A MONTE CARLO TOOL FOR SIMULATION OF SATELLITE ORBIT DETERMINATION BY RADIO INTERFEROMETRY

by

Michael S. Pavloff

A.B., Physics
Harvard University, 1988

Submitted to the Department of Aeronautics and Astronautics in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

at the

Massachusetts Institute of Technology

September 1993

Signature of Author___ _____

Department of Aeronautics and Astronautics
July 31, 1993

Certified by ___ _____

Professor Richard H. Battin
Thesis Supervisor

Accepted by___ _____

Professor Harold Y. Wachman
Chairman, Department Graduate Committee

A MONTE CARLO TOOL FOR SIMULATION OF SATELLITE ORBIT DETERMINATION BY RADIO INTERFEROMETRY

by

Michael S. Pavloff

Submitted to the Department of Aeronautics and Astronautics in partial fulfillment of the requirements for the Degree of Master of Science

ABSTRACT

Radio interferometry offers a novel approach to satellite orbit determination. The high level of precision from the group or phase delay observables that translate into a differential range measurement between tracking stations makes interferometry an attractive technique for satellite tracking. This study summarizes the geometry of the group and phase delay observables and presents a Monte Carlo simulation tool for assessing the accuracy of various orbit determination scenarios. The Interferometric Satellite Orbit Determination Accuracy Estimator (ISODAE) models the process of batch satellite state vector estimation from a potentially overdetermining set of measurements taken over time with error injected due to inherent observable imprecision. State vector accuracies for various orbits and station location geometries are presented.

Thesis Supervisor: Professor Richard H. Battin

Title: Adjunct Professor of Aeronautics and Astronautics

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SECTION 1

INTRODUCTION

1.1 SATELLITE ORBIT DETERMINATION BY RADIO INTERFEROMETRY

Radio interferometry is a novel approach to satellite orbit determination, but it is an approach well worth consideration for several reasons. First, the inherent level of precision of the differential range measurement that results from interferometry is greater than that available from conventional range measurements by radar. Second, radio interferometry may be carried out on any signal emanating from the satellite; neither a special transponder nor time on a communications link are required. Radio interferometry does not require a coherent signal to be carried out; radio astronomers make measurements on quasars, whose radio signal structure is essentially random noise. Third, extremely short baselines between ground stations are feasible, thus ameliorating geographical and political constraints on measuring station siting.

To date, experiments have been carried out in academia and through NASA research and development funding to demonstrate the feasibility of radio interferometry for satellite orbit determination, but it is far from the point of being a widely used method for operational satellite tracking. In this study, we hope to demonstrate the utility of interferometry for orbit determination by showing how the high precision of the differential range measurement that it produces translates into high satellite ephemeris prediction accuracy.

It was not until the late 1960's that radio interferometry was first applied to the problem of satellite orbit determination. In an experiment devised by Irwin Shapiro, Alan Whitney, and others, very long baseline interferometric (VLBI) measurements were made on the TACSAT I communications satellite in geosynchronous orbit (GEO), and the semi-major axis of the orbit was measured with accuracy on the order of several hundred meters [1]. Subsequent experiments were performed in the 1980's by Jim Ray, Curt Knight, and others to determine the position of the Tracking and Data Relay Satellite (TDRS) [2]. The achieved accuracy, estimated to be on the order of 75 meters [2], was encouraging. As discussed in Section 3.1, the theoretically achievable level of precision with the differential range measurement resulting from an interferometric group delay or phase delay observable is extremely high. In short, radio interferometry provides an attractive alternative to traditional means of satellite orbit determination.

Current orbit determination programs, such as the Goddard Trajectory Determination System (GTDS) [3] and GEODYN [4] do not allow the differential range measurement type. The derivation of the differential range measurement from the group delay observable, as shown in Section 2.3, was developed independently for this study. However, the batch orbit determination algorithm presented in Section 2.1 is essentially that used in GTDS and GEODYN.

For this study, a Monte Carlo simulation tool was built around an orbit determination algorithm in order to be able to inject error in the differential range measurements provided by radio interferometry. The user can specify a satellite orbit, any set of ground stations between which differential range measurements (or other types of measurements) are to be made, and the statistical properties of the error in those measurements. Upon

each iteration of the Monte Carlo simulation, the orbit of the satellite is determined based on measurements with errors, and the errors in the resulting satellite ephemerides are recorded. Thus, the user may study the statistical properties of the error in the batch orbit determination process resulting from the use of differential range measurements.

1.2 MONTE CARLO SIMULATION FOR ACCURACY ASSESSMENT

This document presents the theory behind a tool developed for the assessment of the error in satellite orbit determination process with the differential range observable. The Monte Carlo simulation that was developed is referred to as the Interferometric Satellite Orbit Determination Accuracy Estimator (ISODAE). ISODAE allows the user to specify the satellite orbit (state vector at epoch), ground station locations, times of observation, and statistical properties of observable errors. Upon each iteration of the simulation, the orbit determination process is carried out with random error of the user-specified characteristics injected into the observations. The error in the satellite state vector at the specified epoch is calculated as the difference between the true satellite state vector specified by the user as part of the ISODAE input and the state vector at epoch calculated in the orbit determination process with error in the measurements. After the simulation, it is then possible to study the statistical properties of the error in the estimated state vectors.

For flexibility and ease of development, ISODAE was implemented in Mathematica on the Macintosh [5]. This document assumes the user is familiar with Mathematica and with the fundamental principles of astrodynamics. Sufficient comments are furnished

within ISODAE so that the user should be able to guide himself through the process of setting up a sample orbit determination scenario and executing the Monte Carlo simulation.

The orbit determination process that ISODAE uses is similar to that employed by such state-of-the-art orbit determination programs as GEODYN and GTDS. Specifically, ISODAE employs the batch weighted least squares estimation process, described in Section 2.1, in which all measurements are used simultaneously to determine the satellite orbit. (With Kalman filtering, another frequently used orbit determination process, measurements taken at later times are used sequentially to update the estimated satellite orbit.) The current implementation of ISODAE uses only two-body trajectory propagation and state transition matrix for reasons of computational expedience. However, the purpose of ISODAE is to study the statistical properties of errors resulting from differential range measurements, and consequently, the orbit determination problem need not be solved precisely. The user is not interested in the actual veracity of the values for satellite ephemerides genereated by ISODAE; instead, he is interested in the effects of errors in differential range upon those ephemerides. In this document, all the mathematical specifications for more accurate trajectory propagation are provided, but they are not currently implemented in the Mathematica model.

1.3 OVERVIEW OF DOCUMENT

This document serves as the mathematical specification for ISODAE. Since the Mathematica procedures are themselves amply commented, this document is not

expected to serve as a detailed user's guide. Section 2 summarizes the orbit determination process, including the least-squares batch estimator, time and coordinate systems, models of trajectory propagation, and methods of numerical integration. In Section 2, the geometry of the group delay and phase delay observables from radio interferometry are presented. Section 3 summarizes error modeling and the Monte Carlo simulation process. Section 4 presents results of some sample applications of ISODAE to various satellite orbits and provides conclusions about optimal sensor site location geometries. While the purpose of ISODAE is to study the properties of the differential range measurement type used in satellite orbit determination, differential range may be combined with other measurement types in an actual satellite tracking scenario. Appendix A presents the measurement and geometrical partial derivatives for other common measurement types. Finally, Appendix B reproduces the Mathematica procedures in which ISODAE is implemented.

SECTION 2

THE ORBIT DETERMINATION PROCESS

The process of satellite orbit determination involves measuring some physical property of electromagnetic wave propagation between a satellite and a set of observing stations. The measured property, called an observable, is then translated into a geometrical measurement, such as range, range rate, or differential range. This study concentrates on differential range measurements derived from group delay or phase delay observables from an interferometer, as described in detail in Section 2.2. It is assumed that the process of generating a differential range measurement, which includes signal cross-correlation, resolution of cycle ambiguities, and estimation of signal propagation rates, has already been accomplished, and thus the differential range measurement is itself available. ISODAE also includes the capability for the user to specify measurement types other than differential range, but the focus of this study is restricted to differential range.

2.1 LEAST-SQUARES BATCH ESTIMATOR

In one commonly-used approach for artificial satellite trajectory analysis, orbit determination is carried out by fitting the best (in a least-squares sense) orbit to a set of weighted measurements in batch. In the formulation of this approach, it is desired to estimate a satellite state vector $\mathbf{x}$ at some reference time t_0. The state vector is taken to contain m components x_i, $i = 1, ..., m$, which are some subset of the satellite's position and velocity vector components measured in an earth-centered inertial (ECI) coordinate frame, plus any additional parameters, such as clock offsets or atmospheric

characteristics, not concerned with satellite dynamics. It is assumed that p measurements, y_i, $i = 1, ..., p$, of the differential range type (or potentially any other type) are given in the measurement vector $\mathbf{y}$. For each measurement y_i, there is an associated time tag t_i. In summary, the state vector to be estimated at reference time t_0 and the measurement vector, respectively, are written as follows:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{bmatrix}$$

The functional relationships between the state vector $\mathbf{x}$ and expected measurements y_i are known and are denoted $f_i(\mathbf{x}) = y_i$. The exact form of f_i for a differential range measurement will be presented in Section 2.3.1, and the forms for other measurement types will be presented in Appendix A. The measurement equations may thus be written $\mathbf{y} = \mathbf{f}(\mathbf{x})$, where

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_p(\mathbf{x}) \end{bmatrix}$$

The p measurements must be sufficient at least to determine the m state vector elements. However, it is generally advantageous for the measurements to overdetermine $\mathbf{x}$ so that the effects of measurement errors may be reduced. In either case, the least-squares batch estimator finds the state vector $\mathbf{x}$ that minimizes the squared errors between the measurements $\mathbf{y}$ and the expected expected measurements $\mathbf{f}(\mathbf{x})$.

2.1.1 The Batch Estimator for the Static Problem

First consider the problem where the p measurements are all made at the same time. If it is desired to weight the i^{th} measurement by the weighting factor w_i, then the diagonal weighting matrix $\mathbf{W}$ on the measurements will have $[\mathbf{W}]_{i,i} = w_i$. We construct the following quadratic form scalar loss function between a state vector $\mathbf{x}$ and the measurement vector $\mathbf{y}$.

$$Q(\mathbf{x}) = [\mathbf{y} - \mathbf{f}(\mathbf{x})]^{\mathrm{T}} \mathbf{W} [\mathbf{y} - \mathbf{f}(\mathbf{x})] \tag{1}$$

Minimizing the scalar loss function (1) with respect to $\mathbf{x}$ results in the Gaussian least squares regression equation.

$$\frac{\partial Q}{\partial \mathbf{x}} = -2[\mathbf{y} - \mathbf{f}(\mathbf{x})]^{\mathrm{T}} \mathbf{W} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \mathbf{0}$$

$$\Rightarrow \quad [\mathbf{y} - \mathbf{f}(\mathbf{x})]^{\mathrm{T}} \mathbf{W} \mathbf{F}(\mathbf{x}) = \mathbf{0} \tag{2}$$

In equation (2), $\mathbf{F}(\mathbf{x})$ is the $p \times m$ Jacobian matrix:

$$\mathbf{F}(\mathbf{x}) \underset{\text{def}}{=} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \partial f_1 / \partial x_1 & \partial f_1 / \partial x_2 & \cdots & \partial f_1 / \partial x_m \\ \partial f_2 / \partial x_1 & \partial f_2 / \partial x_2 & \cdots & \partial f_2 / \partial x_m \\ \vdots & \vdots & \ddots & \vdots \\ \partial f_p / \partial x_1 & \partial f_p / \partial x_2 & \cdots & \partial f_p / \partial x_m \end{bmatrix} \tag{3}$$

The exact functional forms for the geometrical partial derivatives in the Jacobian (3) are given in Section 2.3.2 and Appendix A, respectively, for differential range and for other measurement types.

Equation (2) is the matrix representation of a system of *m* non-linear equations in the *m* unknowns of **x**. Analytical solution of equation (2) is impossible for the case of differential range measurements, and so solution will be carried out numerically by Newton-Raphson iteration. To implement a Newton-Raphson iteration procedure, we find a Taylor series expansion of **f**(**x**) around some *a priori* estimate of the state vector, $\mathbf{x}_0$, with $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0$:

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_0) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_0} \Delta\mathbf{x} + O(\Delta\mathbf{x}^2) \tag{4}$$

The second term on the right hand side of the Taylor series (4) may be rewritten as $\mathbf{F}(\mathbf{x}_0)\Delta\mathbf{x}$, and for notational purposes, we shall let we let $\mathbf{F}_0 = \mathbf{F}(\mathbf{x}_0)$. Also, let $\Delta\mathbf{y} = \mathbf{y} - \mathbf{f}(\mathbf{x}_0)$ be the difference between the true measurements **y** and what the measurements would be if the satellite's state vector were $\mathbf{x}_0$. We may find an approximation for the factor [**y** – **f**(**x**)] in equation (2) by substituting Taylor series (4) for **f**(**x**) as follows:

$$\mathbf{y} - \mathbf{f}(\mathbf{x}) = \mathbf{y} - \left[\mathbf{f}(\mathbf{x}_0) + \mathbf{F}(\mathbf{x}_0)\Delta\mathbf{x} + O(\Delta\mathbf{x}^2)\right] = \underbrace{\mathbf{y} - \mathbf{f}(\mathbf{x}_0)}_{\Delta\mathbf{y}} - \mathbf{F}_0\Delta\mathbf{x} + O(\Delta\mathbf{x}^2)$$

$$\Rightarrow \quad \mathbf{y} - \mathbf{f}(\mathbf{x}) \approx \Delta\mathbf{y} - \mathbf{F}_0\Delta\mathbf{x} \tag{5}$$

Then to first order, equation (2) may be rewritten with approximation (5):

$$[\mathbf{y}-\mathbf{f}(\mathbf{x})]^{\mathrm{T}}\mathbf{W}\mathbf{F}(\mathbf{x}) \approx [\Delta\mathbf{y}-\mathbf{F}_0\Delta\mathbf{x}]^{\mathrm{T}}\mathbf{W}\mathbf{F}_0 = \mathbf{0}$$

$$\Rightarrow \quad (\mathbf{F}_0\Delta\mathbf{x})^{\mathrm{T}}\mathbf{W}\mathbf{F}_0 = \Delta\mathbf{y}^{\mathrm{T}}\mathbf{W}\mathbf{F}_0 \quad \Rightarrow \quad \mathbf{F}_0^{\mathrm{T}}\mathbf{W}\mathbf{F}_0\Delta\mathbf{x} = \mathbf{F}_0^{\mathrm{T}}\mathbf{W}\Delta\mathbf{y}$$

The $m\times m$ information matrix is $\mathbf{S}_0 = \mathbf{F}_0^{\mathrm{T}}\mathbf{W}\mathbf{F}_0$, and so long as $\mathbf{S}_0$ is non-singular, its $m\times m$ inverse $\mathbf{P}_0 = \mathbf{S}_0^{-1}$ may be found. Thus, the correction $\Delta\mathbf{x}$ may be found to the initial satellite state vector estimate, $\mathbf{x}_0$:

$$\Delta\mathbf{x} = \mathbf{P}_0\mathbf{F}_0^{\mathrm{T}}\mathbf{W}\Delta\mathbf{y}$$

The Newton-Raphson iteration algorithm for finding the state vector $\mathbf{x}$ that best fits the observations $\mathbf{y}$ is summarized in Table 1. This algorithm converges fairly rapidly (usually in four steps with $|\Delta\mathbf{x}|$ small enough for most applications) and is fairly insensitive to relatively large errors in the initial estimate $\mathbf{x}_0$. Clearly, it is possible, even with overdetermining observations, to choose inadvertently an initial state estimate $\mathbf{x}_0$ that causes the information matrix to be singular or poorly-behaved. In such a situation, it is best to increment $\mathbf{x}_0$ by a small amount in a randomly selected direction.

As a final note, once the Newton-Raphson iteration has converged, the matrix $\mathbf{P}$ is the covariance matrix between the measurements and the state vector components. If, for example, $\mathbf{x} = [x \quad y \quad z]^{\mathrm{T}}$ is a satellite's state vector in Cartesian coordinates, then the

Table 1. Algorithm for Solving the Static Orbit Determination Problem

1. Start with the *a priori* state vector estimate $\mathbf{x}_0$, and let $i = 0$ be the initial iteration number. Choose a threshold δx for quitting the iteration.

2. Let $\Delta\mathbf{y} = \mathbf{y} - \mathbf{f}(\mathbf{x}_i)$ and compute the Jacobian matrix $\mathbf{F}_i = \mathbf{F}(\mathbf{x}_i)$.

3. Compute the information matrix $\mathbf{S}_i = \mathbf{F}_i^T\mathbf{W}\mathbf{F}_i$. If the information matrix is singular or ill-behaved, attempt the iteration again with a different *a priori* state vector estimate $\mathbf{x}_0$. If the iteration still fails to converge, it is likely that the measurements do not determine the full state vector, and the particular orbit determination scenario is degenerate. Otherwise, compute $\mathbf{P}_i = \mathbf{S}_i^{-1}$.

3. Calculate the correction to the state vector estimate: $\Delta\mathbf{x} = \mathbf{P}_i\mathbf{F}_i^T\mathbf{W}\Delta\mathbf{y}$.

4. Calculate the new state vector estimate: $\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x}$.

5. If $|\Delta\mathbf{x}| < \delta x$, then quit the iteration with the best state vector estimate $\mathbf{x} = \mathbf{x}_{i+1}$. Otherwise, let $i = i + 1$, and repeat the iteration at step 2.

covariance matrix will be of the following form:

$$\mathbf{P} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{xz}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 & \sigma_{yz}^2 \\ \sigma_{xz}^2 & \sigma_{yz}^2 & \sigma_{zz}^2 \end{bmatrix}$$

The position dilution of precision (PDOP), or position error, may be estimated as follows:

$$\text{PDOP} = \sqrt{\sigma_{xx}^2 + \sigma_{yy}^2 + \sigma_{zz}^2}$$

2.1.2 The Batch Estimator for the Dynamic Problem

Now consider the problem where the p measurements, y_1, y_2, ... , y_p, are made at times t_1, t_2, ... , t_p, possibly all different. Again, weight the i^{th} measurement by the weighting factor w_i, so that the diagonal weighting matrix **W** will have $[\mathbf{W}]_{i,i} = w_i$. Now each functional relationship t_f gives the expected measurement y_i if the satellite state vector were $\mathbf{x}(t_i)$ at time t_i, and so the vector **f** depends on the satellite state vector **x** at different times:

$$\mathbf{f} = \begin{bmatrix} f_1[\mathbf{x}(t_1)] \\ f_2[\mathbf{x}(t_2)] \\ \vdots \\ f_p[\mathbf{x}(t_p)] \end{bmatrix}$$

It is desired to find the m components of the state vector $\mathbf{x}$ at reference time t_0 that minimize the squared error between the expected and actual measurements. Thus, the following scalar loss function is created:

$$Q = [\mathbf{y} - \mathbf{f}]^{\mathrm{T}} \mathbf{W} [\mathbf{y} - \mathbf{f}] \tag{6}$$

Quadratic form (6) involves measurements and calculations at different times; for example, row i of the vector $\Delta\mathbf{y} = \mathbf{y} - \mathbf{f}$, for a particular $\mathbf{x}(t_i)$, is given as follows:

$$\left[\Delta\mathbf{y}[\mathbf{x}(t_t)]\right]_i = y_i - f_i[\mathbf{x}(t_i)]$$

Each row of $(\mathbf{y} - \mathbf{f})$ in equation (6) is referenced to time t_i. Therefore, the computation of the partial derivative of equation (6) with respect to $\mathbf{x}$ must be carried out with respect to time as well as geometry. We are interested in the partial derivative of the scalar loss function with respect to the state vector at reference time t_0. Calculating this partial derivative proceeds most easily with the scalar loss function written in summation notation:

$$Q = \sum_{i=1}^{p} w_i \left\{ y_i - f_i[\mathbf{x}(t_i)] \right\}^2$$

Now,

$$\left.\frac{\partial Q}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t_0)} = \sum_{i=1}^{p} -2w_i \left\{ y_i - f_i[\mathbf{x}(t_i)] \right\} \left.\frac{\partial f_i}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t_i)} \cdot \frac{\partial \mathbf{x}(t_i)}{\partial \mathbf{x}(t_0)}$$

Here, the $m \times m$ partial derivative of the state vector at time t_i with respect to the state vector at time t_0 is the state transition matrix from time t_0 to time t_i:

$$\Phi_0(t_i) \underset{\text{def}}{=} \frac{\partial \mathbf{x}(t_i)}{\partial \mathbf{x}(t_0)}$$

The computation of the elements of the state transition matrix is described in Section 2.6.

Computation of the geometrical partial derivatives is identical to the computation required for equation (3) in the static case. The $p \times m$ matrix function $\mathbf{F}$ is defined row-by-row as follows:

$$[\mathbf{F}]_i = \frac{\partial f_i}{\partial \mathbf{x}(t_i)} \cdot \Phi_0(t_i) \tag{7}$$

Thus, the necessary condition for minimization of the scalar loss function becomes

$$\left.\frac{\partial Q}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t_0)} = -2(\mathbf{y}-\mathbf{f})^{\mathrm{T}}\mathbf{W}\mathbf{F} = \mathbf{0}$$

$$\Rightarrow \quad (\mathbf{y}-\mathbf{f})^{\mathrm{T}}\mathbf{W}\mathbf{F} = \mathbf{0} \tag{8}$$

Equation (8) is identical in form to equation (2) for the static problem, but in (8), the state transition matrix to time t_i has been applied to each row i of the Jacobian. Equation (8) is a matrix representation of m non-linear equations in the m unknowns of the state vector $\mathbf{x}(t_0)$ at epoch, and again solution of the equation will proceed by Newton-Raphson

iteration. First, from the *a priori* estimate of the state vector at epoch, $\mathbf{x}_0(t_0)$, we find the state vector at each time t_i:

$$\mathbf{x}_0(t_i) = \Phi_0(t_i)\,\mathbf{x}_0(t_0)$$

Then we find the Taylor series expansion of each function f_i around the estimate of the state vector at each time t_i, $\mathbf{x}_0(t_i)$, with $\Delta\mathbf{x}(t_i) = \mathbf{x}(t_i) - \mathbf{x}_0(t_i)$:

$$f_i[\mathbf{x}(t_i)] = f_i[\mathbf{x}_0(t_i)] + \left.\frac{\partial f_i}{\partial \mathbf{x}(t_i)}\right|_{\mathbf{x}=\mathbf{x}_0(t_i)} \Delta\mathbf{x}(t_i) + O(\Delta\mathbf{x}^2) \tag{9}$$

Subtracting both sides of equation (9) from the measurement y_i and dropping all terms of higher degree than $\Delta\mathbf{x}$, we obtain a row-by-row approximation for the factor $(\mathbf{y} - \mathbf{f})$ in the necessary condition equation (8):

$$y_i - f_i[\mathbf{x}(t_i)] \approx y_i - f_i[\mathbf{x}_0(t_i)] - \left.\frac{\partial f_i}{\partial \mathbf{x}(t_i)}\right|_{\mathbf{x}=\mathbf{x}_0(t_i)} \Delta\mathbf{x}(t_i) \tag{10}$$

We may rewrite $\Delta\mathbf{x}(t_i)$, the correction to the state vector at time t_i appearing on the far right of equation (10), in terms of the correction at time t_0 by applying the state transition matrix:

$$\Delta\mathbf{x}(t_i) = \Phi_0(t_i)\Delta\mathbf{x}(t_0)$$

With substitution of this expression for $\Delta\mathbf{x}(t_i)$ in equation (10), we obtain

$$y_i - f_i[\mathbf{x}(t_i)] \approx y_i - f_i[\mathbf{x}_0(t_i)] - \left.\frac{\partial f_i}{\partial \mathbf{x}(t_i)}\right|_{\mathbf{x}=\mathbf{x}_0(t_i)} \Phi_0(t_i)\,\Delta\mathbf{x}(t_0)$$

$$= [\Delta\mathbf{y}_0]_i - [\mathbf{F}]_i \Delta\mathbf{x}(t_0) \tag{11}$$

In equation (11), the vector $\Delta\mathbf{y}_0$ is defined row-by-row as follows:

$$[\Delta\mathbf{y}_0]_i = y_i - f_i[\mathbf{x}_0(t_i)]$$

When equation (11) is collected for each $i = 1, ..., p$, we obtain the desired approximation for $(\mathbf{y} - \mathbf{f})$ in the necessary condition equation (8):

$$\mathbf{y} - \mathbf{f} \approx \Delta\mathbf{y}_0 - \mathbf{F}_0 \Delta\mathbf{x}(t_0) \tag{12}$$

Here, $\mathbf{F}_0$ is the matrix $\mathbf{F}$ as defined in equation (7) with the i^{th} row evaluated at $\mathbf{x} = \mathbf{x}_0(t_i)$. Then to first order, equation (8) may be rewritten with approximation (12) as follows:

$$[\mathbf{y} - \mathbf{f}]^{\mathrm{T}} \mathbf{W}\mathbf{F}_0 \approx [\Delta\mathbf{y}_0 - \mathbf{F}_0 \Delta\mathbf{x}(t_0)]^{\mathrm{T}} \mathbf{W}\mathbf{F}_0 = \mathbf{0}$$

$$\Rightarrow \quad \mathbf{F}_0^{\mathrm{T}} \mathbf{W}\mathbf{F}_0 \Delta\mathbf{x}(t_0) = \mathbf{F}_0^{\mathrm{T}} \mathbf{W}\Delta\mathbf{y}_0$$

As in the static case, the $m \times m$ information matrix is $\mathbf{S}_0 = \mathbf{F}_0^{\mathrm{T}} \mathbf{W}\mathbf{F}_0$, and so long as $\mathbf{S}_0$ is non-singular, its $m \times m$ inverse $\mathbf{P}_0 = \mathbf{S}_0^{-1}$ may be found. Thus, the correction $\Delta\mathbf{x}(t_0)$ may be found:

$$\Delta\mathbf{x}(t_0) = \mathbf{P}_0\mathbf{F}_0^{\mathrm{T}}\mathbf{W}\Delta\mathbf{y}_0$$

The Newton-Raphson iteration to find the state vector $\mathbf{x}(t_0)$ at epoch $t = t_0$ that best fits the p observations in the measurement vector $\mathbf{y}$ is summarized in Table 2. In the algorithm, i always refers to iteration number.

Table 2. Algorithm for Solving the Dynamic Orbit Determination Problem

1. Start with the *a priori* state vector estimate $\mathbf{x}_0(t_0)$, and let $i = 0$ be the initial iteration number. Compute the p state transition matrices $\Phi_0(t_j)$ from epoch t_0 to each measurement time t_j. Choose a threshold δx for quitting the iteration.

2. Compute $\Delta\mathbf{y}_i$ row-by-row from $[\Delta\mathbf{y}_i]_j = y_j - f_j[\mathbf{x}_i(t_j)]$. This expression gives row j of $\Delta\mathbf{y}_i$.

(Continued)

Table 2. Algorithm for Solving the Dynamic Orbit Determination Problem (Concluded)

3. Compute the Jacobian $\mathbf{F}_i$ with the j^{th} row evaluated at $\mathbf{x} = \mathbf{x}_i(t_j)$ and with the state transition matrices already calculated above:

$$[\mathbf{F}_i]_j = \left.\frac{\partial f_j}{\partial \mathbf{x}(t_j)}\right|_{\mathbf{x}=\mathbf{x}_i(t_j)} \Phi_0(t_j)$$

   This expression gives row j of the Jacobian.

4. Compute the information matrix $\mathbf{S}_i = \mathbf{F}_i^{\mathrm{T}}\mathbf{W}\mathbf{F}_i$. If the information matrix is singular or ill-behaved, attempt the iteration again with a different *a priori* state vector estimate $\mathbf{x}_0(t_0)$. If the iteration still fails to converge, it is likely that the measurements do not determine the full state vector, and the particular orbit determination scenario is degenerate. Otherwise, compute $\mathbf{P}_i = \mathbf{S}_i^{-1}$.

5. Calculate the correction to the state vector estimate at epoch:
   $\Delta\mathbf{x}(t_0) = \mathbf{P}_i\mathbf{F}_i^{\mathrm{T}}\mathbf{W}\Delta\mathbf{y}_i$.

