Nonlinear Methods for Spacecraft Guidance and Trajectory Optimization

by

Spencer Boone

B.S., McGill University, 2013

M.S., University of Colorado, 2016

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

2022

Committee Members:

Jay McMahon, Chair

Marcus Holzinger

Gregory Lantoine

Daniel Scheeres

Zachary Sunberg

Boone, Spencer (Ph.D., Aerospace Engineering Sciences)

Nonlinear Methods for Spacecraft Guidance and Trajectory Optimization

Thesis directed by Prof. Jay McMahon

Many future spacecraft missions are planned to operate far from Earth in highly nonlinear environments, while performing complex navigational maneuvers. This increased complexity in spacecraft trajectories will necessitate the development of new guidance, maneuver design, and trajectory planning algorithms that are suitable for these intricate mission designs. In particular, there is a need for new methods that strike a balance between computationally expensive, full-fidelity trajectory optimization algorithms, and simplified, linearized guidance methods. This dissertation seeks to bridge this gap by developing computationally efficient, accurate, and flexible algorithms using state transition tensors (STTs) to model the nonlinear spacecraft dynamics. First, a higher-order impulsive spacecraft guidance scheme with both a fixed and variable time-of-flight is developed using the STTs of a reference trajectory. Next, these methods are extended to consider continuous-thrust trajectory optimization by combining STTs with differential dynamic programming, a second-order optimization method. STTs are also shown to be useful for accounting for the effects of state uncertainty propagated through nonlinear dynamics, with application to impulsive statistical maneuver design. Building on this, a method is developed to accurately and efficiently model probabilistic constraints on non-Gaussian state distributions, which are frequently encountered in spacecraft dynamics. Finally, a strategy to approximate the higher-order STTs without losing important information is introduced, which improves the efficiency of the underlying algorithms. The STT-based methods are applied to a variety of complex trajectory scenarios, with a particular emphasis on spacecraft operating in cislunar space. These algorithms are shown to be computationally efficient while accurately capturing the effects of nonlinear dynamics. Altogether, this research provides the mathematical and computational tools to use higher-order STTs to achieve a variety of different objectives in spacecraft guidance and trajectory optimization.

Dedication

I would like to dedicate this thesis to my parents, Renée and Tom, without whose support this work would not have been possible.

Acknowledgements

First of all, I would like to express my gratitude to my Ph.D. advisor Jay McMahon for the tremendous amount of support, encouragement, and guidance I have received during the past few years. We have worked on some challenging and exciting problems, and it has been a ton of fun. It is hard to put into words how fortunate I feel to have been given this opportunity. Thank you! I would also like to thank my committee members Marcus Holzinger, Gregory Lantoine, Daniel Scheeres, and Zachary Sunberg for their helpful feedback and suggestions which have improved this thesis.

The members of the ORCCA lab (past and present) have been a huge source of encouragement and friendship over the last few years. In general, the CCAR community has been a wonderful resource for exciting conversations and research ideas. I am immensely thankful for my friends in Colorado, from McGill, and from elsewhere, for the constant encouragement and always welcome distractions.

This dissertation would not have been possible without the incredible support of my family. To Mom and Dad, I am so grateful for the constant support and encouragement you have given me. You have continually pushed me to reach my full potential, and your advice throughout the years has been invaluable, even if I may not always have listened at the time! I would also like to thank my brother Kyle and sister Jocelyne for being awesome friends and role models.

Finally, I would like to thank NASA for funding this Ph.D. through the grants 80NSSC19K0222 and 80NSSC18K0260.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

1.1 Motivation

Many future spacecraft missions are planned to operate far from Earth in highly nonlinear dynamic systems, while performing complex navigational maneuvers. For example, there is growing interest in operating missions in cislunar space, where the spacecraft dynamics are perturbed by both the Earth and Moon gravitational forces. Such missions include the Lunar Gateway, which will maintain a near-rectilinear halo orbit (NRHO) in the Earth-Moon system [115] (see Fig. 1.1), and the Demonstration Rocket for Agile Cislunar Operations (DRACO) [1], which will demonstrate the use of a nuclear thermal propulsion engine in cislunar space. There are also planned and proposed missions performing highly complex tours of Jupiter's moons, using many close flybys of the moons to control the spacecraft trajectories. These include NASA's Europa Clipper, which will frequently fly by Jupiter's moon Europa [68] (see Fig. 1.2), and ESA's JUICE mission [49], which will perform multiple moon flybys and eventually enter into orbit around Ganymede. In addition, many upcoming deep-space missions will employ low-thrust propulsion systems, such as NASA's Psyche mission, which will insert itself into orbit around a small body using low-thrust propulsion [53], and JAXA's DESTINY+ mission, which will use low-thrust propulsion to target Moon gravity assists and perform flybys of multiple small bodies [87].

This increased complexity in spacecraft trajectories is tremendously exciting, and will enable a large amount of compelling scientific observations and discoveries. However, this complexity will necessitate the development of new trajectory planning, guidance, and maneuver design algorithms

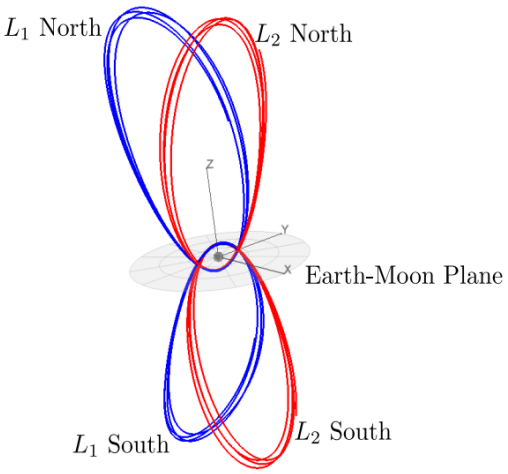

Figure 1.1: Earth-Moon NRHOs plotted in rotating Earth-Moon frame. Source: Ref. [115]

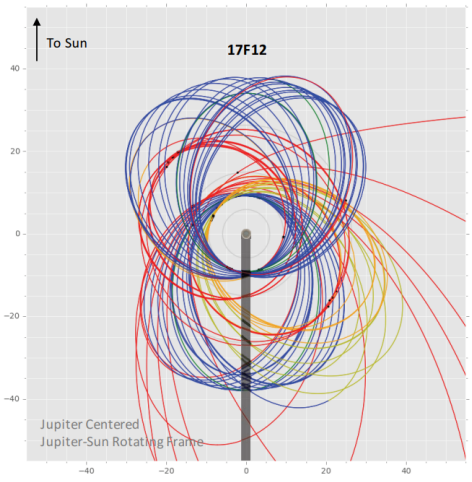

Figure 1.2: Candidate trajectory for Europa Clipper mission, plotted in rotating Jupiter-Europa frame. Source: Ref. [68]

that are suitable for these intricate mission designs. There is a particular need to develop capabilities that can both reduce the computational time to generate these trajectories, and increase the robustness of the mission to operational uncertainties. In addition, as the requirements for new missions move towards more autonomous on-board capabilities, efficient algorithms that are tractable for use on flight hardware, which have limited computational resources, will become crucial.

Currently, most spacecraft trajectory planning algorithms are run on the ground with powerful computers; however, these algorithms are generally computationally intensive and not suitable for use on flight computers. The workflow for computing new trajectories for an ongoing mission typically involves estimating the orbit and state uncertainty of the spacecraft, iterating over a range of control parameters to converge on maneuvers that meet the desired constraints, and uploading maneuver commands to the spacecraft prior to execution. Human navigation or maneuver design analysts are directly involved. The turnaround for this process can range from hours to days [89], limiting missions operating in sensitive dynamical regimes that required rapid responses, or missions in deep space that suffer from communication delays with the ground. A more robust and efficient maneuver planning capability could improve the efficiency of the spacecraft's flight plan, and greatly reduce the burden of routine navigational support.

The long-term objective for spacecraft guidance and maneuver design is to execute these tasks autonomously on-board the spacecraft. This would reduce the reliance on ground contacts for commanding maneuvers, and minimize the impacts of a potential loss of ground communications with the spacecraft. Most flight-proven on-board guidance methods have been used on spacecraft operating in low-Earth orbit, particularly upper stage launch vehicles [6, 28, 30]. These methods commonly rely on linearized predictor-corrector methods to target a future reference state and compute the controls required to steer a spacecraft back to this reference, following some parameterized guidance law. While they have been successful for their intended applications, their convergence region in highly nonlinear dynamic systems outside of low-Earth orbit can be very small. For deep-space missions, a previous implementation of autonomous guidance is the AutoNav system developed by the Jet Propulsion Laboratory (JPL) [38]. This system was used on NASA's Deep Impact mission, but also relies on a linear propagation using the first-order state transition matrix (STM) to compute corrections.

The existing methods are therefore limited to situations where a good initial guess is available and the dynamics are well-approximated with the first-order partial derivatives. They can also require repeated integrations of the dynamics. This can work well for simple dynamical models, but can constitute a prohibitively expensive computational effort when full-fidelity models with perturbations are included, as they would be when performing ground-based maneuver planning. In order to perform these computations rapidly (and eventually on a flight computer), the dynamics would need to be either simplified to make numerical integration feasible (which may not achieve the desired accuracy for spacecraft operating in highly nonlinear systems), or approximated so that the bulk of the computational effort is completed offline, or prior to the maneuver.

1.2 Background

1.2.1 Current methods for efficient spacecraft guidance

There are a number of ongoing research efforts seeking to address these issues. Several recent works have focused on applying advances in machine learning to spacecraft guidance and trajectory optimization problems, with the objective of learning the controls required to correct for state deviations or navigation errors. For example, Parrish [94] and Izzo et al.[59] used neural networks to compute corrections to a low-thrust trajectory. Sullivan and Bosanac[105] and Lafarge et al.[67] applied reinforcement learning methods to develop low-thrust guidance laws for perturbed trajectories in the circular-restricted three body problem (CR3BP). Bonasera et al. [14] used reinforcement learning to compute impulsive stationkeeping maneuvers for a spacecraft operating in a Sun-Earth halo orbit. Another recent advancement in developing efficient, autonomous guidance methods is the *Theory of Functional Connections*, proposed and developed by Daniele Mortari[78, 46]. Convex optimization methods [55, 77] have also been shown to perform efficiently for spacecraft guidance.

1.2.2 Higher-order numerical methods in astrodynamics

An alternative approach for efficient guidance and maneuver design is to expand the reference trajectory in terms of its higher-order derivatives, and integrate and store these over the desired timespan. A number of higher-order methods for spacecraft state propagation have been researched by the astrodynamics community, particularly for uncertainty propagation and space-situational awareness applications. Such methods include state transition tensors (STTs) [93, 118] and differential algebra (DA) [116, 110]. These techniques can be used to obtain accurate approximations of the variations of a dynamical system around a reference; in fact, they can be thought of as a higher-order extension of the first-order state transition matrix (STM). Higher-order methods have been demonstrated for a variety of applications in astrodynamics. Although the integration of the higher-order terms in itself can be computationally intensive, once complete for a given reference trajectory, the resulting terms can be repeatedly evaluated algebraically in order to pre-

dict the effect of any state deviation or control on the spacecraft's state, removing the need for repeated on-board or real-time integrations. This can allow for significantly faster computation of the controls required to reach a desired target on (or near) the reference. In addition, because they include higher-order derivative information, these methods can more accurately capture the effects of nonlinear dynamics. The area of convergence for these methods should therefore be larger than linearized methods. A guidance scheme based on these methods could enable robust, near-real-time trajectory replanning in reaction to navigational errors, off-nominal performance, or unforeseen events. These methods are particularly well-suited for use in deep-space flight dynamics because, for this class of dynamic system, the dynamics are highly nonlinear but well-modeled, and have relatively low uncertainties when compared with systems such as autonomous vehicles or atmospheric reentry systems.

The concept of STTs was introduced to the astrodynamics community by Park and Scheeres [92], and was first applied to nonlinear statistical maneuver design. STTs have since been used primarily for highly accurate nonlinear navigation and uncertainty propagation [93, 72]. Several efforts have also focused on improving the accuracy of computing STTs for two-body dynamics [44, 91]. However, there was little further development in using STTs for control and guidance applications until Ref. [75] was presented, which subsequently formed the foundation of the initial work in this dissertation

Differential algebra (DA) as a technique was originally developed for modeling beam dynamics in the field of particle physics [10], but has since been embraced by the astrodynamics community as a useful tool. As with STTs, DA has been used extensively for nonlinear navigation and uncertainty propagation [116, 110, 111, 3]. DA has also been successfully employed in spacecraft trajectory control problems, including interplanetary transfers [40, 39] and proximity operations [39, 41].

For both STTs and DA, the current applications to control schemes are preliminary and rely on certain key assumptions which limit their use in most applications. The existing works use first-order targeting schemes to converge on control solutions. These require that the system be sufficiently constrained such that there is a unique solution for a given deviation from the reference.

For most control schemes, this is undesirable as it may result in suboptimal or overly constrained solutions. It also cannot allow for solving the optimal control problem with different constraints or cost functions from the reference.

1.2.3 Continuous-thrust trajectory optimization

Trajectory replanning becomes especially cumbersome for continuous or low-thrust trajectories, where the spacecraft is thrusting for a significant portion of the mission. Parametric or feedback guidance laws can result in suboptimal trajectories, which may not sufficiently meet mission requirements. Consequently, for many low-thrust enabled spacecraft, all future controls need to be optimized any time updates are required; this is a difficult task due to the high dimensionality of the underlying control problem. Currently, most of these algorithms are computationally expensive and are run on the ground. For example, for the Dawn mission, which successfully entered into orbit around two asteroids using a low-thrust ion propulsion system for all of its orbital maneuvers, thrust sequences would be redesigned in an open-loop fashion in reaction to navigation errors [89, 90]. These re-design periods were specifically scheduled to maintain a realistic schedule for the Dawn Flight Team. This schedule was found to perform well for the Dawn maneuver operations, but may not be feasible for spacecraft operating in chaotic regions of space with shorter transfer times, such as in cislunar space. Executing these tasks in real-time or on a flight computer is generally not feasible at the moment except for some limited applications, but would certainly be desirable as it would reduce the amount of off-hours support required, and could allow for more responsive thrust sequences.

Existing low-thrust trajectory optimization schemes rely on a variety of techniques including direct optimization methods [52], hybrid methods [88], collocation techniques [97] and differential dynamic programming [69, 5, 32, 114]. As stated previously, without simplifications or approximations these methods are mostly intractable for use on flight computers. Still, decades of research have led to the development and refinement of well-understood tools such as quadratic trust-region and penalty methods [80, 33, 117]. Many of the current proposed rapid low-thrust trajectory

planning schemes cannot fully take advantage of these methods, and can have difficulty proving optimality or incorporating constraints.

Differential algebra has previously been used to succesfully obtain higher-order low-thrust guidance laws [40, 39] in the vicinity of a reference low-thrust trajectory. In these works, the guidance laws have again been sufficiently parameterized such that there is a unique solution for a given state deviation. In addition, the methods cannot allow for different cost functions or constraints from the reference, which can limit their use in practical situations.

1.2.4 Spacecraft guidance under uncertainty

It is important for any spacecraft guidance or maneuver design system to be robust with respect to operational uncertainties, such as navigation and maneuver execution errors. Given the significant upfront cost for any spacecraft mission, guaranteeing mission safety under the presence of uncertainty is critical. Missions operating in highly nonlinear dynamic systems are particularly sensitive to any deviations from the nominal trajectory. Most operational spacecraft guidance methods assume a deterministic system where uncertainty is not considered in the model. Existing nonlinear stochastic control schemes, such as stochastic model predictive control (SMPC) [76] and stochastic DDP [86] are often prohibitively expensive to run without making any assumptions or approximations. As such, there is interest in developing efficient stochastic control algorithms that are suitable for use in real-time or on-board a spacecraft. Several guidance law formulations which take into account state uncertainty information [62, 82] have been specifically developed for spacecraft systems. These formulations typically rely on linearized approximations of the nonlinear dynamics, which may be insufficiently accurate for spacecraft operating in uncertain, chaotic orbital regimes.

The term "stochastic" can refer to several different formulations for incorporating uncertainty in the context of spacecraft control. For example, it can refer to the problem of targeting the mean of a target state distribution [92, 86]. It can also refer to chance-constrained control, where the probability that certain states be "safe" is bounded [84, 82]. Finally, it can refer to the problem of

minimizing the state uncertainty in a system at a given final time [62, 63]. Higher-order methods such as STTs have been used extensively for propagating state uncertainties through nonlinear dynamics. However, since the STT concept was introduced for statistical maneuver targeting in Ref. [92], there has been little additional work building on the concept of using STTs for stochastic control.

Most existing algorithms for stochastic spacecraft control rely on the assumption that the spacecraft state uncertainties obey Gaussian distributions (e.g., [84]). In highly nonlinear dynamics, an initially Gaussian state uncertainty distribution will quickly lose its Gaussian properties as the state is propagated through the dynamics. One approach to address this issue is to express the states in a coordinate system where distributions remain close to Gaussian when propagated through the dynamics, such as modified equinoctial elements [83] or Milankovitch elements [81]. However, for many applications, such as collision avoidance [95], it is most desirable to express the spacecraft state distribution as a non-Gaussian distribution in Cartesian coordinates. Propagating these types of state distributions through nonlinear dynamics can require the use of higher-order methods to accurately capture their non-Gaussian properties.

1.2.5 Efficiency of higher-order methods

STTs and other higher-order methods come with a significant tradeoff in increased storage requirements, and increased computational requirements for both computing the higher-order terms and performing any subsequent mathematical operations involving them. These requirements increase exponentially as the maximum order of STT considered increases. This has limited their adoption in many operational settings. In order to address these issues, several strategies for approximating higher-order STTs have been developed [44, 91, 99]. The methods in these works are straightforward to implement and have been shown to reduce the computational time required to compute the STTs. However, they are restricted to applications in two-body, periodic dynamics. The focus of this dissertation work is on highly nonlinear systems such as the Earth-Moon system that do not always meet these criteria. The existing approximation strategies are therefore not

always suitable for use in these dynamical regimes.

1.3 Dissertation Overview

This dissertation focuses on developing and expanding on the use of higher-order methods such as state transition tensors (STTs) for spacecraft guidance and trajectory optimization. The power and flexibility of these methods has arguably been under-explored in the astrodynamics research community. In this dissertation, several computationally efficient, accurate, and flexible algorithms are derived by using STTs to model the effects of nonlinear dynamics. In doing so, the goal of this thesis is to bridge the current gap that exists between computationally expensive full-fidelity trajectory optimization methods and simplified, linearized spacecraft guidance schemes.

1.4 Thesis Statement

The work completed in this dissertation can be summarized in the following thesis statement: ***Nonlinear methods such as state transition tensors can be used to formulate efficient, accurate, and flexible spacecraft guidance and trajectory optimization algorithms, which could enable future spacecraft missions with highly complex trajectories.***

1.4.1 Organization

This thesis is organized as follows. In Chapter 2, the mathematical concepts that form the foundation of this thesis work are introduced. These include the dynamic models that are used to approximate the dynamics of a spacecraft operating in a two-body or multi-body system. The concept of the state transition tensors (STTs) of a spacecraft trajectory is also introduced, along with their associated mathematical properties.

In Chapter 3, a spacecraft guidance scheme using the STTs of a reference trajectory is derived, with simplifications provided for the case where the controls are modeled as impulsive maneuvers. This scheme is validated on a stationkeeping problem for a spacecraft operating in a halo orbit in the Earth-Moon system. In Chapter 4, numerical methods to expand the STTs of a reference

trajectory with respect to the reference trajectory's final time are derived. These STTs are then used to develop an analytical optimization scheme targeting optimal maneuvers with a variable time-of-flight, with improved performance when compared to the results from Chapter 3.

In Chapter 5, we address the low-thrust trajectory optimization problem by using the higher-order STTs of a reference trajectory to run an analytical approximation of differential dynamic programming (DDP), a second-order trajectory optimization algorithm. The resulting algorithm is able to compute new continuous-thrust trajectories in cislunar space significantly faster than a numerical algorithm.

In Chapter 6, the impulsive spacecraft guidance scheme from Chapters 3 and 4 is extended to consider the nonlinear evolution of state uncertainties. Formulations for targeting the mean state at the final time, applying state chance constraints, and minimizing uncertainty are derived using only the STTs of a reference trajectory. In Chapter 7, an algorithm is derived for maneuver design with non-Gaussian state chance constraints using Gaussian mixture models and iterative risk allocation.

In Chapter 8, the problem of storage and computational requirements for using STTs is addressed. A novel method for approximating the effects of higher-order STTs is introduced called directional state transition tensors (DSTTs). Through the DSTT concept, the most important higher-order information can be stored in a small number of terms. By retaining only these terms, a good approximation of the effects of the full STTs can be achieved while significantly reducing storage and computational requirements.

1.4.2 Contributions

The main contributions of this dissertation can be summarized as:

Spacecraft guidance and maneuver design using STTs:

- Derived a spacecraft guidance scheme using the STTs of a reference trajectory to compute the controls required to return to a reference (Ref. [20])

- Derived numerical methods to expand the STTs of a reference trajectory with respect to the trajectory time-of-flight (Ref. [21])
- Applied the variable time-of-flight STTs to the problem of optimal impulsive spacecraft maneuver targeting (Ref. [21])

Rapid trajectory optimization using STTs and differential dynamic programming

- Developed an algorithm to run differential dynamic programming, a second-order optimization method used for low-thrust trajectory optimization, within higher-order STT approximations of a reference (Ref. [18])
- Applied the STT/DDP algorithm to compute trajectories in the vicinity of the reference with varying initial and target conditions, and using a different cost function from the reference (Ref. [18])
- Demonstrated how the STT/DDP algorithm can be used to expedite large-scale preliminary mission design tradeoff analyses (Ref. [22])

Stochastic control using STTs

- Derived the necessary equations to use a reference trajectory's STTs to analytically formulate mean state constraints, state chance constraints, a minimum-uncertainty cost function, and a cost function maximizing the initial state uncertainty in a trajectory (Ref. [23])
- Derived an algorithm for spacecraft maneuver design with non-Gaussian chance constraints, using Gaussian mixture models and iterative risk allocation (GMM IRA) (Ref. [24])
- Demonstrated how STTs can be used to significantly improve the performance of the GMM IRA algorithm (Ref. [25])

Directional state transition tensors

- Developed a strategy called directional state transition tensors (DSTTs) to approximate the effects of higher-order STTs while requiring significantly fewer terms (Ref. [19])

- Applied the DSTT concept to nonlinear spacecraft state and state uncertainty propagation examples (Ref. [19])

1.5 Associated Publications

1.5.1 Journal articles

(1) (March 2021) S. Boone and J. McMahon, "Orbital guidance using higher-order state transition tensors," *Journal of Guidance, Control, and Dynamics.*

(2) (November 2021) S. Boone and J. McMahon, "Variable time-of-flight spacecraft maneuver targeting using state transition tensors," *Journal of Guidance, Control, and Dynamics.*

(3) (Accepted October 2022) S. Boone and J. McMahon, "Directional state transition tensors for capturing dominant nonlinear effects in orbital dynamics," *Journal of Guidance, Control, and Dynamics.*

(4) (Under review) S. Boone and J. McMahon, "Rapid spacecraft trajectory optimization using state transition tensors and differential dynamic programming," *Journal of Guidance, Control, and Dynamics.*

(5) (In preparation) S. Boone and J. McMahon, "Semi-analytic stochastic spacecraft maneuver design," *Journal of Guidance, Control, and Dynamics.*

(6) (In preparation) S. Boone and J. McMahon, "Spacecraft maneuver design with non-Gaussian chance constraints using Gaussian mixtures and risk allocation," *Journal of Guidance, Control, and Dynamics.*

1.5.2 Peer-reviewed conference papers

(1) S. Boone and J. McMahon, "Non-Gaussian chance-constrained trajectory control using Gaussian mixtures and risk allocation", *2022 Conference on Decision and Control*, Cancun, Mexico, 2022.

(2) S. Boone and J. McMahon, "Semi-analytic spacecraft maneuver design with stochastic constraints", *2022 American Control Conference*, Atlanta, GA, 2022.

1.5.3 Other conference papers and presentations

(1) S. Boone, O. Boodram, J. McMahon, "Improved near rectilinear halo orbit navigation using efficient nonlinear filtering techniques", *45th AAS Guidance, Navigation and Control Conference*, Breckenridge, CO, 2023.

(2) S. Boone and J. McMahon, "Spacecraft maneuver design with non-Gaussian chance constraints using Gaussian mixtures", *2022 AAS/AIAA Astrodynamics Specialist Conference*, Charlotte, NC, 2022.

(3) O. Boodram, S. Boone, J. McMahon, "Efficient nonlinear navigation using directional state transition tensors", *2022 AAS/AIAA Astrodynamics Specialist Conference*, Charlotte, NC, 2022.

(4) T. Kim, S. Boone, J. McMahon, "Higher-order feedback law for low-thrust spacecraft guidance", *2022 AAS/AIAA Astrodynamics Specialist Conference*, Charlotte, NC, 2022.

(5) S. Boone and J. McMahon, "Rapid local trajectory optimization in cislunar space", *44th AAS Guidance, Navigation and Control Conference*, Breckenridge, CO, 2022, (First place in Student Paper Competition)

(6) S. Boone, S. Bonasera, J. McMahon, N. Bosanac, N. Ahmed, "Incorporating observation uncertainty into reinforcement learning-based spacecraft guidance schemes" *2022 AIAA SciTech Forum*, San Diego, CA, 2022.

(7) S. Boone and J. McMahon, "Directional state transition tensors for capturing dominant nonlinear dynamical effects", *2021 AAS/AIAA Astrodynamics Specialist Conference*, Virtual, 2021 (AAS 21-701).

(8) S. Boone and J. McMahon, "Optimal maneuver targeting using state transition tensors with variable time-of-flight," *31st AAS/AIAA Space Flight Mechanics Meeting*, Virtual, 2021 (AAS 21-404), (John V. Breakwell Student Award).

(9) S. Boone and J. McMahon, "Rapid local trajectory optimization using higher-order state transition tensors and differential dynamic programming", *AAS/AIAA Astrodynamics Specialist Conference*, Virtual, 2020 (AAS 20-582).

(10) (Poster presentation) S. Boone and J. McMahon, "Spacecraft trajectory control using higher-order state transition tensors", *2020 American Control Conference*, Virtual, 2020

Chapter 2

Mathematical Background

In this chapter the mathematical foundations that are used in all further chapters of this thesis are presented.

2.1 Spacecraft Dynamics

2.1.1 Two-body dynamics

The equations of motion for the two-body problem (an object with negligible mass orbiting a single point-mass) are

$$\ddot{x} = -\frac{\mu x}{r^3} \qquad \ddot{y} = -\frac{\mu y}{r^3} \qquad \ddot{z} = -\frac{\mu z}{r^3} \tag{2.1}$$

where the state vector $\boldsymbol{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$ contains 3 position (x, y, z) and 3 velocity $(\dot{x}, \dot{y}, \dot{z})$ terms. The distance r is the distance from the spacecraft to the central body (i.e., $r = \sqrt{x^2 + y^2 + z^2}$), and μ is the gravitational parameter for the central body.

2.1.2 Circular restricted three-body problem

The dynamics of a spacecraft operating in a multi-body system such as the Earth-Moon system can be approximated using the circular restricted three-body problem (CR3BP). In the CR3BP, the spacecraft mass is assumed to be negligible in comparison to that of the primary and secondary bodies, which have masses m_1 and m_2, respectively. It is assumed that the masses follow circular orbits about their mutual barycenter. The system is defined by a mass ratio parameter μ,

which is the ratio of the secondary mass to the total system mass:

$$\mu = \frac{m_2}{m_1 + m_2} \tag{2.2}$$

The spacecraft dynamics are modeled in a rotating frame, $(\hat{x}, \hat{y}, \hat{z})$, with the $\hat{x}$ axis directed from the Earth to the Moon, the $\hat{z}$ axis directed along the angular momentum vectors of the primaries, and the $\hat{y}$ axis defined to complete the right-handed cooredinate frame. The spacecraft state is chosen as a Cartesian representation of the spacecraft position and velocity in the synodic frame:

$$\boldsymbol{r} = \begin{bmatrix} x & y & z \end{bmatrix}^T \tag{2.3}$$

$$\boldsymbol{v} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T \tag{2.4}$$

It is common to normalize the CR3BP so that the system mass, angular velocity, and distance between the primaries are all equal to one. The distance from the barycenter to the primary then becomes $-\mu$ and the distance from the barycenter to the secondary becomes $1 - \mu$. We then define the distances $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ as the distances from the spacecraft to the primary and secondary bodies, respectively. These become

$$\boldsymbol{r}_1 = \begin{bmatrix} x + \mu & y & z \end{bmatrix}^T \tag{2.5}$$

$$\boldsymbol{r}_2 = \begin{bmatrix} x - (1 - \mu) & y & z \end{bmatrix}^T \tag{2.6}$$

Using these definitions, the equations of motion for the CR3BP can be stated as

$$\ddot{x} = 2\dot{y} + x - \frac{\mu(-1 + x + \mu)}{r_1^3} - \frac{(1 - \mu)(x + \mu)}{r_2^3} \tag{2.7}$$

$$\ddot{y} = -2\dot{x} + y - \frac{\mu y}{r_1^3} - \frac{y(1 - \mu)}{r_2^3} \tag{2.8}$$

$$\ddot{z} = -\frac{\mu z}{r_1^3} - \frac{z(1 - \mu)}{r_2^3} \tag{2.9}$$

with $r_1 = \|\boldsymbol{r}_1\|$ and $r_2 = \|\boldsymbol{r}_2\|$. The CR3BP possesses five equilibrium points, also referred to as Lagrange points. These are commonly denoted as L_1 through L_5. In the context of the CR3BP,

a number of periodic and quasi-periodic trajectories exist in the vicinity of these Lagrange points. Many of the numerical examples in this thesis will consider the scenario of a spacecraft operating on or near such orbits.

2.2 State Transition Tensors

2.2.1 Definition

As stated in the introduction, state transition tensors (STTs) are effectively a higher-order extension of the commonly-used state transition matrix (STM); in fact, the STM can be thought of as the first-order STT. The basics of their derivation are repeated below, reproduced based on Park and Scheeres [92]. The solution to the evolution of a dynamic system with state vector $\mathbf{x} \in \mathbb{R}^n$ can be described through its solution flow,

$$\mathbf{x}(t) = \boldsymbol{\phi}(t; \mathbf{x}_0, \hat{\mathbf{u}}_{(t,t_0)}, t_0) \tag{2.10}$$

where $\hat{\mathbf{u}}_{(t,t_0)}$ is the time history of all nominal (or reference) controls applied over the interval $[t_0, t)$. Note that, given a control history, the nominal controls can be regarded as part of the "natural" dynamical system so that their effect is wrapped up within the STTs. The STTs are partial matrices of this solution flow with respect to the initial conditions. Thus, the STTs of order p can be defined as

$$\phi^{i,\gamma_1 \ldots \gamma_p}_{(t,t_0)} = \frac{\partial^p \boldsymbol{\phi}^i(t; \mathbf{x}_0, \hat{\mathbf{u}}_{(t,t_0)}, t_0)}{\partial x_0^{\gamma_1} \ldots \partial x_0^{\gamma_p}} \tag{2.11}$$

Index (Einstein summation) notation is used heavily throughout this dissertation. Superscripts indicate components of a vector, matrix, or tensor. The order of a tensor is determined by the number of superscript indices. Subscripts indicate the time of the vector, or for a matrix or tensor, (t, t_0) indicates a mapping from t_0 to t. Repeated indices indicate summation, e.g.,

$$\frac{1}{2}\phi^{i,\gamma_1\gamma_2}_{(t,t_0)} \delta x_0^{\gamma_1} \delta x_0^{\gamma_2} = \sum_{\gamma_1=1}^{n} \sum_{\gamma_2=1}^{n} \frac{1}{2}\phi^{i,\gamma_1\gamma_2}_{(t,t_0)} \delta x_0^{\gamma_1} \delta x_0^{\gamma_2} \tag{2.12}$$

The comma in the superscript indicates that the i-th component is not summed over. The STTs can also be thought of as the partial derivatives of the state vector at time t with respect to the initial state vector at time t_0:

$$\phi_{(t,t_0)}^{i,\kappa_1...\kappa_p} = \frac{\partial^p x^i}{\partial x_0^{\kappa_1}...\partial x_0^{\kappa_p}} \tag{2.13}$$

The solution of the variation of the state at time t around the nominal state can be approximated with STTs as [92]

$$\delta x^i \simeq \sum_{p=1}^{m} \frac{1}{p!} \phi_{(t,t_0)}^{i,\gamma_1...\gamma_p} \delta x_0^{\gamma_1}...\delta x_0^{\gamma_p} \tag{2.14}$$

This corresponds to a Taylor series expansion about the reference trajectory including terms up to order m. As m increases, the approximation for δx^i will generally become more and more accurate, up to the limit $m \to \infty$, where the approximation converges on the true solution.

The integration of the higher-order STTs represents a significant computational burden, and is certainly not feasible to conduct at a large scale on a flight computer, let alone in real time. We can, however, integrate the higher-order STTs of a reference trajectory, and use the resulting STTs to analytically predict the effect of any state deviation on the final state. Because no further integrations of the dynamics are required, these mappings can be performed very efficiently. There will be a region around the reference where the STT mappings are accurate approximations of the true dynamics; outside this region, the STTs will no longer be accurate. The size of this region will depend on the order of STTs included in the approximation.

2.2.2 Computing STTs

The differential equations for integrating the STTs up to fourth-order are[92]

$$\dot{\phi}^{i,a} = A^{i,\alpha}\phi^{\alpha,a} \tag{2.15}$$

$$\dot{\phi}^{i,ab} = A^{i,\alpha}\phi^{\alpha,ab} + A^{i,\alpha\beta}\phi^{\alpha,a}\phi^{\beta,b} \tag{2.16}$$

$$\dot{\phi}^{i,abc} = A^{i,\alpha}\phi^{\alpha,abc} + A^{i,\alpha\beta}(\phi^{\alpha,a}\phi^{\beta,bc} + \phi^{\alpha,ab}\phi^{\beta,c} + \phi^{\alpha,ac}\phi^{\beta,b}) + A^{i,\alpha\beta\gamma}\phi^{\alpha,a}\phi^{\beta,b}\phi^{\gamma,c} \tag{2.17}$$

$$\begin{aligned}\dot{\phi}^{i,abcd} =& A^{i,\alpha}\phi^{\alpha,abcd} + A^{i,\alpha\beta}(\phi^{\alpha,abc}\phi^{\beta,d} + \phi^{\alpha,abd}\phi^{\beta,c} + \phi^{\alpha,ac}\phi^{\beta,b} + \phi^{\alpha,ab}\phi^{\beta,cd} + \phi^{\alpha,ac}\phi^{\beta,bd} \\ &+ \phi^{\alpha,ad}\phi^{\beta,bc} + \phi^{\alpha,a}\phi^{\beta,bcd}) + A^{i,\alpha\beta\gamma}(\phi^{\alpha,ab}\phi^{\beta,c}\phi^{\gamma,d} + \phi^{\alpha,ac}\phi^{\beta,b}\phi^{\gamma,d} + \phi^{\alpha,ad}\phi^{\beta,b}\phi^{\gamma,c} \\ &+ \phi^{\alpha,a}\phi^{\beta,bc}\phi^{\gamma,d} + \phi^{\alpha,a}\phi^{\beta,bd}\phi^{\gamma,c} + \phi^{\alpha,a}\phi^{\beta,b}\phi^{\gamma,cd}) + A^{i,\alpha\beta\gamma\delta}\phi^{\alpha,a}\phi^{\beta,b}\phi^{\gamma,c}\phi^{\delta,d}\end{aligned} \tag{2.18}$$

The integration of the STTs necessitates knowledge of the A tensors, which represent the partial derivatives of the state rates with respect to the state. These tensors can be analytically derived, though beyond the second order, this will generally become prohibitively tedious. For simple dynamics, these can be derived using symbolic manipulators such as Mathematica or SymPy. However, for complex dynamical systems with many perturbations, the resulting equations are often not optimally formulated and must be repeatedly re-derived each time the dynamics equations are modified. Thus, it is beneficial to employ some form of automatic differentiation. For all STT integrations in this work, we made use of the freely-available PyAudi package developed by the European Space Agency [58], which uses the Taylor polynomial automatic differentiation method.

2.2.3 Multiplying STTs

Reference trajectory STTs can be segmented and integrated separately from t_0 to t_k, and from t_k to t_f. The STTs mapping from t_0 to t_f can then be obtained using a combination of these STTs. For the first order, this results in the well-known STM multiplication property shown in Eq. 2.19. For the second through fourth orders, Eqs. 2.20 - 2.22 define these relations. Letting $\phi_{k0} = \phi_{(t_k,t_0)}$ and $\phi_{fk} = \phi_{(t_f,t_k)}$,

$$\phi^{i,a}_{(t_f,t_0)} = \phi^{i,\alpha}_{fk}\phi^{\alpha,a}_{k0} \tag{2.19}$$

$$\phi^{i,ab}_{(t_f,t_0)} = \phi^{i,\alpha}_{fk}\phi^{\alpha,ab}_{k0} + \phi^{i,\alpha\beta}_{fk}\phi^{\alpha,a}_{k0}\phi^{\beta,b}_{k0} \tag{2.20}$$

$$\phi^{i,abc}_{(t_f,t_0)} = \phi^{i,\alpha}_{fk}\phi^{\alpha,abc}_{k0} + \phi^{i,\alpha\beta}_{fk}\left(\phi^{\alpha,a}_{k0}\phi^{\beta,bc}_{k0} + \phi^{\alpha,ab}_{k0}\phi^{\beta,c}_{k0} + \phi^{\alpha,ac}_{k0}\phi^{\beta,b}_{k0}\right) + \phi^{i,\alpha\beta\gamma}_{k0}\phi^{\alpha,a}_{k0}\phi^{\beta,b}_{k0}\phi^{\gamma,c}_{k0} \tag{2.21}$$

$$\begin{aligned}\phi^{i,abcd}_{(t_f,t_0)} = \quad & \phi^{i,\alpha}_{fk}\phi^{\alpha,abcd}_{k0} + \phi^{i,\alpha\beta}_{fk}(\phi^{\alpha,abc}_{k0}\phi^{\beta,d}_{k0} + \phi^{\alpha,abd}_{k0}\phi^{\beta,c}_{k0} + \phi^{\alpha,ac}_{k0}\phi^{\beta,b}_{k0} + \phi^{\alpha,ab}_{k0}\phi^{\beta,cd}_{k0} + \phi^{\alpha,ac}_{k0}\phi^{\beta,bd}_{k0} \\ & + \phi^{\alpha,ad}_{k0}\phi^{\beta,bc}_{k0} + \phi^{\alpha,a}_{k0}\phi^{\beta,bcd}_{k0}) + \phi^{i,\alpha\beta\gamma}_{fk}(\phi^{\alpha,ab}_{k0}\phi^{\beta,c}_{k0}\phi^{\gamma,d}_{k0} + \phi^{\alpha,ac}_{k0}\phi^{\beta,b}_{k0}\phi^{\gamma,d}_{k0} + \phi^{\alpha,ad}_{k0}\phi^{\beta,b}_{k0}\phi^{\gamma,c}_{k0} \\ & + \phi^{\alpha,a}_{k0}\phi^{\beta,bc}_{k0}\phi^{\gamma,d}_{k0} + \phi^{\alpha,a}_{k0}\phi^{\beta,bd}_{k0}\phi^{\gamma,c}_{k0} + \phi^{\alpha,a}_{k0}\phi^{\beta,b}_{k0}\phi^{\gamma,cd}_{k0}) + \phi^{i,\alpha\beta\gamma\delta}_{fk}\phi^{\alpha,a}_{k0}\phi^{\beta,b}_{k0}\phi^{\gamma,c}_{k0}\phi^{\delta,d}_{k0}\end{aligned} \tag{2.22}$$

2.2.4 STT derivatives

In general, $\phi^{i,\gamma_1\ldots\gamma_p}_{(t,t_0)}$ will be symmetric along the $\gamma_1\ldots\gamma_p$ axes. This means that the first-order derivatives of the STT approximation for $\delta\mathbf{x}$ at time t in Eqn. 2.14, with respect to the state deviation $\delta\mathbf{x}_0$ at time t_0, can be expressed analytically solely as a function of the reference STTs mapping from t_0 to t:

$$\frac{\partial(\delta x^i)}{\partial(\delta x_0^{\gamma_1})} = \phi^{i,\gamma_1}_{(t,t_0)} + \sum_{p=2}^{m}\frac{1}{(p-1)!}\phi^{i,\gamma_1\gamma_2\ldots\gamma_p}_{(t,t_0)}\delta x_0^{\gamma_2}\ldots\delta x_0^{\gamma_p} \tag{2.23}$$

Note that for the $m = 2$ case, the summation in Eq. 2.23 becomes an empty sum, giving

$$\frac{\partial(\delta x^i)}{\partial(\delta x_0^{\gamma_1})} = \phi^{i,\gamma_1}_{(t,t_0)} \tag{2.24}$$

The second-order derivatives of Eqn. 2.14 can also be expressed analytically as a function of the reference STTs:

$$\frac{\partial^2(\delta x^i)}{\partial(\delta x_0^{\gamma_1})\partial(\delta x_0^{\gamma_2})} = \phi^{i,\gamma_1\gamma_2}_{(t,t_0)} + \sum_{p=3}^{m}\frac{1}{(p-2)!}\phi^{i,\gamma_1\gamma_2\gamma_3\ldots\gamma_p}_{(t,t_0)}\delta x_0^{\gamma_3}\ldots\delta x_0^{\gamma_p} \tag{2.25}$$

The procedure to derive these equations can easily be extended to obtain the third and higher-order derivatives of the STT approximation.

Chapter 3

Orbital Guidance using State Transition Tensors

3.1 Introduction

As state in the introduction, a robust on-board or real-time maneuver planning and guidance capability would reduce both the reliance on ground contacts for commanding maneuvers as well as the burden of routine navigational support. The main on-board methods that have been previously implemented focus on either neighboring optimal methods, which solve a linear quadratic regulation (LQR) or tracking problem[104], or more commonly the parametric methods that use a predictor-corrector to solve an initial value problem (IVP) to hit a targeted state or terminal function without explicit regard for optimality[28, 30]. In either of these methods, the main optimization of the reference trajectory has been carried out offline, and the on-board guidance method is concerned with either steering back to the nominal trajectory, or with solving an IVP from the current state to achieve the targeted state or orbit. The linearized predictor-corrector methods have proven successful for spacecraft operating in low-Earth orbit, particularly upper-stage launch vehicles, but their convergence region in highly nonlinear systems can be very small.

This chapter focuses on providing the mathematical derivations for an STT-based guidance problem. The objective of this chapter is to develop a higher-order method that can incorporate any number of perturbations while remaining both computationally efficient and accurate, which would make it suitable for real-time or on-board use in operational applications. This chapter is organized as follows: in Section 3.2 we review the formulation of the standard predictor-corrector method, and highlight key scenarios where the method may be insufficient. We derive the full STT expansions

up to the 4th order in terms of the control variations in Section 3.3, with simplifications provided for using an impulsive $\Delta\mathbf{v}$ control in place of continuous control. The STT guidance problem with constraint, error, and targeting equations is developed in Section 3.4. Numerical methods for automating the computation of the dynamics partials are discussed in Section 3.5. Finally, in Section 3.6, we present an example comparing the standard numerical predictor-corrector approach with the STT-based method for a stationkeeping problem in an unstable Earth-Moon L_1-orbiting halo orbit.

3.2 Predictor-Corrector Method

The predictor-corrector method uses an iterative shooting method to predict the effects of changing the control variables on the target constraints. This method is also commonly referred to as a differential corrector or single shooter algorithm. Many commercial software packages use this method in their native targeters. This formulation is agnostic to what method is used to construct the control - it can be used for targeting impulsive $\Delta\mathbf{v}$'s as well as targeting parametric guidance law parameters. The problem can be formulated as

$$-\mathbf{e} = \Gamma\delta\mathbf{u} \tag{3.1}$$

where $\mathbf{e} = \mathbf{C}_f(\mathbf{x}+\delta\mathbf{x},\mathbf{u}) - \mathbf{C}_f^*$ is the error in the target constraint $\mathbf{C}_f$ with respect to the reference constraint $\mathbf{C}_f^*$ at the desired target time, from propagating the current state $\mathbf{x}_0 + \delta\mathbf{x}_0$ to the final time. Here, $\delta\mathbf{x}_0$ is an error in the current state from the nominal reference trajectory at the initial time t_0. Note that we use superscripts throughout this thesis to indicate the components of a vector, matrix or tensor. Γ is the control partials matrix

$$\Gamma = \frac{\partial\mathbf{C}_f}{\partial\mathbf{u}}|_{\mathbf{u}+\delta u^i} \tag{3.2}$$

which is computed via a single control perturbation shooting method such that

$$\Gamma^{a,i} = \frac{C_f^a(\mathbf{x}+\delta\mathbf{x},\mathbf{u}+\delta u^i) - C_f^a(\mathbf{x}+\delta\mathbf{x},\mathbf{u})}{\delta u^i} \tag{3.3}$$

This requires $M+1$ integrations of the dynamics for each iteration, where M is the number of control variables in $\mathbf{u}$.

As discussed in the introduction for this chapter, this method has been shown to work well for a large number of applications, especially since the dynamics are re-integrated at each iteration. However, the fact that this method requires repeated on-board integrations each time the state changes (for example, due to navigation and performance errors) limits its use to situations where the dynamics model is sufficiently simple such that real-time on-board integration is feasible. Other key weaknesses of this method are the linear assumption - which can lead to divergence of successive guesses if the dynamics are highly nonlinear - and the fact that the Γ matrix is constructed through finite differencing using some pre-chosen values for $\delta\mathbf{u}$. Trial and error may be required to determine the best values of $\delta\mathbf{u}$ for the given problem.

3.3 State Transition Tensor Control

The previous section reviewed how STTs can be used to map the evolution of the perturbed state assuming that the controls stay at their nominal values. In this section, we explore how we can similarly use STTs to map the effects of a change in the controls. This will form the basis of the guidance problem.

Recall from Eq. 2.10 that the solution flow is a function of both the state and the control history. Since this is a nonlinear problem, these effects will become coupled. In a similar manner to the derivation of the STTs, we can derive the change in the state due to variations in the control as

$$\begin{aligned}\delta x_f^i \simeq & \sum_{p=1}^{m} \frac{1}{p!} \phi_{(t_f,t_0)}^{i,\gamma_1\ldots\gamma_p} \delta x_0^{\gamma_1} \ldots \delta x_0^{\gamma_p} \\ & + \sum_{\tau=t_0}^{t_f-1} \left[\sum_{q=1}^{m} \frac{1}{q!} \beta_{(t_f,\tau)}^{i,\kappa_1\ldots\kappa_q} \delta u_\tau^{\kappa_1} \ldots \delta u_\tau^{\kappa_q} \right. \\ & \left. + \sum_{q=1}^{m-1} \sum_{p=1}^{m-q} \frac{1}{p!q!} \Xi_{(t_f,\tau)}^{i,\kappa_1\ldots\kappa_q,\gamma_1\ldots\gamma_p} \delta u_\tau^{\kappa_1} \ldots \delta u_\tau^{\kappa_q} \delta x_\tau^{\gamma_1} \ldots \delta x_\tau^{\gamma_p} \right]\end{aligned} \tag{3.4}$$

where τ indicates a number of discrete times for control applications. To solve the full planning

problem, all of these points where the control could be modified would need to be addressed; however, to make the problem more tractable for the initial demonstrations in this work, we will only plan for the current epoch ($t_0 = 0$).

The mapping of the effects of the controls to the final state are given via

$$\beta^{i,\kappa_1 \ldots \kappa_q}_{(t_f,t_0)} = \frac{\partial^q \phi^i(t_f; \mathbf{x}_0, U^*_{(t_f,t_0)}, t_0)}{\partial u_0^{\kappa_1} \ldots \partial u_0^{\kappa_q}} \tag{3.5}$$

One can recognize that the solution flow moves through all time, $t \in [t_0, t_f]$, and is a function of both $\mathbf{x}_k$ and U^*. As such, we can write the solution flow as

$$\mathbf{x}_f = \boldsymbol{\phi}(t_f; \mathbf{x}_k, U^*_{(t_f,t_k)}, t_k) = \boldsymbol{\phi}(t_f; \boldsymbol{\phi}(t_k; \mathbf{x}_0, U^*_{(t_k,t_0)}, t_0), U^*_{(t_f,t_k)}, t_k) \tag{3.6}$$

We can thus see that the partials due to the controls, β, can be evaluated via the chain rule with respect to $\mathbf{x}_k$ and $\mathbf{u}_0$. For the first order, $q = 1$, this becomes

$$\beta^{i,\kappa_1}_{(t_f,t_0)} = \frac{\partial x_f^i}{\partial x_k^{\gamma_1}} \frac{\partial x_k^{\gamma_1}}{\partial u_0^{\kappa_1}} \tag{3.7}$$

where the state vectors have been used in place of the solution flow to improve readability. The first term is recognized to simply be the STM, $\phi^{i,\gamma_1}_{(t_f,t_k)}$, while the second term is simply a different β such that we can write

$$\beta^{i,\kappa_1}_{(t_f,t_0)} = \phi^{i,\gamma_1}_{(t_f,t_k)} \beta^{\gamma_1,\kappa_1}_{(t_k,t_0)} \tag{3.8}$$

The control partials up to $q = 4$ are presented in Eqs. 3.9 - 3.11. In what follows, the time indices will be left out of the right-hand side of the equations, with the understanding that all follow the conventions $\phi_{(t_f,t_k)}$ and $\beta_{(t_k,t_0)}$. Thus we find

$$\beta^{i,\kappa_1\kappa_2}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2} \beta^{\gamma_1,\kappa_1} \beta^{\gamma_2,\kappa_2} + \phi^{i,\gamma_1} \beta^{\gamma_1,\kappa_1\kappa_2} \tag{3.9}$$

$$\beta^{i,\kappa_1\kappa_2\kappa_3}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2\gamma_3} \beta^{\gamma_1,\kappa_1} \beta^{\gamma_2,\kappa_2} \beta^{\gamma_3,\kappa_3} + 3\phi^{i,\gamma_1\gamma_2} \beta^{\gamma_1,\kappa_1\kappa_2} \beta^{\gamma_2,\kappa_3} + \phi^{i,\gamma_1} \beta^{\gamma_1,\kappa_1\kappa_2\kappa_3} \tag{3.10}$$

$$\begin{aligned} \beta^{i,\kappa_1\kappa_2\kappa_3\kappa_4}_{(t_f,t_0)} =& \phi^{i,\gamma_1\gamma_2\gamma_3\gamma_4} \beta^{\gamma_1,\kappa_1} \beta^{\gamma_2,\kappa_2} \beta^{\gamma_3,\kappa_3} \beta^{\gamma_4,\kappa_4} + 6\phi^{i,\gamma_1\gamma_2\gamma_3} \beta^{\gamma_1,\kappa_1\kappa_2} \beta^{\gamma_2,\kappa_3} \beta^{\gamma_3,\kappa_4} \\ &+ 4\phi^{i,\gamma_1\gamma_2} \beta^{\gamma_1,\kappa_1\kappa_2\kappa_3} \beta^{\gamma_2,\kappa_4} + 3\phi^{i,\gamma_1\gamma_2} \beta^{\gamma_1,\kappa_1\kappa_2} \beta^{\gamma_2,\kappa_3\kappa_4} + \phi^{i,\gamma_1} \beta^{\gamma_1,\kappa_1\kappa_2\kappa_3\kappa_4} \end{aligned} \tag{3.11}$$

The nonlinear effects of variations in both the state and the controls can be captured by the Ξ partials. These only exist at 2nd order, $p+q=2$, at a minimum. The previous results can be used to define the Ξ partials, which are of the form

$$\Xi^{i,\kappa_1\ldots\kappa_q,\gamma_1\ldots\gamma_p}_{(t_f,t_0)} = \frac{\partial^{q+p}\phi^i(t_f;\mathbf{x}_0,U^*_{(t_f,t_0)},t_0)}{\partial u_0^{\kappa_1}\ldots\partial u_0^{\kappa_q}\partial x_0^{\gamma_1}\ldots\partial x_0^{\gamma_p}} \tag{3.12}$$

with the understanding that any higher-order partial derivative which takes the partial of some $\mathbf{x}$ with respect to itself as well as some $\mathbf{u}$ is zero, for example

$$\frac{\partial^2 x_k^{\gamma_1}}{\partial x_k^{\gamma_2}\partial u_0^{\kappa_1}} = 0 \tag{3.13}$$

$$\frac{\partial^3 x_k^{\gamma_1}}{\partial x_k^{\gamma_2}\partial u_0^{\kappa_1}\partial u_0^{\kappa_2}} = 0 \tag{3.14}$$

Using this fact, and the same implicit time notation on the right-hand side as in Eqs. 3.8-3.11, the Ξ partials to fourth order (up to $p+q=4$) are

$$\Xi^{i,\kappa_1,\gamma_1}_{(t_f,t_0)} = \phi^{i,ab}\beta^{a,\kappa_1}\theta^{b,\gamma_1} + \phi^{i,a}\Xi^{a,\kappa_1,\gamma_1} \tag{3.15}$$

$$\Xi^{i,\kappa_1,\gamma_1\gamma_2}_{(t_f,t_0)} = \phi^{i,abc}\beta^{a,\kappa_1}\theta^{b,\gamma_1}\theta^{c,\gamma_2} + \phi^{i,ab}\left(2\Xi^{a,\kappa_1,\gamma_1}\theta^{b,\gamma_2} + \beta^{a,\kappa_1}\theta^{b,\gamma_1\gamma_2}\right) + \phi^{i,a}\Xi^{a,\kappa_1,\gamma_1\gamma_2} \tag{3.16}$$

$$\Xi^{i,\kappa_1\kappa_2,\gamma_1}_{(t_f,t_0)} = \phi^{i,abc}\beta^{a,\kappa_1}\beta^{b,\kappa_2}\theta^{c,\gamma_1} + \phi^{i,ab}\left(2\Xi^{a,\kappa_1,\gamma_1}\beta^{b,\kappa_2} + \beta^{a,\kappa_1\kappa_2}\theta^{b,\gamma_1}\right) + \phi^{i,a}\Xi^{a,\kappa_1\kappa_2,\gamma_1} \tag{3.17}$$

$$\begin{aligned}\Xi^{i,\kappa_1,\gamma_1\gamma_2\gamma_3}_{(t_f,t_0)} = {} & \phi^{i,abcd}\beta^{a,\kappa_1}\theta^{b,\gamma_1}\theta^{c,\gamma_2}\theta^{d,\gamma_3} + \phi^{i,abc}\left(3\Xi^{a,\kappa_1,\gamma_1}\theta^{b,\gamma_2}\theta^{c,\gamma_3} + 3\beta^{a,\kappa_1}\theta^{b,\gamma_1\gamma_2}\theta^{c,\gamma_3}\right) \\ & + \phi^{i,ab}\left(3\Xi^{a,\kappa_1,\gamma_1\gamma_2}\theta^{b,\gamma_3} + 3\Xi^{a,\kappa_1,\gamma_1}\theta^{b,\gamma_2\gamma_3} + \beta^{a,\kappa_1}\theta^{b,\gamma_1\gamma_2\gamma_3}\right) + \phi^{i,a}\Xi^{a,\kappa_1,\gamma_1\gamma_2\gamma_3}\end{aligned} \tag{3.18}$$

$$\begin{aligned}\Xi^{i,\kappa_1\kappa_2,\gamma_1\gamma_2}_{(t_f,t_0)} = \; & \phi^{i,abcd}\beta^{a,\kappa_1}\beta^{b,\kappa_2}\theta^{c,\gamma_1}\theta^{d,\gamma_2} \\ & + \phi^{i,abc}\left(4\Xi^{a,\kappa_1,\gamma_1}\beta^{b,\kappa_2}\theta^{c,\gamma_2} + \beta^{a,\kappa_1}\beta^{b,\kappa_2}\theta^{c,\gamma_1\gamma_2} + \beta^{a,\kappa_1\kappa_2}\theta^{b,\gamma_1}\theta^{c,\gamma_2}\right) \\ & + \phi^{i,ab}\left(2\Xi^{a,\kappa_1,\gamma_1\gamma_2}\beta^{b,\kappa_3} + 2\Xi^{a,\kappa_1,\gamma_1}\Xi^{a,\kappa_2,\gamma_2} + 2\Xi^{a,\kappa_1\kappa_2,\gamma_1}\theta^{b,\gamma_2} + \beta^{a,\kappa_1\kappa_2}\theta^{b,\gamma_1\gamma_2}\right) \\ & + \phi^{i,a}\Xi^{a,\kappa_1\kappa_2,\gamma_1\gamma_2} \end{aligned} \tag{3.19}$$

$$\begin{aligned}\Xi^{i,\kappa_1\kappa_2\kappa_3,\gamma_1}_{(t_f,t_0)} = \; & \phi^{i,abcd}\beta^{a,\kappa_1}\beta^{b,\kappa_2}\beta^{c,\kappa_3}\theta^{d,\gamma_1} + \phi^{i,abc}\left(3\Xi^{a,\kappa_1,\gamma_1}\beta^{b,\kappa_2}\beta^{c,\kappa_3} + 3\beta^{a,\kappa_1}\beta^{b,\kappa_1\kappa_2}\theta^{c,\gamma_1}\right) \\ & + \phi^{i,ab}\left(3\Xi^{a,\kappa_1,\kappa_2,\gamma_1}\beta^{b,\kappa_3} + 3\Xi^{a,\kappa_1,\gamma_1}\beta^{b,\kappa_2\kappa_3} + \beta^{a,\kappa_1\kappa_2\kappa_3}\theta^{b,\gamma_1}\right) + \phi^{i,a}\Xi^{a,\kappa_1\kappa_2\kappa_3,\gamma_1} \end{aligned} \tag{3.20}$$

where a, b, c, and d are extra internal indices used to carry out the chain rule multiplications, and where we have defined $\theta^{i,\gamma_1\cdots\gamma_p}$, in order to avoid having to explicitly state the timespan of the STTs, as

$$\theta^{i,\gamma_1\cdots\gamma_p} = \phi^{i,\gamma_1\cdots\gamma_p}_{(t_k,t_0)} \tag{3.21}$$

To this point, we have only made the problem simpler if we can solve the control partials $\beta_{(t_k,t_0)}$ and cross-coupled partials $\Xi_{(t_k,t_0)}$ for some time t_k. Generally speaking, this will not be analytically tractable. Even for the familiar linear case, the control contribution to the final state must be evaluated numerically

Until now, we have been examining a generic, controlled dynamical system. However we will now apply some assumptions about our intended application of spacecraft trajectory control to simplify the problem. In spacecraft trajectory control, the control parameters that make up $\mathbf{u}$ are some set that control the thrust vector, $\mathbf{a}_T$. This thrust vector applies a continuous force to the spacecraft that, when changed in orientation and/or magnitude, will change the trajectory of the spacecraft. When specifically discussing spacecraft trajectories, it is often useful to use the impulsive assumption, which states that over some Δt, the applied continuous thrust can be

approximated as a discrete change in velocity

$$\Delta\mathbf{v} = \int_0^{\Delta t} \mathbf{a}_T(\tau)\mathrm{d}\tau \tag{3.22}$$

and this change in velocity can be used to determine the change in the state due to the controls applied over the time Δt. The key to this assumption being valid is that Δt must be small enough such that the position of the spacecraft does not change appreciably over this period. In the extreme, setting $\Delta t \to 0$ further simplifies the modeling of the application of the thrust.

