Applications of Covariance Matrices and Jacobians in Orbital Mechanics

Erick White and Audra Rissmeyer

December 11, 2023

Abstract

This paper will explore the concepts of observational covariance matrices, transformations between celestial coordinate systems, and transformations of covariance matrices (and why these transformations would be done). We will use MATLAB functions provided by NASA CARA to run the transformations discussed in Alfano and Vallado's *Updated Analytical Partials for Covariance Transformations and Optimization* and illustrate covariance through our own MATLAB visualizations. We will also illustrate the difference between Cartesian and equinoctial covariance distributions and briefly explain the significance that these distributions have.

Attribution

Erick White wrote the MATLAB code not retrieved from NASA CARA's GitHub, wrote the introduction and summaries of celestial coordinate systems, and derived the examples and numerical results section. Audra Rissmeyer further elaborated on covariance matrices and Jacobians and wrote up the Discussion and Conclusions section. Additionally, she formated the appendix and added the matrices in appendices III, IV, V.

We would like to thank NASA CARA for providing the test cases and much of the MATLAB code used in our paper, as well as Doyle Hall and Luis Baars at Omitron for their invaluable assistance in learning the background material needed to understand the requisite papers.

Introduction

The recent (over the course of only the past couple of decades) surge of satellite deployments into orbit around the Earth has led to a slew of previously unforeseen challenges in the rapidly-developing field of Space Situational Awareness (SSA) [6]. Beyond the difficulties of simply getting the satellites into orbit in the first place, the issue of preventing satellite collisions in an increasingly-cluttered orbital region has been brought to the forefront of space

science, with entire teams (such as NASA's CARA) devoted solely to predicting the positions of satellites, planning maneuvers, and ensuring that the near-Earth space environment remains a safe one.

Among the issues orbital analysts face is the fundamental one of determining where in space a given object is at any time. There are a variety of ways of doing this, but a chief one involves taking measurements using RADAR or other tools from the surface of the Earth; this inherently causes error in the ephemeris (position and velocity) determination of objects, especially if they are small and not well tracked [6]. Additionally, even once position and velocity are found, determining where the satellite will be at a future point in time presents an entirely new set of challenges [1].

There are several concepts the reader must be familiar with to understand the analysis conducted in this paper. The first is the idea of celestial coordinate systems and the differences between them. The most commonly-used system in daily life, and the one the reader is likely most familiar with, is the *Cartesian* system. In this system, position and velocity are represented by three distinct components each, one of which is associated with each basis direction of the coordinate system [5]. A standard Cartesian ephemeris vector may look as follows:

$$\mathbf{v} = \begin{bmatrix} r_x \\ r_y \\ r_z \\ v_x \\ v_y \\ v_z \end{bmatrix}$$

In this configuration, r_x, r_y, and r_z represent the x, y, and z positions respectively of the satellite as measured from the origin of the reference frame and v_x, v_y, and v_z represent the x, y, and z velocity components respectively. Note that there is more than one reference frame in use in orbital mechanics, and reference frames can be divided into "inertial" (non-rotating, although they *do* move with the Earth and so are not strictly inertial in the Newtonian sense) and non-inertial (rotating) frames [5]. The analysis of these frames is outside the scope of this paper, and for all ephemeris given, assume that Cartesian coordinates refers to Cartesian coordinates *in the J2000 frame*, a standard Earth-Centered Inertial (ECI) frame. The use of an inertial frame greatly simplifies the calculations given here [3].

While very easy to use and visualize for objects undergoing linear motion, Cartesian coordinates quickly become impractical for use in orbit determination and propagation. Even modeling simple two-body motion in Cartestian coordinates forces the use of numerical differential equations, something not ideal when processing large numbers of objects [2].

Despite their inconvenience, Cartesian coordinates have the significant advantage of having no singularities; that is, there are no orbital arrangements that cannot be defined in terms of equinoctial coordinates. This is not the case for all celestial coordinate systems.

The other system of significance in this paper is the *equinoctial* system. This system, while much less intuitive than Cartesian elements, is of critical importance in orbital analysis due to the fact that ephemeris error is normally distributed in this system [6]. This fact means that statistical analysis involving satellite positions (*i.e.*, for conjunction risk assessment) is best done in this system. Unfortunately, there is no direct analogue for what each element "means" in the same way that there is for Cartesian elements; it is more of a purely mathematical tool than anything else, and attempts to visualize it are often very difficult [2]. An equinoctial ephemeris vector is as follows:

$$\mathbf{v} = \begin{bmatrix} n \\ a_f \\ a_g \\ \chi \\ \psi \\ \lambda_M \end{bmatrix}$$

In terms of Cartesian elements, these formulations are very complex, requiring many intermediate steps [2]:

- $n = \sqrt{\frac{\mu}{a^3}}$[1]
- $a_f = \mathbf{e} \cdot \hat{\mathbf{f}}$
- $a_g = \mathbf{e} \cdot \hat{\mathbf{g}}$
- $\chi = \frac{w_e}{1 + f_r w_w}$
- $\psi = -\frac{w_q}{1 + f_r w_w}$
- $\lambda_M = F + a_g \cos(F) - a_f \sin(F)$

The intermediate quantities are defined as follows:

- $a = \frac{\mu r}{-v^2 r + 2\mu}$
- $\hat{\mathbf{w}} = \hat{\mathbf{r}} \times \hat{\mathbf{v}} = \begin{bmatrix} w_e \\ w_q \\ w_w \end{bmatrix}$
- $\mathbf{e} = \frac{\left(v^2 - \frac{\mu}{r}\right)\mathbf{r} - (\mathbf{r} \cdot \mathbf{v})\mathbf{v}}{\mu}$[2]

[1] This term is known as the *mean motion*; $\mu = GM$ is the gravitational parameter of the Earth and a is the semimajor axis of the orbit. a is sometimes used instead of n.

[2] Note that this is the eccentricity vector, defined as having magnitude equal to the eccentricity of the orbit and pointing along the line of nodes.

- $\hat{\mathbf{f}} = \dfrac{1}{1+\chi^2+\psi^2}\begin{bmatrix}1-\chi^2+\psi^2\\ 2\chi\psi\\ -2f_r\chi\end{bmatrix} = \begin{bmatrix}1-\frac{w_e^2}{1+w_w}\\ \frac{-w_e w_q}{1+w_w}\\ -w_e\end{bmatrix}$
- $\hat{\mathbf{g}} = \dfrac{1}{1+\chi^2+\psi^2}\begin{bmatrix}2f_r\chi\psi\\ (1+\chi^2-\psi^2)f_r\\ 2\psi\end{bmatrix} = \begin{bmatrix}\frac{-w_e w_q}{1+w_w}\\ 1-\frac{w_q^2}{1+w_w}\\ -w_w\end{bmatrix}$
- $X = r\hat{\mathbf{f}}$
- $Y = r\hat{\mathbf{g}}$
- $b = \dfrac{1}{1+\sqrt{1-a_g^2-a_f^2}}$
- $\sin(F) = a_g + \dfrac{\big(1-a_g^2 b\big)Y - a_g a_f b X}{a\sqrt{1-a_g^2-a_f^2}}$
- $\cos(F) = a_f + \dfrac{\big(1-a_f^2 b\big)X - a_g a_f b Y}{a\sqrt{1-a_g^2-a_f^2}}$

Note the f_r term in these calculations. Equinoctial coordinates, unfortunately, suffer from a singularity for a perfectly retrograde (*i.e.*, $i = 180°$) orbit; however, this singularity is removable if this factor is set to -1 (it is set as 1 otherwise)[3]. This reduction in the number of singularities and the ease of removability makes equinoctial coordinates much more reliable and robust than systems with more singularities, such as classical coordinates[4]. Importantly, equinoctial elements have only one fast variable assuming two-body motion, so orbit propagation is very simple over short time periods (and is performed by varying λ_M).

