INTERNATIONAL UNIVERSITY LIAISON INDONESIA (IULI)

BACHELOR'S THESIS

NUMERICAL ANALYSIS OF ORBIT PREDICTION ERRORS OF LAPAN'S SATELLITES

By

Prana Paramartha Rao
11201601009

Presented to the Faculty of Engineering
In Partial Fulfilment Of the Requirements for the Degree of

SARJANA TEKNIK
In
AVIATION ENGINEERING

FACULTY OF ENGINEERING

BSD City 15345
Indonesia
August 2020

APPROVAL PAGE

NUMERICAL ANALYSIS OF ORBIT PREDICTION ERRORS OF LAPAN'S SATELLITES

PRANA PARAMARTHA RAO
11201601009

Presented to the Faculty of Engineering
In Partial Fulfillment of the Requirements for the Degree of

SARJANA TEKNIK
In
AVIATION ENGINEERING

FACULTY OF ENGINEERING

Triwanto Simanjuntak, PhD	__________
Thesis Advisor	Date
Dr. Ir. Prianggada Indra Tanaya, M.M.E	__________
Dean of Faculty of Engineering	Date

EXAMINERS APPROVAL PAGE

Dr. Ir. Prianggada Indra Tanaya, MME.
Examiner 1 Date

Neno Ruseno, ST., M.Sc.
Examiner 2 Date

Dipl.-Ing. Ghani Harahap, M.Sc
Examiner 3 Date

STATEMENT BY THE AUTHOR

I hereby declare that this submission is my own work and to the best of my knowledge, it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at any educational institution, except where due acknowledgement is made in the thesis.

Prana Paramartha Rao
Student

Date

ABSTRACT

Numerical Analysis of Orbit Prediction Errors of LAPAN's Satellites

by

Prana Paramartha Rao

Triwanto Simanjuntak, PhD, Advisor

Indonesia's space agency organization named LAPAN had also launched several satellites to orbit the Earth. They are there for several reasons, from Earth monitoring to the measurement of the Earth's magnetic field. These missions that they have would be useless if there is no means of keeping track of the satellites. So there is an apparent need for determining the whereabouts of the satellites that orbit the Earth. Orbit determination system that is conventionally used is the Two-Line Elements (TLE) which only gives non-so frequent updates of the satellites' whereabout. Propagation methods are then used to compensate for this lack of updates, and to name some of the propagation methods there are the idealized two-body propagation, two-body + J_2 that considers the Earth oblateness, and the more complex SGP4 propagation method.

The focus of this research is to see how the different propagation methods would behave when applied to the satellites' that are owned by LAPAN, for which they are orbiting in the Low-Earth Orbit and have an almost perfectly circular orbit. Furthermore, the propagation results would then be compared to the observed values in the TLE historical data of each satellite. The evaluated values would then be analyzed using some statistical method of analyzing, linear regression and R^2 correlation, to see how the errors in the orbital elements and the state vectors behave over the propagation period. Based on how the propagation method generally implies, the errors should be in the acceptable region especially for the SGP4. What that means is that the error would be relatively small in respect to the historical data, and its behavior would increase at a small value over time.

Keyword: *Two-Line Elements, Satellites, Orbit Determination, Orbit Prediction Errors*

ACKNOWLEDGEMENTS

First and foremost the author would like to express his gratitude to god almighty *Allah SWT.* for the blessing and the opportunity given to the author to conduct this research. The author would also give gratitudes to the following beautiful souls who have believed in what the author does.

- Triwanto Simanjuntak, PhD, as my thesis advisor, supervisor, and also mentor for the past few years. much gratitude for the patience, the support, and the trust for the author.
- Dr. Ir. Tutuko Prajogo M.S.Mfg.E as the rector of the International University Liaison Indonesia, and Dr. Ir. Prianggada Indra Tanaya, M.M.E. as the Dean of Faculty of engineering.
- Both parents of the author, much gratitude for the support and love for all these years. The author would not be the way he is if not for them.
- Agha and Maheswara for the endless nights of discussion, anger, and laughter. Annisa who gave the author a lot of ideas, critics, and support. Ifnu and Kirana for the constant support. Heizkel and Gede for being the best role model for the author. Jihan, Andin, Indah, and Amel for the various ideas and support.
- My dearest friends who always there to hear what the author has to say with regards to his idea and bizarre way of thinking. Much gratitude and love for them all.
- Lastly, the readers, the author's creation might not be for everyone, but the author tip his hat for those who read his creation.

Contents

Approval Page i

Examiners Approval Page ii

Statement by The Author iii

Abstract iv

Acknowledgements v

Contents vi

List of Figures x

List of Tables xiii

1 Introduction 1
- 1.1 Background 1
- 1.2 Objective of Research 2
- 1.3 Significant of Study 3
- 1.4 Scope and Limitation 3

2 Literature Review 5
- 2.1 Two-Body Problem 5
  - 2.1.1 Kepler's Law 5
  - 2.1.2 Newton's Universal Law of Gravitation 7
  - 2.1.3 Two-Body Equation 8
- 2.2 Satellite State Representation 12
  - 2.2.1 Semimajor Axis, a 14

2.2.2 Eccentricity, e 14
2.2.3 Inclination, i 15
2.2.4 Right Ascension of the Ascending Node (RAAN), Ω 15
2.2.5 Argument of Periapsis (AoP), ω 16
2.2.6 True Anomaly, ν 16
2.3 Space Surveillance 17
2.4 Two-Line Elements 19
2.4.1 Format Description 20
2.4.2 Accuracy and Limitation 22
2.5 Satellite State Propagation 22
2.5.1 Orbit Perturbations 23
Gravity Field 23
Atmospheric Drag 25
Third-Body Perturbation 25
2.5.2 Satellite State Propagator 26
Two-Body 27
Simplified General Perturbation Models 27
2.5.3 Propagation Error 28
2.5.4 Regression Analysis 29
2.5.5 Fourier Series 31
2.5.6 R^2 Correlation 32

3 Research Methodology **33**
3.1 Research Outline 33
3.2 Computational Tools 35
3.2.1 Python Programming Language 35
3.2.2 Anaconda 36
3.2.3 Numpy 36
3.2.4 SciPy 36
3.2.5 Pandas 36
3.2.6 Matplotlib 37
3.2.7 cysgp4 37
3.2.8 Scikit-learn 37
3.2.9 symfit 38

3.3 Data Collection . . . 38
3.3.1 Lembaga Penerbangan dan Antariksa Nasional (LAPAN) . . 38
LAPAN-TUBSAT / LAPAN-A1 . . . 39
LAPAN-ORARI / LAPAN-A2 . . . 39
LAPAN-IPB / LAPAN-A3 . . . 40
3.3.2 TLE Data . . . 44
Obtaining TLE Data . . . 44
Extraction of Orbital Elements for Each Satellites . . . 46
3.4 Mathematical Model . . . 50
3.4.1 Conversion between Orbital Elements and State Vectors . . 50
Orbital Elements into State Vector . . . 51
State Vector into Orbital Element . . . 52
3.4.2 Satellite State Propagation . . . 54
Two-Body Propagation . . . 55
Two-Body Propagation + J_2 Propagation . . . 57
SGP4 Propagation . . . 59
3.5 Simulations Regimes . . . 61
3.5.1 Selecting the $State_0$ and t_0 . . . 62
3.5.2 Defining the Time Frame and Time Steps . . . 62
3.6 Error Analysis . . . 63
3.6.1 Controlled Values . . . 64
3.6.2 Propagation Error . . . 64
3.6.3 Regression Analysis . . . 66
3.6.4 Fourier Analysis . . . 68

4 Results and Discussions 71
4.1 Processed TLE Data . . . 71
4.1.1 Processed TLE Data result . . . 71
4.1.2 TLE Historical Data Graphs Analysis . . . 75
4.2 Propagation Results . . . 78
4.2.1 Two-Body Propagation Results . . . 78
4.2.2 Two-Body Propagation Graphs Analysis . . . 83
4.2.3 Two-Body + J_2 Propagation Results . . . 85
4.2.4 Two-Body + J_2 Propagation Graphs Analysis . . . 90

4.2.5 SGP4 Propagation Results 92
4.2.6 SGP4 Propagation Graphs Analysis 97
4.3 Propagation Error Analysis (Linear Regression) 99
4.3.1 Two-Body Propagation Error 99
4.3.2 Error Analysis 103
4.3.3 Two-Body + J_2 Propagation Error 107
4.3.4 Error Analysis 111
4.3.5 SGP4Propagation Error 115
4.3.6 Error Analysis 119
4.4 Propagation Error Analysis (Fourier Analysis) 123
4.4.1 Two-Body Propagation Error 124
4.4.2 Two-Body + J_2 Propagation Error 126
4.4.3 SGP4 Propagation Error 129

5 Summary, Conclusion, Recommendation 133

5.1 Summary 133
5.2 Conclusion 134
5.3 Recommendation 137

References 140

Appendices 143

Turnitin Report 194

Curriculum Vitae 209

List of Figures

2.1 Conic Sections (McClain, 2013) 6
2.2 Kepler's 2nd Law representation (*Kepler's Laws of Planetary Motion*, 2020) 7
2.3 The visualization of Newton's Universal Law of Gravitation (*Newton's Universal Law of Gravity*, n.d.) 8
2.4 Geometry of two bodies in an inertial frame of reference (McClain, 2013) 9
2.5 Earth Centered Inertial coordinate system (T. Kelso, n.d.-b) 12
2.6 Classical Orbital Elements (Tombasco, 2011) 13
2.7 Classical orbital elements representation(Tombasco, 2011) 17
2.8 Space surveillance network observation location (T. S. Kelso, n.d.) . 19
2.9 Two-Line Element Sets data description (*Definition of Two-line Element Set Coordinate System*, 2011) 20
2.10 Spherical Coordinate System (Curtis, 2014) 24
2.11 Representation of other body as the third-body perturbation (McClain, 2013) 26
2.12 Propagated initial state with its initial errors (Li, 2017) 29
2.13 Linear regression analysis example plot (*Regression Analysis*, n.d.) . 30
2.14 Square wave represented by different order of Fourier series 31
2.15 High value of R^2 and low value of R^2 (left to right) (Frost, n.d.) . . 32

3.1 LAPAN-A1 (*LAPAN-A1 / LAPAN-TUBSAT*, n.d.) 39
3.2 LAPAN-A2 (*LAPAN-A2 / LAPAN-ORARI*, n.d.) 40
3.3 LAPAN-A3 (*LAPAN-A3 / LAPAN-IPB*, n.d.) 40
3.4 TLE update frequency of LAPAN's satellites 46
3.5 TLE data extractions 48

3.6 Two-Body Propagation Flowchart 57
3.7 Two-Body + J2 Propagation Flowchart 59
3.8 Two-Body + J2 Propagation Flowchart 61
3.9 Time differences between $epoch_n$ to the first epoch 63
3.10 Positive and negative relationship between the variables (left to right) (*Linear Models*, 2020) 67

4.1 LAPAN-A1 extracted COEs 71
4.2 LAPAN-A2 extracted COEs 72
4.3 LAPAN-A3 extracted COEs 72
4.4 LAPAN-A1 state vector representation according TLE historical data 73
4.5 LAPAN-A2 state vector representation according TLE historical data 74
4.6 LAPAN-A3 state vector representation according to TLE historical data 75
4.7 LAPAN-A1 Two-Body propagated COEs 78
4.8 LAPAN-A2 Two-Body propagated COEs 79
4.9 LAPAN-A3 Two-Body propagated COEs 79
4.10 LAPAN-A1 state vector representation according to Two-Body propagation 80
4.11 LAPAN-A2 state vector representation according to Two-Body propagation 81
4.12 LAPAN-A3 state vector representation according to Two-Body propagation 82
4.13 LAPAN-A1 Two-Body + J_2 propagated COEs 85
4.14 LAPAN-A2 Two-Body + J_2 propagated COEs 86
4.15 LAPAN-A3 Two-Body + J_2 propagated COEs 86
4.16 LAPAN-A1 state vector representation according to Two-Body + J_2 propagation 87
4.17 LAPAN-A2 state vector representation according to Two-Body + J_2 propagation 88
4.18 LAPAN-A3 state vector representation according to + J_2 Two-Body propagation 89
4.19 LAPAN-A1 SGP4 propagated COEs 92
4.20 LAPAN-A2 SGP4 propagated COEs 93

4.21 LAPAN-A3 SGP4 propagated COEs 93
4.22 LAPAN-A1 state vector representation according to SGP4 94
4.23 LAPAN-A2 state vector representation according to SGP4 propagation 95
4.24 LAPAN-A3 state vector representation according to SGP4 propagation 96
4.25 LAPAN-A1 two-body propagation error 100
4.26 LAPAN-A2 two-body propagation error 101
4.27 LAPAN-A3 two-body propagation error 102
4.28 LAPAN-A1 two-body + J_2 propagation error 108
4.29 LAPAN-A2 two-body + J_2 propagation error 109
4.30 LAPAN-A3 two-body + J_2 propagation error 110
4.31 LAPAN-A1 SGP4 propagation error 116
4.32 LAPAN-A2 SGP4 propagation error 117
4.33 LAPAN-A3 SGP4 propagation error 118
4.34 Inclination error of LAPAN-A1 satellite using Two-Body propagation 124
4.35 LAPAN-A1 Inclination and RAAN error using Two-Body + J_2 propagation 126
4.36 LAPAN-A1 and LAPAN-A2 Inclination, RAAN, and Semiparameter error using Two-Body + J_2 propagation 129

List of Tables

2.1 Two-Line Elements set's line 1 format definition 21
2.2 Two-Line Elements set's line 2 format definition 21

3.1 LAPAN-A1 Specification 41
3.2 LAPAN-A2 Specification 42
3.3 LAPAN-A3 Satellite Specification. 43
3.4 Satellites data to be requested 44
3.5 Necessary elements to be extracted from TLE 47
3.6 Initial timestamps and states of each satellites 62
3.7 LAPAN-A2's orbital element according to TLE historical data . . . 65

4.1 Linear regression equation for two-body propagated orbital elements 103
4.2 Linear regression equation for two-body propagated state vector . . 103
4.3 LAPAN's satellites correlation value of two-body propagation error 104
4.4 Linear regression equation for two-body + J_2 propagated orbital elements 111
4.5 Linear regression equation for two-body + J_2 propagated state vector 111
4.6 LAPAN's satellites correlation value of two-body + J_2 propagation error 112
4.7 Linear regression equation for SGP4 propagated orbital elements . . 119
4.8 Linear regression equation for SGP4 propagated state vector 119
4.9 LAPAN's satellites correlation value of SGP4 propagation error . . 120
4.10 The seemingly periodic graphs 123
4.11 Coefficient, frequency, and R^2 values of the Inclination error Fourier analysis 125
4.12 Coefficient, frequency, and R^2 values of the Inclination error Fourier analysis 127

4.13 Coefficient, frequency, and R^2 values of the RAAN error Fourier analysis 128
4.14 Coefficient, frequency, and R^2 values of the Inclination error Fourier analysis 130
4.15 Coefficient, frequency, and R^2 values of the RAAN error Fourier analysis 131
4.16 Coefficient, frequency, and R^2 values of the Semiparameter error Fourier analysis 132

List of Abbreviations

AoP	Argument of Perigee
COE	Classical Orbital Element
ECI	Earth Centered Inertial
LAPAN	Lembaga Penerbangan dan Antariksa Nasional
NORAD	North American Aerospace Defence Command
RAAN	Right Ascension of the Ascending Node
SCC	Space Control Center
SGP4	Simplified General Propagation(4)
TLE	Two-Line Element

The Start of an Endless Journey

CHAPTER 1
INTRODUCTION

1.1 Background

Lembaga Penerbangan dan Antariksa Nasional or LAPAN is a Space Agency Organization, accountable to the President through the Minister responsible for government affairs in the field of research and technology. Historically, with the task given to LAPAN, they have done research on rockets, remote sensing, satellites, and space sciences for decades. Regarding satellite innovation, LAPAN currently operates and owns three satellites. The first satellite is the LAPAN-TUBSAT / LAPAN-A1, the second one is LAPAN-ORARI / LAPAN-A2, and the third one is LAPAN-IPB / LAPAN-A3; All of them are orbiting within the Low-Earth Orbit or LEO for short.

All of those satellites are up there with their own specific reasons, from Earth monitoring, communications, to the measurement of the Earth's magnetic field. These missions are useless if there are no means of keeping track of the satellites' whereabout, preferably, at all times. But, those satellites are also not equipped with any on board orbit determination system, such as GPS. In other words, LAPAN uses other means to keep their satellites in surveillance. The most common way of determining the satellite's orbit, which is also implemented by LAPAN, is to utilize Two Line Elements (TLE) data that is publicly available and is provided by the North American Aerospace Defense Command (NORAD).

The TLE set gives an encoded list of orbital elements of Earth-orbiting objects only at a given point of time. That means, rather than giving a continuous or real-time feed of the satellites' whereabout, the updates of the TLEs of satellites might vary from hours to days between updates. This does create one or two problems, that are mainly in regards of the accuracy, but the common way to compensate this is to predict the future position of the satellites from the latest update of the

TLE. But of course, The prediction only gives the value of position of the satellite to some degree of accuracy.

There are many kinds of prediction methods, or commonly phrased as propagation method, that can be used. Varying from the idealized, that are more of an educational purpose kind of prediction, to the complex ones that considers the complexity of the events that occurs in space. The ones that are used commonly for back of the envelop calculation is the two-body propagation method. With some additional consideration of, say, the Earth's zonal harmonic, the two-body propagation method becomes more complex. And the one that are most commonly and recommended to be used because the model considers quite a lot of disturbances is the Simplified General Perturbation (SGP4) propagation model.

Back to what was stated, those propagation models will will only be accurate to some degree of accuracy. For better or worse, the parameters of propagation method are continuously updated with the update of the TLE. One main reason is to keep the error to a minimum value due to the fact that the TLE itself gave off errors in the observation or even in the generation of the TLE itself. Although, the errors in the TLE is not the one that the author has interest in, rather, the errors in the propagation intrigues the author to analyze how different propagation methods would differ from the "actual" value that are given within the TLE. That said, with this study, the author intended to find and create a baseline with the error analysis in a hope for further improvement in regards to orbit analysis and minimization of error in propagation.

1.2 Objective of Research

The main goal of this research is henceforth to create a baseline of analysis on how different propagation methods, and specifically the error value between the propagation methods and the "actual" value of a historical data provided by NORAD. Along the way of doing so there are some side goals that can and should be achieved:

1. Utilizing the satellites' TLE historical data gathered from NORAD for preliminary orbit determination; and

2. Building numerical models of the different propagation models– Two-Body, Two-Body + J2, and SGP4, and also the method in using the models;

1.3 Significant of Study

1. This study may and presumably can be utilized and be implemented for a more extensive orbital analysis for LAPAN's satellites.
2. This study may also become a baseline for further analysis on different satellites with different types of orbits.
3. The orbit propagation error from this study may also be used to a rough plan on optimization on the propagation methods.

1.4 Scope and Limitation

1. The satellites that were used as the subject of the research were limited to the ones operated by LAPAN.
2. The research only used the publicly accessible data, namely Two-Line Elements, for which it is used to as the ”actual” value of the state of the satellites.
3. The TLE are not error-free per se. That is because it might encounter error in the observation or even the generation of the TLE itself. But those errors are neglected for they are out of the author's comprehension.
4. The raw data that are used are of different sample sizes for each satellite.
5. The research analyzed how different propagation models– Two-Body, Two-Body + J2, and SGP4, and the use of them behave over time.
6. The Two-Body and Two-Body + J2 propagation models and the method of using the models were set and within the control of the author.
7. The SGP4 model however, due to the complexity of the dynamical model, was "outsourced" with an already existing package called $cysgp4$.

8. The research analyzed how does the errors of propagation behave with respect to the actual values over time, all respective to each historical data.

9. The error analysis used were only the linear regression analysis and the Fourier analysis for a more straightforward and intuitive sense of behavior.

CHAPTER 2
LITERATURE REVIEW

2.1 Two-Body Problem

An understanding of the two-body problem is crucial in astrodynamics, for it is a useful "rough" starting point of calculations for more complex problems. In the two-body problem, it is assumed that the only force acting on either of the masses is that of gravitational attraction of which is applied by the other mass. What it means is that there are no other forces, such as the gravitational forces from other bodies nor other perturbation forces, other than the two gravitational force of the two masses taken into account in this problem. The Newton's Law of motion and Newton's law of gravitation, especially when combined with Kepler's Law, are some powerful starting points for orbital problems, especially the ones that will be talked about later.

2.1.1 Kepler's Law

Johannes Kepler, 1571-1630, was the one responsible for the formulation of the laws of planetary motion. The first two laws that he formulated was within the completion of his *Astronomia Nova* in 1609. In 1619, Kepler published his *Harmonices Mundi Libri V* where he buried hist third law within other theorems he came out with. The three laws, of which he wrote separately through out his research, states as the following:

1. The orbit of each planet is an ellipse with the Sun at one focus;
2. The line joining the planet to the Sun sweeps out equal areas in equal times; and

3. The square of the period of a planet is proportional to the cube of its semi-major axis.

The first law gives an information that planets, in fact all objects in space, travels in an elliptical path or conic section-like path, namely circles, ellipses, parabolas, and hyperbolas, with the Sun or other central body at one focus. Note that every conic section has two *foci*, where one of it reserves as the gravitational center of attraction called the *primary focus*. For circles the two *foci* coincides in the center of the orbital plane. From Fig. 2.1, it can be noticed that hyperbolas and parabolas are *open orbits* because they do not repeat their position, while circles and ellipses are *closed orbits* because they tend to retrace their path over time.

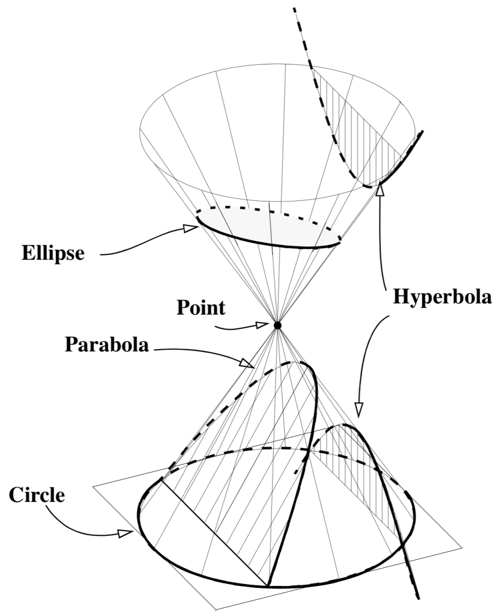


FIGURE 2.1: Conic Sections (McClain, 2013)

The second law describes the speed of which the orbiting body moves through its path. As it is getting closer to the main attracting body, the speed of the orbiting object would increase and would achieve its maximum orbiting speed at the closest point to the attracting body or *periapsis*, When the orbiting body is "catapulted" and moving further from the main attracting body, the orbiting speed of the object would decrease and would achieve its lowest speed at the furthest point of its orbit or *apoapsis*, as shown in Fig. 2.2. This phenomenon explains the equal areas at equal time of observation.

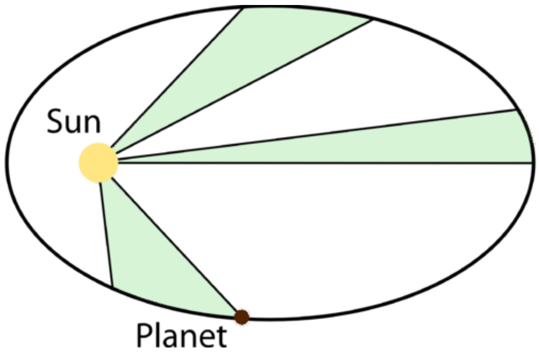


FIGURE 2.2: Kepler's 2nd Law representation (*Kepler's Laws of Planetary Motion*, 2020)

The third law imposed that the squared orbital period of a planet is proportional to the cube of its semimajor axis ($T^2 \propto a^3$). Now, getting a little bit of a sneak peek to the next subsection, the third law of Kepler can be proven with the help of the Newton's Universal Law of Gravitation, where the equation would lead to this,

$$T^2 = \frac{4\pi^2}{GM}a^3 \tag{2.1}$$

2.1.2 Newton's Universal Law of Gravitation

Kepler's Law was the initial statement for the expansion of the orbital motion. Newton then tied it all up together nicely with his Universal Law of Gravitation in his *Philosophiae Naturalis Principia Mathematica*, in which he stated that any masses would attract any other mass with a force that is acting on the line in which the both masses are intersecting. The magnitude of the force is proportional to the product of the two masses and inversely proportional to the square of their distance. The statement could be written as,

$$\vec{F_{21}} = -G\frac{M_1 M_2}{r^2}\hat{r} \tag{2.2}$$

where,

- $\vec{F_{21}}$, is the force acting on M_2 due to M_1;
- G, is the universal gravitational constant;

- M_1 and M_2, are the two attracting body, mass 1 and mass 2;
- r, is the distance between the two masses; and
- $\hat{r}$, is the unit vector of the distance between the two masses.

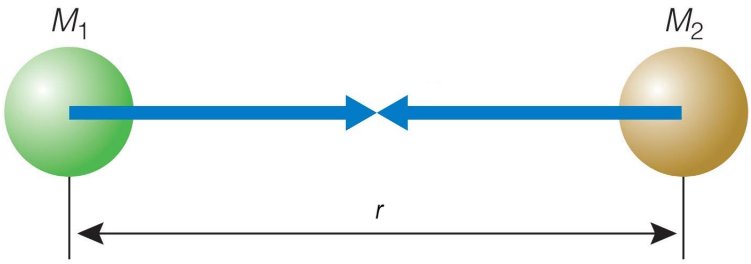


FIGURE 2.3: The visualization of Newton's Universal Law of Gravitation (*Newton's Universal Law of Gravity*, n.d.)

As shown in the Fig. 2.3 and also from the Eq. 2.2, it can be seen that in the system, both of the masses would experience the same amount of force. The only difference is that the sign of which the force vector is acting.

2.1.3 Two-Body Equation

First of all, an assumption of a coordinate system or reference frame is needed to be defined, for it will create an easier analysis of the two-body problem. That said, the inertial reference frame, a frame or a system that is fixed in inertial space, is needed for the derivation of the two-body equation of motion. Let the illustration in the Fig. 2.4 be the inertial frame of reference.

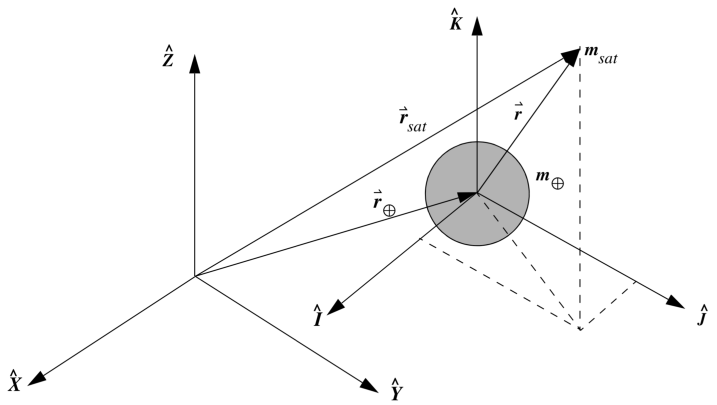


FIGURE 2.4: Geometry of two bodies in an inertial frame of reference (McClain, 2013)

Here are the details on the inertial system,

- $\hat{X}$, $\hat{Y}$, $\hat{Z}$, are the axis of the inertial coordinate system;
- $\hat{I}$, $\hat{J}$, $\hat{K}$, are the axis the geocentric coordinate system;
- $\vec{r}_{sat}$, is the position vector of the satellite in reference to the inertial reference frame;
- $\vec{r}_{\oplus}$, is the position vector of the Earth in reference to the inertial reference frame;
- $m_{\oplus}$,is the mass of the Earth; and
- m_{sat}, is the mass of the satellite.

Per what have been mentioned before, the Two-Body equation utilizes the Newton's Law of Motion and the Newton's Universal Law of Gravitation. Now, taking the geometry in reference to Fig. 2.4, we can derive the Two-Body equation. Let us identify the forces acting on the system, for that we took the Newton's Law of Universal Gravitation and write it as such,

$$\vec{F}_{\oplus\,sat} = -\frac{Gm_{\oplus}m_{sat}}{r^2}\frac{\vec{r}}{r} \tag{2.3}$$

where,

- $\vec{F}_{\oplus sat}$, is the force acting on the m_{sat} due to the $m_{\oplus}$;
- G, is the gravitational constant;
- $m_{\oplus}$, is the mass of the Earth;
- m_{sat}, is the mass of the satellite;
- $\vec{r}$, is the position vector of the m_{sat} relative to $m_{\oplus}$; and
- r, is the distance of the m_{sat} relative to $m_{\oplus}$.

The $\vec{r}$, as mentioned, is the position vector of the satellite relative to the Earth, for it is not to be confusing, here is how the vector from the Earth to the satellite looks like in mathematical form,

$$\vec{r}_{\oplus sat} = \vec{r}_{sat} - r_{\oplus} \tag{2.4}$$

The convenience of defining an inertial frame of reference is that it allows differentiation of the vectors without considering the derivatives of each axis of the coordinate system. That said, the acceleration of the satellite relative to the Earth can be found in a quite straightforward manner with the following equation,

$$\ddot{\vec{r}}_{\oplus sat} = \ddot{\vec{r}}_{sat} - \ddot{\vec{r}}_{\oplus} \tag{2.5}$$

and taking the Newton's Second Law of Motion, shown here,

$$\Sigma \vec{F} = \frac{d(m\vec{v})}{dt} = m\vec{a} \tag{2.6}$$

in combination with Newton's Universal Law of Gravitation, will permit us to directly write the *inertial force* on the satellite as the following,

$$\vec{F}_{g_{sat}} = m_{sat}\ddot{\vec{r}}_{sat} = -\frac{Gm_{\oplus}m_{sat}}{r^2}\frac{\vec{r}}{r} \tag{2.7}$$

and the force on the Earth as,

$$\vec{F}_{g_{\oplus}} = m_{\oplus}\ddot{\vec{r}}_{\oplus} = \frac{Gm_{\oplus}m_{sat}}{r^2}\frac{\vec{r}}{r} \tag{2.8}$$

Per Newton's Third Law of Motion, to every action there is always opposed an equal reaction, the magnitude forces acting on both satellite and the Earth are the same, though the difference is only the sign. In this case, the satellite receive the negative sign and the Earth receive the positive sign. Now, grouping these two equations and solve them for the relative acceleration, $\ddot{\vec{r}}$, gives,

$$\ddot{\vec{r}} = -\frac{Gm_{\oplus}}{r^2}\frac{\vec{r}}{r} - \frac{Gm_{sat}}{r^2}\frac{\vec{r}}{r} \tag{2.9}$$

or

$$\ddot{\vec{r}} = -\frac{G(m_{\oplus} + m_{sat})}{r^2}\frac{\vec{r}}{r} \tag{2.10}$$

Here is where the next assumption takes place. Assume that the mass of the Earth is much bigger, by many orders of magnitudes, in comparison to that of the satellite. As a result the sum of the two masses would just virtually be the mass of the Earth itself. The equation then could be rewritten into,

$$\ddot{\vec{r}} = -\frac{Gm_{\oplus}}{r^2}\frac{\vec{r}}{r} \tag{2.11}$$

or

$$\ddot{\vec{r}} = -\frac{\mu}{r^2}\frac{\vec{r}}{r} \tag{2.12}$$

where μ is the gravitational parameter of the referenced "celestial body", (McClain, 2013)

$$\mu = GM \tag{2.13}$$

With the assumption of the sum of both masses are equal to the Earth's mass, the inertial frame of reference for both bodies are shifted, virtually, to the center of the Earth. This creates a frame of reference that is going to be used within this thesis that is the Earth Centered Inertial (ECI). Now for the ECI, the $\hat{Z}$ is within the line joining the two poles of the Earth pointing north, the $\hat{X}$ axis is pointing to the what it is called *vernal equinox* (two points in space where the Sun and the Earth equatorial plane intersect), and the $\hat{Y}$ axis completes the right-handed orthogonal system, that is 90° to the East.

