Development of a Discretized Model for the Restricted Three-Body Problem

THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Richard Matthew Jedrey

Graduate Program in Aeronautical and Astronautical Engineering

The Ohio State University

2011

Master's Examination Committee:

Professor Hayrani Öz, Advisor

Professor Rama Yedavalli
Copyright by

Richard Matthew Jedrey

2011
Abstract

Spacecraft trajectory design is a science that requires high precision with little error. One of the most classic trajectory design problems is the restricted three-body problem. Two methods to develop the trajectory of a spacecraft under the influence of two celestial bodies are through the use of the equations of motion, and the patched-conic approximation. Popular tools such as MATLAB can be used to solve the equations of motion if great care is taken when selecting an ODE solver since the results are dramatically different between different solvers. As a result, these tools aren’t very robust and can create significant errors, so a different approach must be used for generalized scenarios when an exact solution for comparison is unavailable. The patched-conic approximation can be easily used in a program such as MATLAB, but its exclusion of one of the two celestial bodies at every point in the trajectory creates drawbacks and significant errors.

To avoid the errors that exist when using the patched-conic approach, research was put into the development of a simple model that could propagate a spacecraft’s trajectory under the effect of two celestial bodies while being robust enough to code and solve in a widely available program such as MATLAB. This model acts as a modification to the patched-conic approach. Throughout the trajectory the effect of the primary celestial body of the system on the spacecraft was calculated, as in the patched-
conic approach, however unlike the patched-conic approach this effect is not ignored when the spacecraft reaches the secondary body’s sphere of influence. Furthermore, the effect of the secondary body was also considered even when the spacecraft is outside the secondary body’s sphere of influence. Then, by applying a weighted average to the spacecraft’s radius and velocity components respective to each celestial body, an updated state would be created that would allow the model to accurately propagate the trajectory. This would be compared to a numerically generated ‘exact’ solution to determine the errors.

Algorithms that propagate the spacecraft’s trajectory out with respect to both celestial bodies were created and tested, including the propagation of the secondary celestial body’s orbit itself. A scheme based on the geometry was used in an attempt to combine the spacecraft’s states with respect to both celestial bodies using a weighted average. This scheme was tested at multiple points throughout the trajectory using a variety of weights, but no attempts were met with any success. However, the routines propagating the trajectories of the celestial bodies and spacecraft were proven to work correctly, and an initial foundation in creating a scheme to combine the spacecraft’s state has been laid out.
Dedication

This document is dedicated to my friends and family, especially to my parents Walter and Sabina, for all of their support through all seven years of my collegiate experience.
Acknowledgments

This thesis could not have been written were it not for the assistance of many individuals. First, I’d like to thank my advisor Dr. Hayrani Öz for taking me on as a graduate student and for the help not only these past two years, but all of the years I’ve known him. Also, I thank Dr. Rama Yedavalli for putting the time aside to be a part of my Master’s committee and for the instruction of my first orbital mechanics class. I’d also like to acknowledge Dr. Ed Overman for the many lessons he has taught me regarding numerical solutions and proper coding techniques. And finally, to the many other individuals that helped me along the way. There’s just not enough space to thank everybody.

Of course I’d also like to thank The Ohio State University’s Graduate School and the Department of Mechanical and Aerospace Engineering. I’ve been fortunate to learn many things in my two short years as a graduate student here. Also, the staff has been very helpful along the way helping to insure my timely graduation.
Vita

June 2004 .................................................. Brecksville-Broadview Heights High School
August 2007 – September 2010 ...................... NASA Co-op, Johnson Space Center
June 2009 .................................................. B.S. Aeronautical and Astronautical Engineering, The Ohio State University
January – March 2010 ................................. Graduate Teaching Associate, Department of Aerospace Engineering, The Ohio State University
July 2011 .................................................... Aerospace Engineer; Flight Mechanics and Trajectory Design Branch, NASA Johnson Space Center

Publications


Field of Study

Major Field: Aeronautical and Astronautical Engineering

vi
# Table of Contents

Abstract ...................................................................................................................... ii

Dedication ................................................................................................................ iv

Acknowledgments ..................................................................................................... v

List of Figures ........................................................................................................... x

List of Tables ........................................................................................................... xii

Nomenclature .......................................................................................................... xiii

Chapter 1: Introduction .............................................................................................. 1

Chapter 2: Concept of Model .................................................................................... 4

Chapter 3: Creation of Numerically ‘Exact’ Solution ............................................... 7

  3.1 Exact Solution Baseline Parameters ................................................................ 7

  3.2 Setting Up the Numerically ‘Exact’ Solution in MATLAB .............................. 8

  3.3 Selection of an ODE Solver .............................................................................. 14

Chapter 4: Development of Model .......................................................................... 18

  4.1 Propagation of Trajectory about Primary Celestial Body .............................. 18

  4.2 Propagation of Trajectory about Secondary Celestial Body ......................... 20

  4.2.1 Working Around Numerical Instability .................................................. 25
4.3 Propagating the Secondary Celestial Body’s Orbit........................................ 28

4.4 Combining Spacecraft States ........................................................................ 31
  4.4.1 Attempt at Combining States................................................................. 31
  4.4.2 Weighting Scheme................................................................................... 35

Chapter 5: Results and Forward Work................................................................. 40

References........................................................................................................... 46

Appendix A: MATLAB Scripts and Functions ..................................................... 47
  A.1 Exact Solution Code...................................................................................... 47
    A.1.1 Master Script for the Exact Solution ....................................................... 47
    A.1.2 Equations of Motion as a System, Mars_EOM ....................................... 47
  A.2 Restricted Three-Body Discretized Model Code ........................................... 49
    A.2.1 Master Script for the Restricted Three-Body Discretized Model Code .... 49
    A.2.2 Propagation of Spacecraft with respect to Primary Celestial Body,
        primary_prop............................................................................................. 51
    A.2.3 Propagation of Spacecraft with Respect to Secondary Celestial Body,
        secondary_prop......................................................................................... 52
    A.2.4 Propagation of Secondary Celestial Body, SBody_prop.......................... 54
    A.2.5 Script Used to Test secondary_prop....................................................... 55
    A.2.6 Kepler’s Equation for an elliptical orbit, Kepler_E............................... 55
A.2.7 Universal Time Formulation, Kepler_U..................................................... 56
List of Figures

Figure 1: Visualization of steps used by algorithm at each time step (not to scale)........ 5
Figure 2: Geometry relating the position of the spacecraft and second celestial body (not
to scale)..................................................................................................................................... 11
Figure 3: Geometry used to find gravitational forcing angle with respect to the secondary
celestial body .................................................................................................................................. 13
Figure 4: (a) shows the trajectory produced by ode45 and (b) shows the trajectory
produced by ode15s ......................................................................................................................... 15
Figure 5: Trajectory produced by ode113......................................................................................... 16
Figure 6: Geometry to determine the flight path angle with respect to the secondary
celestial body .................................................................................................................................. 22
Figure 7: Different spacecraft states resulting from different celestial bodies (not to
scale)................................................................................................................................................. 32
Figure 8: Test of subroutine that propagates spacecraft with respect to the primary
celestial body .................................................................................................................................. 36
Figure 9: Test of subroutine that propagates spacecraft with respect to the secondary
celestial body .................................................................................................................................. 37
Figure 10: Diagrams used to estimate appropriate values for weights: (a) shows the
radii, (b) the velocity components (scaled relatively, not exactly)................................................. 38
Figure 11: Points used to calculate weights. Green points were obtained from the primary body sub-model, magenta points from the secondary.......................... 41

Figure 12: Calculated weights vs. their respective positions............................ 43
List of Tables

Table 1: Data used in weight calculations ................................................................. 41
Table 2: Calculated weights ..................................................................................... 42
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>Distance between Primary and Secondary Celestial Body (Assuming a Circular Orbit)</td>
</tr>
<tr>
<td>$E$</td>
<td>Eccentric Anomaly</td>
</tr>
<tr>
<td>$F$</td>
<td>Hyperbolic Anomaly or Gravitational Attraction Force</td>
</tr>
<tr>
<td>$G$</td>
<td>Gravitational Constant</td>
</tr>
<tr>
<td>$M$</td>
<td>Mass of Primary Celestial Body</td>
</tr>
<tr>
<td>$T$</td>
<td>Gravitational Force from Secondary Celestial Body</td>
</tr>
<tr>
<td>$TOF$</td>
<td>Time Of Flight</td>
</tr>
<tr>
<td>$V$</td>
<td>Velocity</td>
</tr>
<tr>
<td>$X$</td>
<td>Universal Anomaly</td>
</tr>
<tr>
<td>$P$</td>
<td>Weight for Primary Celestial Body</td>
</tr>
<tr>
<td>$S$</td>
<td>Weight for Secondary Celestial Body</td>
</tr>
<tr>
<td>$a$</td>
<td>Semimajor Axis</td>
</tr>
<tr>
<td>$d$</td>
<td>Distance Between Spacecraft and Secondary Celestial Body</td>
</tr>
<tr>
<td>$e$</td>
<td>Eccentricity</td>
</tr>
<tr>
<td>$h$</td>
<td>Angular Momentum</td>
</tr>
<tr>
<td>$m$</td>
<td>Mass</td>
</tr>
<tr>
<td>$r$</td>
<td>Radius</td>
</tr>
<tr>
<td>$\text{realmax}$</td>
<td>Largest Real Number MATLAB Can Store</td>
</tr>
<tr>
<td>$t$</td>
<td>Time</td>
</tr>
<tr>
<td>$x$</td>
<td>Generic State Variable</td>
</tr>
<tr>
<td>$z$</td>
<td>Substitution Variable for $\alpha x^d$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Thrusting Angle due to $T$</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Angle Formed Between Position Vectors Connecting the Spacecraft and Primary Celestial Body</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>Flight Path Angle</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Orbital Energy</td>
</tr>
<tr>
<td>$\theta$</td>
<td>True Anomaly</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Angle Formed Between Position Vectors Connecting the Spacecraft and Secondary Celestial Body</td>
</tr>
<tr>
<td>$\mu$</td>
<td>Gravitational Parameter</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Angle Formed by $r$ and $d$</td>
</tr>
<tr>
<td>$\tau$</td>
<td>Ratio of Primary Celestial Body’s Gravitational Force with the Secondary Celestial Body’s</td>
</tr>
<tr>
<td>$\nu$</td>
<td>Angled Formed by the Local Horizon with respect to the Secondary Celestial Body and the Extended Position Vector of the Spacecraft with respect to the Primary Celestial Body</td>
</tr>
<tr>
<td>$\varphi$</td>
<td>Angle Formed by Elongated $r$, $\alpha x^d$, and $d$</td>
</tr>
<tr>
<td>$\psi$</td>
<td>Angle Formed Between Position Vectors Connecting the Spacecraft with Both Celestial Bodies</td>
</tr>
<tr>
<td>$\xi$</td>
<td>Rendezvous Offset Angle</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>Flight Path Angle with respect to the Secondary Celestial Body</td>
</tr>
</tbody>
</table>
Subscripts:

- **SOA**  Sphere Of Accuracy
- **SOI**  Sphere Of Influence
- **T**  Total
- **a**  Apogee
- **b**  Rendezvous Point with respect to Primary Celestial Body
- **e**  Elliptical Orbit
- **i**  Initial

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>Radial Component</td>
</tr>
<tr>
<td>s</td>
<td>Secondary Celestial Body</td>
</tr>
<tr>
<td>s/c</td>
<td>Spacecraft</td>
</tr>
<tr>
<td>v</td>
<td>Velocity</td>
</tr>
<tr>
<td>0,1,</td>
<td></td>
</tr>
<tr>
<td>...j,...</td>
<td>Iterate Point</td>
</tr>
<tr>
<td>⊥</td>
<td>Perpendicular Component</td>
</tr>
<tr>
<td>⊙</td>
<td>Primary Celestial Body</td>
</tr>
<tr>
<td>/</td>
<td>“With Respect To”</td>
</tr>
</tbody>
</table>
As is the case with many subjects in engineering, before a subject can be completely mastered its fundamental concepts must be learned. The restricted two-body problem is a classic problem that is studied thoroughly in introductory orbital mechanics classes. When referring to n-body problems, a restricted problem is when one of the bodies considered has a mass that is negligible to the other bodies in the system. This is the case when one of the bodies examined is a spacecraft. However for problems that involve a third body, such as flying from the Earth to the Moon, which was the mission scenario used as the basis in creating this model, the restricted two-body problem on its own is not enough. Some methods exist that allow for students who only know the restricted two-body problem to solve a problem such as flying from the Earth to the Moon. One example is the patched-conic approximation. However, the patched-conic approximation essentially solves two different restricted two-body problems, except for the patching point which is simply a coordinate transformation. The patched-conic approximation does not account for the effect of the Moon at any time when the spacecraft is outside the Moon’s Sphere of Influence. This creates significant drawbacks in accuracy when examining the problem of having a spacecraft fly from Earth orbit to the Moon.
This lack of accuracy was very apparent when work was done trying to solve the problem of having a spacecraft in orbit around Mars do a flyby of the Martian moon Phobos. The patched-conic approximation was tried when the spacecraft was within Phobos’ Sphere of Influence, albeit a solution was obtained, it was not at all practical. Phobos’ Sphere of Influence is in fact smaller than the mean radius of Phobos. According to the patched-conic approximation then, the spacecraft would not feel the effect of Phobos. Obviously this is not the case. Thus, the question regarding how to model the effect of Phobos came into question.

Upon consideration of how to model the effect of Phobos, the more general question of whether or not it was possible to create an algorithm that could model the restricted three-body problem came into play. Of course an ODE solver could be used to model the equations of motion, and this topic is addressed in Chapter 3, but many issues may arise when using an ODE solver with orbital mechanics. Even though the equations of motion can be solved analytically, all computers are, of course, numerical in nature. Therefore, the scaling of such a problem may create catastrophic problems numerically. Of more interest was whether or not a model could be constructed that would solve the problem using simple orbital equations without using differential equations. If such a model could be created, it could easily be implemented in popular analysis tools such as MATLAB. It would also eliminate the need of determining which ODE solver offered by different packages would be the most accurate.

Such a model would have many benefits. Not only would it provide a much more accurate solution than the patched-conic approximation, but it would be easily applicable
in popular software. Also, it would allow students the opportunity of learning a bit on how to account for the dynamics of the restricted three-body problem without having to worry about the equations of motion.
Chapter 2: Concept of Model

The main objective of this model was to discretize the restricted three-body problem in such a way that it could be implemented in popular programming tools such as MATLAB and still have an accurate solution. This chapter briefly outlines how the model created a solution. At each transition time, the state of the spacecraft was found with respect to the primary celestial body, as well as the secondary. These two different states would preferably be combined using a linear weighting scheme into a single state relative to the primary celestial body, and then this updated spacecraft state would be used as the new initial conditions for propagation about the celestial bodies individually (ignoring the other celestial body) and propagated to the next transition time. This is shown generically in Figure 1a. Then, the secondary body moved along its trajectory, and the process was repeated with new weights until the spacecraft reached its destination. This step is shown in Figure 1b.
This approach acts as a numerical solution to the restricted three-body problem by splitting it into three two-body problems. The initial conditions used are the same as those used in the patched-conic approximation; namely the spacecraft’s initial position, velocity, flight path angle with respect to the primary celestial body, as well as the angle between the position vectors connecting the spacecraft and also the primary celestial body with the secondary celestial body upon reaching the secondary body’s Sphere of Influence. Due to numerical limitations, this model can only be applied in a realm of numerical stability. Thus, as will be elaborated upon in Section 4.2.1, the model didn’t begin to include the effect of the secondary celestial body on the spacecraft until a point defining the Sphere of Accuracy around the secondary celestial body was reached. In the case of the Earth-Moon model used, the Sphere of Accuracy makes up almost ninety percent of the distance between the Earth and the Moon. Once the Sphere of Accuracy
was reached, the model propagates the trajectory out as described above. Until then, the model assumed the primary body had the only effect on the spacecraft.

Since this model split the problem into multiple restricted two-body problems, the possibility may exist to expand this model to include n-bodies. This is discussed more in Chapter 5.
Chapter 3: Creation of Numerically ‘Exact’ Solution

A basis for comparison is required in order to validate any sort of numerical model. In most cases this involves the comparison to an analytical solution. Since much of the development of the discretized model required examining several time steps, it was much more prudent to use an ODE solver in MATLAB as an exact solution so that the spacecraft’s state could be examined at any point. Since there are several ODE solvers in MATLAB the parameters of the analytical solution needed to be found by hand to ensure the result in MATLAB was accurate.

3.1 Exact Solution Baseline Parameters

An example borrowed from Fundamentals of Astrodynamics\(^3\) was used for a typical trans-lunar trajectory. This example was used by the author to illustrate the patched-conic approximation. The injection conditions provided at perigee were:

- \( r = 1.05 \text{ DU}, \)
- \( V = 1.372 \text{ SU}, \)
- \( \gamma = 0, \)
- \( \theta = 0, \)

where \( r \) is the radius, \( V \) is the velocity, and \( \gamma \) is the flight path angle, \( \theta \) is the true anomaly, and the eccentricity of the resulting trajectory was

- \( e = 0.977. \)
With this information the angular momentum and apogee radius could be found according to

\[ h = rV \cos \gamma = 1.44 \text{ DU}^2 / \text{TU} \quad (1) \]

and

\[ r_a = \frac{h^2 / \mu}{1 - e} = 90 \text{ DU}, \quad (2) \]

respectively. Canonical units were used to avoid issues with numerics.

Now that the basic parameters of the resulting earth-centric orbit were known, ODE solvers could be checked against them to ensure the numerically ‘exact’ solution developed by MATLAB was accurate.

3.2 Setting Up the Numerically ‘Exact’ Solution in MATLAB

In order to have an analytical, or exact, solution, the equations of motion need to account for all three bodies in the system. However, by assuming the spacecraft had impulsive acceleration and allowing the secondary celestial body to have acted as an external force on the spacecraft, the equations of motion for a two-body system were able to be used\(^3\). The equations of motion for a two-body system are given by

\[ \ddot{r} = r \dot{\theta}^2 - \frac{GM}{r^2} - \frac{T}{m_{s/c}} \sin \beta \quad (3a) \]

and

\[ \ddot{\theta} = -\frac{2r \dot{\theta}}{r} + \frac{T}{m_{s/c} r} \cos (\pi - \beta). \quad (3b) \]

A dot accent above a variable indicated a derivative with respect to time, \( G \) is the Gravitational Constant, \( M \) is the mass of the dominant body, the subscript \( s/c \) is a
reference to the spacecraft, \( T \) is the gravitational force on the spacecraft due to the secondary celestial body, and \( \beta \) is the angle \( T \) forms with the spacecraft’s local horizon.