In applying the impulsive assumption to the partials in Eqs. 3.8 - 3.11 and Eqs. 3.15 - 3.20, all that changes is that $t_k \to t_0$ on the right hand side. This greatly simplifies the $\Xi_{(t_f,t_0)}$ terms in Eqs. 3.15 - 3.20 as all $\Xi_{(t_0,t_0)}$ and higher-order $\theta_{(t_0,t_0)}$ terms will become zero, and the first-order $\theta_{(t_0,t_0)}$ term is unity. We thus obtain the following equations for the Ξ partials up to fourth order

$$\Xi^{i,\kappa_1,\gamma_1}_{(t_f,t_0)} = \phi^{i,\gamma_1 a}\beta^{a,\kappa_1} \tag{3.23}$$

$$\Xi^{i,\kappa_1,\gamma_1\gamma_2}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2 a}\beta^{a,\kappa_1} \tag{3.24}$$

$$\Xi^{i,\kappa_1\kappa_2,\gamma_1}_{(t_f,t_0)} = \phi^{i,\gamma_1 ab}\beta^{a,\kappa_1}\beta^{b,\kappa_2} + \phi^{i,\gamma_1 a}\beta^{a,\kappa_1\kappa_2} \tag{3.25}$$

$$\Xi^{i,\kappa_1,\gamma_1\gamma_2\gamma_3}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2\gamma_3 a}\beta^{a,\kappa_1} \tag{3.26}$$

$$\Xi^{i,\kappa_1\kappa_2,\gamma_1\gamma_2}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2 ab}\beta^{a,\kappa_1}\beta^{b,\kappa_2} + \phi^{i,\gamma_1\gamma_2 a}\beta^{a,\kappa_1\kappa_2} \tag{3.27}$$

$$\Xi^{i,\kappa_1\kappa_2\kappa_3,\gamma_1}_{(t_f,t_0)} = \phi^{i,\gamma_1 abc}\beta^{a,\kappa_1}\beta^{b,\kappa_2}\beta^{c,\kappa_3} + 3\phi^{i,\gamma_1 ab}\beta^{a,\kappa_1\kappa_2}\beta^{b,\kappa_3} + \phi^{i,\gamma_1 a}\beta^{a,\kappa_1\kappa_2\kappa_3} \tag{3.28}$$

The impulsive assumption does not lead to a lack of control authority because an impulsive control can, in fact, modify the state directly at the time of the impulse. Furthermore, the actual

control parameters have changed - whereas before they were dependent on $\mathbf{a}_T$, they will now control the impulse $\Delta\mathbf{v}$. We can make an additional assumption that the control vector is an impulsive change solely in the velocity state components at time t_0

$$\delta\mathbf{u}_0 = \Delta\mathbf{v}|_{t_0} \tag{3.29}$$

This form of the control vector further simplifies the control and cross-coupled partials. Only the first partials in the state with respect to the control are non-zero - all higher-order partials with respect to the control disappear, giving

$$\beta^{\gamma_1,\kappa_1\ldots\kappa_p}_{(t_0,t_0)} = 0 \tag{3.30}$$

for $p > 1$. Furthermore, the control only affects the velocity in the same direction; thus for a three-dimensional system the control vector will be of length 3, and the first-order $\beta^{\gamma_1,\kappa_1}_{(t_0,t_0)}$ term will be of dimension 6×3. As a result,

$$\beta^{\gamma_1,\kappa_1}_{(t_0,t_0)} = 1 \tag{3.31}$$

for $\kappa_1 = \gamma_1 - 3$ and $\gamma_1 \in 4, 5, 6$. All other entries are zero. Using these facts, the control partials in Eqs. 3.9 - 3.11 simplify to

$$\beta^{i,\kappa_1\kappa_2}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2}\beta^{\gamma_1,\kappa_1}\beta^{\gamma_2,\kappa_2} \tag{3.32}$$

$$\beta^{i,\kappa_1\kappa_2\kappa_3}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2\gamma_3}\beta^{\gamma_1,\kappa_1}\beta^{\gamma_2,\kappa_2}\beta^{\gamma_3,\kappa_3} \tag{3.33}$$

$$\beta^{i,\kappa_1\kappa_2\kappa_3\kappa_4}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2\gamma_3\gamma_4}\beta^{\gamma_1,\kappa_1}\beta^{\gamma_2,\kappa_2}\beta^{\gamma_3,\kappa_3}\beta^{\gamma_4,\kappa_4} \tag{3.34}$$

and the cross-coupled partials from Eqs. 3.23 - 3.28 simplify to

$$\Xi^{i,\kappa_1,\gamma_1}_{(t_f,t_0)} = \phi^{i,\gamma_1 a}\beta^{a,\kappa_1} \tag{3.35}$$

$$\Xi^{i,\kappa_1,\gamma_1\gamma_2}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2 a}\beta^{a,\kappa_1} \tag{3.36}$$

$$\Xi^{i,\kappa_1\kappa_2,\gamma_1}_{(t_f,t_0)} = \phi^{i,\gamma_1 ab}\beta^{a,\kappa_1}\beta^{b,\kappa_2} \tag{3.37}$$

$$\Xi^{i,\kappa_1,\gamma_1\gamma_2\gamma_3}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2\gamma_3 a}\beta^{a,\kappa_1} \tag{3.38}$$

$$\Xi^{i,\kappa_1\kappa_2,\gamma_1\gamma_2}_{(t_f,t_0)} = \phi^{i,\gamma_1\gamma_2 ab}\beta^{a,\kappa_1}\beta^{b,\kappa_2} \tag{3.39}$$

$$\Xi^{i,\kappa_1\kappa_2\kappa_3,\gamma_1}_{(t_f,t_0)} = \phi^{i,\gamma_1 abc}\beta^{a,\kappa_1}\beta^{b,\kappa_2}\beta^{c,\kappa_3} \tag{3.40}$$

3.4 State Transition Tensor Guidance

In Section 3.3, it was shown how to use STTs to predict changes in the final state due to variations in the initial state and the control vector. However, in guidance problems, the target conditions are not always the final state itself, but are functions of the final state that act as constraints to achieve the desire target. A notable example for spaceflight would be that after a burn, we would like the spacecraft's final orbit to have certain orbital elements, but we do not necessarily care where on this targeted orbit the spacecraft ends up.

There are two important factors then for computing controls in a guidance system. First, the error equations must be derived to tell the system by how much the constraint equations are being violated - this error must be nulled to achieve the desired target. Second, the "best" update to the control vector must be selected in order to null, or else to minimize, the error. This section explores these aspects of the guidance problem using STTs.

3.4.1 Constraint equations

The constraint equations are assumed broadly to be nonlinear equations of the final state, written as

$$\mathbf{C}_f(\mathbf{x}_f) = \mathbf{C}_f^* \tag{3.41}$$

where $\mathbf{C}_f^* = \mathbf{C}_f(\mathbf{x}_f^*)$ is the set of fixed values that define the target. Note that in general these values could be functions of time; however, in this work we will consider static targets at a fixed time.

The error in the desired set of final constraints can be determined as

$$\Delta\mathbf{C}_f = \mathbf{C}_f(\mathbf{x}_f) - \mathbf{C}_f(\mathbf{x}_f^*) \tag{3.42}$$

which admits expansion in terms of $\delta\mathbf{x}_f$ as

$$\Delta C_f^i \simeq \sum_{r=1}^{s} \frac{1}{r!} \Psi^{i,\eta_1\ldots\eta_r} \delta x_f^{\eta_1} \ldots \delta x_f^{\eta_r} \tag{3.43}$$

where

$$\Psi^{i,\eta_1\ldots\eta_r} = \frac{\partial^r C_f^i}{\partial x_f^{\eta_1} \ldots \partial x_f^{\eta_r}} \tag{3.44}$$

To formulate the guidance problem, we need to understand how changes in the epoch state, $\delta\mathbf{x}_0$, and the controls, $\delta\mathbf{u}_0$, affect the constraint at the final time. We could approach this by expanding with respect to these variables. However, the constraint equation is only an explicit function of $\mathbf{x}_f$. Therefore, our approach is to expand the constraint equation as in Eq. 3.43, and then to substitute in the results from Eq. 3.4. These two approaches are equivalent when all expansions are taken to infinity. However, our chosen approach allows us to work in a STT-centric framework. We choose m, our desired expansion order in the STTs, and then all of the information we gain from that is brought into the constraint adjustment. It is important to understand the difference - if the constraint equation were expanded with respect to the epoch state and control directly, then the higher-order STTs only show up with the same higher-order derivatives of $\mathbf{C}_f$. In our formulation, with a limited order of expansion, the higher-order STTs show up even if the

constraint equation only is expanded to $s = 1$. This is similar to the classical shooting method where the integrator will capture the nonlinear dynamics even when formulating a linear approximation to the constraint adjustment.

First we will concisely write Eq. 3.4 as

$$\delta x_f^i = h_x^i(\delta \mathbf{x}_0) + h_u^i(\delta \mathbf{u}_0) + h_{xu}^i(\delta \mathbf{x}_0, \delta \mathbf{u}_0) \tag{3.45}$$

where each of the h functions are the summations up to order m from Eq. 3.4 dependent on the state or control (or both) at time t_0; for example

$$h_x^i(\delta \mathbf{x}_0) = \sum_{p=1}^{m} \frac{1}{p!} \phi_{(t_f,t_0)}^{i,\gamma_1 \ldots \gamma_p} \delta x_0^{\gamma_1} \ldots \delta x_0^{\gamma_p} \tag{3.46}$$

Thus, upon the substitution of Eq. 3.45 into Eq. 3.43, we obtain

$$\Delta C_f^i = \sum_{r=1}^{s} \frac{1}{r!} \Psi^{i,\eta_1 \ldots \eta_r} \left[h_x^{\eta_1}(\delta \mathbf{x}_0) + h_u^{\eta_1}(\delta \mathbf{u}_0) + h_{xu}^{\eta_1}(\delta \mathbf{x}_0, \delta \mathbf{u}_0)\right] \ldots \left[h_x^{\eta_r}(\delta \mathbf{x}_0) + h_u^{\eta_r}(\delta \mathbf{u}_0) + h_{xu}^{\eta_r}(\delta \mathbf{x}_0, \delta \mathbf{u}_0)\right] \tag{3.47}$$

From this point forward the $\delta \mathbf{x}_0$ and $\delta \mathbf{u}_0$ arguments will be dropped from the h functions for conciseness. Depending on the order of the expansion of the constraint equations, s, these terms can appear multiple times as signified by the trailing subscript.

3.4.2 Error equations

Using the previous notation, the error equation can be written as

$$e^i = \Delta C_f^i - \sum_{r=1}^{s} \frac{1}{r!} \Psi^{i,\eta_1 \ldots \eta_r} h_x^{\eta_1} \ldots h_x^{\eta_r} \tag{3.48}$$

This represents the error in achieving the constraints if nothing is changed from the current control scheme with a state deviation from the nominal trajectory at time t_0. Generally we would want $\Delta C_f^i = 0$, so that this term drops out. However, leaving this term in the equation allows us to have the flexibility to retarget on-the-fly by including a non-zero value if desired in the future.

The goal of our guidance system is to choose a $\delta \mathbf{u}$ to null this error vector as efficiently as possible. The expression that is dependent on the control can quickly become cumbersome, so some

intermediate results are in order. First, we will define

$$g^{\eta_i,\kappa_1\ldots\kappa_q} = \frac{1}{q!}\beta^{\eta_i,\kappa_1\ldots\kappa_q}_{(t_f,t_0)} + \sum_{p=1}^{m-q}\frac{1}{p!q!}\Xi^{\eta_i,\kappa_1\ldots\kappa_q,\gamma_1\ldots\gamma_p}_{(t_f,t_0)}\delta x_0^{\gamma_1}\ldots\delta x_0^{\gamma_p} \tag{3.49}$$

where $q \leq m$, such that for the $q = m$ term, the Ξ summation becomes an empty sum and only the β term remains. This allows us to accumulate all terms of the same order of $\delta\mathbf{u}_0$ (i.e. what we can control). This becomes helpful if we are iterating over a range of $\delta\mathbf{u}_0$ values as we can avoid having to recompute any terms with the initial perturbation $\delta\mathbf{x}_0$ at each iteration.

Using this, we can write

$$h_u^{\eta_i} + h_{xu}^{\eta_i} = \sum_{q=1}^{m} g^{\eta_i,\kappa_1\ldots\kappa_q}\delta u_0^{\kappa_1}\ldots\delta u_0^{\kappa_q} \tag{3.50}$$

We can thus rewrite the error equation that we will use to determine our control as

$$-e^i = \sum_{r=1}^{s}\frac{1}{r!}\Psi^{i,\eta_1\ldots\eta_r}[h_u^{\eta_1} + h_{xu}^{\eta_1}]\ldots[h_u^{\eta_r} + h_{xu}^{\eta_r}] + \sum_{r=2}^{s}\frac{1}{r!}\Psi^{i,\eta_1\ldots\eta_r}\mathbf{H}_r \tag{3.51}$$

where

$$\mathbf{H}_2 = h_x^{\eta_1}[h_u^{\eta_2} + h_{xu}^{\eta_2}] + h_x^{\eta_2}[h_u^{\eta_1} + h_{xu}^{\eta_1}] \tag{3.52}$$

and for $r \geq 3$ this function follows the recursion

$$\mathbf{H}_r = \mathbf{H}_{r-1}[h_x^{\eta_r} + h_u^{\eta_r} + h_{xu}^{\eta_r}] \tag{3.53}$$

At this point it is interesting to directly write out the expressions for the first and second orders of Eq. 3.51 in r. This allows us to formulate the solutions in orders of $\delta\mathbf{u}_0$, which allows for clearer examination of the effects of the controls. For $s = 1$ this becomes

$$-e^i = \Psi^{i,\eta_1}\sum_{q=1}^{m} g^{\eta_1,\kappa_1\ldots\kappa_q}\delta u_0^{\kappa_1}\ldots\delta u_0^{\kappa_q} \tag{3.54}$$

Upon examination of this equation, there are terms in the control vector up to order m.

For $s = 2$ (i.e., the constraint vector is expanded up to order 2), the error equation quickly explodes due to the multiplication of series. However we can still write this fairly compactly as

$$\begin{aligned}-e^i =&\Psi^{i,\eta_1}\sum_{q=1}^{m} g^{\eta_1,\kappa_1\ldots\kappa_q}\delta u_0^{\kappa_1}\ldots\delta u_0^{\kappa_q} + \frac{1}{2}\Psi^{i,\eta_1\eta_2}\Bigg[\sum_{q_1=1}^{m}\sum_{q_2=1}^{m} g^{\eta_1,\kappa_1\ldots\kappa_{q_1}}g^{\eta_2,\alpha_1\ldots\alpha_{q_2}}\delta u_0^{\kappa_1}\ldots\delta u_0^{\kappa_{q_1}}\delta u_0^{\alpha_1}\ldots\delta u_0^{\alpha_{q_2}}\\ &+ 2\sum_{q=1}^{m} g^{\eta_1,\kappa_1\ldots\kappa_q}\delta u_0^{\kappa_1}\ldots\delta u_0^{\kappa_q}\sum_{p=1}^{m}\frac{1}{p!}\phi^{\eta_2,\gamma_1\ldots\gamma_p}_{(t_f,t_0)}\delta x_0^{\gamma_1}\ldots\delta x_0^{\gamma_p}\Bigg]\end{aligned} \tag{3.55}$$

An interesting point to note is that the $r = 2$ order contribution added terms in the control vector from 1 to $2m$. A decision can be made on whether to keep the terms of order $> m$ for the controls when the initial expansion in Eq. 3.4 only went to order m.

3.4.3 Solving for the control vector

At this point we have derived equations for an arbitrarily high expansion in terms of the control vector. In solving for the control vector, we will often look to minimize the norm of the post-control error from our desired constraint. For the linear expansion in the constraints where $r = 1$, this can be expressed as

$$e_f^i = -e^i - \Psi^{i,\eta_1} \sum_{q=1}^{m} g^{\eta_1,\kappa_1\ldots\kappa_q} \delta u_0^{\kappa_1} \ldots \delta u_0^{\kappa_q} \tag{3.56}$$

If the number of control variables is greater than the number of constraint variables, in general there is not a unique solution to what control vector will solve the guidance problem. Thus, some sort of numerical root-finding scheme must be used; usually we would want to null the final state error at time t_f, or minimize the total $\Delta\mathbf{v}$ or fuel consumption while reducing the constraint error to within a certain tolerance. For these cases an optimization method would be the desired approach, and since evaluating the STTs many times requires no further integrations of the dynamics, large-scale problems which are tedious to find solutions for numerically can become more tractable for potential on-board computation.

For the types of problems where a predictor-corrector or differential corrections method is used, we will often have an equal and sufficiently small number of control and constraint variables, such that the Γ matrix is invertible. In this case we can formulate an iterative Newton (predictor-corrector) type method for the controls update using the STTs. Taking the partial derivative of Eq. 3.56 with respect to the initial control vector $\delta\mathbf{u}_0$, we can formulate an equation for Γ as

$$\Gamma^{i,\kappa_1} = \frac{\partial e_f^i}{\partial u_0^{\kappa_1}} = -\Psi^{i,\eta_1} \left[g^{\eta_1,\kappa_1} + \sum_{q=2}^{m} q \cdot g^{\eta_1,\kappa_1\kappa_2\ldots\kappa_q} \delta u_0^{\kappa_2} \ldots \delta u_0^{\kappa_q} \right] \tag{3.57}$$

An iterative formula for the control update can then be constructed as

$$\delta\mathbf{u}_0^{n+1} = \delta\mathbf{u}_0^n - (\Gamma^n)^{-1}\mathbf{e}_f^n \tag{3.58}$$

where n represents the iteration number. This can be repeated using Eq. 3.56 to re-compute the estimated final error $\mathbf{e}_f$ for each guess of $\delta\mathbf{u}_0$ until $\|\mathbf{e}_f\|$ is nulled within the desired tolerance. An initial guess for $\delta\mathbf{u}_0$ for $m > 1$ can easily be obtained using the linear solution (i.e. when $m = 1$)[92]. For an unequal number of control and error parameters, the pseudo-inverse can be used to obtain Γ^{-1}, although this does not guarantee the existence of a unique solution.

A summary of how the equations derived in Sections 3.3 and 3.4 can be applied is given in Algorithm 2.

Algorithm 1 Compute impulsive correction maneuver at time t_0

Given: Reference trajectory
  Integrate and store reference trajectory STTs $\phi_{(t_f,t_0)}$ up to desired order m
  Compute $\beta_{(t_f,t_0)}$ tensors using Eqs. 3.31- 3.34
  Compute $\Xi_{(t_f,t_0)}$ tensors using Eqs. 3.35- 3.40

Given: State deviation $\delta\mathbf{x}_0$ from reference at time t_0
  Compute initial error vector $\mathbf{e}$ using Eq. 3.48
  Compute g tensors using Eq. 3.49

Given: Initial guess $\delta\mathbf{u}_0$
  Compute initial post-control error vector $\mathbf{e}_f$ using Eq. 3.56

*** Iterate to find optimal $\delta\mathbf{u}_0$ ***
while $\|\mathbf{e}_f\| > \epsilon_{\text{tol}}$ **do**
  Compute $\mathbf{\Gamma}$ using Eq. 3.57
  Compute updated guess for $\delta\mathbf{u}_0$ using Eq. 3.58
  Compute updated post-control error vector $\mathbf{e}_f$ using Eq. 3.56
end while

3.5 Automatic Differentiation

The integration of the state transition tensors (see Eqs. 2.15 - 2.18) necessitates knowledge of the A tensors, which represent the partial derivatives of the state rates with respect to the state. These tensors can be analytically derived and implemented by hand, though beyond the second order this will generally become prohibitively tedious. For simple dynamics, these can be derived using symbolic manipulators such as Mathematica, Matlab's Symbolic Toolbox, or SymPy. However, for complex dynamical systems with many perturbations, the resulting equations are often

not optimally formulated and must be repeatedly re-derived each time the dynamics equations are modified. For the intended use cases of an STT guidance scheme, the dynamics are highly nonlinear and we would ideally like to include the full perturbations that would be used in ground-based maneuver planning. Thus, we will want to employ some form of automatic differentiation.

A number of automatic differentiation methods have previously been applied in astrodynamics, including forward-mode automatic differentiation [98], differential algebra/Taylor polynomial differentiation [41], multicomplex numbers [70], and dual numbers [94]. Through a variety of numerical techniques, these methods each provide a framework to automatically compute the first and higher-order derivatives of a function. In general, these can be accurate to machine precision. Since it is particularly well-suited to computing higher-order partials, the Taylor polynomial differentiation method was used here, though any of the other methods could have been used, as the derivation of the STT guidance equations is the same regardless of how the A tensors are computed. Also note that this method was used strictly to compute the A tensors, which are then provided to the differential equations detailed in Eqs. 2.15 - 2.18. This differs from the typical implementation of the Taylor polynomial automatic differentiation method in astrodynamics (which is referred to as differential algebra (DA) [41]), where the DA variable is provided to an integrator (which is able to be overloaded with the DA variable type). Using STTs, only the dynamics equations require operator overloading for the automatic differentiation, and any integrator can be used to integrate the STTs. For the following examples, we made use of the freely-available PyAudi package developed by the European Space Agency [58] and the SciPy **solve_ ivp** integrator.

Using STTs, the computation of the dynamics partials is decoupled from the assembly of the STT differential equations (which, at higher orders, represents the bulk of the computational effort). This limits the number of operations that are performed on the overloaded-type variables, meaning that this formulation was found to achieve very good scaling in computational time as the dynamics become more complex. A simple comparison of two automatic differentiation methods was conducted to illustrate this observation. In this comparison, a spacecraft's state was propagated for 100 seconds in a low-Earth orbit. The state derivatives were also integrated up to the

Method	**2-body**	**2-body+** J_2	**2-body+** $J_2 + J_3$	**2-body+** $J_2 + J_3 + J_4$
Forward-mode auto-differentiation	1.00	2.46	9.62	13.47
Taylor polynomial auto-differentiation	1.00	1.00	1.01	1.03

Table 3.1: Normalized propagation time comparison of higher-order state propagation methods

fourth order using two different methods: STTs with the forward-mode chain-rule automatic differentiation (which is widely used in the machine learning and data science communities, but can be inefficient for higher-order derivatives), and STTs with the Taylor polynomial differentiation. This process was repeated while adding successive spherical harmonic perturbations (J_2 through J_4). The computational time for propagating the 1st through 4th-order derivatives is shown in Table 3.1, normalized relative to the 2-body propagation time for each method. For this scenario, the computational time for the Taylor polynomial differentiation method does not significantly increase with the additional perturbations; thus, this formulation is promising for operational situations where full-fidelity models of the dynamics are necessary to achieve the desired accuracy. The forward-mode differentiation method scales exponentially as additional perturbations are added and would likely become unreasonably slow for complex dynamical systems. These examples were run on an ASUS personal laptop with 16 GB RAM and an Intel©Core™i7-6500U CPU at 2.50 GHz. The forward-mode automatic differentiation was implemented in the Julia programming language [98], and the Taylor polynomial differentiation was implemented in Python [58].

It is important to note that, for many problems, DA and STTs will yield similar results (i.e. higher-order state derivative information). In general, any trajectory that is expanded to higher-order in terms of STTs can be similarly expanded using DA. However, the STT formulation is more explicit, which can lead to interesting observations about the nature of the guidance problem and the underlying dynamics. The STT framework thus provides a good basis from which to develop numerical targeting and optimization methods.

3.6 Application: Circular Restricted Three-Body Problem

In order to demonstrate a potential use-case for the complex equations derived in Sections 3.3 and 3.4, we study a simple application in a commonly used dynamical system, the circular restricted three-body problem (CR3BP). As the CR3BP is a chaotic system, it will often exhibit highly nonlinear dynamical behavior which can lead to convergence difficulties when using linearized predictor-corrector methods. We will show that the incorporation of higher-order terms can allow for rapid and accurate computation of the controls required to correct for an initial state error, in addition to improved convergence for longer-horizon targeting situations.

3.6.1 Scenario

A hypothetical unstable periodic halo orbit in the Earth-moon system is used as a test scenario for the different methods. The initial conditions for this orbit in the normalized CR3BP are shown in Table 3.2. These initial conditions are propagated forward for several revolutions and shown in Figs. 3.1 and 3.2; we can clearly see the periodic nature of this orbit.

Table 3.2: Unstable halo orbit scenario parameters

Parameter	Value
μ	0.0121505856
x_0	0.8249600133110965
y_0	0.0
z_0	0.0704
$\dot{x}_0$	0.0
$\dot{y}_0$	0.182764953514506
$\dot{z}_0$	0.0

This periodic orbit is highly unstable (with a stability index of $\nu = 584.1$) [50], which results in highly nonlinear dynamical behavior. Though the instability of this orbit would likely make it undesirable for long-term spacecraft operations, it presents an interesting test scenario where iterative linearized predictor-corrector methods can be insufficient for targeting impulsive maneuvers.

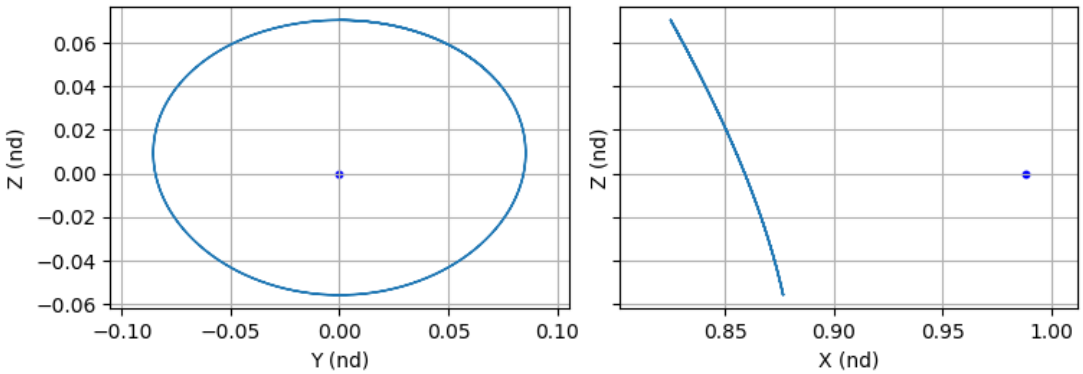

Figure 3.1: Unstable halo orbit initial conditions propagated forward for several revolutions, 2-dimensional view

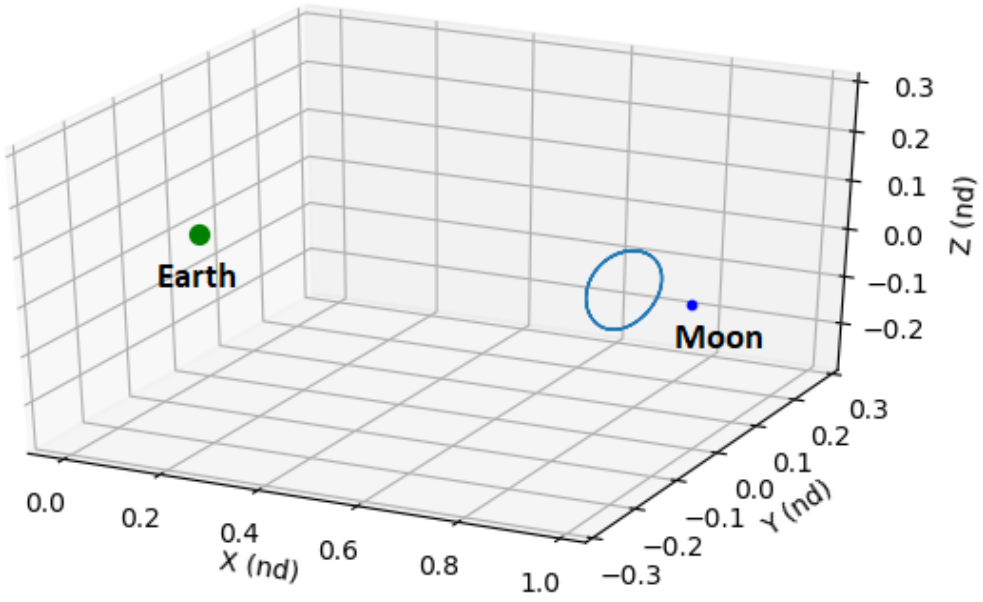

Figure 3.2: Unstable halo orbit initial conditions propagated forward for several revolutions, 3-dimensional view

In order to test out the different methods, an initial perturbation vector of

$$\delta\mathbf{x}_0 = \begin{bmatrix} 2.5\times10^{-5} & -2.5\times10^{-5} & 2.5\times10^{-5} & 1\times10^{-5} & 1\times10^{-5} & 1\times10^{-5} \end{bmatrix}^T \tag{3.59}$$

is applied to the state. These values roughly correspond in magnitude to 10 km for the position errors and 10 cm/s for the velocity errors, and line up with the preliminary expected 3-σ navigation errors for the Lunar Gateway mission while it is operating in cislunar space [79]. The unperturbed halo orbit is treated as the reference trajectory.

The impulsive controls $\delta\mathbf{u}_0$ (applied at time t_0) required to null the final position error vector with respect to the reference trajectory at a desired time t_f were then computed, first using the linearized predictor-corrector method, and next using the STT formulation. For both scenarios, the time t_f at which to null the position vector was varied up to $t_f = 5.5$ (corresponding to roughly 2 periods), in order to show situations where including higher-order terms results in improved convergence. Matching velocity is not considered in this scenario since the spacecraft could presumably perform another impulsive maneuver at time t_f to correct its velocity.

Note that the intent of this example is to demonstrate the STT method's capability for rapidly computing the controls to return to a reference position on the periodic orbit, similar to how the standard predictor-corrector method would function. In practice, halo orbit station-keeping strategies will often target maneuvers that seek to maintain the stability of the orbit rather than achieve a specified target position [103, 56].

3.6.2 Classical predictor-corrector

First, the scenario was run using the classical predictor-corrector algorithm detailed in Section 3.2. The process was iterated until convergence, which was set to be when the norm of the final position error vector is less than a tolerance of 10^{-12}. The choice of $\delta\mathbf{u}$ for computing the Γ matrix can affect the convergence and accuracy of the algorithm. For this scenario non-dimensional values of $\delta\mathbf{u} = 10^{-8}$ in all velocity directions were found to achieve the best results. An initial guess of all

zeros for the control parameters was used.

The predictor-corrector method was found to converge on good solutions for all t_f values up to $t_f = 2.9$, after which the method was no longer able to converge on a good solution (instead diverging dramatically with each successive iteration). The dynamics are thus sufficiently nonlinear after this time, such that the linearized corrections computed using the predictor-corrector method do not yield improved guesses for the controls, and the method is unable to converge on an impulsive maneuver to correct for even a relatively small perturbation. The time required to compute the controls for each time t_f for which convergence was achieved is shown in Fig. 3.7. All simulations were programmed in Python and run on an ASUS personal laptop with 16 GB RAM and an Intel©Core™i7-6500U CPU at 2.50 GHz.

3.6.3 State transition tensor method

The same scenario was then tested using the STT guidance method. In this method, we use Eq. 3.48 to compute the error vector at time t_f as a result of the initial perturbation. Our desired constraint is a fixed position at a fixed time - we are targeting the reference trajectory position at time t_f - thus, the constraint equations will only need to be expanded up to order $s = 1$. We will place equal weight on each direction of the position error. The Ψ constraint matrix will be a 3×6 matrix with

$$\Psi(1,1) = \Psi(2,2) = \Psi(3,3) = 1 \tag{3.60}$$

and all other terms equal to 0.

We can then use Eq. 3.56 to compute the post-control error vector $\mathbf{e}_f$, and use a root-finding scheme to minimize the norm of the error vector. For this scenario, we have an equal number of control and target states; we can therefore use the iterative root-finding method from Eq. 3.58 to compute the controls. The tolerance was set to 10^{-12}.

Once computed, the controls were applied and numerically integrated forward to test the accuracy of the approximations of the STT guidance method. The error between the corrected trajectory position and the target position at t_f is shown in Fig. 3.3 for values of t_f up to $t_f = 5.5$,

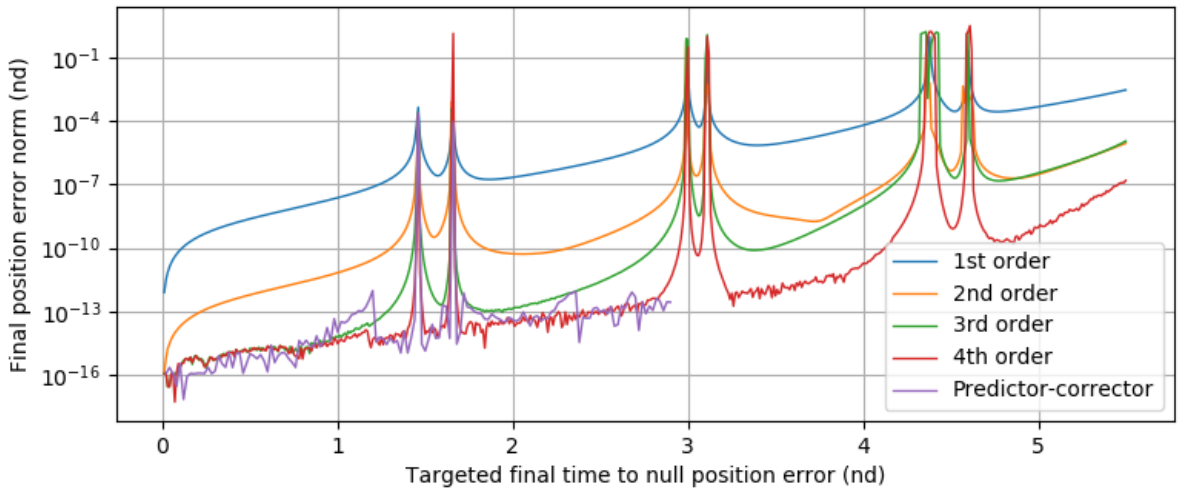

Figure 3.3: Final position error norm resulting from applying controls computed using STT method up to order $m = 4$

and for orders m up to $m = 4$. The STT method was found to converge on accurate solutions for all t_f values up to $t_f = 5.5$, except for when the impulsive maneuver required a significant plane change to reach the target (such as at $t_f = 1.45, 1.66, 3.00, 4.36$), which results from having a fixed time-of-flight. We can also see that including higher-order terms improves the accuracy of the computed controls, generally by several orders of magnitude for each additional order of STT included. The controlled orbit obtained from the controls targeting $t_f = 5.5$, using the STT method with $m = 4$, is shown along with the uncontrolled perturbed orbit in Figs. 3.4 and 3.5. The uncontrolled orbit clearly exhibits highly nonlinear behavior that diverges significantly from the reference orbit, which resulted in the convergence difficulties for the linearized predictor-corrector method. The STT method successfully converges on a maneuver that maintains the periodic nature of the orbit, despite the nonlinearities present in the trajectory.

The $\Delta\mathbf{v}$ magnitude computed for each t_f using the 4th-order STT method is shown in Fig. 3.6. The spikes in Fig. 3.3 can then be explained by the fact that the required $\Delta\mathbf{v}$ to null the position vector at these times is very large, which results in the corrected trajectory being far from the reference trajectory, and therefore outside the convergence region of the m-th order STTs.

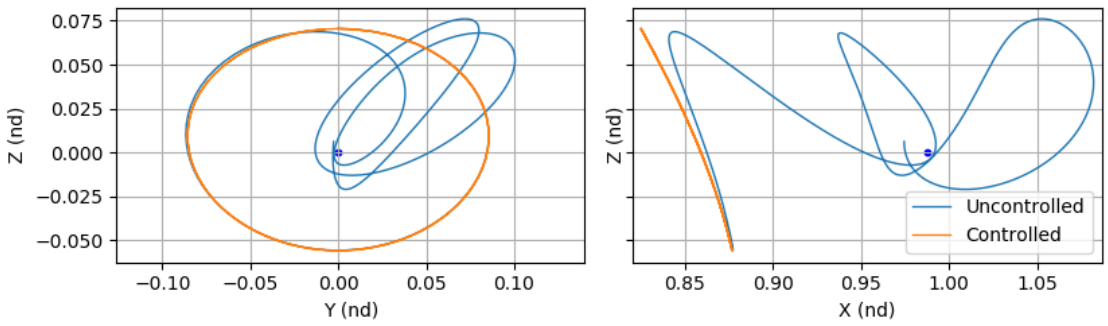

Figure 3.4: Uncontrolled and STT-controlled perturbed halo orbit, 2D view, $t_f = 5.5$

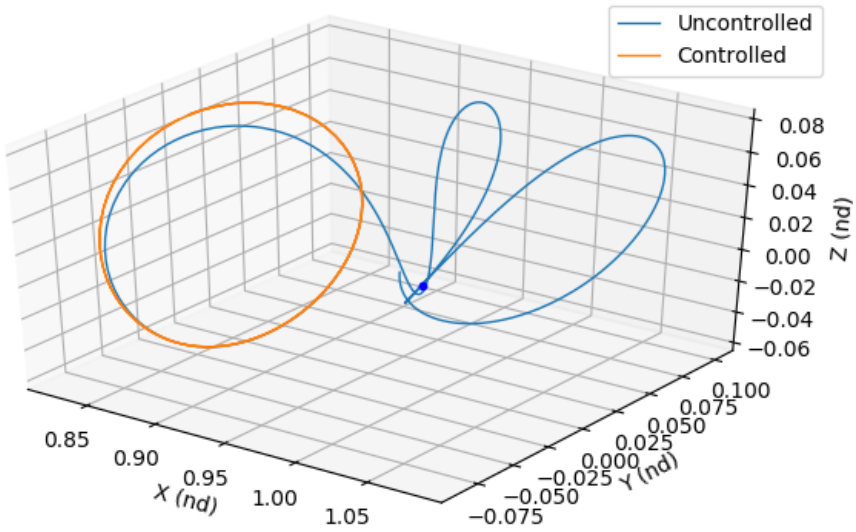

Figure 3.5: Uncontrolled and STT-controlled perturbed halo orbit, 3D view, $t_f = 5.5$

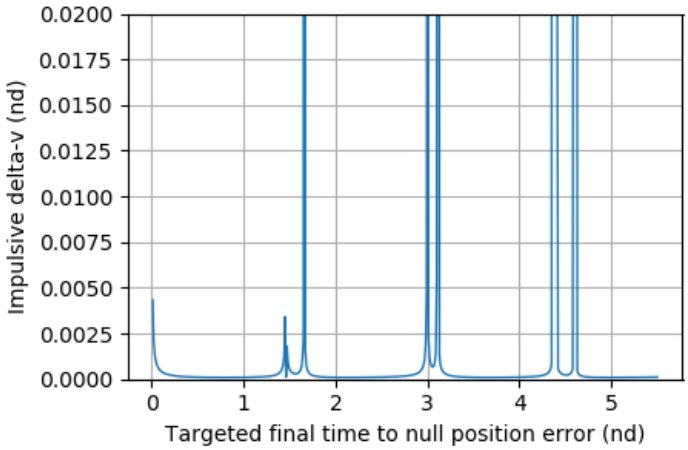

Figure 3.6: Computed $\Delta\mathbf{v}$ magnitude using 4th-order STT method

Because the maneuvers for these times would require very large $\Delta\mathbf{v}$'s relative to other times, these maneuvers would not be a desirable option.

3.6.4 Comparison of methods

The computational time required to compute the controls using the numerical predictor-corrector and the STT method (up to order $m = 4$) is shown for the range of target times t_f in Fig. 3.7. The times displayed for the STT method only include the time required to evaluate the STTs after they have been integrated, since the integration of the reference STTs would be performed prior to execution of the maneuver. Using the higher-order STT method, the computational time is reduced by several orders of magnitude, and the method is able to converge for t_f values past where the numerical predictor-corrector method diverges due to the highly nonlinear dynamics. In addition, when using the STT method, the computational time to evaluate the STTs for each iteration would not increase as the dynamics model becomes more complex, whereas it typically would increase when using numerical integration. It is therefore particularly well-suited to situations requiring rapid evaluations of complex dynamics.

A comparison of the convergence properties of the linearized predictor-corrector and the

Figure 3.7: Computational time required to compute impulsive correction maneuver, using the STT method at orders up to $m = 4$, and the predictor-corrector method. Note that the predictor-corrector method fails to converge after $t_f = 2.9$.

iterative STT method from Eq. 3.58 is shown in Figs. 3.8- 3.11 . This comparison is shown for two different target times: first, for $t_f = 1.5$, where the numerical and STT methods both converge, and second, for $t_f = 3.5$, where the predictor-corrector method does not converge on the solution. Note that the higher-order STT methods use the linear correction (i.e., the control obtained for the $m = 1$ case) for the first iteration. Figs. 3.8 and 3.10 show the norm of the predicted final position error vector (from Eq. 3.56 for the STT method) at each iteration, for the two different target times. Figs. 3.9 and 3.11 show the norm of the final position error vector when the control vector from each iteration is applied and numerically integrated. The differences between Figs. 3.8 and 3.9, and between Figs. 3.10 and 3.11, correspond to the approximation errors from using the STT method.

We can see from Fig. 3.8 that when the linearized predictor-corrector does converge, the number of iterations using either the higher-order STT method or the predictor-corrector is comparable. However, for $t_f = 3.5$ (Figs. 3.10 and 3.11), the predictor-corrector diverges while the STT method still converges rapidly. This illustrates two potential benefits of an STT-based maneuver computation scheme: improved convergence properties, and a decrease in computational requirements.

3.6.5 Monte Carlo analysis

In order to illustrate the robustness of the proposed STT method, a Monte Carlo analysis was performed for a range of initial perturbation vectors, for $t_f = 3.5$. The values for the initial perturbation vectors were sampled from a normal distribution with zero mean, and 1-σ standard deviations equal to

$$\boldsymbol{\sigma} = \begin{bmatrix} 2.5 \times 10^{-5} & 2.5 \times 10^{-5} & 2.5 \times 10^{-5} & 1 \times 10^{-5} & 1 \times 10^{-5} & 1 \times 10^{-5} \end{bmatrix}^T \tag{3.61}$$

As stated previously, these values correspond to the expected 3-σ navigation errors for the Lunar Gateway mission while it is operating in cislunar space[79] (or three times the expected 1-σ

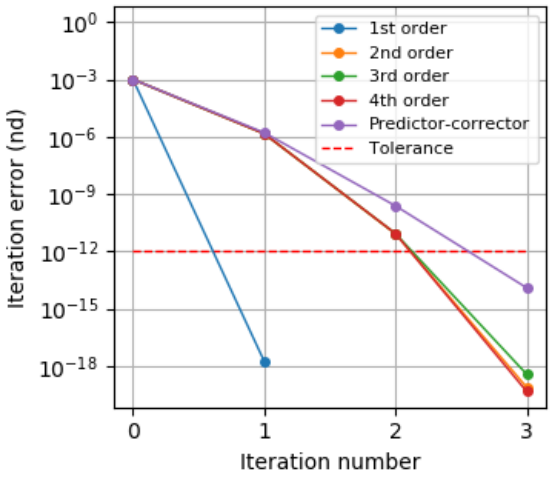

Figure 3.8: Evolution of iteration errors for $t_f = 1.5$. STT errors are computed using Eq. 3.56

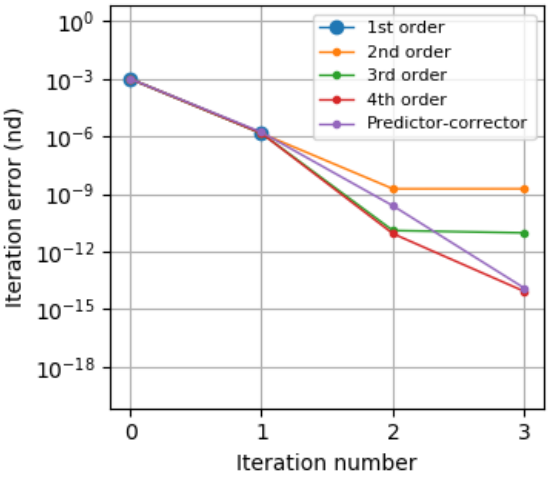

Figure 3.9: Evolution of iteration errors for $t_f = 1.5$ when control guess is applied and numerically integrated

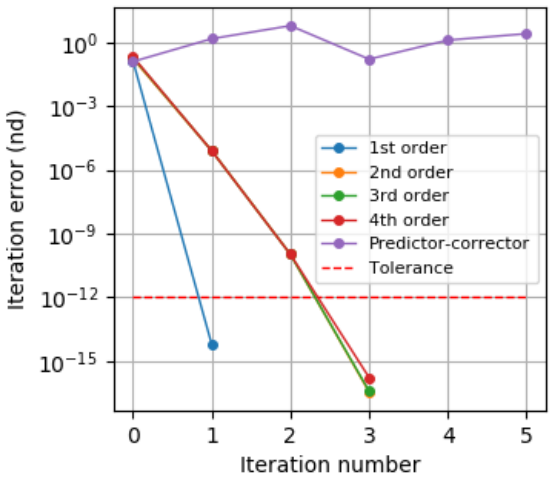

Figure 3.10: Evolution of iteration errors for $t_f = 3.5$. STT errors are computed using Eq. 3.56

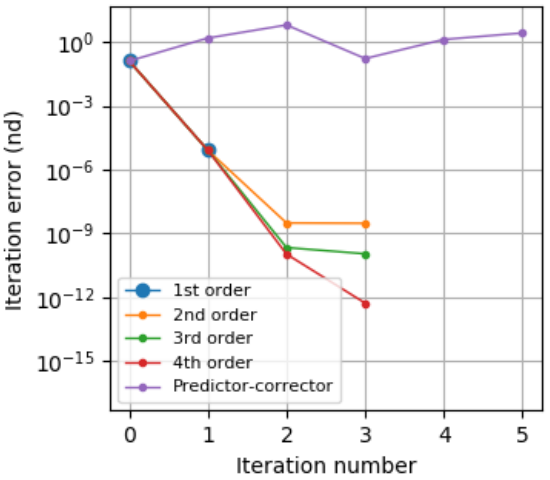

Figure 3.11: Evolution of iteration errors for $t_f = 3.5$ when control guess is applied and numerically integrated

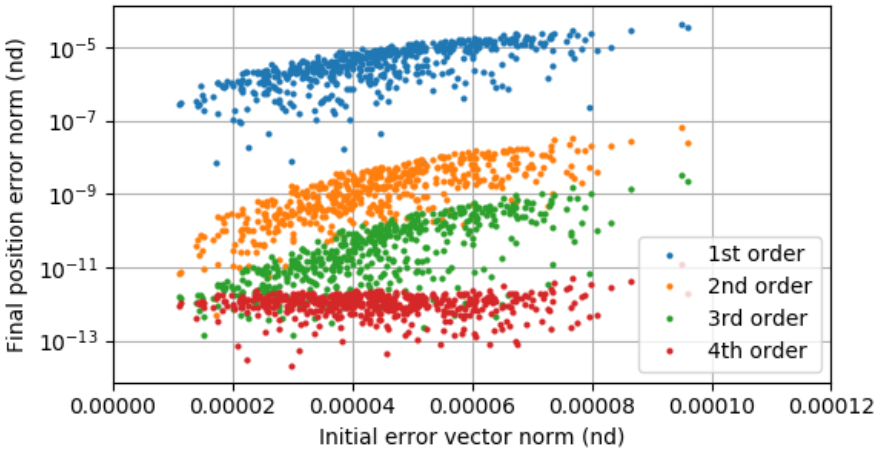

Figure 3.12: Final position errors for Monte Carlo simulation using STT method, plotted as a function of initial perturbation magnitude

navigation errors). 500 perturbation vectors were sampled from this distribution, and the STT method was run for each of these perturbations, up to order $m = 4$. The STT method successfully computed accurate impulsive maneuvers for all 500 cases. The final position error norm after applying the computed maneuvers and integrating to $t_f = 3.5$ is shown in Fig. 3.12, as a function of the norm of the perturbation error vectors. The uncontrolled and controlled orbits for all 500 cases (for $m = 4$) are plotted in Figs. 3.13 and 3.14.

3.6.6 Discussion

The examples provided here illustrate how the STT method can be used to rapidly compute the controls for an impulsive maneuver at a pre-specified initial time. In order to extend this to an on-board spacecraft guidance system, the STTs would need to be integrated and stored for a range of epoch times. These could then be rapidly evaluated at each discrete time to compute the controls at each time to reach the desired target. Alternatively, the STTs could be integrated and stored in stages divided over the entire transfer time, and subsequently multiplied together to construct the STTs for the desired time range (see Eqs. 2.19 - 2.22).

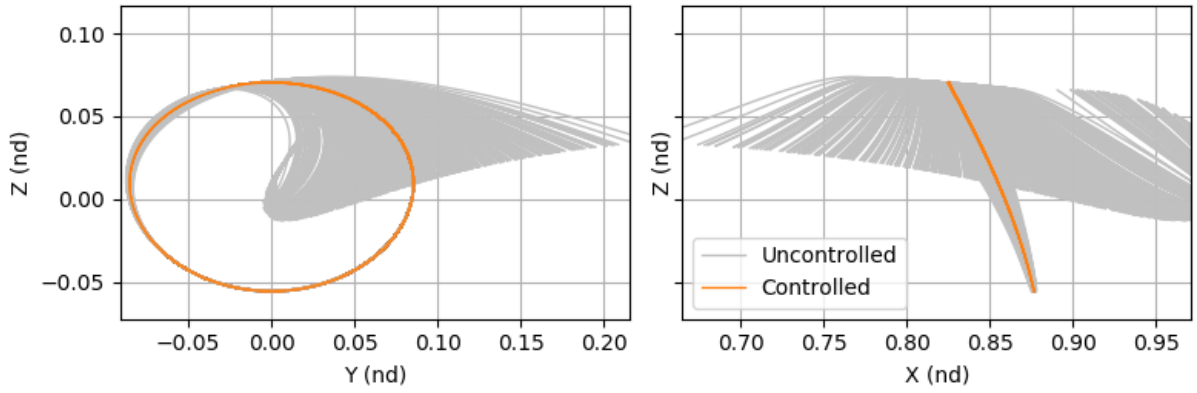

Figure 3.13: Uncontrolled and controlled perturbed halo orbits for Monte Carlo simulation, 2D view

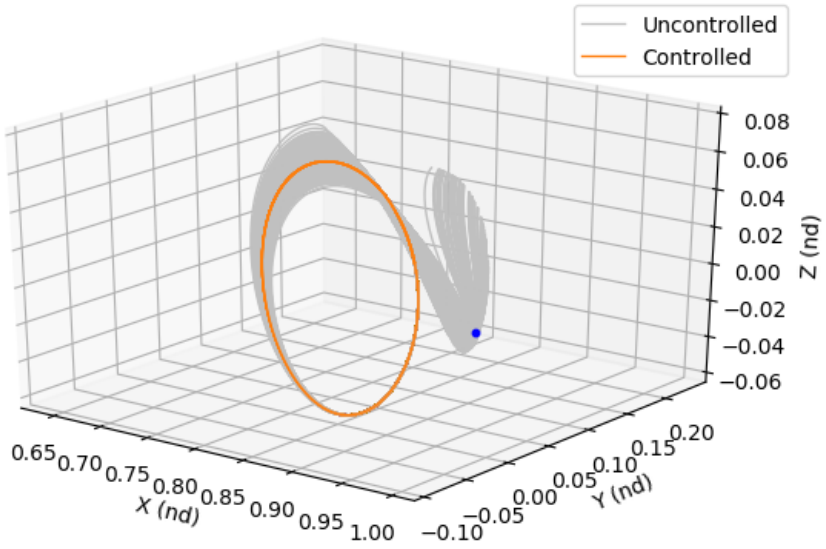

Figure 3.14: Uncontrolled and controlled perturbed halo orbits for Monte Carlo simulation, 3D view

3.7 Conclusions

In this chapter we derived the math necessary to use state transition tensors (STTs) for a spacecraft guidance scheme. Simplifications were made for the case where the target is the state at the end of the propagation time, and the controls are impulsive $\Delta\mathbf{v}$'s at the current epoch. An example is presented for computing an impulsive stationkeeping maneuver in an unstable halo orbit around the Earth-Moon L_1 point. The results show that the STT method can successfully compute accurate maneuver parameters, and requires significantly less time to compute these controls than a linearized numerical predictor-corrector. In addition, the STT method is shown to converge on good solutions for instances where the predictor-corrector method diverges. The STT method thus shows promise for situations where real-time or on-board numerical integration of the dynamics is impractical due to dynamic complexity or computational limitations. The subsequent chapters in this thesis will focus on improving the applicability and flexibility of this STT-based control scheme.

Chapter 4

Variable Time-of-Flight Maneuver Targeting using State Transition Tensors

4.1 Introduction

The higher-order STT formulation developed in Chapter 3 (and most higher-order formulations developed in previous works, e.g. Refs. [92] and [39]) can have limited use given that the STTs are integrated over a fixed period of time. In many operational settings, the target parameters for a maneuver may be to satisfy a constraint at a crossing of a specific plane or axis, rather than at a specific time. In this chapter, we show how the STT method can be expanded on to solve the problem of maneuver targeting to achieve specific geometric goals with a variable time-of-flight.

A method for including a variable integration time in a higher-order expansion, deemed state transition polynomial with time expansion (STP-T), has previously been derived by Sun et al. [106] using differential algebra (DA). The STP-T method was shown to provide a good approximation of the final state within valid ranges of initial state and final time values. However, while it is suitable for DA integration, the method used to obtain the time expansion is impractical for use with many implementations for computing STTs, which provides the motivation for the work in this chapter: to derive a method to expand the STTs of a reference trajectory with respect to the final time, and to apply these STTs to a variable time-of-flight spacecraft maneuver targeting problem.

This chapter is organized as follows. First, two methods for including the derivatives of the final state with respect to the final time in the STT expansion are derived and numerically validated. The STTs are subsequently used to compute halo orbit stationkeeping maneuvers with a variable target time in the Earth-Moon circular restricted three-body problem (CR3BP) - a highly

nonlinear dynamical system. This is a direct extension of the fixed time-of-flight methodology developed in Chapter 3. The algorithm is able to rapidly and analytically converge on accurate maneuvers in the vicinity of the reference. The results in this chapter demonstrate that allowing for a variable time-of-flight in the STT-based maneuver targeting scheme results in more optimal maneuvers and improved accuracy when compared to using the fixed time-of-flight algorithm from Chapter 3.

4.2 STTs with Time Expansion

Previous efforts using STTs for control applications[92, 20] have considered an integration time from t_0 to t_f with fixed values for t_0 and t_f. As stated in the introduction for this chapter, in this work we seek to expand the STTs with respect to the final time to allow for a variable t_f. In this section, two different methods are presented for computing STTs which include derivatives with respect to time. First, we show that we can obtain the time derivatives by simply evaluating the time and coupled time/state derivatives of the dynamics equations at the final time, and appending these to the state STTs (which are typically obtained through numerical integration). We next present a second method where a time scaling parameter is included in the equations of motion and the derivatives of the state vector are integrated with respect to this parameter, similar to how the state STTs are computed. Both methods are then numerically validated with a simple example to show that they produce equivalent and accurate results, before proceeding to a more complex application in Sec. 4.4.

4.2.1 Appending time derivatives

In the first method we will simply derive equations for the derivatives of the final state vector with respect to the final time t_f. If we consider a spacecraft state vector $\mathbf{x}$ that is expressed in Cartesian coordinates (i.e. $\mathbf{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$, where x, y, and z are the position components, and $\dot{x}$, $\dot{y}$, and $\dot{z}$ are the velocity components), then the parameter t_f can be appended

to the state vector $\mathbf{x}$ to form an augmented state vector $\mathbf{X}$, giving

$$\mathbf{X} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & t_f \end{bmatrix}^T \tag{4.1}$$

Thus, the augmented first-order state transition matrix Φ for the augmented state vector $\mathbf{X}$ becomes

$$\Phi_{(t_f,t_0)} = \begin{bmatrix} \phi_{(t_f,t_0)} & \frac{\partial \mathbf{x}_f}{\partial t_f} \\ 0 & 1 \end{bmatrix} \tag{4.2}$$

where $\phi_{(t_f,t_0)}$ is the standard STM for the six-component state vector $\mathbf{x}$ that is generally obtained through numerical integration. This augmentation requires knowledge of the $\frac{\partial \mathbf{x}_f}{\partial t_f}$ terms, which correspond to the state rates at time t_f (i.e. the velocity and acceleration components). In order to keep the notation concise for higher orders, we will write this out using the state vectors and index notation:

$$\Phi^{i,\tau} = \frac{\partial x_f^i}{\partial t_f} \tag{4.3}$$

where x_f^i refers to the i-th component of the state vector $\mathbf{x}_f$ at time t_f, and τ is the index of the time parameter t_f in the state vector (here, $\tau = 7$). Because τ corresponds to one index, in this notation it is not summed over as with the other superscripts. Using this notation, we can see that:

$$\Phi^{\tau,\tau} = 1 \tag{4.4}$$

and all other first and higher-order $\Phi^{\tau,\cdots}$ terms are equal to 0 (i.e., the partials of the final time with respect to the initial state vector are 0).