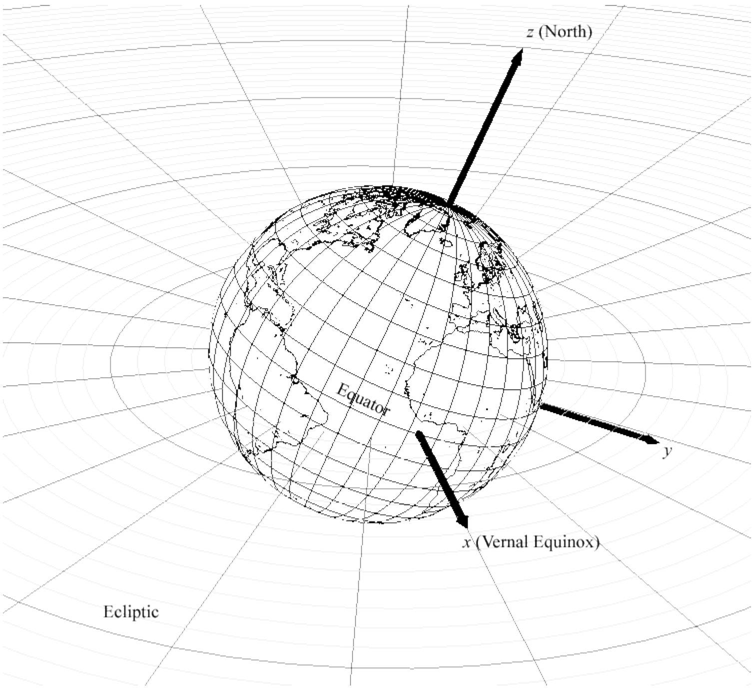


FIGURE 2.5: Earth Centered Inertial coordinate system (T. Kelso, n.d.-b)

2.2 Satellite State Representation

One way to define the state of a satellite is through its position and the velocity, in terms of vector with respect to a reference frame just like in the previous sections, Fig. 2.5. the vectors can be written as shown,

$$\vec{r} = r_x\hat{I} + r_y\hat{J} + r_z\hat{K}$$
$$\vec{v} = v_x\hat{I} + v_y\hat{J} + v_z\hat{K}$$

Just like any applications that uses these two parameters, these two vectors are used to predict or determine the displacement of the satellite at a point of time. However, there are other parameters of state representation that are a little bit more understandable, and they can also be converted from the position vector and the velocity vector. These quantities may take on many equivalent forms. Regardless of their form, the collection is usually referred to as Element Sets (McClain, 2013).

Element Sets are typically associated with scalar magnitude and angular orbital representations called orbital elements. Element Sets have many forms due to the variety of the orbital elements. These orbital elements are define in six quantities required to describe the orbit's size, shape, and orientation as well as the satellite's whereabout at an instantaneous point of time. The most common element set is the Classical Orbital Elements. The six quantities described in the Classical Orbital Elements are:

- Semimajor axis, a;
- Eccentricity, e;
- Inclination, i;
- Right Ascension of the Ascending Node (RAAN), Ω;
- Argument of Perigee (AoP), ω; and
- True Anomaly, ν.

Several other element sets have also been developed for convenience, and one of them is the *Two-Line Element* sets, which would be this thesis' interest source of data, that will be explained later on in this section.

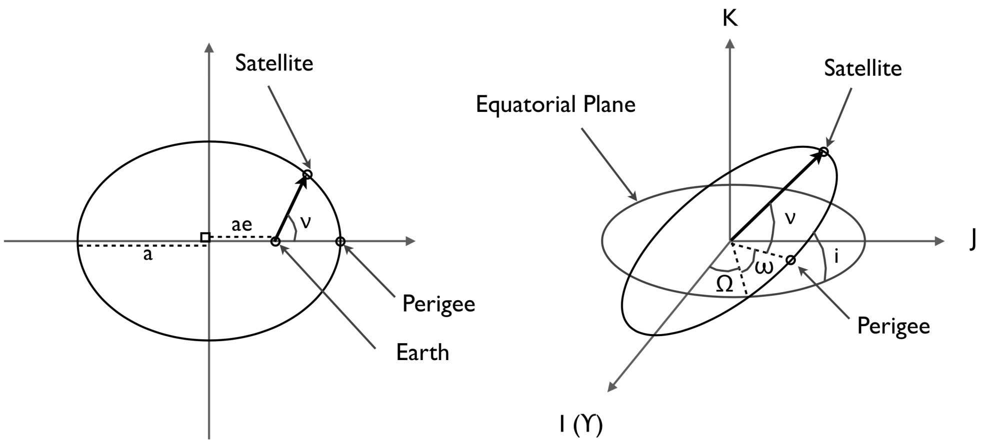


FIGURE 2.6: Classical Orbital Elements (Tombasco, 2011)

2.2.1 Semimajor Axis, a

Semimajor axis defines the size of the orbit. it is the sum of the length of the apoapsis, r_a, and the length of periapsis, r_p, divided by two. Note here that the periapsis is the closest point of the orbit and the apoapsis is the furthest point of the orbit, both in reference to a central body.

$$a = \frac{r_a + r_p}{2} \tag{2.14}$$

The semimajor axis can also be calculated using the *vis-viva* equation that describes the interaction of the orbiting body with the central body with the consideration of the energy (McClain, 2013).

$$a = (\frac{2}{r} - \frac{v^2}{\mu})^{-1} \tag{2.15}$$

2.2.2 Eccentricity, e

Eccentricity defines the shape of the orbit. It is a fixed constant for each type of conic section (circle, ellipse, parabola, and hyperbola) and it ranges from zero to one, though with some exceptions. It never gives a negative value. if the value is equal to zero then it is a *circular orbit*, if the value is between zero and one then the orbit is an *elliptical orbit*, if the value is equal to one then the orbit is *parabolic orbit*, but if the value is greater than one then the orbit is *hyperbolic orbit*. Eccentricity can also be defined as the ratio between the distance of the two foci and the semimajor axis, though it is only true for elliptical orbits (*Describing Orbits*, 2018),

$$e = \frac{c}{a} \tag{2.16}$$

where c is the distance between the two $foci$. For all conic sections, the eccentricity can be obtained from the magnitude of the eccentricity vector, that is (McClain, 2013),

$$\vec{e} = \frac{(\vec{v} - \frac{\mu}{r})\vec{r} - (\vec{r} \cdot \vec{v})\vec{v}}{\mu} \quad , \quad e = |\vec{e}| \tag{2.17}$$

2.2.3 Inclination, i

Inclination defines the tilt of the orbital plane. The tilt angle is measured from the equatorial plane of the central body to the orbital plane. It ranges from 0° to 180°. For all orbits that are inclined at 0° and 180° are called *equatorial orbits*, everything in between is called *inclined orbit*. The inclination of the orbit also determines the direction of the satellite revolving around the central body, say the Earth. If the inclination is between 0° and 90°, then the motion is in sync with the Earth or called *prograde orbits*. If the inclination is between 90° and 180°, then the motion of the satellite is opposing the rotation of the Earth or called *retrograde orbits*, and inclination of 90° is called polar orbit. Inclination can be written mathematically as (McClain, 2013),

$$\cos(i) = \frac{\hat{K} \cdot \vec{h}}{|\hat{K}||\vec{h}|} \tag{2.18}$$

where $\vec{h}$ is the specific angular momentum in vector form and $\hat{K}$ is the unit vector of K axis. the $\vec{h}$ can be obtained by the cross product of position vector $\vec{r}$ and the velocity vector $\vec{v}$.

2.2.4 Right Ascension of the Ascending Node (RAAN), Ω

Right Ascension of the Ascending Node or RAAN is the angle that describes the orientation of the orbital plane in space. It is the angle measured in the equatorial plane measured eastward as positive from the $\hat{I}$ unit vector or the X axis of the celestial body to the location of which the satellite's ascending node, or the point in which the orbital plane intersects the equatorial plane. The ascending node itself is whenever the satellite passes the equatorial plane from south to north, hence the name *ascending*. In the case of equatorial orbit, the RAAN is *undefined*. The angle ranges from 0° to 360°. RAAN can be written mathematically as (McClain, 2013),

$$\cos(\Omega) = \frac{\hat{I} \cdot \vec{n}}{|\hat{I}||\vec{n}|} \tag{2.19}$$

where $\vec{n}$ is the node vector and $\hat{I}$ is the unit vector of the I axis. The $\vec{n}$ can be obtained by crossing the unit vector of K and the the specific angular momentum $\vec{h}$.

2.2.5 Argument of Periapsis (AoP), ω

Argument of periapsis defines the location of the *periapsis* of the orbit. It is an angle measured from the ascending node to the periapsis in the direction of the motion of the satellite. The angle ranges from 0° to 360°. a perfectly circular or an equatorial orbit do not have an argument of periapsis due to the fact that they don't have periapsis or ascending node. Argument of periapsis can be written mathematically as (McClain, 2013),

$$\cos(\omega) = \frac{\vec{n} \cdot \vec{e}}{|\vec{n}||\vec{e}|} \tag{2.20}$$

2.2.6 True Anomaly, ν

True anomaly defines the location of the satellite in the orbit at an instant of time of observation. It is the angle measured from the periapsis to the location of the satellite in the motion of the satellite. It is undefined for a perfectly circular orbit due to the fact that it has no periapsis, though one point in an orbit can be defined as the referenced periapsis. True anomaly can be written mathematically as (McClain, 2013),

$$\cos(\nu) = \frac{\vec{e} \cdot \vec{r}}{|\vec{e}||\vec{r}|} \tag{2.21}$$

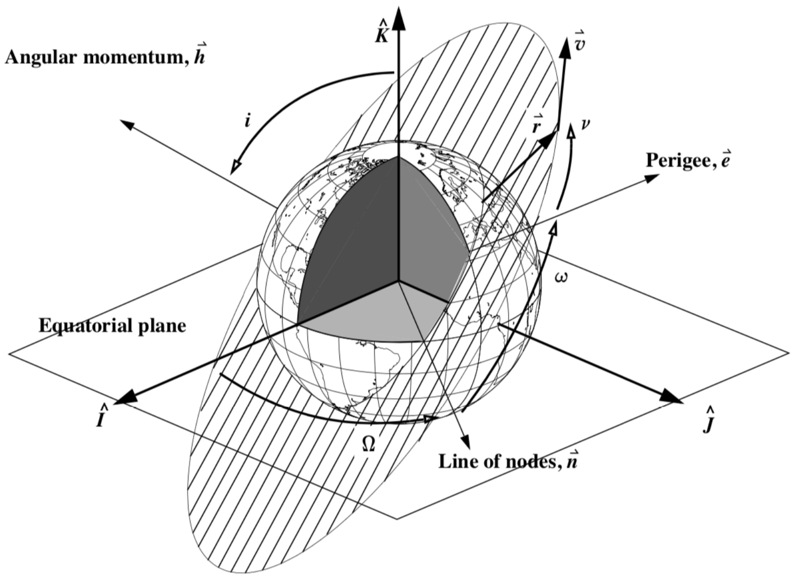


FIGURE 2.7: Classical orbital elements representation(Tombasco, 2011)

2.3 Space Surveillance

A need of a surveillance system for all of the orbiting objects, among other, to avoid unwanted events and to determine the whereabout of all orbiting objects has become the number one importance for all space mission. To give a perspective, the number of man-made objects that orbits the Earth kept on increasing throughout time, creating a hazardous space in which the objects are orbiting. In addition to that, a more apparent importance of space surveillance comes when there is an unavailability of the satellite's location in space. That unavailability would make any communication virtually impossible, making the satellites useless per their purpose to be orbiting the Earth (Greene, 2009).

One of the long-standing organization of several space surveillance is the Space Control Center (SCC), operated by NORAD. They are responsible for the detection, identification, and daily tracking of all man-made objects in space. When an Earth orbiting object is detected, NORAD releases a set of data of that particular detected orbiting object in the form of Two-Line Element (TLE) sets. In doing so, they utilize the traditional and phased-array radar systems, as well as some electro-optical methods (T. S. Kelso, n.d.).

The traditional/conventional radar system
Operating in bistatic mode, meaning that one antenna transmits a pulse another antenna receives the return pulse (T. S. Kelso, n.d.).

The phased-array radar system
Scan a large volume of space due to the fact that there are no mechanically moving parts. Instead, they are composed of thousands of small elements that can be phased to electronically steer the antenna. (T. S. Kelso, n.d.).

The Ground-Based Electro-Optical Deep Space Surveillance
Composed of three telescopes to image objects in space 10,000 times dimmer than that of the naked eyes are capable of. Computer processing removes the stars and other background light sources to produce a clear observations (T. S. Kelso, n.d.).

Those three methods of observations generate up to 80,000 satellite observations per day. Note that currently there are 47,381 catalogued objects with 22,095 of them orbiting the Earth and 3,674 of which are the operational satellites (*SATCAT Boxscore*, n.d.). Though 80,000 observations per day may seem like a rather big number of observations, it is still far from achieving a continuous let alone real-time observation and orbit determination of all Earth-orbiting objects. The main reason for this is because of the geographical distributions of ground-based sensors, see Fig. 2.8. The solution that the SSN implemented to this is that they use a predictive technique in monitoring the catalog of the space objects. What it means that they periodically making sure that each object is where it is predicted to be and they will generate a new set of elements when they are not where they are predicted to be (T. S. Kelso, n.d.).

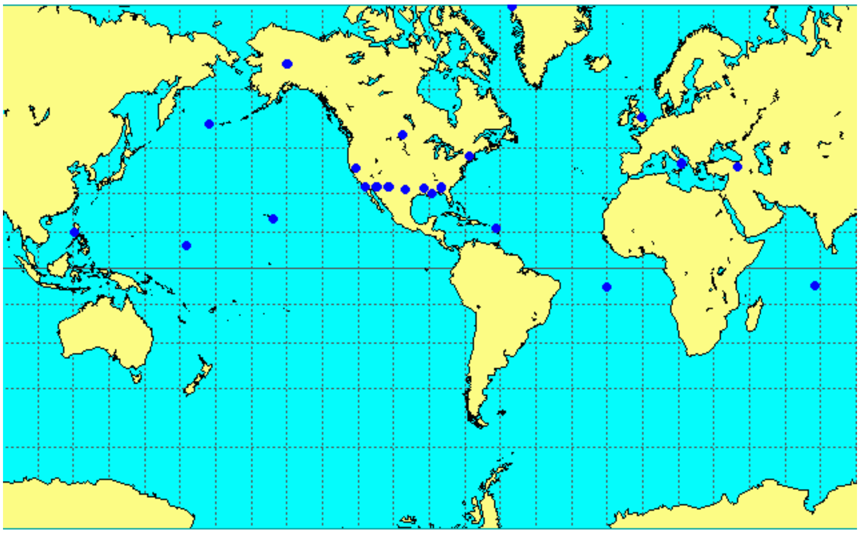


FIGURE 2.8: Space surveillance network observation location (T. S. Kelso, n.d.)

2.4 Two-Line Elements

Two-Line Elements or TLE, as mentioned before, is a set of state description data at a given point of time upon the detection of an Earth orbiting object. TLE data consists of two 69 characters lines of data which can be used to determine the state– the whereabout in term of orbital elements, of an Earth-orbiting object for an instantaneous point of time, or so it is called the *epoch*. The TLE, as mentioned also before, contains a lot of information for the use of determining the state of the Earth-orbiting object.

Recalling back in regards of all Earth-orbiting objects are in surveillance, every day, by NORAD and the observation is released as an ephemeris data in a form of TLE. The TLE generation, in terms of how frequent they are generated, for each objects are different. The updates are not base on a fixed time table, rather the frequency of updates is dependent on various factors. As an example, satellites within the Low-Earth orbit would receive a more frequent updates, due to the somewhat unpredictable atmospheric drag which may cause the satellite to alter the orbit ever so slightly.

TLE contains, among others, the information regarding the mean orbital elements of the Earth-orbiting objects. The "mean" here is resulted from the removal of the short-periodic and long-periodic variations (Greene, 2009). The mean values has an effect on the element, thus the set becomes just slightly different from what have been listed in the section 2.2. There are two classical orbital elements that are not listed inside the TLE, the semimajor axis, a and the true anomaly, ν. However, it is easily obtained from converting some elements inside the TLE into the desired classical orbital elements. The details on the conversion are in the next chapter.

2.4.1 Format Description

TLE is available for the public, which means that it is relatively easy to access. The TLE can be acquired from `https://www.space-track.org/` or `https://celestrak.com/`. both of which give the same TLE, though the procedures are slightly different. the TLE can be acquired in two ways. One that is the most updated TLE of the satellite. The other one is the set of TLEs of some period of time or historical data. the first one is generally available on the website's home page. the latter one should be requested first beforehand through a special data request. TLE, as stated before, is made out of two 69 character lines per ephemeris. In general, the only valid characters of the TLE is any number between 0-9, the character between A-Z, the spaces, the periods, and the plus and minus signs– and all of these characters are also valid within the specific columns. The description on the content of the TLE sets will be given in the following figure and the following tables.

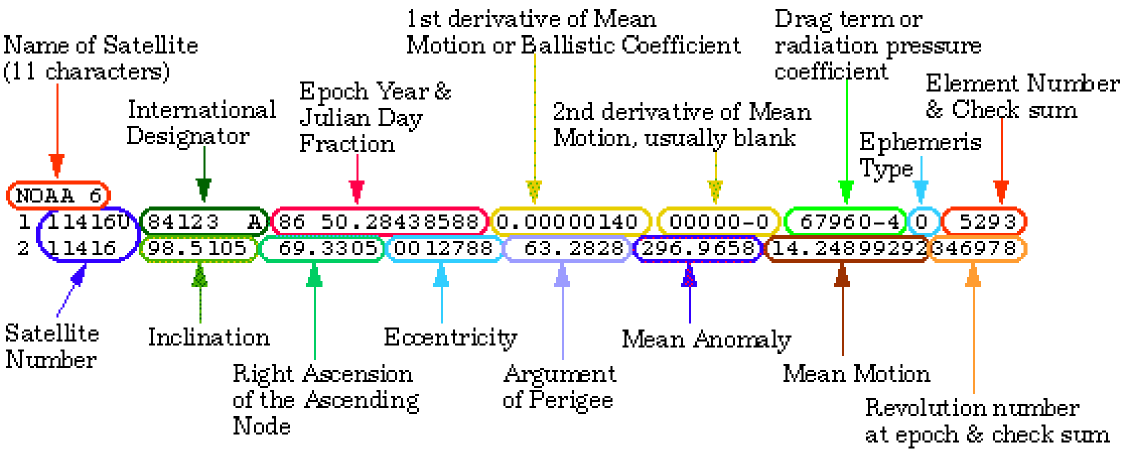


FIGURE 2.9: Two-Line Element Sets data description (*Definition of Two-line Element Set Coordinate System*, 2011)

Fields	Collumns	Content
1	01 - 01	Line number
2	03 - 07	Satellite catalog number
3	08 - 08	Classification (U=Unclassified, C=Classified, S=Secret)
4	10 - 11	International Designator (last two digits of launch year)
5	12 - 14	International Designator (launch number of the year)
6	15 - 17	International Designator (piece of the launch)
7	19 - 20	Epoch Year (last two digits of the year)
8	21 - 32	Epoch (day of the year and fractional portion of the day)
9	34 - 43	First Derivative of Mean Motion (Ballistic Coefficient)
10	45 - 52	Second Derivative of Mean Motion
11	54 - 61	Drag Term (Radiation Pressure Coefficient or BSTAR)
12	63 - 63	Ephemeris type
13	65 - 68	Element set number
14	69 - 69	Checksum (modulo 10)

Table 2.1: Two-Line Elements set's line 1 format definition

Fields	Collumns	Content
1	01 - 01	Line number
2	03 - 07	Satellite Catalog Number
3	09 - 16	Inclination (degree)
4	18 - 25	Right Ascension of Ascending Node (degree)
5	27 - 33	Eccentricity (decimal point assumed)
6	35 - 42	Argument of Perigee (degree)
7	44 - 51	Mean Anomaly (degree)
8	53 - 63	Mean Motion (revolution per day)
9	64 - 68	Revolution Number at Epoch (revolutions)
10	69 - 69	Checksum (modulo 10)

Table 2.2: Two-Line Elements set's line 2 format definition

2.4.2 Accuracy and Limitation

Just like any other observation, there should be an assessment on how accurate and precise the observation is. In this case, there are things that affects the accuracy of the TLE. The sensors or equipment used to detect the satellites, the amount of data gathered, the type of orbit the satellites' are in, the atmospheric condition in the space they are orbiting are some of the factors that affect the accuracy of the TLE sets. Unfortunately, these aspects are unique with each TLE set, which implies that the accuracy of each Earth-orbiting body is often so specific (T. Kelso, n.d.-a).

A consistency assessment of the TLE datasets is more probable to do rather than assessing the accuracy of the TLE itself. This implies to how well does the particular datasets' prediction are in accordance with those of its predecessors dataset (T. S. Kelso, 2007). By comparing the difference of the vector magnitude of the predictions from two successive element sets at the epoch of the newer element set (when it should be most accurate), it is possible to gauge the consistency between those element sets.

2.5 Satellite State Propagation

A new perspective of what was thought to be challenging or even impossible was able to be achieved with the advancement of satellite development. From worldwide communication, remote sensing of the Earth's surface, to the deep space astronomy and exploration are the examples of the advancement. These advancements would be useless if there are no means of keeping track and communication with them. Therefore, the knowledge of satellite tracking and orbital determination, prediction, propagation, and trajectory correction are genuinely vital. These vital pieces of knowledge boils down to one starting point, which is determining the satellite's initial state.

To be completely blunt, there are no means of keeping a continuous or real-time tracking of satellites. However, there is one way to at least know the state of a satellite at a given point of time, that is through the publicly available Two-Line Elements provided by NORAD, as what the previous sections describes. The TLE, as known now, does not update in a frequent or even a set timetable. That is actually enough to at least predict, using some mathematical model, the state of the satellite in the future or the past. The mathematical model are vary from the, relatively, simple to the ones that considers various factors that might affect the motion of the orbiting objects, called orbit perturbations.

2.5.1 Orbit Perturbations

Another crucial foundation in astrodynamics is the study and model perturbations. Perturbation itself are defined as the deviation of an idealized or undisturbed state of motion. Recalling back from the two-body problem, it is assumed that the only forces acting on the bodies are only the forces of gravity exerted by each of the body toward one another, meaning that the motion is "idealized". The real world, however, actually have other forces or disturbance, just like the definition of perturbation. Now, taking account of the perturbation(s) into the two-body problem is necessary. In a nutshell, the equation becomes (Curtis, 2014),

$$\ddot{\vec{r}} = -\frac{\mu}{r^2}\frac{\vec{r}}{r} + \vec{p} \tag{2.22}$$

Note here that the $\vec{p}$ is the net perturbative acceleration from sources, other than the gravitational attraction of the two-bodies. Some paper also uses $\vec{a}$ to describe the perturbative acceleration.

Gravity Field

The Earth or any other celestial bodies are not perfectly spherical, they are more like oblate spheroids, due to the fact that the celestial bodies are spinning or rotating in its axis of rotation. That is where the centrifugal effects comes and causes the equatorial radius of the body to be relatively larger in comparison to the polar radius. Due to this, the gravity of the Earth is not exactly uniform for all points on the surface, but rather varies with the latitude and the radius. This oblateness of a shape, if you may call it, has actually one of the stronger effect on the satellite orbit, near the planet, and when written, the perturbation Φ is given by the infinite series,

$$\Phi(r,\phi) = \frac{\mu}{r}\sum_{k=2}^{\infty} J_k(\frac{R}{r})^k P_k \cos(\phi) \tag{2.23}$$

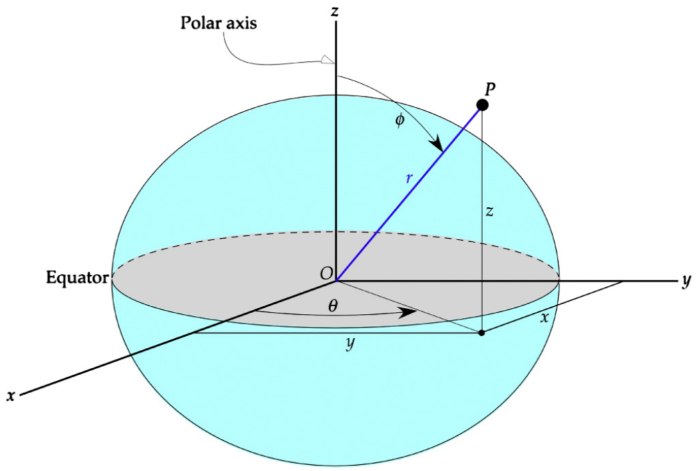


FIGURE 2.10: Spherical Coordinate System (Curtis, 2014)

where,

- J_k, in the equation is the zonal harmonics of the planet, that is a dimensionless number unique to each planet;
- R, here the Equatorial radius;
- P_k, is the Legendre polynomials; and
- J, number extends to infinity, as it is an infinite series

The most significant is the J_2, where the value is J_2 is 0.00108263 (Curtis, 2014). The perturbation acceleration due to J_2 can be written as,

$$a_I = -\frac{3}{2}\frac{J_2\mu R_{\oplus}^2 \vec{r_x}}{r^5}(1 - \frac{5\vec{r_z^2}}{r^2}) \tag{2.24}$$

$$a_J = -\frac{3}{2}\frac{J_2\mu R_{\oplus}^2 \vec{r_y}}{r^5}(1 - \frac{5\vec{r_z^2}}{r^2}) \tag{2.25}$$

$$a_K = -\frac{3}{2}\frac{J_2 \mu R_{\oplus}^2 \vec{r_z}}{r^5}(3 - \frac{5\vec{r}_z^2}{r^2}) \tag{2.26}$$

Atmospheric Drag

The planet's oblateness aside, the atmospheric drag is also one of the stronger influence on the motion of the satellite, especially orbits that are near the planet. Despite the fact that 99% of the Earth's atmosphere is below 100 km, the air density at higher altitude and even though there are way less air molecules, is still sufficient enough to cause drag considering the orbiting velocity for that altitude itself is very high (± 7.9 km/s) (Curtis, 2014). Based on the drag equation,

$$D = -\frac{1}{2}\rho v_{rel}^2 C_D A \tag{2.27}$$

where,

- ρ, is the atmospheric density;
- A, is the frontal area of the satellite or spacecraft; and
- C_D, is the dimensionless drag coefficient.

Then the acceleration for $\vec{p}$ can be written as,

$$\vec{p} = -\frac{1}{2}\rho v_{rel}\Big(\frac{C_D A}{m}\Big)v_{rel} \tag{2.28}$$

Third-Body Perturbation

The "real world" has other celestial bodies that can effect the orbital motion of a satellite. When another celestial body's gravitational attraction is taken into an account, the problem is called the three-body problem. For the Earth, one of the celestial body that can and should be taken into an account is the gravitational pull of the Earth's moon, or the Sun for that matter can also be considered as another third-body perturbation. The satellites that is in a higher orbits have greater effects in comparison to the lowers orbits.

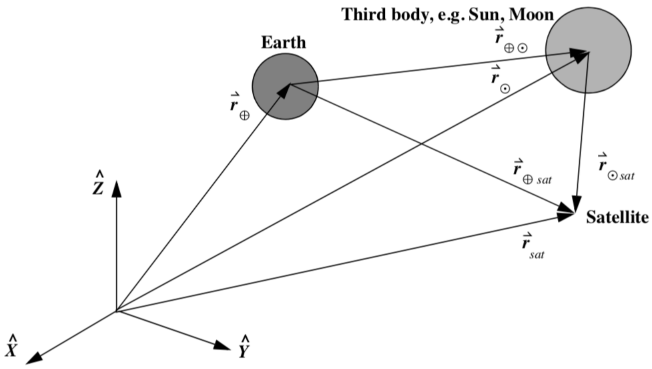


FIGURE 2.11: Representation of other body as the third-body perturbation (McClain, 2013)

2.5.2 Satellite State Propagator

The propagation models make use of the current state of the satellite to predict the state of the satellite to any given point in time of our interest. A simplification of the prediction would be like predicting a car position within a highway where we know the initial position and the initial velocity, also where it is heading. By that, we would be able to predict its whereabouts in, let us just say, an hour. Similarly, if the satellite's initial state is known, its future state could be predicted to some reasonable accuracy.

Of course, that is if the condition of the motion is unperturbed or idealized. However, perturbation occurs in the path of the satellite motion. These perturbations are caused by the Earth's shape (spherical harmonics), drag, radiation, and gravitational effects from other celestial bodies (the sun and moon generally) (Miura, 2009). The propagation model's job is to take those perturbation(s) into account for it to predict the satellite state at any time of interest within some degree of accuracy.

In accordance with the use of the propagation models, all of them represent "solutions" to the equation of motion for two or more bodies. The propagation models are also divided into four categories, namely Two-Body, General Perturbation, Special Perturbation, and Semi-Analytical Theories (*What is an Orbit Propagator?*, 2000), though this thesis will cover more on the two-body, two-body with the first zonal harmonic of the Earth perturbation (J_2), and the general perturbation (SGP4 in specific).

Two-Body

Two-Body propagator is an exact solution to the equation of motion for two mutually attracting bodies, see section 2.1.3. A two-body propagator is the ideal modeling and for educational purposes, but should not be used for any practical use. The Two-Body propagator considers only the force of gravity from the Earth, just like the first section of this chapter, that is modeled as a point mass. Virtually the orbit in an idealized system would virtually repeats its exact path. Meaning that there would not be any changes in the orbital elements of the satellite.

The use of J_2 perturbation in conjunction with the two-body propagator only accounts for secular variations in the orbit elements due to Earth oblateness. Due to the oblateness of the Earth the orbital elements would not be constant. The two of the orbital elements that mainly affected by this perturbation is the RAAN and the AoP, in other words the orbit would shift ever so slightly over time.

Simplified General Perturbation Models

The development of the Simplified General Perturbation or the SGP model series was back in the 1963. The model includes the first three zonal harmonic for the gravity model, and the approximation of atmospheric drag through rate of change in mean motion, and also it does not include the effect any third body gravitational effects (McClain, 2013). The development of the models peaked with the release of the Simplified General Perturbation 4 (SGP4).

With the introduction of Spacetrack Report No. 3, a user compatibility survey on space surveillance and the official users, the SGP4 propagation models' source code had been further optimized (Vallado, 2006). It considers secular and short- and long-periodic variations due to Earth oblateness, solar and lunar gravitational effects, gravitational resonance effects and orbital decay (e.g., atmospheric drag that uses power density functions) (Greene, 2009). The models specifically relied on the Two-Line Element (TLE) dataset, as this thesis is also relying on the TLE dataset.

2.5.3 Propagation Error

In doing measurement there will always be uncertainties that prevent the measurement to be 100% accurate. The inevitable uncertainties that occurs in doing measurements are what it is called the *error*, not to be confused with mistakes (Taylor, 1996). The purpose of the error analysis is to study and evaluate these inevitable uncertainties in the measurements in which to refine such measurements in achieving an asymptotical value of that the *actual* or *real* value of an event. The actual or real value of an event is not actually known, that is why a preliminary or an initial observation or measurement or maybe a theoretical prediction is needed as a guideline for which purpose is to approximate the value of an event. As stated before, all measurement including for orbital problems, namely orbital observation, orbital determination, and orbital prediction would encounter errors. These errors are mainly measuring errors, modeling errors, and methodology errors (Li, 2017).