Mathematical Formulation

Covariance Matrices

The first key mathematical idea that must be introduced is that of a *covariance matrix*. Whenever measurements are made of a quantity, there is an associated uncertainty; when measuring satellite ephemeris, there is an uncertainty associated with every element of the ephemeris, and so if a probabilistic determination of the satellite's position is desired at any time, these uncertainties need to be taken into account [4].

For every measured variable i, define a variable σ_i containing the standard deviation in the measurement of i; additionally, for every pair of variables i, j, define an associated variable μ_{ij} containing the correlation between the two variables. Ideally, these correlational terms

[3]Some authors use $f_r = -1$ for any orbit with $i > 90°$; others use it only when $i = 180°$.

[4]See Appendix I for a formulation and discussion of classical elements.

would be zero, but this is not always (and in fact, is hardly ever) the case. For an arbitrary set of variables $i_1, i_2, \ldots, i_n$, the covariance matrix is as follows:

$$\begin{bmatrix} \sigma_{i_1}^2 & \mu_{i_1 i_2}\sigma_{i_1}\sigma_{i_2} & \cdots & \mu_{i_1 i_{n-1}}\sigma_{i_1}\sigma_{i_{n-1}} & \mu_{i_1 i_n}\sigma_{i_1}\sigma_{i_n} \\ \mu_{i_2 i_1}\sigma_{i_2}\sigma_{i_1} & \sigma_{i_2}^2 & \cdots & \mu_{i_2 i_{n-1}}\sigma_{i_2}\sigma_{i_{n-1}} & \mu_{i_2 i_n}\sigma_{i_2}\sigma_{i_n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu_{i_{n-1} i_1}\sigma_{i_{n-1}}\sigma_{i_1} & \mu_{i_{n-1} i_2}\sigma_{i_{n-1}}\sigma_{i_2} & \cdots & \sigma_{i_{n-1}}^2 & \mu_{i_{n-1} i_n}\sigma_{i_{n-1}}\sigma_{i_n} \\ \mu_{i_n i_1}\sigma_{i_n}\sigma_{i_1} & \mu_{i_n i_2}\sigma_{i_n}\sigma_{i_2} & \cdots & \mu_{i_n i_{n-1}}\sigma_{i_n}\sigma_{i_{n-1}} & \sigma_{i_n}^2 \end{bmatrix}$$

Note that $\mu_{ij} = \mu_{ji}$, so this matrix is symmetric; moreover, a valid covariance matrix is always positive definite[5]. In orbital analysis, the covariance matrices contain the error in the elements of whichever coordinate system is being used. Although data is usually communicated in classical elements in TLEs (Two Line Element sets), covariance matrices are usually distributed in ECI (Earth Centered Inertial) Cartesian coordinates, so the covariance matrices would correspond directly to the error in position and velocity measurements [1].

Covariance matrices serve as a very convenient way of storing variance and covariance information; importantly, since they are symmetric, only slightly over half the data needs to be communicated rather than the entire matrix. They are often written as upper or lower triangular for simplicity. Additionally, covariance matrices provide all the necessary data for Monte Carlo simulations of satellite position, an important tool for modeling and predictions.

Covariance matrices for satellite ephemerides come in multiple sizes; in addition to measuring the position and velocity of the satellite, ephemeris data is sometimes also given regarding solar radiation pressure, atmospheric drag, and other parameters, leading to larger covariance matrices. This paper will deal only with 6×6 covariance matrices for the purpose of predicting satellite position.

Aside: Monte Carlo Simulations

Although not a main focus of the techniques discussed in this paper, Monte Carlo (MC) simulations were used to generate the visualizations given in the analysis, and so should be discussed briefly. These simulations involve taking a mean vector (the measured ephemeris) and varying it according to the uncertainty of the variables (from the covariance matrix) [3]. This produces a large number of possible positions the satellite could be, with more likely regions more densely populated with output results, esentially resulting in a numerical probability density function (PDF). High-fidelity conjunction risk assessments use Monte Carlo analysis as it is the "holy grail" of modeling methods (as it does not need to make any simplifying assumptions as analytical approximations do); however, MC analysis is very time-consuming and computationally intensive, so it is only used in high-risk events. A basic form of MC simulation was used to generate the visuals by creating a PDF at one point in the orbit.

[5]NPD matrices sometimes come up in *covariance synthesis*, but such issues are outside the scope of this paper and NPD covariance matrices must be numerically corrected or discarded.

Jacobian Matrices

Jacobian matrices are a crucial tool in coordinate system transformations. Jacobian matrices are commonly introduced in multivariable calculus as an essential tool when integrating through a change of variables; however, they are also critical in transforming covariance matrices between coordinate systems. Given an initial set of variables $x_1, x_2, \ldots, x_n$ and a final set of variables $y_1, y_2, \ldots, y_n$, the Jacobian matrix associated with the transformation between them is as follows:

$$\begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_{n-1}} & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_{n-1}} & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial y_{n-1}}{\partial x_1} & \frac{\partial y_{n-1}}{\partial x_2} & \cdots & \frac{\partial y_{n-1}}{\partial x_{n-1}} & \frac{\partial y_{n-1}}{\partial x_n} \\ \frac{\partial y_n}{\partial x_1} & \frac{\partial y_n}{\partial x_2} & \cdots & \frac{\partial y_n}{\partial x_{n-1}} & \frac{\partial y_n}{\partial x_n} \end{bmatrix}$$

Note that the direction of the transformation matters; the Jacobian for transforming one system to another is *not* the same as the Jacobian for transforming it back [1].

Consider a covariance matrix A in an arbitrary coordinate system α with a Jacobian matrix J to convert *from* α *to* β. The covariance matrix B in coordinate system β is obtained by:

$$B = JAJ^{-1}$$

Jacobian matrices are critical tools for transforming covariance matrices between the many coordinate systems used in orbital mechanics for satellite position determination. The following Jacobian matrices can be found in Alfano and Vallado's *Updated Analytical Partials for Covariance Transformations and Optimization*: Cartesian to Spherical, Spherical to Cartesian, Cartesian to Classical, Classical to Cartesian, Classical to Equinoctial, Equinoctial to Classical, Equinoctial to Cartesian, and Cartesian to Equinoctial. The formula for the Cartesian to Equinoctial Jacobian matrix used in this paper can be found in Appendix III.

Examples and Numerical Results

The first result derived comes in the form of a basic orbit propagation and visualization for two different satellites. The ephemerides for these satellites are provided as part of CARA's unit testing suite; all measurements are given with base distance in units of meters[6], base time in units of seconds, and are in the Cartesian system using the J2000 reference frame,

[6]The transformation routine utilizes kilometers as the base unit, so a conversion is performed before passing in the data.

in order $\begin{bmatrix} r_x \\ r_y \\ r_z \\ v_x \\ v_y \\ v_z \end{bmatrix}$:

$$\mathbf{c}_1 = \begin{bmatrix} -9.841950433215101 \times 10^5 \\ +3.932342044549424 \times 10^5 \\ +6.991223682230414 \times 10^6 \\ +4.883696742000000 \times 10^3 \\ +5.689086045000000 \times 10^3 \\ +3.665361590000000 \times 10^2 \end{bmatrix} \qquad \mathbf{c}_2 = \begin{bmatrix} -9.839696058965517 \times 10^5 \\ +3.936845951174244 \times 10^5 \\ +6.991219291625473 \times 10^6 \\ +1.509562687000000 \times 10^3 \\ +7.372938617000000 \times 10^3 \\ -1.492509430000000 \times 10^2 \end{bmatrix}$$