These equations were implemented into one of MATLAB’s ODE solvers. However, MATLAB is limited to solving first-order differential equations, so Equations (3a) and (3b) were first converted into a system of first-order equations. Since MATLAB is designed to operate with vectors and matrices, the state variables were converted into the single state vector

\[
\dot{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} r \\ \theta \\ \dot{r} \\ \dot{\theta} \end{bmatrix}.
\]

(4)

Now the equations of motion were rewritten as a system using the new set of state variables:

\[
\begin{align*}
\dot{x}_1 &= x_3, \\
\dot{x}_2 &= x_4, \\
\dot{x}_3 &= x_1 x_4^2 - \mu x_1^2 / x_1^2 - T / m_{s/c} \sin \beta, \\
\dot{x}_4 &= -2 x_3 x_4 / x_1 + T / m_{s/c} \cos (\pi - \beta).
\end{align*}
\]

(5a) (5b) (5c) (5d)

\( \mu \) is the gravitational parameter defined by the product of the Gravitational Constant and celestial body’s mass. MATLAB automatically handles the state variables, but a
formulation for $T$ as well as $\beta$ was needed. As stated above, $T$ is the gravitational force due to the secondary celestial body. This is expressed as follows:

$$T = \frac{Gm_s m_{s/c}}{d^2} = -\frac{\mu_s m_{s/c}}{d^2},$$

(6)

where $s$ denotes reference to the secondary celestial body and $d$ is the distance between the spacecraft and secondary celestial body. With all of the initial conditions being central to the primary celestial body, $d$ was found according to the law of cosines using the geometry in Figure 2.
Figure 2: Geometry relating the position of the spacecraft and second celestial body (not to scale)

From this geometry, the law of cosines gave \( d \) as

\[
d^2 = r^2 + D^2 - 2rD\cos(\theta_{s/c} - \theta_s). \tag{7}
\]
$D$ is the distance between the primary and secondary celestial bodies, which was treated as constant for this problem. In order to know $\theta$, however, the initial position of the secondary celestial body as well as the total angle it covered during the spacecraft’s flight. The secondary celestial body’s location was found simply by finding the difference between the rendezvous point, $\theta_b$, which was given, and the angle covered during the time of flight, $\theta_T$, which is given by Equation (8). The time of flight and rendezvous point were once again taken from the example mentioned earlier as

- $TOF = 223.98$ TU,

and

- $\theta_b = 2.95$ rad,

respectively. The subscript $b$ is the point where the spacecraft crosses the secondary celestial body’s Sphere of Influence. By dividing the time of flight by the secondary celestial body’s orbital period, the total angle covered during the time of flight was found to be

$$\theta_T = \frac{TOF}{P_s} \cdot 2\pi. \quad (8)$$

$P_s$ is the period of the secondary celestial body’s orbit. With this information, the secondary celestial body’s angular position was found using the linear formulation

$$\theta_s = \frac{t}{TOF} \theta_T + \theta_i. \quad (9)$$
$t$ is the time covered since the spacecraft left its initial orbit, and $\theta_i$ is the true anomaly of the secondary celestial body when the spacecraft had its initial impulsive maneuver.

Now there was enough information to find the gravitational force on the spacecraft due to the secondary celestial body at any time. Next, the angle of the gravitational force due to the secondary celestial body acting on the spacecraft, $\beta$, was found according to the geometry in Figure 3.

![Figure 3: Geometry used to find gravitational forcing angle with respect to the secondary celestial body](image)

Equations (10) and (11) demonstrate how the geometry in Figure 3 can be used to find the gravitational forcing angle due to the secondary celestial body. According to the law of cosines,

$$\alpha = \cos \left( \frac{D^2 + d^2 - r^2}{2Dd} \right).$$

(10)
$a$ is the semimajor axis. Then from simple geometry,

$$180^\circ = (\theta_{s/c} - \theta_s) + \alpha + \sigma = \varphi + \sigma, \quad (11a)$$

so

$$\sigma = 180^\circ - (\theta_{s/c} - \theta_s) - \alpha, \quad (11b)$$

and

$$\varphi = 180^\circ - \sigma, \quad (11c)$$

$$\beta = 90^\circ + \varphi. \quad (11d)$$

$\sigma$ is the angle formed by the position vectors $d$ and the spacecraft’s radius with respect to the primary celestial body, $\varphi$ is the angle formed by the spacecraft’s local vertical with respect to the primary celestial body and the position vector $d$, and $\alpha$ is the angle formed by the position vectors $d$ and $D$. All of this information was then put together in MATLAB such that it could use an ODE solver to resolve the restricted three-body problem.

3.3 Selection of an ODE Solver

As has already been mentioned, MATLAB has several ODE solvers. There are many to choose from since differential equations can have many different properties such as whether they are stiff, how stringent the error tolerances are, what the numerics are like, among many others. However, since there are not that many different solvers, they all solved the equations discussed in Section 3.2 relatively quickly, and this author did not feel confident enough in identifying the necessary properties of the equations of interest, all ODE solvers were tested and compared to the hand calculations in Section
3.1. Figure 4 shows a couple of the trajectories resulting from rejected ODE solvers, and Figure 5 shows the trajectory from the chosen ODE solver, ode113.

Figure 4: (a) shows the trajectory produced by ode45 and (b) shows the trajectory produced by ode15s
These trajectories are the only ones shown as they are the only ones that came somewhat close to the apogee solved for in Section 3.1. It’s clear in Figure 4(b) that the trajectory reached an apogee of 80 DU whereas the trajectory should reach 90 DU. Figures 4(a) and 5 both go beyond 80 DU but when they are extrapolated out ode113 provides a trajectory that travels out further which was the more sensible of the two since a flyby of the Moon’s trailing side would increase the spacecraft’s speed and therefore increase the apogee radius, if not put the spacecraft on a non-returning trajectory.
With this solution in-hand, a comparison between the numerically ‘exact’ solution and the model being developed could be done at any time. This became very helpful in testing the development of the model.
Chapter 4: Development of Model

Chapter 2 described the objective and fundamental algorithms used to create the discretized model for the restricted three-body problem. Before the algorithm could be developed for the entire model though, ‘sub-models,’ or subroutines, for the trajectories based on either celestial body needed to be developed.

4.1 Propagation of Trajectory about Primary Celestial Body

As it was explained in Chapter 2, the spacecraft’s state was found in relation to one celestial body at a time while disregarding the other celestial body, and then those states would be combined. With the initial conditions being centric to the primary body it wasn’t difficult to propagate this trajectory out. These sub-models moved the trajectory using a small transition time. Since we are interested in position given the time of flight, Kepler’s equation was utilized. For an elliptic orbit, Kepler’s equation is\(^4\)

\[
M_e = E - esinE. \tag{12}
\]

Here, \(M_e\) is the mean anomaly for an elliptic orbit, and \(E\) is the eccentric anomaly. The initial eccentricity was given as indicated in Section 3.1, and \(M_e\) was solved for according to\(^4\)
\[ M_{e_{j+1}} = \frac{\mu^2 (1 - e_j)^{3/2} t}{\dot{h}_j^3} + M_{e_j}. \]  

(13)

The subscript \( j \) refers to an arbitrary iteration. However, even with both \( e \) and \( M_e \) known, \( E \) cannot be solved for directly in Equation (12). Instead, Newton’s method was used.

The algorithm used can be seen in Appendix A.2.6. Now the spacecraft’s updated position, indicated by the subscript \( j+1 \), was calculated with the following equations:\(^4\):

\[
\cos \theta_{j+1/\oplus} = \frac{e_j - \cos E_{j+1}}{e_j \cos E_{j+1} - 1} \]  

(14)

and

\[
\frac{h_j^2}{\mu} \]  

\[
r_{j+1/\oplus} = \frac{h_j^2}{1 + e_j \cos \theta_{j+1}}. \]  

(15)

The subscript \( \oplus \) indicates it is in reference to the primary celestial body. Again, the eccentricity and angular momentum were kept constant since we are only including the effect of one celestial body. With this information, the updated velocity components and total velocity were found using Equation (16)\(^4\):

\[
V_{\perp_{j+1/\oplus}} = \frac{h_j}{r_{j+1}}, \]  

(16a)

\[
V_{r_{j+1/\oplus}} = \frac{\mu \rho \sin \theta_{j+1}}{h_j}, \]  

(16b)

and

\[
V_{j+1/\oplus} = \sqrt{V_{\perp_{j+1/\oplus}}^2 + V_{r_{j+1/\oplus}}^2}. \]  

(16c)
The subscript $\perp$ refers to the tangential component, and the subscript $r$ refers to the radial component. The flight path angle was then updated by

$$\gamma = \arccos \left( \frac{V_{\perp,j+1/B}}{V_{j+1/B}} \right)$$  \hspace{1cm} (17)

This completed one transition time for calculating the spacecraft’s state with respect to the primary celestial body when the trajectory was elliptic. In the case where the trajectory becomes hyperbolic around the primary celestial body the universal formulation was used. This formulation is discussed in the next section, Section 4.2, since the universal formulation was used exclusively with the secondary celestial body.

