ORBITAL PROPAGATORS FOR HORIZON SIMULATION FRAMEWORK

A Thesis
presented to
the Faculty of California Polytechnic State University,
San Luis Obispo

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Aerospace Engineering

by
Mitra Farahmand
September 2009
ABSTRACT

ORBITAL PROPAGATORS FOR HORIZON SIMULATION FRAMEWORK

Mitra Farahmand

This thesis describes the models of four common orbital propagators and outlines the process of integrating them into the Horizon Simulation Framework (HSF). The results of the Two-Body, $J_2$, and $J_4$ propagators from the HSF are then compared against the outcomes of these propagators in MATLAB and Satellite Toolkit (STK). The MATLAB algorithms verify the functionality of the propagators and determine the accuracy of the HSF implementation. The compassion against STK validates the formulation of the HSF propagators.

In order to equip the HSF with a more precise means of orbit determination, adding the Simplified General Perturbations 4 (SGP4) propagator to the HSF has been the principal goal of this project. A brief description of the algorithm explains the process of configuring the original code into a format compatible with the HSF. Further, the orbital data from the SGP4 propagator across different implementations are examined. The outcomes demonstrate that the HSF algorithm generates reasonably accurate orbital data.
ACKNOWLEDGEMENTS

I wish to acknowledge the support and encouragement of my family and in particular, to dedicate this thesis to the memory of my late father, Jahangir Farahmand, whose lifelong interest in science and technology fueled my passion for nature and the science of space. My mother, Malaekheh Farahmand, has provided unwavering support throughout my life and remains a constant source of encouragement and loving friendship.

I thank my graduate advisor, Dr. Eric Mehiel, for his academic guidance, patience, and thoughtful critique of this report. Finally, I thank my friends and colleagues at Cuesta College and Cal Poly for their kind support and encouragement during the past two years.
# Table of Contents

List of Figures .............................................................................................................................. vii  
Nomenclature ....................................................................................................................................... viii  
I. Introduction ........................................................................................................................................ 1  
   1.1 Horizon Simulation Framework ................................................................................................. 2  
   1.2 Computational implementations ................................................................................................. 4  
   1.3 Objectives .................................................................................................................................... 6  
II. A review of propagators ............................................................................................................... 7  
   2.1 Spherical and nonspherical central body .................................................................................... 7  
   2.2 Simplified General Perturbations – 4 (SGP4) ............................................................................ 11  
      2.2.1 Background ............................................................................................................................ 11  
      2.2.2 Transfer to the HSF. .............................................................................................................. 12  
   2.3 Perturbation Techniques ............................................................................................................ 14  
      2.3.1 Numerical Techniques .......................................................................................................... 14  
      2.3.2 Analytical Techniques .......................................................................................................... 21  
III. Methods of Analysis ..................................................................................................................... 26  
   3.1 HSF vs. MATLAB ...................................................................................................................... 26  
   3.2 HSF vs. STK ............................................................................................................................... 28  
   3.3 Implementations of the SGP4 propagator ................................................................................... 29  
IV. Results and Discussion ................................................................................................................ 30  
   4.1 HSF vs. MATLAB ...................................................................................................................... 30  
      4.1.1 Two-Body .............................................................................................................................. 30  
      4.1.2 $J_2$ and $J_4$ Propagators ..................................................................................................... 35  
      4.2 HSF vs. STK ............................................................................................................................. 38  
      4.2.1 Two-Body .............................................................................................................................. 38  
      4.2.2 $J_2$ and $J_4$ propagators ..................................................................................................... 41  
      4.3 Analysis of the SGP4 propagator .............................................................................................. 44  
      4.3.1 Near Earth ............................................................................................................................ 44  
      4.3.2 Deep Space ............................................................................................................................ 45  
V. Conclusion ......................................................................................................................................... 50  
References ............................................................................................................................................. 52  
Appendix A. Models of propagators ................................................................................................. 53  
Appendix B. TLE Data ......................................................................................................................... 55  
Appendix C. User Guide ..................................................................................................................... 56
List of Figures

Figure 1.1 The essential components of the HSF design. ................................. 3
Figure 2.1 Errors: local and global. ................................................................. 18
Figure 2.2 Variations: secular and periodic. ......................................................... 22

Figure 4.1 HSF vs. MATLAB: Two-Body .............................................................. 33
Figure 4.2 HSF vs. MATLAB: Two-Body, periodic behavior .................................. 33
Figure 4.3 HSF vs. MATLAB: Two-Body, distance ................................................. 34
Figure 4.4 HSF vs. MATLAB: Variation in step-size ............................................... 34
Figure 4.5 HSF vs. MATLAB: $J_2$ propagator ....................................................... 36
Figure 4.6 HSF vs. MATLAB: $J_2$ propagator, periodic behavior ......................... 36
Figure 4.7 HSF vs. MATLAB: $J_4$ propagator ....................................................... 37
Figure 4.8 HSF vs. MATLAB: $J_4$ propagator, periodic behavior ......................... 37
Figure 4.9 HSF vs. STK: Two-Body ................................................................. 39
Figure 4.10 HSF vs. STK: Two-Body, periodic behavior ........................................ 40
Figure 4.11 HSF vs. STK: Two-Body, position ...................................................... 40
Figure 4.12 HSF vs. STK: $J_2$ propagator ............................................................ 42
Figure 4.13 HSF vs. STK: $J_2$ propagator, periodic behavior .................................. 42
Figure 4.14 HSF vs. STK: $J_2$ propagator, distance .............................................. 43
Figure 4.15 HSF vs. STK: $J_4$ propagator, distance .............................................. 43
Figure 4.16 HSF vs. Vallado’s version, SGP4, Near Earth ..................................... 46
Figure 4.17 HSF vs. STK, SGP4, Near Earth ....................................................... 47
Figure 4.18 HSF vs. STK, SGP4, Near Earth, periodic behavior ......................... 47
Figure 4.19 HSF vs. STK, SGP4, Near Earth, distance ......................................... 48
Figure 4.20 HSF vs. Vallado’s version, SGP4, Deep Space .................................... 48
Figure 4.21 HSF vs. STK, SGP4, Deep Space ...................................................... 49
Figure 4.22 HSF vs. STK, SGP4, Deep Space, distance ....................................... 49
# Nomenclature

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSF</td>
<td>Horizon Simulation Framework</td>
</tr>
<tr>
<td>STK</td>
<td>Satellite Toolkit</td>
</tr>
<tr>
<td>SGP4</td>
<td>Simplified General Perturbations 4</td>
</tr>
<tr>
<td>HPOP</td>
<td>High-Precision Orbit Propagator</td>
</tr>
<tr>
<td>SPICE</td>
<td>Spacecraft Planet Instrument C-matrix Events Kernels</td>
</tr>
<tr>
<td>WGS</td>
<td>World Geodetic Survey</td>
</tr>
</tbody>
</table>
I. Introduction

One of the fundamental yet challenging topics in Astrodynamics is orbit determination. Propagators are considered the tools for orbital analysis, which are selected based on both application and precision requirements. Two main factors lead to choosing the right propagator. One is recognizing the forces that control or perturb the orbit; two is estimating their range of influence.

From Classical Mechanics, it can be shown that the central body forces a satellite to rotate on a circular or an elliptical path. Johann Kepler (1571 - 1630) discovered this phenomenon for the first time, as he expressed the highly accurate celestial observations conducted by Tycho Brahe (1546 - 1601), into mathematical formula. Later, Sir Isaac Newton (1642 - 1727) provided the physical explanations for these empirical results by discovering that the gravitational force of the central body is the driving factor. After a few centuries and with the contribution of many physicists, geophysicists, astronomers, and mathematicians, finally a sophisticated model for the Earth’s gravitational field has become available. In addition, other external forces that affect the motion of satellites such as air drag, third-body, etc. have been carefully mapped out.

The focus of this thesis is to integrate four of the most popular propagators into the Horizon Simulation Framework (see section 1.1). The propagators that are discussed in this thesis are well researched and studied in literature. The \textit{Two-Body} propagator is formed when the gravity of the central body is considered the sole affecting force. As long as the gravitational field is assumed symmetric, the results exhibit Keplerian motion. The next two are the \textit{J}_2 \text{ and } \textit{J}_4
propagators, which result when the nonspherical terms of the central body’s gravitation are considered as well. The other prominent propagators are SGP4, HPOP, and SPICE. These include a number of external perturbations in addition to the Earth’s gravitational field. This thesis is primarily concerned with the SGP4, Simplified General Propagator 4, a sophisticated model that includes the Earth’s gravitational field with the first few perturbing terms, air drag, and solar / lunar effects. These propagators will be discussed in more detail in chapter 2.