The higher-order STTs need to be augmented in a similar fashion. In what follows, the time indices will be left out of the STT terms, with the understanding that all follow the conventions $\Phi_{(t_f,t_0)}$ and $\phi_{(t_f,t_0)}$. In order to augment the second-order STT, the terms that must be appended to the standard STT are:

$$\Phi^{i,\tau a} = \Phi^{i,a\tau} = \frac{\partial^2 x_f^i}{\partial t_f \partial x_0^a} = A^{i,\alpha}\phi^{\alpha,a} \tag{4.5}$$

$$\Phi^{i,\tau\tau} = \frac{\partial^2 x_f^i}{\partial t_f^2} \tag{4.6}$$

where A corresponds to the state rate matrix evaluated at time t_f, the equations for which have presumably been formulated in order to integrate the STM. The $\frac{\partial^2 \mathbf{x}_f}{\partial t_f^2}$ term corresponds to the vector of acceleration and jerk terms at time t_f, the latter of which would need to be derived as they are not typically used in orbit propagation.

In order to extend this to third order, the additional terms to be computed are:

$$\Phi^{i,\tau ab} = \Phi^{i,a\tau b} = \Phi^{i,\tau ab} = \frac{\partial^3 x_f^i}{\partial t_f \partial x_0^a \partial x_0^b} = A^{i,\alpha}\phi^{\alpha,ab} + A^{i,\alpha\beta}\phi^{\alpha,a}\phi^{\beta,b} \tag{4.7}$$

$$\Phi^{i,\tau\tau a} = \Phi^{i,\tau a\tau} = \Phi^{i,a\tau\tau} = \frac{\partial^3 x_f^i}{\partial t_f^2 \partial x_0^a} = \dot{A}^{i,\alpha}\phi^{\alpha,a} \tag{4.8}$$

$$\Phi^{i,\tau\tau\tau} = \frac{\partial^3 x_f^i}{\partial t_f^3} \tag{4.9}$$

where $\dot{A}$ represents the time derivative of the corresponding A matrix/tensor, which would generally need to be derived analytically or symbolically. $\phi^{\alpha,ab}$ corresponds to the second-order STT for the standard six-component state vector $\mathbf{x}$, obtained through numerical integration. The $\frac{\partial^3 \mathbf{x}_f}{\partial t_f^3}$ term corresponds to the vector of jerk and snap (time derivative of jerk) terms evaluated at time t_f.

Finally, for the fourth-order STTs, the required terms are:

$$\begin{aligned}\Phi^{i,\tau abc} = \Phi^{i,a\tau bc} = \Phi^{i,ab\tau c} = \Phi^{i,abc\tau} &= \frac{\partial^4 x_f^i}{\partial t_f \partial x_0^a \partial x_0^b \partial x_0^c} \\ &= A^{i,\alpha}\phi^{\alpha,abc} + A^{i,\alpha\beta}\left(\phi^{\alpha,a}\phi^{\beta,bc} + \phi^{\alpha,ab}\phi^{\beta,c} + \phi^{\alpha,ac}\phi^{\beta,b}\right) + A^{i,\alpha\beta\gamma}\phi^{\alpha,a}\phi^{\beta,b}\phi^{\gamma,c}\end{aligned} \tag{4.10}$$

$$\Phi^{i,\tau\tau ab} = \Phi^{i,\tau a\tau b} = \Phi^{i,\tau ab\tau} = \Phi^{i,a\tau\tau b} = \Phi^{i,a\tau b\tau} = \Phi^{i,ab\tau\tau} = \frac{\partial^4 x_f^i}{\partial t_f^2 \partial x_0^a \partial x_0^b} = \dot{A}^{i,\alpha}\phi^{\alpha,ab} + \dot{A}^{i,\alpha\beta}\phi^{\alpha,a}\phi^{\beta,b} \tag{4.11}$$

$$\Phi^{i,\tau\tau\tau a} = \Phi^{i,\tau\tau a\tau} = \Phi^{i,\tau a\tau\tau} = \Phi^{i,a\tau\tau\tau} = \frac{\partial^4 x_f^i}{\partial t_f^3 \partial x_0^a} = \ddot{A}^{i,\alpha}\phi^{\alpha,a} \tag{4.12}$$

$$\Phi^{i,\tau\tau\tau\tau} = \frac{\partial^4 x_f^i}{\partial t_f^4} \tag{4.13}$$

where $\ddot{A}$ represents the second time derivative of the corresponding A matrix/tensor. Clearly, the number of terms that must be derived increases significantly with each additional order of STT. Though symbolic differentiation packages can be used to obtain equations for these terms, these can quickly become complex and cumbersome for higher orders. They also need to be re-derived any time the equations of motion are modified. However, if these can be formulated analytically, then this method does not require the integration of additional equations.

4.2.2 Time scaling

A second method for expanding the STTs with respect to time is to add a time scaling parameter γ to the equations of motion, and add this parameter to the state vector. The derivatives of the state with respect to γ can then be numerically integrated, in the same way that the state component STTs are integrated. We will define this time scaling parameter as

$$\gamma = \frac{t_f}{t_f^*} \tag{4.14}$$

where t_f is the time at which the STTs are to be evaluated, and t_f^* is the final time of the reference trajectory. As a result, under this definition, the time scaling parameter for the reference trajectory is $\gamma^* = 1$. With the time scaling parameter, the dynamics are now specified relative to a shifted timescale t', defined by

$$\Delta t' = \frac{1}{\gamma}\Delta t \tag{4.15}$$

Any derivatives with respect to t will be replaced by derivatives with respect to t' through the relation

$$\frac{\mathrm{d}}{\mathrm{d}t} = \frac{1}{\gamma}\frac{\mathrm{d}}{\mathrm{d}t'} \tag{4.16}$$

The derivatives of the equations of motion with respect to γ are straightforward to derive - generally these will be far easier to compute than the terms required for the first method. These can be obtained analytically, by using a symbolic differentiation package, or through an automatic differentiation scheme such as the one implemented in Ref. [20], which uses Taylor polynomial

automatic differentiation [58]. The time scaling parameter γ can then be included in the state vector to produce an alternative augmented state vector $\mathbf{X}$:

$$\mathbf{X} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \gamma \end{bmatrix}^T \tag{4.17}$$

The scaled differential equations are integrated with respect to the scaled time t'. The velocity components of the state vector are therefore shifted by $\dot{x} \to x'$, $\dot{y} \to y'$, $\dot{z} \to z'$. For clarity we will define the augmented state vector in the t' space as $\mathbf{Y}$:

$$\mathbf{Y} = \begin{bmatrix} x & y & z & x' & y' & z' & \gamma \end{bmatrix}^T = \begin{bmatrix} x & y & z & \gamma\dot{x} & \gamma\dot{y} & \gamma\dot{z} & \gamma \end{bmatrix}^T \tag{4.18}$$

In order to construct the STM/STTs mapping the unscaled state deviation $\delta\mathbf{X}_0$ to $\delta\mathbf{X}_f$, while fully capturing the effects of a change in the time scaling parameter $\delta\gamma$, the derivatives will need to be multiplied together as required from the chain rule. We will introduce some further notation here in order to write this out concisely. We will define the operator $\circ$ to signify the multiplication of two STT expansions, resulting in a single STT expansion. The notation $\frac{\partial^p X_f^i}{\partial X_0^{\kappa_1} ... \partial X_0^{\kappa_p}}$ is used to represent the series of STTs mapping from $\mathbf{X}_0$ to $\mathbf{X}_f$. The STTs obtained from numerically integrating in t' therefore correspond to $\frac{\partial^p Y_f^i}{\partial Y_0^{\kappa_1} ... \partial Y_0^{\kappa_p}}$. We can then express the full mapping from $\mathbf{X}_0$ to $\mathbf{X}_f$ as

$$\Phi^{i,\kappa_1 ... \kappa_p} = \frac{\partial^p X_f^i}{\partial X_0^{\kappa_1} ... \partial X_0^{\kappa_p}} = \left[\frac{\partial^p X_f^i}{\partial Y_f^{\kappa_1} ... \partial Y_f^{\kappa_p}}\right] \circ \left[\frac{\partial^p Y_f^i}{\partial Y_0^{\kappa_1} ... \partial Y_0^{\kappa_p}}\right] \circ \left[\frac{\partial^p Y_0^i}{\partial X_0^{\kappa_1} ... \partial X_0^{\kappa_p}}\right] \tag{4.19}$$

For first-order expansions, this expression reduces to the familiar STM multiplication equations. For higher orders, the STT multiplication equations must be used to chain these derivatives together; these equations are listed in Appendix B.

The next step is to explicitly derive the equations for $\frac{\partial^p Y_0^i}{\partial X_0^{\kappa_1} ... \partial X_0^{\kappa_p}}$ and $\frac{\partial^p X_f^i}{\partial Y_f^{\kappa_1} ... \partial Y_f^{\kappa_p}}$, i.e. the mappings between the scaled and unscaled dynamics at times t_0 and t_f. We present the equations for these mappings up to the fourth order, which matches the previous literature. Beginning with

$\frac{\partial^p Y_0^i}{\partial X_0^{\kappa_1} ... \partial X_0^{\kappa_p}}$, for the first-order case ($q = 1$), we can write this out explicitly in matrix form:

$$\frac{\partial \mathbf{Y}_0}{\partial \mathbf{X}_0} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \gamma & 0 & 0 & \dot{x}_0 \\ 0 & 0 & 0 & 0 & \gamma & 0 & \dot{y}_0 \\ 0 & 0 & 0 & 0 & 0 & \gamma & \dot{z}_0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.20}$$

For the second-order case, we obtain

$$\frac{\partial^2 Y_0^i}{\partial X_0^{\kappa_1} \partial X_0^{\kappa_2}} = 1 \tag{4.21}$$

for $i = \kappa_1 = [4, 5, 6]$ and $\kappa_2 = 7$, and vice-versa. All other terms in the second-order mapping are zero. Subsequently, all third and higher order mappings are zero, i.e.

$$\frac{\partial^3 Y_0^i}{\partial X_0^{\kappa_1} \partial X_0^{\kappa_2} \partial X_0^{\kappa_3}} = 0 \tag{4.22}$$

$$\frac{\partial^4 Y_0^i}{\partial X_0^{\kappa_1} \partial X_0^{\kappa_2} \partial X_0^{\kappa_3} \partial X_0^{\kappa_4}} = 0 \tag{4.23}$$

for all i and $\kappa_1 ... \kappa_4$. Note that for the typical case, we would be numerically integrating along the reference trajectory, giving $\gamma = \gamma^\star = 1$ for these terms.

We will also need to formulate the derivative matrix/tensors for the mapping at the final time, $\frac{\partial^p X_f^i}{\partial Y_f^{\kappa_1} ... \partial Y_f^{\kappa_p}}$. We can start from

$$\mathbf{X}_f = \begin{bmatrix} x_f & y_f & z_f & \frac{x'_f}{\gamma} & \frac{y'_f}{\gamma} & \frac{z'_f}{\gamma} & \gamma \end{bmatrix}^T \tag{4.24}$$

The first-order mapping becomes

$$\frac{\partial \mathbf{X}_f}{\partial \mathbf{Y}_f} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\gamma} & 0 & 0 & -\frac{x'_f}{\gamma^2} \\ 0 & 0 & 0 & 0 & \frac{1}{\gamma} & 0 & -\frac{y'_f}{\gamma^2} \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\gamma} & -\frac{z'_f}{\gamma^2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.25}$$

For the second-order mapping,

$$\frac{\partial^2 X_f^i}{\partial Y_f^{\kappa_1} \partial Y_f^{\kappa_2}} = \frac{2Y_f^i}{\gamma^3} \tag{4.26}$$

for $i = [4, 5, 6]$ and $\kappa_1 = \kappa_2 = 7$, and

$$\frac{\partial^2 X_f^i}{\partial Y_f^{\kappa_1} \partial Y_f^{\kappa_2}} = -\frac{1}{\gamma^2} \tag{4.27}$$

for $i = \kappa_1 = [4, 5, 6]$ and $\kappa_2 = 7$ and vice-versa. For the third-order mapping,

$$\frac{\partial^3 X_f^i}{\partial Y_f^{\kappa_1} \partial Y_f^{\kappa_2} \partial Y_f^{\kappa_3}} = -\frac{6Y_f^i}{\gamma^4} \tag{4.28}$$

for $i = [4, 5, 6]$ and $\kappa_1 = \kappa_2 = \kappa_3 = 7$, and

$$\frac{\partial^3 X_f^i}{\partial Y_f^{\kappa_1} \partial Y_f^{\kappa_2} \partial Y_f^{\kappa_3}} = \frac{2}{\gamma^3} \tag{4.29}$$

for $i = \kappa_1 = [4, 5, 6]$ and $\kappa_2 = \kappa_3 = 7$ and vice-versa (i.e. $i = \kappa_2 = [4, 5, 6]$ and $\kappa_1 = \kappa_3 = 7$, etc.). For the fourth-order mapping,

$$\frac{\partial^4 X_f^i}{\partial Y_f^{\kappa_1} \partial Y_f^{\kappa_2} \partial Y_f^{\kappa_3} \partial Y_f^{\kappa_4}} = \frac{24Y_f^i}{\gamma^5} \tag{4.30}$$

for $i = [4, 5, 6]$ and $\kappa_1 = \kappa_2 = \kappa_3 = \kappa_4 = 7$, and

$$\frac{\partial^4 X_f^i}{\partial Y_f^{\kappa_1} \partial Y_f^{\kappa_2} \partial Y_f^{\kappa_3} \partial Y_f^{\kappa_4}} = -\frac{6}{\gamma^4} \tag{4.31}$$

for $i = \kappa_1 = [4, 5, 6]$ and $\kappa_2 = \kappa_3 = \kappa_4 = 7$ and vice-versa. Again, for the typical case, we would have $\gamma = \gamma^\star = 1$, which simplifies the above equations.

Thus, using Eqs. 4.20-4.23 and 4.25-4.31, the STTs corresponding to $\Phi^{i,\kappa_1\ldots\kappa_p} = \frac{\partial^p X_f^i}{\partial X_0^{\kappa_1}\ldots\partial X_0^{\kappa_p}}$ can be formulated using Eq. 4.19. Note that the equations for these mappings do not depend on the dynamics equations.

4.2.3 Numerical validation

Both methods were tested to show that they can produce accurate approximations of the final state around a reference trajectory when a deviation is applied to the final time, and to show that they both produce equivalent results. A scenario was considered using the circular restricted three-body problem (CR3BP), a highly nonlinear dynamical system. The unscaled equations of motion for the CR3BP are given in Eqs. 2.7 - 2.9.

For the first method, the equations for the derivatives of the final state vector with respect to the final time were obtained using a symbolic differentiation package. The equations of motion for the CR3BP with the time scaling parameter (required for the time scaling method) are:

$$x'' = 2\gamma y' + \gamma^2 \left[x - \frac{(1-\mu)(x+\mu)}{r_1^3} - \frac{\mu(-1+x+\mu)}{r_2^3} \right] \tag{4.32}$$

$$y'' = -2\gamma x' + \gamma^2 \left[y - \frac{y(1-\mu)}{r_1^3} - \frac{\mu y}{r_2^3} \right] \tag{4.33}$$

$$z'' = \gamma^2 \left[-\frac{z(1-\mu)}{r_1^3} - \frac{\mu z}{r_2^3} \right] \tag{4.34}$$

Thus, for the reference trajectory (where $t_f = t_f^*$), we have $\gamma = \gamma^* = 1$, and we recover the standard, unscaled CR3BP equations.

Both methods were tested for an unstable halo orbit scenario around the L_1 Lagrange point in the Earth-Moon system. This is the same reference orbit that was used to generated the results in Chapter 3. The scenario parameters and initial conditions are given in Table 4.1. The period for this halo orbit is $T \simeq 2.7707$. These initial conditions were propagated forward for several revolutions and shown in Fig. 3.2; we can clearly see the periodic nature of this orbit. In this example and throughout this chapter, we use the normalized non-dimensional CR3BP units (also

Table 4.1: Earth-Moon halo orbit scenario parameters

Parameter	Value
μ	0.0121505856
x_0	0.8249600133098401
y_0	0.0
z_0	0.0704
$\dot{x}_0$	0.0
$\dot{y}_0$	0.1827649535351789
$\dot{z}_0$	0.0
T	2.77073806332875

referred to as *nd*). The time-expanded STTs were computed (up to order $m = 4$) using both the appending and time scaling methods, with a reference final time of $t_f^* = T$. The STTs were then evaluated for a range of final times $t_f^* + \delta t_f$, for $\delta t_f \in [-0.1, 0.1]$. The resulting states were compared to the true final state for each final time, computed through numerical integration. The norm of the state error vector was computed for each final time and order of STT. These are plotted in Fig. 4.1. It is clear that both methods produce the same results for all orders of STT. We can also see that, as expected, the accuracy of the STT approximation decreases as δt_f becomes larger, and increases as the maximum order of STT included in the approximation increases.

4.2.4 Discussion

The ease of implementing each method will vary depending on the formulation of the dynamics equations. The first method is relatively straightforward to implement up to second order, but requires symbolic derivation of the higher-order terms, and would require re-deriving these equations any time the dynamics are modified, for example by increasing the fidelity of the system with more perturbations. On the other hand, the time scaling method may be more straightforward to implement, particularly if using an automatic differentiation scheme to integrate the higher-order STTs, but involves the integration of additional equations for each new term. Because it is easily incorporated with automatic differentiation, the time scaling method was used to produce all

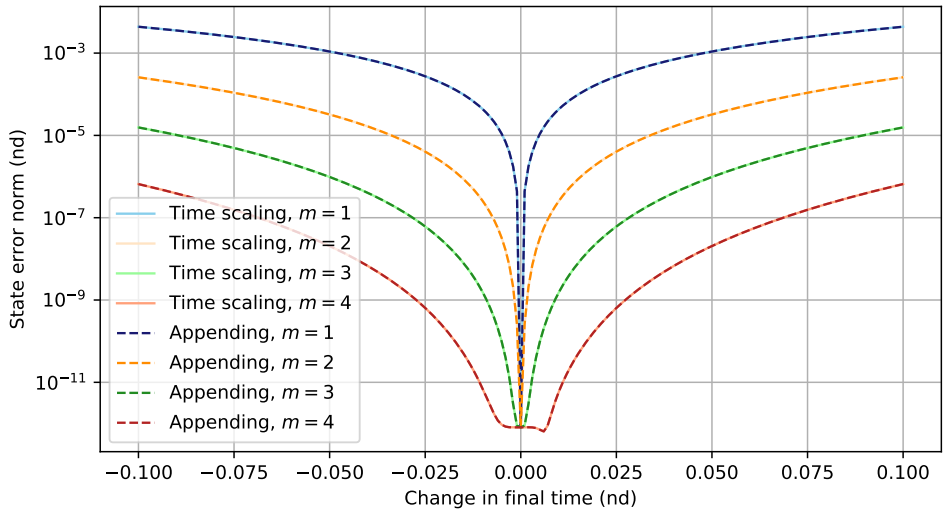

Figure 4.1: Numerical accuracy of STT time expansion methods for halo orbit scenario

subsequent examples in this chapter.

Note that both methods assume the dynamical system is autonomous - i.e., the dynamics do not explicitly depend on time. In practice, this is not always the case. For example, full-fidelity planetary ephemerides, solar flux values, and drag coefficients will contain terms that explicitly depend on time. In these situations, these parameters will need to be formulated or parametrized in a way that allows for them to be differentiable with respect to time. For the simple demonstrations in this chapter, we will only consider autonomous dynamical systems, so this is not needed.

4.3 Optimization Scheme

In Chapter 3, impulsive maneuvers to correct for an initial state deviation were computed using STTs with a fixed final time, targeting only the position components of the reference at this time. Because there are three control parameters and three target parameters in this example, there exists a unique solution, and this solution can be computed through an efficient iterative procedure that is equivalent to the STT version of Newton's root-finding method. However, with the addition of a time dependency parameter, there are now four parameters which can be varied for a single impulsive maneuver (3 control components, and one time parameter). If only three or fewer state components are being targeted, there will be an infinite number of solutions to the problem, and the iterative procedure to find the unique root can no longer be used. In order to address this issue, we developed an optimization scheme based on the principles of sequential quadratic programming (SQP), that can be used to analytically find the solution that minimizes the control magnitude while satisfying trajectory constraints, using only the reference trajectory STTs.

SQP is a direct optimization method that consists of performing successive second-order Taylor expansions of a cost function around a reference, and finding the linear update that minimizes the quadratic expansion [80]. This process is continued sequentially until convergence is achieved, usually defined by having converged on a local minimum for the cost function. SQP requires knowledge of the first and second-order derivatives of the cost function and constraints with respect to the control parameters. When using STTs, we can take advantage of the fact that the first and

second-order derivatives of an STT approximation of $\delta\mathbf{x}_f$ (Eq. 2.14) with respect to the initial state vector, evaluated at some point $\mathbf{x}_0 + \delta\mathbf{x}_0$, can be expressed exactly in terms of the STTs using Eqs. 2.23 and 2.25. Thus, if we consider the scenario where we seek to minimize an impulsive control applied at time t_0, while achieving some geometric constraint at time t_f, then the objective function and its first and second-order derivatives can be expressed analytically as a function of the reference trajectory STTs. The resulting optimization scheme is therefore computationally lightweight since it requires no further integrations of the dynamics. The full derivation and details of this procedure can be found in Ref. [21]. This optimization scheme was used to produce the examples in this chapter.

4.4 Application: Halo Orbit Stationkeeping

In this section, the STTs with time expansion are applied to a stationkeeping problem for a spacecraft operating in an unstable halo orbit around the L_1 Lagrange point in the Earth-Moon system. The unstable halo orbit was chosen to demonstrate the proposed methodology due to its pronounced chaotic behavior when compared with typical operational or proposed halo orbit regimes, such as the near-rectilinear halo orbit [115]. The initial conditions and scenario parameters for the halo orbit are given in Table 4.1 and the orbit is illustrated in Fig. 3.2; these are the same parameters that were used for the example in Sec. 4.2.3.

The high-level objective of performing stationkeeping for a halo orbit mission is generally to maintain stability of the dynamics, as any large deviation can lead to chaotic behavior and dramatic departure from the desired periodic (or quasi-periodic) motion. Operational stationkeeping strategies[51, 26] in halo orbits generally do not consider a fixed time-of-flight for the target parameters - this would result in overconstrained and suboptimal maneuvers, since any deviation in the initial state will result in a small shift in the period of the motion. This scenario therefore provides an excellent example to illustrate the benefits of expanding the STTs with respect to time.

4.4.1 Targeting a reference position vector

We first consider the scenario investigated in Section 3.6.1, where a perturbation was applied to the spacecraft's state at the epoch time t_0, and an impulsive control $\mathbf{u}_0$ (applied at time t_0) was computed using the reference STTs to null the final position error vector with respect to the reference trajectory at a desired time t_f. The time t_f^* at which to null the position vector was varied up to $t_f^* = 5.5$ (corresponding to roughly 2 periods). Matching velocity was not considered in this scenario since the spacecraft could presumably perform another impulsive maneuver at time t_f to correct its velocity. The results from Chapter 3 showed that an STT-based targeting method (with a fixed time-of-flight) could compute accurate maneuvers in most situations; however, there were certain values for t_f^* where the computed maneuver magnitudes were very large, and not accurately approximated with the STTs. For these cases, it was found that a significant plane change was required to return to the reference position at exactly the fixed target time. By using the variable time-of-flight method, this restriction can be removed and a more optimal and realistic maneuver can be obtained.

In order to test out the STT time expansion algorithm, the STTs of the reference trajectory (see Table 4.1) were integrated and stored for a range of reference t_f^* values using the time scaling method described in Sec. 4.2.2. A perturbation $\delta\mathbf{x}_0$ was then applied at time t_0:

$$\delta\mathbf{x}_0 = \begin{bmatrix} 2.5\times10^{-5} & -2.5\times10^{-5} & 2.5\times10^{-5} & 1\times10^{-5} & 1\times10^{-5} & 1\times10^{-5} \end{bmatrix}^T \tag{4.35}$$

The optimization scheme developed in Ref. [21] was then used to compute the controls required to return to the reference's position, for order $m = 3$, and for reference t_f^* values of $t_f^* \leq 5.5$. The algorithm was successfully able to converge on maneuvers for all cases. Note that, using the time expansion method, the controls will ensure that the spacecraft returns to the reference position at a shifted time $t_f^* + \delta t_f$. If using the time scaling method to obtain the STT time derivatives, this shift in the final time δt_f can be computed using the optimal value for the change in time scaling

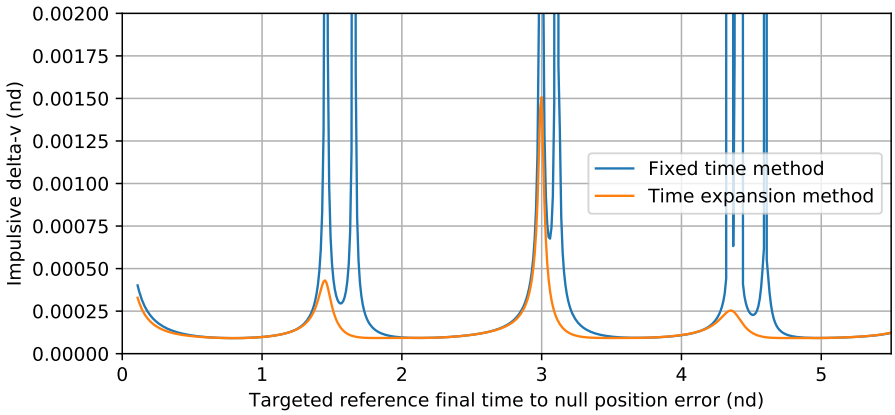

Figure 4.2: Control magnitudes computed using standard STT and STT time expansion methods, for $m = 3$

($\delta\gamma$) that was found through the optimization scheme:

$$\delta t_f = t_f^* \cdot \delta\gamma \tag{4.36}$$

The magnitude of the computed control vectors for both the standard STT method [20] and the STT time expansion method are shown in Fig. 4.2. For all figures in this Section, quantities are expressed in normalized non-dimensional CR3BP units (also referred to as nd). The computed controls were applied and integrated to time t_f^* (for the standard STT method) and to time $t_f^* + \delta t_f$ (for the STT time expansion method). The errors in the final position vectors with respect to the target position were computed; these are shown in Fig. 4.3. It can clearly be seen that the large spikes in control magnitude and final position error with the fixed time-of-flight method t (such as at $t_f^* = 1.45, 1.66, 3.00, 4.36$) are greatly reduced when the time-of-flight restriction is removed. This allows for more optimal maneuvers that do not incur a large plane shift, thus resulting in both decreased control requirements and increased accuracy of the STT approximation. The shift in the computed final time is shown for each reference value of t_f^* that was targeted in Fig. 4.4. We can see that larger shifts in the final time generally correspond to the target times where the control magnitude was very large with the fixed time-of-flight.

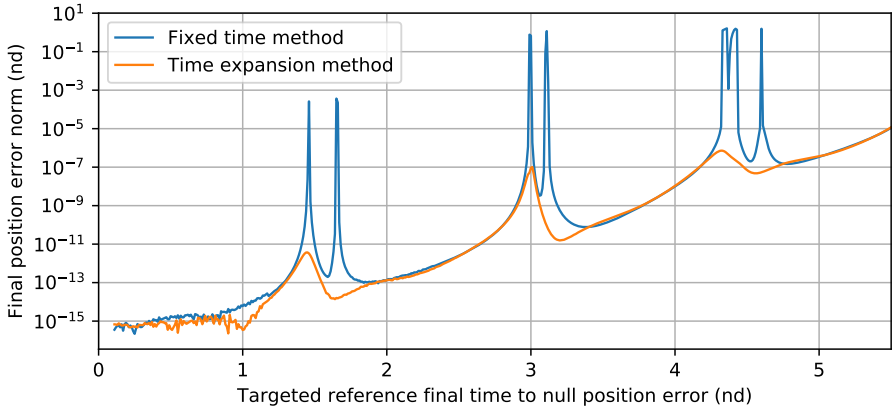

Figure 4.3: Final position error magnitudes for controls computed using standard STT and STT time expansion methods, for $m = 3$

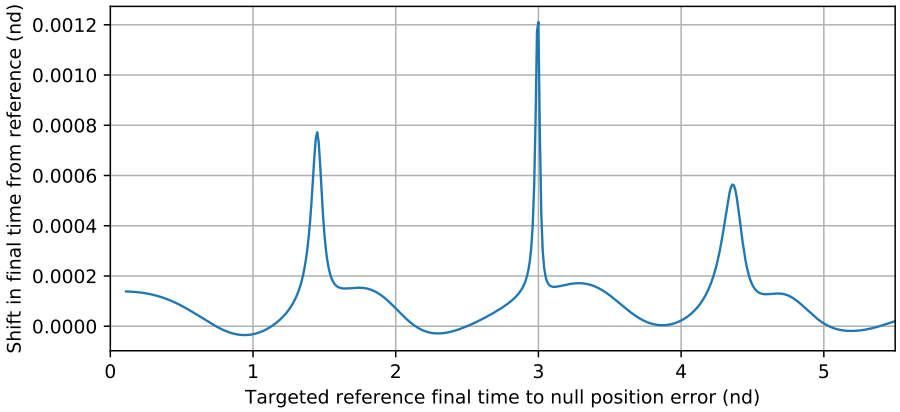

Figure 4.4: Optimal shift in final target time vs. reference final target time

4.4.2 X-axis crossing algorithm

The previous scenario involved computing a maneuver to return to a specified target position, In practice, halo orbit stationkeeping strategies have been developed that seek to maintain the stability of the halo orbit rather than closely track a specific reference trajectory. This has been found to reduce the stationkeeping fuel requirements while ensuring the spacecraft remains in the desired orbital regime.

Therefore we consider the scenario where a perturbation $\delta\mathbf{x}_0$ is applied to the spacecraft's state, and the spacecraft must execute an impulsive maneuver $\mathbf{u}_0$ to maintain the stability of the halo orbit over one period, rather than target a specific position state (see Fig. 4.5 for an illustration). From Guzzetti et al.[51] and Davis et al. [35] a stationkeeping strategy was developed for spacecraft operating in an Earth-Moon halo orbit. In this strategy, only the x velocity of the reference trajectory at the $x-z$ plane crossing is targeted. This was found to be sufficient to efficiently maintain the long-term stability of the halo orbit. The spacecraft's state is propagated until the $x-z$ plane crossing is reached - the timing of this crossing will therefore vary slightly if a perturbation is applied to the initial state. We will show that by allowing a variable time-of-flight through the STT time expansion method, improved stationkeeping maneuvers can be recovered with no initial guess.

The time-expanded STTs of the reference trajectory were integrated over one revolution and stored. The optimization scheme (at orders $m = 2$ through $m = 4$) was then run for two cases: first, with no time scaling parameter included in the control vector, and therefore a fixed time-of-flight (i.e., $\mathbf{u} = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^T$); and second, with the time scaling parameter in the control vector and a variable time-of-flight (i.e., $\mathbf{u} = \begin{bmatrix} u_x & u_y & u_z & \delta\gamma \end{bmatrix}^T$). This allows for an evaluation of the improvements in optimality and accuracy when using the time expansion method. Both cases were run for the same 100 initial state deviation vectors $\delta\mathbf{x}_0$, sampled from a zero-mean normal distribution with the following 1-σ standard deviations

$$\sigma = \begin{bmatrix} 2.5\times10^{-5} & 2.5\times10^{-5} & 2.5\times10^{-5} & 1\times10^{-5} & 1\times10^{-5} & 1\times10^{-5} \end{bmatrix}^T \tag{4.37}$$

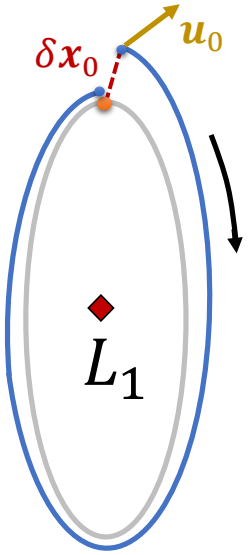

Figure 4.5: Diagram of halo orbit stationkeeping scenario (not to scale)

These roughly correspond to the expected 3-σ navigation errors for a spacecraft operating in cislunar space [79]. For all cases, an initial guess of zero for all control directions and the time scaling parameter $\delta\gamma$ was used. For each initial state deviation vector, the computed maneuvers were applied and numerically integrated in the full dynamics in order to validate the numerical accuracy of the algorithms. The results for both cases are shown in Table 4.3. The uncontrolled and controlled orbits for the $m = 2$ case are shown in Fig. 4.6. Again, for all tables and figures in this Section, quantities are expressed in normalized non-dimensional CR3BP units (also referred to as nd).

The inclusion of a time parameter and variable time-of-flight in the targeting scheme results in a significant decrease in the control requirement to target the desired states: the average control magnitude is reduced by 30%. Including a variable time-of-flight also improves the accuracy of the computed control. This may be somewhat paradoxical as we are including an additional parameter which can be varied. However, this can be explained by the fact that the optimal maneuver with the variable time-of-flight has a smaller magnitude and results in a trajectory that remains "closer" to the reference (and is thus better approximated using the reference STTs) - the average position deviation from the reference over the entire orbit is 40% lower for the variable time-of-flight method. Including higher orders of STTs improves the accuracy of the computed control, but also increases the computational time for the optimization scheme. There is therefore a tradeoff between accuracy, convergence, and runtime that must be taken into account when using this method.

The directions of the computed controls for the $m = 2$ cases (with and without the time parameter) are shown in Fig. 4.7. The directions for all 100 maneuvers computed using the time expansion method closely align with the direction of the eigenvector corresponding to the stable mode of the STM propagated for one revolution (also called the monodromy matrix). The previous studies [57, 103, 27] have shown that optimal stationkeeping maneuvers for unstable periodic orbits generally align with the direction of the stable eigenvector. Even though the initial guesses for our algorithm did not take this knowledge into account, Fig. 4.7 shows that the STT time expansion strategy is successfully able to compute optimal maneuvers that align with the stable eigenvector

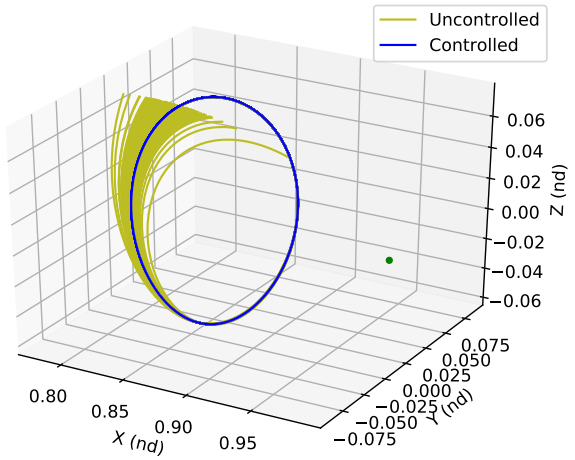

Figure 4.6: Uncontrolled and controlled orbits for 100 initial state deviation vectors for x-axis targeting scheme, using control computed with time expansion, for $m = 2$

Table 4.2: STT optimization method results for x-axis crossing stationkeeping algorithm

STT optimization algorithm	Average control magnitude (nd)	Average final target state error norm (nd)
Fixed time method ($m = 2$)	8.85×10^{-5}	3.42×10^{-9}
Time expansion method ($m = 2$)	6.20×10^{-5}	1.50×10^{-9}
Fixed time method ($m = 3$)	8.85×10^{-5}	2.63×10^{-11}
Time expansion method ($m = 3$)	6.20×10^{-5}	9.46×10^{-12}
Fixed time method ($m = 4$)	8.85×10^{-5}	1.65×10^{-12}
Time expansion method ($m = 4$)	6.20×10^{-5}	3.45×10^{-13}

Table 4.3: STT optimization method results for x-axis crossing stationkeeping algorithm (continued)

STT optimization algorithm	Average runtime (ms)	Average position deviation from reference orbit (nd)
Fixed time method ($m = 2$)	2.6	3.28×10^{-5}
Time expansion method ($m = 2$)	2.8	1.93×10^{-5}
Fixed time method ($m = 3$)	63.5	3.28×10^{-5}
Time expansion method ($m = 3$)	7.2	1.93×10^{-5}
Fixed time method ($m = 4$)	78.5	3.28×10^{-5}
Time expansion method ($m = 4$)	17.6	1.93×10^{-5}

(even when expanded only to second order), while the fixed time-of-flight algorithm clearly does not.

4.5 Conclusions

In this chapter, two methods were derived for including the derivatives of a dynamical system with respect to the final time in a state transition tensor (STT) approximation. These augmented STTs are shown to yield an accurate approximation of the dynamics around the reference final time. The STTs are used to compute guidance maneuvers with a variable time-of-flight for perturbed trajectories around the reference, as an extension to the fixed time-of-flight methods developed in Chapter 3. The resulting procedure is shown to be remarkably computationally lightweight and requires only the reference STTs. The strategy is applied to various targeting problems for stationkeeping in an unstable halo orbit in the Earth-Moon circular restricted three-body problem. The variable time-of-flight algorithm is shown to compute maneuvers with improved fuel requirements when compared with the fixed time-of-flight method.

In an operational use case, the variable time-of-flight STT algorithm could provide an efficient method for computing near-optimal maneuvers on-board a spacecraft in reaction to off-nominal performance or navigation errors. Previous on-board maneuver computation strategies that have relied on first-order corrections strategies to target maneuver parameters are generally sufficiently con-

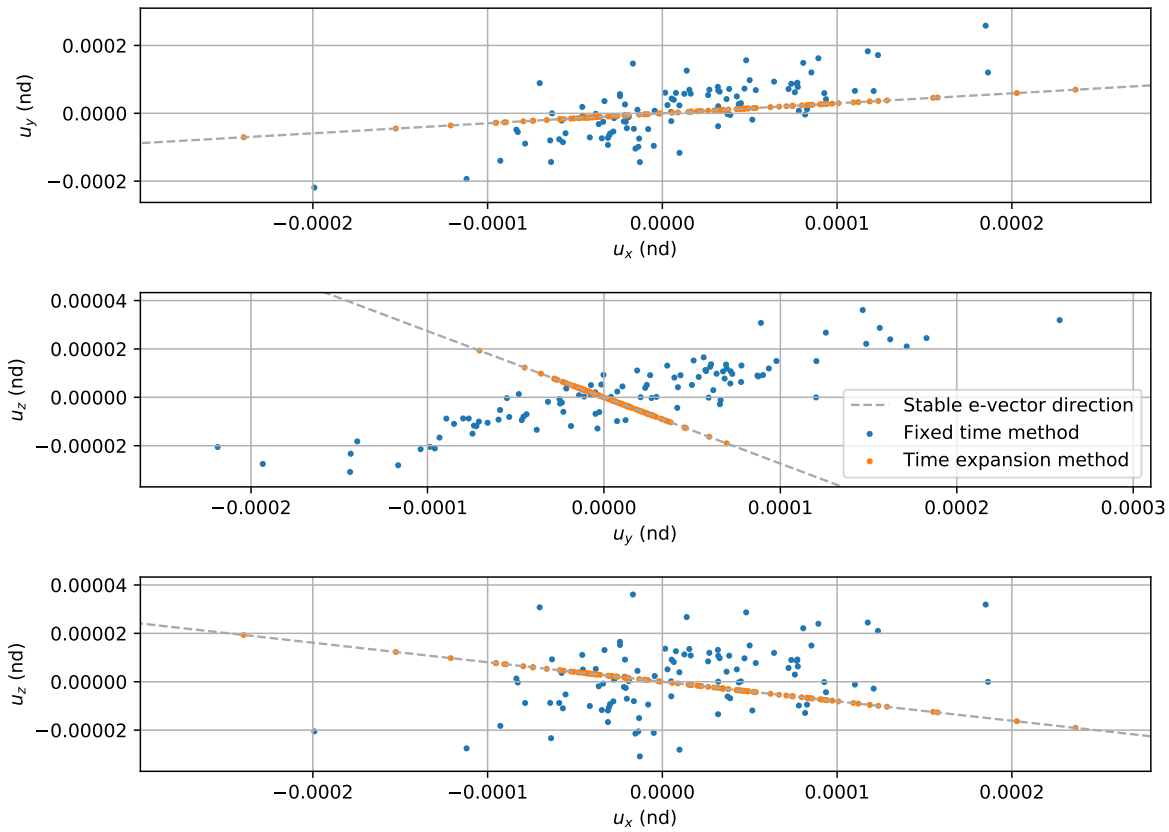

Figure 4.7: Directions of computed controls for 100 initial state deviation vectors for x-axis targeting scheme, with and without time parameter, for $m = 2$

strained that there is no need to optimize the control magnitude. The proposed algorithm provides greater flexibility in the target state, which could lead to large improvements in fuel requirements for on-board maneuver computation strategies, or could enable on-board computations for missions which require optimal or near-optimal maneuvers. Beyond the stated application for spacecraft maneuver targeting, the numerical methods presented in this chapter could also be useful for efficient nonlinear state and state uncertainty propagation with a variable final time.

Chapter 5

Rapid Trajectory Optimization using State Transition Tensors and Differential Dynamic Programming

5.1 Introduction

There has been significant interest in the space community in operating missions in complex dynamical regimes with low-thrust or continuous-thrust propulsion systems - for example, NASA's Lunar Gateway [74] and the DRACO program [1], which will both operate in cislunar space. Due to the highly nonlinear dynamics, optimizing trajectories in these regimes can be a time-consuming and sensitive process. Particularly in cislunar space, the timeframe for transfers between orbits can be on the order of days (as opposed to weeks or months). As such, the time, communication, and personnel requirements for ground-based navigation procedures could be a limiting factor for replanning transfers in reaction to navigation or maneuver execution errors. In addition, particularly for the large number of CubeSats that are planned to operate in cislunar space with low-thrust propulsion systems [36, 42, 34], there may be uncertain or varying departure and arrival conditions. This may necessitate conducting large-scale sensitivity analyses in order to ensure that the proposed low-thrust trajectories are sufficiently robust to the varying conditions of the transfer. Currently, these analyses are mostly conducted using either numerical Monte Carlo simulations [43] (which may be very expensive), or linearized techniques such as JPL's ADAM maneuver analysis system [29] (which may be inaccurate in highly nonlinear systems). A capability to rapidly generate accurate and near-optimal trajectories in the vicinity of some reference could therefore be highly beneficial.

A common feature of many existing ground-based trajectory optimization algorithms is that

they rely on the first (and often second) order derivatives of the dynamics to understand the local dynamical behavior and update controls guesses accordingly [69, 114]. When using full-fidelity models of the dynamics with realistic perturbations, no exact analytic representations of the dynamics and these derivatives are available. Running these algorithms therefore either requires repeated integrations of the dynamics (which can be prohibitively slow in real-time or on a flight computer), or the use of a lower-fidelity model to approximate the true dynamics. Several recent efforts have focused on yielding accurate approximations for these trajectories; for example, shape-based models have been used to obtain analytic approximations of continuous-thrust trajectories [100, 2]. These models are accurate for specific transfer geometries and dynamical regimes, but may not be suitable for multi-body or highly perturbed systems. Recently, a number of efforts have also concentrated on applying machine learning techniques [94, 105, 66, 59] or convex programming [55] to develop continuous-thrust guidance laws. These efforts are very promising, but may require novel transcriptions for incorporating constraints or defining objective functions. As such, they may not be able to rely on the well-understood methods from the full-fidelity optimization algorithms that are used in the design of the reference trajectory.

The algorithms developed in Chapters 3 and 4 can be used to optimize maneuvers for a multi-impulse transfer; however, they will likely become intractable for use when computing low-thrust trajectories, which may require optimizing over hundreds of thrusting parameters. In this chapter, we present a method to run differential dynamic programming (DDP), an existing low-thrust trajectory optimization algorithm [69], within higher-order approximations of the dynamics of a reference trajectory. As minimal modifications are required to the nominal DDP algorithm, most stage constraints and penalty methods that would be included in the standard algorithm can be accommodated. The resulting algorithm, called STT/DDP, can be used to efficiently and accurately optimize trajectories in the vicinity of a reference, which could enable more efficient trajectory re-planning and guidance, or expedite large scale mission design analyses.

This chapter is organized as follows. First, we provide a brief overview of the DDP algorithm for the optimization of spacecraft trajectories, and describe the modifications needed for the

STT/DDP algorithm. The STT/DDP algorithm is then used to compute a variety of near-optimal continuous-thrust trajectories in the Earth-Moon circular restricted three-body problem (CR3BP), a highly nonlinear dynamical regime for which no analytical solution is available. The method is able to compute new trajectories in a matter of seconds for significant deviations in both the initial conditions and the final target state, and for transfers using a different cost function than the reference.

5.2 Differential Dynamic Programming

The analytical derivative properties of an STT-approximated dynamical system (Eqs. 2.23 and 2.25 can be used to construct an analytical optimization algorithm using differential dynamic programming (DDP). In this section we will briefly describe DDP, before we outline in Section 5.3 the modifications and additions to the standard DDP formulation required for the STT/DDP algorithm. For a complete derivation of DDP, readers can refer to Refs. [60], [69] or [4]. We note that the purpose of the work in this chapter is not to develop the most sophisticated DDP implementation but rather to show that repeated numerical integrations can be accurately approximated using a reference trajectory's STTs.

DDP is a second-order local dynamic programming algorithm, in which a quadratic approximation of the cost-to-go around a trajectory is computed and correspondingly, a local linear-feedback controller of the form $\mathbf{u}_k = \bar{\mathbf{u}}_k + B_k \delta \mathbf{x}_k$ is obtained. DDP consists of successive backward sweeps and forward passes. The trajectory is discretized into $N+1$ stages, and the backward sweep solves the sequence of quadratic subproblems that minimize the cost-to-go from stage $k = N, N-1, ..., 0$ to obtain a prediction for the optimal control update at each stage $\delta \mathbf{u}_k$. In the forward pass, the dynamics are re-integrated, the new control updates are applied and a new quadratic expansion is performed around the resulting trajectory. Terminal constraints can be adjoined to the cost function using a constant vector of Lagrange multipliers λ. Iterations are repeated until the expected reduction in the cost function is below a pre-specified tolerance ϵ_{opt}, and terminal constraints are satisfied within the tolerance ϵ_{feas}.

This DDP implementation borrows several developments from the algorithm developed by Lantoine and Russell [69] for spacecraft trajectory optimization, deemed hybrid differential dynamic programming (HDDP), namely the use of the state transition matrix and tensor to propagate the cost-to-go through stages during the backward sweep. A derivation of the general DDP algorithm is included in the following sections. In this thesis, we use index notation to represent matrix and tensor multiplication operations, in order to remain consistent with the STT notation.

5.2.1 Forward pass

In the forward pass, a control law $\mathbf{u} = \bar{\mathbf{u}} + \delta\mathbf{u}$ is applied. Using the solution flow notation from Eq. 2.10, the forward pass can be expressed as

$$\mathbf{x}(t_{k+1}) = \boldsymbol{\phi}(t_k; \mathbf{x}_k, \mathbf{u}_{(t_{k+1},t_k)}, t_k) \tag{5.1}$$

5.2.2 Second-order expansion

If a forward pass iterate is accepted by the DDP algorithm, a second-order expansion of the dynamics around the new trajectory is performed. In the HDDP algorithm [69], the second-order expansion is obtained by numerically integrating the STM and second-order STT for the new trajectory, using Eqs. 2.15 and 2.16. Computing these derivatives is typically by far the most computationally expensive portion of the standard numerical DDP algorithm.

5.2.3 Backward sweep

The backward sweep solves the sequence of quadratic subproblems that minimize the cost-to-go from stage $k = N, N-1, ..., 0$. This can be stated as [4]

$$J_k^* = \min_{\delta\mathbf{u}_k} [J_k] \tag{5.2}$$

where J_k represents the cost-to-go at stage k, and J_k^* is the locally minimized cost-to-go at state k after applying the optimal control update $\delta\mathbf{u}_k^*$. L_k is the local cost at stage k; the cost-to-go at

stage k is obtained by summing the local cost and the cost-to-go from the subsequent stage.

$$J_k = L_k + J^*_{k+1} \tag{5.3}$$

The quadratic subproblem is formulated by performing a second-order expansion of J_k as a function of $\delta\mathbf{x}_k$, $\delta\mathbf{u}_k$, and $\delta\lambda$.

$$\begin{aligned}\delta J_k = \quad & ER_{k+1} + J^{\alpha_1}_{x,k}\delta x^{\alpha_1}_k + J^{\beta_1}_{u,k}\delta u^{\beta_1}_k + J^{\eta_1}_{\lambda,k}\delta\lambda^{\eta_1} + \frac{1}{2}J^{\alpha_1\alpha_2}_{xx,k}\delta x^{\alpha_1}_k\delta x^{\alpha_2}_k + \frac{1}{2}J^{\beta_1\beta_2}_{uu,k}\delta u^{\beta_1}_k\delta u^{\beta_2}_k \\ & + \frac{1}{2}J^{\eta_1\eta_2}_{\lambda\lambda,k}\delta\lambda^{\eta_1}\delta\lambda^{\eta_2} + J^{\alpha_1\beta_1}_{xu,k}\delta x^{\alpha_1}_k\delta u^{\beta_1}_k + J^{\alpha_1\eta_1}_{x\lambda,k}\delta x^{\alpha_1}_k\delta\lambda^{\eta_1} + J^{\beta_1\eta_1}_{u\lambda,k}\delta u^{\beta_1}_k\delta\lambda^{\eta_1}\end{aligned} \tag{5.4}$$

In this equation, ER_{k+1} is the predicted change in the cost for stages $k+1$ through N. Again, to be consistent throughout the thesis, we use index notation to express the various matrix and tensor operations. Taking the derivative of Eq. 5.4 with respect to $\delta\mathbf{u_k}$ and setting it equal to zero gives a linear feedback control law for the locally optimal control $\delta\mathbf{u}^*_k$:

$$\delta u^{*,\beta_1}_k = A^{\beta_1}_k + B^{\beta_1\alpha_1}_k\delta x^{\alpha_1}_k + D^{\beta_1\eta_1}_k\delta\lambda^{\eta_1} \tag{5.5}$$

$$A^{\beta_1}_k = -\left(J^{-1}_{uu,k}\right)^{\beta_1\beta_2} J^{\beta_2}_{u,k} \tag{5.6}$$

$$B^{\beta_1\alpha_1}_k = -\left(J^{-1}_{uu,k}\right)^{\beta_1\beta_2} J^{\beta_1\alpha_1}_{ux,k} \tag{5.7}$$

$$D^{\beta_1\eta_1}_k = -\left(J^{-1}_{uu,k}\right)^{\beta_1\beta_2} J^{\beta_2\eta_1}_{u\lambda,k} \tag{5.8}$$

Before proceeding upstream from stage k to $k-1$, the stage update equations predict the effects of the updated control $\delta\mathbf{u}^*_k$ on stage k. To obtain these we perform a quadratic expansion of J^*_k as a function of $\delta\mathbf{x}_k$ and $\delta\lambda_k$, giving

$$\delta J^*_k = ER_k + J^{*,\alpha_1}_{x,k}\delta x^{\alpha_1}_k + J^{*,\eta_1}_{\lambda,k}\delta\lambda^{\eta_1} + \frac{1}{2}J^{*,\alpha_1\alpha_2}_{xx,k}\delta x^{\alpha_1}_k\delta x^{\alpha_2}_k + \frac{1}{2}J^{\eta_1\eta_2}_{\lambda\lambda,k}\delta\lambda^{\eta_1}\delta\lambda^{\eta_2} + J^{\alpha_1\eta_1}_{x\lambda,k}\delta x^{\alpha_1}_k\delta\lambda^{\eta_1} \tag{5.9}$$

We can then insert the optimal control law from Eq. 5.5 into Eq. 5.4, and match coefficients of equivalent orders of $\delta\mathbf{x}_k$ and $\delta\lambda_k$ to obtain the following expressions:

$$ER_k = ER_{k+1} + J_{u,k}^{\beta_1} A_k^{\beta_1} + \frac{1}{2} J_{uu,k}^{\beta_1\beta_2} A_k^{\beta_1} A_k^{\beta_2} \tag{5.10}$$

$$J_{x,k}^{*,\alpha_1} = J_{x,k}^{\alpha_1} + J_{u,k}^{\beta_1} B_k^{\beta_1\alpha_1} + J_{uu,k}^{\beta_1\beta_2} A_k^{\beta_1} B_k^{\beta_2\alpha_1} + J_{ux,k}^{\beta_1\alpha_1} A_k^{\beta_1} \tag{5.11}$$

$$J_{\lambda,k}^{*,\eta_1} = J_{\lambda,k}^{\eta_1} + J_{u,k}^{\beta_1} D_k^{\beta_1\eta_1} + J_{uu,k}^{\beta_1\beta_2} A_k^{\beta_1} D_k^{\beta_2\eta_1} + J_{u\lambda,k}^{\beta_1\eta_1} A_k^{\beta_1} \tag{5.12}$$

$$J_{xx,k}^{*,\alpha_1\alpha_2} = J_{xx,k}^{\alpha_1\alpha_2} + J_{uu,k}^{\beta_1\beta_2} B_k^{\beta_1\alpha_1} B_k^{\beta_1\alpha_2} + 2 J_{ux,k}^{\beta_1\alpha_2} B_k^{\beta_1\alpha_1} \tag{5.13}$$

$$J_{\lambda\lambda,k}^{*,\eta_1\eta_2} = J_{\lambda\lambda,k}^{\eta_1\eta_2} + J_{uu,k}^{\beta_1\beta_2} D_k^{\beta_1\eta_1} D_k^{\beta_2\eta_2} + 2 J_{u\lambda,k}^{\beta_1\eta_2} D_k^{\beta_1\eta_1} \tag{5.14}$$

$$J_{x\lambda,k}^{*\alpha_1\eta_1} = J_{x\lambda,k}^{\alpha_1\eta_1} + J_{uu,k}^{\beta_1\beta_2} B_k^{\beta_1\alpha_1} D_k^{\beta_2\eta_1} + J_{u\lambda,k}^{\beta_1\eta_1} B_k^{\beta_1\alpha_1} + J_{ux,k}^{\beta_1\alpha_1} D_k^{\beta_1\eta_1} \tag{5.15}$$

Finally, to proceed from stage $k+1$ to stage k, the derivatives of the optimal cost-to-go at stage $k+1$ must be mapped to stage k. For clarity, we will combine the state vector $\mathbf{x}$ and control vector $\mathbf{u}$ into an augmented state vector $\mathbf{X}^T = [\mathbf{x}^T \mathbf{u}^T]$. We can then write the equations for the stage cost-to-go derivatives from the backward sweep for stage k as

$$J_{X,k}^{\kappa_1} = L_{X,k}^{\kappa_1} + J_{X,k+1}^{*,\gamma_1} \Phi^{\gamma_1\kappa_1} \tag{5.16}$$

$$J_{\lambda,k}^{\eta_1} = J_{\lambda,k+1}^{*,\eta_1} \tag{5.17}$$

$$J_{XX,k}^{\kappa_1\kappa_2} = L_{XX,k}^{\kappa_1\kappa_2} + J_{XX,k+1}^{*\gamma_1\gamma_2} \Phi^{\gamma_1\kappa_1} \Phi^{\gamma_2\kappa_2} + J_{X,k+1}^{*\gamma_1} \Phi^{\gamma_1\kappa_1\kappa_2} \tag{5.18}$$

$$J_{\lambda\lambda,k}^{\eta_1\eta_2} = J_{\lambda\lambda,k+1}^{*,\eta_1\eta_2} \tag{5.19}$$

$$J_{X\lambda,k}^{\kappa_1\eta_1} = J_{X\lambda,k+1}^{*,\gamma_1\eta_1} \Phi^{\gamma_1\kappa_1} \tag{5.20}$$

where the Φ terms correspond to the STM and second-order STT mapping the augmented state vector X from time t_k to t_{k+1}, and the κ_n and γ_n indices are used, for clarity, to represent derivatives with respect to the augmented state vector at stage k and $k+1$, respectively.

5.2.4 Augmented Lagrangian function

As in Lantoine and Russell[69], we add a quadratic penalty parameter σ to place additional weight on the terminal constraints. The augmented cost function then becomes

$$\tilde{J} = J + \lambda^i \psi^i + \sigma^{ij} \psi^i \psi^j \tag{5.21}$$

where the vector $\boldsymbol{\psi}$ is the terminal constraint vector - i.e. $\boldsymbol{\psi} = \mathbf{x}_{N+1} - \mathbf{x}_{\text{target}}$. The terminal constraints will be satisfied when $\|\boldsymbol{\psi}\| < \epsilon_{\text{feas}}$. For this work we keep σ constant between all iterations as this was found to be sufficient for producing feasible trajectories for the scenarios that were investigated. Some approaches will continually increase the penalty weight to push successive iterations towards feasibility [69].

The backward sweep portion of the DDP algorithm is thus initialized at the final stage $N+1$ by setting

$$J^*_{N+1} = \lambda^i \psi^i + \sigma^{ij} \psi^i \psi^j \tag{5.22}$$

With this form of terminal constraints, the optimal cost-to-to derivatives are straightforward to compute at stage $N+1$ as:

$$J^{*,\kappa_1}_{x,N+1} = \lambda^{\kappa_1} + 2\sigma^{\kappa_1 j} \psi^j \tag{5.23}$$

$$J^{*,\eta_1}_{\lambda,N+1} = \psi^{\eta_1} \tag{5.24}$$

$$J^{*,\kappa_1\kappa_2}_{xx,N+1} = 2\sigma^{\kappa_1\kappa_2} \tag{5.25}$$

$$J^{*,\kappa_1\eta_1}_{x\lambda,N+1} = \mathbb{I}^{\kappa_1\eta_1}_{n\times n} \tag{5.26}$$

$$J^*_{u,N+1} = J^*_{ux,N+1} = J^*_{u\lambda,N+1} = J^*_{\lambda\lambda,N+1} = ER_{N+1} = 0 \tag{5.27}$$

5.2.5 Trust-region subproblem

DDP will often take large steps toward the minimum; if these are not constrained, the steps may lie outside the accuracy region of the local quadratic approximation, or may lead to infeasible iterates. Many implementations [69, 5] address this issue by solving a trust-region quadratic subproblem (TRQP) at each stage. An extensive review of trust-region methods is available in Conn et. al.[33]; the methods from this source were found to be sufficient for the problems at hand. The TRQP for stage k can be stated as

$$\min_{\delta \mathbf{u}_k} \left[J_{u,k}^T \delta \mathbf{u}_k + \frac{1}{2} \delta \mathbf{u}_k^T J_{uu,k} \delta \mathbf{u}_k \right] \tag{5.28}$$

$$\text{subject to } \|D \delta \mathbf{u}_k\| \leq \Delta \tag{5.29}$$

The parameter Δ represents the size of the trust region, and D is a scaling matrix that can be used to adjust the shape of the trust region. We found that setting D to identity was sufficient for our applications. In this implementation we use an algorithm to adaptively increase and decrease Δ, following Conn et al. [33]. For the STT/DDP algorithm, the trust region method also serves to keep successive iterations in proximity to the reference trajectory, and thus within the expected accuracy region of the higher-order STTs.

