-target debris de-orbiting where an active satellite delivers "de-orbiting packages" to several debris objects.

3.2.3.1 Solution Methods

Brute Force Solution The brute force solution method is quite viable for this problem for relatively small N, especially given that the costs to transfer between target orbits are quickly calculable using an analytic expression. Because the transfers used here are time free, the cost to transfer from target i to target j is the same as the cost to transfer from target j to target i, and the problem is symmetric. Thus, to calculate all transfer costs between all targets, there are $N(N-1)/2$ different values to calculate. These values can be pre-computed and then stored in a lookup array. Then, each possible tour sequence can be enumerated and its cost can be computed. For closed loop paths, there are $(N-1)!/2$ different possible sequences, where 1 is subtracted from N to account for the fact that in a closed loop the "starting" target in the sequence is unimportant. There is a factor of two in the denominator because of the symmetry of the problem; reversing the order of a single sequence does not change the cost, but it does constitute a separate permutation. For open loop paths, the starting point does impact the cost, so the total number of possible sequences is $N!/2$ at most. For the brute force method, the cost of all possible sequences is calculated and lowest cost one is taken as the solution. This guarantees that a globally optimum solution is found, because all options are considered.

Genetic Algorithms Genetic algorithms (GAs) have been widely used to solve traveling salesman problems and other integer programming problems. [73] provides an excellent overview of genetic algorithms specifically with respect to traveling salesman problems, but the process is quickly summarized here. In short, genetic algorithms are a type of evolutionary algorithm that encode decision variables into "chromosomes," which are essentially lists of variables that are to be manipulated by the algorithm. A genetic algorithm begins by initializing a number N_p of chromosomes, where each chromosome represents a full solution to the problem and has a corresponding scalar "fitness" value that is to be optimized by the algorithm (analogous to an objective function). In a single "generation," "mutation" and "crossover" operators are applied to the chromosomes in the population to produce the next generation of N_p chromosomes. The crossover ratio Cr is the fraction of the next generation that is generated using the crossover operator, with roughly $1 - Cr$ being the fraction of the next generation that is generated using the mutation operator. A certain number of "elite" chromosomes are also included in the next generation. These "elite" chromosomes preserve the N_e best (with respect to fitness) chromosomes from the previous generation.

The time to run the genetic algorithms and their accuracies are largely a function of the user specified stopping criteria. The GA is considered "stalled" when the average change in the best value of the objective function over a given number of generations is below a given threshold ($10^{-5}\%$ here). The GA can also simply be stopped after a maximum number of generations have been generated. After the stopping criteria is reached, the resulting solution is the chromosome in the last generation with the best fitness.

The first step for crossover is the selection of two "parent" chromosomes, which is typically done using a process that selects chromosomes with better fitness values with a higher probability than other chromosomes. In this analysis tournament selection is used, which is not discussed by Potvin but is widely employed and has been described by [75] and others. Given two parents, a crossover operator combines some of the parameters ("genes") of the first parent chromosome with some of the genes of the second parent to create a "child" chromosome that represents a valid solution. A mutation is an operation that takes a single parent chromosome and modifies one or more of its genes to create a child chromosome that is inserted into the next generation.

Gene:	**1**	**5**	**2**	**4**	**6**	**3**
City ID:	1/A	5/E	2/B	4/D	6/F	2/B
Tour Order:	1	2	3	4	5	6

Table 3.6: Example path representation of the TSP in a chromosome encoding a path through 6 "cities"

There are a large number of different selection, mutation, and crossover operators that have been developed and investigated for use in genetic algorithms. How a given problem's decision variables are encoded into a chromosome greatly impacts the available operators that can be used. Suppose that there are N cities to visit with each city having an integer label from 1 to N. In our sample representations, we also include a corresponding letter with the city ID number to help distinguish between city ID and tour order numbers (A=1, B=2, etc.). The typical representation of a traveling salesman problem in a chromosome is to use a list of integers called the "path representation," seen in Table 3.6. The first city visited in this route is city 1, the second city visited is city 5, and so on. For a problem of N targets to visit, the chromosome has N genes. This is a very natural way to transcribe the problem, but precludes one from using more standard mutation and crossover algorithms due to the need to make child chromosomes valid. Any operators on a path type chromosome must ensure that children have all N targets listed in the child chromosome exactly once, which will not occur if standard genetic algorithms operators are used. [73] explores many different operators specific to the TSP that ensure that child chromosomes meet this criteria. In this work the "order crossover" and "scramble" mutation operator discussed by Potvin is used when using the path representation.

We also propose a different type of encoding of the genes that allows the use of standard genetic algorithm operators while still producing valid child chromosomes. This approach has the advantage of giving the user more options of well-studied mutation and crossover operators and allows the user to more easily encode the TSP genes into a single chromosome that might contain additional problem parameters to be optimized with standard genetic algorithm operators. In this

Gene:	**0.05**	**0.21**	**0.9**	**0.3**	**0.15**	**0.45**
City ID:	1/A	2/B	3/C	4/D	5/E	6/F
Tour Order: (dictated by gene relative magnitudes)	1	3	6	4	2	5
Path Representation:	1/A	5/E	2/B	4/D	6/F	3/C

Table 3.7: Example rank representation of the TSP in a chromosome encoding a path through 6 “cities”

representation, there are N genes where each gene is assigned a number between 0 and 1. Each gene in this case corresponds to a numbered target city (i.e. the first gene corresponds to city 1, the second gene corresponds to city 2, etc.). The relative magnitudes of the gene values dictate the order in which each city is visited. In the example shown in Table 3.7 we encode the same sequence as in Table 3.6. City 1 is visited first because its gene has the smallest value out of all genes. City 5 is visited second because its gene has the second smallest value, and so on. This representation is less natural to understand, but is simple to implement in code and has the aforementioned benefits of allowing the user to implement standard genetic algorithm crossover and mutation operators. In this work, a uniform mutation and 2-point crossover are used with the rank representation.

Lin-Kernighan-Helsgaun (LKH3) The Lin-Kernighan-Helsgaun (LKH3) solver [76] uses an approximate heuristic method to solve the TSP. It relies on the highly successful Lin-Kernighan (LK) algorithm, which is a method of performing a local search for optimal solutions. Beginning with a feasible solution to the TSP, the LK algorithm works by removing a set of “edges” (connections between cities) and then re-connecting cities such that the result produces a valid solution and reduces the tour cost. The term “k-opt” is typically used when describing the number of “edges” that are removed and replaced in a given iteration. Once a local minimum where no further cost reductions are possible with a “k-opt” move, the optimization can be terminated. Depending on the implementation, a randomized set of initial feasible solutions can be locally optimized with the LK algorithm to provide a better chance of finding the true optimal solution. More detail on the LK algorithm can be found in work by [77].

Concorde Solver The concorde solver[2] is widely regarded to be the current state-of-the-art in methods for finding an exact solution to the TSP [78, 79]. A full description of the method is outside the scope of this section, but in short the concorde solver first poses the problem as a relaxed linear programming (LP) problem, and in the process ignores constraints that require decision variables to be integers. It then uses branch-and-bound, and branch-and-cut, which uses a concept called “cutting planes” to iteratively fix instances where the LP solver gives non-integer solutions to decision variables that must be integers. Further, it uses the Lin-Kernighan heuristic as a starting point in its solution method. Full information can be found in [22]. Importantly, this solver should always produce the globally optimal solution to a given TSP, because it uses LP duality to prove that the solutions it produces are optimal. Source code for the Concorde is available online[1] for academic use and allows the user to generate an executable that can solve “tsplib” files, which are a standardized format of defining TSP problems.

Benchmarks Sample run times for the different solution methods have been included in Fig. 3.20 to give rough estimates of accuracy and the time to compute solutions for a single traveling satellite. These values should be taken as **order of magnitude estimates only**, because a number of different factors will affect computation time and accuracy (processor type, implementation details, etc.). Further, while problems with a larger number of targets will generally take longer to solve, the exact initial conditions of each problem also greatly impacts the problem difficulty. For this reason the computation times do not strictly trend upward for each method as N increases. The various tuning parameters that are available for each of the different solution methods have not specifically optimized. A great deal of effort could be spent on finding specific parameters that work well for astrodynamics problems, but this work is better suited for scenarios where there is a specific mission application in mind. In the absence of any single real-world application in mind, we instead aim to provide a starting point for future investigations of varying types.

Fig. 3.20 shows the variation in mean computation time for each method as the number of target orbits grows. For a given number N of target orbits, the same set of randomly generated

[2] http://www.math.uwaterloo.ca/tsp/concorde.html

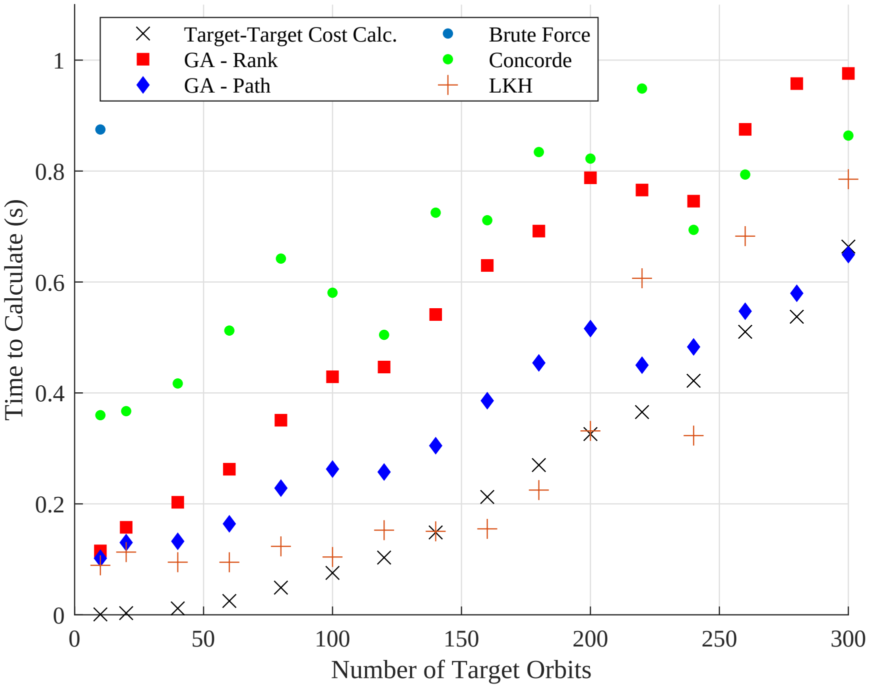


Figure 3.20: Mean computation time for each method to find a solution as the number of target orbits increases.

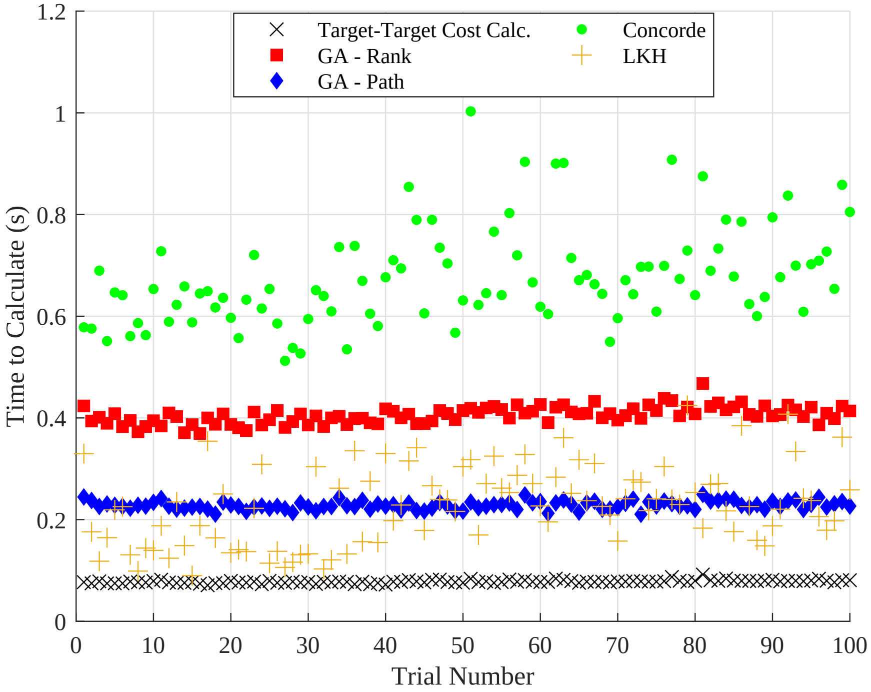


Figure 3.21: Mean computation time for each method to find a solution where each trial has a different randomized set of 100 target orbits.

orbits is solved 100 times and averaged to give an idea of the mean computation time for a given set of targets. The mean computation time for the brute force method is only included for $N = 10$ due to the prohibitively long computation times for $N \geq 20$. In Fig. 3.21 we instead hold the number of target orbits constant at 100, and display how the computation time varies for different randomly initialized sets of $N = 100$ target orbits. The sets are randomly initialized using a uniform distribution, and for each trial the mean computation time for a total of 100 repeated calculations for the same set of target orbits is plotted.

The time to compute all inter-target orbit transfer costs is also included in Figs. 3.20-3.21 with "x" markers. These costs must be computed prior to using each of the solution methods, but the computation time value plotted for each method does not include the transfer cost computation time.

These sample solutions were produced with code written in the Julia programming language on an Intel Core i7-9700k desktop processor. The genetic algorithm methods were directly written and executed in Julia, while the Concorde and LKH solvers were interfaced with through available pre-compiled binaries using text file inputs. Additional speed would likely be gained by directly interfacing with the available code for both programs, so the time to write the text file to disk was not included in the above benchmarks. The costs written to text files were also necessarily converted into integers to be compatible with the pre-compiled Concorde solver, so results were only optimal to within a conversion tolerance. In this case, the costs were multiplied by 10^6 and then rounded to the nearest integer.

As expected, the Concorde solver always produced the exact minimum solution found with the brute force method, but solved the problem much more quickly. The LKH3 solver also happened to always find the globally minimal solution in every case in Fig. 3.20, though for the results in Fig. 3.21 three cases out of the 100 trials found sub-optimal results (with $< 0.1\%$ over-estimation of the optimal cost). The LKH method has a clear time advantage over Concorde, and is likely the ideal method for computing the optimal cost unless an iron-clad optimum is needed.

The genetic algorithm solutions are shown here simply for reference, but far underperform the Concorde and LKH solvers in terms of accuracy. These un-tuned solution methods reach average percent error rates of well over 1,000% as the number of targets exceeds 150. The rank representation GA has a slightly longer calculation time than the path representation GA, but consequently has slightly lower average error rates. There is essentially a trade between number of generations (computation time) and how close a GA will get to the global optimum, so these results are in line with what is expected from a GA. For both of these GAs, a maximum number of 1400 generations were allowed, with 600 generations being the "stall limit" criteria to stop the GA. Accuracy improvements may be found by implementing a "k-opt" mutation operator as suggested by [73], which is similar to the method of the LKH solver.

3.2.3.2 Sample Results

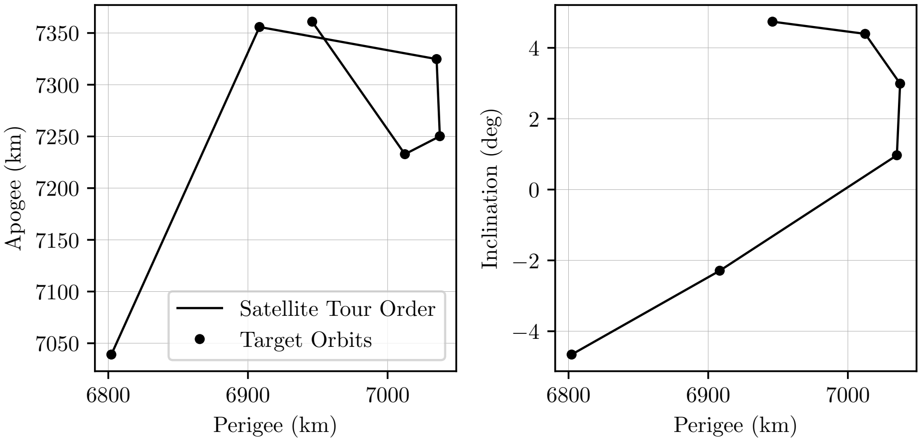


Figure 3.22: Open loop single spacecraft tours, two views of the same set of target orbits

Exact solutions for sample tours through 6 target orbits are shown in Figs. 3.22 and 3.23 for open and closed loop paths respectively. Each figure shows two different views of the same target orbits that are placed in 3-dimensional $a - e - i$ space, with a line connecting them in the ΔV optimal order. There are several different aspects of the solution to note. First, the open loop target sequence is different from the closed loop target sequence, as seen in Figs. 3.22 and 3.23. That is, the closed loop solution is not simply to connect the starting and ending target orbits of

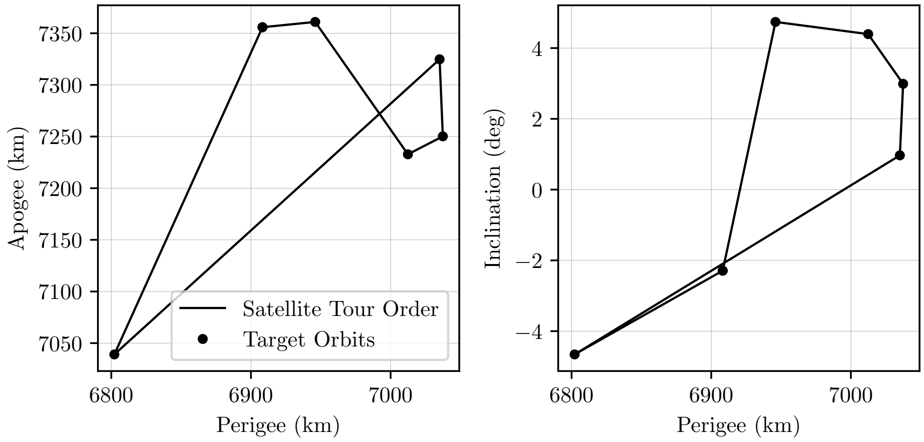


Figure 3.23: Closed loop single spacecraft tours, two views of the same set of target orbits

the open loop tour. Additionally, in the open loop case, the path almost always follows from one extreme of the inclination distribution to the other extreme. For example, in Fig. 3.22, the tour starts at the highest inclination target, and each subsequent target is selected by choosing the next nearest target in inclination space. This matches the perhaps intuitive expectation that minimizing inclination changes should be a higher priority than minimizing periapsis and apoapsis changes in a tour.

This inclination heuristic for the open loop case has been found to hold for large numbers of spacecraft, but it breaks down when target orbits are within roughly 0.1 deg of one another in the simulations we have run. This can be seen in Fig. 3.24, where in addition to the targets from Fig. 3.22 an additional target has been added with $q = 7000$ km, $Q = 7300$ km, and $i = 3.01$ degrees. The additional target's nearest neighbor has $q = 7037.41$ km, $Q = 7250.2$ km, and $i = 2.99$ degrees. If the heuristic were strictly followed, then the additional target would come before the nearest neighbor when starting from the highest inclination target. But in Fig. 3.24 that is not the case, and the heuristic sequencing method would result in a total cost increase of about 0.8 m/s (about 0.06% of the total cost) as compared to the optimal solution. The penalty for using the heuristic ordering will grow as the heuristic-breaking target moves farther from its neighboring targets in the optimal order. That is, as the heuristic causes the tour to go “farther out of the way,” the cost penalty grows. The size of the inclination difference threshold that may break the heuristic (found to be roughly 0.1 deg here) will depend on the periapsis and apoapsis changes that must also be made for each transfer.

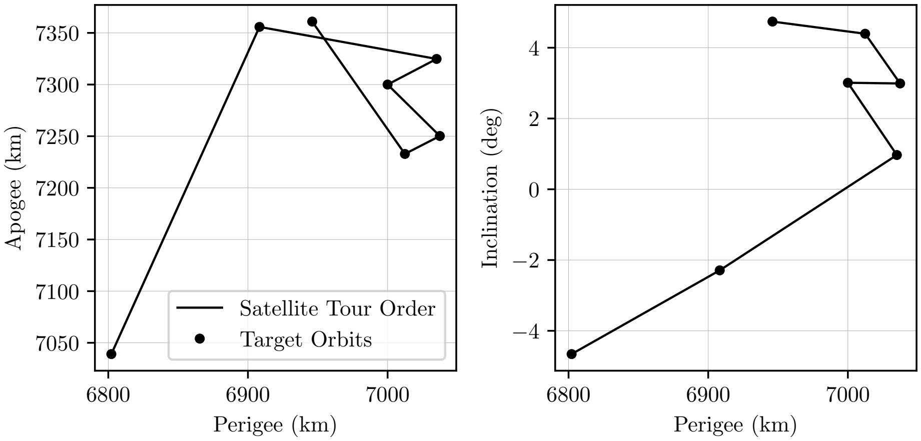


Figure 3.24: Open loop single spacecraft tours, two views of the same set of target orbits

3.2.4 Multiple Satellite Tours

Several different multiple traveling satellite problems are considered here. In the first, a set of N target orbits, which must each be visited by exactly one of up to m total active spacecraft, is considered. The total ΔV expended by all m spacecraft is to be minimized. This is an extension of the single satellite tour, and similarly has variants where each active spacecraft has an either open or closed loop path through the targets. The problem extends similarly to real world scenarios as the single traveling satellite problems. The closed loop case applies to scenarios such as having multiple supply depots that each have an active servicing spacecraft and set of target objects to service. The open loop case applies to scenarios such as having multiple spacecraft selecting and visiting multiple debris objects for de-orbit package delivery. Thus, both the open and closed loop variants of this problem are referred to here as the “multiple servicer” problem (MSP).

A scenario where there are N active spacecraft that all must rendezvous with exactly one of m target orbits is also considered here. The objective is to place the m target orbits such that the total ΔV cost for each of the N spacecraft to rendezvous with its nearest ("nearest" meaning lowest ΔV cost) target is minimized. This problem can be thought of as a spacecraft constellation where each spacecraft in the constellation regularly visits a supply depot for fuel or other supplies. Alternatively, there could be a single active servicing spacecraft based at the supply depot, which on average visits all target orbits equally and thus the total ΔV to transfer from each target to the supply depot should be minimized. This scenario assumes that the single active spacecraft re-visists the supply depot before and after visiting each target. Whereas the previous problems described here are integer programming problems, this scenario has continuous variables to optimize. That is, there are $3m$ orbit elements to optimize (a, e, and i for each depot), each of which is a continuous variable. Here, this problem is referred to as the "depot placement" problem. While in this statement of the problem each spacecraft simply uses its nearest target orbit for rendezvous, alternative formulations of the problem that might, for example, explicitly assign each of the N spacecraft to one of the m targets, are not explored here.

3.2.4.1 Multiple Servicer Problem

The multiple servicer problem as stated here is related to the multiple depot, multiple traveling salesman problem, which is one of the numerous variants of the TSP that have been considered by the mathematics and operations research communities. However, the multiple depot, multiple TSP typically has salesmen departing from and arriving at a set of depot locations that are fixed and known a priori, whereas in the MSP the start and endpoints of a tour are not fixed. The typical multiple traveling salesman problem (mTSP) is similarly related, but additionally constrains that all salesmen start and end at the same city. In the MSP, the multiple traveling satellites ("servicers") specifically start at different orbits that are not predetermined.

A mathematical statement of the closed loop MSP is to minimize the linear cost

$$J = \sum_{k \in M} \sum_{(i,j) \in E} d_{ij} x_{ijk} \tag{3.40}$$

where x_{ijk} is now a binary variable indicating if servicer k will depart city i and arrive at city

j in its tour, M is the set of all m servicers, and k is used to index the servicers. In this formulation the incoming and outgoing edges are considered to be separate, so index i specifically corresponds to the departure city and index j corresponds to the arrival city. There are also several constraints that apply to this problem. The first three constraints (Eqs. (3.41 - 3.43)) simply constrain each city to have one incoming and one outgoing edge, and constrains the decision variables to be binary.

$$\sum_{k\in M}\sum_{j\in V} x_{ijk} = 1, \quad \forall i \in V \tag{3.41}$$

$$\sum_{k\in M}\sum_{i\in V} x_{ijk} = 1, \quad \forall j \in V \tag{3.42}$$

$$x_{ijk} \in \{0,1\} \tag{3.43}$$

The next constraint in Eq. (3.44) constrains both the arrival and departure edges for a single city to belong to a tour with the same servicer k.

$$\frac{1}{2}\left(\sum_{j=1}^{N} x_{\ell jk} + \sum_{i=1}^{N} x_{i\ell k}\right) \in \{0,1\}, \quad (\forall \ell \in V \text{ and } \forall k \in M) \tag{3.44}$$

The final closed loop MSP constraints in Eqs. (3.45-3.47) prevent subtours from forming in a single servicer's set of assigned cities.

$$x_{ijk} = 0, \quad (i = j, |V_k| > 1, \forall k \in M) \tag{3.45}$$

$$x_{ijk} + x_{jik} \in \{0,1\}, \quad (|V_k| > 2, \forall k \in M) \tag{3.46}$$

$$\sum_{i,j\in S_{V_k}} x_{ijk} \leq |S_{V_k}| - 1, \quad (S_{V_k} \subset V_k; 3 \leq |S_{V_k}| \leq |V_k| - 2, \forall k \in M) \tag{3.47}$$

V_k is the set of all cities assigned to each servicer k and S_{V_k} is a proper subset of V_K. In this formulation, while symmetric costs $d_{ij} = d_{ji}$ are still being used, the edge decision variables $x \in E$ are no longer taken to be symmetric (i.e. $x_{ijk} \neq x_{jik}$). Thus, solutions to the problem will have a specific direction through the targets, though in practice the cost of doing the same tour in reverse will be the same because of the symmetric path costs. This change was made to the MSP to accommodate the assignment of a single servicer to a single target, which is now encoded by having $x_{ijk} = 1$ for $i = j$.

Equation (3.41) enforces that each city must have an edge departing itself, and Eq. (3.42) enforces that each city must have an edge arriving at itself. Equation (3.44) enforces that a single city must have either no connected edges included in a single servicer's tour, or exactly two connected edges included in a single servicer's tour. This prevents a city from being connected to one servicer on one edge and another servicer with a different edge. Together with Eqs. (3.41-3.42), this ensures that each city is included in exactly one of the m possible tours.

The standard subtour elimination constraint, Eq. (3.39) from the TSP, must also be modified to suit the scenario of the MSP, and is now represented in Eqs. (3.45-3.47). Equation (3.45) prevents an edge that leaves from and arrives at the same city, which was never allowable in the TSP but is potentially allowable here if only one target is assigned to a single servicer. This equation ensures that if the number of targets assigned to a single servicer is more than 1, then there can be no complete tour through a single city. Equation (3.46) prevents a subtour of just two cities from forming, which was previously impossible in the TSP due to the constraints and selection of the set of edges. It states that if an edge from city i to city j is selected for a tour, then the edge from city j to city i cannot be selected. This is not allowed when more than two cities are assigned to a single servicer. Equation (3.47) is then the similar subtour elimination constraint as found in the TSP to account for cases where $|V_k| \geq 3$. While the MSP subtour elimination constraint is similar to the TSP subtour elimination constraint, it is not as easily accounted for because there is no a priori knowledge of V_k or $|V_k|$. This is discussed in more detail in Sec. 3.2.4.1 “Integer Linear Programming Approach” where the use of an ILP problem MSP solution method similar to that used for the TSP is explored.

For the open loop MSP, m cities must be added to the set of N targets V, each with a path cost of $d = 0$ to all other cities. Assuming that these new cities correspond to the cities $i = 1, ..., m$, the additional open loop MSP constraints are

$$\sum_{j=1}^{N} x_{ijk} = 1, \quad (i = k = 1, ..., m) \tag{3.48}$$

$$\sum_{i=1}^{N} x_{ijk} = 1, \quad (j = k = 1, ..., m) \tag{3.49}$$

Equation (3.48) constrains each servicer's tour to have a departing edge from the corresponding zero cost city, and Eq. (3.49) constrains each servicer's tour to have an arriving edge to the corresponding zero cost city. Together with the MSP subtour constraints, this prevents multiple zero cost cities from being linked together in the same tour, and thus each tour will have a single zero cost city that indicates the starting and ending point.

Brute Force Method Brute force solutions to the multiple servicer can actually be relatively fast for small N and m. However, as N and m grow, the number of possible solutions grows exponentially. The number of ways of assigning the N targets to one of m servicers is $h = m^N$, because for each target N there are m possible assignments. For each of the h distributions, there are m sub-problems where the optimal sequence of the assigned targets must be found for each servicer. This sub-problem is equivalent to the single satellite tour discussed earlier, and thus could be solved exactly with the Concorde solver or quickly with the LKH method. However, intelligent consideration of the symmetries present in the problem can reduce the number of calculations for an overall brute force method. For example, the same set of targets will be assigned to multiple servicers, but the optimal tour through a given set of targets must only be calculated once. Optimal paths and path costs can be stored for given sets of target spacecraft. Further code optimizations are also possible, though are non-trivial to implement.

Figures 3.25-3.26 show exact solutions to a sample MSP problem with 2 traveling satellites and 11 target orbits. Target orbits are shown with different colored points depending on which spacecraft it is assigned to, and the two spacecraft paths through their respective targets are shown with lines. In Fig. 3.25 the inclination heuristic again holds for the most part, except for two of the targets which have similar inclinations. There is also a wide discrepancy in the total number of targets assigned to each spacecraft, which is a common feature seen when exploring solutions

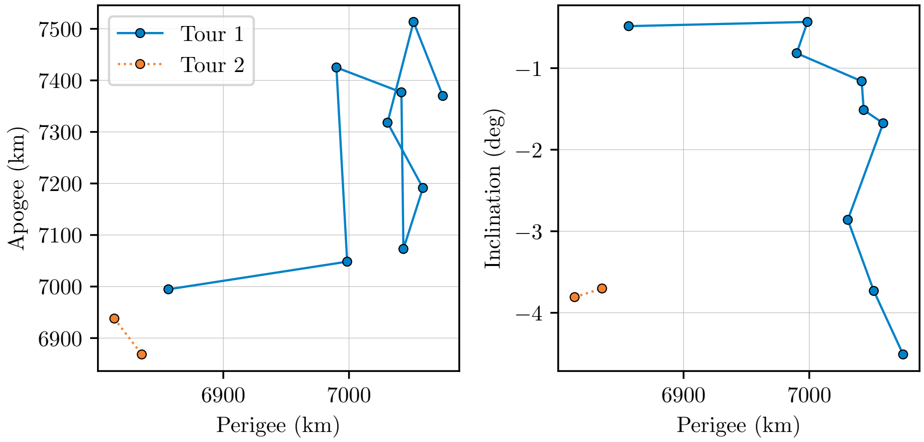


Figure 3.25: Exact solution to open loop 2 spacecraft MSP, two views of the same set of target orbits

to different scenarios. Some solutions even have a single target assigned a single spacecraft. This is not necessarily indicator that a target orbit is an "outlier," because even when all targets are "closely" grouped, assigning a single spacecraft to a single target is essentially a free way to reduce cost because the launch cost of having additional spacecraft is not accounted for here. However, solutions where only one or a few targets are assigned to a single spacecraft may perhaps prompt a redesign or re-selection of target orbits to enable more efficient transfers. For a more even distribution of the targets between the different traveling satellites the number of targets could be fixed, or a different objective function could be used.

Note that when the tours are changed to closed loop, the assignment of targets to the spacecraft is quite different, as seen in Fig. 3.26. The open and closed loop cases are not necessarily always so different, but it is interesting to note that the total ΔV optimal distribution of targets can change significantly with the change in the problem statement.

Genetic Algorithm The flexibility of the previously discussed rank representation of the single TSP problem makes it relatively easy to extend that transcription to the multiple servicer problem. Table 3.8 shows a sample representation of the MSP as a chromosome. The ordering of the cities is done exactly as in the single TSP problem, but instead of having N genes for N targets,

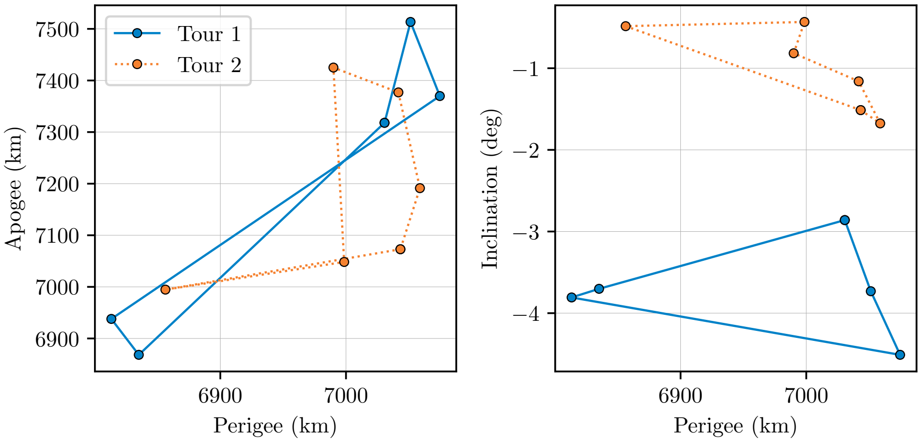


Figure 3.26: Exact solution to closed loop 2 spacecraft MSP, two views of the same set of target orbits

an additional gene is added to encode the number of targets assigned to each traveling satellite. Gene $N + 1$ is also a value between 0 and 1, but is used to determine the "split point" to divide the targets between the two traveling satellites. Fig. 3.27 shows how the last gene in Table 3.8 is used to determine how the targets are assigned to the two satellites. In this case, the first satellite is assigned the first 4 targets of the equivalent path representation (1/A, 5/E, 2/B, 4/D), and the second satellite is assigned the last two targets of the equivalent path representation (6/F, 3/C). The first satellite is assigned the first 4 targets because the value of the last gene, 0.54, lies in the 4th "bin" of Fig. 3.27. Bin widths are determined by dividing 1 by the number of total target cities N, and start/end points of each bin are defined by creating N bins between 0 and 1. The first four values of the equivalent path representation must be used as opposed to using the first four values of the rank representation gene so that each satellite has the possibility of being assigned any of the target orbits.

This transcription allows the GA to select both the assignment of each target to a specific traveling satellite, as well as the order in which each satellite visits its assigned targets. Alternatively, this transcription can be used only to assign each target to a specific traveling satellite, and instead use the Concorde or LKH methods to select the proper tour order (i.e. the ΔV cost for each

Gene:	**0.05**	**0.21**	**0.9**	**0.3**	**0.15**	**0.45**	**0.54**
City Number:	1/A	2/B	3/C	4/D	5/E	6/F	-
Tour Order: (dictated by gene relative magnitudes)	1	3	6	4	2	5	-
Path Representation:	1/A	5/E	2/B	4/D	6/F	3/C	-

Table 3.8: Example of encoding a path for two satellites through 6 “cities” in the multiple servicer problem

tour is given by using the Concorde or LKH to solve for the optimal target order). This method was found to be ideal for solving larger scale problems. Another possibility is using the Concorde or LKH method as a mutation operator so that the “local” optimization is not necessarily always applied.

A sample inexact solution produced by the MSP genetic algorithm is shown in Fig. 3.28. This solution is 5% more costly than the true solution, and was found by allowing the GA to run for the same amount of time as the brute force 2-spacecraft 11-target brute force calculation. However, due to their stochastic nature, the error in the GA solutions from run to run vary widely even when the GA is given this amount of time. The GA given this run time has by chance found the exact minimum solution, but has also found a solution 30% worse than the true solution as its best guess on different runs. A good strategy for finding good solutions with the GA is to run it several times, where in each run the initial population is randomly generated. This tends to give better results than running a GA once for many generations (if both are given roughly the same amount of total time to run). The GA gives good results for moderately sized problems, but will become less accurate as the problem size increases.

While the GA solution method is of relatively little use in this case where a brute force solution is relatively quick to compute ($\sim$ 30 sec), it can handle cases that would take prohibitively long to compute with a brute force method. In cases with large N solved with the GA, there will be a trade-off between computation time and accuracy, with perhaps an unknown accuracy at the

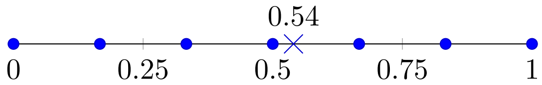


Figure 3.27: Visualization of translating the final gene to a cut point. 0.54 falls in the fourth bin demarcated by the blue points, so the first traveling satellite is assigned the first four cities of the path representation.

end of the GA calculation if no brute force calculation is performed. Better tuning of the GA for problems that are easier to verify with brute force methods may improve the performance of the GA for problems that are not easily verifiable with a brute force approach. An example closed loop larger scale scenario is shown in Fig. 3.29 where the GA presented here is used only for target assignment, with target ordering performed with the LKH solver during each GA cost function evaluation and with the Concorde solver used to refine the final solution. The GA commonly finds solutions where single targets are assigned to two of the three servicers, and the rest of the targets are assigned to the third servicer. This is perhaps because this type of solution allows the "free" removal of four edges, but in the next paragraph ("Integer Linear Programming Approach") a better solution will be found where there are no single target tours. Still, the GA finds a solution only 3% more costly in ΔV than the method in Sec. 3.2.4.1 "Integer Linear Programming Approach," but at a significantly longer run time.

Integer Linear Programming Approach First consider the closed loop MSP as formulated in Eqs. (3.40-3.47), which is in a similar form to the standard TSP problem posed in Eqs. (3.35-3.39). While similar solution methods can be used to solve the MSP, constraints in Eqs. (3.44) and (3.45-3.47) constitute significant departures from the TSP. Equation (3.44) can be handled by introducing Nm new binary decision variables $b_{\ell,k}$. Then, there are Nm new constraints where

$$\frac{1}{2}\left(\sum_{j=1}^{N} x_{\ell jk} + \sum_{i=1}^{N} x_{i\ell k}\right) = b_{\ell,k}, \quad \forall \ell \in V \text{ and } \forall k \in M \tag{3.50}$$

$$b_{\ell,k} \in \{0,1\} \tag{3.51}$$

These new decision variables have no associated cost and are simply added as an simple method of translating the constraint in Eq. (3.44) into a form easily handled by standard ILP

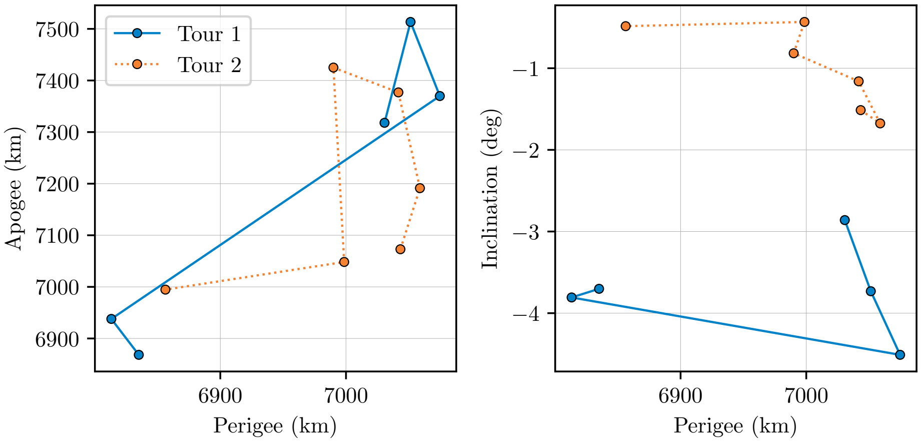


Figure 3.28: Open loop 2 spacecraft tours solved with a GA, two views of the same set of target orbits. This path is 5% higher cost than the true minimum cost path.

solvers. Equation (3.46) could also be implemented in a similar way if $|V_k|$ were known.

Equations (3.45-3.47) are much more difficult to account for, because as formulated it requires a priori knowledge of V_k or $|V_k|$ in order to implement the constraint in an ILP solver. This is not possible, because indeed one of the desired outputs of the problem solution is the knowledge of V_k and $|V_k|$.

Closed Loop MSP

An alternative method of solving the closed loop MSP is to leverage an ILP solution method (Algorithm 2) used to solve the TSP. This method of solving the TSP is one where a modified TSP posed with Eqs. (3.35 - 3.38) (ignoring the subtour elimination constraint) is first solved with standard ILP techniques with a standard ILP solver. The resulting solution is then checked to see if any subtours exist. If one or more subtours exist, the smallest subtour is used to implement the constraint in Eq. (3.39). That is, instead of implementing Eq. (3.39) for all valid subsets S_V, Eq. (3.39) is implemented for the smallest subset S_V that violates Eq. (3.39). The new problem is then again solved with the addition of the new constraint, and the process is repeated until there is no violation of the constraint in Eq. (3.39). This is a “lazy constraint” that allows the problem to be solved without enumerating all possible subsets S_V.

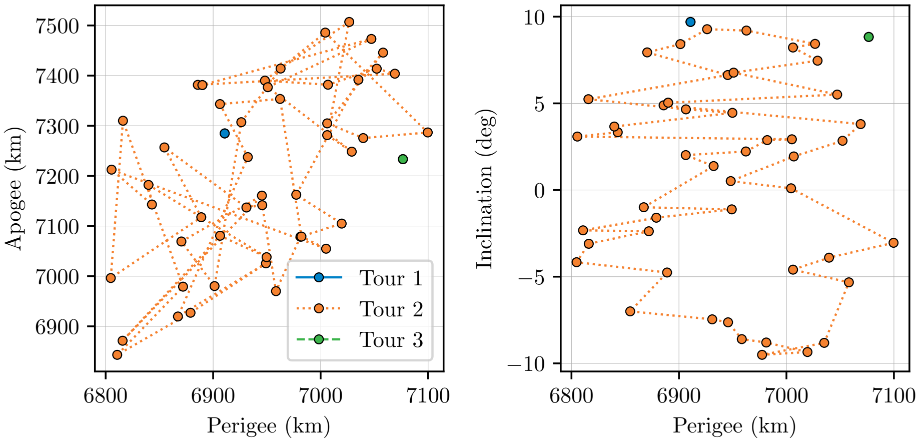


Figure 3.29: Large scale ($N = 50$, $m = 3$) MSP problem solved with a genetic algorithm. Total $\Delta V = 5.656$ km/s, total run time was about 3 hours

```
Algorithm 2: TSP Solution Method with ILP
  Result: Optimal set of edges to form a TSP tour
  solve modified TSP (Eqs. (3.35 - 3.38)) with ILP solver;
  while Eq. (3.39) violated in solution do
      add Eq. (3.39) as a constraint to the problem for the smallest violating subset $S_V$ ;
      solve problem with added constraint ;
  end
```

This method can be simply modified to solve the MSP, as shown in Algorithm 3. First, note that solving the modified TSP with no subtour elimination constraint is essentially equivalent to solving the MSP but without a limit on the number of servicers. Each subtour corresponds to a tour for a single servicer. This can be interesting from the standpoint of understanding the maximum number of servicers that minimizes path costs, but for certain scenarios this can be an unrealistically large number. Instead, to bound the number of servicers (i.e. bound the number of subtours), Eq. (3.39) is implemented in a similar manner to Algorithm 2 except it is no longer implemented when the total number of subtours is less than or equal to the desired maximum number of servicers m.

Algorithm 3: MSP Closed Loop Solution Method with ILP

```
Result: Optimal set of edges to form closed loop MSP tours
solve modified TSP (Eqs. (3.35 - 3.38)) with ILP solver;
while number of subtours in solution $> m$ do
    add Eq. (3.39) as a constraint to the problem for the smallest $S_V$ that contains a
     subtour ;
    solve problem with added constraint ;
end
```

This method ignores the addition of the k indices in the MSP formulation in Eqs. (3.40 - 3.47). As a consequence of using a modified TSP solution method from the formulation posed in Eqs. (3.35 - 3.38), this method now constrains the smallest number of targets assigned to a servicer to be 3. Further, this method is in effect a heuristic method of solving the MSP, because adding constraints to remove the smallest subtour may cause the removal of a subtour that might actually be a component of the truly optimal solution. However, the method in Algorithm 3 has matched the brute force results for almost all relatively small-scale test cases that were compared, though there can be no guarantee that this always holds. Thus, the Algorithm 3 method is the best way to solve the closed loop MSP when servicers are not desired to have 2 or fewer assigned targets The only cases that were not found to match the brute force results where those in which a single servicer was assigned less than 3 target orbits. In this study the Gurobi solver [80] through its Python interface was used to solve the ILP, and run times usually were less than 1 second to solve the MSP.

Also note that the heuristic of removing the smallest subtour speaks to the goal of having some degree of balance in the number of targets assigned to each servicer. While such a goal is not explicitly considered here, it is very much of interest in scenarios where each servicer is to have roughly equal capabilities. While from a path planning perspective it may be optimal to assign a disproportionately large set of targets to a single servicer, it may overall be too expensive to equip a single servicer with the capability to service a large number of targets. Accounting for such objectives can be highly case specific, and is considered out of scope for this study.

The same large scale example shown in Fig. 3.29 and solved with a GA is also shown solved with Algorithm 3 in Fig. 3.30. This solution not only has a lower total path cost than the GA solution, it also qualitatively looks closer to what one might expect to be the optimal solution. Further, another grouping by inclination can be seen, again speaking to the usefulness of an inclination-based target grouping heuristic.

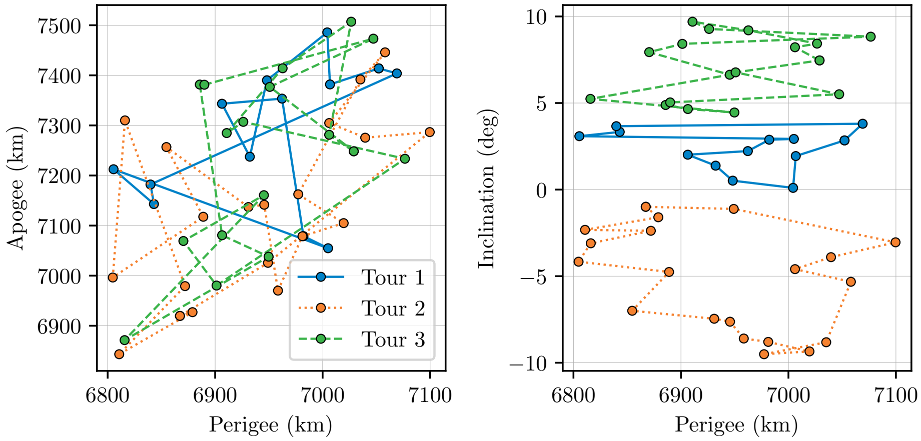


Figure 3.30: Large scale ($N = 50$, $m = 3$) MSP problem solved with Algorithm 3. Total $\Delta V = 5.487$ km/s, total run time was about 1 second

Open Loop MSP

In the open loop case, the results of the mTSP can be leveraged along with the idea of adding zero cost cities to a closed loop TSP in order to solve an open loop TSP. Recall that the mTSP is a scenario in which multiple traveling salesman are located at a single depot, and closed loop tours for each salesman must be found such that each target city is visited exactly once by a single salesman. The mTSP can be stated as the problem of minimizing the cost

$$J = \sum_{(i,j)\in E} d_{ij}x_{ij} \tag{3.52}$$

subject to

$$\sum_{j \in V} x_{ij} = 2m, \quad i = 1 \tag{3.53}$$

$$\sum_{j \in V} x_{ij} = 2, \quad i = 2, 3, ..., N \tag{3.54}$$

$$x_{ij} \in \{0, 1\} \tag{3.55}$$

$$x_{ij} = 0, \quad i = j \tag{3.56}$$

$$\sum_{i,j \in S_V} x_{ij} \leq |S_V| - 1, \quad (\forall S_V \subseteq V \setminus \{1\}, \ S_V \neq \emptyset) \tag{3.57}$$

[81]. Here the same set of edges E as in the TSP are used ($x_{ijk} = x_{jik}$), so only one unique edge exists between any two cities. Thus, Eq. (3.53) enforces that m servicers both leave from and arrive back at the depot "city" which has an index of 1. Equation (3.54) enforces that each non-depot city has two connected edges, and Eq. (3.57) is the subtour elimination constraint for the mTSP. This constraint enforces that all subtours not containing the depot city must be eliminated. S_V is no longer a proper subset because it is now a subset of V not including the depot city 1 ($V \setminus \{1\}$ is the set V not including the member $\{1\}$). Together with Eq. (3.53), this enforces that there will be exactly m subtours, which each correspond to a single salesman.

To transform the mTSP into solving the open loop MSP, the single depot city can simply be assigned a cost of 0 to transfer to any other city. That is,

$$d_{1j} = 0, \quad \forall j \in V \tag{3.58}$$

This, along with Eq. (3.53), enforces that each of the subtours has a start and end point at a zero cost city, mirroring the TSP transformation into an open loop TSP. This formulation similarly has the limitation that the smallest tours will include at least 3 cities, though the starting depot is no longer a "real" city, so the smallest tour length is now 2 cities. Further, this formulation also enforces that all m servicers be used instead of allowing that at most m servicers be used. This is likely expected to optimal because there is no additional cost associated solely with adding servicers (there is only the cost associated with edges included in tours). However, the use of all servicers may be always optimal with respect to total tour costs only when tours of a single target are allowable. This is because if a single target tour is allowable, then adding more servicers can be done at zero path cost if they are only assigned single target tours. Single target tours here would only use edges to and from the depot city, both of which would have 0 cost.