4.2 Propagation of Trajectory about Secondary Celestial Body

Updating the spacecraft’s state about the secondary celestial body required a conversion of all the trajectory’s parameters that were centric to the primary celestial body so that they would be relative to the secondary body. Then Kepler’s equation would be used, but this time the universal formulation was used instead of the Kepler’s hyperbolic formulation to try working around numerical instability that resulted from using the hyperbolic formulation. This did not initially resolve the problem, but since either approach would be acceptable the universal formulation was still used. Plus, it was found that if the initial conditions into the sub-model which propagates the spacecraft about the secondary celestial body were a certain value, the resulting trajectory could either be elliptical or hyperbolic. More discussion is given about resolving the numerical instability in Section 4.2.1. As before, Newton’s method was used, but a different set of
inputs were required. This algorithm used the time step, initial radius, velocity, the gravitational parameter, and the inverse of the semi major axis, \( \alpha \).

The radius of the spacecraft with respect to the secondary celestial body at the point \( j \), \( r_{j/s} \), was found using Equation (7) except now \( d \) is replaced with \( r_{j/s} \). Converting the primary celestial body-centric velocity components to be centric to the secondary celestial body required both knowledge of the secondary body's orbital speed and the flight path angle with respect to the secondary celestial body as shown in Figure 6 and indicated by Equation (18)\(^5\) given by

\[
V_{\perp j/s} = V_{\perp/\Sigma} \cos \epsilon_j + V_s \cos \left( -\frac{\pi}{2} + \theta_s + \lambda \right),
\]

(18a)

and

\[
V_{r j/s} = V_{\perp/\Sigma} \sin \epsilon_j + V_s \sin \left( -\frac{\pi}{2} + \theta_s + \lambda \right).
\]

(18b)

\( \epsilon \) is the angle formed between the spacecraft's velocity with respect to the primary celestial body and the local horizontal of the spacecraft with respect to the secondary celestial body, and \( \lambda \) is the angle formed between the spacecraft's position vector with respect to the secondary celestial body and \( D \). For the Earth-Moon system, the Moon was assumed to be on a circular orbit with a constant orbital velocity of

- \( V_s = 0.1288 \) SU.

The flight path angle with respect to the secondary celestial body and \( \lambda \) were developed using the geometry in Figure 6 and their formulations are shown in Equation (19).
Figure 6: Geometry to determine the flight path angle with respect to the secondary celestial body

Utilizing the law of sines and the above geometry, $\epsilon$ was found as follows:

\[
\frac{\sin (\theta_s - \theta_{s/c})}{r_{j/s}} = \frac{\sin (\pi - \psi)}{D},
\]

which gives

\[
\sin (\pi - \psi) = \frac{D \sin(\theta_s - \theta_{s/c})}{r_{j/s}},
\]

so that

\[
\pi = \psi + \frac{\pi}{2} + v,
\]

or

\[
v = \pi - \psi - \frac{\pi}{2},
\]

which thereby gives

\[
\frac{\pi}{2} + (\theta_{s/c} - \gamma) = \epsilon + v + \theta_{s/c},
\]

so that finally

\[
\epsilon = \frac{\pi}{2} + \theta_{s/c} - \gamma - v - \theta_{s/c},
\]

or

\[
\epsilon = \frac{\pi}{2} - \gamma - v.
\]

Then, $\lambda$ was found simply by

\[
\lambda = 180^\circ - |\theta_s - \theta_{s/c}| - \psi.
\]
\( \psi \) is the angle formed by the position vectors of the spacecraft with respect to both celestial bodies, and \( \nu \) is the angle formed by the local horizontal with respect to the secondary celestial body and the local vertical with respect to the primary celestial body.

Substituting the results from Equation (19) into Equation (18) then yields the velocity components with respect to the secondary celestial body. Then, the total velocity with respect to the secondary celestial body was found simply by finding the root-sum-square of the velocity components. Next, the semimajor axis, \( a \), was found using

\[
a_{j/s} = \frac{1}{2/v_{j/s} - \sqrt{V_{j/s}^2/\mu_s}}.
\] (20)

This gave all of the information required to use the universal time formulation, which is given in Appendix A.2.7.

The algorithm for for the universal formulation gave the universal ‘anomaly’, \( X \). The change in the hyperbolic anomaly, \( \Delta F \), was found by

\[
\Delta F_{j+1/s} = \frac{X_{j+1/s}}{\sqrt{-a_{j/s}}}
\] (21)

However in order to find \( F_{j+1/s} \), \( F_j \) needed to be found. The formulation for \( F_j \) used was

\[
F_{j/s} = 2\text{atanh} \left( \sqrt{\frac{e_{j/s} - 1}{e_{j/s} + 1}} \tan \frac{\theta_{j/s}}{2} \right).
\] (22)
The eccentricity and true anomaly terms given in Equation (22) were found with the following equations:

\[ a_{j/s} = \frac{h_{j/s}^2}{\mu_s} \frac{1}{1 - e_{j/s}^2}, \quad (23a) \]

Giving the eccentricity as

\[ e_{j/s} = \frac{-h_{j/s}}{\sqrt{a_{j/s} \mu_s} + 1}, \quad (23b) \]

And knowing the radius is defined by

\[ r_{j/s} = \frac{h_{j/s}}{\mu_s} \frac{1}{1 + e_{j/s} \cos \theta_{j/s}}, \quad (23c) \]

We can rearrange the terms to get

\[ \theta_{j/s} = a \cos \left( \frac{h_{j/s}^2}{r_{j/s} \mu_s e_{j/s}} - \frac{1}{e_{j/s}} \right). \quad (23d) \]

The angular momentum was simply obtained by taking the product of the radius and tangential velocity component at point \( j \). This allowed the calculation of \( F_{j+1/s} \) simply by

\[ F_{j+1/s} = F_{j/s} - \Delta F_{j+1/s}, \quad (24) \]

Knowing \( F_{j+1/s} \) allowed for the calculation of the spacecraft’s updated position with respect to the secondary celestial body. The updated true anomaly was found by rearranging Equation (22) and is shown by

\[ \theta_{j+1/s} = 2 \tan \left( \frac{e_{j/s} + 1}{e_{j/s} - 1} \tanh \frac{F_{j+1/s}}{2} \right). \quad (25) \]
Then Equation (23c) was used to solve for the updated radius, \( r_{j+1/8} \) using the new true anomaly.

Then the updated velocity with respect to the secondary celestial body was found. This was done by substituting the updated position parameters into Equation (16), except here the variables are with respect to the secondary celestial body as opposed to the primary.

### 4.2.1 Working Around Numerical Instability

An issue with numerical instability was mentioned briefly in Section 4.2. When Kepler’s equation for a hyperbolic trajectory was used MATLAB always returned NaN (Not A Number) as the hyperbolic anomaly. This was the initial motivation in trying to use the universal formulation. Unfortunately, the same problem occurred. Upon further investigation it was found that the cause was due to one of the Stumpff functions (when using the universal formulation). With parameters given, the Stumpff function used was

\[
S(z) = \frac{\sinh(-z) - \sqrt{-z}}{(\sqrt{-z})^3},
\]

where \( z \) was defined by

- \( z = \alpha X^2 \),

where \( \alpha \) is the inverse of the semimajor axis. Since the semimajor axis is negative for a hyperbolic orbit, \( z \) is negative. However, since the universal anomaly was very large at the onset, \( z \) became very large (around the order of \( 10^4 \)) and the hyperbolic sine of a number of that magnitude is very large. This number is too large for MATLAB’s
numeric capability, which results in MATLAB returning a value of infinity for the
hyperbolic sine. However, as the orbit progresses, the universal anomaly decreases, so
the point at which MATLAB would become numerically stable was found by initially
finding the largest value of $z$ that MATLAB could tolerate.

This value of $z$ was found using the internal MATLAB variable `realmax` which is
the largest real number that MATLAB can store and using the equation

$$-z_0 = [\text{asinh (realmax)}]^2 \approx 5.0478 \times 10^5. \quad (27)$$

However, $z$ itself does not define a point. According to the definition of $z$, we have

$$z = \alpha X^2 = \alpha (\sqrt{\mu \Delta t})^2 = \alpha^3 \mu (\Delta t)^2. \quad (28)$$

Equation (28) shows that either $\alpha$ or $\Delta t$ must be constrained so that $z$ can always
be numerically stable. In order to make the model as robust as possible, a maximum $\Delta t$
was set to 100 TU. Even though the final model would run at a much smaller transition
time, this prevents issues from arising if another user later on would use a large transition
time. Additionally, as it will be shown shortly, this still permits the model to involve the
secondary celestial body for the vast majority of the spacecraft’s transit. With our
maximum time step set to 100 TU, we have

$$-5.0478 \times 10^5 = \alpha^3 \times 0.0123 \times 100^2, \quad (29a)$$

which gives

$$\alpha \approx -16.0103 \Rightarrow \alpha \approx -0.0625. \quad (29b)$$
Equation (29) gives the largest value allowable for the semimajor axis with respect to the spacecraft’s orbit about the secondary celestial body in order to maintain numeric stability. In order to determine where the model became numerically stable with respect to the primary celestial body, the exact solution was set to pause within the code execution the instant that the value for the absolute value of the semimajor axis with respect to the secondary celestial body was greater than or equal to 0.0625. This was found by implementing Equations (18) through (20). Fortunately, the MATLAB ODE solver also keeps track of time elapsed, and it was found about 10.5 TU elapsed before the spacecraft got into a region of numerical stability. This is not at all significant as 10.5 TU is approximately 2 hours. The spacecraft’s and secondary body’s positions at this point were:

- \( r_{/\oplus} = 6.4815 \) DU,
- \( \theta_{/\oplus} = 2.3433 \) rad = 134.3°,

and

- \( \theta_s = 2.4921 \) rad = 142.8°.