These ephemerides were then provided to CARA's `convert_cartesian_to_equinoctial` routine[7] (from the same code base as the test cases). These transformations are the ones outlined in the background section of the paper for converting Cartesian elements to equinoctial elements, and the Jacobian is given here in partial derivative form (see Appendix III for the fully-expanded matrix):

$$\begin{bmatrix} \frac{\partial a}{\partial r_x} & \frac{\partial a}{\partial r_y} & \frac{\partial a}{\partial r_z} & \frac{\partial a}{\partial v_x} & \frac{\partial a}{\partial v_y} & \frac{\partial a}{\partial v_z} \\ \frac{\partial a_f}{\partial r_x} & \frac{\partial a_f}{\partial r_y} & \frac{\partial a_f}{\partial r_z} & \frac{\partial a_f}{\partial v_x} & \frac{\partial a_f}{\partial v_y} & \frac{\partial a_f}{\partial v_z} \\ \frac{\partial a_g}{\partial r_x} & \frac{\partial a_g}{\partial r_y} & \frac{\partial a_g}{\partial r_z} & \frac{\partial a_g}{\partial v_x} & \frac{\partial a_g}{\partial v_y} & \frac{\partial a_g}{\partial v_z} \\ \frac{\partial \chi}{\partial r_x} & \frac{\partial \chi}{\partial r_y} & \frac{\partial \chi}{\partial r_z} & \frac{\partial \chi}{\partial v_x} & \frac{\partial \chi}{\partial v_y} & \frac{\partial \chi}{\partial v_z} \\ \frac{\partial \psi}{\partial r_x} & \frac{\partial \psi}{\partial r_y} & \frac{\partial \psi}{\partial r_z} & \frac{\partial \psi}{\partial v_x} & \frac{\partial \psi}{\partial v_y} & \frac{\partial \psi}{\partial v_z} \\ \frac{\partial \lambda_M}{\partial r_x} & \frac{\partial \lambda_M}{\partial r_y} & \frac{\partial \lambda_M}{\partial r_z} & \frac{\partial \lambda_M}{\partial v_x} & \frac{\partial \lambda_M}{\partial v_y} & \frac{\partial \lambda_M}{\partial v_z} \end{bmatrix}$$

The converted ephemerides are as follows, in order: $\begin{bmatrix} n \\ a_f \\ a_g \\ \chi \\ \psi \\ \lambda_M \end{bmatrix}$:

$$\mathbf{e}_1 = \begin{bmatrix} 0.00106234856601108 \\ -0.00016403932830575 1 \\ 0.00033621122551993 6 \\ -0.86981213549355 \\ -0.757308556918845 \\ -0.76558198088406 \end{bmatrix} \qquad \mathbf{e}_2 = \begin{bmatrix} 0.00105358439130502 \\ 0.00382709339591507 \\ -0.00782905340313533 \\ -1.13725714941296 \\ -0.229344655594952 \\ -0.185988318096853 \end{bmatrix}$$

[7] This routine uses n and λ_M as its equinoctial parameters, hence their earlier usage in the background section.

The covariance matrices are also provided by CARA and are similarly transformed; the Cartesian and equinoctial matrices are provided in their full forms in Appendices IV and V.

Propagating a two-body orbit in Cartesian coordinates is computationally difficult, requiring differential equations to be numerically solved for a large number of time steps around the orbit; however, propagating a two-body orbit in equinoctial coordinates requires only varying the mean longitude. Performing this propagation for both test cases yields the following orbits:

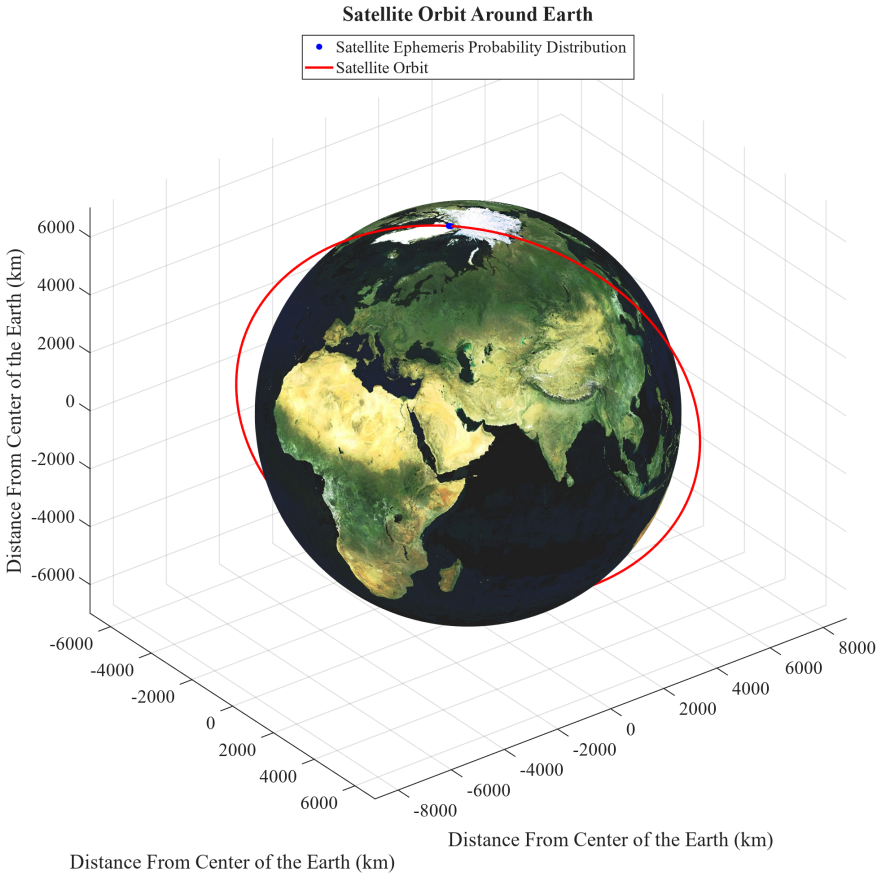

Figure 1: The orbit of a satellite whose covariance can be modeled using the rectilinear assumption.

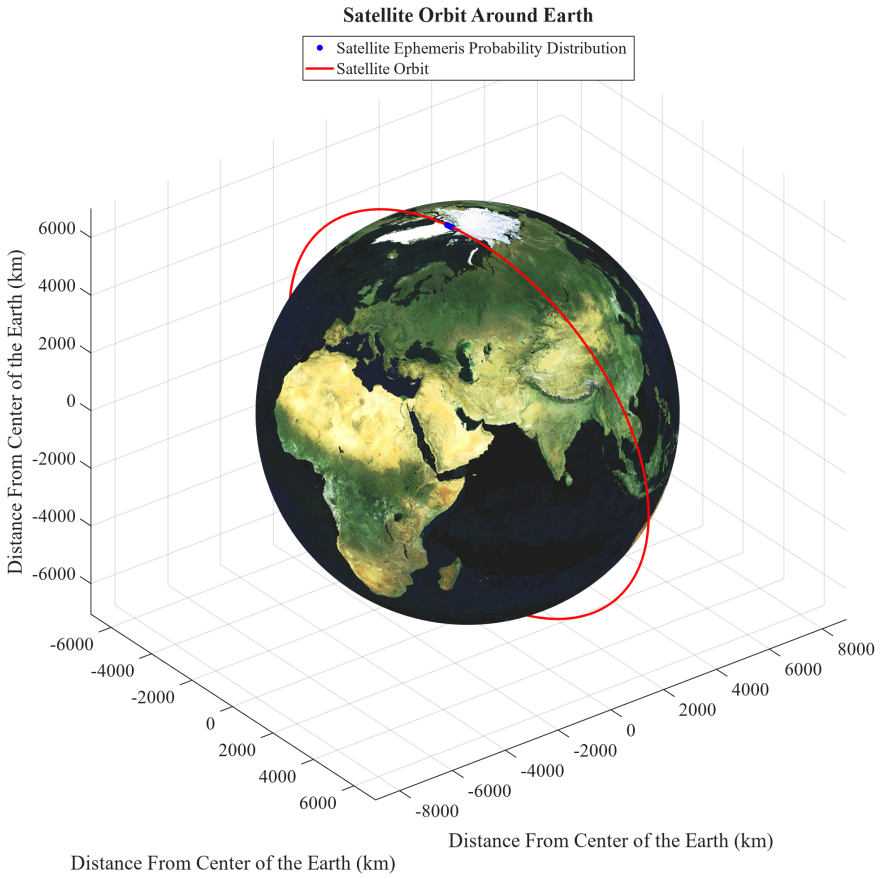

Figure 2: The orbit of a satellite whose covariance cannot be modeled using the rectilinear assumption.