The solution method implemented here, shown in Algorithm 4 is the exact algorithm referred to as the "straight algorithm" by [81]. This method is very similar to Algorithm 3, where the problem is first solved without any subtour elimination constraints, and then constraints are added for subsets that violate Eq. (3.57) to avoid having to enumerate all subsets of V. In this case, the iterative addition of constraints is not a heuristic, because the subtour elimination constraint in Eq. (3.57) only depends on the known size of the set that violates the constraint. See [81] for a full exploration of the mTSP and the various possible solution methods that may be faster or scale to larger problems more effectively, but for the cases considered here the straight algorithm was sufficient.

Algorithm 4: MSP Open Loop Solution Method with ILP

```
Result: Optimal set of edges to form open loop MSP tours
solve modified TSP (Eqs. (3.52 - 3.56)) with ILP solver;
while Eq. (3.57) violated do
    add Eq. (3.57) as a constraint to the problem for all $S_V$ that violate Eq. (3.57) ;
    solve problem with added constraint ;
end
```

Algorithm 4 was implemented with the Gurobi solver and Python interface in a similar manner to Algorithm 3. Results from Algorithm 4 have exactly matched brute force open loop MSP results in several test cases, and the method can also quickly solve larger problems. Figure 3.31 shows an open loop solution to the large scale example target set solved for the closed loop solutions in Figs. 3.29 and 3.30 with the GA and Algorithm 3 respectively.

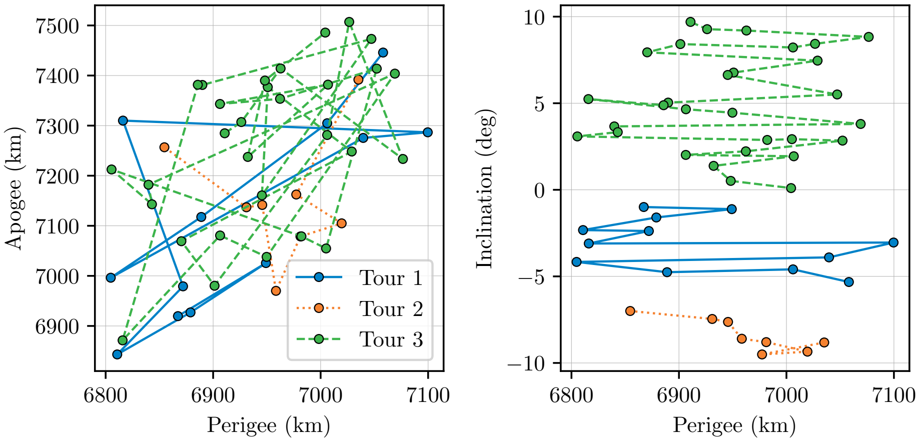


Figure 3.31: Large scale ($N = 50$, $m = 3$) open loop MSP problem solved with Algorithm 4. Total $\Delta V = 3.968$ km/s, total run time was about 1.1 seconds

3.2.4.2 Depot Placement Problem

This problem is essentially the extension of the rendezvous problem found in previous work [17] from $m = 1$ target rendezvous orbits to $m \geq 1$ rendezvous rendezvous orbits.

In this case, for N active spacecraft labeled from $i = 1 : N$ and m depots labeled from $d = 1 : m$, the function to be minimized is

$$\Phi = \sum_{i=1}^{N} J^*(i, d_{nearest}) \tag{3.59}$$

where

$$J^*(i, d_{nearest}) = \min\{J^*(i, d_1), J^*(i, d_2), ..., J^*(i, d_m)\}. \tag{3.60}$$

$J^*(i, d_{nearest})$ is the cost of transferring from spacecraft i to the lowest ΔV cost target depot d, so the total ΔV for all N spacecraft to rendezvous with their lowest cost option d is being minimized here. $J^*(i, d_1)$ is given by Eq. (3.33).

Solution methods for the $m = 1$ case are discussed at length in the previous work [17], but it is especially important to highlight the discontinuities present in the problem. The analytic equations for the transfer costs between orbits will change depending on the relative orbital states between the initial and final orbits (see Section 3.1.2). This leads to discontinuities in the derivatives of the cost, which complicates the use of gradient-based optimization methods. For the $m \geq 1$ case considered here, additional discontinuities are introduced by having each spacecraft rendezvous with the "nearest" fuel depot. In the case where the candidate orbital placements of depots d_1 and d_2 gives a spacecraft the same cost to rendezvous with both depot d_1 and depot d_2, then the derivative of the total cost will be different as depot d_1 is moved "closer" to the spacecraft than the derivative of the cost as depot d_1 is moved "farther" from the spacecraft.

For these reasons, particle swarm optimization (PSO) was chosen to solve for the orbit elements of each of the m targets. PSO also has the benefit of finding a solution relatively quickly, even for large values of N. A standard genetic algorithm was also used to solve this problem, though it typically takes longer than PSO to produce similar results. If a more accurate solution is desired, a nonlinear programming (NLP) solver may be used to further optimize the output solution of the PSO, but again care must be taken around the discontinuities. The discontinuities here are more complicated than in the $m = 1$ case because the lines of discontinuous derivatives now depend both on the orbital positions of the N spacecraft as well as on the placement of the m depots which changes as the problem is optimized.

Fig. 3.32 shows a sample solution to this problem. Here, the same set of 50 orbits from the large scale MSP problems (Figs. 3.29-3.31) is used as the set of active spacecraft, and 3 supply depots, represented by x markers, were optimally placed. Active spacecraft orbits have the same color as their "nearest" supply depot to indicate the "service area" of each depot. A trend in the solution based on the inclination distribution is again found, as seen in the Perigee-Inclination view. Each fuel depot serves a certain range of inclinations, without any apparent overlaps. While a heuristic method (PSO) is again being used to calculate the optimum solution, the dimensionality of this problem is relatively low, and gives more confidence in the result. Regardless of how many spacecraft or targets N exist, the number of optimization variables is only $3m$.

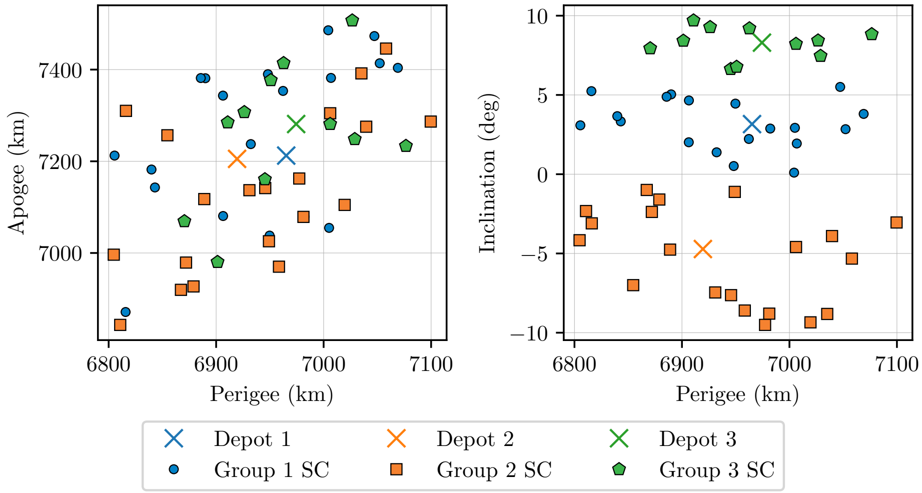


Figure 3.32: Optimal placement of 3 supply depots for a system of 50 active spacecraft. Total ΔV for all spacecraft to rendezvous with the nearest depot is 11.5 km/s, total run time was about 6 minutes using Matlab

3.2.5 Conclusions

In this work we have formulated and solved for the lower bounds on a number of different "traveling satellite problems" that are of interest to the astrodynamics community. First we address the problem of optimally selecting the path for a single spacecraft to travel through a given set of targets. This problem is applicable to a wide array of different mission scenarios. We then explore two scenarios with multiple traveling satellites, one where multiple satellites are optimally assigned subsets of a full set of targets to visit (MSP), and another where supply depots are optimally placed on orbit to minimize the total ΔV to transfer from target orbits to the supply depot (depot placement problem). In the single traveling satellite problem and the MSP, we demonstrate how to exactly solve for the optimal ΔV value that provides a lower bound. In the single spacecraft case, tools developed for the standard TSP can be applied to exactly solve problems with large N fairly quickly. In the MSP case, we show how one might exactly calculate the lower bound ΔV by enumerating all possible target assignments and solving the single traveling satellite subproblem. For larger N MSPs where this is not possible in a feasible amount of time, both a GA and ILP methods are used to solve for what are potentially over-approximations of the lower bound ΔV .

Finally, in the case of the depot placement problem, because it is a problem of solving for optimal real parameters (i.e. $\boldsymbol{x} \in \mathbb{R}$), there is no analogous brute force technique to guarantee that a global optimum is found. In that case a heuristic method (particle swarm) is also used to solve for what is likely again an over-approximation of the bounding optimal ΔV . These multiple spacecraft problems are especially challenging due to their high-dimensionality, though the solutions presented here provide another method with which to compare results to.

The bounding values that can be found using the solution methods discussed here can be an important way to contextualize the optimality of solutions that are obtained with consideration of additional constraints, higher fidelity calculations or optimizations of transfer costs, consideration of system-level architecture design parameters, and a multitude of other factors that would impact the size of the search space. Having a bounding value gives an idea of whether or not a feasible point solution is very close to a bounding value (meaning there is little opportunity for improvement), or very far from a bounding value (meaning there is the potential for improvement, though there is no guarantee that any feasible trajectories exists to gain such improvements). The methods described here can also be used for early mission concept development in order to quickly eliminate options that will fundamentally cost too much ΔV prior to using a more full but computationally expensive optimization. That is, this technique could be used to eliminate options that are too ΔV expensive when only considering $a - e - i$, and produce a smaller trade space in which to use more computationally expensive methods. Finally, because the time-free transfers used here are actually realizable in finite time, the techniques here can also be used to inform system level design for scenarios such as constellation servicing. For example, the depot placement problem solution may be used to inform the design of a constellation and supply depots for optimum serviceability given that the locations of the targets and depots are all potentially design choices. Further, insight from solving the problems discussed here may inform heuristics for solving more constrained problems as well. The techniques here can also be augmented with additional problem-specific constraints to better inform analyses of different scenarios.

The Concorde and LKH-3 solvers are excellent choices to use to solve single traveling satellite problems, and are also likely good candidates for “inner loop” solvers where a theoretical outer-loop optimizer may be selecting different sets and numbers of target orbits. The rank representation GA presented here may be of interest for single spacecraft problems where an inner loop is not desired, but the path is to be simultaneously optimized with other variables encoded into the GA chromosomes. It could also be used in situations where instead of using the pre-computed target-target time-free transfer costs to calculate the full tour cost, instead some other trajectory optimization technique is used to calculate the cost of traversing a tour of order given by a GA gene. Such a technique could account for “true” rendezvous to match all orbit elements, and could be used in conjunction with the bounding values found here to find a suitable optimum tour.

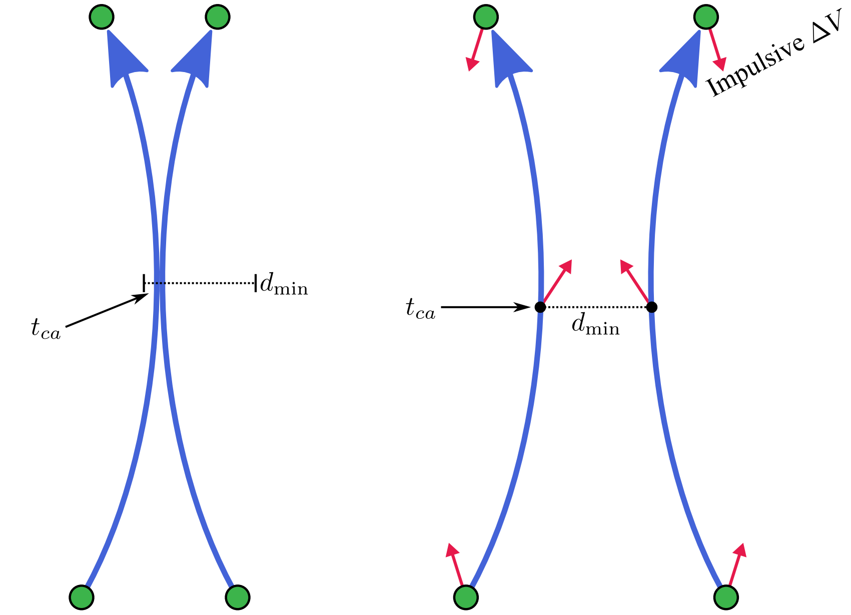
3.3 Nonlinear Programming: Impulsive, Cooperative Spacecraft Collision Avoidance

3.3.1 Introduction

Spacecraft collision avoidance is becoming increasingly important as the orbital space around Earth becomes more congested. A great deal of space debris objects must be tracked and avoided by active satellites, and the proliferation of so-called “mega-constellations” is increasing the likelihood that multiple spacecraft with active maneuvering capabilities will need to perform collision avoidance maneuvers. Collision avoidance is imperative not only to preserve the nominal functionality of the potentially impacting spacecraft, but also to ensure the the space environment does not become further polluted with more debris. Collisions between space objects that create more space debris make safely navigating the space environment more difficult for all actors.

Because of how critical collision avoidance is, it has received considerable attention in the literature from many different perspectives. One perspective analyzes a pursuit-evasion game where a pursuing spacecraft attempts to match the position of an evading spacecraft. Aside from the application of such a scenario to missile avoidance, it also provides a worst-case analysis of strategies to avoid an inert object with unknown dynamics (e.g. unknown drag force, unknown outgassing, etc.). Significant effort has also been made to develop closed loop control strategies that enable some goal to be reached (e.g. formation reconfiguration) while also avoiding any collisions. Many analyses have also aimed to improve methods of determining if a collision is going to occur. Better dynamical models of spacecraft motion can enable more precise long-term state estimation with more certainty in the probability of a collision occurring in the future. Other work focuses on finding maneuver strategies to avoid collisions with inert debris[3] [82, 83]; in this dissertation we address this aspect of collision avoidance as well but with both objects being maneuverable.

3.3.2 Problem Formulation



(a) Uncontrolled close approach (b) Controlled close approach

Figure 3.33: Uncontrolled close approach vs. three-impulse controlled close approach

The two spacecraft collision case is considered here with both spacecraft having maneuvering capabilities. Each spacecraft can maneuver impulsively three times, and crucially, must begin and end on the same trajectory/orbit. The spacecraft must return to their initial orbit to minimize the impact of the collision avoidance maneuvers on their primary task. While a spacecraft encountering

[3] for current conditions in orbit, this is the most frequent case

a close approach may not desire to return to its pre-close-approach orbit because it is in the process of performing an orbit transfer, we assume that returning to its pre-close-approach orbit is operationally desired so that the orbit transfer can occur as originally planned. When the spacecraft are uncontrolled as shown in Fig. 3.33a, they pass less than $d_{\min}$ km away from one another at the time of closest approach t_{ca}. In the controlled case shown in Fig. 3.33b, the spacecraft are constrained to be exactly $d_{\min}$ km away from one another at the time of closest approach t_{ca}. This constraint can be written as

$$\Delta\boldsymbol{r}(t_{ca})^T\Delta\boldsymbol{r}(t_{ca}) - d_{\min}^2 = 0 \tag{3.61}$$

where $\Delta\boldsymbol{r}(t_{ca}) = \boldsymbol{r}_2(t_{ca}) - \boldsymbol{r}_1(t_{ca})$ is the relative position vector at t_{ca}. The times $t_0 = 0$ and t_f are fixed, while in the controlled case t_{ca} is allowed to vary. Impulse 1 occurs at t_0, impulse 2 occurs at t_{ca}, and impulse 3 occurs at t_f. The condition at t_{ca} is

$$\Delta\boldsymbol{r}(t_{ca})^T\Delta\dot{\boldsymbol{r}}(t_{ca}) = \boldsymbol{0} \tag{3.62}$$

where $\Delta\dot{\boldsymbol{r}}(t_{ca})$ is the relative velocity vector. The desired safe close approach distance $d_{\min}$ can be used as a safe threshold given the time horizon and uncertainties in both spacecraft states. In a higher fidelity problem, the distance constraint could instead be changed to a constraint on impact probability. This could potentially allow the spacecraft to pass closer than a conservative $d_{\min}$ value while still having an acceptable probability of collision. This extension to a probabilistic constraint is not considered here.

In this formulation there is a single decision maker that decides the control strategy for both spacecraft. Because there are many solutions that result in successful avoidance, but each with a different relative control effort expended by each spacecraft, a single decision maker must decide how much control effort each spacecraft must use. This could be from the perspective of a single spacecraft operator that owns multiple spacecraft that might risk impacting one another. Such an occurrence should be unlikely, but might be a consideration when implementing autonomous collision avoidance capabilities. The single decision maker perspective could also be that of a "space traffic control" (STC) entity, similar to that of air traffic control (ATC) in that it can dictate how craft should maneuver to avoid one another. Such an entity does not exist at present, and significant regulatory and geopolitical factors make creating a STC challenging, but as space becomes more congested it may become necessary. A STC entity or set of common rules that all operators adhere to might attempt to equitably distribute how much control effort each spacecraft must expend by using the type of calculations developed in this section. In Section 6.2 a similar scenario is addressed but with continuous thrust and separate decision makers to decide each spacecraft's control.

3.3.3 Numeric Solutions: Rectilinear Dynamics

One short timescales close to the time of impact, the two spacecraft trajectories can be approximated by rectilinear dynamics in three-dimensional space with position $\boldsymbol{r} \in \mathbb{R}^3$. That is, the uncontrolled dynamics of spacecraft i can be written as:

$$\boldsymbol{r}_i(t) = \boldsymbol{r}_i(0) + \boldsymbol{v}_i(0)t \quad i = 1, 2 \tag{3.63}$$

Given these simple dynamics, and impulsive maneuvers $\Delta v_{j,i}$ where i indexes the spacecraft number and j indexes the maneuver number, the relative position at t_{ca} is

$$\Delta\boldsymbol{r}(t_{ca}) = [\boldsymbol{r}_2(0) + (\boldsymbol{v}_2(0) + \Delta v_{1,2})t_{ca}] - [\boldsymbol{r}_1(0) + (\boldsymbol{v}_1(0) + \Delta v_{1,1})t_{ca}] \tag{3.64}$$

while the relative velocity at t_{ca} is

$$\Delta\dot{\boldsymbol{r}}(t_{ca}) = [(\boldsymbol{v}_2(0) + \Delta v_{1,2})t_{ca}] - [(\boldsymbol{v}_1(0) + \Delta v_{1,1})t_{ca}] \tag{3.65}$$

The constraints on final states are

$$\boldsymbol{r}_i(0) + \boldsymbol{v}_i(0)t_f = \boldsymbol{r}_i(0) + (\boldsymbol{v}_i(0) + \Delta v_{1,i})t_{ca} + (\boldsymbol{v}_i(0) + \Delta v_{1,i} + \Delta v_{2,i})(t_f - t_{ca}) \quad i = 1, 2 \tag{3.66}$$

to ensure each spacecraft returns to its originally planned position at t_f, and

$$\boldsymbol{v}_i(0) = \boldsymbol{v}_i(0) + \Delta v_{1,i} + \Delta v_{2,i} + \Delta v_{3,i} \quad i = 1, 2 \tag{3.67}$$

to ensure each spacecraft returns to its original, unperturbed velocity at t_f. For convenience, we also define

$$^{i}\Delta v = \sum_{j=1}^{3} |\Delta\boldsymbol{v}_{j,i}| \tag{3.68}$$

A benefit of the rectilinear formulation is that all potential impacts can be described within a single plane defined by the velocity vectors of both spacecraft at the impact point. Given this collision plane, all possible impact configurations can be described by changing the angle between and magnitudes of the two velocity vectors at the impact.

If we first suppose that spacecraft 2 does not maneuver at all, then the collision avoidance optimization problem is then

$$\min_{\Delta\boldsymbol{v}_{1,1},\Delta\boldsymbol{v}_{2,1},\Delta\boldsymbol{v}_{3,1},t_{ca}} \sum_{j=1}^{3} |\Delta\boldsymbol{v}_{j,1}| \tag{3.69}$$

such that constraints from Eqs. (3.61 - 3.62) and Eqs. (3.66 - 3.67) are satisfied. That is, select t_{ca} and each impulsive maneuver for spacecraft 1 such that the total Δv is minimized and the constraints are satisfied.

If spacecraft 1 does not maneuver at all, then the problem is

$$\min_{\Delta\boldsymbol{v}_{1,2},\Delta\boldsymbol{v}_{2,2},\Delta\boldsymbol{v}_{3,2},t_{ca}} \sum_{j=1}^{3} |\Delta\boldsymbol{v}_{j,2}| \tag{3.70}$$

such that constraints from Eqs. (3.61 - 3.62) and Eqs. (3.66 - 3.67) are satisfied. The solutions to the problem of Eq. (3.69) and the problem of Eq. (3.70) give the extreme ends of the Pareto front that dictates the optimal trade-offs in Δv costs for each spacecraft. If the optimal Δv cost when only spacecraft 1 manuevers is

$${}^{1}\Delta v^{*} = \min_{\Delta v,t_{ca}} \sum_{j=1}^{3} |\Delta v_{j,1}| \quad \text{with} \quad \sum_{j=1}^{3} |\Delta v_{j,2}| = 0 \tag{3.71}$$

then the Pareto front can be found using the following problem:

$$\min_{\substack{\Delta\boldsymbol{v}_{1,2},\Delta\boldsymbol{v}_{2,2},\\ \Delta\boldsymbol{v}_{3,2},\Delta\boldsymbol{v}_{1,1},\\ \Delta\boldsymbol{v}_{2,1},\Delta\boldsymbol{v}_{3,1},\\ t_{ca}}} \sum_{j=1}^{3} |\Delta\boldsymbol{v}_{j,2}| \tag{3.72}$$

subject to

$${}^{1}\Delta v = k \quad k \in [0, {}^{1}\Delta v^{*}] \tag{3.73}$$

and constraints from Eqs. (3.61 - 3.62) and Eqs. (3.66 - 3.67). This is, to find Pareto optimal solutions, we use the control impulses from both spacecraft and t_{ca} as decision variables, and minimize the total ${}^2\Delta v$ such that ${}^1\Delta v$ is constrained to a single value $k \in [0, {}^1\Delta v^*]$ along with the other problem constraints. By solving the problem in Eq. (3.72) for different values of k, the Pareto front showing the optimal trade off between ${}^1\Delta v$ and ${}^2\Delta v$ can be found.

While the dynamics here are linear, the problems in Eqs. (3.70), (3.69), and (3.72) are NLP problems due to nonlinearities in the objectives and constraints. In this section we use the NLP solver IPOPT [28] to solve the NLP problems.

3.3.3.1 Rectilinear results

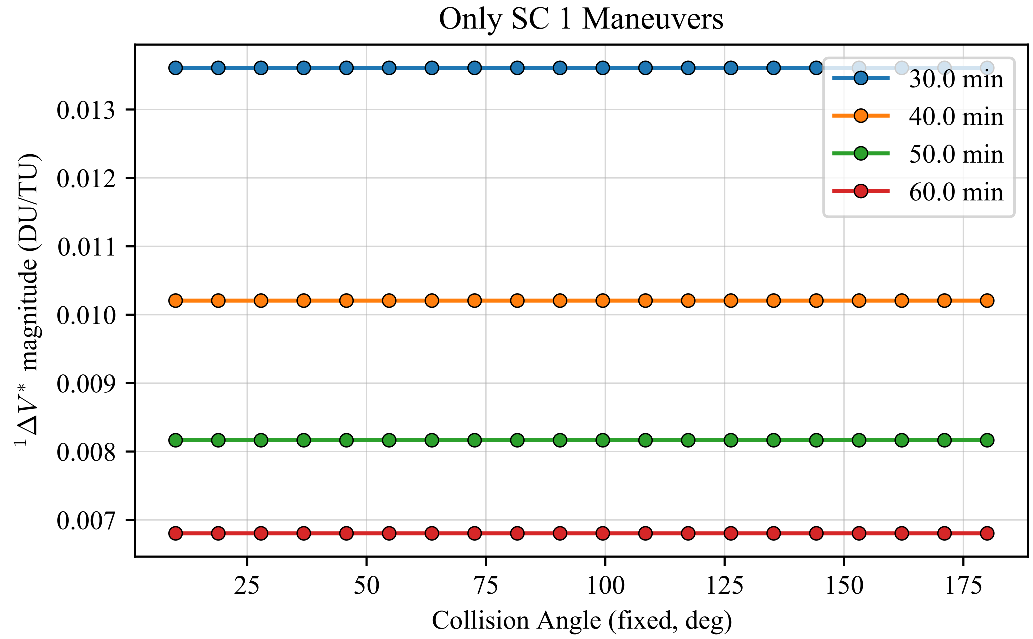


Figure 3.34: ΔV cost for Spacecraft 1 to avoid collision with linear dynamics, varying collision angle and time

First, we explore the case where only spacecraft 1 manuevers. The speed of both spacecraft is set to 7.5 DU/TU, and in Fig. 3.34 the total time is varied from 30 to 60 “minutes” (a “minute” for linear problems here are TU/60 given that the circular orbit speed around the Earth at 7000 km is roughly 7.5 km/s; DU and TU then roughly correspond to kilometers and seconds). The minimum distance is fixed at 5 DU. Note that the total ΔV cost is insensitive to the collision angle; this is less true for very short time span engagements in which case smaller collision angles have a lower

ΔV cost. Note the nonlinear relationship between engagement time and ΔV cost.

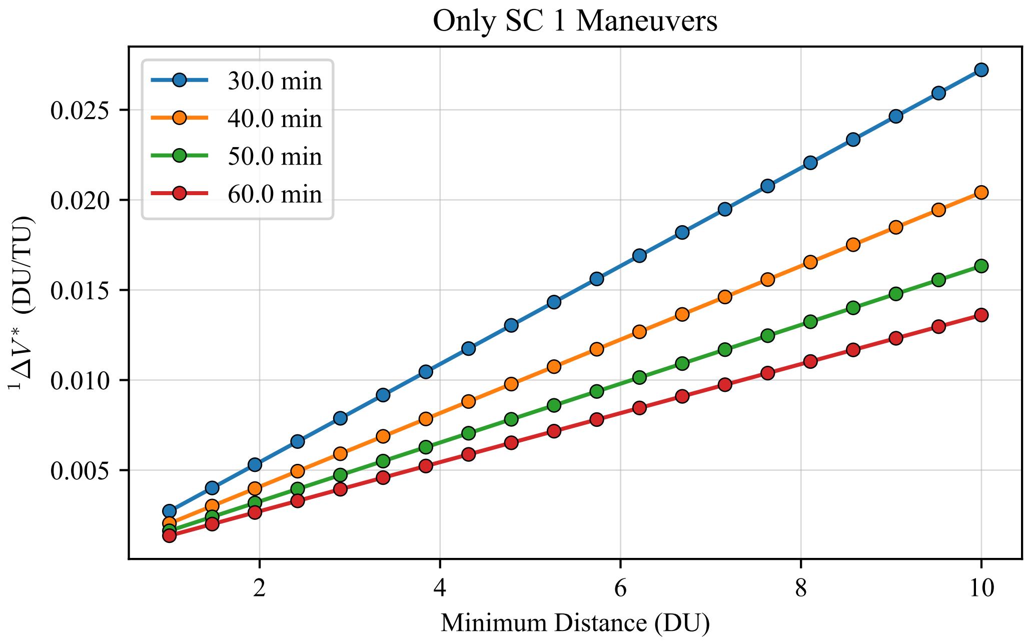


Figure 3.35: ΔV cost for Spacecraft 1 to avoid collision with linear dynamics, varying minimum distance and time

In Fig. 3.35 we fix the collision angle at 50° and instead vary the minimum distance as well as the time. While the minimum distance has a linear relationship with ΔV cost, note that the relationship changes depending on the engagement time.

Figure 3.36 shows the Pareto front of optimal trade-offs in ΔV costs between the two spacecraft as the minimum distance d_{min} is varied. Increasing the minimum distance appears to shift the Pareto front by roughly the same amount for each d_{min}. Figure 3.37 shows the Pareto front of optimal trade-offs in ΔV costs between the two spacecraft as the total engagement time t_f ($t_0 = 0$) is varied. Increasing the engagement time shifts the Pareto front by a different amount depending on the previous value of t_f.

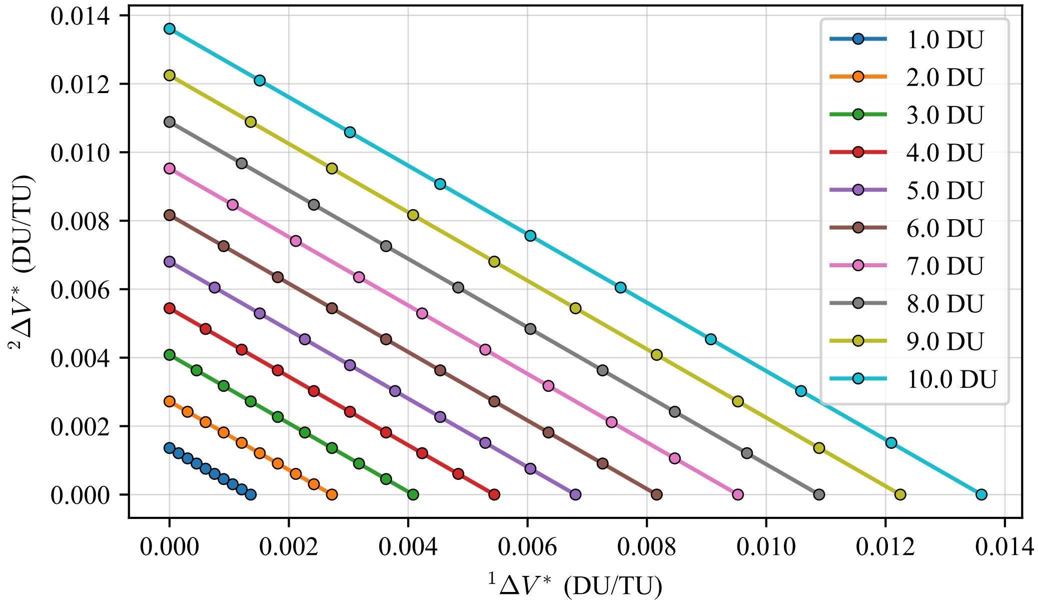


Figure 3.36: ΔV cost Pareto front with linear dynamics, varying minimum distance $d_{\min}$

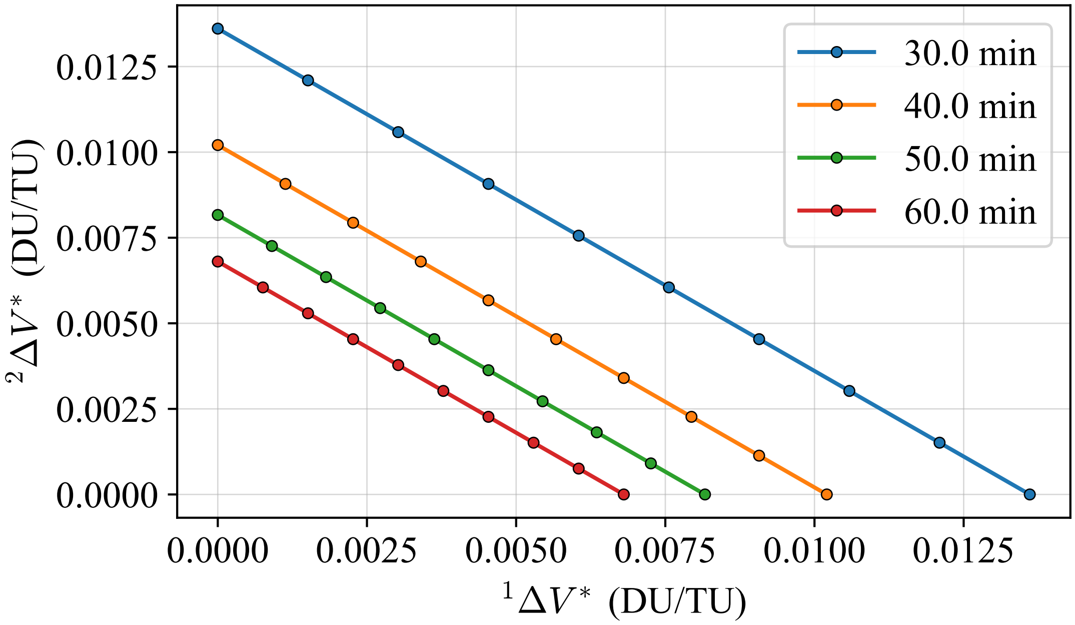


Figure 3.37: ΔV cost Pareto front with linear dynamics, varying time t_f

3.3.4 Numeric Solutions: Keplerian Dynamics

When the dynamics of the spacecraft are Keplerian as opposed to rectilinear, the problem can be solved in almost exactly the same way. The notable changes are that the positions and velocities over time must be calculated by propagating the orbital state; in this work we use the Bate, Mueller, and White universal variable propagation method [84]. This changes how $\Delta\boldsymbol{r}$ and $\dot{\boldsymbol{r}}$ as well as the desired end states $\boldsymbol{r}_i(t_f)$ and $\boldsymbol{v}_i(t_f)$ are calculated. Otherwise, the constraints and objectives are the same as in the rectilinear problem. In Fig. 3.38 we show how the cost for collision avoidance varies if only spacecraft 1 maneuvers and the total time is varied. Spacecraft 1 has $a = 7350$ km, $e = 0.0476$, $i = 50°$, $\omega = 0°$; spacecraft 2 has $a = 7075$ km, $e = 0.0106$, $i = 25°$, $\omega = 0°$, and a direct collision occurs at the ascending node.

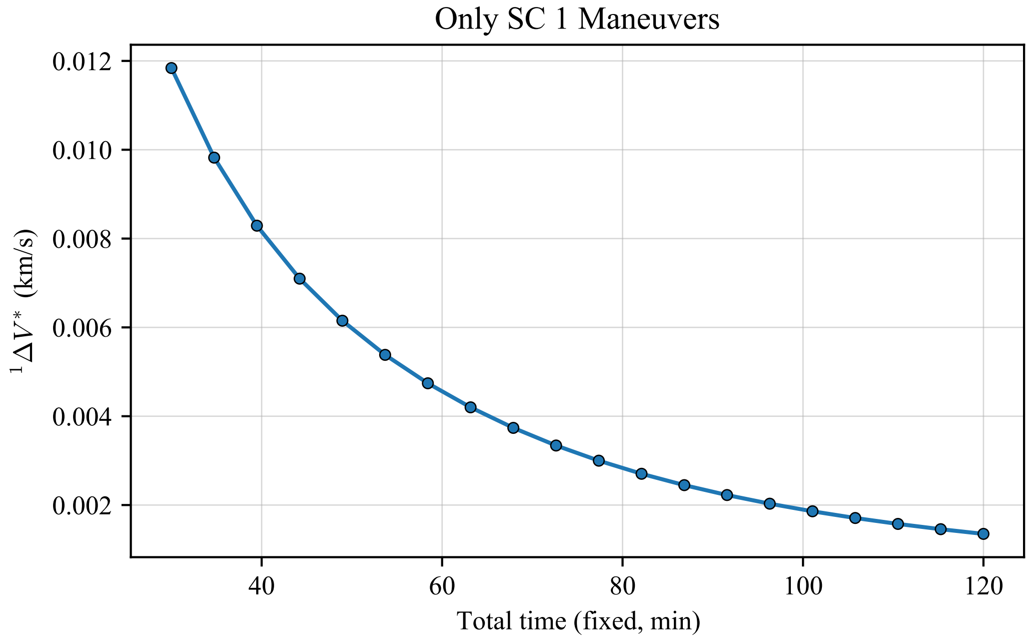


Figure 3.38: ΔV cost for Spacecraft 1 to avoid collision, varying time t_f

We can similarly generate Pareto fronts for optimal trade-offs in ΔV expenditure between each spacecraft for the Keplerian case. Figure 3.39 shows how the Pareto front evolves as $d_{\min}$ is varied, and Fig. 3.40 shows how the Pareto front evolves as t_f is varied. The relationships in the Keplerian case match trends seen in the linear case, despite the nonlinearity introduced by Keplerian dynamics. This is likely due to the small magnitude of the maneuvers needed for successful collision avoidance in these cases; the total ΔV needed for avoidance is a small percentage of the speed of each spacecraft.

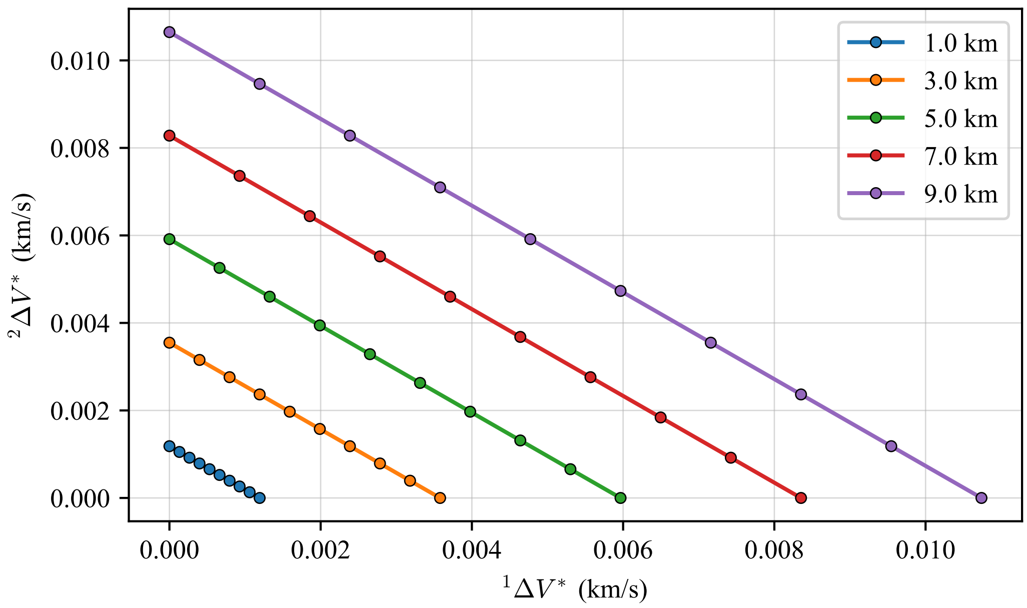


Figure 3.39: ΔV cost Pareto front with Keplerian dynamics, varying distance $d_{\min}$

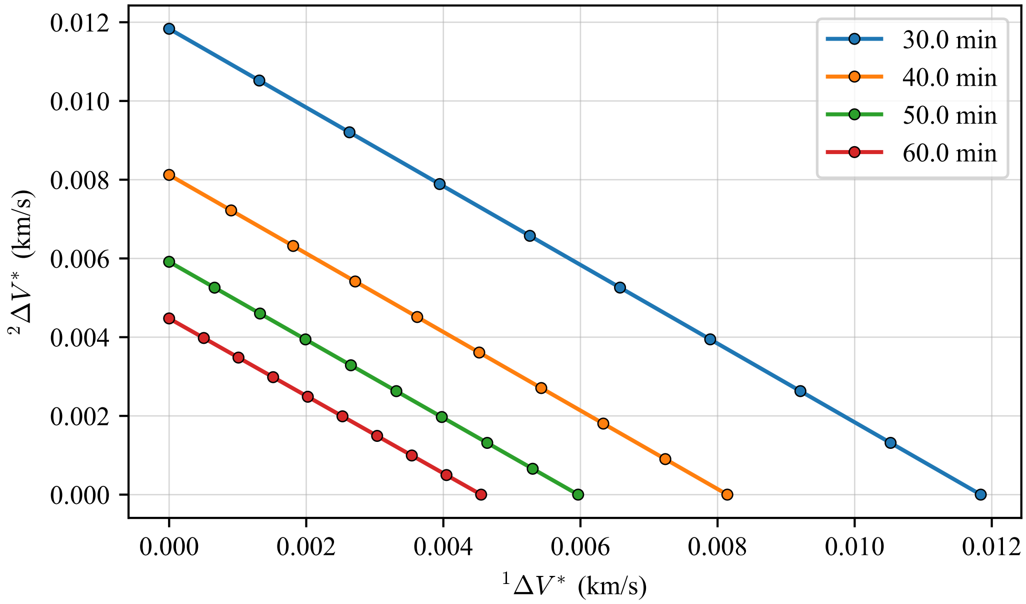


Figure 3.40: ΔV cost Pareto front with Keplerian dynamics, varying time t_f

Chapter 4

Optimal Control Approaches

4.1 Direct Simultaneous Trajectory Optimization: Application to Robust, Pareto-Optimal Low-Thrust Trajectory Design

4.1.1 Introduction

Low-thrust propulsion methods, especially solar electric propulsion (SEP), are becoming increasingly used in space missions. These methods are useful for their efficiency and ability to, in many cases, deliver more massive payloads to a target than traditional chemical propulsion methods. However, low-thrust methods typically require that a thruster be on for long periods of time, whereas a chemical mission may only require the firing of a thruster for comparatively fewer instances and shorter periods of time in each instance. In the event that some issue prevents a spacecraft with low-thrust propulsion from following its nominal thrust profile (e.g., spacecraft transitions to safe mode and ceases thrusting), the outage has the potential to render the desired target inaccessible given the available propellant on-board and time available to reach the destination. In other words, a surprise loss of thruster operations may cause the mission to fail if the nominal trajectory is not designed to be robust to such losses. In this work, we define the missed thrust recovery margin (MTRM) as the longest amount of time a spacecraft may coast away from a nominal trajectory while still being able to reach a terminal manifold once thruster operations are resumed. In other words, the MTRM at a given point on a trajectory is the length (e.g., days) of the longest forced coast period that still allows the spacecraft to reach its target. While MTRM is important in a variety of contexts, this work specifically focuses on the MTRM problem in interplanetary trajectories. An Earth orbiting low-thrust spacecraft may rely less on precise timing and orbital periods around Earth are relatively short, while an interplanetary low thrust spacecraft will likely be relying on relative phasing of various celestial bodies and likely cannot afford to wait a full orbital period before resuming operations.

A great deal of work on trajectory optimization has focused on mass- and time-optimal trajectories, but there is a gap in work on directly optimizing the MTRM of a trajectory in the literature. Laipert and Longuski [85] address the missed thrust problem by evaluating how much additional propellant is needed to allow a certain amount of MTRM throughout a given nominal trajectory. They also explore the trade-off between the additional propellant margin and how late the spacecraft arrives at the target. Critically, however, their technique does not enable the reshaping of the nominal trajectory to be more robust to missed thrust events. While their technique enables the analysis of how much reserve propellant is needed to meet robustness requirements, the result is not necessarily giving the most propellant optimal way of doing so. Laipert and Imkin [86] analyze the potential impact of multiple missed thrust events using a Monte-Carlo approach given historical data and statistics on missed thrust events in past space missions. This analysis similarly uses propellant margin and lateness to measure the robustness of a given trajectory to the multiple missed thrust events.

Ozaki et. al. [87] explore a stochastic differential dynamic programming approach to dealing with uncertainty in low-thrust spacecraft missions. However, the theory in its current form does not appear to have the ability to directly account for the missed thrust problem. Instead, uncertainty is modeled as Gaussian disturbances. Olympio [88] also takes a stochastic approach to the problem. Olympio and Yam [89] address the missed thrust problem with a constrained optimization problem where the MTRM at each point in the nominal, mass-optimized trajectory is constrained to be greater than or equal to some threshold MTRM value. MTRM in the optimization problem is calculated using an approximation which is found by fitting a function to discrete points in state space where MTRM is evaluated. The technique developed in this section is not totally dissimilar, though does not use a function approximation to constrain MTRM. As reported by Oh et. al. [90], the Dawn mission used a “rolling coast” method to ensure that a minimum of 28 days of forced shutdown time was always possible at any point in the nominal trajectory. Ad hoc methods include inserting coast arcs into more sensitive (low MTRM) parts of a nominal trajectory [91] and lowering the planned thruster duty cycle for sensitive parts of a trajectory.

Concurrent with the development of the work in this section, McCarty and Grebow [92] also developed a similar method to address the missed thrust problem. The “ghost” trajectories discussed in their work are quite similar to the “virtual” trajectories developed in this section. Ghost trajectories are applied in the three-body problem to make cis-lunar trajectories more robust to missed thrust events. The virtual swarm method developed here in Sections 4.1.2 and 4.1.3 is generally applicable across dynamical models, but the example results focus on interplanetary trajectories using Keplerian dynamics (Sections 4.1.4 - 4.1.6). McCarty and Grebow use a fixed number of ghost trajectories (five) in targeted locations to improve MTRM at points in the nominal trajectory known to have the worst robustness to missed thrust events. The limited number of ghost trajectories reduces the number of trajectories that must be simultaneously optimized to lessen the computational load, but does not necessarily allow MTRM to be fully constrained or optimized across the entire trajectory. The virtual swarm method developed here requires a variable number of virtual spacecraft to be added to the problem to ensure that worst-case MTRM is controlled across the entire nominal trajectory. Given the more complex dynamics in the three-body problem, applying the virtual swarm method to cis-lunar trajectories would likely be more challenging than the interplanetary example trajectories that are explored in this section. Notably, the ghost trajectories in [92] also appear to have a fixed thrusting structure, meaning that the optimization process cannot uncover alternative thrust profiles that may be ideal for recovery trajectories. The Sims-Flanagan transcription used in the examples here provides more freedom for the optimizer to select thrusting and coasting arcs.

Rather than relying on ad hoc, heuristic, or potentially over-conservative approaches to making a nominal low-thrust trajectory robust, we instead aim to directly optimize or constrain the worst-case MTRM along a nominal, deterministic trajectory. This allows a mission designer to either meet a given robustness constraint throughout a trajectory, or allows them to understand the best possible value of the worst-case MTRM. This section introduces a “virtual swarm” technique where a nominal spacecraft trajectory is simultaneously optimized with a discrete number of recovery trajectories with consideration for MTRM. The MTRM can either (1) be constrained in each recovery trajectory to be some minimum value (enforce a shutdown of at least some amount of time) with some other variable being optimized, or alternatively (2) the lower bound constraint on the MTRM for each recovery trajectory can be used as the optimization variable to find a maximally robust nominal trajectory. Note that in this work, when optimizing MTRM, we specifically are interested in optimizing the worst-case MTRM to ensure that the worst-case recovery time is as long as possible. This is in line with typical requirements for past missions such as the Dawn mission (28 days minimum recovery time at any point [93]), but if more uniformity or certain distributions of MTRM throughout a trajectory are desired, future work may consider a different optimization objective.

In addition to addressing the single-objective problem of optimizing worst-case MTRM along a trajectory, this work also addresses the multi-objective problem because such considerations are critically important in the mission development phase. It is rare for a mission designer to have only a single objective to consider; frequently they are dealing with many different aspects of the mission to find the “best” overall trajectory. Thus, the impact of optimizing worst-case MTRM on other aspects of the trajectory is also of interest. The virtual swarm method developed here allows a multi-objective analysis of the trade-off between worst-case MTRM along a nominal trajectory, delivered mass, and target arrival date. Specifically, we aim to develop an automated method of generating a Pareto front for these objectives that presents the best case scenario for one objective given that the other objectives are fixed. Solutions that are represented along a Pareto front cannot improve one of the objectives without performing worse in another. A Pareto front is perhaps one of the best ways to give decision makers the ability to weigh different objectives that may not have a “correct” answer. For example, each mission may prioritize the competing objectives of missed thrust robustness and payload mass in different ways, and an informed decision may be best made by considering the full curve of Pareto optimal solutions. Further, methods that can be automated are important in allowing the efficient exploration of a trade space; too much need for human intervention may make exploration of the trade space too difficult to do in a tractable amount of time.

