Distribution Statement A – Approved for Public Release

A Partial State Collocation Method for Covariance Optimal Control

by

Tyler J. Kapolka
B.S., United States Air Force Academy (2016)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Master of Science in Aeronautics and Astronautics
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2018

©2018 Tyler Joseph Kapolka, All rights reserved.
The author hereby grants to MIT and The Charles Stark Draper Laboratory, Inc. permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Author .....
Department of Aeronautics and Astronautics
May 24, 2018

Certified by .....
Steven R. Hall
Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by .....
Matthew J. Weinstein
Member of Technical Staff
The Charles Stark Draper Laboratory, Inc.
Thesis Supervisor

Accepted by .....
Hamsa Balakrishnan
Associate Professor of Aeronautis and Astronautics
Chair, Graduate Program Committee

Distribution Statement A – Approved for Public Release

A Partial State Collocation Method for Covariance Optimal Control

by

Tyler J. Kapolka

Submitted to the Department of Aeronautics and Astronautics
on May 24, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

An overview is presented for two methods of incorporating the covariance in an optimal control problem. Including the covariance in the optimal control problem can be beneficial in the field of navigation where it is desirable to design trajectories which either minimize navigation error or maximize observability for instrument calibration. The *full state collocation* method uses Legendre Gauss Radau collocation to discretize the deterministic states and controls as well as the unique elements of the covariance matrix. The problem is then transcribed to a nonlinear progamming problem (NLP) and is solved with an NLP solver. This method, however, results in problems with many constraints and variables, which is computationally expensive. The *partial state collocation* method, the main focus of this thesis, collocates the deterministic states and controls but uses a shooting method to incorporate the covariance matrix. The problem is then transcribed to a nonlinear programming problem, which has fewer constraints and variables than the full state collocation method.

Both of these methods are demonstrated by solving for the trajectory that minimizes the final position uncertainty for a spacecraft reentering Earth's atmosphere. The problem is tested with different sized covariance matrices, which shows how the time it takes to solve the problem increases as the covariance matrix increases in size. The partial state collocation method is generally faster and converges in fewer NLP iterations than the full state collocation method. As the covariance matrix increases in size, the time it takes to solve the problem increases at a smaller rate for the partial state collocation method.

Thesis Supervisor: Steven R. Hall
Title: Professor of Aeronautics and Astronautics

Thesis Supervisor: Matthew J. Weinstein
Title: Member of Technical Staff
The Charles Stark Draper Laboratory, Inc.

Distribution Statement A – Approved for Public Release

Acknowledgments

I would first like to thank my advisor at Draper, Dr. Matthew Weinstein, and my advisor at MIT, Prof. Steven Hall. Your insight and mentorship throughout my two years here have been invaluable. I would like to thank you both for taking time each week to meet with me and guide me through my research and writing my thesis. I would like to thank Draper, MIT, the Navy and the Air Force for giving me the opportunity to perform this research. Finally, I would like to thank my family for the support they have given me throughout the years. Your continual support and encouragement is the reason I have gotten to this point and completed my thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

This material is based upon work supported by, or in part by, the Department of the Navy's Strategic Systems Programs under contract number N00030-16-C-0014.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the Department of the Navy's Strategic Systems Programs.

The views expressed in this article are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution Statement A – Approved for Public Release

Contents

Distribution Statement A – Approved for Public Release

Distribution Statement A – Approved for Public Release

List of Figures

Distribution Statement A – Approved for Public Release

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution Statement A – Approved for Public Release

List of Tables

Distribution Statement A – Approved for Public Release

Distribution Statement A – Approved for Public Release

Chapter 1

Introduction

A traditional optimal control problem consists of an objective function, state dynamics, path constraints, and boundary conditions. The optimal control problem minimizes the objective function while satisfying the state dynamics, path constraints, and boundary conditions. This problem formulation, however, does not take into account the uncertainties in the states. There are certain problems where it is beneficial to optimize over the uncertainties in the states in addition to the problem states, controls, and time. One such application, where the uncertainty needs to be included in the optimal control problem, is planning the trajectory for a vehicle where the goal is to minimize the navigation errors. When planning a vehicle's trajectory, it is often important to ensure the navigation errors are not too large. Another application, where the uncertainty needs to be included in the optimal control problem, is to optimize the calibration technique for an instrument by maximizing the observability. By including the uncertainty in the optimal control problem, an objective function can be formulated that maximizes the observability of each error source. Another application is to plan a trajectory that maximizes the navigation errors. This trajectory could be used for sensitivity analysis. When planning a trajectory to test the sensitivity to errors, it can be beneficial to use a trajectory which maximizes the navigation errors.

There are multiple methods for solving optimal control problems and multiple ways of looking at the uncertainty in a problem. This thesis aims to introduce two

Distribution Statement A – Approved for Public Release

methods of incorporating uncertainty in an optimal control problem. This thesis will use the covariance matrix for calculating the uncertainty, and it will use the Legendre Gauss Radau (LGR) pseudospectral optimization method for solving the optimal control problem.

1.1 Covariance

There are a few ways of looking at the uncertainty in the states of a problem. Some methods of looking at the uncertainty use Monte Carlo analysis, the sensitivity matrix, and the covariance matrix. This thesis will only use the covariance matrix for tracking these uncertainties [1, 2, 3]. The covariance matrix includes the standard deviations of all of the states as well as the correlation coefficients between the standard deviations of different states. The covariance matrix is also used because its linearized dynamics can easily be calculated for any trajectory. Both methods allow the user to place elements of the final or initial covariance matrix in the objective function, and they allow bounds to be placed on the initial and final covariance matrices. The covariance can also be calculated when measurements are included. Measurements can be incorporated in the covariance through the Kalman filter, which allows for an additional type of problem to be solved. These two methods of including the covariance allows the trajectory to be optimized with respect to the uncertainty of the states, and the methods have the option of including continuous measurements.

1.2 Pseudospectral optimization methods

There are multiple ways of solving optimal control problems. One of the more prevalent ways is through the use of a pseudospectral method. These pseudospectral methods are a class of direct collocation, and they convert the optimal control problem into an NLP, and then an NLP solver can be used to find the optimal solution [4, 5, 6, 7, 8, 9]. The reason for choosing a direct collocation method for solving the optimal control problem is these methods are easy to implement, are robust with re-

Distribution Statement A – Approved for Public Release

gards to enforcing path constraints, and generally have good convergence properties. There are different types of pseudospectral methods, but this thesis will only look at the Legendre-Gauss-Radau method. The main difference between the different pseudospectral methods is how the collocation points are chosen. The differences will be further examined in chapter 2, but only the process of including the covariance in the LGR pseudospectral method is discussed. The principle of adding the covariance to any of these pseudospectral methods is the same and works for any of them, but the actual means of incorporating the covariance does vary.

1.3 Methods of Including Covariance in Optimal Control Problem

This thesis uses the covariance matrix for calculating the uncertainties in the states. It also uses the Legendre-Gauss-Radau (LGR) pseudospectral optimization method for solving the optimal control problem. Two different techniques for incorporating the covariance in the optimal control problem are included. The full state collocation method uses Legendre Gauss Radau collocation to discretize the deterministic states and controls as well as the unique elements of the covariance matrix. The problem is then transcribed to an NLP and is solved with an NLP solver. This method does not require that any changes be made to the LGR pseudospectral optimization structure. This method is the most intuitive way of including the covariance in the optimal control problem, but it causes the number of variables and constraints to grow rapidly, and thus, the time it takes to solve the problem increases rapidly. With the full state collocation method, the states, controls, and covariance must be calculated and stored at every collocation point, which results in a very large problem.

In an attempt to keep the time it takes to solve the problem from increasing so rapidly, the partial state collocation method was developed. This method combines the direct collocation of the states and controls with a shooting method for the covariance. This method includes the unique elements of the initial and final covariance

Distribution Statement A – Approved for Public Release

matrices as parameters in the problem and includes a constraint that the final covariance equals the initial covariance after it has been propagated to the final time. This method also requires the derivatives of these constraints to be calculated with respect to the states, controls, time, and parameters. Propagating the covariance forward and calculating the derivatives can also be a time consuming process. This method is meant to keep the problem size from increasing drastically while maintaining the convergence properties seen by direct collocation methods. This thesis will test to see how this partial state collocation method compares to the full state collocation method.

1.4 Thesis overview

This thesis first introduces the mathematical techniques necessary for including the covariance matrix in an optimal control problem. This background information includes how the covariance matrix is propagated with and without measurements and how the LGR pseudospectral optimization method works. Then, it describes the two processes of including the covariance in the optimal control problem. The full state collocation method is rather straight forward, while the partial state collocation method requires adding additional constraints and their derivatives to the LGR pseudospectral framework. Finally, these two methods are tested on a couple of example problems. The first example problem is designing a trajectory for a spacecraft reentry vehicle such that the final latitude and longitude uncertainties are minimized. The next example takes the same spacecraft but includes an inertial measurement unit (IMU). The goal of that problem is to minimize the north and east position error. The problem is run for different combinations of error sources from the IMU. By changing which error sources are included, the size of the covariance matrix can be varied. Varying the size of the covariance matrix shows how the time it takes to solve the problem changes as the covariance matrix increases in size. Both of the methods of including the covariance are tested on all the examples, and their solutions and run time are compared. These problems are also tested with and without an altimeter.

The altimeter provides measurements and allows for the Kalman filter to be tested. It also shows how the time varies with and without measurements.

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution Statement A – Approved for Public Release

Chapter 2

Background Mathematical Techniques

Multiple approaches can be taken for shaping a trajectory based on the uncertainty, all of which share two things in common. They use some method of calculating the uncertainty, and they use an optimizer to optimize over the uncertainty. How the uncertainty is calculated can vary depending on the approach, as can the technique used for optimizing over the trajectory. The purpose of this thesis is to examine two covariance-trajectory shaping techniques which utilize both linear covariance analysis for calculating the covariance and an LGR pseudospectral optimization method for solving the optimal control problem. The methods differ according to how the covariance is included in the LGR pseudospectral optimization technique, but the basis for both techniques is rooted in linear covariance analysis and LGR pseudospectral optimization. This chapter will review the theory and equations which make the linear covariance analysis and the LGR pseudospectral optimization method possible. This chapter will also review the continuous Kalman filter, which is needed so the covariance-trajectory shaping methods can be used for problems which include measurements. The Kalman filter does not change either method in any fundamental way; it only changes the equations used for calculating the covariance.

Distribution Statement A – Approved for Public Release

2.1 Covariance Matrix

In a stochastic system, the performance is not deterministic, so there will be uncertainty in the states, and in analyzing a system, it is beneficial to be able to track the uncertainty of the states. There are a number of ways in which that uncertainty can be determined, such as Monte Carlo simulations. Monte Carlo simulations use repeated random sampling to determine the error in the states. These methods are computationally expensive and are better suited for determining the error in the states after the trajectory has been determined. However, in this thesis the goal is to optimize over the variance of the states, and so Monte Carlo simulations do not work. An alternative is using the covariance matrix.

The covariance matrix consists of the variances of the states and the correlations between the states. A benefit to using the covariance matrix is that it can be propagated through a linear model. Because it is being propagated with a linear model, it must be assumed that any nonlinearities in the covariance matrix are small. The propagation through the linear model also means that the covariance can be quickly calculated analytically for any given trajectory, which is necessary when trying to optimize over elements of the covariance matrix. The linearized model also allows the derivative of the covariance matrix to be computed, which will be necessary when integrating the covariance matrix with the LGR pseudospectral optimization method.

2.1.1 Covariance Matrix Structure

The covariance matrix contains the statistical behavior of the stochastic system. Within the covariance matrix is the standard deviation of each state and the correlation coefficients between states. To calculate the covariance, let the uncertainties in the system model be expressed as random variables. Let $\mathbf{x}$ be a vector of random variables, and let x be a scalar random variable. The variance of a scalar value is calculated as

$$\sigma^2 = E[(x - \mu)^2] \ , \tag{2.1}$$

where σ is the standard deviation and μ is the mean of the scalar random variable. This variance calculation can be expanded for the covariance of a vector through the expectation

$$\mathbf{P} = \mathbf{E}[\delta\mathbf{x}\delta\mathbf{x}^{\mathrm{T}}] \ , \tag{2.2}$$

where

$$\delta\mathbf{x} = \mathbf{x} - \bar{\mathbf{x}} \ , \tag{2.3}$$

and $\bar{\mathbf{x}}$ is the mean of the vector of random variables, which is the same as the expected value of the random variables,

$$\bar{\mathbf{x}} = E[\mathbf{x}] \ . \tag{2.4}$$

Equation (2.3) can be expanded to

$$\mathbf{P} = \begin{bmatrix} E[\delta x_1 \delta x_1] & E[\delta x_1 \delta x_2] & \dots & E[\delta x_1 \delta x_n] \\ E[\delta x_2 \delta x_1] & E[\delta x_2 \delta x_2] & \dots & E[\delta x_2 \delta x_n] \\ \vdots & \vdots & \ddots & \vdots \\ E[\delta x_n \delta x_1] & E[\delta x_n \delta x_2] & \dots & E[\delta x_n \delta x_n] \end{bmatrix} \ , \tag{2.5}$$

which shows how each element of the covariance matrix is calculated.

The covariance matrix has a number of properties, which can be employed. As stated previously, the covariance matrix contains the standard deviation of each state as well as the correlation coefficients between the various states. The covariance matrix from Equation (2.5) can be rewritten as

$$\mathbf{P} = \begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_{1,2} & \dots & \sigma_1\sigma_n\rho_{1,\mathrm{n}} \\ \sigma_1\sigma_2\rho_{2,1} & \sigma_2^2 & \dots & \sigma_2\sigma_n\rho_{2,\mathrm{n}} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_n\sigma_1\rho_{\mathrm{n},1} & \sigma_n\sigma_2\rho_{\mathrm{n},2} & \dots & \sigma_n^2 \end{bmatrix} \ , \tag{2.6}$$

which includes standard deviations and correlation coefficients [10, 11, 12]. In Equation (2.6), σ_i^2 is the variance of state i, and $\rho_{i,j}$ is the correlation coefficient relating the standard deviation of state i with that of state j. $\mathbf{P}$ is a symmetric, positive-

Distribution Statement A – Approved for Public Release

semidefinite $n \times n$ matrix. It is clear from the above equation that the matrix is symmetric because the correlation of state i to state j is the same as the correlation of state j to state i. Because the matrix is symmetric, there are only $\frac{n(n+1)}{2}$ unique elements to the matrix. When storing these values, only the standard deviations and correlation coefficients need to be kept, and the entire matrix can be reconstructed from these elements.

The n states of the covariance matrix do not have to correspond directly to the original states used in the optimal control problem. While oftentimes it will be beneficial to examine the covariance of the system's states, nevertheless, the covariance of states not in the problem can be used instead. For example, in this thesis two example problems will be used. The first is a spacecraft reentering the atmosphere, and the goal is to minimize the variance of the final position of the vehicle. To solve this problem we will look at the covariance of the original states of the system. In the second example, we will continue to look at the same spacecraft, but this time with an IMU on board. The states of the system are the same as in the first example, but for the covariance matrix, we will use the states of the IMU instead of the states of the reentry vehicle. As long as the linearized dynamics of the covariance of the states whose uncertainty is being tracked is known and is a function of the original problem's states and controls, the covariance can be added to the optimal control problem.

2.1.2 Covariance Dynamics

To incorporate the covariance into the optimal control problem, the dynamics of the covariance matrix must be known. The previous section described what the covariance matrix is, and this section will address how to calculate the time derivative of the covariance matrix. In order to calculate the dynamics of the covariance matrix, the linearized state dynamics need to be known. If $\mathbf{x}$ is the state vector, the dynamics are

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{w} \ , \tag{2.7}$$

Distribution Statement A – Approved for Public Release

where $\mathbf{A}$ is the linearized state dynamics and $\mathbf{w}$ is the zero mean Gaussian process noise. The dynamics of the covariance matrix can be written as

$$\dot{\mathbf{P}} = \mathbf{AP} + \mathbf{PA}^{\mathrm{T}} + E[\mathbf{ww}^{\mathrm{T}}] \ . \tag{2.8}$$

Because the noise has a Gaussian distribution and the mean is zero,

$$E[\mathbf{w}] = 0 \ , \tag{2.9}$$

and

$$E[\mathbf{ww}^{\mathrm{T}}] = \mathbf{Q} \ , \tag{2.10}$$

where Q is the spectral densisty process noise. The covariance dynamics can be rewritten as

$$\dot{\mathbf{P}} = \mathbf{AP} + \mathbf{PA}^{\mathrm{T}} + \mathbf{Q} \ . \tag{2.11}$$

The A and Q matrices are problem dependent. Because of the symmetry of the covariance matrix, the Ricatti equation can be rewritten as

$$\dot{\mathbf{P}} = \mathbf{AP} + (\mathbf{AP})^{\mathrm{T}} + \mathbf{Q} \ , \tag{2.12}$$

which reduces the complexity because only one matrix multiplication and one matrix transpose need to be performed to calculate $\dot{\mathbf{P}}$, as opposed to the two matrix multiplications needed in the original form of the equation. In order for Equation (2.12) to be a good approximation of the covariance rate of change, the nonlinearities in the $\mathbf{A}$ matrix must be small. This equation does not take into account measurements, but the Kalman Filter, introduced later in this chapter, will describe how the equation changes to incorporate measurements.

Distribution Statement A – Approved for Public Release

2.2 Legendre Gauss Radau Pseudospectral Optimization Method

There are multiple approaches that can be taken for solving an optimal control problem, but this thesis will only address using a pseudospectral method because they transform an optimal control problem into a nonlinear programming problem (NLP), which allows the problem to be solved by an NLP solver such as SNOPT or IPOPT. This transformation is accomplished by parameterizing the states and controls and collocating the differential-algebraic equations.

The continuous optimal control problems solved by these pseudospectral methods take on the same structure. These optimal control problems consist of an objective function to be minimized, state dynamics to be satisfied, path constraints to be satisfied, and boundary conditions to be satisfied. The objective function is of the form

$$J = \Phi(\mathbf{x}_\mathrm{f}, t_\mathrm{f}, \mathbf{x}_\mathrm{i}, t_\mathrm{i}) + \int g(\mathbf{x}, \mathbf{u}, \mathbf{t})dt \ . \tag{2.13}$$

The state dynamics to be satisfied are

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{t}) \ . \tag{2.14}$$

The constraints take the form

$$c(\mathbf{x}, \mathbf{u}, \mathbf{t}) \leq 0 \ . \tag{2.15}$$

The boundary conditions that must be enforced are

$$\phi(\mathbf{x}_\mathrm{f}, t_\mathrm{f}, \mathbf{x}_\mathrm{i}, t_\mathrm{i}) \leq 0 \ . \tag{2.16}$$

Collocation involves representing the states and controls over the given interval by a polynomial and then choosing a number of points on the interval where the constraints must be enforced. Gauss collocation methods are commonly used for determining the collocation points. The Gauss collocation points are chosen to give the best accuracy for whichever form of Gaussian quadrature being used. These

Distribution Statement A – Approved for Public Release

points are defined over the domain [-1,1], so the time interval $\tau \in [-1, 1]$ will be used. This interval can be transformed to the interval $[t_0, t_f]$ through the affine transform, which is given by

$$\mathbf{t} = \frac{t_f - t_0}{2}\tau - \frac{t_f + t_0}{2} \ . \tag{2.17}$$

Three common forms of Gauss collocation are Legendre Gauss (LG), Legendre Gauss Lobatto, and Legendre Gauss Radau (LGR), which is the technique used in this thesis. Each of these collocation methods choose the nodes for different forms of Gauss quadrature. The nodes are obtained from the roots of a Legendre polynomial, roots of the derivative Legendre polynomial, or some linear combination of the aforementioned polynomials. The LG method lies on the interval (-1,1), and the collocation points do not include either endpoint. The LGL method lies on the interval [-1,1], and the collocation points contain both endpoints. The LGR method lies on the interval (-1,1] or [-1,1) and contains either the left or right endpoint [8, 9, 5]. For this paper, the LGR method will contain the left endpoint as a collocation point. The collocation points for the form of the Gauss quadrature used for the LG method are the roots to the Legendre polynomial

$$P_N(\tau) \ , \tag{2.18}$$

where P_N is a Legendre polynomial of degree N, and N is the number of collocation points. The LGL collocation points are the roots of

$$\dot{P}_{N-1}(\tau) \ , \tag{2.19}$$

along with -1 and 1. The LGR collocation points are the roots of the polynomial

$$P_N(\tau) + P_{N-1}(\tau) \ . \tag{2.20}$$

The Legendre polynomials are defined recursively by

$$(n+1)P_{n+1}(\tau) = (2n+1)tP_n(\tau) - nP_{n-1}(\tau), n \geq 1 \ , \tag{2.21}$$

Distribution Statement A – Approved for Public Release

where

$$P_N(1) = 1 \ . \tag{2.22}$$

For the recursion the first two polynomials must be known, and they are

$$\begin{aligned} P_0(\tau) &= 1 \\ P_1(\tau) &= \tau \ . \end{aligned} \tag{2.23}$$

The LG, LGL, and LGR collocation points can all be used with the pseudospectral method, but the different collocation methods do change the way the problem is solved, with the biggest difference being in the structure of the differentiation matrix. The resulting differentiation matrix from the LG and LGR methods is nonsingular and invertible, while the differentiation matrix from the LGL method may not be invertible [5]. The LGR method has a benefit over the LG method because one of the endpoints is contained as a collocation point. For these reasons the LGR method is used in this thesis, and only the framework for the LGR method will be presented.

One of the steps to the pseudospectral method is to collocate the differential algebraic equations. To begin this step, the states are approximated by polynomials of degree at most N. The collocation points along with an additional point at $\tau_{N+1} = 1$ are used to create a basis of Lagrange polynomials. The collocation points include $\tau_0 = -1$ and $N-1$ interior points, but it does not contain the right endpoint, and so it is added here as an additional point for approximating the states. The polynomials used to approximate the states are at most order N, but $N+1$ points are used to approximate the state, which provides an additional degree of freedom. The i-th element of the state is approximated by

$$x_i(\tau) = \sum_{i=1}^{N+1} \mathbf{X_i} L_i(\tau) \ , \tag{2.24}$$

where i in $\mathbf{X_i}$ is the row of the state matrix and L_i is a basis of Lagrange polynomials given by

$$L_i(\tau) = \prod_{i=1, j \neq i}^{N+1} \frac{\tau - \tau_j}{\tau_i - \tau_j}, \quad (i = 1, \ldots, N+1) \ . \tag{2.25}$$

The time derivative of Equation (2.24) is

$$\dot{\mathbf{x}}(\tau) = \sum_{i=1}^{N+1} \mathbf{X_i} \dot{L}_i \ , \tag{2.26}$$

which is equivalent to

$$\dot{\mathbf{x}}(\tau) = \sum_{i=1}^{N+1} D_{\mathrm{ki}} \mathbf{X_i} \ , \tag{2.27}$$

where $\mathbf{D}$ is the $N \times N+1$ differentiation matrix.

The dynamic constraint ensures that the derivatives of the Lagrange polynomials at the collocation points are equal to the differential algebraic equations at the collocation points. Let the state matrix $\mathbf{X}$ contain the states at each of the collocation points as well as the state at $\tau = 1$, so the state matrix is $N+1 \times n$, where n is the number of states. The derivatives, which are a product of $\mathbf{D}$ and $\mathbf{X}$, are only for the states at the collocation points and can be expressed as $\dot{\mathbf{X}}_{\mathbf{1:N}}$. Therefore, the derivatives calculated from differential algebraic equations are only functions of the collocated states, $\mathbf{X_{1:N}}$. To ensure the collocated states satisfy the differential equations, the dynamic constraints can be written as

$$\sum_{i=1}^{N+1} D_{\mathrm{ki}} \mathbf{X_i} - \frac{t_f - t_0}{2} \mathbf{f}(\mathbf{X_k}, \mathbf{U_k}, \tau_k; t_0, t_f), (k = 1, \ldots, N) \ . \tag{2.28}$$

Equation (2.28) only ensures the dynamic constraints are satisfied at the collocation points and does not include the right endpoint. In Equation (2.28), $\mathbf{X_k} \equiv \mathbf{X}(\tau_k)$, $\mathbf{U_k} \equiv \mathbf{U}(\tau_k)$, $(k = 1, \ldots, N)$, and $(i = 1, \ldots, N+1)$. Including the additional column in the $\mathbf{D}$ matrix, from the noncollocated state at $\tau = 1$, provides an additional degree of freedom and makes the matrix nonsingular. As stated previously, one of the reasons the LGR method is used over the LGL method is because the $\mathbf{D_{1:N}}$ matrix is nonsingular, and this property makes the correspondence between control and state much less complex for LGR than it is for the LGL method [5].

In addition to the dynamic constraints needing to be satisfied, the continuous objective function must also be written as a function of states and controls at the

collocation points and at the right endpoint. The continuous objective function is ordinarily in the form

$$J = \Phi(\mathbf{x}_0, t_0, \mathbf{x}_f, t_f) + \int_{t_0}^{t_f} g(\mathbf{x}, \mathbf{u}, \mathbf{t})\mathbf{dt} \ . \tag{2.29}$$

In order to rewrite the objective function in terms of the collocated states and controls and the right endpoint, $\mathbf{X}(t_0)$ is replaced with $\mathbf{X_1}$, which is the first row of the state matrix, and $\mathbf{X}(t_f)$ is replaced with $\mathbf{X_{N+1}}$, which is the last row of the state matrix and is the state at the right endpoint. The integral component of the cost function can be replaced with a Gauss quadrature, which results in the objective function,

$$J = \Phi(\mathbf{X_1}, t_0, \mathbf{X_N}, t_f) + \frac{t_f - t_0}{2} \sum_{k=1}^{N} w_k g(\mathbf{X_k}, \mathbf{U_k}, \tau_k; t_0, t_f) \ , \tag{2.30}$$

where w_k are the quadrature weights associated with the LGR points.

In addition to the dynamic constraints and the objective function, there are boundary conditions and path constraints which must be satisfied. Similar to the cost function, the boundary conditions and path constraints are written in terms of the state at the first collocation point and the state at the right endpoint. The boundary conditions can be written as

$$\phi(\mathbf{X_1}, t_0, \mathbf{X_{N+1}}, t_f) = 0 \ . \tag{2.31}$$

The path constraints, which are only evaluated on the LGR collocation points, can be writen as

$$\mathbf{C}(\mathbf{X_k}, \mathbf{U_k}, \tau_k; t_0, t_f) \leq 0, \quad (k = 1, \ldots, N) \ . \tag{2.32}$$

The optimization problem is now in a format which allows it to be solved by an NLP solver, where Equation (2.30) is the objective function subject to the constraints given by Equations (2.28), (2.31), and (2.32).

Distribution Statement A – Approved for Public Release

2.3 Extended Kalman Filter

Section 2.1 outlined the dynamics for the covariance matrix without measurements. This section will examine using a Kalman filter so that the problem can be solved with measurements [13]. The measurements, $\mathbf{y}$, are calculated as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{z} \ , \tag{2.33}$$

where $\mathbf{z}$ is the zero-mean Gaussian noise and $\mathbf{H}$ is the $k \times n$ observation matrix. Because the noise is zero-mean, the expected value is

$$E[\mathbf{z}] = 0 \tag{2.34}$$

and

$$E[\mathbf{z}\mathbf{z}^{\mathrm{T}}] = \mathbf{R} \ , \tag{2.35}$$

where $\mathbf{R}$ is the $k \times k$ observation noise matrix. Because the Kalman filter is an optimal estimator for linear systems with Gaussian noise, it works for problems with linear state dynamics, and the Extended Kalman filter is for problems with nonlinear state dynamics. The EKF works the same way as the Kalman filter except that the state dynamics are linearized about the current point. As long as the linearization is a close approximation to the true dynamics, the EKF works fairly well.