For further reference, the spacecraft’s radius with respect to the secondary celestial body was found at the point of numerical stability. This radius was found by using the geometry in Figure 3 and applying the law of cosines once again as in Equation (7) but with the appropriate values to obtain

- \( r_{SOA} = 53.867 \) DU = 343,571 km.

This radius was called the Sphere of Accuracy, as indicated by the subscript \( SOA \). Now with this point determined the model was adjusted to propagate the trajectory out
approximately 10.5 TU under the influence of the primary celestial body before applying the effect of the secondary celestial body.

4.3 Propagating the Secondary Celestial Body’s Orbit

Throughout the spacecraft’s flight the secondary body’s position must also be tracked. This was done at the initiation of the spacecraft’s transit even when the secondary body’s influence was ignored due to numerical instability, and of course, when its effect was applied to the spacecraft. The secondary celestial body’s position was tracked in much the same way that it was for the exact solution described in Section 3.2. However, since the trajectories are likely to be different than the exact solution, the rendezvous point may also be different, so the conditions at rendezvous were found explicitly.

Once again, the geometry in Figure 3 was applied with appropriate values so that the law of cosines would yield

$$r_b = \sqrt{D^2 + r_{SOI}^2 - 2Dr_{SOI} \cos \lambda}. \quad (30)$$

Next, the true anomaly of the secondary celestial body was found using

$$\theta_b = \acos \left( \frac{h^2 \mu_B}{r_b e_\alpha} - \frac{1}{e_\alpha} \right). \quad (31)$$
The subscript $a$ here refers to the initial conditions (before the spacecraft begins its transit to the secondary celestial body). Therefore, $h_a$ was determined by the product of the initial radius and velocity (since we have a flight path angle of 0), and the value of $e_a$ was given.

Similarly to Section 3.2, the total angle covered by the secondary celestial body was needed to determine its initial position. Again, Equation (8) was used to determine the total angle covered by the secondary celestial body. However, this time the time of flight was not given and was solved using the formula\textsuperscript{2}

\[
    TOF = \frac{a^3}{\sqrt{\mu}} \left[ (E_b - esinE_b) - (E_a - esinE_a) \right].
\]

If the spacecraft initiates its transfer at a true anomaly of 0, the second term encompassed by parentheses within the brackets will become 0. However, $a$, $e$, and $E_b$ were kept variable as they may change as the spacecraft’s state gets updated throughout the trajectory. These variables were calculated using parameters from before the trajectory was propagated with respect to the celestial bodies. First however, the angular momentum of the orbit respective to the primary celestial body was calculated by the product of the radius and tangential velocity component with respect to the primary celestial body as given according to the last known values. Then, the remaining variables mentioned were found with the following equations with all variables being centric to the primary celestial body:
\( e = 0, \quad (33a) \)
\[
\varepsilon = \frac{v^2}{2} - \frac{\mu}{r}, \quad (33b)
\]
\[
a = \frac{-\mu}{2\varepsilon}, \quad (33c)
\]
and
\[
E_b = \arccos \left( \frac{e + \cos \theta_b}{\sqrt{1 + e \cos \theta_b}} \right). \quad (33d)
\]
\( \varepsilon \) is the energy of the orbit. Then, the total angle covered by the secondary celestial body was given by Equation (8).

The initial true anomaly of the secondary celestial body was not found in the same manner as it was in the exact solution. This is due to the rendezvous point being defined as when the spacecraft reaches the secondary celestial body’s Sphere of Influence, which may not be at the same true anomaly for both. Figures 2 and 3 show that the angle formed by the spacecraft’s and secondary celestial body’s position vectors with respect to the primary celestial body was found by once again utilizing the law of cosines according to
\[
\xi = \arccos \left( \frac{D^2 + r_b^2 - r_{sb}^2}{2Dr_b} \right). \quad (34)
\]
\( \xi \) is called the rendezvous offset angle. Then, from the geometry, it was found that the initial true anomaly was given by
\[
\theta_s = (\theta_b + \xi) - \theta_T. \quad (35)
\]
Finally, the secondary celestial body’s true anomaly was found using Equation (9).

4.4 Combining Spacecraft States

As described in Chapter 2, once the spacecraft was propagated out to the region of numerical stability and its state was determined with respect to both celestial bodies, the next step would be to combine the different states into a single, updated state. Then, the state would be propagated out with respect to the primary body again, converted into being centric to the secondary celestial body and propagated out about the secondary body, the secondary celestial body would progress along its orbit, and the process would repeat until reaching the target or end condition. An approach that utilized the geometry of the spacecraft states was used to combine the two spacecraft states into one.

4.4.1 Attempt at Combining States

After each iteration, the spacecraft has essentially two different states: one that is respective to the primary celestial body and another to the secondary. This is shown in Figure 7.
Figure 7: Different spacecraft states resulting from different celestial bodies (not to scale)

Not only would the radius and position be different, but so will the velocities, among other parameters resulting from propagation about the different celestial bodies. However, since the rate of change of the true anomaly is relatively small, this was assumed to be equal for the two different states that result from propagation of the spacecraft due to the celestial bodies. Once the necessary updated parameters were obtained by combining the spacecraft states, an updated true anomaly was found. This is similar to the approach used by predictor-corrector methods. In order for the radii to be combined, the radius of the spacecraft with respect to the primary celestial body as a result of propagation about the secondary celestial body, $r_{j/s/\Theta}$, must be found. This was done simply by using the law of sines according to
such that

\[ \frac{r_{j/s/\oplus}}{\sin \lambda} = \frac{D}{\sin \psi}, \]  

(36a)

\[ r_{j/s/\oplus} = \frac{D\sin \lambda}{\sin \psi}. \]  

(36b)

The angles \( \lambda \) and \( \psi \) were found using Equations (19h) and (19b), respectively. Again, the subscript \( j/s/\oplus \) refers to the parameter resulting from the spacecraft’s propagation due to the secondary celestial body being converted such that it is respective to the primary celestial body. Then the updated radius was found using

\[ r_j = \mathbb{P}_r r_{j/\oplus} + \mathbb{S}_r r_{j/s/\oplus}. \]  

(37)

The scripts \( \mathbb{P} \) and \( \mathbb{S} \) represent the weights used in the linear combination. Section 4.4.2 discusses how these weights were found. In order to convert the velocities resulting from the spacecraft’s state with respect to the secondary celestial body to align with the local horizontal and vertical with respect to the primary celestial body, Equation (19d) was used in conjunction with the following formulas:

\[ \gamma_s = \arccos \left( \frac{V_{\perp j/s}}{V_{j/s}} \right), \]  

(38a)

\[ V_{r_{j/s/\oplus}} = V_{j/s} \cos(\nu + \gamma_s), \]  

(38b)

and

\[ V_{\perp j/s/\oplus} = V_{j/s} \sin(\nu + \gamma_s). \]  

(38c)
Then, similarly to how the updated radius was found, updated velocity components were found by

\[ V_{r_j} = \mathbb{P}_V V_{r_j/\Theta} + \mathbb{S}_V V_{r_j/s/\Theta'}, \quad (39a) \]

and

\[ V_{\perp_j} = \mathbb{P}_V V_{\perp_j/\Theta} + \mathbb{S}_V V_{\perp_j/s/\Theta}. \quad (39b) \]

The updated total velocity, \( V_j \), was found by taking the root-sum-square of the velocity components solved for in Equation (39). Then the updated true anomaly and flight path angle with respect to the primary celestial body were found by taking the following steps:

\[ h_j = r_j \times V_{\perp_j}, \quad (40a) \]

\[ e_j = \frac{V_j^2}{2} - \frac{\mu_\Theta}{r_j}, \quad (40b) \]

\[ e_j = \frac{2\varepsilon_j h_j^2}{\mu_\Theta^2} + 1, \quad (40c) \]

so

\[ \theta_j = \arccos \left( \frac{h_j^2}{r_j \mu_\Theta e_j} - \frac{1}{e_j} \right), \quad (40d) \]

and

\[ \gamma_{j/\Theta} = \arccos \left( \frac{V_{\perp_j}}{V_j} \right). \quad (40e) \]

With a combination scheme devised, the next step was to decide on what weights to use for \( \mathbb{P} \) and \( \mathbb{S} \).
4.4.2 Weighting Scheme

Clearly, when the spacecraft was close to the primary celestial body, much more emphasis needed to be put on the primary rather than the secondary celestial body, and vice versa. Initially, values that ‘seemed’ reasonable were estimated and put into the code as constants. This was attempted at three different points: near the primary celestial body, about halfway between the celestial bodies, and very near the secondary celestial body. For example, when the spacecraft was very near the primary celestial body the weights used were

- \( I^p = 0.99 \)
- \( I^s = 0.01. \)

This was not met with any success. Therefore, the subroutines that propagated the trajectories out with respect to either celestial body were tested on their own. The subroutine which propagated the spacecraft with respect to the primary celestial body was tested simply by setting \( I^p \) to 1 and \( I^s \) to 0. This produced an elliptical orbit around the primary celestial body as would be expected. Several data points from this test are shown in Figure 8.
Figure 8: Test of subroutine that propagates spacecraft with respect to the primary celestial body

A separate script was prepared to test the subroutine which propagated the spacecraft with respect to the secondary celestial body since this subroutine takes inputs that are centric to the primary celestial body and converts them to the secondary body’s frame. Therefore, the subroutine which propagates the spacecraft with respect to the primary celestial body was used to send the spacecraft to a point at which it was very near the secondary celestial body. This testing script is shown in Appendix A.2.5 and the resulting trajectory as seen from the secondary celestial body is shown in Figure 9.
Since the trajectory shown in Figure 9 is a conic section it was assumed with confidence that this subroutine was also functioning properly.