4.1.2 Problem Statements

First, consider the low-thrust spacecraft nominal trajectory optimization problem of maximizing the total delivered mass to a target. In the terminology of optimal control, this is the problem of maximizing the objective

$$J = m_f \tag{4.1}$$

with general state dynamics

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t; \boldsymbol{\alpha}) \tag{4.2}$$

where $\boldsymbol{x}$ is the spacecraft state

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \\ m \end{bmatrix} \tag{4.3}$$

with Cartesian position $\boldsymbol{r}$, Cartesian velocity $\boldsymbol{v}$, and mass m. $\boldsymbol{u}$ is the control vector provided by the propulsion system and $\boldsymbol{\alpha}$ is a set containing spacecraft parameters (e.g. solar panel size, thruster performance, etc.). There is the terminal constraint to match the state of the target at the final time

$$\boldsymbol{g}_f(\boldsymbol{x}_{rv}(t_f), t_f) = \boldsymbol{x}_{rv}(t_f) - \boldsymbol{x}_{rv}^{\text{target}}(t_f) = \boldsymbol{0} \tag{4.4}$$

where

$$\boldsymbol{x}_{rv} = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \end{bmatrix} \tag{4.5}$$

refers only to Cartesian state, and there are inequality constraints on initial and final times

$$t_{0,\min} \leq t_0 \leq t_{0,\max} \tag{4.6}$$

$$t_{f,\min} \leq t_f \leq t_{f,\max} \tag{4.7}$$

and potentially an inequality constraint to ensure that the time of flight (TOF) is less than a desired maximum TOF, $T_{\max}$

$$t_f - t_0 \leq T_{\max} \tag{4.8}$$

There is also a constraint on the initial state

$$\boldsymbol{g}_0(\boldsymbol{x}(t_0), t_0) = \boldsymbol{x}(t_0) - \boldsymbol{\psi}(t_0, \boldsymbol{\rho}) = \mathbf{0} \tag{4.9}$$

Where the function $\boldsymbol{\psi}$ determines the initial state of the spacecraft as a function of time and as a function of the set of auxiliary parameters $\boldsymbol{\rho}$. The set $\boldsymbol{\rho}$ can include parameters such as C_3, launch asymptote, and related parameters. At its simplest, $\boldsymbol{\psi}$ may simply fix the initial state of the spacecraft on some orbit (e.g., Earth's heliocentric orbit). Parameters in $\boldsymbol{\rho}$ may also have associated equality or inequality constraints.

The objective J is optimized by selecting decision variables $\boldsymbol{u}$, t_0, t_f, and potentially some or all of the auxiliary parameters $\boldsymbol{\rho}$. This problem will be referred to as the "reference problem" or "mass-optimal" problem, and is frequently a baseline problem being solved by mission designers who use a variety of additional methods to take into account other mission objectives and constraints that are not explicitly accounted for in this formulation.

This work focuses on augmenting the reference problem with consideration for missed thrust recovery margin, which can be considered as either a constraint or an objective. The missed thrust recovery margin β at an arbitrary point in state space $\boldsymbol{x}$ at time t is defined as

$$\beta = M(\boldsymbol{x}, t; \boldsymbol{g}(\cdot), \boldsymbol{f}(\cdot)) = \max_{\boldsymbol{u_r}(\cdot) \in \mathbb{U}} t_{sd} \tag{4.10}$$

where t_{sd} is the forced shutdown time when the spacecraft cannot thrust. There is the constraint

$$\boldsymbol{u}(t) = \mathbf{0} \quad \text{for } t_0 \leq t \leq t_{sd} \tag{4.11}$$

enforcing that the spacecraft cannot thrust until $t > t_{sd}$ and the constraint

$$m_f \geq m_{f,\min} \tag{4.12}$$

The maximization problem in Eq. (4.10) is also subject to the same state dynamics $\boldsymbol{f}(\cdot)$ as the reference problem and the same Cartesian terminal constraint in Eq. (4.4). The values of $t_{f,\min}$, $t_{f,\max}$, and $m_{f,\min}$, however, can be selected either to match the values used and found in the reference problem or selected to be more permissive to allow recovery trajectories to have worse performance than the reference. For example, if $m_{f,\min}$ is chosen to be the optimized value of m_f found in the reference problem and the same time bounds on t_f are used as in the reference problem, the value of β represents the amount of time t_{sd} that is tolerable at the point in state space while still being able to deliver a mass of m_f to the target within the original time bounds. A more permissive approach would calculate β with a longer allowed time of flight and/or lower amount of delivered mass (e.g., if a missed thrust event occurs it is ok to arrive at the target 30 days later than the original $t_{f,\max}$).

The set $\mathbb{U}$ contains all control functions $\boldsymbol{u}(\cdot)$ that generate a state trajectory $\boldsymbol{x}(\boldsymbol{u}(t), t)\forall t$ that satisfy the terminal constraints given a starting point $\boldsymbol{x}$ and t as well as spacecraft parameters such as thruster performance characteristics. Some optimal recovery control function $\boldsymbol{u}^*_{\boldsymbol{r}}(\cdot) \in \mathbb{U}$, will both meet the constraint in Eq. (4.11) and maximize t_{sd}. That is, $\boldsymbol{u}^*_{\boldsymbol{r}}(\cdot)$ is a function describing the recovery optimal control that will allow the longest forced shutdown period t_{sd}.

Though β is a quantity that can be defined at arbitrary points in state space, here it is of interest when evaluated along the nominal trajectory of a low-thrust spacecraft. Especially important in mission design is the worst-case missed thrust margin of a nominal spacecraft trajectory. That is, considering all points along a nominal trajectory, what is the shortest amount of time a spacecraft has to recover from a forced shutdown event before it can no longer reach its target. For a given spacecraft state trajectory $\boldsymbol{x}(\boldsymbol{u}(t), t)$, the worst-case missed thrust margin γ is

$$\gamma = \min_t M(\boldsymbol{x}(\boldsymbol{u}(t), t), t; \boldsymbol{g}(\cdot), \boldsymbol{f}(\cdot)) \tag{4.13}$$

In other words, γ is the lowest value of β found along a nominal trajectory as t is varied from t_0 to t_f.

Frequently, there is some lower bound requirement on γ for a low-thrust mission (e.g. $\gamma \geq 28$ days for the Dawn mission [90]). Thus, the problem of accounting for this can be considered as adding the constraint

$$\gamma \geq \gamma_{\min} \tag{4.14}$$

to the reference problem such that delivered mass is optimized while the trajectory is still forced to be robust. This is referred to here as the robust-constrained problem. Alternatively, there may be some desired lower-bound constraint on delivered mass (e.g. spacecraft design has been mostly finalized), and a mission designer may wish to maximize γ for a trajectory. This is referred to here as the robust-optimal problem.

Current methods of accounting for these robustness considerations are frequently ad-hoc, rely on the experience and intuition of the mission designer, and can require a significant time expenditure on the part of the designer. Further, while current methods may enable a designer to meet the constraint in Eq. (4.14), such methods may not enable the simultaneous optimization of delivered mass or other variables while also meeting the constraint.

In this section a method is developed that can both optimize spacecraft delivered mass with a constraint on γ as well as optimize γ with constraint on delivered mass. The method can also be automated to enable more efficient exploration of the engineering trade space and Pareto front of solutions. While this section is focused on the lower bound of β throughout a trajectory, the method developed here is not incompatible with alternative criteria for β or alternative optimization objectives.

4.1.3 Virtual Swarm Method

The central idea of the virtual swarm method is to simultaneously optimize a nominal spacecraft trajectory along with its recovery trajectories after simulated forced shutdown events. Each "virtual" spacecraft provides a method of fixing or optimizing β at discrete points along a nominal trajectory. Simultaneously optimizing the nominal trajectory along with its recovery trajectories enables the nominal trajectory to be reshaped in conjunction with values related to the recovery trajectory (e.g. recovery trajectory forced coast time, delivered mass).

A virtual spacecraft is one that has a fixed "spawn point" where its state $\boldsymbol{x}_0 = \boldsymbol{x}_{\text{nominal}}(t_{\text{spawn}})$ where t_{spawn} is the time along the nominal trajectory the virtual spacecraft is spawned. Each spawn point can be thought of as a discrete control point for missed thrust recovery margin along the nominal trajectory. The virtual spacecraft themselves can be equivalently thought of as spacecraft that have the same control history as the nominal spacecraft from $t = t_0$ to $t = t_{\text{spawn}}$. At t_{spawn}, the virtual spacecraft has a forced shutdown/coasting period of t_{sd} days, after which it can resume thrusting with controls and other decision variables (e.g. TOF, m_f) that are independent of the nominal spacecraft trajectory. The nominal and the virtual spacecraft must all satisfy the specified terminal constraints. An arbitrary number N of these recovery trajectories can be added, and here are indexed by the variable i. Care must be exercised when selecting the number N and spawn times of virtual spacecraft; this is discussed later in this section. At each spawn point, a virtual spacecraft has a forced shutdown (coast) time of $t_{sd,i}$, where $i = 1, 2, ..., N$. Let $i = 0$ refer to the nominal spacecraft. This method is illustrated in Fig. 4.1.

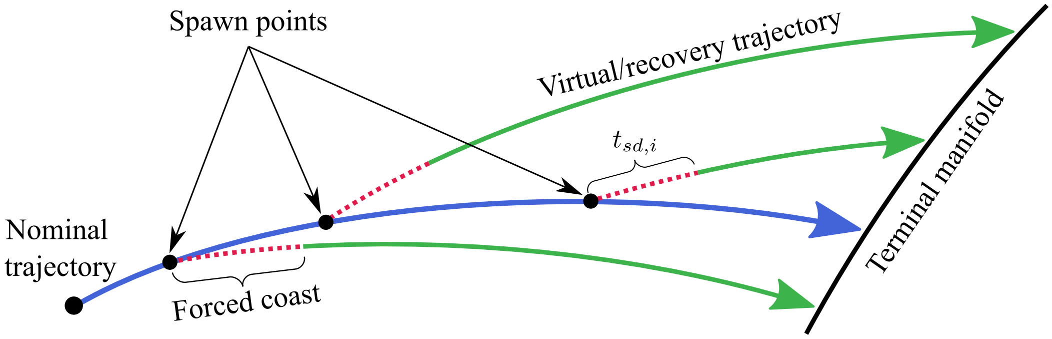


Figure 4.1: Illustration of the virtual swarm method

For the robust-constrained reference problem, the optimization is formulated as the minimization of

$$J = -m_{f,\min} \tag{4.15}$$

subject to

$$t_{sd,i} = \gamma_{\text{min}} \quad i = 1, 2, ..., N \tag{4.16}$$

$$m_{f,i} \geq m_{f,min} \quad \forall i \tag{4.17}$$

$$t_{f,min} \leq t_{f,i} \leq t_{f,max} \quad \forall i \tag{4.18}$$

While the nominal spacecraft still has the initial constraints of Eq. (4.9), each virtual spacecraft has the aforementioned initial constraint

$$\boldsymbol{x}_{0,i} = \boldsymbol{x}_{\text{nominal}}(t_{\text{spawn},i}) \quad i = 1, 2, ..., N \tag{4.19}$$

with

$$t_{0,i} = t_{\text{spawn},i} \quad i = 1, 2, ..., N \tag{4.20}$$

In short, this formulation fixes γ by assigning that value to each $t_{sd,i}$ and then optimizes the lower bound on delivered mass to ensure that the worst-case delivered mass is the highest possible. The lower bound is the focus because the spacecraft design must be able to account for worst-case performance.

The resulting value of $m_{f,\text{min}}$ represents the worst-case delivered mass. While $m_{f,i}$ is a decision variable for all spacecraft, in the results of the optimization problem only the limiting case $m_{f,i} = m_{f,\text{min}}$ represents a maximum delivered mass for that spacecraft. The other virtual spacecraft can be separately optimized to find the maximum delivered mass for each case.

For the robust-optimal problem, the optimization is formulated as the minimization of

$$J = -t_{sd,min} \tag{4.21}$$

subject to

$$t_{sd,i} \geq t_{sd,min} \quad i = 1, 2, ..., N \tag{4.22}$$

$$m_{f,i} \geq m_{f,min} \quad \forall i \tag{4.23}$$

$$t_{f,min} \leq t_{f,i} \leq t_{f,max} \quad \forall i \tag{4.24}$$

Again, the nominal spacecraft still has the initial constraints of Eq. (4.9), and each virtual spacecraft has the aforementioned initial constraint in Eq. (4.19).

This formulation fixes a lower bound on delivered mass $m_{f,i}$, ensuring that the nominal, as well as all recovery trajectories, deliver sufficient mass while ensuring that the lower bound value γ is as large as possible. This ensures that the weakest point in the trajectory in terms of β has the largest possible value. Similar to the robust-constrained reference problem, $\beta_i = t_{sd,i}$ only for the trajectory where $t_{sd,i} = t_{sd,min}$. For other trajectories, $\beta_i \geq t_{sd,i}$, and β for each point can be found by re-solving for it at each point along the nominal trajectory. If a sufficient number N of virtual spacecraft are used in proper spawn locations, then $t_{sd,min} = \gamma$.

Note that both the robust-optimal and robust-constrained problems can both be used to uncover the same solution. Consider a robust-constrained problem where an optimal $m_{f,\mathrm{min}}$ is found with fixed $t_{sd,i}$. Using that same $m_{f,\mathrm{min}}$ as a fixed constraint value in the robust-optimal problem and optimizing $t_{sd,\mathrm{min}}$ will result in $t_{sd,\mathrm{min}} = t_{sd,i}$ from the robust-constrained problem. Solving both problems and ensuring that the solutions match as expected can increase the confidence that a solution has been properly found.

The number and placement of the N virtual spacecraft is extremely important in ensuring that the constraint of γ is properly met or in ensuring that the lower bound γ is properly optimized. Theoretically, an infinite number of virtual spacecraft could be added such that one is spawned at each time of the nominal trajectory. A large number of equally spaced virtual spacecraft could be added throughout the nominal trajectory in lieu of constraining every point. However, a large number of virtual spacecraft will require a large number of decision variables that may slow efforts to numerically converge on an optimal solution. Alternatively, virtual spacecraft can be iteratively added as constraint violations are found. First, N_{init} initial virtual spacecraft can be placed at locations in the nominal trajectory where it is expected to be sensitive to missed thrust events (e.g. right before arrival at the final target, right before a gravity assist). Then, more virtual spacecraft can be iteratively added as shown in Algorithm 5.

Algorithm 5: Iterative scheme of adding virtual spacecraft

```
Result: Optimized nominal and recovery trajectories
add $N_{\text{init}}$ virtual spacecraft;
solve Eq. (4.15) or Eq. (4.21);
evaluate $\beta$ at each point along the trajectory, calculate $\gamma$;
while $\gamma < \gamma_{\mathbf{min}}$ do
    add $N_w$ virtual spacecraft, one at each the $N_w$ worst violations points where $\beta < \gamma_{\text{min}}$;
    solve Eq. (4.15) or Eq. (4.21);
    evaluate $\beta$ at each point along the trajectory, calculate $\gamma$;
end
```

This is the same scheme used for many optimization problems where explicitly implementing all constraints can be computationally expensive. It does not, however, dilute the result of the solution; constraints need not be explicitly enforced as long as they are not violated in the final solution. Algorithm 5 can be automated or done manually by the mission designer.

4.1.4 Low Thrust Trajectory Transcription

While the virtual swarm method can be implemented numerically using a number of different trajectory transcriptions, in this section low-thrust trajectories will be described using a multiple phase version of the well known Sims-Flanagan direct transcription of the low-thrust trajectory optimization problem [33, 94]. This transcription is well suited to global optimization problems and enables faster convergence on local optima for low-thrust trajectories. While it is relatively low fidelity, it enables efficient exploration of wide search spaces and provides initial guesses for higher fidelity methods that are not as able to explore wide regions of state space. This is especially important for efficient exploration of virtual swarm solutions because they can be very high-dimensional problems.

In the Sims-Flanagan transcription, low-thrust control is modeled as a series of impulsive maneuvers spaced throughout a trajectory in discrete segments as seen in Fig. 4.2. In this work, a single phase is represented by a Sims-Flanagan transcribed trajectory with forward shooting from

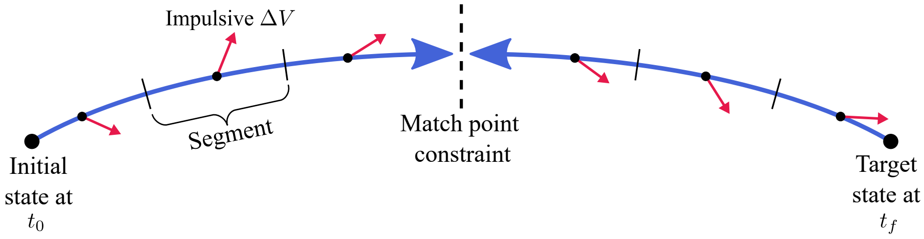


Figure 4.2: Single phase represented with a Sims-Flanagan transcription

the initial body, and backward shooting from the final body. The forward and backward parts of the trajectories are constrained to match at the midpoint of the phase, and Keplerian dynamics are used to propagate the trajectory between impulses. The universal variable formulation from Bate, Mueller, and White [84] is used for the Keplerian propagation. Multiple phases can be connected to give a trajectory with one or more gravity assists; a single phase mission here goes directly from the initial body to the final body. In a multi-phase mission, the final state of one phase is the initial state of the next phase. Phases are indexed from $k = 1, 2, ..., N_{ph}$. In each phase, there are τ_k impulses numbered from $j = 1, 2, ..., \tau_k$, each of which is centered in the segment with the same number identifier. The magnitude of each impulsive ΔV is limited in proportion to how much ΔV a low thrust engine may be able to provide in the time period of the segment. The method is summarized for a single spacecraft in Fig. 4.2. The total number of impulses τ is

$$\tau = \sum_{k=1}^{N_{ph}} \tau_k \tag{4.25}$$

and the time period of each segment is

$$q = \frac{t_f - t_0}{\tau} \tag{4.26}$$

Because continuous periods of low-thrust acceleration are approximated by impulsive ΔVs, virtual spacecraft are only spawned at the start time of a segment. For similar reasons, the maximum number of virtual spacecraft spawned is $N_{\max} = \tau$. For $N > \tau$, virtual spacecraft will spawn from points inside segments, where thrust is theoretically being applied but is not actually apparent until the impulse in the center of the segment. These points inside segments have larger differences from a higher fidelity, finite burn approximation of the low thrust trajectory than points at the start of segments. For greater control and fidelity, τ can be increased.

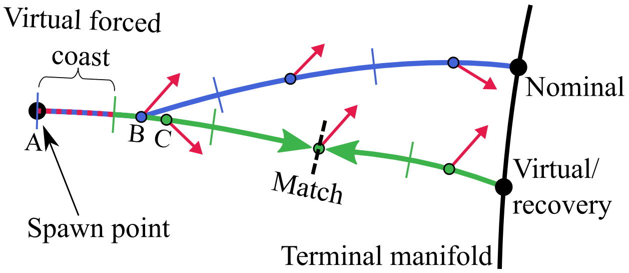


Figure 4.3: Detail on a single virtual/recovery trajectory

Figure 4.3 shows a detailed view of how a single recovery trajectory is handled in conjunction with the nominal trajectory. A virtual spacecraft can be added to the start of any segment, and is uniquely described by the nominal phase and segment it spawns from. While in general virtual trajectories immediately diverge from the nominal trajectory at the spawn point (see Fig. 4.1), when applied in a Sims-Flanagan transcription the virtual trajectory does not deviate from the nominal until one of the two spacecraft applies an impulse. In the notional example shown in Fig. 4.3, the nominal and virtual trajectories do not deviate until the nominal spacecraft applies an impulse at point B. Alternatively, the deviation could be caused by the first impulse applied by the virtual spacecraft (point C), in which case the full effect of the missed thrust event would not impact the virtual spacecraft trajectory. This occurs when

$$t_{sd,i} + \frac{q_i}{2} < \frac{q_0}{2} \tag{4.27}$$

where the time length of a single segment of the nominal trajectory is q_0 and the time length of a single segment of the virtual trajectory is q_i. The times q_0 and q_i are frequently of similar magnitude, and usually $t_{sd,i} > 0$ is desired; both factors lead to the condition in Eq. (4.27) not holding.

For a virtual spacecraft spawned at segment j of phase k, the total number of impulses

$\tau_{\text{virtual}_{j,k}}$ it has for its mission lifetime is

$$\tau_{\text{virtual}_{j,k}} = \max\{(\tau_k - j + 1), \tau_{k,\min}\} + \sum_{k+1}^{N_{ph}} \tau_k \tag{4.28}$$

The virtual spacecraft typically gets the same number of impulses as remain in the phase for the nominal trajectory, including the spawn segment, and has the full number of impulses the nominal spacecraft has in subsequent phases. A lower bound $\tau_{k,\min}$ on the number of impulses in the spawn phase is included, however, so that virtual spacecraft always have some minimum control authority to change thrust direction. In this work, $\tau_{k,\min} = 5$ is generally used, though Fig. 4.3 shows $\tau_{k,\min} \leq 3$.

The transcription introduces a number of constraints. These include explicit match point constraints to ensure the trajectory is continuous and explicit control magnitude constraints (see Ellison [94] for details). Note that the terminal constraint in Eq. (4.4) and the position component of the initial constraint in Eq. (4.9) are implicitly satisfied by the transcription and thus do not need to be explicitly stated in the problem set up for the NLP solver.

Launch auxiliary decision variables are shown in Table 4.1. The launch C_3 and maximum allowable m_0 value are related by a function $m_{0,\max} = l(C_3)$. Here, a polynomial fit to publicly available launcher data provided by NASA Launch Services Program[1] is used to model that function. The underload factor $\eta_{\text{underload}} \in (0, 1]$ is used when the initial mass is not fixed, and is instead based on the maximum possible payload mass for the selected launcher and C_3 value. When the underload factor is used, $m_0 = \eta_{\text{underload}} l(C_3)$.

Table 4.1: Launch auxiliary decision variables

Variable	Description
C_3	Characteristic energy
RLA	Right ascension of launch asymptote
DLA	Declination of launch asymptote
$\eta_{\text{underload}}$	Launch mass underload factor

For the robust-constrained reference problem, the additional decision variables are shown in Table 4.2. The number of entries contributed to the decision vector for each variable is listed in

[1] `https://elvperf.ksc.nasa.gov/Pages/Query.aspx`

the "Number of Values" column. To express the number of decision variables, a binary variable $\nu_{j,k} \in \{0,1\}$ is used to express if a virtual spacecraft is spawned at segment j in phase k of the nominal trajectory. If $\nu_{j,k} = 1$, a virtual spacecraft is spawned at that point. The binary variable relates to the total number of virtual spacecraft N with the relation

$$N = \sum_{k=1}^{N_{ph}} \sum_{j=1}^{\tau_k} \nu_{j,k} \tag{4.29}$$

Table 4.2: Robust-constrained reference problem decision variables

Variable	Description	Number of Values
t_0	Initial time at start of each phase	$N+1$
TOF	Time of flight from t_0 to t_f	$N+1$
$\boldsymbol{u}$	Control vector components	$3\tau + \sum_{k=1}^{N_{ph}} \sum_{j=1}^{\tau_k} 3\nu_{j,k}\tau_{\text{virtual}_{j,k}}$
m_f	Final mass at end of each phase	$N+1$
$m_{f,min}$	Minimum delivered mass constraint	1

For the robust-optimal problem, the additional decision variables are are shown in Table 4.3.

Table 4.3: Robust-optimal problem decision variables

Variable	Description	Number of Values
t_0	Initial time at start of each phase	$N+1$
TOF	Time of flight from t_0 to t_f	$N+1$
$\boldsymbol{u}$	Control vector components	$3\tau + \sum_{k=1}^{N_{ph}} \sum_{j=1}^{\tau_k} 3\nu_{j,k}\tau_{\text{virtual}_{j,k}}$
m_f	Final mass at end of each phase	$N+1$
$t_{sd,i}$	Coast times	N
$t_{sd,min}$	Minimum coast time	1

4.1.5 Software Implementation

Given the discrete optimization variables and constraints from the transcription, the resulting nonlinear programming problem is solved here using the commercial optimization package SNOPT [30]. While analytic partial derivates of objectives and constraints with respect to decision variables are available in prior work [94, 95], for fast development the tool created for this analysis (named the N Spacecraft Trajectory Optimizer, or NSTOP) uses automatic differentiation [96, 97] and all functionality is written in the Julia programming language. Automatic differentiation provides exact, machine precision partial derivatives without user specification of analytic partials. However, this comes at the cost of slower run time speed as compared to using analytic partials only. The NLP solver is augmented with monotonic basin hopping in an outer loop for a stochastic search, similar to methods previously used [95, 98, 99]. The stochastic search is parallelized by running multiple processes with a stochastic search at the same time. Each process shares its best result with the other processes, so all workers use the current consensus best solution as a starting point for continued stochastic searching. Basin hopping alone enables convergence on a solution even with a very poor initial guess, but the speed of the process is further improved with parallelization. The use of a stochastic search in general also gives a measure of ability to find an approximate global optimum as opposed to only being able to find a local optimum with the NLP solver alone. NSTOP has been validated against the open source version of NASA Goddard's Evolutionary Mission Trajectory Generator (EMTG) [100] and produces nearly identical optimal trajectories when the same problem is posed in both tools.

The design of NSTOP is relatively complex, and has been shaped significantly by the design of the Julia programming language in which NSTOP is implemented. NSTOP is designed to optimize an array of `Mission` structs[2] . A `Mission` struct represents a single spacecraft trajectory, and has fields (variables) such as:

- Mission type (e.g. nominal or virtual)
- Unique Identifying Number
- Parent mission ID and spawn point (if virtual)
- Thruster type and related parameters (e.g. I_{sp}, Max Thrust, thruster model for polynomial models of $T_{\max}$, solar array coefficients, P_0, etc.)
- Array of `Phase` structs

Each `Mission` struct contains at least one `Phase` struct, which has fields describing the trajectory

[2] In an object oriented language the structs might instead be implemented as objects.

between two celestial bodies, or between the spawn of a virtual spacecraft and the next celestial body. The `Phase` struct has fields to describe parameters such as:

- Unique Identifying Number
- Start point type (e.g. Earth launch, fixed state, celestial body, spawn from parent)
- Start point parameters (e.g. launch C_3, DLA, RLA, launch vehicle, initial celestial body state spline function)
- V_∞ arrays out of start point / in to end point
- Initial and final masses
- Bounds on initial and final masses
- Initial and final states
- Initial time
- Time of flight
- Initial and final forced coasting time (e.g. for virtual spacecraft spawn or post-launch/flyby coast)
- Set of $\boldsymbol{u}$ arrays for each impulse
- Decision variables (indicates which fields might be modified by the optimizer)
- Decision variable bounds and scaling factors
- Objective variable & sign (indicates if a field in this phase is the objective variable)
- Global decision variable (indicates if a field in this phase should be updated from a global decision variable, e.g. $m_{f,\min}$)
- Set of constraint functions (e.g. constraint impulse magnitude, match state at midpoint, time of flight, final mass)

- Constraint bounds and scaling factors

Each spacecraft trajectory is then defined using `Mission` and `Phase` structs, and NSTOP provides a function to return a single decision vector from an array of `Mission` structs, as well as a function to update all structs based on an updated decision vector called `updateMissions!()`. This function must not only update any fields in a `Mission` or `Phase` struct that are directly decision variables, but also update any fields in each struct that change as a result of an updated field. For example, when the control arrays in a parent `Mission` are changed, the initial state in the first `Phase` of any child spacecraft must be updated based on the new trajectory of the parent spacecraft. NSTOP also provides functions to get arrays of evaluated constraint values and constraint bounds given an array of `Mission` structs. These functions, along with the automatic-differentiation-based Jacobian calculation function, allow NSTOP to transform the spacecraft trajectory optimization problem into an NLP for SNOPT to solve. Each time the NLP solver updates the decision vector, the fields in all `Mission` and `Phase` structs are updated, and objectives & constraints can be calculated using the updated field values. Additional functions handle many other tasks, such as constructing `Mission` and `Phase` structs for a virtual mission based on a nominal spacecraft and a specific spawn point.

NSTOP also provides the monotonic basin hopping and seed sharing functionality that is wrapped around the local SNOPT NLP solver. NSTOP allows the user to select a probability distribution with which to perturb decision vectors after a local NLP solve, and specify if the program should terminate after a set number of hops or after a set amount of time. NSTOP can also seed share; if multiple trajectories are being optimized in parallel, the best decision vector found by each process can be provided to other processes as an updated initial guess. This can be used to speed up the basin hopping process if the multiple processes are optimizing the same problem, or be used to share good neighboring solutions if different, but similar problems are being optimized in parallel.

4.1.6 Examples

All examples here use the same polynomial XR-5 Hall thruster model as in earlier work by Laipert [85]. Maximum thrust and mass flow rates are estimated by polynomials that are functions of available power. The thrust function is

$$T(P) = (-8.597 + 77.34P - 2.119P^2 - 1.151P^3 + 0.1739P^4) \times 10^{-3} \tag{4.30}$$

where T is thrust in newtons and P is power in kilowatts limited to the range $0.302 \leq P \leq 4.839$. The mass flow function is

$$\dot{m}(P) = (3.524 + 68.48P - 16.32P^2 + 2.351P^3 - 0.1195P^4) \times 10^{-7} \tag{4.31}$$

where $\dot{m}$ is the mass flow rate in kg/s. Two thrusters are used on the spacecraft at a 95% duty cycle, and the power per thruster is limited to the range of 0.302 kW to 4.839 kW. The power system modeling, thruster switching (maximum number), and thruster smoothing logic used here are described by Ellison [95], with solar array coefficients taken from the example used by Laipert [85]. The solar array coefficients used result in the power expression

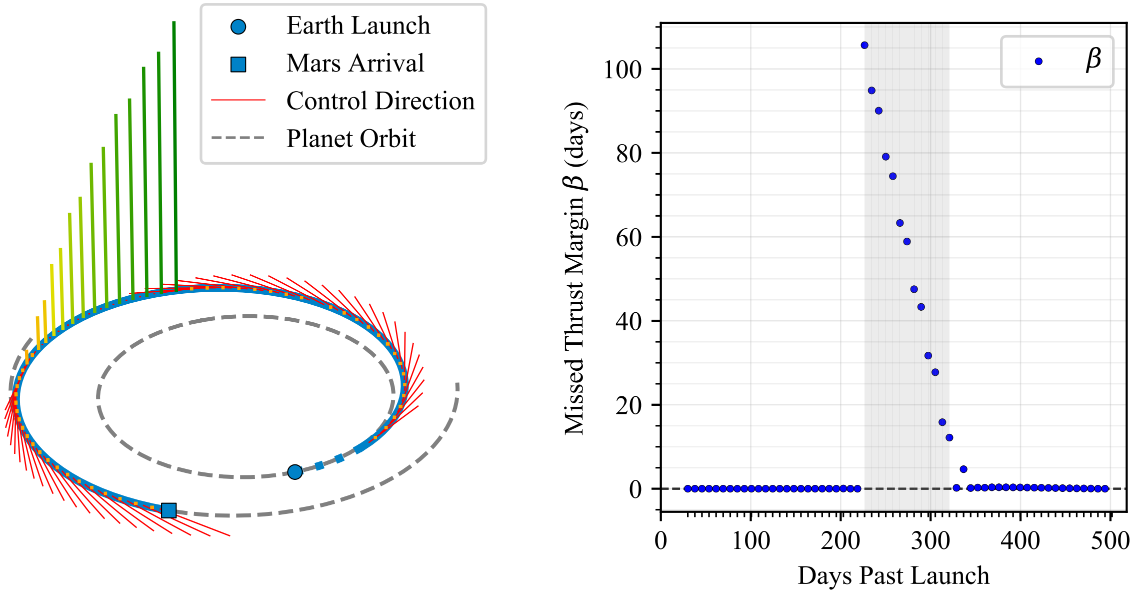
$$P(r) = \frac{P_0}{r^2}\left(\frac{1.321 - (0.108/r) - (0.117/r^2)}{1 + 0.108r - 0.013r^2}\right) \tag{4.32}$$

where r is the spacecraft distance from the sun in astronomical units and P_0 is the power available when the spacecraft is at a distance of 1 AU from the Sun. The Falcon 9 ASDS model is used for launch vehicle performance and constraints; the polynomial relating C_3 ($\mathrm{km}^2/\mathrm{s}^2$) to the maximum possible launch mass (kg) is

$$m_{0,\max}(C_3) = 0.7226C_3^2 - 116.14C_3 + 3310.8 \tag{4.33}$$

and is valid for $C_3 \in [0, 10]$ $\mathrm{km}^2/\mathrm{s}^2$. In all examples a mandatory 30 day coasting period immediately after launch is enforced. Thus, any virtual spacecraft are only spawned after that initial coast, and evaluation of MTRM occurs afterward. States of target celestial bodies are approximated using spline fits to ephemeris data provided by JPL and the SPICE toolkit.

4.1.6.1 Evaluating Missed Thrust Recovery Margin for a Nominal Trajectory



(a) Vertical lines are drawn proportional to how much MTRM there is at each point in the trajectory; larger lines indicate more MTRM

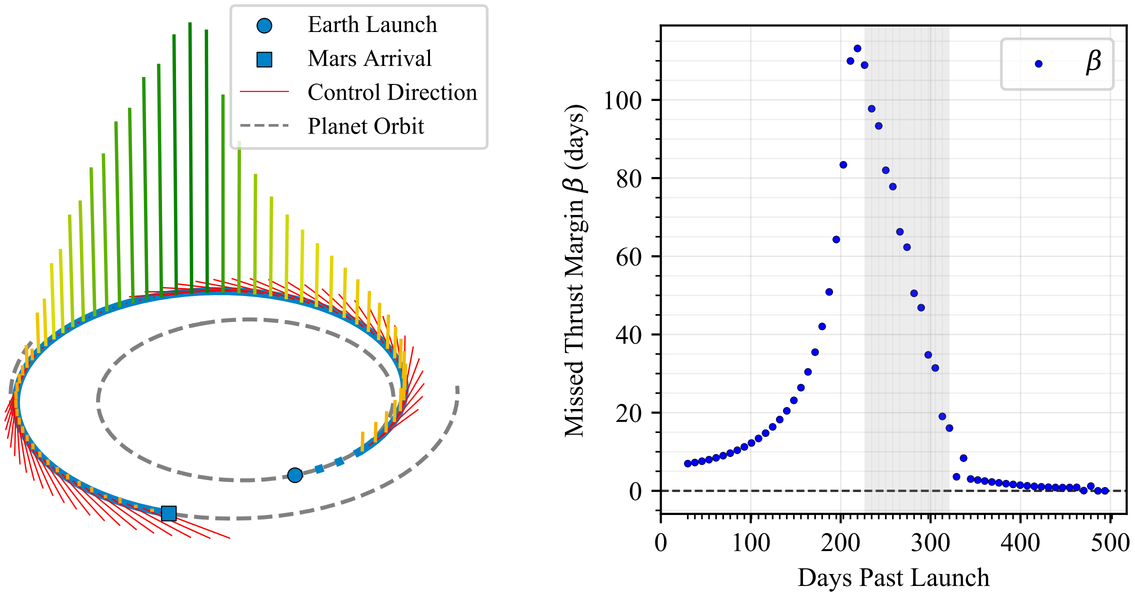
(b) Shaded region indicates a coasting period

Figure 4.4: Missed thrust recovery margin along an Earth-Mars low-thrust transfer

Given a trajectory transcription, now the optimization problem of evaluating β in Eq. (4.10) can be solved. Note that in evaluating β, the initial state $\boldsymbol{x}_0$ is fixed. In this case, x_0 is generated by selecting specific points along the nominal trajectory. Evaluating β within a Sims-Flanagan segment has limited utility, because in each segment the impulsive ΔV is a surrogate for distributed thrusting throughout the segment. For this reason, non-physical discontinuities in β are found when it is evaluated within a segment. Instead, β is only evaluated at the start points of segments in order to more accurately approximate the value of MTRM along the trajectory. As in the nominal case, a larger number of Sims-Flanagan segments can be used when greater accuracy in modeling the true low-thrust problem is desired. A "non-permissive" example of evaluating the MTRM along a low-thrust mass optimal trajectory from Earth to Mars can be seen in Fig. 4.4. Earth launch occurs at the circular point in Fig. 4.4a, a mandatory 30 day post-launch coast is shown by a blue dotted line, impulsive ΔV vectors are shown with red lines, and arrival at Mars is shown with a blue square. Vertical lines indicate the MTRM β at the start of each segment, with taller, green lines indicating larger β values and shorter, orange lines indicating smaller β values. Numeric values of β can be seen more clearly in Fig. 4.4b.

In this non-permissive example, β is optimized with the same limit $t_{f,\max}$ as the nominal trajectory and had the constraint that $m_{f,\text{recovery}} \geq m_{f,\text{nominal}}$. While in this specific case, the nominal trajectory had $t_f < t_{f,max}$ by 11 days, the additional time is not enough to allow $\beta > 0$ at all points along the nominal trajectory. Indeed, one indicator that the nominal trajectory is truly mass optimal is that there is at least one point along the trajectory where it cannot withstand any amount of forced coasting time without inducing an additional mass penalty in the recovery trajectory. From 0.42TOF ($\beta = 106$ days) to 0.62TOF ($\beta = 12.17$ days) the spacecraft is coasting, so forced coasts starting at those times overlap with planned nominal coasts and the decline in MTRM should be linear. In this evaluation the decline is close to, but not exactly, linear due to the discrete number of impulses used to approximate the post-forced-shutdown trajectory.

A more permissive example of evaluating β along a low-thrust mass optimal trajectory from Earth to Mars can be seen in Fig. 4.5. In this case, the same time bound $t_{f,\max}$ from the nominal case is used, but the lower bound on the recovery delivered mass is $m_{f,\text{recovery}} \geq m_{f,\text{nominal}} - 30$ kg. With more permissive bounds, the spacecraft can withstand longer forced shutdown periods at each point along its nominal trajectory, at the expense of using more propellant. This method could be used in a similar manner as in previous work [85] to evaluate the robustness of a nominal trajectory and determine how much additional propellant margin is needed to make a given nominal trajectory robust. However, note that towards the end of this example trajectory very little robustness is gained with 30 kg of additional fuel use; more must be done to gain robustness.



(a) Vertical lines are drawn proportional to how much MTRM there is at each point in the trajectory; larger lines indicate more MTRM

(b) Shaded region indicates a coasting period

Figure 4.5: Permissive missed thrust recovery margin along an Earth-Mars low-thrust transfer

4.1.6.2 Earth-Mars Transfer

Single Example The first example of solving the robust-constrained reference problem shown here is an Earth-Mars transfer using 30 impulses in the nominal spacecraft's trajectory transcription. User-specified values for the reference problem without robustness considerations compared to User-specified values for the related robust-constrained problem are given in Table 4.4, while details of the optimal results are given in Table 4.5. In this case, the robust-constrained reference problem solved here enforces $\gamma_{\text{min}} = 20$ days by setting a fixed forced shutdown time $t_{sd,i}$ for all virtual spacecraft. The maximum arrival date is set to 25 days later than in the reference problem, and the lower bound on delivered mass for all spacecraft in the swarm is optimized. To use a common measure, the propellant margin κ needed for the robust trajectory can be calculated as

$$\kappa = \frac{m_{f,\text{mass optimal only}} - m_{f,\text{min,swarm}}}{m_{\text{propellant,mass optimal only}}} \tag{4.34}$$

Here the propellant margin is how much additional propellant, as compared to the mass optimal only solution, is needed to recover from the worst-case missed thrust event. However, this requires that the robust-constrained trajectory be flown. If the mass optimal only trajectory is flown with the additional propellant margin, it cannot be guaranteed to recover from the worst-case missed thrust event with the given propellant margin.

Table 4.4: Robust-constrained Earth-Mars transfer problem: user specified & constant values

Parameter	Reference problem	Robust-constrained
Nominal SC number of impulses	30	30
Power available at 1AU	10 kW	10 kW
t_0 bounds (lower, upper)	Aug. 11, 2024 - Oct. 10, 2024	Aug. 11, 2024 - Oct. 10, 2024
t_f bounds (lower, upper)	Nov. 7, 2025 - Jan. 6 2026	Nov. 7, 2025 - Jan. 31 2026
Virtual spacecraft spawn segments	-	1-4, 18-30
Virtual spacecraft t_{sd}	-	20 days
Number of decision variables	97	794
Number of constraints	38	424

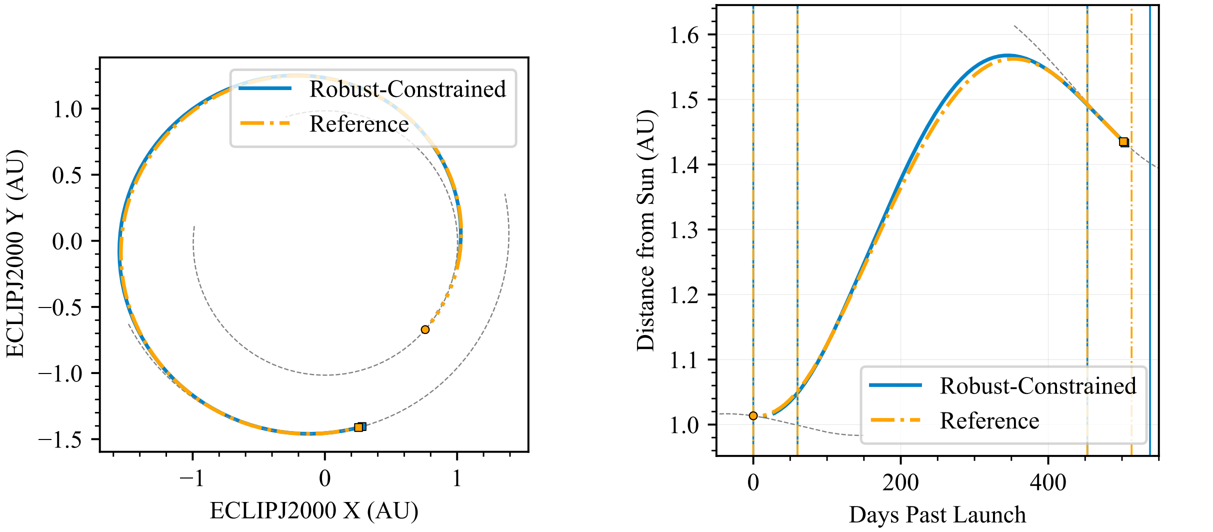
Table 4.5: Robust-constrained Earth-Mars transfer problem: optimizer selected values

Parameter	Reference problem	Robust-constrained
Launch date	Aug. 11, 2024	Aug. 11, 2024
Nominal SC arrival date	Dec. 25, 2025	Dec. 27, 2025
Nominal SC delivered mass	2343 kg	2304 kg
Nominal SC propellant mass	695 kg	673 kg
Launch mass	3038 kg	2977 kg
C_3	$2.38\,\mathrm{km}^2/\mathrm{s}^2$	$2.92\,\mathrm{km}^2/\mathrm{s}^2$
Optimized $m_{f,\min}$	-	2272 kg
Propellant margin κ	-	10%

In the robust-constrained solution, the optimizer has reduced the launch mass from 3038 kg in the reference problem to 2977 kg in the robust-constrained problem. This has the effect of enabling a higher C_3 and giving the low-thrust system more control over acceleration due to the lighter initial mass. The robust-constrained solution, however, has little qualitative difference from the reference solution, as seen in Fig. 4.6. Figure 4.6 shows the reference solution in orange and the robust-constrained solution in blue. Figure 4.6b shows the distance of both solutions from the Sun over time, with their respective time limits on t_0 and t_f shown with vertical lines. While the robust-constrained spacecraft arrives two days later than the reference spacecraft, it still arrives considerably before its $t_{f,\max}$. This allows virtual/recovery trajectories spawned later in the nominal trajectory to have time to arrive at Mars within the allowed limits.

Figure 4.7b demonstrates how the control profile of the robust-constrained trajectory shifts as compared to the mass-optimal reference trajectory. The mass-optimal trajectory for the most part has the familiar bang-bang control structure; the short segment where the thrust is neither full nor zero is likely due to the approximate nature of the Sims-Flanagan transcription. that is slightly off due to the transcription. The robust trajectory coasting time is comparatively shorter and earlier, and thrusting is no longer at 100% in the lead up to arrival at the target. Instead, the thrust level steps down as it nears arrival and reaches 50% immediately before t_f. These changes make the trajectory much less sensitive to missed thrust events. Figure 4.7a shows how the control profiles of the nominal and virtual spacecraft evolve over time. Figure 4.8 shows individual delivered mass and arrival dates for both the nominal trajectory and each virtual trajectory in the robust-constrained case. Note that nearly all virtual spacecraft spawned in the second half of the trajectory have arrival dates and arrival mass that match the limiting value. In the case of delivered mass shown in Fig. 4.8a, the values shown don't necessarily represent the maximum possible arrival mass for each recovery trajectory. This is because only the lower bound for the whole swarm is being optimized.

Finally, Fig. 4.9 shows β evaluated at each segment start point along the nominal robust-constrained trajectory. Evaluated values of β are shown as blue circles, while the value of t_{sd} for each virtual spacecraft included in the swarm is shown with red squares. Because each value of $\beta \geq t_{sd}$, the number and placement of virtual spacecraft in this case was sufficient to constrain $\gamma \geq \gamma_{\min} = t_{sd}$.



(a) X-Y planar view of the Earth-Mars trajectories

(b) Distance from sun vs. time of the Earth-Mars trajectories

Figure 4.6: Reference optimal transfer (orange) vs. robust-constrained nominal (blue) Earth-Mars low-thrust transfer. Vertical lines show limits on t_0 and t_f for their corresponding trajectory

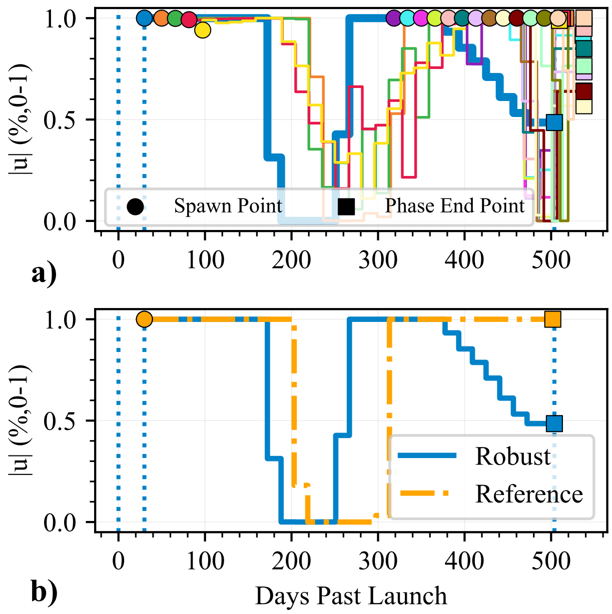
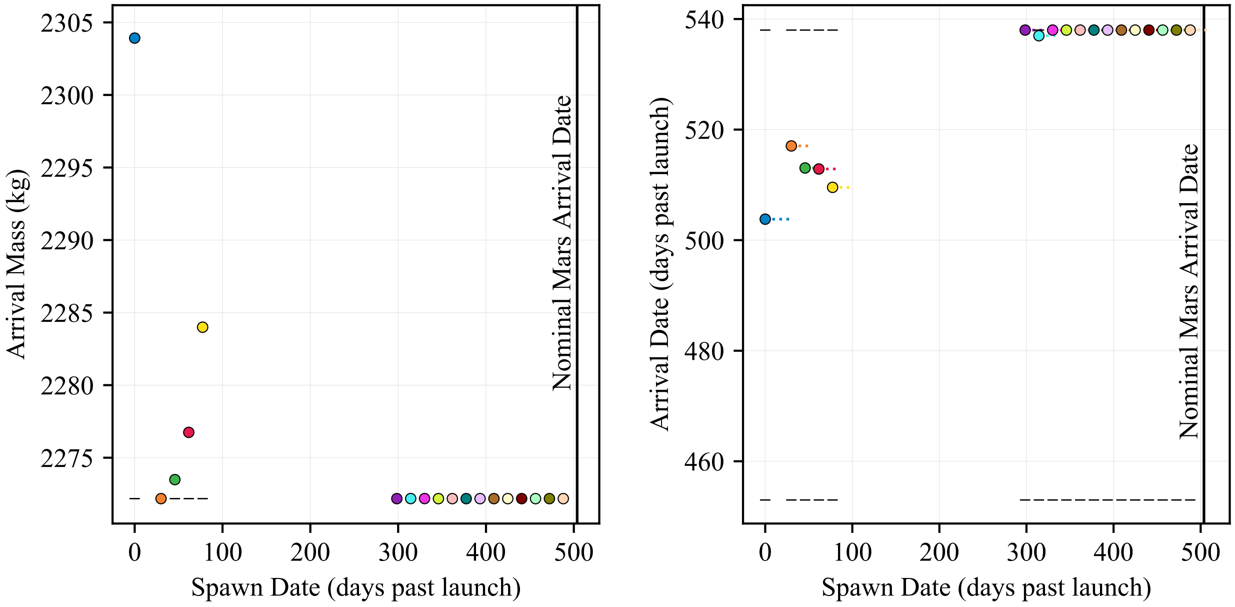


Figure 4.7: a) control history of all swarm (multi-colored) and reference (thick blue) spacecraft. b) control history of reference mass-optimal transfer (orange) vs. optimal robust-constrained nominal (blue) Earth-Mars low-thrust transfer



(a) Spawn date vs. delivered mass

(b) Spawn date vs. arrival date

Figure 4.8: Results for nominal and virtual spacecraft in the robust-constrained Earth-Mars problem; colors correspond to the virtual spacecraft colors in Fig. 4.7a

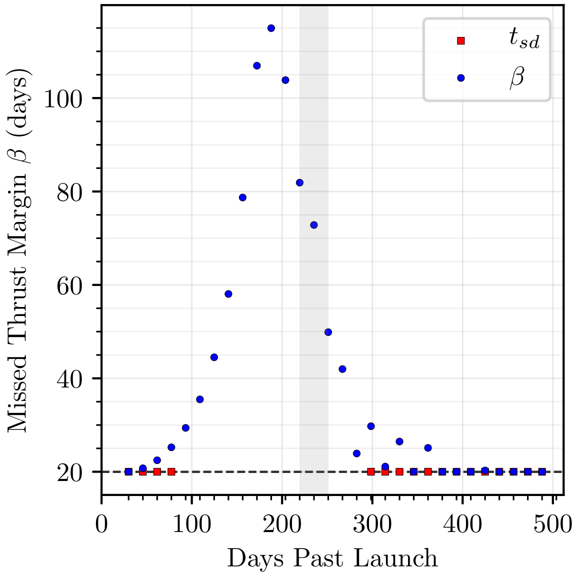


Figure 4.9: β values calculated along the robust-constrained Earth-Mars trajectory. The shaded region indicates coasting, and a dashed horizontal line is placed at the desired minimum β value of 20 days.