1.1 Horizon Simulation Framework

The Horizon Simulation Framework (HSF) is an on going project developed by the faculty and students of the Aerospace Engineering Department at California Polytechnic State University, San Luis Obispo (Cal Poly). The HSF is considered a systems engineering tool that employs a time-driven simulation algorithm to understand the functionality of a set of subsystems or model of a satellite. One such example is the Long View Space Telescope;¹ a design project that has also been created at Cal Poly. Figure1.1 summarizes the HSF design philosophy² and depicts an important feature of this design; the main scheduling algorithm operates independently from the system model. Also, notice that the system model is created from subsystems that are essentially modular, but the interfaces between them and the main algorithm can be established as desired.
The analysis and simulation of HSF generates operational schedules. The subsystems can be added as needed and are designed to essentially encapsulate a complete but independent functionality, such as the power subsystem, sensor subsystem, etc. Along with the main scheduling, and subsystem development, a set of mathematical utilities, geometrical facilities, and input / output capabilities have been created for the HSF to deliver it as a stand-alone framework. The current project provides the diverse methods of orbit determination to expand the geometrical resources of HSF.

The HSF is written in C++, an object-oriented language. A class structure in object-oriented programming helps create modular code. All the methods in HSF including the geometrical resources are written as a class. Two classes, EOMS and Position are in particular relevant to this project. The EOMS class defines the Equations of Motion, thus it captures the core of propagators’
functionality including the perturbing forces. The Position class, on the other hand, offers a selection of coordinate systems and conversion among them, as well as providing the input / output data, and finally facilitating the routines for numerical integration. The EOMS and Position class together determine the path of motion for different types of objects. The methodology for the satellites’ motion is that the EOMS class provides the model of a propagator and the Position class facilitates the solution method. Throughout this project, the EOMS and Position class were upgraded to compute the position of satellites with a better precision. This was accomplished by adding the $J_2$, $J_4$, and SGP4 propagators to the HSF.

### 1.2 Computational implementations

The algorithms of the propagators were first formed in C++, the language of HSF. Initially, the simplest form of the propagators, the Two-Body propagator, was available in the HSF. So, the next step forward was to form the $J_2$ and $J_4$ propagators that address the asymmetry of Earth’s gravitational field and known to draw in the largest discrepancy to the unperturbed orbit. To solve these propagators, the numerical integration is used (see section 2.3.1). Next, the SGP4 propagator was added to the HSF, which utilizes the analytical approach for orbit determination (see section 2.3.2). The source code that provides the functionality of the SGP4 model is an encapsulation of Vallado’s C++ version.\(^3\)

In order to verify the numerical accuracy of the propagators in HSF, the propagators were also modeled in MATLAB. While the HSF is still in the stages of development, MATLAB has been used for decades in aerospace and other
engineering fields. Thus, one can safely assume that there is less chance of a programming flaw in MATLAB than in HSF. Also, MATLAB provides a fairly fast and easy assessment for the programming resolutions, for example by plotting the orbital path.

Further, the results of the propagators in HSF were compared to those given by Satellite Toolkit (STK) to validate the accuracy of the various propagators considered in this thesis. STK offers sophisticated data analysis for a broad range of applications such as navigation, communication, and mission design and operation. From this thesis’s standpoint, STK offers an intriguing functionality for orbit determination. It provides not only common types of propagators but also through the Astrogator component let the user to create new propagators, such as ‘Venus $J_2$’. All the propagators created for the HSF are available through STK. However, the details of the models and solution methods are somewhat different.

There are several similarities and differences between the HSF and STK propagators to consider (for more details see section 3.2). The STK Help manual confirms that the Two-Body and $J_2$ models are essentially the same in HSF and STK. However, it points out: “Two-Body, J2Perturbation, and J4Perturbation are analytical propagators that generate ephemeris by evaluating a formula.” This marks a significant difference between the methods of analysis for the Two-Body, $J_2$, and $J_4$ propagators in HSF and STK. While, STK adapts analytical techniques for orbit determination, the HSF utilizes numerical integration. On the other hand, the brief history of the SGP4 propagator given in the STK Help manual and the
fact that Vallado’s original codes generate the output in the form of STK’s ephemeris strongly implies that STK has also adapted Vallado’s SGP4 model.

1.3 Objectives

The current project discussed in this thesis analyzes four propagators. The analysis of Two-Body, $J_2$, and $J_4$ propagators serves two goals. First, adding these propagators to the HSF expands the list of the propagators available to the HSF users. The basic structure of these propagators makes them particularly suitable for rough estimates. The second goal related to these propagators is to verify the results and analysis of the HSF implementation by comparing the computational accuracy of the HSF and MATLAB codes.

Adding the SGP4 propagator to the HSF has been, however, the main goal of this thesis. This model compared to the other three models is known for generating the orbital data with a better accuracy. In order to verify the functionality of the SGP4 propagator in HSF, the results of the HSF version and the original code are compared.

Further, the results of all four propagators are compared across HSF and STK to demonstrate the compatibility of the HSF with a commercial software package.

The discussion of the current project takes on the following order. Chapter 2 describes the mathematical model of each propagator. In addition, the solution techniques for orbit determination are briefly discussed. Chapter 3 explains the motivation for each pair of comparisons, and chapter 4 displays the results. Finally, chapter 5 presents a conclusion for this thesis.
II. A review of propagators

This chapter first presents a theoretical background of the propagators that are used throughout the analysis of the current project. Section 2.1 begins with the most basic propagator, the Two-Body model. It then extends the discussion to the $J_2$ and $J_4$ models, which include the perturbation terms originated from the Earth’s gravitational field asymmetry. Section 2.2 reviews the historical development of the SGP4 model and explains the process of integrating it into the HSF. Finally, section 2.3 introduces the perturbation techniques known as common methods for solving the orbit propagation problems.

2.1 Spherical and nonspherical central body

The problem of orbit determination addresses two fundamental questions. One, could the orbital state of an object determine the time of flight and two, does having the time of flight yield the location of object? The Two-Body model introduced and analyzed by Kepler is perhaps the first mathematical model for any orbit determination. While this model is the simplest form of propagators, it addresses the questions stated above and unveils some of the existing complexity in the orbit determination problems. From Newtonian physics, the gravitational field of spherically symmetric body is given by

$$U_{\text{Two-body}} = \frac{GM_\oplus}{r},$$

(2.1)
where, \( r \) is the distance of the satellite from the center of Earth, \( G \) is the universal gravitational constant, and \( M_{\oplus} \) is the mass of Earth. It can be also shown that the acceleration is given as

\[
\ddot{r} = -\frac{\mu \; \bar{r}}{r^3},
\]

where,

\[
\mu = G \cdot M_{\oplus}.
\]

The attempt to solve this problem results in Kepler's equation presented in terms of the Classical Orbital Elements (COE) by

\[
\sqrt{\frac{a^3}{\mu}} = \frac{(t - T)}{E - e \sin E},
\]

with \( E \), the eccentric anomaly, \( a \), the semi-major axis, and \( e \), the eccentricity of the orbit. The COE defines where the object is in its orbit around a central body. Equation 2.4 shows that the time of flight from the periapsis, \( t - T \), is easily found once the position of the object is known. Thus, the answer to the first question of the orbit determination problem is fairly straightforward. The second question, also known as Kepler’s problem or more generally propagation, is rather complicated. As Vallado points out: “Solving Kepler’s equation has intrigued scientists for centuries.” Having the time of flight, Eq. 2.4 suggests that the eccentric anomaly could be only found by a transcendental operation. This leads to approximate estimations such as series expansion or recursion methods (ex. Newton-Raphson iteration). On the other hand, the perturbation techniques as

\[
^* \text{The COE are semi-major axis, eccentricity, inclination, right ascension of node, argument of perigee, and mean anomaly.}
\]
explained in section 2.3, have commanded the attention of scientists and engineers. The perturbation techniques are of particular interest because they include other forces besides the gravitation, such as air drag, third body effect, etc. First, a re-examination of the central body’s gravity, the largest factor in orbit determination, is presented.