6. Calculate the new state vector estimate at epoch: $\mathbf{x}_{i+1}(t_0) = \mathbf{x}_i(t_0) + \Delta\mathbf{x}(t_0)$.

7. If $|\Delta\mathbf{x}| < \delta x$, then quit the iteration with the best state vector estimate $\mathbf{x}(t_0) = \mathbf{x}_{i+1}(t_0)$. Otherwise, let $i = i + 1$, and repeat the iteration at step 2.

2.2 GROUP DELAY AND PHASE DELAY MEASUREMENTS

This section provides a brief introduction to the process of radio interferometry. Since the purpose of this study is to assess the statistical properties of the error that results from using differential range measurements to determine satellite orbits, precise details on the cross-correlation process for the generation of the differential range measurement from interferometric observables will not be provided. Readers interested in further information on the cross-correlation or other aspects of radio interferometry are referred to reference [6].

A differential range measurement is calculated from a phase delay or group delay observable generated by radio interferometry. Group delay is the differential time of arrival of a signal between two ground stations and is usually generated by a VLBI system. Each site within a VLBI system has an independent local oscillator for frequency and time standards. To generate a group delay observable, the signal from a satellite (which need not be a special signal for orbit determination) is time-tagged, sampled, digitized, and transmitted to the correlator site from each other site. At the correlator site, the recorded signals from each element of the interferometer are cross-correlated at various time offsets. The interpolated time offset that creates the peak of the cross-correlation function is the group delay measurement, $\tilde{\tau}$, where the tilda indicates that the true group delay, τ, is the measured group delay plus the clock offset between the two measuring sites minus delays due to the various systematic error sources such as tropospheric delays, ionospheric delay, and equipment biases, minus random noise error in the measurement. If Δt is the clock offset between two sites, τ_t is the tropospheric

delay, τ_t is the ionospheric delay, and τ_b is the delay due to equipment biases, and ε is the random noise error in the measurement, then the true group delay, τ, will be given by

$$\tau = \tilde{\tau} + \Delta t - \tau_t - \tau_i - \tau_b - \varepsilon \tag{13}$$

The tropospheric and ionospheric delays (or any other delays deriving from changes in the signal propagation rate) are referred to as nuisance delays. Tropospheric delay is a function of temperature and total water vapor content in the signal propagation path (i.e., a reading of relative humidity at the station is insufficient for predicting tropospheric delay). Ionospheric delay is a function of the signal frequency, and in certain regions of the spectrum (such as K-band), ionospheric delays are zero. ISODAE users may estimate clock offsets as additional solve-for parameters in the orbit determination process, or clock offsets may be specified to be zero. ISODAE assumes that the nuisance delays and equipment biases can be measured or predicted, and so they are not modeled in the simulation. Thus, for the purposes of ISODAE, equation (13) reduces to

$$\tau = \tilde{\tau} + \Delta t - \varepsilon \tag{14}$$

Phase delay is the phase offset of a signal between two different receiving stations and is generally measured by a collected element interferometry (CEI) system. The concept of phase delay is easily understood so long as the signal is in the form of coherent sinusoid; however, phase delay can be measured even on signals with a structure of random noise. Radio astronomers measure phase delay on quasars, for example, whose signal structure is essentially random noise. The measured phase delay, $\tilde{\phi}$,

between two elements of a CEI system is related to the true group delay, τ, by the relationship

$$\tilde{\phi} + 2\pi N - 2\pi\nu\tau_t - 2\pi\nu\tau_i - 2\pi\nu\tau_b - 2\pi\nu\varepsilon = 2\pi\nu\tau - \phi_{LO} \tag{15}$$

where ϕ_{LO} is the phase offset of the local oscillator reference signal at the antennas, N is the unknown cycle ambiguity, ν is the reference center frequency of the phase delay measurement, and again ε is the random noise error in the measurement. The true group delay is calculated from measured phase delay by solving equation (15) for τ:

$$\tau = \frac{\tilde{\phi} + 2\pi N}{2\pi\nu} + \frac{\phi_{LO}}{2\pi\nu} - \tau_t - \tau_i - \tau_b - \varepsilon = \tilde{\tau}_\phi + \Delta t_\phi - \tau_t - \tau_i - \tau_b - \varepsilon \tag{16}$$

where the definitions

$$\tilde{\tau}_\phi \underset{\text{def}}{=} \frac{\tilde{\phi} + 2\pi N}{2\pi\nu} \qquad \Delta t_\phi \underset{\text{def}}{=} \frac{\phi_{LO}}{2\pi\nu} \tag{17}$$

give equation (16) the same form as equation (13).

The reference frequency ν is known and the cycle ambiguity N may usually be deduced from *a priori* information about the satellite's position vector, but the local oscillator phase offset ϕ_{LO} must be estimated as one of the parameters in the orbit determination process. Even with temperature controlled fiberoptic cable links between antennas and the oscillator in a CEI system, it is virtually impossible to calculate ϕ_{LO} by carefully calibrating the system, and so it is necessary to estimate it as a parameter in the

orbit determination process. This is analogous to the estimation of the unknown clock offset between stations in the case of a VLBI system measuring group delay. Thus, equation (16) is of the same form as equation (13). If *a priori* satellite position information is insufficient for the determination of the cycle ambiguity N, then redundant measurements must be made at different reference center frequencies. For the purposes of ISODAE, it has been assumed that sufficient *a priori* information is available to determine the value of N. In the case of a CEI system, as for a VLBI system, it is assumed in ISODAE that nuisance delays and equipment bias delays may be measured or predicted. Thus, equation (16) reduces to the form of (14), again where measured group delay is derived from the measured phase delay as in definition (17) and Δt derives from ϕ_{LO} as in definition (17).

In a real scenario where an interferometry system is being used to determine the orbit of a satellite, it is necessary to calculate tropospheric and ionospheric parameters, to calibrate for equipment biases, and to calculate differential range as a function of group delay. These topics are beyond the scope of this study; ISODAE assumes that the differential range measurement has already been generated, clock offsets or local oscillator offsets notwithstanding. ISODAE allows the user to model error in the group delay measurements, but details in this section on group and phase delay observables are provided to the reader for background only.

2.3 GEOMETRY OF THE DIFFERENTIAL RANGE OBSERVABLE

Consider an interferometric orbit determination scenario in which O is the origin of an earth-centered inertial (ECI) coordinate system, $\mathbf{r}$ is the position vector of a satellite with respect to O, $\mathbf{b}_1$ and $\mathbf{b}_2$ are the position vectors of two ground stations from which measurements are to be made, and $\mathbf{d}_1$ and $\mathbf{d}_2$ are the position vectors of the satellite with respect to those ground stations, as pictured in Figure 1. All the position vectors $\mathbf{r}$, $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{d}_1$, and $\mathbf{d}_2$ are functions of time. Given the latitude, longitude, and height above sea level of a ground station, the position vector of that ground stations can be calculated as a function of time, as shown in Section 2.4.1. Assign ECI coordinates to the ground station position vectors as follows:

$$\mathbf{b}_1 = \begin{bmatrix} b_{1x} \\ b_{1y} \\ b_{1z} \end{bmatrix} \quad \text{and} \quad \mathbf{b}_2 = \begin{bmatrix} b_{2x} \\ b_{2y} \\ b_{2z} \end{bmatrix}$$

Notice that the sum of a station position vector $\mathbf{b}_k$ and the satellite position vector $\mathbf{d}_k$ measured from that station is simply the satellite position vector $\mathbf{r}$, or $\mathbf{b}_k + \mathbf{d}_k = \mathbf{r}$. Therefore, $\mathbf{d}_k = \mathbf{r} - \mathbf{b}_k$. If the propagation rate, c, of the signal through the atmosphere is known, then the transit time, T_k, of the signal from the satellite at point P to ground station k at point B_k will be given by

$$T_k = \tfrac{1}{c}|\mathbf{d}_k| = \tfrac{1}{c}\sqrt{(\mathbf{r}-\mathbf{b}_k)\cdot(\mathbf{r}-\mathbf{b}_k)}$$

(ISODAE takes the signal propagation rate to be the speed of light, which is

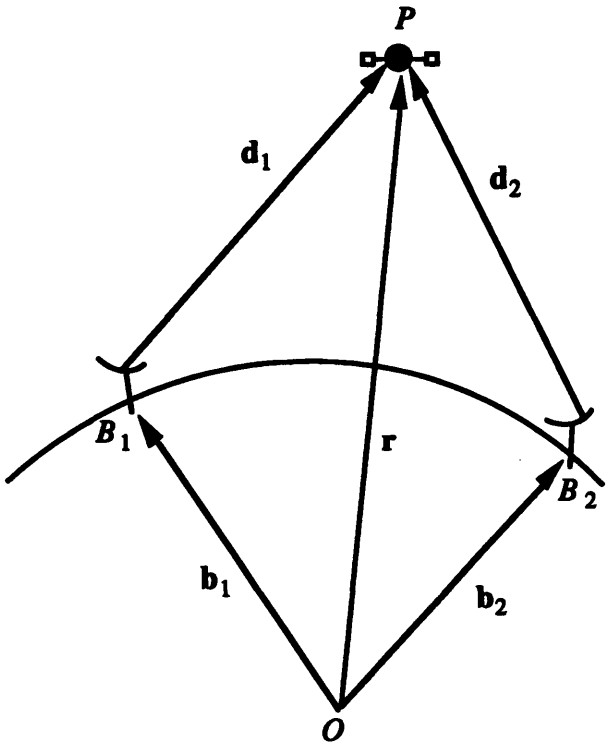

Figure 1. Illustration of the Interferometric Measurement Scenario

$c = 299{,}792{,}458$ m·sec^{-1} [7].) The true group delay, τ, is the differential transit time of the signal between the two sites:

$$\tau = T_2 - T_1 = \frac{1}{c}\left(|\mathbf{d}_2| - |\mathbf{d}_1|\right) = \frac{1}{c}\left[\sqrt{(\mathbf{r}-\mathbf{b}_2)\cdot(\mathbf{r}-\mathbf{b}_2)} - \sqrt{(\mathbf{r}-\mathbf{b}_1)\cdot(\mathbf{r}-\mathbf{b}_1)}\right] \qquad (18a)$$

If the position vector of the satellite is expressed in ECI coordinates as $\mathbf{r} = [x \quad y \quad z]^{\mathrm{T}}$ and the magnitude of station k's position vector is written $b_k = |\mathbf{b}_k|$, then the group delay may be expressed as follows:

$$\tau = \frac{1}{c}\sqrt{x^2 + y^2 + z^2 - 2\left(b_{2x}x + b_{2y}y + b_{2z}z\right) + b_2^2} - \frac{1}{c}\sqrt{x^2 + y^2 + z^2 - 2\left(b_{1x}x + b_{1y}y + b_{1z}z\right) + b_1^2} \tag{18b}$$

2.3.1 Differential Range Measurement Function

One subtlety of equations (18) is that the satellite position vector **r**, station 1 position vector $\mathbf{b}_1$, and station 2 position vector $\mathbf{b}_2$ are all referenced to different times. If the measured signal emanates from the satellite at time t, then it will arrive at station 1 at time $t + T$, and it will arrive at station 2 at time $t + T + \tau$, where T is the signal transit time from the satellite to station 1, and τ is the true group delay between stations 1 and 2. If the satellite position vector **r** is measured at time t, then the station 1 position vector is measured at time $t + T$, and the station 2 position vector is measured at time $t + T + \tau$. Thus, we write $\mathbf{r} = \mathbf{r}(t)$, $\mathbf{b}_1 = \mathbf{b}_1(t + T)$, and $\mathbf{b}_2 = \mathbf{b}_2(t + T + \tau)$. The group delay equation (18a) is, therefore, more properly written as follows:

$$\tau = \frac{1}{c}\sqrt{\left[\mathbf{r}(t) - \mathbf{b}_2(t + T + \tau)\right] \cdot \left[\mathbf{r}(t) - \mathbf{b}_2(t + T + \tau)\right]} - \frac{1}{c}\sqrt{\left[\mathbf{r}(t) - \mathbf{b}_1(t + T)\right] \cdot \left[\mathbf{r}(t) - \mathbf{b}_1(t + T)\right]} \tag{19}$$

In ISODAE, a user specifies a scenario that includes satellite position and velocity vectors at epoch, $t = t_0$, ground station locations from which differential range measurements are to be made, and times, $t_1, t_2, \ldots, t_p$, at which measurements are to be taken. It is, therefore, necessary for ISODAE to calculate the true group delay observables, after which measurement errors are added to obtain measured group delays.

If measurement number j is made at known time t_j from station 1, so that $t + T = t_j$, then the station 1 position vector, $\mathbf{b}_1(t_j)$, can be calculated from equation (25) in Section 2.4.1. The signal transit time to station 1, T, is unknown because it depends on the satellite position vector at the unknown time $t = t_j - T$, but it is obeys the relationship

$$T = \frac{1}{c}\sqrt{\left[\mathbf{r}(t_j - T) - \mathbf{b}_1(t_j)\right] \cdot \left[\mathbf{r}(t_j - T) - \mathbf{b}_1(t_j)\right]} \tag{20}$$

The signal transit time, T, may be calculated iteratively from equation (20), as shown in Table 3.

Once the algorithm in Table 3 has been applied, the signal transit time, T, to station 1 and the position vector of the satellite at the time of signal emanation, $\mathbf{r}(t_j - T)$, will have been computed. Unfortunately, the true group delay still cannot be computed directly from equation (19) because the right hand side is a function of τ. Thus, group delay must be calculated iteratively. The position vector for ground station 1, $\mathbf{b}_1(t_j)$, the position vector of the satellite at the time of signal emanation, $\mathbf{r}(t_j - T)$, and the signal transit time T to station 1 have already been computed and will remain fixed throughout the iteration, which is summarized in Table 4.

In setting up an orbit determination scenario, a user of ISODAE will specify an epoch time t_0, the satellite's position vector at epoch, $\mathbf{r}(t_0)$, and the satellite's velocity vector at epoch, $\mathbf{v}(t_0)$. The user will also specify a set of p measurement functions. For each measurement j of the differential range type, the user will specify the time of the

Table 3. Algorithm for Finding Signal Transit Time

1. Let t_0 be the epoch at which the satellite state vector $\mathbf{x}(t_0)$ is known. Compute the position vector for ground station 1, $\mathbf{b}_1(t_j)$, from equation (25) in Section 2.4.1. Choose a threshold δT for quitting the iteration.

2. Let $i = 0$ be the initial iteration number, and let $T_0 = 0$ be the initial estimate of the signal transit time T.

3. Compute the state transition matrix, $\Phi_0(t_j - T_i)$, from time t_0 to the current estimated time of signal emanation from the satellite, $t_j - T_i$, as described in Section 2.6. Compute the satellite state vector at time $t_j - T_i$ from

$$\mathbf{x}(t_j - T_i) = \Phi_0(t_j - T_i)\mathbf{x}(t_0)$$

Let $\mathbf{r}(t_j - T_i)$ be the position vector components of the state vector $\mathbf{x}(t_j - T_i)$.

4. Calculate T_{i+1} from the following:

$$T_{i+1} = \frac{1}{c}\sqrt{\left[\mathbf{r}(t_j - T_i) - \mathbf{b}_1(t_j)\right] \cdot \left[\mathbf{r}(t_j - T_i) - \mathbf{b}_1(t_j)\right]}$$

5. Quit the iteration if $|T_{i+1} - T_i| < \delta T$. Otherwise, let $i = i + 1$, and repeat at step 3.

Table 4. Algorithm for Finding True Group Delay

1. Let $i = 0$ be the initial iteration number, and let $\tau_0 = 0$ be the initial estimate of the true group delay. Choose a threshold $\delta\tau$ for quitting the iteration.

2. Compute the ground station 2 position vector, $\mathbf{b}_2(t_j + \tau_i)$, at the current estimated time of the signal arrival, $t_j + \tau_i$ from equation (25) in Section 2.4.1.

3. Calculate τ_{i+1} from the following:

$$\tau_{i+1} = \frac{1}{c}\sqrt{\left[\mathbf{r}(t_j - T) - \mathbf{b}_2(t_j + \tau_i)\right]\cdot\left[\mathbf{r}(t_j - T) - \mathbf{b}_2(t_j + \tau_i)\right]} - T$$

4. Quit the iteration if $|\tau_{i+1} - \tau_i| < \delta\tau$. Otherwise, let $i = i + 1$, and repeat at step 2.

measurement, t_j, the geodetic latitude, longitude, and height above sea level of the two stations between which the differential range was measured, and the clock offset (for group delay) or local oscillator offset (for phase delay). ISODAE will then generate the true group delay, τ_j, for the j^{th} measurement by applying the algorithms from Tables 3 and Table 4.

Once the true group delay corresponding to measurement number j, τ_j, has been computed, the true differential range measurement may be computed to be $c\tau_j$. If Δt_j is the given clock offset between the interferometer elements $j1$ and $j2$, then the differential

range equivalent of the clock offset is $\Delta r_j = c\Delta t_j$. The differential range measurement function may then be written as follows:

$$f_j\left[\mathbf{r}\left(t_j - T_j\right)\right] = c\tau_j - \Delta r_j \tag{21}$$

In equation (21), T_j is the signal transit time between the satellite and station $j1$, the first element of the interferometer generating the differential range measurement. This signal transit time is computed from the algorithm in Table 3. Finally, to generate a measurement for Monte Carlo simulation, ISODAE will add random measurement error with user-specified statistical properties to the measured differential range function in equation (21). This process is described in further detail in Section 3.

2.3.2 Geometrical Partial Derivatives of Differential Range

Shown in equations (22) are the geometrical partial derivatives of the differential range geometry function (21). In this formulation, $\mathbf{b}_{j1}$ and $\mathbf{b}_{j2}$ are the station position vectors for the two elements of the interferometer making measurement number j, and their ECI components are denoted as follows:

$$\mathbf{b}_{j1} = \begin{bmatrix} b_{j1x} \\ b_{j1y} \\ b_{j1z} \end{bmatrix} \quad \text{and} \quad \mathbf{b}_{j2} = \begin{bmatrix} b_{j2x} \\ b_{j2y} \\ b_{j2z} \end{bmatrix}$$

In equations (22), the magnitudes of the station position vectors are written as follows: $b_{j1} = \left|\mathbf{b}_{j1}\right|$ and $b_{j2} = \left|\mathbf{b}_{j2}\right|$.

$$\frac{\partial f_j}{\partial x} = \frac{b_{j1x} - x}{\sqrt{x^2 + y^2 + z^2 - 2\left(b_{j1x}x + b_{j1y}y + b_{j1z}z\right) + b_{j1}^2}} - \frac{b_{j2x} - x}{\sqrt{x^2 + y^2 + z^2 - 2\left(b_{j2x}x + b_{j2y}y + b_{j2z}z\right) + b_{j2}^2}} \tag{22a}$$

$$\frac{\partial f_j}{\partial y} = \frac{b_{j1y} - y}{\sqrt{x^2 + y^2 + z^2 - 2\left(b_{j1x}x + b_{j1y}y + b_{j1z}z\right) + b_{j1}^2}} - \frac{b_{j2y} - y}{\sqrt{x^2 + y^2 + z^2 - 2\left(b_{j2x}x + b_{j2y}y + b_{j2z}z\right) + b_{j2}^2}} \tag{22b}$$

$$\frac{\partial f_j}{\partial z} = \frac{b_{j1z} - z}{\sqrt{x^2 + y^2 + z^2 - 2\left(b_{j1x}x + b_{j1y}y + b_{j1z}z\right) + b_{j1}^2}} - \frac{b_{j2z} - z}{\sqrt{x^2 + y^2 + z^2 - 2\left(b_{j2x}x + b_{j2y}y + b_{j2z}z\right) + b_{j2}^2}} \tag{22c}$$

$$\frac{\partial f_j}{\partial\left(\Delta r_j\right)} = -1 \tag{22d}$$

Equations (22) may be written more compactly with a few additional definitions. Let $\mathbf{r} = [x \quad y \quad z]^T$ be the satellite position vector, let $\mathbf{d}_{j1} = \mathbf{r} - \mathbf{b}_{j1}$, let $\mathbf{d}_{j2} = \mathbf{r} - \mathbf{b}_{j2}$, and finally let $d_{j1} = |\mathbf{d}_{j1}|$ and $d_{j2} = |\mathbf{d}_{j2}|$. (The vectors $\mathbf{d}$ are the position vectors of the satellite measured from the ground stations.) Then equations (22) become

$$\frac{\partial f_j}{\partial \mathbf{r}} = \frac{\mathbf{d}_{j2}}{d_{j2}} - \frac{\mathbf{d}_{j1}}{d_{j1}}$$

The geometrical partial derivatives of the differential range geometry function with respect to all velocity, clock offset, or other state vector components are 0. In equations (22), the satellite position vector and ground station vector components must be evaluated at the appropriate times. In the orbit determination algorithm from Table 2, the computation of the following geometrical partial derivative is required:

$$\left.\frac{\partial f_j}{\partial \mathbf{x}(t_j)}\right|_{\mathbf{x}=\mathbf{x}(t_j)}$$

In this derivative, the satellite position vector is evaluated at time $t_j - T_j$, where T_j is the signal transit time from the satellite to station j,1. This transit time and the satellite position vector at the time of the signal emanation, $\mathbf{r}(t_j - T_j)$, are computed by the algorithm in Table 3. Therefore, x, y, and z in equations (22) are the components of $\mathbf{r}(t_j - T_j)$. The station 1 position vector for the measurement at time t_j is $\mathbf{b}_{j1}(t_j)$, which is computed as described subsequently in Section 2.4.1. Finally, the station 2 position vector, $\mathbf{b}_{j2}(t_j + \tau_j)$, is evaluated (also as described in Section 2.4.1) at time $t_j + \tau_j$, where τ_j is the true group delay between station 1 and 2, as computed by the algorithm in Table 4.

2.4 COORDINATE AND TIME SYSTEM

ISODAE uses the B1950 ECI coordinate system for all measurements and calculations. In the B1950 ECI coordinate system, the mean equator and mean equinox for Julian Ephemeris Date (JED) 2433282.5, or January 1.0, 1950, Ephemeris Time (ET), are used to define the z-axis and x-axis, respectively. The B1950 ECI coordinate system is commonly used for trajectory propagation and orbit determination; therefore, polynomial expressions for useful parameters such as solar and lunar ephemerides and the right ascension of the earth's prime meridian are readily available in this system.

2.4.1 Computation of Ground Station ECI Coordinates

This section describes the computation of the ECI coordinates of a site on the surface of the earth. To specify the location of the site, ISODAE requires the geodetic latitude of the site, L_d, the longitude of the site, λ, and the height of the site above mean sea level, h. Because the measure of latitude given on most maps is geodetic, ISODAE accepts geodetic and not geocentric latitude as input.

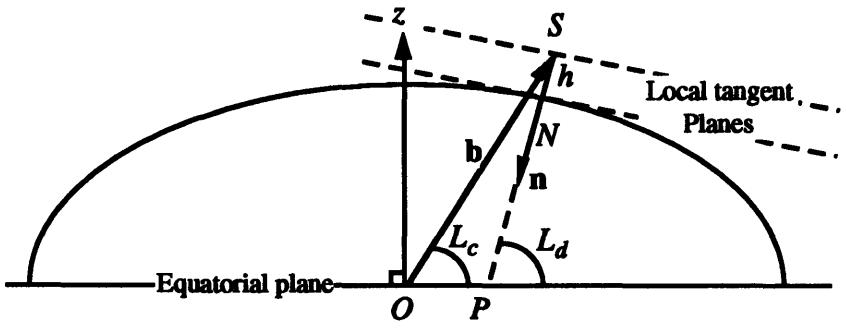

Figure 2. Geodetic (L_d) Versus Geocentric (L_c) Latitude

The difference between geodetic latitude, L_d, and geocentric latitude, L_c, is illustrated in Figure 2. In this model, the earth is treated as being an ellipsoid with circular cross-section normal to the z-axis and with elliptical cross-section normal to the xy-plane. This elliptical cross section will have semi-major axis a_e and eccentricity e_e, where $a_e = 6378.137$ km and $e_e = 0.08182$ [7]. Geocentric latitude is measured from the origin O and is the angle between the xy-plane and the site's position vector, $\mathbf{b}$. If $\mathbf{i}_z$ is a unit vector in the $+z$ direction, and $b = |\mathbf{b}|$, then

$$L_c = 90° - \arccos\frac{\mathbf{b}\cdot\mathbf{i}_z}{b}$$

Sites differing only in height above sea level will have different geocentric latitudes. Geodetic latitude, on the other hand, is computed so as to be independent of height above sea level. If a site is at point S in an ECI system with origin O, as shown in Figure 3, then line segment SP is constructed so that it intersects the oblate earth normal to the ellipsoid at the point of intersection, N. Line segment SN is then extended until it intersects the xy-(or equatorial) plane at point P. Geodetic latitude is then the angle measured from point P between line segment SP and the equatorial plane, where sites below the equator have negative latitudes. If $\mathbf{n}$ is the unit normal vector to the ellipsoidal earth from point N, as shown in Figure 3, and then

$$L_d = \arccos(\mathbf{n}\cdot\mathbf{i}_z) - 90°$$

The ECI coordinates, $\mathbf{b}(t) = \begin{bmatrix} b_x(t) & b_y(t) & b_z(t) \end{bmatrix}^{\mathrm{T}}$, of a site with geodetic latitude L_d, longitude λ, and height h above sea level at time t may be computed as follows. Let Δt be the difference in ephemeris days between 0000 hours, 1 January 1950, and the

observation time t. Then the right ascension, α, will be given by the site longitude plus the right ascension of the prime meridian, θ_{PM}. The following numerical approximation for θ_{PM} is given by NASA [8]:

$$\theta_{PM} = 99°.87 + 24\underbrace{\left(15.041067178\ °\cdot \text{day}^{-1}\right)}_{\text{Earth's mean rotation rate}}\Delta t \tag{23}$$

Thus, the right ascension of the site is given by

$$\alpha = \theta_{PM} + \lambda \tag{24}$$

Let **a** be the vector pointing from O to N, as shown in Figure 3. Let **h** be the vector pointing from N to the site location S, so that **h** has magnitude h and is normal to the reference ellipsoid. It may be verified from geometry the vector **a** may be written as follows:

$$\mathbf{a} = a_e \begin{bmatrix} \dfrac{\cos\alpha}{\sqrt{1+\left(1-e_e^2\right)\tan^2 L_d}} \\ \dfrac{\sin\alpha}{\sqrt{1+\left(1-e_e^2\right)\tan^2 L_d}} \\ \dfrac{\left(1-e_e^2\right)\sin L_d}{\sqrt{1-e_e^2\sin^2 L_d}} \end{bmatrix}$$

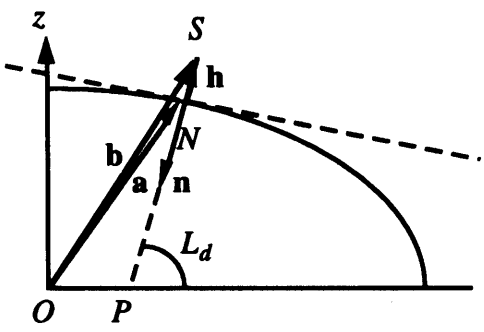

Figure 3. Geometry for Calculating the Site Position Vector

It may be verified from analytical geometry that the unit normal vector **n** to the ellipsoidal earth may be written as follows:

$$\mathbf{n} = -\begin{bmatrix} \cos\alpha\cos L_d \\ \sin\alpha\cos L_d \\ \sin L_d \end{bmatrix}$$

From Figure 3, it may be seen that the station position vector **b** is the vector sum of **a** and **h**, where $\mathbf{h} = -h\mathbf{n}$. Thus,

$$\mathbf{b} = \mathbf{a} - h\mathbf{n} = \begin{bmatrix} \dfrac{a_e\cos\alpha}{\sqrt{1+\left(1-e_e^2\right)\tan^2 L_d}} + h\cos\alpha\cos L_d \\ \dfrac{a_e\sin\alpha}{\sqrt{1+\left(1-e_e^2\right)\tan^2 L_d}} + h\sin\alpha\cos L_d \\ \dfrac{a_e\left(1-e_e^2\right)\sin L_d}{\sqrt{1-e_e^2\sin^2 L_d}} + h\sin L_d \end{bmatrix} \tag{25}$$

It should be noted that equation (25) for the ECI coordinates of a site position vector **b** is a function of time because the right ascension α is a function of time.