Pareto Front To explore the optimal trade-offs between maximum arrival date, γ, and delivered mass, the robust-constrained problem can be solved many times with different fixed values of $t_{f,\max}$ and t_{sd}. This will provide the Pareto front for this trade space. The same information could be gathered by optimizing $\gamma_{\min}$ and constraining $m_{f,\min}$ and $t_{f,\max}$, though in this example $m_{f,min}$ is the optimization variable. Other parameters of interest could also be varied to give an understanding of how different parameters impact a trajectory. To give an easy reference for how robust trajectories compare to the mass optimal only reference trajectory, the Pareto fronts in Fig. 4.10 are shown in terms of propellant margin κ instead of in terms of delivered mass. Each point represents a single solution to the robust-constrained problem with different constraints. Each curve has a corresponding γ, which is enforced in each solution through fixed t_{sd} in numerous virtual spacecraft. For each solution, β has been calculated throughout the trajectory to ensure that $\gamma = t_{sd}$. The x-axis describes how many days past the reference problem $t_{f,\max}$ that the swarm spacecraft were allowed to arrive at the target. The point solution described in more detail in the previous section is at $\gamma = 20$ days, with an allowable arrival date of 25 days past the reference problem and a propellant margin of 10.2%.

The propellant margin needed if the spacecraft is not allowed to arrive any later than the reference $t_{f,\max}$ can be quite high, especially as γ is increased to 25 days and beyond. The propellant margin drops off relatively quickly as $t_{f,\max}$ is increased from 0, though it does level off with changes from $t_{f,\max} = +50$ to $t_{f,\max} = +100$ being mostly minor. This is likely because a local minimum is found at a certain arrival date due to the relative positions of Earth and Mars at launch. However, as γ is increased, greater benefits in propellant margin κ are seen as $t_{f,\max}$ is increased.

4.1.6.3 Earth-Mars-Psyche Transfer

Single Example The same procedure can be applied to a trajectory that includes a gravity assist. In this example, an Earth-Mars-Psyche mission is used to demonstrate the technique. This trajectory is unrelated to the NASA Psyche mission, and while it is similar to the gravity assist in Laipert [85], the reference trajectory is different. A twenty day coasting period prior to arrival at the

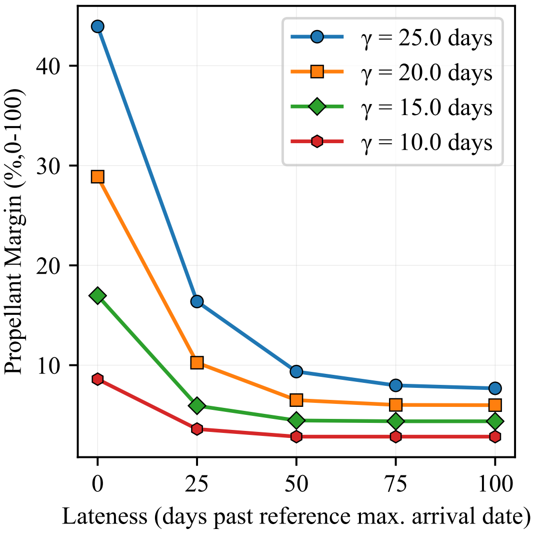


Figure 4.10: Earth-Mars Pareto front, γ, $t_{f,\max}$, propellant margin

Mars gravity assist is enforced for all trajectories to allow for navigation and targeting maneuvers to occur. This applies to nominal spacecraft and any virtual spacecraft that spawn prior to the gravity assist; if a missed thrust event occurs before the gravity assist, the spacecraft must be able to withstand a shutdown of at least $\gamma_{\min}$ days in addition to the twenty-day forced coast prior to arrival at the gravity assist. It is also possible to allow recovery trajectories to have a shorter pre-gravity assist coast than the nominal spacecraft to potentially reduce the propellant needed for the worst-case recovery trajectory at the expense of reduced time to perform pre-gravity assist operations in recovery scenarios. Such a strategy, however, is not used in these results. Details of the problem set up and optimal results for the reference problem without robustness considerations and the robust-constrained problem are given in Tables 4.6 and 4.7. The robust-constrained result is allowed to arrive 75 days later than the reference problem, and its lower bound mass $m_{f,\min}$ is optimized with a fixed $t_{sd} = 20$ days for all virtual spacecraft. Note that this robust-constrained solution again has a lower launch mass and higher C_3 than the reference problem, as was found in the Earth-Mars transfer.

In the gravity assist case, as visualized in Fig. 4.11, qualitative differences between the

Table 4.6: Robust-constrained Earth-Mars-Psyche transfer problem: user specified & constant values

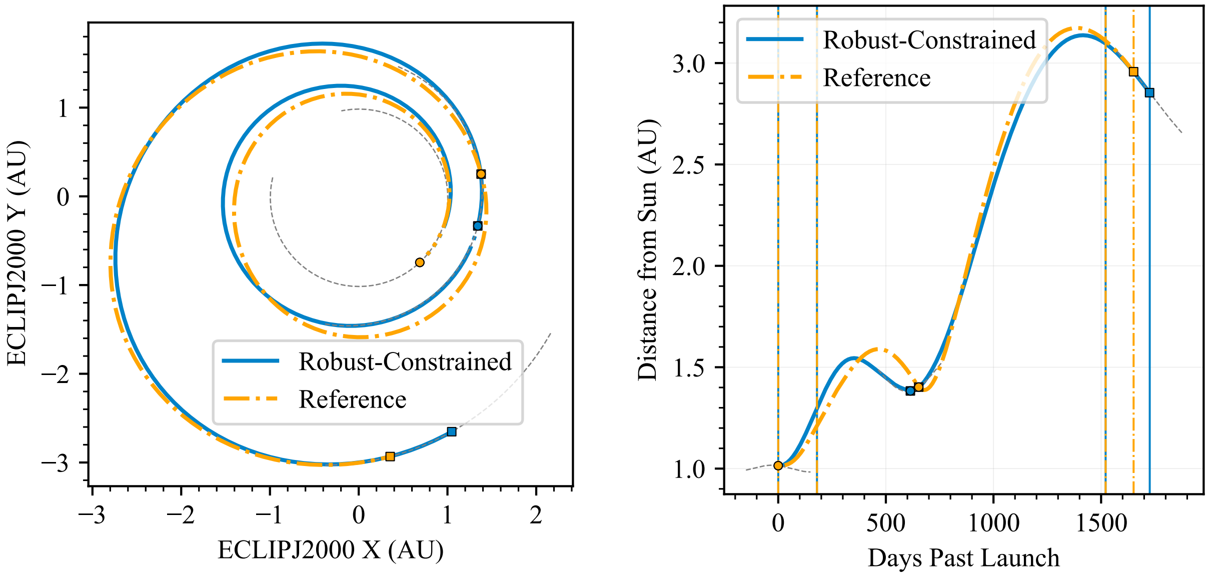
Parameter	Reference problem	Robust-constrained
Nominal SC number of impulses	60	60
Power available at 1AU	20 kW	20 kW
t_0 bounds (lower, upper)	Aug. 5, 2024 - Feb. 1, 2025	Aug. 5, 2024 - Feb. 1, 2025
t_f bounds (lower, upper)	Oct. 4, 2028 - Feb. 11 2029	Oct. 4, 2028 - Apr. 27 2029
Virtual SC phase 1 (Earth-Mars) spawn segments	-	28, 30
Virtual SC phase 2 (Mars-Psyche) spawn segments	-	1, 21, 23, 25-30
Virtual SC t_{sd}	-	20 days
Number of decision variables	194	680
Number of constraints	78	358

Table 4.7: Robust-constrained Earth-Mars-Psyche transfer problem: optimizer selected values

Parameter	Reference problem	Robust-constrained
Launch date	Aug. 5, 2024	Aug. 5, 2024
Nominal SC Mars gravity assist date	May 19, 2026	Apr. 10, 2026
Nominal SC Psyche arrival date	Feb. 11, 2029	Apr. 26, 2029
Nominal SC delivered mass	1891 kg	1580 kg
Nominal SC propellant mass	1333 kg	1523 kg
Launch mass	3224 kg	3103 kg
C_3	$0.75\,\mathrm{km}^2/\mathrm{s}^2$	$1.81\,\mathrm{km}^2/\mathrm{s}^2$
Optimized $m_{f,\min}$	-	1575 kg
Propellant margin κ	-	24%

reference and robust solutions are clearly evident. The nominal gravity assist date has been moved earlier in the robust case, which allows recovery trajectories enough time to make corrections and arrive at the gravity assist before it becomes infeasible. The robust nominal trajectory also remains closer to the Sun than the reference trajectory providing more power for thrusting. The control profile shown in Fig. 4.12 interestingly shows the robust trajectory does have some thrusting immediately prior to the gravity assist, while the reference trajectory is coasting for a long period of time prior to its gravity assist. The large spike in applied control immediately before the robust spacecraft's gravity assist is largely applied in the anti-velocity direction (see Fig. 4.11a). This indicates that the control for the robust trajectory at this point is largely being applied to control the timing for the nominal and virtual spacecraft near the gravity assist. In the second phase from Mars to Psyche, the robust trajectory has more control applied earlier in the phase, and again steps the control magnitude down as it approaches its target, much like the Earth-Mars example.

Figure 4.13 shows the swarm arrival dates and masses for the gravity assist case. Note that in Fig. 4.13a the virtual spacecraft spawned immediately prior to the gravity assist and the virtual spacecraft spawned immediately prior to arrival at Psyche are the limiting factors in the optimization of $m_{f,\min}$. This is in line with our understanding of the problem; the point where the gravity assist occurs is highly sensitive to missed thrust events. Figure 4.13b shows that several, but not all spacecraft in the swarm arrive at roughly $t_{f,\max}$. Finally, Fig. 4.14 shows β along the nominal trajectory to verify that $\gamma = t_{sd}$.



(a) X-Y planar view of the Earth-Mars-Psyche trajectories

(b) Distance from sun vs. time of the Earth-Mars-Psyche trajectories

Figure 4.11: Reference optimal transfer (orange) vs. robust-constrained nominal (blue) Earth-Mars-Psyche low-thrust transfer. Mars gravity assists are marked by square-enclosed circles, vertical lines show limits on t_0 and t_f for their corresponding trajectory

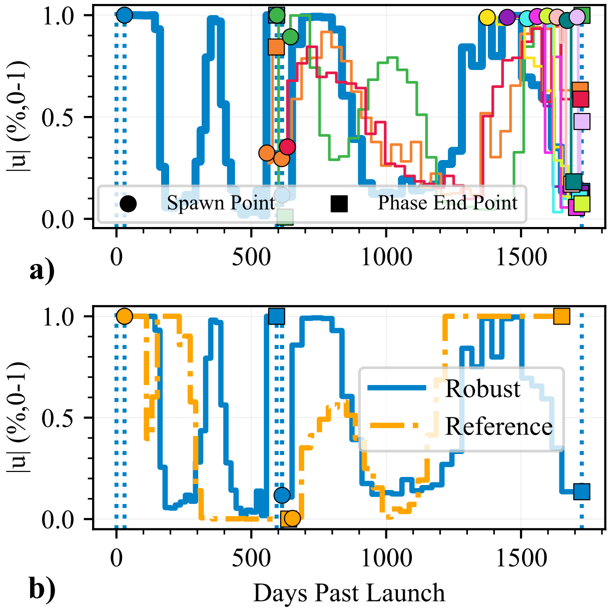
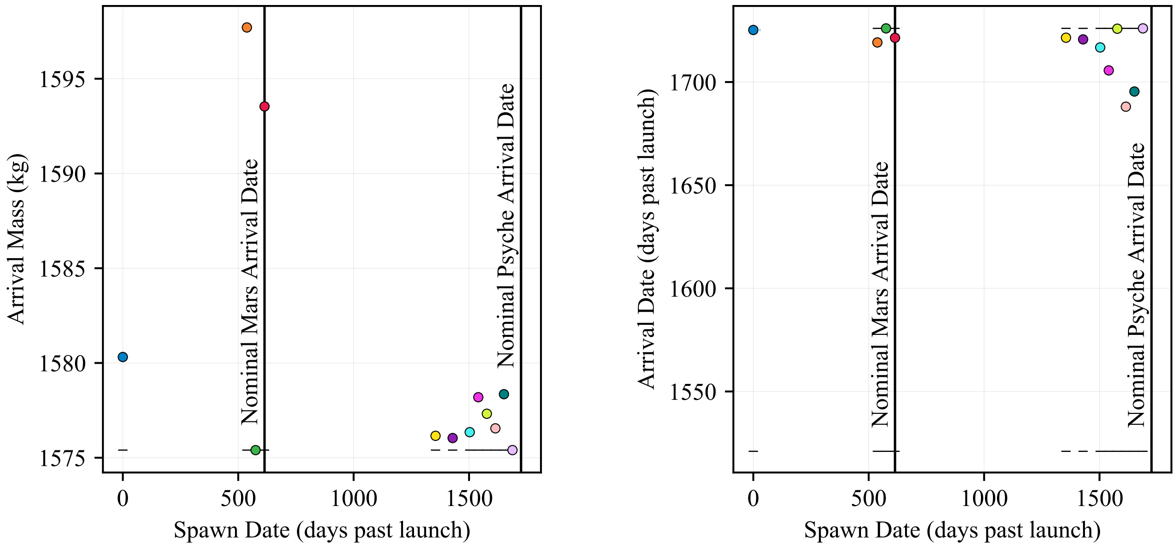


Figure 4.12: a) control history of all swarm spacecraft (thin multi-colored) and reference spacecraft (thick blue). b) control history of reference optimal transfer (orange) vs. robust-constrained nominal (blue) Earth-Mars-Psyche low-thrust transfer



(a) Spawn date vs. delivered mass

(b) Spawn date vs. arrival date

Figure 4.13: Results for nominal and virtual spacecraft in the robust-constrained Earth-Mars-Psyche problem; colors correspond to the virtual spacecraft colors in Fig. 4.12a

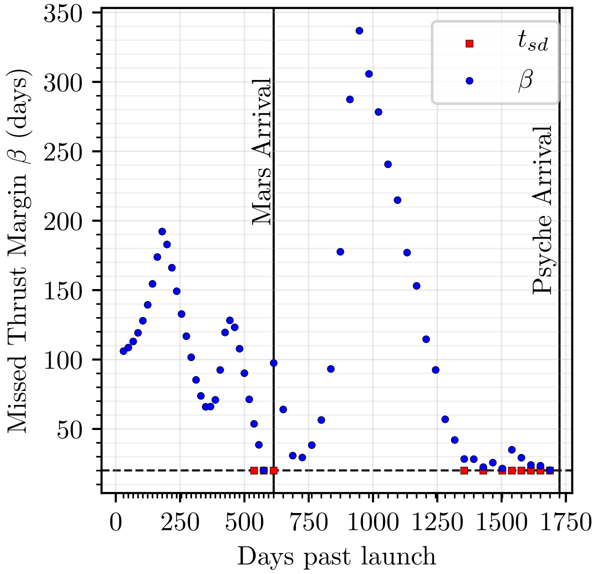


Figure 4.14: β values calculated along the robust-constrained Earth-Mars-Psyche trajectory

Pareto Front and Sensitivity

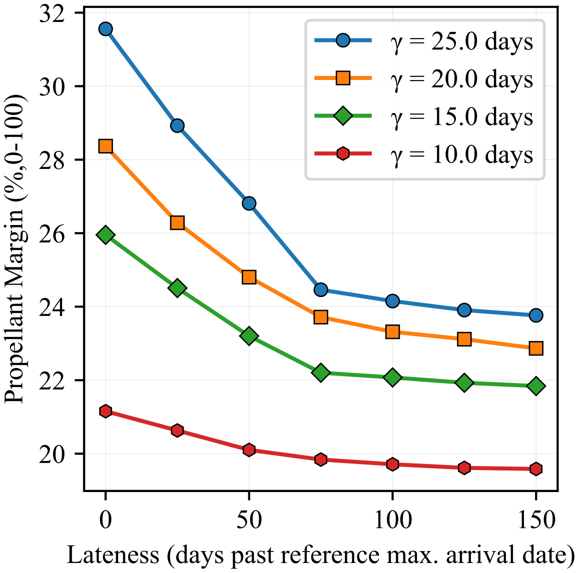


Figure 4.15: Earth-Mars-Psyche Pareto front for γ, $t_{f,\max}$, and propellant margin

A Pareto front can be generated for this case using the same method as the Earth-Mars example discussed in Sec. 4.1.6.2. The result is shown in Fig. 4.15. Note that the lowest propellant margin found in this example is significantly higher than the lowest propellant margin found in the Earth-Mars case seen in Fig. 4.10. This speaks to the additional sensitivity to missed thrust events introduced by using gravity assists in an interplanetary trajectory. Importantly, individual gravity-assist opportunities (i.e. a specific launch date and sequence of flyby locations and dates) will each have varying levels of sensitivity to missed thrust events; these examples are not necessarily representative of how much propellant margin is generally required to ensure trajectories are robust.

To visualize how the sensitivity of the trajectory to missed thrust events changes from the mass-optimal case to the robust constrained case, sensitivity plots are given in Figs. 4.16 - 4.18. In these plots, missed thrust events are simulated at various points along the nominal trajectory with varying lengths. After the simulated missed thrust event, a mass-optimal trajectory is generated with the initial state fixed to the post-shutdown state and no constraints on robustness. This gives insight into what the optimal delivered mass will be for missed thrust events occurring at different points and for different lengths of time. The sensitivity of the Psyche mass-optimal reference trajectory is shown in Fig. 4.16, where each recovery trajectory is allowed to arrive at Psyche up to 75 days late but the nominal trajectory still arrives at the date given in Table 4.7. The sensitivity of the 20-day robust and up to 75-day late Earth-Mars-Psyche trajectory discussed in detail in this section is shown in Figs. 4.17 - 4.18. While the robust trajectory was only constrained to be 20 days robust, shutdown events of up to 25 days were simulated to understand the sensitivity beyond the constrained robustness. In the contour plots (Figs. 4.17 and 4.16), white unshaded space indicates that a feasible solution was not found, likely indicating that such a trajectory is not possible. The discrete points where optimization problems were solved in order to generate the contours are shown with black points, and the contour colors correspond to the optimized mass delivered to Psyche. Figure 4.18 shows the same data as Fig. 4.17 but in a different format to more clearly show which case is limiting.

In the mass-optimal case, Fig. 4.16, there are very significant regions where a missed thrust event would render the target unreachable, even with significant additional propellant expenditure. Contrast this with the robust-constrained case in Figs. 4.17 - 4.18, where nearly all points are feasible though many have significant propellant requirements to arrive at the target. The point immediately prior to the gravity assist (575 days past launch, the purple steeply downward-sloped line in Fig. 4.18b) indicates that a missed thrust event just prior to the gravity assist is the limiting case where the least amount of mass can be delivered to Psyche. A missed thrust event immediately prior to Psyche arrival is similarly sensitive (but slightly less so) to 20-day missed thrust events than the point prior to the gravity assist. The significantly steeper slope of the pre-gravity assist delivered mass vs shutdown length curve demonstrates how much more sensitive the point is than others. Information in Fig. 4.13a similarly shows these two points to be sensitive, but does not show how much mass might be saved if a shorter missed thrust event occurs at those points. Information like this can be used to decide if a lesser guarantee on allowable missed thrust event length can be used at highly sensitive points in the trajectory in order to reduce the amount of contingency propellant needed to account for missed thrust events. With the robust-constrained virtual swarm method, each point can have different constraints on the missed thrust margin, so in this case the 20 day requirement may be relaxed for the pre-gravity assist point if the resulting decrease in contingency propellant mass is deemed more important. Alternative mitigations for missed thrust events at such points may include standing up additional ground support resources during times when the trajectory is less robust and additional spacecraft testing to ensure nominal operations during less robust points in the trajectory. These system-level decisions can be supported by using the virtual swarm method to understand the optimal propellant trade-offs with time, robustness, and other parameters of interest.

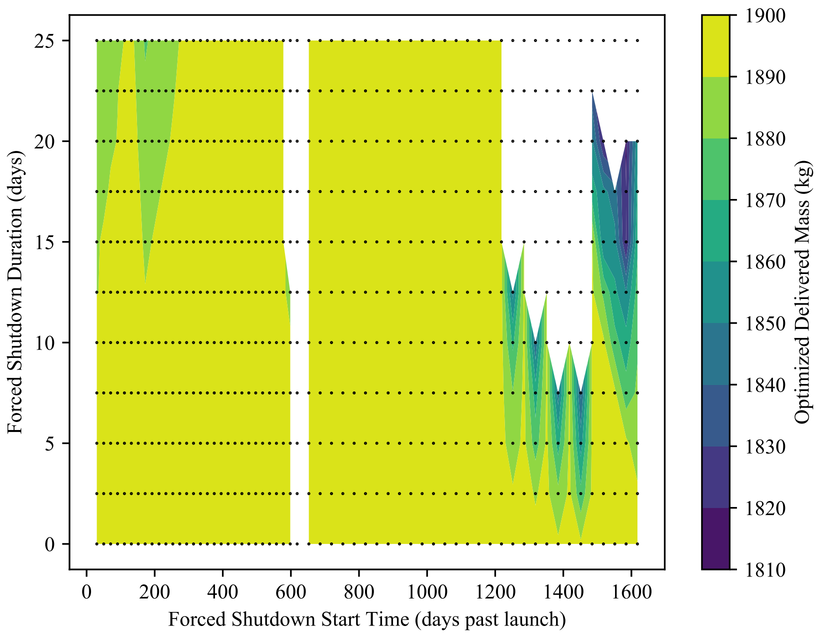


Figure 4.16: Sensitivity of a reference mass-optimal Earth-Psyche trajectory to missed thrust events; recovery trajectories can arrive up to 75 days later than the nominal maximum arrival date

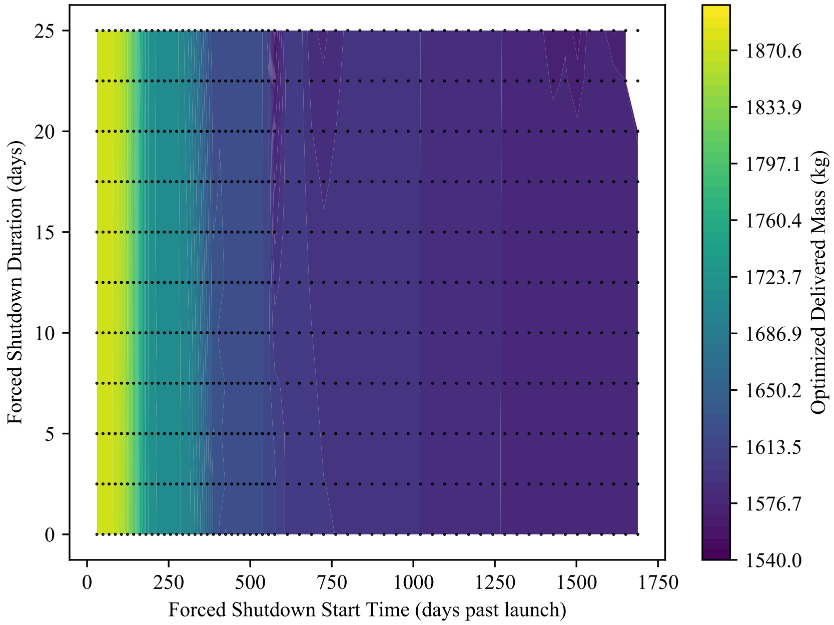
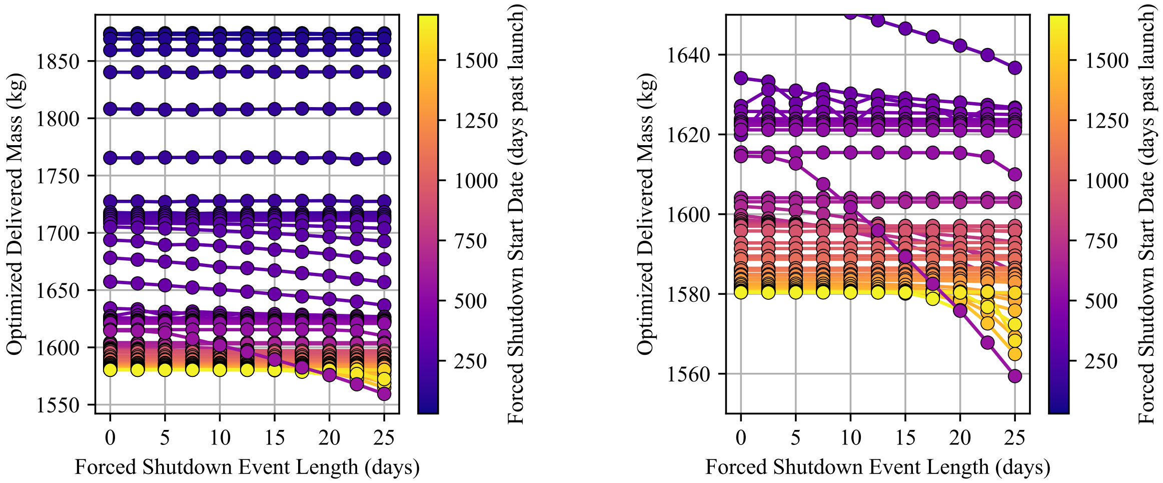


Figure 4.17: Sensitivity of a robust-constrained (20-day robustness, up to 75 days late arrival) Earth-Psyche trajectory to missed thrust events



(a) All optimized delivered mass values shown

(b) Detail view of lower delivered mass values

Figure 4.18: Sensitivity of a robust-constrained (20-day robustness, up to 75 days late arrival) Earth-Psyche trajectory to missed thrust events, line plot views

4.1.7 Discussion

The virtual swarm method that has been developed here is able to describe the optimal trade-offs between γ, $t_{f,\max}$, and delivered mass, without requiring excessive user intervention. Additional trade-offs can be explored in a similar manner by varying other parameters of interest (e.g. launch vehicle, thruster model, etc.) and optimizing the problem. Pareto fronts using the robust-constrained formulation could also consider optimizing other continuous variables not considered here, such as target flyby speed in missions where arrival at the final target is a flyby. The case without a gravity assist, which is less sensitive than the gravity assist case, is particularly amenable to automation. Using a poor initial guess, even with a large number of virtual spacecraft included in the swarm, does not prevent the optimizer from converging on a solution in a relatively short amount of time. Because of this, the method is well suited for trade space exploration. Note that neither the Earth-Mars transfer nor the Psyche transfer examples shown here are specifically selected for amenability to being robust to missed thrust events. Rather, these examples show how a nominal trajectory is transformed once MTRM is constrained along the trajectory. Wide searches using this method can potentially be used to identify specific launch opportunities where the propellant margin penalty for a robust mission as compared to a mass-optimal only mission is less than the penalties found in the examples presented here.

The gravity assist (multi-phase) case presents greater challenges in searching for solutions. While controlling the robustness of the trajectory immediately prior to arrival at the gravity assist introduces high sensitivities into the problem, controlling this point alone does not require too much computational time. However, in this case each virtual spacecraft can potentially have many more impulses, and consequently many more decision variables, than virtual spacecraft spawned in a single-phase trajectory. The introduction of more decision variables creates a wider search space, and additionally slows linear algebra operations performed with the Jacobian matrix of partial derivatives of constraints with respect to decision variables. In this section these linear algebra operations are performed by the NLP solver SNOPT. Further, because automatic differentiation is being used in this work, the inclusion of more decision variables translates into a greater amount of time spent calculating derivatives.

These problems are not insurmountable, and have not prevented automated searches for multi-phase trajectories as was done to generate Fig. 4.15. These barriers do, however, motivate more careful selection of initial guesses to speed up convergence and enable wider searches of the state space as problems become more complex. For more complex problems each virtual spacecraft can be initialized by optimizing its mass or initial coast time based on a static initial state on a nominal reference trajectory. This would provide the optimizer a feasible trajectory as a starting point, instead of it initially having to make the trajectories both feasible and optimal. Complex missions may also benefit from using fewer virtual spacecraft to start with and only adding one or two spacecraft at a time where constraint violations occur (use a small N_w in Algorithm 1). There is a balance to be struck there, however, because while it can take some time to converge on a solution with many virtual spacecraft, there is also a time cost to repeatedly evaluating β along the nominal trajectory. Another strategy is to start with a smaller number of Sims-Flanagan segments to reduce the number of decision variables, then progressively increase the fidelity as solutions are found.

The general computation speed of the virtual swarm method is difficult to describe for a number of reasons. First, varying problems will have varying sensitivities to missed thrust events, and thus will require different numbers of virtual spacecraft to adequately constrain the nominal trajectory. Further, the additional computational complexity introduced by each virtual spacecraft is different depending on its spawn point when using the Sims-Flanagan transcription; virtual spacecraft that spawn earlier will have more control parameters to add to the problem than virtual spacecraft that spawn later. The stochastic monotonic basin hopping method also makes each optimization run non-deterministic, and sometimes a fortuitous hop occurs early in the process that greatly speeds up how soon a solution is found. Because of this stochasticity, reporting single run times from start to finish are of limited utility. To give a rough idea of how computationally expensive adding the additional virtual spacecraft are, however, Table 4.8 includes the time to

Table 4.8: Computation times for each example

Case	Number of Decision Variables	Number of Constraints	Constraint Evaluation Time (ms)	Constraint Jacobian Evaluation Time (ms)
Earth-Mars mass optimal	97	38	0.7004	12.0108
Earth Mars robust-constrained (17 virtual spacecraft)	794	424	9.0954	139.7524
Earth-Mars-Psyche mass optimal	194	78	1.7909	43.3259
Earth-Mars-Psyche robust-constrained (11 virtual spacecraft)	680	358	6.8641	665.0238

evaluate all problem constraints and constraint Jacobian for each example. The constraints are the most expensive part of each iteration in the NLP solver; the objective function and gradient of the objective function are trivial to compute because the objective is directly a decision variable. Each time is an average over 100 trials of the same computation for a more accurate measurement, and a desktop machine with an Intel Core i7 9700k processor is used for the calculations. Finally, note that the NSTOP tool developed for this work was primarily developed with the intention of proving the virtual swarm concept to be viable, and was not specifically optimized for fast evaluation in large trade studies or large individual problems. There are many areas in which the code could be improved to have a faster run time, such as using analytic derivatives or optimizing the speed of the trajectory propagation function which is run many times as the NLP solver iterates.

Also note that models of varying fidelity can be used in the virtual swarm method. For example, virtual spacecraft could use smaller numbers of Sims-Flanagan segments or analytic approximations for low-thrust dynamics. Conversely, where a higher fidelity solution is of interest, larger numbers of Sims-Flanagan segments could be used or the low-thrust equations of motion could be integrated using finite-burn thruster modeling. The correct model to use depends largely on the use case and the overall goals of the mission designer, but the virtual swarm method can support a number of different options. The low-fidelity Sims-Flanagan transcription is the focus of this section to efficiently search a wider space and to provide better initial guesses for higher fidelity optimization problems.

The virtual swarm technique can also theoretically account for an arbitrary number of sequential missed thrust events. Throughout this work only a single missed thrust event is accounted for, but it is possible that a recovery trajectory also has a missed thrust event along its nominal path so a second layer of virtual spacecraft could be used to constrain robustness to the second event. The recovery trajectory from the second missed thrust event could also have a third missed thrust event, so a third layer of virtual spacecraft could be spawned, and so on. The NSTOP tool built to do the analysis in this section can compute solutions with an arbitrary number of missed thrust events, though including too many virtual spacecraft will present challenges to finding optimal and feasible solutions. The exponential growth of the number of virtual spacecraft needed to adequately constrain robustness across multiple missed thrust events will make handling multiple missed thrust events with this method difficult. One potential solution is to use the method as described in this section as a way to provide an initial guess for a series of indirect optimization problems for spacecraft in the swarm. An indirect formulation of the optimization problem is more sensitive and difficult to find solutions for, but has fewer decision variables to optimize and thus may scale to larger numbers of spacecraft better. Once indirect solutions are found for the swarm at one level of missed thrust, the next level of missed thrust robustness might be constrained with either direct- or indirect-optimized virtual spacecraft. Alternatively, it may be most prudent to use real world data [86] to guide the placement of virtual spacecraft to ensure that β is constrained at the most important locations along the trajectory. If the robustness need not be constrained at all points along a trajectory, accounting for multiple missed thrust events with virtual spacecraft becomes much easier. The method presented here takes a deterministic view of the robustness problem, which has the benefit of giving concrete recovery trajectories, but makes accounting for an arbitrary number of missed thrust events more difficult. A fully stochastic optimal control approach would be ideal to constrain successful arrival probability (e.g. 99% chance of successfully arriving given a known distribution of multiple possible missed thrust events), but current methods are not able to incorporate missed thrust events into a stochastic optimal control problem with chance constraints.

This method could also be used to analyze the safe-mode robustness of spacecraft using solar sails for propulsion. During a safe mode event for a spacecraft equipped with a solar sail, the thrust level will not necessarily be zero until the spacecraft resumes normal operations. An interesting avenue to explore is what the ideal attitude or attitude control law would be during a safe-mode event, assuming the spacecraft can maintain some desired attitude instead of simply pointing radially away from the Sun. This safe-mode attitude would need to be balanced against other spacecraft health needs such as pointing solar panels towards the sun. The virtual swarm method might also be adapted to other off-nominal scenarios, such as constraining a crewed spacecraft trajectory such that abort trajectories are always available to return astronauts to safety within a specified amount of time. Finally, the tools developed for this work can extend to real spacecraft swarms with relative ease. Perhaps the most directly applicable real swarm that this method extends to is one in which a low-thrust parent spacecraft launches one or more child spacecraft along its trajectory. These child spacecraft could be destined for the same or different targets than the parent.

4.1.8 Conclusions

The missed thrust recovery margin of a low-thrust spacecraft trajectory can be constrained or optimized using the virtual swarm method developed here. The virtual swarm method simultaneously optimizes a nominal trajectory with its recovery trajectories, enabling the nominal trajectory to be reshaped to account for robustness constraints or objectives. This has the benefit of allowing a mission designer to either optimize delivered mass with path constraints for a minimum missed thrust margin along the way, or it allows a mission designer to optimize worst-case missed thrust recovery margin with a constraint that nominal and recovery trajectories must deliver a minimum amount of mass. The process can be automated to enable efficient search space exploration and the generation of Pareto fronts to give decision makers information about optimal trade-offs between different objectives.

4.2 Indirect Simultaneous Trajectory Optimization: Application to Finite Time Rendezvous and Deployment

4.2.1 Introduction

While much effort has been devoted to optimal active-passive rendezvous trajectories where one spacecraft actively maneuvers to rendezvous with another spacecraft or orbit, there has been limited exploration of the active-active case. In active-active rendezvous, both spacecraft can actively maneuver, and the end goal of rendezvous is more important than the exact location final rendezvous orbit. Prussing has briefly explored the cooperative, active-active rendezvous problem from an indirect optimal control perspective, though without numerically solving for solutions [44]. Feng et al. [101] have actually developed a similar method of solving for optimal cooperative rendezvous trajectories as the one developed in this work, though they focus solely on the rendezvous problem and limit the transfer to two spacecraft. Additionally, they use evolutionary algorithms which are not used here. Previous work has also explored the cooperative rendezvous and optimal deployment problems, but with significant restrictions on maneuver types and the types of rendezvous orbits that could be considered [13] (Section 3.1). We have also previously considered simultaneously optimizing multiple spacecraft trajectories in coupled problems using direct methods [15] (Section 4.1). While the method could quite successfully be applied to the missed thrust problem, the direct transcription required a large number decision variables to fully describe the solution of all spacecraft.

In this work, we develop a method to optimize active-active rendezvous with an unconstrained final orbit using an indirect optimal control perspective. We are specifically interested in the capability of solving the problem with large separation between spacecraft, so fully non-linear Keplerian dynamics are used and we generate full control profiles that achieve rendezvous in finite time. Additionally, we consider the dual problem of finding an optimal deployment orbit for multiple spacecraft; the fuel optimal rendezvous orbit for multiple spacecraft in their final intended orbits can be used as a fuel optimal deployment orbit. A general formulation for N_{SC} spacecraft is found with a specific interest in finding an optimal deployment orbit for several, and potentially many spacecraft.

4.2.2 Problem Formulation

The problem is formulated as an indirect optimal control problem with the objective of minimizing the total fuel mass expended by all spacecraft. For N_{SC} spacecraft system, we have

$$J = \int_{t_0}^{t_f} \left(\sum_{i=1}^{N_{SC}} \frac{T_i}{c_i} \right) dt \tag{4.35}$$

and the masses and thrust levels of each spacecraft are allowed to vary throughout the rendezvous scenario which is of fixed length $t_f - t_0$. While the transfer time is fixed, this fixed time is only an upper bound on the allowable times that the spacecraft can maneuver. The indirect method here allows for terminal coasting to be found if such a strategy is optimal. This also means that the spacecraft are not constrained to arrive at the rendezvous orbit at the same time, nor are they required to depart from a deployment orbit at the same time. To ensure that all spacecraft meet at the same final orbit, there are $N_{SC} - 1$ terminal constraints. These are described as

$$\boldsymbol{g}_j(\boldsymbol{X}_f) = \boldsymbol{x}_{j+1,f} - \boldsymbol{x}_{j,f} = \boldsymbol{0} \quad \text{for} \quad j = 1, 2, ..., N_{SC} - 1 \tag{4.36}$$

where $\boldsymbol{X}$ is the full state of all spacecraft (including masses), $\boldsymbol{x}_i$ is the orbital state (not including mass) of spacecraft i, and the f subscript indicates a value at the final time. Because solving the two-point boundary value problem that arises in the indirect formulation is highly numerically sensitive, we use modified equinoctial elements (MEE) to represent each spacecraft's orbital state. The sensitivity of the final state with respect to the initial state and costates is greatly reduced using MEEs. The states are defined as

$$\boldsymbol{x}_i = \begin{bmatrix} p_i \\ f_i \\ g_i \\ h_i \\ k_i \\ L_i \end{bmatrix} \quad \boldsymbol{X}_i = \begin{bmatrix} \boldsymbol{x}_i \\ m_i \end{bmatrix} \quad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_{N_{SC}} \end{bmatrix} \tag{4.37}$$

where m_i is the mass of spacecraft i. Similarly, the costates are written as

$$\boldsymbol{\lambda}_i = \begin{bmatrix} \lambda_{p_i} \\ \lambda_{f_i} \\ \lambda_{g_i} \\ \lambda_{h_i} \\ \lambda_{k_i} \\ \lambda_{L_i} \\ \lambda_{m_i} \end{bmatrix} \quad \boldsymbol{\lambda} = \begin{bmatrix} \boldsymbol{\lambda}_1 \\ \boldsymbol{\lambda}_2 \\ \vdots \\ \boldsymbol{\lambda}_{N_{SC}} \end{bmatrix} \tag{4.38}$$

and the full state is

$$\boldsymbol{Z}_i = \begin{bmatrix} \boldsymbol{X}_i \\ \boldsymbol{\lambda}_i \end{bmatrix} \quad \boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_1 \\ \boldsymbol{Z}_2 \\ \vdots \\ \boldsymbol{Z}_{N_{SC}} \end{bmatrix} \tag{4.39}$$

Though using Cartesian coordinates for a mass-optimal trajectory results in the elegant primer vector solution for the optimal thrust direction, a similar optimal thrust direction can be found when modified equinoctial elements are used, albeit with a less elegant solution. The relation between the state and the optimal thrust direction is more complex when using MEE.

The optimal control Hamiltonian is

$$H = \sum_{i=1}^{N_{SC}} \left(\frac{T_i}{c_i} + \boldsymbol{\lambda}_i^T \boldsymbol{F}(\boldsymbol{X}_i, \boldsymbol{u}_i) \right) \tag{4.40}$$

where

$$\dot{\boldsymbol{X}}_i = \boldsymbol{F}(\boldsymbol{X}_i, \boldsymbol{u}_i) \tag{4.41}$$

and

$$c_i = I_{sp,i} g_0 \tag{4.42}$$

with $g_0 = 0.00980665$ km/s^2 and the symbol λ used for the adjoints (costates). The Hamiltonian in Eq. (4.40) can be expanded when using modified equinoctial elements to give

$$H = \sum_{i=1}^{N_{SC}} \left(\frac{T_i}{c_i} + \boldsymbol{\beta}_i^T \frac{T_i}{m_i} \hat{\boldsymbol{u}}_i - \lambda_{m_i} \frac{T_i}{c_i} \right) \tag{4.43}$$

with the control $\boldsymbol{u}$ being decomposed into its acceleration magnitude T_i/m_i and direction $\hat{\boldsymbol{u}}$ (control direction is expressed here in radial, tangential, normal components). The vector $\boldsymbol{\beta}_i$ is

$$\boldsymbol{\beta}_i = \boldsymbol{\lambda}_{\boldsymbol{x},i}^T \boldsymbol{F}(\boldsymbol{x}_i, \boldsymbol{u}_i) \tag{4.44}$$

Fully expanded, the vector $\boldsymbol{\beta}_i$ in radial, tangential, and normal components is

$$\boldsymbol{\beta}_i = \begin{bmatrix} \beta_R \\ \beta_T \\ \beta_N \end{bmatrix} \tag{4.45}$$

with

$$\beta_R = \sqrt{\frac{p}{\mu}} \Big(\lambda_{f,i} \sin L - \lambda_{g,i} \cos L \Big) \tag{4.46}$$

$$\beta_T = \sqrt{\frac{p}{\mu}} \left(\frac{\lambda_{f,i}}{w} ((w+1)\cos L + f) + \frac{\lambda_{g,i}}{w} ((w+1)\cos L + g) \right) \tag{4.47}$$

$$\beta_N = \sqrt{\frac{p}{\mu}} \Bigg(\frac{-\lambda_{f,i}}{w} (h \sin L - k \cos L) g + \frac{\lambda_{g,i}}{w} (h \sin L - k \cos L) g + \frac{\lambda_{h,i}}{w} \frac{s^2}{2} \cos L + \frac{\lambda_{k,i}}{w} \frac{s^2}{2} \sin L + \frac{\lambda_{L,i}}{w} (h \sin L - k \cos L) \Bigg) \tag{4.48}$$

The Hamiltonian is linear in the control, so Pontryagin's minimum principle is used to find the optimal control direction

$$\hat{\boldsymbol{u}}_i^* = \frac{-\boldsymbol{\beta}_i}{|\boldsymbol{\beta}_i|} \tag{4.49}$$

The optimal Hamiltonian is then

$$H^* = \sum_{i=1}^{N_{SC}} \left(\frac{T_i}{c_i} - \frac{|\boldsymbol{\beta}_i| T_i}{m_i} - \lambda_{m_i} \frac{T_i}{c_i} \right) \tag{4.50}$$

Given the known optimal thrust direction given in Eq. (4.49), the remaining control variable to find is the thrust magnitude. Each spacecraft has a maximum thrust level $T_{\text{max},i}$, and will thrust at maximum or zero. The switching function S_i that determines which thrust level to provide is

$$S_i \equiv \left(\frac{1 - \lambda_m}{c_i} - \frac{|\boldsymbol{\beta}_i|}{m_i} \right) \tag{4.51}$$

and

$$T_i = T_{\text{max},i} \quad S_i > 0 \tag{4.52}$$

$$T_i = 0 \quad S_i < 0 \tag{4.53}$$

The adjoint dynamics rely on the partial derivative of the Hamiltonian with respect to the state, i.e.

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{X}} \tag{4.54}$$

The mass adjoint dynamics are then

$$\dot{\boldsymbol{\lambda}}_{m,i} = -\frac{|\boldsymbol{\beta}| T_i}{m_i^2} \tag{4.55}$$

The dynamics for the MEE adjoints are quite tedious to derive by hand, so a computational tool was used to symbolically calculate those values.

The transversality conditions dictate that the final mass adjoint for each spacecraft

$$\boldsymbol{\lambda}_{m_{i,f}} = 0 \tag{4.56}$$

The final orbit element adjoints for each spacecraft all must sum to zero, i.e.

$$\boldsymbol{\lambda}_{\boldsymbol{x}_{1,f}} + \boldsymbol{\lambda}_{\boldsymbol{x}_{2,f}} + ... + \boldsymbol{\lambda}_{\boldsymbol{x}_{N_{SC},f}} = \boldsymbol{0} \tag{4.57}$$

This pattern is expected to continue for any number N of spacecraft.

This formulation in MEE is very similar to that found in Junkins and Taheri[102], though is derived slightly differently. A significant difficulty in solving indirect mass optimal orbital trajectories is the discontinuities in thrust when the optimal solution switches from thrust on to off or off to on. When using gradient based techniques to solve the two-point boundary value problem, this switching can make converging quite difficult. As in Junkins and Taheri[102], we smooth the transition from off to on or on to off using the function

$$\sigma(S;\rho) = \frac{1}{2}\left[1 + \tanh\left(\frac{S}{\rho}\right)\right] \tag{4.58}$$

where S is the value of the switching function and ρ is a tunable smoothness parameter. Larger values of ρ result in smoother transitions between thrust levels and less optimal solutions, while smaller values of ρ more closely approximate the instantaneous transition from full thrust to no thrust found in optimal solutions. This approach is taken as opposed to a potential alternative approach where the number of switches between full and zero thrust (or vice versa) is pre-selected; such an approach limits the optimal solutions that can be found. We are instead interested in recovering the optimal number of switches from the solution. While Junkins and Taheri[102] started with larger ρ values and progressively stepped them down, we frequently found that large (larger than about 0.1) values of ρ would lead to convergence on poor solutions dominated by numerical noise in the propagation of near-zero initial adjoints. We instead often used starting values of $\rho < 0.1$ to start with to avoid convergence on such solutions (see next section for more detail).

4.2.3 Solution Method

The problem as described previously is now a two-point boundary value problem (TPBVP) that can be solved by guessing the initial adjoints at t_0 that when integrated forward satisfy all of the problem constraints. Perhaps the easiest method of solution is to simply guess the initial values of the adjoints randomly, propagate forward to t_f, calculate the error in each of the constraints, and then correct the initial values of the unknown adjoints until all constraints are satisfied (single shooting). This is usually quite difficult to accomplish because the constraints can be highly sensitive to the initial adjoints and it is not easy to understand the domain of the adjoints over which random guesses should be made. The region of initial guesses that would converge can also be quite small.

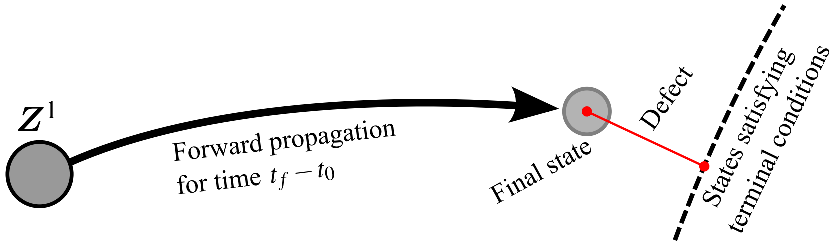

$\boldsymbol{Z}^1$
Forward propagation for time $t_f - t_0$
Final state
Defect
States satisfying terminal conditions

Figure 4.19: Illustration of the single shooting method applied here

In this section however, a single shooting method has been found to be sufficient to solve a surprising number problems when using MEE coordinates; in contrast the problem appeared to be intractable when using cartesian coordinates and single shooting. This method is illustrated in Fig. 4.19. At node 1 ($\boldsymbol{Z}^1$), the decision variables to find are each of the adjoints for each spacecraft at time t_0 for a total of $7N_{SC}$ decision variables. The decision variables must be found such that when the initial condition is integrated forward to the final time, the resulting final state satisfies the terminal conditions (i.e. the "defect" in Fig. 4.19 is zero). A potential alternative approach would be to guess the states (orbit elements and adjoints) at the "final" condition of the rendezvous or deployment orbit and propagate in the other direction. In this alternative case, the number of decision variables is $6 + N_{SC} + 6(N_{SC} - 1) = 7N_{SC}$, the same number of values to guess when shooting forward. The values to guess would be the final orbital state, the final spacecraft masses, and $N_{SC} - 1$ final orbit element adjoint sets. The N_{SC}^{th} set of orbit element adjoints are determined by Eq. (4.57). Here, shooting forward by guessing initial adjoints is used instead of this alternative method.