When deriving Kepler’s equation, one must assume that the attracting body is spherically symmetrical with uniform density. However, careful observations suggest that the Earth is an oblate spheroid with eccentricity of 0.0818, and flatness of 0.0034. Therefore, the first step would be to address the orbital discrepancy resulted from the Earth’s asymmetric shape. In the general case using the potential function theory, Newton’s law of gravitation (Eq. 2.1) can be rewritten as

\[
U = G \int_{\text{central body}} \frac{1}{\rho_Q} \, dm ,
\]  

(2.5)

where, \(dm\) is the infinitesimal element of the central body’s mass, and \(\rho_Q\) is the distance from this mass to the observing point (satellite’s position). Further, Eq. 2.5 can be written in terms of the geometrical properties of the central body as

\[
U = \frac{G}{r} \int_{\text{central body}} \alpha \cdot P_1[\cos (\Lambda)] \, dm ,
\]

(2.6)

\[
\alpha = \frac{r_Q}{r} ,
\]

where, \(r\) is the distance from the center of the Earth to the observing point, \(r_Q\) is the distance from the center of the Earth to \(dm\), and \(P_1[\cos (\Lambda)]\) is the Legendre polynomial with \(\Lambda\) the angle between \(r\) and \(r_Q\). Using the decomposition formula
of spherical harmonics, one can expand this expression into a form that separates the mathematical modeling of the Earth’s shape from the satellite’s position. The details of this calculation are not discussed here and the interested reader should consult Vallado.\(^4\) The final result for the expanded gravitational potential function is stated as,

\[
U = \frac{\mu}{r} \sum_{l=0}^{\infty} \sum_{m=0}^{l} \left( \frac{R_\oplus}{r} \right)^l P_{l,m} \sin (\varphi_{eq}) \{ C_{l,m} \cos (m\lambda_{sat}) + S_{l,m} \sin (m\lambda_{sat}) \},
\]

where, \(C\) and \(S\) functions depend on the shape and size of the central body only.\(^*\) The analysis of this expression highlights some of the prominent features of the Earth’s asymmetric shape. First, note that when \(l = m = 0\), the spherical potential is resulted. Since \(S_{l,0}\) is zero from the definition, we can separate the terms with \(m = 0\) and form a summation including only the \(C\) functions. In literature, it is common to use \(J_l\) notation (known as zonal harmonics) where \(J_l = -C_{l,0}\). Finally, since \(C_{1,1}\) and \(S_{1,1}\) are zero, we find:

\[
U = \frac{\mu}{r} \left[ 1 - \sum_{l=2}^{\infty} J_l \left( \frac{R_\oplus}{r} \right)^l P_l \sin (\varphi_{eq}) \right] \\
+ \sum_{l=2}^{\infty} \sum_{m=1}^{l} \left( \frac{R_\oplus}{r} \right)^l P_{l,m} \sin (\varphi_{eq}) \{ C_{l,m} \cos (m\lambda_{sat}) + S_{l,m} \sin (m\lambda_{sat}) \}.
\]

The Legendre polynomials behave as independent eigenfunctions, or basis functions, and form the spherical harmonics for the Earth’s gravitational field. They are categorized as zonal, sectorial, and tesseral, among which the zonal

\(^*\) For their formula see Ref. 4, p 542.
harmonics have been primarily studied in literature. The $J_2$ term in particular, is responsible for the largest perturbation due the Earth’s asymmetrical shape. Section 2.3 discusses the application of perturbation techniques to the $J_2$ model.

2.2 Simplified General Perturbations – 4 (SGP4)

2.2.1 Background.

The history of the SGP4 model goes back to almost a half century ago. It was first introduced publicly in 1980 in the Spacetrack Report Number 3.\textsuperscript{5} The significance of the SPG4 model is that for the first time a model for orbit determination was proposed where the results were consistent with the data generated by North American Aerospace Defense Command (NORAD). The source code was originally arranged in five subroutines. The original SGP4 routine was used for Near Earth satellites and the SDP4 routine for the Deep Space objects. The most recent version of the SGP4 code by Vallado,\textsuperscript{3} however, combines these two subroutines into a single model. As mentioned in Spacetrack Report Number 3, the SGP4 model employs Brouwer’s gravitational and atmospheric drag models, and SDP4 in addition includes the third body effects of the moon and the sun, and certain sectorial and tesseral spherical harmonics of the Earth. It must be noted that a lot of ambiguity exists around the development of the SGP4 code simply because no mathematical formulation has been published to backup the source code. For example, Vallado refers to the Brouwer

\textsuperscript{*} For more details see Ref. 4, pp 545-547.
and Clemence algorithm in his book and works through the gravitational effects up to \( J_5 \), whereas in his published code, he only uses the data up to \( J_4 \).

### 2.2.2 Transfer to the HSF.

The SGP4 propagator for the HSF was created based on Vallado’s original C++ version. Vallado’s SGP4 code is organized in four main files: TESTcpp manages the Input and Output; SGpio converts the original “TLE data” into orbital parameters; SGP4UNIT includes the propagator’s main algorithms; SGP4ext provides the mathematical operations used in SGP4UNIT. In order to integrate the original code into the HSF (see section 1.1), the files had to be reformatted so that they either become part of a class or form a new class. In the following, the process of integrating each file into the HSF is described.

In the original code, TESTcpp provides the following elements essential to start the SGP4 propagator: TLE file, the Earth’s gravity model, and the time of propagation. As noted previously, the SGP4 code was initially created to reproduce the NORAD’s orbital data, which are published in a particular format known as Two Line Element (TLE). The first few terms in the TLE file are allocated for the objects’ identification and then followed by a sequence of parameters that characterizes the shape, size, and orientation of the orbit. T.S. Kelso manages the publication of TLE data for a wide range of satellites and space objects. For more information on each parameter in the TLE data represents and how it should be used in calculation, see the CelesTrak website.

---

* See Ref. 4, section 9.7.3.
Converting the TLE data to proper forms for using in the SGP4 algorithm was left the same as in the original code to avoid any loss or misreading of the data, however, encapsulated in a new class. The “TLElements” class was purposely formed as an independent unit to allow accessing the TLE data from other parts of the HSF.

The next step was to include the Earth’s gravity model. Vallado\(^3\) considers the following Worlds Geodetic Surveys, wgs72, wgs72old, and wgs84. For each gravity model, the Earth’s properties such as radius, gravitational constant \((\mu)\), etc. as well as the zonal terms \((J_2, J_3, J_4)\) were saved as a *structure* in “geomUtil”, a class already existed in the HSF. In order to avoid calling this structure repeatedly to retrieve the data for the SGP4 subroutines, a member variable representing the gravity model was added to the “TLElements” class. Moreover, in Vallado’s code a number of parameters that are frequently used for both Near Earth and Deep Space orbit determination are retained in a structure called “Satrec”. During the transition of SGP4 code to the HSF, this structure was upgraded to a class to be compatible with the rest of the HSF code.

Vallado furnishes a number of ways to enter the time of propagation, however this becomes irrelevant in HSF since the main scheduling algorithm controls the time of propagation. The time intervals are set in terms of *minutes from epoch*. Finally, the original code provides the SGP4UNIT file with all the required subroutines to determine the orbit of Near Earth and Deep Space objects. A new EOMS class, “sgp4”, was created for the HSF to encapsulate this functionality. Note that prior to running the code for any desired time interval, the subroutines
must be initialized. By adding a number of member functions to the “sgp4” class, the order of running the subroutines were kept the same as the original code.

### 2.3 Perturbation Techniques

There are three approaches to solve the orbit propagation problems; analytical, numerical, and semi-analytical. This thesis only discusses the first two methods. The semi-analytical method is a combination of the other two, and the interested reader should consult Vallado. The following sections first express the motivation for each technique, then discuss the theory and application of each method, and finally describes their advantages and disadvantages.

#### 2.3.1 Numerical Techniques

The numerical approach is well suited for special perturbation problems, which are defined in terms of specific force models and initial conditions. In mathematical terms such problems are referred to as “Initial Value Problems”. The numerical methods have lately become more popular because of the enhancement in the computational power available to scientists and engineers. In a numerical approach the effect of disturbing forces are added to the Two-Body acceleration as following:

\[
\ddot{r} = -\frac{\mu}{r^3} \dot{r} + \bar{a}_{\text{perturbed}},
\]

where, \(\bar{a}_{\text{perturbed}}\) represents the additional perturbing component. Equation 2.9 is known as Cowell’s formulation. Since the perturbing term can have different forms depending on the affecting forces, it is not possible to derive a general
equation like Kepler’s equation (Eq. 2.4). Vallado\(^4\) derives the perturbing accelerations for a wide range of perturbing forces, among which the results of the \(J_2\) model are presented at the end of this section.

Integrating the acceleration in Eq. 2.9 determines the state of the object on the orbital path. The result of the integration, however, can not be found analytically. Therefore a discussion of numerical integrators is appropriate. There are two main types of numerical integrators, single-step and multi-step. The Runge Kutta integrator is an example of single-step method; as described later, the Runge-Kutta methods evaluate the function at an initial time and add the weighted rates of the function at some intermediate points in order to calculate the function at a final time. The multi-step integrators are predictor-corrector type, where the predictor gives an initial estimate, and the corrector further refines the result. The predictor-corrector types are known to give better accuracy but at the expense of more complexity. For both methods, the ultimate accuracy is obtained by varying the integrator’s step-size. This is particularly useful to eccentric orbits, where over a slowly varying course, large steps, and during abrupt changes, small steps are desirable.