The trajectory shaping method used in this thesis works in continuous time, so the continuous time Kalman filter equations will be used instead of the more common discrete time Kalman filter equations. The continuous Kalman filter has two parts. The first involves calculating the state dynamics with the measurements, which is given by

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \mathbf{F}(t)\hat{\mathbf{x}}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{K}(t)(\mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t) - \mathbf{H}(t)\hat{\mathbf{x}}(t)) \ . \tag{2.36}$$

The second involves calculating the time derivative of the covariance matrix, which

Distribution Statement A – Approved for Public Release

is given by

$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^{\mathrm{T}}(t) + \mathbf{Q}(t) - \mathbf{K}(t)\mathbf{R}(t)\mathbf{K}^{\mathrm{T}}(t) \ . \tag{2.37}$$

$\mathbf{K}$ is the Kalman gain, given by

$$\mathbf{K}(t) = \mathbf{P}(t)\mathbf{H}^{\mathrm{T}}(t)\mathbf{R}^{-1}(t) \ . \tag{2.38}$$

Only the continuous Kalman filter equation for the covariance matrix is needed for adding covariance to the LGR Pseudospectral optimization method. The covariance differential equation from the Kalman filter is independent of the measurements, so it can be calculated without having the measurement values. However, the covariance differential equation does require knowledge of the measurement matrix. As long as there is knowledge of what the initial covariance matrix is and what is being measured, the covariance matrix can be propagated to show how the covariance changes over time if measurements are used, without having to incorporate any measurements. The aim of this thesis is to develop a method of optimizing a trajectory relative to elements of the covariance matrix. It does not use measurements to update the state, but since the covariance differential equation is not dependent on the measurements, the algorithm is still able to accomplish its goal of optimizing a trajectory relative to elements of the covariance matrix when measurements are being taken, even though no measurements are actually used in the problem.

Distribution Statement A – Approved for Public Release

Chapter 3

Including Covariance in the LGR Pseudospectral Optimization Method

The LGR Pseudospectral method optimizes a trajectory by minimizing an objective function while ensuring all necessary constraints are satisfied. The objective function is a function of the states, controls, and time. Once a solution has been generated, analysis on the variance of the states can be performed. Performing this analysis after the solution has been generated provides insight into the errors of the states, but it does not allow for the trajectory to be optimized relative to the variance of the states, and it does not allow any constraints to be placed on the variances. This chapter will introduce two methods of including the covariance in the LGR Pseudospectral method so that bounds can be placed on elements of the covariance matrix and those elements can be included in the objective function.

The optimization problem to be solved will take on a similar form to the optimal control problem introduced in chapter 2, except this problem will include the covariance matrix. The cost function will still be a function of the states, controls, and time, but it can also be a function of the initial and final covariance matrices. The cost function can be written as

$$J = \Phi(\mathbf{x}_f, t_f, \mathbf{x}_i, t_i, \mathbf{P}_i, \mathbf{P}_f) + \int g(\mathbf{x}, \mathbf{u}, \mathbf{t})dt \ . \tag{3.1}$$

Distribution Statement A – Approved for Public Release

The state dynamics are still

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{t}) \ . \tag{3.2}$$

For this problem, in addition to satisfying the state dynamics, the covariance dynamics must also be satisfied. The covariance dynamics are given by

$$\dot{\mathbf{P}} = \mathbf{AP} + \mathbf{PA}^{\mathrm{T}} + \mathbf{Q} - \mathbf{KRK}^{\mathrm{T}} \ , \tag{3.3}$$

where $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ are all functions of states, controls, and time. For the covariance dynamics, $\mathbf{KRK}^{\mathrm{T}}$ goes away if there are no measurements for the problem. One of the reasons this problem formulation is possible is the covariance can be separated from the other states. The covariance is a function of other problem states, but the problem states are not a function of the covariance. For this reason, the covariance can be treated differently than the other states. This problem formulation does not allow path constraints on the covariance, so the path constraints are still of the form

$$c(\mathbf{x}, \mathbf{u}, \mathbf{t}) \leq 0 \ . \tag{3.4}$$

Boundary conditions can be placed on the initial and final covariance matrices, so the boudary conditions can be written as

$$\phi(\mathbf{x}_f, t_f, \mathbf{x}_i, t_i, \mathbf{P}_i, \mathbf{P}_f) \ . \tag{3.5}$$

3.1 Full State Collocation

The first technique for adding covariance to the optimal control problem is including elements of the covariance matrix as additional states of the problem. This method does not require any changes to be made to the LGR pseudospectral method. This approach is the most intuitive way of including the covariance in the optimal control problem. For an optimal control problem, the state and control boundary conditions and state dynamics must be provided. In this formulation, the covariance is included as a state and the covariance dynamics from equation (2.37) are added to the original

problem's dynamics. Since the elements of the covariance matrix are treated as states, path constraints can be enforced, boundary conditions can be enforced, and they can be included in the objective function.

In the optimal control problem without the covariance, there are n states, so the state vector is $\mathbf{x} \in \mathbb{R}^{n\times 1}$, and there are m controls, so the control vector is $\mathbf{u} \in \mathbb{R}^{m\times 1}$. The covariance matrix tracks the variances of z states. These z states could be the same as the n states used in the original problem, or they could be different, as long as the linearized dynamics of the covariance matrix are a function of the original problem's states and controls. The resulting covariance matrix is $z \times z$, but because the matrix is symmetric, we do not need to include every element of the matrix as an additional state to the problem, which would require an additional z^2 states. Instead we can just add the variances and the correlation coefficients as states to the problem, which results in $\dfrac{z(z+1)}{2}$ additional states. The resulting state vector will be denoted as $\mathbf{x_a}$ because elements of the covariance matrix have been adjoined to the original state vector, $\mathbf{x}$. The new state vector takes on the following structure

$$\mathbf{x_a} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ \sigma_1 \\ \vdots \\ \sigma_z \\ \rho_{1,2} \\ \rho_{1,3} \\ \rho_{2,3} \\ \vdots \\ \rho_{z-1,z} \end{bmatrix}. \tag{3.6}$$

Other than adjoining elements of the covariance matrix to the state vector and including the dynamics of the covariance matrix, there are no other changes to the problem. A benefit of this covariance shaping technique is it is not specific to the

Distribution Statement A – Approved for Public Release

LGR pseudospectral method. It can be applied to any method of solving an optimal control problem without needing to make any adjustments. However, one of the drawbacks to this approach is the rate at which the size of the state matrix increases, and correspondingly how long it takes to solve the problem. The other method of adding covariance to the optimal control problem aims to solve this issue.

3.2 Partial State Collocation

For the second method, instead of adding elements of the covariance matrix to the state vector, the initial and final covariance matrices are included in the problem as parameters. The state vector and control vector remain unchanged from the original control problem. Similarly to when the covariance was included as a state, not every element of the covariance matrix needs to be included as a parameter. Only the $\frac{z(z+1)}{2}$ unique elements, in the form of the variances and correlation coefficients, will be stored as parameters and then the covariance matrix can be constructed from these elements. The parameterized states and controls are used to propagate the covariance matrix from the initial time to the final time. Then a constraint is added that ensures the parameters representing the final covariance are equal to the propagation of the initial covariance. Boundary conditions can then be placed on the parameters, and the parameters can be included in the objective function. From henceforth this method will be referred to as the partial state collocation method because the states and controls are solved using the LGR pseudospectral framework, but the covariance is just propagated forward from the initial to the final time like in a shooting method.

The benefit to this partial state collocation method is it requires fewer states to be used for solving the problem, but it requires changing the LGR pseudospectral method. The change in the LGR pseudospectral method comes from having to enforce the constraint that the final covariance is equal to the initial covariance after it has been propagated to the final time. Including this constraint in the LGR pseudospectral method has two parts. The first is propagating the covariance from

the initial time to the final time, and the second part is taking the derivatives of the parameters with respect to the other parameters, the states, the controls, and the time. The covariance propagation is accomplished by using the Kalman filter to calculate the time rate of change of the covariance matrix and the Runge-Kutta 4 (RK4) integration scheme to then propagate the covariance [14]. The derivatives will be calculated by using adjoint differentiation.

3.2.1 Runge-Kutta 4

Runge-Kutta is a class of numerical methods for integrating ordinary differential equations. The differential equation being integrated here is the covariance differential equation. The difference between the Runge-Kutta methods is at how many points they calculate the slope. The RK4 scheme being used here calculates the slope at four points and then averages these slopes before propagating to the next time step. The first step is

$$\mathbf{k}_1 = \dot{\mathbf{P}}(\mathbf{A}_n, \mathbf{Q}_n, \mathbf{H}_n, \mathbf{R}_n, \mathbf{P}_n) \ , \tag{3.7}$$

where $\mathbf{k}$ is the time rate of change of the covariance matrix, $\mathbf{A}_n$ is the linearized system dynamics at time n, $\mathbf{Q}_n$ is the process noise at n, and $\mathbf{P}_n$ is the covariance at time n. This covariance time derivative is used to propagate the covariance matrix forward by half a time step so that the next covariance time derivative can be calculated by

$$\mathbf{k}_2 = \dot{\mathbf{P}}(\mathbf{A}_{n+1/2}, \mathbf{Q}_{n+1/2}, \mathbf{H}_{n+1/2}, \mathbf{R}_{n+1/2}, \mathbf{P}_n + \frac{h}{2}\mathbf{k}_1) \ , \tag{3.8}$$

where h is the time step. The final two slopes needed for the RK4 method are given by

$$\mathbf{k}_3 = \dot{\mathbf{P}}(\mathbf{A}_{n+1/2}, \mathbf{Q}_{n+1/2}, \mathbf{H}_{n+1/2}, \mathbf{R}_{n+1/2}, \mathbf{P}_n + \frac{h}{2}\mathbf{k}_2) \tag{3.9}$$

and

$$\mathbf{k}_4 = \dot{\mathbf{P}}(\mathbf{A}_{n+1}, \mathbf{Q}_{n+1}, \mathbf{H}_{n+1}, \mathbf{R}_{n+1}, \mathbf{P}_n + h\mathbf{k}_3) \ . \tag{3.10}$$

Distribution Statement A – Approved for Public Release

The four slopes are then averaged, with the second and third weighted higher than the other two, and the covariance is propagated to the next time step, as shown by

$$\mathbf{P}_{n+1} = \mathbf{P}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4) \ . \tag{3.11}$$

In order to perform the RK4 integration, $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{H}$ must be known at each time n and at the halfway point between each time n. Because $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{H}$ are functions of the states, controls, and time, these matrices are known at every point where the states and controls are known. If the initial covariance matrix is known as well as the states at every step n and every point halfway between each n, then RK4 can be used to propagate the covariance to the final time. From the LGR pseudospectral method, the states are only known at the collocation points and the right endpoint. These collocation points are not evenly spaced, and the value of the state at the halfway point between nodes is not known. The collocation points do not provide any flexibility with determining the step size. In general, the step size can be decreased for better accuracy, but it will take more time and more iterations to calculate the final covariance, or the step size can be increased for less accuracy but it takes less time and fewer iterations to get the final covariance. In order to gain this flexibility in the step size, and in order to know the states at the halfway points between steps, Lagrange polynomials are used to approximate the states and controls at any time.

The Lagrange polynomials will be generated the same way as in Equation (2.24), and the states and controls at the collocation points will be used to calculate the states and controls at the times needed for the RK4 integration. Before interpolating the states and controls, the time steps for the RK4 integration need to be determined. For this thesis, rather than using one constant step size, we set a maximum value for the step size, but allow the step sizes to vary. Initially the midpoint between each collocation point and the midpoint between the final collocation point and the right endpoint were determined. If any of these steps was greater than the max step size then the midpoint of that step was determined. This process continued

Distribution Statement A – Approved for Public Release

until each step was less than the maximum step size. The symmetry of the process ensures that each step is divided into two equal parts even though each step is not the same size. The reason equally spaced steps are not used is because with the mesh refinement used in solving the problem, more collocation points are placed in less stable regions. Because there are more collocation points in those regions, the step size for the covariance propagation will be smaller. Allowing the step sizes to be based on the number of collocation points in a region allows smaller steps to be taken in less stable regions without wasting time on having small steps in the stable regions.

3.2.2 Covariance Constraint

With this problem formulation, there must be a constraint placed on the parameters which ensures the final covariance parameters equal the initial covariance parameters after they have been propagated. Rather than including this constraint, it is possible to directly add elements of the final covariance to the objective function, which would eliminate the need to store the initial and final covariance matrices as parameters. Not including the parameters would have no impact on the propagation, but it would affect how the derivatives are calculated. The main benefit from not including the parameters is that when calculating the derivatives, the derivatives could be calculated directly from the cost function. In the current form, there are $\frac{z(z+1)}{2}$ constraints for the parameters and the derivative of each of these constraints with respect to every state, control, time, and other parameter needs to be calculated and added to the Jacobian. If the covariance matrices were not stored as parameters, but directly incorporated in the cost function, then only the derivative of the cost with respect to the states, controls, and time would need to be included in the Jacobian. The derivative of the final covariance with respect to the states, controls, and time would still be calculated, but it would all be wrapped into the derivative of the cost with respect to the states, controls, and time. Although this method would decrease the size of the Jacobian, it gives less control over the covariance. No bounds could be placed on the final covariance matrix and the initial covariance matrix would have to be pro-

Distribution Statement A – Approved for Public Release

vided. With the method where the initial and final covariance matrices are included as parameters, bounds can be placed on the final and initial covariance. Rather than having to provide the exact value of the initial covariance, upper and lower bounds can be provided and then the optimal value can be chosen in the solution. In order to have more control over the covariance matrix, the covariance will be included as parameters, and the derivatives of the constraints applied to those parameters will be calculated using adjoint differentiation.

3.3 Adjoint Differentiation

There are a number of ways to calculate the derivatives, and two of the common methods are the forward mode and adjoint differentiation. The forward mode starts at the beginning of an algorithm and calculates the derivatives at each line in the algorithm, combining them with the other derivatives through the chain rule. Adjoint differentiation is a method of calculating derivatives in the reverse order in which the original algorithm runs. Both methods rely on repeatedly using the chain rule to calculate the derivatives. The adjoint differentiation works in the opposite direction, starting with the derivatives at the end of the algorithm and working its way to the beginning. For cases where there are many variables which converge to fewer variables, adjoint differentiation generally runs faster than the forward mode. In this problem there are many states and controls, all of whichl converge to $\frac{z(z+1)}{2}$ parameters, so it makes sense to use the adjoint differentiation instead of the forward mode.

The adjoint differentiation needs to be developed for the Kalman filter and for the RK4 integration. Because these functions work for any covariance matrix and are not problem dependent, the adjoint code only needs to be developed once, and then it can be used for any covariance matrix. The adjoint code will only give the derivatives of the constraints with respect to the $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ matrices from the Kalman filter. To get the derivatives of the constraints with respect to the states, controls, and time, the derivative of the $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ matrices must be computed. Because these matrices are problem dependent, there is no generalized adjoint code

Distribution Statement A – Approved for Public Release

that can be used; instead, some automatic differentiation software must be used to calculate these derivatives, or they must be provided by the user. The adjoint code starts with the derivative of the parameters with respect to the constraints applied to them. Using those derivatives as the starting point, adjoint differentiation can be used to get the derivatives of the constraints with respect to the states, controls, and time.

We will look at a sample line of code to show how the adjoint method is applied. For this example we will assume the result of the algorithm is some cost, J. The original algorithm would already have been run, and we have J. Let a line of code from the original algorithm be

$$y = z_1^2 + 5z_2 \ . \tag{3.12}$$

Running the adjoint differentiation back to this line, we would have $\frac{\partial J}{\partial y}$. The adjoint for this line will result in $\frac{\partial J}{\partial z_1}$ and $\frac{\partial J}{\partial z_2}$ and is given by

$$\frac{\partial J}{\partial z_1} = \frac{\partial J}{\partial z_1} + \frac{\partial J}{\partial y} 2z_1 \tag{3.13}$$

and

$$\frac{\partial J}{\partial z_2} = \frac{\partial J}{\partial z_2} + \frac{\partial J}{\partial y} 5 \ . \tag{3.14}$$

In general, the chain rule applied in the adjoint differentiation takes the form [15]

$$\frac{\partial J}{\partial x} = \left(\left(\left(\left(\ldots\left(\frac{\partial F_k}{\partial z_{k-1}}\frac{\partial F_{k-1}}{\partial z_{k-2}}\right)\ldots\right)\frac{\partial F_3}{\partial z_2}\right)\frac{\partial F_2}{\partial z_1}\right)\frac{\partial F_1}{\partial x}\right) \ . \tag{3.15}$$

Because the covariance is propagated before the derivatives are calculated, all of the intermediate values must be stored, so they can be used when calculating the derivatives. Because the RK4 propagation is an iterative process, this can result in many values needing to be stored. This additional storage is one of the disadvantages of using adjoint differentiation. It may require fewer calculations compared to a forward mode, but it requires more storage.

In order to speed up the derivative calculations, the derivatives of the $\mathbf{A}$, $\mathbf{Q}$,

$\mathbf{H}$, and $\mathbf{R}$ matrices are not calculated in each loop. Because the $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ matrices are only dependent on the states, controls, and time and not the covariance, the $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ matrices are known at all times and so are their derivatives. The derivatives of the $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ matrices can be calculated before the derivatives of the parameters are calculated. These derivatives are then stored and incorporated into the adjoint differentiation loop, so the output is the derivative of the covariance constraint with respect to the states, controls, and time. The resulting derivatives are used to calculate the derivatives of the final covariance with respect to all of the interpolated states, controls, and time and not to the states and controls at the collocation points. It is with respect to the states and controls at the collocation points that the derivatives are needed. In order to obtain the correct derivatives, the adjoint code is continued to include the code that was used to interpolate the states and controls. Extending the adjoint to include the interpolation gives the correct derivatives for the Jacobian.

3.3.1 Adjoint Application

This section will show how the adjoint differentiation is applied to the RK4 and Kalman filter algorithms. The inputs to the Runge Kutta 4 integration are the states, controls, times, and initial σ and ρ values, and the outputs are the final σ and ρ values. The Runge Kutta 4 algorithm calculates $\dot{\mathbf{P}}$ four times and then takes an average of these values to propagate the covariance to the next step. Equations (3.7) to (3.11) show the general form of the RK4 algorithm. Those equations will be redefined here to put them in a format where it will be easier to show how the adjoint differentiation is applied. The first equations are

$$\mathbf{k}_{1,i} = \dot{\mathbf{P}}(\mathbf{A}_i(\mathbf{x}_i, \mathbf{u}_i, t_i), \mathbf{Q}_i(\mathbf{x}_i, \mathbf{u}_i, t_i), \mathbf{H}_i(\mathbf{x}_i, \mathbf{u}_i, t_i), \mathbf{R}_i(\mathbf{x}_i, \mathbf{u}_i, t_i), \mathbf{P}_i) \tag{3.16}$$

and

$$\mathbf{P}_{\text{temp1,i}} = \mathbf{P}_i + h\mathbf{k}_1/2 \ , \tag{3.17}$$

where $\dot{\mathbf{P}}$ is being calculated at step i, and then the covariance is propagated to $i+1$, which is the halfway point between steps. The steps occur at every other i, but the halfway points are needed for the RK4 integration. The next three covariance derivatives are calculated and the covariance is propagated, and these calculations are given by

$$\begin{aligned}\mathbf{k}_{2,i} = \dot{\mathbf{P}}(&\mathbf{A}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}), \mathbf{Q}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}), \mathbf{H}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}),\\ &\mathbf{R}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}), \mathbf{P}_{\text{temp1,i}})\end{aligned} \tag{3.18}$$

$$\mathbf{P}_{\text{temp2,i}} = \mathbf{P}_i + h\mathbf{k}_2/2 \tag{3.19}$$

$$\begin{aligned}\mathbf{k}_{3,i} = \dot{\mathbf{P}}(&\mathbf{A}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}), \mathbf{Q}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}), \mathbf{H}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}),\\ &\mathbf{R}_{i+1}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, t_{i+1}), \mathbf{P}_{\text{temp2,i}})\end{aligned} \tag{3.20}$$

$$\mathbf{P}_{\text{temp3,i}} = \mathbf{P}_i + h\mathbf{k}_3 \tag{3.21}$$

$$\begin{aligned}\mathbf{k}_{4,i} = \dot{\mathbf{P}}(&\mathbf{A}_{i+2}(\mathbf{x}_{i+2}, \mathbf{u}_{i+2}, t_{i+2}), \mathbf{Q}_{i+2}(\mathbf{x}_{i+2}, \mathbf{u}_{i+2}, t_{i+2}), \mathbf{H}_{i+2}(\mathbf{x}_{i+2}, \mathbf{u}_{i+2}, t_{i+2}),\\ &\mathbf{R}_{i+2}(\mathbf{x}_{i+2}, \mathbf{u}_{i+2}, t_{i+2}), \mathbf{P}_{\text{temp3,i}}) \ .\end{aligned} \tag{3.22}$$

The $\mathbf{P}_{temp}$ values in Equations (3.17), (3.19), and (3.21) are intermediary values for the covariance matrix. These values are used for calculating the slopes at different points but are not the actual value of the covariance matrix at any time. The entire RK4 loop is run before the derivatives are calculated, and the values for each of these slopes and temporary covariance matrices for each iteration need to be stored for use with the adjoint differentiation. The slopes are then used to propagate the covariance to the next step, $i+2$, which is shown by

$$\mathbf{P}_{i+2} = \mathbf{P}_i + h_i/6(\mathbf{k}_{1,i} + 2\mathbf{k}_{2,i} + 2\mathbf{k}_{3,i} + \mathbf{k}_{4,i}) \ . \tag{3.23}$$

The covariance is only calculated at half of the points where the $\mathbf{A}$ matrix is calculated. Since i increases by two for each loop, there will be $\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4, \ldots, \mathbf{A}_N$, but $\mathbf{P}$ is only calculated at $\mathbf{P}_0, \mathbf{P}_2, \mathbf{P}_4, \ldots, \mathbf{P}_N$.

3.3.2 RK4 Adjoint

When calculating the adjoint, the RK4 algorithm begins with the derivative of the final σ and ρ constraints with respect to the final covariance matrix. For notation purposes, the σ and ρ constraints will be represented by the vector, J, where

$$J = [\sigma_1, \ldots, \sigma_n, \rho_{1,2}, \ldots, \rho_{n-1,n}]^{\mathrm{T}} \ . \tag{3.24}$$

For consistency with Equations (3.16) to (3.23), the partial of the constraints with respect to the covariance will be written as $\dfrac{\partial J}{\partial \mathbf{P}_{i+2}}$. If the RK4 integration runs for $i = 1 : N$, the RK4 adjoint loop runs from $i = N : 1$. The calculations are the same for every loop, so this section will just show the calculations performed to go from $i+2$ to i. The derivatives of the σ and ρ constraints are calculated for Equation (3.23). These derivatives are with respect to $\mathbf{k}_{1,i}, \mathbf{k}_{2,i}, \mathbf{k}_{3,i}, \mathbf{k}_{4,i}, h$, and $\mathbf{P}_i$. The derivative calculations for $\mathbf{k}_{1,i}, \mathbf{k}_{2,i}, \mathbf{k}_{3,i}$, and $\mathbf{k}_{4,i}$ are

$$\frac{\partial J}{\partial \mathbf{k}_{4,i}} = \frac{\partial J}{\partial \mathbf{P}_{i+2}} \cdot h_i/6 \tag{3.25}$$

$$\frac{\partial J}{\partial \mathbf{k}_{3,i}} = \frac{\partial J}{\partial \mathbf{P}_{i+2}} \cdot h_i/3 \tag{3.26}$$

$$\frac{\partial J}{\partial \mathbf{k}_{2,i}} = \frac{\partial J}{\partial \mathbf{P}_{i+2}} \cdot h_i/3 \tag{3.27}$$

$$\frac{\partial J}{\partial \mathbf{k}_{1,i}} = \frac{\partial J}{\partial \mathbf{P}_{i+2}} \cdot h_i/6 \tag{3.28}$$

The next derivative is with respect to h and is given by

$$\frac{\partial J}{\partial h_i} = \frac{\partial J}{\partial \mathbf{P}_{i+2}} \cdot 1/6((\mathbf{k}_{1,i} + 2\mathbf{k}_{2,i} + 2\mathbf{k}_{3,i} + \mathbf{k}_{4,i}). \tag{3.29}$$

Next, the derivative with respect to the covariance at the previous time step is calculated by

$$\frac{\partial J}{\partial \mathbf{P}_i} = \frac{\partial J}{\partial \mathbf{P}_{i+2}}. \tag{3.30}$$

It is important to note that every time a derivative is calculated, it is added to that same derivative if it has already been calculated from a different line of code, unless the variable has been redefined. For example, in Eq 3.30, $\frac{\partial J}{\partial \mathbf{P}_i}$ is calculated, and this is the first time that it is being calculated, so there is nothing to add to it, but later on, other equations which are dependent on $\mathbf{P}_i$ will have derivatives $\frac{\partial J}{\partial \mathbf{P}_i}$ and the two derivatives are added together.

Because $\mathbf{k}_{1,i}, \mathbf{k}_{2,i}, \mathbf{k}_{3,i}$, and $\mathbf{k}_{4,i}$ are all calculated with the Kalman filter, the derivatives are provided by the Kalman filter adjoint. For the derivatives from Equation (3.22), $\frac{\partial J}{\partial \mathbf{k}_{4,i}}$ is provided to the Kalman filter adjoint, and the outputs are the derivatives of the constraints with respect to the states, controls, time, and $\mathbf{P}_{\text{temp3,i}}$. The derivatives are then calculated for Equation (3.21). These derivatives are

$$\frac{\partial J}{\partial \mathbf{P}_i} = \frac{\partial J}{\partial \mathbf{P}_i} + \frac{\partial J}{\partial \mathbf{P}_{\text{temp3,i}}} \ , \tag{3.31}$$

$$\frac{\partial J}{\partial \mathbf{k}_{3,i}} = \frac{\partial J}{\partial \mathbf{k}_{3,i}} + \frac{\partial J}{\partial \mathbf{P}_{\text{temp3,i}}} \cdot h \ , \tag{3.32}$$

and

$$\frac{\partial J}{\partial h_i} = \frac{\partial J}{\partial h_i} + \frac{\partial J}{\partial \mathbf{P}_{\text{temp3,i}}} \cdot \mathbf{k}_{3,i} \ . \tag{3.33}$$

Now, $\frac{\partial J}{\partial \mathbf{k}_{3,i}}$ can be used by the Kalman filter adjoint to get the derivatives from Equation (3.20). Using $\frac{\partial J}{\partial \mathbf{P}_{\text{temp2,i}}}$ from the Kalman filter adjoint, the derivatives from Equation (3.19) can be calculated. This process is continued through Equation (3.16), and then the loop is repeated.

3.3.3 Kalman Filter Adjoint

Just like the RK4 algorithm called the Kalman filter algorithm four times for each time step, the RK4 adjoint calls the Kalman filter adjoint four times for each iteration. The inputs to the adjoint of the Kalman filter are the derivatives of the σ and ρ constraints with respect to the current covariance time rate of change, and the outputs are the derivatives of the σ and ρ constraints with respect to the previous states, controls,

time and the previous covariance matrix.