2.5 EQUATIONS OF MOTION AND FORCE MODELS

Expressed in the B1950 ECI coordinate frame, the equations of motion for a satellite can be directly integrated in rectangular coordinates. If the satellite's position vector is **r** and the velocity vector is **v**, then the equations of motion may be written as follows:

$$\frac{d^2}{dt^2}\mathbf{r} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_d \qquad \text{or} \qquad \frac{d}{dt}\begin{bmatrix}\mathbf{r}\\ \mathbf{v}\end{bmatrix} = \begin{bmatrix}\mathbf{v}\\ -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_d\end{bmatrix} \tag{26}$$

where $\mu = 398{,}600.45\ \text{km}^3\cdot\text{sec}^{-2}$ [7] is the product of the universal gravitational constant and the mass of the earth, $r = |\mathbf{r}|$, and $\mathbf{a}_d$ is the sum of the disturbing forces acting on the satellite. The first term on the right hand side of equation (26) is the point mass gravitational force of the earth. If there were no disturbing forces, $\mathbf{a}_d = \mathbf{0}$, then the satellite would obey Keplerian motion, and the solution to (26) would be the two-body solution described in Section 2.6.1.

Since the earth's gravity is usually modeled by a gravitational potential function V, the first term of which gives the point mass gravitational potential, equation (26) is usually rewritten as follows:

$$\frac{d}{dt}\begin{bmatrix}\mathbf{r}\\ \mathbf{v}\end{bmatrix}=\begin{bmatrix}\mathbf{v}\\ \nabla V+\mathbf{a}_{SRP}+\mathbf{a}_S+\mathbf{a}_M+\mathbf{a}_D\end{bmatrix} \tag{27}$$

where

∇V = $\left[\frac{\partial V}{\partial x} \quad \frac{\partial V}{\partial y} \quad \frac{\partial V}{\partial z}\right]^T$

$\mathbf{a}_{SRP}$ = acceleration due to solar radiation pressure

$\mathbf{a}_S$ = acceleration due to solar gravity

$\mathbf{a}_M$ = acceleration due to lunar gravity

$\mathbf{a}_D$ = acceleration due to atmospheric drag

In the rest of this section, the forms of the earth's gravitational potential, the solar radiation pressure acceleration, the solar and lunar gravitational accelerations, and the acceleration due to atmospheric drag will be presented.

2.5.1 Geopotential Model

The earth's gravitational field is often expressed with a spherical harmonic expansion. The gravitational potential V at any point P in space is given by the spherical harmonic series

$$V=\frac{\mu}{r}\left\{1+\sum_{l=2}^{\infty}\sum_{m=0}^{l}\left(\frac{a_e}{r}\right)^l P_{lm}(\sin\phi)\left[C_{lm}\cos m\lambda+S_{lm}\sin m\lambda\right]\right\} \tag{28}$$

where

r	=	distance of P from the origin (center of the earth in an ECI coordinate system)
ϕ	=	geocentric latitude of P
λ	=	right ascension of P
a_e	=	mean equatorial radius of the earth (6378.137 km)
P_{lm}	=	associated Legendre function
C_{lm}	=	spherical harmonic cosine coefficient of degree l and order m
S_{lm}	=	spherical harmonic sine coefficient of degree l and order m

The polar coordinates of the point P are (r, ϕ, λ), and the conversion from polar to Cartesian coordinates is as usual

$$x = r\cos\phi\cos\lambda$$

$$y = r\cos\phi\sin\lambda$$

$$z = r\sin\phi$$

The associated Legendre functions are given by

$$P_{lm}(\sin\phi) = \cos^m\phi\sum_{t=0}^{k}\frac{(-1)^t(2l-2t)!}{2^l t!(l-t)!(l-m-2t)!}\sin^{l-m-2t}\phi$$

where k is the integer part of $(l-m)/2$.

Because the order of magnitude of the coefficients C and S decreases fairly quickly with l, known as the *degree*, and especially with m, known as the *order*, it has been common practice to present the harmonic coefficients in *normalized form*, in which the order of magnitude remains approximately constant. Kaula [9] introduced the normalization factor that has become standard:

$$C_{lm} = \sqrt{\frac{(l-m)!(2l+1)(2-\delta_{0m})}{(l+m)!}}\,\bar{C}_{lm} \tag{29}$$

where C_{lm} is the standard coefficient used in equation (28), $\bar{C}_{lm}$ is the normalized coefficient, and δ_{0m} is the Kronecker delta between 0 and m. The normalization factor in equation (29) also applies to the S_{lm} coefficients.

For the purposes of this discussion, a gravity model refers to a set of coefficients in the harmonic expansion of the earth's gravitational field. Models are generated by regressions on highly precise satellite tracking data, usually in the forms of laser ranging, Doppler, and satellite-to-satellite ranging. There are a number of different models that have been developed, including GEM-T2, Goddard's latest earth gravity model generated from 2.4 million observations [10]. It is GEM-T2 that has been chosen for use by ISODAE. While GEM-T2 extends to 50th degree and 43rd order, the coefficients only through fourth degree and order, inclusive, have been used for ISODAE. These are given, both in normalized and in standard form, in Table 5.

Table 5. GEM-T2 Normalized and Standard Gravity Coefficients

l	m	$\overline{C}$	C	$\overline{S}$	S
2	0	$-4.841652998\times10^{-4}$	-1.082627×10^{-3}	0	0
2	1	0	0	0	0
2	2	2.4390067×10^{-6}	1.574372×10^{-6}	-1.400870×10^{-6}	-9.042577×10^{-7}
3	0	9.570331×10^{-5}	2.532072×10^{-6}	0	0
3	1	2.0307524×10^{-6}	2.193463×10^{-6}	2.496027×10^{-5}	2.696017×10^{-7}
3	2	9.035391×10^{-5}	3.086174×10^{-7}	-6.189858×10^{-5}	-2.114239×10^{-7}
3	3	7.215073×10^{-5}	1.006094×10^{-7}	1.4137252×10^{-6}	1.971346×10^{-7}
4	0	5.399078×10^{-5}	1.619723×10^{-6}	0	0
4	1	-5.352557×10^{-5}	-5.077881×10^{-7}	-4.741332×10^{-5}	-4.498022×10^{-7}
4	2	3.482596×10^{-5}	7.787321×10^{-8}	6.640236×10^{-5}	1.484802×10^{-7}
4	3	9.913108×10^{-5}	5.924215×10^{-8}	-2.014288×10^{-5}	-1.203767×10^{-8}
4	4	-1.893677×10^{-5}	-4.001123×10^{-9}	3.089680×10^{-5}	6.528140×10^{-9}

2.5.2 Third-Body Gravity Model

For earth-orbiting missions, third-body gravity refers to the disturbing forces exerted on a spacecraft by all gravitational bodies other than the earth. For such missions, the predominant third-body forces are due to the sun and moon. In Figure 4, which illustrates the situation for the disturbing acceleration due to the moon, O is the origin of the ECI coordinate frame, $\mathbf{r}$ is the position vector of the spacecraft in the ECI coordinate frame, P is the point at which the spacecraft is located, $\mathbf{r}_M$ is the position vector of the moon, M is the point at which the moon is located, and $\mathbf{d}_M$ is the position vector of the spacecraft with

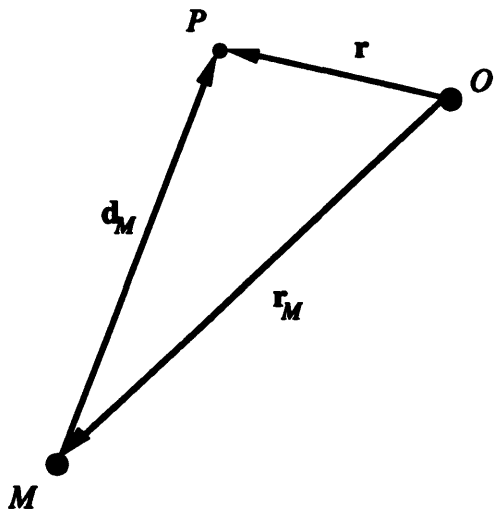

Figure 4. Third-Body Disturbing Gravity

respect to the moon ($\mathbf{d}_M = \mathbf{r} - \mathbf{r}_M$). Equation (30) gives the disturbing acceleration on a spacecraft due to the moon, where μ_M is the product of the mass of the moon and the universal gravitational constant (4,902.79 $km^3 \cdot sec^{-2}$ [11]). The analogous expression for the disturbing acceleration due to the sun is shown in equation (31), where μ_S is the product of the mass of the sun and the universal gravitational constant (1.3271244×10^{11} $km^3 \cdot sec^{-2}$ [11]).

$$\mathbf{a}_M = -\mu_M \left(\frac{1}{d_M^3} \mathbf{d}_M + \frac{1}{r_M^3} \mathbf{r}_M \right) = -\mu_M \left(\frac{\mathbf{r} - \mathbf{r}_M}{|\mathbf{r} - \mathbf{r}_M|^3} + \frac{\mathbf{r}_M}{r_M^3} \right) \tag{30}$$

$$\mathbf{a}_S = -\mu_S \left(\frac{\mathbf{r} - \mathbf{r}_S}{|\mathbf{r} - \mathbf{r}_S|^3} + \frac{\mathbf{r}_S}{r_S^3} \right) \tag{31}$$

The following are the osculating orbital elements for the sun, as given by JPL [8], with T being the integer number of Julian centuries (36,525 ephemeris days) from epoch January 1.0, 1950 ET, and with d being the number of ephemeris days from the same epoch.

$$a = 1.49597927 \times 10^{8} \quad \text{(km)}$$

$$e = 1.67301085 \times 10^{-2} - 4.1926 \times 10^{-5} T - 1.26 \times 10^{-7} T^{2}$$

$$i = 23.4457888616 - 1.30141669 \times 10^{-2} T - 9.445 \times 10^{-7} T^{2} + 5.000 \times 10^{-7} T^{3} \quad (^{\circ})$$

$$\Omega = 0^{\circ}$$

$$\omega = 282.08053 + 0.32328T + 1.5 \times 10^{-4} T^{2} \quad (^{\circ})$$

$$M = 358.000682 + 0.9856002623d - 1.550000 \times 10^{-4} T^{2} - 3.3333 \times 10^{-6} T^{3} \quad (^{\circ})$$

The following are the osculating orbital elements of the moon, with T and d as given above [8].

$$a = 3.843993 \times 10^{5} \quad \text{(km)}$$

$$e = 0.054900489$$

$$i = 5.145396 \quad (^{\circ})$$

$$\Omega = 12.112791 - 1934.139929T + 2.081 \times 10^{-3} T^{2} + 2.0 \times 10^{-6} T^{3} \quad (^{\circ})$$

$$\omega = 196.731198 + 6003.163629T - 1.2425 \times 10^{-2} T^{2} - 1.4 \times 10^{-5} T^{3} \quad (^{\circ})$$

$$M = 215.531463 + 4.77198858310 \times 10^{5} T + 9.214 \times 10^{-3} T^{2} + 1.4 \times 10^{-5} T^{3} \quad (^{\circ})$$

The lunar and solar ephemerides that are computed from these polynomials will be in the B1950 ECI coordinate frame. ISODAE computes lunar and solar position vectors from

the osculating orbital elements as follows. First, the eccentric anomaly, E, is computed from the mean anomaly, M, and the eccentricity, e, by solving Kepler's equation:

$$M = E - e\sin E$$

This equation is transcendental in the desired parameter, E, and so a numerical solution must be obtained. The Newton's method procedure for the solution of Kepler's equation is provided in Table 8 of Section 2.6.1. Now the value of the true anomaly, f, is computed as follows:

$$f = 2\arctan\left[\sqrt{\frac{1+e}{1-e}}\tan\tfrac{1}{2}E\right]$$

and the value of the angle θ is defined to be the sum of the true anomaly and the argument of perigee: $\theta = f + \omega$. Next, the magnitude of the position vector of the disturbing body, $r_D = |\mathbf{r}_D|$, is computed as follows:

$$r_D = a(1 - e\cos E)$$

Finally, the components of the position vector of the disturbing body are computed as follows:

$$\mathbf{r}_D = r_D\begin{bmatrix}\cos\Omega\cos\theta - \sin\Omega\sin\theta\cos i\\ \sin\Omega\cos\theta + \cos\Omega\sin\theta\cos i\\ \sin\theta\sin i\end{bmatrix}$$

2.5.3 Solar Radiation Pressure Model

ISODAE uses the static solar radiation pressure model, which states that the acceleration due to the solar photons impinging on the surface of a spacecraft is given by

$$\mathbf{a}_{SRP} = k_e P_S r_S^2 (1+\eta)\left(\frac{A_S}{m}\right)\frac{(\mathbf{r}-\mathbf{r}_S)}{|\mathbf{r}-\mathbf{r}_S|^3} \tag{32}$$

where $\mathbf{r}$ is the satellite position vector, P_S is the solar pressure flux at one astronomical unit, 4.5×10^{-6} N·m^{-2} [12], $\mathbf{r}_S$ is position vector of the sun (and r_S is its magnitude), η is the reflectance of the surface of the satellite, $-1 \le \eta \le 1$ (–1 for perfectly transparent material, zero for a perfectly black body, and unity for a perfectly reflective material), A_S is the orbit-average area of the spacecraft projected into the plane normal to the line of sight from the spacecraft to the sun, m is the mass of the spacecraft, and k_e is zero if the earth is eclipsing the sun from the spacecraft or one otherwise.

The user of ISODAE must specify the spacecraft mass, m, and the solar radiation pressure cross-sectional area, A_S. ISODAE must compute the position vector of the sun, which is done exactly as described in Section 2.5.2 for third-body gravity.

2.5.4 Atmospheric Drag Model

ISODAE uses the static atmospheric drag model, which states that the acceleration due to a spacecraft moving through the atmosphere is given by

$$\mathbf{a}_D = -\tfrac{1}{2} C_D \left(\frac{A_D}{m} \right) \rho[h(\mathbf{r})] v_r \mathbf{v}_r \tag{33}$$

where $\mathbf{r}$ is the satellite's position vector, $h(\mathbf{r})$ is the height of the satellite above the surface of the earth, $\rho[h(\mathbf{r})]$ is the density of the atmosphere at the satellite's position, C_D is the satellite's coefficient of drag (usually around 2.0), $\mathbf{v}_r$ is the component of the satellite's velocity vector in the direction of the rotating atmosphere (and v_r is its magnitude), A_D is the orbit-average area of the spacecraft projected into the plane normal to $\mathbf{v}_r$, and m is the mass of the spacecraft.

The user of ISODAE must specify the spacecraft mass, m, the drag cross-sectional area, A_D, and the coefficient of drag, C_D. ISODAE must compute the density of the atmosphere at the satellite position and the satellite velocity relative to the atmosphere. The latter is accomplished by applying the Coriolis theorem, from which the following obtains:

$$\mathbf{v}_r = \mathbf{v} - \boldsymbol{\omega} \times \mathbf{r} \tag{34}$$

where $\mathbf{v}$ is the satellite's ECI velocity vector, $\mathbf{r}$ is the satellite's ECI position vector, and $\boldsymbol{\omega}$ is the angular momentum vector of the earth, which is given as

$$\boldsymbol{\omega} = \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix}$$

where $\omega = 7.2921151464 \times 10^{-5}$ rad·sec^{-1} is the mean rotation rate of the earth.

Atmospheric density is modeled as being a function of the height of the spacecraft above the surface of the earth, which is in turn a function of the satellite's position vector, **r**. The height, $h(\mathbf{r})$, will be given as follows:

$$h(\mathbf{r}) = r - a_e \sqrt{\frac{1 - e_e^2}{1 - e_e^2 \cos^2 L_c}} \tag{35}$$

where $r = |\mathbf{r}|$, a_e and e_e are the mean equatorial radius and eccentricity, respectively, of the ellipsoidal earth, as given in Section 2.4.1, and L_c is the geocentric latitude of the satellite, as calculated from equation (25) in section 2.4.1 (with $\mathbf{b} = \mathbf{r}$).

ISODAE uses the 1976 U.S. Standard Atmosphere model adapted by JPL [13, 11]. This model provides tabular densities for heights ranging from 86 km to 1000 km. Altitudes lower than 86 km are given the density value at 86 km, though a satellite at such an altitude will be in a highly unstable orbit. The atmosphere is generally considered to have no effect at altitudes greater than 1000 km. For heights between the values listed in the table, ISODAE performs a linear interpolation. The 1976 U.S. Standard Atmosphere is shown in Table 6.

Table 6. The 1976 U.S. Standard Atmosphere

Altitude		Temperature			Pressure			Density	
Z (m)	H (m)	T (K)	t (°C)	T_M (K)	P (mb)	P (torr)	P/P_0	ρ (kg/m^3)	ρ/ρ_0
86000	84852	186.87	-86.28	186.95	3.7338e-3	2.8006e-3	3.6850e-6	6.958e-6	5.000e-6
86500	85339	186.87	-86.28	186.96	3.4163	2.5624	3.3716	6.366	5.196
87000	85825	186.87	-86.28	186.98	3.1259	2.3446	3.0850	5.824	4.754
87500	86312	186.87	-86.28	187.03	2.8602	2.1454	2.8228	5.328	4.350
88000	86798	186.87	-86.28	187.03	2.6173	1.9631	2.5831	4.875	3.980
88500	87285	186.87	-86.28	187.06	2.3951	1.7965	2.3638	4.460	3.641
89000	87771	186.87	-86.28	187.11	2.1919	1.6440	2.1632	4.081	3.331
89500	88257	186.87	-86.28	187.16	2.0060	1.5044	1.9797	3.734	3.048
90000	88744	186.87	-86.28	187.21	1.8359e-3	1.3771e-3	1.8119e-6	3.416e-6	2.789e-6
90500	89230	186.87	-86.28	187.28	1.6804	1.2604	1.6584	3.126	2.552
91000	89716	186.87	-86.28	187.36	1.5381	1.1536	1.5179	2.860	2.335
91500	90202	186.89	-86.26	187.47	1.4078	1.0560	1.3894	2.616	2.136
92000	90688	186.96	-86.19	187.64	1.2887	9.6662e-4	1.2719	2.393	1.953
92500	91173	187.08	-86.07	187.87	1.1798	8.8490	1.1643	2.188	1.786
93000	91659	187.25	-85.90	188.16	1.0801	8.1014	1.0660	2.000	1.632
93500	92145	187.47	-85.68	188.51	9.8896e-4	7.4178	9.7602e-7	1.826	1.492
94000	92630	187.74	-85.41	188.92	9.0568	6.7925	8.9375	1.670	1.363
94500	93116	188.05	-85.10	189.39	8.2937	6.2208	8.1852	1.526	1.245
95000	93601	188.42	-84.73	189.92	7.5966e-4	5.6979e-4	7.4973e-7	1.393e-6	1.137e-6
95500	94087	188.84	-84.31	190.52	6.9592	5.2199	6.8682	1.273	1.039
96000	94572	189.31	-83.84	191.17	6.3765	4.7828	6.2932	1.162	9.486e-7
96500	95057	189.83	-83.32	191.88	5.8439	4.3833	5.7675	1.061	8.668
97000	95542	190.40	-82.75	192.69	5.3571	4.0181	5.2870	9.685e-7	7.906
97500	96027	191.04	-82.11	193.55	4.9122	3.6844	4.8480	8.842	7.218
98000	96512	191.72	-81.43	194.48	4.5057	3.3795	4.4468	8.071	6.588
98500	96997	192.47	-80.68	195.49	4.1342	3.1009	4.0802	7.367	6.014
99000	97482	193.26	-79.87	196.58	3.7948	2.8463	3.7452	6.725	5.490
99500	97967	194.15	-79.00	197.74	3.4846	2.6137	3.4390	6.139	5.011
100000	98451	195.04	-78.07	198.99	3.2011e-4	2.4010e-4	3.1593e-7	5.604e-7	4.575e-7
101000	99428	197.16	-75.99	201.75	2.7192	2.0396	2.6837	4.695	3.833
102000	100389	199.53	-73.62	204.88	2.3144	1.7359	2.2841	3.935	3.212
103000	101358	202.23	-70.92	208.42	1.9742	1.4808	1.9484	3.308	2.694
104000	102326	205.31	-67.84	212.41	1.6882	1.2663	1.6661	2.769	2.260
105000	103294	208.84	-64.31	216.93	1.4477	1.0859	1.4288	2.325	1.898
106000	104261	212.89	-60.26	222.09	1.2454	9.3411e-5	1.2291	1.954	1.595
107000	105229	217.63	-55.52	226.82	1.0751	8.0642	1.0611	1.643	1.341
108000	106196	223.29	-49.86	235.80	9.3188e-5	6.9897	9.1970e-8	1.381	1.128
109000	107162	230.33	-42.82	243.53	8.1142	6.0862	8.0081	1.161	9.475e-8
110000	108129	240.00	-33.15	254.93	7.1042e-5	5.3286e-5	7.0113e-8	9.788e-8	7.925e-8
111000	109095	252.00	-21.15	268.91	6.2614	4.6965	6.1790	8.111	6.622
112000	110061	264.00	-9.15	283.06	5.5547	4.1664	5.4821	6.838	5.582
113000	111026	276.00	2.85	297.17	4.9570	3.7180	4.8922	5.811	4.744
114000	111992	288.00	14.85	311.46	4.4473	3.3358	4.3892	4.975	4.061
115000	112957	300.00	26.85	325.69	4.0096	3.0075	3.9572	4.289	3.501
116000	113921	312.00	38.85	340.04	3.6312	2.7236	3.5837	3.720	3.037
117000	114885	324.00	50.85	354.43	3.3022	2.4768	3.2598	3.240	2.650
118000	115849	336.00	62.85	368.98	3.0144	2.2610	2.9758	2.847	2.324
119000	116813	348.00	74.85	383.37	2.7615	2.0713	2.7254	2.509	2.048
120000	117777	360.00	86.85	397.91	2.5382e-5	1.9038e-5	2.5050e-8	2.222e-8	1.814e-8
121000	118740	371.89	98.74	412.38	2.3401	1.7552	2.3095	1.977	1.614
122000	119703	383.55	110.40	426.66	2.1635	1.6228	2.1352	1.767	1.442
123000	120665	394.99	121.84	440.74	2.0055	1.5043	1.9793	1.585	1.294
124000	121627	406.22	133.07	454.64	1.8635	1.3977	1.8391	1.428	1.166
125000	122589	417.23	144.08	468.35	1.7354	1.3016	1.7127	1.291	1.054
126000	123551	428.04	154.89	481.89	1.6194	1.2147	1.5983	1.171	9.557e-9
127000	124512	438.64	165.49	495.26	1.5141	1.1357	1.4943	1.065	8.694
128000	125473	449.04	175.89	508.46	1.4183	1.0638	1.3997	9.717e-9	7.932
129000	126434	459.25	186.10	521.49	1.3307	9.9810e-6	1.3133	8.889	7.257
130000	127395	469.27	196.12	534.36	1.2505e-5	9.3795e-6	1.2341e-8	8.152e-9	6.655e-9
131000	128355	479.09	205.94	547.08	1.1769	8.8275	1.1615	7.494	6.118
132000	129315	488.74	215.59	559.64	1.1092	8.3196	1.0947	6.904	5.636
133000	130274	498.20	225.05	572.06	1.0468	7.8513	1.0331	6.374	5.204
134000	131234	507.48	234.33	584.32	9.8907e-6	7.4187	9.7614e-9	5.897	4.814
135000	132193	516.59	243.44	596.44	9.3568	7.0182	9.2365	5.465	4.461
136000	133151	525.53	252.38	608.42	8.8617	6.6488	8.7459	5.074	4.142
137000	134110	534.29	261.14	620.25	8.4018	6.3019	8.2919	4.719	3.852
138000	135068	542.90	269.75	631.95	7.9739	5.9809	7.8696	4.396	3.588
139000	136026	551.34	278.19	643.51	7.5751	5.6818	7.4760	4.101	3.348
140000	136983	559.63	286.48	654.94	7.2028e-6	5.4026e-6	7.1087e-9	3.831e-9	3.128e-9
141000	137940	567.76	294.61	666.23	6.8550	5.1414	6.7653	3.584	2.926
142000	138897	575.73	302.58	677.40	6.5294	4.8974	6.4440	3.358	2.741
143000	139854	583.56	310.41	688.44	6.2243	4.6686	6.1429	3.150	2.571
144000	140810	591.24	318.09	699.36	5.9380	4.4539	5.8604	2.958	2.415
145000	141766	598.78	325.63	710.15	5.6691	4.2522	5.5950	2.781	2.270
146000	142722	606.17	333.02	720.82	5.4162	4.0625	5.3454	2.618	2.137
147000	143677	613.43	340.28	731.36	5.1781	3.8839	5.1104	2.466	2.013
148000	144633	620.55	347.40	741.81	4.9538	3.7156	4.8890	2.326	1.899
149000	145587	627.54	354.39	752.14	4.7421	3.5569	4.6801	2.196	1.793

Table 6. The 1976 U.S. Standard Atmosphere (Continued)