5.2.6 Form of control and cost function

The choice of control affects how the state transition matrices and tensors in Eqs. 5.16-5.20 are computed. If we consider a continuous control, additional differential equations would need to be integrated to obtain the augmented STM. To simplify the formulation for the initial demonstrations in this chapter, we treat the control vector as an impulsive change in velocity at the beginning of each stage.

$$\mathbf{u}_k = \Delta \mathbf{v}_{t_k} \tag{5.30}$$

Initially in this chapter we will use the minimum-energy cost function, which corresponds to the sum of the square of the magnitude of the velocity at each stage. The local cost L_k at stage k becomes

$$L_k = \|\mathbf{u}_k\|^2 = u_{k,x}^2 + u_{k,y}^2 + u_{k,z}^2 \tag{5.31}$$

Note that this form of the cost function will result in different optimal control policies than using the total fuel usage - the optimal minimum-energy policy will have the controls spread out over all stages as opposed to a bang-bang thrust profile. We will later show that we can use the STT/DDP method to compute trajectories with a different cost function than the reference. For this we use the minimum-fuel formulation of the cost, which can be written as

$$L_k = \|\mathbf{u}_k\| = \sqrt{u_{k,x}^2 + u_{k,y}^2 + u_{k,z}^2 + \epsilon_{ml}} \tag{5.32}$$

where ϵ_{ml} is a small mass leak term. This term is necessary to resolve the singularity for the derivatives of the minimum-fuel cost function during coast arcs.

5.3 State Transition Tensor DDP

Many spacecraft trajectory optimization algorithms (including DDP) rely on repeated evaluations of the first and second-order derivatives of the dynamics to converge on locally optimal linear updates to the control [69, 114]. One can expand upon this formulation and develop algorithms based on higher-order derivatives [8, 107, 71]. In some cases, these have been shown to yield larger convergence regions and more robust control laws, but at a tremendous increase in the computational requirements. Ultimately, in this tradeoff, the linear-quadratic expansion has been shown to achieve a "sweet spot" in the balance between computational complexity and efficiency. As such, the vast majority of controls and optimization literature has focused on developing methods using successive second-order expansions of the dynamics.

Over the past few decades, a wealth of literature has been developed on trust region methods,

penalty methods, and safeguards, specifically for quadratic expansions of the dynamics [117, 33]. We would like to take advantage of the existing methods as much as possible, while retaining the improved dynamical knowledge given by the higher-order terms. Eqs. 2.23 and 2.25 give expressions for the first and second-order derivatives of the STT approximation for a perturbed trajectory in the vicinity of a reference, in terms of the reference trajectory's higher-order STTs. These expressions will allow us to run the second-order differential dynamic programming algorithm from Section 5.2 within a higher-order STT approximation of the dynamics.

5.3.1 Algorithm

The computation of the first and second-order derivatives, used in Eqs. 5.16-5.20 for the backward sweep in each iteration, is the most computationally expensive portion of the numerical DDP algorithm. In order to alleviate this burden, we can integrate the higher-order STTs along a reference trajectory, and subsequently use the exact derivatives of the STT-approximated dynamics (see Eqs. 2.23 and 2.25) to run an "approximation" of the numerical DDP algorithm, with significant improvements in computational time. We refer to this strategy as STT/DDP. The STT/DDP algorithm will be accurate so long as the successive quadratic expansions around each DDP iteration lie within the accuracy region of the reference trajectory's higher-order STTs.

The formulation of the STT/DDP algorithm is as follows. Given a reference trajectory with an associated reference control history $\hat{\mathbf{u}}$, we can first divide the reference trajectory into $N+1$ stages, and integrate the higher-order STTs for each stage along this reference. The numerical DDP algorithm is modified by replacing any integrations of the dynamics with evaluations of these STTs. For clarity, we use $\delta\hat{\mathbf{x}}_k$ to refer to state deviations from the **original reference trajectory**. Similarly, $\hat{\phi}^{i,\gamma_1\cdots\gamma_p}_{(t_{k+1},t_k)}$ refers to the reference trajectory STTs of order p, mapping from stage k to stage $k+1$. The definitions of the state and state deviations for the reference trajectory and STT/DDP iteration are illustrated in Fig. 5.1.

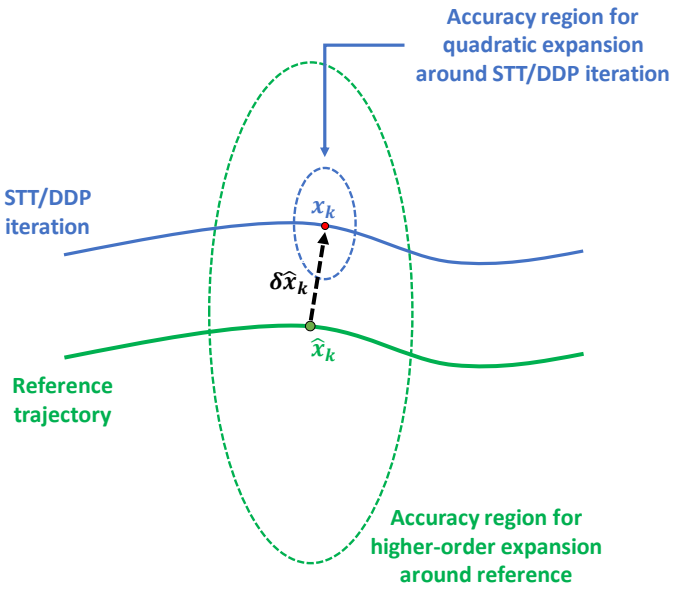

Figure 5.1: STT/DDP state and state deviation definitions. The quadratic expansion around the STT/DDP iteration lies within the accuracy region for the higher-order expansion around the reference; thus, the quadratic expansion can be accurately approximated using the higher-order reference STTs.

For each iteration, the forward pass in the DDP algorithm can be restated as

$$x^i_{k+1} = \left[\hat{x}^i_{k+1} + \sum_{p=1}^{m} \frac{1}{p!} \hat{\phi}^{i,\gamma_1 \ldots \gamma_p}_{(t_{k+1},t_k)} \delta\hat{x}^{\gamma_1}_k \ldots \delta\hat{x}^{\gamma_p}_k \right] + u^i_{k+1} \tag{5.33}$$

Since the dynamics are approximated using the reference STTs, Eqs. 2.23 and 2.25 can then be used to obtain the STM and second-order STT for the new trajectory computed during the forward pass:

$$\phi^{i,\gamma_1}_{(t_{k+1},t_k)} \simeq \hat{\phi}^{i,\gamma_1}_{(t_{k+1},t_k)} + \sum_{p=2}^{m} \frac{1}{(p-1)!} \hat{\phi}^{i,\gamma_1,\gamma_2 \ldots \gamma_p}_{(t_{k+1},t_k)} \delta\hat{x}^{\gamma_2}_k \ldots \delta\hat{x}^{\gamma_p}_k \tag{5.34}$$

$$\phi^{i,\gamma_1\gamma_2}_{(t_{k+1},t_k)} \simeq \hat{\phi}^{i,\gamma_1\gamma_2}_{(t_{k+1},t_k)} + \sum_{p=3}^{m} \frac{1}{(p-2)!} \hat{\phi}^{i,\gamma_1,\gamma_2,\gamma_3 \ldots \gamma_p}_{(t_{k+1},t_k)} \delta\hat{x}^{\gamma_3}_k \ldots \delta\hat{x}^{\gamma_p}_k \tag{5.35}$$

No further modifications are required. We note again that while Eqs. 5.34 and 5.35 are approximations of the true state derivatives, they in fact correspond to the exact derivatives of the STT-approximated dynamics from Eq. 5.33.

5.3.2 Enforcing accuracy of the STTs

The STT/DDP algorithm will yield a locally optimal trajectory within the STT-approximated dynamics. If this trajectory lies within the accuracy region of the reference's STTs, it will correspond to a nearly-optimal trajectory in the true dynamics. However, if the optimal trajectory obtained from the STT/DDP algorithm is too far from the reference STTs, the STT approximations may not be sufficiently accurate. In this case, the algorithm is no longer useful as it is not an accurate representation of the actual dynamics. It is therefore important to implement a method to enforce successive iterations to remain within the accuracy region of the reference STTs.

Because the STTs represent a Taylor expansion up to order m integrated through time, it is difficult to explicitly predict the error from ignoring terms of $\mathcal{O}(\varepsilon^{m+1})$ (where $\varepsilon \ll 1$ and $\delta\mathbf{x}_0 \sim \mathcal{O}(\varepsilon)$). Nevertheless, we can use elements of perturbation theory [7] and knowledge of how a convergent series should behave to derive a penalty method to force these errors to be small. If we

consider that an STT of order m contains secular terms that grow over time like $t, t^2, ..., t^m$ [99], we can see that the expansion will break down when $t \sim \mathcal{O}(1/\varepsilon)$. In this case, the asymptotic ordering of the terms in the series breaks down and the series is no longer convergent.

However, if the order m term $\frac{1}{m!}\phi^{i,\gamma_1 \cdots \gamma_m}\delta\hat{x}_k^{\gamma_1}...\delta\hat{x}_k^{\gamma_m}$ (i.e. the highest-order term in the expansion) is sufficiently small, and the integration time is not too long ($t \sim \mathcal{O}(1)$), then we can assume that the series is convergent, and that the approximation error from ignoring all terms of order $m+1$ (and greater) is of $\mathcal{O}(\varepsilon^{m+1}) \ll \mathcal{O}(\varepsilon^m)$. Thus, if the order m term is small, we can assume that the truncation error is smaller than this term, and that the series is sufficiently accurate. In order to enforce a small order m term, we can apply a quadratic penalty at each stage on its magnitude. This is scaled with a weight W to ensure that the order m term is of the desired order of magnitude. The penalty parameter to be added to the local cost function at each stage k then becomes

$$L_k = W(\frac{1}{m!}\phi^{i,\gamma_1 \cdots \gamma_m}\delta\hat{x}_k^{\gamma_1}...\delta\hat{x}_k^{\gamma_m})(\frac{1}{m!}\phi^{i,\gamma_1 \cdots \gamma_m}\delta\hat{x}_k^{\gamma_1}...\delta\hat{x}_k^{\gamma_m}) \tag{5.36}$$

where $\delta\hat{\mathbf{x}}_k$ represents the deviation from the **reference** trajectory. For this work, W was set to be the same for all state components, but it could be replaced by a vector of weights to place emphasis on specific components.

In order to include this penalty parameter in the DDP formulation, its first and second order partial derivatives with respect to the state vector X must be derived. Fortunately, this form of penalty parameter is relatively straightforward to differentiate. First, we will define the i-th component of the order m term as β^i:

$$\beta^i = \frac{1}{m!}\phi^{i,\gamma_1 \cdots \gamma_m}\delta\hat{x}_k^{\gamma_1}...\delta\hat{x}_k^{\gamma_m} \tag{5.37}$$

The first and second order derivatives of β^i with respect to the states can be expressed

analytically as a function of the reference STTs:

$$\beta^{i,a} = \frac{1}{(m-1)!}\phi^{i,a\gamma_2\ldots\gamma_m}\delta\hat{x}_k^{\gamma_2}\ldots\delta\hat{x}_k^{\gamma_m} \tag{5.38}$$

$$\beta^{i,ab} = \frac{1}{(m-2)!}\phi^{i,ab\gamma_3\ldots\gamma_m}\delta\hat{x}_k^{\gamma_3}\ldots\delta\hat{x}_k^{\gamma_m} \tag{5.39}$$

Note that the number of superscripts after the i indicates the order of derivative of β^i with respect to the state vector. We can then write the stage quadratic penalty function from Eq. 5.36 as

$$L_k = W\beta^i\beta^i \tag{5.40}$$

and the stage derivatives of this local cost function can be written compactly as

$$L^a_{X,k} = 2W\beta^{i,a}\beta^i \tag{5.41}$$

$$L^{ab}_{XX,k} = 2W\left[\beta^{i,a}\beta^{i,b} + \beta^i\beta^{i,ab}\right] \tag{5.42}$$

The weighting parameter W must be carefully chosen to be large enough to ensure that the terms of order m are maintained sufficiently small, but not too large, in which case the terms may be over-penalized to the point that they barely contribute to the approximation. An alternative formulation of the penalty term is presented in Ref. [45], where a penalty is applied to the magnitude of the full STT state deviation summation rather than the highest-order term.

5.3.3 Resulting feedback policy

The STT/DDP method yields a feedback law of the form $\mathbf{u}_k = \bar{\mathbf{u}}_k + B_k\delta\mathbf{x}_k$, following Eqs. 5.5 and 5.7. The open-loop component $\bar{\mathbf{u}}_k$ of the feedback law is optimal in the STT-approximated dynamics. However, when applying the resulting control law in the true dynamics, there will be approximation errors of $\mathcal{O}(\varepsilon^{m+1})$ at each stage. Over the course of an entire transfer, these errors may compound if they are too large, and potentially result in a large final error in reaching

the desired target. This can be corrected for by applying the feedback law at each stage, where $\delta\mathbf{x}_k$ corresponds to the difference between the STT-predicted state at stage k and the observed or numerically integrated state. Thus, if the STTs perfectly approximate the true dynamics to within numerical precision, $\delta\mathbf{x}_k$ would equal 0 at each stage. Additionally, in an operational trajectory planning setting, the control law could be used to correct for navigation errors, off-nominal performance, or unforeseen events.

5.4 L_2 to L_1 Halo Orbit Transfer in the Earth-Moon System

We first apply the STT/DDP formulation to a continuous-thrust transfer in the Earth-Moon system, from a halo orbit around the L_2 Lagrange point to a halo orbit around the L_1 Lagrange point. Through this example, we will show how the algorithm can be used to rapidly generate transfers with a different target state from the reference, and we will compare its performance with the numerical DDP algorithm. For this example, and all subsequent examples in this chapter, we approximate the dynamics of the Earth-Moon system using the circular restricted three-body problem (CR3BP) - see Eqs. 2.7- 2.9 for the equations of motion. For the Earth-Moon system, we use $\mu = 0.0121505856$.

5.4.1 Scenario

First, a reference transfer was generated using the numerical DDP algorithm. The initial and target state data for this halo orbit transfer was obtained from Ref. [5] and is shown in Table 5.1. These halo orbits have differing Jacobi constants C. The transfer time was chosen to be 4.4 non-dimensional time units (roughly 20 days) and the transfer was split into 110 stages of equal time length. The numerical DDP algorithm as outlined in the Section 5.2 was used to obtain a nominal transfer trajectory governed by a nominal control law of the form $\mathbf{u}_k = \bar{\mathbf{u}}_k + B_k\delta\mathbf{x}_k$. This nominal trajectory and the initial and final halo orbits are shown in Fig. 5.2. The higher-order STTs of this reference trajectory were then integrated separately for each stage and stored, and we can explore the potential for efficiently computing new continuous-thrust trajectories in the vicinity of this

Table 5.1: Halo-to-halo transfer orbit scenario parameters

Orbit	x	y	z	$\dot{x}$	$\dot{y}$	$\dot{z}$	C
Initial	1.160797311	0.0	-0.122697	0.0	-0.207683284	0.0	3.0942
Reference final	0.848710153	0.0	0.173890	0.0	0.263500947	0.0	3.0090
New final	0.874998280	0.0	0.1914	0.0	0.232342084	0.0	2.9979

reference using the STT/DDP method. For this case study, we set $\sigma = 1 \times 10^3$, $\epsilon_{opt} = 1 \times 10^{-12}$, and $\epsilon_{feas} = 1 \times 10^{-8}$ for both the numerical and STT/DDP algorithms, and used $W = 0.5$ for the penalty term to enforce accuracy of the STT method.

The feedback law obtained from any DDP algorithm can be used to compute the controls to steer a spacecraft towards its original target state in response to navigation or performance errors in the initial state. If the spacecraft needs to reach a new target state, however, a new open-loop control policy and corresponding feedback law is required. Therefore, for this scenario, we will show that the STT/DDP method can be used to target a different halo orbit from the reference target. The new target orbit parameters are shown in Table 5.1 and the orbits are illustrated in Fig. 5.3. The new target is far from the original target orbit, with position differences on the order of 10,000 km.

5.4.2 Comparison of different STT orders

To begin with, we will show the benefits of including higher orders of reference STTs in the approximation. The STT/DDP method was run for values of $m \in [2, 3, 4]$, where m is the maximum order of STT included in the approximation. For each order, the optimal feedback law of the form $\mathbf{u}_k = \bar{\mathbf{u}}_k + B_k \delta \mathbf{x}_k$ was obtained using the STT/DDP method (with the penalty parameter to enforce accuracy), and the STT state predictions from the optimal forward pass were saved for each stage. To validate the resulting transfer, the trajectory was then numerically integrated in the true dynamics and the feedback law was applied at each stage to correct for small approximation errors, as outlined in Section 5.3.3. The resulting transfers for both the STT/DDP algorithm at

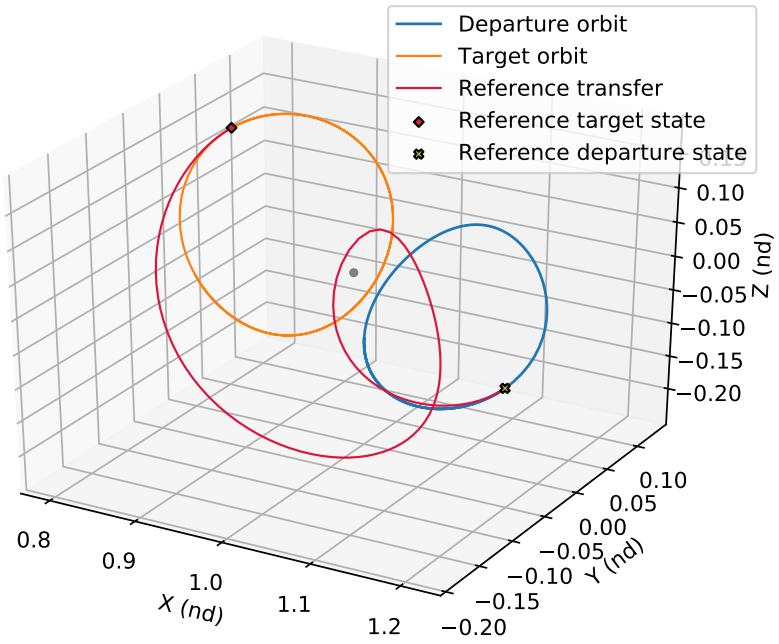

Figure 5.2: Reference halo orbit transfer

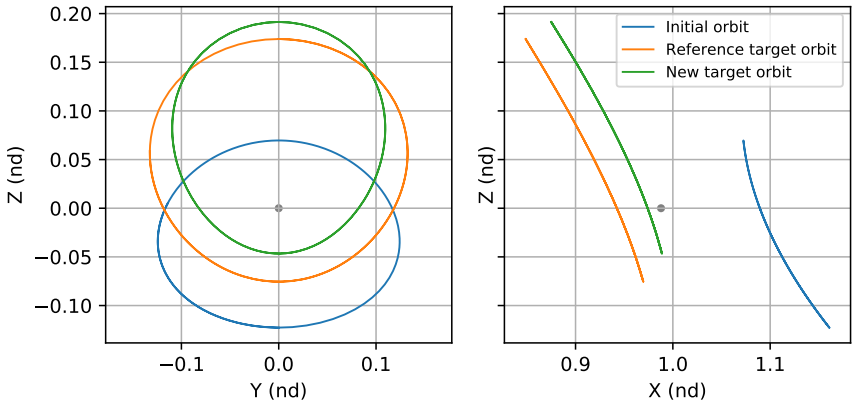

Figure 5.3: Reference and new target halo orbits, 2D view

Table 5.2: STT/DDP algorithm performance at various orders m for targeting new halo orbit

Algorithm	Final cost J	Computation time (s)	Number of iterations	Final state error $\|\boldsymbol{\psi}\|$ with feedback law
STT/DDP, $m = 2$	0.003179	1.91	20	4.51e-5
STT/DDP, $m = 3$	0.003149	2.46	23	7.23e-6
STT/DDP, $m = 4$	0.003142	3.27	24	1.18e-6
Numerical DDP	0.003137	118.32	24	4.92e-15

$m = 4$ and the numerical DDP algorithm are illustrated in Figs. 5.4 and 5.5. The thrust profiles for the trajectories computed using each algorithm are shown in Fig. 5.6.

Table 5.2 shows the performance of the STT/DDP algorithm for this scenario up to fourth order. This is compared to the performance of the numerical DDP algorithm. The dynamics in this transfer are sufficiently nonlinear that they are not perfectly approximated using only the second-order STTs - the cost using the second-order method was found to be 1.34% higher than when using numerical DDP. Including higher orders of STTs improves the accuracy of the results; the discrepancy from the numerical DDP result was found to be 0.38% and 0.16% for the third and fourth-order STT/DDP methods, respectively. In addition, as no further numerical integrations are required, the solution using the STT/DDP method is computed significantly faster than the standard numerical DDP method.

5.4.3 Performance comparison

In order to compare the performance of the two methods, we will examine the fourth-order STT solution in further detail. All computations performed by the DDP algorithms are grouped into three main categories: forward pass, backward sweep, and STM integration. The computational time and fraction of total time spent performing each category of computation is shown in Table 5.3. The time required to perform the backward sweep is similar for both algorithms, but the time to perform the forward pass and, in particular, the computation of the STMs at each stage is significantly reduced when using the STT/DDP method. In fact, the backward sweep, which

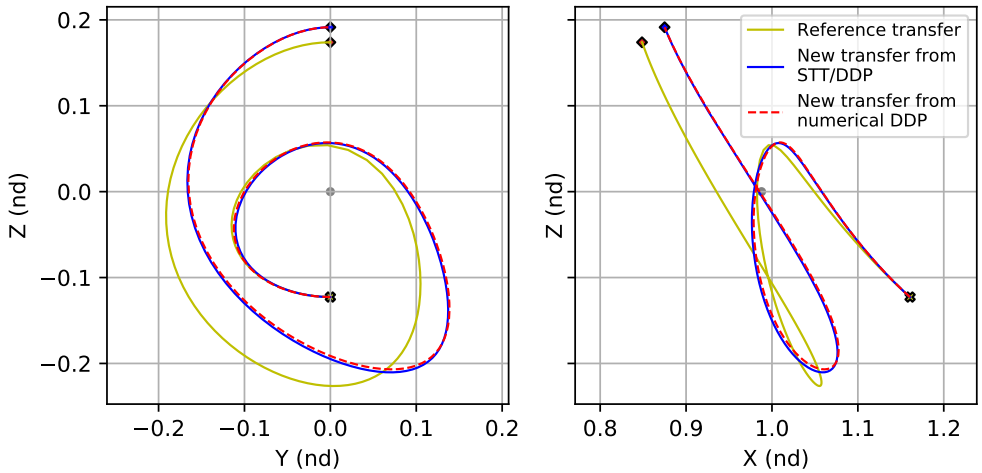

Figure 5.4: Transfer to new target halo orbit, 2D view

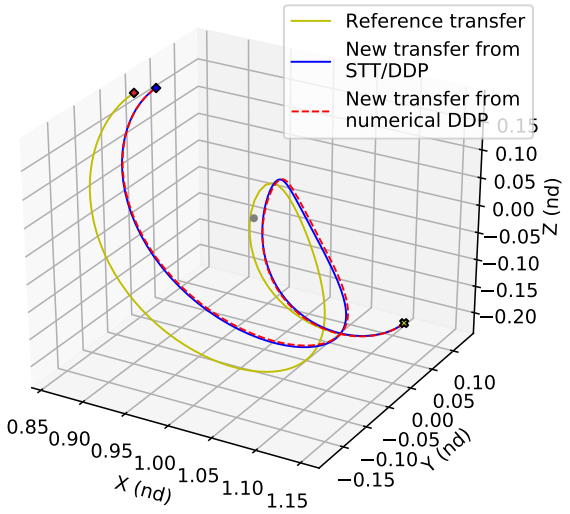

Figure 5.5: Transfer to new target halo orbit, 3D view

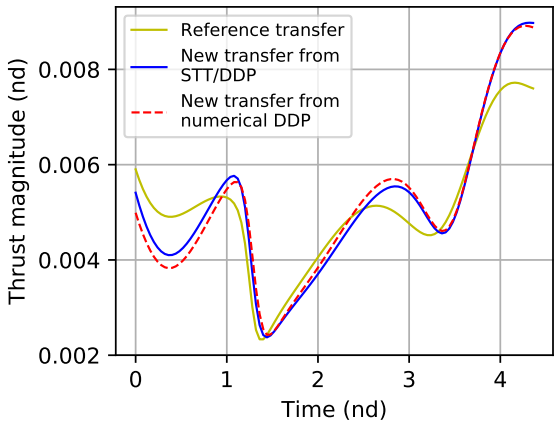

Figure 5.6: Thrust magnitude over time, transfer to new target orbit

Table 5.3: Computational time for different segments of DDP algorithms

	STT/DDP, $m = 4$			Numerical DDP		
	Time (s)	Percent of total time	Time (s) per 1 iteration	Time (s)	Percent of total time	Time (s) per 1 iteration
Forward pass	0.63	19.4%	0.03	19.62	16.6%	0.82
Backward sweep	1.95	59.6%	0.08	2.12	1.8%	0.09
STM computation	0.66	20.2%	0.03	96.12	81.2%	4.00

represents only 1.8% percent of the total computational time for the numerical DDP algorithm, becomes the most computationally expensive segment of the STT/DDP algorithm.

We note that all code was written in Python, which is a relatively slow language. Using a compiled language such as C or Julia would result in computational improvements for both algorithms. We also note that the time required to integrate the reference STTs is not included in this comparison; we assume that this would be done offline prior to the execution of the STT/DDP algorithm. All code was run in serial - the computation of the derivatives at each stage could be parallelized to improve performance, though this may not be possible on all hardware configurations. The important point to note from these results is that, when using STTs to approximate the local dynamics, the computational requirements decrease significantly. In addition, the time required to evaluate the reference STTs will not increase as additional perturbations are added to the dynamics, whereas the time required for numerical integration generally will. Thus, for more complex dynamics, such as a full-ephemeris model of the Earth-Moon system, the performance improvements may become even more notable.

5.5 DRO to NRHO Transfer in the Earth-Moon System

We next apply the STT/DDP formulation to a continuous-thrust transfer from a distant retrograde orbit (DRO) to a near-rectilinear halo orbit (NRHO) in the Earth-Moon circular restricted three-body problem (CR3BP). Earth-Moon NRHOs are currently of very high interest for the civilian and military space communities, as it is the planned operating location for NASA's

Table 5.4: DRO-to-NRHO transfer orbit scenario parameters

Orbit	x	y	z	$\dot{x}$	$\dot{y}$	$\dot{z}$	C
Initial DRO	0.983368093	-0.259208967	0.0	-0.351341295	-0.008333464	0.0	2.925
Target NRHO	1.021968177	0.0	-0.18206	0.0	-0.103140143	0.0	3.047

Lunar Gateway [115]. On the other hand, DROs, which are planar in the CR3BP, have been shown have favorable long-term stability properties [11], and could be desirable for missions requiring very low station-keeping costs. Efficient transfers between the two families of orbits may be necessary in the near-future. The state data for the reference NRHO and DRO used in this example is given in Table 5.4. The state data for the DRO was obtained from Ref. [94]. The target state for the NRHO is selected to lie at apolune. A reference trajectory was optimized using the numerical DDP algorithm and minimum-energy cost function, with 100 stages and a transfer time of 2.45 non-dimensional CR3BP units, corresponding to roughly 10.6 days. This reference transfer is shown in Fig. 5.7. The higher-order STTs of this reference trajectory (up to order $m = 4$) were then integrated separately for each stage and stored. For this case study, we set $\sigma = 10^4$, $\epsilon_{opt} = 10^{-10}$, and $\epsilon_{feas} = 5 \times 10^{-7}$ for both the numerical and STT/DDP algorithms, and used $W = 0.4$ for the penalty term to enforce accuracy of the STT method.

For the DRO-to-NRHO transfer example, we will demonstrate how the STT/DDP algorithm can be used as a "guidance" scheme, in order to steer a spacecraft with a perturbed initial state back towards the reference target state. Recall that the last iteration of the numerical DDP algorithm results in an optimal linear feedback law that can also be used as a guidance scheme. We will compare the accuracy regions for the linear feedback law and the re-optimized trajectories computed using the STT/DDP algorithm.

The initial DRO state was perturbed by selecting a range of fifty alternative departure points along the DRO, by changing the initial phasing on the orbit between $\delta\tau \in [-0.4, 0.6]$ non-dimensional CR3BP time units. These alternative departure points could be operationally desirable in order to achieve the correct phasing between the departure and arrival orbits, if for example a

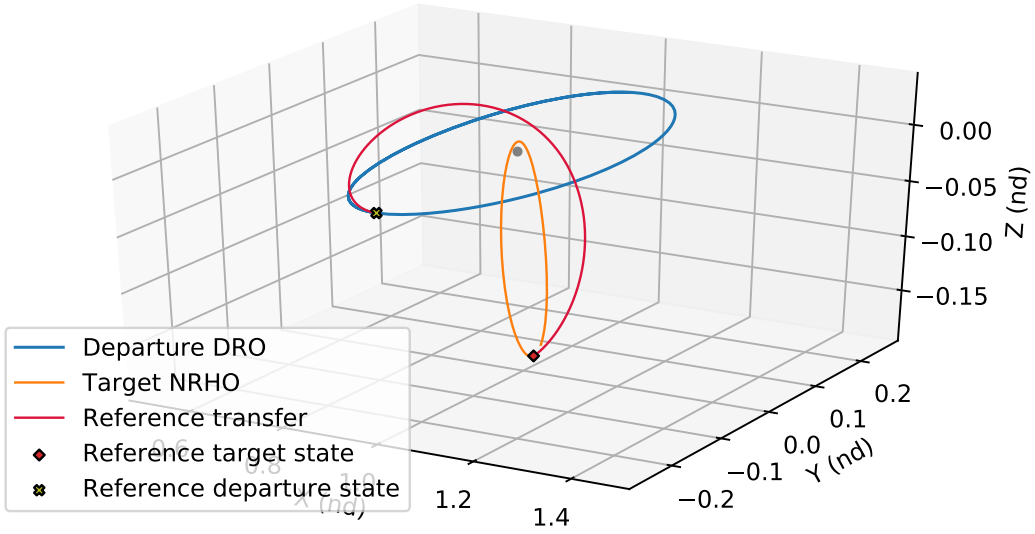

Figure 5.7: Reference DRO to NRHO transfer in Earth-Moon CR3BP

Table 5.5: Average STT/DDP algorithm performance at various orders for perturbed initial state

Algorithm	Computation time (s)	Number of iterations
STT/DDP, $m = 2$	1.61	17.2
STT/DDP, $m = 3$	1.92	19.2
STT/DDP, $m = 4$	2.48	19.5
Numerical DDP	44.7	16.5

spacecraft is seeking to rendezvous with another spacecraft along the NRHO. This large range of departure points was chosen to illustrate when the trajectories computed using each method diverge from the numerical DDP trajectories. New trajectories were computed for the range of perturbed initial DRO states to reach the reference target state, using the linear feedback law obtained from numerical DDP, and the STT/DDP method for orders $m \in [2, 3, 4]$. As a comparison, optimal trajectories for each departure point were also computed using the numerical DDP method. The optimized trajectories computed using the STT/DDP method with $m = 3$, along with the reference transfer and departure and target orbits, are shown in Fig. 5.8.

The total transfer costs for the range of alternative departure points for each method are shown in Figs. 5.9 and 5.10. The linear feedback law is accurate for a small region in the vicinity of the reference departure point, but rapidly loses accuracy and results in transfers with very large thrust requirements. The second and higher-order STT/DDP methods clearly result in transfers with lower thrust requirements further from the reference. As expected, as the maximum order of STT included in the approximation increases, the trajectories approach the optimal trajectories obtained through the numerical DDP. The average computational time and number of DDP iterations to compute the fifty alternative transfers is shown in Table 5.5. All orders of the STT/DDP algorithm are at least an order of magnitude faster to run than the numerical DDP algorithm.

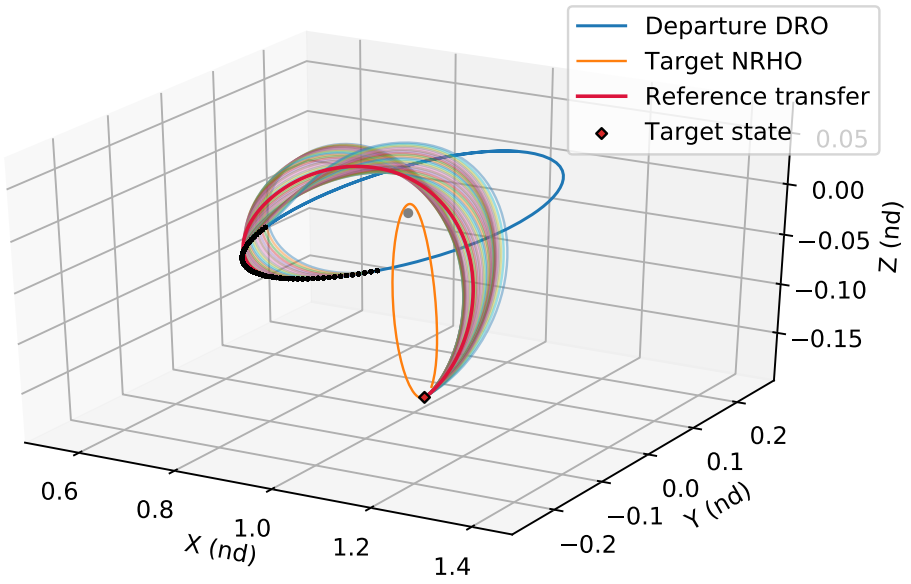

Figure 5.8: DRO to NRHO transfers with varying departure location. Trajectories generated using STT/DDP method ($m = 3$)

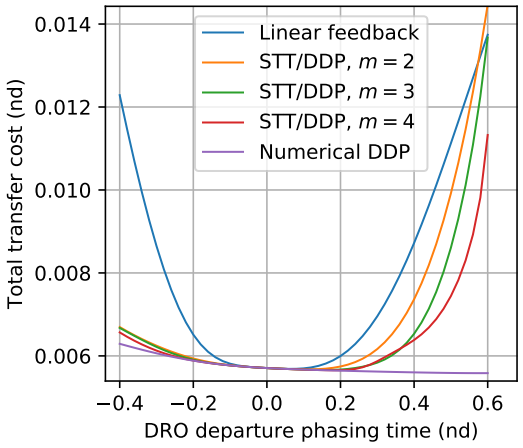

Figure 5.9: Minimum-energy transfer costs for DRO-to-NRHO transfers

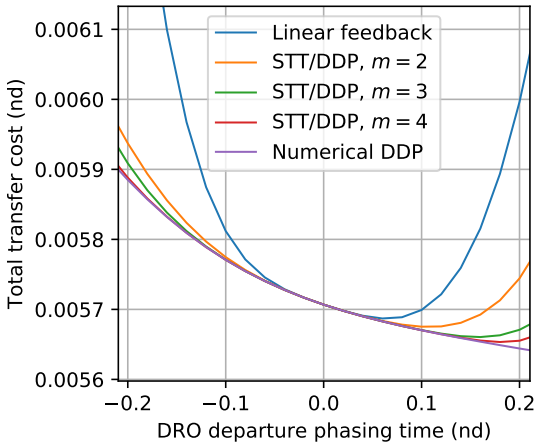

Figure 5.10: Minimum-energy transfer costs for DRO-to-NRHO transfers (zoomed view)

Table 5.6: NRHO-to-GSO transfer orbit scenario parameters

Orbit	x	y	z	$\dot{x}$	$\dot{y}$	$\dot{z}$
Initial NRHO	1.021968177	0.0	-0.18206	0.0	-0.103140143	0.0
Target GSO	0.080981568	0.0	0.0	0.0	3.113042895	0.0

5.6 NRHO to Geosynchronous Orbit Transfer in the Earth-Moon System

Next, we apply the STT/DDP method to a transfer from an NRHO to a geosynchronous orbit (GSO) around Earth. Since geosynchronous and geostationary orbits are currently in high use for a number of applications, efficient methods to compute transfers between NRHOs and these orbits are likely to be of interest in the near-future. Entering into geosynchronous orbit requires significant thrusting capability, resulting in highly sensitive and nonlinear trajectories. The precise geometry of the target geosynchronous orbits may not be exactly known **a priori** when designing a reference trajectory; thus, a method to rapidly generate transfers targeting a variety of orbital configurations could enable more flexible mission designs.

Again, a reference transfer was generated using the numerical DDP algorithm with the minimum-energy cost function from Eq. 5.31, with a transfer time of 1.32 non-dimensional CR3BP time units, corresponding to roughly 5.7 days. This trajectory was segmented into 200 equally spaced stages - a larger number of stages is required for this transfer due to the high sensitivity of the geosynchronous orbit insertion portion of the transfer. The target reference geosynchronous orbit configuration was arbitrarily chosen to lie in the Earth-Moon plane. The initial and target orbit state parameters are given in Table 5.6. The higher-order STTs along this reference (up to order $m = 4$) were integrated separately for each stage and stored. We will investigate using these STTs to rapidly compute new trajectories around the reference.

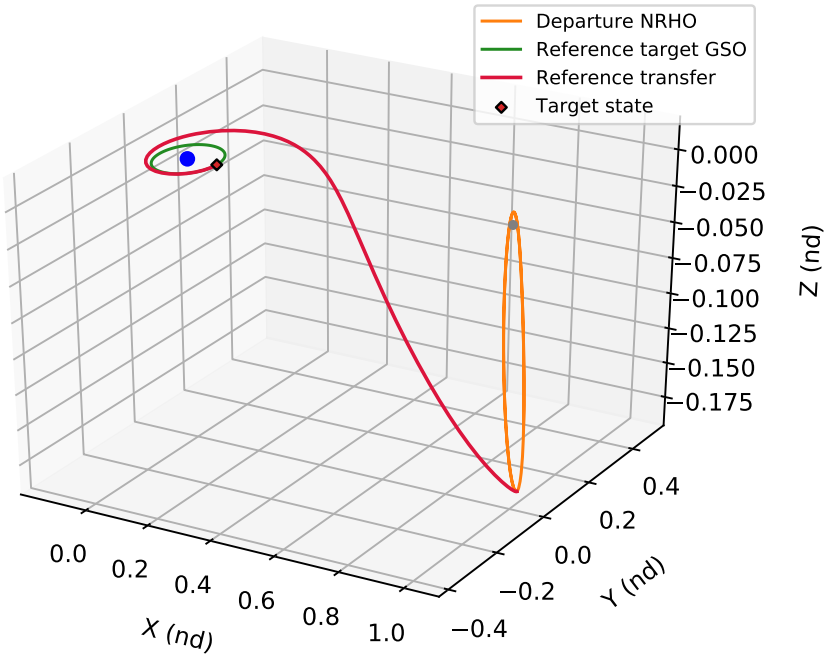

Figure 5.11: Reference NRHO to GSO transfer in Earth-Moon CR3BP

Table 5.7: STT/DDP algorithm performance at various orders m for targeting new geosynchronous orbit

Algorithm	Final cost J	Computation time (s)	Number of iterations	Final state error $\|\boldsymbol{\psi}\|$ with feedback law
STT/DDP, $m = 2$	0.139014	4.96	30.9	1.35e-3
STT/DDP, $m = 3$	0.069420	6.46	32.4	2.58e-4
STT/DDP, $m = 4$	0.065928	8.14	31.8	4.45e-5
Numerical DDP	0.065789	291.04	32.4	4.47e-13

5.6.1 Varying target parameters

First, we will investigate using the STT/DDP method to target a new GSO configuration from the same initial NRHO state at apolune. As with the example in Section 5.4, the linear feedback law from the numerical DDP method cannot be used here because the target state is different from the reference target state. However, the STT/DDP method can easily be applied to this scenario.

Fifty different new target geosynchronous orbits were selected, with the inclination varying from 0 to 15° relative to the Earth-Moon plane. The right ascension of the ascending node (RAAN) was allowed to vary between 0 and 360°, again relative to the Earth-Moon plane. The STT/DDP methods at orders $m \in [2, 3, 4]$ were run to generate new optimal transfers to reach these new targets. The numerical DDP method was also run as a comparison. The fifty transfers are illustrated in Fig. 5.12, for the $m = 4$ method. The thrust profiles for each of the fifty transfers are shown in Figs. 5.13 and 5.14. The resulting average cost, computational time, and final state errors are shown in Table 5.7.

As there is a less than 0.25% difference between the average transfer cost when using the fourth-order STT/DDP method and numerical DDP method, we can conclude that the STT/DDP method produces transfers that are an accurate approximation of the optimal transfers. The STT/DDP method can therefore be used to run large-scale analyses that would be expensive to run with a standard numerical method. Using the same range of inclination and RAAN parameters

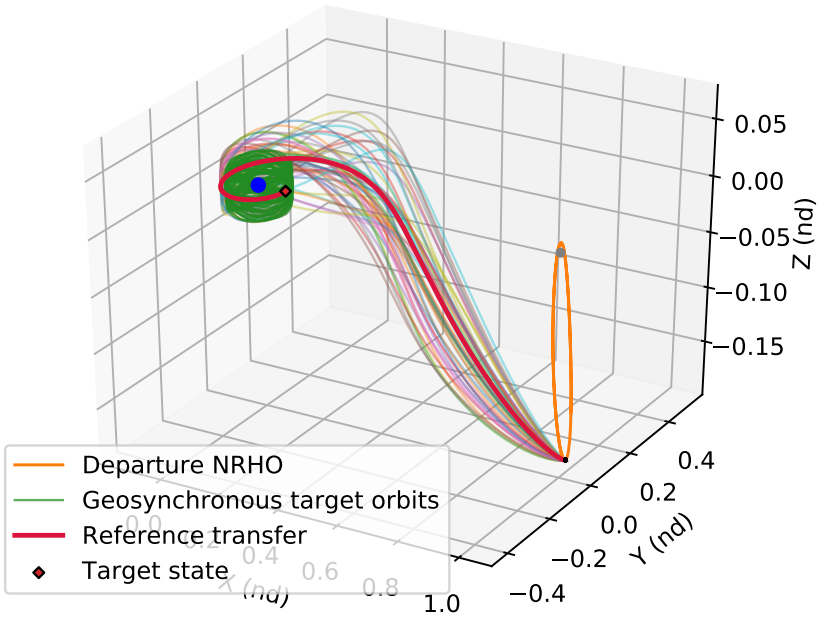

Figure 5.12: Transfers from NRHO to GSOs with varying parameters in Earth-Moon CR3BP

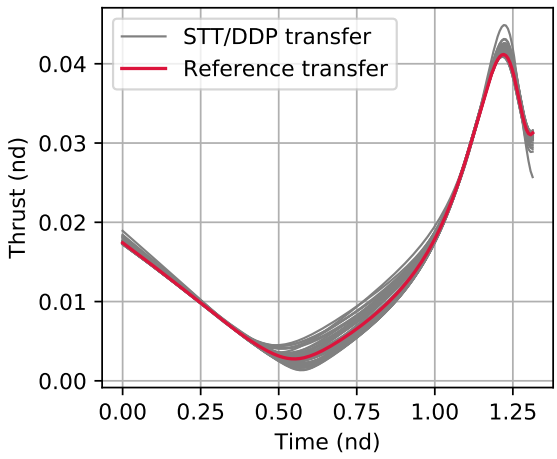

Figure 5.13: Thrust magnitudes for various NRHO to GSO transfers, generated using STT/DDP method with $m = 4$

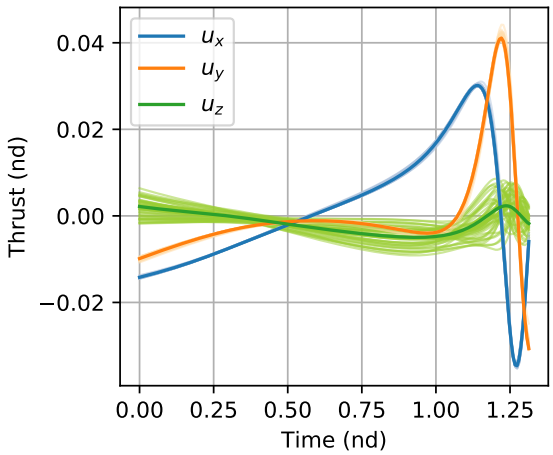

Figure 5.14: Thrust profiles for various NRHO to GSO transfers, generated using STT/DDP method with $m = 4$. Reference thrust profile in bold.

as detailed above, the STT/DDP method was used to optimize trajectories for 1000 different target conditions. These parameters, and the resulting cost for each combination of parameters, were used to generate the contour plot shown in Fig. 5.15. This type of large-scale tradeoff analysis can allow a mission designer to rapidly identify reachable orbital configurations, and was completed around 30× faster when using the STT/DDP method instead of the numerical DDP method. For reference, optimizing these 1000 trajectories took roughly two hours to run in serial using Python code and a standard laptop computer. The equivalent numerical DDP code would have taken over three days to run.

5.6.2 Transfer from different initial conditions

In order to demonstrate the flexibility of the proposed algorithm, the NRHO-to-GSO transfer scenario was also run with varying initial conditions. The phasing of the departure location along the NRHO was allowed to vary between $\delta\tau \in [-0.7, 0.5]$, in non-dimensional CR3BP units. The same range of target GSO parameters (inclination and RAAN) as in the previous section was used. The STT/DDP method ($m = 4$) was used to generate fifty new transfers with these characteristics. It was successfully able to optimize new trajectories for all fifty scenarios. These are illustrated in Fig. 5.16

5.7 Transfers Using a Different Cost Function

The STT/DDP method can also be used to optimize transfers using a different cost function than was used to generate the reference trajectory. For example, for all previous transfers computed in this chapter, the minimum-energy cost (Eq. 5.31) was employed. This form of the cost will result in thrust profiles with the thrust spread relatively evenly over each stage. However, this form of the cost will not result in fuel-optimal trajectories, and is therefore not typically used in the design of optimal trajectories. Instead, the minimum-fuel form of the control (Eq. 5.32) is most often used. This form of the cost function results in bang-bang thrust profiles, with the control switching on and off over the course of the transfer. The minimum-energy form of the cost results in a

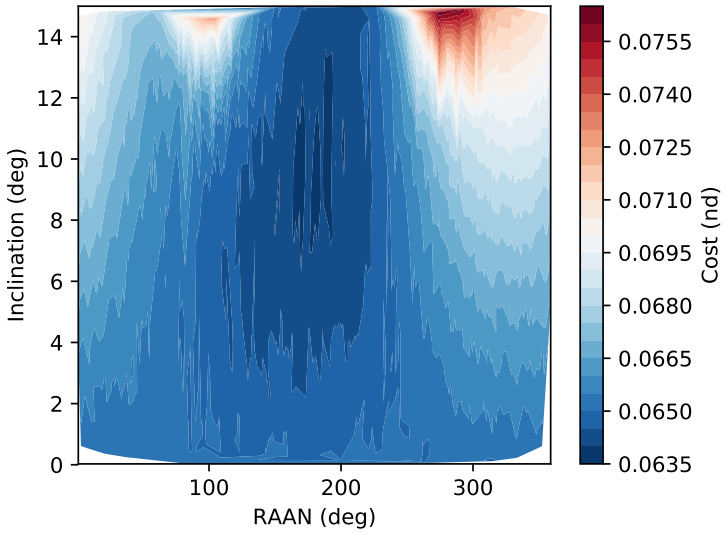

Figure 5.15: Cost to transfer from NRHO to 1000 different GSO configurations, generated using STT/DDP with $m = 4$

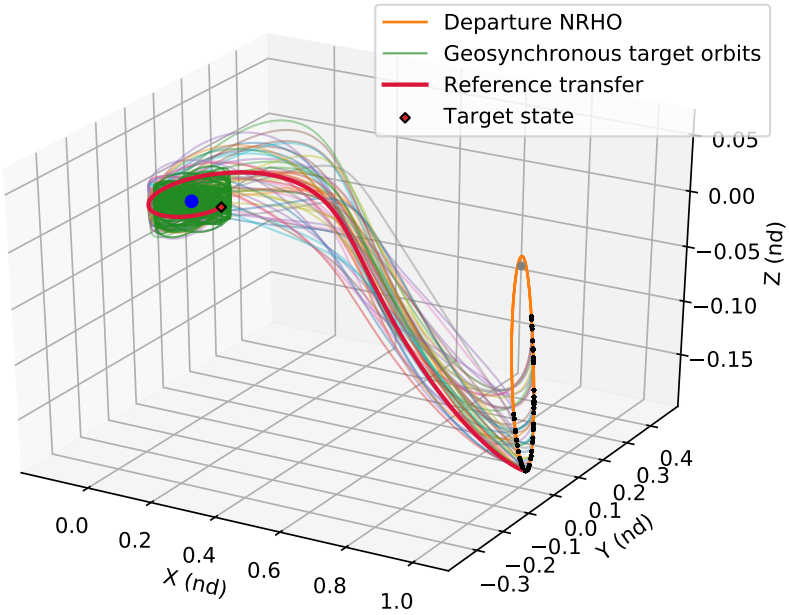

Figure 5.16: Low-thrust transfers from varying initial points along NRHO to various target GSOs, generated using STT/DDP ($m = 4$)

more numerically stable optimization problem which requires fewer iterations to solve. Thus, an additional potential use for the STT/DDP method is to design a reference trajectory using an "easier" cost function (which can be solved with a small number of iterations). Then, the higher-order STTs of this reference can be integrated, and the STT/DDP method can be used to rapidly compute an optimal trajectory using a different cost function (e.g. minimum-fuel), which may require more iterations to converge on a good solution. This could also be useful for operational situations to rapidly re-optimize a trajectory if priorities shift during the course of a mission.

In order to illustrate this strategy, the STT/DDP algorithm was run using the same reference STTs for the DRO-to-NRHO transfer as in Section 5.5, but using the minimum-fuel cost function from Eq. 5.32, with $\epsilon_{ml} = 1 \times 10^{-5}$. The target state was kept the same as the reference transfer, and the initial state was allowed to vary along the departure DRO. The thrust magnitude was constrained to be less than 0.015 nondimensional units. The tolerance and weighting parameters were set to $\epsilon_{opt} = 1 \times 10^{-6}$, $\epsilon_{feas} = 1 \times 10^{-6}$, $\sigma = 1 \times 10^{5}$, and $W = 1 \times 10^{3}$. The STT/DDP method was successfully able to compute optimal minimum-fuel trajectories for several different departure points along the reference DRO, using only the reference STTs from the minimum-energy transfer. The resulting orbit and thrust profiles are shown in Figs. 5.17 and 5.18; the bang-bang thrust profiles of the minimum-fuel trajectories are evident.

5.8 Conclusions

This chapter presents an algorithm for rapid local trajectory optimization around a reference. The method relies on the higher-order state transition tensors (STTs) of a reference trajectory to approximate the dynamics and derivatives of perturbed trajectories around the reference. This method is used to run an approximation of differential dynamic programming (DDP), a commonly used optimization method for controlled nonlinear dynamical systems. The algorithm, referred to as STT/DDP, is applied to several complex spacecraft transfers in the Earth-Moon circular restricted three-body problem (CR3BP). Results show that the STT/DDP algorithm yields similar results to a numerical DDP algorithm when computing new trajectories in the vicinity of the reference, but at

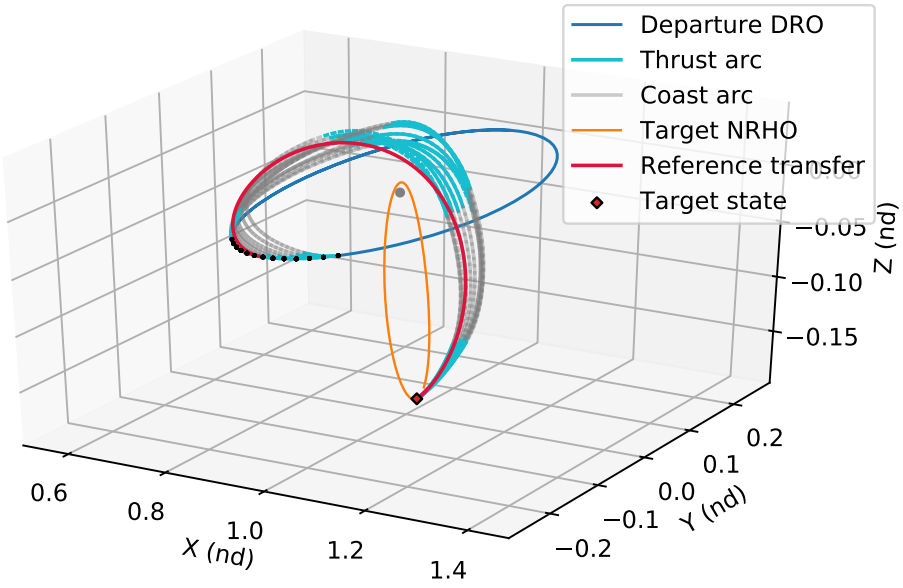

Figure 5.17: DRO-to-NRHO transfers using minimum-fuel cost function, computed using STT/DDP algorithm with $m = 4$. Turquoise arcs indicate thrust arcs.

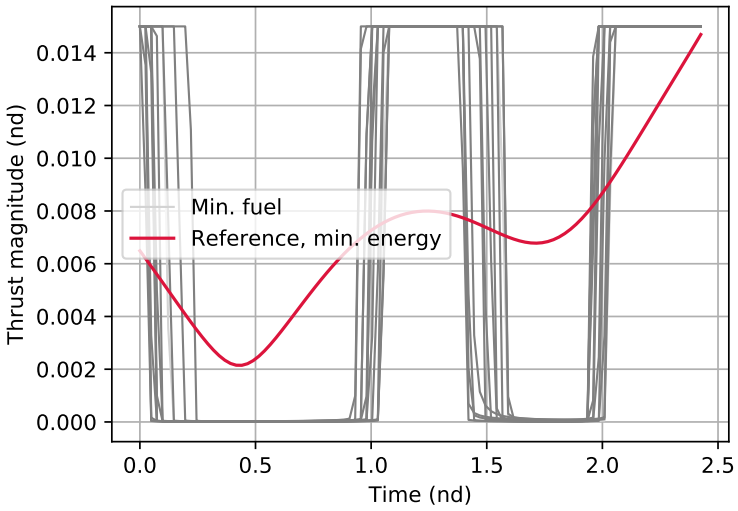

Figure 5.18: Thrust magnitude over time for transfers using minimum-fuel cost function, computed using STT/DDP algorithm with $m = 4$

a fraction of the computational cost. The method can accommodate any number of perturbations in the dynamics, can incorporate most stage or penalty constraints that are used in a standard DDP algorithm, and can handle different cost functions or stage constraints than were used to generate the reference.

In an operational setting, one could integrate the higher-order STTs of a reference trajectory prior to mission execution, and use the STT/DDP algorithm to rapidly recompute near-optimal controls in response to changes in the initial or target states. The algorithm is particularly suitable to cislunar applications due to the highly nonlinear nature of orbits in the Earth-Moon regime, and the relatively short timescales which can present limitations for the traditional full-fidelity ground-based maneuver planning workflow. The capability to incorporate different cost functions or constraints from the reference could be used to enable a more agile or flexible trajectory design process for these types of missions. The proposed algorithm could also be beneficial for use on-board CubeSats with limited computational capabilities and lean flight dynamics operations teams. It could additionally be used to rapidly conduct large-scale tradeoff analyses for missions with variable departure, arrival and constraint conditions.

Chapter 6

Stochastic Maneuver Design with State Transition Tensors

Given the significant upfront cost for any spacecraft mission, guaranteeing mission safety under the presence of uncertainty is critical. Missions operating in highly nonlinear dynamic systems are particularly sensitive to any navigation and maneuver execution errors. Existing nonlinear stochastic control schemes, such as stochastic model predictive control (SMPC) [76] are often prohibitively expensive to execute on flight hardware without making any assumptions or approximations. As such, there is interest in developing efficient stochastic control algorithms that are suitable to use on-board a spacecraft or for real-time applications.

An important component of a spacecraft maneuver design scheme is the concept of **statistical maneuvers**, which are included after large trajectory correction maneuvers to correct for maneuver execution and navigation errors. In a reference trajectory, these maneuvers will generally have a ΔV of zero; however, since the mission will never perfectly conform to its baseline trajectory, these maneuvers are required to ensure mission success. Statistical ΔV maneuvers are generally scheduled to occur at pre-determined times. In this case, the STTs of the reference trajectory could be integrated between the pre-defined maneuver times, and could be stored for later use in real time or on-board the spacecraft. Subsequently, prior to maneuver execution, the STTs could be used to predict the effect of a correction maneuver on the final state while considering the effects of the nonlinear dynamics on the state uncertainty propagation. For the types of complex transfers that are being proposed for upcoming missions, the timescale between these maneuvers may be on the order of days or hours, which may limit the ability for ground-based operations teams to plan these

maneuvers and upload the commands. A real-time or on-board maneuver planning capability that can take into account the effects of uncertainty could reduce the risk and operations requirements for future spacecraft missions.

Several guidance law formulations which take into account state uncertainty information have been specifically developed for spacecraft systems [62, 82]. These formulations typically rely on linearized approximations of the nonlinear dynamics, which may be insufficiently accurate for spacecraft operating in chaotic orbital regimes, such as in cislunar space. In Chapters 3 and 4, efficient spacecraft guidance and maneuver design algorithms were developed using STTs to approximate the nonlinear spacecraft dynamics. However, since these methods were developed for deterministic nonlinear systems with the assumption that perfect state knowledge is available, they are not designed to be robust to operational uncertainties. This motivates the problem considered in this chapter, which is to extend the previously developed algorithms to account for state uncertainties. To this end, analytic equations for a variety of different formulations to incorporate state uncertainty are derived as a function of a reference trajectory's STTs.