There is nothing of particular note with these two orbits; however, it is important again to note the ease with which these trajectories were calculated. When considering large data sets, performing these conversions and propagating in equinoctial elements (assuming the two-body assumption is valid and the propagation is over an appropriately short time scale) can save a significant amount of time as compared to working in Cartesian coordinates directly.

The other major result is the visualization of the satellites' probability density functions. To obtain a PDF (Probability Density Function), the mean ephemeris vector was varied using the provided covariance matrix to create a point cloud, which was then plotted in three-dimensional space. Since the original ephemeris vectors and covariance matrices are provided in Cartesian coordinates, calculating the PDF using these uncorrected quantities will result in PDFs that do not necessarily reflect the actual probability distributions of the satellites' positions; the elliptical orbits of the satellites mean that they are more likely to lie along points on the orbital path than off the path entirely, meaning that the distribution is not normal in Cartesian space. However, the distributions *are* normal in equinoctial space; therefore, converting to equinoctial coordinates, calculating the distribution there, and then converting back to Cartesian coordinates to plot it results in a more accurate PDF.

These corrections are made using the covariance Jacobian matrices to convert to and from equinoctial space; the results are illustrated below.

Note the distinct difference in the shape of the two PDFs; here, it is plainly visible that the first satellite's motion almost certainly satisfies the rectilinear assumption while the second almost certainly does not. Although not a rigorous enough check for use in more advanced calculations requiring the rectilinear assumption to be satisfied, the visible difference is a good indicator of the validity of the assumption. Additionally, these plots illustrate the aforementioned distribution along the curvature of the orbit rather than in an ellipsoid in Cartesian space.

Aside: The Rectilinear Assumption

One item of significance in these two example satellites is the validity of the rectilinear assumption in the first but not the second. The rectilinear assumption is a common simplifying assumption in which the velocity vector of an orbiting body is treated as constant; this greatly simplifies the analysis of conjunction probability, although such applications are outside the direct scope of this paper. However, the rectilinear assumption manifests in the plots shown here in the shape of the probability density functions: the first satellite's PDF point cloud appears roughly straight, while the PDF of the second satellite visibly curves with the orbital path.

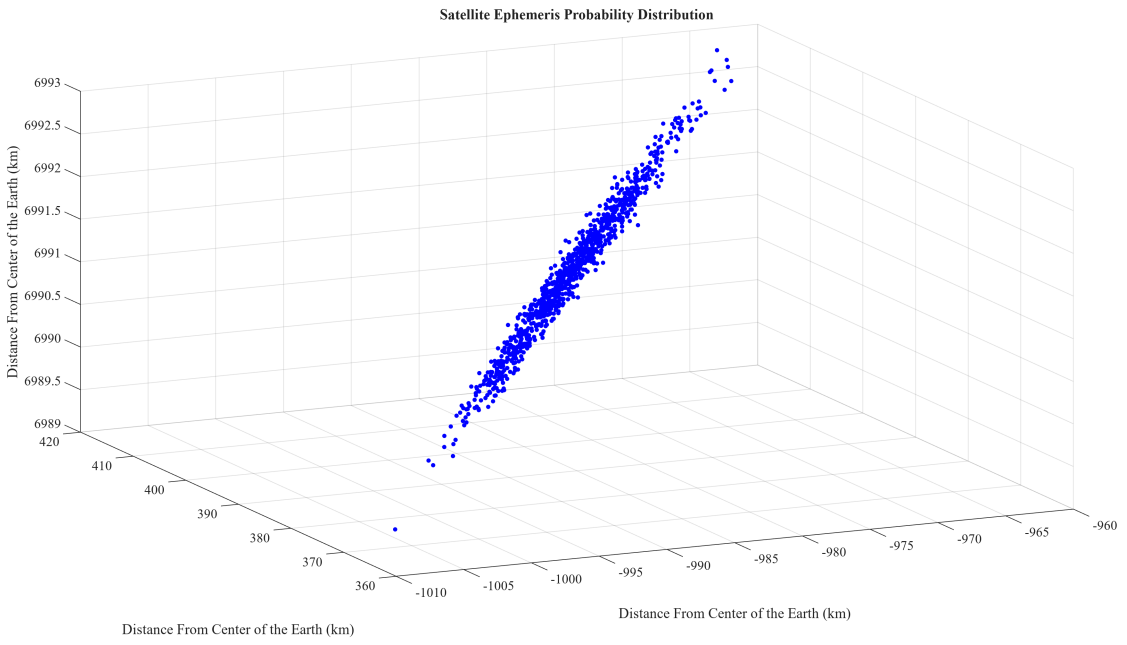

Figure 3: Close up view of the covariance distribution of a satellite whose covariance can be modeled using the rectilinear assumption.

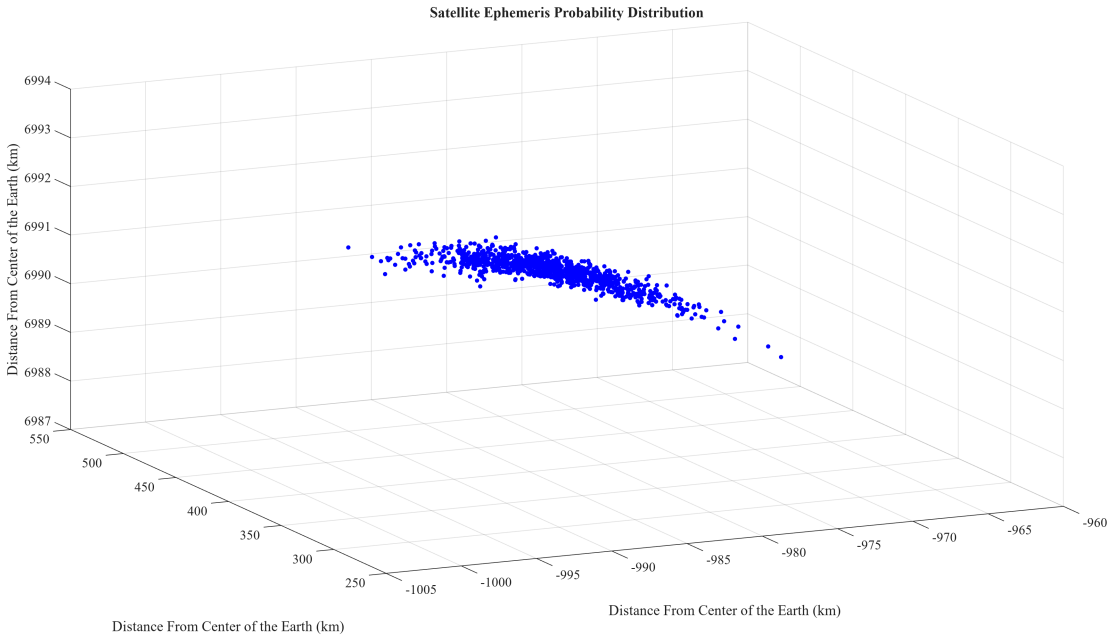

Figure 4: Closeup view of the covariance distribution of a satellite whose covariance cannot be modeled using the rectilinear assumption.