Therefore, a more elaborate approach was used to determine what weights would be more appropriate to use. At each point, a diagram relating the radii and velocity components to the exact solution obtained from MATLAB’s ODE solver was used to get an idea of how the vectors related with each other. This idea was used to visually get an understanding of how the vectors were related to each other and to hopefully motivate the use of certain weights. An example of the diagrams used is shown in Figures 10(a) and (b).
Figure 10: Diagrams used to estimate appropriate values for weights: (a) shows the radii, (b) the velocity components (scaled relatively, not exactly)

Figure 10(a) also shows an example of why the true anomaly was neglected when combining the weights, and also why it was assumed constant in the combination scheme discussed in Section 4.4.1. Figure 10(a) also motivated setting up an equation to determine one weight given another. For example, from the geometry \( \varpi_r \) was set to 0.985, and then \( S_r \) was found by

\[
0.985 \times 8.8684 + S_r \times 9.4001 = 8.6328, \quad (41a)
\]

which gives

\[
38
\]
The geometry in Figure 10(b) also motivated a mathematical approach. However, this time there were two equations and two unknowns, therefore the weights could be solved by

\[ \begin{align*}
P_v \times 0.4204 + S_v \times 0.6052 &= 0.4238 \\
P_v \times 0.1624 + S_v \times 0.2202 &= 0.1667,
\end{align*} \]

(42a)

giving

\[ \begin{align*}
P_v &= 1.3245 \\
S_v &= -0.2198.
\end{align*} \]

(42b)

Equations (41) and (42) not only provide weights that were determined mathematically, but they also are values that were not thought of when weights were being selected arbitrarily. Unfortunately, these weights did not yield any more desirable results for any of the three data points examined. This is discussed in greater detail in Chapter 5. This suggests there is a problem elsewhere in the algorithm, or that this methodology is physically flawed. Although this approach was simple, it was used merely in an attempt to search for any underlying trends that could be observed and utilized in constructing a more refined approach. This is discussed further in Chapter 5 as well as some initial results for weights at different points throughout the trajectory from utilizing this approach.
Chapter 5: Results and Forward Work

Significant progress was made towards the creation of a model of the restricted three-body problem without the use of the classic equations of motion. A numerically ‘exact’ solution was created for comparison, the architecture of the model was determined, and the model propagating the trajectory of the spacecraft with respect to both celestial bodies, as well as a method propagating the secondary celestial body was constructed and tested. This required an understanding of the complex geometry relating the spacecraft’s state with both celestial bodies at every point throughout the trajectory. Many difficulties were encountered, but all of them were overcome and equations relating the spacecraft’s state and effect of both celestial bodies were successfully applied.

Also, initial attempts were made at combining the spacecraft states with respect to the celestial bodies such that their effect would be balanced and an accurate trajectory could be obtained. A combination scheme based on the geometry of the spacecraft’s state with both celestial bodies was constructed and utilized a linear combination with weights. Multiple configurations of values for the weights were tested including arbitrary selection and selection based on diagrams and mathematical equations. Several sets of data used in finding weights are shown in Table 1. Figure 11 shows the radii as calculated by the sub-models with relation to the numerically exact solution.
Figure 11: Points used to calculate weights. Green points were obtained from the primary body sub-model, magenta points from the secondary

Table 1: Data used in weight calculations

<table>
<thead>
<tr>
<th></th>
<th>Exact Solution</th>
<th></th>
<th>Primary Model</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Radius (DU)</td>
<td>True Anomaly (deg)</td>
<td>Radial Velocity (SU)</td>
<td>Tangential Velocity (SU)</td>
</tr>
<tr>
<td>8.6328</td>
<td>141.23</td>
<td>0.4238</td>
<td>0.1667</td>
<td></td>
</tr>
<tr>
<td>20.4099</td>
<td>157.19</td>
<td>0.2625</td>
<td>0.0705</td>
<td></td>
</tr>
<tr>
<td>30.4719</td>
<td>162.98</td>
<td>0.1987</td>
<td>0.0474</td>
<td></td>
</tr>
<tr>
<td>46.1144</td>
<td>168.49</td>
<td>0.1389</td>
<td>0.032</td>
<td></td>
</tr>
<tr>
<td>55.9049</td>
<td>171.03</td>
<td>0.1243</td>
<td>0.0295</td>
<td></td>
</tr>
</tbody>
</table>
Applying this data to the scheme given in Section 4.4.2 the following weights shown in Table 2 and Figure 12 were obtained.

Table 2: Calculated weights

<table>
<thead>
<tr>
<th>Radius Weights</th>
<th>Velocity Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary</td>
<td>Secondary</td>
</tr>
<tr>
<td>0.985</td>
<td>-0.0108</td>
</tr>
<tr>
<td>0.95</td>
<td>0.0418</td>
</tr>
<tr>
<td>0.9</td>
<td>0.0851</td>
</tr>
<tr>
<td>0.6</td>
<td>0.38297</td>
</tr>
<tr>
<td>0.3522</td>
<td>0.65</td>
</tr>
</tbody>
</table>
Figure 12: Calculated weights vs. their respective positions
Unfortunately, none of these weights when applied to the scheme described in Section 4.4.1 were met with any success. At all of the points examined, as shown in Figure 11, the combination scheme only lasted for a few iterations before failing. Either the spacecraft became ‘stuck’ and the radius would not increase at an appropriate rate, or MATLAB began producing NaN’s. The cause for these failures is not clear. Further investigation ensuring that there are no ‘bugs’ in how the code is executing is required. Also, the manner in which the scheme is applied should also be examined more closely. The problem is likely to be a simple coding error, a mistake in the coding logic, and/or the problem is overconstrained in selection of the weights. Fortunately, the subroutines that propagate the spacecraft trajectory out with each celestial body were proven to work correctly. Therefore it is logical to assume the error is most likely located in the master script that calls the subroutines and operates the combination scheme as discussed in Section 4.4.1.

Whether the error is something as simple as a misplaced sign or something more complicated such as logic, the foundation of work that has been created here will make it much easier for a future investigator to seek out the problem and correct it.

If and when a solution is found, there will be many benefits. Trajectories could be created using popular analysis tools such as MATLAB without worrying about the equations of motion, the material is simple enough that students just learning orbital mechanics can gain an understanding of the kinematics involved with having a second celestial body, and the problem could potentially be expanded to include n-bodies. Since a separate algorithm is included for each celestial body, all that may be required would be
the addition of another algorithm and an extra term in the weighting scheme to include respective spacecraft states with respect to the additional celestial body and a respective weight. The significance of such a model and its benefits are immeasurable.
References


5. Öz, Hayrani, Hyperbolic Approach (Fly-By) and Escape To/From a Planet, The Ohio State University.

Appendix A: MATLAB Scripts and Functions

A.1 Exact Solution Code

A.1.1 Master Script for the Exact Solution

% Phobos_Analytic uses MATLAB's ODE solver, ode113, to solve the
% equations of motion for the restricted three-body problem. This code
% was originally created for the Mars-Phobos system, but adjustments
% were made such that the Earth-Moon system could be solved as well.
% This code works in conjunction with the function Mars_EOM.
%
% Created By:  Richard Jedrey (5/15/2011)

% Mars-Phobos setup
% [tout yout]=ode45(@Mars_EOM,[0 3999.5],[3488.1 0 0 .0013])

% Earth-Moon setup
[tout yout]=ode113(@Mars_EOM,[0 1000],[1.05 0 0 1.44/1.05^2]);

polar(yout(:,2),yout(:,1),'b')

A.1.2 Equations of Motion as a System, Mars_EOM

function X=Mars_EOM(t,x)

% Mars_EOM sets up the equations of motion for either the Mars-Phobos
% or Earth-Moon system by making them a system of first-order
% differential equations.
%
% SYNTAX:
%   X=Mars_EOM(t,x)
%
% Inputs:
%   x    Vector containing state variables and their first
derivatives
%   t    Time
%
% Output:
%   X    Vector containing the first and second derivatives of the
state variables
%
% See Reference 6 for astronomical constants used
% % Created By: Richard Jedrey (5/15/2011)

%mu_Mars=42814; % km^3/s^2 - Gravitational Parameter of Mars
mu_Mars=1; % Canonical - Gravitational Parameter of Earth

%D=9378; % km - Distance of Mars to Phobos
D=384400/6378.145; % Canonical - Distance of Earth to Moon

%m_Pho=7.1134e-4; % km^3/s^2 - mu for Phobos
%m_Pho=10000; % Forced value for mu to see effect on trajectory
m_Pho=4.90287e3/3.986012e5; % Canonical - mu for Moon

m_0=11000; % kg - Hard-coded s/c Initial Mass
m_dot=0; % kg/s - Hard-coded s/c Prop Consumption Rate

theta_b=2.2067; % rad - Hard-coded True Anomaly at Rendezvous for 3999.5 s TOF
theta_b=2.95; % rad - True anomaly for rendezvous with Earth's Moon
theta_T=.2904*pi; % rad - Hard-coded Total True Anomaly Covered by Phobos During Transfer for 3999.5 s TOF