As stated in the thesis introduction, the term "stochastic" can refer to several different formulations for incorporating uncertainty in the context of spacecraft control. For example, it can refer to the problem of targeting the mean of a target state distribution [92, 86], to chance-constrained control [84, 82], or to the problem of minimizing the state uncertainty in a system at a given final time [62, 63]. In this chapter, we will show that our nonlinear approach using higher-order STT approximations of the dynamics can efficiently yield accurate solutions to each of these formulations for considering uncertainty. This could enable flexible maneuver design algorithms that can adapt to changing mission priorities or requirements. An important benefit for this strategy when compared to other methods for "frontloading" computations, such as machine learning-based algorithms [15, 17], is that the cost function and constraints can differ from the reference trajectory cost function and constraints.

This chapter is organized as follows. First, we present the theory behind nonlinear uncertainty propagation using STTs. Following this, we develop analytic formulations for several cost function

and constraint formulations strictly as a function of the STTs. Finally, a numerical example is presented for a spacecraft executing a complex multi-impulse transfer in the Earth-Moon circular restricted three-body problem.

6.1 Dynamics model

In this chapter we will consider a discrete-time nonlinear time varying system with impulsive controls. Let $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ be the state vector at time t_k, $k = 0, 1, ..., N-1$. The system can be expressed as:

$$\boldsymbol{x}_{k+1} = \phi(\boldsymbol{x}_k + \beta \boldsymbol{u}_k) \tag{6.1}$$

where ϕ represents the nonlinear solution flow of the system, $\boldsymbol{u}_k$ is the impulsive control applied at time t_k, and the matrix $\beta \in \mathbb{R}^{n_x \times n_u}$ is the mapping between the impulsive control and state vectors.

For a generic nonlinear system, iterative direct optimization algorithms will generally be required to find optimal solutions that minimize some cost function while satisfying constraints. This can require repeated integrations of the dynamics. For a spacecraft system, this is often prohibitively expensive to compute using on-board processors. In Chapters 3 and 4, a computationally efficient spacecraft maneuver design scheme was developed using higher-order STTs, which will form the basis for the work in this chapter.

When using STTs to approximate the dynamics around a reference trajectory, the controlled system from Eq. 6.1 becomes:

$$\hat{x}^i_{k+1} + \delta\check{x}^i_{k+1} = \phi^i(\hat{\boldsymbol{x}}_k + \beta\hat{\boldsymbol{u}}_k) + \sum_{p=1}^{m} \frac{1}{p!} \phi^{i,\gamma_1 ... \gamma_p}_{(t_{k+1},t_k)} \delta x^{\gamma_1}_k ... \delta x^{\gamma_p}_k \tag{6.2}$$

$$\delta x^i_k = \delta\check{x}^i_k + \beta^{i,j}\delta u^j_k \tag{6.3}$$

where $\delta\check{\boldsymbol{x}}_k$ is simply the deviation from the reference state $\hat{\boldsymbol{x}}_k$ before the control update $\delta\boldsymbol{u}_k$ (with respect to the reference control $\hat{\boldsymbol{u}}_k$) is applied. Using this notation, the effects of the reference controls are incorporated into the reference STTs. Since the reference trajectory dynamics do not

vary, the above system can be rewritten as:

$$\delta\check{x}^i_{k+1} = \sum_{p=1}^{m} \frac{1}{p!} \phi^{i,\gamma_1 \ldots \gamma_p}_{(t_{k+1},t_k)} \delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p}$$

$$\delta x^i_k = \delta\check{x}^i_k + \beta^{i,j} \delta u^j_k \tag{6.4}$$

Using Eqs. 2.23 and 2.25, the first and second-order derivatives of $\delta\boldsymbol{x}_{k+1}$ with respect to the state deviation $\delta\boldsymbol{x}_k$ at time t_k, can be expressed analytically solely as a function of the reference STTs mapping from t_k to t_{k+1}. Using the chain rule, the derivatives of $\delta\boldsymbol{x}_{k+1}$ with respect to the control update $\delta\boldsymbol{u}_k$ at time t_k are also straightforward to obtain analytically as a function of the reference STTs. This property means that exact derivative information can be obtained at virtually no cost. This property can be extended to form expressions for the third and higher-order derivatives of $\delta\boldsymbol{x}_{k+1}$.

6.2 Nonlinear Uncertainty Propagation with STTs

In previous chapters we have assumed the spacecraft state to be deterministic, neglecting any uncertainty in the spacecraft's knowledge of its own state. We will now relax this assumption, and consider the spacecraft state vector at time t_k to be a Gaussian random vector $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{m}_k, P_k)$, where $\boldsymbol{m}_k$ is the mean state vector and P_k is the covariance matrix at time t_k.

6.2.1 Mean propagation

STTs can be used to analytically propagate a spacecraft's mean state vector through nonlinear dynamics to time t_{k+1}, following Ref. [93]:

$$\delta\check{m}^i_{k+1} = \sum_{p=1}^{m} \frac{1}{p!} \phi^{i,\gamma_1 \ldots \gamma_p}_{(t_k,t_{k+1})} \mathbb{E}\left[\delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p}\right] \tag{6.5}$$

where $\mathbb{E}\left[\delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p}\right]$ corresponds to the p-th order moment. Assuming that $\delta\boldsymbol{x}_k$ is a Gaussian random vector, then these higher-order moments can be expressed as functions of $\delta\boldsymbol{m}_k$ and P_k.

The first four moments of a Gaussian vector are given by [91]

$$\mathbb{E}\left[\delta x_k^a\right] = \delta m_k^a \tag{6.6}$$

$$\mathbb{E}\left[\delta x_k^a \delta x_k^b\right] = \delta m_k^a \delta m_k^b + P_k^{ab} \tag{6.7}$$

$$\mathbb{E}\left[\delta x_k^a \delta x_k^b \delta x_k^c\right] = \delta m_k^a \delta m_k^b \delta m_k^c + \delta m_k^a P^{bc} + \delta m_k^b P^{ac} + \delta m_k^c P_k^{ab} \tag{6.8}$$

$$\begin{aligned}\mathbb{E}\left[\delta x^a \delta x^b \delta x^c \delta x^d\right] = {} & \delta m^a \delta m^b \delta m^c \delta m^d + \delta m^a \delta m^b P_k^{cd} + \delta m^a \delta m^c P_k^{bd} + \delta m^b \delta m^c P_k^{ad} + \\ & \delta m^a \delta m^d P_k^{bc} + \delta m^b \delta m^d P_k^{ac} + \delta m^c \delta m^d P_k^{ab} + P_k^{ab} P_k^{cd} + P_k^{ac} P_k^{bd} + P_k^{ad} P_k^{bc}\end{aligned} \tag{6.9}$$

If the deviation from the reference is zero, then $\delta \boldsymbol{m}_k = 0$, which greatly simplifies the above equations, causing all odd moments to vanish. In the case, the first four moments become:

$$\mathbb{E}\left[\delta x_k^a\right] = 0 \tag{6.10}$$

$$\mathbb{E}\left[\delta x_k^a \delta x_k^b\right] = P_k^{ab} \tag{6.11}$$

$$\mathbb{E}\left[\delta x_k^a \delta x_k^b \delta x_k^c\right] = 0 \tag{6.12}$$

$$\mathbb{E}\left[\delta x^a \delta x^b \delta x^c \delta x^d\right] = P_k^{ab} P_k^{cd} + P_k^{ac} P_k^{bd} + P_k^{ad} P_k^{bc} \tag{6.13}$$

Similar to Eqs. 2.23 and 2.25, the first and second-order derivatives of the STT-propagated mean deviation (Eq. 6.5), with respect to the initial mean deviation from a reference, can be expressed analytically solely as a function of the reference STTs:

$$\frac{\partial(\delta \check{m}_{k+1}^i)}{\partial(\delta m_k^{\gamma_1})} = \phi_{(t_{k+1},t_k)}^{i,\gamma_1} + \sum_{p=2}^{m} \frac{1}{(p-1)!} \phi_{(t_{k+1},t_k)}^{i,\gamma_1\gamma_2\ldots\gamma_p} \mathbb{E}\left[\delta x_k^{\gamma_2} \ldots \delta x_k^{\gamma_p}\right] \tag{6.14}$$

$$\frac{\partial^2(\delta \check{m}_{k+1}^i)}{\partial(\delta m_k^{\gamma_1})\partial(\delta m_k^{\gamma_2})} = \phi_{(t_{k+1},t_k)}^{i,\gamma_1\gamma_2} + \sum_{p=3}^{m} \frac{1}{(p-2)!} \phi_{(t_{k+1},t_k)}^{i,\gamma_1\gamma_2\gamma_3\ldots\gamma_p} \mathbb{E}\left[\delta x_k^{\gamma_3} \ldots \delta x_k^{\gamma_p}\right] \tag{6.15}$$

6.2.2 Covariance propagation

The spacecraft's state covariance can also be analytically propagated through the nonlinear dynamics, following Ref. [92]:

$$P_{k+1}^{ij} = \Big(\sum_{p=1}^{m} \sum_{q=1}^{m} \frac{1}{p!q!} \phi_{(t_{k+1},t_k)}^{i,\gamma_1\ldots\gamma_p} \phi_{(t_{k+1},t_k)}^{j,\eta_1\ldots\eta_q} \times \mathbb{E}\left[\delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p} \delta x_k^{\eta_1} \ldots \delta x_k^{\eta_p}\right] \Big) - \delta \check{m}_{k+1}^i \delta \check{m}_{k+1}^j \tag{6.16}$$

Substituting Eq. 6.5 into Eq. 6.16 gives the following alternative expression:

$$P_{k+1}^{ij} = \sum_{p=1}^{m}\sum_{q=1}^{m} \frac{1}{p!q!} \phi_{(t_{k+1},t_k)}^{i,\gamma_1 \ldots \gamma_p} \phi_{(t_{k+1},t_k)}^{j,\eta_1 \ldots \eta_q} \times \left(\mathbb{E}\left[\delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p} \delta x_k^{\eta_1} \ldots \delta x_k^{\eta_p}\right] - \mathbb{E}\left[\delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p}\right] \mathbb{E}\left[\delta x_k^{\eta_1} \ldots \delta x_k^{\eta_q}\right] \right) \tag{6.17}$$

Note that both Eqs. 6.16 and 6.17 require computing the moments of the initial state distribution up to order $2m$. As can be seen from Eqs. 6.6 - 6.9, the number of terms required to compute the higher-order moments increases exponentially as the order increases. These computations become prohibitively expensive at higher orders and outweigh the benefits of using higher-order methods over sampling-based methods. It is therefore beneficial to derive an approximation of Eq. 6.16 that truncates any negligible terms that are expensive to compute.

6.2.3 Tractable covariance propagation

Upon examining the equations for the moments of a Gaussian vector (Eqs. 6.6-6.9), we can see that the expressions contain various permutations and orders of $\delta \boldsymbol{m}_k$ and P_k, which correspond to the state deviation from the reference trajectory at time t_k, and the state covariance at time t_k, respectively. To derive the truncated covariance expression, consider that the full expression in Eq. 6.16 is a convergent series, where for example, terms of $\mathcal{O}(\delta \boldsymbol{m}_k) \gg \mathcal{O}(\delta \boldsymbol{m}_k \delta \boldsymbol{m}_k)$. Let the vector $\boldsymbol{\sigma}_k$ refer to the 1-σ error component vector in each direction (i.e. $\sigma_k^i = \sqrt{P_k^{ii}}$). The main assumption in deriving this covariance propagation approximation is that the error in the spacecraft state estimate will be significantly smaller than the state deviation from the reference trajectory. This will generally be the case for a spacecraft system. This implies that $\mathcal{O}(\delta \boldsymbol{m}_k) \gg \mathcal{O}(\boldsymbol{\sigma}_k)$, meaning that, for example, $\mathcal{O}(\delta \boldsymbol{m}_k \delta \boldsymbol{m}_k) \gg \mathcal{O}(P_k)$, or $\mathcal{O}(\delta \boldsymbol{m}_k \delta \boldsymbol{m}_k \delta \boldsymbol{m}_k \delta \boldsymbol{m}_k) \gg \mathcal{O}(P_k P_k)$, etc. As such, terms of $\mathcal{O}(P_k P_k)$ and smaller can be neglected when compared with the equivalent-order terms of $\mathcal{O}(\delta \boldsymbol{m}_k \delta \boldsymbol{m}_k \delta \boldsymbol{m}_k \delta \boldsymbol{m}_k)$ or smaller.

It is therefore helpful to group terms from Eq. 6.17 of equivalent orders of P_k. To do so, Eq. 6.17 can be applied to the STT-approximated dynamic system in Eqs. 6.4. From Eqs. 2.23

and 2.25, we can define the STM and higher-order STTs of this system as

$$\theta^{i,\gamma_1}_{(t_{k+1},t_k)} = \frac{\partial(\delta x^i_{k+1})}{\partial(\delta x^{\gamma_1}_k)} = \phi^{i,\gamma_1}_{(t_{k+1},t_k)} + \sum_{p=2}^{m} \frac{1}{(p-1)!}\phi^{i,\gamma_1\gamma_2\ldots\gamma_p}_{(t_{k+1},t_k)}\delta x^{\gamma_2}_k \ldots \delta x^{\gamma_p}_k \tag{6.18}$$

$$\theta^{i,\gamma_1\gamma_2}_{(t_{k+1},t_k)} = \frac{\partial^2(\delta x^i_{k+1})}{\partial(\delta x^{\gamma_1}_k)(\delta x^{\gamma_2}_k)} = \phi^{i,\gamma_1\gamma_2}_{(t_{k+1},t_k)} + \sum_{p=3}^{m} \frac{1}{(p-2)!}\phi^{i,\gamma_1\gamma_2\gamma_3\ldots\gamma_p}_{(t_{k+1},t_k)}\delta x^{\gamma_3}_k \ldots \delta x^{\gamma_p}_k \tag{6.19}$$

In order to derive this approximation, consider now a zero-mean deviation $\delta\boldsymbol{z}_k$ from $\delta\boldsymbol{x}_k$, where $\delta\boldsymbol{z}_k \sim \mathcal{N}(0, P_k)$. This represents the state uncertainty for a state estimate with deviation $\delta\boldsymbol{x}_k$ from the reference trajectory. The covariance propagation for this system about $\delta\boldsymbol{z}_k$ can be rewritten as

$$P^{ij}_{k+1} = \sum_{p=1}^{m}\sum_{q=1}^{m} \frac{1}{p!q!}\theta^{i,\gamma_1\ldots\gamma_p}_{(t_{k+1},t_k)}\theta^{j,\eta_1\ldots\eta_q}_{(t_{k+1},t_k)} \times \left(\mathbb{E}\left[\delta z^{\gamma_1}_k \ldots \delta z^{\gamma_p}_k \delta z^{\eta_1}_k \ldots \delta z^{\eta_p}_k\right] - \mathbb{E}\left[\delta x^{\gamma_1}_k \ldots \delta x^{\gamma_p}_k\right]\mathbb{E}\left[\delta x^{\eta_1}_k \ldots \delta x^{\eta_q}_k\right]\right) \tag{6.20}$$

Because $\mathbb{E}[\delta\boldsymbol{z}_k] = 0$, we can use the simplified moment equations from Eqs. 6.10- 6.13 to reduce the above equation. For example, for $m = 2$, it becomes

$$P^{ij}_{k+1} = \theta^{i,\gamma_1}_{(t_{k+1},t_k)}\theta^{j,\eta_1}_{(t_{k+1},t_k)}P^{\gamma_1\eta_1}_k + \frac{1}{2}\theta^{i,\gamma_1\gamma_2}_{(t_{k+1},t_k)}\theta^{j,\eta_1\eta_2}_{(t_{k+1},t_k)}P^{\gamma_1\eta_1}_k P^{\gamma_2\eta_2}_k \tag{6.21}$$

For $m > 2$, we can see that all additional terms that emerge will be of $\mathcal{O}(P_k P_k)$ or smaller. Thus, for $m > 2$, Eq. 6.20 can be truncated to

$$P^{ij}_{k+1} = \theta^{i,\gamma_1}_{(t_{k+1},t_k)}\theta^{j,\eta_1}_{(t_{k+1},t_k)}P^{\gamma_1\eta_1}_k \tag{6.22}$$

with approximation errors with respect to Eq. 6.16 of $\mathcal{O}(P_k P_k)$.

It is interesting to note that since $\theta^{i,\gamma_1}_{(t_{k+1})}$ corresponds to the 1st-order STM for the STT-approximated system defined by Eq. 6.4, Eq. 6.22 corresponds to the linear covariance propagation equation $P_{k+1} = \phi_{(t_{k+1},t_k)}P_k\phi^T_{(t_{k+1},t_k)}$ for this system. Though derived differently, this is analogous to the procedure described in Ref. [44].

Eq. 6.22 is significantly cheaper to compute than Eq. 6.16. When incorporating the effects of propagating uncertainty through a nonlinear system, the system from Eq. 6.4 can be reformulated

as:

$$\delta\check{m}_{k+1}^{i} = \sum_{p=1}^{m} \frac{1}{p!} \phi_{(t_k,t_{k+1})}^{i,\gamma_1\ldots\gamma_p} \mathbb{E}\left[\delta x_k^{\gamma_1} \ldots \delta x_k^{\gamma_p}\right] \tag{6.23}$$

$$\delta m_k^i = \delta\check{m}_k^i + \beta^{i,j}\delta u_k^j \tag{6.24}$$

$$P_{k+1}^{ij} = \theta_{(t_{k+1},t_k)}^{i,\gamma_1} P_k^{\gamma_1\eta_1} \theta_{(t_{k+1},t_k)}^{j,\eta_1} \tag{6.25}$$

The derivatives of θ^{i,γ_1} with respect to $\delta\boldsymbol{x}_k$ can be expressed analytically as a function of the reference STTs:

$$\frac{\partial(\theta_{(t_{k+1},t_k)}^{i,\gamma_1})}{\partial(\delta x_k^{\gamma_2})} = \theta_{(t_{k+1},t_k)}^{i,\gamma_1\gamma_2} = \phi^{i,\gamma_1\gamma_2} + \sum_{p=3}^{m} \frac{1}{(p-2)!} \phi^{i,\gamma_1\gamma_2\gamma_3\ldots\gamma_p} \delta x_k^{\gamma_3} \ldots \delta x_k^{\gamma_p} \tag{6.26}$$

$$\frac{\partial^2(\theta_{(t_{k+1},t_k)}^{i,\gamma_1})}{\partial(\delta x_k^{\gamma_2})\partial(\delta x_k^{\gamma_3})} = \theta_{(t_{k+1},t_k)}^{i,\gamma_1\gamma_2\gamma_3} = \begin{cases} 0 & m = 2 \\ \phi^{i,\gamma_1\gamma_2\gamma_3} + \sum_{p=4}^{m} \frac{1}{(p-3)!} \phi^{i,\gamma_1\gamma_2\gamma_3\gamma_4\ldots\gamma_p} \delta x_k^{\gamma_4} \ldots \delta x_k^{\gamma_p} & m > 2 \end{cases} \tag{6.27}$$

These derivatives are again significantly cheaper to compute than the derivatives of the full covariance propagation equation (Eq. 6.16).

6.2.4 Control-linear noise

In many situations it can be critical to consider maneuver execution errors in the maneuver design process. A common formulation to model maneuver execution errors for an impulsive maneuver is the Gates model [47], which includes fixed and proportional errors in the maneuver magnitude, and fixed and proportional errors in the maneuver direction. If we assume that a scheduled maneuver will necessarily be executed at time t_k, then we can assume that the fixed maneuver execution errors can be included in the fixed initial covariance matrix $P_{k,fixed}$. The proportional errors in the maneuver magnitude can then be approximated as being linear with respect to the control vector. The initial covariance matrix P_k can be inflated by adding this noise at time t_k, giving $P_k = P_{k,fixed} + P_{k,prop}$, where

$$P_{k,prop}^{ij} = \sigma\beta^{i,a}\delta u_k^a \delta u_k^b \beta^{j,b}\sigma = \sigma^2\beta^{i,a}\delta u_k^a \delta u_k^b \beta^{j,b} \tag{6.28}$$

where σ is the expected 1-σ proportional error in the control (i.e. if the expected 1-σ proportional maneuver execution error is 1%, then $\sigma = 0.01$). The first-order derivative of $P_{k,prop}$ with respect to the control update $\delta \boldsymbol{u}_k$ can be expressed analytically as a function of the reference STTs:

$$\frac{\partial P_k^{\gamma_1 \eta_1}}{\partial (\delta u_k)^{\gamma_2}} = P_k^{\gamma_1 \eta_1, \gamma_2} = \frac{\partial P_{k,prop}^{\gamma_1 \eta_1}}{\partial (\delta u_k)^{\gamma_2}} = 2\sigma^2 \beta^{\gamma_1, a} \delta u_k^a \beta^{\eta_1, \gamma_2} \tag{6.29}$$

6.3 Problem Formulation

From Eqs.6.14 and 6.26, we can see that the system described by Eq. 6.4 is analytically differentiable with respect to changes in the control and state parameters. As such, any constraints or cost functions formulated using $\delta \boldsymbol{x}_k$, $\delta \boldsymbol{m}_k$, or P_k are also analytically differentiable. Thus, any first or second-order optimization algorithm which requires first and/or second-order derivative information can be run within the STT-approximated dynamics without requiring any further integrations of the dynamics. Computing these derivatives is typically the most computationally expensive portion of a gradient-based optimization algorithm; thus, the optimization scheme using STTs will run significantly faster than a typical method using numerical integrations of the dynamics.

In this chapter, we will use the Sequential Least Squares Programming (SLSQP) solver available in the SciPy **minimize** function. This solver is based on the principles of sequential quadratic programming (SQP) [80]. Analytical gradient information for all problem formulations are derived in this chapter, which can be provided to the SLSQP solver to significantly speed up the optimization algorithm. It is straightforward to derive analytical second-order derivatives of the cost function and constraint formulations as well. However, since the SLSQP solver used in this work uses a quasi-Newton method to approximate the Hessian, only the first-order derivative information is needed. In order to simplify the problem, we restrict the problem to solve the control parameters at each stage as separate problems (i.e. solving for an optimal control update $\delta \boldsymbol{u}_k$ to satisfy some constraints at time t_{k+1} or t_k). This amounts to solving for each TCM separately, which is the standard workflow for spacecraft maneuver planning operations.

In the optimization scheme formulation, we will seek to minimize a cost function J while

satisfying the vectors of equality constraints $\boldsymbol{\psi}_{eq} = \mathbf{0}$ and inequality constraints $\boldsymbol{\psi}_{ineq} \leq \mathbf{0}$. The following sections will detail the various cost functions and constraint expressions that can be formulated analytically using only a reference trajectory's STTs.

6.4 Cost Functions

In this section we will outline several cost functions that can be formulated analytically using a reference trajectory's STTs.

6.4.1 Minimum-energy cost function

The minimum-energy cost function consists of the square of the control magnitude at stage k:

$$J = \frac{1}{2}\|\boldsymbol{u}_k\|^2 = \frac{1}{2}\|\hat{\boldsymbol{u}}_k + \delta\boldsymbol{u}_k\|^2 \tag{6.30}$$

The first and second-order derivatives of J with respect to $\delta\boldsymbol{u}_k$ are:

$$J_u^i = \hat{u}_k^i + \delta u_k^i \tag{6.31}$$

$$J_{uu}^{ij} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases} \tag{6.32}$$

For the examples in this chapter we will be applying our formulation to the computation of statistical maneuvers; for these types of maneuvers, the reference control magnitude is zero ($\hat{\boldsymbol{u}}_k = 0$).

6.4.2 Minimum-covariance cost function

A minimum-covariance cost function can be formulated to minimize the trace of certain components of the final covariance matrix. This can be expressed using index notation as:

$$J = \mathcal{W}^{\alpha i} P_{k+1}^{ij} \mathcal{W}^{\alpha j} \tag{6.33}$$

where $\mathcal{W}$ is a weighting matrix indicating which diagonal components of the covariance matrix to include, and α is an internal index to carry out the trace operation. By substituting Eq. 6.22 into

Eq. 6.33, we can write the minimum-covariance cost function J as a function of $\phi^{i,\gamma_1\ldots\gamma_p}_{(t_{k+1},t_k)}$, P_k, and $\delta\boldsymbol{x}_k$:

$$J = \mathcal{W}^{\alpha i}\theta^{i,\gamma_1}P_k^{\gamma_1\gamma_2}\theta^{j,\gamma_2}\mathcal{W}^{\alpha j} = \tilde{\theta}^{\alpha,\gamma_1}P_k^{\gamma_1\gamma_2}\tilde{\theta}^{\alpha,\gamma_2} \tag{6.34}$$

where $\tilde{\theta}^{\alpha,\gamma_1} = \mathcal{W}^{\alpha i}\theta^{i,\gamma_1}$.

The first-order derivatives of this cost function with respect to the control update $\delta\boldsymbol{u}_k$ can be expressed analytically as a function of the reference STTs:

$$J_u^{\gamma_2} = 2\tilde{\theta}^{\alpha,\gamma_1\gamma_2}P_k^{\gamma_1\eta_1}\tilde{\theta}^{\alpha,\eta_1} \tag{6.35}$$

where $\tilde{\theta}^{\alpha,\gamma_1\gamma_2} = \mathcal{W}^{\alpha i}\theta^{i,\gamma_1\gamma_2}$ can be obtained analytically using Eq. 6.26.

6.4.3 Minimum-covariance cost function with control-linear noise

The minimum-covariance cost function can be expanded to consider the control-linear noise formulation from Section 6.2.4:

$$J = \tilde{\theta}^{\alpha,\gamma_1}\left(P_{k,fixed} + P_{k,prop}\right)^{\gamma_1\gamma_2}\tilde{\theta}^{\alpha,\gamma_2} = \tilde{\theta}^{\alpha,\gamma_1}P_{k,fixed}^{\gamma_1\gamma_2}\tilde{\theta}^{\alpha,\gamma_2} + \tilde{\theta}^{\alpha,\gamma_1}\sigma\beta^{\gamma_1,a}\delta u_k^a\delta u_k^b\beta^{\eta_1,b}\sigma\tilde{\theta}^{\alpha,\eta_1} \tag{6.36}$$

The first-order derivative of Eq. 6.28 can be fully expressed analytically as:

$$J_u^{\gamma_2} = 2\tilde{\theta}^{\alpha,\gamma_1\gamma_2}\left(P_{k,fixed}^{\gamma_1\eta_1} + P_{k,prop}^{\gamma_1\eta_1}\right)\tilde{\theta}^{\alpha,\eta_1} + 2\sigma^2\tilde{\theta}^{\alpha,\gamma_1}\beta^{\gamma_1,a}\delta u_k^a\beta^{\eta_1,\gamma_2}\tilde{\theta}^{\alpha,\eta_1} \tag{6.37}$$

6.4.4 Maximizing initial covariance

An alternative cost function can be formulated to maximize the magnitude of an initial covariance matrix at time t_k while satisfying constraints at time t_{k+1}. This could be beneficial to determine the navigation requirements to in order to satisfy the constraints, and thus safely perform the maneuver. In order to scale the overall state covariance rather than individual components of the covariance matrix, the problem can be formulated by using a scaling parameter ξ. The initial unscaled covariance matrix $P_{k,unscaled}$ is scaled by ξ, and the optimization scheme can then seek to

find the maximum value for ξ that satisfies the constraints. The cost function is written as:

$$J = -\xi \tag{6.38}$$

Thus, the optimization scheme will seek to maximize the scaling parameter ξ by minimizing J. The covariance matrix at time t_k for this cost function becomes

$$P_k = \xi P_{k,unscaled} \tag{6.39}$$

6.5 Constraints

Several formulations for stochastic constraints $\boldsymbol{\psi}$ can be expressed in terms of reference STTs. For these constraints, the exact first-order derivative $\boldsymbol{\psi}_u$ (and second-order derivative $\boldsymbol{\psi}_{uu}$) can be obtained solely as a function of these reference STTs.

6.5.1 Constraint on expected state

A constraint on the expected value of a linear combination of the state components of $\boldsymbol{x}_{k+1}$ at time t_{k+1} can be expressed as:

$$\psi^q = \mathbb{E}[g^{q,i} x^i_{k+1}] - \psi^q_{\text{target}} \tag{6.40}$$

$$= g^{q,i} \left(\hat{x}^i_{k+1} + \delta m^i_{k+1} \right) - \psi^q_{\text{target}} \tag{6.41}$$

where $\boldsymbol{g}^q$ is the linear mapping vector for the q-th constraint, and ψ^q_{target} is the target value for the constraint. The derivatives of this constraint can be expressed analytically as a function of the reference STTs, following Eq. 6.14. A constraint can also be formulated on a parameter that is a nonlinear function of the state components:

$$\psi^q = \mathbb{E}[\boldsymbol{z}(\boldsymbol{x}_{k+1})] - \psi^q_{\text{target}} \tag{6.42}$$

This can also be formulated analytically with STTs so long as the function $\boldsymbol{z}(\boldsymbol{x}_{k+1})$ is differentiable with respect to $\delta \boldsymbol{u}_k$.

6.5.2 State chance constraint

A chance constraint inequality on a linear combination of state parameters can be expressed as:

$$\Pr\left[x_{k+1}^i \in \mathcal{S}_{k+1}^q\right] \geq 1 - \delta_{k+1}^q \tag{6.43}$$

where $\mathcal{S}_{k+1}^q$ is the feasible region for the q-th chance constraint at time t_{k+1}. δ_{k+1}^q is the desired probability for the q-th chance constraint at stage $k+1$.

In general, evaluating this chance constraint is intractable; however we can reformulate it into a deterministic constraint by assuming the state uncertainties at time t_k are subject to Gaussian distributions, i.e. $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{m}_k, P_k)$, and the feasible region $\mathcal{S}_{k+1}^q$ is a half-space defined by the linear state mapping vector $\boldsymbol{h}^q$ and the desired constraint value d_{k+1}^q. With this assumption, it is shown that Eq. 6.43 is equivalent to [84]:

$$\psi^q = h^{q,i}\left(\hat{x}_{k+1}^i + \delta m_{k+1}^i\right) + \Phi^{-1}(1-\delta_{k+1}^q)\sqrt{h^{q,i}P_{k+1}^{ij}h^{q,j}} - d_{k+1}^q \leq 0 \tag{6.44}$$

where $\Phi^{-1}(\circ)$ denotes the inverse cumulative distribution function of the standard normal distribution. P_{k+1} can be expressed analytically as a function of the STTs mapping from t_k to t_{k+1} using the approximation from Eq. 6.22. Eq. 6.44 then becomes:

$$\psi^q = h^{q,i}\left(\hat{x}_{k+1}^i + \delta m_{k+1}^i\right) + \Phi^{-1}(1-\delta_{k+1}^q)\sqrt{h^{q,i}\theta^{i,\gamma_1}_{(t_{k+1},t_k)}P_k^{\gamma_1\eta_1}\theta^{j,\eta_1}_{(t_{k+1},t_k)}h^{q,j}} - d_{k+1}^q \leq 0 \tag{6.45}$$

Letting $\tilde{\theta}_{(t_{k+1},t_k)} = h^{q,i}\theta^{i,\gamma_1}_{(t_{k+1},t_k)}$, this simplifies to:

$$\psi^q = h^{q,i}\left(\hat{x}_{k+1}^i + \delta m_{k+1}^i\right) + \Phi^{-1}(1-\delta_{k+1}^q)\sqrt{\tilde{\theta}^{q,i}_{(t_{k+1},t_k)}P_k^{ij}\tilde{\theta}^{q,j}_{(t_{k+1},t_k)}} - d_{k+1}^q \leq 0 \tag{6.46}$$

Assuming that $\boldsymbol{h}^q$ is a constant mapping, since this is strictly a function of the reference STTs, the derivatives of this constraint with respect to the control vector can be computed analytically. We will write out explicitly the first-order derivative of the $\sqrt{\tilde{\theta}^{q,i}_{(t_{k+1},t_k)}P_k^{ij}\tilde{\theta}^{q,j}_{(t_{k+1},t_k)}}$ expression (omitting the (t_{k+1},t_k) timespan for conciseness):

$$\frac{\partial\left(\sqrt{\tilde{\theta}^{q,i}_{(t_{k+1},t_k)}P_k^{ij}\tilde{\theta}^{q,j}_{(t_{k+1},t_k)}}\right)}{\partial(\delta u_k^{\gamma_1})} = \frac{\tilde{\theta}^{q,i\gamma_1}P_k^{ij}\tilde{\theta}^{q,j}}{\sqrt{\tilde{\theta}^{q,i}P_k^{ij}\tilde{\theta}^{q,j}}} \tag{6.47}$$

where $\tilde{\theta}^{q,i\gamma_1} = \partial\tilde{\theta}^{q,i}/\partial(\delta u_k^{\gamma_1})$ can be computed from Eq. 6.14. If using the control-linear noise formulation described in Section 6.4.3, then $\partial P_k/\partial(\delta \boldsymbol{u}_k) \neq 0$, in which case the first-order derivative of $\sqrt{\tilde{\theta}^{q,i}_{(t_{k+1},t_k)} P_k^{ij} \tilde{\theta}^{q,j}_{(t_{k+1},t_k)}}$ becomes

$$\frac{\partial\left(\sqrt{\tilde{\theta}^{q,i}_{(t_{k+1},t_k)} P_k^{ij} \tilde{\theta}^{q,j}_{(t_{k+1},t_k)}}\right)}{\partial(\delta u_k^{\gamma_1})} = \frac{\tilde{\theta}^{q,i\gamma_1} P_k^{ij} \tilde{\theta}^{q,j} + \frac{1}{2}\tilde{\theta}^{q,i}\frac{\partial(P_k^{ij})}{\partial(\delta u_k^{\gamma_1})}\tilde{\theta}^{q,j}}{\sqrt{\tilde{\theta}^{q,i} P_k^{ij} \tilde{\theta}^{q,j}}} \tag{6.48}$$

where $\partial(P_k^{ij})/\partial(\delta u_k^{\gamma_1})$ for the control-linear noise formulation is given in Eq. 6.29.

6.6 Application to an Impulsive Transfer in the Earth-Moon System

We illustrate a potential use case for the proposed methodology for a spacecraft operating in cislunar space. The dynamics in the Earth-Moon system are highly nonlinear and chaotic, and linearized methods often have very small convergence regions in this regime. In addition, operational uncertainties will likely be large for spacecraft operating in this regime, which may necessitate stochastic maneuver design strategies.

6.6.1 Scenario

We consider a scenario with a spacecraft executing a multi-impulse transfer from a periodic near-rectilinear halo orbit (NRHO) in the Earth-Moon CR3BP ($\mu = 0.0121505856$), to a geosynchronous orbit around Earth in the Earth-Moon plane. As with several of the examples in other chapters in this thesis, this example is motivated by recent interest in transfers to and from NRHOs, since this is the planned operational orbit for the future Lunar Gateway mission [35]. A 5-impulse reference transfer was generated. The initial maneuver $\boldsymbol{u}_0$ at time t_0 departs the NRHO and targets a close flyby of the Moon. The second maneuver $\boldsymbol{u}_1$ at time t_1 is a statistical maneuver that occurs 24 hours prior to the close flyby of the Moon, to correct for any maneuver execution errors from $\boldsymbol{u}_0$. In the reference trajectory, this maneuver has zero ΔV since it is a statistical maneuver. The third maneuver $\boldsymbol{u}_2$ at time t_2 occurs near the closest approach of the Moon during the flyby and targets the geosynchronous orbit position. The fourth maneuver $\boldsymbol{u}_3$ at time t_3 is a statistical

maneuver. Again, in the reference trajectory, this maneuver has zero ΔV, but in practice this maneuver will be required to correct for the effects of maneuver execution errors from $\boldsymbol{u}_2$, which is scheduled to occur during the dynamically sensitive Moon flyby. The final maneuver $\boldsymbol{u}_4$ is required to match the velocity of the desired geosynchronous orbit. The controls are modeled as impulsive ΔVs; thus, $\beta = [0_{3\times3}, I_3]^T$ for each stage. This reference transfer is shown in Figs. 6.1 and 6.2. The STTs up to order $m = 4$ for this reference orbit were integrated and stored for each segment of the transfer. These are then used to analytically compute updated controls for the two statistical maneuvers given maneuver execution errors from the deterministic maneuvers, and state deviation and uncertainty values.

6.6.2 First statistical maneuver

We will first demonstrate the previously derived equations to compute a variety of maneuvers incorporating uncertainty by investigating the first planned statistical maneuver $\boldsymbol{u}_1$. The first deterministic maneuver $\boldsymbol{u}_0$ is scheduled to depart the NRHO and target the close flyby of the Moon. We assume a value of $\sigma = 0.02$ for the control-linear noise - i.e. that the 1-σ maneuver execution error is 2%. Since the flyby can be a highly sensitive portion of the trajectory, this level of maneuver execution error may result in a large deviation from the reference trajectory. The statistical maneuver $\boldsymbol{u}_1$ is therefore planned to correct for these potential deviations, and target the close Moon flyby where the deterministic flyby maneuver $\boldsymbol{u}_2$ is schedule to occur.

We will show how maneuver targeting schemes using different cost functions can be formulated using the same reference STTs. To do so, we first integrate the reference trajectory STTs up to $m = 4$ for the (t_2, t_1) segment. We simulate ten different trajectories by applying the deterministic flyby maneuver $\boldsymbol{u}_0$ and adding random noise sampled from a distribution proportional to the maneuver magnitude, according to σ. These ten trajectories are then propagated to time t_1, where the statistical maneuver $\boldsymbol{u}_1$ will be applied. Note that $\boldsymbol{u}_1$ is a statistical maneuver that has zero ΔV in the baseline trajectory. The reference trajectory STTs can be used to re-target this maneuver and update the nominal maneuver plan with a different cost function or new constraints, in reaction

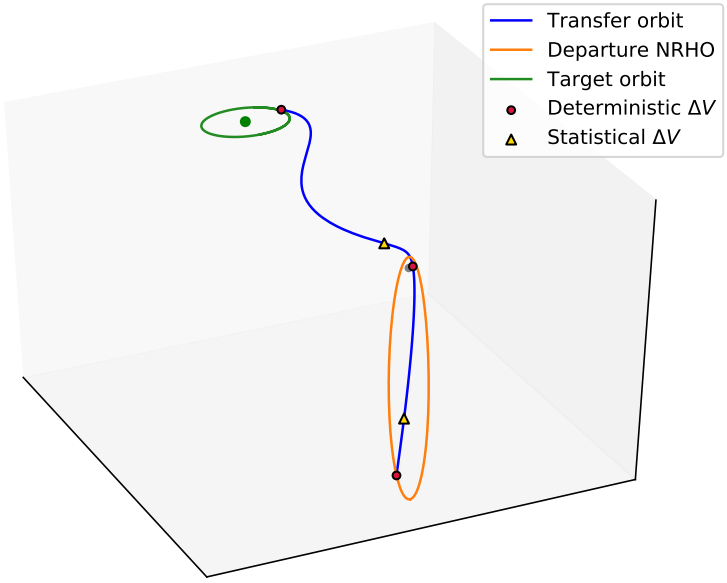

Figure 6.1: Reference 5-impulse NRHO to GEO transfer

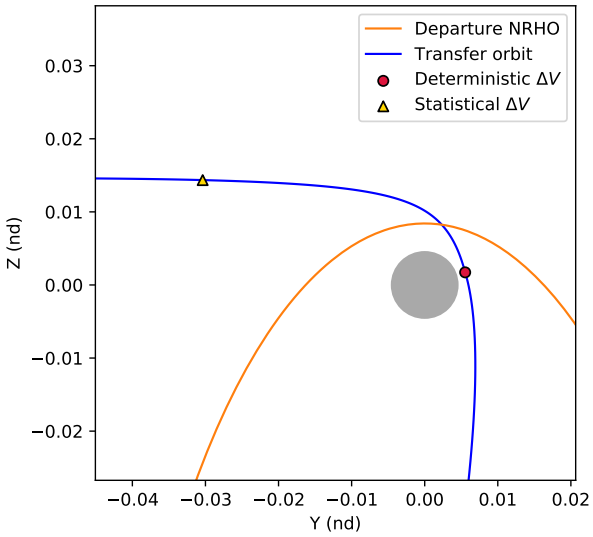

Figure 6.2: 2D-view of $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ location during Moon flyby

to maneuver execution errors, navigation errors, or changing mission priorities.

We first demonstrate how the state chance constraint formulation from Eq. 6.44 can be applied. For $\boldsymbol{u}_1$, if the uncertainty P_1 at time t_1 is found to be unexpectedly large, the most important factor for computing an updated control would be to ensure that, to some desired degree of certainty, the spacecraft is guaranteed not to fly too close to the Moon. In order to achieve this, we can place two constraints on $\boldsymbol{x}_2$. We will restrict the constraints to the $y-z$ plane in the CR3BP frame to facilitate visualization, though we note that the methods are derived to allow for constraints applied in any number of directions. A first constraint can be placed to ensure that the $y-z$ coordinates of $\mathbb{E}[\boldsymbol{x}_2]$ lie on the same axis from the center of the Moon as the reference state $\hat{\boldsymbol{x}}_2$ when projected on the $y-z$ plane, with $\boldsymbol{g}^{q_1}$ defined appropriately. A chance constraint can then be formulated to ensure that the 3-σ covariance ellipse in the $y-z$ plane intersects a line tangent to the minimum desired flyby radius in the $y-z$ plane, with $\boldsymbol{h}^{q_2}$ defined appropriately. This ensures that the spacecraft will almost certainly be no closer to the Moon than this desired radius. We assume the initial 1-σ uncertainty on this state deviation is $\boldsymbol{\sigma} = [2.5, 2.5, 2.5, 1.0, 1.0, 1.0]^T \times 10^{-4}$, which corresponds to roughly 100 km for the position values and 100 cm/s for the velocity values, and that $P_1 = \text{diag}([\boldsymbol{\sigma}^{\boldsymbol{2}}])$. We set the minimum desired 3-σ flyby radius to be 500 km above the surface of the Moon. Given these parameters, the optimization scheme was able to converge on optimal solutions for all cases. The uncontrolled and corrected trajectories are shown in Fig. 6.3. The resulting mean target state and associated 3-σ covariance ellipse are illustrated in Fig. 6.4 for the $m = 3$ case; we can see that the STT-based optimization scheme ensures the constraints are satisfied without requiring further integrations of the dynamics.

It is also possible to approximate chance constraints in non-Cartesian coordinates as a linear chance constraint by performing a first-order expansion of the constraint at each iteration to obtain $\boldsymbol{h}^{\boldsymbol{q}}$. We demonstrate this by applying a chance constraint on the radius (measured in the $y-z$ plane). We will seek to probabilistically ensure that the flyby radius at time t_2 is larger than 1000 km above the surface the Moon. The reference trajectory targeted a flyby altitude of 500 km; the larger radius may be desirable in some situations to ensure mission safety or avoid potential

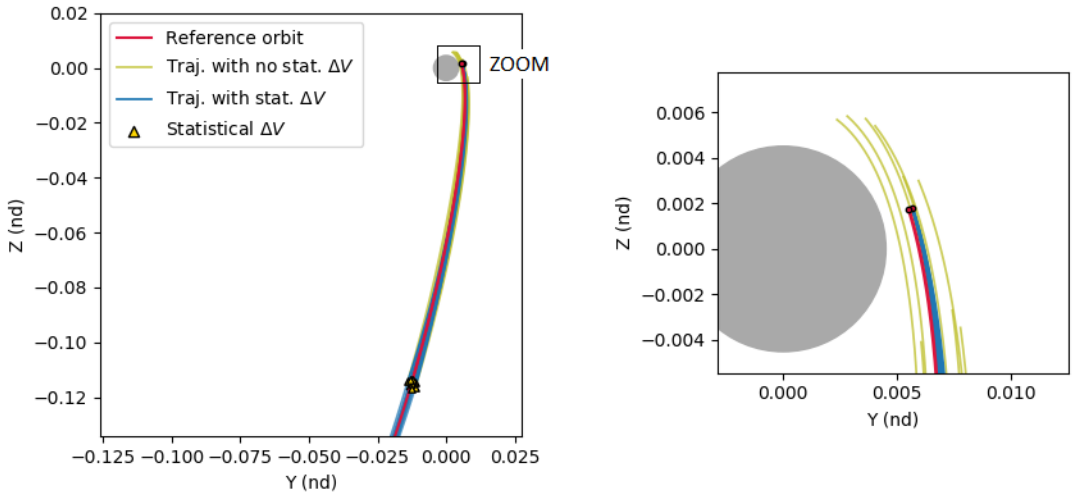

Figure 6.3: Trajectories with and without statistical maneuver $\boldsymbol{u_1}$

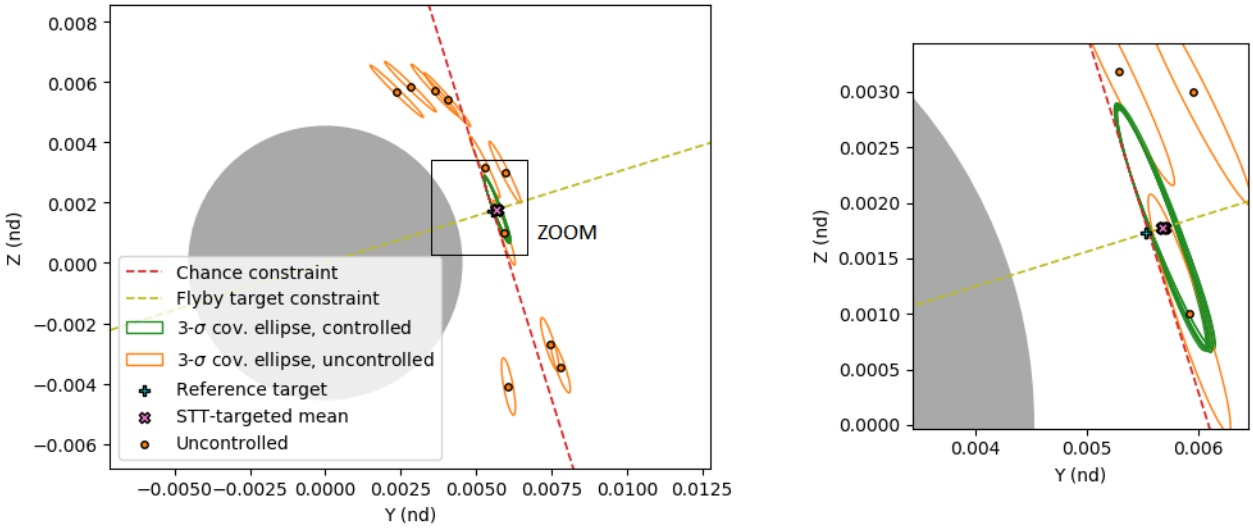

Figure 6.4: 3-σ target state covariance ellipses for transfers with and without statistical maneuver $\boldsymbol{u}_1$, with single chance constraint. The chance constraint is successfully satisfied for all ten trajectories.

conjunctions with other spacecraft in low lunar orbit. We also add two linear chance constraints on the y and z position states to enforce that the distributions remain within the top-right quadrant of the $y-z$ plane. The constraints were applied for the same ten perturbed trajectories from the previous scenario. The optimization scheme using STTs up to $m = 3$ was successfully able to converge on trajectories that satisfied all constraints. The resulting mean target state and associated 3-σ covariance ellipse are illustrated in Fig. 6.5.

Finally, we will demonstrate the utility of the maximum-covariance cost function developed in Section 6.4.4. Again, we show how a maneuver using this cost function can be computed analytically using the STTs of a reference trajectory. For this scenario, we use $P_{1,unscaled} = \text{diag}([\boldsymbol{\sigma^2}])$, where $\boldsymbol{\sigma} = [2.5, 2.5, 2.5, 1.0, 1.0, 1.0]^T \times 10^{-4}$. As with the first example in this Section, we apply a first constraint to ensure that the $y-z$ coordinates of $\mathbb{E}[\boldsymbol{x}_2]$ lie on the same axis from the center of the Moon as the reference state $\hat{\boldsymbol{x}}_2$ when projected on the $y-z$ plane, with $\boldsymbol{g}^{q_1}$ defined appropriately. Then, we apply two separate chance constraints perpendicular to this first constraint, one to ensure that the distribution is at least 500 km from the surface of the Moon, and the second to ensure that the distribution is at most some varying altitude parameter. We use altitude parameters of 540 km, 750 km, and 1000 km to demonstrate the utility of the cost function. These give allowable radius ranges of 40 km, 250 km, and 500 km, respectively. The optimization scheme was able to compute the maneuvers that enable the maximum covariance scaling ξ. The resulting covariance ellipses for all three cases are shown in Figs. 6.6-6.8. The average optimal values for the covariance scaling were found to be $\xi = 0.027$ for the 40 km case, $\xi = 0.969$ for the 200 km case, and $\xi = 5.806$ for the 500 km case. This type of analysis could be useful to determine if the current state estimate knowledge is sufficient to safely perform a maneuver given some constraints, or if the state covariance should be reduced through additional measurements prior to executing the maneuver.

6.6.3 Second statistical maneuver

Next, we will demonstrate how reference STTs can be used compute a further variety of maneuvers incorporating uncertainty by investigating the second planned statistical maneuver. The

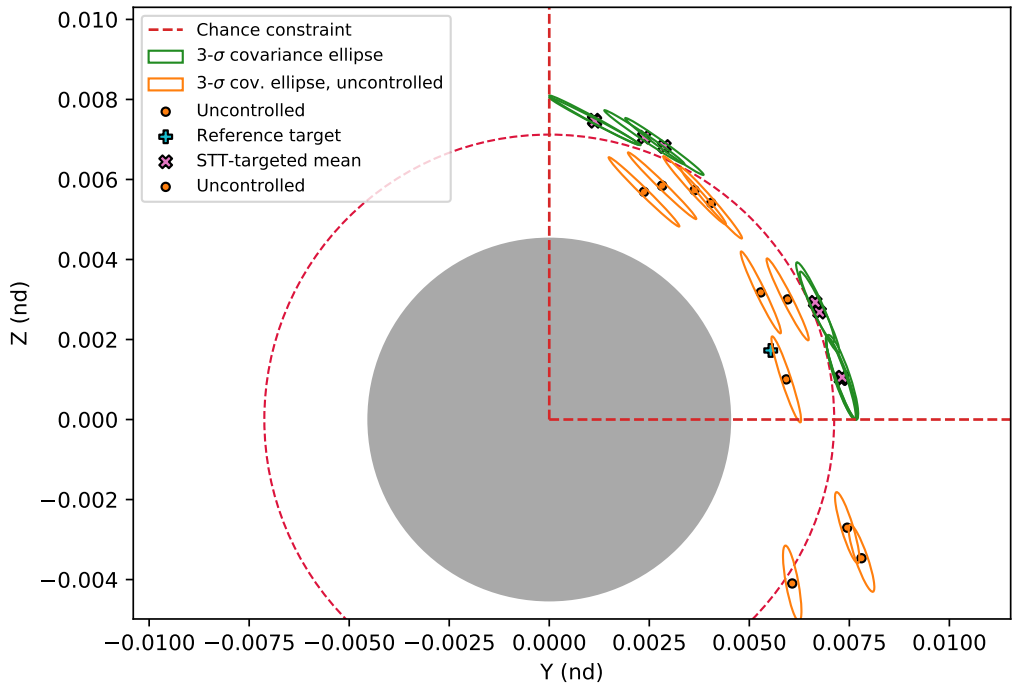

Figure 6.5: 3-σ target state covariance ellipses for transfers with and without statistical maneuver $\boldsymbol{u}_1$, with radius chance constraint. The chance constraints are successfully satisfied for all ten trajectories.

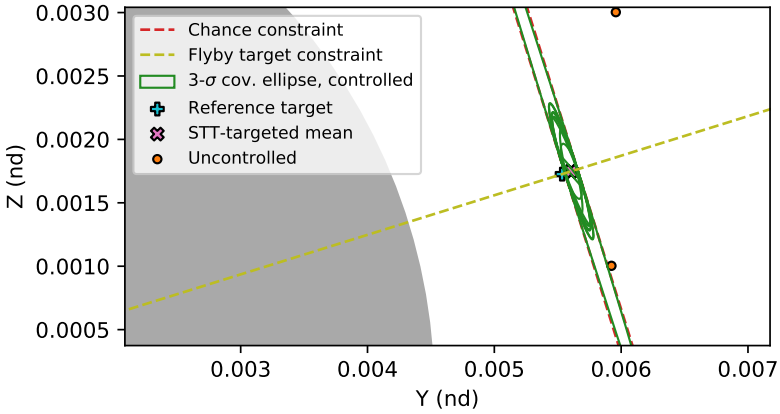

Figure 6.6: 3-σ target state covariance ellipses for maximum-covariance transfers with 40 km radius range, computed using reference STTs, $m = 3$

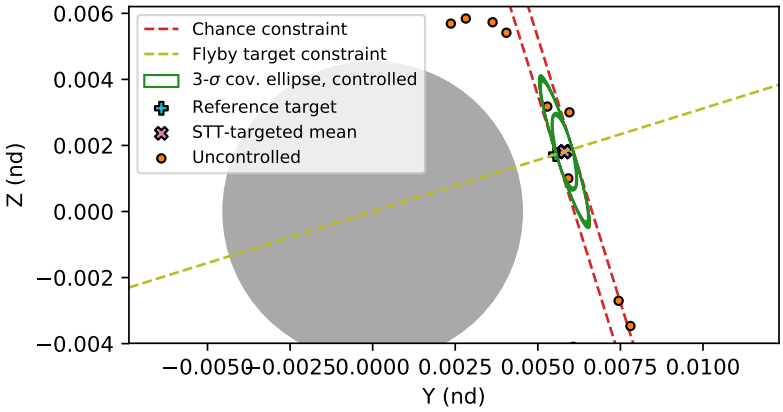

Figure 6.7: 3-σ target state covariance ellipses for maximum-covariance transfers with 200 km radius range, computed using reference STTs, $m = 3$

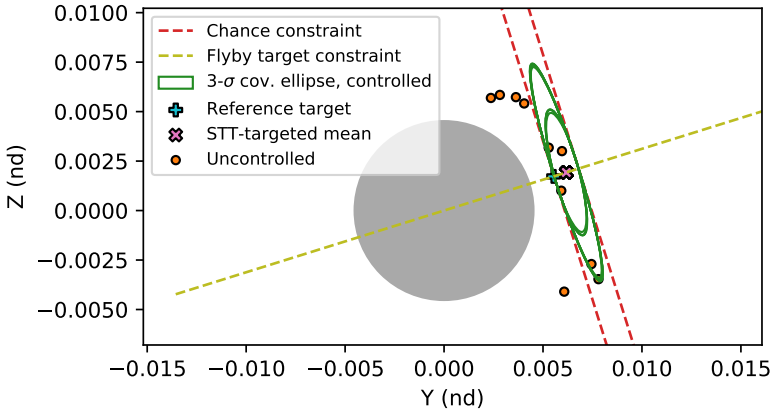

Figure 6.8: 3-σ target state covariance ellipses for maximum-covariance transfers with 500 km radius range, computed using reference STTs, $m = 3$

second deterministic maneuver $\boldsymbol{u}_2$ is planned to occur during the close flyby of the Moon. This is a highly nonlinear segment of the trajectory. We assume again a value of $\sigma = 0.02$ for the control-linear noise. $\boldsymbol{u}_2$ is a large deterministic maneuver; as such, this level of maneuver execution error will result in a large deviation from the reference trajectory. We want to use the reference STTs to efficiently compute the statistical maneuver $\boldsymbol{u}_3$ to successfully re-target the geosynchronous orbit. Since $\boldsymbol{u}_3$ is scheduled to occur only 6 hours after $\boldsymbol{u}_2$, a ground-based maneuver design procedure would not be able to meet this cadence. This provides an excellent example where a computationally lightweight on-board STT-based maneuver design scheme could find use.

For this statistical maneuver, we will target any position along the geosynchronous orbit. This target orbit can be approximated using two constraints, one enforcing the nonlinear function $R(\hat{\boldsymbol{x}}_4 + \delta\boldsymbol{x}_4) = (\hat{\boldsymbol{x}}_4 + \delta\boldsymbol{x}_4)^2 + (\hat{\boldsymbol{y}}_4 + \delta\boldsymbol{y}_4)^2 = r_{\text{geo}}^2$, and a constraint enforcing $z_4 = 0$. Recall that the reference transfer targets this orbit, so $\hat{z}_4 = 0$, and the second constraint is therefore equivalent to $\delta z_4 = 0$. We want to consider the nonlinear effects of the uncertainty propagated from t_3 to t_4, so we constrain the expected values of these constraints. The constraints can thus be written as:

$$\psi^{q_1} = (\hat{x}_4 + \mathbb{E}[\delta x_4])^2 + (\hat{y}_4 + \mathbb{E}[\delta y_4])^2 - r_{\text{geo}}^2 \tag{6.49}$$

$$\psi^{q_2} = \mathbb{E}[\delta z_4] \tag{6.50}$$

or using Eq. 6.6 this becomes:

$$\psi^{q_1} = (\hat{x}_4 + \delta m_4^x)^2 + (\hat{y}_4 + \delta m_4^y)^2 - r_{\text{geo}}^2 \tag{6.51}$$

$$\psi^{q_2} = \delta m_4^z \tag{6.52}$$

where δm_k^z refers to the z component of $\delta\boldsymbol{m}_k$. These constraints are easily differentiable with respect to $\delta\boldsymbol{u}_3$ following Eq. 6.14.