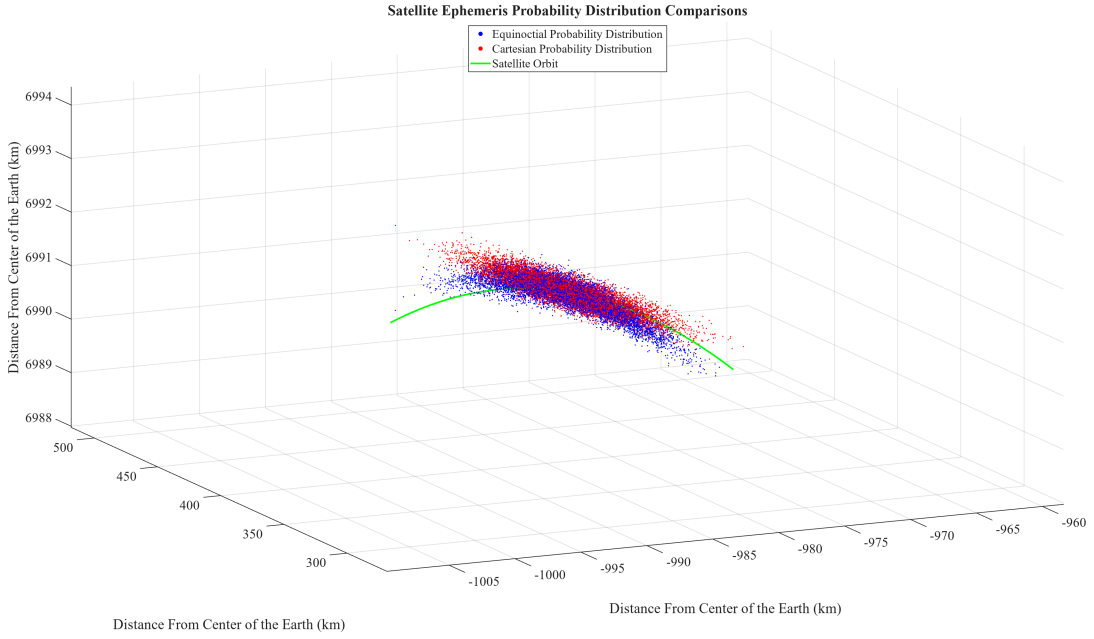

Figure 5: Comparison of the distribution of satellite position predictions in Cartesian (red) vs. equinoctial (blue) systems. Note how the equinoctial distribution better follows the curvature of the orbit.

The final plot is of particular note, as it illustrates the difference between a PDF generated in Cartesian space and one generated in equinoctial space for a satellite whose motion cannot

be modeled under the rectilinear assumption. The equinoctial PDF follows the shape of the orbit, while the Cartesian PDF is only tangent to it; when the PDFs are propagated forwards through time, this result becomes even more pronounced, with equinoctial PDFs taking on a banana-like shape while Cartesian PDFs remain roughly elliptical in shape.

Discussion and Conclusions

The numerical testing conducted in this paper confirms the results of Alfano and Vallado's *Updated Analytical Partials for Covariance Transformations and Optimization*. Using the test cases and MATLAB code provided by NASA CARA, as well as the Jacobian matrices detailed in Alfano and Vallado's paper, test case covariance matrices were successfully transformed from the Cartesian coordinate system to the equinoctial coordinate system. The covariance matrices, before and after transformation, can be found in the appendices.

The results were then plotted using a Monte Carlo simulation to display the probability distribution of two satellites' positions. Visual inspection of the plots shows that the Monte Carlo simulations run with the equinoctial covariance matrices more closely follow the actual curvatures of the satellites' orbits than the simulations run with the Cartesian covariance matrices do; this shows that having a covariance matrix in the equinoctial coordinate system helps to more accurately predict the locations of satellites at any given time. This is especially important when modeling satellites whose behavior cannot be approximated using the rectilinear assumption.

These results are not surprising given the current literature, but nonetheless serve as a more visual demonstration of real-world astrodynamics and satellite orbital analysis. The most surprising result was how pronounced the curvature of the probability density function was for a satellite whose motion could not be modeled using the rectilinear assumption and the amount by which it deviated from the PDF generated purely in Cartesian coordinates.

The next steps to take with this work would likely be to investigate time-dependent covariance matrices and how ephemeris prediction and propagation can be carried out more accurately. Additionally, there is still much work to be done in investigating non-two-body motion, and modeling such systems is a very computationally challenging problem; it is in the best interest of all involved to further develop algorithms to efficiently and effectively anticipate and prevent satellite conjunction events.

In the increasingly crowded near-Earth space environment, there is little room for error. These observations reaffirm the importance of using the proper coordinate system for the given application and knowing when to convert between systems. This paper serves to illustrate techniques for ephemeris and covariance manipulation through the use of Jacobian matrices and the associated statistical results that can be obtained through numerical analysis.

References

[1] S. Alfano and D. Vallado. "Updated Analytical Partials for Covariance Transformations and Optimization". In: 2015.

[2] P.J. Cefola. "Equinoctial Orbit Elements - Application to Artificial Satellite Orbits". In: 1972.

[3] NASA Office of the Chief Engineer. *NASA Spacecraft Conjunction Assessment and Collision Avoidance Best Practices Handbook*. Tech. rep. NASA CARA, 2023.

[4] CueMath. *Covariance Matrices*. `https://www.cuemath.com/algebra/covariance-matrix/`. Retrieved 2023.

[5] T.S. Kelso. *Orbital Coordinate Systems, Part I*. `https://celestrak.org/columns/v02n01/`. Retrived October, 2023.

[6] Navigation and Ancillary Information Facility. *An Overview of Reference Frames and Coordinate Systems in the SPICE Context*. `https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Tutorials/pdf/individual_docs/17_frames_and_coordinate_systems.pdf`. 2023.

Appendices

Appendix I: Classical Orbital Elements

Satellite ephemeris data is typically given as a two-line element (TLE) set in *classical* (or *Keplerian*) elements. This coordinate system has the distinct advantage of being designed specifically for orbital mechanics; rather than giving position and velocity as two distinct entities, they are tied together in terms of quantities much more relevant to the shape of the orbit itself. A classical ephemeris vector looks as follows:

$$\mathbf{v} = \begin{bmatrix} a \\ e \\ i \\ \Omega \\ \omega \\ \nu \end{bmatrix}$$

The quantities in the vector are defined as follows:

- a is the semimajor axis of the (presumed elliptical) orbit
- e is the eccentricity of the orbit
- i is the inclination of the orbit with respect to a reference plane (usually the equatorial plane of the Earth)
- Ω is the longitude of the ascending node (the angle between a predefined direction vector $\mathbf{i}$ and the ascending node, the point at which the satellite's orbit rises from South to North above the orbital plane, measured in the reference plane)
- ω is the argument of perigee (the angle between the ascending node and periapsis, the closest point of the orbit to Earth, measured in the orbital plane)
- ν is the true anomaly (the angle between periapsis and the current position of the satellite, measured in the orbital plane)[8]

Classical elements make it very easy to visualize an orbit for back-of-the-envelope calculations and, importantly, have only one fast variable; unlike with Cartesian coordinates, only the true anomaly changes over time, while the other five parameters remain fixed. (Note that this ignores anomalous gravitational effects such as perturbation and nutation that cause the orbit itself to move over time; for the remainder of the paper, simplified two-body motion is assumed. This assumption is valid over short periods since orbital precession is a long-term process, and no long-term propagations requiring a more advanced propagation model will be discussed here.)

[8]Mean anomaly M and eccentric anomaly E are also sometimes used for these calculations and have slightly different formulations.

However, despite how cleanly classical elements work with satellite orbits, they suffer from severe singularity issues. For a perfectly circular orbit, for instance, there is no closest point to the Earth, so ν and ω are both undefined (since they rely on a measurement of periapsis), while for an orbit with zero inclination (an equatorial orbit), Ω and ω are both undefined (since there is no ascending node). Most severely, for equatorial circular orbits, Ω, ω, and ν (fully half of the elements) are undefined. This is undesirable for obvious reasons, and while such edge cases are rare, they are not unheard of; while there are corrections that can be made to classical elements to handle such cases, these corrections are not universally applied, and not all algorithms converting data to and from classical elements are guaranteed to support them (and near-singular orbits may be highly numerically unstable when performing calculations).

