SPACECRAFT TRAJECTORIES IN A SUN, EARTH, AND MOON EPHEMERIS MODEL

A Project

Presented to

The Faculty of the Department of Aerospace Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Aerospace Engineering

by

Romalyn Mirador

ABSTRACT

SPACECRAFT TRAJECTORIES IN A SUN, EARTH, AND MOON EPHEMERIS MODEL

by Romalyn Mirador

This project details the process of building, testing, and comparing a tool to simulate spacecraft trajectories using an ephemeris N-Body model. Different trajectory models and methods of solving are reviewed. Using the Ephemeris positions of the Earth, Moon and Sun, a code for higher-fidelity numerical modeling is built and tested using MATLAB. Resulting trajectories are compared to NASA's GMAT for accuracy. Results reveal that the N-Body model can be used to find complex trajectories but would need to include other perturbations like gravity harmonics to model more accurate trajectories.

ACKNOWLEDGEMENTS

I would like to thank my family and friends for their continuous encouragement and support throughout all these years. A special thank you to my advisor, Dr. Capdevila, and my friend, Dhathri, for mentoring me as I work on this project. The knowledge and guidance from the both of you has helped me tremendously and I appreciate everything you both have done to help me get here.

Table of Contents

List of Symbols v

1.0 INTRODUCTION 1

1.1 Project Objective 1

1.2 Literature Review 1

1.2.1 Different Models 1

1.2.2 Approaches to Solutions 4

1.2.3 Use Cases .... **Error! Bookmark not defined.**

2.0 THE N-BODY MODEL 7

2.1 Derivation of Equations of Motion 7

3.0 METHODOLOGY 12

3.1 MATLAB 12

3.2 SPICE and HORIZONS 12

3.3 Benchmark Data 14

3.4 GMAT 16

4.0 PROCESS 18

4.1 MATLAB and SPICE 18

4.1.1 Code Flowchart 18

4.1.2 Tool Correction 19

4.2 GMAT 21

5.0 RESULTS 24

5.1 Circular Orbit Results 24

5.1.1 MATLAB Results for Circular Orbit Case 24

5.1.2 GMAT Results for Circular Orbit Case 25

5.1.3 MATLAB and GMAT Comparisons for Circular Orbit Case 26

5.2 LADEE Trajectory Results 27

5.2.1 MATLAB Results for LADEE Trajectory Case 27

5.2.2 GMAT Results for LADEE Trajectory Case 29

5.2.3 MATLAB and GMAT Comparisons for LADEE Trajectory Case 31
6.0 SUMMARY 33
6.1 Future Work 33
References 34
Appendix A. SPICE/MICE Functions 36
Appendix B. MATLAB Function File for N-Body Equations of Motion 37
Appendix C. MATLAB Function File for Fixed-Step Runge-Kutta Method 39
Appendix D. MATLAB Script for N-Body Trajectories, Plots and Comparisons 40

List of Symbols

Symbol	**Definition**	**Units (SI)**
$\vec{f}$	Forces exerted on body of interest	N
G	Universal Gravitational Constant	$km^3/kg\ s^2$
m	Mass	m
P	Particle/Body	
r	Magnitude of position vector	km
$\vec{r}$	Position vector	km
$\dot{\vec{r}}$	Velocity vector	km/s
$\ddot{\vec{r}}$	Acceleration vector	km/s^2
x	x position Cartesian coordinate	km
y	y position Cartesian coordinate	km
z	z position Cartesian coordinate	km
$\dot{x}$	x velocity Cartesian coordinate	km/s
$\dot{y}$	y velocity Cartesian coordinate	km/s
$\dot{z}$	z velocity Cartesian coordinate	km/s
$\ddot{x}$	x acceleration Cartesian coordinate	km/s^2
$\ddot{y}$	y acceleration Cartesian coordinate	km/s^2
$\ddot{z}$	z acceleration Cartesian coordinate	km/s^2
Subscripts		
$()_i$	Body of interest (Spacecraft)	
$()_j$	Additional bodies in system	
$()_q$	Central body (Earth)	
$()_{ij}$	Motion of body of interest relative to additional body	
$()_{iq}$	Motion of body of interest relative to central body	
$()_{qj}$	Motion of central body relative to additional body	
$()_0$	Initial condition	
$()_1$	Additional body in system (Moon)	
$()_2$	Additional body in system (Sun)	
Acronyms		
2BP	Two-Body Problem	
GMAT	General Mission Analysis Tool	
ICRF	International Celestial Reference Frame	
LADEE	Lunar Atmosphere and Dust Environment Explorer	
NAIF	Navigation and Ancillary Information Facility	
ODE	Ordinary Differential Equation	
RK	Runge-Kutta	

1.0 INTRODUCTION

1.1 Project Objective

Finding accurate spacecraft trajectory, especially in regimes where multiple bodies are acting on the spacecraft, is not a trivial task. The Two-Body Problem (2BP) is used to model spacecraft trajectories in the close vicinity of a heavenly body. However, the 2BP does not take gravitational effects from other large bodies or other perturbations like solar radiation pressure, into account. When more bodies are added, the model becomes more complex, albeit accurate.

This project develops a higher-fidelity model - an ephemeris N-Body model. Using the Ephemeris positions of the Earth, Moon and Sun, the goal of this investigation is to build and test a code of a higher-fidelity numerical model for trajectory design. The tool is tested and compared to NASA's General Mission Analysis Tool (GMAT) for accuracy.

1.2 Literature Review

Since the 1600s, Johannes Kepler's research into planetary motion combined with Newton's Laws have shaped the ideas of orbital mechanics and spaceflight [1]. Kepler and Newton's laws are the basis of spacecraft trajectory models that have been extensively studied and researched today.

1.2.1 Different Models

The most used model is the 2BP as it has a complete analytical solution. Practically, orbital mechanics problems can be solved by using the 2BP model as a starting point to calculate approximate parameters and trajectories [2]. For example, an article presented by Hsiao et al. [3]

investigates spacecraft trajectories with a photonic laser propulsion system. The authors use the 2BP to help design the spacecraft mission.

Derivation for the two-body problem can found in Bate et al. [1, pp.11-14] using the assumptions:

- The two bodies are spherically symmetric
- No other external or internal forces are acting on the system

The equation of motion for the 2BP is presented in equation 1.1 where $\ddot{\vec{r}}$ is the acceleration vector, $\vec{r}$ is the position vector, r is the magnitude of the position, and μ is the gravitational parameter ($\mu = GM$).

$$\ddot{\vec{r}} + \frac{\mu}{r^3}\vec{r} = 0 \tag{1.1}$$

Adding one more body into the system, the resulting three-body problem arises. A known three-body model is called the Circular Restricted Three Body Problem (CRTBP). A comprehensive text on and full derivation of the CRTBP can be found in *The Theory of Orbits* by Victor Szebehely [4]. From the text, the following simplifying assumptions are used for derivation of the CRTBP:

- Two of the bodies revolve in a circular orbit about their common center of mass (barycenter)
- The two revolving bodies are considered point masses
- The third body has a mass much smaller than the other two bodies that it does not affect the motion of the larger bodies

The CRTBP is one of the models that is seen commonly used for trajectory design in multi-body dynamical regimes. An example of CRTBP's use in recent studies comes from Pavlak's dissertation [5]. A set of nondimensional, scalar, second order ODE equations of motion of a body of interest (the spacecraft) was derived and presented here in equations 1.2 to 1.4. [5]

$$\ddot{x} - 2\dot{y} - x = -\frac{(1-\mu)(x+\mu)}{\left(\sqrt{(x+\mu)^2+y^2+z^2}\right)^3} - \frac{\mu(x-1+\mu)}{\left(\sqrt{(x-1+\mu)^2+y^2+z^2}\right)^3} \tag{1.2}$$

$$\ddot{y} + 2\dot{x} - y = -\frac{(1-\mu)y}{\left(\sqrt{(x+\mu)^2+y^2+z^2}\right)^3} - \frac{\mu y}{\left(\sqrt{(x-1+\mu)^2+y^2+z^2}\right)^3} \tag{1.3}$$

$$\ddot{z} = -\frac{(1-\mu)z}{\left(\sqrt{(x+\mu)^2+y^2+z^2}\right)^3} - \frac{\mu z}{\left(\sqrt{(x-1+\mu)^2+y^2+z^2}\right)^3} \tag{1.4}$$

In these equations, μ represents the nondimensional mass parameter of the mass of a second body, m_2, divided by the characteristic mass, m^*. Pavlak uses these derived equations of motion to better prepare for its numerical integration and to easily compare solutions of different three body systems. [5]

Other examples of CRTBP use includes:

- A thesis by Brick [6] that explores high altitude satellite trajectories to remotely sense the surface of the Earth for military applications (such as surveillance and global communications)
- An article by Tantardini et al. [7] discussing possible missions to the third libration point, L_3, to insert a space observatory satellite

Other variations of equations of motion for the general three body problem have also been derived. For example, in an article presented by Bombardelli and Bernal Mencia [8], the CRTBP equations of motion are derived in cylindrical curvilinear coordinates instead of Cartesian coordinates. With the new derivation, a new analytical tool and an expansion to the averaged third body disturbing function were found to aid in their study of co-orbital motion. [8]

When deriving the equations of motion for multiple bodies without using simplifying assumptions (as in the case for the 2BP or CRTBP), the resulting derivation gives the higher-fidelity N-body model. The N-Body model can consider all gravitational and other external forces (such as drag and solar radiation pressure) acting on the body of study. A derivation of the N-body model is seen in Section 2.1 of this paper based on the derivation from Roy [2] and Capdevila [9].

1.2.2 Numerical Methods

Unlike the 2BP where a closed form solution can be found, models with more than 2 bodies present challenges in obtaining a solution. From the reviewed research, the common concept to solve for multi-body trajectory problems is to approximate solutions using numerical methods. This could be done by building a code or by using applications that already have a library of numerical integration solvers.

In a thesis presented by Wilmer [10], Wilmer uses MATLAB and its library of solvers to numerically integrate the CRTBP equations of motion found in his thesis. One of MATLAB's ODE solvers (ODE45) is based on an explicit Runge-Kutta formula, the Dormand-Prince pair [11]. The Runge-Kutta formula is used to solve for a first order system of ODEs in the form of equation 1.5 and a known initial condition, $y(x_0)$.

$$y'(x) = f[x, y(x)] \tag{1.5}$$

The explicit Runge -Kutta formula [12] is given by equation 1.6 to approximate the solution, y_n, at points, x_n, where $x_{n+1} = x_n + h_n, h_n = \theta(h_n)h$ and $0 < \theta(x_n) < 1$.

$$y_{n+1} = y_n + h_n\Phi(y_n, h_n) = y_n + \sum_{i=1}^{s} b_i k_i \tag{1.6}$$

$$k_1 = h_n f(y_n)$$

$$k_i = h_n f\left(y_n + \sum_{j=1}^{i-1} a_{ij} k_j\right)$$

A similar numerical method also in the Runge-Kutta family is the Fixed-Step Runge-Kutta (RK) function. A known and commonly used fixed-step RK method is the classic fourth order RK method (RK4) detailed by Cheever [13]. Similar to the explicit RK formula for ODE45, the RK4 method approximates a solution for a first order ODE in the form of equation 1.7 and initial condition of $y(t_0) = y_0$.

$$\frac{dy(t)}{dt} = y'(t) = f(y(t), t) \tag{1.7}$$

To find the estimation of the solution, $y(t)$, slope estimates are calculated first with a fixed step-size of *h* (shown in equations 1.8 to 1.11) and then summed in equation 1.12.

$$k_1 = f(y(t_0), t_0) \tag{1.8}$$

$$k_2 = f(y(t_0) + k_1 \frac{h}{2}, t_0 + \frac{h}{2}) \tag{1.9}$$

$$k_3 = f(y(t_0) + k_2 \frac{h}{2}, t_0 + \frac{h}{2}) \tag{1.10}$$

$$k_4 = f(y(t_0) + k_3 h, t_0 + h) \tag{1.11}$$

$$y(t_0 + h) = y(t_0) + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6} h \tag{1.12}$$

Scott and Martini [14] tested another numerical integrator. The authors used a Taylor Series integration and NASA’s Spacecraft N-body Analysis Program (SNAP) to calculate spacecraft trajectories. By using the Taylor series, a solution is found quicker than using an eighth-order Runge-Kutta Fehlberg method. [14]

Other approaches to finding a spacecraft’s trajectory does not always use a numerical integrator. For example, a method described by Zhang et al. [15] is to use a periodic orbit computation. The article describes utilizing the symmetry of the CRTBP to derive a new kind of periodic orbit computation that does not require a starting analytic approximation or state transition matrix. Erwin and Bernstein [16] takes an approach to use incoming data to estimate a target satellite’s trajectory. Using an extended Kalman filter algorithm and range measurements from the other satellites in orbit, the authors’ found an estimation of the eccentricity and inclination of the target satellite’s orbit.

2.0 THE N-BODY MODEL

2.1 Derivation of Equations of Motion

The geometry of *n* number of bodies (depicted by *P*) of mass, *m* in a Newtonian (or non-accelerating) frame is shown in Figure 2.1. Motions of the bodies in inertial space are relative to a non-accelerating point, *O*. Positions of the particles relative to the origin, *O*, are denoted by $\vec{r_i}$ and $\vec{r_j}$. Positions of the particles relative to the particle of interest, P_i, are denoted by $\overrightarrow{r_{ij}}$.

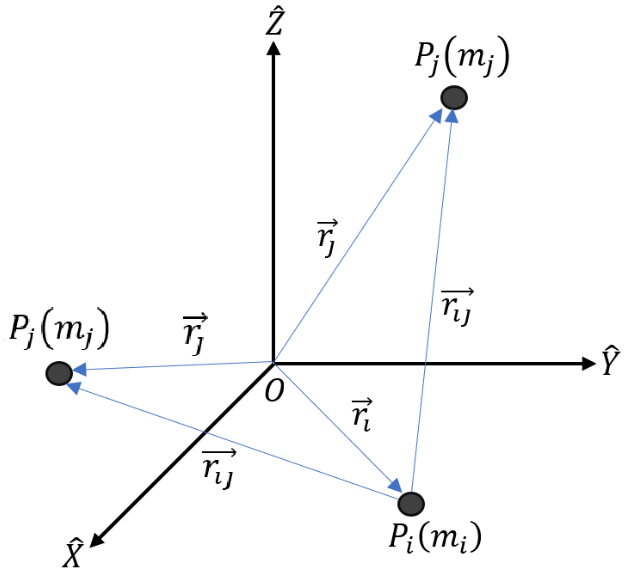


Figure 2.1 N-Body Model Geometry in Inertial Frame

The equation of motion for the N-Body system can be derived by using Newton's Law of Universal Gravitation and Newton's Law of Motion. Starting with Newton's Law of Universal Gravitation, the sum of the forces exerted on the spacecraft, P_i, is shown in equation 2.1.

$$\vec{f_i} = -G \sum_{j=1, j\neq i}^{n} \frac{m_i m_j}{r_{ij}^3} \overrightarrow{r_{ij}} \tag{2.1}$$

Using Newton's Law of Motion and substituting in equation 2.1 and the vector property, $\overrightarrow{r_{ij}} = -\overrightarrow{r_{ji}}$, equation 2.2 is the resulting N-Body model.

$$m_i \ddot{\vec{r_i}} = \vec{f_i}$$

$$\ddot{\vec{r_i}} = G \sum_{j=1, j \neq i}^{n} \frac{m_j}{r_{ji}^3} \overrightarrow{r_{ji}} \tag{2.2}$$

When evaluating the N-Body model, every body, P_j, that is introduced creates six more unknowns (x, y and z positions and velocities). To reduce the number of unknowns, find the motion of a particle, P_i, relative to the motion of another particle, P_q, as shown in Figure 2.2.