Altitude		Temperature			Pressure			Density	
Z (m)	H (m)	T (K)	t (°C)	T_M (K)	P (mb)	P (torr)	P/P_0	ρ (kg/m^3)	ρ/ρ_0
150000	146562	634.39	361.24	762.35	4.5422- 6	3.4070- 6	4.4828- 9	2.076- 9	1.694- 9
151000	147696	641.12	367.97	772.45	4.3533	3.2651	4.2964	1.963	1.603
152000	148830	647.72	374.57	782.44	4.1746	3.1312	4.1200	1.859	1.517
153000	149964	654.28	381.13	792.32	4.0054	3.0043	3.9538	1.761	1.438
154000	150357	660.56	387.41	802.19	3.8451	2.8840	3.7948	1.670	1.363
155000	151311	666.88	393.65	811.77	3.6938	2.7708	3.6457	1.585	1.294
156000	152263	672.92	399.77	821.34	3.5487	2.6617	3.5023	1.505	1.229
157000	153216	678.93	405.78	830.81	3.4116	2.5589	3.3670	1.431	1.168
158000	154168	684.83	411.68	840.18	3.2813	2.4612	3.2384	1.361	1.111
159000	155120	690.61	417.46	849.45	3.1574	2.3683	3.1161	1.295	1.057
160000	156072	696.29	423.14	858.63	3.0395- 6	2.2798- 6	2.9997- 9	1.233- 9	1.007- 9
161000	157023	701.86	428.71	867.71	2.9272	2.1956	2.8889	1.175	9.593- 10
162000	157974	707.33	434.18	876.70	2.8201	2.1153	2.7833	1.121	9.148
163000	158925	712.70	439.55	885.60	2.7181	2.0387	2.6825	1.069	8.726
164000	159875	717.96	444.81	894.41	2.6207	1.9657	2.5864	1.021	8.333
165000	160826	723.13	449.98	903.13	2.5278	1.8960	2.4947	9.750- 10	7.959
166000	161775	728.20	455.05	911.77	2.4394	1.8294	2.4071	9.319	7.607
167000	162725	733.18	460.03	920.32	2.3551	1.7657	2.3233	8.911	7.274
168000	163674	738.07	464.92	928.78	2.2750	1.7049	2.2432	8.525	6.960
169000	164623	742.86	469.71	937.16	2.1953	1.6466	2.1666	8.161	6.662
170000	165572	747.57	474.42	945.46	2.1210- 6	1.5909- 6	2.0933- 9	7.815- 10	6.380- 10
171000	166521	752.18	479.03	953.68	2.0499	1.5375	2.0231	7.488	6.113
172000	167469	756.71	483.56	961.82	1.9817	1.4864	1.9558	7.178	5.859
173000	168417	761.16	488.01	969.89	1.9164	1.4374	1.8913	6.883	5.619
174000	169364	765.53	492.38	977.87	1.8537	1.3904	1.8295	6.604	5.391
175000	170311	769.81	496.66	985.78	1.7936	1.3453	1.7702	6.339	5.174
176000	171256	774.01	500.86	993.62	1.7360	1.3021	1.7132	6.086	4.968
177000	172205	778.14	504.99	1001.38	1.6806	1.2605	1.6584	5.844	4.773
178000	173151	782.19	509.04	1009.07	1.6274	1.2206	1.6061	5.614	4.586
179000	174098	786.17	513.02	1016.69	1.5763	1.1823	1.5557	5.401	4.409
180000	175043	790.07	516.92	1024.24	1.5271- 6	1.1455- 6	1.5072- 9	5.194- 10	4.240- 10
181000	175989	793.89	520.74	1031.72	1.4799	1.1100	1.4606	4.997	4.079
182000	176934	797.64	524.50	1039.13	1.4345	1.0759	1.4157	4.809	3.926
183000	177879	801.34	528.19	1046.47	1.3907	1.0431	1.3726	4.630	3.779
184000	178824	804.96	531.81	1053.75	1.3487	1.0116	1.3310	4.459	3.640
185000	179768	808.51	535.36	1060.96	1.3081	9.8117- 7	1.2910	4.295	3.506
186000	180712	812.00	538.85	1068.11	1.2691	9.5149	1.2525	4.138	3.379
187000	181656	815.42	542.27	1075.19	1.2315	9.2334	1.2154	3.988	3.257
188000	182600	818.78	545.63	1082.21	1.1952	8.9664	1.1796	3.843	3.141
189000	183543	822.08	548.93	1089.17	1.1603	8.7028	1.1451	3.711	3.029
190000	184486	825.31	552.16	1096.07	1.1266- 6	8.4499- 7	1.1118- 9	3.581- 10	2.923- 10
191000	185428	828.49	555.34	1102.90	1.0940	8.2066	1.0797	3.456	2.821
192000	186371	831.61	558.46	1109.68	1.0627	7.9707	1.0488	3.336	2.723
193000	187313	834.67	561.52	1116.40	1.0324	7.7435	1.0189	3.222	2.630
194000	188255	837.67	564.52	1123.06	1.0031	7.5242	9.9803- 10	3.112	2.540
195000	189196	840.62	567.47	1129.67	9.7491- 7	7.3124	9.6216	3.006	2.454
196000	190137	843.51	570.36	1136.21	9.4783	7.1078	9.3524	2.905	2.372
197000	191078	846.35	573.20	1142.71	9.2127	6.9101	9.0922	2.809	2.293
198000	192019	849.14	575.99	1149.14	8.9588	6.7190	8.8404	2.716	2.217
199000	192959	851.87	578.72	1155.52	8.7117	6.5343	8.5970	2.626	2.144
200000	193899	854.56	581.41	1161.85	8.4710- 7	6.3557- 7	8.3628- 10	2.541- 10	2.074- 10
201000	194839	857.20	584.05	1168.13	8.2432	6.1829	8.1355	2.458	2.007
202000	195779	859.78	586.63	1174.35	8.0204	6.0158	7.9150	2.379	1.942
203000	196718	862.32	589.17	1180.52	7.8048	5.8541	7.7028	2.303	1.880
204000	197657	864.82	591.67	1186.64	7.5962	5.6976	7.4984	2.230	1.820
205000	198595	867.26	594.11	1192.71	7.3942	5.5461	7.2975	2.160	1.763
206000	199534	869.67	596.52	1198.73	7.1986	5.3994	7.1045	2.092	1.708
207000	200472	872.02	598.87	1204.70	7.0092	5.2573	6.9175	2.027	1.655
208000	201410	874.34	601.19	1210.62	6.8257	5.1197	6.7364	1.964	1.603
209000	202347	876.61	603.46	1216.49	6.6479	4.9863	6.5610	1.904	1.554
210000	203284	878.84	605.69	1222.31	6.4756- 7	4.8571- 7	6.3910- 10	1.846- 10	1.507- 10
211000	204221	881.03	607.88	1228.09	6.3087	4.7319	6.2262	1.790	1.461
212000	205158	883.18	610.03	1233.82	6.1468	4.6105	6.0664	1.736	1.417
213000	206094	885.29	612.14	1239.50	5.9899	4.4928	5.9115	1.683	1.374
214000	207030	887.36	614.21	1245.14	5.8377	4.3788	5.7613	1.633	1.333
215000	207966	889.39	616.24	1250.73	5.6903	4.2679	5.6156	1.585	1.294
216000	208902	891.39	618.24	1256.27	5.5468	4.1604	5.4743	1.538	1.256
217000	209837	893.35	620.20	1261.77	5.4078	4.0562	5.3371	1.493	1.219
218000	210772	895.27	622.12	1267.23	5.2729	3.9550	5.2040	1.450	1.183
219000	211706	897.16	624.01	1272.65	5.1420	3.8568	5.0748	1.408	1.149
220000	212641	899.01	625.86	1278.02	5.0149- 7	3.7615- 7	4.9494- 10	1.367- 10	1.116- 10
221000	213575	900.83	627.68	1283.34	4.8915	3.6689	4.8276	1.328	1.084
222000	214509	902.62	629.47	1288.63	4.7717	3.5791	4.7093	1.290	1.053
223000	215442	904.37	631.22	1293.87	4.6553	3.4918	4.5944	1.253	1.023
224000	216375	906.09	632.94	1299.07	4.5422	3.4069	4.4828	1.218	9.943- 11
225000	217308	907.78	634.63	1304.23	4.4324	3.3245	4.3744	1.184	9.665
226000	218241	909.44	636.29	1309.35	4.3258	3.2445	4.2690	1.151	9.395
227000	219173	911.07	637.92	1314.43	4.2219	3.1667	4.1666	1.119	9.134
228000	220105	912.67	639.52	1319.47	4.1210	3.0910	4.0671	1.088	8.882
229000	221037	914.24	641.09	1324.47	4.0230	3.0175	3.9704	1.058	8.638

Table 6. The 1976 U.S. Standard Atmosphere (Continued)

Altitude		Temperature			Pressure			Density	
Z (m)	H (m)	T (K)	t (°C)	T_M (K)	P (mb)	P (torr)	P/P_0	ρ (kg/m^3)	ρ/ρ_0
230000	221969	915.78	642.63	1329.43	3.9278E-7	2.9468E-7	3.8783E-10	1.029E-10	8.402E-11
231000	222908	917.29	644.14	1334.35	3.8349	2.8764	3.7846	1.001	8.173
232000	223831	918.78	645.63	1339.23	3.7448	2.8088	3.6956	9.741E-11	7.952
233000	224742	920.24	647.09	1344.07	3.6571	2.7430	3.6093	9.479	7.738
234000	225642	921.67	648.52	1348.87	3.5718	2.6791	3.5251	9.225	7.530
235000	226622	923.07	649.92	1353.64	3.4888	2.6168	3.4432	8.979	7.329
236000	227552	924.45	651.30	1358.37	3.4080	2.5562	3.3634	8.740	7.135
237000	228481	925.81	652.66	1363.06	3.3294	2.4973	3.2858	8.509	6.946
238000	229411	927.14	653.99	1367.71	3.2529	2.4399	3.2103	8.285	6.764
239000	230340	928.44	655.29	1372.33	3.1784	2.3840	3.1368	8.068	6.586
240000	231268	929.73	656.58	1376.91	3.1059E-7	2.3296E-7	3.0653E-10	7.858E-11	6.415E-11
241000	232197	930.98	657.83	1381.46	3.0353	2.2767	2.9956	7.654	6.248
242000	233125	932.22	659.07	1385.97	2.9665	2.2251	2.9277	7.456	6.087
243000	234053	933.43	660.28	1390.45	2.8996	2.1749	2.8616	7.265	5.930
244000	234980	934.62	661.47	1394.88	2.8343	2.1259	2.7973	7.079	5.779
245000	235906	935.79	662.64	1399.29	2.7708	2.0783	2.7346	6.898	5.631
246000	236835	936.94	663.79	1403.66	2.7089	2.0318	2.6735	6.723	5.488
247000	237761	938.07	664.92	1407.99	2.6486	1.9866	2.6139	6.553	5.350
248000	238688	939.18	666.03	1412.30	2.5899	1.9425	2.5559	6.388	5.215
249000	239614	940.26	667.11	1416.56	2.5325	1.8995	2.4994	6.228	5.084
250000	240540	941.33	668.18	1420.80	2.4767-7	1.8577-7	2.4443-10	6.073-11	4.957-11
251000	241466	942.38	669.23	1425.00	2.4222	1.8168	2.3906	5.922	4.834
252000	242391	943.41	670.26	1429.16	2.3692	1.7770	2.3382	5.775	4.714
253000	243316	944.42	671.27	1433.30	2.3175	1.7382	2.2871	5.633	4.598
254000	244241	945.41	672.26	1437.40	2.2670	1.7004	2.2374	5.494	4.485
255000	245165	946.38	673.23	1441.47	2.2178	1.6635	2.1888	5.360	4.375
256000	246089	947.34	674.19	1445.51	2.1698	1.6275	2.1415	5.229	4.268
257000	247013	948.28	675.13	1449.51	2.1230	1.5924	2.0953	5.102	4.165
258000	247937	949.20	676.05	1453.49	2.0774	1.5582	2.0502	4.979	4.064
259000	248860	950.10	676.95	1457.43	2.0328	1.5248	2.0063	4.859	3.967
260000	249784	950.99	677.84	1461.34	1.9894-7	1.4922-7	1.9634-10	4.742-11	3.871-11
261000	250706	951.86	678.71	1465.22	1.9470	1.4604	1.9215	4.629	3.779
262000	251629	952.72	679.57	1469.07	1.9056	1.4293	1.8807	4.519	3.689
263000	252551	953.56	680.41	1472.89	1.8652	1.3990	1.8408	4.412	3.601
264000	253473	954.39	681.24	1476.68	1.8258	1.3695	1.8020	4.307	3.516
265000	254395	955.20	682.05	1480.44	1.7874	1.3406	1.7640	4.206	3.433
266000	255316	955.99	682.84	1484.17	1.7498	1.3125	1.7269	4.107	3.353
267000	256237	956.78	683.63	1487.87	1.7131	1.2850	1.6907	4.011	3.274
268000	257158	957.54	684.39	1491.54	1.6773	1.2581	1.6554	3.918	3.198
269000	258079	958.30	685.15	1495.18	1.6424	1.2319	1.6209	3.827	3.124
270000	258999	959.04	685.89	1498.80	1.6083-7	1.2063-7	1.5872-10	3.738-11	3.054-11
271000	259919	959.77	686.62	1502.38	1.5749	1.1813	1.5543	3.652	2.981
272000	260839	960.48	687.33	1505.94	1.5424	1.1569	1.5222	3.568	2.913
273000	261758	961.18	688.03	1509.46	1.5106	1.1330	1.4908	3.486	2.846
274000	262678	961.87	688.72	1512.96	1.4795	1.1097	1.4602	3.407	2.781
275000	263597	962.54	689.39	1516.43	1.4492	1.0870	1.4302	3.329	2.718
276000	264515	963.21	690.06	1519.88	1.4195	1.0647	1.4010	3.254	2.656
277000	265434	963.86	690.71	1523.30	1.3906	1.0430	1.3724	3.180	2.596
278000	266352	964.50	691.35	1526.68	1.3623	1.0218	1.3444	3.108	2.538
279000	267269	965.13	691.98	1530.05	1.3346	1.0010	1.3171	3.039	2.481
280000	268187	965.75	692.60	1533.38	1.3076-7	9.8075-8	1.2905-10	2.971-11	2.425-11
281000	269104	966.35	693.20	1536.69	1.2811	9.6093	1.2644	2.904	2.371
282000	270021	966.95	693.80	1539.98	1.2553	9.4156	1.2389	2.840	2.318
283000	270938	967.53	694.38	1543.23	1.2301	9.2263	1.2140	2.777	2.267
284000	271854	968.11	694.96	1546.47	1.2054	9.0412	1.1896	2.715	2.217
285000	272771	968.67	695.52	1549.67	1.1813	8.8603	1.1658	2.656	2.168
286000	273686	969.22	696.07	1552.85	1.1577	8.6834	1.1426	2.597	2.120
287000	274602	969.77	696.62	1556.01	1.1346	8.5104	1.1199	2.540	2.074
288000	275517	970.30	697.15	1559.14	1.1121	8.3413	1.0975	2.485	2.028
289000	276432	970.83	697.68	1562.24	1.0900	8.1759	1.0758	2.431	1.984
290000	277347	971.34	698.19	1565.32	1.0685-7	8.0141-8	1.0545-10	2.378-11	1.941-11
291000	278262	971.85	698.70	1568.38	1.0474	7.8559	1.0337	2.326	1.899
292000	279176	972.34	699.19	1571.41	1.0267	7.7012	1.0133	2.276	1.858
293000	280090	972.83	699.68	1574.42	1.0066	7.5498	9.9340-11	2.227	1.818
294000	281004	973.31	700.16	1577.41	9.8682-8	7.4018	9.7392	2.179	1.779
295000	281917	973.78	700.63	1580.37	9.6751	7.2569	9.5486	2.133	1.741
296000	282830	974.24	701.09	1583.31	9.4862	7.1152	9.3622	2.087	1.704
297000	283743	974.70	701.55	1586.22	9.3014	6.9766	9.1797	2.043	1.668
298000	284656	975.14	701.99	1589.12	9.1205	6.8410	9.0013	1.999	1.632
299000	285568	975.58	702.43	1591.99	8.9436	6.7082	8.8268	1.957	1.598
300000	286480	976.01	702.86	1594.83	8.7704-8	6.5783-8	8.6557-11	1.914-11	1.564-11	1.499
301000	287391	976.44	703.29	1597.66	8.6011	6.4511	8.4884	1.873	1.530
302000	288302	976.85	703.70	1600.46	8.4351	6.3268	8.3248	1.833	1.497
303000	289212	977.25	704.10	1603.24	8.2729	6.2051	8.1648	1.794	1.465
304000	290122	977.65	704.50	1606.00	8.1139	6.0859	8.0078	1.760	1.437
305000	291031	978.03	704.88	1608.74	7.9581	5.9693	7.8541	1.722	1.406
306000	291940	978.41	705.26	1611.46	7.8061	5.8551	7.7040	1.686	1.378
307000	292848	978.78	705.63	1614.16	7.6572	5.7433	7.5572	1.651	1.349
308000	293756	979.14	706.00	1616.84	7.5112	5.6338	7.4129	1.618	1.321
309000	294663	979.50	706.35	1619.50	7.3685	5.5267	7.2720	1.586	1.295
310000	295570	979.90	706.75	1622.13	7.2285	5.4218	7.1339	1.552	1.267
312000	297383	980.60	707.45	1627.34	6.9574	5.2185	6.8666	1.489	1.216
314000	299220	981.26	708.13	1632.47	6.6975	5.0235	6.6099	1.429	1.167
316000	301035	981.93	708.78	1637.53	6.4484	4.8365	6.3639	1.372	1.120
318000	302850	982.55	709.40	1642.51	6.2098	4.6572	6.1278	1.317	1.075

Table 6. The 1976 U.S. Standard Atmosphere (Continued)

Altitude		Temperature			Pressure			Density	
Z (m)	H (m)	T (K)	t (°C)	T_M (K)	P (mb)	P (torr)	P/P_0	ρ (kg/m^3)	ρ/ρ_0
320000	304603	983.16	710.01	1647.42	5.9796- 8	4.4856- 8	5.9814- 11	1.20e- 11	1.032- 11
322000	306476	983.74	710.59	1652.76	5.7593	4.3199	5.7604	1.214	9.913- 12
324000	308347	984.31	711.16	1657.83	5.5468	4.1613	5.4754	1.18e	9.521
326000	310217	984.85	711.70	1661.73	5.3450	4.0091	5.2751	1.121	9.147
328000	312086	985.37	712.22	1666.37	5.1502	3.8629	5.0828	1.077	8.789
330000	313714	985.88	712.73	1670.94	4.9630	3.7226	4.8981	1.035	8.447
332000	315521	986.37	713.22	1675.45	4.7833	3.5877	4.7207	9.946- 12	8.119
334000	317387	986.84	713.69	1679.90	4.6108	3.4582	4.5503	9.561	7.805
336000	319132	987.29	714.14	1684.29	4.4447	3.3338	4.3865	9.193	7.505
338000	320935	987.73	714.58	1688.62	4.2852	3.2142	4.2292	8.841	7.217
340000	322738	988.15	715.00	1692.90	4.1320- 8	3.0992- 8	4.0779- 11	8.503- 12	6.941- 12
342000	324539	988.56	715.41	1697.13	3.9844	2.9887	3.9325	8.179	6.677
344000	326340	988.96	715.81	1701.30	3.8430	2.8825	3.7927	7.869	6.424
346000	328139	989.34	716.19	1705.42	3.7068	2.7803	3.6583	7.572	6.181
348000	329938	989.70	716.55	1709.48	3.5758	2.6821	3.5290	7.287	5.948
350000	331735	990.06	716.91	1713.49	3.4498	2.5878	3.4047	7.014	5.725
352000	333531	990.40	717.25	1717.45	3.3288	2.4966	3.2851	6.751	5.511
354000	335326	990.73	717.58	1721.46	3.2126	2.4092	3.1700	6.500	5.308
356000	337120	991.05	717.90	1725.36	3.0998	2.3250	3.0592	6.259	5.109
358000	338913	991.35	718.20	1729.22	2.9918	2.2440	2.9526	6.027	4.920
360000	340705	991.65	718.50	1733.05	2.8878- 8	2.1661- 8	2.8501- 11	5.805- 12	4.739- 12
362000	342496	991.94	718.79	1736.84	2.7878	2.0910	2.7513	5.592	4.565
364000	344286	992.21	719.06	1740.59	2.6915	2.0188	2.6563	5.387	4.397
366000	346076	992.48	719.33	1744.32	2.5987	1.9492	2.5647	5.190	4.237
368000	347862	992.74	719.59	1748.02	2.5094	1.8822	2.4766	5.001	4.083
370000	349648	992.98	719.83	1751.68	2.4234	1.8177	2.3917	4.820	3.934
372000	351434	993.22	720.07	1755.32	2.3405	1.7556	2.3099	4.645	3.792
374000	353218	993.44	720.31	1758.94	2.2607	1.6957	2.2312	4.478	3.655
376000	355002	993.66	720.53	1762.52	2.1839	1.6380	2.1553	4.316	3.524
378000	356784	993.89	720.74	1766.10	2.1098	1.5825	2.0822	4.162	3.397
380000	358565	994.10	720.95	1769.66	2.0384- 8	1.5289- 8	2.0117- 11	4.013- 12	3.276- 12
382000	360346	994.30	721.15	1773.19	1.9696	1.4773	1.9439	3.870	3.159
384000	362125	994.50	721.35	1776.71	1.9033	1.4276	1.8784	3.732	3.046
386000	363903	994.68	721.53	1780.22	1.8394	1.3797	1.8153	3.599	2.938
388000	365680	994.86	721.71	1783.72	1.7778	1.3335	1.7545	3.472	2.834
390000	367456	995.04	721.89	1787.20	1.7184	1.2889	1.6959	3.350	2.734
392000	369231	995.21	722.06	1790.66	1.6611	1.2460	1.6394	3.232	2.638
394000	371005	995.37	722.22	1794.15	1.6059	1.2045	1.5849	3.118	2.545
396000	372778	995.53	722.38	1797.61	1.5527	1.1646	1.5324	3.009	2.456
398000	374549	995.68	722.53	1801.08	1.5013	1.1261	1.4817	2.904	2.370
400000	376320	995.83	722.68	1804.54	1.4518- 8	1.0889- 8	1.4326- 11	2.803- 12	2.288- 12
402000	378090	995.97	722.82	1808.00	1.4040	1.0531	1.3856	2.705	2.208
404000	379858	996.10	722.95	1811.47	1.3579	1.0185	1.3401	2.611	2.132
406000	381626	996.23	723.08	1814.94	1.3134	9.8514- 9	1.2962	2.521	2.058
408000	383392	996.36	723.21	1818.41	1.2705	9.5295	1.2539	2.434	1.987
410000	385158	996.49	723.34	1821.90	1.2291	9.2189	1.2130	2.350	1.918
412000	386922	996.60	723.45	1825.39	1.1891	8.9192	1.1736	2.269	1.853
414000	388686	996.72	723.57	1828.90	1.1506	8.6299	1.1355	2.192	1.789
416000	390448	996.83	723.68	1832.42	1.1133	8.3507	1.0988	2.117	1.728
418000	392210	996.94	723.79	1835.96	1.0774	8.0812	1.0633	2.044	1.669
420000	393970	997.04	723.89	1839.52	1.0427- 8	7.8211- 9	1.0291- 11	1.975- 12	1.612- 12
422000	395729	997.14	723.99	1843.09	1.0092	7.5699	9.9605- 12	1.908	1.557
424000	397487	997.24	724.09	1846.69	9.7692- 9	7.3275	9.6414	1.843	1.504
426000	399245	997.33	724.18	1850.31	9.4578	7.0933	9.3333	1.781	1.453
428000	401001	997.42	724.27	1853.96	9.1558	6.8673	9.0350	1.720	1.404
430000	402756	997.50	724.35	1857.63	8.8645	6.6489	8.7466	1.662	1.357
432000	404510	997.59	724.44	1861.33	8.5824	6.4381	8.4712	1.606	1.311
434000	406263	997.67	724.52	1865.07	8.3119	6.2345	8.2032	1.553	1.267
436000	408015	997.75	724.60	1868.83	8.0497	6.0378	7.9444	1.501	1.225
438000	409766	997.82	724.67	1872.64	7.7964	5.8478	7.6944	1.450	1.184
440000	411516	997.90	724.75	1876.48	7.5517- 9	5.6642- 9	7.4529- 12	1.402- 12	1.144- 12
442000	413265	997.97	724.82	1880.36	7.3153	5.4869	7.2196	1.355	1.106
444000	415013	998.03	724.88	1884.26	7.0869	5.3156	6.9942	1.310	1.070
446000	416760	998.10	724.95	1888.20	6.8662	5.1500	6.7766	1.267	1.034
448000	418505	998.16	725.01	1892.17	6.6529	4.9901	6.5659	1.225	9.998- 13
450000	420250	998.22	725.07	1896.17	6.4468	4.8355	6.3625	1.184	9.668
452000	421994	998.28	725.13	1900.21	6.2477	4.6861	6.1660	1.145	9.349
454000	423737	998.34	725.19	1904.28	6.0552	4.5418	5.9760	1.108	9.041
456000	425478	998.40	725.25	1908.39	5.8691	4.4022	5.7924	1.071	8.744
458000	427219	998.45	725.30	1912.54	5.6893	4.2673	5.6149	1.036	8.457
460000	428959	998.50	725.35	1916.73	5.5155- 9	4.1370- 9	5.4434- 12	1.002- 12	8.180- 13
462000	430698	998.55	725.40	1920.97	5.3474	4.0109	5.2775	9.694- 13	7.913
464000	432435	998.60	725.45	1925.22	5.1850	3.8891	5.1172	9.377	7.655
466000	434172	998.65	725.50	1930.74	5.0279	3.7712	4.9621	9.072	7.406
468000	435907	998.69	725.54	1935.32	4.8760	3.6573	4.8123	8.777	7.165
470000	437642	998.73	725.58	1939.97	4.7292	3.5472	4.6673	8.492	6.932
472000	439376	998.78	725.63	1944.78	4.5871	3.4406	4.5272	8.217	6.708
474000	441108	998.82	725.67	1949.58	4.4498	3.3376	4.3916	7.952	6.491
476000	442840	998.86	725.71	1954.37	4.3170	3.2380	4.2605	7.695	6.282
478000	444570	998.89	725.74	1959.32	4.1885	3.1416	4.1337	7.447	6.079

Table 6. The 1976 U.S. Standard Atmosphere (Continued)

Altitude		Temperature			Pressure			Density	
Z (m)	H (m)	T (K)	t (°C)	T_M (K)	P (mb)	P (torr)	P/P_0	ρ (kg/m^3)	ρ/ρ_0
480000	446300	998.93	725.78	1964.36	4.0642- 9	3.0484- 9	4.0111- 12	7.28e- 13	5.86e- 13
482000	448826	998.97	725.82	1969.47	3.9448	2.9583	3.8925	6.976	5.605
484000	449756	999.00	725.85	1974.68	3.8278	2.8711	3.7777	6.753	5.513
486000	451482	999.03	725.88	1979.97	3.7153	2.7867	3.6667	6.537	5.336
488000	453206	999.07	725.92	1985.35	3.6064	2.7051	3.5593	6.328	5.166
490000	454932	999.10	725.95	1990.83	3.5011	2.6261	3.4554	6.127	5.001
492000	456656	999.13	725.98	1996.40	3.3993	2.5497	3.3548	5.932	4.842
494000	458378	999.15	726.00	2002.07	3.3007	2.4757	3.2575	5.743	4.688
496000	460100	999.18	726.03	2007.84	3.2053	2.4041	3.1633	5.561	4.540
498000	461820	999.21	726.06	2013.71	3.1129	2.3349	3.0722	5.385	4.396
500000	463540	999.24	726.09	2019.69	3.0236- 9	2.2679- 9	2.9840- 12	5.215- 13	4.257- 13
505000	467834	999.30	726.15	2035.12	2.8125	2.1096	2.7757	4.814	3.930
510000	472122	999.35	726.20	2051.27	2.6179	1.9636	2.5837	4.446	3.629
515000	476404	999.40	726.25	2068.18	2.4385	1.8290	2.4066	4.107	3.353
520000	480679	999.45	726.30	2085.90	2.2729	1.7048	2.2431	3.796	3.099
525000	484949	999.50	726.35	2104.47	2.1200	1.5901	2.0923	3.509	2.865
530000	489212	999.54	726.39	2123.94	1.9789	1.4843	1.9530	3.246	2.650
535000	493469	999.57	726.42	2144.35	1.8485	1.3865	1.8244	3.003	2.452
540000	497719	999.61	726.46	2165.75	1.7281	1.2962	1.7055	2.780	2.269
545000	501964	999.64	726.49	2188.18	1.6167	1.2126	1.5956	2.574	2.101
550000	506202	999.67	726.52	2211.70	1.5137- 9	1.1354- 9	1.4939- 12	2.384- 13	1.946- 13
555000	510435	999.69	726.54	2236.35	1.4184	1.0639	1.3999	2.210	1.804
560000	514661	999.72	726.57	2262.16	1.3303	0.9778- 10	1.3129	2.049	1.672
565000	518881	999.74	726.59	2289.24	1.2486	0.3692	1.2323	1.900	1.551
570000	523095	999.76	726.61	2317.58	1.1729	0.7978	1.1576	1.763	1.439
575000	527303	999.78	726.63	2347.24	1.1028	0.2719	1.0884	1.637	1.336
580000	531505	999.80	726.65	2378.28	1.0378	7.7843	1.0243	1.520	1.241
585000	535701	999.81	726.66	2410.74	9.7752- 10	7.3320	9.6473- 13	1.413	1.153
590000	539890	999.83	726.68	2444.66	9.2155	6.9122	9.0950	1.313	1.072
595000	544074	999.84	726.69	2480.10	8.6958	6.5224	8.5821	1.221	9.971- 14
600000	548252	999.85	726.70	2517.10	8.2130- 10	6.1602- 10	8.1056- 13	1.137- 13	9.279- 14
605000	552424	999.86	726.71	2555.69	7.7642	5.8236	7.6626	1.058	8.640
610000	556589	999.88	726.73	2595.92	7.3466	5.5108	7.2507	9.859- 14	8.048
615000	560749	999.88	726.73	2637.82	6.9585	5.2193	6.8675	9.190	7.502
620000	564903	999.89	726.74	2681.43	6.5969	4.9481	6.5105	8.571	6.996
625000	569051	999.90	726.75	2726.78	6.2601	4.6954	6.1782	7.998	6.529
630000	573193	999.91	726.76	2773.89	5.9461	4.4608	5.8684	7.468	6.096
635000	577329	999.92	726.77	2822.79	5.6533	4.2404	5.5794	6.977	5.695
640000	581459	999.92	726.77	2873.49	5.3801	4.0354	5.3097	6.523	5.325
645000	585583	999.93	726.78	2926.01	5.1249	3.8440	5.0579	6.102	4.981
650000	589701	999.93	726.78	2980.36	4.8865- 10	3.6651- 10	4.8226- 13	5.712- 14	4.663- 14
655000	593814	999.94	726.79	3036.53	4.6635	3.4979	4.6025	5.350	4.368
660000	597920	999.94	726.79	3094.52	4.4549	3.3414	4.3966	5.015	4.094
665000	602021	999.95	726.80	3154.32	4.2595	3.1949	4.2038	4.704	3.840
670000	606116	999.95	726.80	3215.92	4.0765	3.0576	4.0232	4.416	3.605
675000	610205	999.96	726.81	3279.29	3.9048	2.9289	3.8534	4.148	3.386
680000	614288	999.96	726.81	3344.39	3.7438	2.8081	3.6940	3.900	3.183
685000	618365	999.96	726.81	3411.19	3.5925	2.6946	3.5455	3.669	2.995
690000	622437	999.97	726.82	3479.66	3.4504	2.5880	3.4052	3.454	2.820
695000	626503	999.97	726.82	3549.69	3.3166	2.4877	3.2733	3.255	2.657
700000	630563	999.97	726.82	3621.27	3.1908- 10	2.3933- 10	3.1491- 13	3.070- 14	2.506- 14
705000	634617	999.97	726.82	3694.31	3.0722	2.3043	3.0320	2.897	2.365
710000	638666	999.97	726.82	3768.74	2.9604	2.2205	2.9217	2.736	2.234
715000	642709	999.98	726.83	3844.47	2.8549	2.1414	2.8176	2.587	2.112
720000	646746	999.98	726.83	3921.41	2.7553	2.0666	2.7193	2.448	1.998
725000	650778	999.98	726.83	3999.46	2.6611	1.9960	2.6263	2.318	1.892
730000	654803	999.98	726.83	4078.53	2.5720	1.9292	2.5384	2.197	1.793
735000	658824	999.98	726.83	4158.49	2.4877	1.8659	2.4551	2.084	1.701
740000	662838	999.98	726.83	4239.25	2.4077	1.8059	2.3762	1.979	1.615
745000	666847	999.99	726.84	4320.67	2.3319	1.7491	2.3014	1.880	1.535
750000	670850	999.99	726.84	4402.64	2.2599- 10	1.6951- 10	2.2303- 13	1.788- 14	1.460- 14
755000	674848	999.99	726.84	4485.04	2.1915	1.6438	2.1628	1.702	1.390
760000	678840	999.99	726.84	4567.74	2.1265	1.5950	2.0987	1.622	1.324
765000	682826	999.99	726.84	4650.61	2.0645	1.5485	2.0375	1.547	1.262
770000	686807	999.99	726.84	4733.52	2.0056	1.5043	1.9793	1.476	1.205
775000	690782	999.99	726.84	4816.36	1.9493	1.4621	1.9238	1.410	1.151
780000	694751	999.99	726.84	4898.99	1.8957	1.4219	1.8709	1.348	1.100
785000	698715	999.99	726.84	4981.29	1.8444	1.3834	1.8203	1.290	1.053
790000	702674	999.99	726.84	5063.15	1.7954	1.3467	1.7719	1.235	1.008
795000	706627	999.99	726.84	5144.44	1.7485	1.3115	1.7256	1.184	9.666- 15
800000	710574	999.99	726.84	5225.06	1.7036- 10	1.2778- 10	1.6813- 13	1.136- 14	9.272- 15
805000	714516	999.99	726.84	5304.89	1.6606	1.2456	1.6389	1.091	8.902
810000	718452	999.99	726.84	5383.84	1.6193	1.2146	1.5982	1.048	8.554
815000	722383	1000.00	726.85	5461.80	1.5797	1.1849	1.5591	1.008	8.225
820000	726309	1000.00	726.85	5538.70	1.5417	1.1564	1.5215	9.697- 15	7.916
825000	730229	1000.00	726.85	5614.45	1.5051	1.1289	1.4854	9.339	7.624
830000	734143	1000.00	726.85	5688.96	1.4699	1.1025	1.4507	9.001	7.348
835000	738052	1000.00	726.85	5762.18	1.4360	1.0771	1.4172	8.682	7.087
840000	741956	1000.00	726.85	5834.03	1.4034	1.0526	1.3850	8.380	6.841
845000	745854	1000.00	726.85	5904.47	1.3719	1.0290	1.3539	8.094	6.607