We will show how maneuver targeting schemes using different cost functions can be formulated using the same reference STTs. Ten different trajectories are simulated by applying the deterministic flyby maneuver $\boldsymbol{u}_2$ and adding random noise sampled from a distribution proportional to the maneuver magnitude. These ten trajectories are then propagated to time t_3, where the statistical

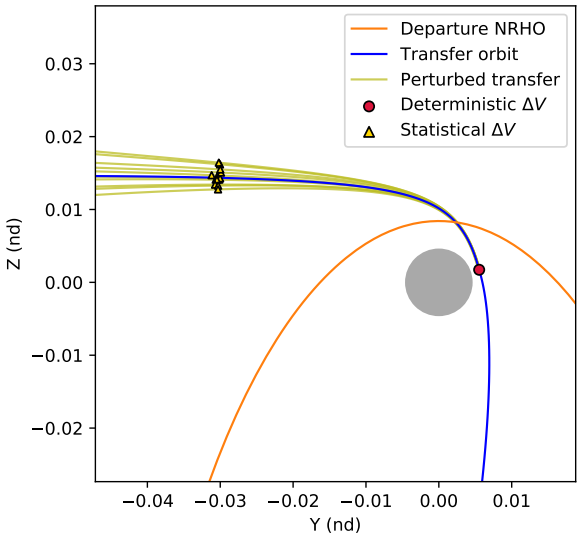

Figure 6.9: 2D-view of $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ location during Moon flyby

maneuver $\boldsymbol{u}_3$ will be applied. These perturbed trajectories are shown in Fig. 6.9. We assume the initial 1-σ uncertainty on the state deviation at time t_3 is $\boldsymbol{\sigma} = [2.5, 2.5, 2.5, 1.0, 1.0, 1.0]^T \times 10^{-4}$, which corresponds to roughly 100 km for the position values and 100 cm/s for the velocity values, and that $P_3 = \text{diag}([\boldsymbol{\sigma^2}])$. We then use the optimization scheme outlined in Section 6.3 to compute the statistical maneuver $\boldsymbol{u}_3$ that satisfies the constraints for each state deviation, while minimizing the cost function.

We first compute statistical maneuvers using the minimum-energy cost function from Eq. 6.30. The optimization scheme successfully computed solutions for all ten trajectories. These minimum-energy transfers are shown in Fig. 6.10. Next, we compute statistical maneuvers using the minimum-covariance cost function (with control-linear noise) from Eq. 6.36. We will seek to minimize uncertainty at the target time t_4 along the z-direction - this may be desirable in order to probabilistically ensure that the spacecraft arrives at Earth at the correct latitude. This also provides a relatively simple example to demonstrate the minimum-covariance cost function. We set $\mathcal{W} = \begin{bmatrix} 0 & 0 & 1000 & 0 & 0 & 0 \end{bmatrix}$ - the scaling is helpful to ensure numerical stability for the optimization algorithm. Again, the optimization scheme successfully computed solutions for all ten trajec-

Table 6.1: Average metrics for $\boldsymbol{u}_3$ computed using minimum-energy and minimum-covariance cost functions

Algorithm	Average final STT position approx. error (nd)	Average control magnitude (nd)	Average 1-σ z uncertainty (nd)
Minimum-energy, $m = 2$	2.18×10^{-4}	0.012	1.08×10^{-4}
Minimum-covariance, $m = 2$	2.73×10^{-3}	0.063	5.63×10^{-5}
Minimum-energy, $m = 3$	4.13×10^{-5}	0.012	1.14×10^{-4}
Minimum-covariance, $m = 3$	1.71×10^{-4}	0.044	5.73×10^{-5}
Minimum-energy, $m = 4$	9.19×10^{-6}	0.012	1.12×10^{-4}
Minimum-covariance, $m = 4$	8.12×10^{-5}	0.045	5.73×10^{-5}

tories. These minimum-covariance transfers are shown in Fig. 6.11. In both Figs. 6.10 and 6.11, the trajectories with no statistical maneuver applied are equivalent.

Average metrics for the different cost functions are shown in Table 6.1. For each order of STT included in the algorithm, the maneuvers computed using the minimum-energy and minimum-covariance cost functions clearly achieve their respective goals: the control magnitude is minimized for the minimum-energy cost function, while the average uncertainty in the z direction is minimized for the minimum-covariance cost function. We can also see that, as expected, including higher orders of STT in the approximation reduces the STT approximation error.

6.7 Conclusions

In this chapter, we derived the equations for using higher-order state transition tensor (STT) approximations of a nonlinear system to solve a stochastic optimal control problem with impulsive controls. An efficient truncation for the STT covariance propagation equations was derived which greatly simplifies this propagation at higher orders. We showed that a reference trajectory's STTs can be used to formulate several types of stochastic cost functions (minimum-covariance and maximum-covariance) and constraints (mean state constraints and state chance constraints). The many examples presented in this chapter illustrate how these equations could be used for computing statistical spacecraft maneuvers with a variety of different geometric and stochastic constraints.

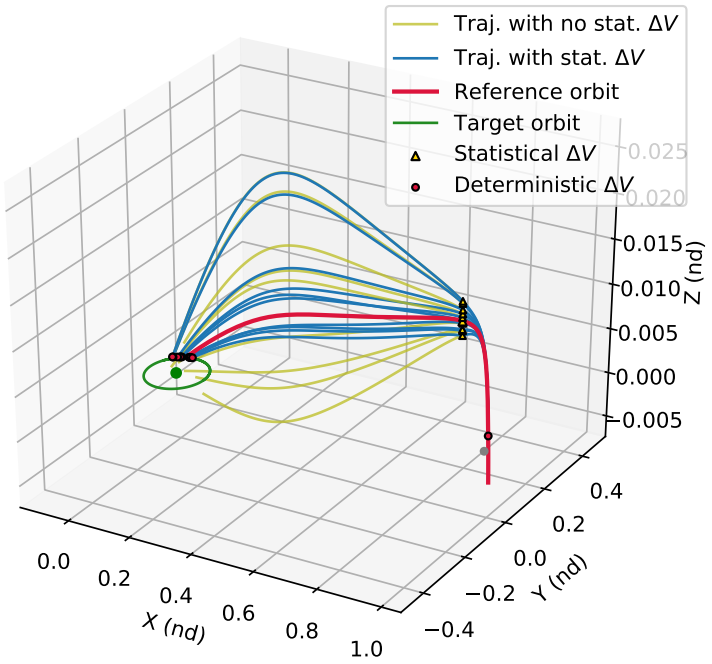

Figure 6.10: Minimum-energy transfers computed using reference STTs, $m = 3$

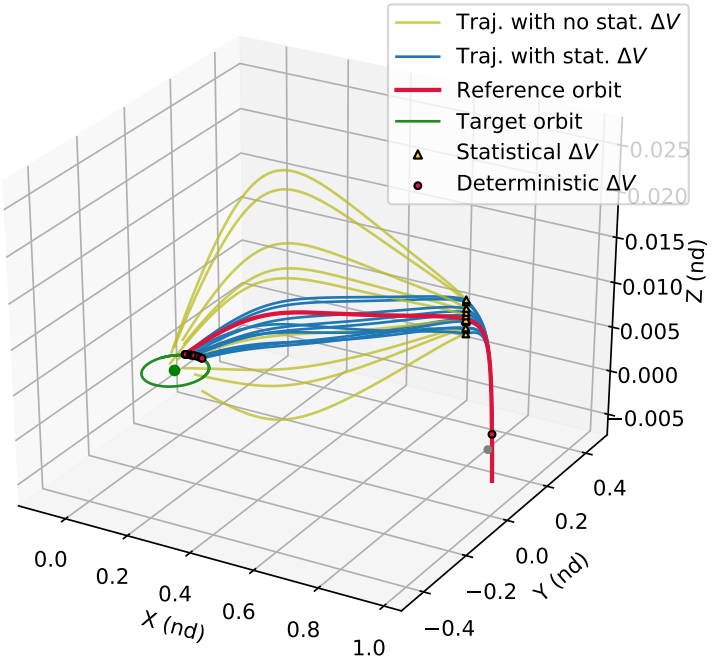

Figure 6.11: Minimum-covariance transfers computed using reference STTs, $m = 3$

In an operational setting, the higher-order STTs of a reference trajectory could be integrated and stored over the desired timespan between some pre-specified maneuver times. The priorities and constraints on the maneuver may be variable up to the maneuver time, which may necessitate using a different cost function or constraint from the reference trajectory. An on-board or real-time STT-based guidance scheme could use the equations derived in this chapter to efficiently compute updated maneuvers which incorporate the effects of state uncertainty, in reaction to maneuver execution errors, navigation errors, or changing mission priorities.

The equations for stochastic maneuver design derived in this chapter are useful beyond this proposed application. The STT/DDP algorithm derived in Chapter 5 could be modified to incorporate the effects of state uncertainty, for example by incorporating state chance constraints or a minimum-covariance cost function. In addition, these equations could be used to speed up any stochastic optimization algorithm that requires a large number of iterations close to a reference - an example of one such algorithm (which requires propagating a Gaussian mixture through nonlinear dynamics over many iterations) is presented in Chapter 7. These equations could also be used to efficiently formulate a robust, nonlinear statistical tool for $\Delta V99$ studies accounting for orbit determination and maneuver execution errors. This could enable more rapid analysis of expected maneuver performance in reaction to these errors.

Chapter 7

Maneuver Design with Non-Gaussian Chance Constraints

7.1 Introduction

As stated in Chapter 6, many existing spacecraft maneuver targeting algorithms consider strictly deterministic systems with no uncertainty in the states or dynamics. In any operational scenario, however, there will always be some degree of uncertainty in the system. A common strategy to ensure that a specific maneuver is robust to operational uncertainties is to perform a Monte Carlo simulation, and examine the resulting statistics of the final state distribution. This can be a computationally intensive procedure, which has lead to the development of **chance-constrained** maneuver design algorithms, which seek to achieve specific goals while probabilistically guaranteeing that the spacecraft satisfies the desired constraints under uncertainty. Most of these existing algorithms rely on the assumption that the spacecraft state uncertainties obey Gaussian distributions (e.g., [84]). This Gaussian assumption is valid if the system dynamics are sufficiently linear, or the measurement update frequency is high enough. For example, chance-constrained trajectory control has been successfully applied for robotics path-planning algorithms [13, 85], where this Gaussian assumption generally holds true.

For astrodynamics applications, this is often an inaccurate assumption when using Cartesian coordinates, due to the high nonlinearities present in the system. One approach to address this issue is to express the chance constraint in a coordinate system where distributions remain close to Gaussian when propagated through the dynamics, such as modified equinoctial elements [83] or Milankovitch elements [81]. However, for many applications, such as collision avoidance [95], it is

most desirable to express a chance constraint in Cartesian coordinates. This motivates developing methods to express a chance constraint on a non-Gaussian distribution.

Some existing studies have focused on non-Gaussian chance-constrained control, either using parametric functions to approximate the distribution [48], or by deriving approximate bounds based on the statistical moments of the underlying distributions [113]. In this chapter, we propose an alternative strategy in which the chance constraint on the non-Gaussian distribution is approximated as a conjunction of individual chance constraints on a mixture of Gaussian distributions (each referred to as a **mixand**). This approximation is valid as long as the Gaussian mixture model (GMM) is a sufficiently accurate approximation of the true distribution. GMMs have been shown to be useful tools for approximating non-Gaussian distributions, and have found particular use for our motivating application of nonlinear spacecraft uncertainty propagation [108, 37]. The general idea behind this concept is visualized in Fig. 7.1. By considering a better representation of the spacecraft state probability density function with the GMM, we can more accurately capture the statistics on how the state distribution violates (or does not violate) the constraint.

Optimally weighting the importance of the individual chance constraints on each mixand will reduce the degree of conservatism in converting the original chance constraint to a conjunction of individual chance constraints. Several strategies exist to determine the optimal weighting for the individual chance constraints [12, 112]. One such strategy is **iterative risk allocation** (IRA), proposed by Ono and Williams [85]. IRA consists of iteratively solving an upper-stage that optimizes risk allocation, and a lower-stage that optimizes the control sequence for the given risk allocation. IRA has been applied to a variety of dynamic systems with multiple individual chance constraints at separate discrete times [85, 96], but has not previously been applied to a non-Gaussian distribution approximated using GMMs.

In this chapter we develop the framework to approximate a non-Gaussian chance constraint on a spacecraft state using a Gaussian mixture model, and apply iterative risk allocation to optimally assign risk to each of the mixture's components. We then show how we can significantly speed up the risk allocation algorithm by using the higher-order state transition tensors (STTs) of

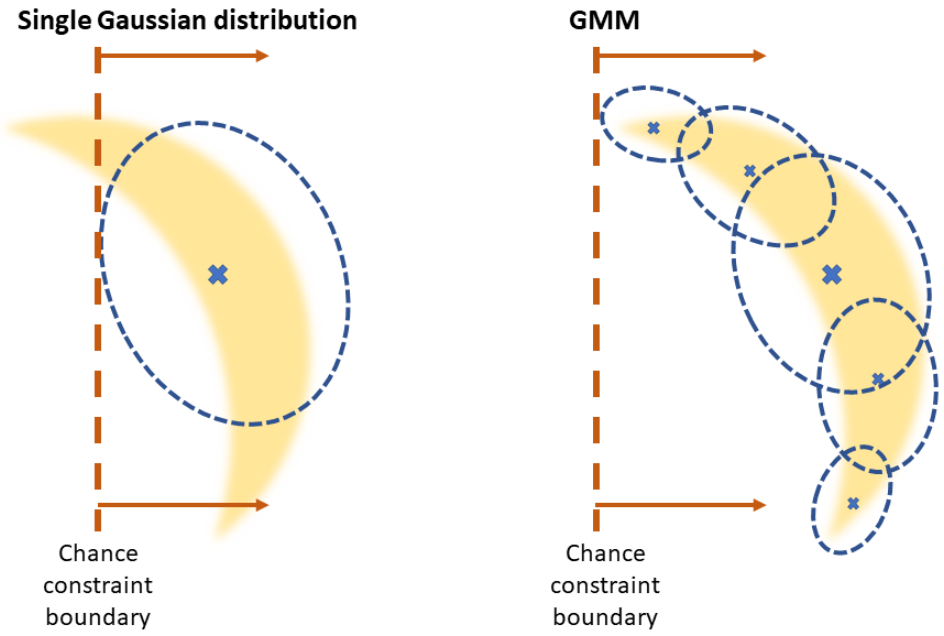

Figure 7.1: Illustration of applying a chance constraint on a Gaussian mixture model to approximate a non-Gaussian chance constraint

some reference trajectory to approximate the linear covariance propagation of each mixand. The chapter is structured as follows. We begin with an introduction on chance-constrained control and Gaussian mixture models. We then derive the theory required to approximate a non-Gaussian chance constraint with a GMM. We present the iterative risk allocation algorithm for the GMM chance constraint, using the algorithm from Ref. [85] as a baseline. We then develop a strategy for significantly reducing the computational time required for the algorithm by using the higher-order STTs of a reference trajectory. Finally, we apply the resulting algorithm to maneuver targeting scenarios for an asteroid-orbiting spacecraft with a box chance constraint, and for a spacecraft approaching the surface of Jupiter's moon Europa on a low-energy trajectory.

7.2 Background

7.2.1 Nonlinear dynamics

It is helpful to restate the mathematical formulation of the dynamics used in this chapter. The dynamics of a spacecraft are modeled as a discrete-time nonlinear time-varying system. Let $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ be the state vector at time t_k $(k = 0, 1, ..., n-1)$. The system can be expressed as:

$$\boldsymbol{x}_{k+1} = \phi(\boldsymbol{x}_k + \beta \boldsymbol{u}_k) \tag{7.1}$$

where ϕ represents the nonlinear solution flow of the system, $\boldsymbol{u}_k$ is the impulsive control applied at time t_k, and the matrix $\beta \in \mathbb{R}^{n_x \times n_u}$ is the mapping between the control and state vectors. We also consider $\boldsymbol{x}_k$ to be a random vector with associated mean $\bar{\boldsymbol{x}}_k$ and covariance P_k. Note that since the dynamic system is nonlinear, we cannot assume that the state distribution at time t_k is Gaussian, even if the state at time t_{k-1} is. For the optimal control problem we will seek to minimize an objective function J.

7.2.2 Chance constraints

A chance constraint inequality on a linear combination of state parameters at time t_k can be expressed as:

$$\Pr\left[\boldsymbol{x}_k \in \mathcal{S}_{k,j}\right] \geq 1 - \delta_{k,j} \tag{7.2}$$

where $\mathcal{S}_{k,j}$ is the feasible region for the j-th chance constraint, $j = 0, 1, ..., N_c$. $\delta_{k,j}$ is the desired probability for the j-th chance constraint. The problem considered in this chapter can thus be formulated as follows.

Problem 1 Optimal control problem with chance constraints

$$\min_{\boldsymbol{u}_k} \; J(\boldsymbol{u_k}) \tag{7.3}$$

$$\text{s.t.:} \quad \boldsymbol{x}_{k+1} = \phi(\boldsymbol{x}_k + \beta \boldsymbol{u}_k) \tag{7.4}$$

$$\Pr\left[\boldsymbol{x}_k \in \mathcal{S}_{k,j}\right] \geq 1 - \delta_{k,j} \tag{7.5}$$

For a general non-Gaussian distribution, evaluating this form of chance constraint is intractable; however, we can reformulate it into a deterministic constraint by assuming the state uncertainties are subject to Gaussian distributions, i.e. $\boldsymbol{x}_k \sim \mathcal{N}(\bar{\boldsymbol{x}}_k, P_k)$, and the feasible region $\mathcal{S}_{k,j}$ is a half-space defined by the linear state mapping vector $\boldsymbol{h}_{k,j}$ and the desired constraint value $d_{k,j}$. With this assumption, it is shown that Eqn. 7.2 is equivalent to [84]:

$$\boldsymbol{h}_{k,j}^T \bar{\boldsymbol{x}}_k + \Phi^{-1}(1 - \delta_{k,j}) \sqrt{\boldsymbol{h}_{k,j}^T P_k \boldsymbol{h}_{k,j}} - d_{k,j} \leq 0 \tag{7.6}$$

where $\Phi^{-1}(\circ)$ denotes the inverse cumulative distribution function of the standard normal distribution.

7.2.3 Gaussian mixtures

An arbitrary distribution $p(\boldsymbol{x}_k)$ at time t_k can be approximated by a weighted sum of N Gaussian distributions:

$$p(\boldsymbol{x}_k) \sim \sum_{i=1}^{N} \alpha^i \mathcal{N}(\bar{\boldsymbol{x}}_k^i, P_k^i) \tag{7.7}$$

where α^i represents the weight associated with the i-th mixture component (also known as a **mixand**). $\bar{\boldsymbol{x}}_k^i$ and P_k^i are the mean state vector and covariance matrix associated with the i-th mixand. The complete Gaussian mixture model (GMM) at time t_k is fully described by α^i, $\bar{\boldsymbol{x}}_k^i$, and P_k^i, $(i = 1, ..., N)$.

The GMM can be propagated through nonlinear dynamics by individually propagating each mixand using traditional uncertainty propagation methods for Gaussian distributions. If the system is well-approximated by a linearization of the dynamics, the state transition matrix ϕ^i can be computed for each component, and each mixand's covariance can be propagated linearly:

$$P_{k+1}^i = \phi_{(t_{k+1},t_k)}^i P_k^i (\phi_{(t_{k+1},t_k)}^i)^T \tag{7.8}$$

For highly nonlinear systems, a more sophisticated method such as the unscented transform [64] or state transition tensors [92] could be desirable. However, if the number of mixands is selected such that each mixand is small enough to remain within its linear accuracy region, Eq. 7.8 will be sufficiently accurate.

A Gaussian distribution at time t_k with mean state vector $\bar{\boldsymbol{x}}_k^g$ and covariance P_k^g can be expressed exactly using a Gaussian mixture model. In this case, the initial mean and covariance values for the mixture components must satisfy the following equations:

$$\bar{\boldsymbol{x}}_k^g = \sum_{i=1}^{N} \alpha^i \bar{\boldsymbol{x}}_k^i \tag{7.9}$$

$$P_k^g = \sum_{i=1}^{N} \alpha^i (P_k^i + \bar{\boldsymbol{x}}_k^i (\bar{\boldsymbol{x}}_k^i)^T) - \left(\sum_{i=1}^{N} \alpha^i \bar{\boldsymbol{x}}_k^i \right) \left(\sum_{i=1}^{N} \alpha^i \bar{\boldsymbol{x}}_k^i \right)^T \tag{7.10}$$

7.3 Non-Gaussian chance constraints

For a general non-Gaussian distribution, there is no exact method to perform the conversion of a chance constraint into a deterministic constraint. We propose to address this issue by approximating the non-Gaussian distribution as a Gaussian mixture. The chance constraint on the full distribution can then be approximated as a joint chance constraint on the mixands.

A chance constraint applied on a GMM with N Gaussian mixands can be expressed as

$$\sum_{i=1}^{N} \alpha^i \left(\Pr \left[\boldsymbol{x}_k^i \in \mathcal{S}_{k,j} \right] \right) \geq 1 - \Delta_{k,j} \tag{7.11}$$

where $\Delta_{k,j}$ represents the probabilistic bound for the j-th chance constraint at time t_k. Eqn. 7.11 cannot generally be converted to a deterministic chance constraint as in Eqn. 7.6. It can, however, be conservatively approximated as a conjunction of individual chance constraints on each of the mixands by using Boole's inequality:

$$\bigwedge_{i=1}^{N} \Pr \left[\boldsymbol{x}_k^i \in \mathcal{S}_{k,j} \right] \geq 1 - \frac{\delta_{k,j}^i}{\alpha^i} \tag{7.12}$$

$$\sum_{i=1}^{N} \delta_{k,j}^i \leq \Delta_{k,j} \tag{7.13}$$

$$\delta_{k,j}^i < \alpha^i \tag{7.14}$$

The risk allocated to the j-th constraint for the i-th mixand is scaled by the mixture weight α^i. We will refer to $\delta_{k,j}^i$ as the **unweighted** risk associated with the i-th component mixture, and $\delta_{k,j}^i / \alpha^i$ as the **weighted** risk. The upper bound in Eqn. 7.14 on each unweighted risk term is required to ensure the weighted risk for each individual chance constraint does not exceed 1.

Since each of these individual constraints consists of a linear constraint on a Gaussian mixture, they can each be converted to deterministic constraints on the specific mixand's mean and

covariance following the procedure in Eq. 7.6. These become

$$(\boldsymbol{h}^i_{k,j})^T \bar{\boldsymbol{x}}^i_{k,j} + \Phi^{-1}\left(1 - \frac{\delta^i_{k,j}}{\alpha^i}\right)\sqrt{(\boldsymbol{h}^i_{k,j})^T P^i_k \boldsymbol{h}^i_{k,j}} - d_{k,j} \leq 0 \tag{7.15}$$

where $\boldsymbol{h}^i_{k,j}$ is the linear state mapping vector for the j-th constraint at time t_k applied on the i-th mixand. We can then reformulate Problem 1 into a tractable form.

Problem 2 Optimal control problem with chance constraints on Gaussian mixture model

$$\min_{\boldsymbol{u}_k} \; J(\boldsymbol{u_k}) \tag{7.16}$$

$$\text{s.t.:} \quad \boldsymbol{x}^i_{k+1} = \phi(\boldsymbol{x}^i_k + \beta \boldsymbol{u}_k) \tag{7.17}$$

$$(\boldsymbol{h}^i_{k,j})^T \bar{\boldsymbol{x}}^i_{k,j} + \Phi^{-1}\left(1 - \frac{\delta^i_{k,j}}{\alpha^i}\right)\sqrt{(\boldsymbol{h}^i_{k,j})^T P^i_k \boldsymbol{h}^i_{k,j}} - d_{k,j} \leq 0 \tag{7.18}$$

$$\sum_{i=1}^{N} \delta^i_{k,j} \leq \Delta_{k,j} \tag{7.19}$$

$$\delta^i_{k,j} < \alpha^i \tag{7.20}$$

Problem 2 is formulated with a fixed risk $\Delta_{k,j}$ associated with the j-th chance constraint at time t_k. If we wish to satisfy multiple chance constraints at time t_k with an overall probability bound Δ_k, we can enforce the additional constraint $\sum_{j=1}^{N_c} \Delta_{k,j} \leq \Delta_k$. A chance constraint consisting of multiple constraints on a GMM at separate discrete times, satisfying the probability bound Δ, can be formulated in a similar manner by enforcing $\sum_{k=0}^{T-1}\sum_{j=1}^{N_c} \Delta_{k,j} \leq \Delta$.

Eq. 7.15 will be convex if $\delta^i_{k,j}/\alpha_i \leq 0.5, \; \forall(i,j)$. This property is necessary in order to use convex optimization solvers to solve Problem 2, as is frequently done in the literature (e.g., [84, 96]). Thus, in order to ensure the problem remains convex, a tighter bound of $\delta^i_{k,j} \leq \alpha^i/2$ would need to be used instead of Eq. 7.14. This can be guaranteed by choosing the mixture weights such that $\alpha^i \geq 2\Delta_{k,j}, \; \forall(i,j,k)$. For the examples provided in this work we do not require that the problem

be convex, so the tighter bounds are not required. For all subsequent sections, we will consider the case for Problem 2 where multiple chance constraints are enforced at time t_k with the additional constraint $\sum_{j=1}^{N_c} \Delta_{k,j} \leq \Delta_k$.

7.4 Risk Allocation

7.4.1 Conservative risk allocation

The risk levels $\delta_{k,j}^i$ become decision variables in the overall problem. A simple, feasible allocation for $\delta_{k,j}^i$ can be obtained by equally dividing the total allowable risk between each individual chance constraint. This can be written as

$$\delta_{k,j}^i = \frac{\Delta_k}{N \times N_c} \tag{7.21}$$

If Eq. 7.21 results in $\delta_{k,j}^i \geq \alpha^i$ holding true for any constraints, then $\delta_{k,j}^i$ should be reduced for that constraint such that $\delta_{k,j}^i < \alpha^i$. This form of risk allocation will ensure that the overall chance constraint is satisfied, but will generally result in overly conservative solutions. In the remainder of this chapter, we will call this form of risk allocation a **conservative risk allocation**.

7.4.2 Iterative risk allocation

A less conservative approach is to consider the allocation of $\delta_{k,j}^i$ values within the optimization problem. When examining Problem 2, we see that this naturally leads to a two-stage optimization framework, with an upper-stage optimization solving for the risk allocation parameters $\delta_{k,j}^i$, and a lower-stage which solves the control problem for the given risk allocation. This procedure can be iterated until an overall solution that satisfies tolerances is obtained. Ref. [85] contains details on a mathematical formulation of this procedure, which has been called Iterative Risk Allocation (IRA).

The basics of the IRA algorithm developed in Ref. [85] are outlined as follows. We will use $\boldsymbol{\delta}_{(n)}$ to refer to the vector of unweighted risk variables $\delta_{k,j}^i$ for the n-th iteration of the algo-

rithm. Given an initial feasible risk allocation $\boldsymbol{\delta}_{k,(0)}$ with an associated optimal cost $J^*(\boldsymbol{\delta}_{k,(0)})$, the IRA algorithm constructs a sequence of feasible risk allocations $(\boldsymbol{\delta}_{k,(0)}, \boldsymbol{\delta}_{k,(1)}, ..., \boldsymbol{\delta}_{k,(n)})$ such that $J^*(\boldsymbol{\delta}_{k,(0)}) \geq J^*(\boldsymbol{\delta}_{k,(1)}) \geq ... \geq J^*(\boldsymbol{\delta}_{k,(n)})$.

Given the feasible risk allocation $\boldsymbol{\delta}_{(n)}$, we first construct $\boldsymbol{\delta}'_{(n)}$ by tightening the inactive constraints. For each mixand i where the j-th constraint is active, we set $(\delta^i_{k,j,(n)})' = \delta^i_{k,j,(n)}$. Then, for each mixand i where the j-th constraint is inactive, we set $(\delta^i_{k,j,(n)})' < \delta^i_{k,j,(n)}$. Following this, $\boldsymbol{\delta}_{(n+1)}$ is constructed by loosening the active constraints. To do so, we set $\delta^i_{k,j,(n+1)} = (\delta^i_{k,j,(n)})'$ for each mixand where the constraint is inactive. For each active constraint, we then choose $\delta^i_{k,j,(n+1)} \geq (\delta^i_{k,j,(n)})'$ while ensuring that Eqs. 7.13 and 7.14 remain satisfied. The procedure is iterated until the change in the objective function from the new risk allocation is below some tolerance ϵ (i.e., $|J^*(\boldsymbol{\delta}_{(n)}) - J^*(\boldsymbol{\delta}_{(n-1)})| < \epsilon$). Initial values for $\boldsymbol{\delta}_{(0)}$ can be obtained using the conservative risk allocation model from Eq. 7.21.

The full IRA algorithm with Gaussian mixture model is detailed in Algorithm 2 [85, 24]. In later sections in this chapter, we will refer to this algorithm as the GMM IRA algorithm. Note that $\Phi(\circ)$ corresponds to the cumulative distribution function of the standard normal distribution. The lower-stage optimization problem to compute the optimal control solution for the current risk allocation for $\delta^i_{k,j}$ is solved in Line 4. In Line 10, inactive constraints are tightened with a parameter γ, $0 < \gamma < 1$. In Line 14, all active constraints are loosened, with a small tolerance parameter ϵ_{IRA} required to enforce $\delta^i_{k,j} < \alpha^i$. Following from the discussions in Ref. [85], the IRA algorithm generates a sequence of feasible risk allocations $(\boldsymbol{\delta}_{k,(0)}, \boldsymbol{\delta}_{k,(1)}, ..., \boldsymbol{\delta}_{k,(n)})$ that monotonically decrease the cost function $J^*(\boldsymbol{\delta})$.

7.5 Tractable Implementation using State Transition Tensors

7.5.1 Motivation

If using strictly numerical integration, Algorithm 2 quickly becomes prohibitively expensive as the number of mixands or chance constraints increases. Integrating the state transition matrix

Algorithm 2 Iterative Risk Allocation with Gaussian Mixture Model

$\forall(i) \quad \delta^i_{k,j} \leftarrow \Delta_k/(N \times N_c)$
while $|J^* - J^*_{prev}| < \epsilon$ **do**
    $J^*_{prev} \leftarrow J^*$
    Solve Problem 2 with $\delta^i_{k,j}$
    $N_{active} \leftarrow$ number of active constraints
    if $N_{active} = 0$ or $N_{active} = N \times N_c$ **then**
        break;
    end if
    for all i such that j-th constraint is inactive for i-th mixand at time t_k **do**
        $$\delta^i_{k,j} \leftarrow \gamma\delta^i_{k,j} + \alpha^i(1-\gamma)\left[\frac{\Phi(\boldsymbol{h}^T_{k,j}\boldsymbol{x}^{i,*}_k - g_{k,j})}{\sqrt{\boldsymbol{h}^T_{k,j}P^i_k\boldsymbol{h}_{k,j}}}\right]$$
    end for
    $\delta_{k,residual} \leftarrow \Delta_k - \sum_{j=1}^{N_c}\sum_{i=1}^{N}\delta^i_{k,j}$
    for all i, j such that j-th constraint is active for i-th mixand at time t_k **do**
        $$\delta^i_{k,j} \leftarrow \min\left\{\delta^i_{k,j} + \delta_{k,residual}/(N \times N_c),\ \ \alpha^i - \epsilon_{IRA}\right\}$$
    end for
end while

for each mixand, in order to propagate each mixand's covariance following Eq. 7.8, is the most computationally expensive portion of this algorithm. Fujimoto and Scheeres [44] previously derived a tractable expression to linearly propagate the covariance of a GMM using the higher-order state transition tensors (STTs) of some reference trajectory. This procedure can be used to significantly speed up the GMM IRA algorithm.

7.5.2 State transition tensors

As state in previous chapters, the first-order derivative of Eq. 2.14 with respect to the state deviation $\delta \boldsymbol{x}_k$ at time t_k corresponds to an approximation of the true linear STM. This first-order derivative (which we will call θ) can be expressed as a function of the reference STTs, following Eq. 6.26.

$$\theta^{i,\gamma_1}_{(t_{k+1},t_k)} = \frac{\partial(\delta x^i_{k+1})}{\partial(\delta x^{\gamma_1}_k)} = \phi^{i,\gamma_1}_{(t_{k+1},t_k)} + \sum_{p=2}^{m} \frac{1}{(p-1)!} \phi^{i,\gamma_1\gamma_2\ldots\gamma_p}_{(t_{k+1},t_k)} \delta x^{\gamma_2}_k \ldots \delta x^{\gamma_p}_k \tag{7.22}$$

The linear covariance propagation of a perturbed trajectory (in the vicinity of a reference trajectory) with initial covariance P_k can thus be approximated using the reference trajectory's STTs, following the procedure from Eq. 6.22:

$$P^{ij}_{k+1} = \theta^{i,\gamma_1}_{(t_{k+1},t_k)} P^{\gamma_1\eta_1}_k \theta^{j,\eta_1}_{(t_{k+1},t_k)} \tag{7.23}$$

Using Eq. 7.23, the chance constraint equation from Eq. 7.6 can be expressed analytically as a function of the reference trajectory's STTs. In addition, the first and second-order derivatives of the analytical expression for Eq. 7.6 can be expressed analytically as a function of the reference STTs, as shown in Eq. 6.47. This means that cheap, analytical Jacobian and/or Hessian information can be provided to the optimization algorithm used for the lower-stage optimization procedure, which can further speed up this portion of the algorithm if using a gradient-based solver.

7.5.3 Gaussian mixture propagation using STTs

Eqs. 7.22 and 7.23 can be used to approximate the linear covariance propagation of a Gaussian mixture model, similar to the procedure detailed in Ref. [44]. The STTs of some reference trajectory can be integrated and stored, and subsequently used to linearly propagate each mixand's covariance in a mixture. For the algorithm developed in this chapter, we can begin by numerically computing the solution for the chance constraint with the single Gaussian distribution, which is a relatively simple problem to solve. We can then integrate the higher-order STTs of the single-Gaussian optimal trajectory, and use these STTs to analytically evaluate the state and covariance propagation for each mixand at each iteration of the IRA algorithm. This can be repeated until the upper-stage risk allocation algorithm converges on a satisfactory allocation. Since the lower-stage optimization would typically be performed by numerically integrating each mixand's state and STM separately, using this approximation will result in significant computational savings. We will also show in later results that the accuracy loss from this procedure is relatively small, so long as the GMM solution remains within the higher-order convergence region of the STT expansion around the single Gaussian solution.

7.6 Application: Asteroid Orbiter

7.6.1 Scenario

We will demonstrate our proposed algorithm, including the tractable implementation with STTs, on the simple impulsive maneuver targeting scenario from Ref. [24]. Consider a spacecraft orbiting a small asteroid ($\mu = 5.2\,\mathrm{m}^3/\mathrm{s}^2$) in a circular terminator orbit. The dynamics of the system can be approximated by assuming the central body is a point mass. The equations of motion for the two-body problem are given in Eq. 2.1. The initial state values for the spacecraft's nominal trajectory at the initial time t_0 are

$$\boldsymbol{x}_0 = \begin{bmatrix} -1000 & 0 & 0 & 0 & 0 & -7.211 \times 10^{-2} \end{bmatrix}^T \ (\mathrm{m}, \mathrm{m/s}) \tag{7.24}$$

This corresponds to a polar circular orbit in the $x-z$ plane, commonly referred to as a **terminator** orbit. The period for this circular orbit is roughly 24 hours.

Inspired by the reconnaissance maneuvers that were conducted for the OSIRIS-REx mission [9] in preparation for the eventual touch-and-go sample maneuver, a maneuver is planned to place the spacecraft on an orbit which approaches the surface of the asteroid at a specific altitude and time. This may be desirable in order to perform reconnaissance over a specific location of the asteroid. Given some initial uncertainty in the spacecraft state, we want to ensure that the spacecraft can satisfy some desired state constraint or constraints at a final time t_f, to a specified probability level. For this scenario, we will define our target state constraint as a box. This can be formulated as a conjunction of six chance constraints defining the boundaries of the target box in position space. The risk threshold is set to $\Delta_f = 0.05$; in other words, we want 95% of the state distribution at the target time to lie within the constraint box. The six chance constraints can be expressed in the asteroid-centric inertial Cartesian frame:

$$\Pr \begin{bmatrix} 495 \leq x_f \leq 505 \text{ m} \\ -80 \leq y_f \leq 80 \text{ m} \\ -25 \leq z_f \leq 25 \text{ m} \end{bmatrix} \geq 0.95 \tag{7.25}$$

This corresponds to a box at a 500 m radius of periapse, or roughly 250 m altitude above the asteroid surface, which could be suitable for performing surface reconnaissance. The tight constraint on the final x-position ensures that the spacecraft will probalistically remain at a safe altitude from the asteroid, while still coming close enough to the surface to perform the desired operations. We will also enforce that the post-maneuver orbit lies within the $x-y$ plane. This can be ensured by enforcing that $\dot{z}_f = 0$. The setup for this scenario is illustrated in Fig. 7.2.

Initial state uncertainty values of

$$\boldsymbol{\sigma}_0 = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 1 \times 10^{-8} & 1 \times 10^{-8} & 1 \times 10^{-8} \end{bmatrix}^T (\text{m}, \text{m/s}) \tag{7.26}$$

are applied. We assume that $P_0 = \text{diag}([\boldsymbol{\sigma}_\mathbf{0}^\mathbf{2}])$ and that these initial state uncertainties are Gaussian. We seek to minimize the magnitude of the control at time t_0, so the objective function is defined as

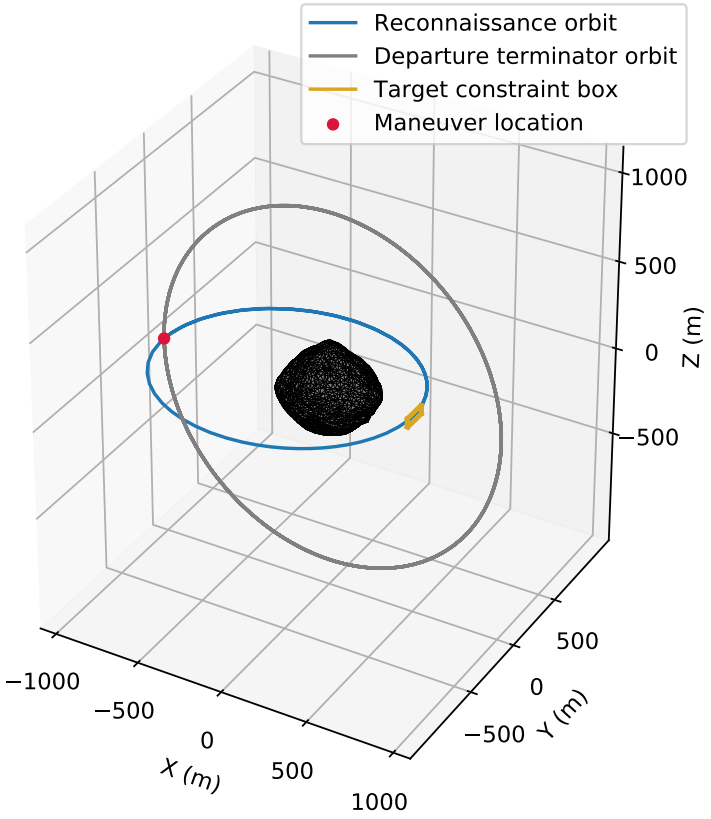

Figure 7.2: Asteroid maneuver targeting scenario

$J = \|\boldsymbol{u}_0\|$. An initial guess for the control vector is set to $\boldsymbol{u}_{0,\text{guess}} = \begin{bmatrix} 0.0 & 0.058 & 0.072111 \end{bmatrix}$ m/s, which helps to achieve the constraint of $\dot{z}_f = 0$. In order to introduce significant nonlinearities into the trajectory, the final time t_f is set to lie at 1.5 revolutions of the reconnaissance orbit from the maneuver location. This corresponds to roughly 23.6 hours. At this target time, the uncertainty distribution becomes decidedly non-Gaussian when expressed in Cartesian coordinates.

The optimal control update $\boldsymbol{u}_0$ is first computed assuming that the state at time t_f obeys a Gaussian distribution, i.e., $\boldsymbol{x}_f \sim \mathcal{N}(\bar{\boldsymbol{x}}_f, P_f)$. For this we use the standard chance constraint formulation from Eq. 7.2, and the deterministic constraint conversion from Eq. 7.6. The problem is solved using the **SLSQP** solver in the SciPy **minimize** function. The optimizer successfully converges on a control solution that satisfies the chance constraint on the single Gaussian distribution.

We next approximate the initial state distribution using a GMM. Following Eqs. 7.9 and 7.10, the initial Gaussian distribution at time t_0 is split into 15 Gaussian distributions along the maximum stretching direction in position space (identified using the Cauchy-Green tensor for the designated time span). We will refer to this direction as ξ. The means of each of the distributions are shifted along this direction by a certain value. These values, along with the associated 1-σ uncertainties along this direction, and the mixand weights, are chosen to be the following:

$$\bar{\boldsymbol{x}}_{\xi,0} = \begin{bmatrix} 0.0 \\ 0.25 \\ -0.25 \\ 0.5 \\ -0.5 \\ 0.75 \\ -0.75 \\ 1.05 \\ -1.05 \\ 1.4 \\ -1.4 \\ 1.8 \\ -1.8 \\ 2.25 \\ -2.25 \end{bmatrix} \text{ m} \qquad \boldsymbol{\sigma}_{\xi} = \begin{bmatrix} 0.55 \\ 0.5 \\ 0.5 \\ 0.45 \\ 0.45 \\ 0.4 \\ 0.4 \\ 0.35 \\ 0.35 \\ 0.35 \\ 0.35 \\ 0.3 \\ 0.3 \\ 0.2208 \\ 0.2208 \end{bmatrix} \qquad \boldsymbol{\alpha} = \begin{bmatrix} 5/24 \\ 2.5/24 \\ 2.5/24 \\ 2.25/24 \\ 2.25/24 \\ 1.5/24 \\ 1.5/24 \\ 1/24 \\ 1/24 \\ 1/24 \\ 1/24 \\ 0.75/24 \\ 0.75/24 \\ 1/48 \\ 1/48 \end{bmatrix} \tag{7.27}$$

These values were manually selected using Eqs. 7.9 and 7.10. A more sophisticated mixture splitting method such as the one in Ref. [109] could also be used.

As a initial feasible guess for the risk allocation, the conservative risk allocation formulation from Eq. 7.21 is used to assign an unweighted risk value of $\delta^i_{f,j} = 0.05/(6 \times 15)$ to each mixand's associated individual chance constraints. We use the STT method for propagating the Gaussian mixtures, given by Eqs. 7.22 and 7.23 to significantly speed up the algorithm. The second-order STTs for the single-Gaussian solution were integrated and stored. The two-stage optimization procedure was then run using these STTs to propagate the mean and covariance for each mixand at each iteration of the algorithm. With a given fixed risk allocation, the inner optimal control problem is solved, using the Python **minimize** function. The outer IRA algorithm detailed in Algorithm 2 is then used to reallocate the risk associated with each mixand and chance constraint. This procedure is repeated until the change in cost from consecutive iterations is below a certain tolerance, at which point we stop the algorithm and assume that the risk has been optimally allocated. We use values of $\gamma = 0.7$, and we set the tolerance values to $\epsilon = \epsilon_{IRA} = 10^{-10}$. The

GMM/IRA algorithm successfully converged on a solution within the specified tolerances in 40 iterations and 3.68 seconds

1000 state perturbations were then sampled from the initial state uncertainty distribution, and the optimal controls for both the single Gaussian and GMM/IRA formulations were applied. These state vectors were propagated forward to verify that the chance constraints were successfully met. The resulting state distributions at time t_f are shown in Figs. 7.3-7.5. Zoomed-in plots for the single Gaussian and GMM/IRA algorithms are shown in Figs. 7.6 and 7.7. We can clearly see that for this scenario, the state distribution becomes non-Gaussian, and the single Gaussian chance constraint algorithm does not accurately capture the true distribution. On the other hand, the simple conservative risk allocation algorithm results in an overly conservative solution. The GMM/IRA algorithm gives a control that accurately satisfies the desired chance constraints.

The percent of Monte Carlo iterations that satisfy all of the chance constraints are given for each algorithm in Table 7.1. We can see that, in practice, the single-Gaussian method does not satisfy the desired chance constraint, due to the non-Gaussian nature of the final distribution. On the other hand, the GMM method with conservative risk allocation results in an overly conservative solution, satisfying the chance constraint to 99.2% probability rather than the desired 95%. This increased conservatism results in a slight increase in the control magnitude, which may be undesirable for spacecraft missions looking to conserve as much propellant mass as possible. While in this particular case, this increase in the control magnitude is relatively small, other scenarios could result in more significant savings. Finally, the GMM/IRA algorithm successfully satisfies the chance constraint to the desired probability of 95.0%. Clearly, it performs significantly better than the single Gaussian method, while being computationally tractable thanks to the GMM covariance propagation scheme with STTs.

7.6.2 State transition tensor approximation

In this section we will illustrate the benefits of using the tractable STT-based approximation for the lower-stage optimization algorithm developed in this chapter. We first run the same

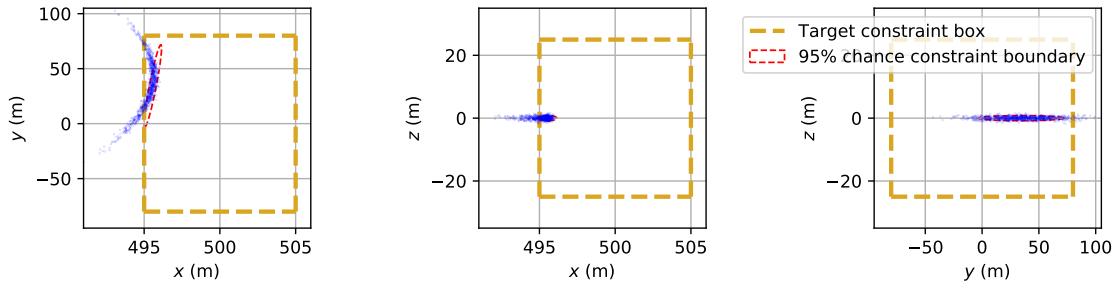

Figure 7.3: Chance constraint results for asteroid maneuver targeting scenario with single Gaussian

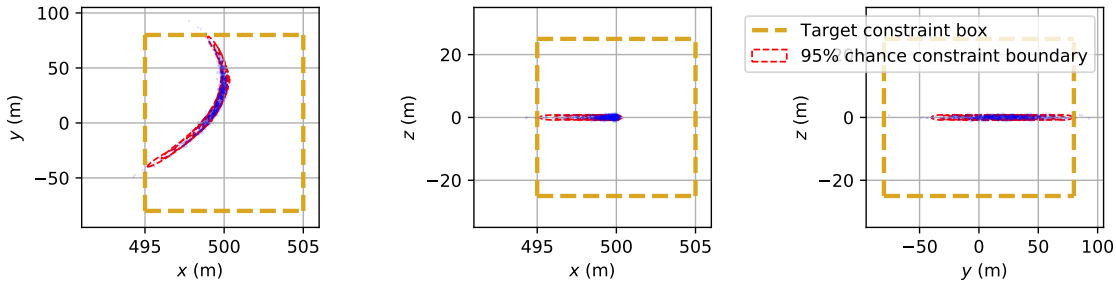

Figure 7.4: Chance constraint results for asteroid maneuver targeting scenario with GMM and conservative risk allocation

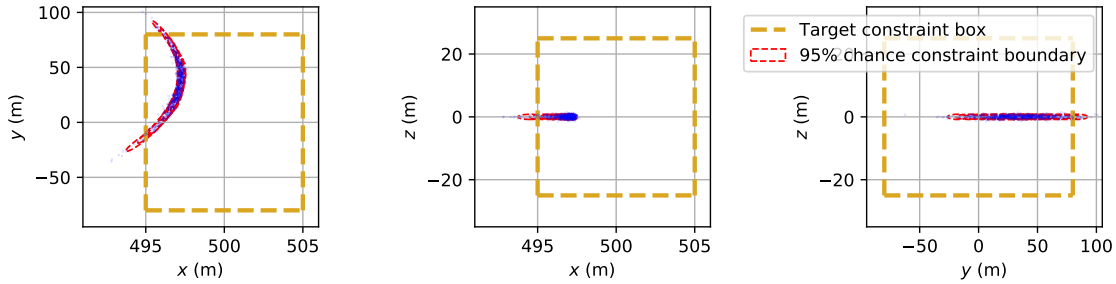

Figure 7.5: Chance constraint results for asteroid maneuver targeting scenario with GMM/IRA

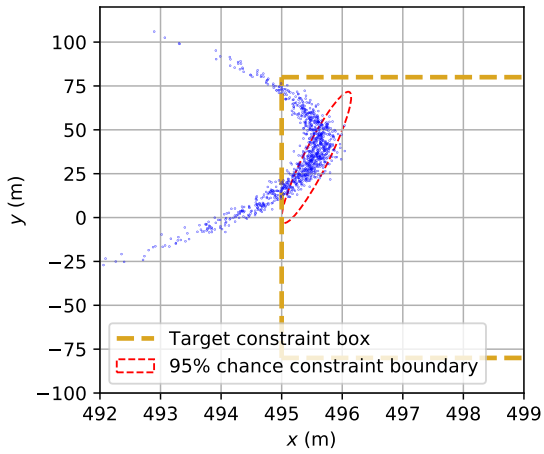

Figure 7.6: Results for single Gaussian (zoom)

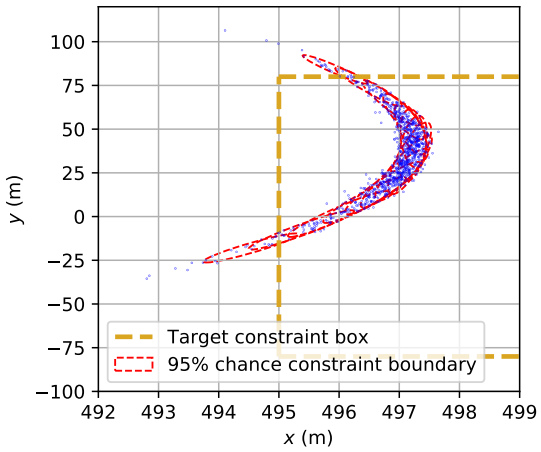

Figure 7.7: Results for GMM/IRA (zoom)

Table 7.1: Algorithm metrics for asteroid scenario with target chance constraint satisfaction of 95%

Algorithm	Monte-Carlo percent chance constraint satisfied	Optimal control magnitude (cm/s)
Single Gaussian	75.4%	9.301
GMM with cons. risk	99.2%	9.309
GMM with IRA	95.0%	9.304

Table 7.2: Comparison metrics for GMM IRA algorithm with and without STT approximation for GMM propagation

GMM propagation method	GMM IRA runtime (s)	Optimal control magnitude (cm/s)
Numerical integration	2004.52	9.304372
STTs	3.93	9.304396
STTs with analytical jacobians	3.68	9.304396

asteroid maneuver targeting scenario from the previous section using numerical integration for the lower-stage optimization. We then run the scenario using the STT approximation strategy both with and without analytical Jacobians provided to the optimizer. To do so, the second-order STTs of the optimal single-Gaussian trajectory are integrated and stored. These are then used to analytically perform all numerical integrations of the dynamics and all instances of linear covariance propagation, following Eqs. 2.14, 7.22, and 7.23. The computational times required to run the three methods (on a standard laptop computer) are given in Table 7.2.

From Table 7.2, we can see that the numerical integration and STT methods yield extremely similar optimal control values, meaning that the second-order STT GMM propagation methods is an accurate approximation of the numerically integrated GMM propagation method for this specific scenario. However, the STT method is over 500× faster than the numerical integration method, which demonstrates its very significant computational benefits.

7.7 Application: Low-Energy Europa Approach Trajectory

7.7.1 Scenario

We next apply the GMM IRA concept to a low-energy trajectory approaching the surface of Europa from a Jupiter-centric orbit. For this example, we use the circular restricted three-body

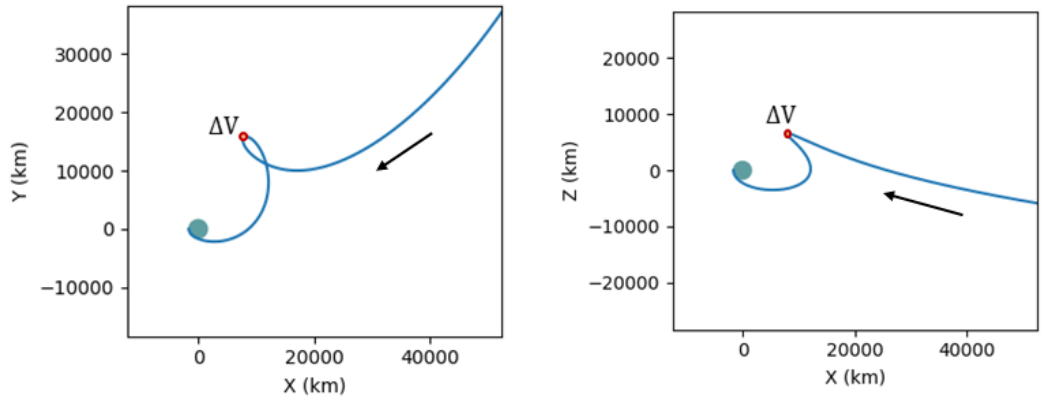

Figure 7.8: Low-energy approach trajectory to Europa, in Europa-centric rotating frame

problem (CR3BP) to approximate the dynamics of the Jupiter-Europa system (see Eqs. 2.7 - 2.9 for the equations of motion). For the Jupiter-Europa system, we use $\mu = 2.52801752854 \times 10^{-5}$.

This scenario is inspired by the results from Ref. [54]. More details on the trajectory generation strategy for the low-energy approach trajectory are given in Section 8.4.3. This low-energy trajectory provides a fuel-efficient strategy for accessing the surface of Europa, approaching from a 5:6 resonant orbit around Jupiter with respect to Europa. A baseline trajectory was generated that approaches Europa tangentially to the surface at an altitude of 50 km, following the trajectory generation strategy detailed in Ref. [54]. This trajectory is illustrated in Fig. 7.8. This specific low-energy trajectory has a loop on the L_2 side before arriving on the sub-Jovian side of Europa (i.e. facing Jupiter). This trajectory is highly nonlinear, which will result in highly non-Gaussian state distributions for any propagation arc of significant duration, even if the initial state distributions are Gaussian. A impulsive statistical maneuver is scheduled to occur at the tip of the loop. The maneuver at this location will be optimized to satisfy a non-Gaussian chance constraint at the final time. The propagation time between the ΔV location and the target time is set to be 1.512 non-dimensional CR3BP time units, or roughly 20.5 hours.

7.7.2 Radius chance constraint

In order to demonstrate the ability of the GMM IRA algorithm to satisfy a number of difference chance constraint formulations, we will investigate a constraint on the final distance from the center of Europa (r_2 in CR3BP notation), rather than a constraint in the Cartesian coordinate space. This constraint is nonlinear with respect to the position states. In order to express this constraint as a linear constraint, as required by Eq. 7.15, we can perform a first-order expansion of r_2 for each mixand. Given that $r_2 = \sqrt{(x-(1-\mu))^2+y^2+z^2}$, we can write the linear constraint for the i-th mixand as a function of its associated mean state $\boldsymbol{x}_f^i$:

$$\boldsymbol{h}_f^i = \begin{bmatrix} \frac{x^i-(1-\mu)}{r_2^i} & \frac{y^i}{r_2^i} & \frac{z^i}{r_2^i} & 0 & 0 & 0 \end{bmatrix} \tag{7.28}$$

The constraint direction is recomputed for each iteration of the optimization and GMM/IRA algorithm. We consider two different constraints which we will call Case 1 and Case 2. For Case 1 we seek to constrain the distance from the surface of Europa to be **smaller** than 50 km (i.e. $r_2 \leq 1611$ km, assuming a value of $r_{eur} = 1561$ km), which might be desirable in order to ensure successful imaging of the surface. For Case 2 we constrain the distance to be **larger** than 50 km (i.e. $r_2 \geq 1611$ km), which might be desirable to ensure mission safety. For both cases, the initial state uncertainties are assumed to be zero-mean and Gaussian with $\boldsymbol{\sigma}_0 = \begin{bmatrix} 2.0 & 2.0 & 2.0 & 0.1 & 0.1 & 0.1 \end{bmatrix}^T$ (km, mm/s). We set the risk threshold to $\Delta_f = 0.05$. Similar to the asteroid maneuver targeting scenario, we first solve the optimal control problem for the single-Gaussian cases. We then integrate and store the second-order STTs along these trajectories, and use them to rapidly run the GMM/IRA algorithm. To formulate the initial GMM, the initial Gaussian distribution is split into 15 Gaussian distributions along the maximum stretching direction in position space ξ, as with the asteroid maneuver targeting scenario. The means of each of the distributions are shifted along ξ by the following values:

$$\bar{\boldsymbol{x}}^g_{\xi,0} = [\, 0.0 \quad 0.5 \quad -0.5 \quad 1.0 \quad -1.0 \quad 1.5 \quad -1.5 \quad ...$$
$$2.1 \quad -2.1 \quad 2.8 \quad -2.8 \quad 3.6 \quad -3.6 \quad 4.5 \quad -4.5 \,]^T \text{ km} \tag{7.29}$$

The values for $\boldsymbol{\sigma}_\xi$ and $\boldsymbol{\alpha}$ were chosen to be the same as in Eq. 7.27. The conservative risk allocation was used to initially assign an unweighted risk value of $\delta_f^i = 0.05/15$ to each mixand's chance constraint. We use values of $\gamma = 0.7$ and $\epsilon = \epsilon_{IRA} = 5 \times 10^{-8}$. The GMM/IRA algorithm sucessfully converged on a solution within these tolerances in 12 iterations and 1.07 seconds for Case 1, and 17 iterations and 2.98 seconds for Case 2.

As with the asteroid scenario, 1000 state perturbations were sampled for each case from the initial state uncertainty distribution, and the optimal controls for both the single Gaussian and GMM/IRA formulations were applied. These state vectors were propagated forward to verify that the chance constraints were successfully met. The percent of Monte Carlo iterations that satisfy the radius chance constraint are given in Table 7.3 for each case. Clearly, the GMM/IRA method satisifies the chance constraint closest to the desired probability. Histogram plots for the achieved final radii for each Monte Carlo iteration are shown in Figs. 7.9-7.12 for the single-Gaussian and GMM/IRA algorithms. We can see that the GMM method more accurately captures the non-Gaussian distribution of the final achieved radius. The resulting state distributions in the $x - y$ plane are shown in Figs. 7.13 and 7.14.