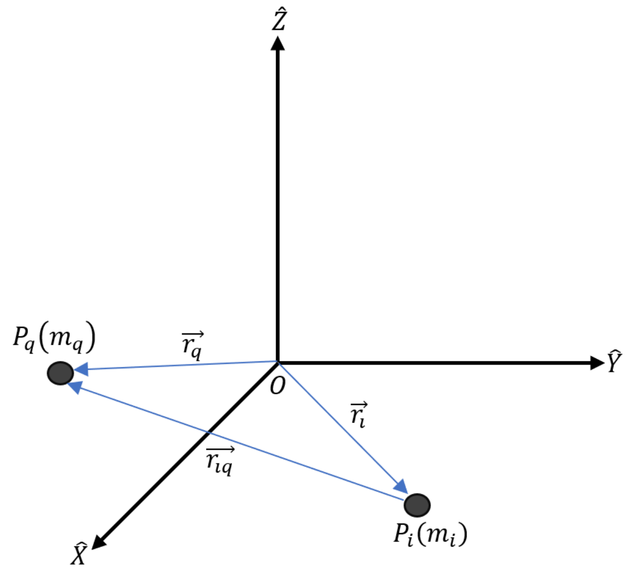


Figure 2.2 Motion of Particle *i* relative to Particle *q*

Equation 2.3 is found by using vector addition and taking the derivatives of equation 2.3 gives the velocity vector (equation 2.4) and the acceleration vector (equation 2.5).

$$\overrightarrow{r_{iq}} = \vec{r_q} - \vec{r_i} \tag{2.3}$$

$$\dot{\vec{r}}_{iq} = \dot{\vec{r}}_q - \dot{\vec{r}}_i \tag{2.4}$$

$$\ddot{\vec{r}}_{iq} = \ddot{\vec{r}}_q - \ddot{\vec{r}}_i \tag{2.5}$$

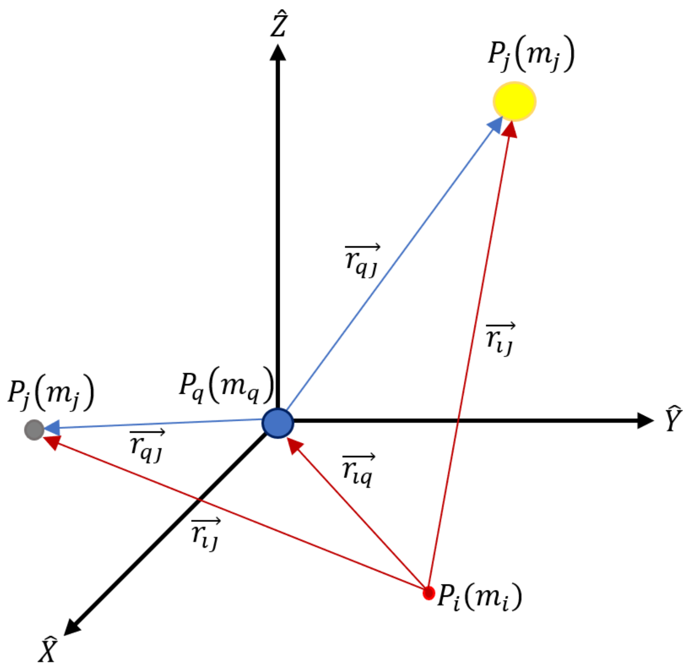
Finding the equations of motion for bodies P_i and P_q, substituting into equation 2.5 and using vector properties yields equation 2.6, the final form of the N-Body model.

$$\ddot{\vec{r}}_i = G \sum_{j=1, j\neq i, j\neq q}^{n} \frac{m_j}{r_{ji}^3}\vec{r}_{ji} + G\frac{m_q}{r_{qi}^3}\vec{r}_{qi}$$

$$\ddot{\vec{r}}_q = G \sum_{j=1, \neq i, j\neq q}^{n} \frac{m_j}{r_{jq}^3}\vec{r}_{jq} + G\frac{m_i}{r_{iq}^3}\vec{r}_{iq}$$

$$\ddot{\vec{r}}_{iq} = G \sum_{j=1, j\neq i, j\neq q}^{n} m_j\left(\frac{\vec{r}_{ij}}{r_{ij}^3} - \frac{\vec{r}_{qj}}{r_{qj}^3}\right) - G\left(m_q + m_i\right)\frac{\vec{r}_{qi}}{r_{qi}^3} \tag{2.6}$$

Although the model is derived using an arbitrary inertial base point, it is easier to calculate the motion of the body of interest with respect to a central body. Figure 2.3 presents the N-Body model geometry where the central body, P_q, represents the Earth, the body of interest, P_i, represents the spacecraft, and all other bodies, P_j, represent additional celestial bodies, specifically the Sun and Moon. The position of the spacecraft relative to the Earth is denoted by $\vec{r}_{iq}$. The position of the spacecraft relative to the Moon and Sun is denoted by $\vec{r}_{ij}$. Positions of the Sun and Moon relative to the Earth are denoted by $\vec{r}_{qj}$.



$\hat{Z}$
$P_j(m_j)$
$\overrightarrow{r_{qj}}$
$\overrightarrow{r_{ij}}$
$P_j(m_j)$
$P_q(m_q)$
$\hat{Y}$
$\overrightarrow{r_{qj}}$
$\overrightarrow{r_{iq}}$
$\overrightarrow{r_{ij}}$
$P_i(m_i)$
$\hat{X}$

Figure 2.3 N-Body Model Geometry relative to Central Body, P_q

Further reducing the number of unknowns, $\overrightarrow{r_{ij}}$ is rewritten in terms of $\overrightarrow{r_{iq}}$ and $\overrightarrow{r_{qj}}$ and the resulting N-Body model is shown in equation 2.7.

$$\ddot{\overrightarrow{r_{iq}}} = G \sum_{j=1, j\neq i, j\neq q}^{n} m_j \left(\frac{\overrightarrow{r_{iq}} + \overrightarrow{r_{qj}}}{\left(r_{iq} + r_{qj}\right)^3} - \frac{\overrightarrow{r_{qj}}}{r_{qj}^3} \right) - G\left(m_q + m_i\right) \frac{\overrightarrow{r_{qi}}}{r_{qi}^3} \tag{2.7}$$

Equation 2.7 does not consider other external forces such as drag, thrust, or solar radiation pressure. For the scope of this project, only gravitation forces will be addressed for simplification. Equations 2.8 – 2.10 are the expanded Cartesian coordinate forms of the N-Body model.

$$\ddot{x}_{iq} = G\sum_{j=1,j\neq i,j\neq q}^{n} m_j\left(\frac{x_{iq}+x_{qj}}{\left(\sqrt{x_{iq}^2+y_{iq}^2+z_{iq}^2}+\sqrt{x_{qj}^2+y_{qj}^2+z_{qj}^2}\right)^3}-\frac{x_{qj}}{\left(\sqrt{x_{qj}^2+y_{qj}^2+z_{qj}^2}\right)^3}\right) - G(m_q+m_i)\frac{x_{iq}}{\left(\sqrt{x_{iq}^2+y_{iq}^2+z_{iq}^2}\right)^3} \tag{2.8}$$

$$\ddot{y}_{iq} = G\sum_{j=1,j\neq i,j\neq q}^{n} m_j\left(\frac{y_{iq}+y_{qj}}{\left(\sqrt{x_{iq}^2+y_{iq}^2+z_{iq}^2}+\sqrt{x_{qj}^2+y_{qj}^2+z_{qj}^2}\right)^3}-\frac{y_{qj}}{\left(\sqrt{x_{qj}^2+y_{qj}^2+z_{qj}^2}\right)^3}\right) - G(m_q+m_i)\frac{y_{iq}}{\left(\sqrt{x_{iq}^2+y_{iq}^2+z_{iq}^2}\right)^3} \tag{2.9}$$

$$\ddot{z}_{iq} = G\sum_{j=1,j\neq i,j\neq q}^{n} m_j\left(\frac{z_{iq}+z_{qj}}{\left(\sqrt{x_{iq}^2+y_{iq}^2+z_{iq}^2}+\sqrt{x_{qj}^2+y_{qj}^2+z_{qj}^2}\right)^3}-\frac{z_{qj}}{\left(\sqrt{x_{qj}^2+y_{qj}^2+z_{qj}^2}\right)^3}\right) - G(m_q+m_i)\frac{z_{iq}}{\left(\sqrt{x_{iq}^2+y_{iq}^2+z_{iq}^2}\right)^3} \tag{2.10}$$

3.0 METHODOLOGY

Building and testing the code to model trajectories of a spacecraft will be done in MATLAB. Reading in the ephemerides of the Earth, Moon and Sun for computations is integrated into MATLAB using NASA JPL's SPICE toolkit. The circular orbit and the LADEE mission trajectory are used as test cases for the numerical model. Finally, the results are compared to NASA's GMAT program.

3.1 MATLAB

MATLAB was chosen as the main tool for programming as it already has built-in math functions and visualization tools, thus saving on time on programming in traditional languages (e.g. C/C++). One of the built-in math functions of importance are the solvers for ordinary differential equations (ODE) – particularly ODE45. Section 1.2 of this paper describes the numerical method used for ODE45.

The ODE45 solver will be used to solve for the N-Body model. Additionally, a Fixed-Step Runge-Kutta (RK) function will be created to also evaluate the N-body model. Initial conditions are taken for a circular orbit and from the LADEE mission (section 3.3). Solutions will then be plotted to visualize and compare the subsequent spacecraft trajectories.

3.2 SPICE and HORIZONS

SPICE is an information system built by NASA JPL's Navigation and Ancillary Information Facility (NAIF) and contains data such as planet ephemerides or a spacecraft's orientation at a given time in space. Data from SPICE can be used to compute observation geometry (e.g. positions and velocities of spacecraft) to help with space mission design. SPICE data can be accessed through files known as kernels which a user can call into their program or application. [17]

HORIZONS, which is from NASA JPL's Solar System Dynamics team, is another way to access ephemerides for celestial objects. Unlike SPICE, the ephemerides can be accessed through a web-based interface and can output text files. Users can easily obtain planetary positions for a few specific times. However, for larger time frames and computations, it is ideal to use SPICE toolkits for an more efficient integration into programs. [18]

For the purposes of this paper, the SPICE toolkit, MICE, will be integrated into MATLAB so that the ephemeris positions of the n-bodies can be called in to compute a spacecraft's trajectory. Functions that will be used is listed in Appendix A.

Kernels as well as time frames that will be used are associated with the LADEE mission (discussed in Section 3.3). Specifically, the kernels used are [19]:

- *naif0010.tls* – kernel containing leap second data up to July 1, 2012.
- *de432s.bsp* – kernel for Planetary Ephmerides DE432.
- *pck00010.tpc* – kernel providing 2009 IAU constants.
- *ladee_r_13250_13279_pha_v01.bsp* – LADEE trajectory kernel for all Earth phasing orbits and latter part of launch. [20]
- *ladee_r_13278_13325_loa_v01.bsp* – LADEE trajectory kernel for lunar orbit acquisition. [21]

- *ladee_r_13325_14108_sci_v01.bsp* – LADEE trajectory kernel for science phase. [22]
- *ladee_r_14108_99001_imp_v01.bsp* – LADEE trajectory kernel for end-of-mission. [23]

An inertial reference frame, centered at the Earth's center, will be used. In SPICE, this frame is the J2000 frame, or the International Celestial Reference Frame (ICRF). SPICE considers the J2000 frame and the ICRF as the same frame because the difference between them is small. When SPICE data is referencing J2000 frame, it is referencing ICRF. The coordinate system lies on the Equatorial plane with the x-axis at the Earth Mean Equator, the y-axis at the Equinox of Reference epoch, and the z-axis normal to the mean equator of date at epoch.

3.3 Benchmark Data

To validate that the code is working as expected, a known solution for the 2BP will be simulated in both MATLAB and GMAT. The initial conditions given are for a geostationary circular orbit shown in Table 3.1.

Table 3.1 Initial Conditions for a Circular Orbit

Initial Condition	**Value**
Initial x position, x_0	42,164 km
Initial y position, y_0	0 km
Initial z position, z_0	0 km
Initial x velocity, $\dot{x}_0$	0 km/s
Initial y velocity, $\dot{y}_0$	3.0747 km/s
Initial z velocity, $\dot{z}_0$	0 km/s

After validation of the code, the next test case used will be the Lunar Atmosphere and Dust Environment Explorer (LADEE) mission. The purpose of the LADEE mission was to observe the lunar atmosphere at low altitudes. On September 7, 2013 03:27 UTC, LADEE was launched on a Minotaur-V launch vehicle and afterwards entered a phasing loop trajectory. There are three phasing loop orbits around the Earth before reaching its final trajectory around the Moon. The amount of time for the phasing loops to be completed is shown in Table 3.2. [24]

Table 3.2 Number of Days to Complete Phasing Loops [24]

Phase	**Days**
Insertion orbit - Earth	6.4
Second phasing loop - Earth	8.2
Third phasing loop - Earth	9.9
Transfer orbit – Earth to Moon	4.9

SPICE data for the LADEE mission is used as benchmark data. The motivation to use the LADEE mission as benchmark data came from wanting to simulate a mission to the moon as well as the availability of trajectory data from SPICE. Although there are maneuvers performed during the phasing loops, the scope of this project will focus on the first Earth phasing loop before the first maneuver. A list of maneuvers performed during the three phasing loops and lunar orbit insertion are shown in Table 3.3.

Table 3.3 LADEE Maneuvers [24]

Maneuver	**Date / Time (UTC)**
Apogee Maneuver 1B	11 Sep 2013 23:00
Perigee Maneuver 1	13 Sep 2013 16:36
Perigee Maneuver 2	21 Sep 2013 11:53
Trajectory Correction Maneuver 1	01 Oct 2013 22:00
Lunar Orbit Insertion 1	06 Oct 2013 10:57
Lunar Orbit Insertion 2	09 Oct 2013 8:16
Lunar Orbit Insertion 3	13 Oct 2013 02:57

Initial conditions were also taken from the SPICE data of LADEE trajectory at September 7, 2013 04:00 TDB (shown in Table 3.4). The LADEE trajectory appears in Figure 3.1.