Appendix II: Formulation of Equinoctial Elements Using Classical Elements

The formulation of equinoctial elements in terms of classical elements is much simpler than that using Cartesian elements. These quantities are defined as follows (in terms of classical elements):

- $n = \sqrt{\frac{\mu}{a^3}}$
- $a_f = e\cos(\omega + f_r\Omega)$
- $a_g = e\cos(\omega + f_r\Omega)$
- $\chi = \tan\left(\frac{i}{2}\right)^{f_r}\sin(\Omega)$
- $\psi = \tan\left(\frac{i}{2}\right)^{f_r}\cos(\Omega)$
- $\lambda_M = f_r\Omega + \omega + M$[9]

[9]This quantity depends on the mean anomaly; a separate formulation, $\lambda_\nu = f_r\Omega + \omega + \nu$, uses true anomaly instead.

Appendix III: Jacobian Matrix for Transforming Cartesian Covariance Matrices to Equinoctial Covariance Matrices

The Jacobian Matrix for the Cartesian to equinoctial covariance matrix transformation is organized as follows:

$$J = \begin{bmatrix} \hat{R}_n^{\mathrm{T}} \\ \hat{R}_{a_f}^{\mathrm{T}} \\ \hat{R}_{a_g}^{\mathrm{T}} \\ \hat{R}_{\chi}^{\mathrm{T}} \\ \hat{R}_{\psi}^{\mathrm{T}} \\ \hat{R}_{\lambda_M}^{\mathrm{T}} \end{bmatrix}$$

Where the $\hat{R}$ quantities are defined as follows:

$$\hat{R}_n = \begin{bmatrix} \frac{-3nar_x}{r^3} \\ \frac{-3nar_y}{r^3} \\ \frac{-3nar_z}{r^3} \\ \frac{-3v_x}{na^2} \\ \frac{-3v_y}{na^2} \\ \frac{-3v_z}{na^2} \end{bmatrix} \hat{R}_{a_f} = \begin{bmatrix} -\frac{aba_fBr_x}{r^3} - \frac{a_g(\chi\dot{X}-f_r\psi\dot{Y})w_e}{AB} - \frac{B}{A}\frac{\partial v_x}{\partial a_g} \\ -\frac{aba_fBr_y}{r^3} - \frac{a_g(\chi\dot{X}-f_r\psi\dot{Y})w_q}{AB} - \frac{B}{A}\frac{\partial v_y}{\partial a_g} \\ -\frac{aba_fBr_z}{r^3} - \frac{a_g(\chi\dot{X}-f_r\psi\dot{Y})w_w}{AB} - \frac{B}{A}\frac{\partial v_z}{\partial a_g} \\ \frac{(2X\dot{Y}-\dot{X}Y)g_e-Y\dot{Y}f_e}{\mu} - \frac{a_g(f_r\psi Y-\chi X)w_e}{AB} \\ \frac{(2X\dot{Y}-\dot{X}Y)g_q-Y\dot{Y}f_q}{\mu} - \frac{a_g(f_r\psi Y-\chi X)w_q}{AB} \\ \frac{(2X\dot{Y}-\dot{X}Y)g_w-Y\dot{Y}f_w}{\mu} - \frac{a_g(f_r\psi Y-\chi X)w_w}{AB} \end{bmatrix}$$

$$\hat{R}_{a_g} = \begin{bmatrix} -\frac{aba_gBr_x}{r^3} - \frac{a_f(\chi\dot{X}-f_r\psi\dot{Y})w_e}{AB} - \frac{B}{A}\frac{\partial v_x}{\partial a_f} \\ -\frac{aba_gBr_y}{r^3} - \frac{a_f(\chi\dot{X}-f_r\psi\dot{Y})w_q}{AB} - \frac{B}{A}\frac{\partial v_y}{\partial a_f} \\ -\frac{aba_gBr_z}{r^3} - \frac{a_f(\chi\dot{X}-f_r\psi\dot{Y})w_w}{AB} - \frac{B}{A}\frac{\partial v_z}{\partial a_f} \\ \frac{(2\dot{X}Y-X\dot{Y})f_e-X\dot{X}g_e}{\mu} - \frac{a_f(f_r\psi Y-\chi X)w_e}{AB} \\ \frac{(2\dot{X}Y-X\dot{Y})f_q-X\dot{X}g_q}{\mu} - \frac{a_f(f_r\psi Y-\chi X)w_q}{AB} \\ \frac{(2\dot{X}Y-X\dot{Y})f_w-X\dot{X}g_w}{\mu} - \frac{a_f(f_r\psi Y-\chi X)w_w}{AB} \end{bmatrix} \hat{R}_{\chi} = \begin{bmatrix} -\frac{C\dot{Y}w_e}{2AB} \\ -\frac{C\dot{Y}w_q}{2AB} \\ \frac{C\dot{Y}w_w}{2AB} \\ \frac{C\dot{Y}w_e}{2AB} \\ \frac{C\dot{Y}w_q}{2AB} \\ \frac{C\dot{Y}w_w}{2AB} \end{bmatrix}$$

$$\hat{R}_{\psi} = \begin{bmatrix} -\frac{Cf_r\dot{X}w_e}{2AB} \\ -\frac{Cf_r\dot{X}w_q}{2AB} \\ -\frac{Cf_r\dot{X}w_w}{2AB} \\ \frac{Cf_r\dot{X}w_e}{2AB} \\ \frac{Cf_r\dot{X}w_q}{2AB} \\ \frac{Cf_r\dot{X}w_w}{2AB} \end{bmatrix} \hat{R}_{\lambda_M} = \begin{bmatrix} -\frac{v_x}{A} + \frac{(\chi\dot{X}-f_r\psi\dot{Y})w_e}{AB} - \frac{bB}{A}(a_g\frac{\partial v_x}{\partial a_g} + a_f\frac{\partial v_x}{\partial a_f}) \\ -\frac{v_y}{A} + \frac{(\chi\dot{X}-f_r\psi\dot{Y})w_q}{AB} - \frac{bB}{A}(a_g\frac{\partial v_y}{\partial a_g} + a_f\frac{\partial v_y}{\partial a_f}) \\ -\frac{v_z}{A} + \frac{(\chi\dot{X}-f_r\psi\dot{Y})w_w}{AB} - \frac{bB}{A}(a_g\frac{\partial v_z}{\partial a_g} + a_f\frac{\partial v_z}{\partial a_f}) \\ -\frac{2r_x}{A} + \frac{a_f\frac{\partial a_g}{\partial v_x}-a_g\frac{\partial a_f}{\partial v_x}}{1+B} + \frac{(f_r\psi Y-\chi X)w_e}{A} \\ -\frac{2r_y}{A} + \frac{a_f\frac{\partial a_g}{\partial v_y}-a_g\frac{\partial a_f}{\partial v_y}}{1+B} + \frac{(f_r\psi Y-\chi X)w_q}{A} \\ -\frac{2r_z}{A} + \frac{a_f\frac{\partial a_g}{\partial v_z}-a_g\frac{\partial a_f}{\partial v_z}}{1+B} + \frac{(f_r\psi Y-\chi X)w_w}{A} \end{bmatrix}$$

Appendix IV: Cartesian Covariance Matrices For CARA Test Cases

The Cartesian covariance matrices for the CARA test cases referenced in the Examples and Numerical Results section of this paper are as follows:

$C_1 =$

$$\begin{bmatrix} +4.976545641899520 \times 10^{4} & +5.787130862568278 \times 10^{4} & +3.370410320935015 \times 10^{3} & \cdot \\ +5.787130862568278 \times 10^{4} & +6.730377643610841 \times 10^{4} & +3.926542932121541 \times 10^{3} & \cdot \\ +3.370410320935015 \times 10^{3} & +3.926542932121541 \times 10^{3} & +2.461403197221289 \times 10^{2} & \cdot \\ +1.137272273949272 \times 10^{1} & +1.321992688238858 \times 10^{1} & +7.586865834476763 \times 10^{-1} & \cdot \\ -4.325472616114674 \times 10^{0} & -5.035560720747812 \times 10^{0} & -3.077848629905763 \times 10^{-1} & \cdot \\ -8.009705480233521 \times 10^{1} & -9.314985106902773 \times 10^{1} & -5.434034460756914 \times 10^{0} & \cdot \end{bmatrix}$$

$$\begin{bmatrix} +1.137272273949272 \times 10^{1} & -4.325472616114674 \times 10^{0} & -8.009705480233521 \times 10^{1} \\ +1.321992688238858 \times 10^{1} & -5.035560720747812 \times 10^{0} & -9.314985106902773 \times 10^{1} \\ +7.586865834476763 \times 10^{-1} & -3.077848629905763 \times 10^{-1} & -5.434034460756914 \times 10^{0} \\ +2.608186227148725 \times 10^{-3} & -9.804181796720670 \times 10^{-4} & -1.829751672999786 \times 10^{-2} \\ -9.804181796720670 \times 10^{-4} & +3.895883508545853 \times 10^{-4} & +6.968892326415779 \times 10^{-3} \\ -1.829751672999786 \times 10^{-2} & +6.968892326415779 \times 10^{-3} & +1.289253320300791 \times 10^{-1} \end{bmatrix}$$