TOF=2.137e-3*223.98; % Canonical - Angle covered by Earth's Moon during TOF

theta_i=theta_b-theta_T;
m_s=m_0-m_dot*t;
theta_Pho=t/TOF*theta_T+theta_i;
d_squared=x(1)^2+D^2-2*x(1)*D*cos(abs(x(2)-theta_Pho));
T=-m_Pho*m_s/d_squared;

X(1)=x(3);
X(2)=x(4);

alpha=acos((D^2+d_squared-x(1)^2)/(2*D*sqrt(d_squared))));
gamma=pi-(x(2)-theta_Pho)-alpha;
phi=pi-gamma;
beta=pi/2+phi;
X(3)=x(1)*x(4)^2-mu_Mars/x(1)^2-T*sin(beta)/m_s;
X(4)=-2*x(3)*x(4)/x(1)+T/(m_s*x(1))*cos(pi-beta);
X=X';
polar(theta_Pho,D,'r')
hold on

% Section used to calculate the semimajor axis of the spacecraft-secondary celestial body orbit
V_Pho=.1288; % Canonical - Velocity of Earth's Moon

r_Pho=sqrt(x(1)^2+D^2-2*x(1)*D*cos(theta_Pho));
h=x(1)^2*x(4);
V_sc_Mars_perp=h/x(1);
V_sc_Mars=sqrt(x(3)^2+V_sc_Mars_perp^2);

FPA=asin(x(3)/V_sc_Mars);
trident=asin(D*sin(theta_Pho-x(2))/r_Pho);
mu=pi-trident-pi/2;
FPA_Pho=pi/2-FPA-mu;
V_sc_Pho_perp=V_sc_Mars*cos(FPA_Pho)-V_pho;
V_sc_Pho_rad=-V_sc_Mars*sin(FPA_Pho);
V_sc_Pho=sqrt(V_sc_Pho_perp^2+V_sc_Pho_rad^2);
a=1/(2/r_Pho-V_sc_Pho^2/mu_Pho); % Used in conjunction with if statement to determine when s/c reaches SOA
if x(1)>55 % Used to pause code at any point to check s/c state
  % keyboard
end

A.2 Restricted Three-Body Discretized Model Code

A.2.1 Master Script for the Restricted Three-Body Discretized Model Code

% disc_model_alg is the master script for the Discretized Restricted Three-Body Problem model. It establishes the initial conditions for both celestial bodies and the spacecraft, and calls the necessary function files. It also attempts to apply a weighting scheme until the spacecraft reaches its destination.
%
% Created By: Richard Jedrey (5/20/2011)

clear
clc
close all

% Initial conditions of spacecraft to get to SOA
r_A=1.05; V_A=1.372; gamma_A=0; theta_A=0; mu=1; Me_A=0;
dt_SOA=10.5119108540282; % dt required to reach Sphere of Accuracy
%d_SOA=262;

% Properties of secondary body, conditions at arrival, and transition time of model
D=60.27; r_SOI=10.395; V_m=1.018/7.90536828;
mu_m=mu/(5.974e24/7.36e22); % See Reference
lambda_B=30*pi/180;
dt=4.4786719594346; % Transition time after reaching SOA
%d_t=2;

[theta_0 r_0 V_0_perp V_0_r V_0 gamma_0
Me_0]=primary_prop(r_A,V_A,gamma_A,theta_A,dt_SOA,mu,Me_A);
$$h_A = r_A V_A \cos(\gamma_A);$$
$$e_A = h_A^2/(\mu r_A) - 1;$$
$$w = 2.137 e^{-3};$$
$$Epsilon_A = V_A^2/2 - \mu/r_A;$$
$$a_A = -\mu/(2 Epsilon_A);$$

```matlab
[theta_m] = SBody_prop(D, r_SOI, lambda_B, dt_SOA, h_A, e_A, mu, w, a_A, theta_A, counter); % Weighting scheme
j1 = 1; % Primary Celestial Body's Weight for radius
k1 = 0; % Secondary Celestial Body's Weight for radius
j2 = 1; % Primary Celestial Body's Weight for velocity components
k2 = 0; % Secondary Celestial Body's Weight for velocity components

% Combination Scheme
trident_1 = pi - asin(D * sin(abs(theta_m - theta_1))/r_1_m); % Primary Celestial Body's Weight for radius
lambda_1 = pi - real(trident_1); % Secondary Celestial Body's Weight for radius
r_1_conv = D * sin(lambda_1)/sin(trident_1); % Primary Celestial Body's Weight for velocity components
r_1_f(counter) = j1 * r_1 + k1 * r_1_conv; % Secondary Celestial Body's Weight for velocity components

gamma_m = acos(V_1_perp_m/V_1_m); % Primary Celestial Body's Weight for radius
nu_1 = pi - trident_1 - pi/2; % Secondary Celestial Body's Weight for radius
V_1_r_conv = V_1_m * cos(nu_1 + gamma_m); % Primary Celestial Body's Weight for velocity components
V_1_perp_conv = V_1_m * sin(nu_1 + gamma_m); % Secondary Celestial Body's Weight for velocity components
V_1_r_f = j2 * V_1_r + k2 * V_1_r_conv; % Primary Celestial Body's Weight for velocity components
V_1_perp_f = j2 * V_1_perp + k2 * V_1_perp_conv; % Secondary Celestial Body's Weight for velocity components
V_1_f(counter) = sqrt(V_1_r_f^2 + V_1_perp_f^2); % Primary Celestial Body's Weight for velocity components

h_f = r_1_f(counter) * V_1_perp_f; % Secondary Celestial Body's Weight for velocity components
```
Energy_f=V_1_f(counter)^2/2-mu/r_1_f(counter);
e_f=sqrt(2*Energy_f*h_f^2/mu^2+1);
theta_f(counter)=acos(h_f^2/(r_1_f(counter)*mu*e_f)-1/e_f);
gamma_f=acos(V_1_perp_f/V_1_f(counter));
polar(theta_m,D,'r')
hold on
polar(theta_f(counter),r_1_f(counter),'b+')
hold on

% Reset values
r_0=r_1_f(counter); V_0=V_1_f(counter); gamma_0=gamma_f;
theta_0=theta_f(counter);
Me_0=Me_1;
end

% Display Results
Table=[r_1_f' V_1_f' theta_f']

A.2.2 Propagation of Spacecraft with respect to Primary Celestial Body, primary_prop

function [theta_1 r_1 V_1_perp V_1_r V_1 gamma_1…
    Me_1]=primary_prop(r_0,V_0, gamma_0,theta_0,dt,mu,Me_0)

% primary_prop propagates the trajectory of a spacecraft under the
% influence of a primary celestial body (in the case of multiple
% bodies) using Kepler's elliptic time of flight formulation for
% elliptical orbits or the universal formulation for hyperbolic
% orbits.
% SYNTAX:
% [theta_1 r_1 V_1_perp V_1_r V_1 gamma_1…
% Me_1]=primary_prop(r_0,V_0, gamma_0,theta_0,dt,mu,Me_0)
% Inputs:
% r_0 Initial radius
% V_0 Initial velocity
% gamma_0 Initial flight path angle
% theta_0 Initial true anomaly
% dt Time step
% mu Gravitational parameter of primary celestial body
% Me_0 Initial mean eccentric anomaly
% Outputs:
% theta_1 Updated true anomaly
% r_1 Updated radius
% V_1_perp Updated tangential velocity component
% V_1_r Updated radial velocity component
% V_1 Updated velocity
% gamma_1 Updated flight path angle
% Me_1 Updated mean eccentric anomaly
% Created By: Richard Jedrey (5/15/2011)

h = r_0 * V_0 * cos(gamma_0);
e = h^2 / (mu * r_0 * cos(theta_0)) - 1 / cos(theta_0);
flag = 1; % flag = 1 for elliptical orbits
if e >= 1
    flag = 2; % flag = 2 for parabolic and hyperbolic orbits
end

if flag == 1
    Me_1 = mu^2 / h^3 * (1 - e^2)^(1.5) * dt + Me_0;
elseif flag == 2
    F_0 = 2 * atanh(sqrt((e - 1) / (e + 1)) * tan(theta_0 / 2));
    Me_1 = [];
end

if flag == 1
    E = kepler_E(e, Me_1);
    theta_1 = acos((e - cos(E)) / (e * cos(E) - 1));
elseif flag == 2
    V_1_r = V_0 * sin(gamma_0);
a_0 = h^2 / (mu * (1 - e^2));
X = kepler_U(dt, r_0, V_0_r, a_0, mu);
del_F = X / sqrt(-a_0);
F_1 = F_0 + del_F;
theta_1 = 2 * atan(sqrt((e + 1) / (e - 1)) * tanh(F_1 / 2));
end

r_1 = (h^2 / mu) / (1 + e * cos(theta_1));
V_1_perp = h / r_1;
V_1_r = mu * e * sin(theta_1) / h;
V_1 = sqrt(V_1_perp^2 + V_1_r^2);
gamma_1 = acos(h / (r_1 * V_1));

A.2.3 Propagation of Spacecraft with Respect to Secondary Celestial Body,

secondary_prop

function [r_1_m, theta_1_m, V_1perp_m, V_1r_m, V_1_m, e_1_m] = secondary_prop(r_0, gamma_0, D, theta_0, theta_m, V_0, V_m, mu_m, dt)