The HSF currently implements the Runge Kutta (RK) integrators of type RK4 and RK45. The RK integrators are based on evaluating only the first order differential equations, whereas in Eq. 2.9 the second order derivative is present. By utilizing the concept of the state equation, one can transform the differential equations of order two or higher to a set of first order equations, which are then
solved using the RK integrators. Therefore, the first task would be rewriting Eq. 2.9 in the form of a state equation as

\[ y(t) = \begin{bmatrix} \dot{r} \\ \omega \end{bmatrix}, \quad \begin{bmatrix} \ddot{r} \\ \omega \end{bmatrix} = f(t, y), \quad y(t_0) = \begin{bmatrix} \dot{r}_0 \\ \omega_0 \end{bmatrix}. \quad (2.10) \]

Equation 2.10 is a set of first order equations, which together with the initial conditions, define an “Initial Value Problem”. In this case, position and the first derivative of position, velocity, form the components of the state variable. The following discussion describes how the RK4 and RK45 methods are applied to solve a single first order “Initial Value Problem”. One could easily extend the results of the scalar form to the full state equation (Eq. 2.10).

Following Shampine,\(^7\) first let’s look at Euler’s integrator (a simpler method than the RK method), which provides the necessary tools to understand numerical integration. Throughout the following discussion, \(y(x_n)\) is used for the solution, whereas \(y_n\) is an approximation to \(y(x_n)\); also, \(x\) replaces \(t\) in Eq. 2.10. The following equations define the scalar form of the “Initial Value Problem”:

\[ \begin{align*}
   y'(x) &= f(x, y(x)), \\
   y(a) &= A. 
\end{align*} \quad (2.11) \]

Euler’s method solves this problem by keeping only the first term in the Taylor series of \(f(x, y(x))\), as,

\[ \begin{align*}
   y_{n+1} &= y_n + hf(x_n, y_n), \quad n = 0, 1, \ldots, N - 1 \\
   y_0 &= A. 
\end{align*} \quad (2.12) \]
where, $h$ is the integration step-size. By studying the convergence of results in Euler’s method,\(^*\) the error in each step is calculated as,

$$\max_{0 \leq n \leq N} |y(x_n) - y_n| \leq \frac{hM^2}{2L} \left( e^{L(b-a)} - 1 \right). \quad (2.13)$$

Thus, the error, $e_n = |y(x_n) - y_n|$, is bounded by a constant times $h$. The right side of the inequality in Eq. 2.13 is usually written as $O(h).$\(^7\) Here, the error is only subject to the approximation of the integration method (in this case Euler’s method), while there are other computational errors resulting from the finite arithmetic precision of computers. By considering the error in calculating $f(x_n, y_n)$, $\varepsilon$, as well as the error in calculating $y_{n+1}$, $\rho$, we ultimately get

$$\max_{0 \leq n \leq N} |y(x_n) - y_n| \leq \frac{e^{L(b-a)}}{L} \left( \frac{hM^2}{2} + \varepsilon + \frac{\rho}{h} \right). \quad (2.14)$$

Equation 2.14 suggests that decreasing the integration step-size, $h$, does not necessarily results in a better accuracy. Therefore, the next step forward is to establish a method through which, varying the step-size controls the error tolerance (that is the ultimate goal of numerical integration). The error handling approach is further explored by examining the RK integrators. The Runge Kutta integrators follow a similar recipe as the Euler’s method. So,

$$y_{n+1} = y_n + h \Phi(x_n, y_n), \quad n = 0, 1, \ldots, N - 1$$

$$y_0 = A,$$ \quad (2.15)

where, function $\Phi(x, y)$ is a generalized form of $f(x, y)$ in Eq. 2.12. By choosing $\Phi(x, y)$ smartly, the integrator’s result converges to the true solution satisfying a desired error tolerance. As mentioned before, the RK integrators evaluate the

\(^*\) See Ref. 7, pp 218-220.
function of integration at a number of intermediate steps indicated by the order of
the RK integration. For example, for RK2:

\[
\Phi(x_n, y_n) = a_1 f(x_n, y_n) + a_2 f(x_n + p_1 h, y_n + p_2 h f(x_n, y_n)) .
\] (2.16)

Similarly the RK4 integrator is formed when the number of intermediate steps
is increased to four. The classical RK4 algorithm follows:

\[
\begin{align*}
\hat{y}_1 &= f(t_0, y_0) , \\
\hat{y}_2 &= f(t_0 + \frac{h}{2}, y_0 + \frac{h}{2} \hat{y}_1) , \\
\hat{y}_3 &= f(t_0 + \frac{h}{2}, y_0 + \frac{h}{2} \hat{y}_2) , \\
\hat{y}_4 &= f(t_0 + h, y_0 + h \hat{y}_3) , \\
y(t) &= y(t_0) + \frac{h}{6} (\hat{y}_1 + 2\hat{y}_2 + 2\hat{y}_3 + \hat{y}_4) + O(h^5) .
\end{align*}
\] (2.17)

This becomes a complete algorithm once the error-handling procedure is
provided to keep the error less than a desired value. As shown in Figure 2.1, in
general there are two interpretations of error, local and global.

---

\* For the values of \( p \) and \( \alpha \) see Ref. 7, p 224.
In theory one is seeking the solution for the differential equation,

\[ y' = f(x, y), \]
\[ y(a) = A. \]  \hfill (2.18)

The global error, \( y_n(x) - y_n \), is the difference in the estimated solution, \( y_n \), from the true unique solution, \( y_n(x) \). But, through the numerical integration, the solution is given only in terms of the local quantities, \( x_n \) and \( y_n \). The integration is therefore based on,

\[ u' = f(x, u), \]
\[ u(x_n) = y_n. \]  \hfill (2.19)

The local error, \( u(x_n) - y_n \), measures the distance between the estimated solution, \( y_n \), and the true local solution, \( u(x_n) \). Therefore, the numerical integration methods deal with the local error.

Further investigation suggests that

\[ e_{\text{global}} = e_{\text{local}} + y_{n+1}(x) - u_{n+1}(x), \]  \hfill (2.20)

where, \( y_{n+1}(x) - u_{n+1}(x) \) denotes the stability of differential equation. From Eq. 2.20, although the local error is not the sole contributor to the global error, they are evidently related. In fact the local error could essentially affect the convergence of the global solution. The local error indeed emerges from the truncation of the terms in the series expansion of \( \Phi(x, y) \), and can be shown to be proportional to the integration step-size.\footnote{See Ref. 7 for more detail.} A formal analysis of two successive integration approximations, such as RK4 and RK5, shows that the difference between the two results gives an estimate of the local error. This provides a
method to improve the solution of the lower order integrator (thus a means for controlling the error) by dictating the step-size to the integration process. A number of approaches have been developed for RK45 error handling procedure.* The HSF employs the RK45 numerical integration technique in a similar fashion to the MATLAB ode45() function.

Next, let’s examine the solution of the $J_2$ propagator by applying the numerical technique. As previously discussed, the first zonal harmonic term in the Earth’s aspherical model consists of the $J_2$ perturbation. From Eq. 2.8, the potential function for the $J_2$ term is:

$$R_2 = -\frac{3J_2 \mu}{2r} \left( \frac{R_\odot}{r} \right)^2 \left( \sin^2 (\varphi_{gc}) - \frac{1}{3} \right). \quad (2.21)$$

By calculating the gradient, the acceleration components are given as,

$$a_i = \frac{\partial R_2}{\partial r_i} = -\frac{3J_2 \mu R_\odot^2 r_i}{2r^5} \left( 1 - \frac{5r_i^2}{r^2} \right),$$

$$a_j = \frac{\partial R_2}{\partial r_j} = -\frac{3J_2 \mu R_\odot^2 r_j}{2r^5} \left( 1 - \frac{5r_j^2}{r^2} \right), \quad (2.22)$$

$$a_k = \frac{\partial R_2}{\partial r_k} = -\frac{3J_2 \mu R_\odot^2 r_k}{2r^5} \left( 3 - \frac{5r_k^2}{r^2} \right).$$

The $J_2$ Integrator subroutine in Appendix A combines the equations above (Eq. 2.22) with those resulted from the symmetric Earth model (Eq. 2.2) and form $f(t, y)$ in Eq. 2.10. Further, applying the RK45 integrator in the HSF or the ode45() function in MATLAB determine the results of the $J_2$ propagator.

* For example see Ref. 7 and Ref. 8.
To wrap up this section, note that the numerical techniques, also known as the special perturbation techniques, are straightforward to implement (as shown in the example above). But the integrators are defined in terms of specific force models and initial conditions. Therefore, if the perturbation model changes, the analysis has to be repeated. Also, numerical integrations usually take longer than analytical techniques to give results, but are more accurate. Finally, there are some inherent errors with numerical analysis such as truncation and round-off errors, which cause the results degrade over a long period of time.