From the orbital observation point of view, which is done by NORAD by using their means of cataloging and keeping the ephemerides of the satellites updated, the errors are due to the particular sensors that are being used for observations. Deviation and noise from the equipment that is used, the atmosphere of which the observation time is in progress to the amount of data that is being collected are also some parts of it. From that they could only acquire and optimally approximated values of the object's true state. The orbital determination, that is the TLEs that are acquired from the observations, only gives the mean value of the object's state that is actually not the true value of the orbit due to the observation's errors. The error of the initial determination is then passed through to the orbital prediction, since in predicting a future state of orbit the initial state should be defined. The other error that is present in the orbital prediction is the modeling of the prediction itself.

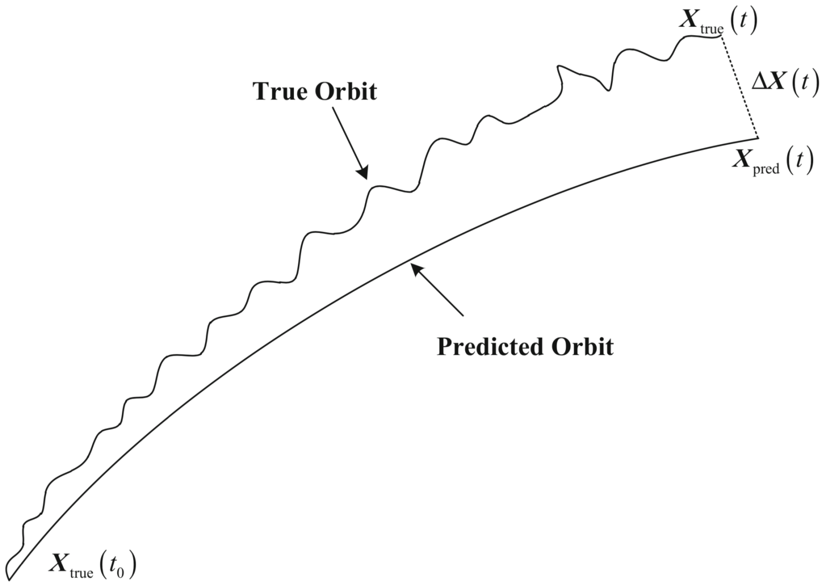


FIGURE 2.12: Propagated initial state with its initial errors (Li, 2017)

2.5.4 Regression Analysis

Regression analysis is one of the most common analysis that is used to analyze the, hypothetical, behavior and relation between variables. Regression analysis is divided into three categories, namely, linear regression, multiple linear regression, and nonlinear regression. The common ones to be used for relatively simple data sets are the linear regression analysis and the multiple linear regression analysis, while the nonlinear regression analysis is for more complicated data sets. In regression analysis there are two kind of variables to be analyzed, the first one is the dependent variables and the second one is the dependent variables. The dependent variable, as its name may suggest, is hypothetically said to change over the change of the independent variable. When graphed the independent variable is depicted as the X axis and the dependent variable is depicted as the Y axis.

$$Y = \beta_0 + \beta_1 X \tag{2.29}$$

where,

- Y, is the expected value of y for a given value of x;
- β_0, is the y-intercept of the regression line; and
- β_1, is the slope of the line.

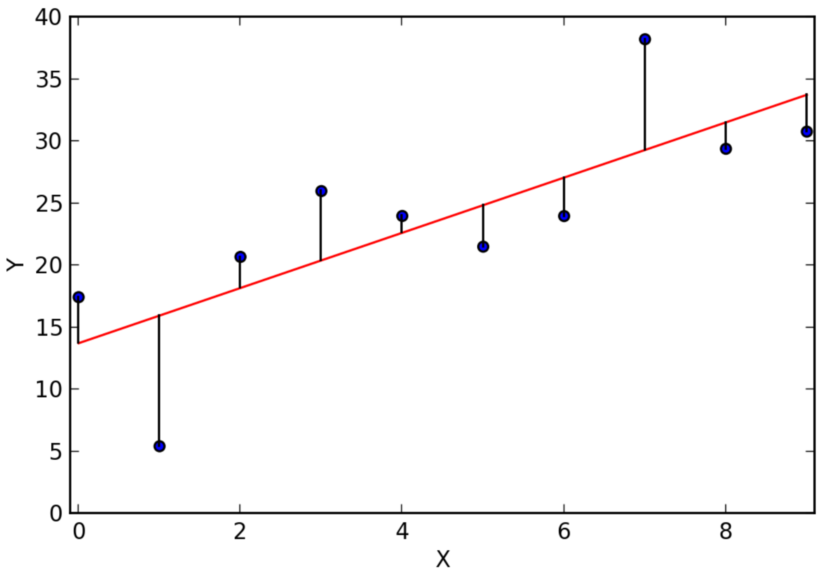


FIGURE 2.13: Linear regression analysis example plot (*Regression Analysis*, n.d.)

What that means is that if the dependent variable increases with the increase of the independent variable, the relation is named positive relation. If the dependent variable decreases with the increase value of the independent variable, then the relationship is negative relation. The relationship between the variables are described or represented with the linear line called the regression line.

2.5.5 Fourier Series

It is a study of a way some functions may be represented or approximated by sums of simpler trigonometric functions. The functions are, in this case, periodic functions, that is composed of harmonically related sinusoids (Howell, 2016). In other descriptions, it also represents a periodic function by a discrete sum of complex exponentials. Putting it into some words, it gives the main idea of an analysis that compare the signal with an infinite sum of sines and cosines of various frequencies. Fourier series itself make use of the orthogonality relationships of the sine and cosine functions, and the equation for the series are given in the following expression,

$$f(x) = a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega x) + \sum_{n=1}^{\infty} b_n \sin(n\omega x) \tag{2.30}$$

where,

- $f(x)$, is real-valued function;
- a_0, a_n and b_n, are the Fourier coefficients; and
- ω, is the frequency.

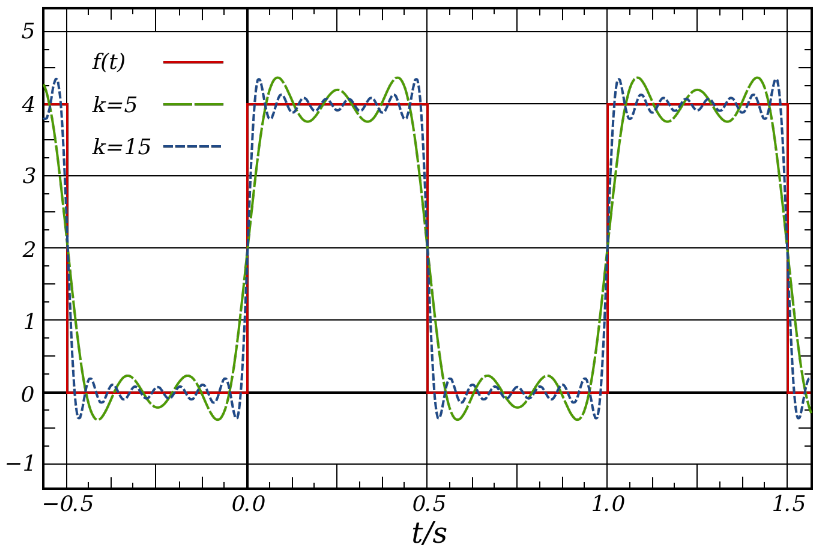


FIGURE 2.14: Square wave represented by different order of Fourier series

2.5.6 R^2 Correlation

The behavior of the data does not really give too much information on how well the model fits the data or how does the variables correlates to one another. That means that there is a need to find a way to measures the strength of the relationship between the two. R-squared or R^2 can be used in conjunction with the linear regression for the correlation analysis for the variables. The value of correlation is measured on a convenient 0% – 100% scale.

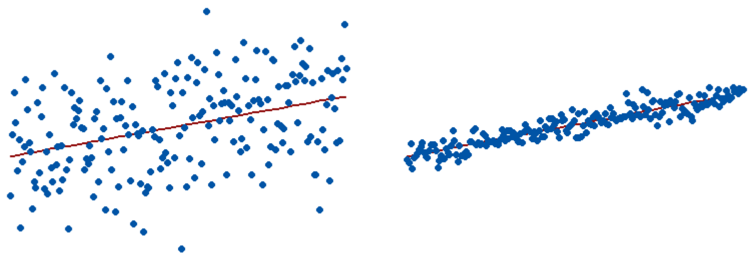


FIGURE 2.15: High value of R^2 and low value of R^2 (left to right) (Frost, n.d.)

The closer the value of R^2 to 0% depicts that none of the variability of a model, response data around its mean. While 100% on the other hand, explains all of the variability of the model, response around its mean. The mean of the dependent variable predicts the dependent variable as well as the regression model. R^2 does not indicate whether a regression model is adequate, a low R^2 value for a good model might occur.

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Research Outline

With this section being named *Research Methodology*, it has the purpose to give an outline of what kind of approach that the author had taken in order to try and "tackle" the problems stated in this thesis. Explicitly speaking, this chapter gives the ever so detailed approaches in doing the thesis. Here is the list of the approaches made by the author:

1. **Problem Statement**
   The author formulated the initial problem based on the lack of a real-time satellite orbit determination, for it is very essential for orbit maneuvering, collision avoidance and accurate orbit tracking for mission purposes. In this case the author chose three satellites, all of them owned and operated by LAPAN, as the subjects for this thesis. The satellites then went through a process called state propagation or orbit propagation in which the state of the satellite would be predicted in the near or long future, all of that was for the propagated values to be analyzed. The propagated states then evaluated with the timestamp of the updates within the satellites' respective TLE historical data.

2. **Literature Study**
   The author took upon various references such as journals, websites, and books that are related to the fields / materials in aiding this research. For the baseline of knowledge the author took the basic astrodynamics references on how satellite's state is represented, propagation method, and orbital mechanics. In addition to that, the goal of this thesis is to analyze the errors in the

propagation, so sufficient knowledge on the error analysis was essentials for the author to get his hand on.

3. **Computational Tools**
   The author used open-source programs for gathering the needed materials for the thesis, creating mathematical models, simulations regimes, and error analysis. The use of the open-source programs gave a massive helping hand for the author due to the community forums of each programs that were very helpful, and also due to the fact that the author did not have to spare a single penny to do what stated.

4. **Data Collection**
   The collection of data is the *base* building block of this thesis, for it was used as the input and the reference values. The author did the data collection through a publicly accessible website. In the data collection itself the author went ahead and request all of the data of the satellites of interest for this thesis.

5. **Mathematical Model**
   The author build mathematical models as of the case in this thesis are the mathematical models for the orbital elements extractions, orbital element conversions, and orbit numerical model of the propagations. The author constructed it with the help of the computational tool stated before, which gave the ability to model all of them with some degree of ease. There were also some external use of some numerical models used by the author, for they were out of the author's comprehension at the time this thesis was conducted.

6. **Simulation**
   For the simulation, the author used the numerical model propagations that the author had previously built. The author set all the common parameters for the propagation of the satellites' states, for which the details are stated later in this chapter. The simulation also used one of the external numerical model that are going to be mention in this chapter. The simulation would be the half way, may so to say, of this thesis, for then the observation of the propagation would be done for further analysis by the author.

7. **Error Analysis**
   With the mathematical tool set and ready to be used by the author, it gave the ability to get the data needed for comparison. The comparison itself was to see the error (over time) of each propagator model in reference to the TLE historical data as the true state of the satellite. The comparison between the propagators in reference to the TLE ephemerides gave some insight of the "behavior" of the propagation over time.

3.2 Computational Tools

In conducting this research, the author has thought and be taught to utilize computers to do all the scientific computing. With the help of computer, the author gained the ability to easily collect data as well as to process them in such way they were needed. The author also was thought to use some programs in a hope to ease the research to some degree. The programs were all open-source so that the author had quite a handful of references on how to utilize the programs to its optimal use. Most of the programs were in the field of utilizing programming language and its packages or modules. All of that helping tools were in a hope to make an easier calculations, models for simulations, and analysis for the thesis.

3.2.1 Python Programming Language

The programming languages that was used in conducting this research was *Python*. The first reason to why the author chose Python was that Python got a lot of already existing building blocks corresponding to classical numerical methods or basic actions, and in a sense it was straight forward to use. The second reason was that it has been a "go to" programming language for the scientific computation for quite some time. The third reason was because Python was relatively easy to comprehend compared to other programming language, and since the author has little programming background, especially in scientific computing, it was a good choice of programming language for the author.

3.2.2 Anaconda

Anaconda is an open-source software distribution package created by the Anaconda Inc. As it is an open-source software, it is accessible for all category of uses, from individual use to enterprise use. Anaconda comes with, for the individual edition, over than 250 packages and also over than 7500 additional plug-ins and packages suited for whatever the user need (*Anaconda Individual Edition*, 2020). The author then believed that by using Anaconda and its Python IDE, Spyder, would be a great choice to be used with the computational and modeling needed in this thesis.

3.2.3 Numpy

Numpy is an open-source additional package for Python with the ability to enable numerical computing (*ABOUT US*, n.d.). This package helped a lot along the way of this research, as the author needed its numerical computing power with powerful numerical arrays objects, and routines to manipulate them. The manipulation of arrays with this package gave so much room for the author to work with. The ease of use of this package was also implemented in parallel to other packages throughout the codes that the author had built and used.

3.2.4 SciPy

Scipy is a collection of open-source software that is used as an *ecosystem* for scientific computing, of course for Python (*1.6. Scipy : high-level scientific computing*, n.d.). Its submodules correspond to different applications, such as integration, optimization, statistics, special functions, etc. Scipy is meant to be operated alongside with Numpy array which provides a high performance. With this, Scipy gave much of the helping hand needed by the author to compute some functions with some degree of complexity.

3.2.5 Pandas

Another useful Python package that the author used was Pandas, an open source data analysis / manipulation tool. It provides a fast, flexible, and expressive data structures that are designed to be easy and intuitive for the users (McKinney & the

Pandas Development Team, 2020). All of the manipulated arrays were efficiently computed with the Numpy and with the use of Pandas in parallel to it, the author gained the ability to intuitively create a more expressive data output, in specific. With the use of Pandas the author also had the ability to store the output in such way that it was compatible to be opened with other programs.

3.2.6 Matplotlib

Matplotlib is an additional library for Python that enables visualization of data in Python (*Matplotlib: Visualization with Python*, n.d.). Using this library allowed the author to visualize a lot of data that were needed to be represented with a high fidelity, as you might say, results fitted with the author's customizations. This python package helped a lot in the post-processing of the data for this thesis. All of the graphs were mostly use for visual representation, with the hope that the readers would have an intuitive sense of what the data and also the analysis behave.

3.2.7 cysgp4

*cysgp*4 is a Cython, an extension of C language in Python programming, powered package that wraps the *C++* SGP4 Satellite library for which it is dependent on the use of the two-line elements (TLE). The intention of this library is of course to calculate or rather predict the orbit of the satellites, in preference of course, around the Earth at any time of interest. The *cysgp*4 works with arrays of TLEs and make use of multi-core platforms to boost processing times a lot (*cysgp4 0.3.3*, n.d.). This module helped the author get the model for the SGP4 propagator easily since it was out of the author comprehension to make.

3.2.8 Scikit-learn

Scikit-learn is a Python module that integrates a wide range of machine learning algorithms for medium-scale supervised and unsupervised problem. This package helped the author as it is intended for non-specialist in machine learning users. It gave an easy feel of use as it uses a consistent, task-oriented interface. It is also relying on the scientific Python ecosystem for which it can easily be integrated into application outside the traditional range of statistical data analysis (l Varoquaux; Alexandre Gramfort, 2011).

3.2.9 symfit

symfit is a Python package that exist to create a more "Pythonic" or simply creating a straightforward code that exist for fitting of data. Due to the symbolic nature of symfit, it saves the author the trouble of having to determine the derivatives. Furthermore, having this Jacobian allows good estimation of the errors in your parameters, something scipy does not always succeed in (*symfit 0.5.3*, 2020).

3.3 Data Collection

In conducting this research, and for it to be inline with the topic, the author needed to acquire the needed parameters. In doing so, the author thought of acquiring the parameter through the publicly accessible site which will be further described in the following subsections. The data collected was an essential part for the starting line of this thesis. The author decided to go with three different satellites operated by LAPAN. To give a general idea on what satellites that were being used, the following subsection will discuss about each satellites.

3.3.1 Lembaga Penerbangan dan Antariksa Nasional (LAPAN)

On 27 November 1963, *Lembaga Penerbangan dan Antariksa Nasional* or LAPAN has finally established with the Presidential Decree No. 236 of 1963 concerning LAPAN, after the presence of an informal space agency prior to the year (*Sejarah LAPAN*, n.d.). LAPAN is a Space Agency Organization, accountable to the President through the Minister responsible for government affairs in the field of research and technology. LAPAN has the task to perform government functions in the field of aerospace research and development and its use, as well as the management of space, in accordance with the provisions of the legislation (*Tugas dan Fungsi*, n.d.).

Historically, with the task given to LAPAN, they have done research on rockets, remote sensing, satellites, and space sciences for decades. Regarding satellite innovation, LAPAN currently operates and owns three satellites. The first satellite is the LAPAN-TUBSAT / LAPAN-A1, the second one is LAPAN-ORARI / LAPAN-A2, and the third one is LAPAN-IPB / LAPAN-A3. There are two other satellites that will orbit the Earth soon enough, LAPAN-A4 and LAPAN-A5.

LAPAN-TUBSAT / LAPAN-A1

LAPAN-A1 / LAPAN-TUBSAT satellite is the first generation LAPAN satellite. The primary objective of this satellite is to monitor the Earth. The LAPAN-A1 / LAPAN-TUBSAT satellite was successfully launched from Sriharikota, India, on 10 January 2007. It has a polar orbit and is intended for monitoring the Earth (*LAPAN-A1 / LAPAN-TUBSAT*, n.d.).

FIGURE 3.1: LAPAN-A1 (*LAPAN-A1 / LAPAN-TUBSAT*, n.d.)

LAPAN-ORARI / LAPAN-A2

The LAPAN-A2 satellite is the second generation LAPAN satellite, the successor to the LAPAN-A1 satellite. The main purpose of this satellite is for communication, earth observation, and Traffic Monitoring. The LAPAN-A2 satellite was launched from Sriharikota, India, on 28 September 2015 (*LAPAN-A2 / LAPAN-ORARI*, n.d.).

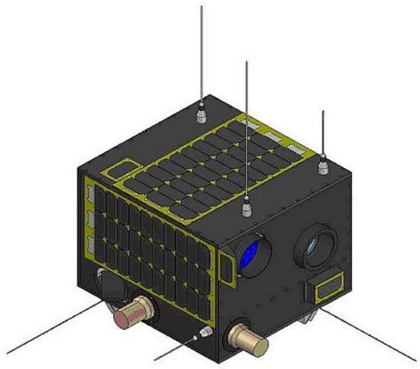


Figure 3.2: LAPAN-A2 (*LAPAN-A2 / LAPAN-ORARI*, n.d.)

LAPAN-IPB / LAPAN-A3

LAPAN-A3 is a third generation LAPAN satellite. This satellite is a successor to the two previous satellites, the LAPAN-A1 and the LAPAN-A2. The main objectives of this satellite is for Earth monitoring, ship monitoring and measurement of the Earth's magnetic field. The LAPAN-A3 satellite was launched from Sriharikota, India, on 22 June 2016 (*LAPAN-A3 / LAPAN-IPB*, n.d.).

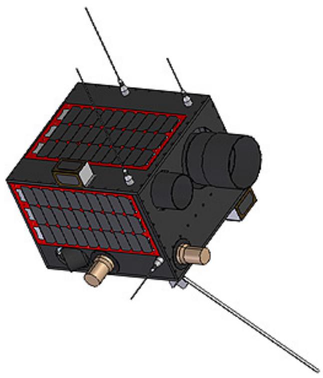


Figure 3.3: LAPAN-A3 (*LAPAN-A3 / LAPAN-IPB*, n.d.)

LAPAN-A1	
Nation	Indonesia, Germany
Type / Application	Earth observation
Operator	LAPAN, TU-Berlin
Contractors	TU-Berlin
Payload	1 x 3CCD Color-Camera with 6 m GSD 1 CCD Color-Camera with 200 m GSD
Attitude Control	3 wheel/gyro pairs (RW 203 wheels + WDE, fiber optical gyros) Star Sensor
Communication	2 TTCs, UHF 437.325 MHz, 1200 bps
Data handling	
	3.5 W RF S-Band Payload Communication 2220 MHz 524 kB external and internal RAM, 524 EEPROM, 16kB ROM, 38.4 kbps SCI Interfaces
Power System	4 Solar Panels, 432 x 243 mm, 35 cells in series, max. 14 W 5 NiH2 batteries, 14 V nominal voltage, 12 Ah
Dimension	450 × 450 × 270 mm
Propulsion	None
Lifetime	1 year (design)
Mass	57 kg
Orbit	635 km × 635 km, 98 deg (polar)

Table 3.1: LAPAN-A1 Specification

LAPAN-A2	
Nation	Indonesia
Type / Application	Earth observation, amateur communications, traffic monitoring
Operator	LAPAN
Contractors	LAPAN
Payload	Digital Space Camera CCD Color Video Camera Automatic Identification System (AIS) Receiver Voice Repeater & Automatic Packet Reporting System (APRS)
Attitude Control	3 Wheel/Fibre Optic Laser Gyros in Orthogonal Axis 2 CCD Star Sensor Magnetic Coil 6 Single Solar Cell for Sun Sensor 3 Axis Magnetic Field Sensor
Communication	2 TT7C UHF 1200 bps, FFSK modulation, 3W output
Data handling	
	S-Band payload Communications, 3.5 W RF output OBDH 32 bit RISC Processor, 128/256 byte internal, 1 Mbyte RAM and 1 Mbyte Flash Memory External
Power System	4 GaAs Solar Array, 465 x 265 mm, 30 cells in series, Max 230W(EOS) 4 Lithium-ion Batteries, 16V nominal Voltage, 18 Ah Total Capacity
Dimension	600 x 470 x 380 mm
Propulsion	None
Power	Solar cells, batteries
Lifetime	
Mass	74 kg
Orbit	638 km $\times$ 658 km, 6 deg (equatorial)

Table 3.2: LAPAN-A2 Specification

LAPAN-A3	
Nation	Indonesia
Type / Application	Earth observation, communications, traffic monitoring
Operator	LAPAN, ORARO
Contractors	LAPAN
Payload	Push-Broom Camera Multi Spectral Digital Space Camera CCD Color Video Camera Automatic Identification System (AIS) Receiver Space Based Magnetometer
Attitude Control	4 Reaction Wheels & Gyros 2 Star Sensors 3 Magnetic Coils 6 Single Solar Cell for Sun Sensors 1 Pitch Sensor 2 Horizon Sensor
Communication	UHF for 2 TTCs; 1200 bps; FFSK
Data handling	
	X-Band: 105 Mbps; 5 W max RF Output S-Band: 3.5 W RF Output OBDH 32 bit RISC Processor On Board Data Solid State Memory 4GB RAM & 16 GB Flash Memory with CCSDS Formatter
Power System	5 GaAs solar arrays: @ 46.5 cm x 26.2 cm, 30 cells in series; max power of 37 Watt Li-On Battery with capacity of 36 Ah with 16 V Nominal Voltage
Dimension	667 x 574 x 960 mm
Propulsion	None
Power	Solar cells, batteries
Lifetime	
Mass	115 kg
Orbit	500 km $\times$ 516 km, 97.51° (polar)

Table 3.3: LAPAN-A3 Satellite Specification.

3.3.2 TLE Data

This thesis solely relied on the data acquired inside of the Two-Line Element (TLE). The TLE itself gives the mean orbital elements of an Earth orbiting objects at a specific and spontaneous time. As stated in the previous chapter, the TLE are available for the public through a website and the author was able to obtain the data fairly easy. The TLE data in this thesis are the TLE historical data of LAPAN-A1, LAPAN-A2, and LAPAN-A3. All of the data were obtained through the *celestrak* website. The TLE data were necessary to give a glimpse about the behavior of the satellite state over time, of course with its respective epoch. Thus, the data were used as the *real* or *controlled* data for which all the propagator need to refer to in the simulation and for further analysis.

Obtaining TLE Data

The first thing the author did in collecting the TLE data was by accessing the special data request webpage provided in *celestrak*, `https://celestrak.com/NORAD/archives/request.php`, where then the author was asked to fill some required personal information and the information in regards to the satellites of interest. The needed to fill data for the satellites were the satellites' catalog number, the start date, and the stop date. For the satellite section, the author input as the following table,

Catalog No.	Satellite Name	Start Date YYYY-MM-DD	End Date YYYY-MM-DD
29709	LAPAN-A1	2007-01-10	2020-12-07
40931	LAPAN-A2	2015-09-28	2020-12-07
41603	LAPAN-A3	2016-06-22	2020-12-07

Table 3.4: Satellites data to be requested

Then the author needed to define starting date is date of which the ephemerides data started, in this case the author set all of the starting time to be the date of launch of each satellites. For LAPAN-A1 the author set to 10 January 2007, for

LAPAN-A2 the author set to 28 September 2015, and for LAPAN-A3 the author set to 22 June 2016. The stop date is the last date of the historical data, and for all the satellites the author set it to stop on 7 December 2020. All of the details are in the Table. 3.4. Then, the requested data of the satellites were sent by an email to the author in a form of *.txt* file. The preview of the content of the raw file can be seen below,

```
1 40931U 15052B   15271.78630001  .00000639  00000-0  00000+0 0  9998
2 40931   5.9992  25.6568 0013796   3.6374  59.9065 14.76340487    80
1 40931U 15052B   15272.31483986  .00000638  00000-0  00000+0 0  9995
2 40931   5.9988  21.9158 0013213  11.0283 349.0707 14.76347526   161
1 40931U 15052B   15272.65260880  .00001350  00000-0  10000-3 0  9995
2 40931   6.0001  19.5263 0013132  15.4783 344.5656 14.76348257   213
1 40931U 15052B   15272.72016628  .00000445  00000-0 -27196-4 0  9992
2 40931   5.9996  19.0388 0013165  16.6402 343.4213 14.76347078   222
```

The total received data of LAPAN-A1 was 32,392 lines, LAPAN-A2 was 4,648 lines, and LAPAN-3 was 20,390 lines. Divide them by two gave how many observations were made for all of them– 16,196 for LAPAN-A1, 2,324 for LAPAN-A2, and 10,194 for LAPAN-A3. Now remember, those numbers described the total of observations made from the day those satellites were launched into their orbit. But of course, there need to be an evaluation of the raw data before the author could process them even further.

All of the epochs or the time of observation between each satellites would differ from one another, as previously stated in the last chapter. This difference in the epochs of the satellite was not entirely a set back for the processing of the TLE data nor the simulations, since the data processing and simulations of each satellite would be done separately and according to each respective epochs. That means the author needed to process the requested TLE data into a more flexible for them to be manipulated or processed further. One other thing to be noted is that some of the data might have duplicates in them, this might be due to multiple updates at once. This set back would be explained after the following subsection.

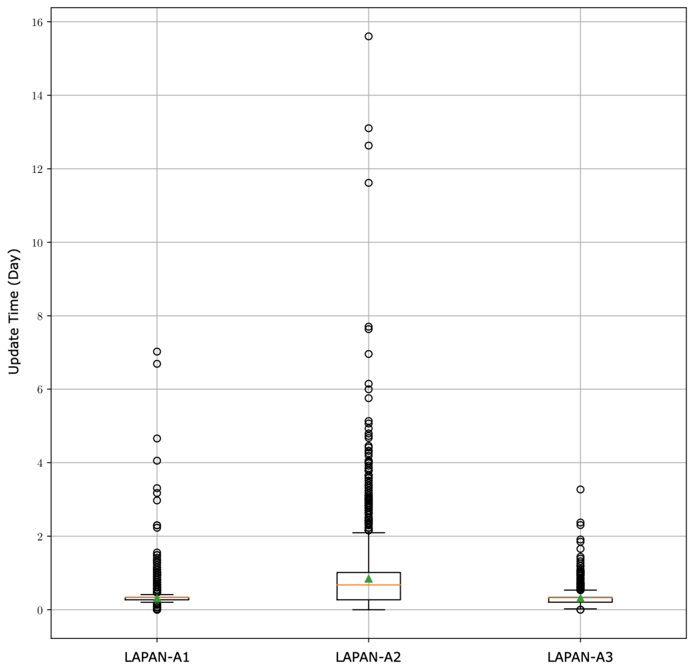


FIGURE 3.4: TLE update frequency of LAPAN's satellites

As shown in the Fifg. 3.4, all of the satellites received different different frequency of updates. For LAPAN-A1 the update frequency were mainly on an hourly basis, though there were some updates that takes longer than one day, depicted by the outliers. For LAPAN-A2, the update frequency were mainly on the daily basis, some oven take more than two days get a new update, depicted by the outliers. As for LAPAN-A3, the update frequency were more frequent in comparison to the two, the satellites mainly received hourly update for the state update, though there were some update that took longer but not longer than four days, depicted by the outliers.

Extraction of Orbital Elements for Each Satellites

As presented in the Table. 2.1 and Table. 2.2 from the previous chapter, the TLE data contains among other, the mean orbital element of the orbiting object. The author only needed the epochs and the orbital elements listed in them. The epochs were needed for the later references of the orbit propagation while the orbital elements were needed for evaluation value for the propagated state. The needed orbital elements were divided into two category, the first one is for the initial state for propagation that is the first observation in the TLE historical data. The latter one is the evaluation values for propagated values that is the rest of the TLE historical data. Since the TLE contains more than what was needed for this thesis, it meant that the author needed to separate or rather select the necessary ones and collected them, accordingly, to a "container" for which it would be much easier to use later on in the process of propagating the states. Here are the needed elements from the TLE,

Elements	Line	Columns
Epoch	1	19 - 32
Inclination	2	09 - 16
Right Ascension of Ascending Node	2	18 - 25
Eccentricity	2	27 - 33
Argument of Perigee	2	35 - 42
Mean Anomaly	2	44 - 51
Mean Motion	2	53 - 63

Table 3.5: Necessary elements to be extracted from TLE

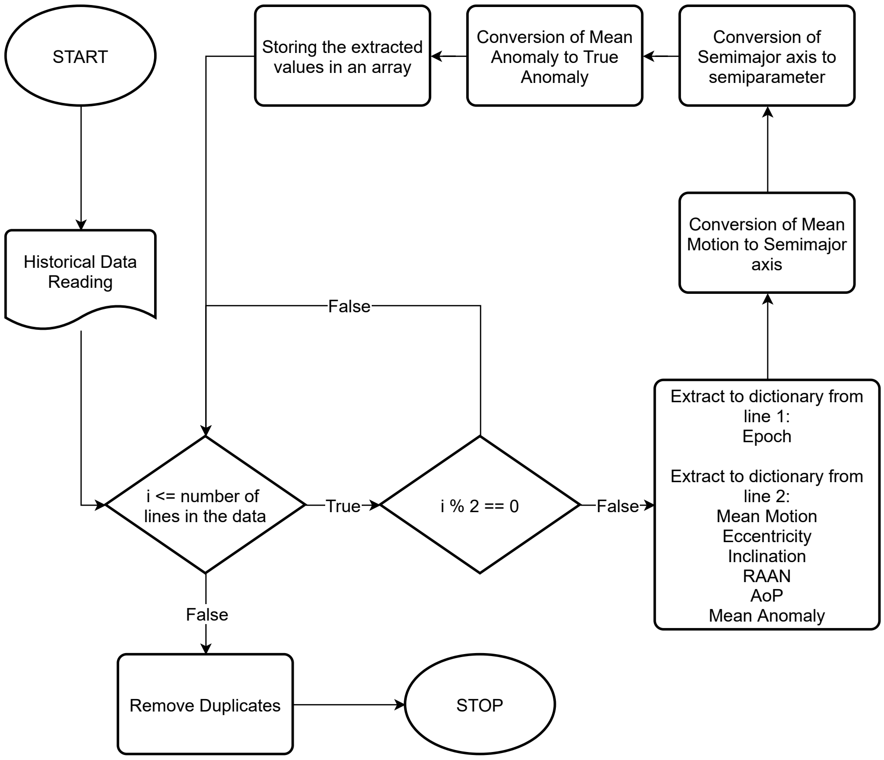


Figure 3.5: TLE data extractions

As mentioned before, there are no direct information regarding the semimajor axis, a and the true anomaly, ν within the TLE. Both of the non-specified elements were relatively straightforward to obtain or rather to convert from other parameters listed in the TLE. Firstly, to get the value of the semimajor axis, the value can be obtained using the Kepler's third law on the mean motion of the satellite,

$$n = \sqrt{\frac{\mu}{a^3}} \tag{3.1}$$

The mean motion of the satellite is expressed in revolution per day, thus the unit should be converted into a more suitable one. In this case radian per second,

$$n = \frac{n_{TLE} 2\pi}{24 * 3,600} \tag{3.2}$$

and when rearranged the conversion into the eq. 3.1, the equation becomes,

$$a = \sqrt[3]{\frac{\mu}{n^2}} \tag{3.3}$$

Now then is for the is true anomaly, ν. This element can be obtain by using the mean anomaly, M, where the two are related through what it is called the Eccentric Anomaly, E, for all elliptical orbit $e < 1.0$, through this equation:

$$M = E - e\sin(E) \tag{3.4}$$

By taking the E as a function of f(x) and solving it by implementing the Newton-Raphson iteration, the equation lead to:

$$E_{n+1} = E_n + \frac{M - E_n + e\sin(E_n)}{1 - e\cos(E_n)} \tag{3.5}$$

This Newton-Raphson iteration would go on until the successive values of eccentric anomaly are close enough (usually within $10e^{-8}$), $|E_{n+1} - E_n| < tolerance$. The calculated eccentric anomaly is then can be directly be put into the following equation to calculate for true anomaly:

$$\tan(\nu) = \frac{\sin(\nu)}{\cos(\nu)} \tag{3.6}$$

where,

$$\sin(\nu) = \frac{\sin(E)\sqrt{1-e^2}}{1 - e\cos(E)} \quad or \quad \cos(\nu) = \frac{\cos(E) - e}{1 - e\cos(E)} \tag{3.7}$$

(McClain, 2013)

To be noted that this thesis solely be relied on this conversion due to the fact that the satellites operated by LAPAN that are the interest of this thesis orbital paths are elliptical orbits with $e < 1.0$. For a full extraction and conversion, they are all listed in the codes in the appendix.