7.8 Conclusions

This chapter presents a methodology to approximate non-Gaussian chance constraints as a conjunction of individual chance constraints on the individual mixands of a Gaussian mixture model (GMM). Iterative risk allocation (IRA) is then used to determine the optimal risk allocation for the mixands in order to reduce the degree of conservatism in the formulation. The resulting algorithm was first demonstrated on a spacecraft maneuver targeting scenario with a conjunction of linear chance constraints forming a box in Cartesian position space. Next, the algorithm was demonstrated

Table 7.3: Algorithm metrics for Europa scenario with target chance constraint satisfaction of 95%

Algorithm	Case 1 $[r_2 \leq (r_{eur} + 50 \text{ km})]$ percent chance constraint satisfied	Case 2 $[r_2 \geq (r_{eur} + 50 \text{ km})]$ percent chance constraint satisfied
Single Gaussian	74.2%	97.8%
GMM with cons. risk	97.7%	99.6%
GMM with IRA	95.0%	95.8%

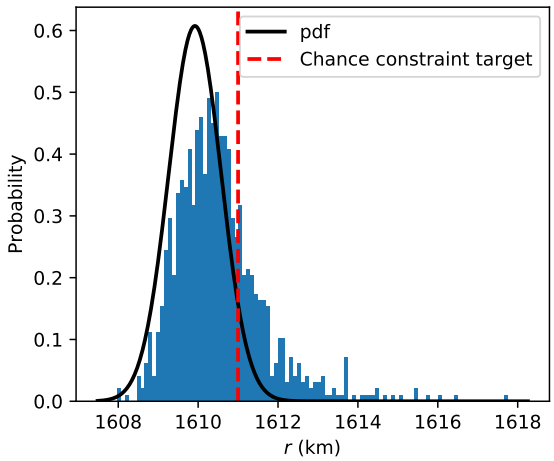

Figure 7.9: Case 1 $[r_2 \leq (r_{eur} + 50 \text{ km})]$ results for single Gaussian algorithm

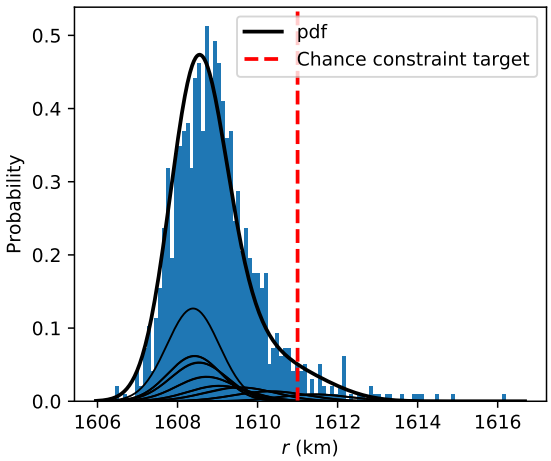

Figure 7.10: Case 1 $[r_2 \leq (r_{eur} + 50 \text{ km})]$ results for GMM IRA algorithm

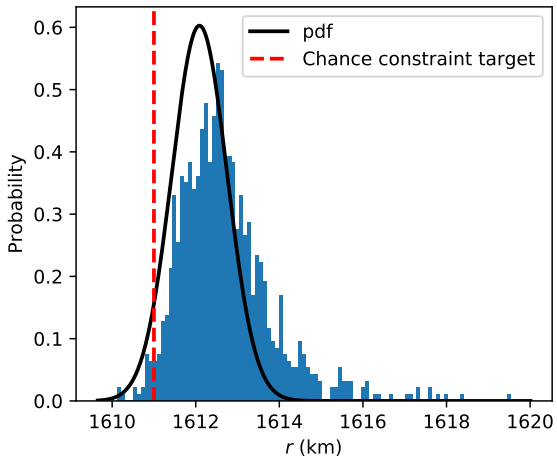

Figure 7.11: Case 2 [$r_2 \geq (r_{eur} + 50$ km)] results for single Gaussian algorithm

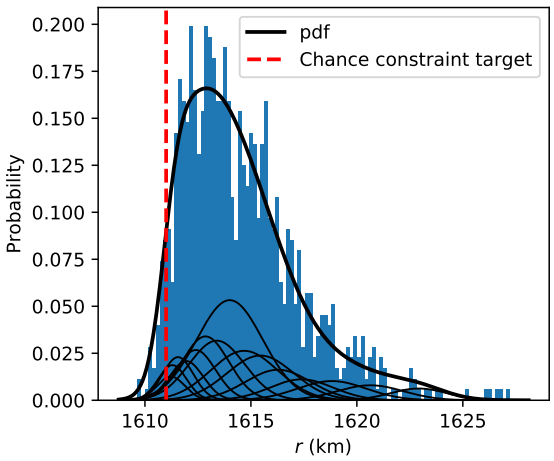

Figure 7.12: Case 2 [$r_2 \geq (r_{eur} + 50$ km)] results for GMM IRA algorithm

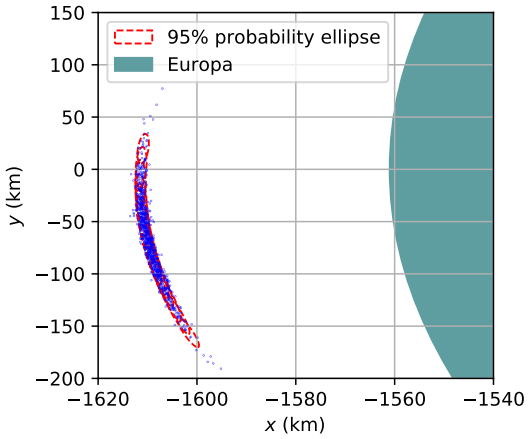

Figure 7.13: Case 1 [$r_2 \leq (r_{eur} + 50$ km)] final states for GMM IRA algorithm

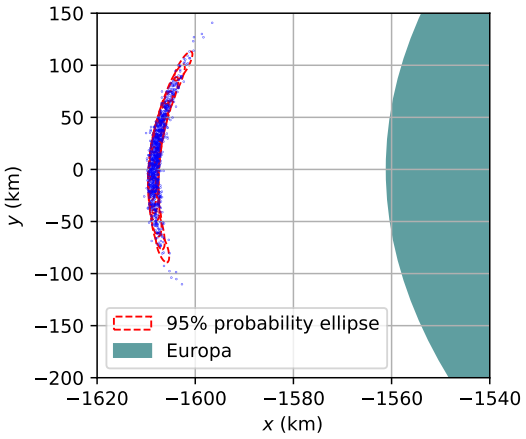

Figure 7.14: Case 2 [$r_2 \geq (r_{eur} + 50$ km)] final states for GMM IRA algorithm

on a low-energy trajectory approaching the surface of Europa with a chance constraint applied on the final radius. Both of these scenarios result in decidedly non-Gaussian state distributions. In both cases, the GMM chance constraint algorithm with IRA is shown to accurately satisfy the desired chance constraint boundaries, while the single Gaussian chance constraint algorithm does not. Beyond the applications presented in this chapter, this strategy could be used for collision avoidance maneuvers in both low-Earth orbits and cislunar space.

Chapter 8

Directional State Transition Tensors for Capturing Dominant Nonlinear Effects

8.1 Introduction

Higher-order methods such as state transition tensors (STTs) come with a significant tradeoff in increased storage requirements, and increased computational requirements for both computing the higher-order tensors and performing any subsequent mathematical operations involving them. These requirements increase exponentially as the maximum order of STT considered increases. This has limited their adoption in many operational settings. In order to address these issues, several strategies for approximating higher-order STTs have been developed [44, 91, 99]. For example, Fujimoto et al. [44] present a method for analytically computing the STTs for the perturbed two-body problem. In addition, Roa and Park [99] derive a method to approximate higher-order STTs by retaining only the dominant secular terms. The methods in these works are straightforward to implement and have been shown to reduce the computational time required to compute the STTs. However, they are restricted to applications in two-body, periodic dynamics. Several current, future, and conceptual missions, including the Europa lander concept [54], the Lunar Gateway [115], and the Sun-Earth libration point orbiting missions [26, 27], operate in orbits that do not meet these criteria. The existing approximation strategies are not always suitable for use in these dynamical regimes. Because these particular orbits are highly nonlinear, guidance, navigation and control operations for these spacecraft would greatly benefit from a method that can efficiently approximate higher-order effects.

Outside of the field of astrodynamics, a number of numerical methods have been developed for

decomposing tensors into a combination of smaller matrices or tensors. These methods include the Tucker decomposition, the parallel factors decomposition, and the canonical decomposition, among others [65]. These methods first appeared in psychometrics and chemometrics literature, and their use has expanded to other fields including signal processing [31] and computer vision [101]. However, these methods tend to be most useful for decomposing sparse tensors with very large dimensions (i.e. on the order of 100 or 1000). The STTs corresponding to a spacecraft orbit are not particularly sparse and have relatively small dimensions (e.g. a dimension of six for a standard Cartesian state). These methods therefore do not perform particularly well for approximating higher-order STTs in astrodynamics problems. In addition, since they are a numerical procedure, it is difficult to extract physical insight as to what information is lost in the decomposition process.

In this chapter we propose a novel method for approximating higher-order STTs which we call **directional state transition tensors** (DSTTs). The derivatives in DSTTs are taken with respect to a particular direction in the state, rather than a single state parameter (as with the standard STTs). In essence, they constitute an orthogonal transformation of the original STTs. By transforming the DSTTs to align with specific directions, we can maximize the amount of information contained in a small number of terms. This allows us to retain only these terms, and achieve a very good approximation of the effects of the full STTs while significantly reducing storage and computational requirements.

This chapter is organized as follows. In Section 8.2, we introduce the problem considered in this chapter - namely, that higher-order STTs can require storing and operating on many unnecessary terms. In Section 8.3, we derive the equations necessary to compute and utilize directional state transition tensors. We then present a strategy for finding a suitable basis for the DSTTs using the Cauchy-Green Tensor (CGT). Examples for using DSTTs for nonlinear state propagation around a reference are presented in Section 8.4 for several scenarios in two and three-body dynamics, including a spacecraft operating at a near-rectilinear halo orbit in the Earth-Moon system, and a spacecraft approaching Europa on a low-energy trajectory in the Jupiter-Europa system. In Section 8.5, we develop an efficient method to estimate the relative importance of the DSTT terms.

Finally, in Section 8.6, we derive the equations to use DSTTs to simplify the STT state covariance propagation equations, and apply these reduced equations to the various spacecraft scenarios in Section 8.7.

8.2 Size of State Transition Tensors

The p-th order STTs will contain n^{p+1} elements - for example, the STM for a standard state vector of size $n = 6$ will have thirty-six elements, the second-order STT will have $6^3 = 216$ elements, and the third-order STT will have $6^4 = 1296$ elements. Clearly, the number of elements increases exponentially as the maximum order p included in the approximation increases. This leads to a tradeoff in accuracy vs. storage and computational requirements when using STTs to approximate nonlinear dynamics. This has been a limiting factor in the adoption of STT-based uncertainty propagation, navigation, and control schemes in practical applications. For many scenarios, simply considering the first-order STM is sufficient to achieve the desired accuracy; however, for spacecraft operating in more complex dynamical systems, including the higher-order terms may become necessary or desirable.

The p-th order summation term in Eq. 2.14 can also be thought of as a sum of n^p terms, which we will refer to as $\delta\mathbf{x}_{f,\kappa_1\ldots\kappa_p}$. For example, the six first-order terms obtained from the STM are

$$\delta\mathbf{x}_{f,\kappa_1} = \sum_{i=1}^{n} \phi^{i,\kappa_1}\delta x_0^{\kappa_1} \tag{8.1}$$

for fixed $\kappa_1 \in [1, n]$ (i.e. not summing over the κ_1 index). For the standard Cartesian state vector $\mathbf{x} = [x\ \ y\ \ z\ \ \dot{x}\ \ \dot{y}\ \ \dot{z}]^T$, the six terms correspond to each of the variations in the final state as a result of variations in each of the initial states, e.g.

$$\delta\mathbf{x}_{f,\kappa_1} = \phi^{i,x}\delta x_0 + \phi^{i,y}\delta y_0 + \phi^{i,z}\delta z_0 + \phi^{i,\dot{x}}\delta\dot{x}_0 + \phi^{i,\dot{y}}\delta\dot{y}_0 + \phi^{i,\dot{z}}\delta\dot{z}_0 \tag{8.2}$$

Similarly, there will be thirty-six second-order terms obtained from the second-order STT:

$$\delta\mathbf{x}_{f,\kappa_1\kappa_2} = \sum_{i=1}^{n} \frac{1}{2}\phi^{i,\kappa_1\kappa_2}\delta x_0^{\kappa_1}\delta x_0^{\kappa_2} \tag{8.3}$$

for fixed $\kappa_1 \in [1,n]$ and $\kappa_2 \in [1,n]$. In general, for a normalized Cartesian state vector (where the position and velocity units are normalized to be roughly of the same order magnitude) and a typical spacecraft orbit, the magnitude of each of these terms will be roughly of the same order of magnitude for a given order p, i.e. for $p = 2$:

$$\mathcal{O}(\|\delta\mathbf{x}_{f,11}\|) \simeq \mathcal{O}(\|\delta\mathbf{x}_{12}\|) \simeq \mathcal{O}(\|\delta\mathbf{x}_{13}\|) \simeq \mathcal{O}(\|\delta\mathbf{x}_{22}\|) \simeq \mathcal{O}(\|\delta\mathbf{x}_{33}\|) \simeq \mathcal{O}(\|\delta\mathbf{x}_{44}\|), \text{ etc.} \tag{8.4}$$

meaning that all thirty-six terms will be needed in order to accurately compute the full second-order nonlinear effects. If any of these terms is neglected in the approximation, the errors would be expected to be of $\mathcal{O}(\|\delta\mathbf{x}_{f,11}\|)$. However, if one (or more) of these terms is significantly larger than all other terms, for example,

$$\mathcal{O}(\|\delta\mathbf{x}_{f,11}\|) \gg \mathcal{O}(\|\delta\mathbf{x}_{f,1\ldots6,2\ldots6}\|) \tag{8.5}$$

then the full second-order effects from Eq. 2.14 can be approximated as

$$\frac{1}{2}\phi^{i,\kappa_1\kappa_2}\delta x_0^{\kappa_1}\delta x_0^{\kappa_2} \simeq \frac{1}{2}\phi^{i,11}\delta x_0^1\delta x_0^1 \tag{8.6}$$

For this example, by using this approximation, the number of elements required in the second-order STT is reduced from 216 to six. This represents a significant reduction in the storage requirements for the second-order STT, and in the computational requirements for any operations using the STTs.

In most scenarios, it is unlikely that using a Cartesian state representation will result in the term along a particular direction dominating over the other terms (as required in Eq. 8.5). We will therefore introduce the concept of **directional** state transition tensors (DSTTs), in which the STT derivatives are taken with respect to a rotated orthogonal basis that is formed through a linear combination of the Cartesian state coordinates. In nonlinear dynamics, there are often one or two particular directions that significantly contribute to final state deviations. These are especially

prominent in scenarios where linearized dynamics are insufficiently accurate and where including higher-order terms may become necessary.

8.3 Directional State Transition Tensors

8.3.1 Derivation

STM and STT derivatives are typically taken with respect to variations in the state vector $\mathbf{x} \in \mathbb{R}^n$, in which the dynamics equations are expressed. These STTs can be rotated such that the derivatives are taken with respect to an alternative basis $\mathbf{y} \in \mathbb{R}^n$ with no loss in information so long as $\mathbf{y}$ constitutes an orthogonal basis that spans $\mathbb{R}^n$. These bases are related through the transformation $\mathbf{y} = R\mathbf{x}$, where R corresponds to a linear transformation matrix. If using index notation, the relation can be written as $y^{\gamma_1} = R^{\gamma_1,\kappa_1} x^{\kappa_1}$. By using the chain rule, the directional STM with respect to $\mathbf{y}$ can thus be calculated as

$$\phi^{i,\gamma_1} = \frac{\partial x_f^i}{\partial y_0^{\kappa_1}} = \frac{\partial x_f^i}{\partial x_0^{\kappa_1}} \frac{\partial x_0^{\kappa_1}}{\partial y_0^{\gamma_1}} = \phi^{i,\kappa_1} R^{\gamma_1,\kappa_1} \tag{8.7}$$

where we use the $\gamma_1 ... \gamma_p$ indices to refer to derivatives taken with respect to the rotated basis $\mathbf{y}$. All directional STTs will have these indices as superscripts.

The higher-order DSTTs can be expressed as a function of the standard STTs $\phi^{i,\kappa_1 \cdots \kappa_p}$ and the transformation matrix R^{γ_1,κ_1}. We will assume all derivatives are taken with respect to the same basis $\mathbf{y}$. This does not necessarily have to be the case, though we note that the symmetry properties along the $\gamma_1 ... \gamma_p$ axes would be lost if the higher-order derivatives were taken with respect to different basis vectors. The chain rule can be used following the procedure in Eq. 8.7 to obtain the second-order DSTT, noting that all higher-order derivatives of $\mathbf{x}_0$ with respect to $\mathbf{y}_0$ vanish since R is a linear transformation matrix:

$$\begin{aligned} \phi^{i,\gamma_1\gamma_2} = \frac{\partial^2 x_f^i}{\partial y_0^{\kappa_1} y_0^{\kappa_2}} &= \frac{\partial^2 x_f^i}{\partial x_0^{\kappa_1} x_0^{\kappa_2}} \frac{\partial x_0^{\kappa_1}}{\partial y_0^{\gamma_1}} \frac{\partial x_0^{\kappa_2}}{\partial y_0^{\gamma_2}} \\ &= \phi^{i,\kappa_1\kappa_2} R^{\gamma_1,\kappa_1} R^{\gamma_2,\kappa_2} \end{aligned} \tag{8.8}$$

Thus, we obtain the following equations for the second through fourth order DSTTs:

$$\phi^{i,\gamma_1\gamma_2} = \phi^{i,\kappa_1\kappa_2} R^{\gamma_1,\kappa_1} R^{\gamma_2,\kappa_2} \tag{8.9}$$

$$\phi^{i,\gamma_1\gamma_2\gamma_3} = \phi^{i,\kappa_1\kappa_2\kappa_3} R^{\gamma_1,\kappa_1} R^{\gamma_2,\kappa_2} R^{\gamma_3,\kappa_3} \tag{8.10}$$

$$\phi^{i,\gamma_1\gamma_2\gamma_3\gamma_4} = \phi^{i,\kappa_1\kappa_2\kappa_3\kappa_4} R^{\gamma_1,\kappa_1} R^{\gamma_2,\kappa_2} R^{\gamma_3,\kappa_3} R^{\gamma_4,\kappa_4} \tag{8.11}$$

Eq. 2.14 can be restated using the DSTTs as

$$\delta x_f^i \simeq \sum_{p=1}^{m} \frac{1}{p!} \phi^{i,\gamma_1\ldots\gamma_p}_{(t_f,t_0)} \delta y_0^{\gamma_1} \ldots \delta y_0^{\gamma_p} \tag{8.12}$$

Eq. 8.12 is equivalent to Eq. 2.14 if $\mathbf{y}$ is an orthogonal basis that spans $\mathbb{R}^n$.

8.3.2 Basis reduction

In order to decrease the number of terms required to obtain a sufficiently accurate approximation, the dimension of the basis $\mathbf{y}$ can be reduced to $\mathbf{y} \in \mathbb{R}^k$, with $k < n$. Eqs. 8.7 and 8.9- 8.11 can still be used to compute the reduced DSTTs with respect to the reduced basis, so long as the components of $\mathbf{y}$ are orthogonal. If the desired reduced basis $\mathbf{y}$ is known prior to integration of the STTs, the DSTTs can also be directly integrated without requiring any modifications to the dynamics equations. If the number of elements in the reduced DSTT is significantly less than the number of elements in the full STT of corresponding order, then the number of equations to be integrated can be greatly reduced. For example, for $n = 6$ and $k = 1$, the second-order DSTT will have $6 \times 1 \times 1 = 6$ elements. In this case, only six differential equations would need to be integrated to obtain the desired second-order elements, instead of the full 216.

The directional state transition tensors can be directly integrated by computing the directional state transition matrix ϕ^{i,γ_1} at each integration time step. The DSTT differential equations

can be obtained using the chain rule; to demonstrate this we will explicitly write out the differential equation to integrate the second-order DSTT:

$$\dot{\phi}^{i,\gamma_1\gamma_2} = \frac{\partial^2 \dot{x}^i}{\partial y_0^{\gamma_1} \partial y_0^{\gamma_2}} = \frac{\partial \dot{x}^i}{\partial x^\alpha} \frac{\partial^2 x^\alpha}{\partial y_0^{\gamma_1} \partial y_0^{\gamma_2}} + \frac{\partial^2 \dot{x}^i}{\partial x^\alpha \partial x^\beta} \frac{\partial x^\alpha}{\partial y_0^{\gamma_1}} \frac{\partial x^\beta}{\partial y_0^{\gamma_2}} \tag{8.13}$$

$$= A^{i,\alpha} \phi^{\alpha,\gamma_1\gamma_2} + A^{i,\alpha\beta} \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2} \tag{8.14}$$

where α and β are extra internal indices used to carry out the chain rule multiplications, and ϕ^{i,γ_1} can be obtained from Eq. 8.7. In these equations, the A tensors represent the partial derivatives of the state rates with respect to the state, in Cartesian space (i.e. $A^{i,\alpha} = \partial \dot{x}^i / \partial x^\alpha$, where $\dot{\mathbf{x}}$ is the state rate vector). These are the same A tensors that would be used to integrate the standard Cartesian STTs. Following this procedure, the associated differential equations to directly integrate the DSTTs (up to fourth order) are

$$\dot{\phi}^{i,\kappa_1} = A^{i,\alpha} \phi^{\alpha,\kappa_1} \tag{8.15}$$

$$\dot{\phi}^{i,\gamma_1\gamma_2} = A^{i,\alpha} \phi^{\alpha,\gamma_1\gamma_2} + A^{i,\alpha\beta} \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2} \tag{8.16}$$

$$\dot{\phi}^{i,\gamma_1\gamma_2\gamma_3} = A^{i,\alpha} \phi^{\alpha,\gamma_1\gamma_2\gamma_3} + A^{i,\alpha\beta} \left(\phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2\gamma_3} + \phi^{\alpha,\gamma_1\gamma_2} \phi^{\beta,\gamma_3} + \phi^{\alpha,\gamma_1\gamma_3} \phi^{\beta,\gamma_2} \right) + A^{i,\alpha\beta\lambda} \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2} \phi^{\lambda,\gamma_3} \tag{8.17}$$

$$\begin{aligned}
\dot{\phi}^{i,\gamma_1\gamma_2\gamma_3\gamma_4} = {} & A^{i,\alpha} \phi^{\alpha,\gamma_1\gamma_2\gamma_3\gamma_4} + A^{i,\alpha\beta} \Big(\phi^{\alpha,\gamma_1\gamma_2\gamma_3} \phi^{\beta,\gamma_4} + \phi^{\alpha,\gamma_1\gamma_2\gamma_4} \phi^{\beta,\gamma_3} + \phi^{\alpha,\gamma_1\gamma_3\gamma_4} \phi^{\beta,\gamma_2} \\
& + \phi^{\alpha,\gamma_1\gamma_2} \phi^{\beta,\gamma_3\gamma_4} + \phi^{\alpha,\gamma_1\gamma_3} \phi^{\beta,\gamma_2\gamma_4} + \phi^{\alpha,\gamma_1\gamma_4} \phi^{\beta,\gamma_2\gamma_3} + \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2\gamma_3\gamma_4} \Big) \\
& + A^{i,\alpha\beta\lambda} \Big(\phi^{\alpha,\gamma_1\gamma_2} \phi^{\beta,\gamma_3} \phi^{\lambda,\gamma_4} + \phi^{\alpha,\gamma_1\gamma_3} \phi^{\beta,\gamma_2} \phi^{\lambda,\gamma_4} + \phi^{\alpha,\gamma_1\gamma_4} \phi^{\beta,\gamma_2} \phi^{\lambda,\gamma_3} \\
& + \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2\gamma_3} \phi^{\lambda,\gamma_4} + \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2\gamma_4} \phi^{\lambda,\gamma_3} + \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2} \phi^{\lambda,\gamma_3\gamma_4} \Big) \\
& + A^{i,\alpha\beta\lambda\delta} \phi^{\alpha,\gamma_1} \phi^{\beta,\gamma_2} \phi^{\lambda,\gamma_3} \phi^{\delta,\gamma_4}
\end{aligned} \tag{8.18}$$

Again, if $k < n$, then the time required to numerically integrate the directional STTs can be greatly reduced when compared to the full STTs.

For this work, we will consider the case where the full first-order STM is computed with respect to the standard Cartesian basis $\mathbf{x} \in \mathbb{R}^n$, but all higher-order DSTTs are integrated with respect to the reduced rotated basis $\mathbf{y} \in \mathbb{R}^k$ with associated transformation matrix $R \in \mathbb{R}^{k \times n}$. Eq. 2.14 can then be approximated as:

$$\delta x_f^i \simeq \phi^{i,\kappa_1} \delta x_0^{\kappa_1} + \sum_{p=2}^{m} \frac{1}{p!} \phi_{(t_f,t_0)}^{i,\gamma_1\gamma_2\ldots\gamma_p} \delta y_0^{\gamma_1} \delta y_0^{\gamma_2} \ldots \delta y_0^{\gamma_p} \tag{8.19}$$

In this approximation, we retain all first-order information, and only neglect certain higher-order terms. Using the form in Eq. 8.19 might be desirable if DSTTs are being used with existing operational software that already uses the standard STM.

8.3.3 Finding a suitable basis

The next step is to find a suitable basis for $\mathbf{y}$. The objective is to find a basis that transforms the state vector such that the maximum amount of information about the higher-order effects is contained in the fewest number of terms. The simplest approach is to find a direction with some physical meaning, along which any perturbation is known to cause a large deviation from the reference trajectory at a later time. For example, in two-body dynamics, for a long propagation time, any perturbation to the along-track velocity will cause a shift in the period from the reference trajectory, and thus lead to a large deviation downstream in the trajectory.

An alternative approach is to use concepts from dynamical systems theory to determine the directions of maximum sensitivity along an orbit without requiring any prior insight into the behavior of the system. This approach is useful for complex orbits for which it is difficult to intuitively determine the directions of maximum deviation, such as orbits in multi-body dynamical systems. We make use of the **Cauchy-Green Tensor** [51, 102, 84] (CGT), which corresponds to

the first-order state transition matrix multiplied by its transpose:

$$C_{(t_f,t_0)} = \Phi^T_{(t_f,t_0)}\Phi_{(t_f,t_0)} \tag{8.20}$$

As in previous sections, for clarity we omit the subscripts corresponding to the timespan, with the understanding that all refer to mappings from an initial time t_0 to a final time t_f. The CGT gives information about the magnitude of the distance from the nominal motion at time t_f given an initial perturbation at time t_0 [51]

$$\|\delta\mathbf{x}_f\|^2 = \delta\mathbf{x}_0^T C \delta\mathbf{x}_0 \tag{8.21}$$

The eigenvalues λ_{ξ_γ} and associated eigenvectors $\boldsymbol{\xi}_\gamma$ of the CGT exactly relate to the magnitude of the first-order term corresponding to each eigenvector:

$$\|\Phi\boldsymbol{\xi}_\gamma\| = \sqrt{\lambda_{\xi_\gamma}}\|\boldsymbol{\xi}_\gamma\| \tag{8.22}$$

Since the CGT is by definition a real symmetric matrix, its eigenvalues are real, and its eigenvectors are real and orthogonal. Thus, its eigenvectors $\boldsymbol{\xi}_\gamma$ form an orthogonal basis that spans $\mathbb{R}^n$, and its eigenvalues present useful information about the magnitude of the first-order STM terms with respect to this basis. If the eigenvalue corresponding to one particular eigenvector of the CGT is significantly larger than the others, e.g. $\lambda_{\xi_1} \gg \lambda_{\xi_{2\ldots n}}$, then the first-order term along this direction will be significantly larger than the others, assuming the components of the perturbation vector $\delta\mathbf{x}_0$ are of similar magnitude.

We will show through examples in later sections that the orthogonal basis obtained using the eigendecomposition of the CGT also provides a useful basis for the higher-order DSTTs. Though the eigenvalues of the CGT only provide exact information about the magnitude of the first-order terms, we found that if there is a particularly sensitive direction (e.g. $\lambda_{\xi_1} \gg \lambda_{\xi_{2\ldots n}}$), the magnitude of the second and higher-order terms along this direction will generally also be significantly larger than the other terms. This property will be demonstrated numerically on an example scenario in Section 8.4.2. Another potential strategy to identify the most sensitive directions would be to

utilize measures of nonlinearity [61] rather than the first-order CGT, though this is not investigated in this work.

In the case where the DSTTs are computed strictly along the CGT eigenvector $\boldsymbol{\xi}_1$ corresponding to the maximum CGT eigenvalue λ_{ξ_1}, then the DSTT basis $\mathbf{y}$ reduces to a single direction ($k = 1$), and the DSTT transformation matrix R becomes $R = \boldsymbol{\xi}_1^T$, where $R \in \mathbb{R}^{1\times n}$. In order to simplify the notation in later sections, when computing DSTTs strictly along the direction $\boldsymbol{\xi}_1$, we write these DSTTs as $\boldsymbol{\psi}_{[1]}, \boldsymbol{\psi}_{[2]}, \boldsymbol{\psi}_{[3]}, \boldsymbol{\psi}_{[4]}$, where $_{[2]}$ refers to the second-order DSTT along $\boldsymbol{\xi}_1$. Note that each $\boldsymbol{\psi}_{[p]}$ is reduced from a $(p+1)$-dimensional matrix or tensor with n^{p+1} elements, to a vector of size n. Using the notation from the previous section, this means that

$$\psi^i_{[1]} = \phi^{i,\xi_1} \tag{8.23}$$

$$\psi^i_{[2]} = \phi^{i,\xi_1\xi_1} \tag{8.24}$$

$$\psi^i_{[3]} = \phi^{i,\xi_1\xi_1\xi_1} \tag{8.25}$$

$$\psi^i_{[4]} = \phi^{i,\xi_1\xi_1\xi_1\xi_1} \tag{8.26}$$

Depending on the scenario, there may be more than one important CGT eigenvector direction. In this case, if we wish to include the DSTTs computed along the directions $\boldsymbol{\xi}_1$ through $\boldsymbol{\xi}_k$, the DSTT transformation matrix R can be expressed as $R = [\boldsymbol{\xi}_1 \ ... \ \boldsymbol{\xi}_k]^T$, with $R \in \mathbb{R}^{k\times n}$.

8.4 Examples: Orbit State Propagation using DSTTs

8.4.1 Two-body dynamics

We first present a simple example in two-body dynamics to showcase the efficiency of the DSTT concept, similar to the initial example provided in Ref. [99]. In this example, a spacecraft is placed in a reference circular terminator orbit on the $x-z$ plane around a small asteroid ($\mu = 5.2$ m^3/s^2, $r_{sc} = 1000$ m, $r_{\text{asteroid}} = 246$ m). We use two-body dynamics for this simplified scenario. Position units are normalized relative to the radius of the asteroid, and time units are normalized relative to the mean motion on the asteroid's surface, such that $\mu_{\text{norm}} = 1$.

The reference state is integrated forward for a number of periods, along with the reference STM and second-order DSTT along $\boldsymbol{\xi_1}$ (computed using the CGT). A string of samples forming a circular pattern in position deviation in the $x-z$ plane around the reference is generated, keeping velocity states equal to the reference velocity. These points' states are integrated forward for a number of periods N_{period}. The final position state computed through this numerical integration is compared to the final state computed using only the first-order STM, and using the second-order DSTT along $\boldsymbol{\xi_1}$. The projection of the final position states on the $x-z$ plane are shown for a range of N_{period} values in Fig. 8.1. Similar to Ref. [99], these projections are rotated to the $X'-Z'$ axes so that the X' direction is aligned with the principal axis of the pattern predicted by the linear expansion. Despite only containing six more elements than the STM, we can clearly see that the second-order DSTT along $\boldsymbol{\xi_1}$ significantly improves the accuracy of the state propagation when compared to the STM, and successfully captures the most important nonlinearities.

8.4.2 Earth-Moon halo orbit

We first consider a periodic orbit in the Earth-Moon system which is an example of a near-rectilinear halo orbit (NRHO). This is the planned regime for long-term operations of the planned Lunar Gateway outpost [115]. The initial conditions for this orbit are given in Table 8.1, and the orbit is illustrated in Fig. 8.2. NRHOs are characterized by a relatively small perilune radius, ranging from approximately 1850 km to 17350 km [51]; for this specific orbit, the perilune radius is around 2000 km. Despite the overall stability of the NRHO family, this perilune region is particularly sensitive due to the very large velocities, and linear approximation methods may suffer in accuracy at these points. Including higher-order STTs in the approximations will improve the predictions; however, as stated before, this can come at a tremendous increase in storage requirements and computational costs. We will show that for propagating a spacecraft's perturbed state to the perilune region of the reference orbit, including only higher-order terms along the $\boldsymbol{\xi_1}$ direction yields state errors with a similar order of magnitude to the full STTs.

In order to demonstrate this, the reference state and STM were propagated for 1.5 orbits

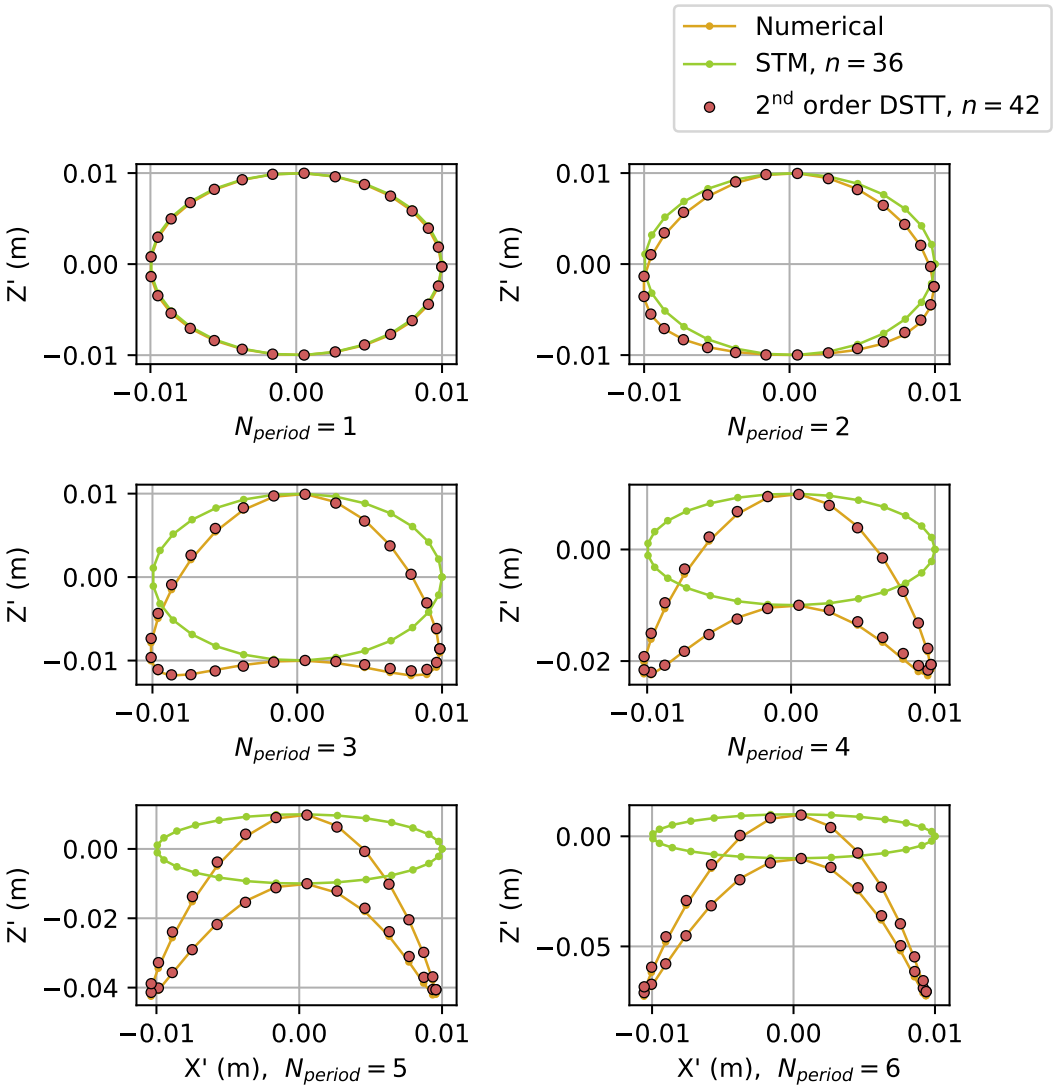

Figure 8.1: Evolution of states in two-body asteroid orbiting scenario

Table 8.1: Earth-Moon NRHO scenario parameters (non-dimensional)

Parameter	Near-Rectilinear Halo Orbit
μ	0.0121505856
x_0	1.013417655693384
y_0	0.0
z_0	-0.175374764978708
$\dot{x}_0$	0.0
$\dot{y}_0$	-0.083721347178432
$\dot{z}_0$	0.0
T	1.396265

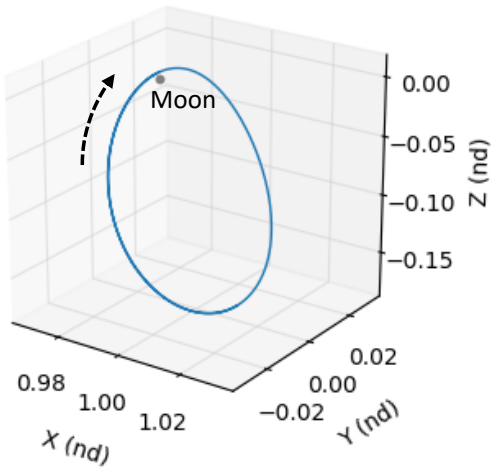

Figure 8.2: NRHO plotted in rotating Earth-Moon CR3BP frame

(to $t_f \simeq 2.094397$). The CGT for this trajectory arc was computed, and an eigendecomposition of the CGT was performed to obtain a suitable orthogonal basis for the DSTTs. For reference, the eigenvalues of the CGT computed from t_0 to t_f for this trajectory are

$$\lambda_{\boldsymbol{\xi}} \in \left\{ 5.6 \times 10^{7}, \quad 1.7 \times 10^{4}, \quad 7.4 \times 10^{2}, \quad 4.1 \times 10^{-4}, \quad 1.9 \times 10^{-4}, \quad 1.8 \times 10^{-8} \right\} \tag{8.27}$$

Clearly, there is a large discrepancy in the expected magnitudes of the terms along each direction, with λ_{ξ_1} being several orders of magnitude larger than all others. We would expect any STT terms computed strictly along this direction to dominate over others of the same order.

The STTs and DSTTs up to order $m = 3$ were then integrated for the reference trajectory. These were used to analytically propagate 1000 perturbed state vectors in the vicinity of the reference, with the initial state deviations sampled from a zero-mean normal distribution with the following 3-σ values for the state components (in non-dimensional CR3BP units):

$$\sigma = \begin{bmatrix} 2.5 \times 10^{-5} & 2.5 \times 10^{-5} & 2.5 \times 10^{-5} & 1 \times 10^{-5} & 1 \times 10^{-5} & 1 \times 10^{-5} \end{bmatrix}^T \tag{8.28}$$

These roughly correspond to the expected 3-σ navigation errors for a spacecraft operating in cislunar space [79]. For each case, the perturbed state was also numerically integrated, and the accuracy of the various orders of STT and DSTT approximations was compared. First, we will show the benefit of rotating the STTs onto the orthogonal basis obtained from the first-order CGT. The magnitude of all thirty-six second-order terms was computed for all 1000 perturbed state vectors using both the STTs in Cartesian state space, and the DSTTs computed with respect to the rotated basis. The magnitudes of the terms along each combination of axes/directions are shown in Fig. 8.3 for the Cartesian STTs and Fig. 8.4 for the DSTTs. In these Figures, "x, y" refers to the second-order term along the x and y axes (e.g., $\phi^{i,xy}\delta x_0 \delta y_0$). Similarly, "$\xi_1, \xi_2$" refers to the second-order term along the ξ_1 and ξ_2 directions from the CGT eigenvector basis (corresponding to the λ_{ξ_1} and λ_{ξ_2} eigenvalues, where $\lambda_{\xi_1} > \lambda_{\xi_2} > ... > \lambda_{\xi_6}$).

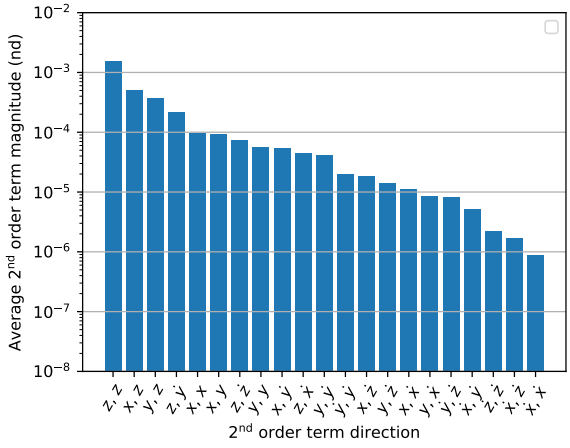

Figure 8.3: Second-order term magnitudes for NRHO scenario using Cartesian STTs

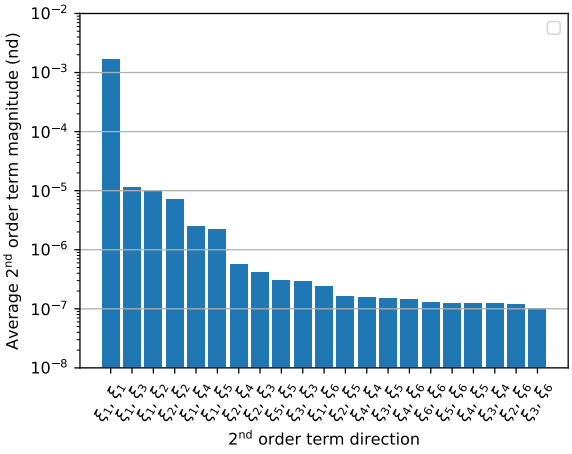

Figure 8.4: Second-order term magnitudes for NRHO scenario using DSTTs

Recall from Eq. 8.5 that we are seeking to find one ore more directions along which the higher-order terms will be orders of magnitude larger than all other terms. This can allow us to ignore all other terms in the approximation without a significant loss in accuracy. From Fig. 8.3, we can see that the largest second-order terms in Cartesian space all have roughly similar orders of magnitude, meaning that all these terms are needed to obtain an accurate second-order approximation. On the other hand, we can see from Fig. 8.4 that the second-order term along $\boldsymbol{\xi}_1$ in the rotated basis is around two orders of magnitude larger than all other second-order terms in this basis (10^{-3} vs. 10^{-5}). Thus, Eq. 8.6 can be used to approximate the effects of the full second-order STTs using only the second-order terms along the $\boldsymbol{\xi}_1$ direction (e.g. $\frac{\partial^2 \mathbf{x}_f}{\partial \xi_1 \partial \xi_1}$, or $\boldsymbol{\psi}_{[2]}$ using the notation from Eq. 8.24). We can then expect errors on the order of 10^{-5} from using this approximation to propagate the state deviations, with respect to using the full second-order STT.

The same procedure can be conducted for the third-order terms. For the same NRHO scenario, the magnitude of all 216 third-order terms was computed for all 1000 perturbed state vectors using both the Cartesian STTs and the CGT eigenvector DSTTs. The magnitudes of the third-order terms along the combinations of axes/directions corresponding to the 20 largest magnitudes are shown in Fig. 8.5 for the Cartesian STTs and Fig. 8.6 for the DSTTs. Again, the largest third-order terms in Cartesian space all have roughly similar orders of magnitude. On the other hand, the third-order term along $\boldsymbol{\xi}_1$ in the DSTT basis is around two orders of magnitude larger than all other third-order terms using this basis. We can therefore approximate the effects of the full third-order STT using only the six third-order terms along this ξ_1 direction (i.e. $\boldsymbol{\psi}_{[3]}$), reducing the number of elements required from 1296 to six. We would expect errors with respect to the full third-order effects on the order of 10^{-6}. Note that the average magnitude of the third-order term along $\boldsymbol{\xi}_1$ is larger than all of the average magnitudes of the second-order terms along the other directions (see Fig. 8.4) - this serves to show that scenarios exist where higher-order terms along $\boldsymbol{\xi_1}$ may be more useful to consider in an approximation than lower-order terms along different directions.

Next, the accuracy of using the STTs and DSTTs to map initial state deviations to the final

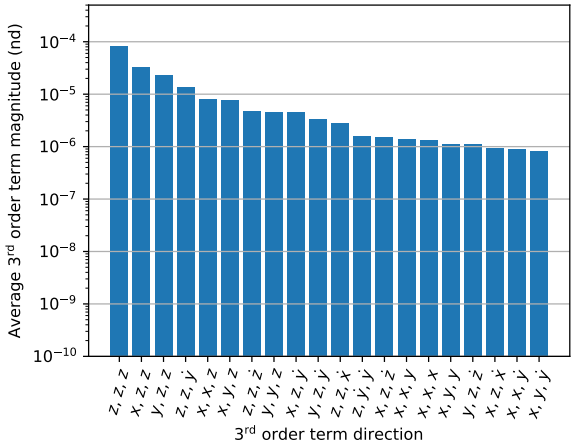

Figure 8.5: Third-order term magnitudes for NRHO scenario using Cartesian STTs

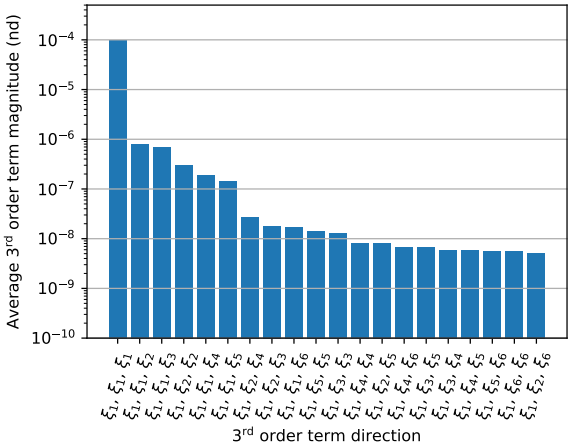

Figure 8.6: Third-order term magnitudes for NRHO scenario using DSTTs

Table 8.2: Average final state errors for STT/DSTT propagation of 1000 perturbed trajectories around NRHO reference

Order/basis	Number of additional elements	Average final state error
Full 1^{st}	36	1.692×10^{-3}
Full 1^{st} + full 2^{nd}	252	9.651×10^{-5}
Full 1^{st} + 2^{nd} DSTT	42	1.001×10^{-4}
Full 1^{st} + full 2^{nd} + full 3^{rd}	1548	6.014×10^{-6}
Full 1^{st} + full 2^{nd} + 3^{rd} DSTT	258	6.191×10^{-6}
Full 1^{st} + 2^{nd} DSTT + 3^{rd} DSTT	48	2.373×10^{-5}

time was compared. First, the state errors were computed using the full first, second, and third-order Cartesian STTs. The number of elements required in these approximations is $n = 36$, $n = 252$, and $n = 1548$, respectively. The state errors were computed using the first-order STM with the second-order DSTT only along $\boldsymbol{\xi}_1$ ($n = 42$), and using the full first and second-order STT with the third-order DSTT along $\boldsymbol{\xi}_1$ ($n = 258$). Finally, the state errors were computed using the full first-order STM, with the second and third-order DSTTs only along $\boldsymbol{\xi}_1$ ($n = 48$). The computed final state deviations were then compared to the true final state deviations obtained through numerical integration, and the approximation errors were computed. The state error magnitudes for all cases and iterations are plotted as a function of the size of the initial state deviation vector along $\boldsymbol{\xi}_1$ in Fig. 8.7. The average state error magnitude for all cases is given in Table 8.2. We can see that the DSTTs give nearly the same accuracy as the full STTs, but require significantly fewer elements to achieve this accuracy. This clearly illustrates the benefits of using higher-order DSTTs along particularly sensitive directions.

8.4.3 Europa lander scenario

We next consider the scenario of a spacecraft approaching the surface of Jupiter's moon Europa on a low-energy trajectory. For this scenario, the Jupiter-Europa CR3BP with $\mu = 2.52801752854 \times 10^{-5}$ is used to approximate the dynamics of the full Jupiter-Europa system. This scenario is inspired by the results from Ref. [54]. This low-energy trajectory provides a fuel-

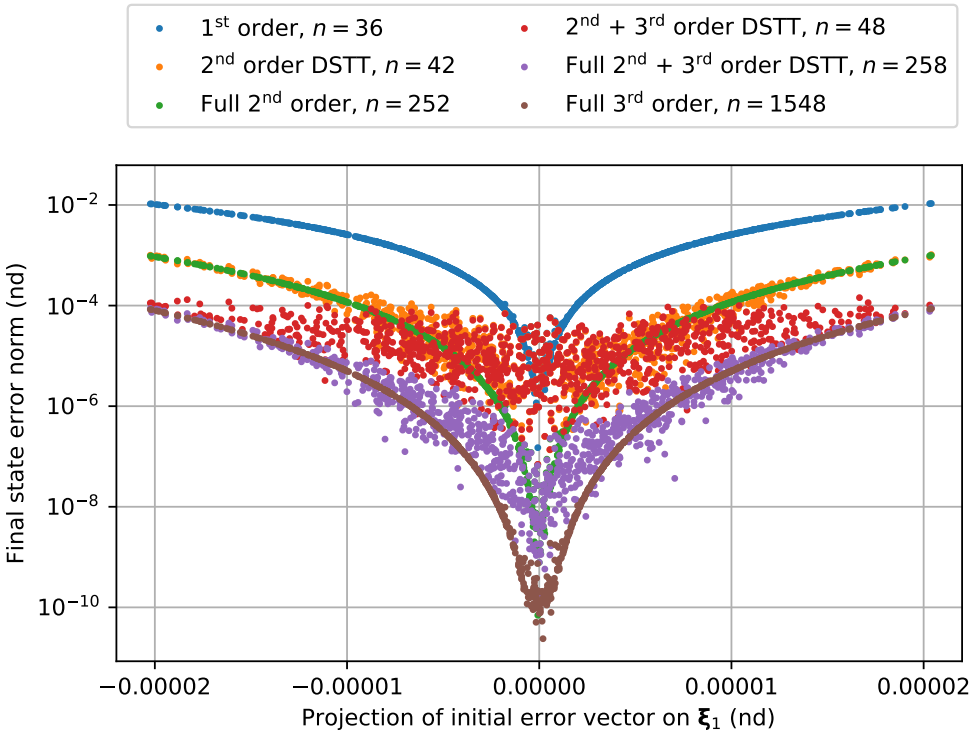

Figure 8.7: State errors for propagating 1000 perturbed trajectories in NRHO scenario using various orders of STTs and DSTTs

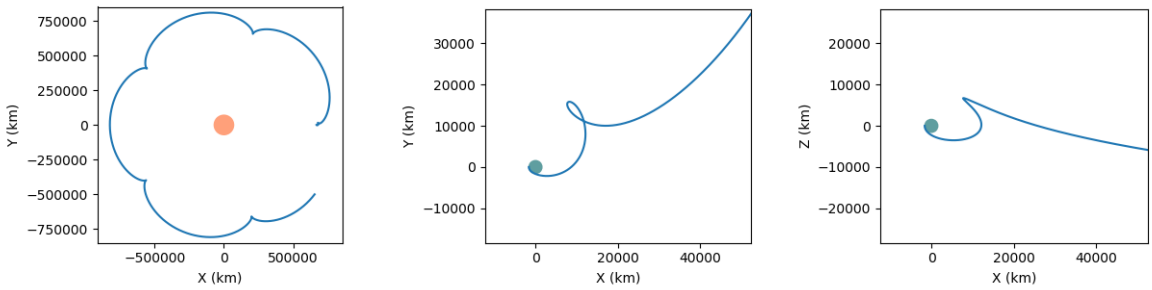

Figure 8.8: Low-energy approach trajectory to Europa, in Jupiter-centric (left) and Europa-centric (center, right) rotating frames

efficient strategy for accessing the surface of Europa, approaching from a 5:6 resonant orbit around Jupiter with respect to Europa. In Ref. [54], the authors found that linearized techniques were no longer valid for performing maneuver design and orbit determination for these types of trajectories. This is therefore a good example to apply the DSTT concept to efficiently improve the accuracy of the linearized approximation.

Following the trajectory generation strategy detailed in Ref. [54], a baseline trajectory was generated that approaches Europa tangentially to the surface at an altitude of 50 km. This trajectory is illustrated in Fig. 8.8. This specific low-energy trajectory has a loop on the L_2 side before arriving on the sub-Jovian side of Europa (i.e. facing Jupiter). This trajectory is highly nonlinear, non-periodic, and is sensitive to small perturbations. Practically, in order to remain on or near this baseline trajectory, statistical trajectory correction maneuvers are necessary to correct for navigation errors and maneuver execution errors from the previous maneuvers. Following Ref. [54], these maneuvers are placed at every apoapsis ($\Delta V_1 - \Delta V_4$). Once close to the L_2 point, more frequent maneuvers must be scheduled ($\Delta V_5 - \Delta V_7$). The locations of these maneuvers are illustrated in Fig. 8.9.

The high nonlinearity and long propagation time of this orbit means that there will be specific directions along which deviations from the nominal orbit will lead to very large state deviations downstream. However, because this orbit is complex and non-periodic, it is not trivial to identify the most significant directions. We can therefore use the CGT procedure detailed in the previous

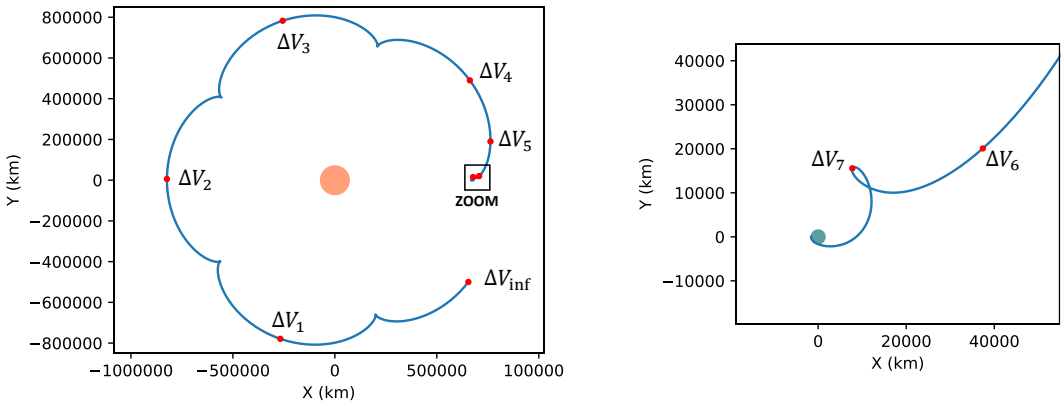

Figure 8.9: Low-energy approach trajectories to Europa with ΔV locations marked, in Jupiter-centric (left) and Europa-centric (center, right) rotating frames

sections to identify the directions of maximum final deviation. For each ΔV location, the reference orbit state at that location was propagated to the final staging point 50 km from the surface of Europa, along with the first-order STM for the specific orbit segment. An eigendecomposition of the CGT was then performed to identify $\boldsymbol{\xi}_1$. For this scenario, there is a distinct direction that becomes more and more important as the propagation time is increased (i.e. the state is propagated from an earlier ΔV point). The magnitude of the eigenvalues of the CGT corresponding to each ΔV point are shown in Fig. 8.10 - it is clear that the eigenvalue $\lambda_{\boldsymbol{\xi}_1}$ corresponding to the most sensitive direction for each orbit arc is significantly more important than the other directions. We can also see that, as expected, it becomes more important as the propagation time increases (i.e. for the earlier ΔV locations).

The higher-order DSTTs (up to order $m = 4$) were then integrated solely along $\boldsymbol{\xi}_1$ from each ΔV point to the staging point. This therefore involves fifty-four additional differential equations being integrated along with the state vector (thirty-six for the STM, and six for each order of DSTT up to $m = 4$). The full higher-order STTs along the Cartesian directions were also integrated for comparison. Note that, due to the long propagation time from ΔV_{inf} to the staging point, it is unlikely that state deviations would need to be mapped from the initial ΔV locations to the final staging point in an operational setting, but it still provides an interesting case study.

1000 perturbation vectors were sampled from a zero-mean normal distribution with 1-σ values of 25 m and 0.5 mm/s for the position and velocity components, respectively. These are smaller than the expected state uncertainty values in this orbital regime [54]; however, these were chosen to avoid very large final state deviations from the earlier ΔV locations. These perturbation vectors were applied to the initial state at each ΔV location. The various orders of STTs and DSTTs were used to propagate the state deviations from each ΔV location to the final time. Each perturbed state vector was also numerically integrated to the final time, and the error in the final position vector from using the STTs and DSTTs for propagating the perturbed states was computed. These are shown in Fig. 8.11. Note that, in these trajectories, ΔVs are not being applied at these locations. We are simply showing how the uncertainty prediction accuracy changes with varying orders of

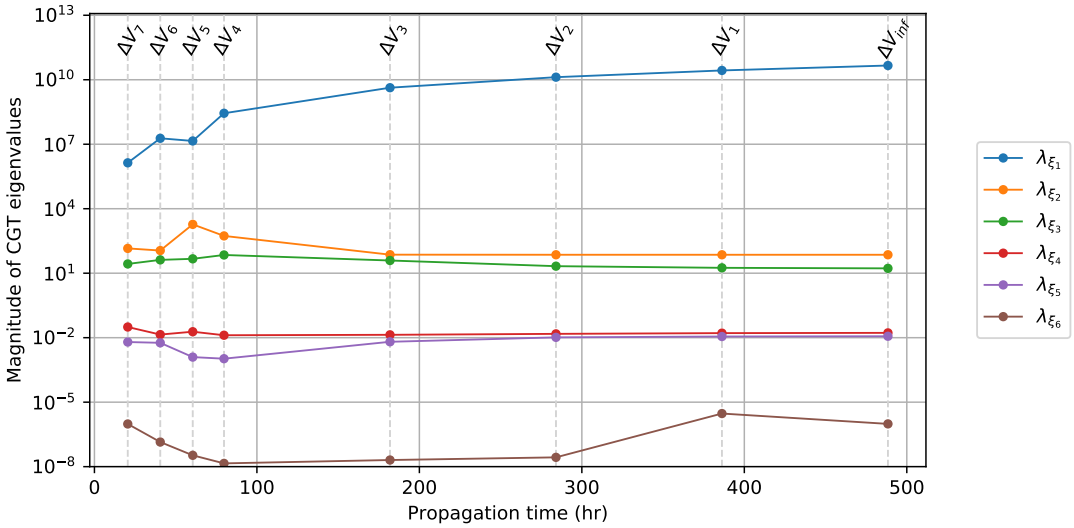

Figure 8.10: Magnitude of CGT eigenvalues for Europa approach trajectories propagated from each ΔV point to staging point

STTs and DSTTs, which can inform statistical maneuver design.