### 2.3.2 Analytical Techniques

One of the advantages of analytical techniques over the numerical ones is that the output provides a qualitative analysis of the orbital path. Therefore, they lead to a better understanding of the perturbation sources and their effects. In analytical methods the original equations of motion are replaced with an analytical approximation, which permits analytical integration. Thus, the analytical technique, also known as the general perturbation technique, results in approximate, *general*, solution that would hold for a limited time interval but any initial condition.

The underlying concept of the analytical approach is the Variation of Parameters (VOP). Based on this method, the solution of the unperturbed system can represent the solution of the perturbed system, provided that the constants in the solution can be generalized as time-varying parameters, or

$$
\frac{d\bar{c}}{dt} = f(\bar{c}, t),
$$

(2.23)
where, \( \vec{c} \) is the set of classical orbital elements. Vallado\(^4\) derives the VOP equations of motion in terms of the classical orbital elements by two methods, Lagrangian VOP for conservative forces, and Gaussian VOP for nonconservative forces. In this section, the method of Lagrangian VOP for the \( J_2 \) perturbation is partially discussed in order to compare the results of the analytical and numerical techniques (see also section 2.2.1).

First, consider how the orbital elements vary over time. Figure 2.2 shows that the variations of orbital elements are categorized as *secular, short-periodic*, and *long-periodic*. Vallado presents the list of the variations caused by different perturbing forces (see Ref. 4, p 722, Table 9-5).

![Figure 9-1. Effect of Perturbation Forces on Orbital Elements. The change in any orbital element, \( c \), is illustrated. The straight line shows secular effects. The large oscillating line shows the secular plus long-periodic effects, and the small oscillatory line, which combines all three, shows the short-periodic effects. You should understand which elements are averaged when using mean elements. The two examples I've shown are greatly exaggerated but representative of possible errors.](image)

**Figure 2.2 Variations: secular and periodic.**\(^4\)

The secular terms vary linearly or proportional to some power of time. Short-periodic effects change on the order of satellite’s period or less, but long-periodic effects have cycles typically one or two orders of magnitude longer. In the analysis of long-range missions usually the mean variables are defined for short-
periodic effects to increase the speed of computation. Also, one may recognize
the type of orbital elements as fast variables (true and mean anomaly) or slow
variables (semi-major axis, eccentricity, inclination, right ascension of the
ascending node, and argument of perigee). The perturbation effects are most
prominently seen in changing the slow variables.

The following discussion demonstrates the application of the analytical
technique to form a solution for the $J_2$ propagator. First, the nature of variations
for each orbital element is examined. Recall from Eq. 2.8, the potential function
with the zonal harmonics only is,

$$U_{zonal} = \frac{\mu}{r} \left[ 1 - \sum_{l=2}^{\infty} J_l \left( \frac{R_{\oplus}}{r} \right) P_l \sin(\phi_{sat}) \right]. \quad (2.24)$$

Applying conservation of mechanical energy (kinetic and potential), one can
express the semi-major axis as

$$\frac{1}{a} = -\frac{2 \xi_2}{\mu} - \frac{2}{r} \sum_{l=2}^{\infty} J_l \left( \frac{R_{\oplus}}{r} \right) P_l \sin(\phi_{sat}), \quad (2.25)$$

where, $\xi_2$ is the Two-Body mechanical energy. Knowing $r$ is periodic (going
from perigee to apogee and back), as well as the spherical harmonics ($P_l$), then $a$
must be purely periodic. Further, the gravitational potential of zonal harmonics is
axially symmetrical; therefore the component of the angular momentum, $h$, along
the polar axis must be a constant, or

$$h \cos(i) = \text{constant}. \quad (2.26)$$
The inclination, therefore, can’t bear any secular variations, or it reaches 90° eventually and forces this component to be zero. Substituting for $h$ in Eq. 2.26 gives

$$h \cos(i) = \sqrt{\mu a(1-e^2)} \cos(i) = \text{constant}.$$ (2.27)

Since $a$ and $i$ can only be periodic, eccentricity turns out to be also periodic. Therefore, among the classical orbital elements, only right ascension of the ascending node, argument of perigee, and mean anomaly experience secular changes as a result of the zonal perturbations. The following example demonstrates the results for right ascension of the ascending node. The Lagrangian VOP equation is

$$\frac{d\Omega}{dt} = \frac{1}{na^2\sqrt{1-e^2}\sin(i)} \frac{\partial R_2}{\partial i},$$ (2.28)

where, $R_2$ represents only the potential function of the first zonal harmonic, $J_2$, (Eq. 2.21). Following Vallado, in order to distinguish the periodic and nonperiodic components, $\sin(\varphi_{b\infty})$ in Eq. 2.21 is substituted by $\sin(i) \sin(\omega + \nu)$. So,

$$R_2 = \frac{-3n^2 J_2 R_\infty^2}{2} \left(\frac{a}{r}\right)^3 \left(\sin^2(i) \sin^2(\omega + \nu) - \frac{1}{3}\right),$$ (2.29)

where, $\omega$ is the argument of perigee, $\nu$, true anomaly, and $i$, inclination. By using a trigonometric identity and further ignoring the short and long periodic terms, Eq. 2.29 simplifies to

$$\tilde{R}_2 = \frac{-3n^2 J_2 R_\infty^2}{2} \left(\frac{a}{r}\right)^3 \left(\frac{\sin^2(i)}{2} - \frac{1}{3}\right).$$ (2.30)
Calculating the average of \((alr)^3\) over one orbital period results in an average for \(\tilde{R}_2\). By substituting \(\tilde{R}_2,_{\text{avg}}\) in Eq. 2.28, the secular variation of right ascension of the ascending node becomes

\[
\dot{\Omega}_{\text{sec}} = -\frac{3n R_\oplus^2 J_2}{2p^2} \cos(i). \tag{2.31}
\]

By examining the nature of the \(J_2\) perturbation, the physical explanation for such variation becomes obvious. The \(J_2\) term, the largest perturbation caused by the Earth’s bulge at the equator, results in an extra gravitational pull toward the equator. The satellite, therefore, crosses the equator sooner than it would if orbiting around a spherical earth (the node regresses for direct orbits). For further discussion on secular and periodic effects consult Vallado.\(^4\)

Let’s conclude this section by discussing some of the features of the analytical methods. The analytical theories are fairly accurate but very complex as demonstrated in the example for the \(J_2\) perturbation. While, they are harder to be developed, they give more insight about the propagation problems. Vallado\(^4\) suggests that most operational analytical theories are limited to the central body and drag. Moreover, when accuracy requires including the effects of the third-body and solar radiation, the numerical integration can typically be implemented just as easily. Also, no principal set of equations like Cowell’s formulation exists to describe every method of analytical solution. In each case, an approach to Variation of Parameters, averaging, and perhaps coordinate transformations are applied to simplify the equations of motion. Finally, the analytical solutions degrade by time same as the numerical ones, but for different reasons.
III. Methods of Analysis

This chapter explains the methods of comparison for the propagators discussed in chapter 2. Section 3.1 presents a description of the algorithms in MATLAB, which is considered a standard for the verification of the HSF code. Section 3.2 discusses the propagators in STK, a commercial software package, which is commonly used to validate the models of the propagators developed for the HSF. The comparison of results in each case is based on the accuracy and precision of the computational implementations.

3.1 HSF vs. MATLAB

The HSF and MATLAB versions of the Two-Body, $J_2$, and $J_4$ propagators were formed in parallel. The models of propagators and the solution techniques were deliberately chosen to be the same in the two implementations to merely assess HSF programming. The algorithms for $J_2$, and $J_4$ propagators are available in appendix A. As mentioned previously, the HSF version of the Two-Body, $J_2$, and $J_4$ propagators are solved through the numerical techniques. In MATLAB, the numerical integration techniques for solving an “Initial Value Problem” are formally known as Ordinary Differential Equations (ODE). Therefore, the ode4 function is equivalent to the RK4 integrator, and ode45 replaces RK45. Multiple sources recommend RK4 and RK45 as proper methods with sufficient precision for general applications. * Knowing that the RK45 integrator has a built-in method

---

* See Ref. 4, p 524; also MATLAB Help manual.
for error handling (see section 2.3.1), the results of propagators presented in
chapter 4 are only generated by RK45.