Table 3.4 Initial Conditions for LADEE Trajectory

Initial Condition	**Value**
Initial x position, x_0	$1190\ km$
Initial y position, y_0	$8209.1\ km$
Initial z position, z_0	$-3341.6\ km$
Initial x velocity, $\dot{x}_0$	$-6.4\ km/s$
Initial y velocity, $\dot{y}_0$	$3.8\ km/s$
Initial z velocity, $\dot{z}_0$	$-5.6\ km/s$

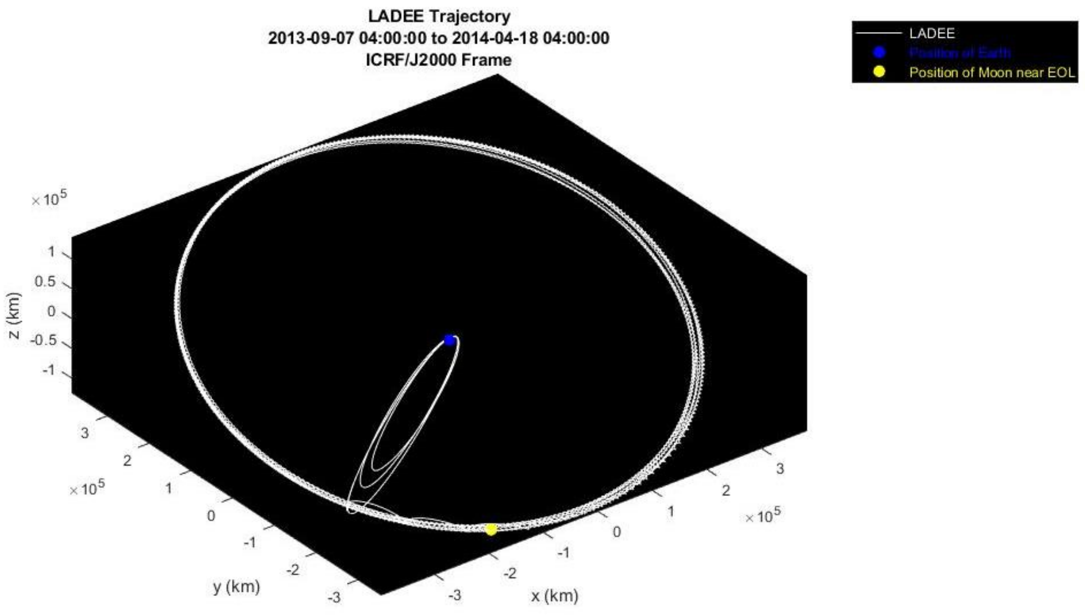


Figure 3.1 – LADEE Trajectory plotted in MATLAB

3.4 GMAT

Another tool used to simulate, analyze, or optimize space trajectories is GMAT – a program developed by a team of NASA Goddard Space Flight Center employees and private/public industry partners [25]. GMAT will be used to simulate the trajectories of a circular orbit as well as the LADEE mission before the first maneuver. The results of GMAT simulations will be compared to MATLAB solutions.

Information necessary to simulate an orbit in GMAT included the following:

- Epoch format: TDBGregorian
- Epoch: 07 Sep 2013 04:00:00.000
- Coordinate system: EarthICRF
- Initial conditions presented in Tables 3.1 and 3.4
- Numerical integrator: Runge-Kutta4
- Propagation time: Elapsed Time: 414,000 seconds
- Output wanted: Orbital view and a report of positions of spacecraft

GMAT has other features and tools to help simulate accurate orbits such as including other perturbation calculations (e.g. drag forces, solar radiation pressure, or point masses), differential correctors, or orbital maneuvers. For comparisons, perturbations not included in the N-Body model used in this paper will be excluded from GMAT simulations. Any differential correctors and orbital maneuvers will also be excluded from GMAT simulations. Although many of the perturbation calculations and differential correctors can be excluded from GMAT, at least one force model is required [26]. The one known force model included in the GMAT simulations is for spherical harmonic gravity – in particular, the Joint Gravity Model (JGM) 2 model.

4.0 PROCESS

4.1 MATLAB and SPICE

Necessary constants needed for computation are listed in Table 4.1.

Table 4.1 Constants for Computations

Constant	**Value**
Universal Gravitational Constant, G	$6.6743015 * 10^{-20}\ km^3/kg\ s^2$
Mass of the spacecraft, m_i	$383\ kg$
Mass of the Earth, m_q	$5.97219 * 10^{24}\ kg$
Mass of the Moon, m_1	$7.34767 * 10^{22}\ kg$
Mass of the Sun, m_2	$1.989 * 10^{30}\ kg$

Also needed are the ephemeris positions of the bodies used for this paper – the Sun, Moon and Earth. The ephemeris positions will be called into MATLAB by using SPICE functions. The time frame used to simulate the beginning of the first LADEE Earth phasing loops starts on September 7, 2013 04:00:00 and ends September 11, 2013 23:00:00.

The equations of motion (equations 2.8 – 2.10) for this investigation are second order ODEs. The ODEs must be rewritten into a series of first order ODEs so that the solvers can compute. An example of rewriting the equations are shown below where eventually $\dot{r_4}, \dot{r_5}$, and $\dot{r_6}$ are equal to equations 2.8, 2.9 and 2.10 respectively.

$$\begin{aligned} r_1 &= x & \dot{r_1} &= \dot{x} \\ r_2 &= y & \dot{r_2} &= \dot{y} \\ r_3 &= z & \dot{r_3} &= \dot{z} \\ r_4 &= \dot{x} & \dot{r_4} &= \ddot{x} \\ r_5 &= \dot{y} & \dot{r_5} &= \ddot{y} \\ r_6 &= \dot{z} & \dot{r_6} &= \ddot{z} \end{aligned}$$

4.1.1 Code Flowchart

Figure 4.1 is the flowchart of the code. Included is the process to load and unload kernels needed from SPICE, use numerical integrators or functions, plotting trajectories, and comparing solutions. The complete MATLAB code can be found in Appendices B, C and D.

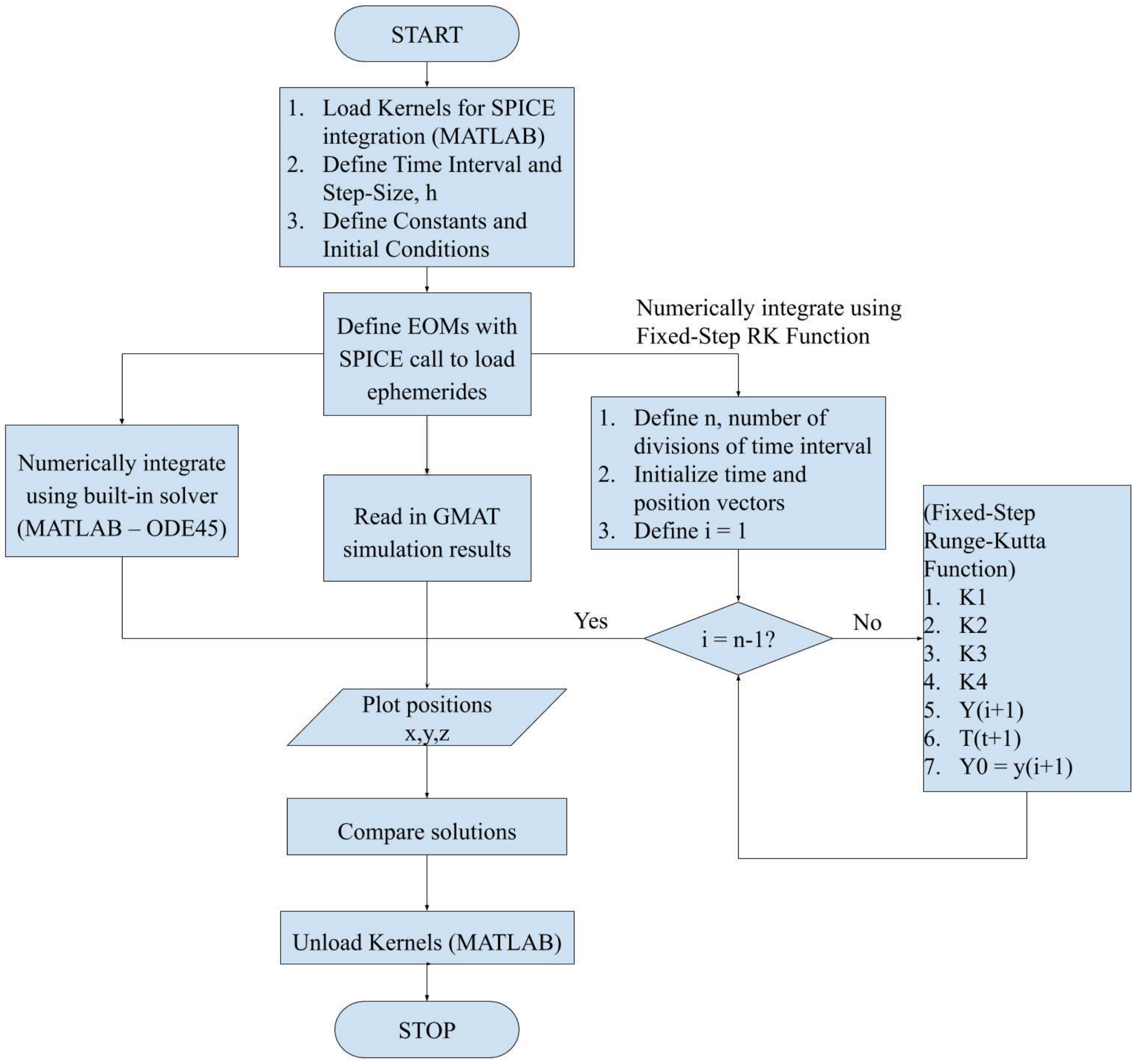

START
1. Load Kernels for SPICE integration (MATLAB)
2. Define Time Interval and Step-Size, h
3. Define Constants and Initial Conditions
Define EOMs with SPICE call to load ephemerides
Numerically integrate using Fixed-Step RK Function
Numerically integrate using built-in solver (MATLAB – ODE45)
Read in GMAT simulation results
1. Define n, number of divisions of time interval
2. Initialize time and position vectors
3. Define i = 1
(Fixed-Step Runge-Kutta Function)
1. K1
2. K2
3. K3
4. K4
5. Y(i+1)
6. T(t+1)
7. Y0 = y(i+1)
Yes
i = n-1?
No
Plot positions x,y,z
Compare solutions
Unload Kernels (MATLAB)
STOP

Figure 4.1 – Flowchart of Code

4.1.2 Tool Correction

During the first round of testing the code, one main issue arose where the output was an erroneous, straight line trajectory. The following list were possible errors tested for debugging the code:

1. ODE45 solver not able to propagate trajectory solution
2. Incorrect ephemerides called in by SPICE
3. Incorrect equations of motion

The first assumption was that the MATLAB ODE45 solver could not correctly propagate the solution. A Fixed-Step RK method was considered instead and required coding a new function. The Fixed-Step RK method is detailed in Section 1.2. The program ran completely with the Fixed-Step RK function, but the same erroneous solution appeared.

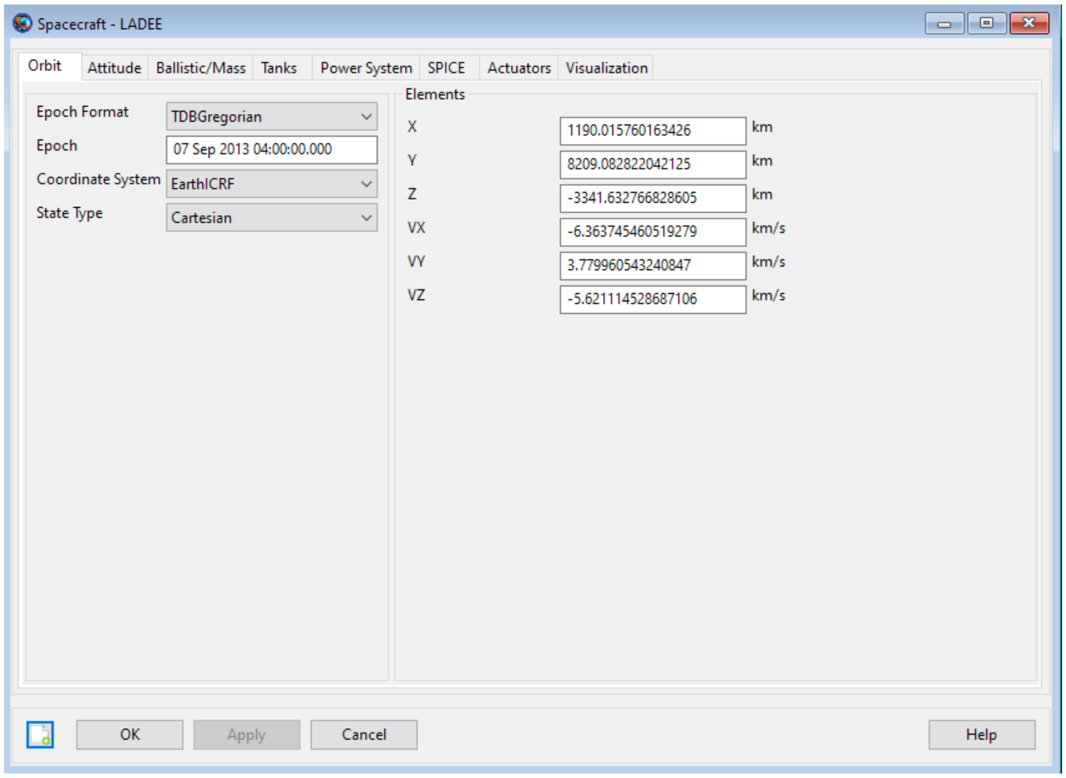
To make sure there were no errors with the newly created Fixed-Step RK function, the 2BP equation of motion was tested to solve for a circular orbit. Running the program resulted in an expected circular orbit outcome. The MATLAB ODE45 solver tested the same circular orbit case and produced the same outcome. This portion of debugging indicated that the Fixed-Step RK function and the ODE45 solver were computing as expected.

The ephemerides from SPICE was investigated next. Using the HORIZONS web-interface, text files of the ephemerides were exported and then read into the MATLAB code in place of SPICE function calls. Output from this test resulted in the same incorrect trajectory as previous test results, thus concluding the SPICE functions are not at fault.

The last test was to rewrite and check the equations of motion in the MATLAB function file. Comparing the derivation of the equations of motion to the coded file provided insight to human error. Equations of motion were corrected, and the code was tested again with the ODE45 solver and the Fixed-Step RK function. Both solvers generated similar and expected trajectories.