As mentioned before, the author needed to reevaluate the raw data before any further processes, one of the evaluation was that there might be duplication of updates within the TLE historical data. This happened with all of the historical TLE data that the author had requested. As an example here is one of several update duplicates of TLE data within the LAPAN-A2 historical TLE data.

```
1 40931U 15052B   20011.77319414  .00000873  00000-0  32448-4 0  9993
2 40931   5.9945  19.8389 0013427 198.6363 161.3444 14.76624730231911
```

```
1 40931U 15052B   20011.77319414  .00000873  00000-0  32448-4 0  9993
2 40931   5.9945  19.8389 0013427 198.6363 161.3444 14.76624730231911
```

The epoch of the two data are the same and the other parameters or values are also the same for the two updates, even though the other parameters has a chance to be different. This duplicate of epoch would give some trouble later on with the simulation, especially when defining the time difference of each epoch. To compensate this duplication phenomenon the author decided to remove one of the duplicate, the latter one to be precise. The duplicate removal process is fairly easy with the help of *pandas* package, The author just need to use the following function after extracting all of the necessary parameters in the TLE into a DataFrame "container", in this case the container is called *coes_df*.

```
coes_df.drop_duplicates(subset=["Epoch"], inplace=True)
```

3.4 Mathematical Model

The mathematical model were used in a way to achieve the suitable and relatively correct results of data extractions, state conversions and orbital propagations. The author, with the help of Python programming and also the literature given in the previous chapter, came up with several mathematical models that were needed for this thesis. Though, one mathematical model has been stated outside of this section, that is the extraction of the TLE datasets, the following are the necessary ones to be used along side with the simulation.

3.4.1 Conversion between Orbital Elements and State Vectors

After the extraction of the necessary orbital elements and the conversion of the mean motion to the semimajor axis, a, and also mean anomaly, M to the true anomaly, ν, the author then have the ability to convert the classical orbital elements into the state vector as well as the other way around. To be noted that these conversion are all based on the Fundamental of Astrodynamics and Application (McClain, 2013). All of the steps are shown in the subsection below.

Orbital Elements into State Vector

In calculating or converting the orbital elements into the state vector, the first thing to keep in mind is that this conversion is to determine the state vector in the perifocal, PQW, coordinate system and after that the state vector are rotated into the geocentric, IJK, coordinate system. Now here are the steps to convert the orbital element into the state vector:

1. The first step is to convert all of the orbital elements angle, from degree to radian, this due to NumPy computational processing is in radian.

2. The next step is to convert the semimajor axis into semiparameter, since the semiparameter is defined for all types of orbit. That said, the semiparameter equation is as follows,

$$p = a(1 - e^2) \tag{3.8}$$

3. Then, there is a need for a new coordinate system. The coordinate system in mind is the perifocal coordinate system, PQW, for then it is to be rotated to the geocentric equatorial system, IJK. perifocal coordinate system is where the orbital plane is laid flat in the coordinate system and is centered in the orbit focus. Now then, begin with finding the position vector of the satellite in the perifocal coordinate system:

$$\vec{r}_{PQW} = \begin{bmatrix} \frac{p\cos(\nu)}{1+e\cos(\nu)} \\ \frac{p\sin(\nu)}{1+e\cos(\nu)} \\ 0 \end{bmatrix} \tag{3.9}$$

Then the velocity of the perifocal coordinate system is defined as,

$$\vec{v}_{PQW} = \begin{bmatrix} -\sqrt{\frac{\mu}{p}}\sin(\nu) \\ \sqrt{\frac{\mu}{p}}(e + \cos(\nu)) \\ 0 \end{bmatrix} \tag{3.10}$$

4. The last part is to rotate the state vector that are already define in perifocal coordinate system to the equatorial geocentric system. the position vector is

defined as,

$$\vec{r}_{IJK} = [ROT3(-\Omega)][ROT1(-i)][ROT3(-\omega)]\vec{r}_{PQW} \tag{3.11}$$

and the velocity vector is defined as,

$$\vec{v}_{IJK} = [ROT3(-\Omega)][ROT1(-i)][ROT3(-\omega)]\vec{v}_{PQW} \tag{3.12}$$

5. The last part is to separate between the position vectors and the velocity vectors of the rotated matrices into a container, in this case the container is in a form *numpy* array.

State Vector into Orbital Element

The conversion of the state vector to the orbital elements is also a common and fundamental routine. This conversion would come in handy later when the propagation have been made. But before getting ahead of ourself, let us see how this conversion is done:

1. Firstly, a definition of the unit vector $I, J, andK$, for which they all contain only the value of *one*.

2. Then, the state vector are put into two different container or array from NumPy, and also the magnitude of both of these vectors are calculated.

3. Several intermediate vectors are needed to be defined before actually converting the state vector to the classical orbital elements. The first one is to define the angular momentum, $\vec{h}$, and its magnitude. The angular momentum acquired from the cross product of the position vector and the velocity vector,
$$\vec{h} = \vec{r} \times \vec{v} \tag{3.13}$$

4. The node vector, $\vec{n}$, and its magnitude are next to be defined. The node vector is resulted from the cross product of the K vector defined earlier and

the angular momentum vector,

$$\vec{n} = \hat{K} \times \vec{h} \tag{3.14}$$

5. Another one to be defined is the specific mechanical energy, ξ, for which it is used in the calculation for the semimajor axis,

$$\xi = \frac{v^2}{2} - \frac{\mu}{r} \tag{3.15}$$

6. From here on out the rest of the orbital parameters can then be calculated. Starting with the eccentricity that is is acquired from the form of its vector, and the magnitude of the vector is the eccentricity value of the orbit. the equation make use of the position vector and the velocity vector,

$$\vec{e} = \frac{(\vec{v} - \frac{\mu}{r})\vec{r} - (\vec{r} \cdot \vec{v})\vec{v}}{\mu} \tag{3.16}$$

and the value of the eccentricity is,

$$e = |\vec{e}| \tag{3.17}$$

7. Then the semimajor axis and the semiparameter can be obtain according to the value of the eccentricity that is. For non-parabolic orbit, the semimajor axis and the semiparameter are defined as,

$$a = -\frac{\mu}{2\xi} \qquad p = a(1 - e^2) \tag{3.18}$$

For parabolic orbit, the semimajor axis is defined as infinity (∞) and the semiparameter is defined as,

$$p = \frac{h^2}{\mu} \tag{3.19}$$

8. From here on out, the res of the angles of the classical orbital element can be calculated. The first one is the inclination, i,

$$\cos(i) = \frac{\hat{K} \cdot \vec{h}}{|\hat{K}||\vec{h}|} \tag{3.20}$$

9. Next is the right ascension of the ascending node, Ω,

$$\cos(\Omega) = \frac{\hat{I} \cdot \vec{n}}{|\hat{I}||\vec{n}|} \qquad IF(n_J < 0)\ \ THEN\ \ \Omega = 360^\circ - \Omega \tag{3.21}$$

10. Next is the argument of perigee ω,

$$\cos(\omega) = \frac{\vec{n} \cdot \vec{e}}{|\vec{n}||\vec{e}|} \qquad IF(e_K < 0)\ \ THEN\ \ \omega = 360^\circ - \omega \tag{3.22}$$

11. The last one is the true anomaly, ν,

$$\cos(\nu) = \frac{\vec{e} \cdot \vec{r}}{|\vec{e}||\vec{r}|} \qquad IF(\vec{r} \cdot \vec{v} < 0)\ \ THEN\ \ \nu = 360^\circ - \nu \tag{3.23}$$

12. The last process of this conversion is to put them all together inside of a *numpy* array container or a list with the according series semiparameter, eccentricity, inclination, RAAN, argument of perigee, and true anomaly.

3.4.2 Satellite State Propagation

As previously mentioned, the TLE datasets or historical data gives a somewhat not-so frequent updates on the orbital element of the satellites. With that, the extraction of the historical data would give a controlled value of the orbital elements, or to put it other words it would give different time steps in between epochs. Nonetheless, the propagation model and its mathematical form should be robust before even going to the point where the author needed to match the time steps nor implementing them in the simulations. With that, in this section the author set the function definition for the propagator that was used in this thesis, namely Two-Body, Two-Body + J_2, and SGP4, although the SGP4 would rely on the an externally existing package.

For the Two-Body and the Two-Body + J_2 propagation models, the author use the SciPy's built-in ODE solver called the *solve_ivp*. It has the option to use various algorithm, namely $RK45$ or the Runge-Kutta method fourth order, $RK23$ or the Runge-Kutta method third order, $DOP853$ or the Runge-Kutta method eight order, *Radau* or the Runge-Kutta method fifth order, BDF or the multi-step variable-order, $LSODA$ or the Adams/BDF method with automatic stiffness detection and switch (community, 2020).

Two-Body Propagation

The two-body propagation is an idealized propagation model for which it only considers the attraction forces from the two interacting bodies. Thus, what the author expected to be the result of this method of propagation is that all of the orbital elements are not changing its value with time. But, that would be just an initial assumption and without the result it would mean nothing. The propagation itself is done by solving for the two-body equation, that is,

$$\ddot{\vec{r}} = -\frac{Gm_{\oplus}}{r^2}\frac{\vec{r}}{r} \tag{3.24}$$

The solve_ivp solver needs a time frame and initial state input as well as a "callable function" as the base arguments. Since the input would be in a form of position vector and velocity vector, the two-body problem needs to be broken down into its components. Here are the steps taken for the "callable function" definition:

1. Firstly, define the function name and arguments for the two-body problem. The arguments are the time period and initial state of the satellite. The reason that the arguments are only those two is because the "callable function" of the *solve_ivp* only need those two arguments.

2. The state is the position and the velocity vector. Both of the vectors are separated into its components,

   $$x, y, z$$

   and

   $$\dot{x}, \dot{y}, \dot{z}$$

3. The there is the need to find the magnitude of the position vector,

$$r = \sqrt{x^2 + y^2 + z^2} \tag{3.25}$$

in which the magnitude is used in the two-body equation.

4. Next is to find the derivative of each velocity vector, as shown here,

$$\ddot{x} = -\frac{\mu}{r^3}x \tag{3.26}$$

$$\ddot{y} = -\frac{\mu}{r^3}y \tag{3.27}$$

$$\ddot{z} = -\frac{\mu}{r^3}z \tag{3.28}$$

Now the acceleration above then assigned to a variable with the respected value $\ddot{x}, \ddot{y}, \ddot{z}$

5. rearranging the results into a sorted *numpy* array as the following $\dot{x}, \dot{y}, \dot{z}$ and $\ddot{x}, \ddot{y}, \ddot{z}$ for the return value.

Now that the function definition has been defined. The solve_ivp then can be used, and the use of the solve_ivp function can be seen in the appendix. But for the sake of this subsection, here is the look of it,

```
scipy.integrate.solve_ivp(func, t_s, y0, method="RK45", t_eval=None)
```

From the solver above and as stated before, the author need to specify the time frame and also the initial state of the data. In addition, all of the steps are being stated inside of a function statement for the first argument of the solver. The t_eval is essentially the only time, within the time frame of course, that is being evaluated, or to be blunt the ones that is going to be the output of the solver.

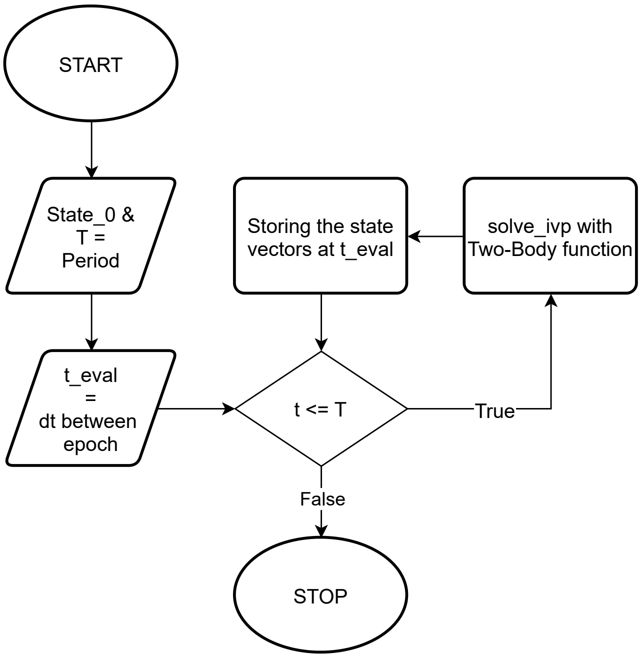


FIGURE 3.6: Two-Body Propagation Flowchart

Two-Body Propagation + J_2 Propagation

The Two-Body Propagation with J_2 perturbation or any other other perturbation is only an expansion on the two-body problem. As mentioned, this propagation method adds in the consideration of the Earth oblateness in shape into the equation. Thus, there should be some changes within the propagated orbital elements, mainly the RAAN and the AoP. Again, this is only an assumption and only gives an initial idea on how would the propagation behave. Now, recalling from the previous chapter regarding the orbit perturbation, the two-body, which includes perturbation, equation becomes,

$$\ddot{\vec{r}} = -\frac{\mu}{r^2}\frac{\vec{r}}{r} + \vec{p} \tag{3.29}$$

where $\vec{p}$ is an acceleration due to perturbation. in this subsection case the perturbation is the J_2 perturbation, and the equation can be rewritten into,

$$a_I = -\frac{3}{2}\frac{J_2 \mu R_\oplus^2 \vec{r_x}}{r^5}(1 - \frac{5\vec{r_z}^2}{r^2}) \tag{3.30}$$

$$a_J = -\frac{3}{2}\frac{J_2 \mu R_{\oplus}^2 \vec{r_y}}{r^5}(1 - \frac{5\vec{r}_z^2}{r^2}) \tag{3.31}$$

$$a_K = -\frac{3}{2}\frac{J_2 \mu R_{\oplus}^2 \vec{r_z}}{r^5}(3 - \frac{5\vec{r}_z^2}{r^2}) \tag{3.32}$$

Now that the acceleration of each axis has been stated, the equation for the two-body + J_2 perturbation can be rewritten as,

$$\ddot{x} = -\frac{\mu}{r^3}x + a_I \tag{3.33}$$

$$\ddot{y} = -\frac{\mu}{r^3}y + a_J \tag{3.34}$$

$$\ddot{z} = -\frac{\mu}{r^3}z + a_K \tag{3.35}$$

Now, the same step of defining the function applies with this two-body + J_2 as the previous two-body one. the only difference to it is in the fourth (4^{th}) steps. rather than using the stated equation in the step, the equations are replaced with the eq. 3.33, eq. 3.34, and eq. 3.35. Same as the previous section, it is to be noted that the initial state vector is the first epoch of the requested TLE ephemerides, and all of the conversion of the orbital elements are the same way as the previous sections. The time frame of which the satellites' states are being propagated is ten (10) days from the epoch.

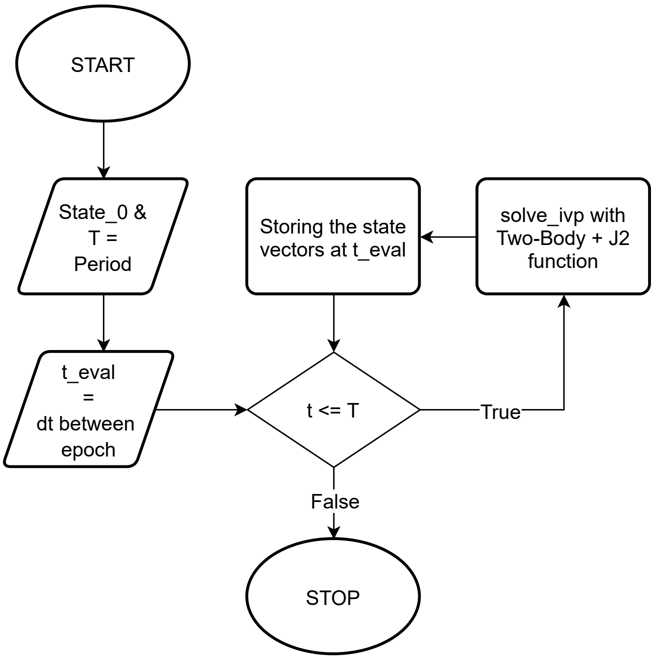


FIGURE 3.7: Two-Body + J2 Propagation Flowchart

SGP4 Propagation

The SGP4 is a propagator that is solely, or rather heavily, relying on the Two-Line Element (TLE) dataset. The propagator itself considers the secular and short- and long-periodic variations due to Earth oblateness, solar and lunar gravitational effects, gravitational resonance effects and orbital decay (e.g., atmospheric drag). That being said, in order to reduce the error of creating the code, the author relied on the an external package called the *cysgp4* as previously mentioned before. This subsection would give a glimpse of how the author used the *cysgp4* package to propagate the satellites' state. The steps of which the author took was different to those the Two-Body propagation and The Two-Body + J_2 propagation. There are two different way on using the *cysgp4*, one is to use the function *Satellite* and the other one is called *propagate_many*. The apparent difference is that the latter will utilize all CPU cores that are available on the system, but for the convenience and also the straightforwardness of use, the author decided to work with the *propagate_many*. The steps on how to use the function is as the following,

1. First of all the author need to fetch all of the lines in the TLE historical data.

In this case the author need to fetch the satellites raw data or still in the *.txt* format.

2. Then the author create and array of list, separating the lines of the TLE historical data into individual $cysgp4.cysgp4.PyTle$ type.

3. The author is then define an observer longitude, latitude and altitude in Earth Centered Inertia Frame (ECI). For this case the author uses his own location. Although it is not mandatory for the author to include the observer if the author only need the ECI position and velocity of the satellite.

4. The author then extracted the time of updates in the second steps and put it into an array in a form of *Modified Julian Date*.

5. The author then set a propagation period and also the time step of the need to be evaluated value in the propagation, in which already settled by the previous step .

6. With the propagation period, observer, and TLE data set, the author then just have to call the *propagate_many* with each of those key arguments to use the function.

```
propagate_many(T, TLE, obs, do_geo=False, do_topo=False)
```

7. The propagation would give the satellite's positions and velocities. Due to that reason that is also the reason why the author want to separate each satellite into an individual assigned value or simply for ease of sorting.

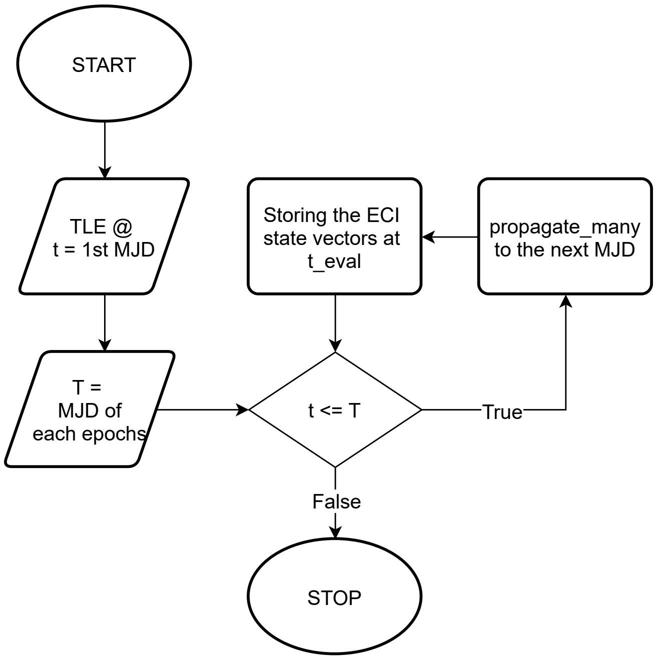


FIGURE 3.8: Two-Body + J2 Propagation Flowchart

3.5 Simulations Regimes

The simulations would only concern about the propagation scenarios of each propagation models. The help of using *solve_ivp* gave the author the ability to propagate the states easier, though for the SGP4 the author uses the help of an already existing package provided for the public, *cysgp4*. All of the satellites are being treated equally in terms of propagation parameters. That said, here are the parameters that are implemented in each of the scenarios,

1. All of the propagation period of each model are from the point when they were launched to the 7 December 2020;

2. The initial value of the state vectors are the value of which in the first epoch of their respective TLE historical data;

3. The only propagated values that are taken into an account or evaluated are that of which at the same time as each epoch in TLE historical data;

4. The propagated values then are contained in a container separating all of the values, namely x, y, z and $\dot{x}$, $\dot{y}$, $\dot{z}$; and

5. For the convenience of the error analysis, the container would also be provided with each respective errors in position and velocity vectors.

3.5.1 Selecting the $State_0$ and t_0

For the selection of the initial state is relatively straightforward, the author just need to select the first states or the first state according to the epoch in the TLE data of each of the satellites' converted TLEs. These first states are the only ones that are needed for the simulations of propagations. The reason why is that so the author can easily analyze how the propagation models behave over time with respect to the control data / TLE data. The following table gives the initial state and the initial time stamp of each of the satellite,

	LAPAN-A1	LAPAN-A2	LAPAN-A3
Epoch	2007-01-11	2015-09-29	2016-06-23
	14:35:14.434944	18:52:16.320864	10:15:05.956128
X (km)	2193.69	112.74	2894.81
Y (km)	456.87	4208.07	-1393.59
Z (km)	-6649.24	1.75	6174.42
V_X ($\frac{km}{s}$)	-0.888	-1.780	-4.970
V_Y ($\frac{km}{s}$)	4.293	-5.124	5.197
V_Z ($\frac{km}{s}$)	-0.907	-2.031e-03	4.117

Table 3.6: Initial timestamps and states of each satellites

3.5.2 Defining the Time Frame and Time Steps

As previously mentioned, all of the time steps between updates of each satellites are different due to one reason or the other. But for defining the time frame of the propagation, the time between first epoch and the last epoch is the most straightforward answer to how the author define the time frame. The time difference on the other hand, got a little bit more treatment. To give a little perspective, the Fig. 3.9 gives an idea on how the steps are looking like.

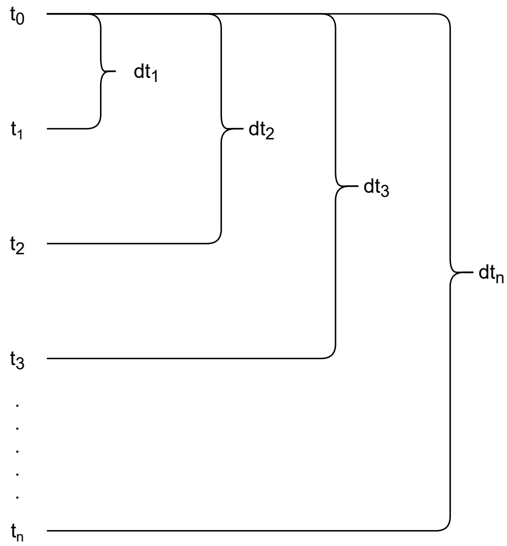


FIGURE 3.9: Time differences between $epoch_n$ to the first epoch

From the figure above, the t_0 until the t_n are the epoch updates of the TLE data. The dt_1 until dt_n are the time difference between between the time of updates and the first epoch. This method is used to be implemented as the *t_eval* in the *solve_ivp* function. time differences are then gathered inside of a list. The apparent problem comes if the duplicates of epoch are not removed. For one and another reason, if there are duplicates of values in *t_eval* then the solver would not work. That is why the removal of duplicates are necessary in the first place. This method of defining the *t_eval* rather than the time difference between each updates is still fairly acceptable and fairly robust and easily understood.

3.6 Error Analysis

The comparison of data was be done in several ways. The first one was a visual representative of each propagation methods and the control value. This, however, only gives little information on what was going on, in this case how does the error of the propagation visually looked like. The second method of comparison was with using tables containing all the data from propagation. This gives the sense of difference since the author know exactly the value of error in the propagation. In addition to that, the author also created a graph visualizing the errors with respect to the propagation time.

3.6.1 Controlled Values

The control value are needed to be a referenced value for the propagation for an error analysis of each propagation and each satellites. The control values take on two form, the first one is the state vectors and the orbital elements from each satellites' TLE historical data. The author just need to take the extracted orbital elements data and convert all of them into their respective state vectors and contain them in a *pandas* DataFrame container for easy access. Table 3.7 gives a preview of the controlled values from LAPAN-A2 satellites.

All of these orbital elements and the state vectors as well as the epoch are then treated as the control values, as stated before. These control values are going to be used as the timestamps and time steps reference for the simulations which then lead to the analysis of errors for each propagation method towards each of the state vectors and orbital elements values. Just to be sure that the readers are all in the same page, all of the state vectors are in reference to the Earth Centered Inertia (ECI), just like what have been explained in Chapter 2.

3.6.2 Propagation Error

The propagation error is inevitable, just like any other measurement. With the models and the scenarios are set, the author could analyze the error of the propagation error with the corresponding *true* value of the orbit at different epoch of the TLE. The outline of how the author analyze the propagation error is as the following:

1. the saved propagation state vectors are then converted into orbital elements using the already stated conversion regimes,
2. the state vectors and the orbital elements then saved into a *numpy* array container,

Epoch	p_i km	e_i $^\circ$	i_i $^\circ$	RAAN_i $^\circ$	AoP_i $^\circ$	nu_i $^\circ$	rx_i km	ry_i km	rz_i km	vx_i $\frac{km}{s}$	vy_i $\frac{km}{s}$	vz_i $\frac{km}{s}$
2020-01-03 14:06:03.209760	7018.1311	0.0014	5.9928	85.5229	76.7647	55.5764	-5508.13	-1345.93	-579.21	1.784	-6.163	-0.315
2020-01-09 17:36:44.749728	7018.1323	0.0013	5.9917	41.4791	160.0202	320.4025	-4306.58	78.22	4773.03	-8.494	-0.503	4.193
2020-01-12 18:32:49.780896	7018.123	0.0013	5.9944	19.842	198.746	117.3274	541.17	7008.36	3753.03	-0.362	-3.08	-6.208
2020-01-12 18:33:23.973696	7018.1229	0.0013	5.9945	19.8389	198.6363	244.2146	3707.26	382.36	389.33	7.357	-8.124	0.738
2020-01-13 17:15:02.756160	7017.6199	0.0015	6.0091	12.9182	247.5866	102.5649	-4806.36	-125.7	-3778.87	1.517	-0.366	8.576
2020-01-14 17:33:56.489184	7018.1244	0.0013	6.0014	5.8582	234.1218	5.911	-427.38	-309.26	5889.18	0.723	7.009	-1.138
2020-01-15 03:17:30.493536	7018.1203	0.0013	6.0048	2.9632	240.078	25.2692	-6642.28	6038.29	609.36	-0.929	-2.68	0.801
2020-01-15 11:23:48.517440	7018.1218	0.0014	6.0004	0.5865	245.6705	62.143	2153.2	626.81	-5117.47	-8.633	0.721	1.023
2020-01-15 17:52:51.284928	7018.1238	0.0014	5.9997	358.6415	248.9552	234.9442	631.08	-2866.33	-3255.31	-0.372	8.592	7.467
2020-01-16 13:19:59.251296	7018.1251	0.0014	5.9989	352.8787	259.9842	321.7374	-1897.08	-6364.38	-354.78	7.463	-0.929	-0.37
2020-01-19 17:31:17.113152	7018.1238	0.0014	5.9994	330.2874	304.6802	283.0284	-5774.36	-634.83	-5767.3	-0.826	0.891	8.34
2020-01-23 05:48:54.670752	7018.124	0.0014	5.9978	305.2532	354.0615	342.4306	-322.18	5555.28	-3955.48	0.72	0.827	-2.282
2020-01-25 16:10:20.307360	7018.1212	0.0014	5.9989	287.9175	28.0383	307.7824	6402.06	-2471.13	-487.59	-7.084	7.498	0.851
2020-01-26 11:37:28.181280	7018.1181	0.0013	6.0013	282.1597	35.1657	257.682	-2840.69	-318.04	3464.0	-4.982	-0.151	2.012
2020-01-27 11:56:23.648352	7018.1244	0.0012	5.996	274.8066	49.2328	180.9504	-508.76	-208.66	-4982.81	-0.838	-1.601	7.267
2020-01-27 18:25:24.072672	7018.1254	0.0012	5.9962	272.8828	53.5953	282.6406	2494.5	-6968.23	-474.45	4.314	8.543	-0.273
2020-01-30 14:30:23.414688	7018.1175	0.0012	5.9973	252.7102	96.1773	4.8826	5522.19	-730.87	-6658.58	-6.169	-0.433	8.34
2020-01-30 15:46:16.992768	7018.118	0.0012	5.9976	252.328	96.9155	111.5048	456.53	6084.9	-2142.87	0.307	-7.368	-2.282
2020-02-02 04:06:17.357472	7018.1357	0.0013	5.9965	234.3327	130.2605	209.5117	-1529.66	-180.28	-542.44	-7.097	-1.408	0.851
2020-02-02 04:06:17.543232	7018.1275	0.0013	5.9946	234.3426	134.2153	343.7525	6814.86	-52.03	1898.15	5.0	-0.693	2.012
2020-02-02 04:06:18.159264	7018.1271	0.0013	5.9945	234.3432	134.3905	335.8616	727.65	4356.18	-5762.73	-0.699	-7.197	7.267
2020-02-02 04:06:18.160128	7018.1271	0.0013	5.9945	234.3432	134.3905	335.8616	-5925.38	-5466.16	-427.13	-4.292	-4.892	-0.273

TABLE 3.7: LAPAN-A2's orbital element according to TLE historical data

3. the propagated values within the container is then being subtracted with the values of that in the TLE historical data, and the results are in absolute values,

4. the error values of the propagation with respect to the TLE historical data then contained in a *pandas* DataFrame container, and

5. the propagation error then serves as the base for the regression analysis, linear to be precise.