The input to the adjoint of the Kalman filter is the derivative of J with respect to $\mathbf{k}_{1,i}, \mathbf{k}_{2,i}, \mathbf{k}_{3,i}$, or $\mathbf{k}_{4,i}$. It does not matter which $\mathbf{k}$ is being used, because the equations are the same for all of them. So, for generality $\mathbf{k}_{1,i}, \mathbf{k}_{2,i}, \mathbf{k}_{3,i}$, and $\mathbf{k}_{4,i}$ will be replaced by $\dot{\mathbf{P}}_i$, and the input to the Kalman filter adjoint becomes $\frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger}$, where $\dot{\mathbf{P}}_i^\dagger$ is the vectorization of the $\dot{\mathbf{P}}_i$ matrix. When using the Kalman filter to calculate the time rate of change of the covariance matrix, there are a few steps and a few functions used. The functions needed are for the $\mathbf{A}$, $\mathbf{Q}$, $\mathbf{H}$, and $\mathbf{R}$ matrices. The $\mathbf{A}$ matrix is the linearized dynamics for the covariance states, the $\mathbf{Q}$ matrix is the process noise, the $\mathbf{H}$ matrix is the observation matrix, and the $\mathbf{R}$ matrix is the observation noise matrix. All of these matrices are functions of the states, controls, and time. After calculating these matrices, the time derivative of the covariance matrix, at i, is calculated by

$$\begin{aligned}\dot{\mathbf{P}}_i =&\mathbf{A}_i(\mathbf{x}_i, \mathbf{u}_i, t_i)\mathbf{P}_i(\dot{\mathbf{P}}_{i-1}) + (\mathbf{A}_i(\mathbf{x}_i, \mathbf{u}_i, t_i)\mathbf{P}_i(\dot{\mathbf{P}}_{i-1}))^{\mathrm{T}} + \mathbf{Q}_i(\mathbf{x}_i, \mathbf{u}_i, t_i) \\ &-\mathbf{K}_i(\mathbf{P}_i, \mathbf{H}_i, \mathbf{R}_i^{-1})\mathbf{R}_i(\mathbf{x}_i, \mathbf{u}_i, t_i)\mathbf{K}_i(\mathbf{P}_i, \mathbf{H}_i, \mathbf{R}_i^{-1})^{\mathrm{T}} \ ,\end{aligned} \tag{3.34}$$

where

$$\mathbf{K}_i = \mathbf{P}_i(\dot{\mathbf{P}}_{i-1})\mathbf{H}_i(\mathbf{x}_i, \mathbf{u}_i, t_i)^T\mathbf{R}_i(\mathbf{x}_i, \mathbf{u}_i, t_i)^{-1} \ . \tag{3.35}$$

Equation (3.34) is the starting point for the adjoint differentiation. $\frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger}$ has already been provided, so now the derivative of the σ and ρ constraints will be calculated with respect to $\mathbf{A}_i$, $\mathbf{P}_i$, $\mathbf{H}_i$, $\mathbf{R}_i$, and $\mathbf{Q}_i$. To calculate the derivative of the σ and ρ constraints with respect to $\mathbf{A}_i$, use

$$\frac{\partial J}{\partial \mathbf{A}_i^\dagger} = \frac{\partial J}{\partial \mathbf{A}_i^\dagger} + \frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger} \cdot \frac{\partial \dot{\mathbf{P}}_i^\dagger}{\partial \mathbf{A}_i^\dagger} \tag{3.36}$$

This calculation requires a $\frac{z(z+1)}{2} \times z^2$ matrix times a $z^2 \times z^2$ matrix. These derivative calculations can be reduced to multiplying smaller matrices. The first step

for this simplification is to define $\frac{\partial J}{\partial \dot{\mathbf{P}}_i}$ as

$$\frac{\partial J}{\partial \dot{\mathbf{P}}_i} = \begin{bmatrix} \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(1,1)} & \cdots & \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(1,z)} \\ \vdots & \ddots & \vdots \\ \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(z,1)} & \cdots & \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(z,z)} \\ \hdashline \frac{\partial J_{2,f}}{\partial \dot{\mathbf{P}}_i} \\ \vdots \\ \frac{\partial J_{z(z+1)/2,f}}{\partial \dot{\mathbf{P}}_i} \end{bmatrix} . \tag{3.37}$$

The difference between $\frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger}$ and $\frac{\partial J}{\partial \dot{\mathbf{P}}_i}$, is $\frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger}$ is the derivative of a vector with respect to the vectorized matrix $\dot{\mathbf{P}}_i$ and is a $\frac{z(z+1)}{2} \times z^2$ matrix, and $\frac{\partial J}{\partial \dot{\mathbf{P}}_i}$ is the derivative of a vector with respect to a matrix and is a $\frac{z^2(z+1)}{2} \times z$ matrix. Now, Eq. (3.36) can be rewritten as

$$\frac{\partial J}{\partial \mathbf{A}_i} = \frac{\partial J}{\partial \mathbf{A}_i} + 2\frac{\partial J}{\partial \dot{\mathbf{P}}_i} \cdot \mathbf{P}_i \ , \tag{3.38}$$

which results in a $\frac{z^2(z+1)}{2} \times z$ matrix times a $z \times z$ matrix. To calculate the derivative of the σ and ρ constraints with respect to $\mathbf{P}_i$, use

$$\frac{\partial J}{\partial \mathbf{P}_i^\dagger} = \frac{\partial J}{\partial \mathbf{P}_i^\dagger} + \frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger} \cdot \frac{\partial \dot{\mathbf{P}}_i^\dagger}{\partial \mathbf{P}_i^\dagger} \ . \tag{3.39}$$

For Eq. (3.39), the simplification requires $\frac{\partial J^{\mathrm{T}}}{\partial \dot{\mathbf{P}}_i^\dagger}$, and this takes the form

$$\frac{\partial J^T}{\partial \dot{\mathbf{P}}_i} = \left[\begin{array}{ccc:c:c:c} \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(1,1)} & \cdots & \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(1,z)} & & & \\ \vdots & \ddots & \vdots & \frac{\partial J_{2,f}}{\partial \dot{\mathbf{P}}_i} & \cdots & \frac{\partial J_{z(z+1)/2,f}}{\partial \dot{\mathbf{P}}_i} \\ \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(z,1)} & \cdots & \frac{\partial J_{1,f}}{\partial \dot{\mathbf{P}}_i(z,z)} & & & \end{array}\right] . \tag{3.40}$$

Using $\frac{\partial J^T}{\partial \dot{\mathbf{P}}_i}$, Eq. 3.39 can be written as

$$\frac{\partial J^T}{\partial \mathbf{P}_i} = 2\mathbf{A}^T \cdot \frac{\partial J^T}{\partial \dot{\mathbf{P}}_i} - 2\Big(\frac{\partial J}{\partial \dot{\mathbf{P}}_i} \cdot \mathbf{KR}\Big)^T + \frac{\partial J^T}{\partial \mathbf{P}_i} \ . \tag{3.41}$$

The next step is to take the derivative of J with respect to K_i,

$$\frac{\partial J}{\partial \mathbf{K}_i^\dagger} = \frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger} \cdot \frac{\partial \dot{\mathbf{P}}_i^\dagger}{\partial \mathbf{K}_i^\dagger} \ , \tag{3.42}$$

which can be written as

$$\frac{\partial J}{\partial \mathbf{K}_i} = -2\frac{\partial J}{\partial \dot{\mathbf{P}}_i}(\mathbf{KR}) \ , \tag{3.43}$$

and this notation takes advantage of multiplying smaller matrices. The derivative of J with respect to $\mathbf{H}_i$ is

$$\frac{\partial J}{\partial \mathbf{H}_i^\dagger} = \frac{\partial J}{\partial \mathbf{K}_i^\dagger} \cdot \frac{\partial \mathbf{K}_i^\dagger}{\partial \mathbf{H}_i^\dagger} \ . \tag{3.44}$$

In order to use the notation from Equations (3.37) and (3.40), which allow for smaller matrices to be multiplied together, a new variable must be defined,

$$\mathbf{B}_i = \mathbf{H}_i^T \mathbf{R}_i^{-1} \ . \tag{3.45}$$

So now Equation (3.44) becomes

$$\frac{\partial J}{\partial \mathbf{H}_i^\dagger} = \frac{\partial J}{\partial \mathbf{K}_i^\dagger} \cdot \frac{\partial \mathbf{K}_i^\dagger}{\partial \mathbf{B}_i^\dagger} \cdot \frac{\partial \mathbf{B}_i^\dagger}{\partial \mathbf{H}_i^\dagger} \ . \tag{3.46}$$

The derivative of J with respect to $\mathbf{B}_i$ becomes

$$\frac{\partial J^T}{\partial \mathbf{B}_i} = \mathbf{P}_i \cdot \frac{\partial J^T}{\partial \mathbf{K}_i} \ , \tag{3.47}$$

and the derivative of J with respect to $\mathbf{H}_i$ becomes

$$\frac{\partial J^T}{\partial \mathbf{H}_i} = \mathbf{R}_i^{-1} \cdot \frac{\partial J^T}{\partial \mathbf{B}_i^T} \ . \tag{3.48}$$

For the derivative of J with respect to $\mathbf{R}_i$, the calculation will be broken up into two parts so that Equations (3.37) and (3.40) can be used. The general equation is given by

$$\frac{\partial J}{\partial \mathbf{R}_i^\dagger} = \frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger} \cdot \frac{\partial \dot{\mathbf{P}}_i^\dagger}{\partial \mathbf{C}_i^\dagger} \cdot \frac{\partial \mathbf{C}_i^\dagger}{\partial \mathbf{R}_i^\dagger} \ , \tag{3.49}$$

where

$$\mathbf{C}_i = \mathbf{R}_i (\mathbf{R}^{-1})^T \mathbf{H}_i \mathbf{P}_i^T \ . \tag{3.50}$$

The first step is

$$\frac{\partial J^T}{\partial \mathbf{C}_i} = -\mathbf{K}_i^T \cdot \frac{\partial J^T}{\partial \dot{\mathbf{P}}_i} \ , \tag{3.51}$$

and the next step is

$$\frac{\partial J}{\partial \mathbf{R}_i} = \frac{\partial J}{\partial \mathbf{C}_i} \cdot \mathbf{K}' \ . \tag{3.52}$$

For this paper $\mathbf{R}_i^{-1}$ is treated as a unique function from $\mathbf{R}_i$, so the derivative of J with respect to $\mathbf{R}_i^{-1}$ is calculated separately from $\mathbf{R}_i$, and it is given in the general form of

$$\frac{\partial J}{\partial (\mathbf{R}_i^{-1})^\dagger} = \frac{\partial J}{\partial \dot{\mathbf{P}}_i^\dagger} \cdot \frac{\partial \dot{\mathbf{P}}_i^\dagger}{\partial \mathbf{B}_i^\dagger} \cdot \frac{\partial \mathbf{B}_i^\dagger}{\partial (\mathbf{R}_i^{-1})^\dagger} \ . \tag{3.53}$$

$\frac{\partial J^T}{\partial \mathbf{B}_i}$ is known from Eq. 3.47, and Eq. 3.53 can be written as

$$\frac{\partial J^T}{\partial (\mathbf{R}_i^{-1})} = \mathbf{H}_i \cdot \frac{\partial J^T}{\partial \mathbf{B}_i} \ . \tag{3.54}$$

The derivatives of $\mathbf{A}_i$, $\mathbf{Q}_i$, $\mathbf{R}_i$, $\mathbf{R}_i^{-1}$, and $\mathbf{H}_i$ with respect to the states, controls, and time are calculated using adigator [16]. The derivatives of J with respect to the states at i, are given by

$$\begin{aligned} \frac{\partial J}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_i} =& \frac{\partial J}{\partial \mathbf{A}_i^\dagger} \cdot \frac{\partial \mathbf{A}_i^\dagger}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_i} + \frac{\partial J}{\partial \mathbf{Q}_i^\dagger} \cdot \frac{\partial \mathbf{Q}_i^\dagger}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_i} + \frac{\partial J}{\partial \mathbf{H}_i^\dagger} \cdot \frac{\partial \mathbf{H}_i^\dagger}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_i} \\ &+ \frac{\partial J}{\partial \mathbf{R}_i^\dagger} \cdot \frac{\partial \mathbf{R}_i^\dagger}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_i} + \frac{\partial J}{\partial (\mathbf{R}_i^{-1})^\dagger} \cdot \frac{\partial (\mathbf{R}_i^{-1})^\dagger}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_i} + \frac{\partial J}{\partial (\mathbf{X}, \mathbf{U}, \mathbf{t})_{ii}} \ . \end{aligned} \tag{3.55}$$

Distribution Statement A – Approved for Public Release

3.4 Integrating Adjoint Code and RK4 into LGR Pseudospectral Optimization

The final step of this partial state collocation method is to integrate it into the LGR Pseudospectral structure. The LGR Pseudospectral method will treat the parameters from the initial and final covariance matrices as it would any other parameters, so it will calculate the derivative of the objective function with respect to the parameters. The difference comes in the constraint, which forces the final covariance to equal the initial covariance after propagation. These constraints are added to the constraints of the problem, and the derivatives of these constraints with respect to the states, controls, parameters, and time are added to the Jacobian. With each iteration of the NLP solver, the covariance is propagated using the states, controls, and time for that iteration, and then the derivatives are calculated. The constraints are adjoined to the other constraints of the problem, and the derivatives are added to the Jacobian. After this step is accomplished, the NLP solver can continue to run and find the optimal trajectory.

Distribution Statement A – Approved for Public Release

Chapter 4

Spacecraft Reentry Problem

4.1 Problem Setup

The two techniques for incorporating the covariance into the optimal control problem were both tested on a reusable, hypersonic spacecraft reentering Earth's atmosphere. This problem was taken from [10] and [11]. Although the purpose of this thesis is not to develop a trajectory for a spacecraft reentry vehicle, this problem is a good example for testing both methods of including the covariance in the optimal control problem. This problem will serve as an example to show that both methods of including the covariance converge to the same trajectory, and it will show how the methods differ in the length of time they take to run. This problem consists of an axisymmetric reentry vehicle entering Earth's atmosphere. This vehicle's states are represented using a modified 3 degree of freedom (DOF) model, and the vehicle is controlled by changing its orientation. This chapter will introduce all of the necessary equations and constants to model this vehicle and optimize the trajectory for two objective functions and compare the results.

4.1.1 Vehicle Model

The vehicle states are modeled using the three DOF which describe the translational motion of the vehicle. To fully model a vehicle, six DOF are needed, and they would

consist of 12 states. These 12 states consist of three states to describe the translational position, three states to describe the translational velocity, three states to describe the attitude, and three states to describe the angular rates. The position is expressed in spherical coordinates in the ECEF frame, $[r, \mu, \lambda]$. These states are the distance from the center of the earth, the longitude, and the latitude. The distance from the center of the earth is not a very intuitive value to use. A better way to represent the position would be the altitude above the equatorial radius of the earth, a. This value makes more sense for the problem, and the dynamics of it are the same as for r. The simple relationship

$$a = r - R_{\oplus} \ , \tag{4.1}$$

where

$$R_{\oplus} = 6,378,137 \text{ m} \tag{4.2}$$

can be used to relate the two values a and r. The velocity is expressed in spherical coordinates in the UEN frame, $[v, \gamma, \psi]$. These states are the velocity, the flight path angle, and the heading angle. The 3 DOF dynamics are given by

$$\dot{a} = v \sin\gamma \tag{4.3}$$

$$\dot{\mu} = \frac{v \cos\gamma \cos\psi}{r \cos\lambda} \tag{4.4}$$

$$\dot{\lambda} = \frac{v \cos\gamma \sin\psi}{r} \tag{4.5}$$

$$\dot{v} = -\frac{D}{m} - \frac{G_{\text{m}}}{r^2} \sin\gamma + \Omega_{\oplus}^2 r \cos\gamma(\sin\gamma \cos\lambda - \cos\gamma \sin\lambda \sin\psi) \tag{4.6}$$

$$\dot{\gamma} = \frac{L \cos\sigma}{mv} + \left(\frac{v}{r} + \frac{G_{\text{m}}}{r^2 v}\right) \cos\gamma + 2\Omega_{\oplus} \cos\lambda \cos\psi + \frac{\Omega_{\oplus}^2 r}{v} \cos\lambda(\cos\gamma + \sin\gamma \sin\lambda \sin\psi) \tag{4.7}$$

$$\dot{\psi} = \frac{L \sin\sigma}{mv \cos\gamma} - \frac{v}{r} \cos\gamma \cos\psi \tan\lambda + 2\Omega_{\oplus}(\tan\gamma \cos\lambda \sin\psi - \sin\lambda) - \frac{\Omega_{\oplus}^2 r}{v \cos\gamma} \sin\lambda \cos\lambda \cos\psi \ , \tag{4.8}$$

where D is the drag, L is the lift, m is the mass of the vehicle, G_m is the gravitational constant, and $\Omega_{\oplus}$ is the angular rate of the Earth. The mass of the vehicle is 226.8

Distribution Statement A – Approved for Public Release

kg, and the gravitational constant and Earth's angular rate are given by

$$G_{\mathrm{m}} = 3.986\,006 \times 10^{14}\ \mathrm{m}^3/\mathrm{s}^2 \tag{4.9}$$

and

$$\Omega_{\oplus} = 7.292\,116 \times 10^{-5}\ \mathrm{rad/s}\ . \tag{4.10}$$

The drag and lift acting on the vehicle and the direction of the velocity vector are functions of the orientation of the vehicle. The 3 DOF model does not include the orientation, and so it needs to be modified to include the angle-of-attack, α, and the bank angle, σ. The roll is not included because it has no effect on the trajectory. The angle-of-attack and bank angle control the lift and drag forces acting on the vehicle and the direction of the velocity vector, so they could be included in the problem as the controls. However, the controls are allowed to change instantaneously, and it would be unrealistic to model the system with an instantaneously changing orientation. Instead, the angle-of-attack and the bank angle are included as states, and their rates of change, $\dot{\alpha}$ and $\dot{\sigma}$, are included as the controls.

This modified 3 DOF model allows the system to be modeled and a trajectory determined. However, an additional three states will be considered. The final three states are constant parameters, so their dynamics are trivial, but they are included to account for uncertainty in the atmospheric density model and uncertainty in the normal and axial coefficients of the vehicle. These states have no bearing on the 3 DOF dynamics, but they do affect the covariance matrix and will be needed for including the covariance in the optimal control problem. C_ρ is the constant parameter included in the atmospheric density calculation. C_{C_N} is the constant used in the calculation of the normal coefficient, and C_{C_X} is the constant used in the axial coefficient calculation. The resulting system is governed by 11 states and two controls, which are given by

$$x = [r\ \mu\ \lambda\ v\ \gamma\ \psi\ \alpha\ \sigma\ C_\rho\ C_{\mathrm{C_N}}\ C_{\mathrm{C_X}}]^{\mathrm{T}} \tag{4.11}$$

Distribution Statement A – Approved for Public Release

and

$$u = [\dot{\alpha},\ \dot{\sigma}]^{\mathrm{T}} \ . \tag{4.12}$$

4.1.2 Lift and Drag

As the vehicle enters the atmosphere, aerodynamic forces and torques act on the vehicle. These aerodynamic forces make controllable flight possible [11]. Two of these forces used to control the trajectory of the vehicle are lift and drag. These forces are functions of the surface area, the aerodynamic coefficients of the vehicle, the dynamic pressure, and the angle-of-attack. Influencing the lift and drag experienced by the vehicle controls the dynamics of the vehicle.

The dynamic pressure, of which the lift and drag are functions, is a function of the atmospheric density and the velocity and is given by

$$q = \frac{1}{2}\rho v^2 \ . \tag{4.13}$$

The atmospheric density is dependent on the geopotential altitude. The geopotential altitude can be thought of as the "gravity-adjusted" altitude, where the dynamic pressure is constant along any given geopotential altitude. The geopotential altitude is a function of the geodetic altitude. The geodetic altitude takes into account the oblateness of the earth. Because the earth is spinning, it bulges at the equator and is not a perfect sphere. The altitude, a, included as a state in the problem is the altitude if the earth were a perfect sphere with radius, $R_{\oplus}$. Using a reference geoid which takes into account the bulge at the equator, the geodetic altitude is calculated as the distance from the vehicle to the earth along the line that passes through the vehicle and is normal to the surface of the geoid. The geodetic altitude, h, can be calculated from the radial distance, r, and the latitude and is given by the equation

$$h = r + R_{\oplus}\Big(\frac{1}{2}f(1-\cos(2\lambda)\Big) + f^2\Big(\frac{R_{\oplus}}{r} - \frac{1}{16}\Big)(1-\cos(4\lambda)) - 1) \ , \tag{4.14}$$

where f is the flattening coefficient given by

$$f = 0.003\,352\,811 \ . \tag{4.15}$$

There is a corresponding geodetic latitude, which relates to the geodetic altitude. The latitude included as a state is the geocentric latitude, but it, along with the radial distance, can be used to calculate the geodetic latitude as

$$\lambda_{\text{geodetic}} = \lambda + f\frac{R_\oplus}{r}\sin(2\lambda) + f^2\frac{R_\oplus}{r}\sin(4\lambda)\Big(\frac{R_\oplus}{r} - \frac{1}{4}\Big) \ . \tag{4.16}$$

The geodetic altitude can be used to calculate the geopotential altitude, h_{p}, by the relation

$$h_{\text{p}} = \frac{hR_0}{h + R_0} \ , \tag{4.17}$$

where R_0 is the radius of the Earth at 45 deg latitude,

$$R_0 = 6,356,766 \text{ m} \ . \tag{4.18}$$

The density is then modeled using a ninth order polynomial fit,

$$\rho = \exp(a_0 + a_1 h_{\text{p}} + a_2 h_{\text{p}}^2 + \cdots + a_9 h_{\text{p}}^9) \ , \tag{4.19}$$

which comes from the 1976 Standard Atmosphere model [17]. Table 5.4, gives the coefficients to be used in Equation (4.19).

In addition to the dynamic pressure, the lift and drag forces are a function of the exposed area, S, and the lift and drag coefficients, C_{L} and C_{D} respectively. Lift is calculated by

$$L = qSC_{\text{L}} \ , \tag{4.20}$$

and drag is calculated by

$$D = qSC_{\text{D}} \ . \tag{4.21}$$

For this problem the exposed surface area is a constant 0.829 m^2. The coefficients

Distribution Statement A – Approved for Public Release

Table 4.1: Density Fitting Coefficients

Parameter Order	Density ρ (lb/ft^3)
0	-2.60
1	-2.18×10^{-5}
2	-4.81×10^{-10}
3	5.61×10^{-15}
4	-6.16×10^{-20}
5	5.34×10^{-25}
6	-2.79×10^{-30}
7	8.10×10^{-36}
8	-1.22×10^{-41}
9	7.41×10^{-48}

of lift and drag are both functions of the axial and normal coefficients, C_X and C_N respectively, and the angle-of-attack. These coefficients are calculated by

$$C_L = C_{lN}\cos(\alpha) - C_X\sin(\alpha) \tag{4.22}$$

and

$$C_D = C_N\sin(\alpha) + C_X\cos(\alpha)\ . \tag{4.23}$$

The normal and axial coefficients are related to the aerodynamic performance of the vehicle. The vehicle in this problem is asymmetric, and the normal and axial coefficients can be modeled through the following equations, where C_{X_a}, C_{X_b}, C_{X_c}, C_{X_d}, C_{X_k}, C_{N_0}, and C_{n_a} are all constants specific to the vehicle, so the normal and axial coefficients become

$$C_X = C_{X_a}e^{-C_{X_b}(M-C_{X_c})} + C_{X_d} + C_{X_k}(\alpha)^2 \tag{4.24}$$

and

$$C_N = C_{N_0} + C_{N_a}\alpha\ . \tag{4.25}$$

The normal and axial fitting coefficients for the reference vehicle are

$$\begin{aligned} C_{X_a} &= 0.317 \\ C_{X_b} &= 0.550 \\ C_{X_c} &= 1.00 \\ C_{X_d} &= 0.083 \\ C_{X_k} &= 1.00 \\ C_{N_0} &= 0.02 \\ C_{X_a} &= 3.00 \end{aligned} \tag{4.26}$$

The mach number, M, is dependent on the speed of sound, c, and velocity of the vehicle. The speed of sound is actually dependent on the temperature, pressure, and density of the air, but for this example it is assumed to be the constant

$$c = 330 \text{ m/s} \ . \tag{4.27}$$

The Mach number is

$$M = \frac{v}{c} \ . \tag{4.28}$$

Now that the lift and drag acting on the vehicle can be calculated, the bounds on the problem need to be defined.

4.1.3 Problem Bounds

The initial values for the states has the vehicle beginning at an altitude of 45,720 m heading due East with a velocity 3,962 m/s. The initial state vector is

$$X_0 = \begin{bmatrix} 45{,}720 \text{ m} \\ 0 \text{ rad} \\ 0 \text{ rad} \\ 3{,}962 \text{ m/s} \\ 0 \text{ rad} \\ 0 \text{ rad} \\ 0 \text{ rad} \\ 0 \text{ rad} \\ 1 \\ 1 \\ 1 \end{bmatrix} . \tag{4.29}$$

Most of the bounds placed on the states are made so that the vehicle has a free range of motion. The bounds on the velocity and altitude are meant to be broad enough so that they should not have to be enforced and are not so strict as to determine the trajectory. The bounds on the angle of attack are set based on the capabilities of the vehicle. Some of the other bounds are implemented to avoid singularities that can occur within the problem dynamics. These singularities occur when

$$\begin{aligned} r &= 0, \\ \lambda &= \pm\frac{\pi}{2} \text{ rad}, \\ v &= 0, \\ \gamma &= \pm\frac{\pi}{2} \text{ rad}, \\ m &= 0 \ . \end{aligned} \tag{4.30}$$

In order to avoid these singularities, while still giving the vehicle a mostly free range of motion within the capability of the vehicle, the following bounds are placed on the states:

$$\begin{bmatrix} -21{,}384 \text{ m} \\ -2\pi \text{ rad} \\ -1.5533 \text{ rad} \\ 1.5240 \text{ m/s} \\ -1.5533 \text{ rad} \\ -2\pi \text{ rad} \\ 0 \text{ rad} \\ -2\pi \text{ rad} \\ 1 \\ 1 \\ 1 \end{bmatrix} \leq X \leq \begin{bmatrix} 3{,}048{,}000 \text{ m} \\ 2\pi \text{ rad} \\ 1.5533 \text{ rad} \\ 30{,}480 \text{ m/s} \\ 1.5533 \text{ rad} \\ 2\pi \text{ rad} \\ 0.261\,80 \text{ rad} \\ 2\pi \text{ rad} \\ 1 \\ 1 \\ 1 \end{bmatrix} . \tag{4.31}$$

The bounds on the controls are based on the performance capabilities of the craft and are

$$\begin{bmatrix} -0.174\,53 \text{ rad/s} \\ -0.523\,60 \text{ rad/s} \end{bmatrix} \leq u \leq \begin{bmatrix} 0.174\,53 \text{ rad/s} \\ 0.523\,60 \text{ rad/s} \end{bmatrix} . \tag{4.32}$$

The terminal contraints placed on four of the states are

$$v_f = 2{,}194.56 \text{ m/s} \tag{4.33}$$

$$\gamma_f = -0.7854 \text{ rad} \tag{4.34}$$

$$\mu_f = 0.1571 \text{ rad} \tag{4.35}$$

$$\alpha_f = 0 \text{ rad} \ . \tag{4.36}$$

There are no set terminal values for the altitude or latitude, but there are terminal conditions involving both of these states which must be satisfied. The first constraint is on the geodetic altitude, which is calculated from Equation (4.14).

$$h_f = 4,572 \text{ m} \ . \tag{4.37}$$

Distribution Statement A – Approved for Public Release

The second condition applies to the geodetic latitude

$$\lambda_{\text{geodetic}}(t_{\text{f}}) = 0.026\,18 \text{ rad} \ . \tag{4.38}$$

There is an additional path constraint on the geodetic altitude,

$$0 \leq h \leq 3{,}048{,}000 \text{ m} \ , \tag{4.39}$$

and this constraint ensures that the vehicle does not hit the ground. This constraint is not applied to the altitude used as a state because that altitude is in reference to the height above the equatorial radius of earth, whereas the geodetic altitude is the distance from Earth's surface. Because the Earth is oblate, the radius of the Earth is not the same for all altitudes, and the geodetic altitude takes that fact into account. It does not, however, take into account any other discrepancies along Earth's surface, such as mountain ranges. These inconsistencies in Earths surface are not modeled for this problem. There is another path constraint

$$10{,}000 \text{ N/m}^2 \leq q \leq 10{,}000{,}000 \text{ N/m}^2 \ , \tag{4.40}$$

which is placed on dynamic pressure calculated from Equation (4.13). The purpose of this path constraint is to ensure there is enough dynamic pressure for the vehicle to use the aerodynamic forces to control its motion.