4.2 GMAT

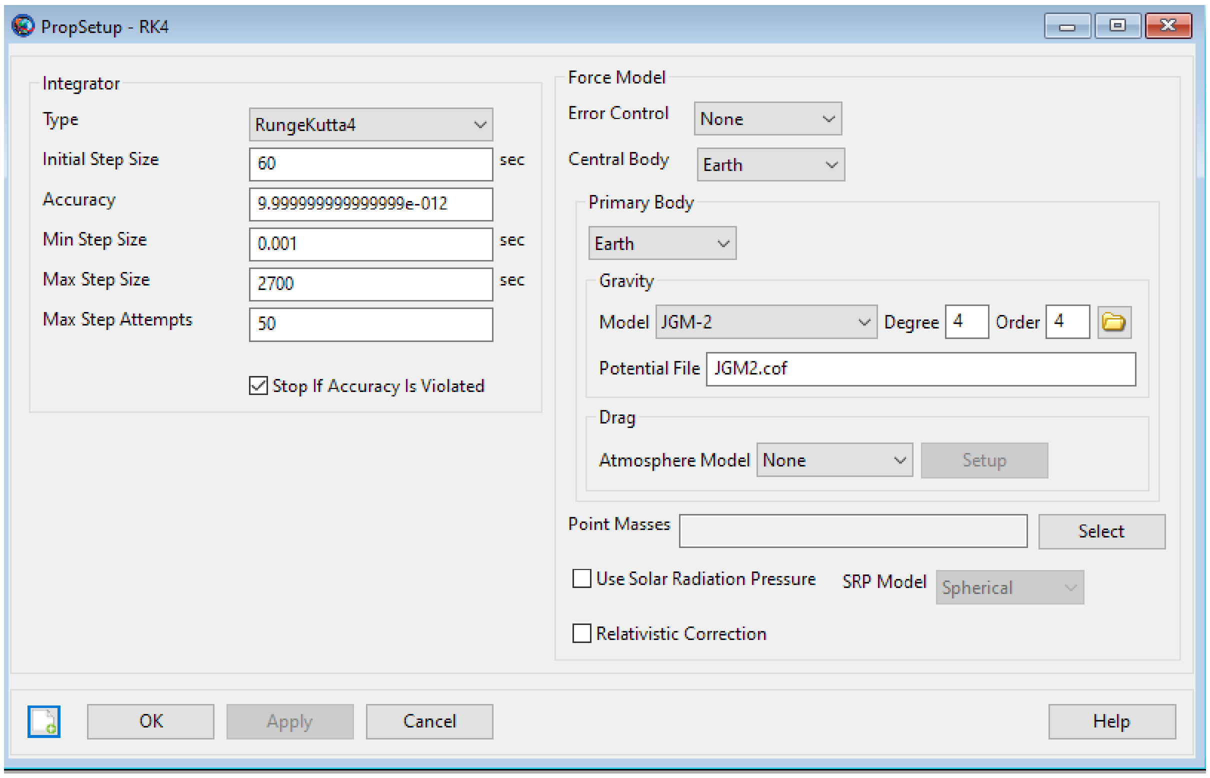
GMAT utilizes a Graphical User Interface (GUI) where users enter input into specified fields. Figure 4.2 to Figure 4.7 depict examples of GMAT windows necessary to obtain a simulation of the LADEE trajectory and output the results. After all information has been set, the GMAT simulation is ran and any outputs will be displayed in the Output tree of the GMAT GUI. Orbital views from the GMAT solution are presented in Section 5 of this paper.



Epoch Format	TDBGregorian
Epoch	07 Sep 2013 04:00:00.000
Coordinate System	EarthICRF
State Type	Cartesian

X	1190.015760163426	km
Y	8209.082822042125	km
Z	-3341.632766828605	km
VX	-6.363745460519279	km/s
VY	3.779960543240847	km/s
VZ	-5.621114528687106	km/s

Figure 4.2 – GMAT Spacecraft Window to Input Epoch and Initial Conditions [25]


PropSetup - RK4

Integrator

Type	RungeKutta4	
Initial Step Size	60	sec
Accuracy	9.999999999999999e-012	
Min Step Size	0.001	sec
Max Step Size	2700	sec
Max Step Attempts	50	

☑ Stop If Accuracy Is Violated

Force Model

Error Control: None

Central Body: Earth

Primary Body: Earth

Gravity: Model JGM-2, Degree 4, Order 4

Potential File: JGM2.cof

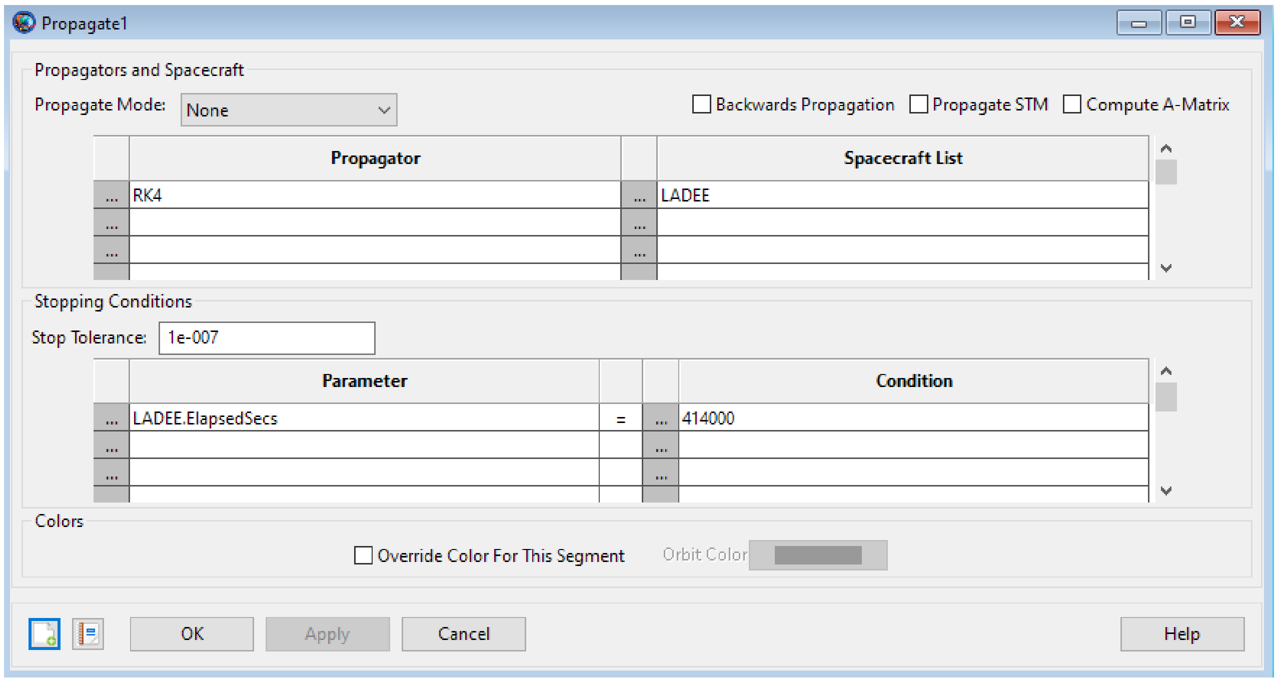
Drag: Atmosphere Model None

Point Masses:

☐ Use Solar Radiation Pressure SRP Model Spherical

☐ Relativistic Correction

Figure 4.3 – GMAT Propagator Window to Input Numerical Integrator and Force Models [25]


Propagate1

Propagators and Spacecraft

Propagate Mode: None

☐ Backwards Propagation ☐ Propagate STM ☐ Compute A-Matrix

Propagator	Spacecraft List
RK4	LADEE

Stopping Conditions

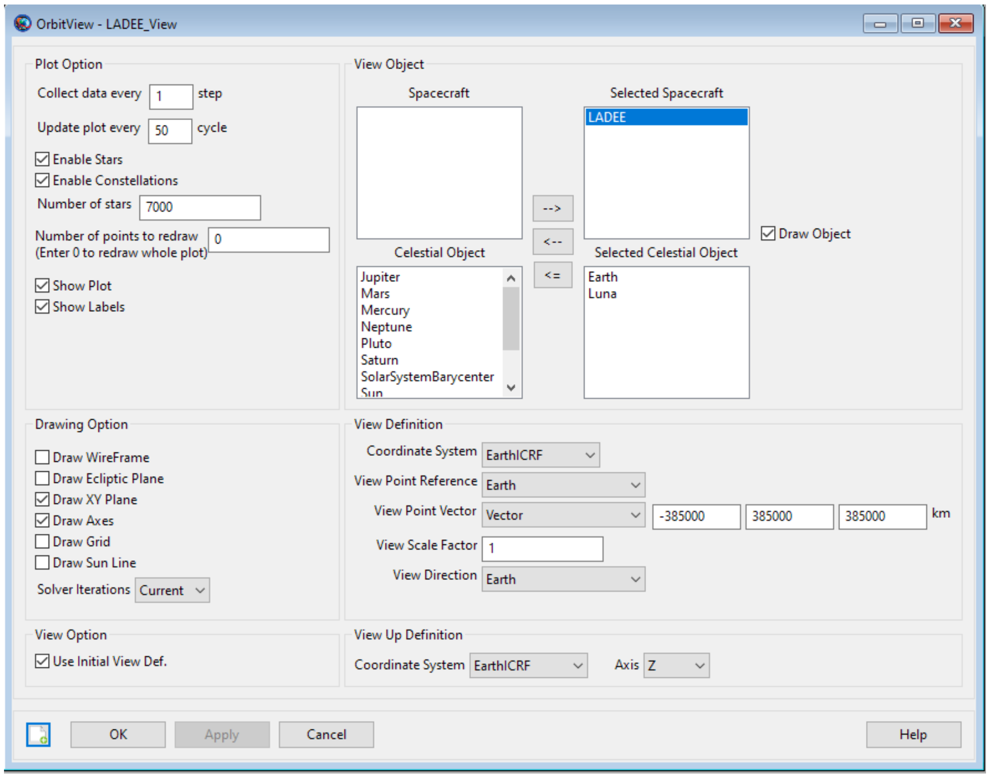
Stop Tolerance: 1e-007

Parameter		Condition
LADEE.ElapsedSecs	=	414000

Colors

☐ Override Color For This Segment Orbit Color

Figure 4.4 – GMAT Mission Window to Input Time Elapsed [25]


OrbitView - LADEE_View

Plot Option
Collect data every 1 step
Update plot every 50 cycle
☑ Enable Stars
☑ Enable Constellations
Number of stars 7000
Number of points to redraw (Enter 0 to redraw whole plot) 0
☑ Show Plot
☑ Show Labels

View Object
Spacecraft
Selected Spacecraft: LADEE
☑ Draw Object
Celestial Object: Jupiter, Mars, Mercury, Neptune, Pluto, Saturn, SolarSystemBarycenter, Sun
Selected Celestial Object: Earth, Luna

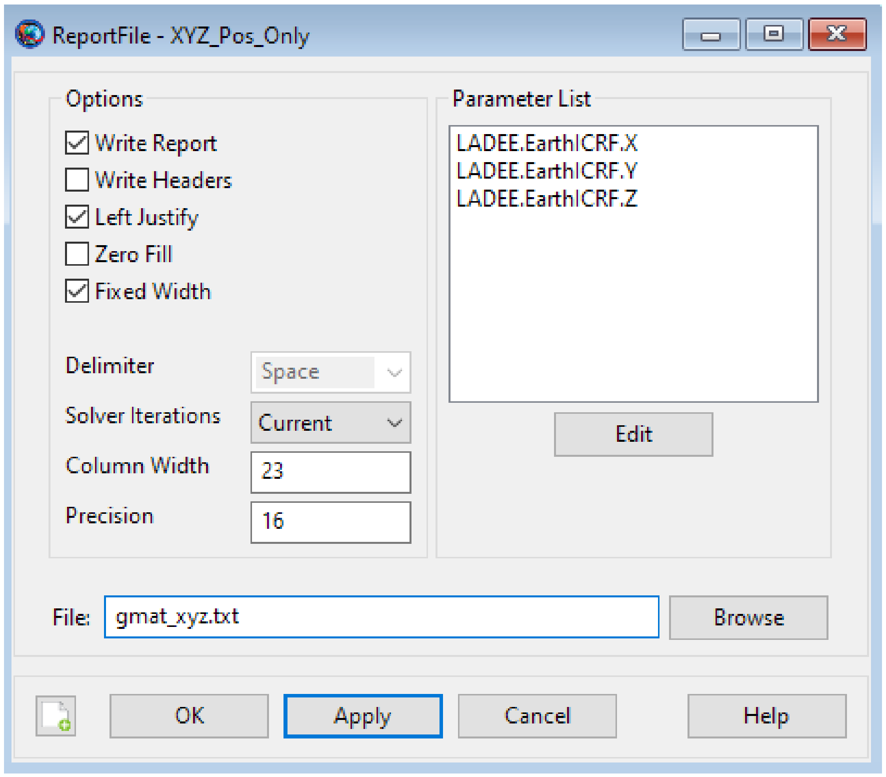
Drawing Option
☐ Draw WireFrame
☐ Draw Ecliptic Plane
☑ Draw XY Plane
☑ Draw Axes
☐ Draw Grid
☐ Draw Sun Line
Solver Iterations Current

View Definition
Coordinate System EarthICRF
View Point Reference Earth
View Point Vector Vector -385000 385000 385000 km
View Scale Factor 1
View Direction Earth

View Option
☑ Use Initial View Def.

View Up Definition
Coordinate System EarthICRF Axis Z

Figure 4.5 – GMAT Output Window to Generate Orbital View of Trajectory [25]


ReportFile - XYZ_Pos_Only

Options
☑ Write Report
☐ Write Headers
☑ Left Justify
☐ Zero Fill
☑ Fixed Width

Delimiter Space
Solver Iterations Current
Column Width 23
Precision 16

Parameter List
LADEE.EarthICRF.X
LADEE.EarthICRF.Y
LADEE.EarthICRF.Z

File: gmat_xyz.txt

Figure 4.6 – GMAT Output Window to Generate Report of X, Y, Z Positions of Spacecraft [25]

5.0 RESULTS

5.1 Circular Orbit Results

5.1.1 MATLAB Results for Circular Orbit Case

Solutions for the circular orbit using MATLAB ODE45 solver and the coded Fixed-Step RK function are shown in Figure 5.1.b and 5.1.c. A comparison of the two solvers is also shown in Figure 5.1.a. As shown in Figure 5.1.a, both solvers resulted in a circular orbit with negligible differences. Computations of the differences in position between the two solvers are summarized in Table 5.1.