3.6.3 Regression Analysis

Through the collected propagation error then the author uses regression analysis, linear regression in particular, to observe how the errors of propagation, in terms of distance, correlates and behave over the period of propagation. As previously mentioned, regression analysis refers to an analysis of an independent variable in which it has an effect towards another variable or called a dependent variable. One regression to be pointed out for this thesis is the linear regression due to the fact that it is the simplest way of regression analysis. To be noted there are two relationships that are describes in the linear regression, the first one to be positive relationship and the other one is negative relationship. The first one means that for the increasing value of the independent variable, the dependent variable increases also in value. The latter one means for the increasing value of the independent variable, the dependent variable will decrease in value. The relationship, when written mathematically, yields this equation,

$$Y = \beta_0 + \beta_1 X \tag{3.36}$$

where,

- Y i the expected value of y for a given value of x;
- β_0 is the y-intercept of the regression line; and
- β_1 is the slope of the line.

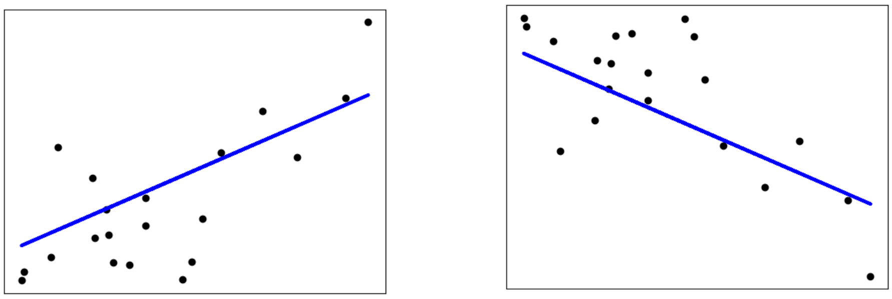


FIGURE 3.10: Positive and negative relationship between the variables (left to right) (*Linear Models*, 2020)

In doing the regression the author uses the $Scikit - Learn$ module to easily analyze the error in the propagation. The following are the steps that the author took in doing the analysis,

1. the author firstly need to call the error container, stated in the previous subsection,
2. then the author selects the "Epoch" column as the independent axis or the x axis, and selects the state vectors and orbital elements as the other values for dependent variables, y
3. the dependent variables in this case are processed separately from each other, this is to see how each parameters behave over time,
4. then the author uses the,

   ```
   LinearRegression().fit(x, y)
   ```

   for each y values, to use the linear regression analysis provided by the module,
5. then the author uses the *.coef_*, *.intercept_*, and *.score(x,y)* to find the regression coefficient, the intercept value, and the R^2 (for the correlation analysis), respectively

6. after that, the author put in regression coefficient, the intercept value, and the R^2 of each satellites for each propagation method in a *pandas* DataFrame container

7. for the graphing, the author just need to set the plot to be a scatter plot and also a line plot, where the scatter plot would give the values of error and the linear would give the regression line of the analysis.

From all of the stated procedural steps of error analysis, the author hoped that the propagated values to give some kind of respectable or acceptable values. With that the author also hoped that the behavior would not be too all over the place, meaning that the error data would not give too much of residuals so that the errors could correlates more to the propagation time. If, however, the data ever gave too much of ambiguity or some kind of non intuitive errors, then that does not mean that the research would considered to be useless.

3.6.4 Fourier Analysis

Also through the collected propagation error and also the result given by the regression analysis, the author did some more analysis on them. The ones that went through more analysis were the ones that were "seemingly" have periodical pattern in their errors of propagation. The periodical pattern would leave the linear analysis in a tight spot as to how the fit would behave overtime, but using the Fourier analysis to replicate the periodic patter of the error with some finite series, the author gained more satisfactory analysis on the errors. Again, here is the equation for the Fourier Series,

$$f(x) = a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega x) + \sum_{n=1}^{\infty} b_n \sin(n\omega x) \tag{3.37}$$

where,

- $f(x)$, is real-valued function;
- a_0, a_n and b_n, are the Fourier coefficients; and
- ω, is the frequency.

The use of the equation was made easy by the use of the *symfit* package. The following are the steps taken when the author was using the package,

1. First off, the author need to eyeball the "seemingly" periodical functions in the error graphs.

2. The author then need to create a variable in the name of the pointed out error data, and of course the time domain of propagation as the x value, later on.

3. Then The author set the x and y variables, errors and time accordingly, and also the frequency of the function.

4. To be noted that every data / graph has different frequency, and the author need only to eyeball the period to get the frequency.

5. Then the author need to declare the function that will be use to fit the data,

   ```
   model_dict = {y: fourier_series(x, f=w, n=3)}
   ```

   This gives the idea on what function is being used, and at what order, n the author used for the Fourier series.

6. Note again, that for every graphs / data, the order vary between small integers to bigger one, and of course the larger the oder the longer the program fit the data.

7. In this case, the author limit the order to 50, if the R^2 value of the fit is relatively still unsatisfactory.

8. Then the author need to input this function,

   ```
   fit = Fit(model_dict, x=xdata, y=ydata)
   ```

   and

   ```
   fit_result = fit.execute()
   ```

these two are needed to actually fit the data, and also to give the result of the fitting, where it will give, among other, the equation of the Finite Fourier series and the R^2 value of the data, only by printing the fit_result.

9. Lastly the author manually save the results and also graph the results for the use of the manuscript.

CHAPTER 4
RESULTS AND DISCUSSIONS

4.1 Processed TLE Data

4.1.1 Processed TLE Data result

This section contains all of the extracted and processed data from the TLE historical data of each satellite. Those data then would be the control data for the simulations as mentioned in Chapter 3. But before getting into the simulations regime and all that, let us just see how the processed and extracted parameters of the historical TLE datasets of each satellites looks like.

Orbital Elements

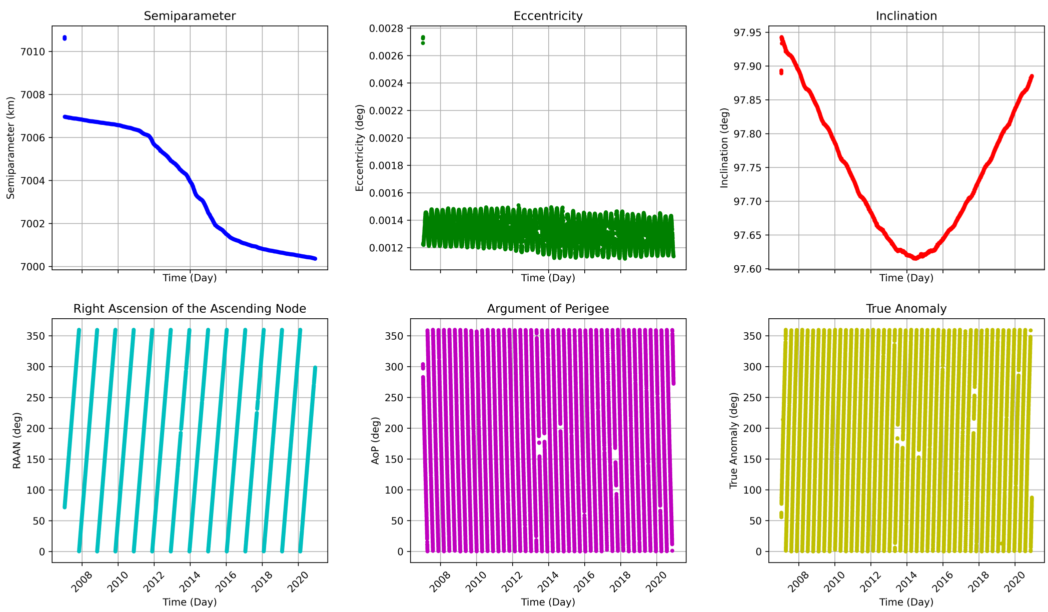


FIGURE 4.1: LAPAN-A1 extracted COEs

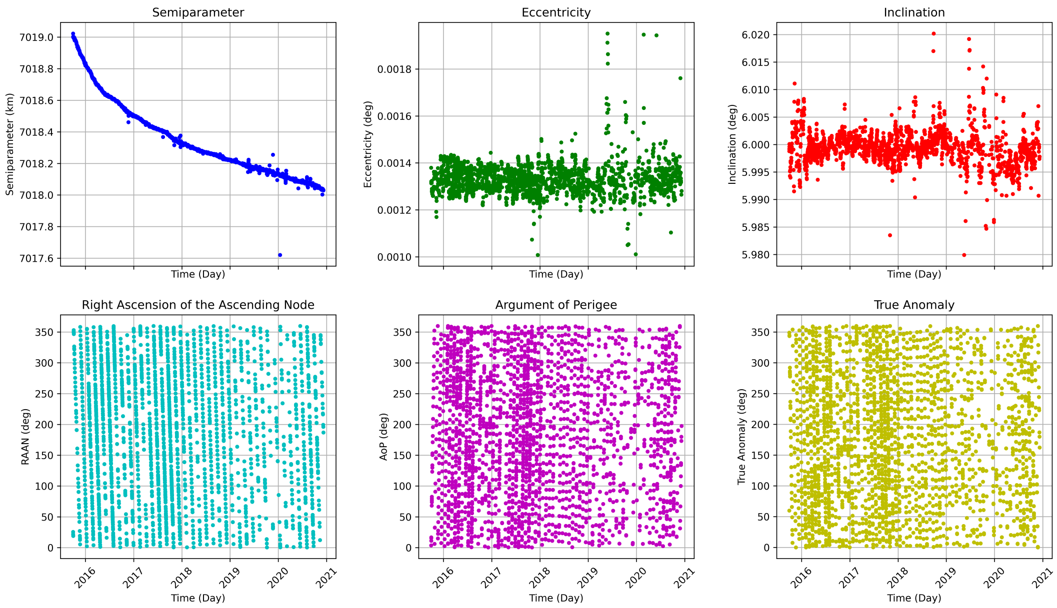


FIGURE 4.2: LAPAN-A2 extracted COEs

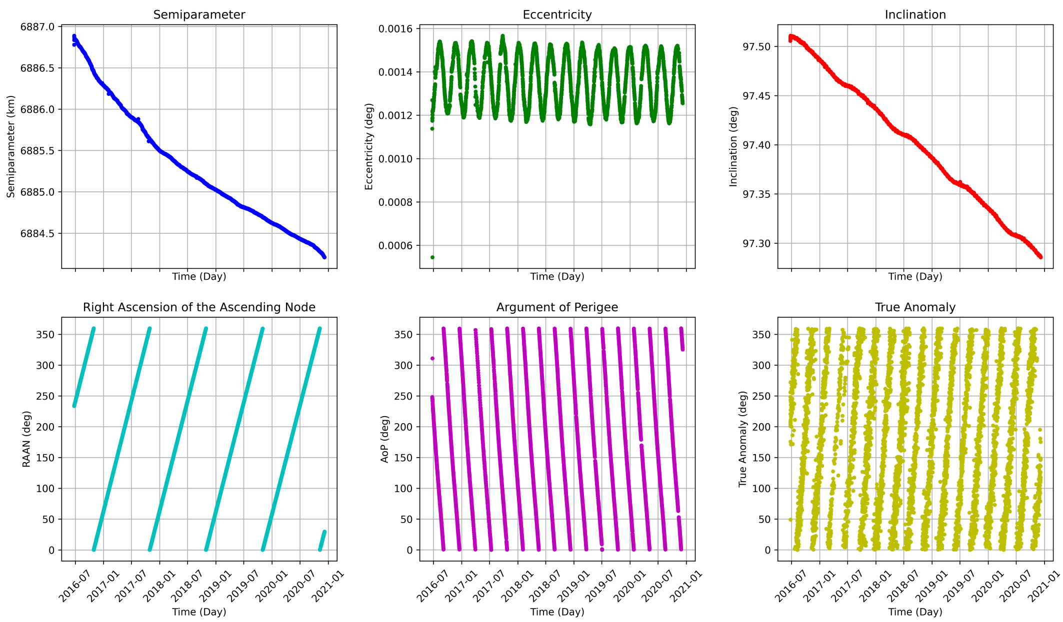
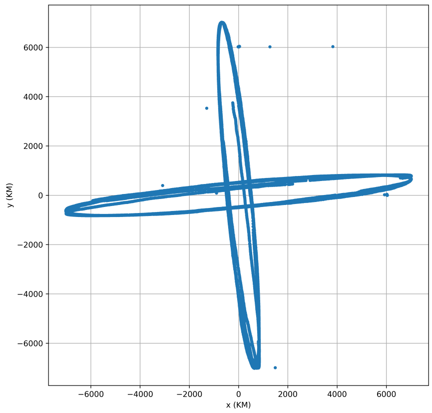
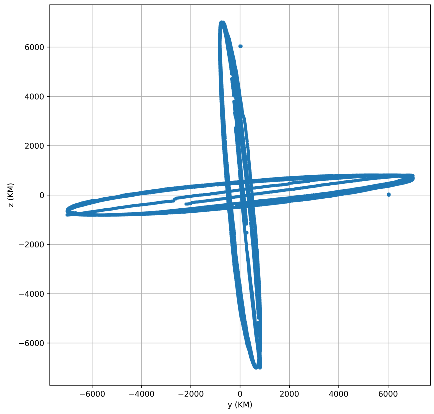


FIGURE 4.3: LAPAN-A3 extracted COEs

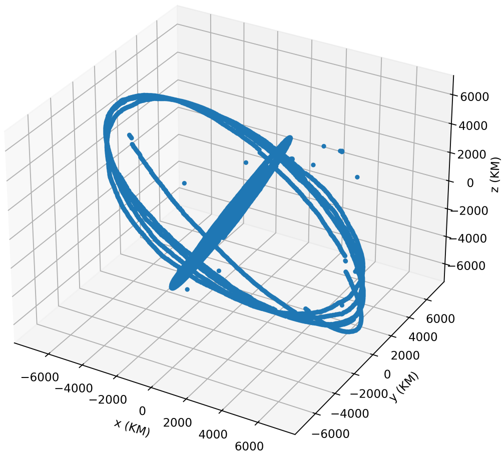
State Vector



(a) LAPAN-A1 X&Y-Plane Position according to TLE historical data

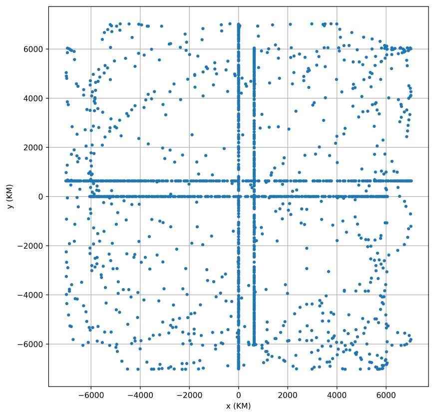


(b) LAPAN-A1 Y&Z-Plane Position according to TLE historical data

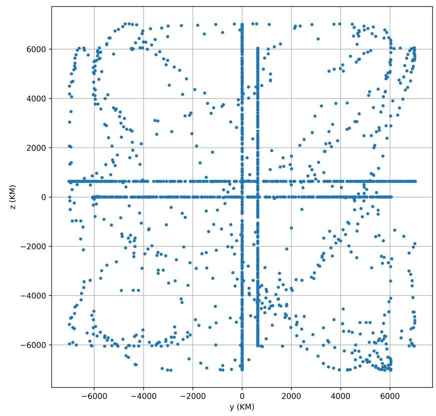


(c) LAPAN-A1 Spatial Position according to TLE historical data

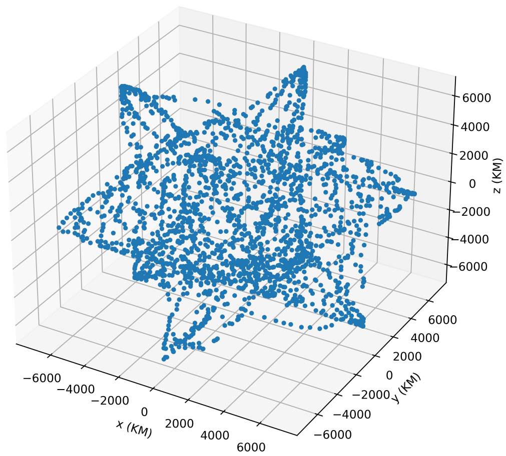
FIGURE 4.4: LAPAN-A1 state vector representation according TLE historical data



(a) LAPAN-A2 X&Y-Plane Position according to TLE historical data

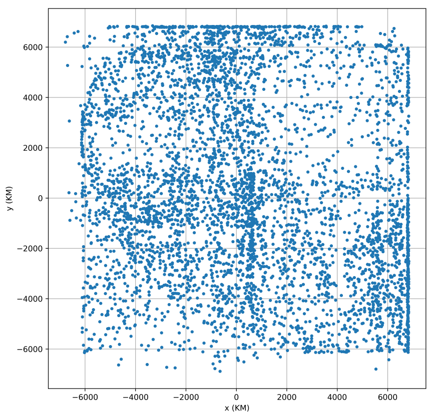


(b) LAPAN-A2 Y&Z-Plane Position according to TLE historical data

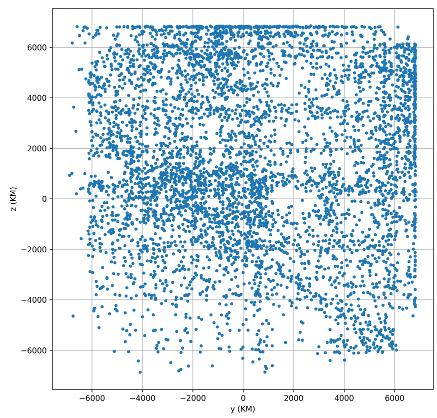


(c) LAPAN-A2 Spatial Position according to TLE historical data

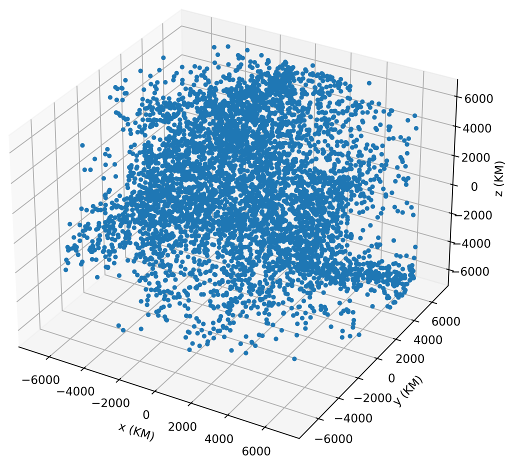
FIGURE 4.5: LAPAN-A2 state vector representation according TLE historical data



(a) LAPAN-A3 X&Y-Plane Position according to TLE historical data



(b) LAPAN-A3 Y&Z-Plane Position according to TLE historical data



(c) LAPAN-A3 Spatial Position according to TLE historical data

FIGURE 4.6: LAPAN-A3 state vector representation according to TLE historical data

4.1.2 TLE Historical Data Graphs Analysis

Orbital Elements Analysis

The six sets of colored graphs above– Fig. 4.1, Fig. 4.2, and Fig. 4.3, shows us how the satellite behave over time according to its orbital elements. some of the orbital elements behave with such patterns so that they seemed to give an intuitive sense towards their behavior overtime.

All three satellites' semiparameters are declining in value over the period from their launch until 7 December 2020. LAPAN-A1 experienced a decrease of about 7 km with a nice and steady slope from 2006 to 2012 followed by a little steep drop where eventually it tried to even out in 2016. LAPAN-A2 experienced a decrease of about 1 km with kind of a steep slope. LAPAN-A3 experienced a decrease of about 3 km at an almost steady rate. The eccentricity of all three satellite experienced fluctuations in values over time. LAPAN-A1 fluctuated between 0.00115° and 0.00150°, LAPAN-A2 fluctuated between 0.0011° and 0.0016° wit some outliers that might be due to the reason stated in the Chapter 2, and LAPAN-A3 fluctuated between 0.00115° and 0.00150°.

The other elements however, behave differently between the satellite. Starting with the inclination, LAPAN-A1's inclination kept decreases from 2008 to 2015 and then the value went back up until the end of the time period. LAPAN-A3's kept on decreasing over the time almost linearly. As for LAPAN-A2's, the inclination fluctuated between 5.990° and 6.015°. LAPAN-A1's and LAPAN-A3's RAAN behaves similarly with each other, the kept on increasing over the time, but LAPAN-A1's rate of change is faster. LAPAN-A2 on the other hand, decreased over time at a fast rate. LAPAN-A1's and LAPAN-A3's AoP kept on decreasing in value, where rate of change of LAPAN-A3 is faster. LAPAN-A2's on the other hand, increased in value over time. the rate of change in RAAN and AoP for LAPAN-A2 are considerably faster in comparison to the other two.

State Vector Analysis

The blue pale blue colored graphs– Fig. 4.4, Fig. 4.5, and Fig. 4.6, shows us the position vectors of each satellites based on the TLE converted orbital elements. They may not seem so intuitive nor make any much of a sense, but each dots represents the position of which the satellites were being observed at each epoch. If we look closely, LAPAN-A1 and LAPAN-A2 position representation might look fairly clear for us to see how they behave, in comparison to the LAPAN-A3.

As described in the Chapter 3, LAPAN-A1 categorized as polar orbit, meaning it moves from pole to pole. from the X&Y-plane position representation, the satellites orbits from polar to polar, as expected, though there are some observation where the satellites seemed to be orbiting in a equatorial orbit, though those are just points of observations. As well as the Y&Z-plane position representation. The spatial position representation shows the best view on how the satellite orbits the Earth, and yes as expected it is orbiting from polar to polar with some changes in the ascending nodes.

For LAPAN-A2, the X&Y-plane position representation and the Y&Z-plane position representation does not really give much of an idea on how the satellites' position according to both respective planes. The spatial position representation however, shows roughly that the satellite, as it was being observed, orbit the Earth in equatorial orbit.

The positions at which the LAPAN-A3 satellite was observed was kinda messy according to the graphs. The X&Y-plane position representation and the Y&Z-plane position representation, really does not give any sense on how the satellite is orbiting the Earth. In addition, the spatial position representation does not provide any intuitive sense as well.

The point that is needed to be noted is that, those points are only the points of observation, they do not have to make any sense, but they do need to represent the satellites relative motion at that particular point of time.

4.2 Propagation Results

4.2.1 Two-Body Propagation Results

In two-body propagation, it is should be noted that the only forces acting on the system are the gravitational attraction of the two body. That means, virtually, there will not be any deviation or change of the states or orbital elements over the propagation period using this technique. But, to be frank, there are actually changes though it is in the order where the changes are insignificant to be accounted for.

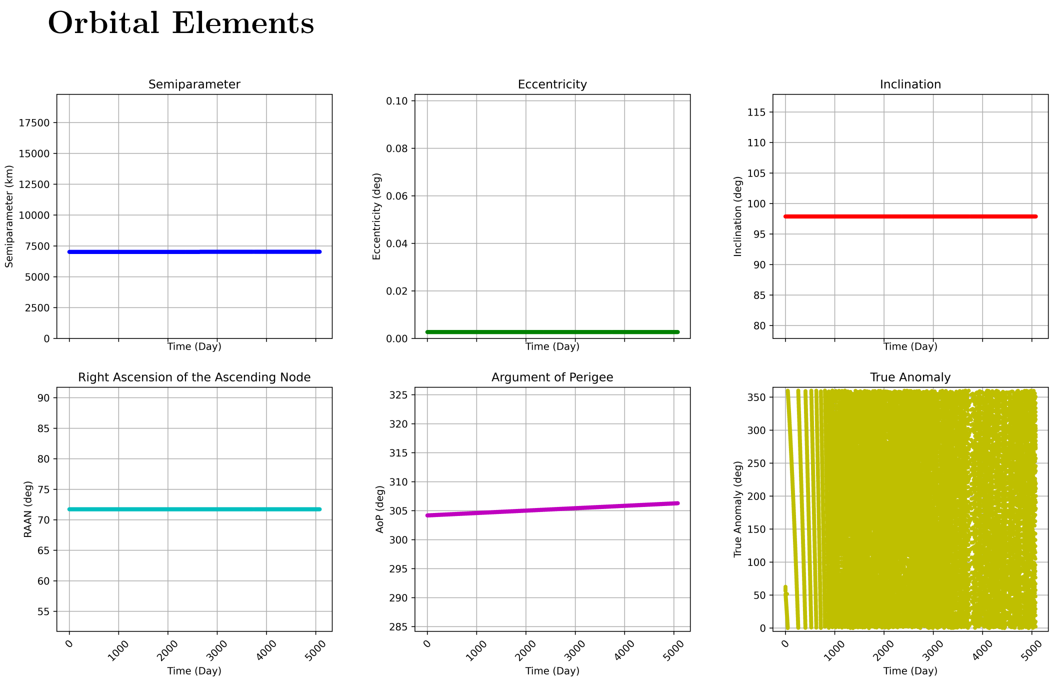


Figure 4.7: LAPAN-A1 Two-Body propagated COEs

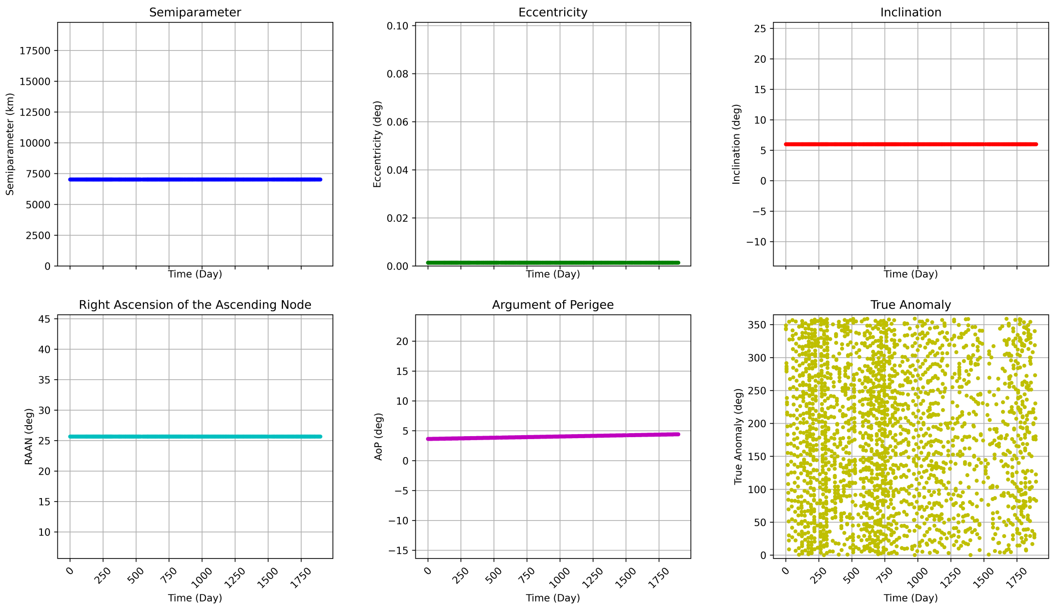


FIGURE 4.8: LAPAN-A2 Two-Body propagated COEs

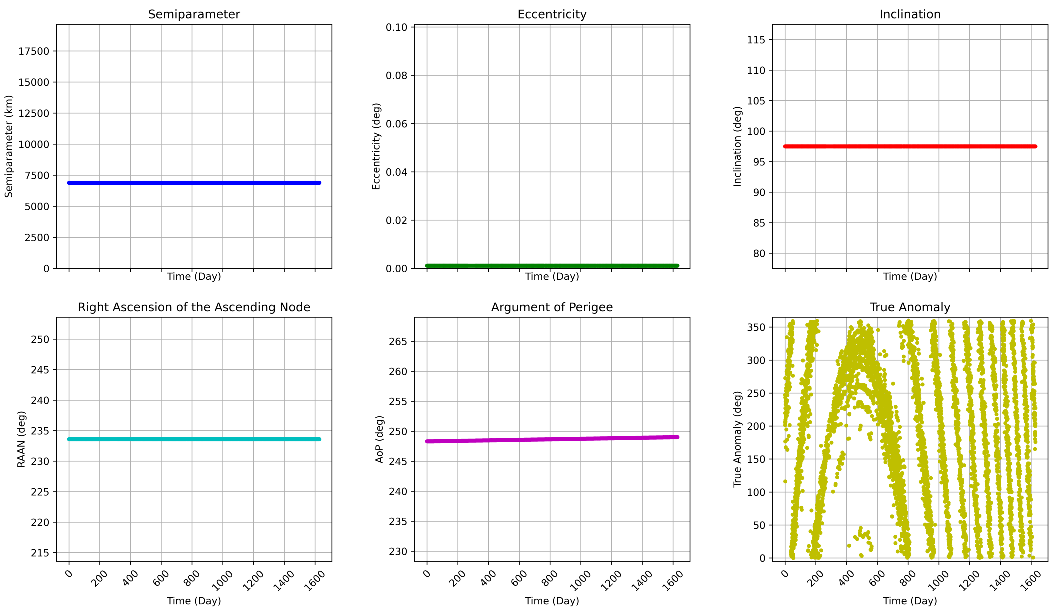
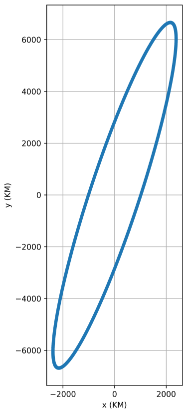
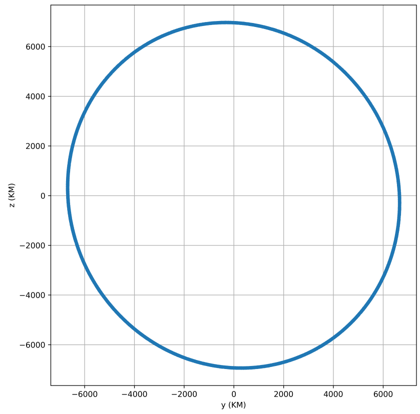


FIGURE 4.9: LAPAN-A3 Two-Body propagated COEs

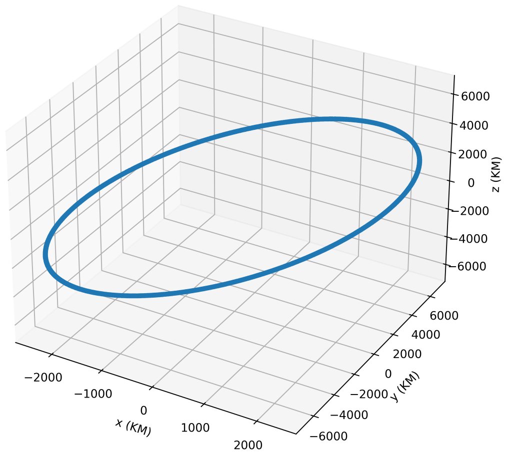
State Vector



(a) LAPAN-A1 X&Y-Plane Position according to Two-Body propagation

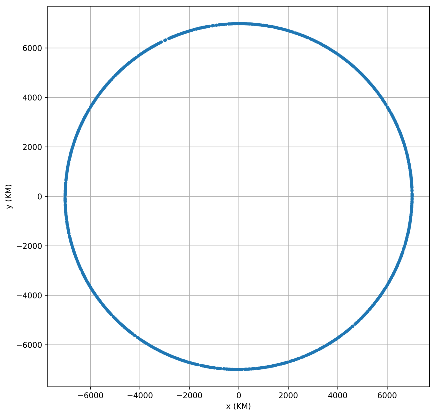


(b) LAPAN-A1 X&Z-Plane Position according to Two-Body propagation

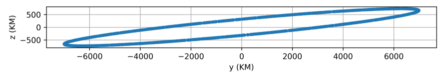


(c) LAPAN-A1 Spatial Position according to Two-Body propagation

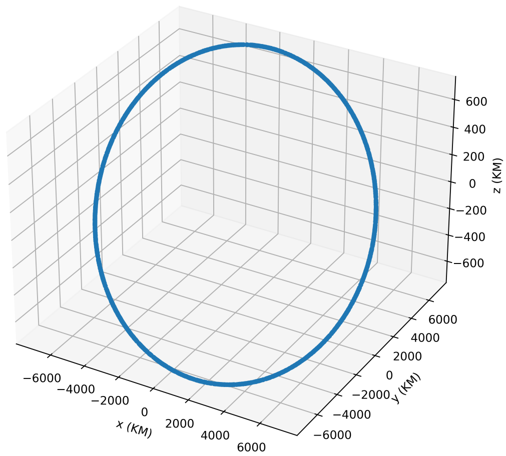
FIGURE 4.10: LAPAN-A1 state vector representation according to Two-Body propagation



(a) LAPAN-A2 X&Y-Plane Position according to Two-Body propagation

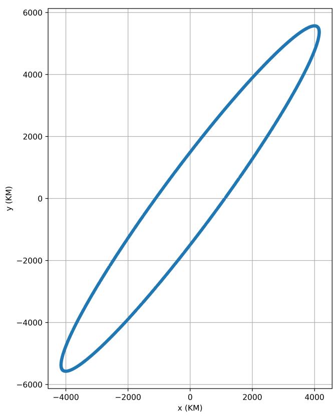


(b) LAPAN-A2 X&Z-Plane Position according to Two-Body propagation

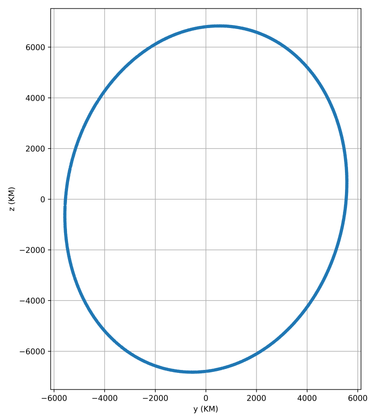


(c) LAPAN-A2 Spatial Position according to Two-Body propagation

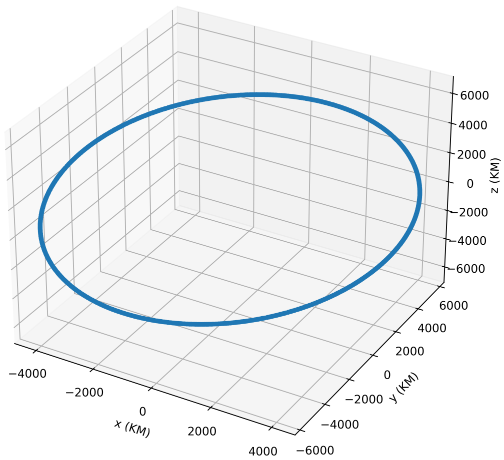
FIGURE 4.11: LAPAN-A2 state vector representation according to Two-Body propagation



(a) LAPAN-A3 X&Y-Plane Position according to Two-Body propagation



(b) LAPAN-A3 X&Z-Plane Position according to Two-Body propagation



(c) LAPAN-A3 Spatial Position according to Two-Body propagation

FIGURE 4.12: LAPAN-A3 state vector representation according to Two-Body propagation

4.2.2 Two-Body Propagation Graphs Analysis

Orbital Elements Analysis

The colored graphs above– Fig. 4.7, Fig. 4.8, and Fig. 4.9, shows us how would the satellite behave over time using the two-body propagation technique. As can be seen in the graphs, the note that the author gave just now has been proven. Virtually, the orbital elements are constant through time, other than the true anomaly of course because. The semiparameter value of LAPAN-A1, LAPAN-A2, and LAPAN-A3 satellites does not differ much– 7,000 km, 7,020 km, and 6,880 km respectively. All three satellites has roughly the same eccentricity over the propagation period at 0.001°. Since LAPAN-A1 and LAPAN-A3 orbit are polar orbit, their inclination are around 97° and 98°, while the LAPAN-A2 is equatorial at the value of approximately 6° of inclination.