Table 6. The 1976 U.S. Standard Atmosphere (Concluded)

Altitude		Temperature			Pressure			Density	
Z (m)	H (m)	T (K)	t (°C)	T_M (K)	P (mb)	P (torr)	P/P_0	ρ (kg/m^3)	ρ/ρ_0
850000	749747	1000.00	726.85	5973.45	1.3415- 10	1.0062- 10	1.3240- 13	7.824- 15	6.387- 15
855000	753634	1000.00	726.85	6048.43	1.3122	9.8420- 11	1.2950	7.567	6.177
860000	757516	1000.00	726.85	6106.47	1.2838	9.6295	1.2670	7.324	5.978
865000	761393	1000.00	726.85	6171.24	1.2564	9.4240	1.2400	7.093	5.790
870000	765264	1000.00	726.85	6234.03	1.2299	9.2251	1.2138	6.873	5.611
875000	769130	1000.00	726.85	6295.22	1.2043	9.0327	1.1885	6.664	5.440
880000	772991	1000.00	726.85	6354.81	1.1794	8.8463	1.1640	6.465	5.278
885000	776846	1000.00	726.85	6412.78	1.1553	8.6657	1.1402	6.276	5.123
890000	780696	1000.00	726.85	6469.15	1.1320	8.4905	1.1172	6.096	4.976
895000	784541	1000.00	726.85	6523.92	1.1093	8.3206	1.0948	5.924	4.836
900000	788380	1000.00	726.85	6577.11	1.0873- 10	8.1556- 11	1.0731- 13	5.759- 15	4.701- 15
905000	792214	1000.00	726.85	6628.72	1.0660	7.9954	1.0520	5.602	4.573
910000	796043	1000.00	726.85	6678.78	1.0452	7.8398	1.0316	5.452	4.451
915000	799866	1000.00	726.85	6727.31	1.0250	7.6885	1.0116	5.308	4.333
920000	803685	1000.00	726.85	6774.34	1.0054	7.5414	9.9229- 14	5.170	4.221
925000	807498	1000.00	726.85	6819.90	9.8635- 11	7.3982	9.7345	5.038	4.113
930000	811305	1000.00	726.85	6864.02	9.6777	7.2589	9.5512	4.912	4.010
935000	815108	1000.00	726.85	6906.73	9.4968	7.1232	9.3727	4.790	3.910
940000	818905	1000.00	726.85	6948.07	9.3207	6.9911	9.1988	4.673	3.815
945000	822697	1000.00	726.85	6988.07	9.1490	6.8623	9.0293	4.561	3.723
950000	826484	1000.00	726.85	7026.78	8.9816- 11	6.7368- 11	8.8642- 14	4.453- 15	3.635- 15
955000	830266	1000.00	726.85	7064.22	8.8184	6.6143	8.7031	4.349	3.550
960000	834043	1000.00	726.85	7100.45	8.6592	6.4949	8.5460	4.248	3.468
965000	837814	1000.00	726.85	7135.49	8.5039	6.3784	8.3927	4.152	3.389
970000	841580	1000.00	726.85	7169.40	8.3523	6.2647	8.2431	4.058	3.313
975000	845342	1000.00	726.85	7202.21	8.2043	6.1537	8.0970	3.968	3.239
980000	849098	1000.00	726.85	7233.96	8.0597	6.0453	7.9543	3.881	3.168
985000	852849	1000.00	726.85	7264.68	7.9185	5.9393	7.8149	3.797	3.100
990000	856594	1000.00	726.85	7294.43	7.7805	5.8358	7.6788	3.716	3.033
995000	860335	1000.00	726.85	7323.24	7.6458	5.7347	7.5457	3.637	2.969
1000000	864071	1000.00	726.85	7351.15	7.5138- 11	5.6358- 11	7.4155- 14	3.561- 15	2.907- 15

2.6 TRAJECTORY PROPAGATION AND STATE TRANSITION MATRIX COMPUTATION

In this section, various means of trajectory propagation and state transition matrix computation are presented. These are both necessary for the dynamic orbit determination algorithm described in Table 2. In the simplest approach, it is assumed that the satellite is in a true Keplerian orbit around the earth, in which case it is necessary to solve only the two-body initial value problem. This approach is most useful for computation of state transition matrices across short periods of time $t - t_0$. If greater accuracy is desired over longer time periods, then the disturbed equations of motion (27) must be numerically integrated. While there are a number of numerical integration approaches in standard use for astrodynamics, it is the relatively new Bulirsch-Stoer integrator that is used for ISODAE. The Bulirsch-Stoer integrator provides high accuracy at a low computational cost, and it is easy to ascertain the level of internal accuracy. The two-body model is shown in Section 2.6.1, and the Bulirsch-Stoer integrator is shown in Section 2.6.2.

2.6.1 Two-Body Model

In the two-body orbital approximation, all disturbing accelerations $\mathbf{a}_d$ in equation (26) are set to $\mathbf{0}$. This approximation is useful over short periods of time (on the order of minutes) when the true satellite orbit will not deviate significantly from the two-body or Keplerian orbit. It is the preferred approach for calculating the state transition matrix from the time of a signal emanation at a satellite to the time of that signal's arrival at a ground station, as required for the signal transit time computation algorithm described in

Table 3, where it may be necessary to calculate a state transition matrix over a period of T on the order of only one second.

The two-body orbit has the advantage of having an analytical solution, which eliminates the need for numerical integration and interpolation between time steps. The solution to equation (26) with $\mathbf{a}_d = \mathbf{0}$ is well documented [14, 15], and so the solution will only be reproduced here. The state vector $\mathbf{x}$ is partitioned into position ($\mathbf{r}$), velocity ($\mathbf{v}$), and non-dynamic components ($\mathbf{s}$), such as clock offsets, so that $\mathbf{x} = [\mathbf{r} \quad \mathbf{v} \quad \mathbf{s}]^{\mathrm{T}}$. Then the state transition matrix $\Phi_0(t)$ from time t_0 to time t will be given in terms of the Lagrange coefficients F, G, F_t, and G_t from the following relationships:

$$\mathbf{r}(t) = F\mathbf{r}(t_0) + G\mathbf{v}(t_0), \qquad \mathbf{v}(t) = F_t\mathbf{r}(t_0) + G_t\mathbf{v}(t_0), \qquad \text{and} \qquad \mathbf{s}(t) = \mathbf{s}(t_0)$$

Properly, the 6×6 state transition matrix for the position and velocity components of the state vector derives from the following:

$$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} F & 0 & 0 & G & 0 & 0 \\ 0 & F & 0 & 0 & G & 0 \\ 0 & 0 & F & 0 & 0 & G \\ F_t & 0 & 0 & G_t & 0 & 0 \\ 0 & F_t & 0 & 0 & G_t & 0 \\ 0 & 0 & F_t & 0 & 0 & G_t \end{bmatrix}}_{\Phi_0(t)} \begin{bmatrix} x(t_0) \\ y(t_0) \\ z(t_0) \\ \dot{x}(t_0) \\ \dot{y}(t_0) \\ \dot{z}(t_0) \end{bmatrix} \tag{36}$$

Thus, the 6×6 state transition matrix may be written in the following partitioned form, where $\mathbf{I}$ is the 3×3 identity matrix:

$$\Phi_0(t) = \left[\begin{array}{c|c} F\mathbf{I} & G\mathbf{I} \\ \hline F_t\mathbf{I} & G_t\mathbf{I} \end{array}\right]$$

Summarized in Table 7 are the steps required to compute the Lagrange coefficients F, G, F_t, and G_t. Let $\mathbf{r}_0 = \mathbf{r}(t_0)$, $\mathbf{v}_0 = \mathbf{v}(t_0)$, $r_0 = |\mathbf{r}_0|$, and $v_0 = |\mathbf{v}_0|$, all of which are known, and let μ be the previously defined gravitational constant. The initial time t_0 and the final time t are also known.

To demonstrate the level of accuracy as a function of time that is achieved with the two-body solution, two-body ephemerides were compared with ephemerides derived from eighth-order Runge-Kutta integration of the fully perturbed equations of motion. A geosynchronous orbit was selected as a representative satellite trajectory for this comparison. The perturbed orbit was computed by the Artificial Satellite Analysis Program (ASAP) [11] developed at the Jet Propulsion Laboratory (JPL) for high-accuracy satellite trajectory propagation. ASAP propagates satellite trajectories by numerically integrating the disturbed equations of motion (27) with an eighth-order Runge-Kutta integrator. ASAP was validated against tracking data by JPL and has been shown to provide high ephemerides prediction accuracy [11]. For all trajectory propagations that were performed on ASAP, the upper accuracy bound on the eighth-order Runge-Kutta integrator was kept at the level of 10^{-12} km. Also, earth's gravitational harmonics to tenth degree and order as given by the Air Force Space Command standard WGS-84 earth gravity model [7] were used. Solar radiation pressure and third-body gravity were included in the perturbing force model.

Table 7. Computation of the Lagrange Coefficients

1. Compute the angular momentum vector, **h**, and the parameter, p, as follows:

$$\mathbf{h} = \mathbf{r}_0 \times \mathbf{v}_0 \qquad p = \frac{\mathbf{h} \cdot \mathbf{h}}{\mu}$$

2. Compute the semi-major axis, a, and eccentricity, e, of the two-body orbit:

$$a = \left(\frac{2}{r_0} - \frac{v_0^2}{\mu} \right)^{-1} \qquad e = \sqrt{1 - \frac{p}{a}}$$

3. Compute the eccentric anomaly, E, at time t_0:

$$E(t_0) = \arccos\left(\frac{a - r_0}{ae} \right)$$

Here, $E(t_0) > 180°$ if $\mathbf{r}_0 \cdot \mathbf{v}_0 < 0$.

4. Compute the mean anomaly, M, at time t_0:

$$M(t_0) = E(t_0) - e \sin E(t_0)$$

(Continued)

Table 7. Computation of the Lagrange Coefficients (Continued)

5. Compute the mean anomaly at time t:

$$M(t) = M(t_0) + \sqrt{\frac{\mu}{a^3}}(t - t_0) \tag{37}$$

6. Compute the eccentric anomaly at time t by solving Kepler's transcendental equation for $E(t)$:

$$M(t) = E(t) - e\sin E(t) \tag{38}$$

This is the one step in the solution of the two-body problem where numerical work is required. It can be shown that Kepler's equation (38) has a unique solution E for any given values of M and e, $0 \leq e < 1$. The Newton's method iteration described in Table 8 provides a readily convergent means of solving Kepler's equation (38). In that iteration, $M(t)$ is held constant at the value calculated in equation (37). Let $E(t)$ be the final estimate of the eccentric anomaly from the Newton's method iteration.

7. Compute the magnitude of the position vector $\mathbf{r}$ at time t:

$$r = a[1 - e\cos E(t)]$$

(Continued)

Table 7. Computation of the Lagrange Coefficients (Concluded)

8. Compute the value of the transfer angle θ:

$$\theta = 2\arctan\left[\sqrt{\frac{1+e}{1-e}}\tan\tfrac{1}{2}E(t)\right] - 2\arctan\left[\sqrt{\frac{1+e}{1-e}}\tan\tfrac{1}{2}E(t_0)\right]$$

9. Compute the F, G, F_t, and G_t coefficients:

$$F = 1 - \frac{r}{p}(1-\cos\theta) \qquad G = \frac{r_0 r}{\sqrt{\mu p}}\sin\theta$$

$$F_t = \frac{\sqrt{\mu}}{r_0 p}\left[\frac{\mathbf{r}_0 \cdot \mathbf{v}_0}{\sqrt{\mu}}(1-\cos\theta) - \sqrt{p}\sin\theta\right] \qquad G_t = 1 - \frac{r_0}{p}(1-\cos\theta) \tag{39}$$

The GEO satellite used for this accuracy assessment was DSCS III with a subsatellite longitude of 52°W. The initial position and velocity vectors for this satellite were taken from an Inter-Range Operation Number (IRON) report from Falcon Air Force Base, which is responsible for determining orbits from tracking data and providing users with ephemeris reports on many Air Force satellites. At 0000 hours UTC on 28 July 1990 in the true-equator mean-equinox of date ECI coordinate frame used in the IRON reports, DSCS III at 52° W had the following position and velocity vectors:

$$\mathbf{r} = \begin{bmatrix} -12{,}413.448 \\ -40{,}299.104 \\ -26.049 \end{bmatrix} \text{km} \qquad \mathbf{v} = \begin{bmatrix} 2937.997 \\ -905.654 \\ 2.431 \end{bmatrix} \text{m} \cdot \text{sec}^{-1}$$

Table 8. Newton's Method Iteration for Solving Kepler's Equation

1. Let $i = 0$ be the initial iteration number, and let $E_0 = 0$ be the initial estimate of the eccentric anomaly transit time t. Choose a threshold δE for quitting the iteration.

2. Compute the next estimate of the eccentric anomaly from

$$E_{i+1} = E_i + \frac{e \sin E_i - E_i + M(t)}{1 - e \cos E_i} \tag{40}$$

3. Quit the iteration if $|E_{i+1} - E_i| < \delta E$. Otherwise, let $i = i + 1$, and repeat at step 2.

The DSCS III position and velocity vectors were propagated in ASAP with the following additional inputs for solar radiation pressure modeling taken from reference [16]: reflectivity coefficient of 1.5, satellite wet mass of 1090 kg, and effective spacecraft area of 17 m^2. Figure 5 shows, as a function of time on two different scales, the magnitude of the difference in the position vectors between the two-body orbit prediction and the reference perturbed position vectors generated by ASAP. Figure 6 shows the velocity vector error in the two-body orbit as a function of time for the same GEO trajectory.

As shown in Figure 5, the error in the two-body GEO position vector propagates at approximately 8 km per day for the a period up to one week from epoch. However,

within the first hour of prediction, the two-body orbit comes within 275 meters of the perturbed satellite position vector. For many applications, where differential range measurements are taken within an hour, and where desired orbital position accuracy is on the order of tens or hundreds of kilometers, it may be sufficient to use the two-body orbit for trajectory propagation and state transition matrix computation. Also, as shown in Figure 6, the error in the two-body GEO velocity vector propagates at less than 0.7 $m \cdot sec^{-1}$ per day. Again, for many applications, this level of error may well be acceptable.

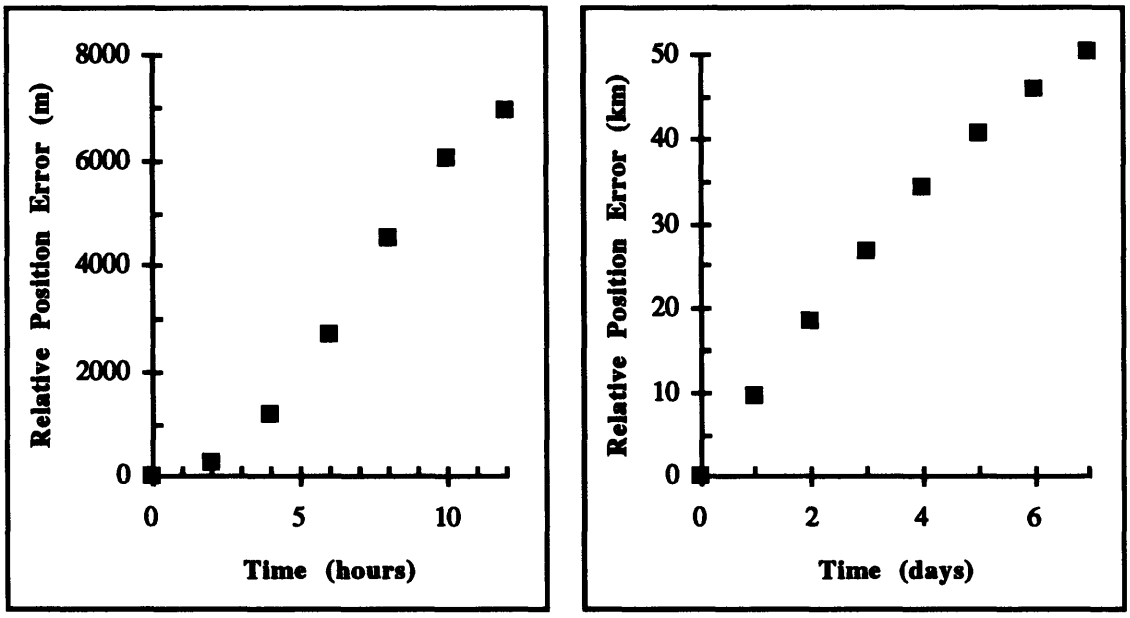

Figure 5. Relative Position Vector Error vs. Time for a GEO Satellite

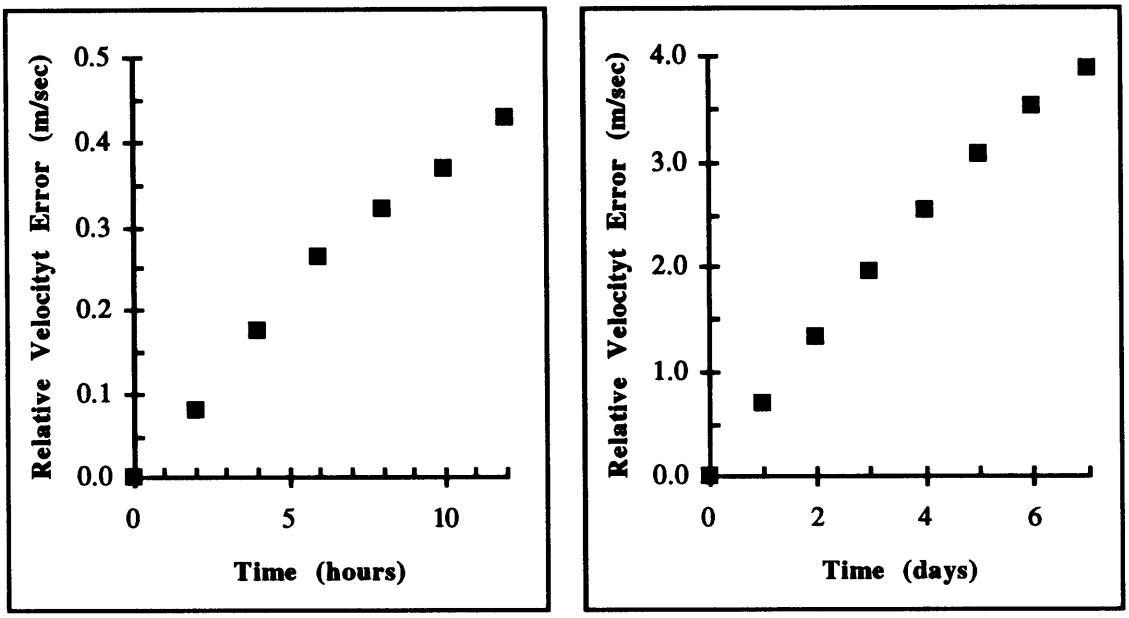

Figure 6. Relative Velocity Vector Error vs. Time for a GEO Satellite

2.6.2 Bulirsch-Stoer Integration of the Disturbed Equations of Motion

The three most popular numerical integrator types for astrodynamic applications are one-step (e.g., Runge-Kutta), multi-step (e.g., predictor-corrector), and extrapolation (e.g., Bulirsch-Stoer). As reported in reference [17], the Bulirsch-Stoer integrator provides high accuracy for numerical integration of Keplerian motion. As will be shown herein for the case of a representative two-body GEO orbit, the Bulirsch-Stoer integrator does provide high accuracy at a relatively low computation cost.

A common numerical integration method is the one-step approach, such as Runge-Kutta, where the interval of integration is divided into small subintervals, and series approximations of the differential equations are made. One-step methods have historically been in common use in astrodynamic applications, but they suffer from several problems. For a low-order technique, the step size must be made fairly small in order to attain a high level of accuracy; however, with a small step size, numerical errors accrue more quickly. Moreover, most one-step techniques are fairly computationally intensive because of the large number of derivative evaluations that are necessary.

The fundamental idea behind the Bulirsch-Stoer technique is to treat the solution of the differential equation as a function of the step size in a one-step integrator. A low-order one-step integrator is used with a sequence of large step sizes, and the results of the integration are fit to a rational function of the step size. A good rational function fit can usually be obtained with a sequence of only four or five one-step integrations (each of which contains a relatively small number of steps, usually 12 at most). The rational

function is then extrapolated to a step size of zero, and the result is the Bulirsch-Stoer solution to the numerical integration.

At the heart of a Bulirsch-Stoer numerical integrator is a one-step integrator, which should be better than first-order, but not so complex that it requires extensive computation. Particularly powerful convergence properties are obtained if the error series of the one-step integrator is even in the step size. Gragg proposed a method that has become known as the modified midpoint method, or Gragg's method [18]. In the initial value problem for ordinary differential equations (ODEs), the given system of differential equations

$$\frac{d}{dt}\mathbf{y} = \mathbf{f}(t, \mathbf{y}) \tag{41}$$

is to be integrated from $t = t_0$ to $t = t_0 + H$, with given initial conditions $\mathbf{y}(t_0) = \mathbf{y}_0$. The solution to the initial value problem is the value of the dependent variable vector at the final time, $\mathbf{y}(t_0 + H)$. Notice that all higher-order ordinary differential equations can be expressed in the form of equation (41) by defining elements of the vector y to be derivatives of the higher-order derivatives. For example, the equations of disturbed motion (26) were initially written in second-order form and then rewritten in first-order form equivalent to (41) with the definition

$$\mathbf{y} = \begin{bmatrix} \mathbf{r} \\ \mathbf{v} \end{bmatrix}$$

In Gragg's method, the interval of integration $[t_0, t_0 + H]$ is divided into n equal subintervals of width $h = H/n$. The first step is the common Euler's method step, and the subsequent steps are modified midpoint method steps. The algorithm is summarized as follows:

$$\begin{aligned} \mathbf{z}_0 &= \mathbf{y}_0 \\ \mathbf{z}_1 &= \mathbf{z}_0 + h\mathbf{f}(t_0, \mathbf{z}_0) \\ \mathbf{z}_{j+1} &= \mathbf{z}_{j-1} + 2h\mathbf{f}(t_j, \mathbf{z}_j), \quad j = 1, 2, \ldots, n-1 \\ \mathbf{y}(t_0 + H) &\approx \tfrac{1}{2}[\mathbf{z}_n + \mathbf{z}_{n-1} + h\mathbf{f}(t_n, \mathbf{z}_n)] \end{aligned} \tag{42}$$

Gragg demonstrated [18] that the power series expansion for the error of this numerical integrator is even in the step size h.

Bulirsch and Stoer [19, 20] apply Gragg's method (42) repeatedly to differential equations (41) and fit the result $\mathbf{y}(t_0 + H)$ to a rational function of the step size h. For the first Gragg's method integration, the interval $[t_0, t_0 + H]$ is traversed with two steps of size $H/2$. For the second Gragg's method integration, four steps of size $H/4$ are used to cross the interval $[t_0, t_0 + H]$. The sequence of number of steps, $\{n_i\}$, $i = 1, 2, 3, \ldots$, is as follows:

$$\{2, 4, 6, 8, 12, 16, \ldots\}, \text{ where } n_i = 2n_{i-2} \text{ for } i > 3$$

Choosing $n_i = 2^i$ makes the step size too small too quickly [21].

Because the details of the rational function fitting and extrapolation processes, described completely in reference [21], become rather involved and are extrinsic to the purposes of this study, they will simply be summarized here. If the dimension of the dependent variable vector **y** is m, and the current Gragg's method trial number is i, so that n_i steps will be made across the interval $[t_0, t_0 + H]$, then m diagonal rational functions, R_j, $j = 1, 2, ..., m$, are fit to each of the elements of the i Gragg's estimated solution vectors $\mathbf{y}(t_0 + H)$ for the i different step sizes. Let us write the dependent variable vector for the integration of equation (41) in component form:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

A sequence of Gragg's method integrations of equation (41) with varying step sizes, $h = H/n_i$ on the i^{th} iteration of the sequence, is used to estimate y_j as a function of h. In particular, the j^{th} diagonal rational function, R_j, is fit to the j^{th} component of **y**, $y_j(t_0 + H)$, for which there are a sequence of values for varying step sizes h. The form of the diagonal rational function is as follows:

$$R_j(h) = \frac{p_{j,0} + p_{j,1}h + \cdots + p_{j,l}h^l}{q_{j,0} + q_{j,1}h + \cdots + q_{j,l}h^l}, \quad i \text{ odd}, \; l = \frac{i-1}{2} \tag{43}$$

$$R_j(h) = \frac{p_{j,0} + p_{j,1}h + \cdots + p_{j,l}h^l}{q_{j,0} + q_{j,1}h + \cdots + q_{j,l}h^l + q_{j,l+1}h^{l+1}}, \quad i \text{ even}, \; l = \frac{i-2}{2}$$

The details of the fitting process are given in references [19, 20, and 21] and will not be repeated here, but it can be seen that so long as $q_{j,0} \neq 0$, the extrapolated solution to the j^{th} row of the differential equation after the i^{th} Gragg's method integration will be given by

$$y_j(t_0 + H) \approx R_j(0) = \frac{p_{j,0}}{q_{j,0}}$$

The Mathematica implementation of the rational function fit and extrapolation process can be seen in subrouting RZEXTR in Appendix B.