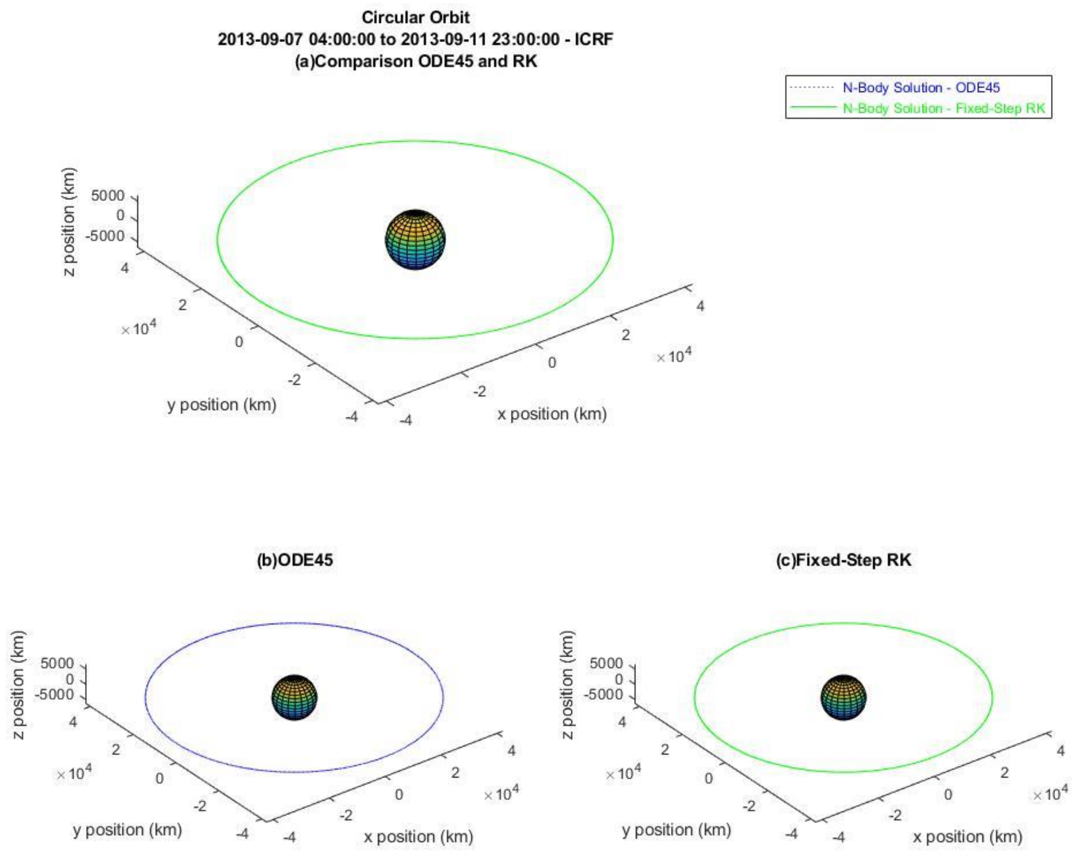


Figure 5.1 Circular Orbit – (a) Comparison of ODE45 and RK Solutions, (b) ODE45 Solution, (c) RK Solution

Table 5.1 Circular Orbit - Summary of Differences for ODE45 and RK Solutions

Maximum Difference (km)	Time (s)	Mean Difference (km)	Mean Percent Difference (%)
0.009863304	413940	0.00056408	7.6279E-06

5.1.2 GMAT Results for Circular Orbit Case

The GMAT simulation for the circular orbit is shown in Figure 5.2. The results of GMAT were read and plotted in MATLAB for comparison. The plotted GMAT solution in MATLAB is shown in Figure 5.3.

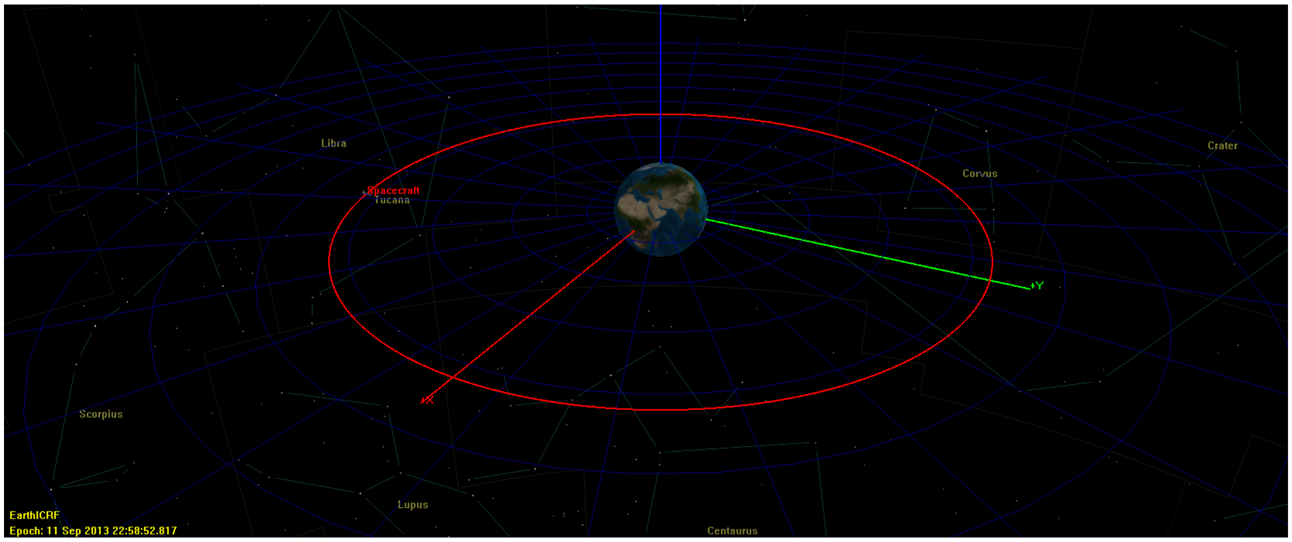


Figure 5.2 Circular Orbit - GMAT Solution

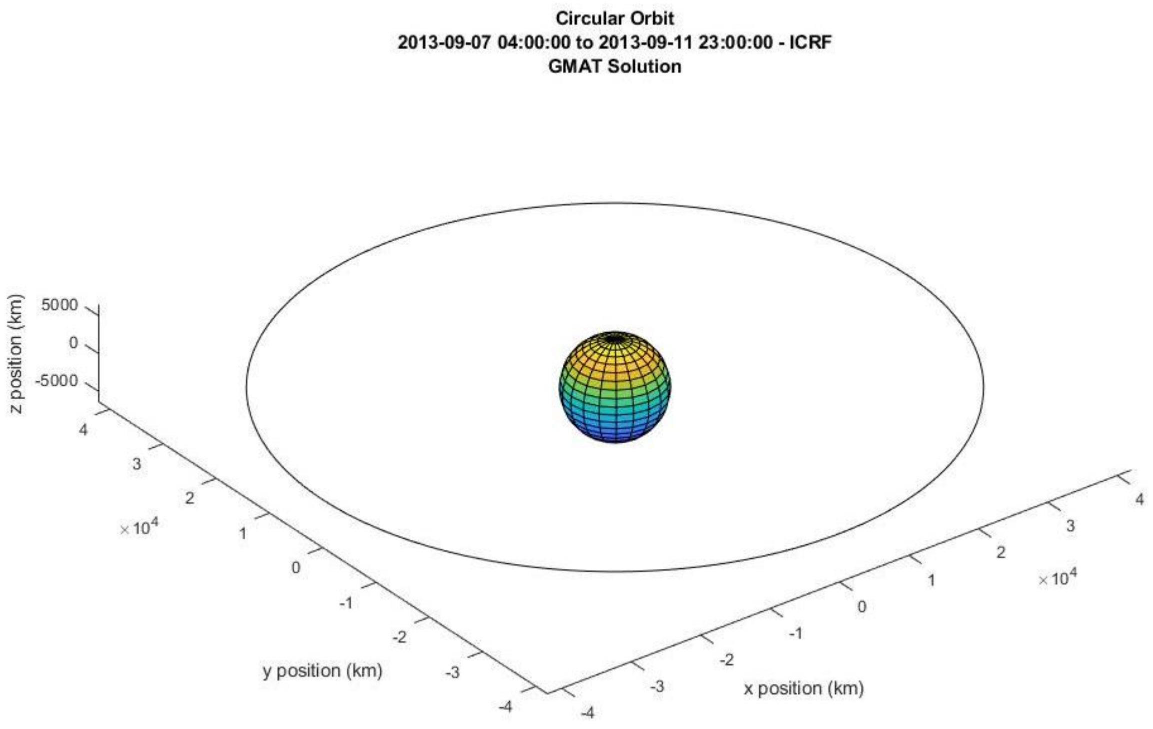


Figure 5.3 Circular Orbit - GMAT Solution Plotted in MATLAB

5.1.3 MATLAB and GMAT Comparisons for Circular Orbit Case

Comparisons of the Fixed-Step RK function and the ODE45 solver against GMAT is shown in Figure 5.4. Comparing the positions of the ODE45 solver and the Fixed-Step RK function against GMAT is summarized in Table 5.2. Since the ODE45 and Fixed-Step RK solutions are similar, the mean differences when comparing them both to GMAT are also similar. Once again, overlaying circular orbits are observed. However, there appears to be a slight variation seen in Figure 5.4 between GMAT and the MATLAB solvers. From calculated data, large differences occur at various time intervals. For example, differences of more than 10 km occurred between the time interval of 65,340 seconds to 95,160 seconds. The large differences between MATLAB and GMAT trajectories could be attributed to GMAT also computing perturbations due to the harmonic gravity field. Furthermore, the truncation errors are likely to accumulate over time which adds to deviations in the solution.

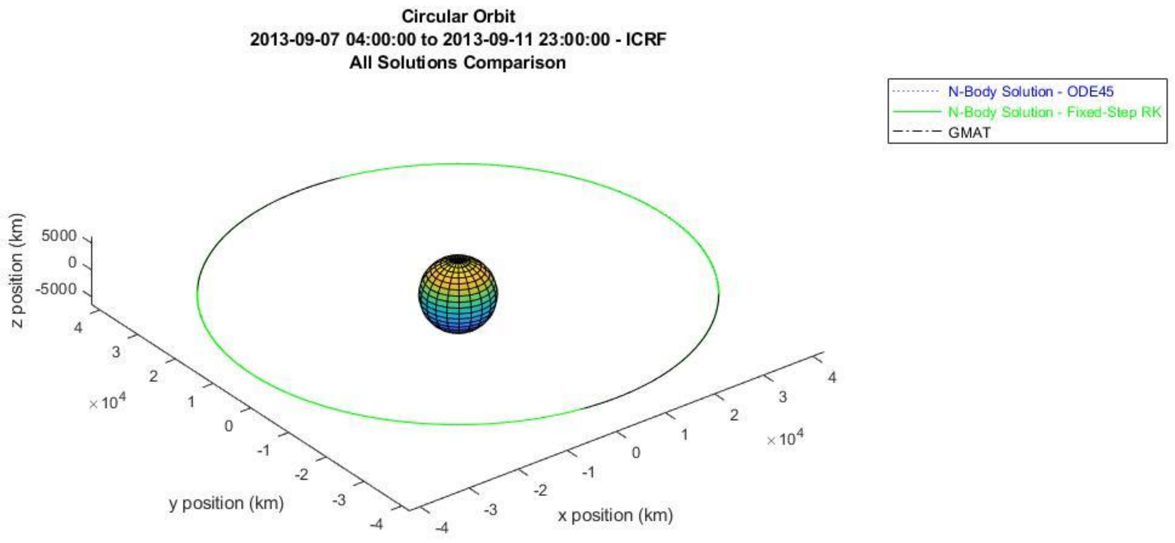


Figure 5.4 Circular Orbit – Comparison of All Solutions

Table 5.2 Circular Orbit - Summary of Differences for ODE45, RK and GMAT Solutions

	Maximum Difference (km)	Time (s)	Mean Difference (km)	Mean Percent Difference (%)
ODE45 vs GMAT	91.39419512	413940	-2.377	0.0735
RK vs GMAT	91.39615857	413940	-2.3776	0.0735

5.2 LADEE Trajectory Results

5.2.1 MATLAB Results for LADEE Trajectory Case

LADEE trajectory solutions are plotted and shown in Figure 5.5. A comparison of ODE45 and Fixed-Step RK appears in Figure 5.5.a. Like the circular orbit, the difference between the two solvers are negligible. The solvers are then compared to the LADEE trajectory taken from SPICE [11] and shown in Figure 5.6. The SPICE LADEE trajectory is considered to be the actual LADEE mission flight data. At the beginning, all trajectories seemingly overlap until the ODE45 and RK solutions start to depart significantly from the SPICE LADEE trajectory near apogee. Differences greater than 10 km start to appear at 128,520 seconds and continues to increase in size. The deviations once again could be contributed to the model not simulating gravity harmonics and accumulating truncation errors. Also, since the plotted LADEE trajectory is considered actual data, it is highly likely that other perturbations like point masses, drag forces, and solar radiation pressure are playing a significant role in its trajectory. The N-Body model in this paper does not include the mentioned perturbations. A summary of the calculated differences is found in Table 5.3.

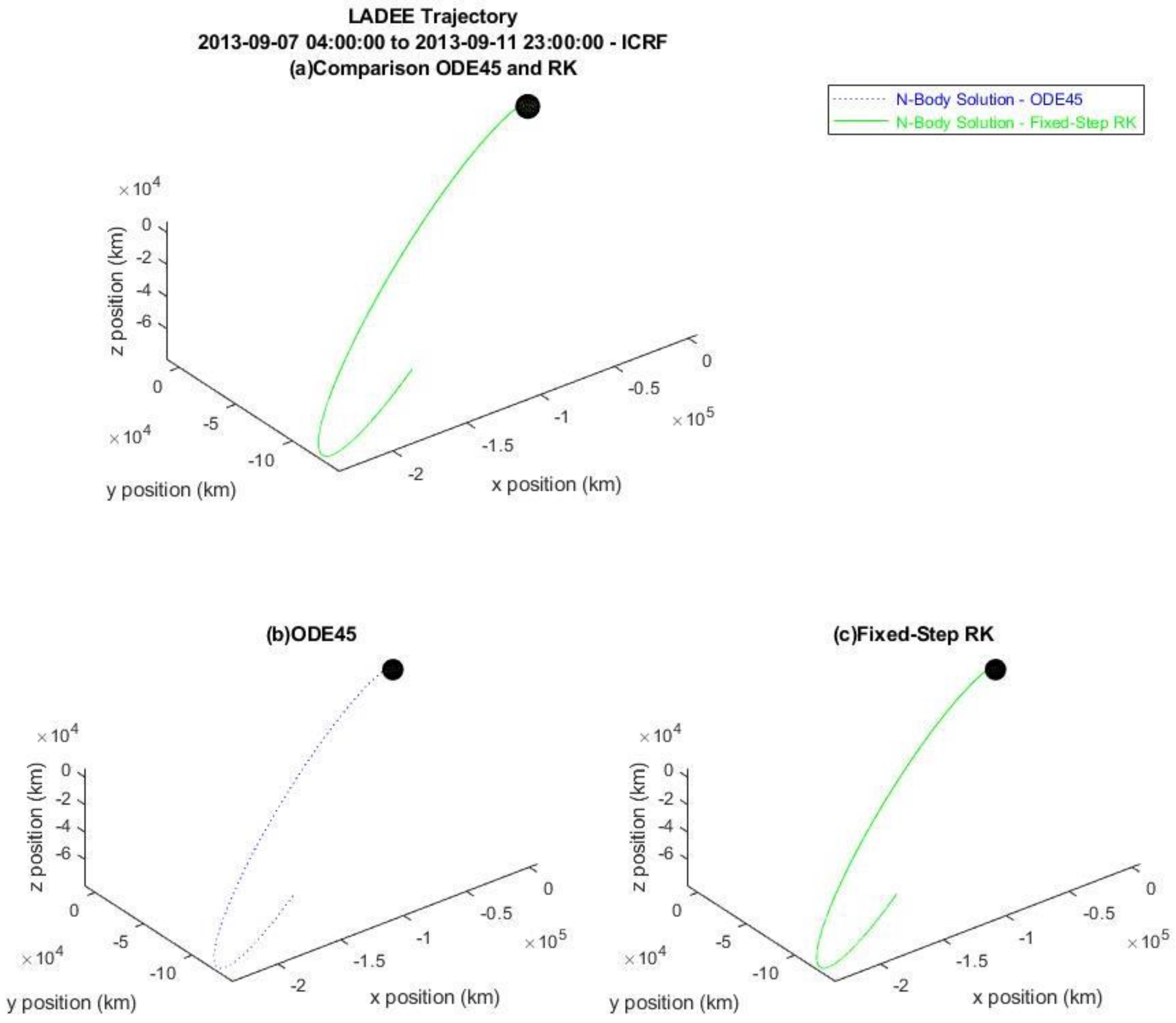


Figure 5.5 LADEE Trajectory – (a) Comparison of ODE45 and RK Solutions, (b) ODE45 Solution, (c) RK Solution