One of the goals of this project is the verification and validation of the orbital
data generated by the HSF propagators. This requires a careful study of the
accuracy and precision of the methods through which the results are computed.
Since the integrators in the HSF and MATLAB are employed in a similar fashion,
the role of the integrators' properties in determining the local errors stays the
same. From the MATLAB help manual, the ode45 solver keeps the local error
below a desired tolerance by

$$|e| \leq \max (\text{RelTol} \cdot \text{abs}(y), \text{AbsTol}), \quad (3.1)$$

where, $e$ is the local error, $y$ is the computed quantity, RelTol is the relative
tolerance, and AbsTol is the absolute tolerance. AbsTol determines the smallest
acceptable value resulted from the integration. RelTol denotes the number of
correct significant digits in the result. In order to employ these parameters
correctly, first note that AbsTol should be set less than the computed values or
they become meaningless. Further, when $\text{abs}(y)$ is larger than one, the first term
in Eq. 3.1, RelTol $\cdot$ abs($y$), determines the maximum local error. During the
analysis, the value of RelTol and AbsTol were set the same in both the HSF and
MATLAB integrators. Therefore, the order of RelTol determines the best
achievable accuracy for the computed values. Chapter 4 presents the
comparison of the HSF and MATLAB propagators by calculating the relative
difference of the results and expresses the accuracy in terms of RelTol. If the
relative difference (that is the ratio of the difference of the HSF and MATLAB
results to the MATLAB result) is less than the value of RelTol, then the results from the two implementations are within the margin of integrators' local error, thus comparable. In order to verify the computational accuracy of the algorithms, the direct results of the integrators (the position components) are compared.

3.2 HSF vs. STK

The orbit determination in STK incorporates a wide range of perturbing forces and solution techniques. This section summarizes the similarities and differences of the HSF and MATLAB models for each propagator discussed in this thesis. As mentioned in section 1.2, the gravitational potential functions for the Two-Body and $J_2$ propagators are defined exactly the same way. However, the HSF applies the RK45 integrator to solve these propagators numerically and STK utilizes the analytical methods to develop the “closed-form solutions”. The details of the analytical methods employed by STK are not given, however the Help manual states that for the $J_2$ model only the secular variations are considered in developing the formula. Zonal harmonics in general manifest themselves in both secular and periodic variations. Since the results of the numerical integrators include all types of variations, the HSF algorithm results in a more accurate data than the STK model, which only includes the “prevailing drifting effect”. As for the $J_4$ propagator, the two implementations become difficult to compare; the STK version includes an extra component, the second order of the $J_2$ term, in addition to the ones already applied in the HSF model (the first order of $J_2$ and $J_4$). Furthermore, the difference in the solution techniques, noted for the $J_2$ propagator, still remains for the $J_4$ propagator.
Despite the discrepancies of the HSF and STK models and solution techniques, the results of the STK models are used to validate the method of orbit determination in HSF. During the analysis, the position components in general showed a larger difference than the distance. Recall that the primary role of a propagator is to determine the orbital path of a satellite. Hence, the comparison of distance given by the HSF and STK propagators is taken to determine the accuracy of the HSF methods.

3.3 Implementations of the SGP4 propagator

The codes for the SGP4 propagator in C++ and MATLAB have been available prior to this project. Section 2.2 describes the process of reformatting the original C++ code to a version compatible with the HSF structure. Since the MATLAB copy of this code is said to be “a line-by-line translation of Vallado’s original C++”, section 4.3 only presents the comparison of the HSF and Vallado’s C++ version. Further, the results of the SGP4 propagator from the HSF and STK are compared. Vallado’s SGP4 version holds the methods of orbit determination for both Near Earth and Deep Space objects, which are examined by generating the orbital data for a sample satellite from each category.
IV. Results and Discussion

This chapter presents the results of the computational comparisons that are outlined in chapter 3. Section 4.1 displays the results of the HSF and MATLAB algorithms. Section 4.2 compares the results of the HSF and STK propagators. Finally, section 4.3 is designated to display the comparison of the SGP4 propagator across the available versions (see section 3.3). In all cases, the results are compared and plotted in MATLAB for discussion.

4.1 HSF vs. MATLAB

The initial inspection of the propagators’ results confirmed that the difference in the outputs of the two programming is minimal. Later, a more comprehensive analysis was carried out to determine the numerical compatibility of HSF with respect to MATLAB. It turned out that the integrator’s properties determine the accuracy of the results as discussed in section 3.1.

4.1.1 Two-Body.

The Two-Body propagator evaluates the orbital path of satellites when the gravitational field of the Earth is considered spherically symmetric. The analysis of the orbital size and shape shows that given the initial position and velocity:

\[ r_0 = (7378.137, 0.0, 0.0) \text{ km} , \]

\[ v_0 = (0.0, 6.02088, 4.215866) \text{ km/s} , \]

semimajor axis, eccentricity, and orbital period become:

\[ a = 7378.13376 \text{ km} , \quad e = 4.3781 \times 10^{-7} , \quad T = 105.12 \text{ min} . \]
These results suggest that the orbital path is effectively a circular mid-Earth orbit (LEO), with a period slightly over one and half hour.

The HSF and MATLAB algorithms compute the results of the propagators while the integrator’s parameters, namely RelTol and AbsTol, are set exactly the same. The graphs for comparing the results are generated by calculating:

\[
\text{relative difference in } x \text{- component} = \frac{x_{\text{MATLAB}} - x_{\text{HSF}}}{x_{\text{MATLAB}}}. \tag{4.3}
\]

The discussion in section 3.1 suggests that RelTol determines the accuracy of integration. Therefore, the results of the HSF and MATLAB propagators are comparable if the relative difference of the results (Eq. 4.3) falls within $\pm$ RelTol. The outcomes of the HSF and MATLAB implementations for the $x$-component of position are comparable for the default value of RelTol, $10^{-3}$, and up to $10^{-8}$, shown in Figure 4.1. The results of $y$-component and $z$-component follow similarly. A closer look at the data, Figure 4.2, suggests a periodic pattern, which appears in the results of $J_2$ and $J_4$ propagators as well (see section 4.1.2).

Figure 4.3 shows the plot for the relative difference in distance. Similar to Eq. 4.3, this quantity is calculated by

\[
\text{relative difference in distance} = \frac{d_{\text{MATLAB}} - d_{\text{HSF}}}{d_{\text{MATLAB}}}, \tag{4.4}
\]

where, $d$ from each implementation is given by

\[
d = \sqrt{(x^2 + y^2 + z^2)}. \tag{4.5}
\]

As noted in section 3.2, the graphs of distance in general show a smoother variation than the graphs of position components. Figure 4.3 suggests that the difference in the computed distance remains within the margin of error when
RelTol is lowered by an order of four (that is $10^{-12}$), and the variations only appear on the order of $10^{-14}$.

Next, the variations in the value of the integration step-size across the two implementations, HSF and MATLAB, were examined. In chapter 2, the role of the integration step-size in error handling procedure was introduced. Although, both the HSF and MATLAB algorithms result in the same number of steps, their values are slightly different. The plot in Figure 4.4 demonstrates that the difference in the step-size (time interval in integration) reaches to .5 seconds ($\approx .008$ minutes) in a little over one day. This difference may seem minimal compared to the orbital period (in this case, 105.12 min), but the comparison of results at these time-steps does not correctly reflect the accuracy of the algorithms. By employing an interpolation technique, the results could be obtained at exact intermediate steps. This remedies the problem of varying time-intervals. Also, the interpolation provides the same accuracy at the intermediate steps as specified for the whole integration.† Throughout the analysis, the time interval is set at 60-seconds, and the total period of propagation is chosen as one week. The time interval turns out to be roughly one-hundredth of the orbital period, a reasonable choice for the step-size, since in literature 100 steps per revolution is suggested.‡

---

* See MATLAB help manual.
† See Ref. 4, page 534.
Figure 4.1 HSF vs. MATLAB: Two-Body.

Figure 4.2 HSF vs. MATLAB: Two-Body, periodic behavior.
Figure 4.3 HSF vs. MATLAB: Two-Body, distance.

Figure 4.4 HSF vs. MATLAB: Variation in step-size.
4.1.2 $J_2$ and $J_4$ Propagators

The $J_2$ and $J_4$ propagators determine the orbital path of satellites with a better precision since the models include the partial effects of the Earth’s asymmetric gravitational field. The parameters of the gravitational field are consistently chosen from the WGS84 model,\(^*\) where

\[
\mu = 398600.5 \text{ kg/m}^3, \quad J_2 = 0.00108262998905, \\
R_\oplus = 6378.137 \text{ km}, \quad J_4 = 0.00000161098761. \tag{4.6}
\]

The outcomes of the analysis for the $J_2$ and $J_4$ propagators closely confirm the results obtained for the Two-Body model. Figure 4.5 and Figure 4.7 indicate that the propagators’ data are accurate with RelTol set as low as $10^{-8}$. Figure 4.6 and Figure 4.8 display a similar periodic behavior as previously noticed for the Two-Body results. The period of this cyclic pattern shown in Figure 4.2, 4.6, and 4.8 is about 50 seconds (that is about half of the orbital period). This result suggests that for all the propagators discussed in this section the integrators’ results become significantly different twice in one revolution. One immediate conclusion is that these high discrepancies occur periodically as the satellite passes through the apogee and perigee that have a considerably sharper curvature than the rest of the orbital path.