$C_2 =$

$$\begin{bmatrix} 4.246862551076427 \times 10^{4} & +2.066374367781032 \times 10^{5} & -5.011108933888592 \times 10^{3} \\ +2.066374367781032 \times 10^{5} & +1.005854717283451 \times 10^{6} & -2.434876491048039 \times 10^{4} \\ -5.011108933888592 \times 10^{3} & -2.434876491048039 \times 10^{4} & +6.131274993037449 \times 10^{2} \\ +3.104606531932427 \times 10^{1} & +1.510022508670080 \times 10^{2} & -3.667147183233717 \times 10^{0} \\ -1.201093683199582 \times 10^{1} & -5.850063541467530 \times 10^{1} & +1.391769957262238 \times 10^{0} \\ -2.207975848324051 \times 10^{2} & -1.074752763805685 \times 10^{3} & +2.601457791444154 \times 10^{1} \end{bmatrix}$$

$$\begin{bmatrix} +3.104606531932427 \times 10^{1} & -1.201093683199582 \times 10^{1} & -2.207975848324051 \times 10^{2} \\ +1.510022508670080 \times 10^{2} & -5.850063541467530 \times 10^{1} & -1.074752763805685 \times 10^{3} \\ -3.667147183233717 \times 10^{0} & +1.391769957262238 \times 10^{0} & +2.601457791444154 \times 10^{1} \\ +2.272826228568773 \times 10^{-2} & -8.778253314778023 \times 10^{-3} & -1.613538091053610 \times 10^{-1} \\ -8.778253314778023 \times 10^{-3} & +3.428801115804722 \times 10^{-3} & +6.251148178133809 \times 10^{-2} \\ -1.613538091053610 \times 10^{-1} & +6.251148178133809 \times 10^{-2} & +1.148404222181769 \times 10^{0} \end{bmatrix}$$

Appendix V: Covariance Matrices After Transformation, Equinoctial

The equinoctial covariance matrices for the CARA test cases referenced in the Examples and Numerical Results section of this paper are as follows:

$E_1 =$

$$\begin{bmatrix} 1.82691451588697 \times 10^{-10} & -1.57604549247866 \times 10^{-10} & -2.23940854560202 \times 10^{-9} \\ -1.57604549248092 \times 10^{-10} & 2.3990304781749 \times 10^{-10} & -2.44566861318093 \times 10^{-9} \\ -2.23940854560232 \times 10^{-9} & -2.44566861318042 \times 10^{-9} & 2.33876898033084 \times 10^{-6} \\ -1.72732657080658 \times 10^{-14} & -1.89483110921661 \times 10^{-14} & 1.74325762110509 \times 10^{-11} \\ 2.74294020098394 \times 10^{-11} & -3.49174615666343 \times 10^{-11} & -8.09294516060298 \times 10^{-10} \\ 2.39823583561893 \times 10^{-11} & -2.00347954780789 \times 10^{-11} & 1.45591076629607 \times 10^{-9} \end{bmatrix}$$

$$\begin{bmatrix} -1.72732657078784 \times 10^{-14} & 2.74294020098865 \times 10^{-11} & 2.3982358356352 \times 10^{-11} \\ -1.89483110921126 \times 10^{-14} & -3.49174615666206 \times 10^{-11} & -2.00347954779788 \times 10^{-11} \\ 1.74325762110509 \times 10^{-11} & -8.09294516060607 \times 10^{-10} & 1.45591076629611 \times 10^{-9} \\ 1.30262052001149 \times 10^{-16} & -5.62958680792519 \times 10^{-15} & 1.04150536362303 \times 10^{-14} \\ -5.62958680794891 \times 10^{-15} & 8.23847142526637 \times 10^{-11} & -1.87620673479503 \times 10^{-11} \\ 1.04150536361471 \times 10^{-14} & -1.87620673479836 \times 10^{-11} & 4.69716328705626 \times 10^{-11} \end{bmatrix}$$

$E_2 =$

$$\begin{bmatrix} 5.00977258091492 \times 10^{-10} & -3.43636052636814 \times 10^{-11} & -1.15038928094219 \times 10^{-8} \\ -3.43636052625379 \times 10^{-11} & 6.42675743373135 \times 10^{-10} & -1.49627779737201 \times 10^{-8} \\ -1.15038928094206 \times 10^{-8} & -1.49627779737258 \times 10^{-8} & 2.0509566451034 \times 10^{-5} \\ -8.02492264892092 \times 10^{-14} & -1.05623258722769 \times 10^{-13} & 1.42217259548482 \times 10^{-10} \\ 2.25903246001087 \times 10^{-10} & 1.02247829619751 \times 10^{-10} & -8.25609559641959 \times 10^{-10} \\ -2.01512819327028 \times 10^{-10} & -8.80341867735264 \times 10^{-11} & 1.54610356361475 \times 10^{-8} \end{bmatrix}$$

$$\begin{bmatrix} -8.02492264891853 \times 10^{-14} & 2.25903246001124 \times 10^{-10} & -2.01512819327292 \times 10^{-10} \\ -1.0562325872176 \times 10^{-13} & 1.02247829621862 \times 10^{-10} & -8.80341867738424 \times 10^{-11} \\ 1.42217259548482 \times 10^{-10} & -8.25609559640839 \times 10^{-10} & 1.54610356361471 \times 10^{-8} \\ 9.87316677055351 \times 10^{-16} & -1.23282174264769 \times 10^{-15} & 9.9775293167244 \times 10^{-14} \\ -1.23282174261936 \times 10^{-15} & 6.12976802726819 \times 10^{-10} & -6.37254978719447 \times 10^{-10} \\ 9.97752931671921 \times 10^{-14} & -6.37254978719254 \times 10^{-10} & 1.09789636164715 \times 10^{-9} \end{bmatrix}$$

Appendix VI: MATLAB code

The following code was used in conjunction with NASA CARA's open source code, which can be found on their public GitHub at `https://github.com/nasa/CARA_Analysis_Tools`.