Given the guessed initial adjoints, the states are propagated forward and the terminal constraints are checked at the final time. To make corrections to the initial adjoints, the Jacobian of the terminal constraints with respect to the initial adjoints is calculated using automatic differentiation with the ForwardDiff.jl[96] package in the Julia programming language. Automatic differentiation provides machine-precision, exact partial derivatives. The initial adjoints are guessed with a random uniform distribution over a domain of $[-0.5, 0.5]$. Because all of the terminal constraints are equality constraints, a multi-variable root finding package NLsolve.jl was used to iteratively correct the initial adjoint guess until convergence or until a set number of iterations has occurred and a new start point is generated.

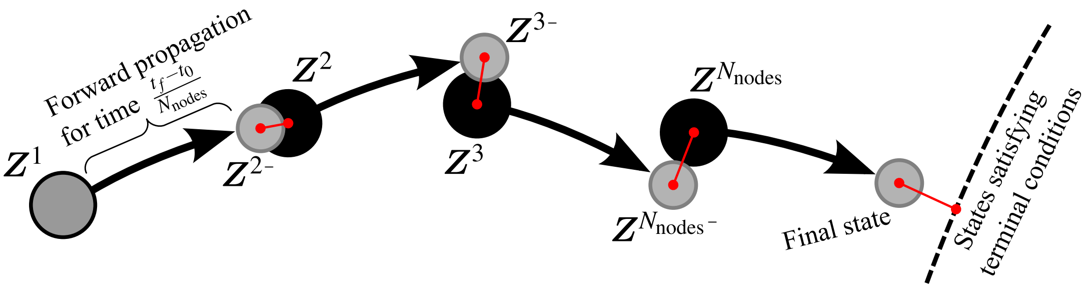

Forward propagation
for time $\frac{t_f - t_0}{N_{\text{nodes}}}$
$\boldsymbol{Z}^1$
$\boldsymbol{Z}^2$
$\boldsymbol{Z}^{2\text{-}}$
$\boldsymbol{Z}^{3\text{-}}$
$\boldsymbol{Z}^3$
$\boldsymbol{Z}^{N_{\text{nodes}}}$
$\boldsymbol{Z}^{N_{\text{nodes}}\text{-}}$
Final state
States satisfying
terminal conditions

Figure 4.20: Illustration of the multiple shooting method applied here

Still, single shooting was found to be unable or inefficient in solving certain problems even using MEE coordinates, so we have also implemented a multiple shooting technique of solving the same problem. The multiple shooting method, illustrated in Fig. 4.20, was especially helpful when solving problems with more than two spacecraft as well as for converging on an initial solution without having a neighboring solution to use as an initial guess. In the multiple shooting method, the set of constraints is less sensitive to changes in the decision variables at the expense of having more constraints to satisfy and decision variables to select. In addition to needing to guess the initial adjoints ($7N_{SC}$ decision variables), the full state $\boldsymbol{Z}$ at each node after node 1 must also be selected ($14N_{SC}$ decision variables per node). The state $\boldsymbol{Z}^j$ is the full state at node j, t^j is the time at node j, and there are N_{nodes} states to propagate. Each state is propagated forward for a time of

$$t^{j+1} - t^j = \frac{t_f - t_0}{N_{\text{nodes}}} \tag{4.59}$$

and the forward propagated state from node j is

$$\boldsymbol{Z}^{(j+1)^-} = \boldsymbol{Z}^j + \int_{t^j}^{t^{j+1}} \dot{\boldsymbol{Z}} dt \tag{4.60}$$

Additionally, there are $N_{nodes} - 1$ continuity or "defect" constraints that must be imposed to ensure that the forward propagation of each node matches the state decision variables of the following node. In Fig. 4.20 these defects are shown with red lines; defect constraints are satisfied when these lines have a length of zero. The defect constraints more formally are

$$\boldsymbol{h}^j = \boldsymbol{Z}^{j^-} - \boldsymbol{Z}^j = \boldsymbol{0} \quad j = 2, 3, ..., N_{\text{nodes}} \tag{4.61}$$

and the terminal conditions (Eqs. (4.36), (4.56-4.57)) are enforced at the state found by forward propagation from the final node. Because of the defect constraints, the $7N_{SC}$ decision variables that are solved at node 1 of the multiple shooting problem would also solve the same problem if it were posed as a single shooting problem. That is, if the multiple shooting transcription is solved, the state at node 1 of the solution can be propagated forward to t_f and satisfy the terminal conditions to solve the necessary conditions for optimality in the indirect problem.

In this work we used the nonlinear programming problem (NLP) solver SNOPT in feasibility mode to find the decision variables that satisfy all constraints. The initial guess is still generated with random numbers if no solution to a similar problem is available. However, the initial guess for the states at each of the intermediate nodes are calculated by propagating the initial state forward in time so that the initial defect constraints are already met for the initial conditions.

This is a naturally sparse problem; instead of assuming that every decision variable affects every constraint, we can specify which decision variables impact which constraints. This speeds up the automatic differentiation process to calculate partial derivatives as well as the iterative process to update the decision variables. For direct optimization problems, sometimes the process of finding which decision variables impact which constraints is done by calculating the full Jacobian for multiple different sets of decision variables. For all calculated Jacobians, the entries that are always zero in the test set can be assumed to be always zero in general. This is only accurate if the number of decision variable sets tested is large enough. Because here we are implementing multiple shooting with an indirect method, we instead manually specify the sparsity pattern and assume that all states from one node may impact the defect constraints at the next node.

For both the single and multiple shooting methods, the general solution process is as follows. First, an "easy" version of the desired problem is solved with one of the shooting methods. Problems can be made easier by shortening the transfer time and ensuring that the amount of thrust available to each spacecraft is high enough to make the problem feasible. If the desired problem requires very large orbit changes, it can also help to initially solve a problem where the orbits of the spacecraft are closer together. Once a solution is found to the easier problem, continuation can be used to then find solutions to the problem of interest. That is, the easier problem solution can be used as an initial guess for more difficult problems. For example, if a transfer time of 12 hours is desired, then the easy problem may be to first solve a problem with a transfer time of 4 hours. Then, the solution of that problem can be an initial guess for a 5 hour transfer problem, then 6 hours, etc.. This process is somewhat easy for a human to perform, but does require human intervention and thus makes fully automating a solution more difficult. For example, as continuation is used to increase the length of the transfer, sometimes the thrusting structure will change and the previous initial guess is not close enough to enable convergence. To remedy this, the smoothing factor ρ in Eq. (4.58) can be increased to get through the difficult transition, then reduced again to make each solution more accurate. Given more experience in the issues that arise in solving problems of interest, small interventions such as that can potentially be implemented in an automated way to make the solution process more hands-off.

4.2.4 Examples

4.2.4.1 Cooperative Rendezvous Examples

A sample summary of a low-thrust cooperative rendezvous solution is shown in Fig. 4.21. The initial orbits of the two 100 kg spacecraft are separated by just 1 km in semimajor axis and

0.5 degrees in inclination; both have the same maximum thrust capability of 0.25 N. Note that both spacecraft thrust at the exact same times, so the switching function and thrust plots only appear to show the second spacecraft's thrusting profile. Time histories of each spacecraft's Keplerian elements are provided to give more physical intuition to how the state evolves over time; the underlying states used in the optimization process are still MEEs.

We have also explored how the solution evolves as the relative maximum thrust level of each spacecraft changes using continuation. In this case, a total of 0.5 N of maximum thrust is available and each case distributes the total of 0.5 N to each spacecraft in different proportions. The example in Fig. 4.21 represents the equally distributed case where both spacecraft are allocated half of 0.5 N for their maximum thrust. Figure 4.22 demonstrates how the final rendezvous orbits evolve as the relative maximum thrust values for each spacecraft are changed. Solid horizontal lines represent the initial state of spacecraft 1, and dotted horizontal lines represent the initial state of spacecraft 2. Because argument of periapsis is not well defined for circular orbits, the optimal rendezvous orbits do not match the initial argument of periapsis for spacecraft 1 or 2 for values $\eta = 1$ or $\eta = 0$ respectively. The final orbit inclination varies linearly, while the final semimajor axis nonlinear relationship with the linearly varied maximum thrust values.

$$T^1_{\max} = (1 - \eta) T^{\text{total}}_{\max} \tag{4.62}$$

$$T^2_{\max} = \eta T^{\text{total}}_{\max} \tag{4.63}$$

If the initial orbits of one of the spacecraft is changed to be elliptic, the features of the optimal transfer change. A single solution for such a case is shown in Fig. 4.24. For this case we similarly varied the relative thrust levels of each spacecraft and solved for the optimal rendezvous orbit for each case; the final rendezvous orbits are all shown in Fig. 4.25 and the thrusting profiles for all of the solutions are superimposed on one another in Fig. 4.26. Note that we no longer find that the thrusting times are the same for all different relative control magnitudes.

4.2.4.2 Deployment Examples

For the deployment cases, we set the "initial" condition at time $t = 0$ to be the final desired orbits for the spacecraft, and propagate backwards in time to satisfy the same terminal conditions as in the rendezvous case. One drawback of doing this is that now the initial mass is unknown, and the final mass is constrained to be the launch mass of the spacecraft (the initial mass at time $t = 0$ will be less than the final mass at the negative time of deployment). Because the example transfers shown here are notional and use fairly small amounts of fuel, we simply set the initial mass to 100 kg as in the rendezvous examples. For trajectories where mass variation is more significant and/or for more accurate trajectories, the terminal condition on the mass adjoint in Eq. (4.56) can be removed and replaced with terminal conditions on the final mass states to equal the initial masses of the spacecraft. Then, the initial masses at time $t = 0$ of the two spacecraft become decision variables to find instead of known quantities.

In Fig. 4.27 we show a sample solution for a low-thrust deployment to a "string of pearls" orbit; the desired orbit for both spacecraft is the same elliptic orbit, but both must be separated by 25 degrees in true anomaly. Spacecraft 1 must be at periapsis, and spacecraft 2 must be 25 degrees ahead of periapsis in true anomaly. Both spacecraft essentially maneuver twice between the deployment and final orbits, and the optimal deployment orbit semimajor axis is close to but not exactly at the desired final semimajor axis. Figure 4.28 shows the same scenario but with an almost impulsive, high thrust control. This significantly changes the optimal trajectories; the first spacecraft now only maneuvers once, while the second spacecraft maneuvers twice. There is also a significant difference in the semimajor axis of the optimal deployment orbit as compared to the final orbit.

Finally, Fig. 4.29 shows a sample solution for a high-thrust three-spacecraft deployment scenario. This is also a "string of pearls" deployment where the spacecraft must have 0, 25, and 50 degree true anomalies respectively. This solution was found using random perturbations to the initial guess from the two spacecraft deployment scenario. Initial convergence was surprisingly quick, and this solution serves as a baseline that enables continuation into other solutions of interest (e.g. longer transfer time, higher/lower thrust, etc.).

4.2.5 Conclusions

This work has shown some preliminary results that indicate that an indirect optimization approach to multi-spacecraft trajectory optimization problems is a promising method. While there are a number of challenges to taking such an approach, we have demonstrated several methods that enable a trajectory designer to overcome these limitations. However, at present, the method does require a fair amount of user intervention to reach a desired solution. The amount of user intervention required can still be reduced with future work. Using the method developed here, we have also been able to uncover some interesting features of the cooperative rendezvous and optimal deployment orbit problems.

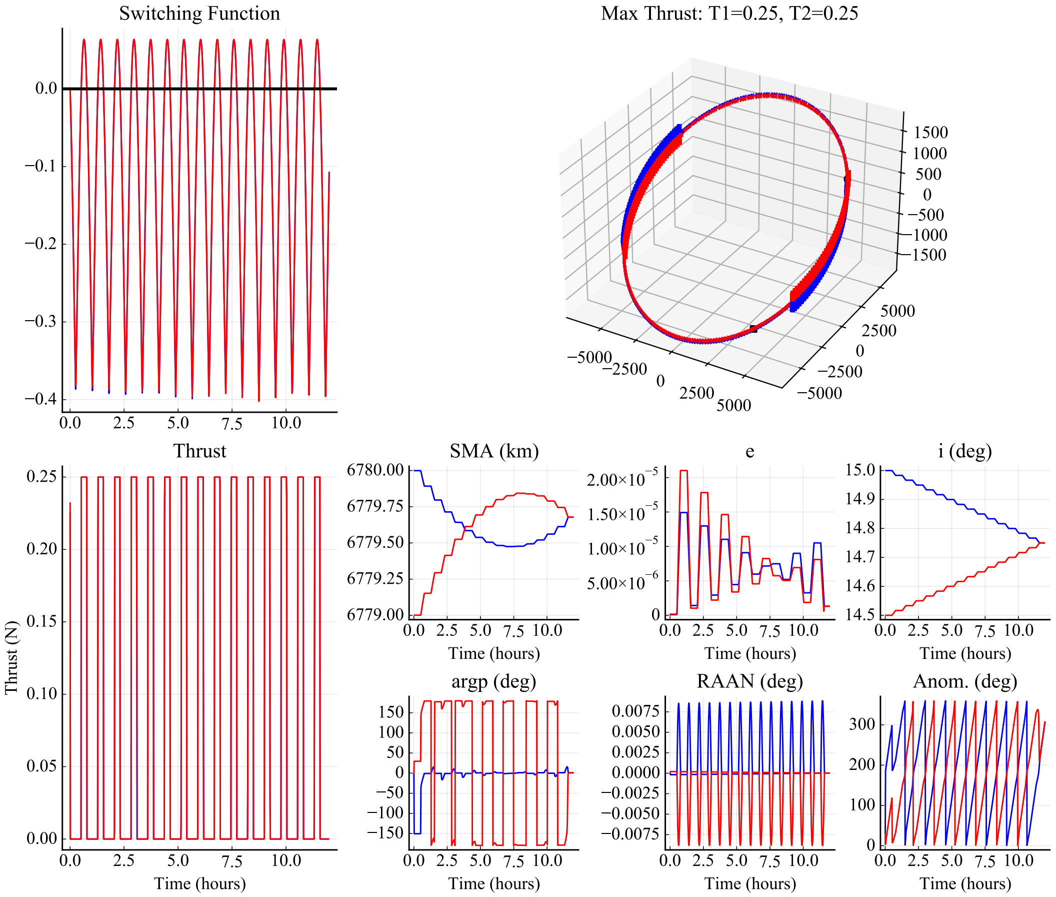


Figure 4.21: Sample optimal cooperative rendezvous solution for spacecraft initially in circular orbits. Blue lines correspond to spacecraft 1, red lines correspond to spacecraft 2. SMA is semimajor axis, e is eccentricity, i is inclination, argp is the argument of perigee, RAAN is the right ascension of the ascending node, and anom. is the true anomaly

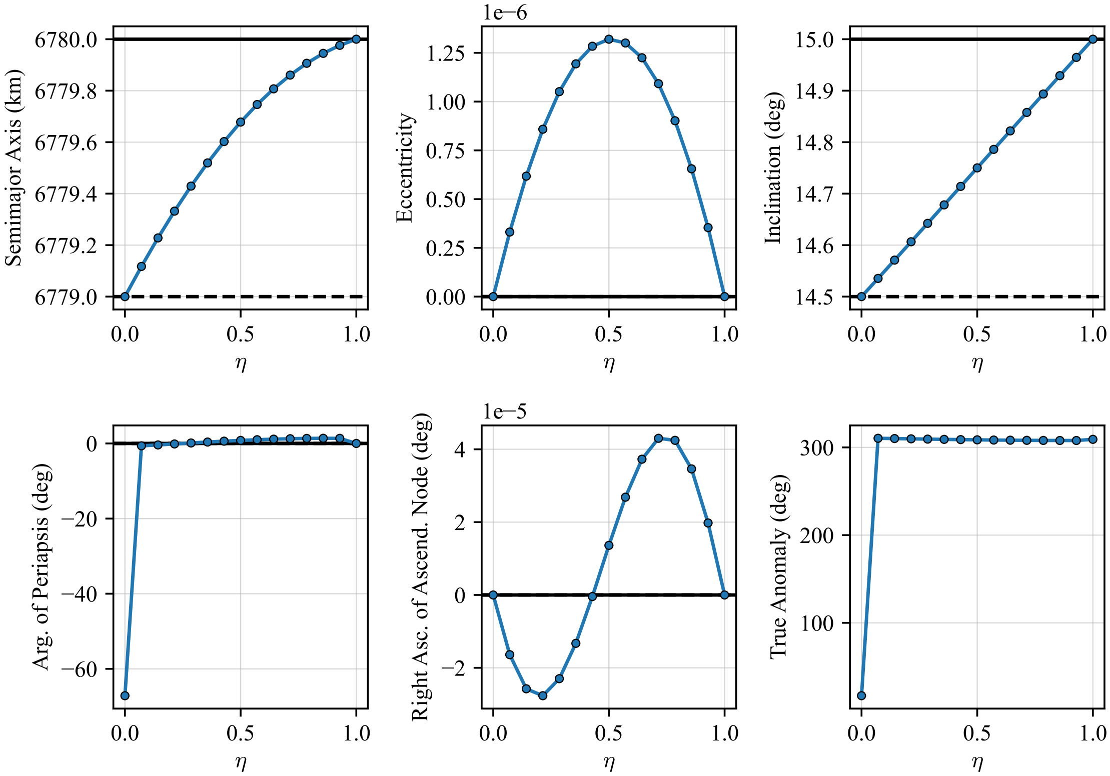


Figure 4.22: Evolution of optimal rendezvous orbit as the the control authority of each spacecraft varies (circular-circular case). The leftmost point on the x-axis has only spacecraft 1 maneuvering, the rightmost point has only spacecraft 2 maneuvering, and the middle point has both spacecraft with the same maximum thrust level of 0.25 N

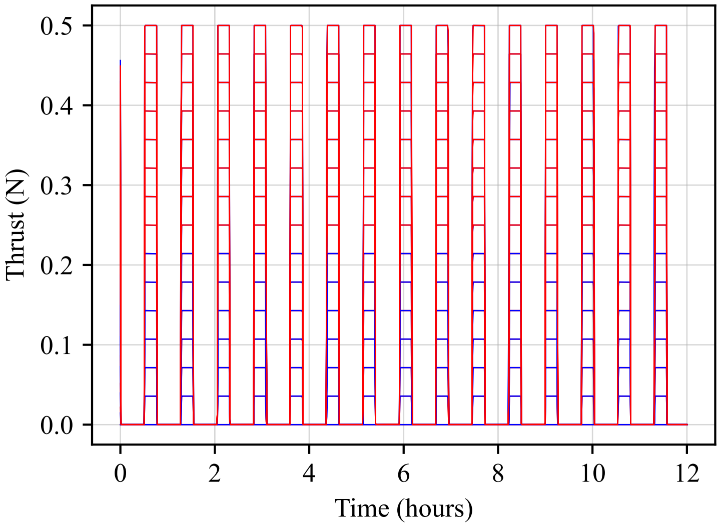


Figure 4.23: Thrusting levels for each spacecraft over time; all thrust profiles

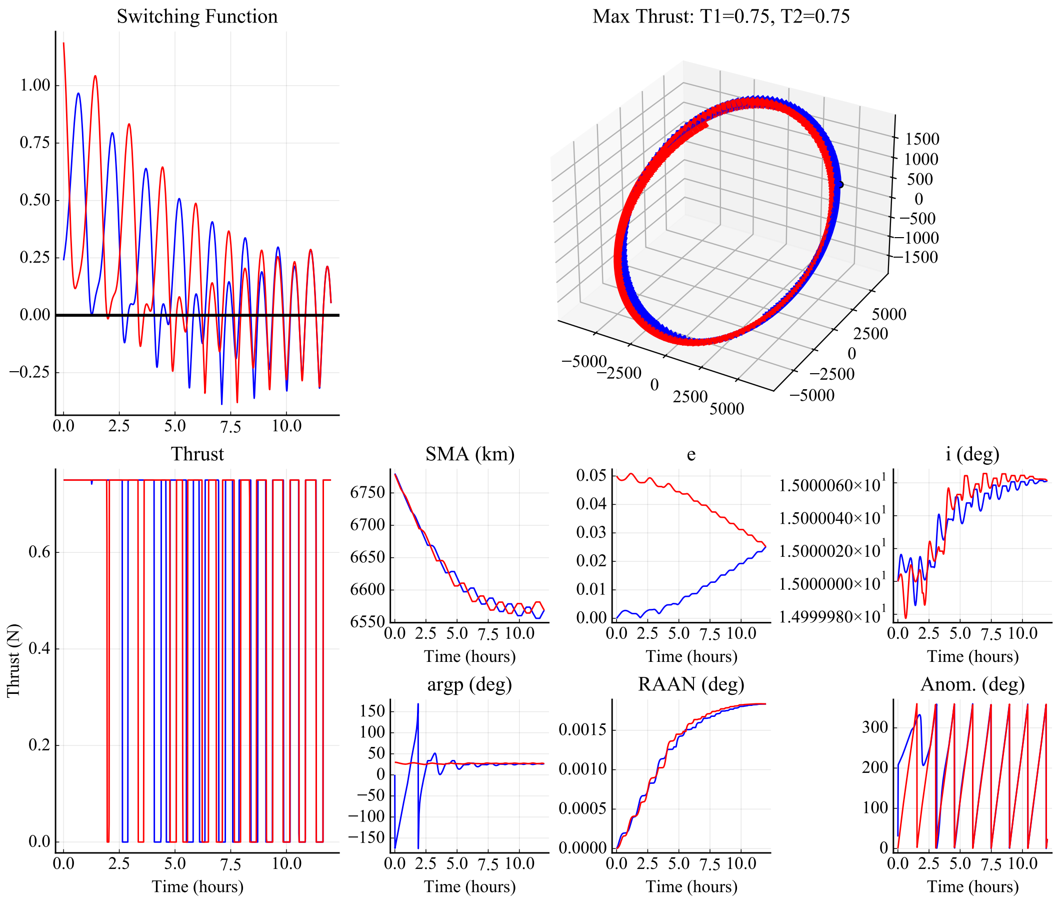


Figure 4.24: Sample optimal cooperative rendezvous solution with spacecraft 1 initially in a circular orbit and spacecraft 2 in an elliptic orbit. Blue lines correspond to spacecraft 1, red lines correspond to spacecraft 2. SMA is semimajor axis, e is eccentricity, i is inclination, argp is the argument of perigee, RAAN is the right ascension of the ascending node, and anom. is the true anomaly

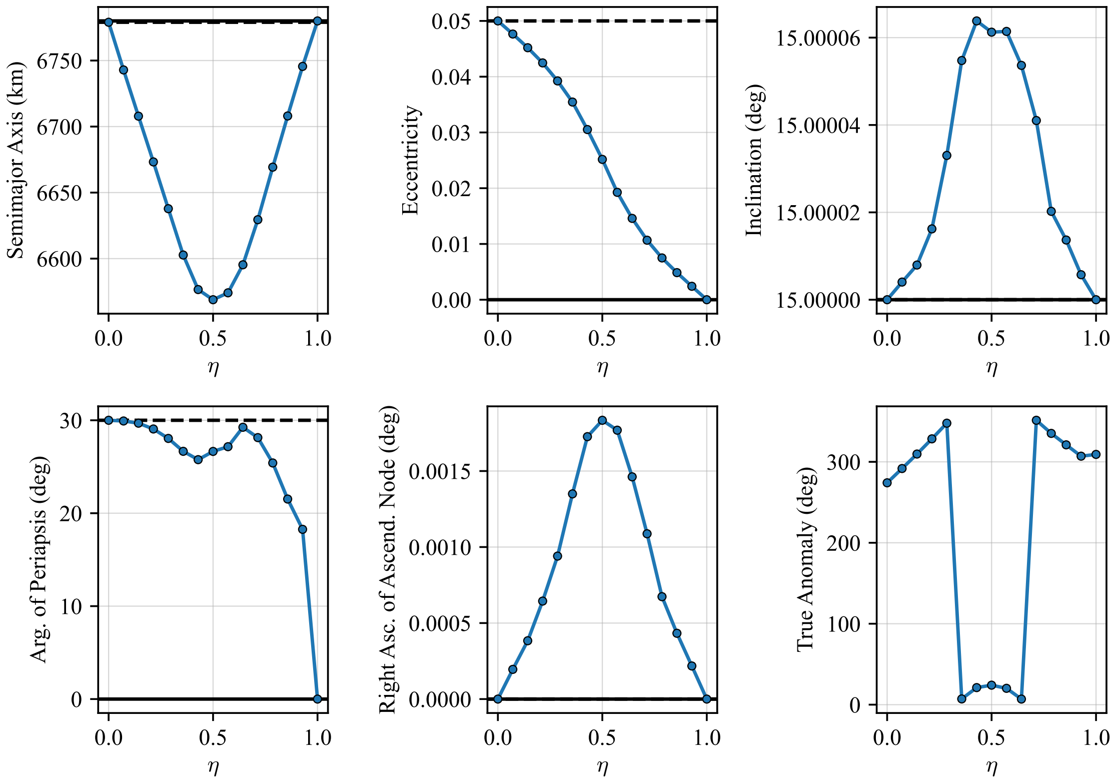


Figure 4.25: Evolution of optimal rendezvous orbit as the the control authority of each spacecraft varies (circular-elliptic case). The leftmost point on the x-axis has only spacecraft 1 maneuvering, the rightmost point has only spacecraft 2 maneuvering, and the middle point has both spacecraft with the same maximum thrust level of 0.25 N

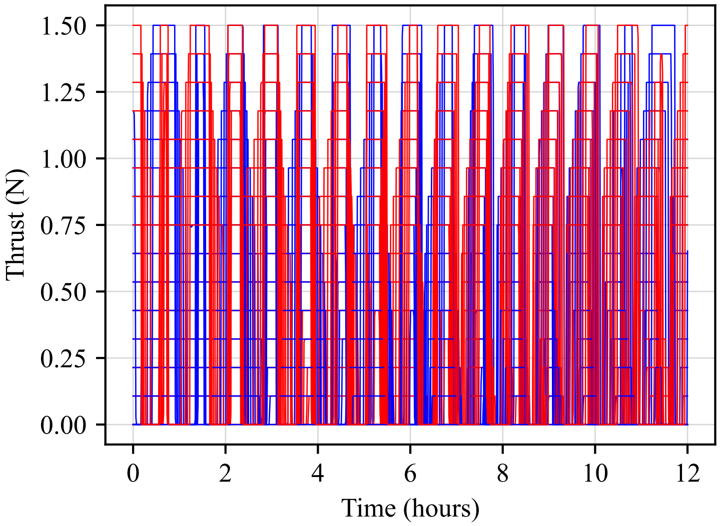


Figure 4.26: Thrusting levels for each spacecraft over time (circular-elliptic case); all thrust profiles for each relative

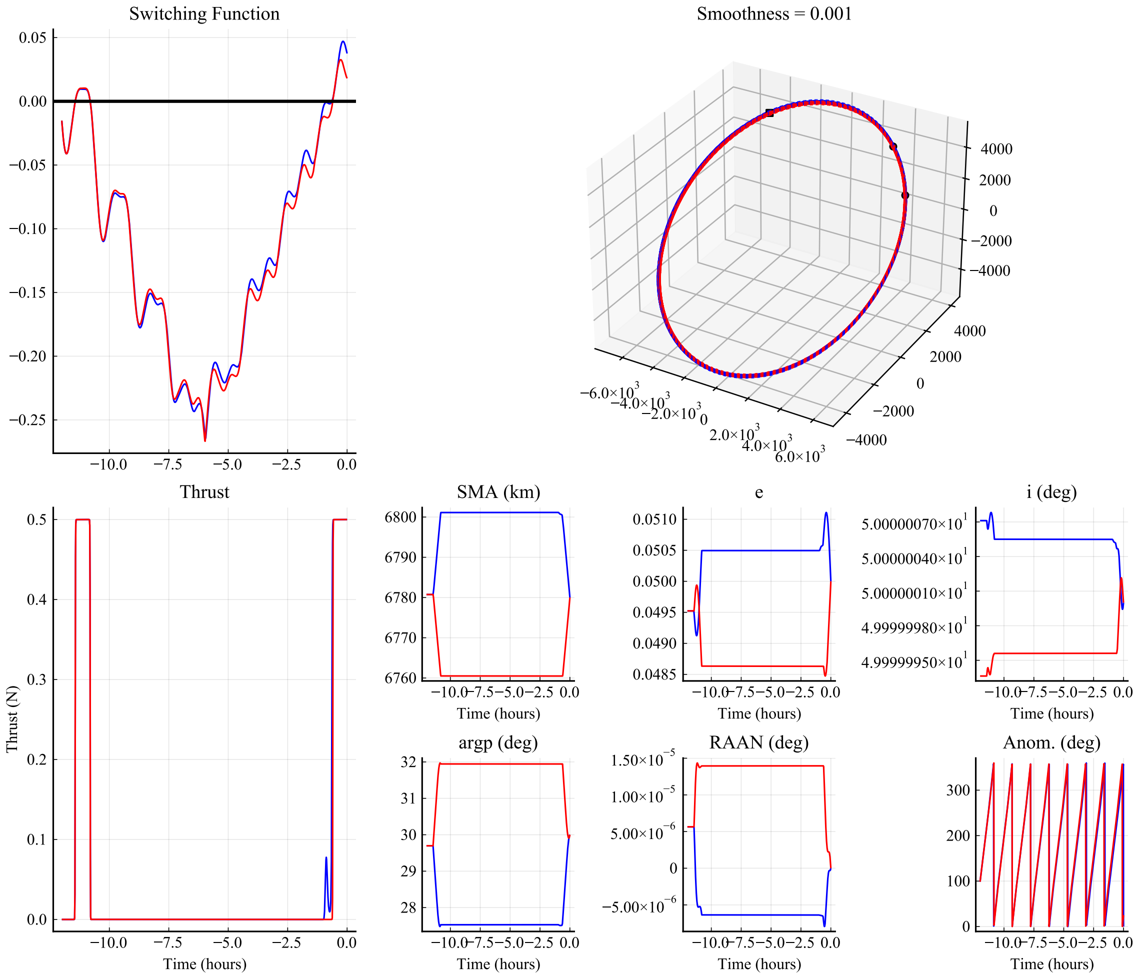


Figure 4.27: Sample two-spacecraft optimal low-thrust deployment solution to a “string of pearls” orbit; both deploy to the same elliptic orbit but are separated by 25 degrees in true anomaly. Deployment orbit is at -12 hours, and the final operational orbits are reached at 0 hours. Blue lines correspond to spacecraft 1, red lines correspond to spacecraft 2. SMA is semimajor axis, e is eccentricity, i is inclination, argp is the argument of perigee, RAAN is the right ascension of the ascending node, and anom. is the true anomaly

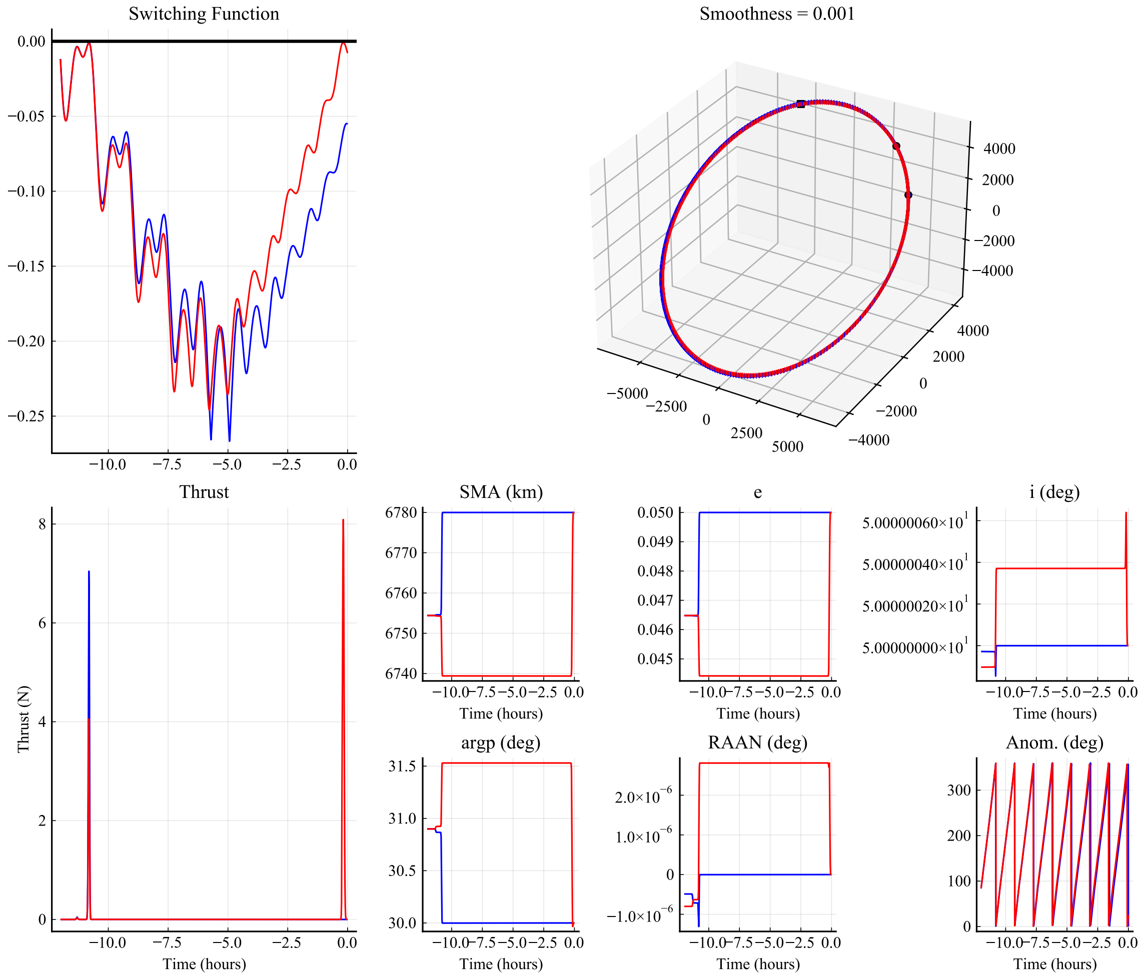


Figure 4.28: Sample two-spacecraft optimal high-thrust deployment solution to a "string of pearls" orbit; both deploy to the same elliptic orbit but are separated by 25 degrees in true anomaly. Deployment orbit is at -12 hours, and the final operational orbits are reached at 0 hours. Blue lines correspond to spacecraft 1, red lines correspond to spacecraft 2. SMA is semimajor axis, e is eccentricity, i is inclination, argp is the argument of perigee, RAAN is the right ascension of the ascending node, and anom. is the true anomaly

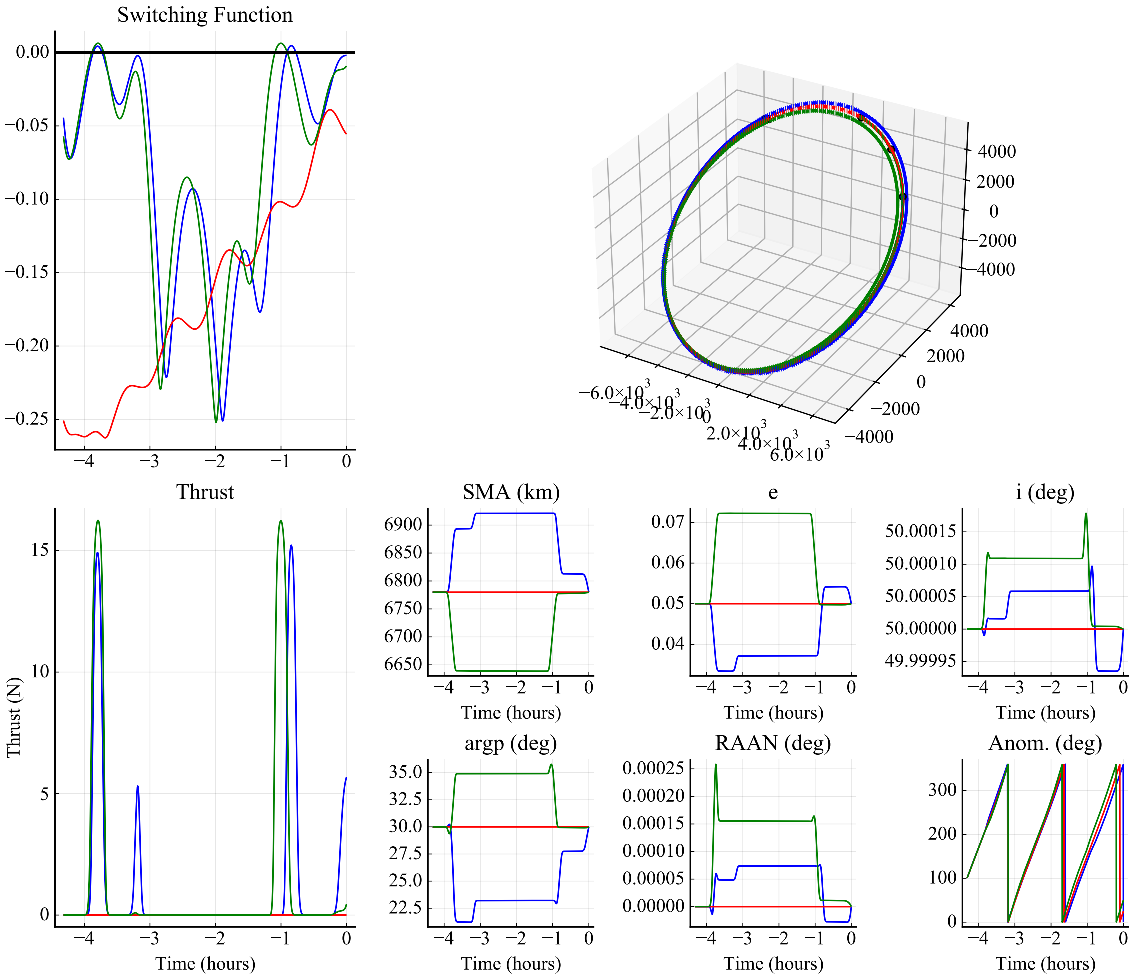


Figure 4.29: Sample optimal high-thrust deployment solution to a "string of pearls" orbit for three spacecraft; all three deploy to the same elliptic orbit but each is separated by 25 degrees in true anomaly. Deployment orbit is at -12 hours, and the final operational orbits are reached at 0 hours. Blue lines correspond to spacecraft 1, red lines correspond to spacecraft 2. SMA is semimajor axis, e is eccentricity, i is inclination, argp is the argument of perigee, RAAN is the right ascension of the ascending node, and anom. is the true anomaly

Chapter 5

Static Game Theory Perspective: Delta-V Based Analysis of Spacecraft Pursuit-Evasion Games

5.1 Introduction

The pursuit-evasion game is a classic example of a non-cooperative multi-agent differential game. In the pursuit-evasion game, a pursuing agent attempts to "capture" an evading agent which attempts to avoid capture. When "capture" is defined as having the cartesian position of both agents match (intercept), such games are not only of interest for scenarios where an optimal evasion strategy is desired to protect an agent from an actively adversarial opponent; they are also of interest in understanding the worst case evasion strategy for avoiding objects that may not be actively pursuing the agent of interest. For the orbital case, such objects include orbital debris, which have uncertain dynamics and can drastically impact an agent's ability to perform its primary functions if proper impact avoidance maneuvers are not made. Impact events also can generate significant debris, which are increasingly complicating space operations. Even those desiring to purposefully disable another spacecraft have an incentive to not impact it and generate debris that complicates operations for themselves as well as others. If "capture" is instead defined as rendezvous (i.e. matching cartesian position and velocity), a pursuit-evasion game is primarily of interest for analyzing fully antagonistic scenarios. Example scenarios addressed by a pursuit-evasion rendezvous game include an unwanted close inspection of a spacecraft by another spacecraft, or an unwanted modification of a spacecraft by another (e.g. disabling instruments, stealing resources). This could be an alternative strategy taken to disable a spacecraft without further polluting space with an impacting event.

The orbital pursuit-evasion game with spacecraft as the agents has been explored in a number of different studies. Recently, Shen and Casalino used an indirect optimal control approach to solve the fully non-linear three-dimensional (spatial) pursuit-evasion differential game [103]. Similarly, Pontani and Conway used semidirect collocation with nonlinear programming (semi-DCNLP) to solve three-dimensional pursuit-evasion games [104]. Hafer et al. addressed a similar scenario as [104] but with the use of sensitivity methods to solve the problem [105]. Stupik, Pontani, and Conway explored pursuit-evasion game solutions with linearized Hill-Clohessy-Wiltshire (HCW) dynamics [106] that apply to more localized problems. Generally, previous approaches have formulated the orbital pursuit-evasion game as an indirect or direct differential game in cartesian (or equivalent) space and then have attempted to numerically solve the problem with various methods. Each of the preceding works and many studies not included here have also used time to intercept as the objective function; the evading spacecraft maximizes time to intercept while the pursuing spacecraft minimizes it. A time-optimal game is equivalent to a mass- and ΔV-optimal game under the assumption that both spacecraft will continuously thrust at maximum levels throughout the entire game; this assumption was proven to be a feature of a time-optimal game in [103]. Notably, in such a scenario, coasting arcs cannot be used to enable a more efficient strategy because efficiency is not a primary goal of that formulation. Some authors have also used a feedback control law approach to explore pursuit-evasion games as well [107, 108]. Another common approach is for the pursuer and evader to minimize and maximize the terminal miss distance respectively [109, 110]. Each of these studies are ultimately focused on final capture, where the pursuing agent reaches some threshold distance from the evading agent or it matches the evading spacecraft's spatial position or full state. Previous work on spacecraft evasion has also considered minimum fuel or minimum ΔV evasion maneuvers, but with a "pursuer" that cannot react to evasive maneuvers [111, 112, 113]. Instead, a given radial distance from the nominal interception point is used for evasion in those studies.

The term "capture" representing the goal of the pursuing spacecraft could mean reaching a relative distance within a missile's blast radius, reaching the same position as the evading spacecraft (intercept), or reaching the same position and velocity as the evading spacecraft (rendezvous). While a majority of previous work has focused on intercept or close approach in position space, this work focuses on rendezvous as the final goal. The use of “capture” throughout will mean rendezvous while the term “intercept” will be used to refer to the more commonly seen intercept goal for pursuit-evasion games.

Though expressing objective functions in terms of continuous variables is incredibly useful in enabling analysis of problems, we note that the fundamental objective for each agent in a pursuit-evasion game is often binary [38]. The evading spacecraft wishes to avoid capture, while the pursuing spacecraft wishes to achieve capture. In this section, a longer-term, strategic approach to pursuit-evasion games is explored with specific interest in understanding optimal strategies for spacecraft with a large initial separation. An approach solely based on time to intercept, as in much of the previous work, may potentially cause either agent to needlessly use fuel when natural dynamics can still be used to their advantage in minimizing fuel use, especially when the initial separation between the agents is large. From a perspective of simply avoiding or achieving capture, maintaining more fuel reserves can make this binary objective more likely for either agent and may allow either agent to have more fuel available to complete other objectives beyond the pursuit-evasion game. In this work ΔV is used instead of instead of time to capture when constructing and evaluating each agent’s cost function, and impulsive maneuvers are also assumed. This formulation prioritizes the preservation of fuel over the minimization/maximization of time to intercept, and is important in enabling solutions where long coasting periods using natural dynamics are allowed. Such a scenario may perhaps be the first phase of a pursuit-evasion game where a pursuing spacecraft allows natural dynamics to do much of the work in coming close to the evading spacecraft, after which in the terminal phase it switches to a continuously on thrust control law based on optimizing time to intercept to ensure it can capture the evading spacecraft. This approach enables a high-level analysis of an evading spacecraft’s risk posture and evasion strategies given different initial orbital placements of a pursuing and evading spacecraft.

Given this focus on using ΔV-based objectives for the pursuing and evading spacecraft, we use ΔV optimal orbit transfers to explore the pursuit-evasion problem where a pursuing spacecraft attempts to match the state of (rendezvous with) an evading spacecraft. Specifically, time-free, impulsive orbit transfers are used because they represent the lowest possible ΔV cost transfers that can be used. In a point mass gravity field perturbed by secular J_2 effects, ΔV must only be expended to match a target semi-major axis (a), eccentricity (e), and inclination (i). Because the J_2 perturbation causes a secular drift rate in right ascension of the ascending node (RAAN, Ω), argument of periapsis (ω), and anomaly (M), any desired final values of those orbit parameters can theoretically be achieved by waiting until natural dynamics perturb them into certain values at the start of the $a - e - i$ transfer such that $\Omega - \omega - M$ drift into the final desired values at the end of the $a - e - i$ transfer [43, 13]. Note that the time for this to occur may be infinite (allowable for time-free transfers), and this will only occur if no two angular rates are equal or commensurate with one another. While in a full pursuit-evasion game some balance between time optimality and ΔV optimality must be found, in this work we specifically focus on the ΔV optimal aspect of the problem as part of an early fuel-conserving phase of the pursuit-evasion problem when the separation between the spacecraft is large in $a - e - i$ space. Thus, the orbit transfers used here only consider costs to achieve a certain $a - e - i$, and the evading spacecraft attempts to ensure that the pursuing spacecraft cannot match its $a - e - i$. We note that these optimal costs can always be achieved to reach a given set of six orbit elements if the spacecraft orbit is placed in a properly phased initial orbit. In a full rendezvous pursuit-evasion game, additional ΔV will potentially need to be expended to match or avoid the matching of $\Omega - \omega - M$, but the strategies described here address considerations for the orbit elements $a - e - i$.

Section 5.2 summarizes the two-impulse optimal orbit transfers used in this section to calculate the cost for either agent to transfer to any other orbit. In Sec. 5.3 we discuss the pursuit-evasion game when the total amount of ΔV available to each spacecraft is known, and the reachable sets of both spacecraft can be used to determine if capture is possible. Though only ΔV costs for changing a, e, and i are considered here, reachable sets based on those transfers provide conservative keep-out zones for the evading spacecraft to avoid. If the evading spacecraft remains outside of the pursuing spacecraft's reachable set, it guarantees that capture cannot occur. Section 5.4 explores evasion strategies in the case where the pursuing spacecraft's available ΔV is unknown. In that case, it is beneficial to explore what maneuvers are the most efficient for the evader that would allow it to potentially escape the pursuing spacecraft's reachable set. Finally, in Sec. 5.5, potential endgame strategies in the context of the maneuvers used here are briefly considered. These strategies focus on the terminal phase where the pursuing spacecraft is close to achieving capture.

5.2 Time- and Orientation-Free Optimal Transfers and Reachable Sets

This section makes use of the two-impulse optimal time-free transfers with Keplerian dynamics that are described in more detail in Section 3.1.2 and [13]. Notably, these transfers are ΔV optimal two-impulse transfers in $a-e-i$ space when transfer time is unconstrained and secular J_2 perturbations to Keplerian dynamics are present. We equivalently use periapsis radius (q) and apoapsis radius (Q) in place of a and e because optimal transfers occur along lines of constant q and Q. Here, we briefly summarize the relevant aspects of the transfers as they pertain to this pursuit-evasion analysis. In short, these transfers are a combination of elliptical Hohmann transfers for changes in semimajor axis (a) and eccentricity (e) as well as "dogleg" maneuvers to distribute plane changes in inclination (i) across both impulses for a minimum ΔV transfer from one orbit in $a-e-i$ space to another. Component maneuvers used here are also shown to be allowable ΔV optimal maneuvers for creating optimal reachable sets as proven in [43]. Maneuvers occur tangentially at periapsis and apoapsis, which must also be at a node ($\omega = 0$). Even if assumptions were relaxed to allow impulses at any point and in any direction outside of those restrictions, such maneuvers would not be strictly ΔV optimal for an orbit transfer [13] or ΔV optimal for maximizing the reachable set of orbits [43]. If transferring to an orbit with a larger apoapsis radius, the elliptical Hohmann transfer sequence is to first change apoapsis to the desired level with a tangential maneuver at periapsis, then to change periapsis to the desired level with a tangential maneuver at apoapsis. If transferring to an orbit with a smaller apoapsis, the sequence is reversed; first periapsis is changed and then apoapsis is changed. For a purely planar transfer with no inclination change, each of the two impulses can be split into an infinite number of smaller impulsive maneuvers as long as the entirety of one apsis change is performed before maneuvering to change the other apsis. For example, if a transfer to a larger apoapsis orbit first requires an apoapsis change maneuver of 50 m/s, this can be split into five maneuvers of 10 m/s at different passes of periapsis all performed before making any changes to the periapsis radius. In the three-dimensional case with a dogleg maneuver, however, all of the maneuvering must be performed within two impulses. If strictly ΔV optimal maneuvers of three or more impulses were allowed, maneuvers through infinity (e.g. bi-parabolic transfer) would be permitted and would result in unbounded reachable sets. Even for orbit transfers where bounded three-impulse transfers are optimal, the required ΔV is large enough to allow escape to infinity as shown in the survey work of Gobetz [41]. Allowing such transfers would result in unbounded reachable sets, while the present analysis is limited to the bounded reachable set case. The assumptions on maneuver number and placement are of course highly restrictive, especially in a pursuit evasion game, but these maneuvers are considered to be part of a high-efficiency phase where spacecraft are attempting to avoid or chase one another in a maximally efficient way, before perhaps transitioning to a more aggressive posture with fewer limits on maneuver types if necessary.