4.1.4 Objective Function

The goal of this problem is to minimize the variance of the latitude and longitude multiplied by some weighting term in addition to minimizing the control input. The control input is in the rate of change of the angle-of-attack and the rate of change of the bank angle. The accumulated control effort can be calculated by integrating the angular rate commands over the time of the trajectory. The solution to this problem is meant to serve as an initial test on both methods. This objective function is not dependent on the covariance matrix, so the trajectory from the two methods tested

Distribution Statement A – Approved for Public Release

in this thesis can be compared to an optimal control solution that does not include the covariance. This cost function is given by

$$J = \int \left\{ \left(\frac{\dot{\alpha}}{\dot{\alpha}^U} \right)^2 + \left(\frac{\dot{\sigma}}{\dot{\sigma}^U} \right)^2 \right\} dt \ . \tag{4.41}$$

The second objective function used seeks to minimize the final variances of the latitude and longitude. Rather than only including the final variances of the latitude and longitude in the objective function, the control effort from the previous cost function is also included. This control effort is multiplied by a weighting term and is included to penalize the control effort in order to keep the trajectory smooth. The weighting term can be increased to penalize the control effort more, or it can be decreased to penalize the control effort less. This objective function is given by

$$J = \left(\sigma_\lambda^2 + \sigma_\mu^2 \right) + w \int \left\{ \left(\frac{\dot{\alpha}}{\dot{\alpha}^U} \right)^2 + \left(\frac{\dot{\sigma}}{\dot{\sigma}^U} \right)^2 \right\} dt \ . \tag{4.42}$$

4.1.5 Covariance Matrix

The dynamics of the covariance matrix take the form of Equation (2.11). There are no measurements in this problem, so the Kalman filter is not used. The initial

covariance matrix is given and the variances of the states are

$$\mathrm{diag}(P_0) = \begin{bmatrix} (0\text{ m})^2 \\ (0.000\,261\,8\text{ rad})^2 \\ (0.000\,261\,8\text{ rad})^2 \\ (5.081\text{ m/s})^2 \\ (0.005\,760\text{ rad})^2 \\ (0.005\,760\text{ rad})^2 \\ (0\text{ rad})^2 \\ (0\text{ rad})^2 \\ 1 \\ 1 \\ 1 \end{bmatrix} . \tag{4.43}$$

All of the correlation coefficients are zero initially, so all of the non-diagonal terms are zero. The diagonal elements of the process noise matrix are

$$\mathrm{diag}(Q) = \begin{bmatrix} (0\text{ m})^2/\text{s} \\ (0\text{ rad})^2/\text{s} \\ (0\text{ rad})^2/\text{s} \\ (0.2581\text{ m/s})^2/\text{s} \\ (1.3537 \times 10^{-8}\text{ rad})^2/\text{s} \\ (1.3537 \times 10^{-8}\text{ rad})^2/\text{s} \\ (0\text{ rad})^2/\text{s} \\ (0\text{ rad})^2/\text{s} \\ 0 \\ 0 \\ 0 \end{bmatrix} . \tag{4.44}$$

Distribution Statement A – Approved for Public Release

The A matrix for the covariance dynamics is the derivative of the state dynamics with respect to all of the states,

$$A = \frac{\partial f}{\partial x} \tag{4.45}$$

All of the information needed to test the two methods of including the covariance in the LGR Pseudospectral method is known, and the performance between the two methods can be compared.

4.2 Results

4.2.1 Minimum Effort Solution

The problem was first solved for the minimum effort objective function, given in Equation (4.41), using a standard LGR Pseudospectral optimization method without covariance. GPOPS was used to generate this solution [18]. This solution was used as a baseline to compare to the solutions generated by the two methods of including the covariance in the problem. The problem was then solved using the full state collocation method. The original problem consists of 11 states and two controls. The covariance matrix is 11×11, but because it is symmetric, only the unique elements of the matrix need to be included as states. The unique elements from this covariance matrix are the 11 standard deviations, one for each of the original states of the problem, and the 55 correlation coefficients which describe the correlations between the standard deviation of each state. The problem resulting from including the covariance as a state has a total of 77 states and 2 controls.

When including the covariance with the partial state collocation method, the number of states of the problem stays the same. However, the unique elements of the initial and final covariance matrices need to be stored as parameters. Because the covariance matrix has 66 unique elements, an additional 132 parameters need to be included in the problem for the initial and final covariance matrices. Although the covariance was not included in the objective function, this method still solved for the final covariance matrix.

Solving this minimum effort problem in three different ways was meant to provide an initial check on the procedures. First, it ensured that all three methods converged to the same trajectory, and secondly, it ensured that the two techniques outlined in this paper resulted in the same final covariance matrix for this minimum effort trajectory. This minimum effort solution was then used as the initial guess for developing the solutions which satisfied the objective function from Equation (4.42). All three methods converged to the same trajectory with only minor differences.

The success of the methods for this problem does not mean that the addition of covariance to the problem was successful, but it shows that even with the addition of the covariance through the two techniques, both methods can still converge on almost the same trajectory when the covariance is not included in the cost. Table 4.2 shows the normalized RMS for the states and the cost function for the full state collocation solution and the partial state collocation solution compared to the baseline solution for the the minimum effort problem. The minor differences shown in table 4.2 can be attributed to the additional constraints that must be satisfied by each method. The full state collocation method must ensure that the covariance satisfies the dynamics at all points, and the partial state collocation method must ensure the constraint that the final covariance equals the initial covariance after it has been propagated. Satisfying these constraints led to minor variations in the states.

4.2.2 Minimum Latitude and Longitude Error

The goal of including the covariance in the optimal control problem is to be able to minimize some of the uncertainties in the problem. More specifically, in this example the goal is to minimize the final latitude and longitude standard deviations. If the goal is to develop a trajectory which guides a vehicle to a specific location, it is important to ensure the error in the final position is as small as possible. The goal in this example is to develop a trajectory which minimizes the latitude and longitude standard deviations in order to show the usefulness of a covariance-shaped trajectory. There are two other goals to this example. One objective is to ensure that the two methods of including the covariance in the optimal control problem converge to the

Distribution Statement A – Approved for Public Release

Table 4.2: Normalized RMS errors for the full state collocation method and the partial state collocation method compared to the baseline solution for the minimum effort objective function

State	NRMS for full state collocation	NRMS for partial state collocation
Altitude	5.20×10^{-5}	4.18×10^{-5}
Longitude	5.07×10^{-6}	3.06×10^{-6}
Latitude	1.20×10^{-5}	1.19×10^{-5}
Velocity	3.79×10^{-5}	2.87×10^{-5}
Flight Path Angle	2.26×10^{-5}	2.15×10^{-5}
Heading Angle	4.76×10^{-5}	3.08×10^{-5}
Angle of Attack	1.07×10^{-4}	2.04×10^{-4}
Bank Angle	5.80×10^{-5}	2.30×10^{-5}
Cost Function	1.03×10^{-5}	1.51×10^{-7}
Max NRMS	1.07×10^{-4}	2.04×10^{-4}

same solution. The other objective is to see if the partial state collocation method converges in less time than the full state collocation method.

The results from using the objective function from Equation (4.41) show that both methods converge to the same trajectory when the covariance is added to the problem. There is no baseline solution for this example as there was for the previous problem, so the only test will be to see if the two solutions converge to the same solution. This convergence does not ensure global optimality, but it does show that the two techniques converge to the same solution. Table 4.3 shows the normalized RMS values for each state and the cost function. These differences are small, but there are differences because of how the covariance is calculated. The covariance is propagated with an RK4 integration scheme for the partial state collocation method, which is different from how the covariance is calculated in the LGR pseudospectral framework for the full state collocation method.

Not only did the two methods converge to the same trajectory, but the partial state collocation method converged in less time. Table 4.4 shows the time it took each method to solve the problem as well as the number of major iterations it took SNOPT to solve the problem. Table 4.4 shows that the partial state collocation method is more than eight times faster than the full state collocation method. The

Distribution Statement A – Approved for Public Release

Table 4.3: Normalized RMS errors for minimum latitude and longitude standard deviation

State	Normalized RMS
Altitude	5.07×10^{-4}
Longitude	3.76×10^{-5}
Latitude	1.39×10^{-4}
Velocity	2.67×10^{-4}
Flight Path Angle	1.84×10^{-4}
Heading Angle	2.65×10^{-4}
Angle of Attack	9.87×10^{-4}
Bank Angle	2.94×10^{-4}
Cost Function	1.51×10^{-6}
Max NRMS	9.87×10^{-4}

Table 4.4: Time and number of major iterations it took each method to solve the minimum latitude and longitude standard deviation problem

	Time (sec)	Major Iterations
First Method	7,865	2,253
Second Method	977	915

reason for this better performance is because including the additional states to the problem is more expensive than the adjoint differentiation required for the partial state collocation method. Also, as is evident from the number of major iterations it took to solve each problem, the partial state collocation method is a better posed problem for SNOPT to solve.

Figure 4-1 shows the trajectories for both the minimum effort solution and the minimum latitude and longitude standard deviation solution. This figure shows the maneuvers performed by the vehicle which drove the standard deviation of the latitude and longitude down. The differences between the solutions from the different methods are so small that they cannot be seen on a plot of the trajectories, so only one of the solutions is shown in figure 4-1. The plots of the remaining nontrivial states for the minimum effort solution and the minimum latitude and longitude standard deviations solution can be seen in the appendix.

Figures 4-2 and 4-3 show the standard deviations for the latitude and longitude from the minimum effort solution and the minimum latitude and longitude standard

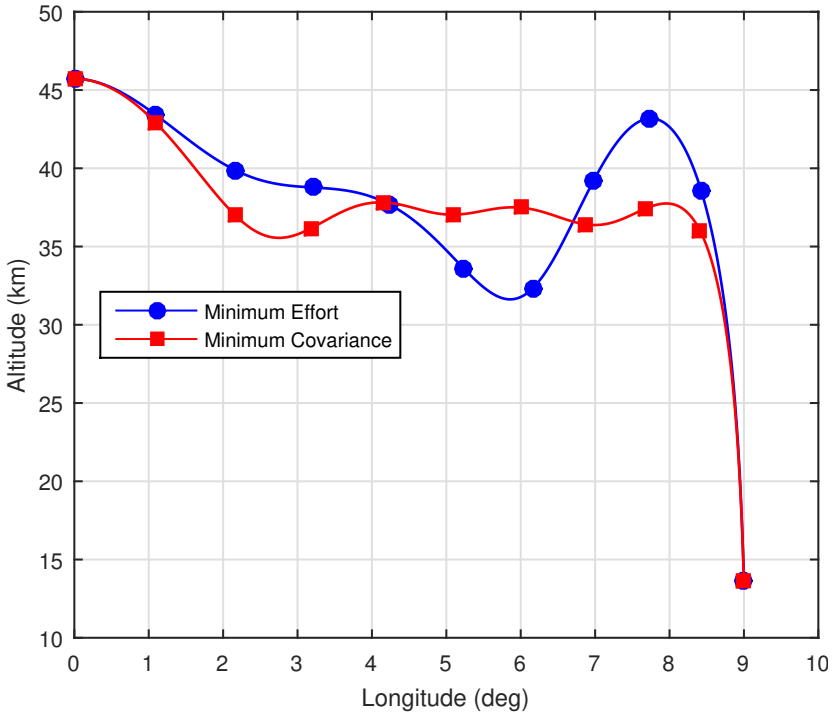

Figure 4-1: Trajectories of the minimum effort solution and the minimum latitude and longitude standard deviations

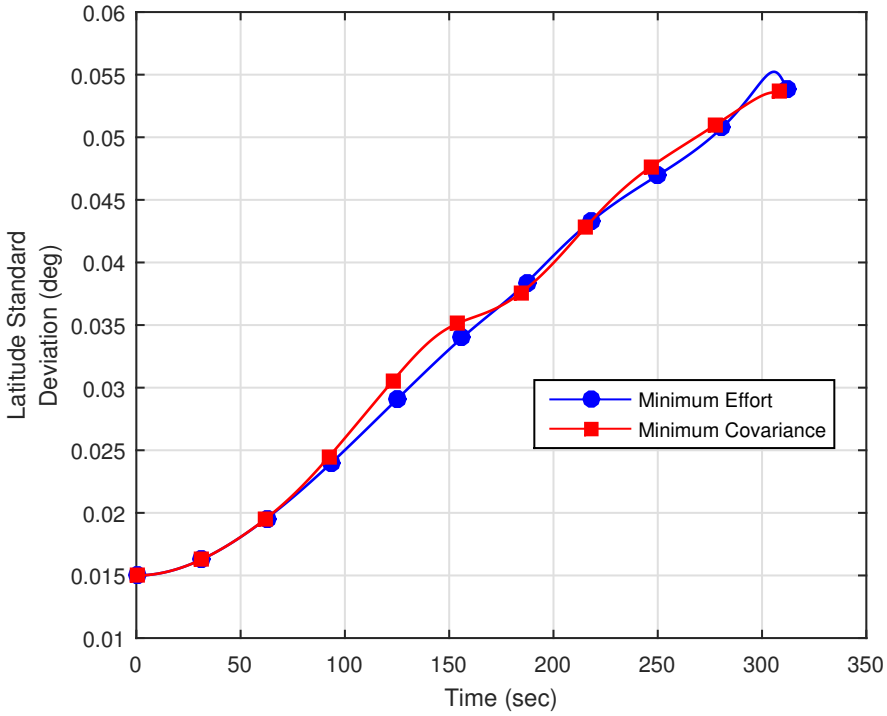

Figure 4-2: Latitude standard deviations of the minimum effort solution and the minimum latitude and longitude standard deviations

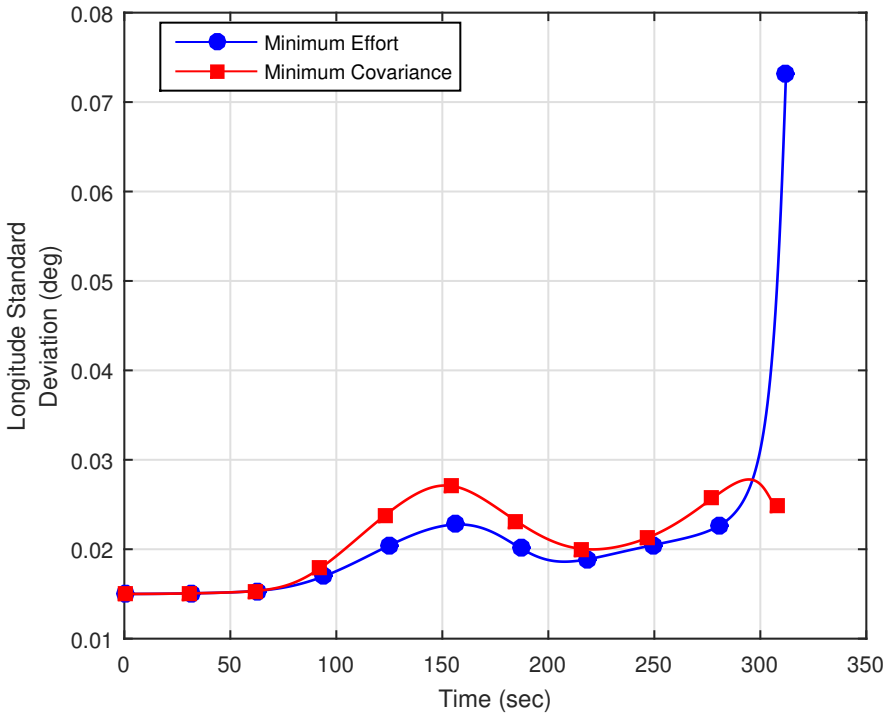

Figure 4-3: Longitude standard deviations of the minimum effort solution and the minimum latitude and longitude standard deviations

Table 4.5: Latitude and Longitude Standard Deviations for Minimum Efficiency and Minimum Latitude and Longitude Standard Deviation Solutions

	Fulll State Collocation		Partial State Collocation	
	Lat (deg)	Long (deg)	Lat (deg)	Long (deg)
Minimum Efficiencey	0.0539	0.0733	0.0539	0.0733
Minimum Lat and Long	0.0537	0.0246	0.0537	0.0246

deviations solution. Table 4.5 shows the final standard deviations for both examples. Table 4.5 and figures 4-2 and 4-3 show how much improvement came from the minimum latitude and longitude standard deviations solution. The optimizer was not able to bring down the latitude standard deviation by much, but it was able to significantly reduce the standard deviation of the longitude. This example shows how much influence the trajectory has on the uncertainty of the states. The second solution requires more control input, but it drastically reduces the uncertainty in the longitude.

These results do not allow us to draw any conclusions about the viability of these solutions to reduce the error in an actual flight. This thesis did not look into the usefulness or accuracy of covariance shaped trajectories, and it did not compare linearized covariance propagation to other methods of calculating the covariance matrix. No tests were performed to see if these covariance shaped trajectories actually drive down the error of the latitude and longitude, but that was not the purpose of this thesis. The purpose of this thesis was to test the viability of using the partial state collocation technique instead of the full state collocation method. Both techniques used the same dynamics for the vehicle states and the covariance, but the second technique did not require the covariance to be treated as additional states. The fact that both methods converged to the same solution, shows that the partial state collocation method is a viable alternative for solving these types of problems.

The previous results showed that the partial state collocation method converges to the same solution as the full state collocation solution. Not only is it a viable alternative, but it is also significantly improves the time it takes to solve the problem. For problems with relatively few states, the LGR Pseudospectral method can be used to solve the optimal control problem in a reasonable amount of time. When solving the minimum effort problem without including the covariance, such that there were only 11 states, GPOPS was able to solve the problem in several seconds. However, as the problem grows the time it takes to solve the problem increases rapidly. The average time it took to solve the minimum latitude and longitude standard deviation problem with the first technique was 7,865 seconds and with the second technique it was 977 seconds. Each technique was used to solve this problem five times and the average of the five runs was used to determine the time it took for each method. The partial state collocation technique is more than 8 times faster than the full state collocation method. The partial state collocation method is able to improve significantly the time it takes to solve this problem, but in saving this time, there is something lost from the solution. No path constraints can be placed on the elements of the covariance matrix in the partial state collocation technique. Also, in order to see the covariance matrix over time for the partial state collocation method, the

covariance matrix must be propagated using the RK4 integration. However, with the full state collocation method, the solution includes the covariance at each node, so no additional calculations need to be performed on the solution. There may be some limitations on the solution from the partial state collocation method, but it does converge on the same trajectory as the full state collocation method and it takes far less time to solve.

Distribution Statement A – Approved for Public Release

Chapter 5

Spacecraft Reentry with Inertial Measurement Unit

This chapter will look at the same spacecraft described in chapter 4, but this time an IMU is added to the problem. The problem is also tested with an altimeter in order to keep the error in the vertical channel from growing too large. The states of the vehicle remain the same, but the covariance matrix will be different. Rather than including the standard deviation for each vehicle state, the covariance matrix will look at the standard deviation for the position, velocity, and attitude in the NED frame as well as the error sources from the IMU. These problems are tested with and without an altimeter. The altimeter is used to update the measurements, which means the Kalman filter will be required for the covariance propagation whenever the altimeter is included. The IMU has different error sources, which contribute to the position, velocity, and attitude errors. Various combinations of these error sources are included in the problem, and the objective is to minimize the east and north position errors. This example serves three purposes. The first is to provide a test for including the Kalman filter in both methods. The second is to test how the time it takes to solve the problem changes as the covariance matrix grows in size; including different error sources from the IMU allows the size of the covariance matrix to be varied. The third is to compare the differences between having measurements and not having measurements.

Distribution Statement A – Approved for Public Release

The reason for testing the examples with and without measurements is to see how the time varies when measurements are included. When they are included, the covariance matrix becomes very dense, but without measurements, for this problem, the covariance matrix is very sparse. When the covariance is sparse, the elements that are identically zero do not need to be included in the full state collocation problem as additional states.

5.1 IMU Background

An IMU generally consists of three gyroscopes and three accelerometers. Additional instruments, such as magnetometers, can be included, but this example only uses gyroscopes and accelerometers. A gyroscope measures the angular rate about one axis, and an accelerometer measures the specific force along one axis. In order to have full knowledge of the motion of the craft, the three gyroscopes must be orthogonally located, and the accelerometers must also be orthogonally located. Additional gyroscopes and accelerometers are sometimes included in an IMU for redundancy.

The IMU is often combined with an Inertial Navigation System (INS), which takes the raw measurements from the IMU and calculates the attitude, velocity, and position. The attitude is calculated by integrating the angular rate and adding it to the initial orientation. The attitude is then used to rotate the specific forces measured by the accelerometers into the correct coordinate frame. The accelerometers measure the specific force in the body frame, but the velocity and position are usually calculated in another frame. After rotating the specific forces into the correct frame, those values are integrated and added to the initial velocity to calculate the current velocity. The velocity is then integrated and added to the initial position to obtain the current position. In order to obtain the attitude, velocity, and position, the measurements from the IMU need to be integrated, which means that IMUs are susceptible to accumulating errors. Any error in the IMU will grow over time as the error is integrated with the measurements.

The accumulating errors in IMUs mean that they are only useful for intervals

Distribution Statement A – Approved for Public Release

before the navigation system needs to be updated with measurements from another source, such as a GPS. The duration that an IMU can be used for navigation before it needs to be updated with an external measurement is dependent on the quality of the IMU. High-end IMUs are still susceptible to accumulating errors, but the errors are much smaller, and it takes longer for those errors to grow to the point where they significantly impact navigation. Low-end IMUs experience much larger errors, which grow quickly, and the navigation system needs to rely on updates much more frequently. However, no matter how good an IMU is, its errors will still accumulate, and eventually the navigation system will have to be updated with other measurements.

The different grades of IMUs differ in the magnitude of the errors but not the source of the errors. This thesis will examine four error sources for the gyroscopes and four error sources for the accelerometers. The first error source is angular random walk (ARW), which affects the gyroscope. This error occurs because of noise in the rate signal, and it gives the average deviation that occurs when the signal is integrated. This error will continue to increase with time until the attitude can be updated with some other measurement. The corresponding error for the accelerometer is the velocity random walk (VRW). Like the ARW, the VRW occurs because of noise in the signal, but this is noise in the accelerometer signal and not the gyroscope signal. The next error sources are the biases, which are constant errors in the outputs of the gyroscopes and accelerometers. The scale factor errors are proportional to the magnitude of the measurements from the gyroscopes and accelerometers. The final error sources considered in this thesis are the nonorthogonality errors, which come from the sensors not being perfectly orthogonal to each other.

5.2 General Navigation Equations

In this thesis the navigation errors will be calculated in the north-east-down (NED) coordinate frame [19, 20]. The attitude, velocity, and position do not need to be calculated in the NED frames, but their deviations are calculated in that frame.

Rather than using the attitude, velocity, and position in the NED frame as the states of the system, the states from the spacecraft reentry problem are used. These states are the position in the spherical ECEF frame, the velocity in the spherical body frame, and the angle of attack and bank angle to determine the attitude. These states can be used to calculate the deviations in the NED frame. The equations for the error sources come from [21]. The errors are given by

$$\sigma = [\psi,\ \delta v,\ \delta r,\ \delta a_{\mathrm{B}},\ \delta\omega_{\mathrm{B}},\ \delta a_{\mathrm{SF}},\ \delta\omega_{\mathrm{SF}},\ \delta a_{\mathrm{NO}},\ \delta\omega_{\mathrm{NO}}]^{\mathrm{T}}\ , \tag{5.1}$$

where δa_{B} is the accelerometer bias, $\delta\omega_{\mathrm{B}}$ is the gyroscope bias, δa_{SF} is the accelerometer scale factor, $\delta\omega_{\mathrm{SF}}$ is the gyroscope scale factor, δa_{NO} is the accelerometer nonorthogonality, and $\delta\omega_{\mathrm{NO}}$ is the gyroscope nonorthogonality. The dynamics of these errors are needed for propagating the covariance matrix. These dynamics are given by

$$\dot{\sigma} = A\sigma + w\ , \tag{5.2}$$

where w is the process noise, or in this case the ARW and VRW and is given by

$$w = [ARW,\ VRW,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0]^{\mathrm{T}}\ , \tag{5.3}$$

and A is

$$\begin{bmatrix} -[\omega^{\mathrm{n}}_{\mathrm{in}}\times] & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -\mathbf{T}^{\mathrm{n}}_{\mathrm{b}} & 0_{3\times3} & -\mathbf{T}^{\mathrm{n}}_{\mathrm{b}}\hat{\omega}_{\mathrm{G}} & 0_{3\times3} & -\mathbf{T}^{\mathrm{n}}_{\mathrm{b}}\tilde{\omega}_{\mathrm{G}} \\ \tilde{a}^{\mathrm{n}} & -[(\omega^{\mathrm{n}}_{\mathrm{ie}}+\omega^{\mathrm{n}}_{\mathrm{in}})\times] & GG^{\mathrm{n}} & \mathbf{T}^{\mathrm{n}}_{\mathrm{b}} & 0_{3\times3} & -\mathbf{T}^{\mathrm{n}}_{\mathrm{b}}\hat{f}_{\mathrm{s}} & 0_{3\times3} & \mathbf{T}^{\mathrm{n}}_{\mathrm{b}}\tilde{f}_{\mathrm{s}} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & -[\omega^{\mathrm{n}}_{\mathrm{en}}\times] & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{18\times3} & 0_{18\times3} & 0_{18\times3} & 0_{18\times3} & 0_{18\times3} & 0_{18\times3} & 0_{18\times3} & 0_{18\times3} & 0_{18\times3} \end{bmatrix} \tag{5.4}$$

In Equation (5.4), ω is the angular rate, where the superscript denotes the frame in which the rate is being expressed, and the subscript denotes between which frames the angular rate is being measured. The frames used here are the inertial frame, i, the NED frame, n, the ECEF frame, e, and the body frame, b. For example, $\omega^{\mathrm{n}}_{\mathrm{in}}$ is the angular rate of the NED frame relative to the inertial frame and expressed in the NED frame. The angular rates measured by the gyroscopes are ω_{G}. For this thesis

we will assume the gyroscopes are aligned with the body frame axes. The specific forces measured by the accelerometers are f_{s}. As with the gyroscopes, we will assume the accelerometers are aligned with the body frame axes. $\mathbf{T}_{\mathrm{b}}^{\mathrm{n}}$ is the rotation matrix from the body frame to the NED frame. The non-gravity acceleration acting upon the vehicle in the NED frame is a^{n}. GG^{n} is the gravity gradient term given by

$$GG^{\mathrm{n}} = \begin{bmatrix} -g/R_{\oplus} & 0 & 0 \\ 0 & -g/R_{\oplus} & 0 \\ 0 & 0 & 2g/R_{\oplus} \end{bmatrix} . \tag{5.5}$$

For notation purposes, $\hat{\omega}$ is the diagonal matrix of the ω vector, given by

$$\hat{\omega} = \begin{bmatrix} \omega_1 & 0 & 0 \\ 0 & \omega_2 & 0 \\ 0 & 0 & \omega_3 \end{bmatrix} . \tag{5.6}$$

The off-diagonal matrix, $\tilde{\omega}$ is

$$\hat{\omega} = \begin{bmatrix} 0 & \omega_3 & \omega_2 \\ \omega_3 & 0 & \omega_1 \\ \omega_2 & \omega_1 & 0 \end{bmatrix} . \tag{5.7}$$

The skew-symmetric cross-product matrix $[\omega\times]$ is given by

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} . \tag{5.8}$$

The states of the spacecraft reentry problem are not given in the NED frame, and so the values needed for Equation (5.4) need to be calculated from the states. The first term needed for the calculation is the angular rate of the ECEF frame with respect to the inertial frame and expressed in the NED frame. Because the Earth is rotating at a constant rate with respect to the inertial frame, the ω vector has a

constant magnitude. Although the magnitude of ω_{ie}^{n} is constant, the direction of the vector is dependent on the latitude of the vehicle. This angular rate can be expressed as

$$\omega_{ie}^{n} = [\Omega_{\oplus} \cos\lambda, 0, -\Omega_{\oplus} \sin\lambda]^{\mathbf{T}} \ . \tag{5.9}$$

The next rate that needs to be calculated is the angular rate of the NED frame with respect to the ECEF frame and expressed in the NED frame. The NED frame is carried with the vehicle but is not dependent on the orientation of the vehicle. The NED frame is centered at the center of the vehicle, with one axis pointed north, the second axis pointed east, and the third axis pointed down. When transforming from the ECEF frame to the NED frame, the first rotation is performed about the i axis by μ radians. This first rotation results in a temporary coordinate frame with axes $[i', j', k']$. The second step is a negative rotation about j' by λ radians. These two rotations can be expressed as

$$\mathbf{T}_{e}^{n} = \begin{bmatrix} -\cos\mu & 0 & -\sin\mu \\ 0 & -1 & 0 \\ \sin\mu & 0 & -\cos\mu \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\lambda & -\sin\lambda \\ 0 & \sin\lambda & \cos\lambda \end{bmatrix} \ . \tag{5.10}$$

Using these Euler angles and their derivatives and the transformation from Equation (5.10), the angular rates of the NED frame with respect to the ECEF expressed in the NED frame can be calculated by

$$\omega_{en}^{n} = \begin{bmatrix} \cos\lambda & 0 & 0 \\ 0 & -1 & 0 \\ -\sin\lambda & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\mu} \\ \dot{\lambda} \\ 0 \end{bmatrix} \ . \tag{5.11}$$

Now that both ω_{en}^{n} and ω_{ie}^{n} are known, ω_{ie}^{n} can easily be calculated by

$$\omega_{in}^{n} = \omega_{ie}^{n} + \omega_{en}^{n} \ . \tag{5.12}$$

The next terms needed for the error dynamics are the non-gravitational acceleration

acting on the body and the transformations from the body frame to the NED frame. Before these values can be computed, a brief overview of the coordinate frames used by the problem is needed.