% secondary_prop takes the spacecraft's primary-body based state and
% converts it to being centered on the secondary body. It then
% propagates that trajectory out as if it were only under the influence
% of the secondary body.
% SYNTAX:
% [r_1_m, theta_1_m, V_1perp_m, V_1r_m, V_1_m, e_1_m] = secondary_prop(r_0, gamma_0, D, theta_0, theta_m, V_0, V_m, mu_m, dt)
% Inputs:
% r_0     Initial radius
% gamma_0 Initial flight path angle
% D      Distance between the primary and secondary celestial bodies
% theta_0 Initial true anomaly
% theta_m True anomaly of secondary celestial body
% V_0     Initial velocity
% V_m     Velocity of secondary celestial body
% mu_m   Gravitational parameter of primary celestial body
% dt     Time step

% Outputs:
% r_1_m    Updated radius
% theta_1_m Updated true anomaly
% V_1perp_m Updated tangential velocity component
% V_1r_m   Updated radial velocity component
% V_1_m    Updated velocity
% e_1_m    Updated eccentricity

% Created By: Richard Jedrey (5/20/2011)

r_0_m=sqrt(r_0^2+D^2-2*r_0*D*cos(theta_0-theta_m));

trident=pi-asin(D*sin(abs(theta_m-theta_0))/r_0_m);
trident=real(trident);
mu=pi-trident-pi/2;
FPA_0_m=(pi/2-gamma_0-nu);
lambda=180-abs(theta_m-theta_0)-trident;
V_0perp_m=V_0*cos(FPA_0_m)+V_m*cos(-pi/2+theta_m+lambda);
V_0r_m=V_0*sin(FPA_0_m)+V_m*sin(-pi/2+theta_m+lambda);
V_0_m=sqrt(V_0perp_m^2+V_0r_m^2);
a_0_m=1/(2/r_0_m-V_0_m^2/mu_m);
alpha_0_m=1/a_0_m;
X_1_m=kepler_U(dt,r_0_m,V_0r_m,alpha_0_m,mu_m);
delF_1_m=X_1_m/sqrt(-a_0_m);

h_0_m=r_0_m*V_0perp_m;
e_0_m=sqrt(h_0_m^2/(r_0_m*mu_m*e_0_m)-1/e_0_m);
theta_0_m=acos(h_0_m^2/(r_0_m*mu_m*e_0_m)-1/e_0_m);
F_0_m=2*atanh(sqrt((e_0_m-1)/(e_0_m+1))*tan(theta_0_m/2));
theta_sign=sign(theta_0_m);
F_1_m=F_0_m-theta_sign*delF_1_m;
theta_1_m=sign(theta_0_m)*2*atan(sqrt((e_0_m+1)/
                                 (e_0_m-1))*tanh(F_1_m/2));
r_1_m=h_0_m^2/(mu_m*(1+e_0_m*cos(theta_1_m)));

V_1perp_m=h_0_m/r_1_m;
V_1r_m=mu_m*e_0_m*sin(theta_1_m)/h_0_m;
V_1_m=sqrt(V_1perp_m^2+V_1r_m^2);
h_1_m=r_1_m*V_1perp_m;
a_1_m=1/(2/r_1_m-V_1_m^2/mu_m);
e_1_m=sqrt(-h_1_m^2/(a_1_m*mu_m)+1);

A.2.4 Propagation of Secondary Celestial Body, SBody_prop

function [theta_m]=...

SBody_prop(D,r_SOI,lambda_B,dt,h,e,mu,w,a,theta_A,counter)

% SBody_prop finds the location of the secondary celestial body (SBody)
% after any elapsed time of flight. It is assumed the secondary body
% has a circular orbit about the primary.
% SYNTAX:
% [theta_m]=SBody_prop(D,r_SOI,lambda_B,dt,h,e,mu,w,a,theta_A)
% Inputs:
% D Distance between primary and secondary celestial bodies
% r_SOI Radius of the Sphere of Influence
% lambda_B Angle formed between the position vectors connecting the
% spacecraft and primary celestial body with the secondary
% dt Time step
% h Angular Momentum of spacecraft's orbit with respect to
% the primary celestial body
% e Eccentricity of the spacecraft's orbit with respect to
% the primary celestial body
% mu Gravitational parameter of primary celestial body
% w Angular speed of secondary celestial body
% a Semimajor axis
% theta_A True anomaly of spacecraft at burn initiation
% counter Iteration used to determine the fraction of the total
% time of flight covered
% Outputs:
% theta_m True anomaly of secondary celestial body
% Created By: Richard Jedrey (5/20/2011)

r_b=sqrt(D^2+r_SOI^2-2*D*r_SOI*cos(lambda_B));
theta_b=acos((h^2/mu)/(r_b*e)-(1/e)); % True Anamoly of the spacecraft at

E_A=acos((e+cos(theta_A))/(1+e*cos(theta_A)));
E_b=acos((e+cos(theta_b))/(1+e*cos(theta_b)));
T=sqrt(a^3/mu)*((E_b-e*sin(E_b))-E_A+e*sin(E_A));
P=2*pi/w;
theta_T=(T/P)*2*pi;
arrival_offset_angle=acos((D^2+r_b^2-r_SOI^2)/(2*D*r_b));
theta_m_0=theta_b+arrival_offset_angle-theta_T;

theta_m=(counter*dt/T)*theta_T+theta_m_0;

A.2.5 Script Used to Test secondary_prop

% secondary_test tests secondary_prop by using a proven primary_prop to
% create an initial condition that secondary_prop iterates over several
% times using an increasing transition time. Then, the results are
% plotted with respect to the secondary celestial body.
%
% Created By: Richard Jedrey (5/20/2011)
clear
r_A=1.05; V_A=1.372; gamma_A=0; theta_A=0; mu_A=0;
dt_SOA=275.519108540282; D=60.27; r_SOI=10.395; V_m=1.018/7.90536828;
mu_m=mu/(5.974e24/7.36e22); % See Reference^6
lambda_B=30*pi/180;
h_A=r_A*V_A*cos(gamma_A);
e_A=h_A^2/(mu*r_A)-1;
w=2.137e-3;
Epsilon_A=V_A^2/2-mu/r_A;
a_A=-mu/(2*Epsilon_A);
[theta_0 r_0 V_0_perp V_0 r V 0 gamma_0 Me_0]=primary_prop(r_A,V_A,gamma_A,theta_A,dt_SOA,mu,Me_A);
[theta_m]=SBody_prop(D,r_SOI,lambda_B,dt_SOA,h_A,e_A,mu,w,a_A,...
  theta_A,1);

dt=5;
for i=1:100
  [r_1_m(i) theta_1_m(i) V_1perp_m V_1r_m V_1_m e_1_m]...
    =secondary_prop(r_0,theta_0,D,theta_0,theta_m,V_0,V_m,mu_m,dt);
dt=dt+5;
end
polar(theta_1_m,r_1_m,'b+')

A.2.6 Kepler's Equation for an elliptical orbit, Kepler_E

function E=kepler_E(e,M)
% This function uses Newton's method to solve Kepler's equation,
% E-e*sin(E)=M, for the eccentric anomaly given the eccentricity and
% the mean anomaly.
%
% SYNTAX:
%   E=kepler_E(e,M)
%
% Inputs:
%   e: Eccentricity
%   M: Mean Anomaly (rad)
%
% Output:
%   E: Eccentric Anomaly (rad)
%
% Credit: See Reference
%
% Written By: Richard Jedrey 3/1/2011

% Set an error tolerance
error=1e-12;

% Select a starting value for E:
if M<pi
    E=M+e/2;
else
    E=M-e/2;
end

% Iterate until E is determined to within the error
tolerance
t=1;
while abs(t)>error
    t=(E-e*sin(E)-M)/(1-e*cos(E));
    E=E-t;
end

A.2.7 Universal Time Formulation, Kepler_U

function x=kepler_U(dt,ro,vro,a,mu)

% This function uses Newton's method to solve the universal
% formulation for the universal anomaly.
%
% mu - gravitational parameter (km^3/s^2)
% x - the universal anomaly (km^.5)
% dt - time since x=0 (s)
% ro - radial position (km) when x=0
% vro - radial velocity (km/s) when x=0
% a - reciprocal of the semimajor axis (1/km)
% z - auxiliary variable (z=a*x^2)
% C - value of Stumpff function C(z)
% S - value of Stumpff function S(z)
% n - number of iterations for convergence
% nMax - maximum allowable number of iterations
%
% Written by: Richard Jedrey (3/28/2011)
% Credit: See Reference

% Set an error tolerance and a limit on the number of iterations
error=1e-2;
nMax=1000000;

% Starting value for x:
x=sqrt(mu)*abs(a)*dt;

% Iterate until convergence occurs within the error % tolerance
n=0;
ratio=1;
while abs(ratio)>error && n<=nMax
    n=n+1;
    C=stumpC(a*x^2);
    S=stumpS(a*x^2);
    F=ro*vro/sqrt(mu)*x^2*C+(1-a*ro)*x^3*S+ro*x-sqrt(mu)*dt;
    dFdx=ro*vro/sqrt(mu)*x*(1-a*x^2*S)+(1-a*ro)*x^2*C+ro;
    ratio=F/dFdx;
    x=x-ratio;
end

% Deliver a value for x, but report that nMax was reached:
if n>nMax
    fprintf('
**# iterations of universal…
formulation')
    fprintf(' = %g',n)
    fprintf('
F/dFdx = %g',F/dFdx)
end