A useful feature of the Bulirsch-Stoer integrator is that so long as the method is convergent, there is a convenient way of checking its internal accuracy. If we denote the i^{th} Bulirsch-Stoer approximation of the solution to the differential equation (i.e., the result of rational function extrapolation after i Gragg's method integrations) as

$$\mathbf{y}_i \approx \mathbf{y}(t_0 + H)$$

then the approximation will have internal accuracy at least on the order $\delta\mathbf{y}_i$, where

$$\delta\mathbf{y}_i = \mathbf{y}_i - \mathbf{y}_{i-1}$$

Thus, for well-behaved differential equations, one may obtain a desired level of accuracy by proceeding down the sequence of Gragg's method step sizes, n_i, until the Bulirsch-Stoer error estimate $\delta\mathbf{y}_i$ becomes sufficiently small.

The ISODAE implementation of the Bulirsch-Stoer integrator is shown Mathematica subroutines MMID, RZEXTR, BSSTEP, and BulirschStoer in Appendix B. This implementation follows that presented in reference [21], with the following additional parameters:

Maximum number of steps to attempt in Gragg's method: 96
(equivalently, a maximum of 11 Gragg's method integrations)

Maximum power of denominator in rational function fit: 3
(equivalently, a maximum of 7 estimates of y)

Step size, H: variable under user control

Internal accuracy level, $|\delta y_i|$: variable under user control

Though the user may select any level of internal integrator accuracy for the Bulirsch-Stoer integrator, it was desired to demonstrate its computational efficiency. As we shall see, for the two-body approximation of a GEO satellite orbit, the Bulirsch-Stoer integrator provides accuracy on the order of 10^{-12} km for an integration step size of $H = 1$ hour and with a series of only $i = 6$ Gragg's method integrations, the last of which has only 16 steps across the one-hour time interval. Since there is one derivative evaluation for each Gragg's method step, plus one evaluation at the end of each Gragg's method integration, a total of only 54 derivative evaluations is necessary to achieve this accuracy. By contrast, to achieve position vector accuracy on the order of 10^{-12} km after one hour with the popular fourth-order Runge-Kutta (RK4) integration of the same two-body GEO

trajectory, a step size of 5 sec is necessary. This was determined empirically with an implementation of the RK4 integrator in Mathematica. The classic RK4 algorithm presented in reference [14] for the integration of differential equation (41) from $t = t_0$ to $t = t_0 + H$, with given initial conditions $\mathbf{y}(t_0) = \mathbf{y}_0$, and with n steps of size $h = H/n$, is as follows:

$$\mathbf{y}_{i+1} = \mathbf{y}_i + \tfrac{1}{6}(\mathbf{k}_0 + 2\mathbf{k}_1 + 2\mathbf{k}_2 + \mathbf{k}_3)h$$

where

$$\mathbf{k}_0 = \mathbf{f}(t_i, \mathbf{y}_i)$$

$$\mathbf{k}_1 = \mathbf{f}\left(t_i + \tfrac{1}{2}h, \mathbf{y}_i + \tfrac{1}{2}h\mathbf{k}_0\right)$$

$$\mathbf{k}_2 = \mathbf{f}\left(t_i + \tfrac{1}{2}h, \mathbf{y}_i + \tfrac{1}{2}h\mathbf{k}_1\right)$$

$$\mathbf{k}_3 = \mathbf{f}(t_i + h, \mathbf{y}_i + h\mathbf{k}_2)$$

and

$$\mathbf{f}(t, \mathbf{y}) = \begin{bmatrix} \mathbf{v} \\ -\frac{\mu}{r^3}\mathbf{r} \end{bmatrix} \qquad \text{where } \mathbf{y} = \begin{bmatrix} \mathbf{r} \\ \mathbf{v} \end{bmatrix}$$

This RK4 algorithm for the two-body initial value problem was implemented in Mathematica. We used as initial conditions a state vector reported by Falcon Air Force Base (AFB) for NATO 3C at 18°W at 0000 hours on 9 February 1990. In the true-equator mean-equinox of date ECI coordinate frame used in the Falcon reports, the position and velocity vectors were as follows:

$$\mathbf{r} = \begin{bmatrix} -21{,}542.98206 \\ 36{,}160.27550 \\ 2697.28210 \end{bmatrix} \text{km} \qquad \mathbf{v} = \begin{bmatrix} -2632.08997 \\ -1579.92061 \\ 154.78188 \end{bmatrix} \text{m}\cdot\text{sec}^{-1}$$

The RK4 integration algorithm required a step size of $h = 5$ sec is necessary to achieve position vector accuracy on the order of 10^{-12} km after one hour [22]. As can be seen from the algorithm, four derivative evaluations per RK4 step are required; thus, a total of 2880 derivative evaluations is necessary in order for RK4 to achieve the same accuracy as Bulirsch-Stoer with only 54 derivative evaluations. While the Bulirsch-Stoer method also requires $i = 6$ rational function extrapolations, each of these requires only $36(i - 1)$ arithmetic operations: $4(i - 1)$ multiplications per dependent variable and $2(i - 1)$ additions per dependent variable, with a total of six dependent variables, in **r** and **v**. Thus, $36(i - 1)$ additional arithmetic operations are necessary for rational function extrapolation after each Gragg's method integration. This amounts to only a moderate amount of computational overhead compared with the derivative evaluations. Thus, for the two-body problem for a GEO satellite, the Bulirsch-Stoer integrator provides the same level of accuracy as the RK4 integrator with at least one order of magnitude less computational effort.

From the initial state for NATO 3C shown above, we calculated the magnitude of the difference in position vectors between the analytical solution to the two-body problem and the solution derived by Bulirsch-Stoer integration of the two-body equations. With working precision on the order of 10^{-20}, and with a total integration interval of five weeks, numerical round-off errors would be many orders of magnitude smaller than

errors due to integrator accuracy. The Bulirsch-Stoer step size was set at 3600 sec, and repeated applications of Bulirsch-Stoer steps were made across the five-week time interval, and on each step, a sequence of the first 6 Gragg's method integrations were used in the rational function extrapolation. Figures 7 show the magnitude of the error in position vector as a function of time on two scales.

The results show that the position error incurred in the first 3600 sec Bulirsch-Stoer step was on the order of 10^{-12} km. It was unexpected that the error did not increase monotonically across the whole interval of integration. Despite the fact that the position error remained on the order of 10^{-9} km for most of the 5-week propagation, the error did reach 2×10^{-8} km by the end of the integration. This is many orders of magnitude more precision than is necessary for many applications, with an implementation of the Bulirsch-Stoer integrator that takes advantage of only the first six Gragg's method integrators in the sequence. We conclude that, to the extent that the two-body initial value problem provides a good model of the behavior of the Bulirsch-Stoer method in the full problem of disturbed satellite motion, this technique of numerical integration provides more than sufficient accuracy at a modest computational expense by comparison with Runge-Kutta techniques.

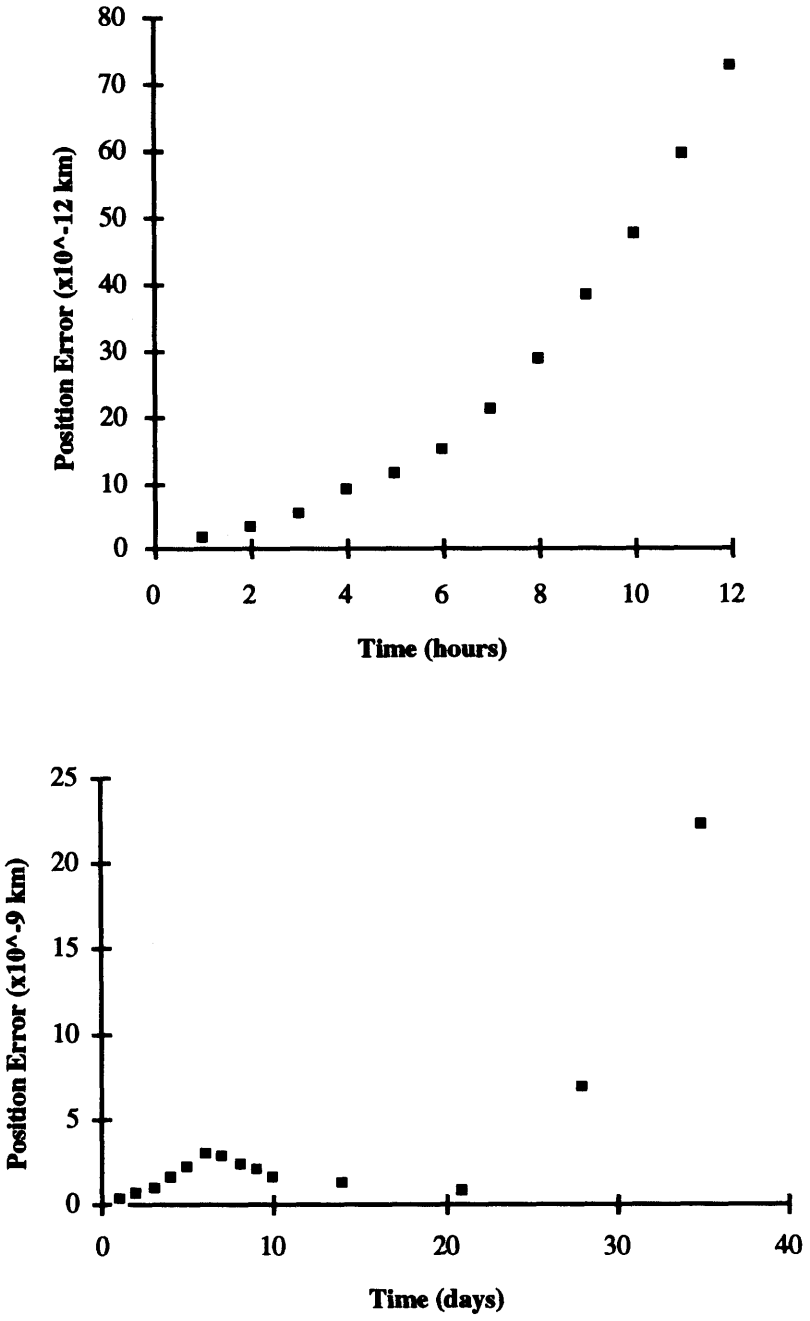

Figure 7. Position Error in NATO 3C Two-Body Trajectory Propagated by Bulirsch-Stoer Integration

SECTION 3

ERROR MODELING

Upon each iteration of the Monte Carlo simulation of the satellite orbit determination process, ISODAE adds random error of user-specified statistical properties where such error occurs. In an actual orbit determination scenario, error sources include inherent observable imprecision, equipment biases, station location uncertainty, and tropospheric and ionospheric delay errors. The current version of ISODAE allows the user to model observable imprecision and equipment biases with independent normal probability distributions. The modeling of clock offsets has already been discussed in Section 2.2. This section describes the generation of normal random variates and presents the statistical properties of interferometric observables. Typical values for the standard deviation of the group delay and phase delay observables due to random measurement error are presented.

3.1 GENERATION OF NORMALLY DISTRIBUTED RANDOM NUMBERS

Mathematica provides the usual pseudorandom 0-1 uniform variates, known as "standard random numbers," and so it is necessary to generate normal random variates from the uniform random variates. After testing the results of various transformations, including Box-Muller [23], from standard random numbers to normally distributed random numbers, the approach that was selected for the best results in Mathemetica was that of Teichroew, reported by Shannon [24]. Table 9 summarizes the process of generating normally random variates.

Table 9. Algorithm for Generating Normally Distributed Random Numbers

1. Generate 12 pseudorandom 0-1 uniform variates, denoted $r_1, r_2, ..., r_{12}$. Then, calculate the number R as follows:

$$R = \frac{1}{4}\left[-6 + \sum_{i=1}^{12} r_i\right]$$

2. Set the following constants:

$$\alpha_1 = 0.029899776$$
$$\alpha_2 = 0.008355968$$
$$\alpha_3 = 0.076542912$$
$$\alpha_4 = 0.252408784$$
$$\alpha_5 = 3.949846138$$

3. Generate a normal random variate, X, with $\mu = 0$ and $\sigma = 1$ as follows:

$$X = \left(\left(\left(\left(\alpha_1 R^2 + \alpha_2\right)R^2 + \alpha_3\right)R^2 + \alpha_4\right)R^2 + \alpha_5\right)R$$

4. Transform X into a normal random variate x with mean μ and standard deviation σ as follows:

$$x = \mu + \sigma X$$

3.2 DIFFERENTIAL RANGE OBSERVABLE PRECISION

Alan Whitney [25] showed that the theoretically achievable precision with the group delay observable, σ_τ, is given by

$$\sigma_\tau = \frac{1}{2\pi(SNR)\Delta\nu} \tag{44}$$

where *SNR* is the signal-to-noise ratio and $\Delta\nu$ is the spanned frequency bandwidth, in Hz, sampled by the interferometer. In the case of TDRS, the downlink to White Sands has an *SNR* of approximately 50 and spans 225 MHz of the Ku-band, of which 20 MHz are sampled by the VLBI system designed by Interferometrics, Inc. [26]. Thus, the theoretically achievable precision of the group delay observable is $\sigma_\tau = 160$ picosec. Differential range could theoretically be measured with 1σ precision of 4.8 cm, or 3σ precision of 14 cm. Preliminary experiments by Interferometrics indicate that their equipment can come within a factor of approximately two of reaching the theoretically achievable group delay observable precision, and so in practice $\sigma_\tau = 320$ picosec, and differential range can be measured with 1σ precision of 9.5 cm.

The theoretically achievable precision with the phase delay observable, σ_ϕ, is given by

$$\sigma_\phi = \frac{1}{2\pi(SNR)\nu} \tag{45}$$

where ν is the center frequency, in Hz, sampled by the interferometer. The TDRS downlink to White Sands is centered at 14 GHz, and so the theoretically achievable precision of the phase delay observable is 0.23 picosec. Differential range could theoretically be measured with 1σ precision of 0.007 cm, or 3σ precision of 0.02 cm. While no TDRS tracking experiments were performed with JPL's CEI equipment at Goldstone, observations were made on natural radio sources at 8.4 GHz to assess the precision of the phase delay observable [27]. The statistical phase error, expressed in radians, is roughly $1/SNR$, and JPL's experiments at Goldstone demonstrated a typical phase error of 0.005 cycles [28]. The achieved SNR was therefore $1/(2\pi\times0.005) \approx 32$. At 8.4 GHz, relationship (45) predicts a phase delay observable precision of 0.59 picosec. JPL demonstrated statistical errors on the phase delay observable to be approximately 1 picosec [27], which is a factor of 1.7 larger than the theoretically achievable value. In this study, the estimated phase delay measurement error on NATO 3C is taken to be on the order of 0.4 picosec, or 1σ differential range precision of 0.01 cm.

In conclusion, relationships (44) and (45) provide the theoretically achievable precision of the group delay and phase delay observables, respectively. In practice, it has been demonstrated that operational VLBI or CEI systems provide observables whose statistical errors are approximately a factor of two larger than the theoretically achievable values. With knowledge of the bandwidth or frequency to be sampled by the interferometer and the signal-to-noise ratio, one may use twice the theoretically achievable observable precision from relationship (44) or (45) as the standard deviation of the group or phase delay observable for ISODAE's error modeling input.

3.3 EQUIPMENT BIASES

The user of ISODAE may simulate equipment biases by specifying a non-zero mean in the measurement error function for a particular observation. With the standard deviation σ specified, as described in Section 3.2 for group delay or phase delay, and with a desired equipment timing bias δ to be modeled, the normal random number generation algorithm in Mathematica (see subroutine "normal" in Appendix B) should be called with the following parameters: mean $c\delta$ and standard deviation $c\sigma$. Here, multiplication by the signal propagation rate c transforms the error function $\varepsilon = c\sigma + c\delta$ into the differential range equivalent error.

SECTION 4

EXAMPLES

In this section, the level of satellite orbit prediction accuracy that can be attained with radio interferometric tracking is demonstrated, and some conclusions about optimal station-satellite geometries are drawn. This was done with sample applications of ISODAE for a geosynchronous satellite orbit. In all subsequent discussion, 1σ position error is measured as the root mean squared of the standard deviations of the errors in the three ECI position vector components of the satellite.

4.1 ISODAE RESULTS FOR A GEOSYNCHRONOUS SATELLITE ORBIT

It was desired to estimate satellite orbit determination accuracy with group or phase delay observables by means of Monte Carlo simulation. To this end, we applied ISODAE to a geosynchronous satellite orbit. (GEO satellites provide an excellent application for radio interferometry because many of the sensors in the Space Surveillance Network (SSN) are radars for which GEO is out of range.) The results indicate that radio interferometry is an accurate means of orbit determination with inherent advantages over more traditional means of satellite tracking.

The GEO satellite state vector and epoch that were selected for study are those for NATO 3C given in Section 2.6.2. Phase delay (i.e., differential range) was measured across three baselines for four ground stations located in a square with edge of length

20 km. In Table 10, the geodetic latitude, longitude, and elevation above sea level are listed for the four stations. The locations are depicted in Figure 8.

Table 10. CEI Station Locations for Sample Scenario

Station Number	Geodetic Latitude (°)	Longitude (°E)	Altitude (km)
1	45.00000	0.0000	0.1
2	45.00000	-0.2545	0.1
3	45.17997	0.0000	0.1
4	45.17997	-0.2545	0.1

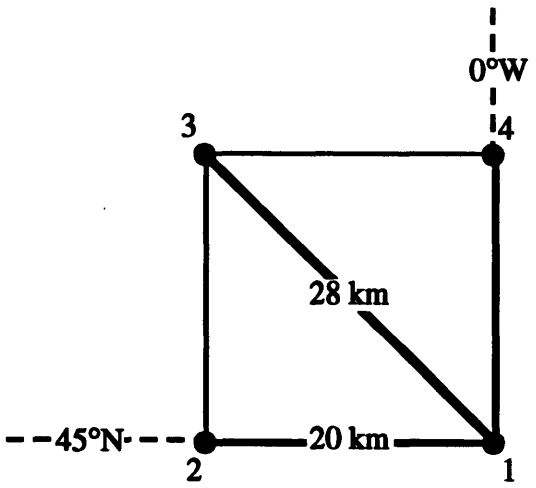

Figure 8. Illustration of CEI Station Locations

Differential range was measured by phase delay simultaneously across the baselines from station 2 to station 1, station 3 to station 1, and station 4 to station 1 (denoted 2-1, 3-1, and 4-1), as illustrated in bold in Figure 8. As discussed in Section 3.2, phase delay in

this scenario can provide group delay equivalent 1σ accuracy of 0.4 picosec, independently normally distributed across each baseline. For this initial study, it was assumed that there were no equipment biases, that all station were connected by fiberoptic cable to one clock and frequency standard, that there were no local oscillator offsets between the four stations, and that station positions were known with perfect accuracy. Thus, the pure effect of measurement geometry and observable precision could be assessed.

ISODAE Monte Carlo simulation of this orbit determination scenario with 200 iterations showed a 1σ position vector accuracy of 3.2 km. The accuracy that can be attained with the use of other sets of baselines was assessed. It is practical to have one station in common for all three measurement pairs so that the common station can act as the correlation center at which the observables are generated. For the particular satellite and ground station locations of this scenario, selection of three measurements where one station is common to each pair (i.e., 2-1, 3-1, 4-1; or 1-2, 3-2, 4-2; or 1-3, 2-3, 4-3; or 1-4, 2-4, 3-4) results in a 1σ position vector accuracy of 3.2 km. Selection of other sets of three measurements (where one station is not common to each measurement pair) would be impractical.

For comparison with a traditional orbit determination scenario, we assessed the position accuracy that can be attained on the same satellite with nine simultaneous measurements by three sensors capable of deep space observations in the Space Surveillance Network. For this purpose, sensors at Eglin AFB (Florida), Millstone (Massachusetts), and Pirinclik (Turkey) were modeled to produce range, elevation angle,

and azimuth angle measurements. Table 11 shows the accuracy of the measurements from these stations.

Table 11. Accuracy of Deep Space Tracking Measurement by the SSN

Sensor	Range Accuracy (km)	Elevation Accuracy (°)	Azimuth Accuracy (°)
Eglin AFB	0.027	0.012	0.012
Millstone	0.044	0.008	0.008
Pirinclik	0.500	0.025	0.027

The result of the Monte Carlo simulation in ISODAE with 200 iterations was position accuracy on the NATO 3C satellite of 3.9 km. Thus, three interferometric measurements across very short baselines produce satellite position vector information with a slightly higher level of accuracy than nine measurements made from geographically dispersed sites in the SSN. It is preferable for operational, economic, and political reasons to carry out satellite tracking across very short baselines. Moreover, radio interferometry offers the advantage of being a purely passive means of orbit determination.

4.2 OPTIMAL INTERFEROMETER SITE GEOMETRY

The starting point for the assessment of the effects of various interferometric measurement geometries on orbit determination accuracy was the tracking scenario for NATO 3C described in Section 4.1. Since it would have been more cumbersome to vary the positions of four ground stations, we instead varied the satellite's initial position vector. (Because the three phase delay observations in this orbit determination scenario are taken simultaneously and only the position vector of the satellite is estimated, the initial velocity vector played no role in this study.)

First, the effect of relative baseline size on orbit determination accuracy by radio interferometry was studied. Range from station 1 was varied while keeping the elevation angle and azimuth angle from that station constant. Because the baseline sizes are small relative to the range to the GEO satellite, the range, elevation angle, and azimuth angle from each of the other three stations is close to that of the first. For this particular orbit determination scenario, range from each site to the satellite is approximately 37,850 km, the elevation angle is approximately 39°, and the azimuth angle is approximately 155°. As shown in Figure 9, the longer the baselines across which phase delay is measured relative to the range to the satellite (or, equivalently, the less the range to the satellite given fixed baselines), the greater the position vector accuracy.

In the original orbit determination scenario from Section 4.1, the satellite's position vector $\mathbf{r}$ is given, the station 1 position vector, $\mathbf{b}_1$, is computed at the measurement time from equation (25), and the position vector of the satellite with respect to station 1, $\mathbf{d}_1$, is

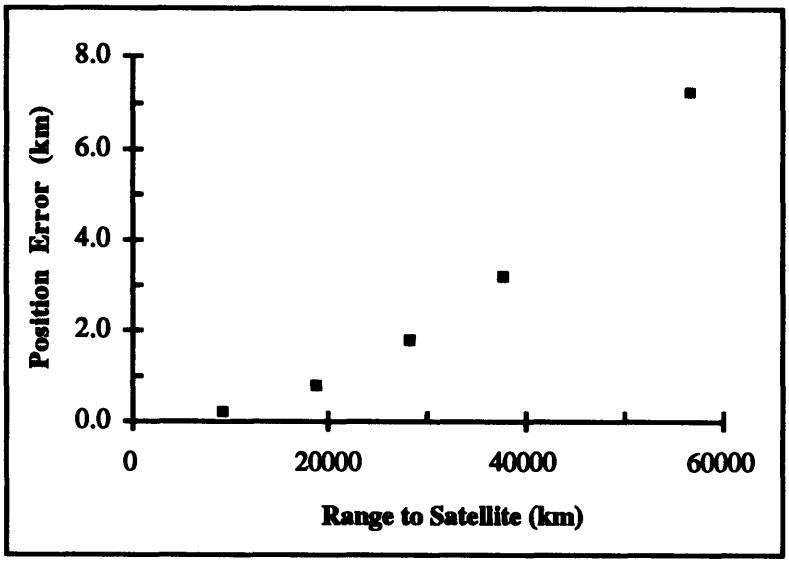

Figure 9. Position Error vs. Range to Satellite

computed as $\mathbf{d}_1 = \mathbf{r} - \mathbf{b}_1$. Selection of a new satellite position vector, $\mathbf{r}^*$, with the same elevation and azimuth angles from station 1, but with a new range d^*, is accomplished as follows:

$$\mathbf{r}^* = \mathbf{b}_1 + \frac{d^*}{|\mathbf{d}_1|}\mathbf{d}_1$$

Next, the effect of satellite elevation angle on orbit determination accuracy by radio interferometry was assessed. The elevation angle of the satellite at station 1 in the original NATO 3C tracking scenario of Section 4.1 was varied while keeping the range and azimuth angle from that station constant. Because of the short baselines relative to the satellite's range, the elevation angles from the other three stations closely match that from station 1. Figure 10 shows the following: position error is minimized for this orbit

determination scenario at the lowest elevation angle, position error increases monotonically with elevation angle, and the orbit determination scenario becomes degenerate when the satellite is at zenith. Again, these results reflect the impact of phase delay measurement error only.

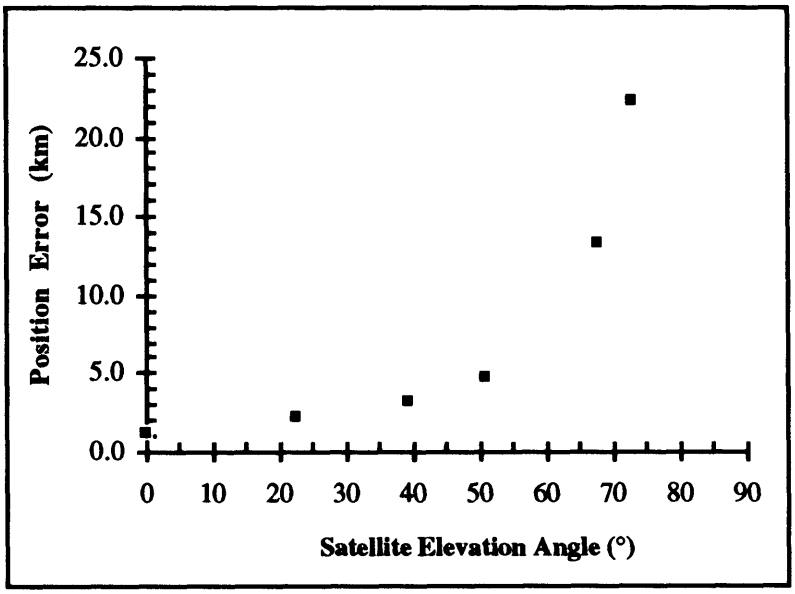

Figure 10. Position Error vs. Satellite Elevation Angle

Selection of a new satellite elevation angle is accomplished in the following manner. Let $\mathbf{n}$ be the unit normal to the plane of ground station 1, as computed in Section 2.4.1. Decompose the range vector $\mathbf{d}_1$ from station 1 to the satellite into components parallel ($\mathbf{d}_{\parallel}$) and perpendicular ($\mathbf{d}_{\perp}$) to $\mathbf{n}$, so that $\mathbf{d}_1 = \mathbf{d}_{\parallel} + \mathbf{d}_{\perp}$. It may be verified that

$$\mathbf{d}_{\parallel} = \frac{\mathbf{d}_1 \cdot \mathbf{n}}{|\mathbf{d}_1|}\mathbf{n} \qquad \text{and} \qquad \mathbf{d}_{\perp} = \mathbf{d}_1 - \frac{\mathbf{d}_1 \cdot \mathbf{n}}{|\mathbf{d}_1|}\mathbf{n}$$

A new station-1-to-satellite range vector $\mathbf{d}'$ with a new elevation angle ε^* and with the same azimuth angle (but with different magnitude from $\mathbf{d}_1$) may be computed as follows:

$$\mathbf{d}' = \mathbf{d}_\perp + \mathbf{d}_\parallel \frac{|\mathbf{d}_\perp|}{|\mathbf{d}_\parallel|} \tan \varepsilon^*$$

The new satellite position vector $\mathbf{r}^*$ such that the elevation angle from station 1 is ε^*, but the range and azimuth angles from station 1 are unchanged, is computed as follows:

$$\mathbf{r}^* = \mathbf{b}_1 + \frac{|\mathbf{d}_1|}{|\mathbf{d}'|} \mathbf{d}'$$

Finally, the effect of satellite azimuth angle on orbit determination accuracy by radio interferometry was assessed. The azimuth angle of the satellite at station 1 in the original NATO 3C tracking scenario of Section 4.1 was varied while keeping the range and elevation angle from that station constant. Again, because of the short baselines relative to the satellite's range, the azimuth angles from the other three stations closely match that from station 1. Figure 11 shows the variation in position accuracy with satellite azimuth angle. The results indicate that for a configuration of four interferometric ground stations at the vertices of a square, position error is minimized when the satellite's azimuth angle is an integer multiple of 45°, and position error is maximized when the satellite's azimuth angle is an integer multiple of 90°.