5.2.1 Coordinate Frames

For the inertial coordinate frame, the Earth Centered Inertial (ECI) frame is used. The ECI frame is located at the center of the Earth. The z_{I} axis is aligned with the Earth's rotation axis, and it points in the direction of the geographic north pole. The x_{I} points from the center of the Earth to the sun on the Vernal Equinox. The y_{I} axis is oriented so as to complete the right-hand orthogonal coordinate frame.

The Earth Centered Earth Fixed (ECEF) frame is located at the center of the Earth like the ECI frame, but unlike the ECI frame, the ECEF frame rotates with the Earth. The z_{e} axis is oriented the same way as the z_{e} axis. The x_{e} axis points from the center of the Earth to the International Reference Meridian (IRM), and the y_{e} axis completes the right-handed orthogonal coordinate frame. The x_{e} and y_{e} axes lie in the same plane as the x_{I} and y_{I}, which is the same plane as the equator of the Earth. The ECEF and ECI frame are aligned once every 24 hours, after one complete rotation of the Earth.

The North East Down frame is a vehicle carried frame. Because the NED frame is carried with the vehicle, the origin is not at the center of the Earth as with the ECI and ECEF frames, but instead, it is located at the center of the vehicle. The translation from the ECEF frame to the NED frame is given by the radial distance, latitude, and longitude. The x_{n} axis is always pointed to the north pole. The y_{n} axis points east, and the z_{n} axis points down. The rotation from the ECEF to the NED frame is a function of the latitude and longitude and was given by Equation (5.10).

The next coordinate frame necessary for the spacecraft reentry problem is the velocity frame. The velocity of the vehicle is given in spherical coordinates in the NED frame. In the spacecraft reentry problem, v is the magnitude of the velocity, ψ is the heading angle, defined as the angle between the velocity vector and the plane, $y_{\mathrm{n}} - z_{\mathrm{n}}$, and γ is the flight path angle, which is defined a the angle between the

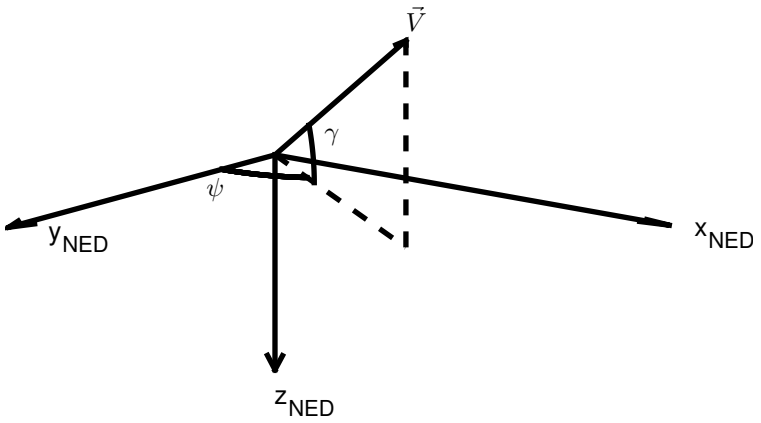

Figure 5-1: Velcoity vector in the NED frame

velocity vector and the plane, $x_{\mathrm{n}} - y_{\mathrm{n}}$. The velocity frame is defined so that x_{v} is aligned with the velocity vector and y_{v} is located in the local horizontal plane. The z_{v} axis completes the right-hand orthogonal coordinate frame. Figure 5-1 shows the velocity vector in the NED frame. The rotation from the the NED to the velocity frame is performed first by a positive rotation about the z_{n} axis by $\pi/2 - \psi$. The next step is a positive rotation about y_{V} by γ. The transformation matrix from the NED to the velocity frame is given by

$$\mathbf{T}_{\mathrm{n}}^{\mathrm{v}} = \begin{bmatrix} \cos(\pi/2-\psi) & \sin(\pi/2-\psi) & 0 \\ -\sin(\pi/2-\psi) & \cos(\pi/2-\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{bmatrix} \tag{5.13}$$

The next two reference frames are based on the orientation of the vehicle. The wind frame is a vehicle carried frame. The x_{w} axis is aligned with the velocity vector, so the x_{w} and x_{v} axes are aligned. The wind frame is obtained by rotating the velocity frame about the x_{v} axis negatively by the bank angle, σ. The transformation matrix for this rotation is

$$\mathbf{T}_{\mathrm{v}}^{\mathrm{w}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\sigma & -\sin\sigma \\ 0 & \sin\sigma & \cos\sigma \end{bmatrix} \tag{5.14}$$

The stability frame would involve rotating negatively about the z_{w} axis by the sideslip

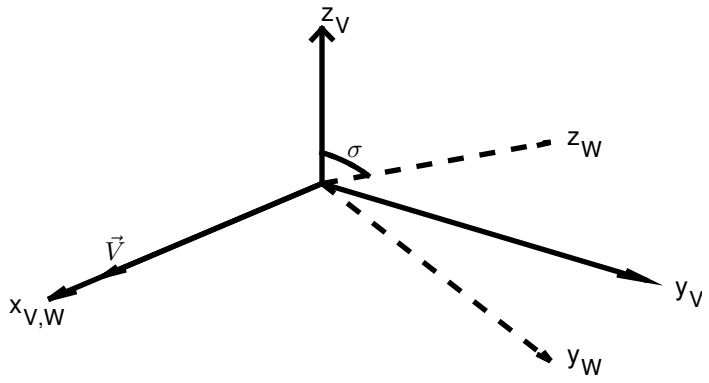

Figure 5-2: Rotation from Velocity frame to the Wind frame

angle. However, the sideslip angle is not modeled in this representation of the spacecraft reentry problem, so the sideslip angle will be assumed to be zero. Since the sideslip angle is zero throughout this problem, the stability frame is equivalent to the Wind frame.

The final reference frame used in this problem is the body fixed frame. This is also a body carried frame, so it shares an origin with the stability frame and the other vehicle carried frames. The body fixed frame is obtained by rotating about the y_s axis by the angle of attack, α. The transformation matrix for this rotation is given by

$$\mathbf{T}_{\mathrm{w}}^{\mathrm{b}} = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix} . \tag{5.15}$$

Figure 5-3 shows the rotation from the Wind frame to the body fixed frame. The body fixed frame is aligned with the axes of the vehicle. The x_b axis points through the nose of the vehicle, the y_b axis points through the right side of the vehicle, and the z_b axis points through the bottom of the vehicle.

With the coordinate frames defined, the final variables for the error dynamics can be calculated. The first variable is the non-gravitational acceleration. The non-gravitational forces acting on the vehicle are lift and drag, which are defined in the Stability frame. The lift, L, acts opposite to the z_s axis, and the drag, D, acts opposite the x_s axis. The drag and lift were calculated in chapter 4 with Equations (4.20) and

Distribution Statement A – Approved for Public Release

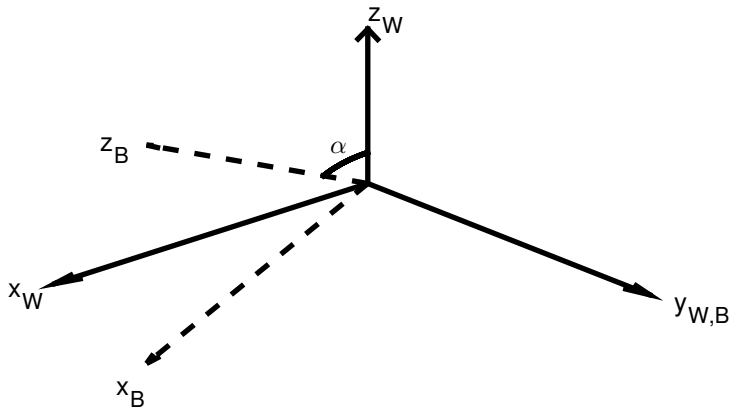

Figure 5-3: Rotation from Velocity frame to the Wind frame

(4.21). These forces are converted into acceleration by dividing the lift and the drag by the mass. The non-gravitational acceleration in the NED frame is calculated by

$$a^{\mathrm{n}} = \mathbf{T}_{\mathrm{v}}^{\mathrm{n}} \mathbf{T}_{\mathrm{w}}^{\mathrm{v}} \begin{bmatrix} D/m \\ L/m \\ 0 \end{bmatrix} . \tag{5.16}$$

The transformation matrix from the NED frame to the Velocity frame and the transformation matrix from the Velocity frame to the Wind frame have been defined in Equations (5.13) and (5.14). In order to obtain the transformation matrix from the Wind frame to the Velocity frame, the transpose is taken of the transformation matrix from the Velocity frame to the Wind frame. To perform the opposite rotation, it is always the transpose of the transformation matrix. The specific forces measured by the accelerometers are the accelerations from the lift and drag. Assuming the accelerometers are aligned with the body axes, the specific forces in the body frame, $f_{\mathrm{s}}^{\mathrm{b}}$, are given by

$$f_{\mathrm{s}}^{\mathrm{b}} = \mathbf{T}_{\mathrm{w}}^{\mathrm{b}} \begin{bmatrix} D/m \\ L/m \\ 0 \end{bmatrix} . \tag{5.17}$$

The final step is to determine the rates measured by the gyroscopes. Because the gyroscopes are aligned with the body frame axes, the gyroscopes measure angular

rates between the body fixed frame and the inertial frame and are expressed in the body fixed frame. This calculation will be performed in three parts. The first step is to calculate the angular rate of the NED frame with respect to the inertial frame expressed in the body frame, $\omega_{\text{in}}^{\text{b}}$. The second step is to calculate the angular rate of the velocity frame with respect to the NED frame expressed in the body fixed frame, $\omega_{\text{nv}}^{\text{b}}$. The final step is to calculate the angular rate of the body fixed frame with respect to the velocity frame expressed in the body fixed frame, $\omega_{\text{vb}}^{\text{b}}$. The angular rate of the NED frame with respect to the inertial frame has already been calculated in the NED frame by Equation (5.12). This angular rate is rotated to the body fixed frame by

$$\omega_{\text{in}}^{\text{b}} = \mathbf{T}_{\text{w}}^{\text{b}} \mathbf{T}_{\text{v}}^{\text{w}} \mathbf{T}_{\text{n}}^{\text{v}} \omega_{\text{in}}^{\text{n}} \ . \tag{5.18}$$

For the second step, we start by calculating the angular rate of the velocity frame with respect to the NED frame expressed in the velocity frame, $\omega_{\text{nv}}^{\text{v}}$. Using the transfomation matrix from the NED frame to the Velocity frame from Equation (5.13), the rate can be calculated by

$$\omega_{\text{nv}}^{\text{v}} = \begin{bmatrix} \sin\psi & 0 & 0 \\ \cos\psi & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\gamma} \\ \dot{\psi} \\ 0 \end{bmatrix} \ . \tag{5.19}$$

This rate now needs to be rotated into the body fixed frame, which can be done by

$$\omega_{\text{nv}}^{\text{b}} = \mathbf{T}_{\text{w}}^{\text{b}} \mathbf{T}_{\text{v}}^{\text{w}} \omega_{\text{nv}}^{\text{v}} \ . \tag{5.20}$$

Now $\omega_{\text{vb}}^{\text{b}}$ needs to be calculated. Using the transformation matrices to rotate from the velocity frame to the body frame, $\omega_{\text{vb}}^{\text{b}}$ can be calculated by

$$\omega_{\text{nv}}^{\text{v}} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & \cos\sigma & 0 \\ 0 & \sin\sigma & 0 \end{bmatrix} \begin{bmatrix} \dot{\sigma} \\ \dot{\alpha} \\ 0 \end{bmatrix} \ . \tag{5.21}$$

This rate is already in the correct coordinate frame, so it does not need to be transformed. The final step to obtain the rates measured by the gyroscopes is

$$\omega_{\mathrm{ib}}^{\mathrm{b}} = \omega_{\mathrm{in}}^{\mathrm{b}} + \omega_{\mathrm{nv}}^{\mathrm{b}} + \omega_{\mathrm{vb}}^{\mathrm{b}} \ . \tag{5.22}$$

5.2.2 Problem Values

This problem will use the initial and terminal conditions and path constraints for the states outlined in chapter 4. The difference in this problem is the covariance matrix being used. The A matrix used for calculating the covariance matrix has already been defined by Equation (5.4). In this chapter, six different covariance matrices and corresponding A matrices will be tested. The A matrix from Equation (5.4) is for a scenario that uses all of the error sources introduced in this chapter. The first test, case a, will just use the navigation equations without any errors. The deviations that are included in the covariance matrix are

$$x_1 = [\psi,\ \delta v,\ \delta r]^{\mathrm{T}} \ . \tag{5.23}$$

The A matrix for this problem will consist of the upper-left 9×9 elements of the A matrix from Equation (5.4).

The next test, case b, will take the problem from the first scenario and incorporate the errors from the gyroscope bias. The deviations included for this second scenario are

$$x_2 = [\psi,\ \delta v,\ \delta r,\ \delta\dot{\omega}_{\mathrm{B}}]^{\mathrm{T}} \ . \tag{5.24}$$

The A matrix for the second scenario will consist of the upper-left 15×15 elements of the A matrix from Equation (5.4).

The next test, case c, will take the problem from the second scenario and incorporate the errors from the accelerometer bias. The deviations included for this third scenario are

$$x_3 = [\psi,\ \delta v,\ \delta r,\ \delta\dot{\omega}_{\mathrm{B}},\ \delta\dot{a}_{\mathrm{B}}]^{\mathrm{T}} \ . \tag{5.25}$$

Distribution Statement A – Approved for Public Release

The A matrix for the third scenario will consist of the upper-left 18×18 elements of the A matrix from Equation (5.4).

The next test, case d, will take the problem from the third scenario and incorporate the errors from the gyroscope scale factor. The deviations included for this fourth scenario are

$$x_4 = [\psi,\ \delta v,\ \delta r,\ \delta\dot{\omega}_{\mathrm{B}},\ \delta\dot{a}_{\mathrm{B}},\ \delta\dot{\omega}_{\mathrm{SF}}]^{\mathrm{T}} \ . \tag{5.26}$$

The A matrix for the fourth scenario will consist of the upper-left 21×21 elements of the A matrix from Equation (5.4).

The next test, case e, will take the problem from the fourth scenario and incorporate the errors from the accelerometer scale factor. The deviations included for this fifth scenario are

$$x_5 = [\psi,\ \delta v,\ \delta r,\ \delta\dot{\omega}_{\mathrm{B}},\ \delta\dot{a}_{\mathrm{B}},\ \delta\dot{\omega}_{\mathrm{SF}},\ \delta\dot{a}_{\mathrm{SF}}]^{\mathrm{T}} \ . \tag{5.27}$$

The $\mathbf{A}$ matrix for the fifth scenario will consist of the upper-left 21×21 elements of the $\mathbf{A}$ matrix from Equation (5.4).

The next test, case f, includes all of the error sources and the full $\mathbf{A}$ matrix. All of these scenarios will be tested with and without an altimeter. When the altimeter is not included, the covariance is propagated without the Kalman updates. When the altimeter is included, it is included in the form of a Kalman filter. For the Kalman filter the observation matrix ,$\mathbf{H}$, is

$$H = [0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1]^{\mathrm{T}} \ . \tag{5.28}$$

This observation matrix is for scenario one. The $\mathbf{H}$ matrix is the length of the number of states in the covariance matrix, but the ninth element will always be one, and the rest of the elements are zero. This structure of the observation matrix means the Kalman filter will be updated with vertical range measurements. The $\mathbf{R}$ matrix for the Kalman filter is

$$R = (1000\mathrm{m})^2 \ . \tag{5.29}$$

The values for the IMU error sources come from the data sheet for the HG1900 [22]. The initial covariance matrix for the scenario without any IMU error sources is

$$P_0 = \begin{bmatrix} (2.618 \times 10^{-4} \text{ rad})^2 \\ (5.498 \times 10^{-3} \text{ rad})^2 \\ (5.760 \times 10^{-3} \text{ rad})^2 \\ (0 \text{ m/s})^2 \\ (0 \text{ m/s})^2 \\ (0 \text{ m/s})^2 \\ (0 \text{ m})^2 \\ (0 \text{ m})^2 \\ (0 \text{ m})^2 \end{bmatrix} . \tag{5.30}$$

5.3 Results With Altimeter

There were six different versions of the spacecraft reentry problem tested. Each of these methods was run with different tolerances. The first run had the tolerances set to 10^{-5}, and the second run had the tolerances set to 10^{-7}. The figures presented in this chapter are from the solution with the tolerances set at 10^{-7}.

The first scenario, case *a*, involved having an IMU and altimeter aboard the IMU, but none of the IMU error sources were included. This scenario tested how the variance of the position error in the north and east directions could be minimized only using the general navigation equations. This example had a covariance matrix of 9×9. The partial state collocation method had 11 states with 90 parameters, and the full state collocation method had 56 states.

Table 5.1 shows the normalized RMS errors between the partial state collocation method and the full state collocation method for the different states and the cost function for each of the cases run. The differences in the solutions are small, but it is worth discussing why they are not exactly the same. One reason is because of how the covariance matrix is scaled. Different scaling in the two methods allows for different amounts of error in the solution. The more notable reason for the differences is the

Distribution Statement A – Approved for Public Release

Table 5.1: Normalized RMS Errors

State	case a	case b	case c	case d
Altitude	1.64×10^{-5}	5.44×10^{-4}	3.83×10^{-5}	5.64×10^{-4}
Longitude	6.04×10^{-7}	9.44×10^{-6}	4.80×10^{-7}	8.53×10^{-6}
Latitude	8.82×10^{-6}	2.98×10^{-5}	1.38×10^{-5}	1.98×10^{-5}
Velocity	9.74×10^{-6}	1.48×10^{-4}	1.02×10^{-5}	1.54×10^{-4}
Flight Path Angle	7.31×10^{-6}	1.92×10^{-4}	1.50×10^{-5}	2.12×10^{-4}
Heading Angle	1.97×10^{-5}	1.21×10^{-4}	4.99×10^{-5}	1.23×10^{-4}
Angle of Attack	2.67×10^{-4}	8.48×10^{-4}	2.09×10^{-4}	1.01×10^{-3}
Bank Angle	1.01×10^{-4}	2.62×10^{-4}	4.49×10^{-5}	2.83×10^{-4}
Cost Function	1.40×10^{-6}	3.98×10^{-5}	1.13×10^{-6}	4.64×10^{-5}
Max NRMS	2.67×10^{-4}	8.48×10^{-4}	2.09×10^{-4}	1.01×10^{-3}

Table 5.2: Timing data for the different methods with different size covariance matrices and the tolerance set to 10^{-5} and 10^{-7}

	Partial State Collocation		Fulll State Collocation	
	tol = 10^{-5}	tol = 10^{-7}	tol = 10^{-5}	tol = 10^{-7}
Case a	55 sec	219 sec	66 sec	247 sec
Case b	220 sec	1,137 sec	925 sec	5,723 sec
Case c	210 sec	4,543 sec	2,605 sec	9,066 sec
Case d	773 sec	8,133 sec	3,604 sec	47,439 sec

difference between the RK4 integration and how the LGR pseudospectral method calculates the covariance. Because there are differences in the covariance matrix between these two methods, that leads to small differences in the optimal solution.

The partial state collocation method converged in less time than the full state collocation method. The other problems tested in this chapter will aim to find at what rate the time increases to solve the problem relative to the size of the covariance matrix. The timing data for each method for each of the cases is given in table 5.2, and the number of major iterations it took to solve each case with each method is given in table 5.3. These two tables show that for case a the partial state collocation method was faster and converged in fewer major iterations. Because it converged in fewer iterations, the partial state collocation method was a better posed problem for SNOPT to solve.

The next scenario, case b, included the error from the gyroscope bias. This ex-

Distribution Statement A – Approved for Public Release

Table 5.3: Number of iterations it takes to solve the different methods with different size covariance matrices and the tolerance set to 10^{-5} and 10^{-7}

	Partial State Collocation		Fulll State Collocation	
	tol = 10^{-5}	tol = 10^{-7}	tol = 10^{-5}	tol = 10^{-7}
Case *a*	137	319	111	238
Case *b*	229	869	378	614
Case *c*	125	827	375	1,030
Case *d*	156	581	226	996

ample shows how the trajectory *c*an be manipulated to minimize the error in the north and east positions coming from bias in the gyroscope. The covariance matrix for this problem is 12×12, so the partial state collocation method has 11 states and 156 parameters, and the full state collocation method has 89 states. The differences between the two methods for case *b* can be seen in table 5.1. Table 5.2 and table 5.3 show that the partial state collocation method was faster for case *b* and that it took fewer major iterations to solve, so once again the partial state collocation method was a better posed problem for SNOPT to solve.

The next example, case *c*, adds the accelerometer biases to the problem, so that the covariance matrix becomes 15×15. The partial state collocation method has 11 states with 240 parameters, while the full state collocation method has 131 states. The gyroscope biases are the driving forces for the north and east position error, so this trajectory is similar to the trajectory when just the gyroscope biases were included. This trajectory makes minor corrections for the accelerometer biases, but the trajectory is mainly governed by trying to minimize the error in the position coming from the gyroscope biases. Table 5.2 and table 5.3 show that the partial state collocation method was faster for case *c* and that it took fewer major iterations to solve, so once again the partial state collocation method was a better posed problem for SNOPT to solve.

This next example, case *d*, adds the gyroscope scale factors to the problem, so that the covariance matrix becomes 18×18. The partial state collocation method has 11 states with 342 parameters, while the full state collocation method has 182 states.

Table 5.2 and table 5.3 show that the trend continues of the partial state collocation method being faster and being better posed for SNOPT to solve.

This next example adds the accelerometer scale factors to the problem, so that the covariance matrix becomes 21×21. The partial state collocation method has 11 states with 462 parameters, while the full state collocation method has 242 states. It is at this point that the problem becomes too large for the full state collocation method. The partial state collocation method is able to converge to a solution, but the full state collocation method cannot. The final example was to add gyroscope and accelerometer nonorthogonality conditions. The resulting covariance matrix is 27×27. The partial state collocation method has 11 states with 756 parameters, while the full state collocation method has 489 states. The partial state collocation method was able to converge to a solution for this problem, but Matlab ran out of memory while running the full state collocation method. Plots for these last two examples are not included because no comparisons can be made between the two methods, since there are only results from the partial state collocation method. Even though no comparisons can be made between the solutions, it is noteworthy that the partial state collocation method is able to converge to a solution when the covariance matrix is this large, while the covariance as state method cannot handle covariance matrices this large. The partial state collocation method converges in 3,781 seconds when the accelerometer scale factors are included and the tolerance is set to 10^{-5} and it converges in 10,583 seconds when the nonorthogonality conditions are included and the tolerance is set to 10^{-5}.

5.3.1 Trajectory Comparison

Table 5.1 showed that both the partial state collocation method and the full state collocation method converged to the same solution for each of the cases where the full state collocation method was able to solve the problem. The minor differences in the solutions can be attributed to the difference in how the covariance was calculated between the two methods and the difference in how the problems were scaled. Figure 5-4 shows the trajectories of the minimum effort solution from chapter 4 and the

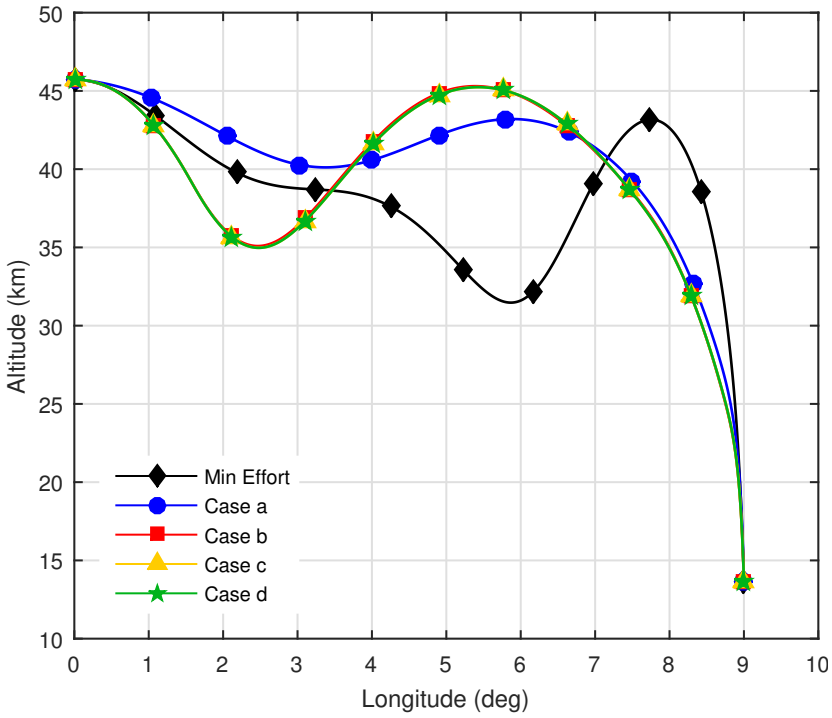

Figure 5-4: Trajectories for minimum effort solution and cases a-d

trajectories from cases a to b. This figure shows each modification the trajectory makes as additional error sources are added to the problem. The trajectory for case a maneuvers to minimize the north and east position standard deviations when no IMU error sources are considered. As the gyroscope bias is added to the problem, the vehicle performs another maneuver to try and cancel out the errors originating from the gyroscope bias. The final two trajectories are very similar to case b. The gyroscope bias is the driving error source from the IMU, so it has the greatest effect on the trajectory.