The planar transfer cost, reproduced here from Eqs. (3.1 - 3.11), is denoted as

$$\Delta V'_{total} = \Delta V'_1 + \Delta V'_2 \tag{5.1}$$

where

$$\Delta V'_1 = |V_{1t} - V_0| \tag{5.2}$$

$$\Delta V'_2 = |V_f - V_{2t}| \tag{5.3}$$

and subscript "t" is for the transfer orbit. V_0 is the velocity on the initial orbit immediately prior to the first impulse, and V_f is the velocity on the final orbit immediately after the second impulse. The optimal two-impulse transfer sequence and ΔV cost relies on the values of the initial and final radius of apoapsis Q_0 and Q_f, where q_0 and q_f are the initial and final periapsis radii. For $Q_f > Q_0$,

$$V_0 = \sqrt{\frac{2\mu}{q_0 + Q_0}\frac{Q_0}{q_0}} \tag{5.4}$$

$$V_{1t} = \sqrt{\frac{2\mu}{q_0 + Q_f}\frac{Q_f}{q_0}} \tag{5.5}$$

$$V_{2t} = \sqrt{\frac{2\mu}{q_0 + Q_f}\frac{q_0}{Q_f}} \tag{5.6}$$

$$V_f = \sqrt{\frac{2\mu}{q_f + Q_f}\frac{q_f}{Q_f}} \tag{5.7}$$

For $Q_f \leq Q_0$,

$$V_0 = \sqrt{\frac{2\mu}{q_0 + Q_0}\frac{q_0}{Q_0}} \tag{5.8}$$

$$V_{1t} = \sqrt{\frac{2\mu}{q_f + Q_0}\frac{q_f}{Q_0}} \tag{5.9}$$

$$V_{2t} = \sqrt{\frac{2\mu}{q_f + Q_0}\frac{Q_0}{q_f}} \tag{5.10}$$

$$V_f = \sqrt{\frac{2\mu}{q_f + Q_f}\frac{Q_f}{q_f}} \tag{5.11}$$

Again, Eqs. (5.2-5.3) can be rewritten without absolute value operators with knowledge of the initial and final states. This is easier to see if Eqs. (5.4-5.11) are rewritten in a different form. For example, Eq. (5.4) can be rewritten as

$$V_0 = \sqrt{2\mu\left(\frac{1}{q_0} - \frac{1}{q_0 + Q_0}\right)}, \tag{5.12}$$

The general expression for the optimal ΔV cost to transfer from one elliptical orbit $\boldsymbol{A}_0 = [q_0, Q_0, i_0]$ to another elliptical orbit $\boldsymbol{A}_f = [q_f, Q_f, i_f]$ in a different plane using a two-impulse sequence is then

$$\Delta V_{total}(\boldsymbol{A}_0, \boldsymbol{A}_f) = \Delta V_{0,f} = \Delta V_1 + \Delta V_2 = \\ \sqrt{V_{1t}^2 + V_0^2 - 2V_{1t}V_0\cos(\eta\Delta i)} + \sqrt{V_f^2 + V_{2t}^2 - 2V_{2t}V_f\cos((1-\eta)\Delta i)} \tag{5.13}$$

This equation, reproduced from Eq. (3.13) combines the planar and out-of-plane maneuvers for each impulse using the law of cosines. Fractional inclination changes at each of the two impulses

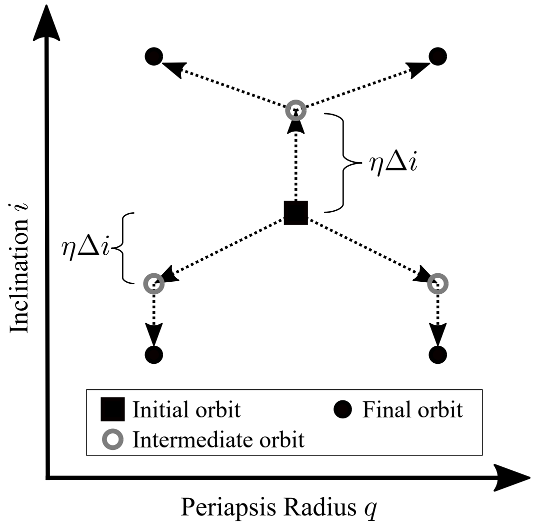


Figure 5.1: 3D orbit transfer paths

are dictated by the parameter η, which is defined in the range $0 \leq \eta \leq 1$. The transfer paths in $q - i$ space are shown in Fig. 5.1, where the initial spacecraft orbit is shown as a black square and paths to various final orbits (solid black circles) are shown with arrows. The open circles in gray indicate the intermediate orbits used in each transfer; they are the orbits reached after the first impulse. For a view of the transfers in $q - Q$ space, see the inset in Fig. 5.4.

Also note that

$$\Delta i = |i_f - i_0|. \tag{5.14}$$

The η^* that minimizes ΔV_{total} can be found by taking the partial derivative of Eq. (5.13) with respect to η and equating it to 0. The result,

$$F = \frac{\partial \Delta V_{total}}{\partial \eta} = \frac{\Delta i V_0 V_{1t} \sin(\eta^* \Delta i)}{\Delta V_1} - \frac{\Delta i V_f V_{2t} \sin((1-\eta^*)\Delta i)}{\Delta V_2} = 0 \tag{5.15}$$

(reproduced from Eq. (3.15) cannot be explicitly solved for η. The implicit function can, however, be solved using a variety of techniques; in this work a bisection method is used to find the correct value for each individual transfer.

5.2.1 Reachable Sets

A reachable set here is described as the set of all possible orbits a given spacecraft can reach given its initial orbit $\boldsymbol{A}_0$ and maximum available $\Delta V_{\max}$. Because transfers here are considered in $q - Q - i$ space, a reachable set R can be written as

$$R(\boldsymbol{A}_0, \Delta V_{\max}) = \{\boldsymbol{A}_f | \Delta V_{total}(\boldsymbol{A}_0, \boldsymbol{A}_f) \leq \Delta V_{\max}\} \tag{5.16}$$

The reachable set is considered here to include all orbits that cost less than $\Delta V_{\max}$ for the spacecraft to transfer to. That is, the set includes both the extremal surface as well as the interior of the surface.

While reachable sets can be quite difficult to calculate, the analytic expression for the optimal transfer cost in Eq. (5.13) can be used to relatively quickly calculate reachable sets for spacecraft in $a - e - i$ space. To generate these reachable sets, costs for each spacecraft to transfer to each orbit in a discretized grid of final orbits is calculated. Using that grid, the geometric surface that represents the reachable set of orbits in $q - Q - i$ space can be interpolated given the amount of ΔV available to each spacecraft. The surface defined by any given $\Delta V_{\max}$ value must be found by interpolating between grid points where the actual ΔV has been calculated, so higher-resolution grids with more points will produce more accurate reachable set approximations. This reachable set computation method gives similar results to the method described in [43], with a major difference being that these sets assume a two-impulse structure for orbit transfers (see [13] for more detail) while [43] does not. The benefit of this method is a faster, more flexible method of computation that makes the analysis of multiple reachable sets easier and enables some analytic insight that will be seen in later sections.

5.3 Known Reachable Set Approach to Pursuit-Evasion Games

First, a scenario where the evading spacecraft has knowledge of the pursuing spacecraft's available ΔV is considered. If a reachable set of orbits can be calculated for a spacecraft given its initial orbit and its available ΔV, the reachable sets dictate whether an evading spacecraft can be guaranteed to successfully avoid capture in a pursuit-evasion game. Given the evading spacecraft's reachable set of orbits E and the pursuing spacecraft's reachable set of orbits P, and assuming that both spacecraft maneuver optimally, evasion cannot be guaranteed if

$$E \subseteq P \tag{5.17}$$

(E is a subset of P). If

$$P \subset E \tag{5.18}$$

(P is a proper subset of E), then the evading spacecraft is guaranteed to have the ability to avoid capture. If

$$P \cap E = \emptyset \tag{5.19}$$

(the intersection of P and E is empty, P and E are disjoint), then the evading spacecraft is guaranteed to never be captured because there is no orbit reachable to both agents. In short, the condition $P \cap E \neq \emptyset$ is a necessary but not sufficient condition for capture to occur; alternatively, the evading spacecraft has a guaranteed ability to escape capture if any part of its reachable set lies outside of the reachable set of the pursuing spacecraft ($E \nsubseteq P$). A minimum ΔV evasion strategy for guaranteed avoidance of capture would be for the evading spacecraft to shrink its reachable set (by reducing ΔV) until exactly one orbit remains outside of the reachable set of the pursuing spacecraft. That orbit would be a minimum ΔV evasion orbit that guarantees that the pursuing spacecraft cannot reach it. The intersection of both reachable sets $P \cap E$ can be used as a safe keep-out zone for the evading spacecraft to ensure that it cannot be captured.

Notably, in the case where E is a subset of P, we can definitively state that capture is possible when the pursuit-evasion game is a sequential, two-stage game. In the sequential, two-stage game, the evading spacecraft is first allowed to make a two-impulse move to transfer orbits, after which the second spacecraft can make a two-impulse move to attempt to match the orbit of the evading spacecraft. Given this scenario, the evading spacecraft can only maneuver to some other orbit that lies within the reachable set of the pursuing spacecraft, so the pursuing spacecraft can always reach the evading spacecraft. A differential game formulation is better able to account for more complex interactions between the pursuing and evading spacecraft when both spacecraft are allowed to maneuver any number of times and to maneuver in response to actions taken by the other. When any number of maneuvers are allowed and when additional factors such as uncertainty in the other agent's intent and state are introduced, the evader may be able to avoid capture in certain scenarios even when the initial reachable set $E_0 \subset P_0$. Pioneering work by Isaacs [38] presents a useful discourse on the nature of differential games as compared to discrete games with small numbers of actions/maneuvers/"moves" allowed for each player. A differential game approach is out of scope for this section.

Figure 5.2 shows various examples of what the reachable sets look like for different amounts of ΔV available to each spacecraft. Figure 5.2a shows a disjoint example where $P \cap E = \emptyset$ (capture is impossible), Fig. 5.2b shows an example where $E \subset P$ (evasion cannot be guaranteed), Fig. 5.2c shows an example where $P \cap E \neq \emptyset$ (evasion can be guaranteed), and Fig. 5.2d shows an example where $P \subset E$ (evasion can be guaranteed).

5.4 Unknown Pursuer Reachable Set Approach to Pursuit-Evasion Games

Alternatively, if the evading spacecraft has no knowledge of the pursuing spacecraft's total available ΔV, it is important to understand how an evading spacecraft can most efficiently expend ΔV to avoid the pursuing spacecraft. In this scenario, given a pursuing spacecraft with initial orbital state $[q_{p,0} \quad Q_{p,0} \quad i_{p,0}]$ and an evading spacecraft with initial orbital state $[q_{e,0} \quad Q_{e,0} \quad i_{e,0}]$ we define the objective function

$$J = \Delta V_{p0,e_f} - \Delta V_{e0,e_f} \tag{5.20}$$

which the evading spacecraft seeks to maximize. $\Delta V_{p0,e_f}$ is the cost for the pursuer to transfer from its initial orbit to the final orbit e_f of the evading spacecraft, and $\Delta V_{e0,e_f}$ is the cost for the evader to transfer from its initial orbit to its final orbit. The effort to maximize this cost function is in effect an effort to make capture as expensive as possible for the pursuing spacecraft. However, the inclusion of the second term $-\Delta V_{e0,e_f}$ balances the cost for the evader to transfer to orbit e_f

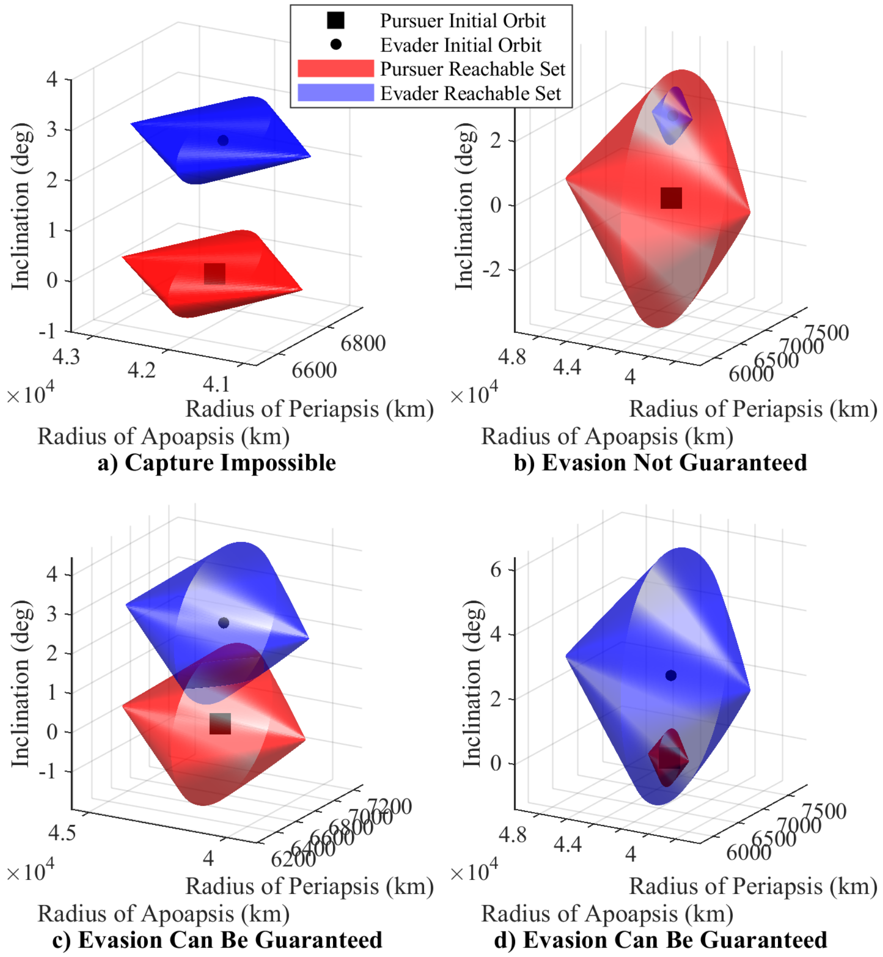


Figure 5.2: Various scenarios for pursuit-evasion reachable sets.

against the increase in $\Delta V_{p_0,e_f}$ that e_f provides. In other words, given that both spacecraft will be expending ΔV as e_f is varied, the evading spacecraft seeks to expend less ΔV to outlast the pursuing spacecraft. The evading spacecraft successfully "outlasts" the pursuer if the pursuing spacecraft runs out of fuel prior to rendezvous occurring. For this reason, the relative amount of total fuel used by each spacecraft in kilograms is not of primary importance; the focus on ΔV is in the context of attempting to make an agent use more or less of its total control authority ΔV_{total}.

The pursuer, in contrast, is potentially only attempting to minimize its cost $\Delta V_{p_0,e_f}$ without regard for the amount of fuel expended by the evading spacecraft. Alternatively, it may seek to to minimize Eq. (5.20), which could potentially cause the evading spacecraft to run out of fuel and enable capture. The pursuer is, however, at the mercy of the evader in that it must go to whichever orbit the evading spacecraft transfers to. In this section, we explore both linearized and non-linear approaches to evaluating strategies for the evading spacecraft to maximize J. In the linearized analyses, we take partial derivatives of J as e_f is varied away from e_0 to understand local evasion strategies. In the non-linear analysis, we perform grid searches and evaluate J across a wide array of e_f values to understand optimal evasion strategies.

5.4.1 Linearized Pursuit-Evasion Analysis

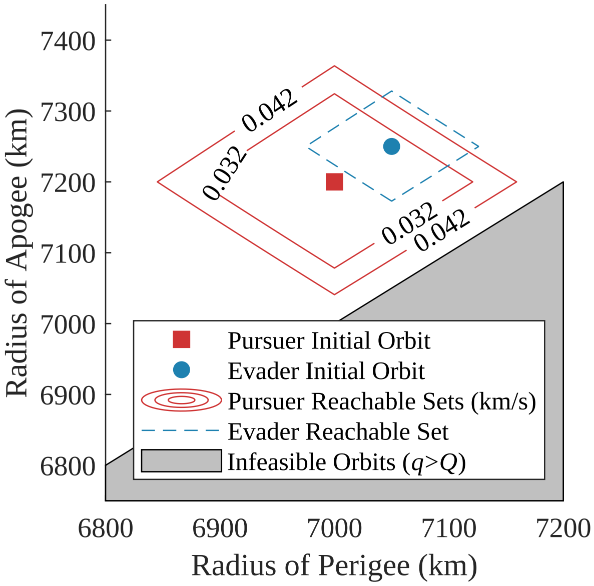


Figure 5.3: Potential evasion scenario showing possible extents of pursuer reachable set.

First order insight into the pursuit-evasion game can be gained by taking partial derivatives of the analytic expressions for the ΔV cost required for each spacecraft to change orbits. From the evading spacecraft's perspective, the strategies of interest are incremental maneuvers it can perform at its initial orbit to most efficiently escape the pursuing spacecraft. Because it is already known that ΔV optimal in-plane maneuvers will be to change periapsis or apoapsis, these are the candidate evasion strategies with which to compare the pursuing spacecraft's ΔV costs to. An example scenario is shown in Fig. 5.3 where the incremental maneuvers found in this section will give the optimal evasion direction for the evading spacecraft to move in order to possibly exit the unknown reachable set of the pursuing spacecraft. In the case where the evading spacecraft maneuvers to change apoapsis, the relation of interest is

$$J_Q \equiv \left.\frac{\partial J}{\partial Q_f}\right|_{Q_f=Q_{0,e}} = \left.\frac{\partial \Delta V_p'}{\partial Q_f}\right|_{Q_f=Q_{0,e}} - \left.\frac{\partial \Delta V_e'}{\partial Q_f}\right|_{Q_f=Q_{0,e}} \tag{5.21}$$

where a "p" subscript indicates the ΔV magnitude of the pursuing spacecraft, and a "e" subscript indicates the ΔV magnitude of the evading spacecraft. Q_f for the pursuing spacecraft is the evading spacecraft's initial apoapsis $Q_{0,e}$, because nominally the pursuing spacecraft must reach the initial orbit of the evading spacecraft. Thus, to explore incremental maneuvers from its nominal state, Q_f for the evading spacecraft is its own initial apoapsis $Q_{0,e}$. The quantity J_Q represents how much more ΔV a pursuing spacecraft must expend than an evading spacecraft to exact the same change in radius of apoapsis from the nominal $Q_f = Q_{0,e}$. A positive value indicates an advantage for the evading spacecraft, a negative value indicates an advantage for the pursuing spacecraft, and a zero value indicates that both spacecraft maneuver with the same efficiency in that direction. Similarly, the relation of interest when the evading spacecraft maneuvers to change periapsis is

$$J_q \equiv \left.\frac{\partial J}{\partial q_f}\right|_{q_f=q_{0,e}} = \left.\frac{\partial \Delta V_p'}{\partial q_f}\right|_{q_f=q_{0,e}} - \left.\frac{\partial \Delta V_e'}{\partial q_f}\right|_{q_f=q_{0,e}} \tag{5.22}$$

The sign of the quantity J_q has the same interpretation as the quantity J_Q, with $q_f = q_{0,e}$, but is directed in the periapsis direction. The analytic expressions for the optimal ΔV cost of in-plane maneuvers allow the direct evaluation of these partial derivatives. However, because the costs in Eqs. (5.2-5.3) have absolute value operators, the partials will be evaluated separately for each case of initial evading spacecraft orbit location relative to the initial orbit of the pursuing spacecraft. The final values for J_Q and J_q are shown in Fig. 5.4 for different relative positions of pursuer and evader, as well as for different evasion directions. Note that the zero value directions all occur in the same directions as optimal transfers noted in the bottom right corner of Fig. 5.4.

For example, if a spacecraft is pursuing an evading spacecraft with a smaller q and larger Q, the the zero value direction is along the direction of the second transfer arrow for an optimal transfer to an orbit with a smaller q and larger Q. Importantly, no positive values of J_Q and J_q are found, so the evading spacecraft can at best maneuver as efficiently as the pursuing spacecraft. Figure 5.5 summarizes Fig. 5.4 by showing the evasion directions where J_q and J_Q are zero for different relative initial orbits between the pursuer and evader.

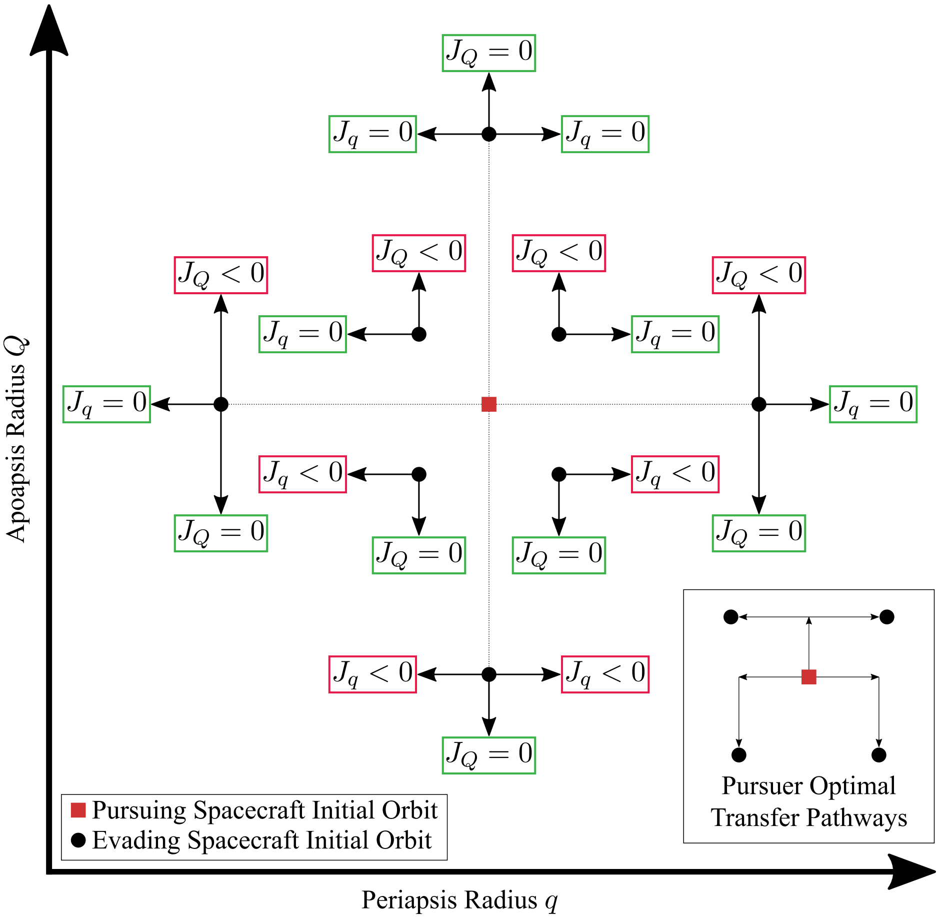


Figure 5.4: J_Q and J_q values for different evasion strategies and different initial pursuer and evader orbits

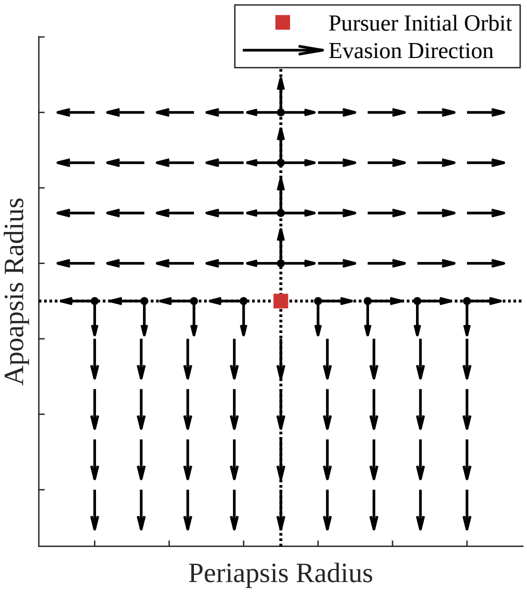


Figure 5.5: Strategy for the evading spacecraft given different initial conditions

5.4.1.1 Cases where the evading spacecraft has a larger apoapsis

When $Q_e > Q_p$, $J_q = 0$ for all relative values of q_e and q_p. That is, both spacecraft maneuver equally as efficiently in the periapsis direction. To exact one unit of change in periapsis radius, both spacecraft must expend the same amount of ΔV. The parameter J_Q has also been numerically found to be negative in general for $Q_e > Q_p$, though in the special case where $q_e = q_p$, $J_Q = 0$. Thus, in general the pursuing spacecraft has an advantage in the apoapsis direction. To exact one unit of change in apoapsis radius, the pursuing spacecraft must expend less ΔV than the evading spacecraft.

To find the values J_q and J_Q, partial derivatives of the cost J are derived and then evaluated. Partial derivatives of orbital speed at periapsis and apoapsis with respect to Q and q are frequently found in the equations of interest, so we first define some convenience functions. Partial derivatives of orbital speed at apoapsis V_Q are defined as

$$\Gamma_1(q,Q) \equiv \frac{\partial V_Q}{\partial q} = \frac{\dfrac{\mu}{(q+Q)^2}}{\sqrt{2\mu\left(\dfrac{1}{Q} - \dfrac{1}{Q+q}\right)}} = \frac{\dfrac{\mu}{(q+Q)^2}}{V_Q} \tag{5.23}$$

$$\Gamma_2(q,Q) \equiv \frac{\partial V_Q}{\partial Q} = \frac{\mu\left(\dfrac{1}{(q+Q)^2} - \dfrac{1}{Q^2}\right)}{\sqrt{2\mu\left(\dfrac{1}{Q} - \dfrac{1}{Q+q}\right)}} = \frac{\mu\left(\dfrac{1}{(q+Q)^2} - \dfrac{1}{Q^2}\right)}{V_Q} \tag{5.24}$$

and partial derivatives of orbital speed at periapsis V_q are defined as

$$\Gamma_3(q,Q) \equiv \frac{\partial V_q}{\partial Q} = \frac{\dfrac{\mu}{(q+Q)^2}}{\sqrt{2\mu\left(\dfrac{1}{q} - \dfrac{1}{Q+q}\right)}} = \frac{\dfrac{\mu}{(q+Q)^2}}{V_q} \tag{5.25}$$

$$\Gamma_4(q,Q) \equiv \frac{\partial V_q}{\partial q} = \frac{\mu\left(\dfrac{1}{(q+Q)^2} - \dfrac{1}{q^2}\right)}{\sqrt{2\mu\left(\dfrac{1}{q} - \dfrac{1}{Q+q}\right)}} = \frac{\mu\left(\dfrac{1}{(q+Q)^2} - \dfrac{1}{q^2}\right)}{V_q} \tag{5.26}$$

For cases where $q_f \geq q_0$ and $Q_f > Q_0$, Eqs. (5.2-5.3) and (5.4-5.7) give the optimal cost. However, the absolute value operators in Eqs. (5.2-5.3) can be removed, because the quantities within the operators will be positive for these values of q and Q. Evaluating the partial derivatives gives the following results:

$$\frac{\partial \Delta V'_{total}}{\partial q_f} = \Gamma_1(q_f, Q_f) \tag{5.27}$$

$$\frac{\partial \Delta V'_{total}}{\partial Q_f} = \Gamma_3(q_0, Q_f) + \Gamma_2(q_f, Q_f) - \Gamma_2(q_0, Q_f) \tag{5.28}$$

For cases where $q_f < q_0$ and $Q_f \geq Q_0$, the only change is to Eq. (5.3).The absolute value operators in Eqs. (5.2-5.3) can be removed, but Eq. (5.3) must be multiplied by -1 to ensure that $\Delta V'_2$ remains positive. Re-evaluating the partial derivatives gives the following results

$$\frac{\partial \Delta V'_{total}}{\partial q_f} = -\Gamma_1(q_f, Q_f) \tag{5.29}$$

$$\frac{\partial \Delta V'_{total}}{\partial Q_f} = \Gamma_3(q_0, Q_f) - \Gamma_2(q_f, Q_f) + \Gamma_2(q_0, Q_f) \tag{5.30}$$

Notice that Eqs. (5.27) and (5.29) have no dependence on initial condition, which is the only factor that differentiates ΔV_{total} between the pursuing and evading spacecraft when both are changing q_f in the same direction. Thus, for $Q_e \geq Q_p$, $J_q = 0$, and both spacecraft must expend ΔV at the same rate while the evading spacecraft maneuvers to change its periapsis radius away from the pursuing spacecraft. The value of J_Q is less clear in this case; J_Q is evaluated as follows:

$$J_Q = \frac{\partial \Delta V_p'}{\partial Q_f}\bigg|_{\substack{q_0=q_p,Q_0=Q_p,\\ q_f=q_e,Q_f=Q_e}} - \frac{\partial \Delta V_e'}{\partial Q_f}\bigg|_{\substack{q_0=q_e,Q_0=Q_e,\\ q_f=q_e,Q_f=Q_e}} \tag{5.31}$$

Because the initial and final orbits are the same for $\frac{\partial \Delta V_e'}{\partial Q}\Big|_{\substack{q_0=q_e,Q_0=Q_e,\\ q_f=q_e,Q_f=Q_e}}$, the value of the partial derivative is different based on whether Q_f is being increased or decreased. Here it is presupposed that $Q_e > Q_p$, so Q_e must be increased to move in a direction that exits the pursuing spacecraft's reachable set. For this reason, Eqs. (5.28) or (5.30) can be used to evaluate Eq. (5.31). For the partial derivative of the evading spacecraft's total ΔV, Eqs. (5.28) and (5.30) are equivalent because $q_0 = q_f = q_e$, but the partial derivative of the pursuing spacecraft's total ΔV changes depending on whether $q_e > q_p$ (use Eq. (5.28)) or $q_e < q_p$ (use Eq. (5.30)).

Evaluating J_Q for $q_e > q_p$ and $Q_e > Q_p$ gives:

$$J_Q = \Gamma_3(q_p, Q_e) + \Gamma_2(q_e, Q_e) - \Gamma_2(q_p, Q_e) - \Gamma_3(q_e, Q_e) \tag{5.32}$$

noting that the Γ_2 terms for the evading spacecraft cancel each other out.

For $q_e < q_p$ and $Q_e > Q_p$, the expression for J_Q is:

$$J_Q = \Gamma_3(q_p, Q_e) - \Gamma_2(q_e, Q_e) + \Gamma_2(q_p, Q_e) - \Gamma_3(q_e, Q_e) \tag{5.33}$$

While inspecting Eqs. (5.32) and (5.33) does not immediately reveal any clear understanding of the sign of J_Q in the general case, it is clear that $J_Q = 0$ for $q_e = q_p$. This is because terms 2 & 3 as well as terms 1 & 4 cancel out in both equations. Thus, when $q_e = q_p$, the evading spacecraft can maneuver either to increase/decrease periapsis or increase apoapsis without disadvantaging itself.

For low-Earth-orbit test cases where $q_e < q_p$ or $q_e > q_p$, J_Q appears to always be negative in numeric test cases. Despite considerable effort, we were unable to find any cases where $J_Q > 0$. This indicates that, in general, for an evading spacecraft with a larger apoapsis than a pursuing spacecraft, it should maneuver to change its periapsis away from the periapsis of the pursuing spacecraft. That is, it should increase periapsis if $q_e > q_p$, or decrease periapsis if $q_e < q_p$. In the special case where $q_e = q_p$ and $Q_e > Q_p$, the evading spacecraft can either maneuver to increase apoapsis or it can maneuver to change periapsis in either direction. Other evasion strategies are certainly possible, but such strategies will either move the evading spacecraft closer to the pursuing spacecraft in $Q - q$ space, or it will require the evading spacecraft to expend more ΔV than a pursuing spacecraft would need to cause the same changes in orbit.

5.4.1.2 Cases where the evading spacecraft has a smaller apoapsis

For cases $Q_e \leq Q_p$ the results essentially mirror the previous case. That is, $J_Q = 0$ for all relative values of q_e and q_p and J_q has been numerically found to be negative in general for $Q_e < Q_p$. However, for the special case where $Q_e = Q_p$, $J_q = 0$. Thus, in general the pursuing spacecraft has an advantage in the periapsis direction.

For cases where $q_f \geq q_0$ and $Q_f \leq Q_0$, Eqs. (5.2-5.3) and (5.8-5.11) give the optimal cost. However, under these constraints, Eqs. (5.2-5.3) can be written without the absolute value operators because the results will still be positive. Then, evaluating the partial derivatives gives the following results:

$$\frac{\partial \Delta V'_{total}}{\partial q_f} = \Gamma_1(q_f, Q_0) + \Gamma_4(q_f, Q_0) - \Gamma_4(q_f, Q_f) \tag{5.34}$$

$$\frac{\partial \Delta V'_{total}}{\partial Q_f} = -\Gamma_3(q_f, Q_f) \tag{5.35}$$

For cases where $q_f < q_0$ and $Q_f < Q_0$, the absolute value operators in Eqs. (5.2-5.3) are removed, and Eq. (5.2) is multiplied by -1 before the partial derivatives are evaluated. This gives the following results:

$$\frac{\partial \Delta V'_{total}}{\partial q_f} = -\Gamma_1(q_f, Q_0) + \Gamma_4(q_f, Q_0) - \Gamma_4(q_f, Q_f) \tag{5.36}$$

$$\frac{\partial \Delta V'_{total}}{\partial Q_f} = -\Gamma_3(q_f, Q_f) \tag{5.37}$$

Similar to the value of J_q for $Q_e \geq Q_p$, Eqs. (5.35) and (5.37) show that for $Q_e < Q_p$, $J_Q = 0$. Finding J_q is more complex for $Q_e < Q_p$, and is evaluated as follows:

$$J_q = \left.\frac{\partial \Delta V'_p}{\partial q_f}\right|_{\substack{q_0 = q_p, Q_0 = Q_p, \\ q_f = q_e, Q_f = Q_e}} - \left.\frac{\partial \Delta V'_e}{\partial q_f}\right|_{\substack{q_0 = q_e, Q_0 = Q_e, \\ q_f = q_e, Q_f = Q_e}} \tag{5.38}$$

In this case, the sign of $\dfrac{\partial \Delta V_e}{\partial q_f}$ will change depending on whether q_e is greater than or less than q_p, but it will have the same magnitude because $Q_0 = Q_f = Q_e$. Evaluating J_q for $q_e > q_p$ and $Q_e < Q_p$ gives:

$$J_q = \Gamma_1(q_e, Q_p) + \Gamma_4(q_e, Q_p) - \Gamma_4(q_e, Q_e) - \Gamma_1(q_e, Q_e) \tag{5.39}$$

When $q_e < q_p$ and $Q_e < Q_p$, evaluating J_q is slightly different because the evading spacecraft will be decreasing periapsis to move away from the pursuing spacecraft. Equation (5.36) gives the partial derivative for positive changes in q, so when using Eq. (5.36) to evaluate J_q it must be multiplied by -1 to account for the fact that both spacecraft will be making negative changes in q. Thus, evaluating J_q for $q_e < q_p$ and $Q_e < Q_p$ where the evading spacecraft is decreasing periapsis gives:

$$J_q = -\Gamma_1(q_e, Q_p) + \Gamma_4(q_e, Q_p) - \Gamma_4(q_e, Q_e) + \Gamma_1(q_e, Q_e) \tag{5.40}$$

Similar to what was found for J_Q when $Q_e > Q_p$, J_q always appears to be negative. This indicates that the most prudent direction for the evader to move is to decrease apoapsis.

All results for J_q and J_Q are summarized in Table 5.1.

Table 5.1: Summary of partial derivative expressions

	$q_e < q_p$	$q_e > q_p$
$Q_e > Q_p$	$J_q = 0$, $J_Q = \Gamma_3(q_p, Q_e) - \Gamma_2(q_e, Q_e) +$ $\Gamma_2(q_p, Q_e) - \Gamma_3(q_e, Q_e)$	$J_q = 0$, $J_Q = \Gamma_3(q_p, Q_e) + \Gamma_2(q_e, Q_e) -$ $\Gamma_2(q_p, Q_e) - \Gamma_3(q_e, Q_e)$
$Q_e < Q_p$	$J_Q = 0$, $J_q = -\Gamma_1(q_e, Q_p) + \Gamma_4(q_e, Q_p) -$ $\Gamma_4(q_e, Q_e) + \Gamma_1(q_e, Q_e)$	$J_Q = 0$, $J_q = \Gamma_1(q_e, Q_p) + \Gamma_4(q_e, Q_p) -$ $\Gamma_4(q_e, Q_e) - \Gamma_1(q_e, Q_e)$

5.4.1.3 Limiting cases

When the evading spacecraft has the same apoapsis as the pursuing spacecraft, it has two evasion strategies where either J_Q or J_q is zero. Because it is on the edge between the two cases $Q_e < Q_p$ and $Q_e > Q_p$ explored above, it has the same properties as the $Q_e < Q_p$ case when decreasing apoapsis, and the same properties as the $Q_e > Q_p$ when changing periapsis away from the pursuing spacecraft. When decreasing apoapsis, $J_Q = 0$ for the same reasons discussed for $Q_e < Q_p$. When moving periapsis away from the pursuing spacecraft, $J_q = 0$ for the same reasons discussed for $Q_e > Q_p$. However, when testing values of J_Q as the evader increases apoapsis, we find that $J_Q < 0$.

As Q_e approaches ∞, we find that

$$\lim_{Q_e \to \infty} J_Q = 0 \tag{5.41}$$

regardless of the relative values of q_e and q_p.

5.4.1.4 Discussion

This analysis has demonstrated that the evading spacecraft has very specific directions along which it can maneuver as efficiently as the pursuing spacecraft; these directions coincide with the optimal orbit transfer directions for the evading spacecraft to move to an orbit farther away from the pursuing spacecraft in $Q - q$ space. If the evading spacecraft maneuvers in a manner inconsistent with the optimal transfer pathways, then it will be maneuvering in a non-ΔV optimal manner and shrinking its reachable set of orbits. Thus, we expect these alternative maneuver directions to not be ideal candidates for ΔV optimal evasion.

General strategies for pursuing and evading spacecraft can be designed based on these results. A limitation not previously discussed is the line of circular orbits where $Q = q$. Note that as seen in Fig. 5.4 the best evasion strategy for a spacecraft with a lower periapsis than a pursuing spacecraft ($Q_e < Q_p$) is to decrease apoapsis. However, once apoapsis is decreased to the point where $Q_e = q_e$, decreasing the radius of any point of the orbit would be equivalent to decreasing periapsis, which is not an efficient maneuver for the evading spacecraft ($J_q < 0$). This can also occur when the evader is in the northeast quadrant ($Q_e > Q_p$, and $q_e > q_p$), because the optimal strategy to increase periapsis can only occur until $q_e = Q_e$. Thus, an evader should avoid being in a circular orbit where $Q_e = q_e$, because from that orbit there is no prudent direction for it to transfer. The other limiting line to consider is where $q_e = r_c$ where r_c is the radius of the central body or the radius of the central body plus some altitude limit. Of course, if the optimal evasion strategy for the evading spacecraft is to decrease periapsis, it can only do so until periapsis is just above this limit. Thus, a pursuer ideally would initially position itself such that it leverages the limit of a circular orbit

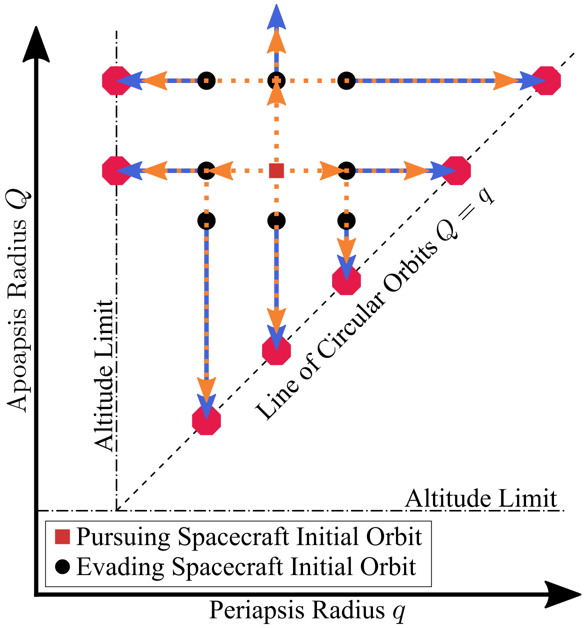

Apoapsis Radius Q
Altitude Limit
Line of Circular Orbits $Q = q$
Altitude Limit
🟥 Pursuing Spacecraft Initial Orbit
● Evading Spacecraft Initial Orbit
Periapsis Radius q

Figure 5.6: Limits on evasion strategies

or the limit of q_e impacting the central body, whichever the evading spacecraft would be closer to. These limits are shown in Fig. 5.6 with red octagons. The only evasion direction without a limit is the scenario where the evading spacecraft has $Q_e > Q_p$ and $q_e = q_p$, but this is a highly constrained position that may not be possible while satisfying other mission objectives. The next best initial orbit for the evading spacecraft would give it the most runway possible before hitting a limit, giving the most potential to move outside of the reachable set of the pursuing spacecraft. That is, the initial orbit with the preferred risk posture would satisfy other mission constraints and maximize the ΔV between the orbit and the limit on the evasion strategy.

The best case strategies for an evading spacecraft found here only allow the evading spacecraft to maneuver equally as efficiently as a pursuing spacecraft. Thus, such a strategy is likely only viable in the case where the evading spacecraft believes it has enough ΔV capability to outlast the pursuing spacecraft. This motivates future investigation into the use of orbit phasing maneuvers as an evasion tool in a pursuit/evasion game. Indeed, the strategies found here do not take into account the need for the pursuer to match the evader's orbit phasing for true capture, because time-free transfers are being used. The need to match phasing can allow an evader to destroy any exact timing used by a pursuing spacecraft to attempt rendezvous.

The preceding analysis has entirely focused on the planar case, but when transfers in inclination space are also considered then we have the additional quantity of interest

$$J_i \equiv \frac{\partial J}{\partial i_f} = \frac{\partial \Delta V_p}{\partial i_f} - \frac{\partial \Delta V_e}{\partial i_f} \tag{5.42}$$

Eqs. (5.21) and (5.22) for J_Q and J_q can similarly be evaluated in the three dimensional case using the full ΔV_{total} equation in Eq. (5.13) to describe the ΔV cost instead of the planar $\Delta V'_{total}$ in Eq. (5.1). In the three dimensional case, however, equations (5.21) and (5.22) can not be easily evaluated or reduced analytically. Through numerical calculation in sample scenarios, J_Q, J_q and J_i appear to always be negative. That is, the pursuing spacecraft appears to always be more efficient, and there do not appear to be any "special" directions where both spacecraft are equally efficient. This, in part, motivates a nonlinear analysis of the objective function in order to gain a better understanding of evasion strategies.

5.4.2 Nonlinear Pursuit-Evasion Analysis

The previous method gives an understanding of local, incremental evasion strategies, but does not give a full understanding of relative advantages for each spacecraft to transfer to any given final orbit. The baseline scalar cost for the pursuing spacecraft to reach the evading spacecraft's initial orbit is denoted as $\Delta V_0 = \Delta V(q_{0,p}, Q_{0,p}, i_{0,p}, q_{0,e}, Q_{0,e}, i_{0,e})$. The "$p$" subscript denotes the pursuer, and the "e" subscript denotes the evader. A grid of pursuer optimal transfer costs is denoted as

$$[\Delta V_p] = \Delta V(q_{0,p}, Q_{0,p}, i_{0,p}, q_{f,1:n_q}, Q_{f,1:n_Q}, i_{f,1:n_i}) \tag{5.43}$$

and the grid of evader optimal transfer costs as

$$[\Delta V_e] = \Delta V(q_{0,e}, Q_{0,e}, i_{0,e}, q_{f,1:n_q}, Q_{f,1:n_Q}, i_{f,1:n_i}) \tag{5.44}$$

Both grids are of dimension $n_q \times n_Q \times n_i$. Using these grids, the "differenced grid" can be calculated as

$$[J_{p-e}] = [\Delta V_p] - [\Delta V_e] - \Delta V_0 \tag{5.45}$$

The differenced grid $[J_{p-e}]$ is a grid where each value is simply the objective function Eq. (5.20) with the baseline capture cost ΔV_0 subtracted out. In this grid, zero values indicate that both spacecraft must expend the same amount of additional ΔV to reach a given orbit. Negative values mean that an evading spacecraft must expend more ΔV to reach the same orbit as the pursuing spacecraft, and positive values mean that an evading spacecraft must expend less ΔV to reach the same orbit as the pursuing spacecraft. In other words, the differenced grid represents how much additional ΔV a pursuing spacecraft must expend as compared to the evading spacecraft in order to capture the evader at a given orbit.

In the planar case we have not found any orbits with positive values in the differenced grid, which indicates that even in a nonlinear sense the evader cannot gain an advantage while in-plane with the pursuing spacecraft. The linear planar evasion strategies that have been found in the previous section can clearly be seen in the differenced grid, as shown in the example in Fig. 5.7 with the 0 contour line.

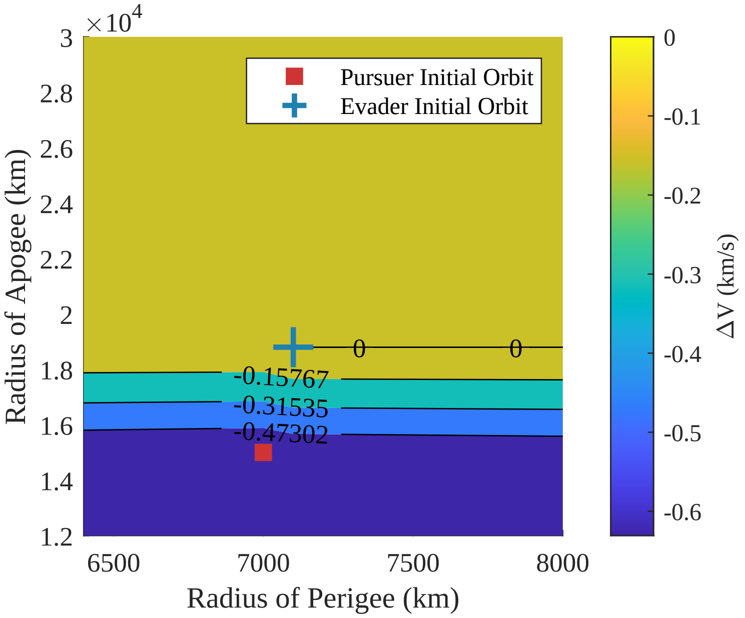
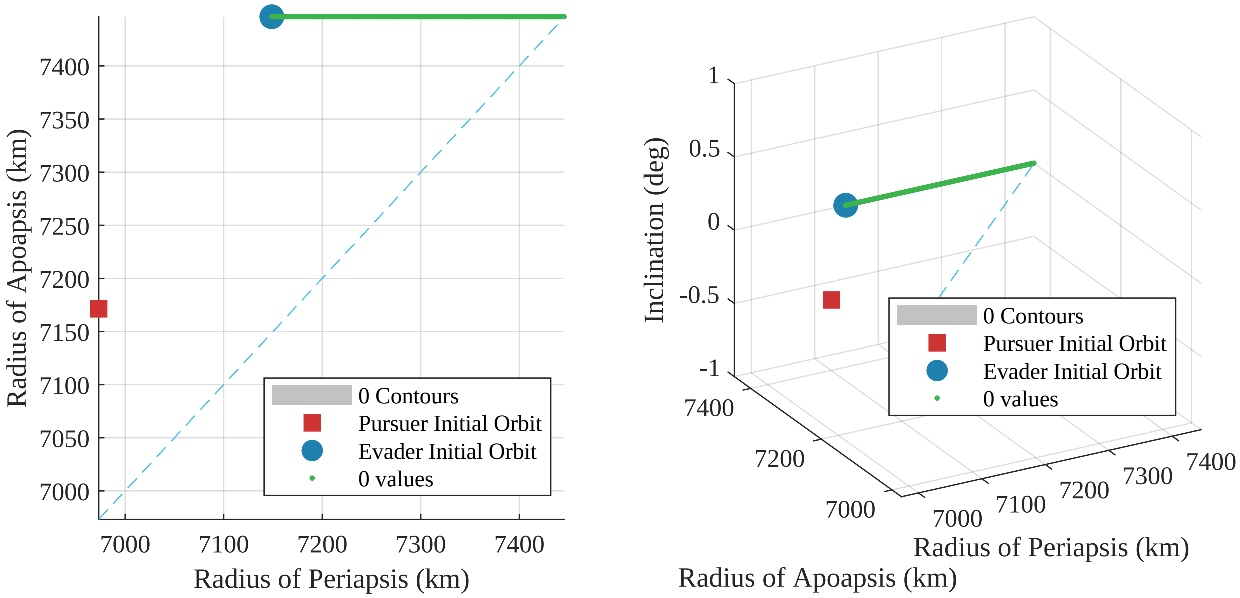


Figure 5.7: Differenced grid contour lines

For the 3-dimensional case, a majority of orbits in the differenced grid are found to have negative values while some orbits have 0 values, similar to the planar case. Figure 5.8 shows the resulting differenced grid for a sample scenario where the pursuer and evader begin in the same orbital plane ($i_p = i_e = 0\,\text{deg}$). Zero values, indicated by green dots, follow exactly the continuous planar evasion strategy as found in Sec. 5.4.1 by examining partial derivatives. The continuous line of 0 values indicates that the evading spacecraft can impulsively raise its periapsis radius multiple times as part of an evasion strategy that allows it to maneuver as efficiently as a pursuing spacecraft.