It may be verified that the new satellite position vector $\mathbf{r}^*$ such that the azimuth angle from station 1 is α^*, but the range and elevation angles from station 1 are unchanged

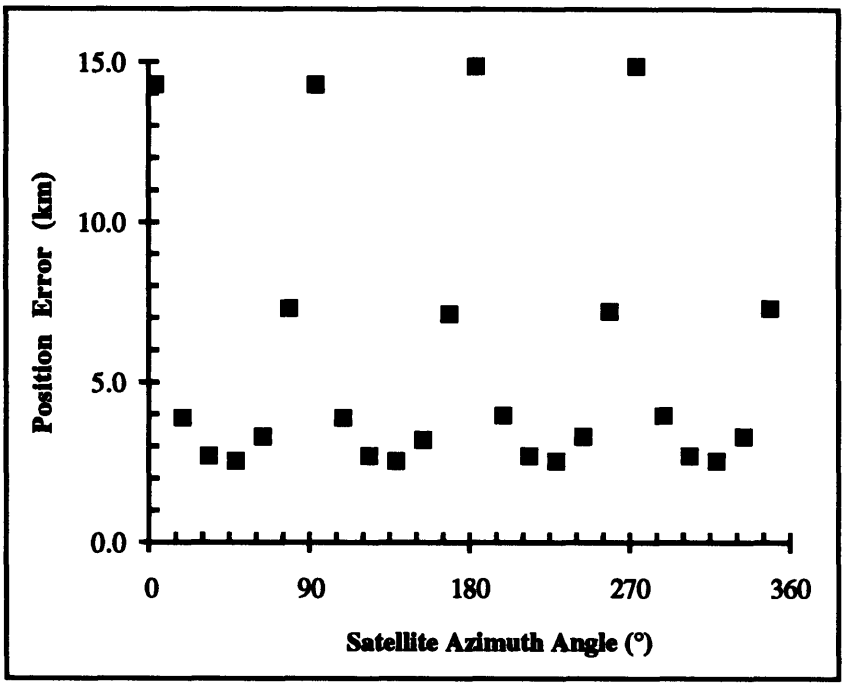

Figure 11. Position Error vs. Satellite Azimuth Angle

from the values determined by the original tracking scenario, is computed as follows:

$$\mathbf{r}^* = \mathbf{b}_1 + \mathbf{d}_{\parallel} + \mathbf{i}_n |\mathbf{d}_{\perp}| \cos\alpha^* + (\mathbf{i}_n \times \mathbf{n}) \frac{|\mathbf{d}_{\perp}|}{|\mathbf{d}_{\parallel}|} (\mathbf{d}_{\perp} \cdot \mathbf{n}) \sin\alpha^*$$

where all values are as described above, and $\mathbf{i}_n$ is the unit vector in the north direction in the plane of ground station 1. The computation of $\mathbf{i}_n$ is given in Appendix A under azimuth angle calculations.

The effects of relative satellite-ground station geometry on the inteferometric tracking of a GEO satellite have been assessed by the ISODAE Monte Carlo tool. In this orbit

determination scenario, phase delay was measured simultaneously across three baselines of four ground stations located at the vertices of a square with edges of length 20 km. For the particular station numbering given in Section 4.1, it was determined that phase delay measurements between the three station pairs 2-1, 3-1, and 4-1 produced a position error of 3.2 km. As expected, longer station baselines increase satellite position estimation accuracy. It was also shown that decreasing the elevation angle to the satellite increases position accuracy. Finally, selecting ground station orientation such that the satellite appears with northwest, northeast, southwest, or southeast azimuth increases position accuracy. These geometrical considerations take into account phase delay measurement error only.

SECTION 5

CONCLUSIONS AND RECOMMENDATIONS

Radio interferometry is an attractive means of satellite orbit determination for several reasons. Interferometric tracking stations can operate passively because measurements can be made on the standard satellite communications downlink. Thus, interferometric tracking may be applied to existing satellite systems. Also, the group or phase delay observable has a very high level of accuracy associated with it. This means that extremely high satellite orbit determination accuracy may be attained across large baselines; but perhaps of more importantance, it means that sufficient orbit determination accuracy is possible with very short baselines, which means that operational tracking systems can be made to be very compact and inexpensive. It has been demonstrated in this study that a connected element interferometry system with 20 km baselines can provide greater orbit determination accuracy on GEO satellites than the geographically diverse and extremely accurate sensors in the Space Surveillance Network. It stands to reason that the potential benefits of the application of radio interferometry to satellite orbit determination merit further study.

In its current form, ISODAE has the following capabilities: ISODAE solves the dynamic batch orbit determination problem; ISODAE allows the user to specify range, range rate, differential range, elevation angle, and azimuth angle measurements; and ISODAE allows the user to model measurement errors, measurement biases, and clock offset errors. With some augmentation, ISODAE can be made to model station location uncertainty, atmospheric effects on signal propagation rate, and errors in the orbit determination process such as numerical integrator errors. Of immediate interest is the

study of atmospheric effects on signal propagation rate. Statistical models of the error in the signal propagation rate through the atmosphere are available [29, 30], and as expected, such error increases as the elevation angle to a satellite decreases. On the other hand, the results of Section 4.2 show that based on geometry alone, lower elevation angles produce less satellite position estimation error. Therefore, a tradeoff between atmospherical and geometrical errors exists and should be studied.

The Monte Carlo simulation developed for this study has the capacity to model a variety of error sources in the orbit determination process. As such, ISODAE is a valuable tool not only for the study of radio interferometry but also for the study of the accuracy of any trajectory determination system. Modified versions of ISODAE have been applied to the problem of determining the accuracy of the satellite ephemerides that can be predicted from the current configuration of the Space Surveillance Network and to the problem of determining the ephemerides prediction accuracy of systems responsible for tracking missiles in a ballistic phase of their trajectory. Because of the potential application of ISODAE to orbit determination scenarios other than the dynamic batch estimator, it was designed in a modular fashion so that with minor modifications, it can assess the accuracy of most trajectory determination algorithms. One potential area for future study is a Kalman filtering system that differentially correct satellite state vectors, such as the scheme employed by the Space Defense Operation Center (SPADOC). There is interest in the space community to estimate the quality of orbital elements in space catalogues (such as SPADOC) for a variety of reasons, including the assessment of probabilities of collision with orbital debris.

Thus, there are a number of potentially useful augmentations or modifications of the capabilities of ISODAE. It will be of interest to assess further the capabilities of radio interferometry for satellite orbit determination with modeling of all error sources and with orbit determination scenarios other than that assessed in Section 4 of this study. Also, modifications of ISODAE could provide a starting point for statistical studies of orbital element quality for the SSN or SPADOC. The potential applications of an element quality study include the development of reasonable specifications for a program such as the anti-satellite (ASAT) weapon and the assessment of the threat of orbital debris.

LIST OF REFERENCES

1. Preston, R. A., et al, 27 October 1972, "Interferometric Observations of an Artificial Satellite," *Science*, Vol. 178, pp. 407-409.

2. Ray, J., et al., October-December 1988, "VLBI Tracking of the TDRS," *The Journal of the Astronautical Sciences*, Vol. 36, No. 4, pp. 347-364.

3. Long, A. C., et al (ed), July 1989, *Goddard Trajectory Determination System (GTDS) Mathematical Theory, Revision 1*, FDD/552-89/001, Greenbelt, MD: Goddard Space Flight Center.

4. Eddy, W. F., et al, 29 August 1990, *GEODYN II Systems Description*, Vol. 1, Lanham, MD: STX Systems Corporation.

5. Wolfram, S., 1988, *Mathematica: A System for Doing Mathematics by Computer*, Wolfram Research Inc.

6. Thompson, A. R., J. M. Moran, and G. W. Swenson, Jr., 1991, *Interferometry and Synthesis in Radio Astronomy*, Malabar, FL: Krieger Publishing Company.

7. Wackernagel, H. B., 5 May 1989, *Proposed Astrodynamic Standards (Second Version)*, DOA Technical Note 89-10, HQ Air Force Space Command, Peterson AFB, CO.

8. Sturms, F. M., 15 January 1971, *Polynomial Expressions for Planetary Equators and Orbit Elements with Respect to the Mean 1950.0 Coordinate System*, NASA Technical Report 32-1508, Jet Propulsion Laboratory, Pasadena, CA.

9. Kaula, W. M., 1966, *Theory of Satellite Geodesy*, Waltham, MA: Blaisdell Publishing Company.

10. Marsh, J. G., et al., October 1989, *The GEM-T2 Gravitational Model*, NASA Technical Memorandum 100746.

11. Kwok, J. H., 20 April 1987, *The Artificial Satellite Analysis Program*, EM 312/87-153, Jet Propulsion Laboratory, Pasadena, CA.

12. Griffin, M. D., and J. R. French, 1991, *Space Vehicle Design*, New York: AIAA.

13. NOAA, NASA, USAF, October 1976, *U.S. Standard Atmosphere, 1976*, Washington, D.C.: U.S. Government Printing Office.

14. Battin, R. H., 1987, *An Introduction to the Mathematics and Methods of Astrodynamics*, New York: AIAA.

15. Bate, R. R., D. D. Mueller, and J. E. White, 1971, *Fundamentals of Astrodynamics*, New York: Dover Publications Inc.

16. Martin, D. H., 31 December 1986, *Communication Satellites, 1958-1988*, El Segundo, CA: The Aerospace Corporation.

17. Fukushima, T., 1989, "Tests of the Extrapolation Method for the Numerical Integration of the Keplerian Motion," *Celestial Mechanics*, Vol. 45.

18. Gragg, W., 1965, "On Extrapolation Algorithms for Ordinary Initial Value Problems," *Numerical Analysis Series B*, Vol. 2.

19. Bulirsch, R., and J. Stoer, 1966, "Numerical Treatment of Ordinary Differential Equations by Extrapolation Methods," *Numerische Mathematik*, Vol. 8, pp. 1–13.

20. Stoer, J., and R. Bulirsch, 1980, *Introduction to Numerical Analysis*, New York: Springer-Verlag Inc.

21. Press, W. H., et al, 1986, *Numerical Recipes*, Cambridge, England: Cambridge University Press.

22. Pavloff, M. S., September 1992, *Analysis of Orbit Prediction Algorithms for the Universal Modem System*, WP-92B0000263V1, Bedford, MA: The MITRE Corporation.

23. Shannon, R. E., 1975, *Systems Simulation: The Art and Science*, Englewood Cliffs, NJ: Prentice-Hall, Inc.

24. Box, G. E. P., and M. E. Muller, 1958, "A note on the Generation of Random Normal Deviates," *Annals of Mathematical Statistics*, Vol. 29, pp. 610-611.

25. Whitney, A. R., January 1974, *Precision Geodesy and Astrometry Via Very Long Baseline Interferometry*, PhD Dissertation at the Massachusetts Institute of Technology, Department of Electrical Engineering.

26. Interferometrics, Inc., 25 January 1991, *TDRS Interferometric Tracking System Field Station Hardware Manual*, Vienna, VA.

27. Edwards, C. D., "Angular Navigation on Short Baselines Using Phase Delay Interferometry," *IEEE Transactions on Instrumentation and Measurement*, Vol. 38, No. 2, April 1989.

28. Edwards, C. D., "Development of Realtime Connected Element Interferometry at the Goldstone Deep Space Communications Complex," AIAA 90-2903, *AIAA/AAS Astrodynamics Conference*, August 1990.

29. Treuhaft, R. N., and G. E. Lanyi, "The Effect of Wet Troposphere on Radio Interferometric Measurements," *Radio Science*, Vol. 22, No. 2, March-April 1987, pp. 251-265.

30. Edwards, C. D., "The Effect of Spatial and Temporal Wet-Troposphere Fluctuations on Connected Element Interferometry," TDA Progress Report 42-97, January-March 1989.

APPENDIX A

MEASUREMENT FUNCTIONS AND GEOMETRICAL PARTIAL DERIVATIVES FOR OTHER OBSERVABLE TYPES

In this appendix, the measurement functions and geometrical partial derivatives for range, range rate, elevation angle, and azimuth angle are presented. These measurement types are available in ISODAE, and their mathematical properties are easily derived or are readily available from other sources [3, 4]. For all discussion in this appendix, the reader is referred to Figure 1 and the definition of variables given in Section 2.3. For measurement number j out of a set of p measurements, the measurement function will be written $f_j(\mathbf{r},\mathbf{v})$. Only in the case of range rate measurements will f be a function of velocity as well as position.

A.1 RANGE

The range measurement, as a function of the satellite ECI position vector $\mathbf{r}$ and a station position vector $\mathbf{b}_i$, is given as follows:

$$f_j(\mathbf{r}) = |\mathbf{r} - \mathbf{b}_i| = \sqrt{(\mathbf{r} - \mathbf{b}_i)\cdot(\mathbf{r} - \mathbf{b}_i)}$$

The geometrical partial derivatives with respect to velocity are clearly 0. With respect to the position vector,

$$\frac{\partial f_j}{\partial \mathbf{r}} = \frac{(\mathbf{r}-\mathbf{b}_i)^{\mathrm{T}}}{\sqrt{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{r}-\mathbf{b}_i)}} = \frac{1}{f_j(\mathbf{r})}(\mathbf{r}-\mathbf{b}_i)^{\mathrm{T}}$$

A.2 RANGE RATE

The range rate measurement, as a function of the satellite ECI position vector $\mathbf{r}$, velocity vector $\mathbf{v}$, and a station position vector $\mathbf{b}_i$, is given as follows:

$$f_j(\mathbf{r},\mathbf{v}) = \frac{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{v}-\mathbf{v}_{\text{site}})}{|\mathbf{r}-\mathbf{b}_i|} = \frac{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{v}-\mathbf{v}_{\text{site}})}{\sqrt{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{r}-\mathbf{b}_i)}}$$

where the velocity vector of the site, $\mathbf{v}_{\text{site}}$, is given in terms of the angular momentum vector of the earth, ω (see Section 2.5.4), as follows:

$$\mathbf{v}_{\text{site}} = \omega \times \mathbf{b}_i$$

The geometrical partial derivative with respect to the velocity vector is as follows:

$$\frac{\partial f_j}{\partial \mathbf{v}} = \frac{(\mathbf{r}-\mathbf{b}_i)^{\mathrm{T}}}{|\mathbf{r}-\mathbf{b}_i|} = \frac{(\mathbf{r}-\mathbf{b}_i)^{\mathrm{T}}}{\sqrt{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{r}-\mathbf{b}_i)}}$$

The geometrical partial derivative with respect to the position vector is as follows:

$$\frac{\partial f_j}{\partial \mathbf{r}} = \frac{(\mathbf{v}-\mathbf{v}_{site})^T}{|\mathbf{r}-\mathbf{b}_i|} - \frac{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{v}-\mathbf{v}_{site})}{|\mathbf{r}-\mathbf{b}_i|^3}(\mathbf{r}-\mathbf{b}_i)^T$$

$$= \frac{(\mathbf{v}-\mathbf{v}_{site})^T}{\sqrt{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{r}-\mathbf{b}_i)}} - \frac{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{v}-\mathbf{v}_{site})}{\sqrt{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{r}-\mathbf{b}_i)}^3}(\mathbf{r}-\mathbf{b}_i)^T$$

A.3 ELEVATION ANGLE

The elevation angle measurement, as a function of the satellite ECI position vector $\mathbf{r}$ and a station position vector $\mathbf{b}_i$, is given as follows:

$$f_j(\mathbf{r}) = -\arcsin\left[\frac{\mathbf{n}\cdot(\mathbf{r}-\mathbf{b}_i)}{|\mathbf{r}-\mathbf{b}_i|}\right] = -\arcsin\left[\frac{\mathbf{n}\cdot(\mathbf{r}-\mathbf{b}_i)}{\sqrt{(\mathbf{r}-\mathbf{b}_i)\cdot(\mathbf{r}-\mathbf{b}_i)}}\right]$$

where $\mathbf{n}$ is the unit normal to the ellipsoidal earth at the station, as given in Section 2.4.1.

The geometrical partial derivatives with respect to velocity are clearly 0. With respect to the position vector,

$$\frac{\partial f_j}{\partial \mathbf{r}} = -\frac{\mathbf{n}^T}{|\mathbf{r}-\mathbf{b}_i|\cos[f_j(\mathbf{r})]}\left[\mathbf{I} - \frac{(\mathbf{r}-\mathbf{b}_i)(\mathbf{r}-\mathbf{b}_i)^T}{(\mathbf{r}-\mathbf{b}_i)^2}\right]$$

$$= -\frac{\mathbf{n}^T}{\sqrt{(\mathbf{r}-\mathbf{b}_i)^2 - [\mathbf{n}\cdot(\mathbf{r}-\mathbf{b}_i)]^2}}\left[\mathbf{I} - \frac{(\mathbf{r}-\mathbf{b}_i)(\mathbf{r}-\mathbf{b}_i)^T}{(\mathbf{r}-\mathbf{b}_i)^2}\right]$$

For the user's convenience, ISODAE computes all angles in degrees. Therefore, factors of $180/\pi$ are multiplied into the measurement function and geometrical partial derivatives for elevation angle calculations.

A.4 AZIMUTH ANGLE

Azimuth is the clockwise angle between the direction of north in the ground station's plane on earth and the projection of the satellite's position vector with respect to that station in that station's plane. The unit vector in the direction of north, measured in the station's plane, $\mathbf{i}_{north}$, can be calculated as follows. First, let $\mathbf{i}_z$ be the unit vector in the z direction in the ECI coordinate frame, let $\mathbf{b}_i$ be the station position vector, let $\mathbf{n}$ be the unit normal to the ellipsoidal earth at the station, as given in Section 2.4.1, and let $\mathbf{r}_z = \mathbf{i}_z - \mathbf{b}_i$. Then,

$$\mathbf{i}_{north} = \frac{\mathbf{r}_z - (\mathbf{r}_z \cdot \mathbf{n})\mathbf{n}}{\sqrt{\mathbf{r}_z^2 - (\mathbf{r}_z \cdot \mathbf{n})^2}}$$

The projection $\mathbf{d}_\perp$ in the ground station's plane of the position vector of the satellite with respect to the station is given as follows:

$$\mathbf{d}_\perp = \mathbf{r} - \mathbf{b}_i - [(\mathbf{r} - \mathbf{b}_i) \cdot \mathbf{n}]\mathbf{n}$$

Finally, the azimuth angle is given as follows:

$$f_j(\mathbf{r}) = \arccos\left[\frac{\mathbf{d}_\perp \cdot \mathbf{i}_{north}}{|\mathbf{d}_\perp|}\right]$$

The geometrical partial derivatives with respect to velocity are clearly 0. With respect to the position vector,

$$\frac{\partial f_j}{\partial \mathbf{r}} = \frac{1}{\sin\left[f_j(\mathbf{r})\right]}\left[\frac{\mathbf{d}_\perp \cdot \mathbf{i}_{north}}{|\mathbf{d}_\perp|^3}\mathbf{d}_\perp^T + \frac{\mathbf{i}_{north}^T}{|\mathbf{d}_\perp|}\left(\mathbf{n}\mathbf{n}^T - \mathbf{I}\right)\right]$$

Factors of $180/\pi$ are multiplied into the measurement function and geometrical partial derivatives for azimuth angle calculations so that all angles appear in degrees.