The value of the RAAN of each satellites are different. For LAPAN-A1's value is constant at 323.9°, LAPAN-A2's value is constant at 85.5°, and LAPAN-A3's value is constant at 59.7°. The AoP are also have the same sense as well, LAPAN-A1's value is constant at 306.1°, LAPAN-A2's value is constant at 76.7°, and LAPAN-A3's value is constant at 73.2°. As for the True anomaly, the value keep on increasing for LAPAN-A1 and LAPAN-A3 at a fairly similar rate, but decreasing for LAPAN-A2 at a rate that is faster in comparison to the other two.

State Vector Analysis

The blue pale blue colored graphs– Fig. 4.10, Fig. 4.11, and Fig. 4.12, show us the position vectors of each satellites that were propagated using the two-body propagation. As expected the orbit does repeat its path perfectly each time the satellites finished one revolution around the Earth. That is why we are seeing such perfect "ellipse", though if we were to look at its perifocal plane it would look like a circle, for each plane, namely in-plane (X&Y Plane), out-plane (Y&Z-Plane), and the spatial positions representation.

LAPAN-A1 and LAPAN-A3 are both orbiting the Earth in an inclination of approximately 97°, that means they are polar orbits. and from the look of the X&Y-plane position representation of both satellites, they both look very elliptical. With the X&Z-plane position representation, it is also creating a solid argument that they are indeed a polar orbit, since they look fairly circular on that plane.

On the other hand, LAPAN-A2 is orbiting the Earth at an inclination of approximately 6°, that mean it is considered as equatorial orbit. From the look of the X&Y-plane position representation of the satellite, the graph looks very much circular. Adding to that argument, the Y&Z-plane position representation shows that it moves very much "elliptical" on that plane, meaning it goes up and down the equator ever so slightly.

These seemingly constant and repetitive path of each satellites are due to the fact that The ascending node and the argument of their perigee is at a constant value. If they were to change its value overtime, then the satellite state representation graphs would give more pattern of how the orbit "changes" through time.

4.2.3 Two-Body + J_2 Propagation Results

As mentioned in the previous chapters, the two-body + J_2 propagation technique considers not only the gravitational attraction of the two body but also the perturbation due to the Earth oblateness in shape. What is it that should expected of the result of this propagation technique is that it would have an effect mainly on the right ascension of the ascending nodes (RAANs) and the argument of Perigees (AoPs) for the are not constant anymore, while the other orbital elements are seemingly still at a constant value– except for the true anomaly again of course.

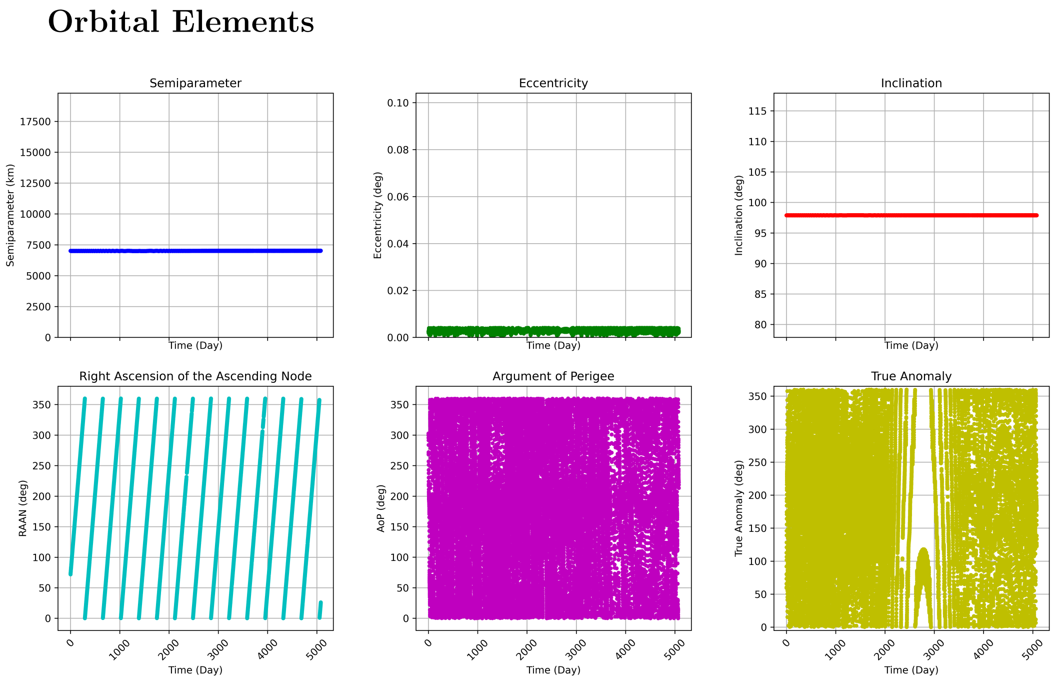


FIGURE 4.13: LAPAN-A1 Two-Body + J_2 propagated COEs

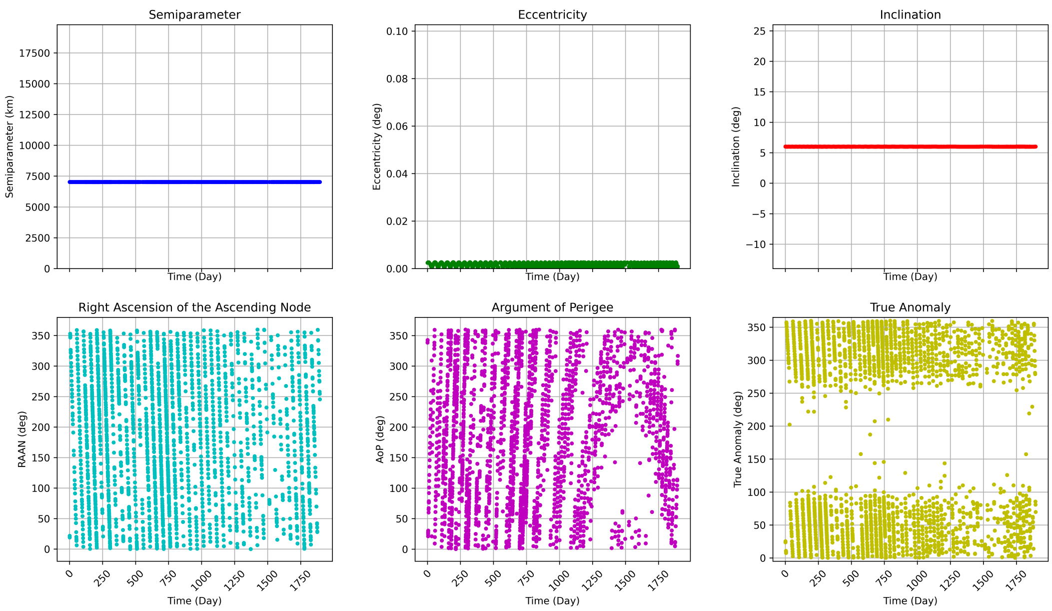


FIGURE 4.14: LAPAN-A2 Two-Body + J_2 propagated COEs

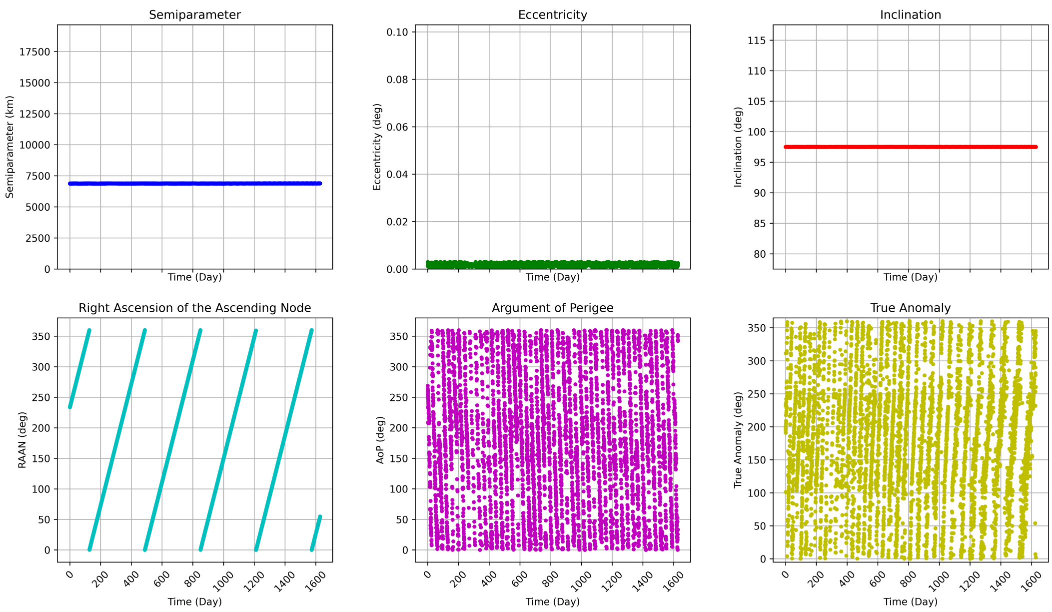
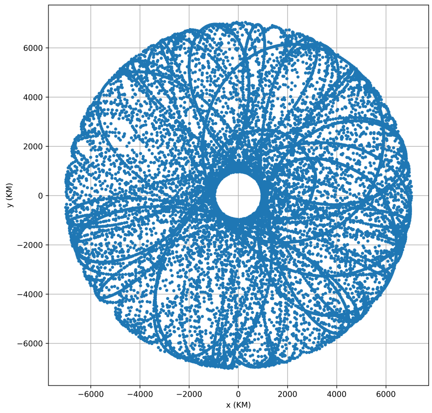
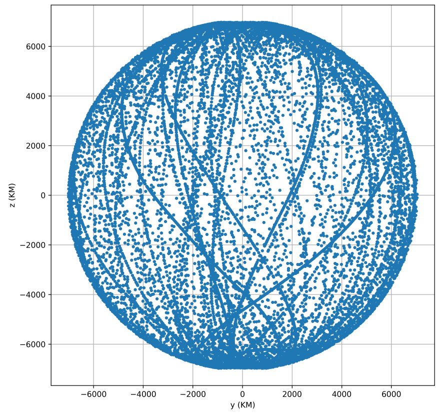


FIGURE 4.15: LAPAN-A3 Two-Body + J_2 propagated COEs

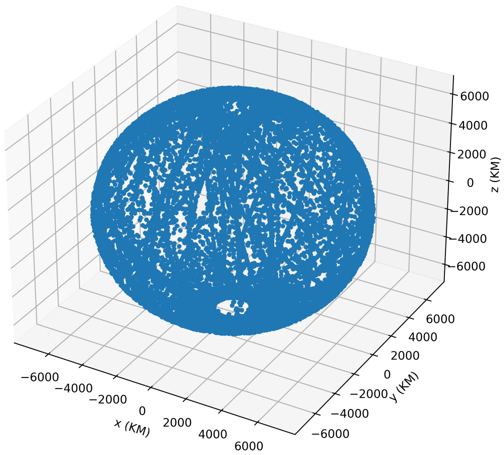
State Vector



(a) LAPAN-A1 X&Y-Plane Position according to Two-Body + J_2 propagation

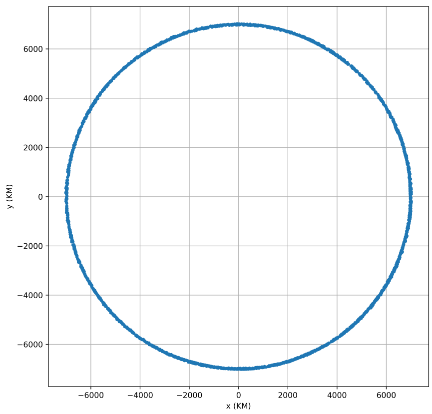


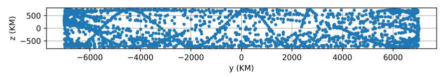
(b) LAPAN-A1 X&Z-Plane Position according to Two-Body + J_2 propagation



(c) LAPAN-A1 Spatial Position according to Two-Body + J_2 propagation

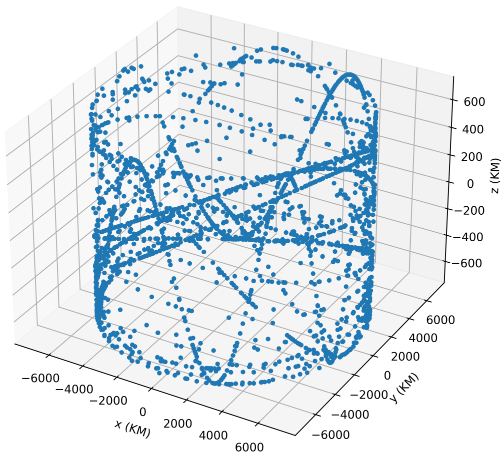
FIGURE 4.16: LAPAN-A1 state vector representation according to Two-Body + J_2 propagation





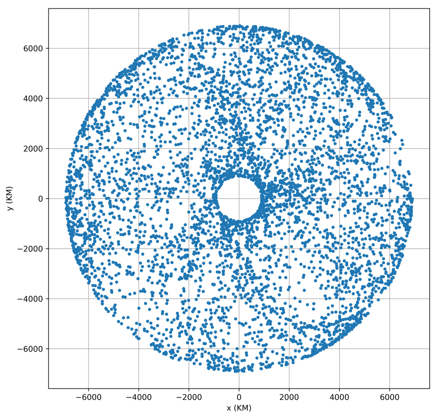
(a) LAPAN-A2 X&Y-Plane Position according to Two-Body + J_2 propagation

(b) LAPAN-A2 X&Z-Plane Position according to Two-Body + J_2 propagation

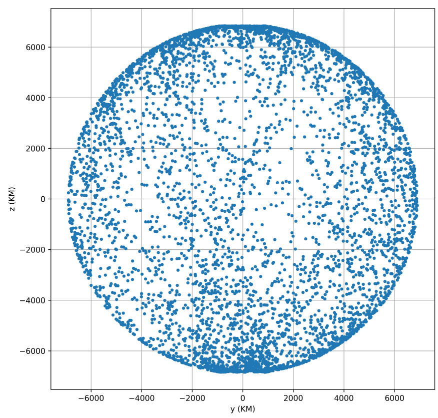


(c) LAPAN-A2 Spatial Position according to Two-Body + J_2 propagation

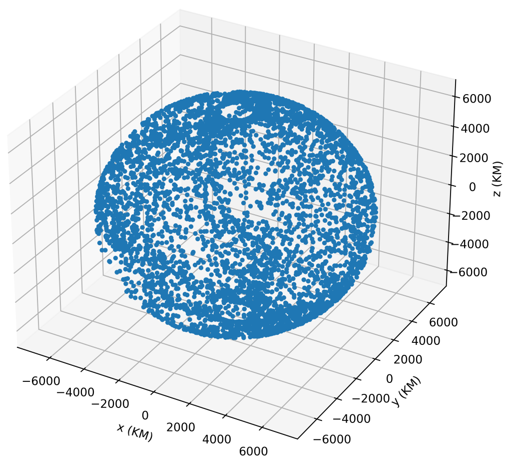
FIGURE 4.17: LAPAN-A2 state vector representation according to Two-Body + J_2 propagation



(a) LAPAN-A3 X&Y-Plane Position according to Two-Body + J_2 propagation



(b) LAPAN-A3 X&Z-Plane Position according to Two-Body + J_2 propagation



(c) LAPAN-A3 Spatial Position according to Two-Body + J_2 propagation

FIGURE 4.18: LAPAN-A3 state vector representation according to + J_2 Two-Body propagation

4.2.4 Two-Body + J_2 Propagation Graphs Analysis

Orbital Elements Analaysis

The graphs above– Fig. 4.13, Fig. 4.14, and Fig. 4.15, shows us how would the satellite behave over time using the two-body + J_2 propagation technique. As can be seen, the semiparameter value for all satellites, LAPAN-A1, LAPAN-A2, and LAPAN-A3, are still at the same value as the two-body propagation, 7,000 km, 7,020 km, and 6,880 km respectively, for this propagation technique. The value of inclination of each satellites are also still in a fairly the same and constant value of 97.8° and 97.3° of inclination for LAPAN-A1 and LAPAN-A3, and 5.9° of inclination for LAPAN-A2. The eccentricity fluctuates a little bit for all of the satellites, but at a point where the fluctuations rather negligible, or in other word the changes are in very small degree of magnitudes.

The apparent difference is the RAANs and the AoPs. Both LAPAN-A1 and LAPAN-A3 RAAN consistently increasing throughout time at a fairly the same rate, while the RAAN of LAPAN-A2 is decreasing with time but with a much faster rate compared to the others. The AoP of both LAPAN-A1 and LAPAN-A3 decreases overtime. at a fairly rapid rate, while the AoP of LAPAN-A2 increases with time, also at a fairly the same rate as the other two.

The true anomaly, just like the sense we get from the RAANs of each LAPAN-A1 and LAPAN-A3. They are increasing with time for LAPAN-A1 and LAPAN-A3, while for LAPAN-A2, the value is decreasing overtime.

State Vector Analaysis

The blue pale blue colored graphs– Fig. 4.16, Fig. 4.17, and Fig. 4.18, shows us the satellite expected behavior over time as depicted if we use the two-body + J_2 propagation technique. As expected out of the consideration of the Earth oblateness effect on the satellites' orbital path, change of RAAN and AoP, the orbital plane move / rotate ever so slightly creating some kind of pattern for each of them.

LAPAN-A1 and LAPAN-A3 orbital paths become more apparent. In comparison to the idealized two-body, their X&Y-plane position representation gives more "donut" pattern on its motion. What the author meant by "donut" is that rather than cutting through the polar axis of the Earth, the satellites' orbit just a little to the side of the axis line, and the accumulated orbiting pattern creates what it seems like a "donut". The X&Z-plane position representation of LAPAN-A1 gives more sense that they are indeed orbiting from polar to polar, as the paths are moving "virtually" vertical. While for LAPAN-A3 does not really give a sense of how it behaves around the equator, unfortunately.

As for LAPAN-A2 it is also more apparent that it is an equatorial orbit. On the X&Y-plane position representation, the satellite is orbiting the Earth at an almost circular pattern. Looking at the X&Z-plane position representation, we can clearly see how the satellite is behaving around the equator. As seen, the satellites moves in a "sinusoidal"-like pattern along its path in revolving the Earth. Moving on to the spatial position representation, the satellites behavior becomes more apparent. The satellite is in fact an equatorial orbit since it never travels through the polar.

4.2.5 SGP4 Propagation Results

As mentioned in the previous chapters, the SGP4 propagation technique considers not only the gravitational attraction of the two body but also the secular and short- and long-periodic variations due to Earth oblateness, solar and lunar gravitational effects, gravitational resonance effects and orbital decay (e.g., atmospheric drag that uses power density functions). That means, to some extent of accuracy, this propagation method should v=give the most accurate results in comparison to the other two methods. But that does not mean that there will not be any errors resulted in the propagations, only the errors should be relatively small compared to the other two.

Orbital Elements

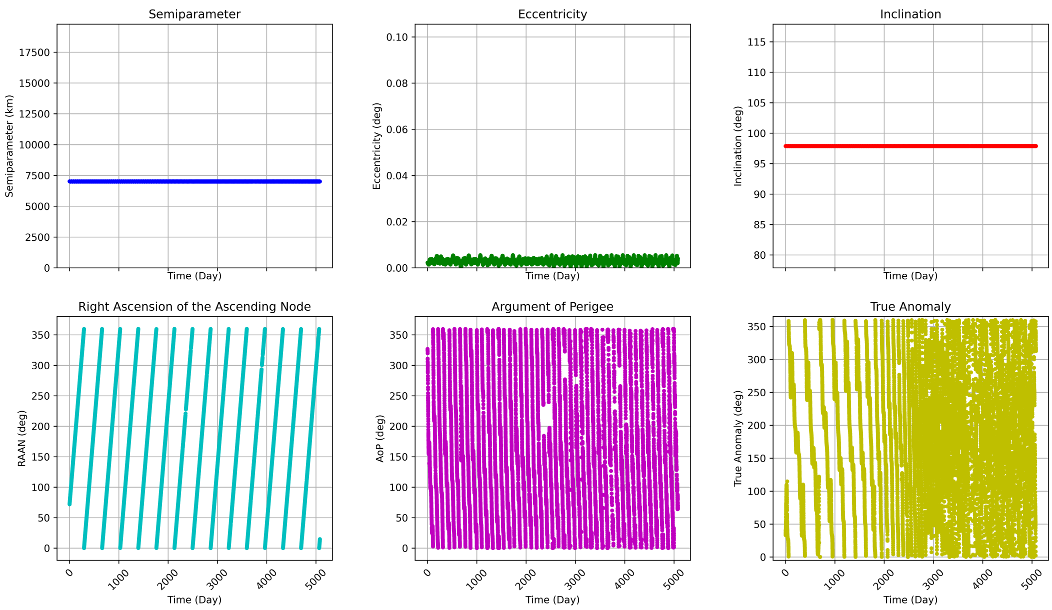


FIGURE 4.19: LAPAN-A1 SGP4 propagated COEs

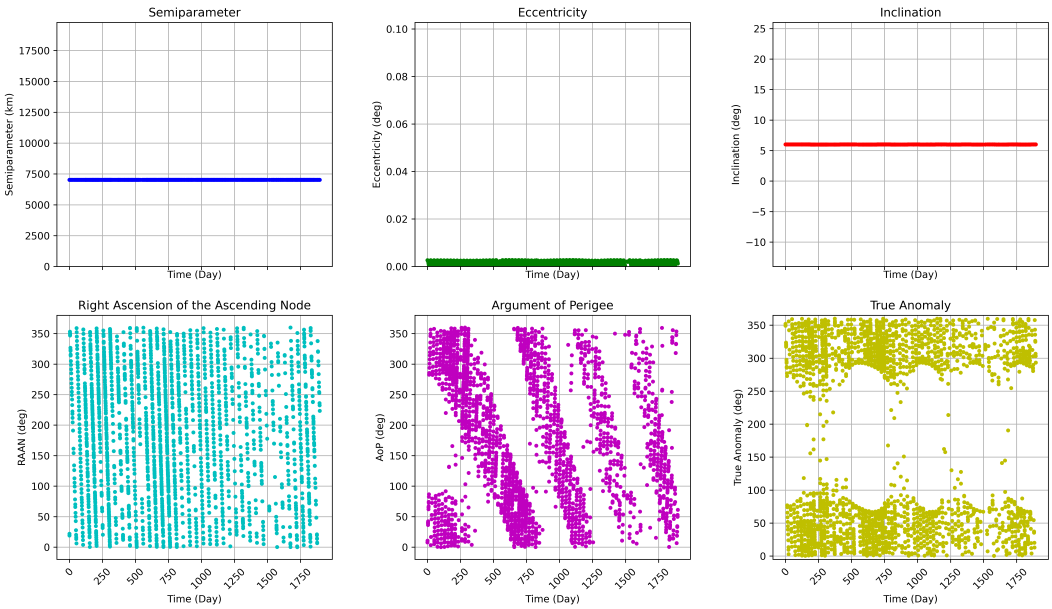


FIGURE 4.20: LAPAN-A2 SGP4 propagated COEs

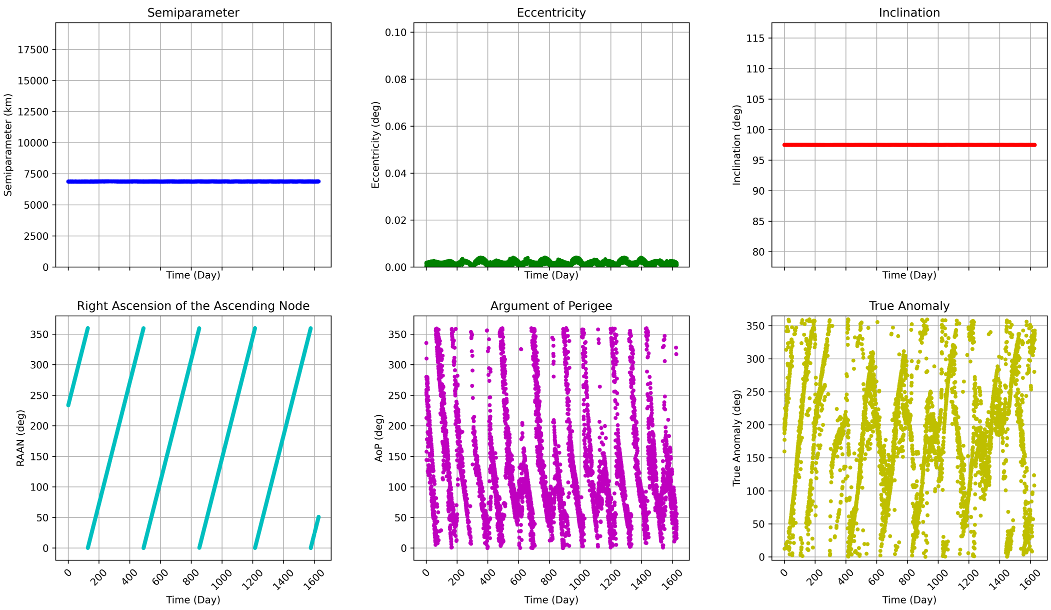
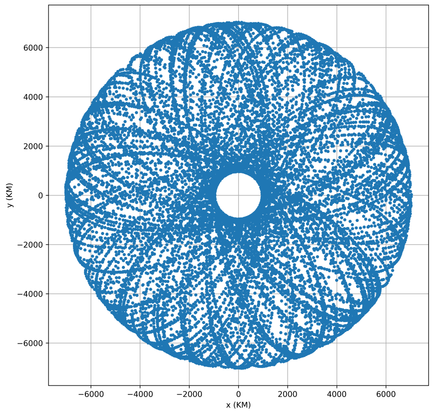
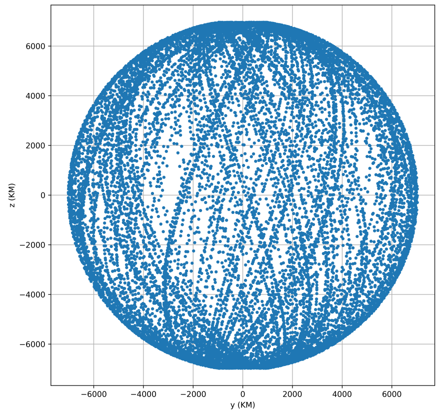


FIGURE 4.21: LAPAN-A3 SGP4 propagated COEs

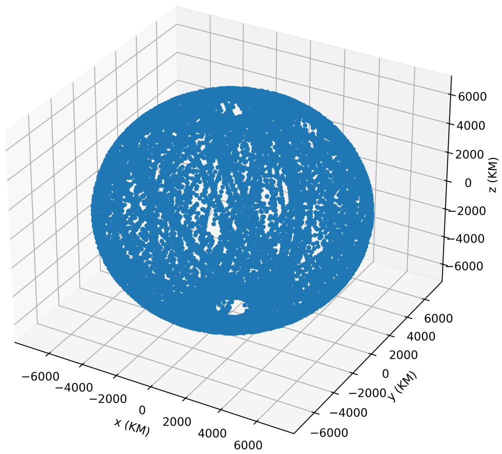
State Vector



(a) LAPAN-A1 X&Y-Plane Position according to SGP4 propagation

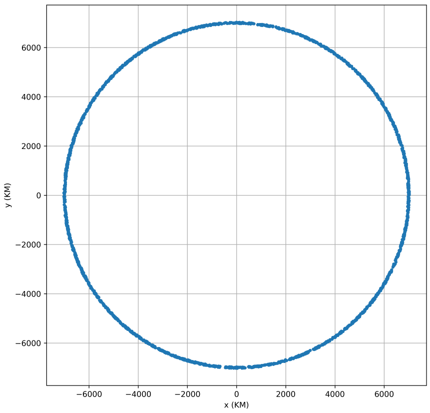


(b) LAPAN-A1 X&Z-Plane Position according to SGP4

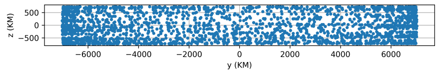


(c) LAPAN-A1 Spatial Position according to SGP4

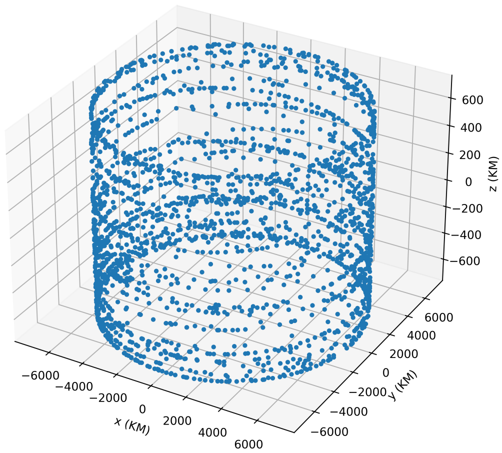
Figure 4.22: LAPAN-A1 state vector representation according to SGP4



(a) LAPAN-A2 X&Y-Plane Position according to SGP4propagation

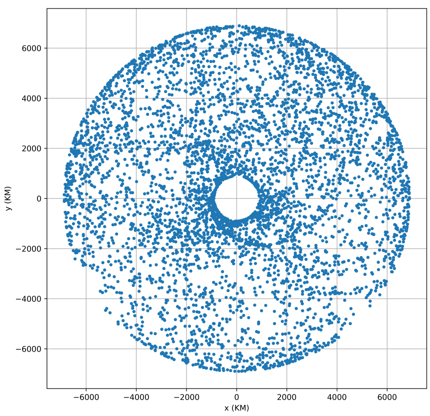


(b) LAPAN-A2 X&Z-Plane Position according to SGP4 propagation

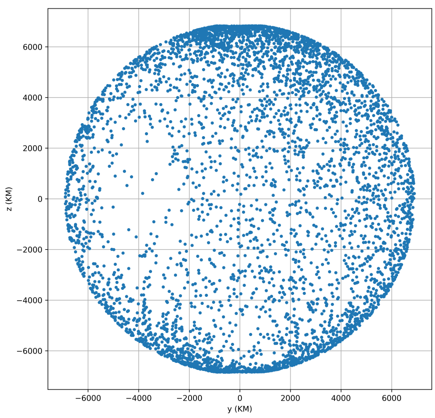


(c) LAPAN-A2 Spatial Position according to SGP4 propagation

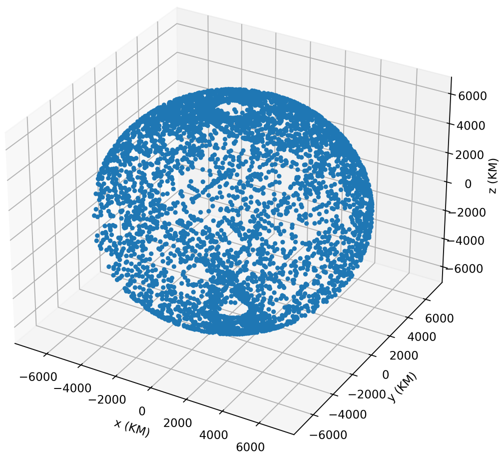
Figure 4.23: LAPAN-A2 state vector representation according to SGP4 propagation



(a) LAPAN-A3 X&Y-Plane Position according to SGP4 propagation



(b) LAPAN-A3 X&Z-Plane Position according to SGP4 propagation



(c) LAPAN-A3 Spatial Position according to SGP4 propagation

FIGURE 4.24: LAPAN-A3 state vector representation according to SGP4 propagation

4.2.6 SGP4 Propagation Graphs Analysis

Orbital Elements Analysis

The colored graphs above– Fig. 4.19, Fig. 4.20, and Fig. 4.21, shows us how would the satellite behave over time by the means of the SGP4 propagation technique. Generally looking at them, it gives a sense that the orbital elements changes more than what the previous two propagation methods describe. As seen the values for the inclination and semiprameter for all of the satellites are virtually does not change with time, while the other values do change.