Interestingly, in certain cases there are regions where the evading spacecraft can transfer at a lower cost than the pursuing spacecraft. These regions are shown by 0 level contours in gray; the fact that the contours have non-zero volume indicates that there are positive values of the differenced grid within them. The contour is easily seen in in the scenario shown in Fig. 5.9. In this particular example, there are two reasons this special region is found. First, the evading

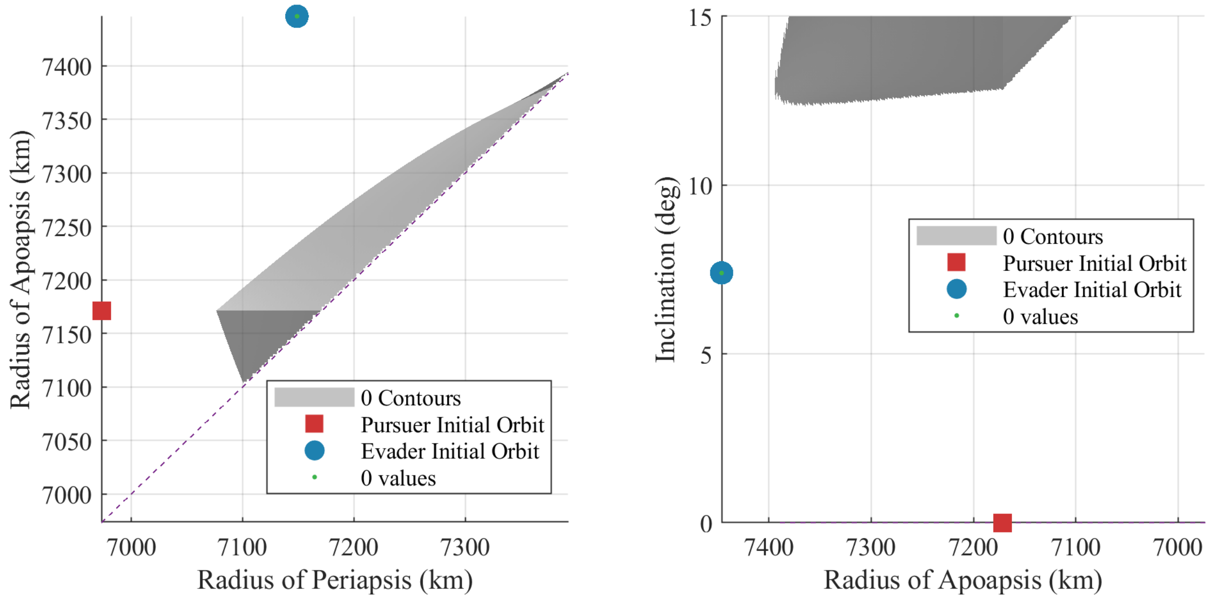


(a) (b)

Figure 5.8: Differenced Grid for a planar pursuit-evasion game

spacecraft has a larger apoapsis radius than the pursuing spacecraft, and thus can perform more efficient inclination changes at slower speeds. The second, more general, reason is that the dogleg maneuver leads to non-intuitive effects on relative costs to transfer orbits. If the dogleg maneuver is not used, and instead the total inclination change for an orbit transfer is just performed at the largest (slowest) apoapsis encountered in the Hohmann transfer (giving a three-impulse transfer), then as expected the same planar evasion strategies as seen in Fig. 5.8 are found for non-planar problems as in Fig. 5.9. In this case, both effects are necessary to create the evader-advantaged region; if the initial conditions of the pursuing spacecraft and evading spacecraft are switched but dogleg maneuvers are used, no positive-valued regions in the differenced grid are found.

These positive regions, however, require quite large ΔV expenditures; in this case they can approach up to $0.13V_c$ (approx. 1 km/s) where V_c is the local circular velocity at periapsis of the evading spacecraft. Further, upon transferring to an orbit in this positive region, there still is no nearby 0 or positive point in the new differenced grid. Further, the advantage the evading



(a)

(b)

Figure 5.9: Differenced Grid for a non-planar pursuit-evasion game

spacecraft has in these positive region is quite low. In results generated for Fig. 5.9, the evader's ΔV advantages are generally on the order of $0.0001V_c$ (a few meters per second) or less. For very large maneuvers this small of an advantage over an evading spacecraft may not truly be useful.

It also appears that when the two spacecraft do not begin in the same plane there is no longer an "incremental" evasion strategy like that seen in the planar case. That is, there is no continuous line of 0 values seen extending from the evading spacecraft's initial orbit. This indicates that there is likely no such strategy directly in the q, Q, or i directions. However, it is certainly possible that there is a more complex, combined maneuver that allows the evader to incrementally maneuver as efficiently as the evading spacecraft. Because this combined maneuver strategy would not have a clean line of grid points evaluated along it, it would not be trivial to see the resulting 0 points in the differenced grids produced here. Differenced grids with very fine resolution centered closely around the evading spacecraft have been generated, but a continuous line of 0 points has not been found.

5.5 Endgame Strategy

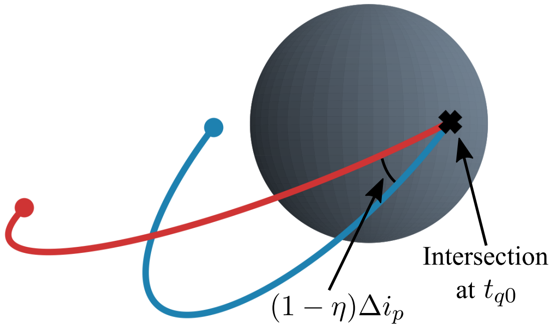


Figure 5.10: Endgame scenario

In this section we explore the terminal phase of the rendezvous pursuit-evasion game given the types of maneuvers considered here. First, suppose that in the course of the pursuit-evasion game the evading spacecraft has expended all available fuel and can no longer maneuver. Further, suppose that the evading spacecraft remains within the reachable set of the pursuing spacecraft. In such a scenario, the terminal phase of the game will appear as in Fig. 5.10. Prior to this terminal phase, the pursuing spacecraft has already matched the right ascension of the ascending node Ω and argument of periapsis ω of the evading spacecraft and has timed its final coast arc so that it intersects the evading spacecraft at the correct time t_{q_0}. At time t_{q_0} the pursuing spacecraft matches the position of the evading spacecraft (with a slight offset), and then performs an impulsive ΔV maneuver to correct the final inclination difference of $(1-\eta)\Delta i_p$ and to match the apoapsis radius of the evading spacecraft. If rendezvous occurs at apoapsis, then a similar maneuver is performed but to match periapsis radius instead of apoapsis radius.

Note that if the evader lies outside of the reachable set of the pursuing spacecraft, the scenario depicted in Fig. 5.10 is still possible with the difference being that the pursuing spacecraft does not have enough ΔV to perform a full rendezvous maneuver at the capture point. Because the rendezvous maneuver is not possible in this case, it is considered a successful evasion and a win for the evading spacecraft. This scenario, however, could potentially result in a successful intercept of the evading spacecraft where the two spacecraft collide. Given that the focus of this work is on the rendezvous pursuit-evasion game, we suppose that the pursuer is not interested in this even as a secondary outcome, and would not destroy the evading spacecraft. Despite the non-cooperative and actively antagonistic nature of the problem, it is realistic to suppose that a pursuing agent may not want to destroy itself, may not want to generate excessive space debris with a collision, may not want to destroy a target without a chance to inspect or modify it, etc.

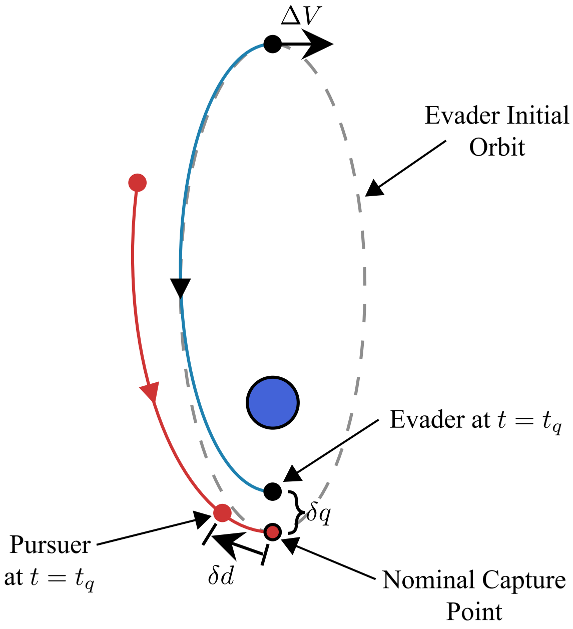


Figure 5.11: Endgame scenario with evasive maneuver

Now in a similar terminal scenario as in Fig. 5.10, suppose that the evading spacecraft has a small amount of fuel left, has a matching inclination with the pursuing spacecraft as well as Ω and ω, and has initial orbit parameters q_0 and Q_0. Figure 5.11 shows this scenario for the case where the pursuing spacecraft is attempting to capture the evading spacecraft at its periapsis. In this scenario, the evading spacecraft is initially at its apoapsis, and can perform a tangential, impulsive maneuver to change its periapsis radius by an amount δq to ensure that the two orbits no longer intersect. The new time of periapsis passage for the evader after it performs a maneuver is t_q. A first order estimate of the distance from the pursuer at time t_q to the position of the pursuer at time t_{q0} is δd. Here, we investigate the relative magnitudes of δq and δd to first order. The first order quantity δd is

$$\delta d = (t_{q_0} - t_q)V_{q_0} = \delta T V_{q_0} \tag{5.46}$$

where V_{q_0} is the orbital speed at periapsis of the evader's initial orbit. To find δT, the partial derivative of the equation for orbital period is used to find

$$\frac{\partial T}{\partial q} = \frac{3\pi}{2\sqrt{\mu}} a_0^{1/2} \frac{\partial a}{\partial q} \tag{5.47}$$

$$\frac{\partial a}{\partial q} = \frac{1}{2} \tag{5.48}$$

$$\delta T = \frac{3\pi}{4\sqrt{\mu}} a_0^{1/2} \delta q \tag{5.49}$$

Combining this with Eq. (5.46) and the expression for V_{q_0} results in the ratio of interest,

$$\frac{\delta d}{\delta q} = \frac{3\pi}{4}\sqrt{\frac{Q_0}{q_0}} \tag{5.50}$$

Because the $Q \geq q$ and $3\pi > 4$, the ratio $\frac{\delta d}{\delta q}$ is always greater than one. That implies that, to first order, one unit change in q will result in a larger change to d.

If the scenario is changed to have the evading spacecraft at its own periapsis at the initial time, and it makes a change δQ to avoid capture, the same ratio becomes

$$\frac{\delta d}{\delta q} = \frac{3\pi}{4}\sqrt{\frac{q_0}{Q_0}} \tag{5.51}$$

In this case, $\frac{\delta d}{\delta q} > 1$ if $\sqrt{\frac{q_0}{Q_0}} > \frac{4}{3\pi}$. The alternative, $\sqrt{\frac{q_0}{Q_0}} \leq \frac{4}{3\pi}$, indicates that the orbit q_0, Q_0 is highly elliptic with an eccentricity of 0.69 or greater. Thus, for most operational orbits the miss distance d will dominate.

This demonstrates one option the evading spacecraft may have to avoid capture while using a ΔV optimal maneuver. However, note that the ΔV costs of such a maneuver has not been discussed. While it is straightforward to relate the changes δq and δQ to corresponding ΔV amounts for the evading spacecraft, the pursuing spacecraft cannot immediately respond with a maneuver because it likely will not be at periapsis or apoapsis of its own orbit. In this analysis, maneuvers can only occur at periapsis and apoapsis to maintain ΔV optimality. The pursuing spacecraft must now re-plan an interception strategy that it can only begin, at the earliest, once it reaches the nominal capture point at periapsis or apoapsis where the evading spacecraft no longer reaches. This replanned strategy is then again subject to being thwarted by changes in q and Q once the pursuer is again on its final coast arc. Thus, the inherent multi-stage nature of the pursuit-evasion game becomes apparent; the true game consists of both agents reacting to the actions of the other over time. A future analysis may consider using the maneuvers here in a carefully constructed multi-stage game, but such a scenario is difficult to pose in a meaningful way. Future work may also consider the use of orbit phase (anomaly), RAAN, and argument of periapsis changes as tools for optimal evasion strategies; orbit phase changes may be of particular interest due to their relatively low ΔV cost and their ability to eliminate the precise conditions a pursuer might use for a ΔV optimal transfer.

The limitations imposed in this analysis illustrates the usefulness of the differential game approach, especially for the endgame phase, because it inherently entails solving for the optimal control at all times with consideration for what the optimal maneuvers of the other agent will be as well. The differential game approach inherently takes into account that the pursuit-evasion game is not a discrete, multi-stage problem, but it is instead essentially an infinite stage problem across time. A differential game approach would also not restrict maneuvers to only occur at periapsis and apoapsis, allowing the pursuing spacecraft to immediately respond to the evasion maneuver made by the evading spacecraft if such a strategy is optimal. An immediate response, especially for small δq or δQ, would likely be better than waiting to maneuver until the next apsis passage. Given an immediate response from the pursuing spacecraft, then it would likely be most prudent for the evading spacecraft to also maneuver in a manner that is not strictly ΔV optimal. The differential game or even closed loop control approaches discussed in Sec. 5.1 are potential candidate methods that could be used to solve for controls in this endgame scenario if they are adaptable to the terminal condition of rendezvous (note that the vast majority of them address the intercept problem). Future work might consider analyzing a two-phase approach where the strategy outlined in this section is used early on when the separation between the two spacecraft is large, before switching to a differential game or closed loop control law approach to generate controls for both spacecraft. The exact threshold distance defining a “large” separation is not immediately clear, so the logic used in deciding when to switch strategies is an interesting topic to explore. The impact of the initial risk posture on the ability to evade capture is also an interesting avenue to explore in a two-phase game. Thus, while the maneuver strategy used in this section could potentially be applied to the terminal phase of the pursuit-evasion game, the primary usefulness of the maneuver strategy is either when using it to maneuver out of the reachable set of the pursuing spacecraft for guaranteed evasion or when the separation between the two spacecraft is large and ΔV (fuel) optimality can be prioritized without adversely impacting the ability to perform or avoid final rendezvous.

5.6 Conclusions

This work has illuminated some aspects of the time-free, ΔV optimal spacecraft rendezvous pursuit-evasion problem. If an evading spacecraft has knowledge of the total ΔV available to the pursuing spacecraft, it can use the knowledge of the pursuing spacecraft’s reachable set of orbits to transfer to an orbit that is guaranteed to avoid capture. Without knowledge of the pursuing spacecraft’s total available ΔV , alternative evasion strategies have also been identified. Notably, for planar pursuit-evasion games there are special directions for the evading spacecraft to move where it will match the efficiency of a pursuing spacecraft. In the 3-dimensional $(q - Q - i)$ case, these special directions were not found, but the evading spacecraft can potentially find certain orbits that it can transfer to at a lower cost than the pursuing spacecraft. While such regions may not be directly useful for practical applications due to the high ΔV of the transfers, they are of theoretical interest and perhaps may inform strategies that do not limit transfers to the time-free ΔV optimal transfers used here.

Chapter 6

Dynamic Game Theory Perspective

6.1 Zero-Sum Game: Mass-Optimal Orbital Pursuit Evasion Game

6.1.1 Introduction

The orbital pursuit-evasion game has most typically been studied in the context of a pursuing agent attempting to match (or get close to matching) the position of an evading spacecraft [103, 104, 106, 109, 110]. Indeed, some of the earliest work in the field of differential games itself focused on the “homicidal chauffeur” game, a type of pursuit-evasion game that roughly models a driver attempting to run down a person on foot [37]. The focus on intersecting the position of an evading spacecraft or achieving a close pass to the target is most typically associated with a kinetic impactor or explosive device intended to disable or destroy the target. However, kinetic and explosive impacts create large amounts of debris that make operating spacecraft for all parties more difficult. Thus, there is interest in offensive and defensive methods to attack or protect space assets in ways that do not generate excessive space debris. This section instead addresses a scenario where the pursuing spacecraft is attempting to rendezvous with the evading spacecraft. This unwanted rendezvous may be for the purposes of unwanted close inspection or unwanted modification of the evading spacecraft. We are also interested in large initial separations between the two spacecraft; linear dynamics are poor approximations in those cases and spacecraft in such scenarios are much more likely to benefit from coasting arcs in their solutions than if they are already close together. We also use a mass-optimal approach to potentially enable these coasting arcs with terminal rendezvous.

The commonly used time-optimal solutions require that both spacecraft have always on thrust; this may be an inefficient method of control, especially when initial spacecraft separation is large. Both terminal rendezvous and mass-optimal orbital pursuit-evasion games have been largely unexplored, with the exception of [114] which found sub-optimal solutions to the non-linear case with quadratic control cost.

6.1.2 Problem Formulation

Here, the mass-optimal pursuit-evasion rendezvous game is posed and solved as a zero-sum differential game with an indirect formulation. In a two player zero sum game the objective of each player $J_i(\cdot)$ always sum to zero, i.e. $J_1 + J_2 = 0$. This implies that as one agent improves its objective the other agent must have an equivalent setback in its own objective. Instead of stating that each player minimizes their own objective function, it can equivalently be said that both players have the same objective function $J(\cdot)$ where one is maximizing the objective while the other is minimizing the objective. In the time-optimal intercept pursuit-evasion game, this objective is time to intercept where the pursuing spacecraft attempts to minimize the objective and the evading spacecraft attempts to maximize the objective.

We seek a saddle point solution to the differential game; at a saddle point each agent will worsen its outcome (as measured by the objective function) if it modifies its strategy. That is, if player 1 is maximizing $J(\cdot)$ and player 2 is minimizing $J(\cdot)$, and they have controls $\boldsymbol{u}_1(\cdot)$ and $\boldsymbol{u}_2(\cdot)$ respectively, then a saddle point or Nash equilibrium solution will be found when

$$J(\boldsymbol{u}_1, \boldsymbol{u}_2^*) \leq J(\boldsymbol{u}_1^*, \boldsymbol{u}_2^*) \leq J(\boldsymbol{u}_1^*, \boldsymbol{u}_2) \tag{6.1}$$

where a superscript asterisk indicates an optimal solution, which in this case synonymous with a saddle point or Nash equilibrium solution. Given the optimal solution $\boldsymbol{u}_1^*, \boldsymbol{u}_2^*$, neither player can improve their cost by varying their control.

As is common in differential game approaches, a terminal condition for the game is set for successful capture. For much of the work concerned with terminal intercept, the terminal condition is then

$$\boldsymbol{g}(\boldsymbol{X}) = \boldsymbol{r}_p(t_f) - \boldsymbol{r}_e(t_f) = \mathbf{0} \tag{6.2}$$

where $\boldsymbol{X}$ is the full state of the game, $\boldsymbol{r}$ is the Cartesian position of the spacecraft, and the subscripts e and p represent the evader and pursuer respectively. In this section, however, the terminal condition for successful capture is

$$\boldsymbol{g}_1(\boldsymbol{X}) = \boldsymbol{x}_p(t_f) - \boldsymbol{x}_e(t_f) = \mathbf{0} \tag{6.3}$$

where $\boldsymbol{x}$ is the orbital state of each spacecraft. This is a significant factor in determining the optimal solution; any solution found with this terminal condition must result in a successful rendezvous, even if the initial conditions are such that the evading spacecraft need not do anything to avoid the pursuing spacecraft (e.g. the evading spacecraft is not reachable by the pursuing spacecraft). Thus, significant care must be taken not only in constructing the rest of the problem, but also in selecting initial conditions such that a solution to the differential game has real meaning.

We first formulate the mass-optimal pursuit evasion with objective as

$$J = \int_{t_0}^{t_f} \left(\frac{T_p}{c_p}\right) dt \tag{6.4}$$

which translates to the total fuel mass used by the evading spacecraft over the course of the game. The evading spacecraft wishes to use as little fuel as possible by minimizing J, while the pursuing spacecraft wishes to make the evading spacecraft use as much fuel as possible (in the hopes that the evader will run out of fuel) by maximizing J. In this problem we use the same state representation as in Section 4.2; spacecraft states are represented by modified equinoctial elements (MEEs) along with the mass of the spacecraft. The initial states of the spacecraft and the final time t_f are fixed and known. Spacecraft 1 is the evading spacecraft, spacecraft 2 is the pursuing spacecraft,

spacecraft numbers are indexed with the variable i, and the states are defined as

$$\boldsymbol{x}_i = \begin{bmatrix} p_i \\ f_i \\ g_i \\ h_i \\ k_i \\ L_i \end{bmatrix} \qquad \boldsymbol{X}_i = \begin{bmatrix} \boldsymbol{x}_i \\ m_i \end{bmatrix} \qquad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \end{bmatrix} \tag{6.5}$$

where m_i is the mass of spacecraft i. The costates are written as

$$\boldsymbol{\lambda}_i = \begin{bmatrix} \lambda_{p_i} \\ \lambda_{f_i} \\ \lambda_{g_i} \\ \lambda_{h_i} \\ \lambda_{k_i} \\ \lambda_{L_i} \\ \lambda_{m_i} \end{bmatrix} \qquad \boldsymbol{\lambda} = \begin{bmatrix} \boldsymbol{\lambda}_1 \\ \boldsymbol{\lambda}_2 \end{bmatrix} \tag{6.6}$$

and the full state is

$$\boldsymbol{Z}_i = \begin{bmatrix} \boldsymbol{X}_i \\ \boldsymbol{\lambda}_i \end{bmatrix} \qquad \boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_1 \\ \boldsymbol{Z}_2 \end{bmatrix} \tag{6.7}$$

Notably, implicit in solving this problem with the formulation used here is the assumption that both spacecraft have perfect information about one another's state at the initial time, and only open loop strategies are found so they do not observe future states of the opponent. This gives baseline information to analyze how to optimally maneuver, but many other factors would be considered for closed-loop strategies (e.g. available sensors, state estimation methods, frequency of state updates, etc.).

Given this state representation, the optimal control Hamiltonian for this problem is then

$$H = \frac{T_2}{c_2} + \sum_{i=1}^{2} \left(\boldsymbol{\lambda}_i^T \boldsymbol{F}(\boldsymbol{X}_i, \boldsymbol{u}_i) \right) \tag{6.8}$$

$$\dot{\boldsymbol{X}}_i = \boldsymbol{F}(\boldsymbol{X}_i, \boldsymbol{u}_i) \tag{6.9}$$

which is expanded to

$$H = \frac{T_2}{c_2} + \sum_{i=1}^{2} \left(\boldsymbol{\beta}_i^T \frac{T_i}{m_i} \hat{\boldsymbol{u}}_i - \lambda_{m_i} \frac{T_i}{c_i} \right) \tag{6.10}$$

with $\boldsymbol{\beta}$ being defined the same as in Section 4.2. Using Pontryagin's maximum/minimum principle, we can define the optimal control direction for each spacecraft as

$$\hat{\boldsymbol{u}}_1 = \frac{\boldsymbol{\beta}_1}{|\boldsymbol{\beta}_1|} \tag{6.11}$$

$$\hat{\boldsymbol{u}}_2 = \frac{-\boldsymbol{\beta}_2}{|\boldsymbol{\beta}_2|} \tag{6.12}$$

to allow each player to optimally maximize/minimize the Hamiltonian. The optimal Hamiltonian is then

$$H^* = \frac{T_2}{c_2} - \frac{|\boldsymbol{\beta}_2|}{m_2} T_2 - \lambda_{m_2} \frac{T_2}{c_2} + \frac{|\boldsymbol{\beta}_1|}{m_1} T_1 - \lambda_{m_1} \frac{T_1}{c_1} \tag{6.13}$$

where the only control variables for each spacecraft are now their thrust levels T_i. These are dictated by the switching functions S_i:

$$S_1 = \frac{|\boldsymbol{\beta}_1|}{m_1} - \frac{\lambda_{m_1}}{c_1} \tag{6.14}$$

$$S_2 = -\left[\frac{1}{c_2} - \frac{|\boldsymbol{\beta}_1|}{m_1} - \frac{\lambda_{m_1}}{c_1} \right] \tag{6.15}$$

where the thrust levels must be:

$$T_i = T_{i,\max} \quad S_i > 0 \tag{6.16}$$

$$T_i = 0 \qquad S_i < 0 \tag{6.17}$$

The costate dynamics are derived from the partial derivative of the Hamiltonian with respect to the state, i.e.

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{X}} \tag{6.18}$$

This partial derivative is the same as in Section 4.2.2 for the orbital states $\boldsymbol{x}_i$ and found using computer aided symbolic differentiation. For the mass costates, the dynamics are:

$$\dot{\boldsymbol{\lambda}}_{m_1} = \frac{|\boldsymbol{\beta}_1| T_1}{m_1^2} \tag{6.19}$$

$$\dot{\boldsymbol{\lambda}}_{m_2} = \frac{-|\boldsymbol{\beta}_2| T_2}{m_2^2} \tag{6.20}$$

$$\tag{6.21}$$

The terminal constraint given in Eq. (6.3) is also applied here. Given that constraint, the transversality conditions dictate that at the final time t_f

$$\boldsymbol{\lambda}_{m_i,f} = 0 \quad i = 1, 2 \tag{6.22}$$

$$\boldsymbol{\lambda}_{\boldsymbol{x}_1,f} + \boldsymbol{\lambda}_{\boldsymbol{x}_2,f} = 0 \tag{6.23}$$

The problem is now a two-point boundary value problem where the initial costates must be found such that Eqs. (6.22 - 6.23) and (6.3) are satisfied, with state dynamics given by Eqs. (6.18) and (6.9), and control given by Eqs. (6.11 - 6.12) and (6.16 - 6.17). The problem is numerically solved here using the multiple shooting method outlined in Section 4.2.

Consider an alternative formulation that uses the evading spacecraft's fuel mass as the optimization objective as opposed to the pursuing spacecraft's fuel mass. In that case the objective is

$$J = \int_{t_0}^{t_f} \left(\frac{T_e}{c_e} \right) dt \tag{6.24}$$

where the evading spacecraft maximizes J and the pursuing spacecraft minimizes J. If one simply swapped the initial conditions of the pursuing and evading spacecraft, solving this problem is exactly equivalent to the problem based on optimizing the the pursuing spacecraft's fuel mass. In other words, given a solution to one of these problems, the designation of which players is the pursuing spacecraft and which player is the evading spacecraft is not strictly defined. Rather, the designation of the players dictates which of the two proposed objective functions has been optimized for a Nash equilibrium solution. This dual problem is important for interpreting the numeric results.

6.1.3 Results

In Figs. 6.1 - 6.2 sample solutions are shown for a scenario in which the 100 kg pursuing and evading spacecraft are on the same, slightly elliptic orbit but separated by 1° in true anomaly. Figure 6.1 shows a shorter time transfer than the example in Fig. 6.2. Each of the six classical orbit elements are plotted in the bottom right corner, the top right corner shows the orbits in cartesian position space, and the left hand side shows the thrust and switching functions. In Fig. 6.1 the pursuing spacecraft in red has some coasting arcs where it is not thrusting, but the evading spacecraft in blue is constantly thrusting at its maximum capability of 1 N. This type of trajectory cannot be found with a time-optimal formulation, but must be accounted for when designing a system that can successfully avoid an efficient pursuer. The majority of the control effort for both spacecraft appears to go into changing the orbit semimajor axis; the out of plane changes to i and Ω ("RAAN") are quite small, as one might expect given how fuel-intensive those changes can be. In Fig. 6.1 both spacecraft are also thrusting in the terminal phase of the game right up until capture.

Figure 6.2 presents a more complicated case, even though it is the same scenario as in Fig. 6.1 but with a longer time t_f. In this case, what should be the evading spacecraft in blue is thrusting in the terminal phase while the pursuing spacecraft is coasting in the terminal phase. This implies that the evading spacecraft actually achieves rendezvous with a passive pursuing spacecraft after the pursuing spacecraft completes its last thrusting arc. Of course, in a true pursuit-evasion game, the evading spacecraft would never do such a thing. This behavior is an artifact of the terminal rendezvous constraint; every problem solved must have rendezvous at the final time. However, given the duality of the two potential formulations discussed in Section 6.1.2, this case can equivalently be thought of as one in which the evading spacecraft is in red and the pursuing spacecraft is in blue. This alternative assignment of the spacecraft also has special meaning if the red spacecraft has expended all of its available fuel after its last thrusting period.

Indeed, solutions to the problems of Eqs. (6.4) and (6.24) have special meaning if the available

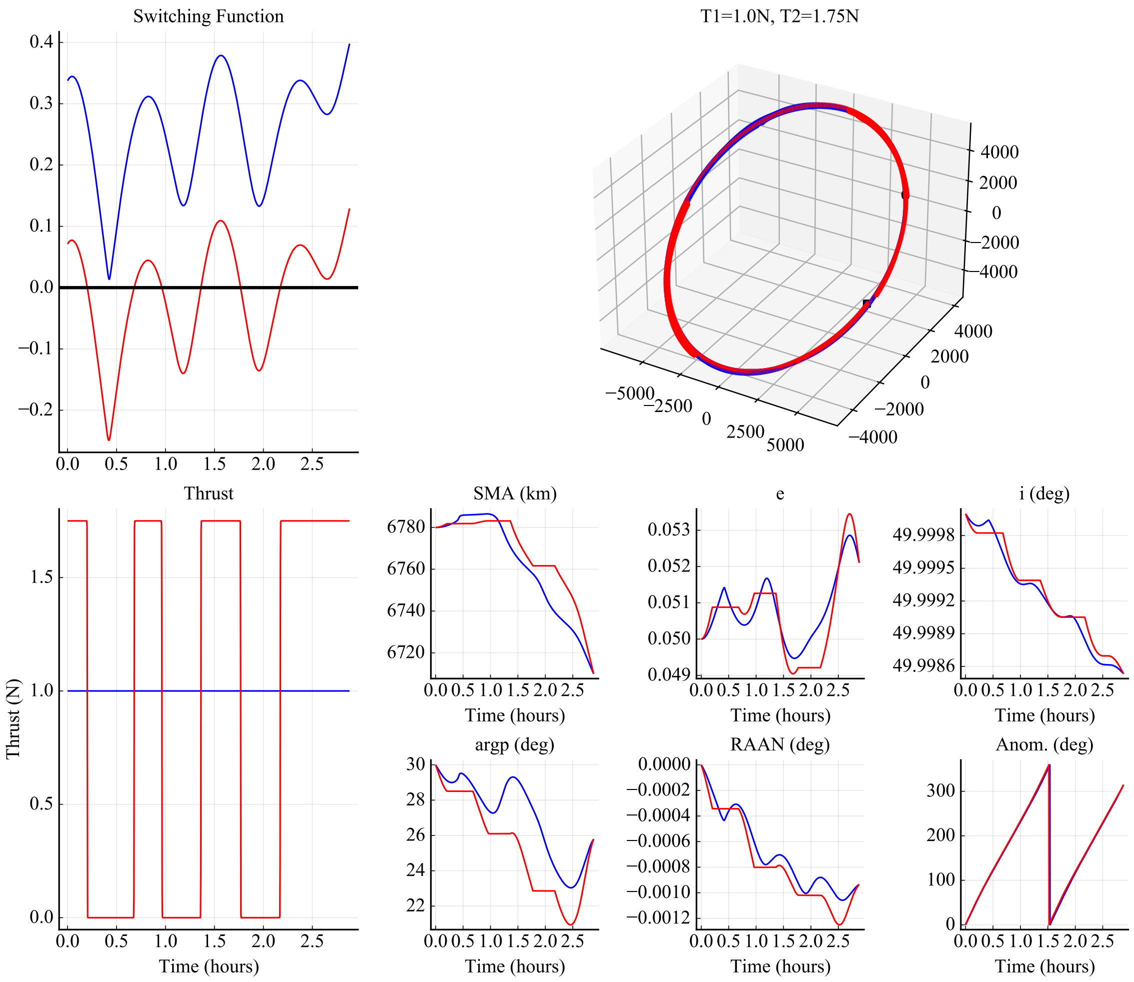


Figure 6.1: Mass-optimal pursuit evasion differential game solution 1

fuel of the spacecraft being optimized is fully depleted at t_f (i.e. if for Eq. (6.4 the fuel of the pursuing spacecraft is fully depleted at t_f or for Eq. (6.24 the fuel of the evading spacecraft is fully depleted). This is because a true end to this pursuit-evasion game would occur when one spacecraft runs out of fuel (though notably, this is not the only reasonable stopping condition[1]). If the evading spacecraft runs out of fuel, the winner of the game is determined by whether or not the pursuing spacecraft still has enough fuel to rendezvous with the evading spacecraft at that time. If the pursuing spacecraft runs out of fuel prior to rendezvous, then the evading spacecraft has won. If the time span of the numeric solution of the game is extended such that the pursuing spacecraft

[1] e.g., another stopping condition could be if the evading spacecraft exits the reachable set of the pursuing spacecraft

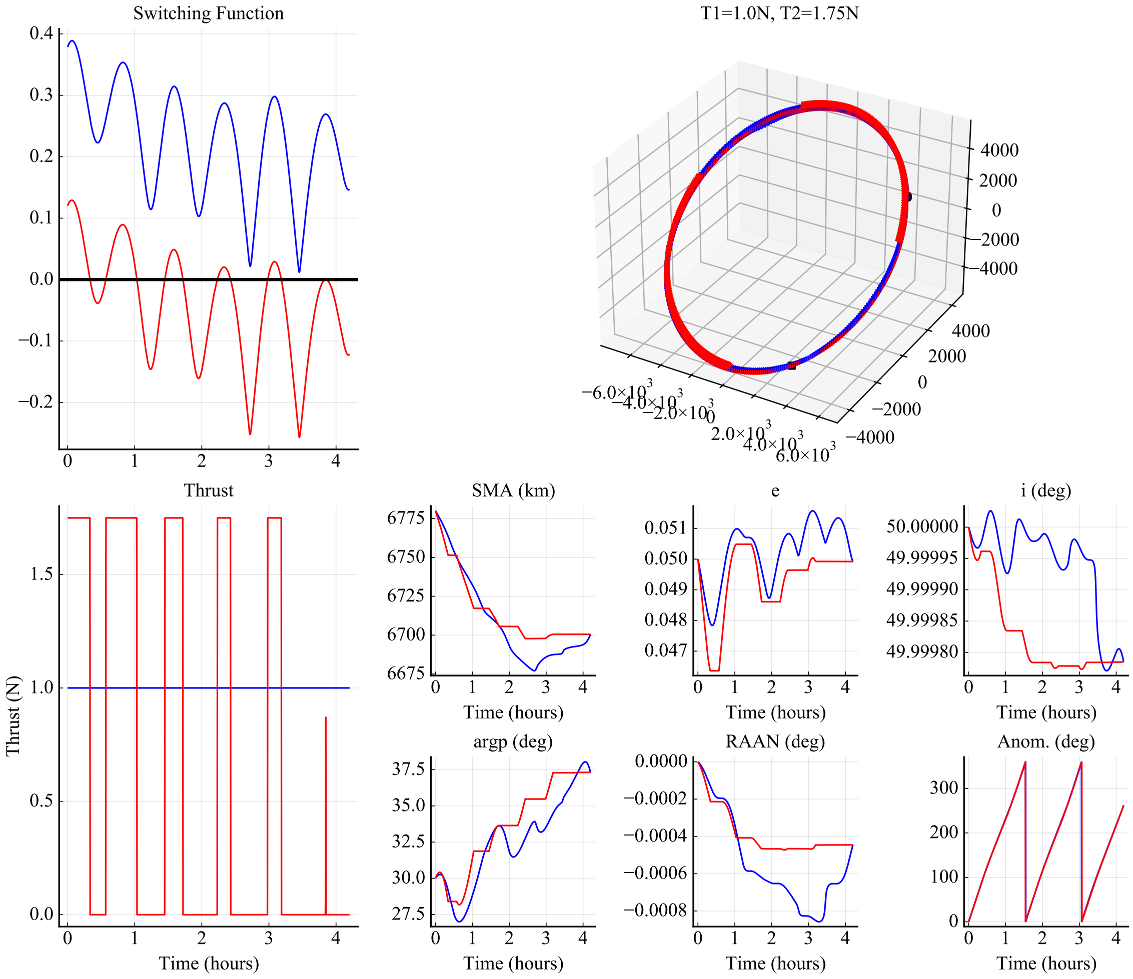


Figure 6.2: Mass-optimal pursuit evasion differential game solution 2

has depleted all of its fuel at the final time if solving Eq. (6.4), then the pursuing spacecraft will fail to accomplish rendezvous if it deviates in any way from the optimal solution (recall that at the equilibrium solution we are solving for, the objective function will be worse for either agent if they deviate from the optimal solution). A similar argument is true for the evading spacecraft if solving Eq. (6.24). Both solutions illuminate the limits of performance of the pursuing spacecraft and the evading spacecraft, and both perspectives are necessary for both players to analyze the scenario and design their space systems. Neither the pursuer nor the evader can optimize their own system without considering how the opposing system would interact with it.

6.2 General-Sum Game: Collision Avoidance

6.2.1 Introduction

In Section 3.3 we introduced the spacecraft collision avoidance scenario in which two maneuverable spacecraft must select a set of controls to avoid an impending collision. In that section we addressed the cooperative case where one or both spacecraft maneuver to avoid one another, but must cooperatively decide in some manner how both spacecraft will proceed. This cooperative problem is one of choosing a Pareto-optimal solution that gives the optimal trade-off in ΔV expended by each spacecraft. Constraints were also imposed to ensure that both spacecraft return to their nominal orbits after the close encounter. However, if both spacecraft choose control solutions from different points on the Pareto front, then collision avoidance cannot be guaranteed.

While uncovering the Pareto front for the cooperative case is of interest, an international "space traffic control" or binding set of rules does not presently exist to dictate control solutions to different spacecraft operators. Thus, in this section we consider a non-cooperative case where there is no accepted authority to determine the controls for both spacecraft. In this case, we consider the problem as a non-zero sum differential game where both spacecraft wish to avoid colliding and both spacecraft also wish to minimize their fuel use. The concept of an "optimal" solution becomes much more complex in a differential game, and we must have more specific criteria in mind when attempting to solve the game. Here, we seek to identify the existence and number of Nash equilibrium solutions. A Nash equilibrium solution is advantageous when neither operator has a high degree of trust in the other. At such a solution, neither operator can improve its objective with local changes to its control. However, some degree of cooperation would be needed. For such a scheme to work, both spacecraft operators would need to calculate solutions using the same method. If only a single Nash equilibrium solution exists, then both operators might only need to agree to apply that single solution (e.g. agree on the same method of calculating that equilibrium point). If multiple Nash equilibria exist, then in addition to the previous agreement they must also agree on which solution to use. Alternatively, they may have to reason about what solution the other operator has chosen based on available tracking data [115] if they assume others are solving a game but not communicating which equilibrium solution they are adhering to. This section does not aim to address the inference problem, but instead aims to understand the solution space of the underlying differential game to illuminate potential strengths and weaknesses of applying such an approach.

6.2.2 Problem Formulation

Consider two spacecraft in Earth orbit with initial orbits $\text{œ}_{0,1}$ and $\text{œ}_{0,2}$. During the engagement with start time $t_0 = 0$ and fixed end time t_f, the spacecraft positions at some time will pass closer to one another than a threshold distance d_{min} if no maneuvers are made. The orbit element set œ used in examples shown here is Cartesian, but it could be Keplerian, modified equinoctial, etc. The state of each spacecraft $\boldsymbol{x}_i$ $(i \in 1, 2)$ is simply the orbit element set

$$\boldsymbol{x}_i = \begin{bmatrix} \text{œ}_i \end{bmatrix}_{6\times 1} \tag{6.25}$$

with a full state vector of

$$\boldsymbol{x} = \begin{bmatrix} \text{œ}_1 \\ \\ \text{œ}_2 \end{bmatrix}_{12\times 1} \tag{6.26}$$

Each spacecraft is interested in minimizing its own control effort. We write the cost functions as

$$J_i = \int_{t_0}^{t_f} \boldsymbol{u}_i(t)^T \boldsymbol{u}_i(t) \quad i \in 1, 2 \tag{6.27}$$

with the path constraint

$$C(\boldsymbol{x}, t) = \Delta\boldsymbol{r}(t)^T \Delta\boldsymbol{r}(t) - d_{\text{min}}^2 \geq 0 \quad \forall t \tag{6.28}$$

where $\Delta\boldsymbol{r}(t)$ is the cartesian position difference between the spacecraft as a function of time.

Though we still aim to find a Nash equilibrium solution, this differential game presents a notable departure from the previous, zero-sum game explored in this chapter. Here, the spacecraft are dynamically coupled through the path constraint in Eq. (6.28), but their objective functions are no longer simply opposites of one another. In the zero sum case, we were able to leverage the indirect formulation because a single objective function and Hamiltonian could be formed, just with both agents optimizing in different directions. In the nonzero-sum case here, different methods must be applied.

To solve the nonzero-sum collision avoidance game we leverage the iterative linear-quadratic (iLQ) method developed by Fridovich-Keil et al.[116]. While differential games are in general quite difficult to solve, the iLQ method takes advantage of the relative ease of solving a differential game with linear dynamics and a quadratic objective function. The general procedure is to first initially guess a control strategy, propagate the full nonlinear dynamics of the game using the control strategy, then linearize the dynamics and quadraticize the cost about the resulting trajectory and solve the resulting linear quadratic (LQ) game. The solution to the LQ game can then be used to update the fully nonlinear control, and the process can then be repeated until the control converges. In this work we use the iLQGames.jl implementation of the iLQ method [117].

The path constraint in Eq. (6.28) is enforced by adding a penalty term to the cost integrand in the form:

$$1000(d_{\min} - |\Delta \boldsymbol{r}(t)|)^2 : |\Delta \boldsymbol{r}(t)| < d_{\min} \tag{6.29}$$

$$0 : |\Delta \boldsymbol{r}(t)| \geq d_{\min} \tag{6.30}$$

and we verify that the final solution does not have $|\Delta \boldsymbol{r}(t)| < d_{\min}$.

6.2.3 Results

A sample collision avoidance result is shown in Fig. 6.3 where the uncontrolled, impacting trajectories are shown with yellow dotted lines and the controlled trajectories are shown with solid navy and orange lines. Because the two cases are difficult to differentiate at orbital scale, Fig. 6.4 shows the game solution meets the desired $d_{\min} = 1$ km constraint. The control profiles for the spacecraft are shown in Fig. 6.5, given an initial guess of zero control for both spacecraft.

Of particular interest is the number of equilibrium solutions that might exist in a given orbital collision avoidance game. In this example case where two spacecraft are on course to exactly impact

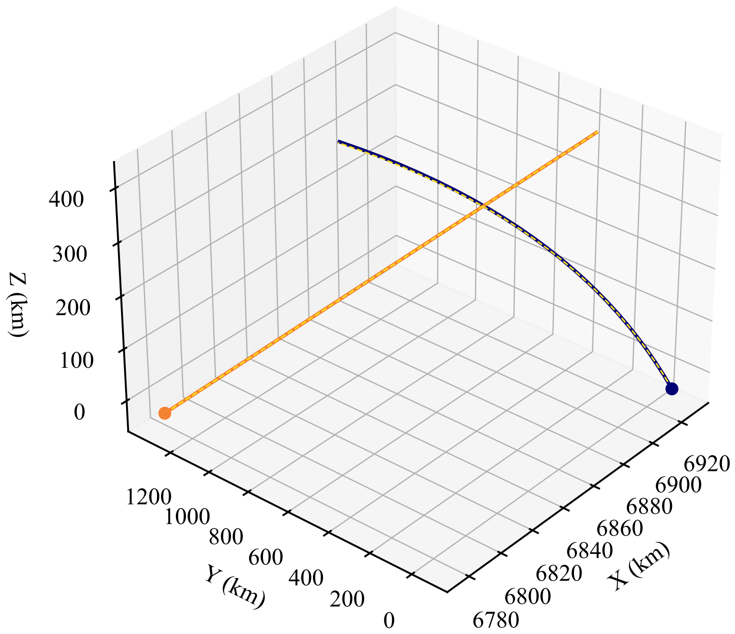


Figure 6.3: Collision avoidance game solution

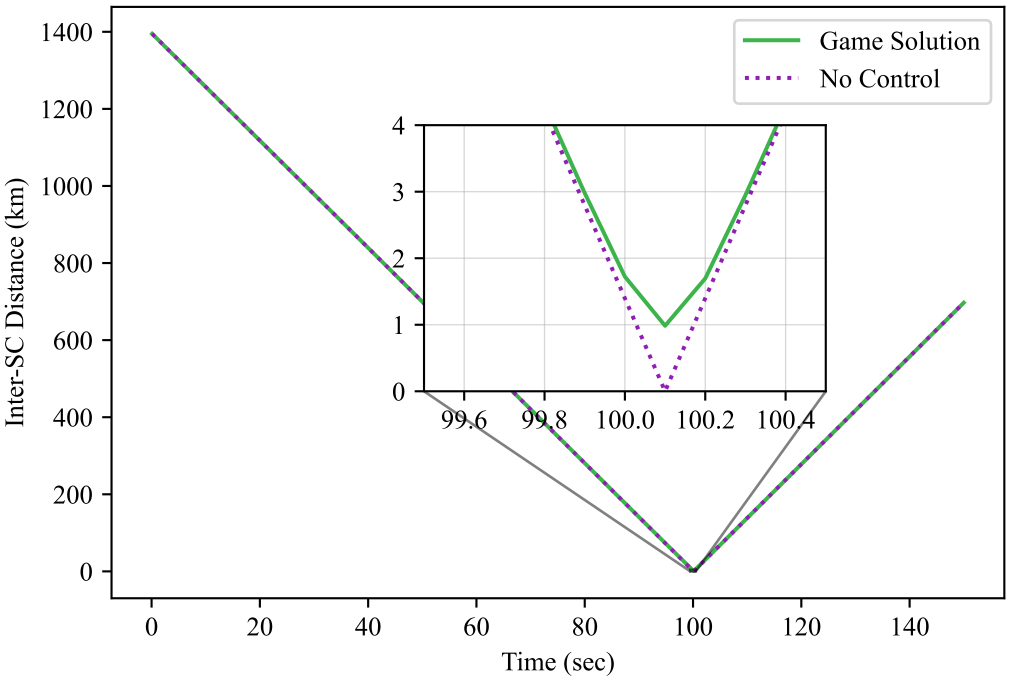


Figure 6.4: Collision avoidance game inter-spacecraft distance

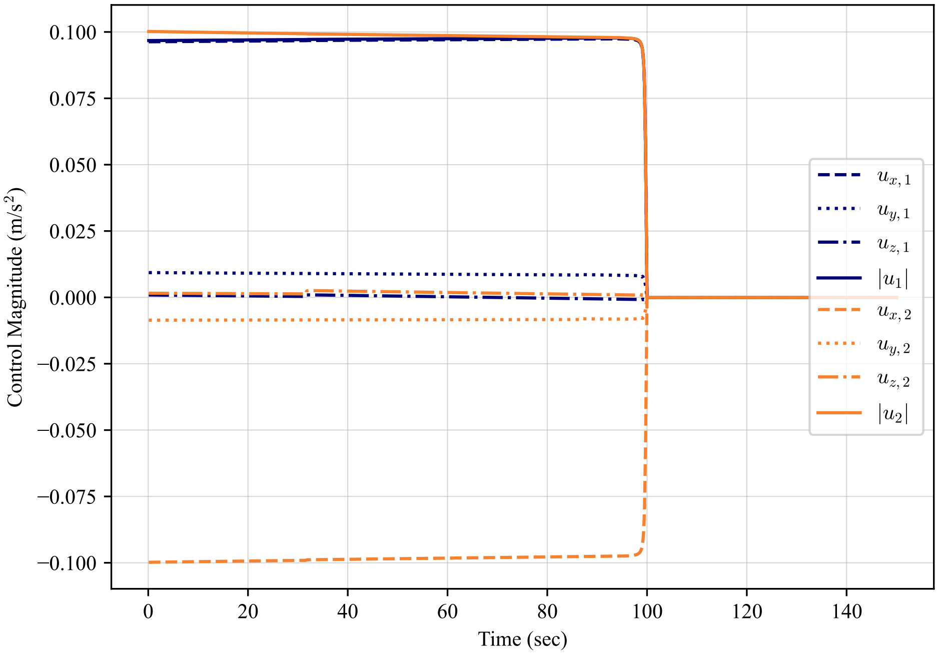
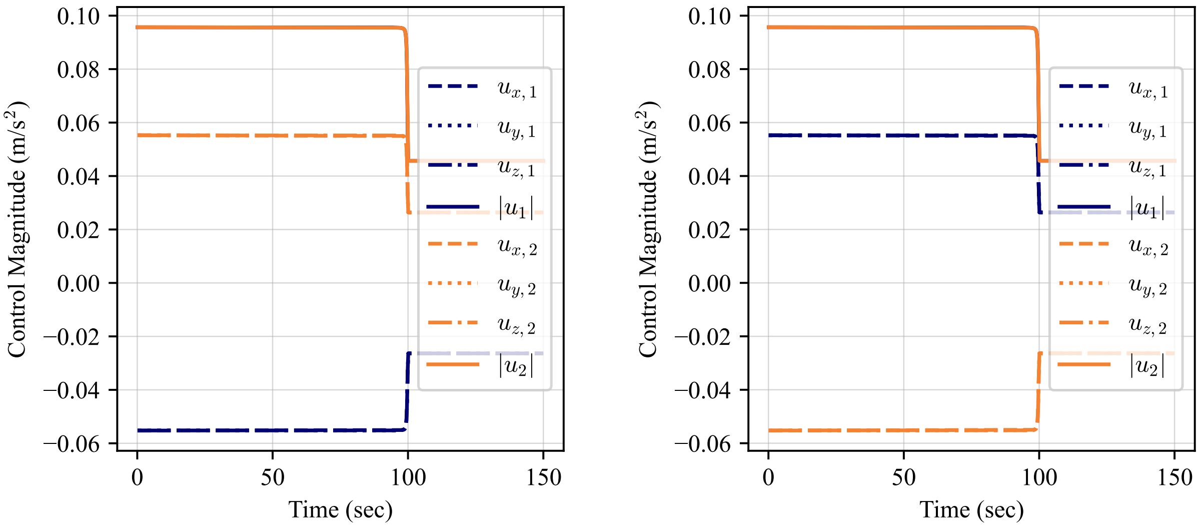


Figure 6.5: Collision avoidance game control profiles

at the time of closest approach, we have been able to find three unique solutions. The first solution shown in Figs. 6.3-6.5 was found with an initial control guess of zero (i.e. no control applied). Two more solutions, shown in Fig. 6.6, were found when initializing the control values of each spacecraft with a small bias instead of zero. The initial control of each spacecraft was set to be equal but opposite of the other with a small, constant value. By changing which of the spacecraft is positively biased and which is negatively biased, we are able to produce the two similar but distinct solutions in Fig. 6.6. These solutions are also distinct from the solution in Fig. 6.5.