\(^*\) World Geodetic Survey 84 (41 × 41).
Figure 4.5 HSF vs. MATLAB: $J_2$ propagator.

Figure 4.6 HSF vs. MATLAB: $J_2$ propagator, periodic behavior.
Figure 4.7 HSF vs. MATLAB: $J_4$ propagator.

Figure 4.8 HSF vs. MATLAB: $J_4$ propagator, periodic behavior.
4.2 HSF vs. STK

As explained in section 3.2, the propagators discussed in this thesis are modeled and solved somewhat differently in HSF and STK. While, the HSF applies an alternative solution technique to the Two-Body propagator (that is the numerical method), the $J_2$ and $J_4$ propagators, in addition, bear differences in the formulated models. The plots for each pair (MATLAB vs. STK and HSF vs. STK) were created in parallel for each propagator. Given that section 4.1 confirms the algorithms in HSF and MATLAB are comparable with $\text{RelTol}$ set as low as $10^{-8}$, only the results form HSF vs. STK are provided for discussion. The results are compared and plotted in MATLAB for discussion.

4.2.1 Two-Body.

The goal of comparing the HSF results against the STK output is to validate the HSF methods of orbit determination. STK gives the results of the Two-Body propagator by an analytical technique, whereas HSF gives the solution through a numerical integration. The plots show the comparison of the two implementations by simply evaluating the difference in the results,

$$\text{Difference in}\ x\ \text{-component} = x_{\text{HSF}} - x_{\text{STK}}. \quad (4.7)$$

Figure 4.9 displays the results of the best achievable accuracy. When $\text{RelTol}$ in the HSF algorithm was changed above or below $10^{-8}$, the results showed a larger discrepancy. The difference could reach as large as 0.5 km over one week. Figure 4.10 displays a similar periodic behavior as observed in section 4.1. The period is about one orbital revolution and again indicates that the discrepancies become significantly large twice in one orbital revolution.
When the difference in distance is compared instead, the variations leveled by almost three orders of magnitude, see Figure 4.11. This compassion was established by simply finding the difference in distance calculated from the two implementations,

\[
\text{Difference in distance} = d_{\text{HSF}} - d_{\text{STK}},
\]  

(4.8)

where, \(d\) is given by Eq. 4.5. Note that the data still vary periodically in the order of one orbital revolution. A rather peculiar feature is that the mean of the results is not at zero, but .5 m above zero. In another words, the STK results are lagging behind the HSF data by .5 m.

Figure 4.9 HSF vs. STK: Two-Body.
Figure 4.10 HSF vs. STK: Two-Body, periodic behavior.

Figure 4.11 HSF vs. STK: Two-Body, position.
4.2.2 $J_2$ and $J_4$ propagators.

The similarities and differences of the HSF and STK implementations in formulating the $J_2$ and $J_4$ propagators have already been discussed in section 3.2. The comparison of these two propagators are expected to show less precision than the Two-Body propagator. Figure 4.12 suggests that in one week, the deviation in the $x$-component of position reaches to almost 30 percent of the mean distance. Figure 4.13 shows that the results still vary periodically. Also, Figure 4.14 indicates that the difference in the distance given by the two implementations is in average 7.5 km apart with a 7.5-km variation, that is two orders of magnitude better than the results of the position components.

The comparison of the position components for the $J_4$ propagator shows far more variations than the $J_2$ propagator. This is likely due the larger differences in the models of the $J_4$ propagator in HSF and STK. Also, Figure 4.12 suggests that one-week propagation time is fairly long for validating the results of either the $J_2$ or $J_4$ propagator. Hence, Figure 4.15 displays the comparison of the distance given by the $J_4$ propagator during the first couple of days (see Figure 4.14 for the $J_2$ propagator). The numerical values of the plots in Figure 4.14 and 4.15 are identical despite the fact that the $J_4$ propagator has the additional $J_4$ term, and also the STK version of the $J_4$ propagator has the $J_2^2$ term. This could be explained by noticing that the values of $J_4$ and $J_2^2$ are almost 1000 times less than the value of $J_2$ (see Eq. 4.6). Thus, over the course of couple of days, the distance computed by the $J_4$ propagator does not show any considerable difference from the distance given by the $J_2$ propagator.
Figure 4.12 HSF vs. STK: $J_2$ propagator.

Figure 4.13 HSF vs. STK: $J_2$ propagator, periodic behavior.
Figure 4.14 HSF vs. STK: $J_2$ propagator, distance.

Figure 4.15 HSF vs. STK: $J_4$ propagator, distance.
4.3 Analysis of the SGP4 propagator

Adding the SGP4 propagator to the HSF has been the main goal of this project. The results presented in this section demonstrate that the SGP4 propagator is successfully reconfigured from its original version,\(^3\) and properly integrated into the HSF. Further, the outcomes of the SGP4 propagator from the HSF and STK are compared. The results of the SGP4 propagator across the two implementations show a smaller difference than the results obtained for the simpler models (see section 4.2). In the following, the sample results are presented for two main categories, Near Earth and Deep Space orbits. The TLE data for these orbits are provided in Appendix B.

4.3.1 Near Earth.

The analysis of Satellite 6251 presents the sample results for the Near Earth orbits. This candidate is chosen from the list of the satellites that Vallado has already used in verification of his SGP4 code.\(^3\) It has a low perigee of 377.26 km, where moderate drag cases are applied.

The plot in Figure 4.16 displays the difference in \(x\)-component of position resulted from the HSF and Vallado’s version. The Vallado’s version generates the data with an accuracy of \(10^{-8}\). The HSF, on the other hand, returns the results with a better precision, \(10^{-10}\). The discrepancies observed in Figure 4.16 are clearly due the extra number of significant figures in the HSF output. Moreover, the results of the comparison for the models in HSF and STK are shown in Figure 4.17. Enlarging the plot, as in Figure 4.18, reveals a periodic behavior similar to the one already seen in section 4.1 and 4.2. The period of variations is
in the order of one orbital revolution (for this satellite ≈ 1.55 h). In this case, the difference reaches to the maximum value of 4 km twice in one revolution.

Finally, the satellite’s distance given by the HSF and STK models are compared in Figure 4.19. The plot indicates that the results for the distance become more accurate (by two orders of magnitude) than the position components. The maximum difference is only 40 m apart.

**4.3.2 Deep Space.**

The analysis of Satellite 28129, a Deep Space objects, wraps up this discussion. This satellite is also selected from the list used by Vallado.\(^3\) It is a GPS satellite in a near circular orbit with a period about 12 hours.

The plot in Figure 4.20 displays similar results as in Figure 4.16, the difference in the \(x\)-component of position from the HSF and Vallado’s version. The consistency in the order of variations, \(10^{-9}\), reflects that the limited accuracy of the results from Vallado’s version compared to the ones from the HSF is the source of discrepancy.

Figure 4.21 shows the periodic pattern that is already noted for the outcomes of the Near Earth case in Figure 4.18. Note that the orbital period of this Deep Space satellite is considerably larger than the period of the Near Earth satellite (almost eight times). Thus, the plot in Figure 4.21 readily displays the periodic variations in the course of one revolution for this Deep Space object. Moreover, the plot of the \(x\)-component of position shows that the discrepancies reach to a maximum of 40 km with a 40-km variation. This variation is larger than the one found for the Near Earth satellite (that is 4 km). But note that the Deep Space
satellite has also a larger mean distance than the Near Earth satellite (semimajor axis_28129 = 4.2 R⊕, whereas semimajor axis_6251 = 1.1 R⊕).

At the end, Figure 4.22 shows that the difference in the distance given by the two implementations reaches to a maximum of 10 m. Thus, the comparison of the distance for the Deep Space orbit again shows a better precision than the position components. Also, by comparing the plots in Figure 4.19 and Figure 4.22 with those in Figure 4.14 and Figure 4.15, it is clear that the results of the SGP4 propagator from the HSF and STK are far closer than the results obtained for the J₂ and J₄ propagators.

Figure 4.16 HSF vs. Vallado's version, SGP4, Near Earth.
Figure 4.17 HSF vs. STK, SGP4, Near Earth.

Figure 4.18 HSF vs. STK, SGP4, Near Earth, periodic behavior.
Figure 4.19 HSF vs. STK, SGP4, Near Earth, distance.