For these specific propagation arcs, the second-order DSTT along $\boldsymbol{\xi}_1$ yields nearly the same results as the the full second-order STT, despite only requiring the integration and storage of forty-two additional elements in addition to the state vector, rather than the full 252. As the propagation arcs become longer, the third and fourth-order DSTTs approach the accuracy of the full third and fourth-order STTs. This is due to the increasing importance of the dominant nonlinear direction relative to all other directions as the propagation time increases. For these situations, the integration and storage benefits become even more significant - including up to the fourth-order DSTT along $\boldsymbol{\xi}_1$ requires only fifty-four elements, compared to the 9324 elements required for the full fourth-order STT.

8.5 Estimating magnitude of STT terms

8.5.1 Derivation

The relative importance of the different DSTT directions and orders (e.g. Figs. 8.3-8.6) may not be immediately apparent. The magnitudes of the expected deviations along each direction will differ, and there may be higher-order contributions along sensitive directions that are more important than lower-order effects along stable directions. Assuming the state vector at time t_0 obeys a zero-mean Gaussian distribution, we can estimate **a priori** the magnitude of the higher-order terms along one or more directions. This is useful for determining along which directions the higher-order terms are significantly larger than others, and for estimating the error in ignoring certain terms. This heuristic can be used to include only the terms that would yield this desired accuracy. In order to derive this, we can first start with the magnitude of the first-order DSTT (also called directional state transition matrix) terms along a given direction:

$$\|\delta x_f\|_{\gamma_1} = \sqrt{\sum_{i=1}^{n} \left(\phi^{i,\gamma_1} \delta y_0^{\gamma_1}\right)^2} \tag{8.29}$$

where n is the size of the state vector $\mathbf{x}_f$, and $\delta y_0^{\gamma_1}$ is the initial perturbation vector projected onto the direction $\boldsymbol{\xi}_{\gamma_1}$, $\gamma_1 \in [1, k]$. We will assume the initial state distribution in the perturbation vector

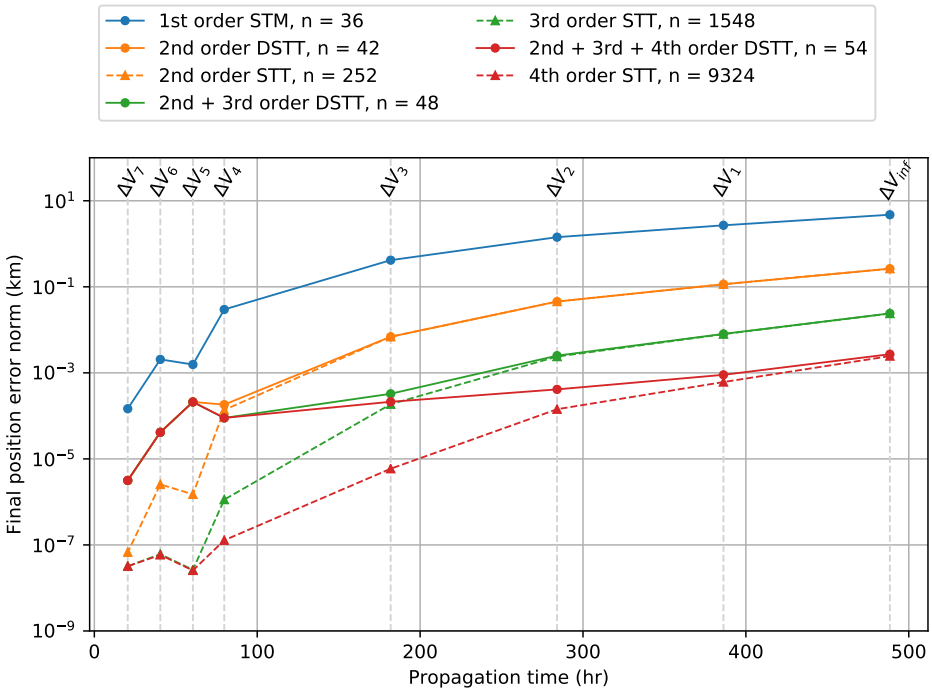

Figure 8.11: Position propagation error for STT/DSTT propagation to staging time for Europa approach scenario

is Gaussian, with zero mean and an initial covariance matrix P_0. We want to find the **expected** magnitude of a term given the distribution in the initial state deviation vectors. We can take the expected value of both sides of Eq. 8.29, and simplify accordingly:

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1}\right] = \mathbb{E}\sqrt{\sum_{i=1}^{n}\left[(\phi^{i,\gamma_1})^2(\delta y_0^{\gamma_1})^2\right]} = \mathbb{E}\left[\sqrt{\sum_{i=1}^{n}(\phi^{i,\gamma_1})^2}\sqrt{(\delta y_0^{\gamma_1})^2}\right] = \mathbb{E}\left[\sqrt{\sum_{i=1}^{n}(\phi^{i,\gamma_1})^2}|\delta y_0^{\gamma_1}|\right] \tag{8.30}$$

Now the directional state transition matrix ϕ^{i,γ_1} is deterministic and can be taken out of the expectation operator, giving:

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1}\right] = \|\phi^{:,\gamma_1}\|\,\mathbb{E}\left[|\delta y_0^{\gamma_1}|\right] \tag{8.31}$$

where $\|\phi^{:,\gamma_1}\|$ is the norm of the STM elements along the γ_1 axis. The $\mathbb{E}\left[|\delta y_0^{\gamma_1}|\right]$ term corresponds to the **half-normal** distribution, where $\mathbb{E}\left[|\delta y_0^{\gamma_1}|\right] = \sqrt{\frac{2}{\pi}}\sigma_{\gamma_1}$, with $\mathbb{E}\left[(\delta y_0^{\gamma_1})^2\right] = \sigma_{\gamma_1}^2$. Thus, the expected value for the magnitude of the first-order term along γ_1 becomes

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1}\right] = \sqrt{\frac{2}{\pi}}\|\phi^{:,\gamma_1}\|\,\sigma_{\gamma_1} \tag{8.32}$$

We can follow a similar procedure in order to estimate the magnitude of the second-order terms. We can start from

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2}\right] = \mathbb{E}\left[\frac{1}{2}\sqrt{\sum_{i=1}^{n}\left(\phi^{i,\gamma_1\gamma_2}\delta y_0^{\gamma_1}\delta y_0^{\gamma_2}\right)^2}\right] = \frac{1}{2}\|\phi^{:,\gamma_1\gamma_2}\|\,\mathbb{E}\left[|\delta y_0^{\gamma_1}\delta y_0^{\gamma_2}|\right] \tag{8.33}$$

The $\mathbb{E}\left[|\delta y_0^{\gamma_1}\delta y_0^{\gamma_2}|\right]$ term will take on different forms for $\gamma_1 \neq \gamma_2$ and $\gamma_1 = \gamma_2$. For $\gamma_1 = \gamma_2$, we obtain

$$\mathbb{E}\left[|\delta y_0^{\gamma_1}\delta y_0^{\gamma_2}|\right] = \mathbb{E}\left[(\delta y_0^{\gamma_1})^2\right] = \sigma_{\gamma_1}^2 \tag{8.34}$$

Now, for the $\gamma_1 \neq \gamma_2$ case, given the presence of the absolute value inside the expectation operator, it is difficult to obtain an exact value for the expected magnitude of the second-order term for an arbitrary initial covariance distribution. However, we can use **Holder's inequality** to obtain an upper bound on the magnitude, which gives the following (relatively simple) equation

$$\mathbb{E}\left[|\delta y_0^{\gamma_1}\delta y_0^{\gamma_2}|\right] \leq \sigma_{\gamma_1}\sigma_{\gamma_2} \tag{8.35}$$

This gives the following estimates for the magnitude of the second-order terms:

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2}\right]_{\gamma_1=\gamma_2} = \frac{1}{2}\|\phi^{:,\gamma_1\gamma_2}\|\ \sigma_{\gamma_1}^2 \tag{8.36}$$

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2}\right]_{\gamma_1\neq\gamma_2} \leq \frac{1}{2}\|\phi^{:,\gamma_1\gamma_2}\|\ \sigma_{\gamma_1}\sigma_{\gamma_2} \tag{8.37}$$

This procedure can again be repeated to estimate the magnitude of the third and fourth-order terms, giving the following:

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2,\gamma_3}\right]_{\gamma_1=\gamma_2=\gamma_3} = \frac{\sqrt{2}}{3\sqrt{\pi}}\|\phi^{:,\gamma_1\gamma_2\gamma_3}\|\ \sigma_{\gamma_1}^3 \tag{8.38}$$

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2,\gamma_3}\right]_{\gamma_1\neq\gamma_2\neq\gamma_3} \leq \frac{\sqrt{2}}{3\sqrt{\pi}}\|\phi^{:,\gamma_1\gamma_2\gamma_3}\|\ \sigma_{\gamma_1}\sigma_{\gamma_2}\sigma_{\gamma_3} \tag{8.39}$$

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2,\gamma_3,\gamma_4}\right]_{\gamma_1=\gamma_2=\gamma_3=\gamma_4} = \frac{1}{8}\|\phi^{:,\gamma_1\gamma_2\gamma_3\gamma_4}\|\ \sigma_{\gamma_1}^4 \tag{8.40}$$

$$\mathbb{E}\left[\|\delta x_f\|_{\gamma_1,\gamma_2,\gamma_3}\right]_{\gamma_1\neq\gamma_2\neq\gamma_3\neq\gamma_4} \leq \frac{1}{8}\|\phi^{:,\gamma_1\gamma_2\gamma_3\gamma_4}\|\ \sigma_{\gamma_1}\sigma_{\gamma_2}\sigma_{\gamma_3}\sigma_{\gamma_4} \tag{8.41}$$

8.5.2 Application: Earth-Moon halo orbit

We apply these estimates to the NRHO example from Section 8.4.2. For this scenario, when looking at the eigenvalues of the CGT in Eq. 8.27, it is relatively clear that $\lambda_{\xi_1} \gg \lambda_{\xi_{2\ldots6}}$, and that the terms along the ξ_1 direction will as a result be significantly larger than along the other directions. However, as stated previously, this is not always immediately apparent. In order to avoid having to test hundreds or thousands of perturbations, we can use Eqs. 8.36- 8.41 to estimate the magnitude of the higher-order terms given some expected uncertainty distribution in the initial states. This can allow us to determine **a priori** which terms will likely have the most important contributions, and to determine whether higher-order terms along certain directions are larger than lower-order terms along other directions.

For the second-order NRHO scenario, for both the Cartesian STTs and DSTTs, Eqs. 8.36 and 8.37 can be used to provide an upper bound on the estimated magnitude of the term along each set of directions, given the initial state uncertainty vector from Eq. 8.28. These bounds are shown

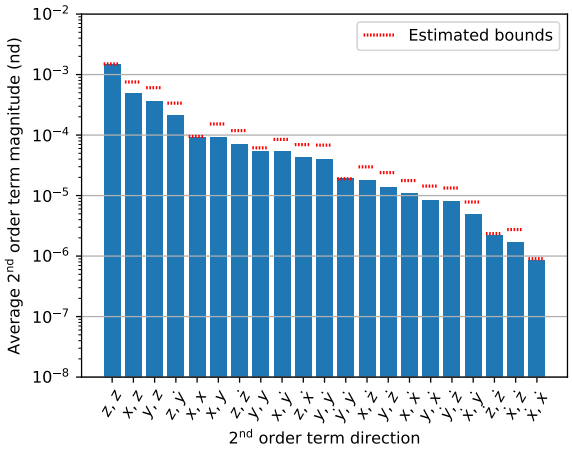

Figure 8.12: Second-order term magnitudes for NRHO scenario using Cartesian STTs with estimated magnitude bounds

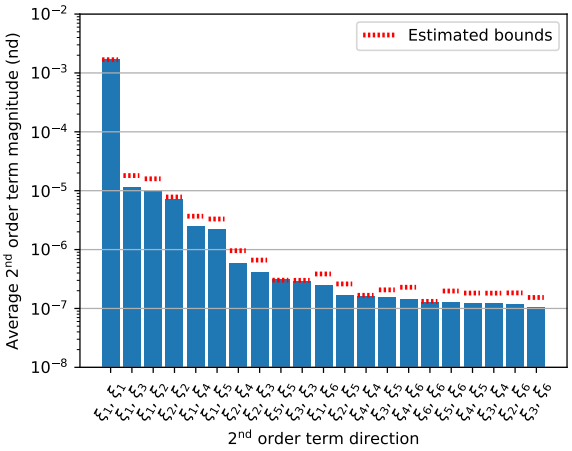

Figure 8.13: Second-order term magnitudes for NRHO scenario using DSTTs with estimated magnitude bounds

in Figs. 8.12 and 8.13, along with the same numerically computed second-order term magnitudes from Figs. 8.3 and 8.4. These show that, for higher-order terms along repeated directions (e.g. x, x or ξ_2, ξ_2), the magnitude estimates do in fact constitute an equality. For terms along different directions, the estimates yield a relatively tight upper bound on their expected magnitude. These estimates provide a simple heuristic to determine the relative importance of each term without requiring sampling a large number of state errors.

8.6 Nonlinear Uncertainty Propagation with Directional State Transition Tensors

8.6.1 Covariance propagation using STTs

We now consider the spacecraft state vector to be a Gaussian random vector $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{m}_0, P_0)$, where $\mathbf{m}_0$ is the mean state vector and P_0 is the covariance matrix at time t_0. STTs can be used to analytically propagate a spacecraft state's mean and covariance through nonlinear dynamics to some

final time t_f, following Ref. [92] (omitting the time subscripts from the ϕ terms for conciseness):

$$\delta m_f^i = \sum_{p=1}^{m} \frac{1}{p!} \phi^{i,\kappa_1 \ldots \kappa_p} \mathbb{E}\left[\delta x_0^{\kappa_1} \ldots \delta x_0^{\kappa_p}\right] \tag{8.42}$$

$$P_f^{ij} = \left(\sum_{p=1}^{m} \sum_{q=1}^{m} \frac{1}{p!q!} \phi^{i,\kappa_1 \ldots \kappa_p} \phi^{j,\eta_1 \ldots \eta_q} \times \mathbb{E}\left[\delta x_0^{\kappa_1} \ldots \delta x_r^{\kappa_p} \delta x_0^{\eta_1} \ldots \delta x_0^{\eta_p}\right] \right) - \delta m_f^i \delta m_f^j \tag{8.43}$$

The moments $\mathbb{E}\left[\delta x_0^{\kappa_1} \ldots \delta x_0^{\kappa_p}\right]$ of the Gaussian probability distribution of $\delta \mathbf{x}_0$ can be written in terms of $\delta \mathbf{m}_0$ and P_0, again following Ref. [92]:

$$\mathbb{E}\left[\delta x^i\right] = \delta m^i \tag{8.44}$$

$$\mathbb{E}\left[\delta x^i \delta x^j\right] = \delta m^i \delta m^j + P^{ij} \tag{8.45}$$

$$\mathbb{E}\left[\delta x^i \delta x^j \delta x^k\right] = \delta m^i \delta m^j \delta m^k + (\delta m^i P^{jk} + \delta m^j P^{ik} + \delta m^k P^{ij}) \tag{8.46}$$

$$\begin{aligned} \mathbb{E}\left[\delta x^i \delta x^j \delta x^k \delta x^l\right] = {} & \delta m^i \delta m^j \delta m^k \delta m^l + (\delta m^i \delta m^j P^{kl} + \delta m^i \delta m^k P^{jl} + \delta m^j \delta m^k P^{il} \\ & + \delta m^i \delta m^l P^{jk} + \delta m^j \delta m^l P^{ik} + \delta m^k \delta m^l P^{ij}) + P^{ij} P^{kl} + P^{ik} P^{jl} + P^{il} P^{jk} \end{aligned} \tag{8.47}$$

Moments of order higher than 4 are lengthy to write out explicitly, containing 26, 76, 231, and 763 terms, respectively, for the fifth through eighth moments. These can be obtained following the procedure described in Ref. [73]. If the STTs have been integrated from $\mathbf{x}_0 = \mathbf{m}_0$, then the initial mean deviation $\delta \mathbf{m}_0 = 0$, which causes all odd-order moments to vanish. Still, the computation of the higher-order moments constitutes a very significant burden on top of integrating the higher-order STTs.

8.6.2 Covariance propagation using DSTTs

If we are using Eq. 8.19 to approximate full higher-order effects using DSTTs along a single direction $\boldsymbol{\xi}_1$, then we can greatly simplify Eqs. 8.42 and 8.43. First, we can simplify higher-order calculations involving the covariance matrix through the following transformation

$$\sigma_{\xi_1} = \sqrt{\xi_1^i P^{ij} \xi_1^j} \tag{8.48}$$

σ_{ξ_1} is a scalar that refers to the 1-σ uncertainty along the $\boldsymbol{\xi}_1$ direction. Using the second-order DSTTs along $\boldsymbol{\xi}_1$, Eqs. 8.42 and 8.43 (for $m = 2$) reduce to

$$\delta m_f^i = \frac{1}{2}\psi_{[2]}^i \sigma_{\xi_1}^2 \tag{8.49}$$

$$P_f^{ij} = \phi^{i,\kappa_1}\phi^{j,\eta_1}P_0^{\kappa_1\eta_1} + \frac{1}{2}\sigma_{\xi_1}^4 \psi_{[2]}^i \psi_{[2]}^j \tag{8.50}$$

where $\boldsymbol{\psi}_{[p]}$ refers to the vector of DSTTs of order p along $\boldsymbol{\xi}_1$. If using the second and third-order DSTTs along $\boldsymbol{\xi}_1$, Eqs. 8.42 and 8.43 - which are more or less intractable at $m > 2$ - reduce to the following relatively simple equations:

$$\delta m_f^i = \frac{1}{2}\psi_{[2]}^i \sigma_{\xi_1}^2 \tag{8.51}$$

$$P_f^{ij} = \phi^{i,\kappa_1}\phi^{j,\eta_1}P_0^{\kappa_1\eta_1} + \left[\frac{1}{2}\psi_{[2]}^i\psi_{[2]}^j + \frac{1}{2}\psi_{[3]}^i\psi_{[1]}^j + \frac{1}{2}\psi_{[1]}^i\psi_{[3]}^j\right]\sigma_{\xi_1}^4 + \left[\frac{5}{12}\psi_{[3]}^i\psi_{[3]}^j\right]\sigma_{\xi_1}^6 \tag{8.52}$$

Similarly, if using the second, third, and fourth-order DSTTs along $\boldsymbol{\xi}_1$, Eqs. 8.42 and 8.43 reduce to

$$\delta m_f^i = \frac{1}{2}\psi_{[2]}^i \sigma_{\xi_1}^2 + \frac{1}{8}\psi_{[4]}^i \sigma_{\xi_1}^4 \tag{8.53}$$

$$\begin{aligned} P_f^{ij} = \phi^{i,\kappa_1}\phi^{j,\eta_1}P_0^{\kappa_1\eta_1} &+ \left[\frac{1}{2}\psi_{[2]}^i\psi_{[2]}^j + \frac{1}{2}\psi_{[3]}^i\psi_{[1]}^j + \frac{1}{2}\psi_{[1]}^i\psi_{[3]}^j\right]\sigma_{\xi_1}^4 + \\ &\left[\frac{5}{12}\psi_{[2]}^i\psi_{[2]}^j + \frac{1}{4}\psi_{[4]}^i\psi_{[2]}^j + \frac{1}{4}\psi_{[2]}^i\psi_{[4]}^j\right]\sigma_{\xi_1}^6 + \left[\frac{1}{6}\psi_{[4]}^i\psi_{[4]}^j\right]\sigma_{\xi_1}^8 \end{aligned} \tag{8.54}$$

This is orders of magnitude cheaper to compute than the full fourth-order covariance propagation calculation, which requires up to 231 7-dimensional terms and 763 8-dimensional terms.

8.7 Examples: Uncertainty Propagation using DSTTs

8.7.1 Near rectilinear halo orbit

We first show the improvements in nonlinear uncertainty propagation for the same NRHO scenario and propagation arc of 1.5 periods from Section 8.4.2. The initial conditions and parameters for this scenario can be found in Table 8.1. An initial diagonal covariance matrix P_0 was constructed using the same 3-σ state uncertainty values as in Eq. 8.28. The inital mean deviation vector and state covariance matrix were then mapped to the final time using the first-order STM

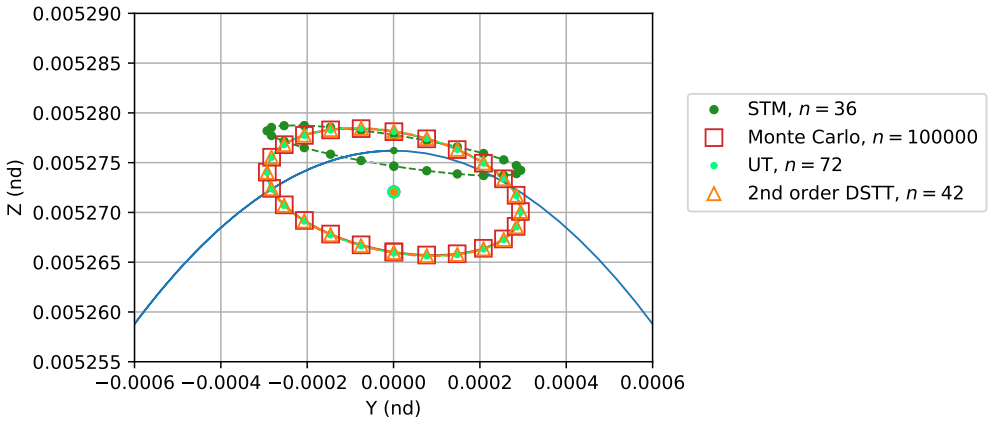

Figure 8.14: 1-σ state covariance at final time for NRHO scenario

($n = 36$), the second-order DSTT ($n = 42$, using Eq. 8.50), and the unscented transform (UT, $n = 12 \times 6 = 72$) [64]. A Monte Carlo simulation with 100000 state deviation vectors sampled from the initial uncertainty distribution was also performed for reference. The mean state and 1-σ uncertainty ellipses along the $y - z$ axes at the final time are shown in Fig. 8.14. For this scenario, we can see that the STM is not sufficient to accurately capture the nonlinearity of the trajectory; however, both the second-order DSTT and the unscented transform do. The second-order DSTT performs comparably to the unscented transform, but requires only around half as many additional equations to be integrated. This showcases a scenario where the DSTT along $\boldsymbol{\xi}_1$ could be a potential improvement over the unscented transform.

8.7.2 Europa lander scenario

We next apply the DSTT concept to nonlinear uncertainty propagation for the Europa lander scenario from Section 8.4.3. We consider the highly nonlinear mapping from the ΔV_{inf} location to the staging location. In Ref. [54], the authors found that linear uncertainty propagation methods were insufficient to accurately capture the statistics of the trajectory arc, and determined that higher-order methods should be investigated. As was shown in the previous section, the trajectory arc has a very distinct unstable direction $\boldsymbol{\xi}_1$, which means that the full higher-order STTs could

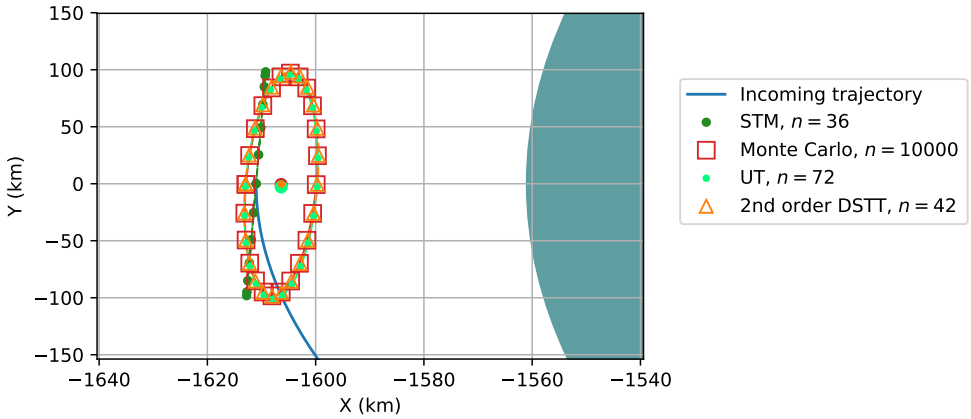

Figure 8.15: 1-σ state covariance at final time for low-uncertainty Europa approach scenario

be reduced to the DSTTs along $\boldsymbol{\xi}_1$, with a minimal loss in accuracy.

First, we consider the initial state uncertainties used in the previous section for this scenario, with 1-σ values of 25 m and 0.5 mm/s for the position and velocity components, respectively. The initial mean state deviation vector and state covariance matrix were then mapped to the final time using the STM, the second-order DSTT, and the unscented transform. In addition, a Monte Carlo simulation with 100000 state deviation vectors sampled from the initial state uncertainty distribution was performed. The resulting final mean deviations and 1-σ covariance ellipses are shown in Fig. 8.15, projected on the $x - y$ plane. The second-order DSTT clearly accurately captures the statistics of the final state uncertainty, while again requiring fewer integrations than the unscented transform.

We next consider the same trajectory, but with four times larger initial state uncertainties (1-σ values of 100 m and 2.0 mm/s for the position and velocity components, respectively). The initial mean state deviation vector and state covariance matrix were then mapped to the final time using the STM, the second-order DSTT ($n = 42$), the unscented transform ($n = 72$), and the second through fourth-order DSTTs along $\boldsymbol{\xi}_1$ (see Eq. 8.54, $n = 54$). Again, a Monte Carlo simulation with 100000 state deviation vectors sampled from the initial state uncertainty distribution was performed. The resulting final mean deviations and 1-σ covariance ellipses are shown in Fig. 8.16,

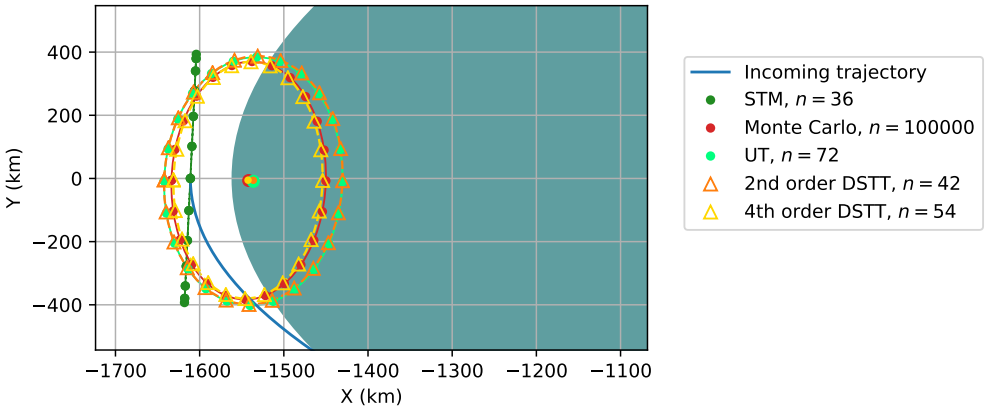

Figure 8.16: 1-σ state covariance at final time for high-uncertainty Europa approach scenario

projected on the $x - y$ plane.

The final state uncertainties become much larger with the larger initial state uncertainties, and the second-order DSTT and unscented transform do not accurately map the uncertainty distribution to the final time. However, the fourth-order DSTT successfully captures the nonlinear evolution of the state uncertainty, despite requiring integrating eighteen fewer equations than the standard unscented transform. Another important advantage to using DSTTs for covariance propagation is that, due to the reduced number of terms, the time required to incorporate the higher-order terms into the covariance calculation is significantly reduced. For this example, the full fourth-order covariance computation using Eq. 8.43 takes around 0.2 seconds to run in Python on a standard laptop computer, whereas the computation using the DSTTs (Eq. 8.54) takes about 0.0002 seconds, or around 1000 times faster.

8.8 Conclusions

The use of higher-order state transition tensors (STTs) for operational astrodynamics applications has been limited by the fact that the integration time and storage requirements increase exponentially as the order of STT included in the approximation increases. In this chapter, we show that by aligning the STTs with particularly important directions, we can distinguish the

dominant terms in the STT approximation, and ignore the less significant terms, which are generally very numerous. We refer to this concept as directional state transition tensors (DSTTs). For the examples provided in this chapter, the DSTT procedure can reduce the number of additional differential equations to be integrated with the state from 252 to forty-two for the second-order DSTT, from 1548 to forty-eight for the third-order DSTT, and from 9324 to fifty-four for the fourth-order DSTT. DSTTs can subsequently be used to greatly speed up nonlinear uncertainty propagation computations, for example reducing the computation time for a fourth-order covariance propagation by a factor of 1000. The DSTT approximation is shown to increase in accuracy as the dominant nonlinear directions increase in importance relative to the other directions. Long propagation arcs and highly nonlinear, chaotic dynamic systems are examples of scenarios where DSTTs are expected to be a good approximation of the full STTs.

There are several potential use cases in an operational setting for DSTTs. Precise navigation and orbit determination in cislunar space can be a challenging process due to the chaotic dynamics, but its importance will only increase with the upcoming Artemis and Lunar Gateway missions [79]. As an application of the DSTT concept, in Ref. [16], an extended Kalman filter (EKF) was augmented with the second-order DSTT (following the higher-order EKF detailed in Ref. [93]), and was shown to provide significant improvement over the standard EKF with minimal overhead cost. In addition, the guidance and trajectory optimization algorithms developed in Chapters 3-6 could benefit from reducing storage requirements for higher-order STTs through the DSTT concept.

Chapter 9

Conclusions

9.1 Dissertation Summary

This dissertation is focused on developing methods to incorporate higher-order Taylor series expansions of spacecraft dynamics, also known as state transition tensors (STTs), in spacecraft guidance, trajectory optimization, and maneuver design. The methods developed in this work are applicable to any nonlinear dynamic system, including highly complex systems that are currently of high interest, such as cislunar space. They could be particularly useful for situations with limited computational resources, such as on-board a spacecraft, or for time-critical trajectory re-planning scenarios. These algorithms are developed to be computationally efficient and flexible, allowing for changing mission constraints and priorities.

The first contribution of this thesis is to derive the general spacecraft guidance formulation using the higher-order STTs of some reference trajectory. Much of the work in the subsequent chapters builds upon this framework. Simplifications are presented for the impulsive maneuver targeting scenario, and the method is improved by allowing for a variable time-of-flight. The focus of the thesis then shifts to consider low-thrust or continuous-thrust trajectory optimization by combining the higher-order approximations of nonlinear dynamics with differential dynamic programming (DDP). DDP is a second-order iterative optimization method which is particularly well-suited to use within STT-approximated dynamics. The next contribution of this thesis work is to incorporate state uncertainty into the STT-based spacecraft maneuver design scheme. This is especially important for the types of missions this thesis work investigates: deep-space spacecraft

missions which can have large state uncertainties, due to their highly nonlinear trajectories with significant gaps between state measurements. This high degree of nonlinearity can lead to the spacecraft state distribution becoming decidedly non-Gaussian. This is addressed by deriving a novel algorithm to consider spacecraft maneuver design with non-Gaussian chance constraints, using Gaussian mixture models and iterative risk allocation. Finally, a strategy to efficiently approximate the effects of higher-order STTs is developed, which could improve the performance of all the STT-based methods in this thesis.

The methods derived in this dissertation are applied to several different scenarios with spacecraft executing highly complex trajectories. A particular emphasis is placed on missions operating in cislunar space, a region of space that is currently of very high interest. Additional applications are presented for spacecraft orbiting around small bodies and approaching the surface of Jupiter's moon Europa. The examples demonstrate the utility of STT-based guidance and trajectory optimization schemes to accomplish a variety of mission objectives. The algorithms are shown to be flexible, robust, and efficient. For each scenario, comparisons with traditional numerical methods are provided, illustrating the benefits of using these nonlinear methods. Beyond the applications shown in this thesis, the STT-based algorithms could find use in any scenario with highly nonlinear but well-modeled dynamics, and with limited computational resources or time constraints.

9.2 Directions for Future Work

This dissertation develops the mathematical foundations to use state transition tensors for a number of different applications, but there are still many interesting and promising directions to continue research on these topics. Some of these are listed below.

Spacecraft guidance and maneuver design using STTs:

- The examples provided in this work strictly considered the design of single maneuvers to achieve targets at a single time. The equations can easily be used to concurrently compute or optimize multiple maneuvers with constraints applied at multiple times, provided the

STTs for the reference trajectory are integrated over the desired timespans.

- The variable time-of-flight algorithm in Chapter 4 could be combined with the stochastic maneuver design formulations derived in Chapter 6 to further improve the flexibility of these algorithms.

- A more rigorous method to estimate the accuracy region for each order of STT included in the guidance scheme should be derived in order to validate the algorithm for use in mission operations.

Trajectory optimization using differential dynamic programming:

- The variable time-of-flight algorithm in Chapter 4, the stochastic maneuver design formulations in Chapter 6, and the non-Gaussian chance constraint algorithm from Chapter 7 could be applied to the STT/DDP algorithm. This would improve the realism and fidelity of the algorithm.

- In general, STTs could be used to propagate uncertainty between stages in a DDP algorithm. The equations in Chapter 6 could be used to replace the unscented transform in existing stochastic DDP algorithms, such as the one in Ref. [86].

- The STT/DDP method could be used to construct a nonlinear Delta V-99 (DV99) tool with realistic navigation and orbit determination models for continuous-thrust trajectories. This could provide an efficient strategy to improve upon existing linear DV99 methods .

Chance-constrained control:

- The DSTT concept from Chapter 8 could be used to develop an efficient higher-order method to express a non-Gaussian distribution. If chance constraints on this distribution can be formulated as geometric constraints, then this could provide a more efficient method to apply non-Gaussian chance constraints than the GMM IRA algorithm in Chapter 7.

- The GMM IRA algorithm from Chapter 7 could be used to compute optimal collision avoidance maneuvers for spacecraft in low-Earth orbit while taking into account the effects of non-Gaussian state distributions.

Bibliography

[1] Demonstration Rocket for Agile Cislunar Operations. Technical report, DARPA, 2020.

[2] Ossama Abdelkhalik and Elsan Taheri. Approximate on-off low-thrust space trajectories using fourier series. Journal of Spacecraft and Rockets, 49(5):962–965, 2012.

[3] Roberto Armellin, Pierluigi Di Lizia, Franco Bernelli-Zazzera, and Martin Berz. Asteroid close encounters characterization using differential algebra: the case of apophis. Celestial Mechanics and Dynamical Astronomy, 107(4):451–470, 2010.

[4] Jonathan Aziz, Jeffrey Parker, and Daniel Scheeres. Low-thrust many-revolution trajectory optimization via differential dynamic programming and a sundman transformation. Journal of the Astronautical Sciences, 65:205–228, 2018.

[5] Jonathan Aziz, D. Scheeres, and Gregory Lantoine. Hybrid differential dynamic programming in the circular restricted three-body problem. Journal of Guidance, Control, and Dynamics, 42:1–13, 2019.

[6] R. H. Battin and R. M. Vaughan. An elegant lambert algorithm. Journal of Guidance, Control, and Dynamics, 7(6):662–670, 1984.

[7] Carl M. Bender and Steven A. Orszag. Advanced mathematical methods for scientists and engineers: I: Asymptotic methods and perturbation theory. Springer, 1999.

[8] Dennis S. Bernstein. Nonquadratic cost and nonlinear feedback control. International Journal of Robust and Nonlinear Control, 3(3):211–229, 1993.

[9] K. Berry, B. Sutter, A. May, Khaliah Williams, Brent Barbee, M. Beckman, and Bobby Williams. Osiris-rex touch-and-go (tag) mission design and analysis. Advances in the Astronautical Sciences, 149:667–678, 01 2013.

[10] Martin Berz. Modern Map Methods in Particle Beam Physics. Academic Press, 1999.

[11] Collin Bezrouk and Jeffrey Parker. Long term evolution of distant retrograde orbits in the earth-moon system. Astrophysics and Space Science, 362(176), 2017.

[12] Lars Blackmore, Masahiro Ono, Askar Bektassov, and Brian C. Williams. A probabilistic particle-control approximation of chance-constrained stochastic predictive control. IEEE Transactions on Robotics, 26(3):502–517, 2010.

[13] Lars Blackmore, Masahiro Ono, and Brian C. Williams. Chance-constrained optimal path planning with obstacles. IEEE Transactions on Robotics, 27(6):1080–1094, 2011.

[14] Stefano Bonasera, Ian Elliott, Christopher J. Sullivan, Natasha Bosanac, Nisar Ahmed, and Jay McMahon. Designing impulsive station-keeping maneuvers near a sun-earth l2 halo orbit via reinforcement learning. In 31st AAS/AIAA Space Flight Mechanics Meeting, Virtual, 02 2021.

[15] Stefano Bonasera, Ian Elliott, Christopher J. Sullivan, Natasha Bosanac, Nisar Ahmed, and Jay McMahon. Designing impulsive station-keeping maneuvers near a sun-earth l2 halo orbit via reinforcement learning. In 31st AAS/AIAA Space Flight Mechanics Meeting, 2021.

[16] Oliver Boodram, Spencer Boone, and Jay McMahon. Efficient nonlinear spacecraft navigation using directional state transition tensors. In 2022 AAS/AIAA Astrodynamics Specialist Conference, 2022.

[17] Spencer Boone, Stefano Bonasera, Jay W. McMahon, Natasha Bosanac, and Nisar R. Ahmed. Incorporating observation uncertainty into reinforcement learning-based spacecraft guidance schemes. In AIAA SCITECH 2022 Forum.

[18] Spencer Boone and Jay McMahon. Rapid local trajectory optimizaion using higher-order state transition tensors and differential dynamic programming. In 2020 AAS/AIAA Astrodynamics Specialist Conference, 2020.

[19] Spencer Boone and Jay McMahon. Directional state transition tensors for capturing dominant nonlinear dynamical effects. In 2021 AAS/AIAA Astrodynamics Specialist Conference, 2021.

[20] Spencer Boone and Jay McMahon. Orbital guidance using higher-order state transition tensors. Journal of Guidance, Control, and Dynamics, 44(3):493–504, 2021.

[21] Spencer Boone and Jay McMahon. Variable time-of-flight spacecraft maneuver targeting using state transition tensors. Journal of Guidance, Control, and Dynamics, 44(11):2072–2080, 2021.

[22] Spencer Boone and Jay McMahon. Rapid local trajectory optimizaion in cislunar space. In 2022 AAS Guidance, Navigation and Control Conference, 2022.

[23] Spencer Boone and Jay McMahon. Semi-analytic spacecraft maneuver design with stochastic constraints. In 2022 American Control Conference, 2022.

[24] Spencer Boone and Jay W. McMahon. Non-gaussian chance-constrained trajectory control using gaussian mixtures and risk allocation. In 2022 IEEE 61st Conference on Decision and Control (CDC), 2022.

[25] Spencer Boone and Jay W. McMahon. Spacecraft maneuver design with non-gaussian chance constraints using gaussian mixtures. In 2022 AAS/AIAA Astrodynamics Specialist Conference, 2022.

[26] Natasha Bosanac, Cassandra M. Webster, Kathleen C. Howell, and David C. Folta. Trajectory design for the wide field infrared survey telescope mission. Journal of Guidance, Control, and Dynamics, 42(9), 2019.

[27] Jonathan Brown and Jeremy Petersen. Applying dynamical systems theory to optimize libration point orbit stationkeeping maneuvers for wind. In 2014 AIAA/AAS Astrodynamics Specialist Conference, 2014.

[28] Robert Brusch. Bilinear tangent yaw guidance. In Guidance and Control Conference, 1979.

[29] C. Chadwick and L.J. Miller. An overview of the adam maneuver analysis system. In 1983 AAS/AIAA Astrodynamics Specialist Conference, 1983.

[30] Doris Chandler and Isaac Smith. Development of the iterative guidance mode with its application to various vehicles and missions. Journal of Spacecraft and Rockets, 4(7):898–903, 1967.

[31] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and HUY ANH PHAN. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. IEEE Signal Processing Magazine, 32(2):145–163, 2015.

[32] Camilla Colombo, Massimiliano Vasile, and Gianmarco Radice. Optimal low-thrust trajectories to asteroids through an algorithm based on differential dynamic programming. Celestial Mechanics and Dynamical Astronomy, 105(1), 2009.

[33] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. Trust Region Methods. Society for Industrial and Applied Mathematics, 2000.

[34] Andrew Cox Kathleen Howell David Folta, Natasha Bosanac. The lunar icecube mission challenge: Attaining science orbit parameters from a constrained approach trajectory. In 27th AAS/AIAA SpaceFlight Mechanics Meeting, 2017.

[35] D. Davis, S. Phillips, K. C. Howell, Srianish Vutukuri, and B. McCarthy. Stationkeeping and transfer trajectory design for spacecraft in cislunar space. 2017.

[36] Diane C. Davis, Emily M. Zimovan-Spreen, Rolfe J. Power, and Kathleen C. Howell. Cubesat deployment from a near rectilinear halo orbit. In AIAA SCITECH 2022 Forum, 2022.

[37] Kyle J. DeMars, Yang Cheng, and Moriba K. Jah. Collision probability with gaussian mixture orbit uncertainty. Journal of Guidance, Control, and Dynamics, 37(3):979–985, 2014.

[38] S. Desai, S. Bhaskaran, W. Bollman, C. Halsell, J. Riedel, S. Synnott, S. Desai, S. Bhaskaran, W. Bollman, C. Halsell, J. Riedel, and S. Synnott. The DS-1 autonomous navigation system - Autonomous control of low thrust propulsion system.

[39] P. Di Lizia, R. Armellin, F. Bernelli-Zazzera, and M. Berz. High order optimal control of space trajectories with uncertain boundary conditions. Acta Astronautica, 93:217 – 229, 2014.

[40] Pierluigi Di Lizia, Roberto Armellin, Amalia Ercoli Finzi, and Martin Berz. High-order robust guidance of interplanetary trajectories based on differential algebra. Journal of Aerospace Engineering, Sciences and Applications, 1, 2008.

[41] G. Di Mauro, M. Schlotterer, S. Theil, and M. Lavagna. Nonlinear control for proximity operations based on differential algebra. Journal of Guidance, Control, and Dynamics, 38(11):2173–2187, 2015.

[42] Sean Phillips Kathleen Howell Diane Davis, Kenza Boudad. Disposal, deployment, and debris in near rectilinear halo orbits. In 29th AAS/AIAA Space Flight Mechanics Meeting, 2019.

[43] Donald Dichmann, Cassandra Alberding, and Wayne Yu. Stationkeeping monte carlo simulation for the james webb space telescope. In 26th International Symposium on Space Flight Dynamics, 2014.

[44] K. Fujimoto, D. J. Scheeres, and K. T. Alfriend. Analytical nonlinear propagation of uncertainty in the two-body problem. Journal of Guidance, Control, and Dynamics, 35(2), 2012.

[45] Masahiro Fujiwara and Ryu Funase. Autonomous trajectory guidance under uncertain dynamical environment using state transition tensors. 03 2022.

[46] Roberto Furfaro and Daniele Mortari. Least-squares solution of a class of optimal space guidance problems via theory of connections. Acta Astronautica, 168:92 – 103, 2020.

[47] C. Gates. A simplified model of midcourse maneuver execution errors. Technical report, NASA Jet Propulsion Laboratory, 1963.

[48] Abebe Geletu, Michael Klppel, Armin Hoffmann, and Pu Li. A tractable approximation of non-convex chance constrained optimization with non-gaussian uncertainties. Engineering Optimization, 47(4):495–520, 2015.

[49] O. Grasset, M.K. Dougherty, A. Coustenis, E.J. Bunce, C. Erd, D. Titov, M. Blanc, A. Coates, P. Drossart, L.N. Fletcher, H. Hussmann, R. Jaumann, N. Krupp, J.-P. Lebreton, O. Prieto-Ballesteros, P. Tortora, F. Tosi, and T. Van Hoolst. Jupiter icy moons explorer (juice): An esa mission to orbit ganymede and to characterise the jupiter system. Planetary and Space Science, 78:1–21, 2013.

[50] Daniel Grebow. Generating periodic orbits in the circular restricted three body problem with applications to lunar south pole coverage. Master's thesis, Purdue University, 2006.

[51] Davide Guzzetti, Emily Zimovan, Kathleen Howell, and Diane Davis. Stationkeeping analysis for spacecraft in lunar near rectilinear halo orbits. In 27th AAS/AIAA Space Flight Mechanics Meeting, 2017.

[52] C.R. Hargraves and S.W. Paris. Direct trajectory optimization using nonlinear programming and collocation. Journal of Guidance, Control, and Dynamics, 10(4):338–342, 1987.

[53] William Hart, G. Brown, Steven Collins, M. Soria-Santacruz, Paul Fieseler, Dan Goebel, Danielle Marsh, David Oh, Steve Snyder, Noah Warner, Gregory Whiffen, Linda Elkins-Tanton, James Bell, David Lawrence, Peter Lord, and Zachary Pirkl. Overview of the spacecraft design for the psyche mission concept. In 2018 IEEE Aerospace Conference, pages 1–20, 03 2018.

[54] Sonia Hernandez, Rodney Anderson, Duane Roth, Yu Takahashi, and Tim McElrath. Navigating low-energy trajectories to land on the surface of europa. In 31st AAS/AIAA Space Flight Mechanics Meeting, 2021.

[55] Christian Hofmann and Francesco Topputo. Rapid low-thrust trajectory optimization in deep space based on convex programming. Journal of Guidance, Control, and Dynamics, 44(7):1379–1388, 2021.

[56] K. C. Howell and H. J. Pernicka. Station-keeping method for libration point trajectories. Journal of Guidance, Control, and Dynamics, 16(1):151–159, 1993.

[57] Kathleen Howell and H. Pernicka. Stationkeeping method for libration point trajectories. Journal of Guidance Control and Dynamics, 16, 02 1990.

[58] Dario Izzo and Francisco Biscani. Audi/pyaudi. 2018.

[59] Dario Izzo and Ekin Öztürk. Real-time guidance for low-thrust transfers using deep neural networks. Journal of Guidance, Control, and Dynamics, 44(2):315–327, 2021.

[60] David H. Jacobson and David Q. Mayne. Differential dynamic programming. American Elsevier Pub. Co New York, 1970.

[61] Erica Jenson and Daniel Scheeres. Semianalytical measures of nonlinearity based on tensor eigenpairs. In 2021 AAS/AIAA Astrodynamics Specialist Conference, 2021.

[62] Erica L. Jenson, Xudong Chen, and Daniel J. Scheeres. Optimal control of sampled linear systems with control-linear noise. IEEE Control Systems Letters, 4(3):650–655, 2020.

[63] Erica L. Jenson and Daniel J. Scheeres. Multi-objective optimization of covariance and energy for asteroid transfers. Journal of Guidance, Control, and Dynamics, 44(7):1253–1265, 2021.

[64] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. Proceedings of the IEEE, 92(3):401–422, 2004.

[65] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. SIAM Review, 51(3):455–500, 2009.

[66] Nicholas B. LaFarge, Daniel Miller, Kathleen C. Howell, and Richard Linares. Guidance for closed-loop transfers using reinforcement learning with application to libration point orbits. In AIAA Scitech 2020 Forum, 2020.

[67] Nicholas B. LaFarge, Daniel Miller, Kathleen C. Howell, and Richard Linares. Autonomous closed-loop guidance using reinforcement learning in a low-thrust, multi-body dynamical environment. Acta Astronautica, 186:1–23, 2021.

[68] Try Lam, Brent Buffington, and Stefano Campagnola. A robust mission tour for nasa's planned europa clipper mission. In 2018 Space Flight Mechanics Meeting.

[69] Gregory Lantoine and Ryan P. Russell. A hybrid differential dynamic programming algorithm for constrained optimal control problems. part 1: Theory. J. Optim. Theory Appl., 154(2), 2012.

[70] Gregory Lantoine, Ryan P. Russell, and Thierry Dargent. Using multicomplex variables for automatic computation of high-order derivatives. ACM Transactions on Mathematical Software, 38(3):16:1–16:21, April 2012.

[71] Michele Maestrini, Pierluigi Di Lizia, Roberto Armellin, and Ryan P Russell. Hybrid differential dynamic programming algorithm for low-thrust trajectory design using exact high-order transition maps. In 69th International Astronautical Congress 2018, 10 2018.

[72] Manoranjan Majji, John Junkins, and Jay Turner. A high order method for estimation of dynamic systems. The Journal of the Astronautical Sciences, 56(3):401–440, 2008.

[73] Peter McCullagh. Tensor methods in statistics. 1987.

[74] Melissa McGuire, Laura Burke, Steven McCarty, Kurt Hack, Ryan Whitley, Diane Davis, and Cesar Ocampo. Low thrust cis-lunar transfers using a 40 kw-class solar electric propulsion spacecraft. In 2017 AAS/AIAA Astrodynamics Specialist Conference, 2017.

[75] Jay McMahon. High-order orbital guidance using state-transition tensors. In Proceedings of the 28th AAS/AIAA Space Flight Mechanics Meeting, 01 2018.

[76] Ali Mesbah. Stochastic model predictive control: An overview and perspectives for future research. IEEE Control Systems Magazine, 36(6):30–44, 2016.

[77] Andrea Carlo Morelli, Christian Hofmann, and Francesco Topputo. Robust low-thrust trajectory optimization using convex programming and a homotopic approach. IEEE Transactions on Aerospace and Electronic Systems, pages 1–1, 2021.

[78] Daniele Mortari. The theory of connections. connecting points. Mathematics, 5, 02 2017.

[79] Clark Newman, Diane C. Davis, Ryan J. Whitley, Joseph R. Guinn, and Mark S. Ryne. Stationkeeping, orbit determination, and attitude control for spacecraft in near rectilinear halo orbits. In 2018 AAS/AIAA Astrodynamics Specialist Conference, 2018.

[80] Jorge Nocedal and Stephen J. Wright. Numerical Optimization. Springer, New York, NY, USA, second edition, 2006.

[81] Kenshiro Oguri and Jay W. McMahon. Risk-aware trajectory design with impulsive maneuvers: Convex optimization approach. In 2019 AAS/AIAA Astrodynamics Specialist Conference, 2019.

[82] Kenshiro Oguri and Jay W. McMahon. Robust spacecraft guidance around small bodies under uncertainty: Stochastic optimal control approach. Journal of Guidance, Control, and Dynamics, 44(7):1295–1313, 2021.

[83] Kenshiro Oguri and Jay W. McMahon. Stochastic primer vector for robust low-thrust trajectory design under uncertainty. Journal of Guidance, Control, and Dynamics, 45(1):84–102, 2022.

[84] Kenshiro Oguri, Masahiro Ono, and Jay W. McMahon. Convex optimization over sequential linear feedback policies with continuous-time chance constraints. In 2019 IEEE 58th Conference on Decision and Control (CDC), pages 6325–6331, 2019.

[85] Masahiro Ono and Brian C. Williams. Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint. In 2008 47th IEEE Conference on Decision and Control, pages 3427–3432, 2008.

[86] Naoya Ozaki, Stefano Campagnola, Ryu Funase, and Chit Hong Yam. Stochastic differential dynamic programming with unscented transform for low-thrust trajectory design. Journal of Guidance, Control, and Dynamics, 41(2):377–387, 2018.

[87] Naoya Ozaki, Takayuki Yamamoto, Ferran Gonzalez-Franquesa, Roger Gutierrez-Ramon, Nishanth Pushparaj, Takuya Chikazawa, Diogene Alessandro Dei Tos, Onur elik, Nicola Marmo, Yasuhiro Kawakatsu, Tomoko Arai, Kazutaka Nishiyama, and Takeshi Takashima. Mission design of destiny+: Toward active asteroid (3200) phaethon and multiple small bodies. Acta Astronautica, 196:42–56, 2022.

[88] M. T. Ozimek and K. C. Howell. Low-thrust transfers in the earth-moon system, including applications to libration point orbits. Journal of Guidance, Control, and Dynamics, 33(2):533–549, 2010.

[89] Daniel Parcher and Gregory Whiffen. Dawn statistical maneuver design for vesta operations. In 21st AAS/AIAA Space Flight Mechanics Meeting, 2011.

[90] D.W. Parcher, Matthew Abrahamson, Alessandro Ardito, Dalila Han, R.J. Haw, B.M. Kennedy, Nickolaos Mastrodemos, S. Nandi, Ryan Park, B.P. Rush, B.A. Smith, J.C. Smith, A.T. Vaughan, and Gregory Whiffen. Dawn maneuver design performance at vesta. Advances in the Astronautical Sciences, 148, 02 2013.

[91] Inkwan Park and Daniel J. Scheeres. Hybrid method for uncertainty propagation of orbital motion. Journal of Guidance, Control, and Dynamics, 41(1):240–254, 2018.

[92] Ryan Park and Daniel Scheeres. Nonlinear mapping of gaussian statistics: Theory and applications to spacecraft trajectory design. Journal of Guidance, Control, and Dynamics, 29, 2006.

[93] Ryan S. Park and Daniel J. Scheeres. Nonlinear semi-analytic methods for trajectory estimation. Journal of Guidance, Control, and Dynamics, 30(6):1668–1676, 2007.

[94] Nathan L. Parrish. Low Thrust Trajectory Optimization in Cislunar and Translunar Space. PhD thesis, University of Colorado Boulder, 2018.

[95] Russell P. Patera. General method for calculating satellite collision probability. Journal of Guidance, Control, and Dynamics, 24(4):716–722, 2001.

[96] Joshua Pilipovsky and Panagiotis Tsiotras. Covariance steering with optimal risk allocation. IEEE Transactions on Aerospace and Electronic Systems, 57(6):3719–3733, 2021.

[97] Robert Pritchett. Strategies for Low-Thrust Transfer Design Based on Direct Collocation Techniques. PhD thesis, 08 2020.

[98] Jarrett Revels, Miles Lubin, and Theodore Papamarkou. Forward-mode automatic differentiation in julia. 2016.

[99] Javier Roa and Ryan S. Park. Reduced nonlinear model for orbit uncertainty propagation and estimation. Journal of Guidance, Control, and Dynamics, 0(0):1–15, 0.

[100] Javier Roa, Jess Pelez, and Juan Senent. New analytic solution with continuous thrust: Generalized logarithmic spirals. Journal of Guidance, Control, and Dynamics, 39(10):2336–2351, 2016.

[101] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In Proceedings of the 22nd International Conference on Machine Learning, 2005.

[102] Cody Short, Kathleen Howell, Amanda Haapala, and Donald Dichmann. Mode analysis for long-term behavior in a resonant earth-moon trajectory. Journal of the Astronautical Sciences, 64:156–187, 2017.

[103] C. Simo, G. Gomez, J. Llibre, R. Martinez, and J. Rodriguez. On the optimal station keeping control of halo orbits. Acta Astronautica, 15(6):391 – 397, 1987.

[104] Robert F. Stengel. Optimal Control and Estimation. Dover, 1994.

[105] Christopher J. Sullivan and Natasha Bosanac. Using reinforcement learning to design a low-thrust approach into a periodic orbit in a multi-body system. In AIAA Scitech 2020 Forum, 2020.

[106] Zhen-Jiang Sun, Pierluigi Di Lizia, Franco Bernelli-Zazzera, Ya-Zhong Luo, and Kun-Peng Lin. High-order state transition polynomial with time expansion based on differential algebra. Acta Astronautica, 163:45 – 55, 2019.

[107] Y. Tassa and E. Todorov. High-order local dynamic programming. In 2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), pages 70–75, 2011.

[108] Gabriel Terejanu, Puneet Singla, Tarunraj Singh, and Peter D. Scott. Uncertainty propagation for nonlinear dynamic systems using gaussian mixture models. Journal of Guidance, Control, and Dynamics, 31(6):1623–1633, 2008.

[109] Kirsten Tuggle and Renato Zanetti. Automated splitting gaussian mixture nonlinear measurement update. Journal of Guidance, Control, and Dynamics, 41(3):725–734, 2018.

[110] M. Valli, R. Armellin, P. Di Lizia, and M. R. Lavagna. Nonlinear mapping of uncertainties in celestial mechanics. Journal of Guidance, Control, and Dynamics, 36(1):48–63, 2013.

[111] M. Valli, R. Armellin, P. Di Lizia, and M. R. Lavagna. Nonlinear filtering methods for spacecraft navigation based on differential algebra. Acta Astronautica, 94(1):363–374, 2014.

[112] Michael P. Vitus and Claire J. Tomlin. On feedback design and risk allocation in chance constrained control. In 2011 50th IEEE Conference on Decision and Control and European Control Conference, pages 734–739, 2011.

[113] Allen Wang, Ashkan Jasour, and Brian C. Williams. Non-gaussian chance-constrained trajectory planning for autonomous vehicles in the presence of uncertain agents. CoRR, abs/2002.10999, 2020.

[114] Gregory Whiffen. Mystic: Implementation of the static dynamic optimal control algorithm for high-fidelity, low-thrust trajectory design. In AIAA/AAS Astrodynamics Specialist Conference, 2006.

[115] Jacob Williams, David E. Lee, Ryan J. Whitley, Kevin A. Bokelmann, Diane C. Davis, and Christopher Berry. Targeting cislunar near rectilinear halo orbits for human space exploration. In 27th AAS/AIAA Space Flight Mechanics Meeting, 2017.

[116] Alexander Wittig, Pierluigi Di Lizia, Roberto Armellin, Kyoko Makino, Franco Bernelli-Zazzera, and Martin Berz. Propagation of large uncertainty sets in orbital dynamics by automatic domain splitting. Celestial Mechanics and Dynamical Astronomy, 122(3):239–261, Jul 2015.

[117] Sidney Yakowitz. Algorithms and computational techniques in differential dynamic programming. volume 31 of Control and Dynamic Systems, pages 75 – 91. Academic Press, 1989.

[118] Ahmad Bani Younes, James Turner, Manoranjan Majji, and John Junkins. High-order uncertainty propagation enabled by computational differentiation. In Recent Advances in Algorithmic Differentiation, pages 251–260, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.