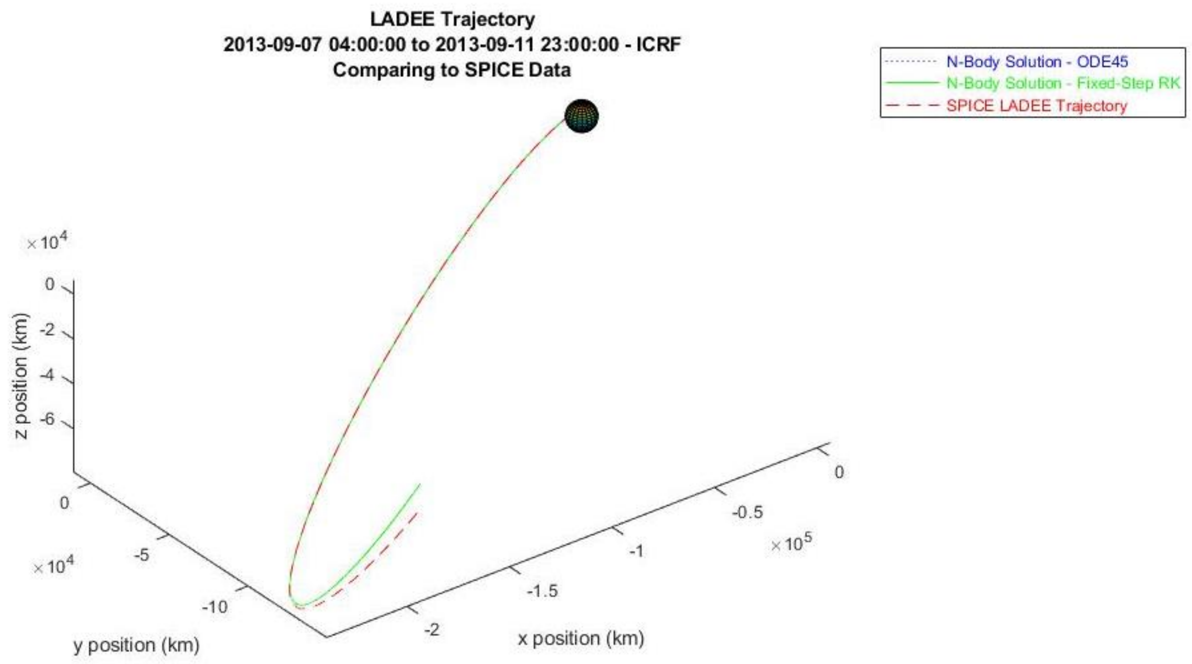


Figure 5.6 LADEE Trajectory – Comparison of ODE45, RK and LADEE

Table 5.3 LADEE Trajectory - Summary of Differences for ODE45 and RK Solutions against LADEE

	Maximum Difference (km)	Time (s)	Mean Difference (km)	Mean Percent Difference (%)
ODE45 vs RK	0.066539333	413940	-0.0381	0.000015153
ODE45 vs LADEE	15973.19946	413940	-3577	1.3752
RK vs LADEE	15973.266	413940	-3577.1	1.3752

5.2.2 GMAT Results for LADEE Trajectory Case

The GMAT simulation for the LADEE mission is shown in Figure 5.7. To compare the results of GMAT to ODE45 and Fixed-Step RK, GMAT data was read into MATLAB (shown in Figure 5.8).

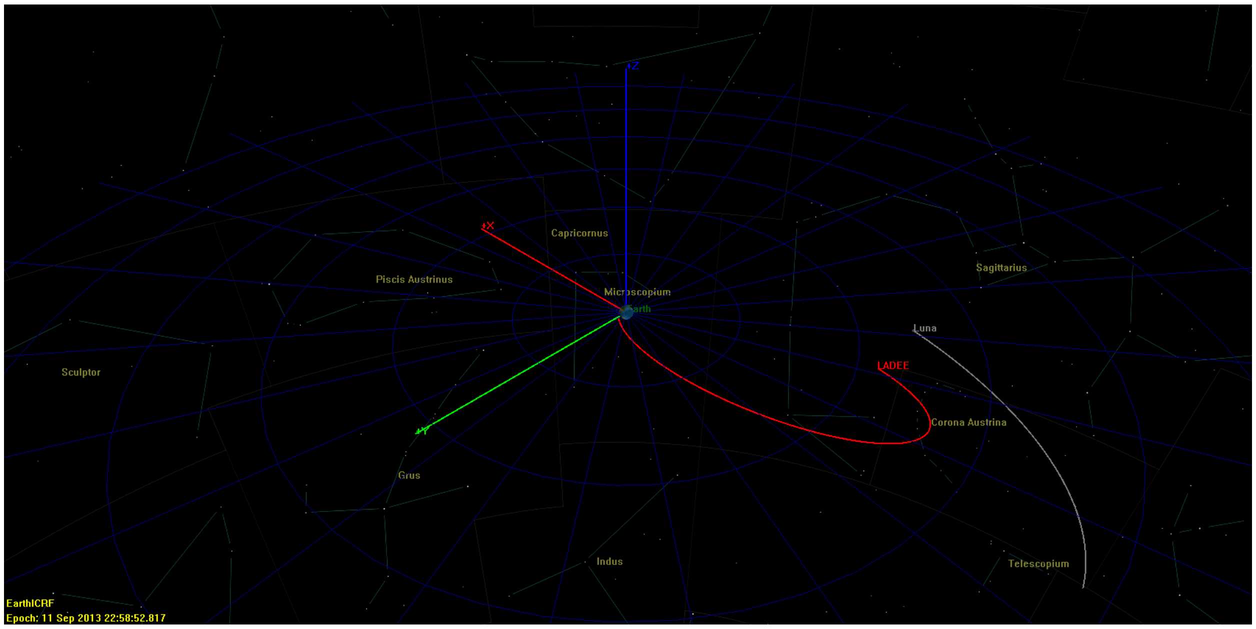


Figure 5.7 LADEE Trajectory – GMAT Solution

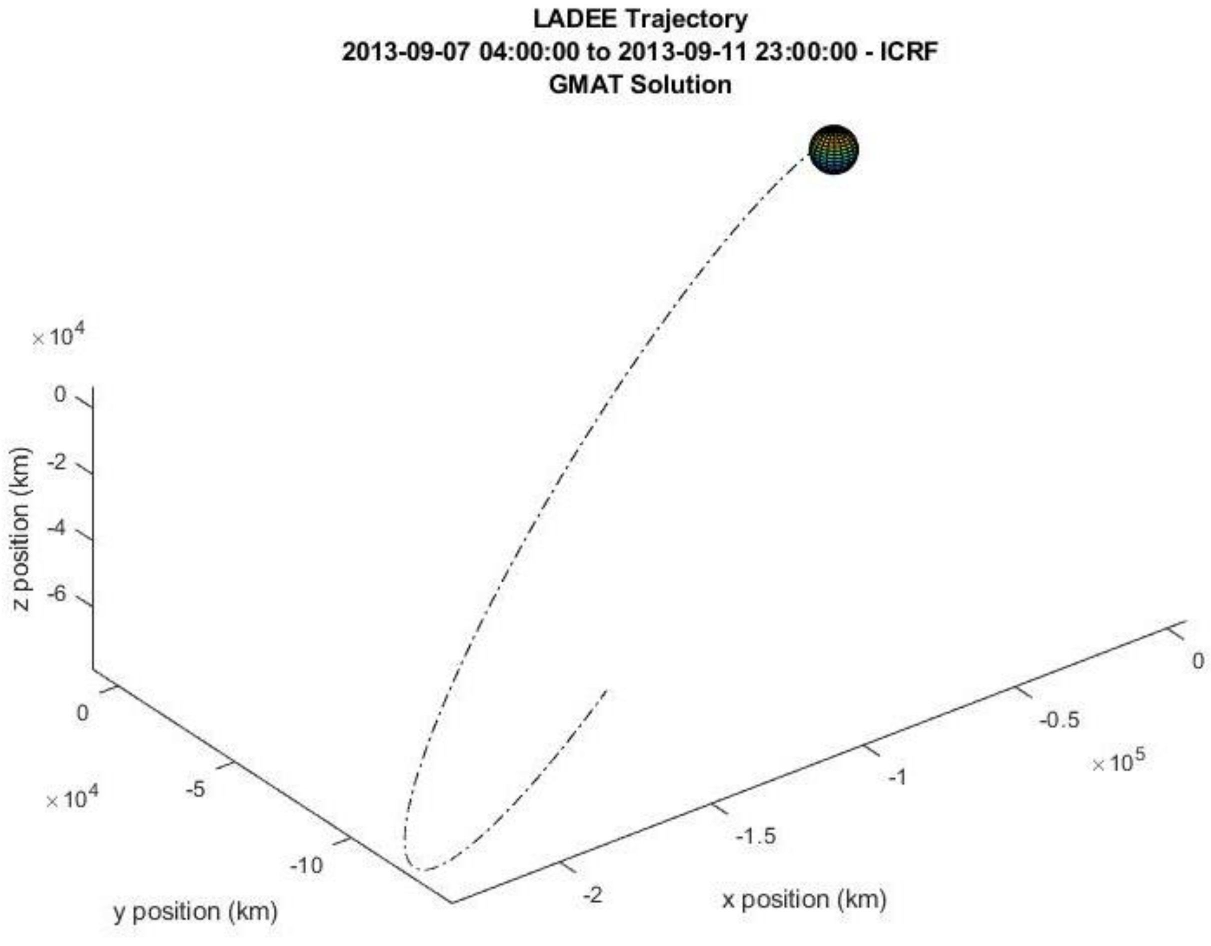


Figure 5.8 LADEE Trajectory – GMAT Solution Plotted in MATLAB

5.2.3 MATLAB and GMAT Comparisons for LADEE Trajectory Case

Comparisons of the Fixed-Step RK function and the ODE45 solver against GMAT is shown in Figure 5.9. All the solutions are also compared to the LADEE SPICE data in Figure 5.9. Compared to the SPICE data for LADEE, the GMAT solution is closer to the ODE45 and Fixed-Step RK solutions. Deviations between GMAT and the ODE45/Fixed-Step RK solver can once again be attributed to gravity harmonic calculations in GMAT and rounding errors. The difference between GMAT and SPICE LADEE trajectory could be attributed to the other force models that were not included in the GMAT computations. A summary of differences can be found in Table 5.4.

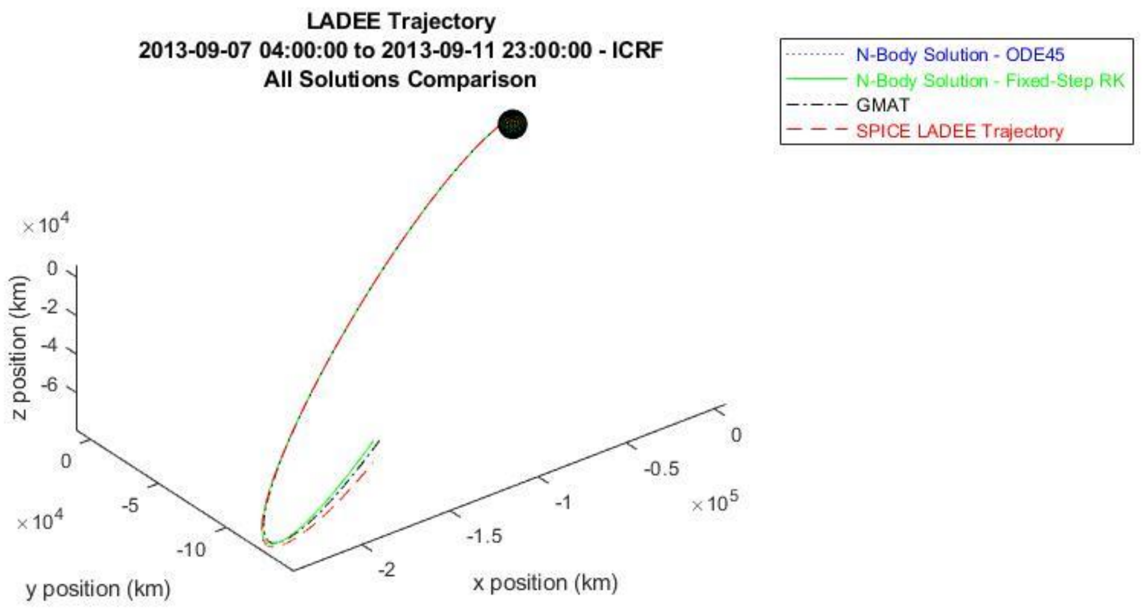


Figure 5.9 LADEE Trajectory – Comparison of All Solutions

Table 5.4 LADEE Trajectory - Summary of Differences for ODE45,RK and LADEE against GMAT

	Maximum Difference (km)	**Time (s)**	**Mean Difference (km)**	**Mean Percent Difference (%)**
ODE45 vs GMAT	1512.149932	413940	-638.7883	0.2523
RK vs GMAT	1512.083392	413940	-638.7502	0.2523
GMAT vs LADEE	17485.34939	413940	-4215.8	1.6004

6.0 SUMMARY

In this paper, a code to simulate spacecraft trajectories using a high-fidelity model proved successful. Simulations of both a circular orbit and the beginning trajectory of the LADEE mission was achieved. However, comparisons of the MATLAB solutions to GMAT showed that the N-Body model without other perturbations (drag forces, gravity harmonics, point masses) is subject to output less accurate trajectories. Additionally, accumulating rounding errors may also be a cause for deviating trajectories. Future work to modify the tool from this paper are considered in the next section.

6.1 Future Work

GMAT provided insight to other force models that should be considered to find more accurate trajectories. For example, in GMAT, adding the point masses of the Sun and the Moon into the simulation made a significant change in the GMAT solution. The resulting GMAT trajectory closely resembled the actual LADEE trajectory from SPICE data.

The paper also simulates a portion of the LADEE mission before the first maneuver. Thrust equations are not modeled so initial conditions after the maneuvers must be taken to continue propagating the solution. This causes unrealistic discontinuities in the orbit. Thus, future work for the MATLAB code can be to include computations for other perturbations as well as orbital maneuvers.