This finding is of particular practical interest. First, given that there is at least one type of case where multiple equilibrium solutions can be found, if this technique were to be used there might need to be some inference or agreement on which equilibrium solution is being selected or should be selected. Even if it could be proven that a single equilibrium solution existed, there would need to be a consensus on how that equilibrium solution is calculated to ensure that all operators are operating as others expect them to. The ideal of a differential game solution is that all rational actors will follow the solution of the game, but in reality many different factors that are unmodeled



(a) Bias 1 solution

(b) Bias 2 solution

Figure 6.6: Collision avoidance game solved with initial control bias values

in the differential game will impact each operator's choice of control. These additional factors range from spacecraft CONOPs limitations to even simply being unaware that a conjunction is imminent. Despite these issues, understanding the solution space of these games can be invaluable in focusing efforts to shape space policy and norms of operation that are effective and reasonably implementable if a broadly tasked and empowered space traffic control cannot direct the actions of spacecraft operators. Future work might also consider the impact of economic factors that affect how much control each decision maker is willing to expend on a single spacecraft. Given each decision makers future plans for its own spacecraft, they might be more or less willing to expend fuel for collision avoidance.

Chapter 7

Conclusion

This dissertation has developed methods to optimize a number of different types of multi-spacecraft trajectory optimization problems. Consequently, a number of different applications and scenarios have been addressed to find solutions to problems of interest in the astrodynamics community while advancing the state-of-the-art of solvable problems across the different classes of optimization problem considered here. In Chapter 3 we explored the time-free ΔV optimal cooperative rendezvous problem and for the two-spacecraft planar problem found a set of optimal, constant total ΔV rendezvous orbits. We also found interesting patterns of solutions for the non-planar and $N > 2$ cases. Chapter 3 also developed a method of finding bounding optimal path costs for multi-spacecraft multi-target missions as well. Future work might apply this time-free approach as a heuristic in solving a finite-time problem (e.g. a certain set of debris objects); the time free approach will likely be more or less useful depending on the specifics of the actual targets of interest. Finally, Chapter 3 briefly explored Pareto-optimal cooperative collision avoidance scenarios.

Chapter 4 first develops a method to solve the problem of finding Pareto-optimal low-thrust trajectories with consideration for robustness to missed thrust events. This method not only allows a mission designer to ensure that a trajectory is robust, it also allows them to weigh the added robustness against the optimal cost of fuel or time of flight needed to enable that robustness. This is crucial information that informs system-level design of space missions. While the method theoretically could be used to account for any number of missed thrust events, the exponential growth in problem complexity when adding additional missed thrust events makes such a process likely numerically infeasible if fully constraining the trajectory with multiple missed thrust events. Future work might consider first using the direct approach as outlined in Section 4.1 for a single level of missed thrust event to arrive at a solution from a poor initial guess, then transition the solved virtual spacecraft trajectories to an indirect formulation (e.g. similar to the formulation in Section 4.2). The indirect formulation provides a lower dimensional method of representing a full solution and might make the problem more tractable as additional missed thrust events are added, while the direct method could be used for initial introduction of virtual spacecraft with poor initial guesses.

Section 4.2 applies an indirect formulation to a similar cooperative rendezvous and deployment problem to that addressed in Chapter 3, but with fewer assumptions and restrictions placed on the problem and allowable control. The approach enables fewer general conclusions to be made, but provides a method to find solutions to such problems. Future work could more comprehensively apply the method to identify architectures and solutions of interest for specific applications.

Perhaps the most important overall takeaway for cooperative problems is that while combining the trajectories of $N \geq 2$ spacecraft into a single problem and finding a solution can be difficult, it is not necessarily impossible. Resistance to this idea can often be found in literature, and authors are not incorrect to state that such problems are difficult. However, with careful construction of problems and thoughtful application of the tools of optimization, certain multi-spacecraft trajectory optimization problems can be solved as demonstrated through the examples in this dissertation.

Chapters 5 and 6 both explore spacecraft pursuit evasion game from a fuel-optimal perspective and with rendezvous as the terminal condition. Fuel-optimal spacecraft pursuit-evasion games have to our knowledge not been solved in previous work, and provide crucial insight into the dynamics of the game when spacecraft can use natural dynamics to their advantage. Future work here might also more comprehensively apply the solution method found in Chapter 6 to identify characteristics of the pursuit-evasion game given different initial conditions and spacecraft capabilities. The pursuit-evasion approaches in either chapter Chapters 5 or 6 could also be investigated as fuel-optimal initial phases of a game where in the terminal phase the objective changes to capture time and the spacecraft constantly thrust.

Section 6.2 applies a non-zero sum differential game approach to solving for optimal collision avoidance maneuvers when two active spacecraft have an impending close approach. This is a promising method of solution that could be adopted by convention if no “space traffic control” authority is given the ability to the direct the actions of potentially impacting spacecraft. Future work should consider other non-zero sum games of interest; even in cases where all spacecraft are owned by a single entity, a non-zero sum differential game approach might give insight into how each spacecraft should make decisions when operating autonomously.

The methods and solutions in this dissertation can also guide the development of solution methods for analogous optimization problems. Some optimization problems similar to those posed and solved in this dissertation might be solved in a manner very similar to or in exactly the same way as shown here. Other analogous problems might not be so amenable to the use of the same solution method; simple changes in problem statement can sometimes wildly change the nature of the problem. However, in this dissertation we have applied a number of different optimization techniques across a number of different problem classes. The justifications provided for the techniques used in each situation might guide the construction of new solution methods for new problems. The methods developed in this work might also provide initial guesses for more complex optimization problems of interest as well.

Bibliography

[1] K. Alfriend, S. R. Vadali, P. Gurfil, J. How, and L. Breger. Spacecraft Formation Flying: Dynamics, Control and Navigation, chapter 10. Butterworth-Heinemann, Oxford, United Kingdom, 2009. https://doi.org/10.1016/C2009-0-17485-8.

[2] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen. A Survey of Spacecraft Formation Flying Guidance and Control (Part I): Guidance. In Proceedings of the 2003 American control conference. IEEE, 2003.

[3] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen. A Survey of Spacecraft Formation Flying Guidance and Control (Part II): Control. In Proceedings of the 2004 American control conference, volume 4, pages 2976–2985. IEEE, 2004.

[4] A. Rahmani, M. Mesbahi, and F. Y. Hadaegh. Optimal balanced-energy formation flying maneuvers. Journal of guidance, control, and dynamics, 29(6):1395–1403, 2006.

[5] G. T. Huntington. Advancement and analysis of a Gauss pseudospectral transcription for optimal control problems. PhD thesis, Citeseer, 2007.

[6] G. S. Aoude, J. P. How, and I. M. Garcia. Two-stage path planning approach for solving multiple spacecraft reconfiguration maneuvers. The Journal of the Astronautical Sciences, 56(4):515–544, 2008.

[7] K. C. Howell. Three-dimensional, periodic,‘halo’orbits. Celestial mechanics, 32(1):53–71, 1984.

[8] N. Baresi. Spacecraft Formation Flight on Quasi-periodic Invariant Tori. PhD thesis, University of Colorado Boulder, 2017.

[9] Dynamic Noncooperative Game Theory, 2nd Edition, chapter 1, pages 1–14. https://doi.org/10.1137/1.9781611971132.ch1.

[10] A. E. Bryson. Applied optimal control: optimization, estimation and control. CRC Press, 1975.

[11] J. Nash. Non-cooperative games. Annals of mathematics, pages 286–295, 1951.

[12] C. Venigalla and D. J. Scheeres. Delta-V-Based Analysis of Spacecraft Pursuit–Evasion Games. Journal of Guidance, Control, and Dynamics, 44(11):1961–1971, 2021. https://doi.org/10.2514/1.G005901. In press.

[13] C. Venigalla and D. J. Scheeres. Minimum Bounds on Multi-Spacecraft ΔV Optimal Cooperative Rendezvous. Journal of Guidance, Control, and Dynamics, 43(12):2333–2348, 2020. https://doi.org/10.2514/1.G004978.

[14] C. Venigalla and D. J. Scheeres. Optimal Multi-Spacecraft Cooperative Rendezvous and Constellation Deployment Trajectories. In 2021 AAS/AIAA Astrodynamics Specialist Conference. AAS Paper 21-718, 2021.

[15] C. Venigalla, J. A. Englander, and D. J. Scheeres. Low-Thrust Trajectory Optimization for Maximum Missed Thrust Recovery Margin. In 2020 AAS/AIAA Astrodynamics Specialist Conference. AAS Paper 20-438, 2020.

[16] C. Venigalla and D. J. Scheeres. Lower Bounds on Delta-V Costs for Traveling Satellite Problems. In 10th International Workshop on Satellite Constellations and Formation Flying, July 2019.

[17] C. Venigalla and D. Scheeres. 3-Dimensional Reachable Set Applications to Multi-Spacecraft Trajectory Coordination. In 2019 AAS/AIAA Space Flight Mechanics Meeting, 2019.

[18] C. Venigalla and D. J. Scheeres. Numerical and Analytical Reachable Set Applications to Cooperative and Non-Cooperative Multi-Spacecraft Trajectory Coordination. In 2018 International Astronautical Congress, October 2018.

[19] C. Venigalla and D. Scheeres. Spacecraft Rendezvous and Pursuit/Evasion Analysis Using Reachable Sets. In 28th AAS/AIAA Space Flight Mechanics Meeting, Kissimmee, FL, 2018. AIAA Paper 2018-0219. https://doi.org/10.2514/6.2018-0219.

[20] J. T. Betts. Survey of Numerical Methods for Trajectory Optimization. Journal of Guidance, Control, and Dynamics, 21(2):193–207, 1998. https://doi.org/10.2514/2.4231.

[21] A. Shirazi, J. Ceberio, and J. A. Lozano. Spacecraft trajectory optimization: A review of models, objectives, approaches and solutions. Progress in Aerospace Sciences, 102:76–98, 2018. https://doi.org/10.1016/j.paerosci.2018.07.007.

[22] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. The traveling salesman problem: a computational study. Princeton university press, 2006.

[23] T. S. Ferguson. Linear Programming: A Concise Introduction. 2000.

[24] V. Chvatal. Linear Programming. Macmillan, 1983.

[25] R. J. Vanderbei. Linear Programming: Foundations and Extensions, volume 285. Springer Nature, 2020. https://doi.org/10.1007/978-3-030-39415-8.

[26] S. Boyd and L. Vandenberghe. Convex Optimization. Cambridge University Press, 2004. https://doi.org/10.1017/CBO9780511804441.

[27] H. H. Rosenbrock. An Automatic Method for Finding the Greatest or Least Value of a Function. The Computer Journal, 3(3):175–184, 01 1960. https://doi.org/10.1093/comjnl/3.3.175.

[28] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical programming, 106(1):25–57, 2006. https://doi.org/10.1007/s10107-004-0559-y.

[29] J. Nocedal and S. Wright. Numerical Optimization. Springer Science & Business Media, 2006. https://doi.org/10.1007/978-0-387-40065-5.

[30] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. SIAM Rev., 47:99–131, 2005. https://doi.org/10.1137/S0036144504446096.

[31] J. M. Longuski, J. J. Guzmán, and J. E. Prussing. Optimal control with aerospace applications. Springer, 2014. https://doi.org/10.1007/978-1-4614-8945-0.

[32] D. E. Kirk. Optimal control theory: an introduction. Courier Corporation, 2004.

[33] J. Sims and S. Flanagan. Preliminary design of low-thrust interplanetary missions. In Advances in the Astronautical Sciences, volume 103, page 538–548, Escondido, CA, 1999. Univelt Inc.

[34] A. E. Petropoulos. Simple control laws for low-thrust orbit transfers. 2003.

[35] A. W. Starr and Y. C. Ho. Nonzero-sum differential games. Journal of Optimization Theory and Applications, 3(3):184–206, Mar 1969. https://doi.org/10.1007/BF00929443.

[36] Dynamic Noncooperative Game Theory, 2nd Edition. https://doi.org/10.1137/1.9781611971132.ch1.

[37] R. Isaacs. Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization. Courier Corporation, 1965.

[38] R. Isaacs. Differential games i: Introduction. Technical report, RAND Corporation, 1954.

[39] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen. A survey of spacecraft formation flying guidance and control (part 1): guidance. In Proceedings of the 2003 American Control Conference, 2003., volume 2, pages 1733–1739, June 2003. https://doi.org/10.1109/ACC.2003.1239845.

[40] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen. A survey of spacecraft formation flying guidance and control. Part II: control. In Proceedings of the 2004 American Control Conference, volume 4, pages 2976–2985 vol.4, June 2004. https://doi.org/10.23919/ACC.2004.1384365.

[41] F. W. Gobetz and J. R. Doll. A Survey of Impulsive Trajectories. AIAA Journal, 7(5):801–834, 1969. https://doi.org/10.2514/3.5231.

[42] W. Hohmann. The attainability of heavenly bodies (Die Ereichbarkeit der Himrnelskorper). NASA Technical Translations F-44, 1960.

[43] M. J. Holzinger, D. J. Scheeres, and R. S. Erwin. On-Orbit Operational Range Computation Using Gauss's Variational Equations with J2 Perturbations. Journal of Guidance, Control, and Dynamics, 37(2):608–622, 2014. https://doi.org/10.2514/1.53861.

[44] J. E. Prussing. Optimal Spacecraft Trajectories. Oxford University Press, Oxford, United Kingdom, 2017. https://doi.org/10.1093/oso/9780198811084.001.0001.

[45] J. Prussing. Terminal Maneuver for an Optimal Cooperative Impulsive Rendezvous (AAS 97-649). Advances in the Astronautical Sciences, 97:759–772, 1998.

[46] J. E. Prussing and B. A. Conway. Optimal terminal maneuver for a cooperative impulsive rendezvous. Journal of Guidance, Control, and Dynamics, 12(3):433–435, May 1989. https://doi.org/10.2514/3.20427.

[47] V. Coverstone-Carroll and J. E. Prussing. Optimal cooperative power-limited rendezvous between neighboring circular orbits. Journal of Guidance, Control, and Dynamics, 16(6):1045–1054, 1993. https://doi.org/10.2514/3.21126.

[48] V. Coverstone-Carroll and J. E. Prussing. Optimal cooperative power-limited rendezvous between coplanar circular orbits. Journal of Guidance, Control, and Dynamics, 17(5):1096–1102, 1994. https://doi.org/10.2514/3.21315.

[49] A. Dutta and P. Tsiotras. Hohmann-Hohmann and Hohmann-Phasing Cooperative Rendezvous Maneuvers. The Journal of the Astronautical Sciences, 57(1):393–417, Jan 2009. https://doi.org/10.1007/BF03321510.

[50] R. Bevilacqua and M. Romano. Rendezvous Maneuvers of Multiple Spacecraft Using Differential Drag Under J2 Perturbation. Journal of Guidance, Control, and Dynamics, 31(6):1595–1607, 2008. https://doi.org/10.2514/1.36362.

[51] D. Thakur, S. Hernandez, and M. R. Akella. Spacecraft Swarm Finite-Thrust Cooperative Control for Common Orbit Convergence. Journal of Guidance, Control, and Dynamics, 38(3):478–488, 2015. https://doi.org/10.2514/1.G000621.

[52] R. A. Broucke and A. F. B. A. Prado. Orbital planar maneuvers using two and three-four (through infinity) impulses. Journal of Guidance, Control, and Dynamics, 19(2):274–282, 1996. https://doi.org/10.2514/3.21615.

[53] K. D. Mease and A. V. Rao. Minimum-Fuel Transfer between Coplanar Elliptic Orbits - Global Results Using Green's Theorem, pages 113–125. Springer US, Boston, MA, 1994. https://doi.org/10.1007/978-1-4615-2425-0_12.

[54] V. A. Chobotov. Orbital Mechanics, chapter 5. American Institute of Aeronautics and Astronautics, Reston, VA, 3 edition, 2002. https://doi.org/10.2514/4.862250.

[55] L. Rider. Characteristic Velocity Requirements for Impulsive Thrust Transfers Between Non Co-Planar Circular Orbits. ARS Journal, 31(3):345–351, 1961. https://doi.org/10.2514/8.5478.

[56] M. Anthony and G. Fosdick. Optimum Two-impulse, Conodal Transfers between Inclined Circular Orbits. In Space Technology and Science, page 443, 1964.

[57] Z. Drezner and H. W. Hamacher. Facility Location: Applications and Theory, page 2. Springer Science & Business Media, Berlin, Germany, 2001.

[58] M. J. Holzinger, D. J. Scheeres, and J. Hauser. Reachability Using Arbitrary Performance Indices. IEEE Transactions on Automatic Control, 60(4):1099–1103, April 2015. https://doi.org/10.1109/TAC.2015.2391451.

[59] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming, 106(1):25–57, Mar 2006. https://doi.org/10.1007/s10107-004-0559-y.

[60] D. J. Scheeres. Orbital Motion in Strongly Perturbed Environments: Applications to Asteroid, Comet and Planetary Satellite Orbiters, page 178. Springer, Berlin, Germany, 2012. https://doi.org/10.1007/978-3-642-03256-1.

[61] H. Schaub and J. L. Junkins. Analytical Mechanics of Space Systems, chapter 12. American Institute of Aeronautics and Astronautics, Reston, VA, 2005. https://doi.org/10.2514/4.105210.

[62] J. Stuart, K. Howell, and R. Wilson. Application of multi-agent coordination methods to the design of space debris mitigation tours. Advances in Space Research, 57(8):1680–1697, 2016. https://doi.org/10.1016/j.asr.2015.05.002.

[63] B. W. Barbee, S. Alfano, E. Piñon, K. Gold, and D. Gaylor. Design of spacecraft missions to remove multiple orbital debris objects. Advances in the Astronautical Sciences, 144:93–110, 2012. https://doi.org/10.1109/AERO.2011.5747303.

[64] J. Bang and J. Ahn. Multitarget Rendezvous for Active Debris Removal Using Multiple Spacecraft. Journal of Spacecraft and Rockets, 56(4):1237–1247, 2019. https://doi.org/10.2514/1.A34344.

[65] K. Alemany and R. D. Braun. Survey of global optimization methods for low-thrust, multiple asteroid tour missions. In 2007 AAS/AIAA Space Flight Mechanics Meeting January 2007, Sedona, Arizona, 2007.

[66] J. R. Stuart, K. C. Howell, and R. S. Wilson. Design of end-to-end trojan asteroid rendezvous tours incorporating scientific value. Journal of Spacecraft and Rockets, 53(2):278–288, 2016.

[67] J. Bang and J. Ahn. Two-phase framework for near-optimal multi-target Lambert rendezvous. Advances in Space Research, 61(5):1273 – 1285, 2018. https://doi.org/https://doi.org/10.1016/j.asr.2017.12.025.

[68] K. T. Alfriend, D.-J. Lee, and N. G. Creamer. Optimal servicing of geosynchronous satellites. Journal of guidance, control, and dynamics, 29(1):203–206, 2006.

[69] H. Shen and P. Tsiotras. Optimal scheduling for servicing multiple satellites in a circular constellation. In AIAA/AAS Astrodynamics Specialist Conference and Exhibit, page 4907, 2002.

[70] J. Foust. Orbit Fab demonstrates satellite refueling technology on ISS. SpaceNews, 06 2019.

[71] M. Di Carlo, J. M. R. Martin, and M. Vasile. Automatic trajectory planning for low-thrust active removal mission in low-earth orbit. Advances in Space Research, 59(5):1234–1258, 2017.

[72] U. Pferschy and R. Staněk. Generating subtour elimination constraints for the TSP from pure integer solutions. Central European journal of operations research, 25(1):231–260, 2017.

[73] J.-Y. Potvin. Genetic algorithms for the traveling salesman problem. Annals of Operations Research, 63(3):337–370, 1996.

[74] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. Journal of the operations research society of America, 2(4):393–410, 1954.

[75] B. L. Miller, D. E. Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. Complex systems, 9(3):193–212, 1995.

[76] K. Helsgaun. An extension of the Lin-Kernighan-Helsgaun TSP solver for constrained traveling salesman and vehicle routing problems. Technical report, Technical Report, Roskilde University, 2017.

[77] K. Helsgaun. General k-opt submoves for the Lin–Kernighan TSP heuristic. Mathematical Programming Computation, 1(2-3):119–163, 2009.

[78] L. Kotthoff, P. Kerschke, H. Hoos, and H. Trautmann. Improving the state of the art in inexact TSP solving using per-instance algorithm selection. In International Conference on Learning and Intelligent Optimization, pages 202–217. Springer, 2015.

[79] S. A. Mulder and D. C. Wunsch. Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks. Neural Networks, 16(5):827 – 832, 2003. https://doi.org/https://doi.org/10.1016/S0893-6080(03)00130-8. Advances in Neural Networks Research: IJCNN '03.

[80] L. Gurobi Optimization. Gurobi Optimizer Reference Manual, 2019.

[81] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega, 34(3):209 – 219, 2006. https://doi.org/https://doi.org/10.1016/j.omega.2004.10.004.

[82] C. Bombardelli and J. Hernando-Ayuso. Optimal Impulsive Collision Avoidance in Low Earth Orbit. Journal of Guidance, Control, and Dynamics, 38(2):217–225, 2015. https://doi.org/10.2514/1.G000742.

[83] J. Hernando-Ayuso and C. Bombardelli. Low-Thrust Collision Avoidance in Circular Orbits. Journal of Guidance, Control, and Dynamics, 44(5):983–995, 2021. https://doi.org/10.2514/1.G005547.

[84] R. R. Bate, D. D. Mueller, and J. E. White. Fundamentals of Astrodynamics. Courier Dover Publications, 1971.

[85] F. E. Laipert and J. M. Longuski. Automated Missed-Thrust Propellant Margin Analysis for Low-Thrust Trajectories. Journal of Spacecraft and Rockets, 52(4):1135–1143, 2015. https://doi.org/10.2514/1.A33264.

[86] F. E. Laipert and T. Imken. A Monte Carlo Approach to Measuring Trajectory Performance Subject to Missed Thrust. In 2018 Space Flight Mechanics Meeting, 2002. https://doi.org/10.2514/6.2018-0966.

[87] N. Ozaki, S. Campagnola, R. Funase, and C. H. Yam. Stochastic Differential Dynamic Programming with Unscented Transform for Low-Thrust Trajectory Design. Journal of Guidance, Control, and Dynamics, 41(2):377–387, 2018. https://doi.org/10.2514/1.G002367.

[88] J. T. Olympio. Designing robust low-thrust interplanetary trajectories subject to one temporary engine failure. In Proceedings of the 20th AAS/AIAA Space Flight Meeting, pages 10–171, 2010.

[89] J. T. Olympio and C. H. Yam. Deterministic method for space trajectory design with mission margin constraints. In 61st International Astronautical Congress. International Astronautical Federation Prague, 2010.

[90] D. Oh, D. Landau, T. Randolph, P. Timmerman, J. Chase, J. Sims, and T. Kowalkowski. Analysis of System Margins on Deep Space Missions Using Solar Electric Propulsion. In 44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, 2008. https://doi.org/10.2514/6.2008-5286.

[91] M. D. Rayman and S. N. Williams. Design of the First Interplanetary Solar Electric Propulsion Mission. Journal of Spacecraft and Rockets, 39(4):589–595, 2002. https://doi.org/10.2514/2.3848.

[92] S. L. McCarty and D. J. Grebow. Missed Thrust Analysis and Design for Low Thrust Cislunar Transfers (AAS 20-535). In 2020 AAS/AIAA Astrodynamics Specialist Conference, 08 2020.

[93] M. D. Rayman, T. C. Fraschetti, C. A. Raymond, and C. T. Russell. Coupling of system resource margins through the use of electric propulsion: Implications in preparing for the Dawn mission to Ceres and Vesta. Acta Astronautica, 60(10):930 – 938, 2007. https://doi.org/https://doi.org/10.1016/j.actaastro.2006.11.012.

[94] D. H. Ellison, B. A. Conway, J. A. Englander, and M. T. Ozimek. Analytic Gradient Computation for Bounded-Impulse Trajectory Models Using Two-Sided Shooting. Journal of Guidance, Control, and Dynamics, 41(7):1449–1462, 2018. https://doi.org/10.2514/1.G003077.

[95] D. H. Ellison, B. A. Conway, J. A. Englander, and M. T. Ozimek. Application and Analysis of Bounded-Impulse Trajectory Models with Analytic Gradients. Journal of Guidance, Control, and Dynamics, 41(8):1700–1714, 2018. https://doi.org/10.2514/1.G003078.

[96] J. Revels, M. Lubin, and T. Papamarkou. Forward-Mode Automatic Differentiation in Julia. arXiv:1607.07892 [cs.MS], 2016.

[97] L. B. Rall and G. F. Corliss. An introduction to automatic differentiation. Computational Differentiation: Techniques, Applications, and Tools, 89, 1996.

[98] J. A. Englander and A. C. Englander. Tuning Monotonic Basin Hopping: Improving the Efficiency of Stochastic Search as Applied to Low-Thrust Trajectory Optimization. In 24th International Symposium on Space Flight Dynamics, May 2014.

[99] C. Yam, D. Lorenzo, and D. Izzo. Low-thrust trajectory design as a constrained global optimization problem. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 225(11):1243–1251, 2011. https://doi.org/10.1177/0954410011401686.

[100] J. A. Englander, J. M. Knittel, K. Williams, D. Stanbridge, and D. H. Ellison. Validation of a Low-Thrust Mission Design Tool Using Operational Navigation Software (AAS 17-204). In 27th AAS/AIAA Space Flight Meeting, 2017.

[101] W. Feng, L. Han, L. Shi, D. Zhao, and K. Yang. Optimal control for a cooperative rendezvous between two spacecraft from determined orbits. The Journal of the Astronautical Sciences, 63(1):23–46, 2016.

[102] J. L. Junkins and E. Taheri. Exploration of Alternative State Vector Choices for Low-Thrust Trajectory Optimization. Journal of Guidance, Control, and Dynamics, 42(1):47–64, 2019. https://doi.org/10.2514/1.G003686.

[103] H.-X. Shen and L. Casalino. Revisit of the Three-Dimensional Orbital Pursuit-Evasion Game. Journal of Guidance, Control, and Dynamics, 41(8):1823–1831, 2018. https://doi.org/10.2514/1.G003127.

[104] M. Pontani and B. A. Conway. Numerical Solution of the Three-Dimensional Orbital Pursuit-Evasion Game. Journal of Guidance, Control, and Dynamics, 32(2):474–487, 2009. https://doi.org/10.2514/1.37962.

[105] W. T. Hafer, H. L. Reed, J. D. Turner, and K. Pham. Sensitivity Methods Applied to Orbital Pursuit Evasion. Journal of Guidance, Control, and Dynamics, 38(6):1118–1126, 2015. https://doi.org/10.2514/1.G000832.

[106] J. Stupik, M. Pontani, and B. Conway. Optimal Pursuit/Evasion Spacecraft Trajectories in the Hill Reference Frame. 2012. https://doi.org/10.2514/6.2012-4882.

[107] A. Jagat and A. J. Sinclair. Optimization Of Spacecraft Pursuit-Evasion Game Trajectories In The Euler-Hill Reference Frame. 2014. https://doi.org/10.2514/6.2014-4131.

[108] G. D. Bohn. Application of a near-optimal closed loop control law to a pursuit-evasion game between two spacecraft. Technical report, Air Force Institute of Technology Wright-Patterson Air Force Base, 1975.

[109] H. J. Kelley, E. M. Cliff, and F. H. Lutze. Pursuit/Evasion in Orbit. The Journal of the Astronautical Sciences, 29(3):277–288, 1981.

[110] P. Menon and A. Calise. Guidance laws for spacecraft pursuit-evasion and rendezvous. In Guidance, Navigation and Control Conference, August 1988. https://doi.org/10.2514/6.1988-4134.

[111] J. E. Prussing and R. S. Clifton. Optimal multiple-impulse satellite evasive maneuvers. Journal of Guidance, Control, and Dynamics, 17(3):599–606, 1994. https://doi.org/10.2514/3.21239.

[112] C. A. Lembeck and J. E. Prussing. Optimal impulsive intercept with low-thrust rendezvous return. Journal of Guidance, Control, and Dynamics, 16(3):426–433, 1993. https://doi.org/10.2514/3.21027.

[113] R. C. Burk and J. W. Widhalm. Minimum impulse orbital evasive maneuvers. Journal of Guidance, Control, and Dynamics, 12(1):121–123, 1989. https://doi.org/10.2514/3.20378.

[114] A. Jagat and A. J. Sinclair. Nonlinear Control for Spacecraft Pursuit-Evasion Game Using the State-Dependent Riccati Equation Method. IEEE Transactions on Aerospace and Electronic Systems, 53(6):3032–3042, 2017. https://doi.org/10.1109/TAES.2017.2725498.

[115] L. Peters, D. Fridovich-Keil, C. Tomlin, and Z. Sunberg. Inference-Based Strategy Alignment for General-Sum Differential Games. In International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2020.

[116] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin. Efficient Iterative Linear-Quadratic Approximations for Nonlinear Multi-Player General-Sum Differential Games. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 1475–1481, 2020. https://doi.org/10.1109/ICRA40945.2020.9197129.

[117] L. Peters and Z. N. Sunberg. iLQGames.jl: Rapidly Designing and Solving Differential Games in Julia, 2020.

Appendix A

Partial Derivatives of the Optimal Time-Free Orbit Transfer Cost

Partial derivatives of the analytic expression for the optimal orbit transfer cost in Eq. (3.13) can be useful for a number of different reasons, including for use with NLP solvers and for verifying that a solution is stationary. In this analysis initial orbit states are taken to be fixed, and changes in the cost function are explored with respect to changing final orbit parameters. Thus, the relevant partial derivatives to calculate are

$$\frac{\partial J}{\partial q_f} = \frac{\partial J}{\partial v_{1t}}\frac{\partial v_{1t}}{\partial q_f} + \frac{\partial J}{\partial \eta}\frac{\partial \eta}{\partial q_f} + \frac{\partial J}{\partial v_{2t}}\frac{\partial v_{2t}}{\partial q_f} + \frac{\partial J}{\partial v_f}\frac{\partial v_f}{\partial q_f} \tag{A.1}$$

$$\frac{\partial J}{\partial Q_f} = \frac{\partial J}{\partial v_{1t}}\frac{\partial v_{1t}}{\partial Q_f} + \frac{\partial J}{\partial \eta}\frac{\partial \eta}{\partial Q_f} + \frac{\partial J}{\partial v_{2t}}\frac{\partial v_{2t}}{\partial Q_f} + \frac{\partial J}{\partial v_f}\frac{\partial v_f}{\partial Q_f} \tag{A.2}$$

$$\frac{\partial J}{\partial i_f} = \frac{\partial J}{\partial \Delta i}\frac{\partial \Delta i}{\partial i_f} + \frac{\partial J}{\partial \eta}\frac{\partial \eta}{\partial i_f} \tag{A.3}$$

In the multi-spacecraft redezvous case, these partial derivatives are necessary for gradient-based parameter optimization algorithms, and help calculate necessary conditions for finding a minimum.

The following components of the partial derivatives are not dependent on the initial and final orbit values

$$\frac{\partial J}{\partial v_{1t}} = \frac{1}{\Delta V_1}\big[v_{1t} - v_0\cos(\eta\Delta i)\big] \tag{A.4}$$

$$\frac{\partial J}{\partial v_{2t}} = \frac{1}{\Delta V_2}\big[v_{2t} - v_f\cos((1-\eta)\Delta i)\big] \tag{A.5}$$

$$\frac{\partial J}{\partial v_f} = \frac{1}{\Delta V_2}\big[v_f - v_{2t}\cos((1-\eta)\Delta i)\big] \tag{A.6}$$

$$\frac{\partial J}{\partial \eta} = \frac{1}{\Delta V_1}\big[v_{1t}v_0\sin(\eta\Delta i)\Delta i\big] - \frac{1}{\Delta V_2}\big[v_f v_{2t}\sin((1-\eta)\Delta i)\Delta i\big] \tag{A.7}$$

$$\frac{\partial J}{\partial \Delta i} = \frac{1}{\Delta V_1}\big[\eta v_{1t}v_0\sin(\eta\Delta i)\big] + \frac{1}{\Delta V_2}\big[(1-\eta)v_f v_{2t}\sin((1-\eta)\Delta i)\big] \tag{A.8}$$

However, the partial derivatives of the velocities v_{1t}, v_{2t}, and v_f with respect to q_f and Q_f are dependent on the relative values of Q_f and Q_0. For $Q_f > Q_0$,

$$\frac{\partial v_{1t}}{\partial q_f} = 0 \tag{A.9}$$

$$\frac{\partial v_{1t}}{\partial Q_f} = \frac{\mu}{v_{1t}(q_0 + Q_f)^2} \tag{A.10}$$

$$\frac{\partial v_{2t}}{\partial q_f} = 0 \tag{A.11}$$

$$\frac{\partial v_{2t}}{\partial Q_f} = -\frac{\mu q_0(q_0 + 2Q_f)}{v_{2t}Q_f^2(q_0 + Q_f)^2} \tag{A.12}$$

$$\frac{\partial v_f}{\partial q_f} = \frac{\mu}{v_f(q_f + Q_f)^2} \tag{A.13}$$

$$\frac{\partial v_f}{\partial Q_f} = -\frac{\mu q_f(q_f + 2Q_f)}{v_f Q_f^2(q_f + Q_f)^2} \tag{A.14}$$

For $Q_f \leq Q_0$,

$$\frac{\partial v_{1t}}{\partial q_f} = \frac{\mu}{v_{1t}(q_f + Q_0)^2} \tag{A.15}$$

$$\frac{\partial v_{1t}}{\partial Q_f} = 0 \tag{A.16}$$

$$\frac{\partial v_{2t}}{\partial q_f} = -\frac{\mu Q_0(Q_0 + 2q_f)}{v_{2t}q_f^2(q_f + Q_0)^2} \tag{A.17}$$

$$\frac{\partial v_{2t}}{\partial Q_f} = 0 \tag{A.18}$$

$$\frac{\partial v_f}{\partial q_f} = -\frac{\mu Q_f(Q_f + 2q_f)}{v_f q_f^2(q_f + Q_f)^2} \tag{A.19}$$

$$\frac{\partial v_f}{\partial Q_f} = \frac{\mu}{v_f(q_f + Q_f)^2} \tag{A.20}$$

While there is no closed form expression for η, partial derivatives of Eq. (3.15) can be used to solve for the needed partial derivatives of η. For example, the partial derivative of Eq. (3.15) with respect to q_f gives

$$\frac{\partial F}{\partial q_f} = \frac{\partial F}{\partial v_{1t}}\frac{\partial v_{1t}}{\partial q_f} + \frac{\partial F}{\partial \Delta V_1}\left(\frac{\partial \Delta V_1}{\partial q_f} + \frac{\partial \Delta V_1}{\partial \eta}\frac{\partial \eta}{\partial q_f}\right) + \\ \frac{\partial F}{\partial v_{2t}}\frac{\partial v_{2t}}{\partial q_f} + \frac{\partial F}{\partial v_f}\frac{\partial v_f}{\partial q_f} + \frac{\partial F}{\partial \Delta V_2}\left(\frac{\partial \Delta V_2}{\partial q_f} + \frac{\partial \Delta V_2}{\partial \eta}\frac{\partial \eta}{\partial q_f}\right) + \frac{\partial F}{\partial \eta}\frac{\partial \eta}{\partial q_f} = 0 \quad \text{(A.21)}$$

which can be solved for $\dfrac{\partial \eta}{\partial q_f}$, giving the result

$$\frac{\partial \eta}{\partial q_f} = -\left[\frac{\partial F}{\partial v_{1t}}\frac{\partial v_{1t}}{\partial q_f} + \frac{\partial F}{\partial \Delta V_1}\frac{\partial \Delta V_1}{\partial q_f} + \frac{\partial F}{\partial v_{2t}}\frac{\partial v_{2t}}{\partial q_f} + \frac{\partial F}{\partial v_f}\frac{\partial v_f}{\partial q_f} + \frac{\partial F}{\partial \Delta V_2}\frac{\partial \Delta V_2}{\partial q_f}\right] \Bigg/ \\ \left[\frac{\partial F}{\partial \eta} + \frac{\partial F}{\partial \Delta V_2}\frac{\partial \Delta V_2}{\partial \eta} + \frac{\partial F}{\partial \Delta V_1}\frac{\partial \Delta V_1}{\partial \eta}\right] \quad \text{(A.22)}$$

Similarly, solving for $\dfrac{\partial \eta}{\partial Q_f}$ gives

$$\frac{\partial \eta}{\partial Q_f} = -\left[\frac{\partial F}{\partial v_{1t}}\frac{\partial v_{1t}}{\partial Q_f} + \frac{\partial F}{\partial \Delta V_1}\frac{\partial \Delta V_1}{\partial Q_f} + \frac{\partial F}{\partial v_{2t}}\frac{\partial v_{2t}}{\partial Q_f} + \frac{\partial F}{\partial v_f}\frac{\partial v_f}{\partial Q_f} + \frac{\partial F}{\partial \Delta V_2}\frac{\partial \Delta V_2}{\partial Q_f}\right] \Bigg/ \\ \left[\frac{\partial F}{\partial \eta} + \frac{\partial F}{\partial \Delta V_2}\frac{\partial \Delta V_2}{\partial \eta} + \frac{\partial F}{\partial \Delta V_1}\frac{\partial \Delta V_1}{\partial \eta}\right] \quad \text{(A.23)}$$

Finally, solving for $\dfrac{\partial \eta}{\partial i_f}$ gives

$$\frac{\partial \eta}{\partial i_f} = -\left[\frac{\partial F}{\partial \Delta i}\frac{\partial \Delta i}{\partial i_f} + \frac{\partial F}{\partial \Delta V_1}\frac{\partial \Delta V_1}{\partial \Delta i}\frac{\partial \Delta i}{\partial i_f} + \frac{\partial F}{\partial \Delta V_2}\frac{\partial \Delta V_2}{\partial \Delta i}\frac{\partial \Delta i}{\partial i_f}\right] \Bigg/ \\ \left[\frac{\partial F}{\partial \eta} + \frac{\partial F}{\partial \Delta V_2}\frac{\partial \Delta V_2}{\partial \eta} + \frac{\partial F}{\partial \Delta V_1}\frac{\partial \Delta V_1}{\partial \eta}\right] \quad \text{(A.24)}$$

The following additional partial derivatives are needed to evaluate Eqs.A.22), (A.23), and (A.24).

$$\frac{\partial \Delta i}{\partial i_f} = \begin{cases} 1 & i_f > i_0 \\ -1 & i_f < i_0 \end{cases} \quad \text{(A.25)}$$

$$\frac{\partial \Delta V_1}{\partial q_f} = \frac{1}{\Delta V_1}\left[v_{1t}\frac{\partial v_{1t}}{\partial q_f} - v_0\cos(\eta\Delta i)\frac{\partial v_{1t}}{\partial q_f}\right] \tag{A.26}$$

$$\frac{\partial \Delta V_1}{\partial Q_f} = \frac{1}{\Delta V_1}\left[v_{1t}\frac{\partial v_{1t}}{\partial Q_f} - v_0\cos(\eta\Delta i)\frac{\partial v_{1t}}{\partial Q_f}\right] \tag{A.27}$$

$$\frac{\partial \Delta V_2}{\partial q_f} = \frac{1}{\Delta V_2}\left[v_f\frac{\partial v_f}{\partial q_f} + v_{2t}\frac{\partial v_{2t}}{\partial q_f} - \big(v_{2t}\cos((1-\eta)\Delta i)\frac{\partial v_f}{\partial q_f} + v_f\cos((1-\eta)\Delta i)\frac{\partial v_{2t}}{\partial q_f}\big)\right] \tag{A.28}$$

$$\frac{\partial \Delta V_2}{\partial Q_f} = \frac{1}{\Delta V_2}\left[v_f\frac{\partial v_f}{\partial Q_f} + v_{2t}\frac{\partial v_{2t}}{\partial Q_f} - \big(v_{2t}\cos((1-\eta)\Delta i)\frac{\partial v_f}{\partial Q_f} + v_f\cos((1-\eta)\Delta i)\frac{\partial v_{2t}}{\partial Q_f}\big)\right] \tag{A.29}$$

$$\frac{\partial \Delta V_1}{\partial \eta} = \frac{v_{1t}v_0\sin(\eta\Delta i)\Delta i}{\Delta V_1} \tag{A.30}$$

$$\frac{\partial \Delta V_2}{\partial \eta} = \frac{-v_{2t}v_f\sin((1-\eta)\Delta i)\Delta i}{\Delta V_2} \tag{A.31}$$

$$\frac{\partial F}{\partial v_{1t}} = \frac{\Delta i v_0\sin(\eta\Delta i)}{\Delta V_1} \tag{A.32}$$

$$\frac{\partial F}{\partial v_{2t}} = \frac{-\Delta i v_f\sin((1-\eta)\Delta i)}{\Delta V_2} \tag{A.33}$$

$$\frac{\partial F}{\partial v_f} = \frac{-\Delta i v_{2t}\sin((1-\eta)\Delta i)}{\Delta V_2} \tag{A.34}$$

$$\frac{\partial F}{\partial \Delta V_1} = \frac{-\Delta i v_0 v_{1t}\sin(\eta\Delta i)}{\Delta V_1^2} \tag{A.35}$$

$$\frac{\partial F}{\partial \Delta V_2} = \frac{\Delta i v_f v_{2t}\sin((1-\eta)\Delta i)}{\Delta V_2^2} \tag{A.36}$$

$$\frac{\partial F}{\partial \eta} = \frac{\Delta i^2 v_0 v_{1t}\cos(\eta\Delta i)}{\Delta V_1} - \frac{\Delta i v_0 v_{1t}\sin(\eta\Delta i)}{\Delta V_1^2}\frac{\partial \Delta V_1}{\partial \eta} + \ldots$$
$$\ldots\frac{\Delta i^2 v_f v_{2t}\cos((1-\eta)\Delta i)}{\Delta V_2} + \frac{\Delta i v_f v_{2t}\sin((1-\eta)\Delta i)}{\Delta V_2^2}\frac{\partial \Delta V_2}{\partial \eta} \tag{A.37}$$

$$\frac{\partial F}{\partial \Delta i} = \frac{v_0 v_{1t}[\sin(\eta\Delta i) + \eta\Delta i\cos(\eta\Delta i)]}{\Delta V1} - \frac{\Delta i v_0 v_{1t}\sin(\eta\Delta i)}{\Delta V_1^2}\frac{\partial \Delta V_1}{\partial \Delta i} - \ldots$$
$$\ldots\frac{v_f v_{2t}[\sin((1-\eta)\Delta i) - \eta\Delta i\cos((1-\eta)\Delta i)]}{\Delta V_2} + \frac{\Delta i v_f v_{2t}\sin((1-\eta)\Delta i)}{\Delta V_2^2}\frac{\partial \Delta V_2}{\partial \Delta i} \tag{A.38}$$

Importantly, the derivatives presented here are only continuous in certain regions of state space. These regions are bounded by planes where one of the orbit elements q_f, Q_f, or i_f are equal to one of the initial orbit elements of the active spacecraft $q_{0,j}$, $Q_{0,j}$, or $i_{0,j}$ respectively.

A.1 Special Case: Initial and Final Orbit Elements are Equal

In the case where $q_0 = q_f$, $Q_0 = Q_f$, and $i_0 = i_f$, the nominal impulsive ΔV magnitudes are zero, and the above expressions break down. However, nominal ΔV cost equations can be re-written based on the parameter to be varied. For example, to change i_f only, the optimal maneuver ΔV cost is

$$\Delta V = 2V_Q \sin(\frac{\Delta i}{2}) \approx V_Q \Delta i \tag{A.39}$$

for small Δi and

$$V_Q = \sqrt{2\mu \frac{q}{Q(q+Q)}} \tag{A.40}$$

is the speed at apoapsis. Thus,

$$\frac{\partial \Delta V}{\partial i_f} = \begin{cases} V_Q & i_f > i_0 \\ -V_Q & i_f < i_0 \end{cases} \tag{A.41}$$

Considering an impulsive maneuver to change just q_f, the ΔV cost is

$$\Delta V = \begin{cases} v_f - v_0 & q_f > q_0 \\ v_0 - v_f & q_f < q_0 \end{cases} \tag{A.42}$$

where

$$v_0 = \sqrt{2\mu \frac{q_0}{Q(q_0+Q)}} \tag{A.43}$$

$$v_f = \sqrt{2\mu \frac{q_f}{Q(q_f+Q)}}. \tag{A.44}$$

This gives the result

$$\frac{\partial \Delta V}{\partial q_f} = \begin{cases} \frac{\partial v_f}{\partial q_f} & q_f > q_0 \\ -\frac{\partial v_f}{\partial q_f} & q_f < q_0 \end{cases} \tag{A.45}$$

with

$$\frac{\partial v_f}{\partial q_f} = \frac{\mu}{v_f(q_f+Q)^2}. \tag{A.46}$$

Finally, considering an impulsive maneuver to change just Q_f, the ΔV cost is

$$\Delta V = \begin{cases} v_f - v_0 & Q_f > Q_0 \\ \\ v_0 - v_f & Q_f < Q_0 \end{cases} \tag{A.47}$$

where

$$v_0 = \sqrt{2\mu \frac{Q_0}{q(q+Q_0)}} \tag{A.48}$$

$$v_f = \sqrt{2\mu \frac{Q_f}{q(q+Q_f)}}. \tag{A.49}$$

This gives the result

$$\frac{\partial \Delta V}{\partial Q_f} = \begin{cases} \frac{\partial v_f}{\partial Q_f} & Q_f > Q_0 \\ -\frac{\partial v_f}{\partial Q_f} & Q_f < Q_0 \end{cases} \tag{A.50}$$

with

$$\frac{\partial v_f}{\partial Q_f} = \frac{\mu}{v_f (q+Q_f)^2}. \tag{A.51}$$