APPENDIX B

MATHEMATICA PROCEDURES FOR ISODAE

```
(*
This file is required for setting up the Interferometric Satellite
Orbit Determination Accuracy Estimator (ISODAE). Select all cells
in this file and have Mathematica evaluate them. Included are
physical constants, polynomial for prime meridian angle
computation, ground station ECI position computation, two-body
trajectory propagation and state transition matrix computation,
Bulirsch-Stoer trajectory propagation, satellite ECI state vector
computation from Keplerian elements, computation of elevation
angle to satellite from ground station, signal transit time and
group delay measurement functions, and generation of normally
distributed random variates.

Note: To run ISODAE, you must open the following files:
    "ISODAE Set up",
    "ISODAE"

    Also, the user must specify which trajectory propagator
    ISODAE is to use. (See procedure 'Propagate' below.)
    The choices currently available are TwoBody and
    BulirschStoer.) The user must also specify the means of
    state transition matrix computation. (See procedure
    'TransitionMatrix' below.) The only choice currently
    available for state transition matrix computation is
    two-body.

You must evaluate all cells in this file first.
*)
```

```
(*-------------------------------------------------------------*)
(*Choose trajectory propagator

 (Propagate state vector x0 from epoch to epoch plus
 deltat seconds)

*)

Propagate[x0_,deltat_]:=Block[{},

 xf=TwoBody[x0,deltat]; (*Two-body orbital approximation*)
 phi={{F,0,0,G,0,0},{0,F,0,0,G,0},{0,0,F,0,0,G},
   {Ft,0,0,Gt,0,0},{0,Ft,0,0,Gt,0},{0,0,Ft,0,0,Gt}};
 xf


(* BulirschStoer[x0,deltat] (*Rational extrapolation
      numerical integrator*)
*)

(*End Block*)]
(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(*Choose transition matrix computation algorithm

 (Compute state transition matrix from epoch t0 to time t)

*)

TransitionMatrix[x0_,t_]:=Block[{},

(* Currently, only the two-body state transition matrix is
available in ISODAE*)

 TwoBody[x0,t];
 phi={{F,0,0,G,0,0},{0,F,0,0,G,0},{0,0,F,0,0,G},
   {Ft,0,0,Gt,0,0},{0,Ft,0,0,Gt,0},{0,0,Ft,0,0,Gt}}

(* In future versions of ISODAE, numerical integration of
the variational equations of motion can be selected for state
transition matrix computation:

 Variational[x0,t]

*)
(*End Block*)]
(*-------------------------------------------------------------*)


(*-------------------------------------------------------------*)
(*Set up physical constants *)

we2=N[15.041067178*Pi/180/3600,20](*rad/sec*);
we=15.041067178(* °/day *);
ae=aearth=6378.14;mu=398600.45;ee=eearth=0.08182;
c=299792.458(*km/sec*)
(*-------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------*)
Cross[x_,y_]:={x[[2]]*y[[3]]-x[[3]]*y[[2]],
   x[[3]]*y[[1]]-x[[1]]*y[[3]],
   x[[1]]*y[[2]]-x[[2]]*y[[1]]}
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
Diad[x_]:={x[[1]]*x,x[[2]]*x,x[[3]]*x}
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(* Give prime meridian at ΔJD=d ephemeris days from 1/1/1950,
 in B1950 ECI coordinate frame *)
pm[d_]:=Block[{pm1,ppm},pm1=99.87+24*we*d;
  ppm=pm1-360*Floor[pm1/360]
  ]
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(* Give B1950 ECI coordinates of a ground site with
 bj={lat,lon,alt};
 lat = geodetic latitude, lon = longitude, alt = altitude,
  (km, °, °) at d ephemeris days from 1/1/1950*)
b[bj_,d_]:=Block[{alpha,lat,alt},
  alt=bj[[3]];
  lat=N[(Pi/180)bj[[1]],20];
  alpha=N[(Pi/180)(bj[[2]]+pm[d]),20];
  bb={ae*Cos[alpha]/Sqrt[1+(1-ee^2)(Tan[lat])^2]+
    alt*Cos[alpha]*Cos[lat],
   ae*Sin[alpha]/Sqrt[1+(1-ee^2)(Tan[lat])^2]+
    alt*Sin[alpha]*Cos[lat],
   ae*(1-ee^2)Sin[lat]/Sqrt[1-(ee^2)(Sin[lat]^2)]+
    alt*Sin[lat]}
 (*End Block*)]
(*-----------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
SolveKepler[E0_,dEmax_,e_,M_]:=

(* Solves Kepler's equation M = E - e Sin[E] for E *)

 Block[{E1,E2,precision},

  precision=2-Floor[Log[10,dEmax]];
  E1=N[E0,precision];
  E2=N[E1+(e*Sin[E1]-E1+M)/(1-e*Cos[E1]),precision];
  While[Abs[E2-E1]>dEmax,
   E1=E2;E2=E1+(e*Sin[E1]-E1+M)/(1-e*Cos[E1])];

  EE=E2

 (* End Block *)]
(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
Cartesian[elts_]:=Block[{a,e,i,bigw,w,M,r,h,theta,f,p},

 a=elts[[1]];e=elts[[2]];i=N[Pi/180*elts[[3]],20];
 bigw=N[Pi/180*elts[[4]],20];
 w=N[Pi/180*elts[[5]],20];M=N[Pi/180*elts[[6]],20];
 SolveKepler[1,10^-25,e,M];
 r=a*(1-e*Cos[EE]);
 f=2ArcTan[Sqrt[(1+e)/(1-e)]*Tan[1/2*EE]];
 theta=(w+f);
 p=a*(1-e^2);
 h=Sqrt[mu*p];

 N[
 { r*(Cos[bigw]*Cos[theta]-Sin[bigw]*Sin[theta]*Cos[i]),
  r*(Sin[bigw]*Cos[theta]+Cos[bigw]*Sin[theta]*Cos[i]),
  r*Sin[theta]*Sin[i],

  -mu/h*(Cos[bigw]*(Sin[theta]+e*Sin[w])
    +Sin[bigw]*(Cos[theta]+e*Cos[w])*Cos[i]),
  -mu/h*(Sin[bigw]*(Sin[theta]+e*Sin[w])
    -Cos[bigw]*(Cos[theta]+e*Cos[w])*Cos[i]),
  mu/h*(Cos[theta]+e*Cos[w])*Sin[i]
 }
 ,20]

(* End Block *)]
(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
TwoBody[y0_,deltat_]:=

 Block[{r00,v00,a,h,p,e,E1,M1,M,EE,theta,rf,vf},

  r00={y0[[1]],y0[[2]],y0[[3]]};
  v00={y0[[4]],y0[[5]],y0[[6]]};

  a=1/(2/Sqrt[r00.r00]-v00.v00/mu);
  h=Cross[r00,v00];
  p=h.h/mu;
  e=Sqrt[1-p/a];
  If[r00.v00<0,E1=N[2Pi,30]-ArcCos[(a-Sqrt[r00.r00])/a/e],
   E1=ArcCos[(a-Sqrt[r00.r00])/a/e]];
  M1=E1-e*Sin[E1];
  M=M1+Sqrt[mu/a^3]*deltat;
  SolveKepler[1,10^-25,e,M];
  r=a*(1-e*Cos[EE]);
  theta=2ArcTan[Sqrt[(1+e)/(1-e)]*Tan[1/2*EE]]-
   2ArcTan[Sqrt[(1+e)/(1-e)]*Tan[1/2*E1]];

  F=1-r/p*(1-Cos[theta]);
  G=Sqrt[r00.r00]*r*Sin[theta]/Sqrt[mu*p];
  Ft=Sqrt[mu]/(Sqrt[r00.r00]*p)*(r00.v00/Sqrt[mu]*
   (1-Cos[theta])-Sqrt[p]*Sin[theta]);
  Gt=1-Sqrt[r00.r00]/p*(1-Cos[theta]);


  rf=N[F*r00+G*v00,15];
  vf=N[Ft*r00+Gt*v00,15];

  xf=Join[rf,vf]

 (* End TwoBody Block *)]
(*-------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------*)
(*Compute subsatellite latitude (geocentric) and longitude, given
 position vector r00 at time jd in Δ(ephemeris days from
 1/1/1950 0000 hrs Z to epoch)*)
subsat[r00_,jd_]:=Block[{lat,lon},

 lat=N[180/Pi*ArcSin[r00[[3]]/Sqrt[r00.r00]],10];
 lon=ArcTan[r00[[2]]/r00[[1]]];
 If[r00[[1]]<0,lon=Pi+lon];
 If[lon<0,lon=2Pi+lon];
 lon=N[180/Pi*lon-pm[jd],10];
 {lat,lon}

]
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(* Elevation angle to satellite, in °,
 given satellite position vector r and site position vector b *)
fel[r_,b_]:=Block[{},

 d=r-b;
 range=Sqrt[d.d];
 If[b[[3]]==0,n=-b,
  n={(ee^2-1)b[[1]]/b[[3]],(ee^2-1)b[[2]]/b[[3]],-1}
 (*End If*)];
 If[b[[3]]<0,n=-n];
 lengthn=Sqrt[n.n];
 el=N[-180/Pi*ArcSin[n.d/lengthn/range],20]

(* End Block *)]
(*-----------------------------------------------------------*)
```

```
(*-----------------------------------------------------------*)
(* Geometrical partial derivatives of elevation angle to satellite,
 given satellite position vector r and site position vector b *)

dfel[r_,b_]:=Block[{},
 fel[r,b];
 N[Join[-180/Pi/lengthn/range/Cos[Pi/180*el]*n.
  (IdentityMatrix[3]-1/range/range*Diad[d]),{0,0,0}],20]

(* End Block *)]
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(* Azimuth angle to satellite, in °,
 given satellite position vector r and site position vector b *)
faz[r_,b_]:=Block[{},

  d=r-b;
  If[b[[3]]==0,n=-b,
   n={(ee^2-1)b[[1]]/b[[3]],(ee^2-1)b[[2]]/b[[3]],-1}
  (*End If*)];
  If[b[[3]]<0,n=-n];
  lengthn=Sqrt[n.n];

  dperp=d-((d.n)/lengthn/lengthn)n;
  iz={0,0,1};
  rz=(iz-b);
  north=(rz-((rz.n)/lengthn/lengthn)n);
  north=north/Sqrt[north.north];
  ldperp=Sqrt[dperp.dperp];
  diadn={n[[1]]n,n[[2]]n,n[[3]]n};
  az=N[180/Pi*ArcCos[(dperp.north)/ldperp],20]

 (*End Block*)]
(*-----------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(* Geometrical partial derivatives of azimuth angle to satellite,
 given satellite position vector r and site position vector b *)

dfaz[r_,b_]:=Block[{n,d,dperp,iz,north},
  faz[r,b];
  N[Join[
   180/Pi/Sin[az*Pi/180]*
   (dperp.north/(ldperp^3)*(d-(n.d)/(lengthn^2)*n)+
    1/ldperp*(diadn.north/(lengthn^2)-north)
      ),{0,0,0}],20]
 (*End Block*)]
(*-------------------------------------------------------------*)

(*-------------------------------------------------------------*)
(* Range rate function, given satellite position vector r,
 velocity vector v, and site position vector b *)

frr[r_,v_,b_]:=Block[{},

 d=r-b;
 vsite=N[Pi/180/3600*we*{-b[[2]],b[[1]],0},20];
 rr=(d.(v-vsite))/Sqrt[d.d]

(* End Block*)]
(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(* Geometrical partial derivatives of range rate,
 given satellite position vector r,
 velocity vector v, and site position vector b *)

dfrr[r_,v_,b_]:=Block[{d,range},

 d=r-b;range=Sqrt[d.d];frr[r,v,b];
 drr=Join[(v-vsite)/range-d*rr/(range^2),d/range]

(* End Block*)]
(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(*Signal transit time algorithm.  b1 is lat(geodetic),lon,alt of
the station 1 for group/phase delay measurement.
tj is the time, in Δ(ephemeris days from 1/1/1950 0000 hrs Z),
at which the signal arrived at station 1.  t0 is the time, in
Δ(ephemeris days from 1/1/1950 0000 hrs Z), at which the satellite
state vector x0 is known.  dt is the accuracy threshold for
quitting the iteration.

This algorithm computes the signal transit time (in seconds)
 from the satellite to station 1.
The position vector of the satellite at the time of signal
emanation is returned in rsat*)

TransitTime[b1_,tj_,x0_,t0_,dt_]:=
 Block[{b1j,xi,i,time2,time,first},
  b1j=b[b1,tj];
  time=time2=0;first=True;

  While[Abs[time2-time]>dt||first,
   time=time2;
   first=False;
   deltat=24*3600*(tj-t0)-time;
    (*Current estimate of time
    of signal emanation, in seconds from  t0*)
   xi=Propagate[x0,deltat];
   rsat=Drop[xi,-3];
   time2=(1/c)Sqrt[(rsat-b1j).(rsat-b1j)]
  (*End While*)];

  transittime=time2

 (*End Block*)]
(*-------------------------------------------------------------*)

(*-------------------------------------------------------------*)
(*Group delay algorithm.  b1,b2 are lat(geodetic),lon,alt of
stations 1 and 2, respectively, for group/phase delay measurement.
```

```
tj is the time, in Δ(ephemeris days from 1/1/1950 0000 hrs Z),
at which the signal arrived at station 1.  t0 is the time, in
Δ(ephemeris days from 1/1/1950 0000 hrs Z), at which the satellite
state vector x0 is known.  dt is the accuracy threshold for
quitting the iteration.

This algorithm computes the group delay (in seconds)
 between stations 1 and 2.

Group delay is returned in global variable "groupdelay".
Station 2 position vector at time of signal arrival is returned
 in global variable "b2j".
Time of signal arrival at station 2 is returned in global variable
 "t2". *)

GroupDelay[b1_,b2_,tj_,x0_,t0_,dt_]:=
 Block[{i,time,time2,first},

  TransitTime[b1,tj,x0,t0,dt];
   (*Compute transit time to station 1;
     Also calculate position vector of satellite at
     time of signal emanation, rsat*)
  time=time2=0;first=True;

  While[Abs[time2-time]>dt||first,
   time=time2;
   first=False;
   t2=tj+time/3600/24;(*Current estimate of time of signal arrival at
       station 2:  tj+time/3600/24 (ΔJD)*)
   b2j=b[b2,t2];
   time2=(1/c)Sqrt[(rsat-b2j).(rsat-b2j)]-transittime
  (*End While*)];

  groupdelay=time2

 (*End Block*)]
(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(* Generate one normal random variate given mean, sd *)
normal[mean_,sd_]:=Block[{r1,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11,r12},

 r1=Random[];r2=Random[];r3=Random[];r4=Random[];
 r5=Random[];r6=Random[];r7=Random[];r8=Random[];
 r9=Random[];r10=Random[];r11=Random[];r12=Random[];
 x1=((r1+r2+r3+r4+r5+r6+r7+r8+r9+r10+r11+r12)-6)/4;
 x1=((((0.029899776(x1^2)+0.008355968)(x1^2)+
  0.076542912)(x1^2)+0.252408784)(x1^2)+
  3.949846138)x1;
 x1=N[mean+sd*x1,20]
(* End Block *)]
(*-------------------------------------------------------------*)

(*-------------------------------------------------------------*)
(* Calculate one Newton-Raphson iteration correction, dx(t0),
 to the state vector at epoch,
 given current guess xi(t0) and measurement vector ym *)

dx[xi_,ym_]:=Block[{site1,site2,xj,tj,j,x,y,z,vx,vy,vz,dy,bj,dj},

 dy={};
 jacobiani={};

  For[j=1,j<=p,j++,

   If[measurements[[j]][[2]]==1,

    site1=sites[[measurements[[j]][[3]]]];
    tj=measurements[[j]][[1]];
    bj=b[site1,tj];
    xj=Propagate[xi,(tj-t0)*3600*24];
    x=xj[[1]];y=xj[[2]];z=xj[[3]];
    dj={x,y,z}-bj;
    AppendTo[dy,ym[[j]]-Sqrt[dj.dj]];
    AppendTo[jacobiani,jacobian[[j]].tm[j]]
```

```
(* End If*)];

If[measurements[[j]][[2]]==2,

 site1=sites[[measurements[[j]][[3]]]];
 tj=measurements[[j]][[1]];
 bj=b[site1,tj];
 xj=Propagate[xi,(tj-t0)*3600*24];
 x=xj[[1]];y=xj[[2]];z=xj[[3]];
 vx=xj[[4]];vy=xj[[5]];vz=xj[[6]];
 AppendTo[dy,ym[[j]]-frr[{x,y,z},{vx,vy,vz},bj]];
 AppendTo[jacobiani,jacobian[[j]].tm[j]]

(* End If*)];

If[measurements[[j]][[2]]==3,

 site1=sites[[measurements[[j]][[3]]]];
 tj=measurements[[j]][[1]];
 bj=b[site1,tj];
 xj=Propagate[xi,(tj-t0)*3600*24];
 x=xj[[1]];y=xj[[2]];z=xj[[3]];
 AppendTo[dy,ym[[j]]-fel[{x,y,z},bj]];
 AppendTo[jacobiani,jacobian[[j]].tm[j]]

(* End If*)];

If[measurements[[j]][[2]]==4,

 site1=sites[[measurements[[j]][[3]]]];
 tj=measurements[[j]][[1]];
 bj=b[site1,tj];
 xj=Propagate[xi,(tj-t0)*3600*24];
 x=xj[[1]];y=xj[[2]];z=xj[[3]];
 AppendTo[dy,ym[[j]]-faz[{x,y,z},bj]];
 AppendTo[jacobiani,jacobian[[j]].tm[j]]

(* End If*)];
```

```
 If[measurements[[j]][[2]]==5,

  site1=sites[[measurements[[j]][[3]]]];
  site2=sites[[measurements[[j]][[4]]]];
  tj=measurements[[j]][[1]];
  GroupDelay[site1,site2,tj,xi,t0,0.0000000001];
  AppendTo[dy,ym[[j]]-c*groupdelay];

  xj=tm[j].xi;
  x=xj[[1]];y=xj[[2]];z=xj[[3]];
  AppendTo[jacobiani,jacobian[[j]].tm[j]]

 (* End If*)];

(*End For j*)];

ft=Transpose[jacobiani];

ft=Drop[ft,m-6];jacobiani=Transpose[ft];

covariancematrix=Inverse[ft.jacobiani];
dx0=covariancematrix.ft.dy

(*End Block*)]
(*-------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------*)
(* Solve orbit determination problem by N-R iteration until
 |r0-r1|<raccuracy and |v0-v1|<vaccuracy*)

Solvex[x00_,ym_,raccuracy_,vaccuracy_]:=

Block[{first,x1,x0,r1,r0,v1,v0},

 first=True;
 x1=x00;x0=x00;
 r1=Drop[x1,-3];v1=Drop[x1,3];
 r0=Drop[x0,-3];v0=Drop[x0,3];
 index=0;

 While[(Sqrt[(r1-r0).(r1-r0)]>raccuracy||
  Sqrt[(v1-v0).(v1-v0)]>vaccuracy||first)&&
  index<10,

  x0=x1;r0=Drop[x0,-3];v0=Drop[x0,3];
  first=False;
  index++;

  If[m==6,
   x1=x0+dx[x0,ym];
   r1=Drop[x1,-3];v1=Drop[x1,3],

   r1=r0+dx[x0,ym];
   x1=Join[r1,v0]

  (*End If*)]

 (*End While*)];

 xest=x1

(*End Block*)]
(*-----------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(* Monte Carlo simulation *)
MonteCarlo[nit_,sigma_,bias_]:=Block[{i,j,percentdone},

 errors={};
 percentdone=0;

 For[i=1,i<=nit,i++,

  If[Floor[i/nit*10]>percentdone,

   percentdone++;
   Print[10percentdone,"% complete"]

  (*End If*)];

  (*Generate p measurement errors*)
  error={};

  For[j=1,j<=p,j++,
   AppendTo[error,normal[bias[[j]],sigma[[j]]]]
  (*End For j*)];

  ym=ytrue+error;

  xi=Solvex[x0,ym,0.0001,1000];

  If[m==6,

   AppendTo[errors,N[
    {x0[[1]]-xi[[1]],
    x0[[2]]-xi[[2]],
    x0[[3]]-xi[[3]],
    x0[[4]]-xi[[4]],
    x0[[5]]-xi[[5]],
    x0[[6]]-xi[[6]]},
     15]],
```

```
  AppendTo[errors,N[
   {x0[[1]]-xi[[1]],
   x0[[2]]-xi[[2]],
   x0[[3]]-xi[[3]]},15]]

 (*End If*)]

(*End For i*)]

(*End Block*)]

(*-------------------------------------------------------------*)


(*-------------------------------------------------------------*)
PDOP[sigmadr_]:=Block[{},

 Solvex[x0,ytrue,0.0001,1000];
 dr=sigmadr{Sqrt[covariancematrix[[1]][[1]]],
    Sqrt[covariancematrix[[2]][[2]]],
    Sqrt[covariancematrix[[3]][[3]]]};
 Print["Sigma x = ",dr[[1]]];
 Print["Sigma y = ",dr[[2]]];
 Print["Sigma z = ",dr[[3]]];
 Print["Sigma pos = ",Sqrt[dr.dr]]

(*End Block*)]
(*-------------------------------------------------------------*)
```

```
MMID[y_,dydx_,xs_,htot_,nstep_]:=

 Block[{h,ym,yn,x,n,swap},

  h=htot/nstep;
  ym=y;
  yn=y+h*dydx;
  x=xs+h;

  For[n=2,n<=nstep,n++,
   swap=ym+2h*DERIVS[x,yn];
   ym=yn;
   yn=swap;
   x=x+h
  (* End For n *)];

  yout=0.5*(ym+yn+h*DERIVS[x,yn])

 (* End Block *)]

(* End MMID function, which returns y(xs+htot) in yout *)
```

```
RZEXTR[iest_,xest_,yest_,nv_,nuse_]:=

 Block[{k,j,m1,v,c,b1,b,dd,ddy,yy},

  x[iest]=xest;
  d[iest]={0,0,0,0,0,0};
  If[iest==1,
   yz=yest;
   d[1]=yest;
   dy=yest
  (* Else *),
   m1=Min[iest,nuse];
   For[k=1,k<=m1-1,k++,fx[k+1]=x[iest-k]/xest];
   For[j=1,j<=nv,j++,
    yy=yest[[j]];
    v=d[1][[j]];
    c=yy;
    dd=d[1];dd[[j]]=yy;d[1]=dd;
    For[k=2,k<=m1,k++,
     b1=fx[k]*v;
     b=b1-c;
     If[b==0,
      ddy=v
     (* Else *),
      b=(c-v)/b;
      ddy=c*b;
      c=b1*b
     (* End If *)];
     If[k==m1,v=v,v=d[k][[j]]];
     dd=d[k];dd[[j]]=ddy;d[k]=dd;
     yy=yy+ddy
    (* End For k *)];
    dy[[j]]=ddy;
    yz[[j]]=yy
   (* End For j *)]
  (* End If *)];
```

```
yz

(* End Block *)]

(* End RZEXTR function, which returns extrapolated function
 values in yz and estimated error in dy *)
```

```
BSSTEP[y0_,dydx0_,x0_,h_,dymax_]:=

(* Integrate DEs given by DERIVS from initial conditions
 y=y0, y'=dydx0 at x=x0 to x=x0+h *)

 Block[{y2,y1,dy,nseq},

  nseq={2,4,6,8,12,16,24,32,48,64,96};

  MMID[y0,DERIVS[0,y0],0,h,2];
  RZEXTR[1,(dt/2)^2,yout,6,7];
  y1=yz;
  MMID[y0,DERIVS[0,y0],0,h,4];
  RZEXTR[2,(dt/4)^2,yout,6,7];
  y2=yz;
  n=2;
  dy=Abs[y2-y1];

  While[dy[[1]]>dymax[[1]]||dy[[2]]>dymax[[2]]||
   dy[[3]]>dymax[[3]]||dy[[4]]>dymax[[4]]||
   dy[[5]]>dymax[[5]]||dy[[6]]>dymax[[6]],

  If[n>11,Print["BS Failed with given step size"],
   n=n+1];

  y1=y2;
  MMID[y0,DERIVS[0,y0],0,h,nseq[[n]]];
  RZEXTR[n,(h/nseq[[n]])^2,yout,6,7];
  y2=yz;
  dy=Abs[y2-y1]

 (* End While *)];

 y=y2

(* End BSSTEP Block *)]
```

```
INTEGRATE[y0_,dydx0_,x0_,h_]:=

(* Integrate DEs given by DERIVS from initial conditions
 y=y0, y'=dydx0 at x=x0 to x=h hours (i.e., x=x0+3600h) *)

 Block[{i,j,y2,y1,nseq,dt,ytrue,yBS,dr,dv},

  nseq={2,4,6,8,12,16,24,32,48,64,96};
  dt=3600;
  y=y0;

  For[i=1,i<=h,i++,

   For[j=1,j<=6,j++,

    MMID[y,DERIVS[(i-1)*dt,y],0,dt,nseq[[j]]];
    RZEXTR[j,(dt/nseq[[j]])^2,yout,6,7]

   (* End For j *)];

   ytrue=TwoBody[y0,dt*i];
   yBS=yz;
   dr=Sqrt[(ytrue[[1]]-yBS[[1]])^2
    +(ytrue[[2]]-yBS[[2]])^2
    +(ytrue[[3]]-yBS[[3]])^2];
   dv=Sqrt[(ytrue[[4]]-yBS[[4]])^2
    +(ytrue[[5]]-yBS[[5]])^2
    +(ytrue[[6]]-yBS[[6]])^2];

   Print[N[yBS,15]];

   Print[i*dt,":  dr = ",N[dr,15],"   dv = ",
     N[dv,15]]

  (* End For i *)]

 (* End INTEGRATE Block *)]
```

```
(*

In this file, the user specifies an orbit determination scenario.
He must specify satellite's B1950 ECI state vector, x0, in km,
km/sec, at time t0, in Δ(ephemeris days from 1/1/1950, 0000 hrs Z).

Note:  To run ISODAE, you must open the following files:
  "ISODAE Set up",
  "ISODAE"

You must evaluate all cells in "ISODAE Set up" before doing
any work here!

*)

(*-------------------------------------------------------------*)
(* Specify the number of parameters to be estimated in the
 satellite state vector for orbit determination.
 If you specify 3, the state vector will be the ECI position
 vector.  If you specify 6, the state vector will be the
 position vector followed by the velocity vector.*)

m=3

(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(*Set up orbit determination scenario here. First, specify
satellite's B1950 ECI state vector (km, km/sec) at time
t0, Δ(ephemeris days from 1/1/1950, 0000 hrs Z)*)

(* NATO 3C @ 18°W, epoch 2/9/90 0000 hrs Z, r in km, v in m/sec *)
rtrue={-21542.98206000000000000,36160.27550000000000000,
2697.28210000000000000};
vtrue={-2632.08997000000000000000,-1579.92061000000000000000,
154.78188000000000000000}/1000;
x0=Join[rtrue,vtrue];
t0=14649 (*ΔJD(2/9/90 0000 hrs Z)*);

(* ALTERNATE satellites
(* DSCS III, epoch 7/28/90 0000 hrs Z, r in km *)
rtrue={-12413.44815000000000000,-40299.10412000000000000000,
-26.04883974000000000000};
vtrue={2.93799731400000000000000,-0.90565441700000000000000,
0.00243091000000000000000};
x0=Join[rtrue,vtrue];
t0=14818 (*ΔJD(7/28/90 0000 hrs Z)*);
*)

(*-------------------------------------------------------------*)


subsat[rtrue,t0]

{3.666609893, -17.31921877}
```

```
(*-------------------------------------------------------------*)
(*
Continue setting up orbit determination scenario.
 Now specify {lat(geodetic),lon,alt} (°,°,km)
 of measuring sites.
*)


sites={{-8.,-14.,0.},(*Ascension Island*)
  {45.,0.,0.},(*France*)
  {20.,-10.,0.}(*Northern Africa*)}

(* Bordeaux *)
(*sites={{45.0,0.0,0.1},{45.0,-0.2545,0.1},
  {45.17997,0.0,0.1},{45.17997,-0.2545,0.1}};*)

  (* ALTERNATE sites *)

(* England *)
(*sites={{54.0,2.0,0.5},{54.0,1.6937,0.5},
  {54.18004,2.0,0.5},{54.18004,1.6937,0.5}};*)

(*-------------------------------------------------------------*)

(*-------------------------------------------------------------*)
(* Compute elevation angle from each site to the satellite
 at the time t0 for the user to check that the
 satellite is visible.
*)
For[i=1,i<=Length[sites],i++,Print[
 fel[rtrue,b[sites[[i]],t0]]]]
(*-------------------------------------------------------------*)

75.74645460671323802
39.26721058211249497
69.10060344415283848
```

```
(*-------------------------------------------------------------*)
(*
Continue setting up orbit determination scenario.

Each measurement is a vector with time of measurement,
type of measurement, and station number(s) making the
measurement.

The first element of each measurement vector is the
time (in ΔJD) at which the measurement is to be
made.

The second element of each measurement vector is the
measurement type, as follows:

 1 Range
 2 Range rate
 3 Elevation angle
 4 Azimuth angle
 5 Differential range (group delay or phase delay)

The third element of each measurement vector is the
station (in "sites" variable index number, i.e.,
station j = sites[[j]]) at which the measurement is
to be made.

E.g., {14818.5,1,2} means that range to the satellite is
measured from site number 2 at 1200 hours UTC on 28 July 1990.

For differential range measurements, the measurement vector
will contain a fourth element that is the second "site"
number between which differential range is being measured.
Also, the time tag is the time of signal arrival at the first
listed station number. E.g., {t1,5,1,2} means that
differential range is measured between stations 1 and 2,
and t1 is the time of signal arrival at station 1.
*)
```

```
measurements={{t0,1,1},
   {t0,1,2},
   {t0,1,3},
   {t0+1/24,3,1},
   {t0+1/24,4,1}}

(*measurements={{t0+30/3600/24,1,2},
   {t0+30/3600/24,1,3},
   {t0+30/3600/24,1,4}

   (*,{t0+0.5,1,2},
   {t0+0.5,1,3},
   {t0+0.5,1,4}*)
   };*)

p=Length[measurements];

(*-------------------------------------------------------------*)
```

```
(*-------------------------------------------------------------*)
(*
This procedure must be executed by the user.  No inputs are
necessary.
The procedure computes the true measurement vector, ytrue,
and the geometrical partial derivative matrix, jacobian.
*)

Block[{j},

 ytrue={};
 jacobian={};
 Clear[tm];

 For[j=1,j<=p,j++,

  t1=measurements[[j]][[1]];
  site1=sites[[measurements[[j]][[3]]]];
  tm[j]=TransitionMatrix[x0,t1-t0];
  r={x,y,z};

  If[measurements[[j]][[2]]==1,

   xj=Propagate[x0,(t1-t0)*3600*24];
   d=Drop[xj,-3]-b[site1,t1];
   AppendTo[ytrue,Sqrt[d.d]];
   d={x,y,z}-b[site1,t1];
   AppendTo[jacobian,Join[d/Sqrt[d.d],{0,0,0}]]

  (* End If*)];

  If[measurements[[j]][[2]]==2,

   xj=Propagate[x0,(t1-t0)*3600*24];
   site=b[site1,t1];
   AppendTo[ytrue,frr[Drop[xj,-3],Drop[xj,3],site]];
   AppendTo[jacobian,dfrr[{x,y,z},{vx,vy,vz},site]]
```

```
(* End If*)];

If[measurements[[j]][[2]]==3,

 xj=Propagate[x0,(t1-t0)*3600*24];
 site=b[site1,t1];
 AppendTo[ytrue,fel[Drop[xj,-3],site]];
 AppendTo[jacobian,dfel[{x,y,z},site]]

(* End If*)];

If[measurements[[j]][[2]]==4,

 xj=Propagate[x0,(t1-t0)*3600*24];
 site=b[site1,t1];
 AppendTo[ytrue,faz[Drop[xj,-3],site]];
 AppendTo[jacobian,dfaz[{x,y,z},site]]

(* End If*)];

If[measurements[[j]][[2]]==5,

 site2=sites[[measurements[[j]][[4]]]];
 GroupDelay[site1,site2,t1,x0,t0,0.0000000001];
 b1=b[site1,t1];
 b2=b2j;

 d1={x,y,z}-b1;
 d2={x,y,z}-b2;

 AppendTo[ytrue,c*groupdelay];

 AppendTo[jacobian,
  {-d1[[1]]/Sqrt[d1.d1]+d2[[1]]/Sqrt[d2.d2],
  -d1[[2]]/Sqrt[d1.d1]+d2[[2]]/Sqrt[d2.d2],
  -d1[[3]]/Sqrt[d1.d1]+d2[[3]]/Sqrt[d2.d2],0,0,0}]
```

```
(* End If*)]

(*End For j*)]

(*End Block*)]

(*-------------------------------------------------------*)
```

```
(*-----------------------------------------------------------*)
(* If you want to verify that your orbit determination scenario
 is solvable, you may do so here.  Specify initial estimates
 close to the true satellite state vector at epoch t0.
 Degenerate orbit determination scenarios usually result in
 a message about ill-conditioned matrices.
*)

r1={-20000,35000,2500};
v1=vtrue;
x1=Join[r1,v1];

Solvex[x1,ytrue,0.01,100];Print[N[xest,20]]
(*-----------------------------------------------------------*)

{-21542.98204116836312, 36160.27551056323496, 2697.282103654370338,

  -2.63208997, -1.57992061, 0.15478188}
```

```
(*-------------------------------------------------------------*)
(*Specify measurement errors and measurement biases.
 The length of each of these vectors must equal the number
 of measurements that you specified above, and the ordering
 of the errors and biases must correspond with the ordering
 that you specified in the vector "measurements".

 Also specify the number of iterations for the Monte Carlo
 simulation.
*)

MeasurementErrors={0.005,0.006,0.007,0.015,0.020};

MeasurementBiases={0.,0.,0.,0.,0.};

Iterations=20;

(*-------------------------------------------------------------*)

(*-------------------------------------------------------------*)
(*Execute Monte Carlo simulation*)

MonteCarlo[Iterations,MeasurementErrors,MeasurementBiases]
(*-------------------------------------------------------------*)

10% complete
20% complete
30% complete
40% complete
50% complete
60% complete
70% complete
80% complete
90% complete
100% complete
```

```
(*Create summary statistics*)
xdev=ydev=zdev=0;vxdev=vydev=vzdev=0;
For[i=1,i<=Length[errors],i++,
 xdev=xdev+errors[[i]][[1]]^2;
 ydev=ydev+errors[[i]][[2]]^2;
 zdev=zdev+errors[[i]][[3]]^2;
 If[m==6,
  vxdev=vxdev+errors[[i]][[4]]^2;
  vydev=vydev+errors[[i]][[5]]^2;
  vzdev=vzdev+errors[[i]][[6]]^2]];

xdev=Sqrt[xdev/Length[errors]];
ydev=Sqrt[ydev/Length[errors]];
zdev=Sqrt[zdev/Length[errors]];
vxdev=Sqrt[vxdev/Length[errors]];
vydev=Sqrt[vydev/Length[errors]];
vzdev=Sqrt[vzdev/Length[errors]];
Print["Sigma x = ",xdev," km"];
Print["Sigma y = ",ydev," km"];
Print["Sigma z = ",zdev," km"];
Print["Sigma position = ",Sqrt[xdev^2+ydev^2+zdev^2]," km"];
If[m==6,
 Print["Sigma vx = ",vxdev," km/sec"];
 Print["Sigma vy = ",vydev," km/sec"];
 Print["Sigma vz = ",vzdev," km/sec"];
 Print["Sigma velocity = ",Sqrt[vxdev^2+vydev^2+vzdev^2],
  " km/sec"]]
```

Sigma x = 0.294188 km
Sigma y = 0.161315 km
Sigma z = 0.0819741 km
Sigma position = 0.345382 km

BIOGRAPHICAL NOTE

Michael Stephen Pavloff was born in West Lafayette, Indiana, in 1966. He grew up in St. Paul, Minnesota, and attended Irondale Senior High School, where he graduated valedictorian in 1984. He attended Harvard from 1984 through 1988 and graduated with an A.B. cum laude in physics. His senior thesis, "Purification and Chemical Characterization of Peptide G_1, an Invertebrate Neuropeptide That Stimulates Cyclic GMP Metabolism," was published in *The Journal of Neurochemistry*, Vol. 55, 1990, co-authored with Michael Goy. After a brief time working as a chef at L'Espalier restaurant in Boston, Mr. Pavloff joined the MITRE Corporation, where he is currently a member of the Orbital Mechanics Specialty Group. Mr. Pavloff's professional interests include satellite orbit determination, tracking system accuracy assessment, high-accuracy trajectory propagation, and spacecraft design; he has been responsible for studies in these areas for the United States Air Force, the United States Army, NASA, and the European Space Agency. Mr. Pavloff is also on the faculty at Harvard, where he teaches calculus and linear algebra and has helped design and write a new calculus curriculum and textbook under a National Science Foundation grant. Mr. Pavloff's recreational interests include squash, running, weight lifting, wine collecting, and French cooking. He was the 1987 winner of the North American *Tournée des Amateurs Gourmands*, a French cooking contest sponsored by the Hotel Sofitel of Lyon, France, where he placed fourth in the international competition.