StartingThe semiparameter value of LAPAN-A1, LAPAN-A2, and LAPAN-A3 satellites does not differ much– 7,007.67 km, 7,025.16 km, and 6883.46 km respectively. The value of inclination of each satellites are also still in a fairly the same and constant value of 97.89° and 97.51° of inclination for LAPAN-A1 and LAPAN-A3, and 5.99° of inclination for LAPAN-A2. The eccentricity fluctuates a little bit for all of the satellites, the changes are in very small degree of magnitudes. LAPAN-A1 fluctuates between 0° - 0.0055°, LAPAN-A2 fluctuates between 0° - 0.0027°, and LAPAN-A3 fluctuates between 0° - 0.0041°.

The more apparent difference, just like in the two-body + J_2 propagation, are the RAANs and the AoPs. Both LAPAN-A1 and LAPAN-A3 RAAN consistently increasing throughout time at different rate of change of which the LAPAN-A3's change is faster, while the RAAN of LAPAN-A2 is decreasing with time but with a much faster rate compared to the others. The AoP of both LAPAN-A1 and LAPAN-A3 decreases overtime at a fairly rapid rate, while the AoP of LAPAN-A2 increases with time, also at a fairly the same rate as the other two.

As of the true anomaly of each satellites, just like the sense we get from the RAANs of each LAPAN-A1 and LAPAN-A3, they are increasing in value with time. For LAPAN-A1 and LAPAN-A3, while for LAPAN-A2, the values are fluctuating between around 250° - 100° but in a sense that the fluctuation occurs crossing the 0° value.

State Vector Analysis

The blue pale blue colored graphs– Fig. 4.22, Fig. 4.23, and Fig. **??**, shows us the satellite expected behavior over time as depicted if we use the SGP4 propagation technique. As expected, just like in the two-body + J_2 propagation, the positions representation would not be constant as depicted by the two-body propagation. The position representation does create some pattern but gives a bit more of "abstract" sense in comparison to the other two.

LAPAN-A1 and LAPAN-A3 orbital paths become more apparent. Just like the two-body + J_2 position representation, their X&Y-plane position representation gives more "donut" pattern on its motion. Again, what the author want to explain by use of "donut" as an image of representation is that rather than cutting through the polar axis of the Earth, the satellites' orbit just a little to the side of the axis line, and the accumulated orbiting pattern creates what it seems like a "donut". The X&Z-plane position representation for both satellites gives the same sense as the two-body + J_2 position representation. LAPAN-A1 gives more sense that they are indeed orbiting from polar to polar, as the paths are moving "virtually" vertical. While for LAPAN-A3 does not really give a sense of how it behaves around the equator, unfortunately.

As for LAPAN-A2 it is also more apparent that it is an equatorial orbit. Looking at the X&Y-plane position representation, the satellite is orbiting the Earth at an almost circular pattern. Now, Looking at the X&Z-plane position representation, we can clearly see how the satellite is only behaving around the equator. Moving on to the spatial position representation, the satellites behavior becomes more apparent. The satellite is in fact an equatorial orbit since it never travels through the polar. In addition to that, the satellite seemed to moves up and down in a sinusoidal-like pattern at a longer period compared to what the two-body + J_2 presented.

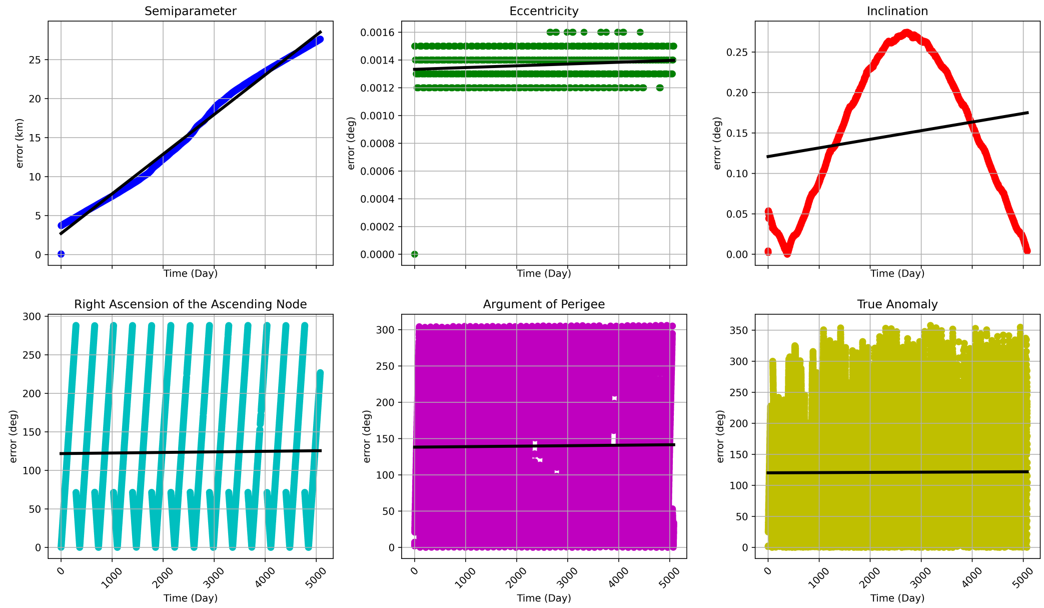
4.3 Propagation Error Analysis (Linear Regression)

For this section the author did the comparison between the propagated state vectors and the propagated orbital elements to the control data of the TLE historical data. The comparison lead to the error of each parameter in each respective state representation. The errors are in absolute value, so that way there will not be any confusion in the "visualization" of the errors. The error analysis of the orbital elements in this section was done by implementing the linear regression analysis.

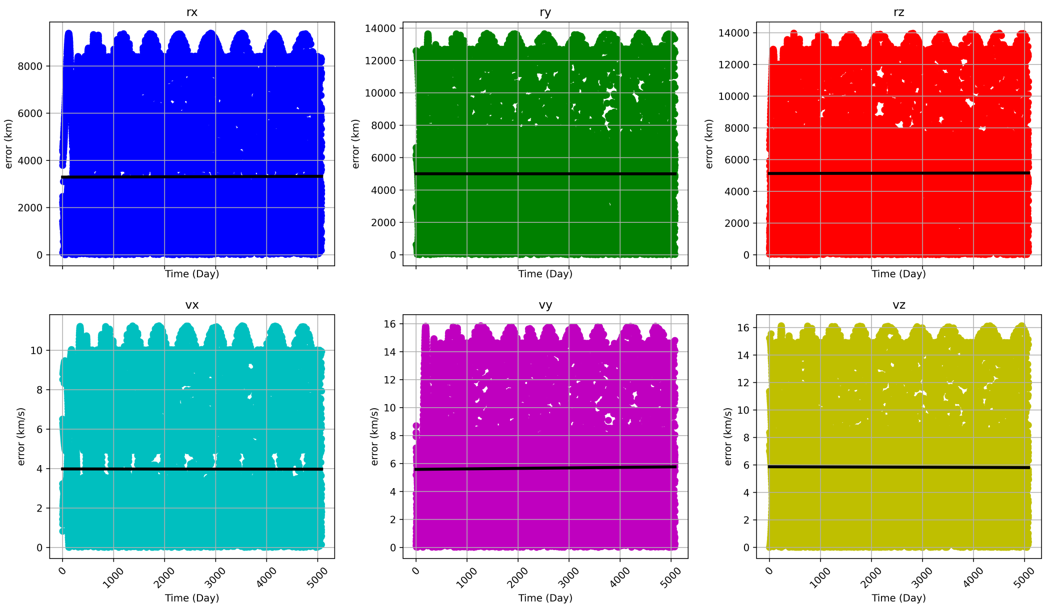
4.3.1 Two-Body Propagation Error

Generally speaking, before getting into the actual value of the error and its analysis, the two-body propagation are expected to have quite big of a difference in comparison to the actual data of the TLE historical data. What can be expected is that, since the two-body does not give any deviation on the orbital elements, the orbital elements should error should be changing its value– keep on increasing or decreasing, over time with respect to the actual TLE historical data.

LAPAN-A1



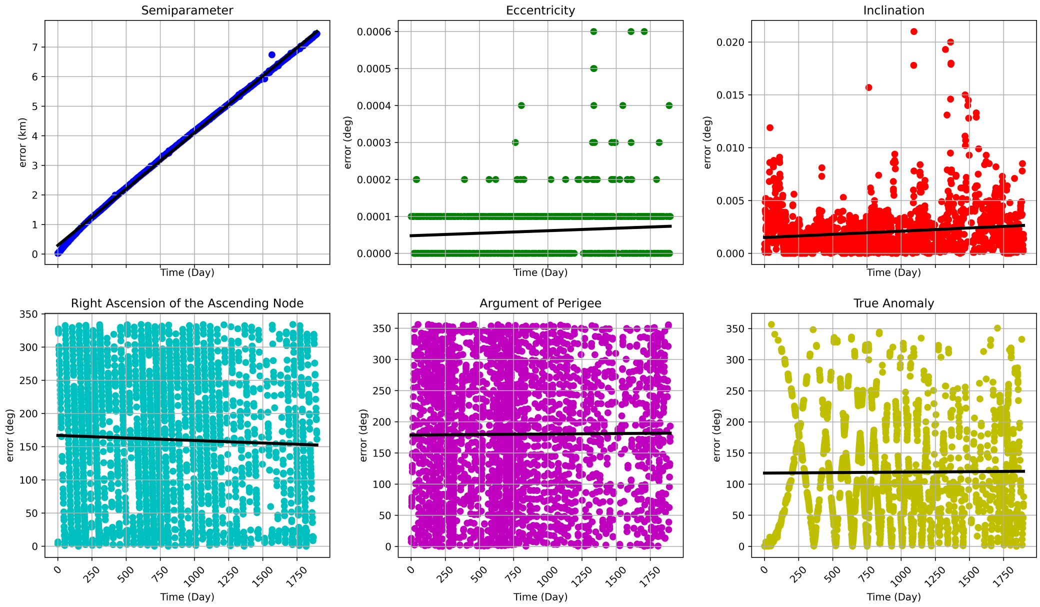
(a) LAPAN-A1 two-body propagation orbital elements error



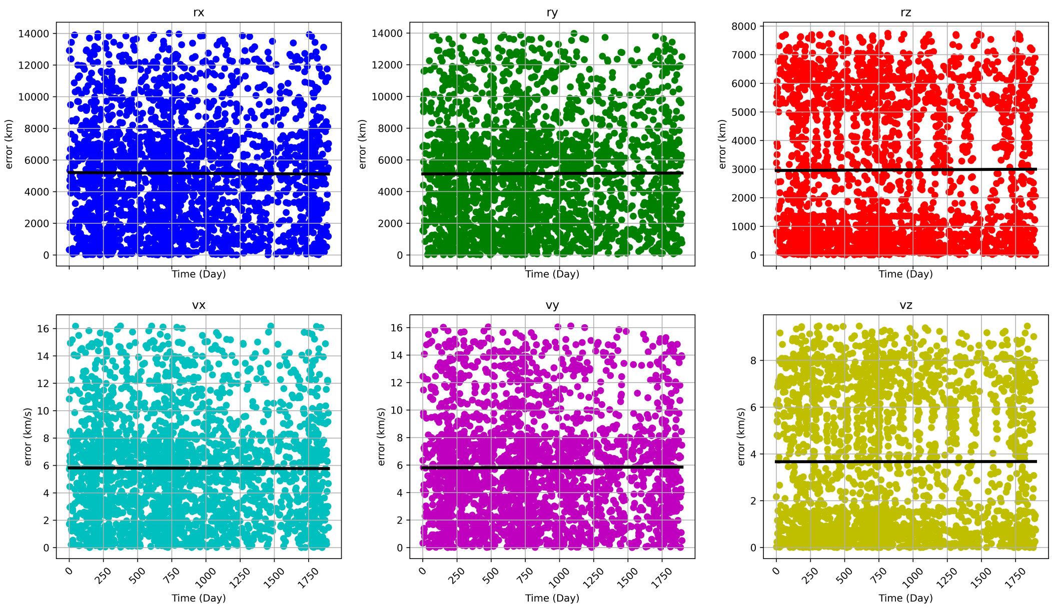
(b) LAPAN-A1 two-body propagation state vector error

FIGURE 4.25: LAPAN-A1 two-body propagation error

LAPAN-A2



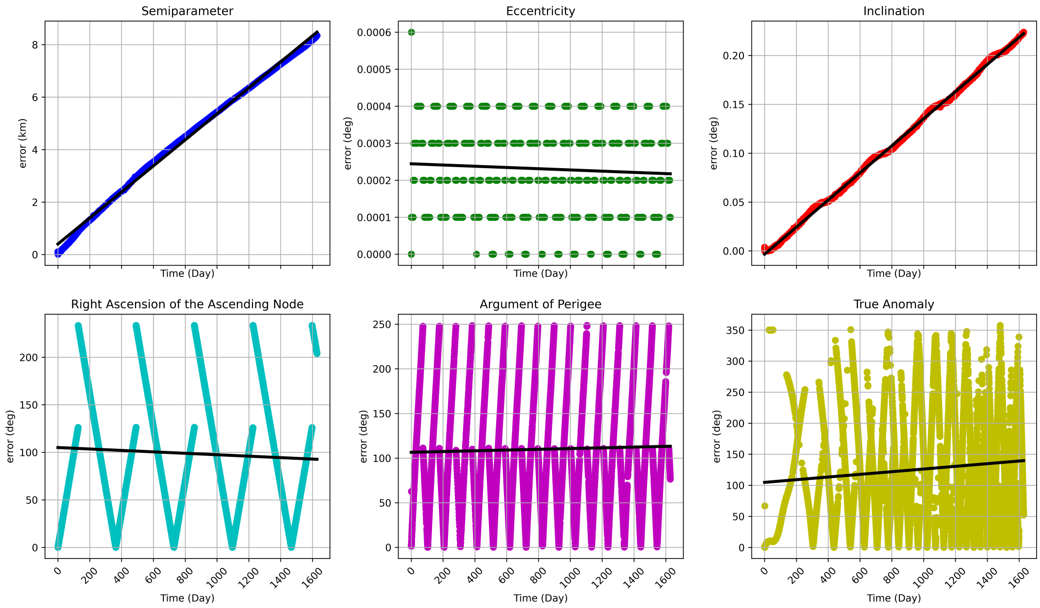
(a) LAPAN-A2 two-body propagation orbital elements error



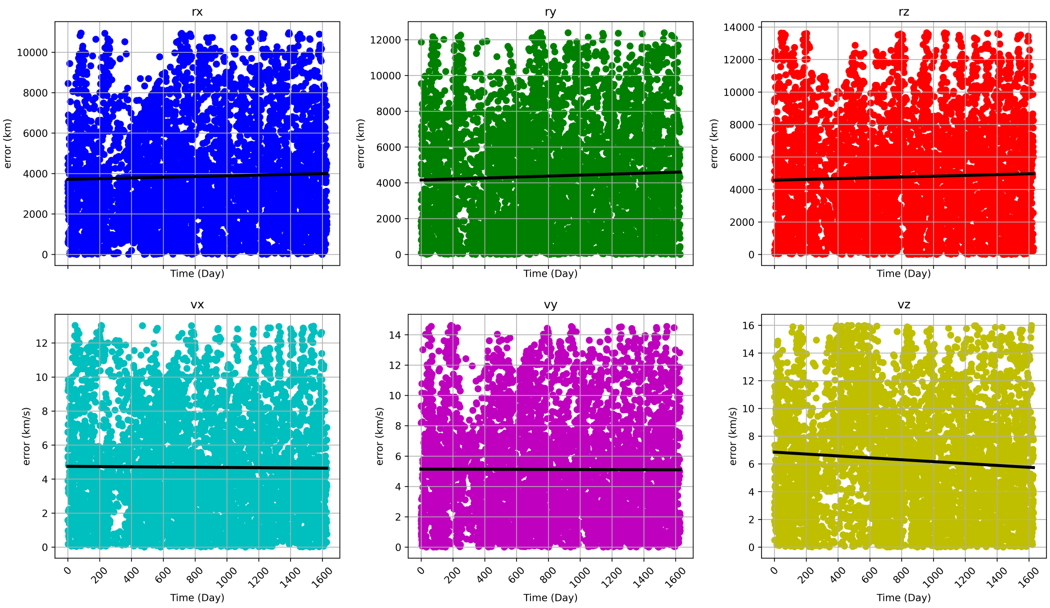
(b) LAPAN-A2 two-body propagation state vector error

FIGURE 4.26: LAPAN-A2 two-body propagation error

LAPAN-A3



(a) LAPAN-A3 two-body propagation orbital elements error



(b) LAPAN-A3 two-body propagation state vector error

FIGURE 4.27: LAPAN-A3 two-body propagation error

4.3.2 Error Analysis

	A1_2B	A2_2B	A3_2B
p =	2.6982 + 0.0051 x	0.2911 + 0.0038 x	0.4012 + 5 x
e =	0.0013 + 0 x	0 + 0 x	0.0002 + 0 x
i =	0.1208 + 0 x	0.0015 + 0 x	-0.0034 + 0.0001 x
Ω =	121.7552 + 0.0007 x	166.948 + (-0.0076) x	105.0284 + (-0.0076) x
ω =	138.1147 + 0.0007 x	178.7672 + (-0.0015) x	106.503 + 0.0042 x
ν =	120.1327 + 0.0003 x	117.6468 + 0.0015 x	104.7099 + 0.0215 x

Table 4.1: Linear regression equation for two-body propagated orbital elements

	A1_2B	A2_2B	A3_2B
rx =	3,290.673 + 0.0064 x	5,206.1286 + (-0.0501) x	3,711.978 + 0.1763 x
ry =	5,000.4565 + (-0.0009) x	5,110.5869 + 0.0307 x	4,157.076 + 0.2723 x
rz =	5,122.3508 + 0.0064 x	2,951.3924 + 0.0238 x	4,569.4084 + 0.2451 x
vx =	3.9783 + 0 x	5.8189 + 0 x	5.142 + 0 x
vy =	5.5796 + 0 x	5.7958 + 0 x	4.0677 + 0.0576 x
vz =	5.8677 + 0 x	3.6662 + 0 x	6.8475 + (-0.0007) x

Table 4.2: Linear regression equation for two-body propagated state vector

	A1_2B_R^2	A2_2B_R^2	A3_2B_R^2
p	0.9933	0.999	0.9967
e	0.0277	0.0135	0.0036
i	0.0318	0.0233	0.999
Ω	0.0002	0.0016	0.003
ω	0.0001	0.0001	0.0008
ν	0	0.0001	0.0125
rx	0	0.0001	0.001
ry	0	0	0.0017
rz	0	0	0.0011

vx	0	0	0.0001
vy	0.0002	0	0
vz	0	0	0.0058

Table 4.3: LAPAN's satellites correlation value of two-body propagation error

Orbital Elements Error Analysis

The graphs in Fig. 4.25(a), Fig. 4.26(a), and Fig. 4.26(a) shows behavior of error as well as the regression analysis of each two-body propagated orbital elements error with respect to the propagation time. To give a more detailed sense of information, Table. 4.3 gives the correlation of each orbital elements error with the propagation time. Linear regression equations are given in the Table. 4.1. By looking at the graphs, it might get a little bit confusing to determine how the error behave over time, that is why the author will only describe them as described with the linear regression analysis that the author had made.

According to the graph and the information of linear regression gathered from the Table. 4.1, LAPAN-A1 the eccentricity error and the inclination error does not change their values, at 0.0013° and 0.1208°, over time as their coefficients are zero. While the other orbital elements do change their values. The RAAN, the AoP, and the true anomaly do change, but at a very slight value over time, 0.0007° each day for RAAN and AoP. An increase 0.0003° each day for the true anomaly. The semiparameter on the other hand keep on increasing at 0.0051 km each day. Though, to be noted, from the Table. 4.3, the only one that has strong correlation is the semiparameter, while the others have very low correlation value.

According to the graph and the information of linear regression gathered from the Table. 4.1 for two LAPAN-A2, same as the LAPAN-A1, the eccentricity error and the inclination error does not change their values, at 0° and 0.0015°, over time as their coefficients are zero. The semiparameter and the true anomaly increases their value over time, while the RAAN and the AoP decreases their value over time. The semiparameter increases 0.0038 km each day, and for the true anomaly, it increases 0.0015° each day. The RAAN decreases 0.0076° of its values every day and for the AoP, the values decreases 0.0015° every day. Same as the correlation in the LAPAN-A1, the only one that has strong correlation between the error and the propagation time is the semiparameter, while the others have very low correlation value.

Lastly, according to the graph and the information of linear regression gathered from the Table. 4.1 for two LAPAN-A3, only the eccentricity that does not have any changes of value, at 0.0002°, over time. The RAAN is the only one that decreases its value over time, while the others increases their value. The value of error of RAAN, as stated before, decreases at a rate of 0.0076°per day. The error of semiparameter increases its value at 5 km each day. The inclination increases its value of error at 0.0001° per day, the AoP at 0.0042° per day, and the true anomaly at 0.0215° per day. For LAPAN-A3, the ones that has strong correlation between the error and the propagation time are the semiparameter and the inclination, while the others, just like before, have very low correlation value.

State Vector Error Analysis
The graphs in Fig. 4.25(b), Fig. 4.26(b), and Fig. 4.27(b) shows behavior of error as well as the regression analysis of each state vector error with respect to the propagation time. To give a more detailed sense of information, Table. 4.3 gives the correlation of each state vectors error with the propagation time. Linear regression equations are given in the Table. 4.2. Looking at the state vector error graphs, it might gives a lot of confusion on what does it mean or how does the state vector behave, that is why the author will only describe them as described with the linear regression analysis that the author had made.

For LAPAN-A1, the position vectors change over time, while the velocity vector does not. The rx error and the rz error keep increasing its value over time at 0.0064 km, for both, every day. while the ry error value decreases by 0.0009 km every day. The velocity vectors, as stated before and according to the equation of regression on the stated table, does not change its value from it initial intercept at 3.9783 $\frac{km}{s}$, 5.5796 $\frac{km}{s}$, and 5.8677 $\frac{km}{s}$ respectively. Now this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

For LAPAN-A2 case, the position vectors change over time, while the velocity vector does not same sense as the LAPAN-A1 but different parameters that changes. The ry error and the rz error keep increasing its value over time at 0.0307 km and 0.0238 km, respectively, every day. while the rx error value decreases by 0.0501 km every day. The velocity vectors, as stated before and according to the equation of regression on the stated table, does not change its value from it initial intercept at 5.8189 $\frac{km}{s}$, 5.7958 $\frac{km}{s}$, and 3.6662 $\frac{km}{s}$ respectively. Just like in LAPAN-A1 case, this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

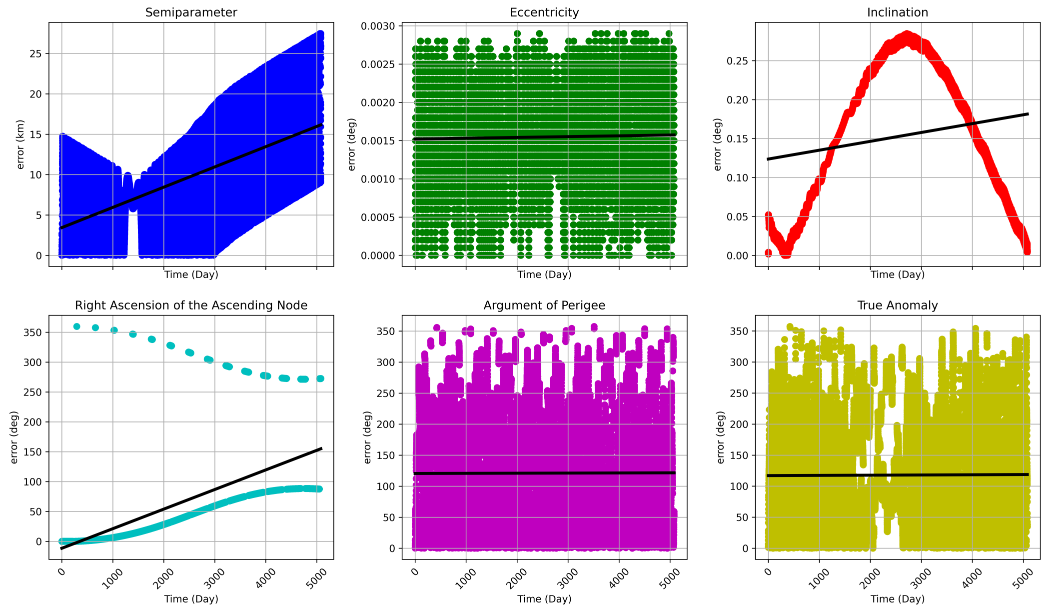
For LAPAN-A3 case, the values of the state vectors are changing, except for the value in the vx for which the equation stated that it stays constant at 5.142 $\frac{km}{s}$. The position vector's error values increases each day of the propagation time. for rx, the value increases 0.1763 km per day, for ry the value increases 0.2723 km per day, and for rz the value increases 0.2451 km per day. The vy value is also increasing with time at 0.0576 $\frac{km}{s}$ per day, while the rz decreases over time at 0.0007 $\frac{km}{s}$ each day. Again this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

The author explicitly separate the correlation values analysis for the state vector because of the fact that there are very little correlation between the state vector errors with the propagation time. That might explain why does the graphs seem so complicated yet the equation does not really give any changes of value over the propagation time.

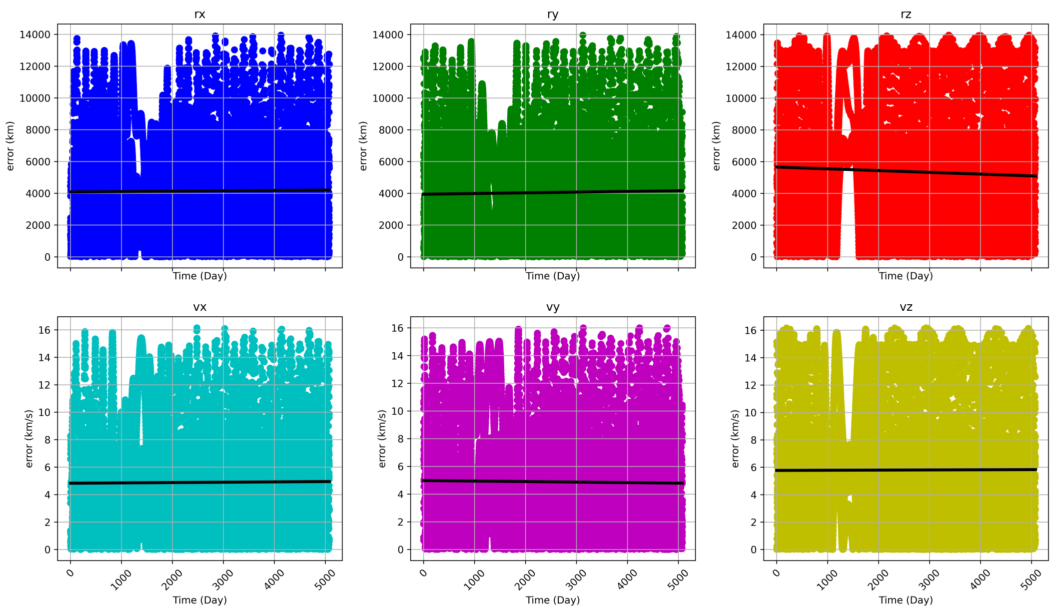
4.3.3 Two-Body + J_2 Propagation Error

For the two-body + J_2 propagation, the values are expected to have some kind of periodic deviation towards the actual data of the TLE historical data. What can be expected is that, since the two-body + J_2 mainly effect the change or right ascension of the ascending node and the argument of perigee and does not give any significant deviation on the other orbital elements, then, the orbital elements should should be changing its value– keep on increasing or decreasing. In comparison to the two-body, there should be less errors.