Other work considered is to test different numerical integrators other than the RK methods used in this investigation. Looking into minimizing or correcting truncation errors may also be of interest. Use of other programming languages other than MATLAB can also be valuable if the solvers show slowness in computation times.

References

1. Bate, R., Mueller, D., White,J.,*Fundamentals of Astrodynamics*, Dover Publications,1971.
2. Roy, A.E., *Orbital Motion*, Taylor & Francis Group, 2005.
3. Hsiao, R., Wang, T., Lee, C., Yang, P., Wei, H., Tsai, P., 'Trajectory of spacecraft with photonic laser propulsion in the two-body problem', *Acta Astronautica*, vol. 84, 2013, p. 215-226. https://doi.org/10.1016/j.actaastro.2012.11.006
4. Szebehely, V., *The Theory of Orbits*, Academic Press, 1967.
5. Pavlak, T. A., 'Trajectory design and orbit maintenance strategies in multi-body dynamical regimes', Purdue University, 2013. Retrieved from http://search.proquest.com.libaccess.sjlibrary.org/docview/1435646690?accountid=10361
6. Brick, J, and Air Force Institute Of Technology Wright-Patterson Afb Oh Wright-Patterson Afb United States. *Military Space Mission Design and Analysis in a Multi-Body Environment: An Investigation of High-Altitude Orbits as Alternative Transfer Paths, Parking Orbits for Reconstitution, and Unconventional Mission Orbits*, 2017.
7. Tantardini, M., Fantino, E., Ren, Y., Pergola, P., Gomez, G., Masdemont, J. J., 'Spacecraft trajectories to the L_3 point of the Sun–Earth three-body problem' *Celestial Mechanics and Dynamical Astronomy*, vol. 108, 2010, p. 215–232. https://doi-org.libaccess.sjlibrary.org/10.1007/s10569-010-9299-x
8. Bombardelli, C., Bernal Mencía, P., 'The circular restricted three-body problem in curvilinear coordinates' *Celestial Mechanics and Dynamical Astronomy*, vol. 130(11)**,** 2018, p. 1-16. https://doi-org.libaccess.sjlibrary.org/10.1007/s10569-018-9870-4
9. Capdevila, L., 'AE 243 Advanced Astrodynamics, CR3BP EOMs and Jacobi Integral', San Jose State University, 2019.
10. Wilmer, M, and Air Force Institute Of Technology Wright-Patterson Afb Oh Wright-Patterson Afb United States. *Military Applications of High-Altitude Satellite Orbits in a Multi-Body Dynamical Environment Using Numerical Methods and Dynamical Systems Theory*, 2016.
11. Ode45, MathWorks, URL: <https://www.mathworks.com/help/matlab/ref/ode45.html>, retrieved March 28, 2020.
12. Dormand, J.R., Prince, P.J., 'A family of embedded Runge-Kutta formulae', *Journal of Computational and Applied Mathematics*, vol. 6, 1980, p. 19-26. https://doi.org/10.1016/0771-050X(80)90013-3
13. Fourth-Order Runge-Kutta, Eric Cheever – Swarthmore College, URL: <https://lpsa.swarthmore.edu/NumInt/NumIntFourth.html>, retrieved April 2020.
14. Martini, Michael C., and James R. Scott. "High Speed Solution of Spacecraft Trajectory Problems Using Taylor Series Integration - NASA/TM-2008-215439." *Astrodynamics Specialist Conference and Exhibit*, no. ulu, hi, 2008, pp. Astrodynamics Specialist Conference and Exhibit; 18–21 Aug. 2008; Honolulu, HI; United States.
15. Zhang, H., Li, Y. & Zhang, K., 'A novel method of periodic orbit computation in circular restricted three-body problem' *Sci. China Technol. Sci.* vol. 54, 2011, p. 2197–2203. https://doi-org.libaccess.sjlibrary.org/10.1007/s11431-011-4441-x
16. Erwin, R, et al. *Spacecraft Trajectory Estimation Using a Sampled-Data Extended Kalman Filter with Range-Only Measurements*, 2005.

17. Acton, C.H.; "Ancillary Data Services of NASA's Navigation and Ancillary Information Facility;" Planetary and Space Science, Vol. 44, No. 1, pp. 65-70, 1996.
18. HORIZONS User Manual, NASA Jet Propulsion Laboratory, URL: <https://ssd.jpl.nasa.gov/?horizons_doc#intro>, retrieved March 2020.
19. LADEE Kernels, Planetary Data System, NAIF, URL: <https://naif.jpl.nasa.gov/pub/naif/LADEE/kernels/spk/>, retrieved March 2020.
20. Policastri, L., 'Lunar Atmosphere & Dust Environment Explorer Phasing SPK file', LADEE/AMES, 2014.
21. Policastri, L., 'Lunar Atmosphere & Dust Environment Explorer LOA SPK file', LADEE/AMES, 2014.
22. Policastri, L., 'Lunar Atmosphere & Dust Environment Explorer SCI SPK file', LADEE/AMES, 2014.
23. 'LADEE Post-Impact Location SPK', BVS/NAIF, 2014.
24. Kam, A., Plice, L., Galal, K., Hawkins, A., Policastri, L., Loucks, M., Nickel, C., Lebois, R., Sherman, R., Carrico, J., 'LADEE FLIGHT DYNAMICS: OVERVIEW OF MISSION DESIGN AND OPERATIONS', *2015 AAS-AIAA Space Flight Mechanics* Meeting, January 2015.
25. NASA Technology Transfer Program, General Mission Analysis Tool [GMAT V. R2016a], Goddard Space Flight Center.
26. Propagator, General Mission Analysis Tool Documentation, URL: <https://documentation.help/GMAT/Propagator.html#Propagator_ForceModel>, retrieved April 2020.

Appendix A. SPICE/MICE Functions

A list of SPICE/MICE functions used when building the code in MATLAB [17]:

- *cspice_furnsh* - function to load SPICE kernels into the MATLAB session.
- *cspice_unload/cspice_kclear* – functions that unload SPICE kernels (typically done at the end of the program) to avoid incorrect results if running multiple programs using SPICE data.
- *cspice_str2et* – function to convert time string inputs (e.g. 2013-09-07 03:28:00 TDB) into ephemeris time (ephemeris seconds past January 1, 2000 12:00:00 or J2000).
- *cspice_spkpos* – function to retrieve position vectors given a specific target, time, reference frame, and observer.
- *cspice_spkezr* – function to retrieve state vectors given a specific target, time, reference frame, and observer.

Appendix B. MATLAB Function File for N-Body Equations of Motion

```
%% N-Body Function
% Romalyn Mirador

%% Rewrite 2nd order to 1st order ODEs
% r(1) = x (x position of spacecraft)
% r(2) = y (y position of spacecraft)
% r(3) = z (z position of spacecraft)
% r(4) = xdot (x velocity of spacecraft)
% r(5) = ydot (y velocity of spacecraft)
% r(6) = zdot (z velocity of spacecraft)

% rdot(1) = xdot (x velocity of spacecraft)
% rdot(2) = ydot (y velocity of spacecraft)
% rdot(3) = zdot (z velocity of spacecraft)
% rdot(4) = xdotdot (x acceleration of spacecraft)
% rdot(5) = ydotdot (y acceleration of spacecraft)
% rdot(6) = zdotdot (z acceleration of spacecraft)

function rdot = nbody_func(t,r)
G = 6.6743015* 10^-20; % universal gravitational constant; units: km^3/kg s^2
m_sc_i = 383; % mi = mass of spacecraft LADEE in kg
m_earth_q = 5.97219*10^24; % mq = mass of Central Body(Earth) = 5.97219 x
10^24 kg
m_moon_1 = 7.34767*10^22; % mj1 = mass of Primary 1 (Moon) = 7.34767 x 10^22
kg
m_sun_2 = 1.989*10^30; % mj2 = mass of Primary 2 (Sun) = 1.989 x 10^30 kg

% Call for Ephmerides for Earth, Moon, Sun with SPICE
% Position for Moon relative to Earth: rqj1 and Position for Sun relative to
Earth: rqj2
% Define targets and observer for Moon and Sun
target1 = 'MOON';
target2 = 'SUN';
observer = 'EARTH';
frame = 'J2000'; %Reference Frame: ICRF/J2000; Coordinate systm: Ecliptic and
Mean Equinox of Reference Epoch

[pos_moon,ltime]=cspice_spkpos(target1,t,frame,'NONE',observer);
r_moon_x=pos_moon(1,:);
r_moon_y=pos_moon(2,:);
r_moon_z=pos_moon(3,:);

[pos_sun,ltime]=cspice_spkpos(target2,t,frame,'NONE',observer);
r_sun_x=pos_sun(1,:);
r_sun_y=pos_sun(2,:);
r_sun_z=pos_sun(3,:);

% Magnitudes
riq_sc2earth = sqrt((r(1)^2)+(r(2)^2)+(r(3)^2)); % magnitude of position of
spacecraft relative to Earth
rqj1_moon = sqrt((r_moon_x^2)+(r_moon_y^2)+(r_moon_z^2)); % magnitude of
position of Moon relative to Earth
```

```
rqj2_sun = sqrt((r_sun_x^2)+(r_sun_y^2)+(r_sun_z^2)); % magnitude of position
of Sun relative to Earth

% Equations of Motion
rdot(1) = r(4);
rdot(2) = r(5);
rdot(3) = r(6);
rdot(4) = (-
G*(m_sc_i+m_earth_q)*(r(1)/riq_sc2earth^3))+(G*m_moon_1*(((r(1)+r_moon_x)/((r
iq_sc2earth+rqj1_moon)^3))-
(r_moon_x/rqj1_moon^3)))+(G*m_sun_2*(((r(1)+r_sun_x)/((riq_sc2earth+rqj2_sun)
^3))-(r_sun_x/rqj2_sun^3)));
rdot(5) =  (-
G*(m_sc_i+m_earth_q)*(r(2)/riq_sc2earth^3))+(G*m_moon_1*(((r(2)+r_moon_y)/((r
iq_sc2earth+rqj1_moon)^3))-
(r_moon_y/rqj1_moon^3)))+(G*m_sun_2*(((r(2)+r_sun_y)/((riq_sc2earth+rqj2_sun)
^3))-(r_sun_y/rqj2_sun^3)));
rdot(6) = (-
G*(m_sc_i+m_earth_q)*(r(3)/riq_sc2earth^3))+(G*m_moon_1*(((r(3)+r_moon_z)/((r
iq_sc2earth+rqj1_moon)^3))-
(r_moon_z/rqj1_moon^3)))+(G*m_sun_2*(((r(3)+r_sun_z)/((riq_sc2earth+rqj2_sun)
^3))-(r_sun_z/rqj2_sun^3)));

rdot = rdot';

return
```

Appendix C. MATLAB Function File for Fixed-Step Runge-Kutta Method

```
%% Runge-Kutta 4 Fixed-Step Method
% Romalyn Mirador

function [t,r] = rk(h,t0,tf,r0)

n = (tf-t0)/h; % n = number of equal divisions in interval [t0,tf] where h is
step-size

%initialize vectors
t = zeros(n,1);
t(1) = t0;
r = zeros(n,6);
r(1,:) = r0';

    for i = 1:n-1
        rprime = nbody_func(h*i,r0);
        k1 = h*rprime;

        rprime = nbody_func(h*i+h/2,r0+k1/2);
        k2 = h*rprime;

        rprime = nbody_func(h*i+h/2,r0+k2/2);
        k3 = h*rprime;

        rprime = nbody_func(h*i+h,r0+k3);
        k4 = h*rprime;

        r(i+1,:) = r(i,:)+((k1/6)+(k2/3)+(k3/3)+(k4/6))';
        t(i+1) = t(i)+h;
        r0 = r(i+1,:)';
    end
end
```