Table 5.4 shows the value of the cost function for each solution when it is applied to each case. For this table the covariance from each of the cases was propagated using the trajectory from each of the cases. From this table we would expect the trajectory from the solution to case a to have the smallest cost compared to all of the other trajectories when only the errors from case a are applied. This trend is exactly what table 5.4 shows. Of all the trajectories developed in this chapter, the optimal one for each case was the trajectory developed specifically for those error sources.

Distribution Statement A – Approved for Public Release

Table 5.4: Value of objective function for each trajectory applied to each case

	case a	case b	case c	case d
Minimum Effort	34.0107	117.2690	122.6437	122.6469
trajectory a	21.9976	44.9628	50.9392	50.9404
trajectory b	31.3027	33.5291	36.4353	36.4398
trajectory c	31.6503	33.5955	36.3675	36.372435
trajectory d	31.6486	33.5951	36.3676	36.372431

Table 5.5: Square root of the sum of the variances for the north and east position

	case a	case b	case c	case d
Minimum Effort	1,834.1 m	3,419.0 m	3,496.7 m	3,496.8 m
trajectory a	1,441.1 m	2,091.2 m	2,229.5 m	2,229.6 m
trajectory b	1,590.3 m	1,658.8 m	1,744.2 m	1,744.4 m
trajectory c	1,597.1 m	1,656.9 m	1,738.5 m	1,738.6 m
trajectory d	1,597.0 m	1,656.8 m	1,738.5 m	1,738.6 m

Table 5.5 shows the square root of the sum of the east and north position variances. One would expect to see the same trend as from table 5.4, where the smallest value for each case comes from the trajectory developed specifically for that case. However, in this table, the square root of the sum of the north and east position variances is smaller for trajectory c applied to case b than for trajectory b applied to case b. The reason for this is that the control input for trajectory b is lower than for trajectory c. The value of the cost function is still lower for trajectory b applied to case b, even though it is not truly the minimum position error.

Figures 5-5 and 5-6 show how the standard deviations change over time for each of the trajectories applied to case a. Obviously, the minimum effort solution has the largest standard deviations. None of the trajectories are able to make much improvement on the east position error, but trajectories a through d all make significant improvements to the north position error, with trajectory a having the lowest north position standard deviation. Figures 5-7 and 5-8 show how these uncertainties change as the gyroscope biases and scale factors and accelerometer biases are added to the problem. Because trajectories b through d are designed to cancel out some of these error sources, they all perform better than trajectory a or the minimum ef-

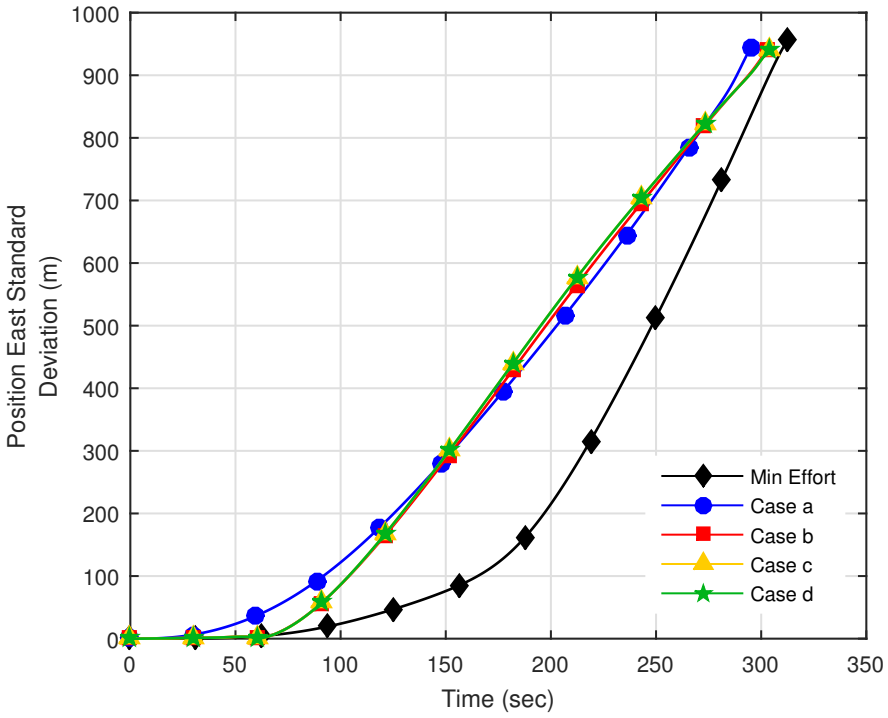

Figure 5-5: North position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *a*

fort trajectory. Trajectories *b* through *d* have very similar results because they are mostly influenced by the gyroscope bias. Trajectory *d* barely outperforms the other trajectories when applied to case *d*.

5.3.2 Timing

Figure 5-9 shows the timing data plotted on a logarithmic scale, and table 5.2 shows the timing data for the examples. From these data it can be calculated that when the tolerance is set to 10^{-5}, the partial state collocation method's time increases at $n^{3.41}$, where n is the number of states whose covariance is included in the problem. The time increases at $n^{5.83}$ for the full state collocation method when the tolerance is set at 10^{-5}. The time increases at $n^{5.36}$ for the partial state collocation method when the tolerance is set at 10^{-7}. The time increases at $n^{7.19}$ for the full state collocation method when the tolerance is set at 10^{-7}. The rates with which the time increases differ based on the tolerance used to solve the problem. These differences are partially

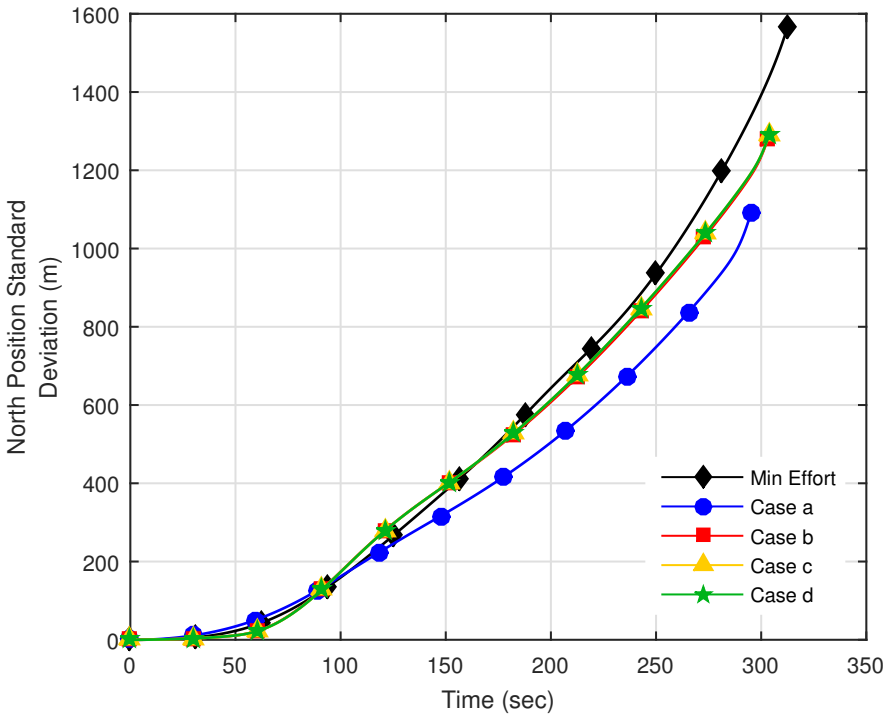

Figure 5-6: East position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *a*

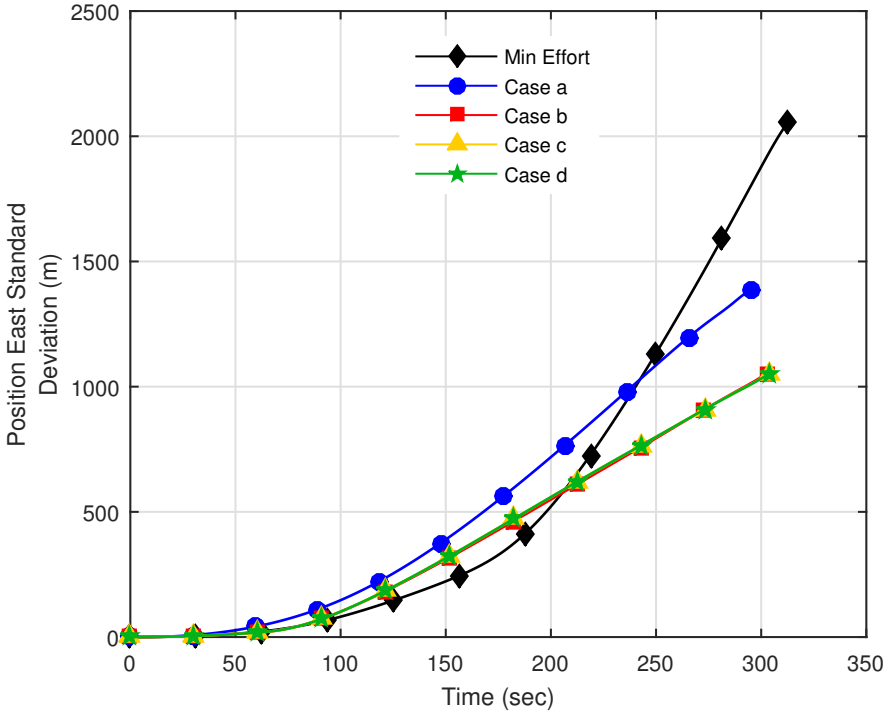

Figure 5-7: North position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *d*

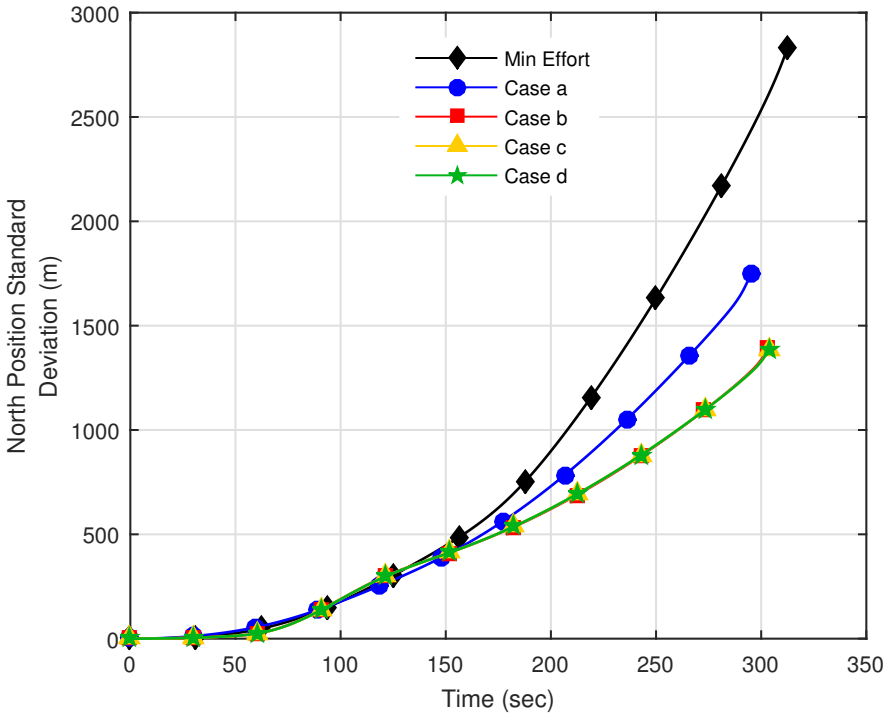

Figure 5-8: East position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *d*

due to the varied number of major iterations it takes to solve each problem. The rates at which the time increases for the tighter tolerance is probably more indicative of the truth because those are the more optimal solutions.

Table 5.3 shows the number of mesh refinements and the total number of major iterations it took to solve each problem. This information helps to explain some of the discrepancies in the timing data. For example going from case *b* to case *c*, we would expect a larger increase in the amount of time it takes to solve the problem for the full state collocation method with the tolerance set to 10^{-5} than we actually see in table 5.2. However, table 5.3 shows that for the covariance as state method with the tolerance set to 10^{-5}, it only took 226 major iterations for case *c*, and it took 375 iterations for case *b*. The number of iterations varies for each example, and this shows that the time it takes to solve the problem is not only a factor of which method is being used and how large the covariance matrix is, but it is also a factor of how well posed the problem is for the NLP solver.

This timing data shows that the time increases at a smaller rate for the partial

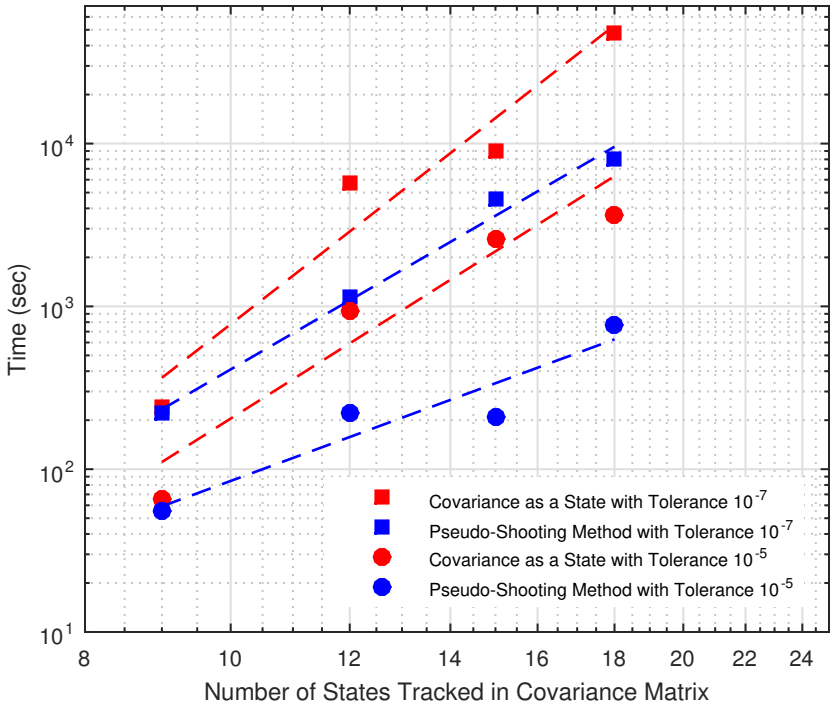

Figure 5-9: Logarithmic plot of timing data for the examples

state collocation method compared to the full state collocation method. Calculating the adjoint for the partial state collocation method is expensive, but in none of the cases tested here was it more expensive than adding the covariance as states. The difference in time between the two methods was not very large for the 9×9 covariance matrix, but that difference continues to increase as the covariance matrix grows in size. The larger the covariance matrix, the greater the benefit of using the partial state collocation method.

Both methods converged to almost the same solution when the tolerance was set to 10^{-7}. Table 5.1 shows the normalized RMS of the states and the cost functions for the different methods when the tolerance is set to 10^{-7}. The differences between these solutions can be attributed to the problem scaling and the difference in how the covariance is calculated in the LGR pseudospectral method and the RK4 integration used by the partial state collocation method. Not only does the time increase at a smaller rate for the partial state collocation method, but it also requires less memory.

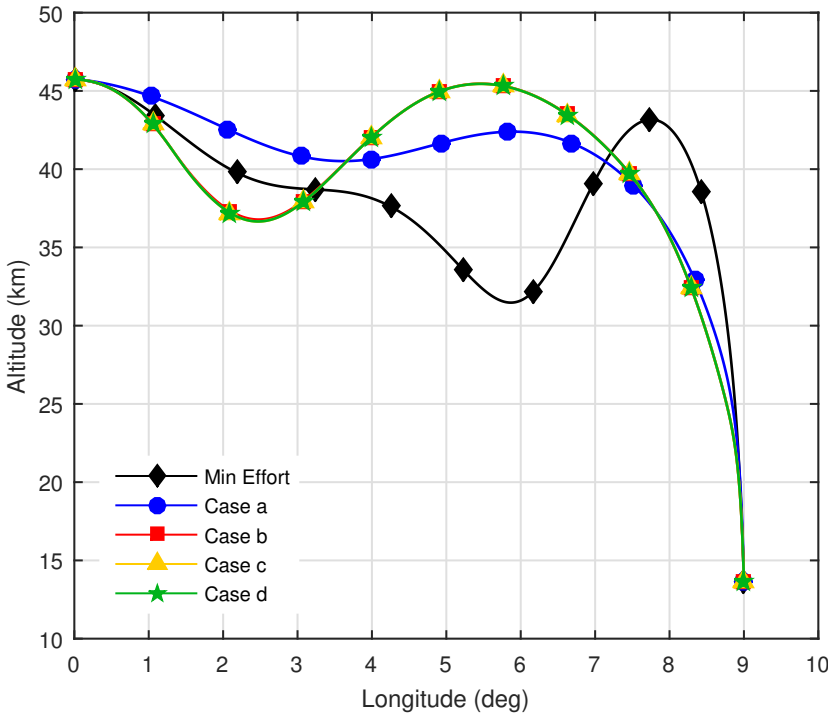

Figure 5-10: Altitude trajectory for minimum north and east position errors with no gyroscope or accelerometer error sources and no altimeter

By the time the covariance matrix was 27×27, Matlab ran out of memory for the full state collocation method, but the partial state collocation method was still able to converge on a solution.

5.4 Results Without Altimeter

The same example problems tested with the altimeter were also tested without the altimeter. In these examples no Kalman update is added to the covariance dynamics. Also, since there are no measurements, the covariance matrix has some identically zero elements in some of the examples. For these cases those elements do not need to be included as states in the full state collocation method.

Once again, the first scenario, case a, had a covariance matrix of 9×9. In this case the covariance matrix was full even without measurements, so there were still 56 states in the full state collocation method. The partial state collocation method had 11 states and 90 parameters.

The second scenario, case b, included the gyroscope biases. The covariance matrix was 12×12. Three of the unique elements of the covariance matrix were identically zero, so only 86 states were included in the full state collocation method. The partial state collocation method still included 11 states and 156 parameters. The third scenario, case c, included the gyroscope biases and accelerometer biases. The covariance matrix was 15×15. Twenty-four of the unique elements of the covariance matrix were identically zero, so only 107 states were included in the full state collocation method. The partial state collocation method still included 11 states and 240 parameters. The fourth scenario, case d, included the gyroscope biases and accelerometer biases. The covariance matrix was 18×18. Fourty-five of the unique elements of the covariance matrix were identically zero, so only 137 states were included in the full state collocation method. The partial state collocation method still included 11 states and 342 parameters. Figure 5-10 shows the trajectory for the minimum effort solution and the trajectories for cases a through d. The trajectories are very similar with and without the altimeter.

Only the results for the first four examples are shown without the altimeter, because those were the only results where the full state collocation method was able to converge to a solution. Table 5.6 shows the normalized RMS values between the different methods for each of the states and the final cost. This table shows that the errors without the measurements are comparable to the errors with the measurements. The reasons for these errors are the same as discussed when reviewing the results from these examples when measurements were included.

Table 5.7 shows the value of the cost function for each solution when it is applied to each case. For this table the covariance from each of the cases was propagated using the trajectory from each of the cases. As with table 5.4, we would expect the trajectory specifically developed for a case to have the smallest cost for that case. This trend is exactly what table 5.4 shows. Of all the trajectories developed in this chapter, the optimal one for each case was the trajectory developed specifically for those error sources. Table 5.5 shows the square root of the sum of the east and north position variances. Because the cost function includes the total control input, the

Table 5.6: Normalized RMS errors for examples without measurments

State	case a	case b	case c	case d
Altitude	7.95×10^{-5}	5.64×10^{-4}	8.69×10^{-4}	2.22×10^{-4}
Longitude	2.13×10^{-6}	8.53×10^{-6}	1.61×10^{-5}	4.96×10^{-6}
Latitude	1.52×10^{-5}	1.98×10^{-5}	4.67×10^{-5}	4.18×10^{-5}
Velocity	3.22×10^{-5}	1.54×10^{-4}	2.45×10^{-4}	6.05×10^{-5}
Flight Path Angle	3.84×10^{-5}	2.12×10^{-4}	3.05×10^{-4}	8.33×10^{-5}
Heading Angle	5.97×10^{-5}	1.23×10^{-4}	1.48×10^{-4}	6.58×10^{-5}
Angle of Attack	2.63×10^{-4}	1.40×10^{-3}	1.40×10^{-3}	5.80×10^{-4}
Bank Angle	3.14×10^{-4}	2.83×10^{-4}	2.59×10^{-4}	2.41×10^{-4}
Cost Function	1.63×10^{-7}	4.64×10^{-5}	3.31×10^{-5}	4.14×10^{-5}
Max NRMS	3.14×10^{-4}	1.40×10^{-3}	1.40×10^{-3}	5.80×10^{-4}

Table 5.7: Value of objective function for each trajectory applied to each case without an altimeter

	case a	case b	case c	case d
Minimum Effort	29.51587	76.1367	78.1725	78.1748
trajectory a	21.771	43.7478	45.5218	45.5248
trajectory b	27.3269	39.6945	41.0733	41.0759
trajectory c	27.5051	39.7027	41.0647	41.0672
trajectory d	27.5018	39.7024	41.0647	41.0672

Table 5.8: Square root of the sum of the variances for the north and east position for each trajectory applied to each case without an altimeter

	case a	case b	case c	case d
Minimum Effort	2,414.3 m	3,892.7 m	3,944.6 m	3,944.7 m
trajectory a	2,037.3 m	2,923.1 m	2,983.2 m	2,983.3 m
trajectory b	2,147.1 m	2,661.5 m	2,712.8 m	2,712.9 m
trajectory c	2,150.6 m	2,657.9 m	2,708.7 m	2,708.8 m
trajectory d	2,150.4 m	2,657.9 m	2,708.7 m	2,708.8 m

trend seen in table 5.7 is not seen in table 5.8. Trajectory b is the optimal trajectory for case b given the objective function, but trajectory c actually reduces the error more. If you were to weight the control input less, then the trend from table 5.7 would be seen in table 5.8 Figures 5-11 and 5-12 show how the standard deviations change over time for each of the trajectories applied to case a. The minimum effort

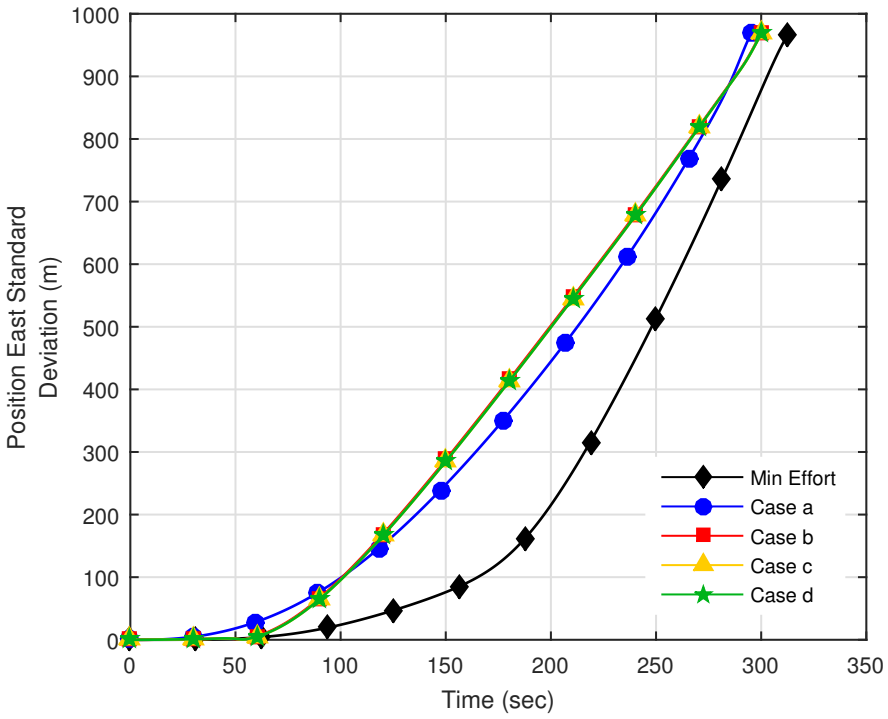

Figure 5-11: North position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *a* without an altimeter

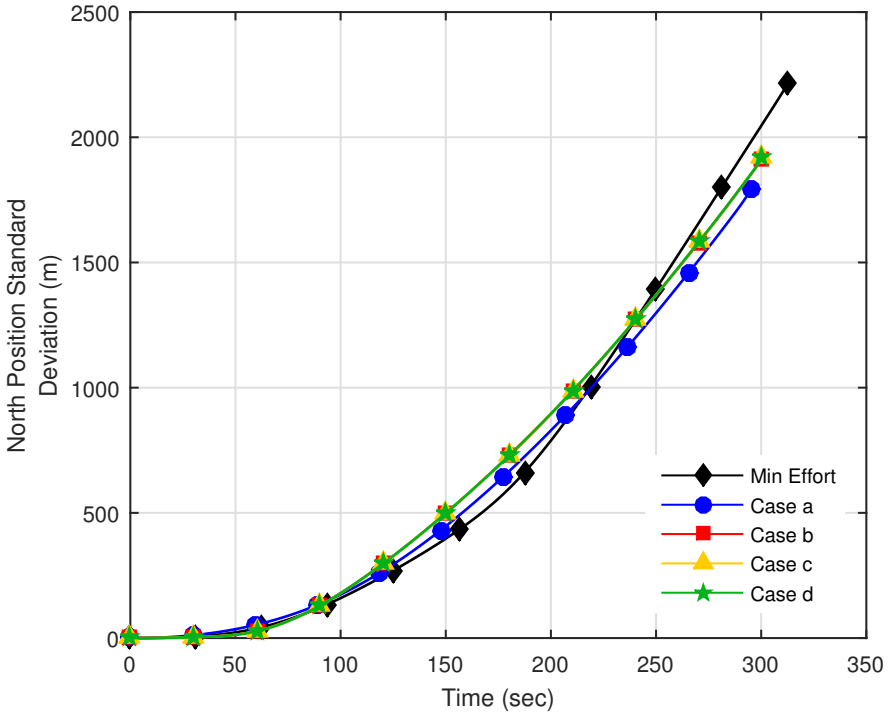

Figure 5-12: East position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *a* without an altimeter