Figure 4.20 HSF vs. Vallado's version, SGP4, Deep Space.
Figure 4.21 HSF vs. STK, SGP4, Deep Space.

Figure 4.22 HSF vs. STK, SGP4, Deep Space, distance.
V. Conclusion

The following discussion summarizes the results and analysis of the four orbital propagators that have been integrated into the HSF during this project. The results presented in chapter 4 show a verification of the HSF algorithms through the comparison with either MATLAB or Vallado’s version of the propagators. The orbital data generated by the HSF and STK propagators validate the models of the Two-Body and SGP4 propagators. The results of the $J_2$ and $J_4$ propagators, however, show large discrepancies that are believed to be due the differences in the formulation of the models in HSF and STK.

The comparison of HSF and MATLAB algorithms for the Two-Body, $J_2$ and $J_4$ propagators indicates that the best accuracy for the HSF propagators is achieved when $\text{RelTol}$ in the integrator is set at $10^{-8}$. On the other hand, the comparison of the results for these propagators from HSF and STK becomes divided. The difference in the results of the Two-Body model is not computationally significant when the numerical or analytical methods of solution are applied, whereas the results of the $J_2$ and $J_4$ propagators display a large discrepancy. One explanation is that the models in STK only include the secular variations of $J_2$ and $J_4$ perturbations. Therefore, the STK propagators lack the precision of the numerical methods employed by the HSF. Note that STK, through its HPOP component, offers a numerical approach for solving the $J_2$ and $J_4$ propagators, which puts forward a source of comparison for further evaluating the functionality of the HSF propagators.
The discussion in chapter 4 only presents the results of the $x$-component of position and distance generated by the propagators. The $y$ and $z$ components generally follow the results of the $x$-component. In comparing any pair of implementations, the values of distance turn out to be closer than the results of the position components (usually by few orders of magnitude).

Moreover, the results of any propagator when compared across two implementations show a periodic pattern such that the orbital data become notably different twice in one revolution. This periodic pattern is found not only when the numerical methods are implemented (HSF vs. MATLAB), but also when the numerical and analytical techniques are employed (HSF vs. STK). A simple explanation is that the results of the propagators in general become less accurate at the perigee and apogee where the orbital path changes sharply.

Finally, a review of the Perturbation Techniques helped developing the solution methods for the propagators. Appendix C presents a User Guide for the HSF applicants to choose any of the propagators discussed in this thesis. The HSF users need to specify the solution method under the “PropagationType” in order for the Position Class to process the orbit determination. It would be reasonable to expand the list of the HSF propagators in this fashion. It is worth noting that the SGP4 propagator compared to the other HSF propagators delivers the orbital data with a better accuracy.
References


Appendix A. Models of propagators

**J2_Integrator**

```matlab
function dy=integrator_EOMS_J2(t,y)

mu=398600.5;
RE=6378.137;
J2=0.00108262998905;

r = norm(y(1:3));
r3=r*r*r;
r5=r*r*r*r*r;

rx=y(1,1);
ry=y(2,1);
rz=y(3,1);

a_sph_x=-mu*rx/r3;
a_sph_y=-mu*ry/r3;
a_sph_z=-mu*rz/r3;

cons1J2 = -3*mu*J2*RE*RE/(2*r5);
cons2J2 =5*rz*rz/(r*r);

ax=a_sph_x+(rx*cons1J2*(1-cons2J2));
ay=a_sph_y+(ry*cons1J2*(1-cons2J2));
az=a_sph_z+(rz*cons1J2*3-cons2J2));

dy(1,1)= y(4,1);
dy(2,1)= y(5,1);
dy(3,1)= y(6,1);

dy(4,1)= ax;
dy(5,1)= ay;
dy(6,1)= az;
```

**J4_Integrator**

```matlab
function dy=integrator_EOMS_J4(t,y)

mu=398600.5;
RE=6378.137;
J2=0.00108262998905;
J4=0.00000161098761;

r = norm(y(1:3));
r3=r*r*r;
r5=r*r*r*r*r;
r7=r5*r*r;
rx=y(1,1);
ry=y(2,1);
```

rz = y(3,1);

a_sph_x = -mu*rx/r3;
a_sph_y = -mu*ry/r3;
a_sph_z = -mu*rz/r3;

cons1J2 = -3*mu*J2*RE*RE/(2*r5);
cons2J2 = 5*rz*rz/(r*r);
cons1J4 = 15*J4*mu*RE*RE*RE/(8*r7);
cons2J4 = 14*rz*rz/(r*r);
cons3J4 = 21*rz*rz*rz*rz/(r*r*r*r);

ax = a_sph_x + (rx*cons1J2*(1-cons2J2)) + (rx*cons1J4*(1-cons2J4+cons3J4));
ay = a_sph_y + (ry*cons1J2*(1-cons2J2)) + (ry*cons1J4*(1-cons2J4+cons3J4));
az = a_sph_z + (rz*cons1J2*(1-cons2J2)) + (rz*cons1J4*(5-
((70*rz*rz)/(3*r*r)) + cons3J4));

dy(1,1) = y(4,1);
dy(2,1) = y(5,1);
dy(3,1) = y(6,1);

dy(4,1) = ax;
dy(5,1) = ay;
dy(6,1) = az;
Appendix B. TLE Data

The following TLE data have been selected from the Test Cases accompanied with Vallado’s SGP4 code.³

Near Earth: Satellite 6251

1 06251U 62025E 06176.82412014 .00008885 00000-0 12808-3 0 3985
2 06251 58.0579 54.0425 0030035 139.1568 221.1854 15.56387291 6774

Deep Space: Satellite 28129

1 28129U 03058A 06175.57071136 -.00000104 00000-0 10000-3 0 459
2 28129 54.7298 324.8098 0048506 266.2640 93.1663 2.00562768 18443
Appendix C. User Guide

The following steps explain how to choose a propagator by filling up the required fields at the input files.

- **<Position>…<Position/>Tag**: Propagators are initiated at Position class, thus all the selections described here goes under this Tag.

- **Attributes and nodes**: There are five attributes and three nodes to the Position Tag. Three of these attributes are required for all types of propagators. The other two attributes and all the nodes depend on the type of the propagator. The required data must be provided, or else results in error, and at times the default value. On the other hand, the optional components are by-passed if not needed for a particular propagator.

The required attributes are:

- **PositionType**: The first attribute can be generally chosen from Static, Predetermined, or Dynamic. For orbital propagation, the Type must be PredeterminedECI, meaning that the orbit is evaluated prior to the HSF scheduling and is based on ECI coordinate system. The other coordinate system LLA is not appropriate for orbit determination, however, has other purposes, for example in assigning the “Target” position.

- **PosDataStep**: The second attribute determines the time interval at which the outputs’ are printed. Note that this is NOT the same as the time interval at which the propagators are numerically or analytically evaluated. The time interval for the propagators is provided by the simulation time.
step from simulation input file (schedParams::SIMSTEP). In numerical cases, this value in fact determines \( n\text{Steps} \), the integrator’s step-size.

- **PropagationType**: As discussed in the thesis, two choices are available, *Numerical* and *Analytical*. For Two-Body, \( J_2 \) and \( J_4 \) propagators, select Numerical, and for the SGP4 propagator, choose Analytical.

The optional attributes are:

- **ICS** (numerical Type only): The Initial Conditions consist of the three components of position and velocity. They must be provided in the appropriate coordinate system (that is ECI).

- **GravityType** (analytical Type only): So far, three gravity models are available, wgs72, wgs72old, and wgs84. The default value is wgs84. Other models can be added by saving the data in the “geomUtil” class.

The optional nodes are:

- **<EOMS EOMSType ... />** (numerical Type only): This node leads to the appropriate equations of motion for the Two-Body, \( J_2 \) and \( J_4 \) propagators when at the input file Orbital_EOMS, J2_EOMS, and J4_EOMS are respectively selected. Note; since the SGP4 propagator is the only analytical method available, by default the “sgp4” class is employed when the analytical method is selected.

- **<INTEGRATOR IntegratorType ... />** (numerical Type only): Three types of integrators, RK4, RK45, and Trapz are available that are respectively represented by rk4, rk45, and trapz at the input file. If the integrator’s type is not chosen correctly, the default, rk45, is applied. The value of the
integrator’s parameters, h, rtol, and atol could be chosen at this node. For more information refer to section 3.1. The default values are h = 0.1, rtol=1e-8, and atol=1e-8.

• <TLE Sourcefile ... /> (analytical Type only): Recall that the SGP4 propagator operates by accessing the orbital data from a TLE file. These files are saved in a particular format discussed in Ref. 6. Thus, the data must be either obtained from the original source,\(^6\) or the user’s input file must be matched with a template.\(^9\)