Appendix D. MATLAB Script for N-Body Trajectories, Plots and Comparisons

```
%% MASTER'S PROJECT
% Numerically integrate the n-body model using fixed-step RK4, ode45,
% and given the ephemerides of Earth, Moon and Sun from JPL's SPICE.
% Compare solutions with GMAT
% Author: Romalyn Mirador

clear; close all; clc;

%% Load Kernels Paths
cspice_furnsh({'sckernel.tm',
'D:\MATLAB2019\kernels\ladee\spk\ladee_r_13250_13279_pha_v01.bsp'});

%% Set up/ Initial Conditions for LADEE
et = cspice_str2et({'2013-09-07 04:00:00 TDB', '2013-09-11 23:00:00 TDB'});
%convert string to ephemeris time in seconds past J2000 (J2000=2000 Jan 1
12:00:00 TDB)
t0 = 0; %initial ephemeris time in seconds past J2000 (J2000=2000 Jan 1
12:00:00 TDB)
tf = et(2)-et(1); %final ephemeris time in seconds past J2000 (J2000=2000 Jan
1 12:00:00 TDB)
ti = [t0,tf];%time interval

time_elapsed = et(2) - et(1);

step_size = 60; % seconds; 60 second intervals (h)
div = (tf-t0)/step_size;
times = (0:div-1)*step_size; % create an array of div times for a given
step_size starting at 0 seconds
times_in_et = times + et(1); % times in actual ephemeris time

% Initial Conditions for circular orbit
rad_Earth=6378.14; % Earth radius in km
mu = 3.98600*10^5; %Gravitational parameter; km^3/s^2

r_cir(1) = 35786+rad_Earth; % initial x position of spacecraft
r_cir(2) = 0; % initial y position of spacecraft
r_cir(3) = 0; % initial z position of spacecraft
r_cir(4) = 0; % initial x velocity of spacecraft
r_cir(5) = sqrt(mu/r_cir(1)); % initial y velocity of spacecraft
r_cir(6) = 0; % initial z velocity of spacecraft

r0_cir = [r_cir(1); r_cir(2); r_cir(3); r_cir(4); r_cir(5); r_cir(6)]; %
Column vector for initial conditions

% Define targets and observer for LADEE spacecraft comparison
target = 'LADEE';
observer = 'EARTH';
frame = 'J2000'; %Reference Frame: ICRF/J2000; Reference plane/Coordinate
system: Earth mean equator and equinox of reference epoch
```

```
[state,ltime]=cspice_spkezr(target,times_in_et,frame,'NONE',observer); %
finding the state using SPICE

% Positions and Velocities of LADEE from SPICE
x = state(1,:); % LADEE x position
y = state(2,:); % LADEE y position
z = state(3,:); % LADEE z position
xv = state(4,:); % LADEE x velocity
yv = state(5,:); % LADEE y velocity
zv = state(6,:); % LADEE z velocity

% Initial Conditions of LADEE from SPICE at time, t0
r(1) = x(1); % initial x position of spacecraft
r(2) = y(1); % initial y position of spacecraft
r(3) = z(1); % initial z position of spacecraft
r(4) = xv(1); % initial x velocity of spacecraft
r(5) = yv(1); % initial y velocity of spacecraft
r(6) = zv(1); % initial z velocity of spacecraft

r0 = [r(1); r(2); r(3); r(4); r(5); r(6)]; % Column vector for initial
conditions



%% Call in RK fixed-step (with nbody) function
% Circular Orbit
[t_rk_cir,r_rk_cir]=rk4_fixed(step_size,t0,tf,r0_cir);

x_rk_cir = r_rk_cir(:,1); % x position of spacecraft - RK Fixed-Step Solution
y_rk_cir = r_rk_cir(:,2); % y position of spacecraft - RK Fixed-Step Solution
z_rk_cir = r_rk_cir(:,3); % z position of spacecraft - RK Fixed-Step Solution
xv_rk_cir = r_rk_cir(:,4); % x velocity of spacecraft - RK Fixed-Step
Solution
yv_rk_cir = r_rk_cir(:,5); % y velocity of spacecraft - RK Fixed-Step
Solution
zv_rk_cir = r_rk_cir(:,6); % z velocity of spacecraft - RK Fixed-Step
Solution

% LADEE Trajectory
[t_rk,r_rk]=rk4_fixed(step_size,t0,tf,r0);

x_rk = r_rk(:,1); % x position of spacecraft - RK Fixed-Step Solution
y_rk = r_rk(:,2); % y position of spacecraft - RK Fixed-Step Solution
z_rk = r_rk(:,3); % z position of spacecraft - RK Fixed-Step Solution
xv_rk = r_rk(:,4); % x velocity of spacecraft - RK Fixed-Step Solution
yv_rk = r_rk(:,5); % y velocity of spacecraft - RK Fixed-Step Solution
zv_rk = r_rk(:,6); % z velocity of spacecraft - RK Fixed-Step Solution

%% Call nbody with ODE45
% Integration tolerances for ODE 45
ode_opt = odeset('AbsTol', 1e-12, 'RelTol', 1e-12);

% Time from GMAT
filename_time = 'gmat_time.txt';
A_time = importdata(filename_time);
```

```
t_gmat = A_time(:,1);

% Circular Orbit
[t_ode45_cir,r_ode45_cir] = ode45(@nbody_func, t_gmat, r0_cir, ode_opt); %
variable time-step

x_ode45_cir = r_ode45_cir(:,1); % calculated x position of spacecraft
y_ode45_cir = r_ode45_cir(:,2); % calculated y position of spacecraft
z_ode45_cir = r_ode45_cir(:,3); % calculated z position of spacecraft
xv_ode45_cir = r_ode45_cir(:,4); % calculated x velocity of spacecraft
yv_ode45_cir = r_ode45_cir(:,5); % calculated y velocity of spacecraft
zv_ode45_cir = r_ode45_cir(:,6); % calculated z velocity of spacecraft

% LADEE Trajectory
[t_ode45,r_ode45] = ode45(@nbody_func, t_gmat, r0, ode_opt); % variable time-
step

x_ode45 = r_ode45(:,1); % calculated x position of spacecraft
y_ode45 = r_ode45(:,2); % calculated y position of spacecraft
z_ode45 = r_ode45(:,3); % calculated z position of spacecraft
xv_ode45 = r_ode45(:,4); % calculated x velocity of spacecraft
yv_ode45 = r_ode45(:,5); % calculated y velocity of spacecraft
zv_ode45 = r_ode45(:,6); % calculated z velocity of spacecraft

%% Circular Orbit - Calculate mean difference and mean percent difference

mag_rk_cir = sqrt((x_rk_cir.^2) + (y_rk_cir.^2) + (z_rk_cir.^2)); % calculate
the position magnitude fixed-step rk4
mag_ode45_cir = sqrt((x_ode45_cir.^2) + (y_ode45_cir.^2) + (z_ode45_cir.^2));
% calculate the position magnitude ode45

% compare rk4 to ode45
mag_diff_rk_ode45_cir = mag_rk_cir - mag_ode45_cir;
mag_ave_rk_ode45_cir = (mag_rk_cir + mag_ode45_cir)/2;
mag_perdiff_rk_ode45_cir =
(abs((mag_diff_rk_ode45_cir./mag_ave_rk_ode45_cir)))*100;
mag_meandiff_rk_ode45_cir = mean(mag_diff_rk_ode45_cir)
mag_meanperdiff_rk_ode45_cir = mean(mag_perdiff_rk_ode45_cir)

%% Circular Orbit - Comparison to GMAT

% Read in gmat results
filename_cir = 'gmat_xyz_2bp.txt';
delimiterIn_cir = ' ';
A_cir = importdata(filename_cir, delimiterIn_cir);

x_pos_cir = A_cir(:,1);
y_pos_cir = A_cir(:,2);
z_pos_cir = A_cir(:,3);

mag_gmat_cir = sqrt((x_pos_cir.^2)+(y_pos_cir.^2)+(z_pos_cir.^2)); %calculate
position magnitude gmat
```

```
%compare rk4 to gmat
mag_diff_gmat_rk_cir = mag_gmat_cir - mag_rk_cir;
mag_ave_gmat_rk_cir = (mag_gmat_cir + mag_rk_cir)/2;
mag_perdiff_gmat_rk_cir =
(abs((mag_diff_gmat_rk_cir./mag_ave_gmat_rk_cir)))*100;
mag_meandiff_gmat_rk_cir = mean(mag_diff_gmat_rk_cir)
mag_meanperdiff_gmat_rk_cir = mean(mag_perdiff_gmat_rk_cir)

%compare ode45 to gmat
mag_diff_gmat_ode45_cir = mag_gmat_cir - mag_ode45_cir;
mag_ave_gmat_ode45_cir = (mag_gmat_cir + mag_ode45_cir)/2;
mag_perdiff_gmat_ode45_cir =
(abs((mag_diff_gmat_ode45_cir./mag_ave_gmat_ode45_cir)))*100;
mag_meandiff_gmat_ode45_cir = mean(mag_diff_gmat_ode45_cir)
mag_meanperdiff_gmat_ode45_cir = mean(mag_perdiff_gmat_ode45_cir)

%% LADEE - Calculate percent difference and percent error

mag_rk = sqrt((x_rk.^2) + (y_rk.^2) + (z_rk.^2)); % calculate the position
magnitude fixed-step rk4
mag_ode45 = sqrt((x_ode45.^2) + (y_ode45.^2) + (z_ode45.^2)); % calculate the
position magnitude ode45
mag_LADEE = sqrt((x.^2) + (y.^2) + (z.^2)); %calculate the position magnitude
LADEE

% compare rk4 to LADEE
mag_diff_rk_LADEE = mag_rk - mag_LADEE';
mag_ave_rk_LADEE = (mag_rk + mag_LADEE')/2;
mag_perdiff_rk_LADEE = (abs((mag_diff_rk_LADEE./mag_ave_rk_LADEE)))*100;
%mag_pererr_rk_LADEE = (abs(mag_diff_rk_LADEE)./abs(mag_LADEE'))*100;
mag_meandiff_rk_LADEE = mean(mag_diff_rk_LADEE)
mag_meanperdiff_rk_LADEE = mean(mag_perdiff_rk_LADEE)

% compare rk4 to ode45
mag_diff_rk_ode45 = mag_rk - mag_ode45;
mag_ave_rk_ode45 = (mag_rk + mag_ode45)/2;
mag_perdiff_rk_ode45 = (abs((mag_diff_rk_ode45./mag_ave_rk_ode45)))*100;
mag_meandiff_rk_ode45 = mean(mag_diff_rk_ode45)
mag_meanperdiff_rk_ode45 = mean(mag_perdiff_rk_ode45)

% compare ode45 to LADEE
mag_diff_ode45_LADEE = mag_ode45 - mag_LADEE';
mag_ave_ode45_LADEE = (mag_ode45 + mag_LADEE')/2;
mag_perdiff_ode45_LADEE =
(abs((mag_diff_ode45_LADEE./mag_ave_ode45_LADEE)))*100;
%mag_pererr_ode45_LADEE = (abs(mag_diff_ode45_LADEE)./abs(mag_LADEE'))*100;
mag_meandiff_ode45_LADEE = mean(mag_diff_ode45_LADEE)
mag_meanperdiff_ode45_LADEE = mean(mag_perdiff_ode45_LADEE)

%% LADEE - Comparison to GMAT

% Read in gmat results
filename = 'gmat_xyz.txt';
```

```
delimiterIn = ' ';
A = importdata(filename, delimiterIn);

x_pos = A(:,1);
y_pos = A(:,2);
z_pos = A(:,3);

mag_gmat = sqrt((x_pos.^2)+(y_pos.^2)+(z_pos.^2)); %calculate position
magnitude gmat

%compare rk4 to gmat
mag_diff_gmat_rk = mag_gmat - mag_rk;
mag_ave_gmat_rk = (mag_gmat + mag_rk)/2;
mag_perdiff_gmat_rk = (abs((mag_diff_gmat_rk./mag_ave_gmat_rk)))*100;
mag_meandiff_gmat_rk = mean(mag_diff_gmat_rk)
mag_meanperdiff_gmat_rk = mean(mag_perdiff_gmat_rk)

%compare ode45 to gmat
mag_diff_gmat_ode45 = mag_gmat - mag_ode45;
mag_ave_gmat_ode45 = (mag_gmat + mag_ode45)/2;
mag_perdiff_gmat_ode45 =
(abs((mag_diff_gmat_ode45./mag_ave_gmat_ode45)))*100;
mag_meandiff_gmat_ode45 = mean(mag_diff_gmat_ode45)
mag_meanperdiff_gmat_ode45 = mean(mag_perdiff_gmat_ode45)

%compare LADEE to gmat
mag_diff_gmat_LADEE = mag_gmat - mag_LADEE';
mag_ave_gmat_LADEE = (mag_gmat + mag_LADEE')/2;
mag_perdiff_gmat_LADEE =
(abs((mag_diff_gmat_LADEE./mag_ave_gmat_LADEE)))*100;
mag_meandiff_gmat_LADEE = mean(mag_diff_gmat_LADEE)
mag_meanperdiff_gmat_LADEE = mean(mag_perdiff_gmat_LADEE)
%% Plot trajectories

% Circular orbit plots
figure(1) %Subplots of ODE45 and RK and comparisons between the two
subplot(2,2,[1,2])
plot3(x_ode45_cir, y_ode45_cir, z_ode45_cir,'b:') % plot nbody ODE45
trajectory
hold on
plot3(x_rk_cir, y_rk_cir, z_rk_cir,'g') % plot nbody fixed rk trajectory
title ({'Circular Orbit';'2013-09-07 04:00:00 to 2013-09-11 23:00:00 - ICRF';
'(a)Comparison ODE45 and RK'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
legend('\color{blue} N-Body Solution - ODE45','\color{green} N-Body Solution
- Fixed-Step RK')
axis equal
```

```
subplot(2,2,3)
plot3(x_ode45_cir, y_ode45_cir, z_ode45_cir,'b:') % plot nbody ODE45
trajectory
title ({'(b)ODE45'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
axis equal

subplot(2,2,4)
plot3(x_rk_cir, y_rk_cir, z_rk_cir,'g') % plot nbody fixed rk trajectory
title ({'(c)Fixed-Step RK'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
axis equal


figure(2) %GMAT Replotted
plot3(x_pos_cir,y_pos_cir,z_pos_cir,'k-.') % plot GMAT results
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
title ({'Circular Orbit'; '2013-09-07 04:00:00 to 2013-09-11 23:00:00 -
ICRF'; 'GMAT Solution'})
axis equal

figure(3) %Comparisons to GMAT
plot3(x_ode45_cir, y_ode45_cir, z_ode45_cir,'b:') % plot nbody ODE45
trajectory
hold on
plot3(x_rk_cir, y_rk_cir, z_rk_cir,'g') % plot nbody fixed rk trajectory
hold on
plot3(x_pos_cir,y_pos_cir,z_pos_cir,'k-.') % plot GMAT results
title ({'Circular Orbit'; '2013-09-07 04:00:00 to 2013-09-11 23:00:00 -
ICRF'; 'All Solutions Comparison'})
hold on
[x_s,y_s,z_s] = sphere();
```

```
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
legend('\color{blue} N-Body Solution - ODE45','\color{green} N-Body Solution
- Fixed-Step RK','\color{black} GMAT')
axis equal

% LADEE trajectory plots
figure(4) %Subplots of ODE45 and RK and comparisons between the two
subplot(2,2,[1,2])
plot3(x_ode45, y_ode45, z_ode45,'b:') % plot nbody ODE45 trajectory
hold on
plot3(x_rk, y_rk, z_rk,'g') % plot nbody fixed rk trajectory
title ({'LADEE Trajectory'; '2013-09-07 04:00:00 to 2013-09-11 23:00:00 -
ICRF'; '(a)Comparison ODE45 and RK'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
legend('\color{blue} N-Body Solution - ODE45','\color{green} N-Body Solution
- Fixed-Step RK')
axis equal

subplot(2,2,3)
plot3(x_ode45, y_ode45, z_ode45,'b:') % plot nbody ODE45 trajectory
title ({'(b)ODE45'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
axis equal

subplot(2,2,4)
plot3(x_rk, y_rk, z_rk,'g') % plot nbody fixed rk trajectory
title ({'(c)Fixed-Step RK'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
axis equal
```

```
figure(5) %LADEE Comparison to ODE45 and RK
plot3(x_ode45, y_ode45, z_ode45,'b:') % plot nbody ODE45 trajectory
hold on
plot3(x_rk, y_rk, z_rk,'g') % plot nbody fixed rk trajectory
hold on
plot3(x,y,z,'r--') % plot LADEE trajectory
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
title ({'LADEE Trajectory'; '2013-09-07 04:00:00 to 2013-09-11 23:00:00 -
ICRF'; 'Comparing to SPICE Data'})
legend('\color{blue} N-Body Solution - ODE45','\color{green} N-Body Solution
- Fixed-Step RK','\color{red} SPICE LADEE Trajectory')
axis equal

figure(6) %GMAT Replotted
plot3(x_pos,y_pos,z_pos,'k-.') % plot GMAT results
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
title ({'LADEE Trajectory'; '2013-09-07 04:00:00 to 2013-09-11 23:00:00 -
ICRF'; 'GMAT Solution'})
axis equal

figure(7) %Comparisons to GMAT
plot3(x_ode45, y_ode45, z_ode45,'b:') % plot nbody ODE45 trajectory
hold on
plot3(x_rk, y_rk, z_rk,'g') % plot nbody fixed rk trajectory
hold on
plot3(x_pos,y_pos,z_pos,'k-.') % plot GMAT results
hold on
plot3(x,y,z,'r--') % plot LADEE trajectory
title ({'LADEE Trajectory'; '2013-09-07 04:00:00 to 2013-09-11 23:00:00 -
ICRF'; 'All Solutions Comparison'})
hold on
[x_s,y_s,z_s] = sphere();
surf( rad_Earth*x_s, rad_Earth*y_s, rad_Earth*z_s ) % sphere with radius
rad_Earth centred at (0,0,0)
hold on
xlabel 'x position (km)'
ylabel 'y position (km)'
zlabel 'z position (km)'
legend('\color{blue} N-Body Solution - ODE45','\color{green} N-Body Solution
- Fixed-Step RK','\color{black} GMAT','\color{red} SPICE LADEE Trajectory')
```

```
axis equal
%% Unload Kernels
cspice_kclear
```