main.m

```
clearvars;
clc;
addpath(genpath(pwd));

r1   = [-9.841950433215101e+05 +3.932342044549424e+05
   +6.991223682230414e+06];
v1   = [+4.883696742000000e+03 +5.689086045000000e+03
   +3.665361590000000e+02];
cov1 = [+4.976545641899520e+04 +5.787130862568278e+04
   +3.370410320935015e+03 +1.137272273949272e+01
   -4.325472616114674e+00 -8.009705480233521e+01; ...
    +5.787130862568278e+04 +6.730377643610841e+04
       +3.926542932121541e+03 +1.321992688238858e+01
       -5.035560720747812e+00 -9.314985106902773e+01; ...
    +3.370410320935015e+03 +3.926542932121541e+03
       +2.461403197221289e+02 +7.586865834476763e-01
       -3.077848629905763e-01 -5.434034460756914e+00; ...
    +1.137272273949272e+01 +1.321992688238858e+01
       +7.586865834476763e-01 +2.608186227148725e-03
       -9.804181796720670e-04 -1.829751672999786e-02; ...
    -4.325472616114674e+00 -5.035560720747812e+00
       -3.077848629905763e-01 -9.804181796720670e-04
       +3.895883508545853e-04 +6.968892326415779e-03; ...
    -8.009705480233521e+01 -9.314985106902773e+01
       -5.434034460756914e+00 -1.829751672999786e-02
       +6.968892326415779e-03 +1.289253320300791e-01];

r2   = [-9.839696058965517e+05 +3.936845951174244e+05
   +6.991219291625473e+06];
v2   = [+1.509562687000000e+03 +7.372938617000000e+03
   -1.492509430000000e+02];
cov2 = [+4.246862551076427e+04 +2.066374367781032e+05
   -5.011108933888592e+03 +3.104606531932427e+01
   -1.201093683199582e+01 -2.207975848324051e+02; ...
    +2.066374367781032e+05 +1.005854717283451e+06
       -2.434876491048039e+04 +1.510022508670080e+02
       -5.850063541467530e+01 -1.074752763805685e+03; ...
```

```
    -5.011108933888592e+03 -2.434876491048039e+04
       +6.131274993037449e+02 -3.667147183233717e+00
       +1.391769957262238e+00 +2.601457791444154e+01; ...
    +3.104606531932427e+01 +1.510022508670080e+02
       -3.667147183233717e+00 +2.272826228568773e-02
       -8.778253314778023e-03 -1.613538091053610e-01; ...
    -1.201093683199582e+01 -5.850063541467530e+01
       +1.391769957262238e+00 -8.778253314778023e-03
       +3.428801115804722e-03 +6.251148178133809e-02; ...
    -2.207975848324051e+02 -1.074752763805685e+03
       +2.601457791444154e+01 -1.613538091053610e-01
       +6.251148178133809e-02 +1.148404222181769e+00];

r1 = r1 / 1000;
v1 = v1 / 1000;
cov1 = cov1 / 1000;

r2 = r2 / 1000;
v2 = v2 / 1000;
cov2 = cov2 / 1000;

[x1, y1, z1, ox1, oy1, oz1] = ProcessEphemeris(r1, v1, cov1);

[x2, y2, z2, ox2, oy2, oz2] = ProcessEphemeris(r2, v2, cov2,
   10000);

baddist = mvnrnd([r2, v2], cov2, 10000);

x3 = baddist(:, 1);
y3 = baddist(:, 2);
z3 = baddist(:, 3);

xrad = (max(x2) - min(x2)) / 2;
yrad = (max(y2) - min(y2)) / 2;
zrad = (max(z2) - min(z2)) / 2;

I = imread('earth.png');
[ex, ey, ez] = sphere(100);
r = 6378.1;

figure (1);
hold on;
grid on;
axis equal;
```

```
scatter3(x1, y1, z1, 'MarkerEdgeColor', 'b', 'MarkerFaceColor',
    'b');
plot3(ox1, oy1, oz1, "r", "LineWidth", 2.5);

warp(-ex * r, ey * r, -ez * r, I);

xlabel("Distance From Center of the Earth (km)");
ylabel("Distance From Center of the Earth (km)");
zlabel("Distance From Center of the Earth (km)");

title("Satellite Orbit Around Earth");

legend("Satellite Ephemeris Probability Distribution", "
   Satellite Orbit", "Location", "north");

set(gca, "FontName", "Times New Roman", "FontSize", 20);
set(gca, "LineWidth", 1);

figure (2);
hold on;
grid on;

scatter3(x1, y1, z1, 'MarkerEdgeColor', 'b', 'MarkerFaceColor',
    'b');

xlabel("Distance From Center of the Earth (km)");
ylabel("Distance From Center of the Earth (km)");
zlabel("Distance From Center of the Earth (km)");

title("Satellite Ephemeris Probability Distribution");

view(-25, 25);

set(gca, "FontName", "Times New Roman", "FontSize", 20);
set(gca, "LineWidth", 1);

figure (3);
hold on;
grid on;
axis equal;

scatter3(x2(1:1000), y2(1:1000), z2(1:1000), 'MarkerEdgeColor',
    'b', 'MarkerFaceColor', 'b');
plot3(ox2, oy2, oz2, "r", "LineWidth", 2.5);
```

```
warp(-ex * r, ey * r, -ez * r, I);

xlabel("Distance From Center of the Earth (km)");
ylabel("Distance From Center of the Earth (km)");
zlabel("Distance From Center of the Earth (km)");

title("Satellite Orbit Around Earth");

legend("Satellite Ephemeris Probability Distribution", "
   Satellite Orbit", "Location", "north");

set(gca, "FontName", "Times New Roman", "FontSize", 20);
set(gca, "LineWidth", 1);

figure (4);
hold on;
grid on;

scatter3(x2(1:1000), y2(1:1000), z2(1:1000), 'MarkerEdgeColor',
    'b', 'MarkerFaceColor', 'b');

xlabel("Distance From Center of the Earth (km)");
ylabel("Distance From Center of the Earth (km)");
zlabel("Distance From Center of the Earth (km)");

title("Satellite Ephemeris Probability Distribution");

view(-25, 25);

set(gca, "FontName", "Times New Roman", "FontSize", 20);
set(gca, "LineWidth", 1);

figure (5);
hold on;
grid on;

scatter3(x2, y2, z2, 2, 'MarkerEdgeColor', 'b', '
   MarkerFaceColor', 'b');
scatter3(x3, y3, z3, 2, 'MarkerEdgeColor', 'r', '
   MarkerFaceColor', 'r');

plot3(ox2, oy2, oz2, "g", "LineWidth", 2.5);

xlabel("Distance From Center of the Earth (km)");
ylabel("Distance From Center of the Earth (km)");
```

```
zlabel("Distance From Center of the Earth (km)");

xlim([mean(x2) - xrad, mean(x2) + xrad]);
ylim([mean(y2) - yrad, mean(y2) + yrad]);
zlim([mean(z2) - zrad, mean(z2) + zrad]);

title("Satellite Ephemeris Probability Distribution Comparisons
   ");

legend("Equinoctial Probability Distribution", "Cartesian
   Probability Distribution", "Satellite Orbit", "Location", "
   north");

view(-25, 25);

set(gca, "FontName", "Times New Roman", "FontSize", 20);
set(gca, "LineWidth", 1);
```

ProcessEphemeris.m

```
function [x, y, z, ox, oy, oz] = ProcessEphemeris(r, v, cov,
   runs, steps)
    if nargin < 4
        runs = 1000;
    end

    if nargin < 5
        steps = 10000;
    end

    [~, n, af, ag, chi, psi, lM] =
       convert_cartesian_to_equinoctial(r, v);
    eqeph = [af, ag, lM, n, chi, psi];
    eqcov = ECI2EQN(cov, r, v);

    orbit = PropagateOrbit(eqeph, steps);

    ox = orbit(:, 1);
    oy = orbit(:, 2);
    oz = orbit(:, 3);

    eqmat = mvnrnd(eqeph, eqcov, runs);

    len = size(eqmat, 1);
```

```
    cartmat = zeros(len, 3);

    n = eqmat(:, 4);
    af = eqmat(:, 1);
    ag = eqmat(:, 2);
    chi = eqmat(:, 5);
    psi = eqmat(:, 6);
    lM = eqmat(:, 3);

    for i = 1:len
        cartmat(i, :) = convert_equinoctial_to_cartesian(n(i),
           af(i), ag(i), chi(i), psi(i), lM(i), 0)';
    end

    x = cartmat(:, 1);
    y = cartmat(:, 2);
    z = cartmat(:, 3);
end
```

PropagateOrbit.m

```
function orbit = PropagateOrbit(eqeph, steps)
    orbit = zeros(steps, 3);

    lM = linspace(0, 2 * pi, steps);

    n = eqeph(4);
    af = eqeph(1);
    ag = eqeph(2);
    chi = eqeph(5);
    psi = eqeph(6);

    for i = 1:steps
        orbit(i, :) = convert_equinoctial_to_cartesian(n, af,
           ag, chi, psi, lM(i), 0)';
    end
end
```