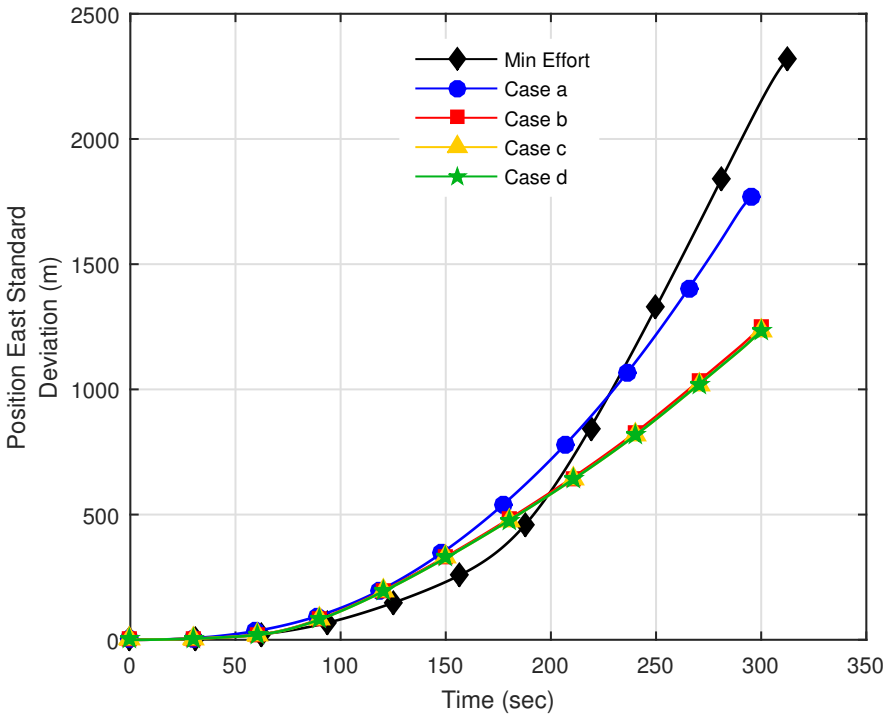

Figure 5-13: North position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *d* without an altimeter

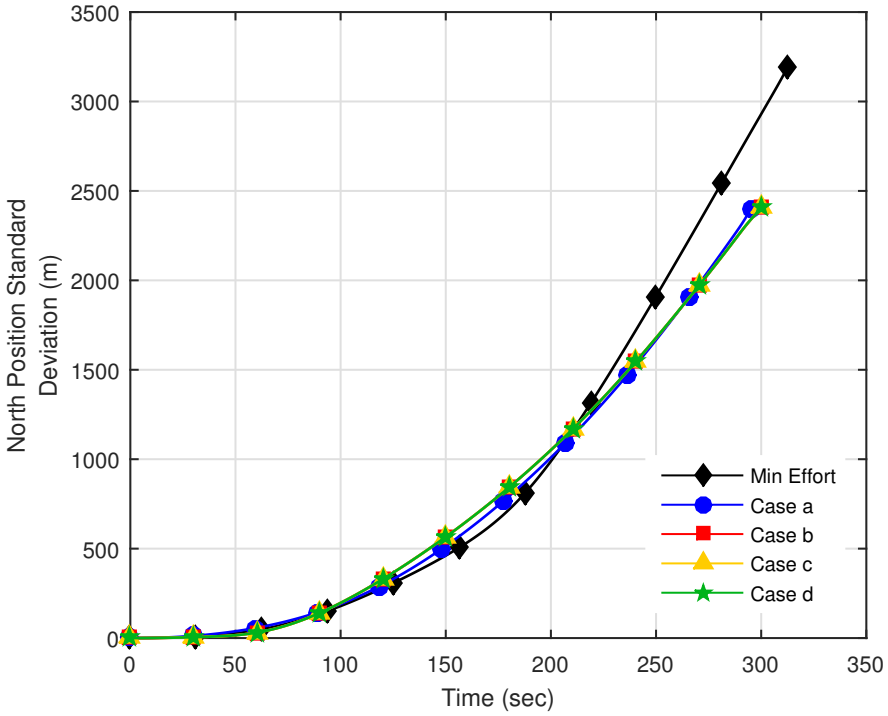

Figure 5-14: East position standard deviations for minimum effort trajectory and trajectories *a*-*d* applied to case *d* without an altimeter

Distribution Statement A – Approved for Public Release

Table 5.9: Timing data for the different methods with no measurements and with different size covariance matrices and the tolerance set to 10^{-5} and 10^{-7}

	Partial State Collocation		Fulll State Collocation	
	tol = 10^{-5}	tol = 10^{-7}	tol = 10^{-5}	tol = 10^{-7}
Case a	25 sec	144 sec	34 sec	127 sec
Case b	120 sec	516 sec	178 sec	2,386 sec
Case c	134 sec	1,684 sec	195 sec	2,322 sec
Case d	278 sec	4,993 sec	532 sec	5,659 sec

solution has the largest standard deviations. None of the trajectories are able to make much improvement on the east position error when no IMU error sources are included. Trajectories a through d all make significant improvements to the north position error, with trajectory a having the lowest north position standard deviation when no IMU error sources are included. Figures 5-13 and 5-14 show how these uncertainties change as the gyroscope bias and scale factors and accelerometer bias are added to the problem. Because trajectories b through d are designed to cancel out some of these error sources, they all perform better than trajectory a or the minimum effort trajectory. Trajectories b through d have very similar results because they are mostly influenced by the gyroscope bias. Trajectory d barely outperforms the other trajectories when applied to case d.

Table 5.9 shows the time it took each method to converge to a solution for tolerances of 10^{-5} and 10^{-7}. Figure 5-15 shows the time it took the different methods to converge to a solution on a logarithmic plot. This table and figure show that in every case, except when no IMU errors are included, the partial state collocation method is faster than the full state collocation method. It can be calculated from theseS data that the time to solve this problem with the partial state collocation method increases on the order of $n^{3.25}$ when the tolerance is set to 10^{-5}, and it increases on the order of $n^{5.09}$ when the tolerance is set to 10^{-7}. The full state collocation method increases on the order of $n^{3.66}$ when the tolerance is set to 10^{-5}, and it increases on the order of $n^{5.10}$ when the tolerance is set to 10^{-7}.

Although the trajectories did not significantly change when removing the altime-

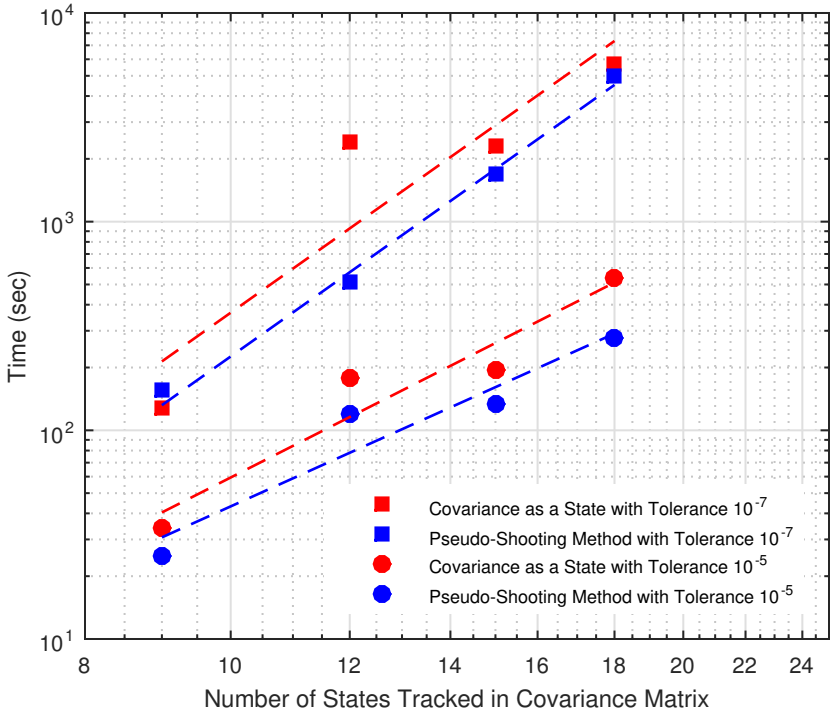

Figure 5-15: Logarithmic plot of timing data for the examples without measurements

Table 5.10: Number of iterations it takes to solve the different methods with no measurements and with different size covariance matrices and the tolerance set to 10^{-5} and 10^{-7}

	Partial State Collocation		Full State Collocation	
	tol $= 10^{-5}$	tol $= 10^{-7}$	tol $= 10^{-5}$	tol $= 10^{-7}$
Case a	80	257	114	306
Case b	184	286	324	1,556
Case c	146	468	245	1,162
Case d	151	469	226	1,132

ter, the time it took to run the scenarios did. Each case took less time to solve when measurements were not included. This result is not surprising because without the Kalman filter, the covariance dynamics become much less complex. What is more significant than the overall improvement in run time, however, is the rate at which the time increases for the full state collocation method. The partial state collocation method increases at a slighltly smaller rate than when measurements were included, but most of the improvement is in the rate the time increases for the covariance as

Distribution Statement A – Approved for Public Release

state method. The reason the full state collocation method increases at such a smaller rate when measurements are not included is because the covariance matrix is more sparse without measurements. When the covariance matrix is sparse, the identically zero elements do not need to be included as states. With fewer states, the problem takes less time to solve. For example, in case d, 24 of the states were removed from the problem when the altimeter was not included.

Table 5.10 shows the number of iterations it took to solve the problems. This table shows that it takes fewer iterations when using the partial state collocation method. These data show that the full state collocation method is a better posed problem for SNOPT to solve. The fact that it takes fewer iterations to solve the partial state collocation method is one of the reasons it converges in less time than the full state collocation method in most of the cases.

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution Statement A – Approved for Public Release

Chapter 6

Conclusion

In this thesis two methods of including covariance in an LGR pseudospectral optimization method were tested. Both of these methods allowed the elements from the final covariance to be included in the cost function so that the trajectory could be optimized with respect to the covariance. The first method was the most intuitive. It took the unique elements of the covariance matrix and included them as additional states to the problem. The problem with this method is that it causes the problem size to increase rapidly, and the time it takes to solve the problem also increases rapidly. In an attempt to solve this timing problem, caused by the problem size becoming so large, the unique elements of the initial and final covariance matrices were included as parameters, and a constraint was added to the problem that the initial covariance equals the final covariance after it has been propagated with an RK4 integration scheme. This method also requires that the derivatives of these additional constraints with respect to the states, controls, time, and parameters be included in the LGR pseudospectral framework. Both of these methods were tested with and without measurements and for different sized covariance matrices.

6.1 Comparison

The partial state collocation method had the additional time expense of having to calculate the adjoint of the constraints imposed on the unique elements of the covari-

Distribution Statement A – Approved for Public Release

ance matrix. Because the RK4 integration and its adjoint require looping over the number of steps in the propagation, the time it takes for this calculation can be large. The amount of time required for these calculations is dependent on the step size and time the problem spans. Including the covariance as a state does not require the added expense of calculating the adjoint of the covariance constraints, but as more states are included in the problem, it takes much longer to solve it.

The first example with the spacecraft reentry problem showed how the two methods compared when measurements were not used. The partial state collocation method was much faster than the full state collocation method. Both solutions converged to almost the same solution. The difference in solutions can be contributed to the differences in calculating the covariance between the RK4 integration and the LGR pseudospectral method. The way the covariance was scaled also attributed to some of the error. The fidelity of either solution can be adjusted by changing the tolerances used by GPOPS. The fidelity of the partial state collocation method can also be adjusted by changing the step size for the RK4 integration. The fidelity is decreased by making the step size larger, and the fidelity of the solution can be increased by making the step size smaller. The step size affects how long it takes for the partial state collocation method to solve the problem. As the step size decreases, the time increases. The user must make a compromise between speed and accuracy when choosing a step size for the problem.

The second set of examples included an IMU. They also tested the examples with and without an altimeter. The altimeter, when included, provided continuous altitude measurements. The covariance was calculated using general navigation equations and various error sources from the IMU. This problem allowed the two methods to be tested with an extended Kalman filter and it allowed the size of the covariance matrix to be varied. By varying the size of the covariance matrix, the rate at which the time increased to solve the problem with relation to the size of the covariance matrix was able to be calculated. From these examples, it was determined that the partial state collocation method increases on the order of about $n^{5.36}$ for the tighter tolerance and when measurements are included. The full state collocation method increases on the

Distribution Statement A – Approved for Public Release

order of $n^{7.19}$ for the tighter tolerance and when the measurements are added, where n is the number of states in the covariance matrix.

When measurements are not included, the time to solve the problem using the partial state collocation method increases at $n^{5.09}$, while the time increases at $n^{5.10}$ for the full state collocation method. The reason the full state collocation method is better without measurements than it was with measurements is because the calculations without measurements are less complicated, and the covariance matrix is more sparse, so the identically zero elements do not need to be included as states. The partial state collocation method still outperformed the full state collocation method when measurements were and were not included, but the benefit is less significant as the covariance matrix becomes more sparse.

The partial state collocation method does have some drawbacks. The adjoint of the RK4 and Kalman filter is expensive. Just how expensive is dependent on the step size and problem duration. This method also does not allow path constraints to be placed on the covariance matrix. The fidelity of the solution is dependent both on the tolerances of the NLP solver and the step size used for the RK4 integration. This method gives the user more control over picking between accuracy and speed by allowing the step size to be changed. For the full state collocation method, the covariance is treated as any other state, so path constraints can be enforced and the accuracy of the solution is just a function of the tolerances of the NLP solver. However, the time to run the full state collocation problem increases much faster than the partial state collocation method.

6.2 Future Work

This thesis only outlined how the partial state collocation method could be applied to an LGR pseudospectral optimization method. However, the principal of the method should work with any pseudospectral optimizer. The actual method of incorporating the covariance would differ slightly, but the general idea of imposing a constraint that the final covariance equals the initial covariance after it has been propagated remains

Distribution Statement A – Approved for Public Release

the same. Other work which could be done is to apply a partial state collocation method to something other than the covariance. The covariance was chosen for this thesis because it is stable and easy to propagate. However, this method can be applied to other states. The states are turned into twice the number of parameters. Half the parameters are the initial value, and the other half are the final value. A constraint is added that the final state equals the initial state after it has been propagated. If the state is stable, so that it can be propagated with RK4 techniques, then this method would work. For this method to be beneficial, it would need to be a large problem, where many of the states are converted to parameters.

Distribution Statement A – Approved for Public Release

Appendix A

Plots of States from Spacecraft Reentry Problem

This appendix shows the states for the minimum effort and the minimum latitude and longitude standard deviations trajectories. Only the altitude and longitude were presented in chapter 4. These figures show the maneuvers performed by the vehicle to drive down the longitude standard deviation.

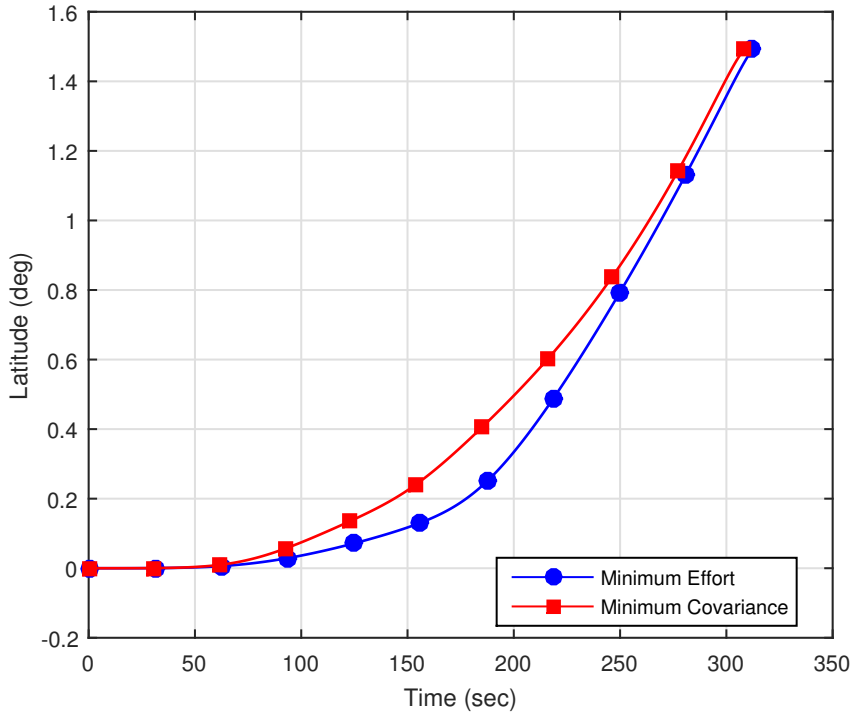

Figure A-1: Latitude for minimum effort trajectory and minimum latitude and longitude standard deviations trajectory

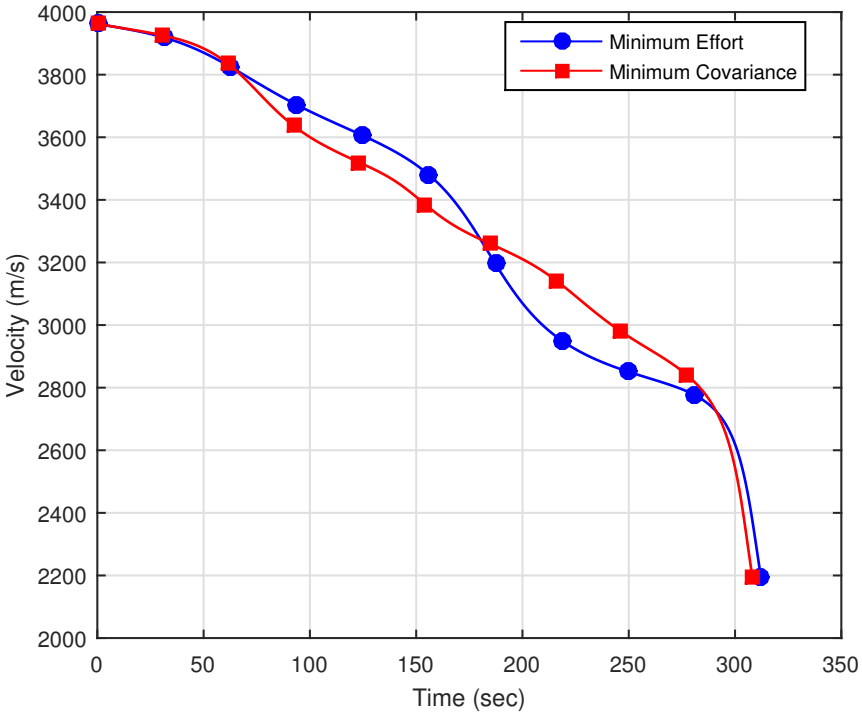

Figure A-2: Velocity for minimum effort trajectory and minimum latitude and longitude standard deviations trajectory

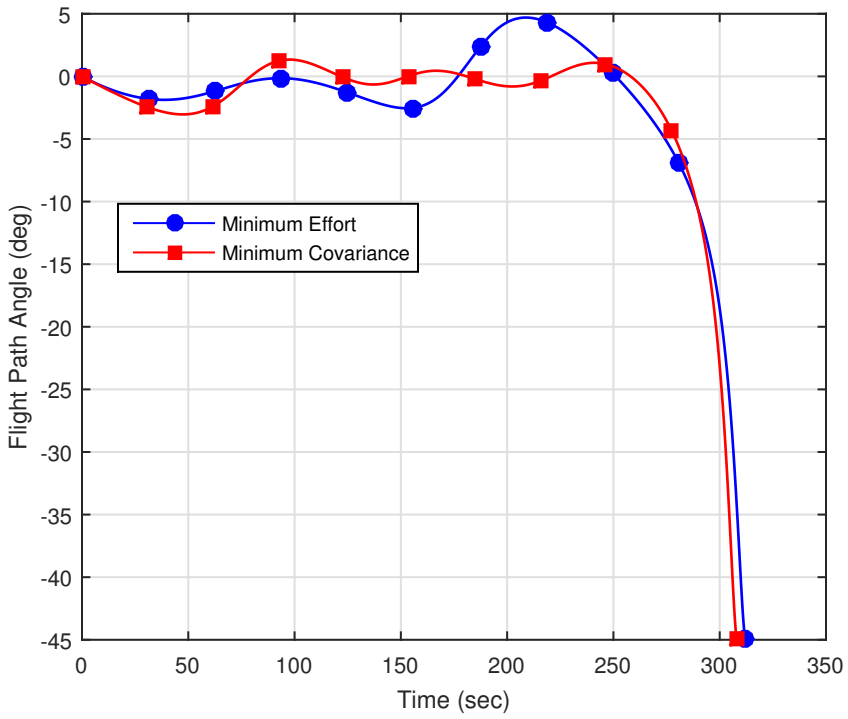

Figure A-3: Flight path angle for minimum effort trajectory and minimum latitude and longitude standard deviations trajectory

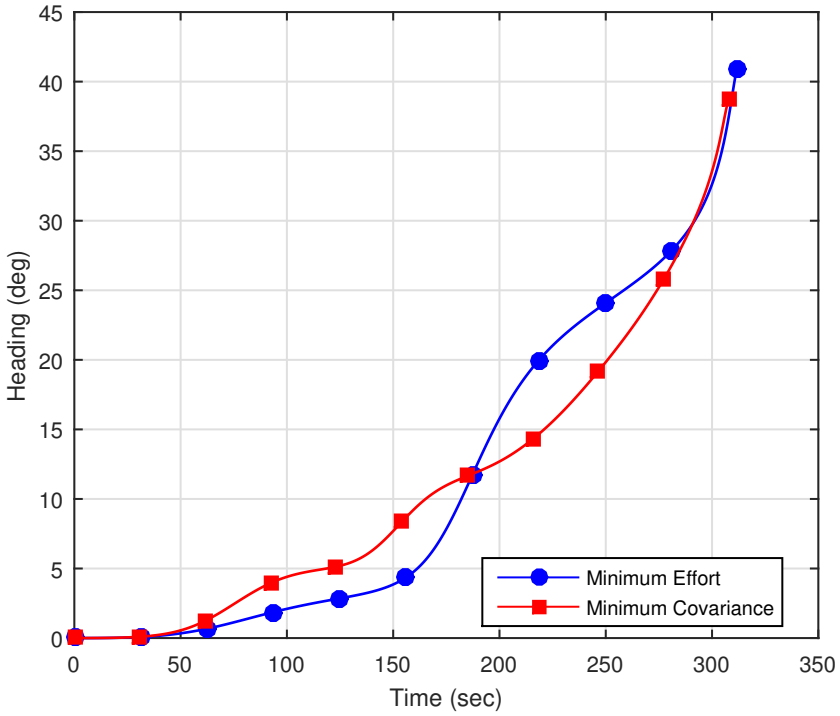

Figure A-4: Heading angle for minimum effort trajectory and minimum latitude and longitude standard deviations trajectory

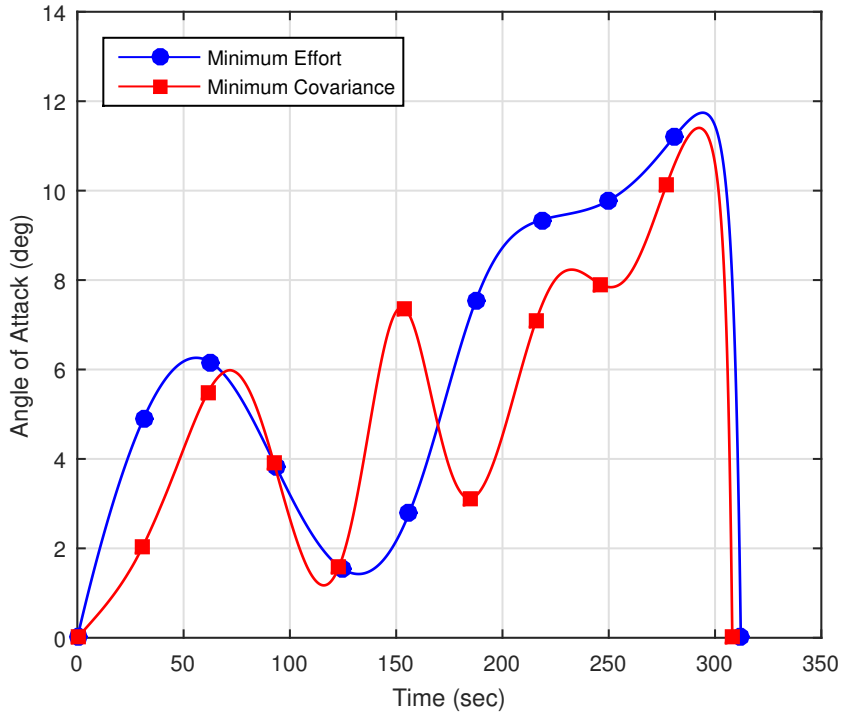

Figure A-5: Angle of attack for minimum effort trajectory and minimum latitude and longitude standard deviations trajectory

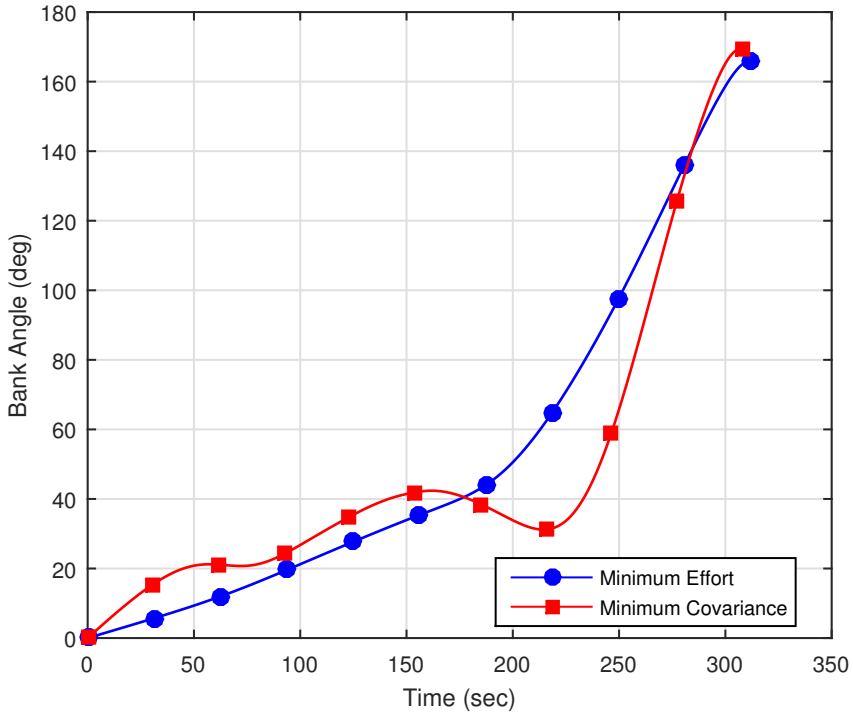

Figure A-6: Bank angle for minimum effort trajectory and minimum latitude and longitude standard deviations trajectory

Distribution Statement A – Approved for Public Release

Appendix B

Plots of States from Spacecraft Reentry Problem with an IMU

In chapter 5 only figures of the trajectories and the east and north position standard deviations were included. The normalized RMS for the different methods were included in tables in chapter 5, but plots of the other states were not included. The actual path that the latitude, velocity, flight path angle, heading angle, angle of attack, and the bank angle took did not contribute to the discussion in chapter 5 so they were omitted. So the reader can see what path the optimal solver converged to for these states, the plots are included here. Figures B-1 to B-6 are the states for he minimum effort trajectory and trajectories *a*-*d*.

Trajectories were also developed for cases *a*-*d* without an altimeter. Figures B-7 to B-12 show the remaining states for the minimum effort trajectory and trajectories *a*-*d* when no altimeter is included in the problem.

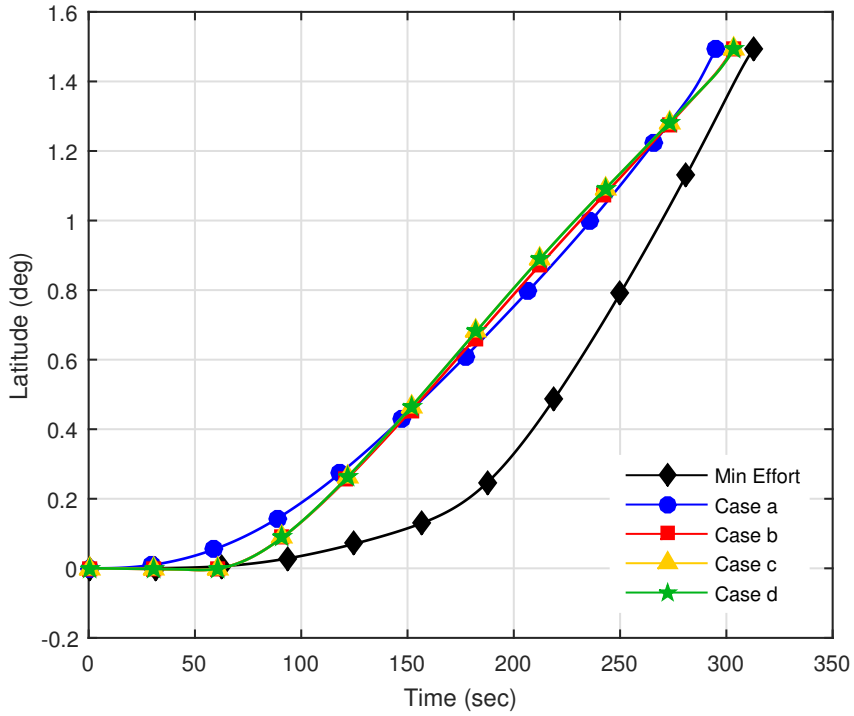

Figure B-1: Latitude of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is included

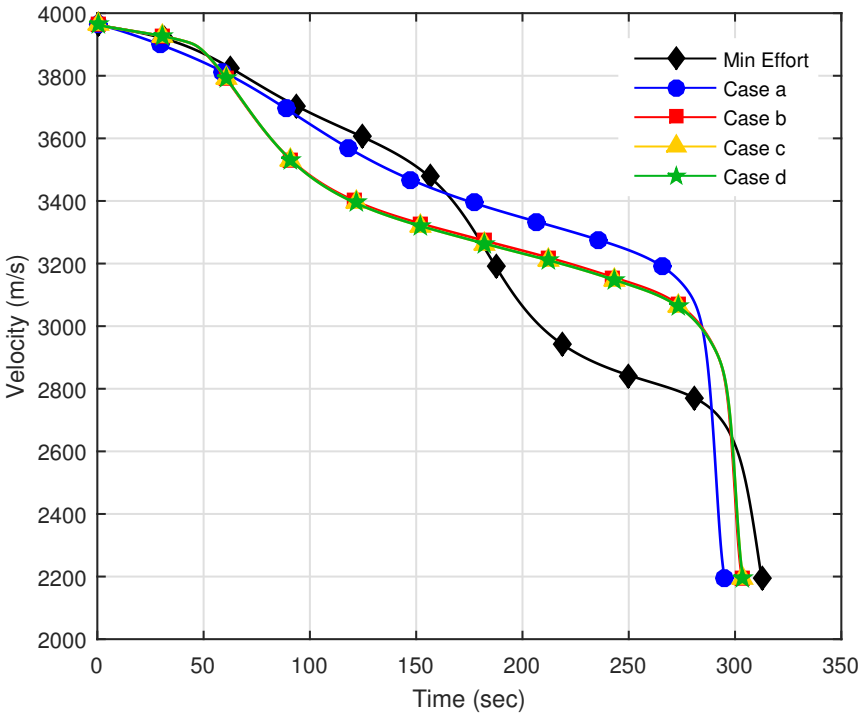

Figure B-2: Velocity of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is included

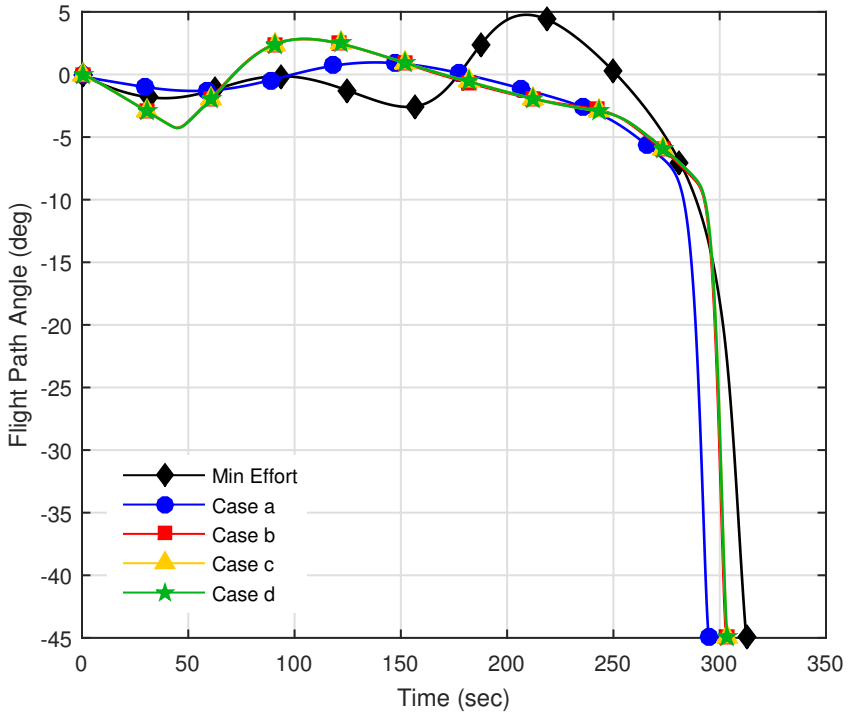

Figure B-3: Flight path angle of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is included

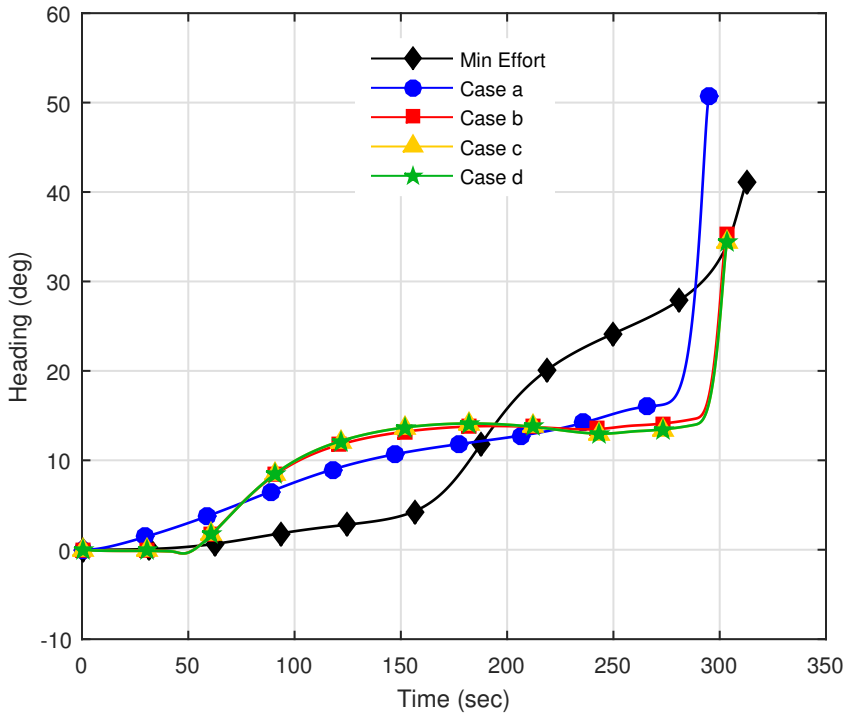

Figure B-4: Heading angle of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is included

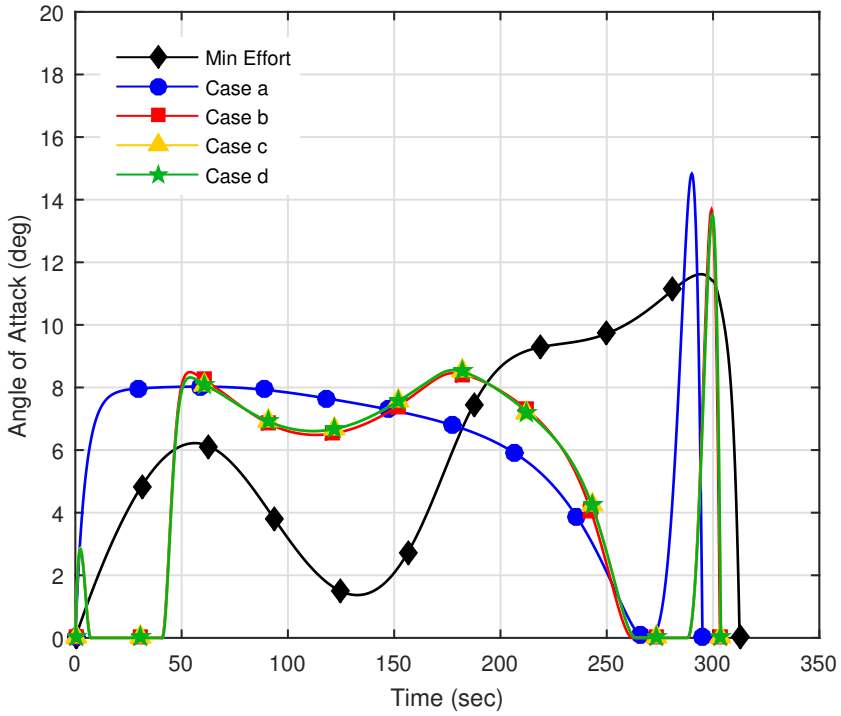

Figure B-5: Angle of attack of reentry vehicle for minimum effort trajectory and cases a-d when an altimeter is included

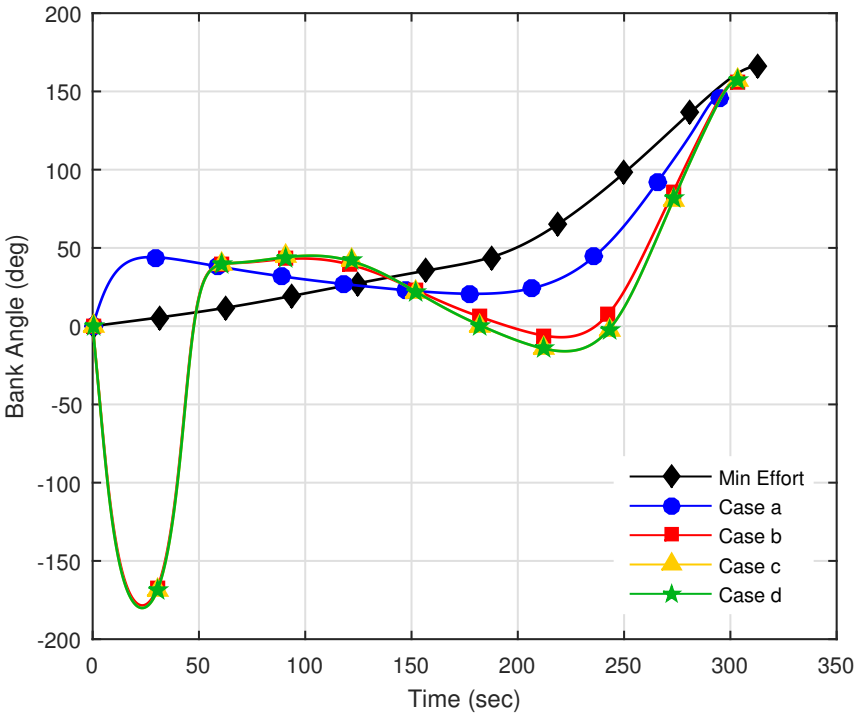

Figure B-6: Bank angle of reentry vehicle for minimum effort trajectory and cases a-d when an altimeter is included

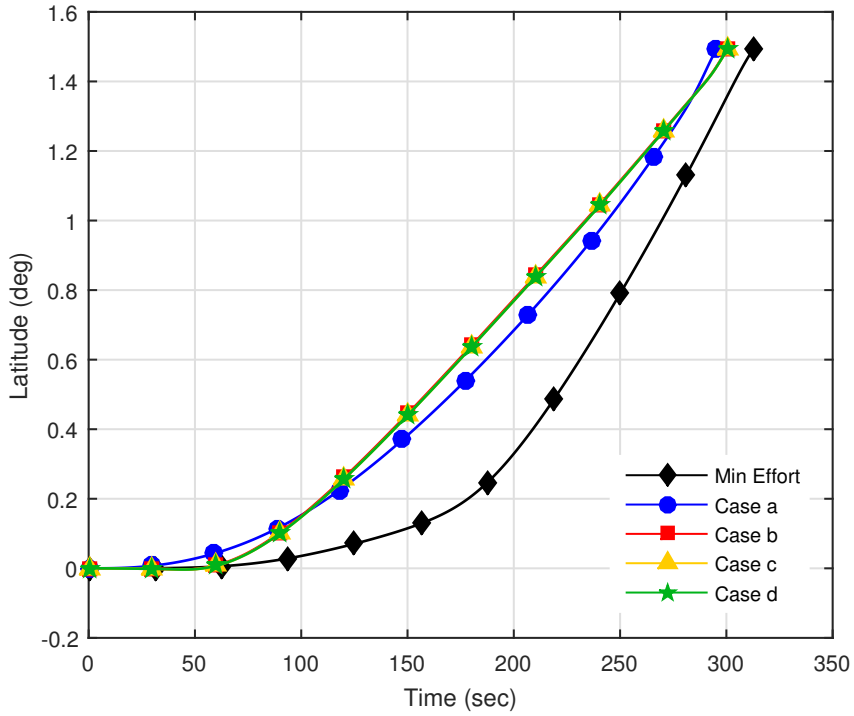

Figure B-7: Latitude of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is not included

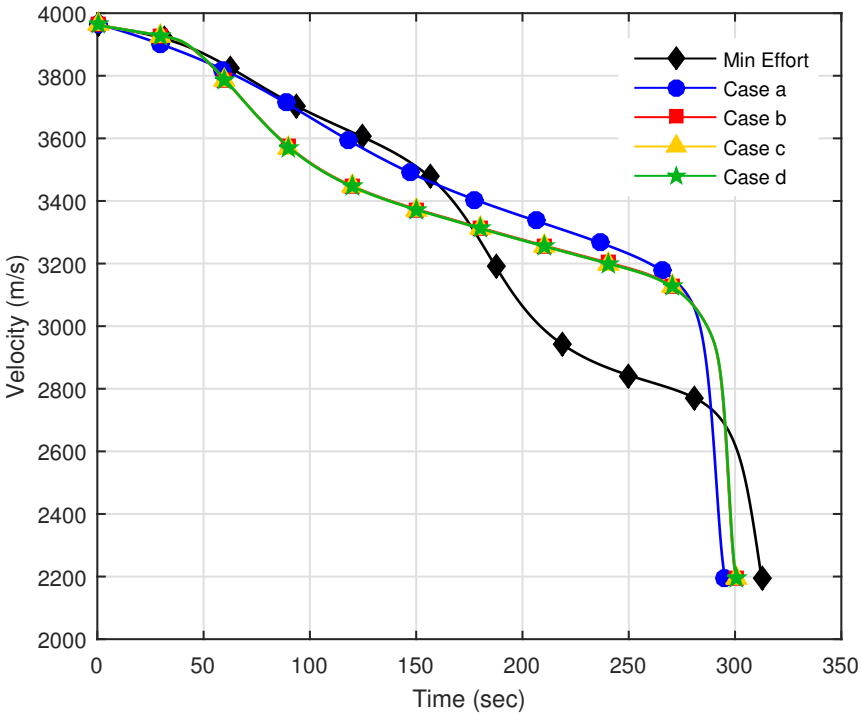

Figure B-8: Velocity of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is not included

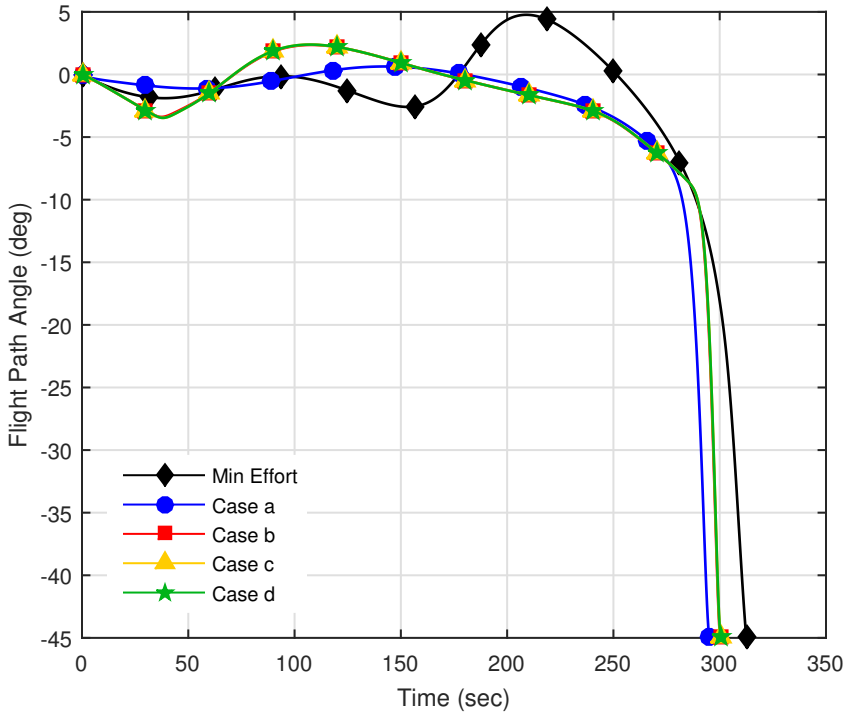

Figure B-9: Flight path angle of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is not included

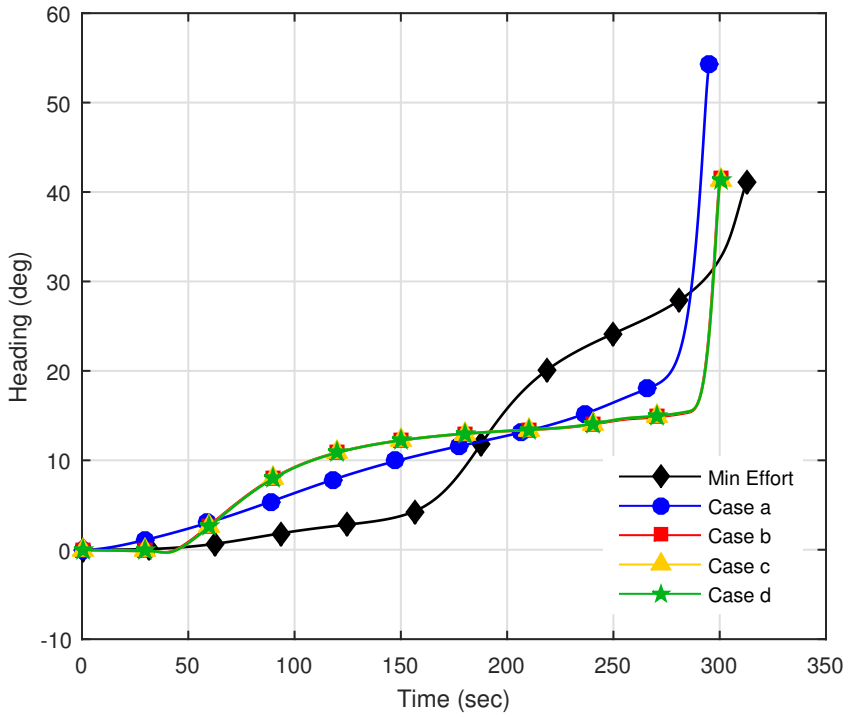

Figure B-10: Heading angle of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is not included

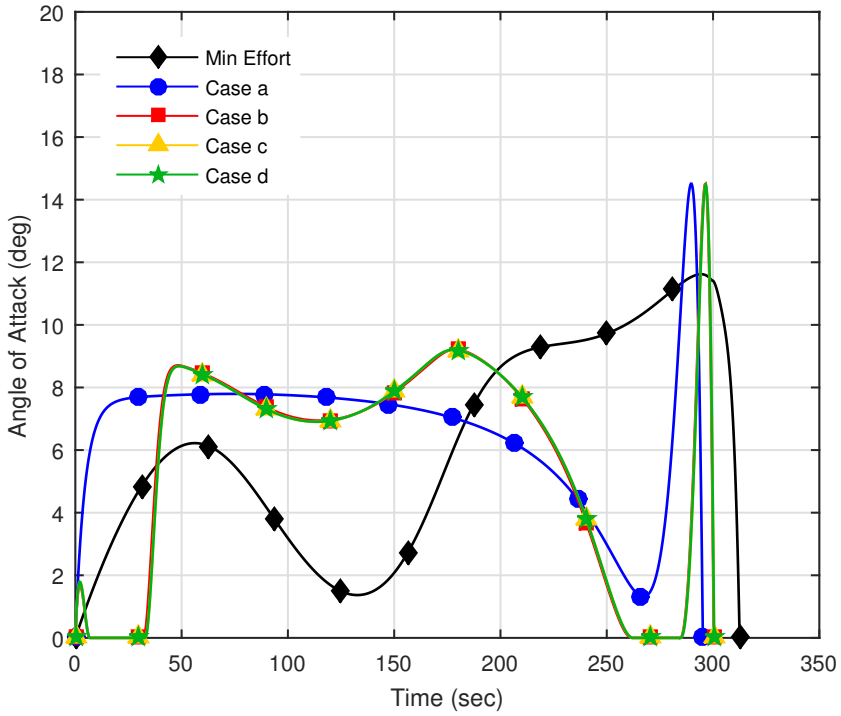

Figure B-11: Angle of attack of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is not included

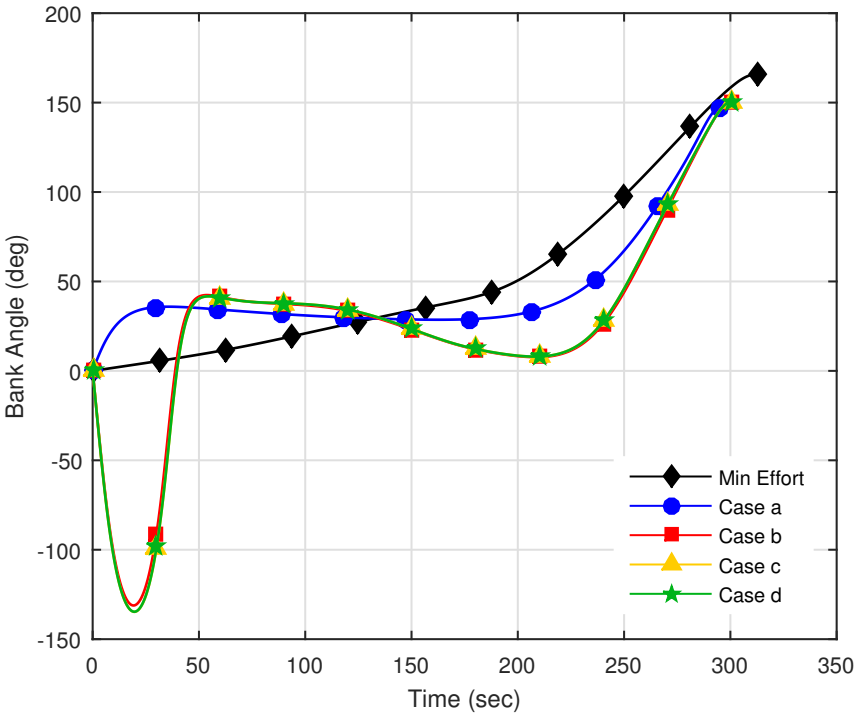

Figure B-12: Bank angle of reentry vehicle for minimum effort trajectory and cases *a*-*d* when an altimeter is not included

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution Statement A – Approved for Public Release

Bibliography

[1] D. Geller, "Linear covariance techniques for orbital rendezvous analysis and autonomous onboard mission planning," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 6, pp. 1404-1414, 2006.

[2] R. Zanetti, D. Woffinden, and A. Sievers, "Multiple event triggers in linear covariance analysis for spacecraft rendezvous," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 353-366, 2012.

[3] D. Vander Stoep, "Trajectory shaping for the minimization of state-variable estiation errors," *IEEE Transactions on Automatic Control*, vol. 13, pp. 284-286, 1968.

[4] H. Hou, W. W. Hager, and A. V. Rao, "Convergence of a gauss pseudospectral method for optimal control," in *2012 AIAA Guidance, Navigation, and Control Conference*, 20124452, AIAA, 8 2012.

[5] D. Garg, Patterson, W. M. Hager, A. V. Rao, D. Benson, and G. Huntington, "A unified framework for the numerical solution of optimal control problems using pseudospectral methods," *Automatica*, vol. 46, no. 11, pp. 1843-1851, 2010.

[6] D. Garg, H. W. W., and A. V. Rao, "Gauss pseudospectral method for solving infinite-horizon optimal control problems," in *2010 AIAA Guidance, Navigation, and Control Conference*, 20107890, AIAA, 8 2010.

[7] D. Garg, M. A. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "An overview of three pseudospectral methods for the numerical

Distribution Statement A – Approved for Public Release

solution of optimal control problems," in *2009 AAS/AIAA Astrodynamics Specialist Conference*, AIAA, 8 2009.

[8] C. Francolin and A. Rao, "Direct trajectory optimization and costate estimation of state inequality path-constrained optimal control problems using a radau pseudospectral method," *AIAA Guidance, Navigation, and Control Conference*, 2012.

[9] C. Darby and A. Rao, "A state approximation-based mesh refinement algorithm for solving optimal control problems using pseudospectral methods," *AIAA Guidance, Navigation, and Control Conference*, Oct 2009.

[10] T. Small, "Optimal trajectory-shaping with sensitivity and covariance techniques," Master's thesis, Massachusetts Institute of Technology, 2010.

[11] B. Saunders, "Optimal trajectory design under uncertainty," Master's thesis, Massachusetts Institute of Technology, 2012.

[12] S. Zimmer, C. Ocampo, and R. Bishop, "Reducing orbit covariance for continuous thrust spacecraft transfers," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 2, pp. 771-791, 2010.

[13] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[14] N. Kasdin, "Runge-kutta algorithm for the numerical integration of stochastic differential equations," *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 1, pp. 114-120, 1995.

[15] S. Hall, "Automatic differentiation methods," February 2017.

[16] M. Weinstein and A. Rao, "Algorithm 984: Adigator, a toolbox for the algorithmic differentiation of mathematical functions in matlab using source transformation via operator overloading," *ACM Transactions on Mathematical Software*, vol. 44, no. 2, 2017.

Distribution Statement A – Approved for Public Release

[17] "1976 u.s. standard atmosphere," 1976.

[18] A. V. Rao, D. A. Benson, C. L. Darby, M. A. Patterson, C. Francolin, I. Sanders, and G. T. Huntington, "Gpops: A matlab software for solving multiple-phase optimal control problems using the gauss pseudospectral method," *ACM Transactions on Mathematical Software*, vol. 37, no. 2, p. 39, 2010.

[19] K. Btitting, *Inertial Navigation Systems Analysis*. Wiley, 1971.

[20] D. Goshen-Meskin and I. Y. Bar-Itzhack, "Unified approach to inertial navigation system error modeling," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 3, pp. 648-653, 1992.

[21] T. Becker, "Approaches to optimal inertial instrument calibration using slewing," Master's thesis, Massachusetts Institute of Technology, 2005.

[22] NovAtel, *IMU-HG1900 Product Sheet*, 2017.