LAPAN-A1



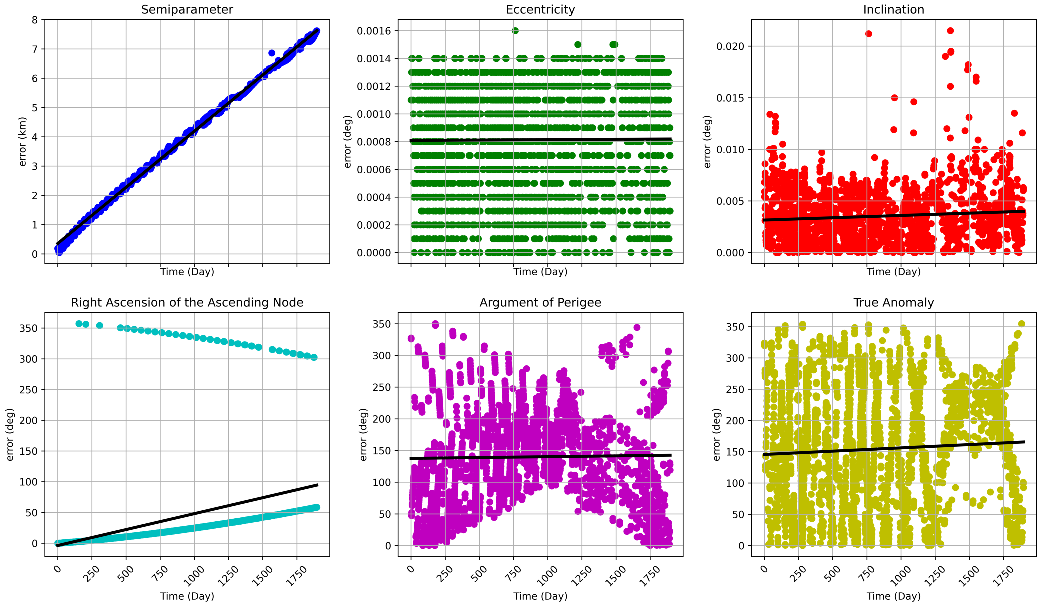
(a) LAPAN-A1 two-body + J_2 propagation orbital elements error



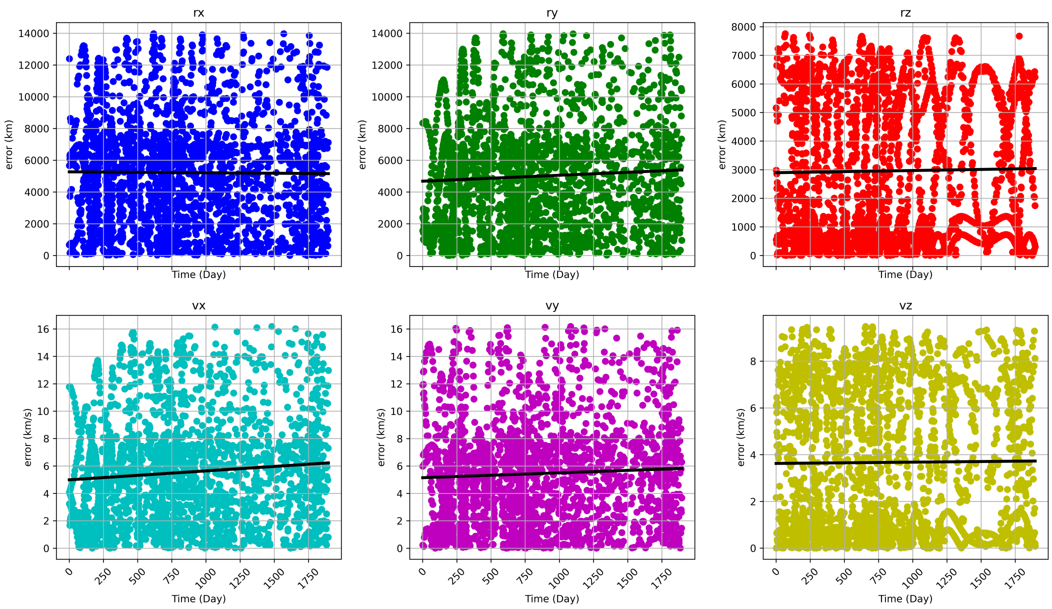
(b) LAPAN-A1 two-body + J_2 propagation state vector error

FIGURE 4.28: LAPAN-A1 two-body + J_2 propagation error

LAPAN-A2



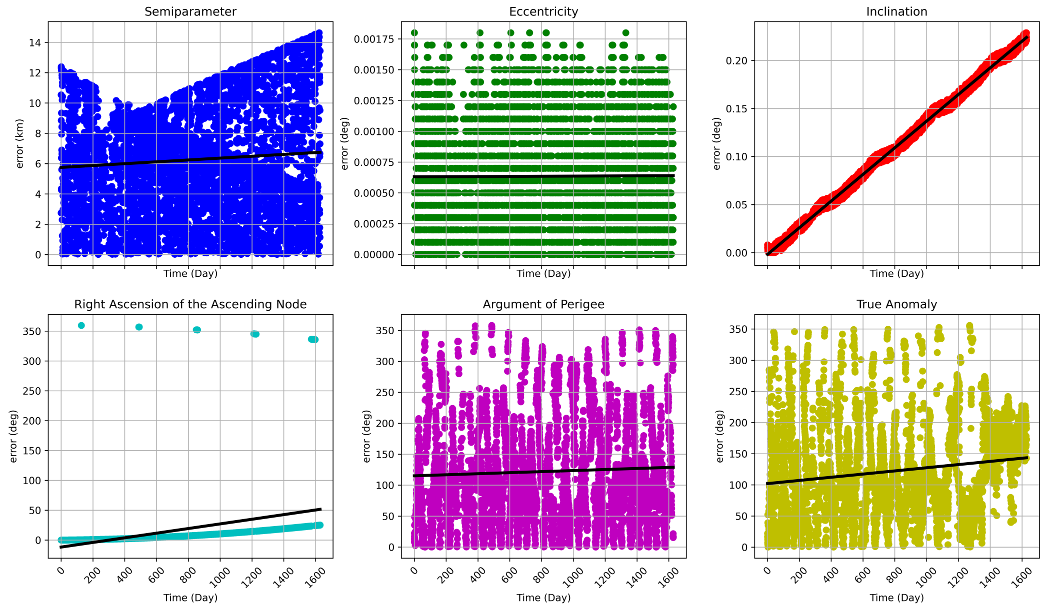
(a) LAPAN-A2 two-body + J_2 propagation orbital elements error



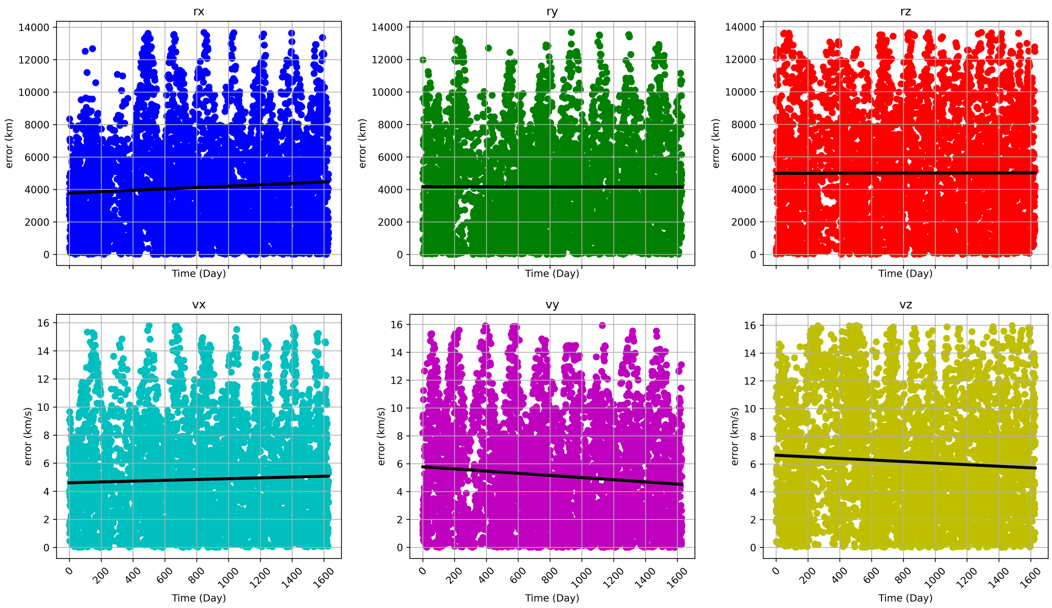
(b) LAPAN-A2 two-body + J_2 propagation state vector error

FIGURE 4.29: LAPAN-A2 two-body + J_2 propagation error

LAPAN-A3



(a) LAPAN-A3 two-body + J_2 propagation orbital elements error



(b) LAPAN-A3 two-body + J_2 propagation state vector error

FIGURE 4.30: LAPAN-A3 two-body + J_2 propagation error

4.3.4 Error Analysis

	A1_J2	A2_J2	A3_J2
p =	3.4365 + 0.0025 x	0.3526 + 0.0038 x	5.7503 + 0.0006 x
e =	0.0015 + 0 x	0.0008 + 0 x	0.0006 + 0 x
i =	0.1237 + 0 x	0.0031 + 0 x	-0.0018 + 0.0001 x
Ω =	-11.6521 + 0.0328 x	-3.781 + 0.0518 x	-11.7375 + 0.0388 x
ω =	120.5276 + 0.0002 x	137.5393 + 0.0026 x	115.0335 + 0.0085 x
ν =	117.0027 + 0.0003 x	145.6562 + 0.0105 x	101.9116 + 0.0254 x

TABLE 4.4: Linear regression equation for two-body + J_2 propagated orbital elements

	A1_J2	A2_J2	A3_J2
rx =	4,079.1916 + 0.0241 x	5,269.2541 + (-0.0596) x	3,774.569 + 0.4279 x
ry =	3,942.1179 + 0.0434 x	4,679.6811 + 0.3728 x	4,167.0299 + (-0.0111) x
rz =	5,655.8373 + (-0.1113) x	2,894.0721 + 0.0762 x	4,982.8045 + 0.0164 x
vx =	4.8253 + 0 x	4.9899 + 0.0007 x	4.6019 + 0.0003 x
vy =	4.9757 + 0 x	5.144 + 0.0004 x	5.7819 + (-0.0008) x
vz =	5.7693 + 0 x	3.6257 + 0 x	6.6371 + 0 x

TABLE 4.5: Linear regression equation for two-body + J_2 propagated state vector

	A1_J2_R^2	A2_J2_R^2	A3_J2_R^2
p	0.3186	0.9977	0.0063
e	0.0005	0	0
i	0.0351	0.0082	0.9958
Ω	0.3124	0.1371	0.0875
ω	0	0.0003	0.0021
nu	0	0.0035	0.0235
rx	0.0001	0.0001	0.0045
ry	0.0005	0.0034	0
rz	0.0022	0.0003	0

vx	0.0001	0.0078	0.0015
vy	0.0002	0.0025	0.0096
vz	0	0.0001	0.0041

TABLE 4.6: LAPAN's satellites correlation value of two-body + J_2 propagation error

Orbital Elements Error Analysis

The graphs in Fig. 4.28(a), Fig. 4.29(a), and Fig. 4.30(a) shows behavior of error as well as the regression analysis of each two-body + J_2 propagated orbital elements error with respect to the propagation time. Table. 4.6 give a more detailed sense of information for which it provides the correlation of each orbital elements error with the propagation time. Linear regression equations are given in the Table. 4.4. Same sense as the error analysis in the two-body, the graphs might get a little bit confusing to determine how the error behave over time, that is why the author will only describe them as described with the linear regression analysis that the author had made.

According to the graph and the information of linear regression gathered from the Table. 4.4, the eccentricity error and the inclination error of LAPAN-A1 do not change their values, at 0.0015° and 0.1237°, over time as their coefficient values are both zero. While the other orbital elements do change their values. The RAAN, the AoP, and the true anomaly do increase their value over time, but at a very slight value, 0.0328° each day for the RAAN, 0.0002° each day for the AoP, and 0.0003° each day for the true anomaly. The semiparameter on the other hand keep on increasing its value at 0.0025 km each day. Though, to be noted, from the Table. 4.6, none of the error value has strong correlation to the propagation time, in fact they all have very low correlation value.

According to the graph and the information of linear regression gathered from the Table. 4.4 for two LAPAN-A2, same as the LAPAN-A1, the eccentricity error and the inclination error does not change their values, at 0.0008° and 0.0031°, over time as their coefficients are zero. All of the other elements are increasing in values. The semiparameter increases 0.0038 km each day. The RAAN increases 0.0518° of its values every day and for the AoP, the values increases 0.0026° every day. and for the true anomaly, it increases 0.0105° each day. The only one that has strong correlation between the error and the propagation time is the semiparameter, while the others have very low correlation value.

Lastly, according to the graph and the information of linear regression gathered from the Table. 4.4 for two LAPAN-A3, only the eccentricity that does not have any changes of value, at 0.0006° over time. While the rest of the orbital elements' error increases with time. The semiparameter error increases 0.0006 km each day, the inclination error increases 0.0001° each day, the RAAN error increases 0.0388° each day, the AoP error increases 0.0085° each day, the true anomaly error increases 0.0254° each day. Those values are only according to the regression equations, but then the correlation should always be in consideration as well, in this case only the inclination error has a strong correlation to the propagation period, while the others are virtually does not correlate whatsoever.

State Vector Error Analysis

The graphs in Fig. 4.28(b), Fig. 4.29(b), and Fig. 4.30(b) shows behavior of error as well as the regression analysis of each state vector error with respect to the propagation time. Table. 4.6 gives a more detailed sense of information of the correlation between each state vectors error with the propagation time. Linear regression equations are given in the Table. 4.5. Looking at the state vector error graphs, it might gives a lot of confusion on what does it mean or how does the state vector behave, that is why the author will only describe them as described with the linear regression analysis that the author had made.

For LAPAN-A1, the position vectors change over time, while the velocity vector does not. The only error value that is decreasing over time is the rz error values at 0.1113 km every day, while the other position vectors are increasing their values. for rx and ry, they increased their value of error by 0.0241 km and 0.0434 km, respectively. The velocity vectors, as stated before and according to the equation of regression on the stated table, does not change its value from it initial intercept at 4.8253 $\frac{km}{s}$, 4.9757 $\frac{km}{s}$, and

5.7693 $\frac{km}{s}$ respectively. Now this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

For LAPAN-A2 case, All of the vectors changes except the vz value for which it stays at its intercept value of 3.6257 $\frac{km}{s}$ over time. The only vector that is decreasing with time is the rx value at which it decreases by 0.0596 km per day. All of the other vectors are increasing with time. The ry error increases by 0.3728 km, rz error increases by 0.0762 km, vx error increases by 0.0007 $\frac{km}{s}$, vy error increases by 0.0004 $\frac{km}{s}$. Just like in LAPAN-A1 case, this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

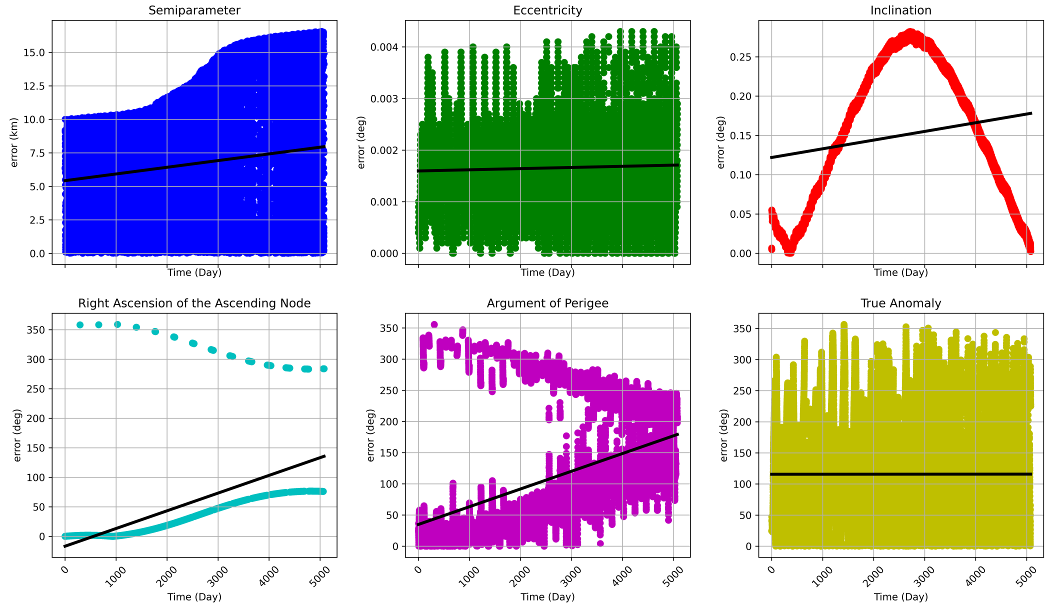
For LAPAN-A3 case, the values of the state vectors are changing except the vz at which it stays at 6.6371 $\frac{km}{s}$. The only two that are decreasing in value over time are the ry and vy at which they decrease by 0.0111 km per day and 0.0008 $\frac{km}{s}$ per day, respectively. The other vectors are increasing in value over the propagation time. The rx error increases by 0.4279 km, rz error increases by 0.0164 km, vx error increases by 0.0003 $\frac{km}{s}$.

Again this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value. The author explicitly separate the correlation values analysis for the state vector because of the fact that there are very little correlation between the state vector errors with the propagation time. That might explain why does the graphs seem so complicated yet the equation does not really give any changes of value over the propagation time.

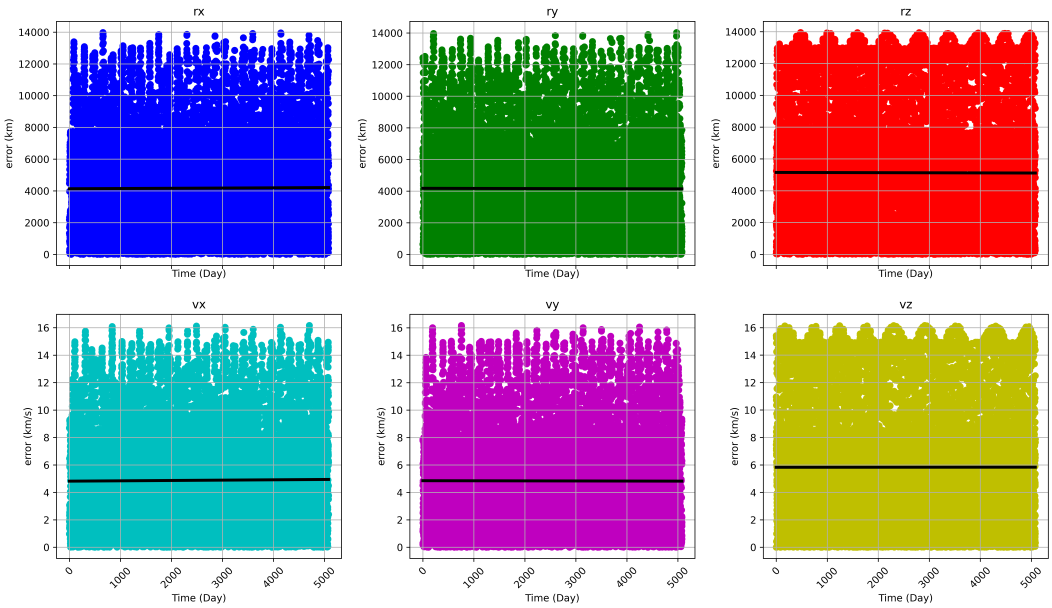
4.3.5 SGP4Propagation Error

For the SGP4 propagation, the values are also expected to have some kind of periodic deviation towards the actual data of the TLE historical data. Since the orbital elements' values are changing, it is to be expected is that the error values are relatively much smaller than that of the other methods. As also the increase or decrease of error should be relatively small. But those are just some assumptions. The figures and tables below showshow exactly the propagator behave.

LAPAN-A1



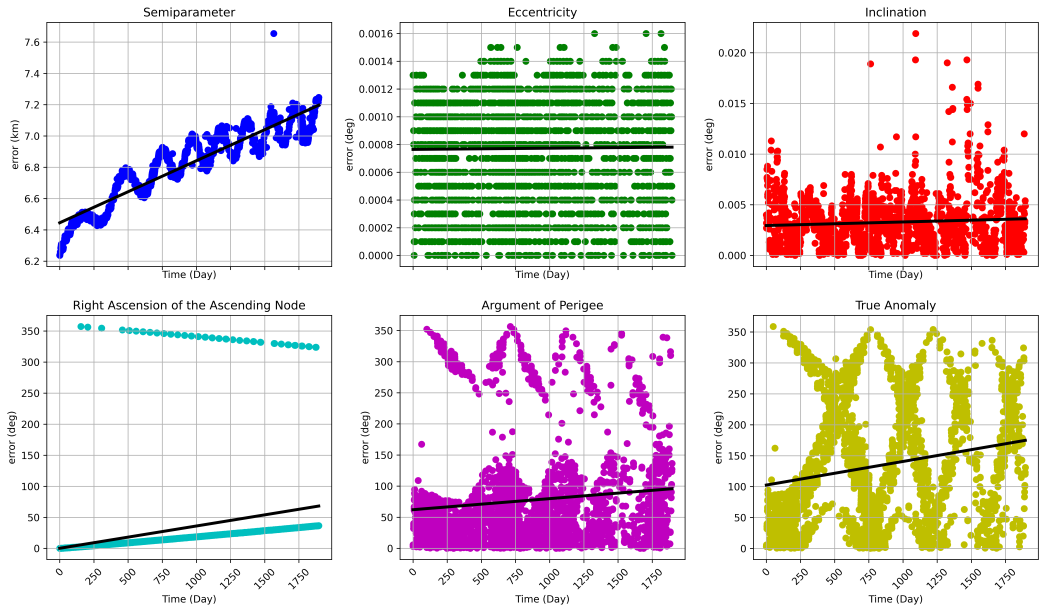
(a) LAPAN-A1 SGP4 propagation orbital elements error



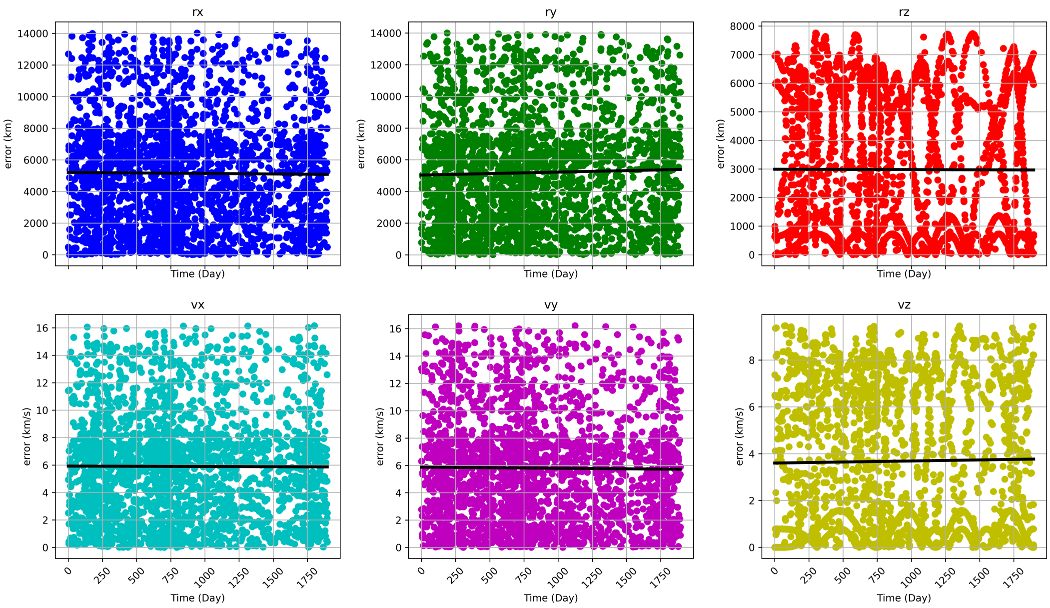
(b) LAPAN-A1 SGP4 propagation state vector error

FIGURE 4.31: LAPAN-A1 SGP4 propagation error

LAPAN-A2



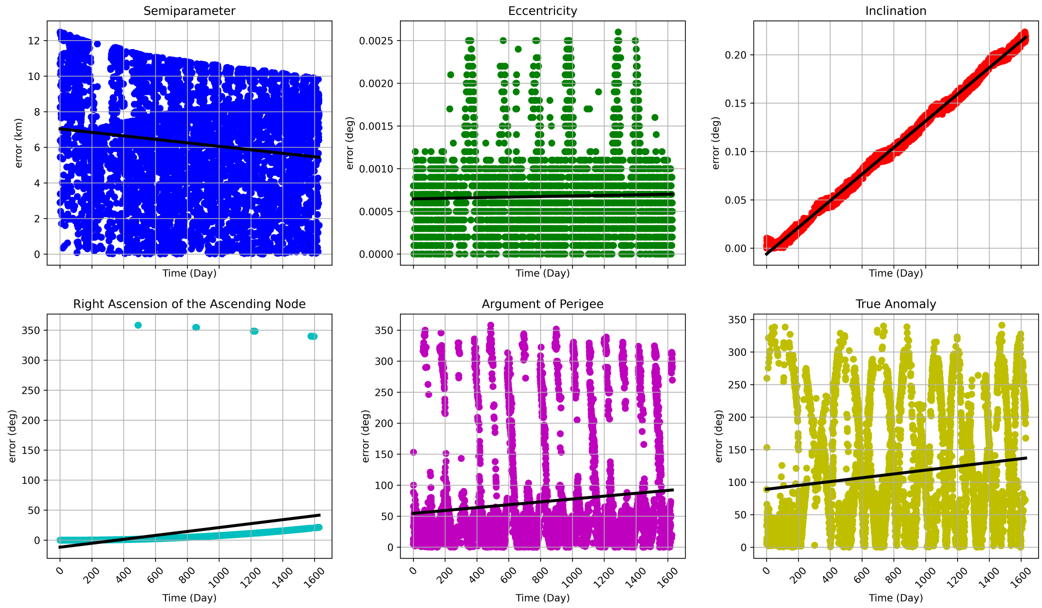
(a) LAPAN-A2 SGP4 propagation orbital elements error



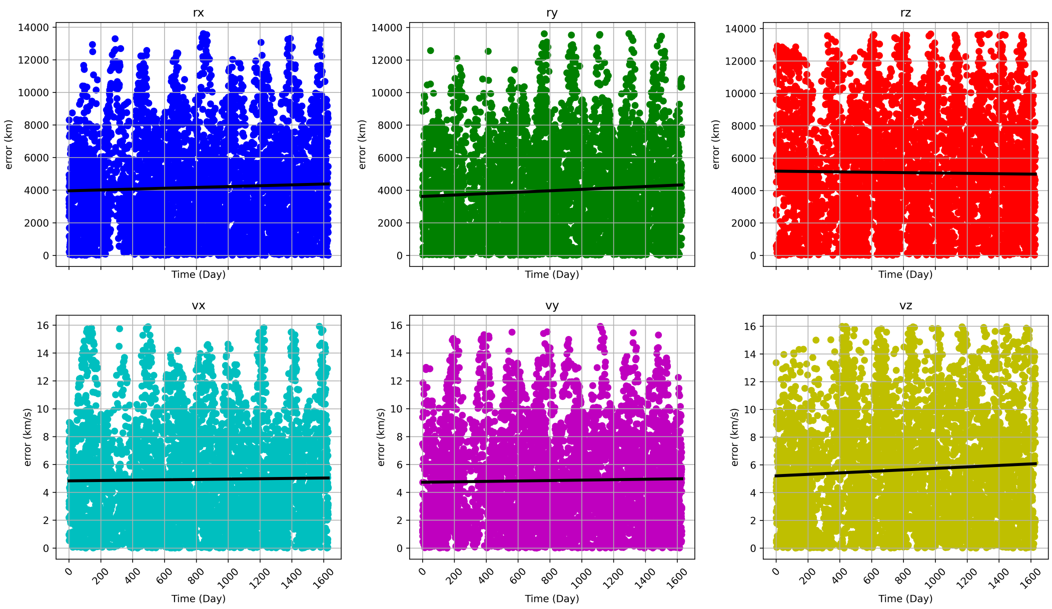
(b) LAPAN-A2 SGP4 propagation state vector error

FIGURE 4.32: LAPAN-A2 SGP4 propagation error

LAPAN-A3



(a) LAPAN-A3 SGP4 propagation orbital elements error



(b) LAPAN-A3 SGP4 propagation state vector error

FIGURE 4.33: LAPAN-A3 SGP4 propagation error

4.3.6 Error Analysis

	A1_SGP4	A2_SGP4	A3_SGP4
p =	5.4237 + 0.0005 x	6.4455 + 0.0004 x	7.0386 + (-1) x
e =	0.0016 + 0 x	0.0008 + 0 x	0.0006 + 0 x
i =	0.1219 + 0 x	0.0029 + 0 x	-0.0061 + 0.0001 x
Ω =	-16.7475 + 0.03 x	0.1677 + 0.0359 x	-11.7663 + 0.0327 x
ω =	34.8795 + 0.0284 x	62.1533 + 0.0178 x	54.5802 + 0.0231 x
ν =	115.6339 + 0 x	102.5064 + 0.0381 x	89.0796 + 0.0294 x

Table 4.7: Linear regression equation for SGP4 propagated orbital elements

	A1_SGP4	A2_SGP4	A3_SGP4
rx =	5,207.3519 + 0.0146 x	5,207.3519 + (-0.0653) x	3,959.4613 + 0.2592 x
ry =	5,033.9398 + (-0.0069) x	5,033.9398 + 187 x	3,618.9749 + 0.4341 x
rz =	2,989.0102 + (-0.0074) x	2,989.0102 + (-0.0125) x	5,196.5038 + (-0.1148) x
vx =	5.927 + 0 x	5.927 + 0 x	4.8237 + 0.0001 x
vy =	5.8694 + 0 x	5.8694 + (-0.0001) x	4.723 + 0.0002 x
vz =	3.5995 + 0 x	3.5995 + 0 x	5.2084 + 0.0005 x

Table 4.8: Linear regression equation for SGP4 propagated state vector

	A1_SGP4_R^2	A2_SGP4_R^2	A3_SGP4_R^2
p	0.0278	0.8443	0.0204
e	0.0013	0.0001	0.001
i	0.0336	0.0059	0.9958
Ω	0.2849	0.0794	0.0812
ω	0.2038	0.0127	0.0129
ν	0	0.0429	0.0218
rx	0.0001	0.0001	0.0016
ry	0	0.0008	0.005
rz	0	0	0.0002

vx	0.0001	0	0.0003
vy	0	0.0001	0.0004
vz	0	0.0002	0.0041

TABLE 4.9: LAPAN's satellites correlation value of SGP4 propagation error

Orbital Elements Error Analysis

The graphs in Fig. 4.31(a), Fig. 4.32(a), and Fig. 4.33(a) shows behavior of error as well as the regression analysis of each SGP4 propagated orbital elements error with respect to the propagation time. Table. 4.9 give a more detailed sense of information for which it provides the correlation of each orbital elements error with the propagation time. Linear regression equations are given in the Table. 4.4. Same sense as the error analysis in the two-body, the graphs might get a little bit confusing to determine how the error behave over time, that is why the author will only describe them as described with the linear regression analysis that the author had made.

According to the graph and the information of linear regression gathered from the Table. 4.7, the eccentricity error, the inclination error, and the true anomaly error of LAPAN-A1 do not change their values, at 0.0016°, 0.1219°, and 115.6339° over time as their coefficient values are both zero. While the other orbital elements do change their values. The RAAN, the semiparameter, and the true anomaly do increase their value over time, but at a very slight value, 0.03° each day for the RAAN, 0.0284° each day for the AoP, 0.0005 km each day for the semiparameter. Though it is a little bit frowned upon to take what the equation describes as is, so from the Table. 4.9 we can see how the variable correlates and turns out none of the error value has strong correlation to the propagation time, in fact they have very low correlation value.

According to the graph and the information of linear regression gathered from the Table. 4.7 for two LAPAN-A2, same as the LAPAN-A1, the eccentricity error and the inclination error does not change their values, at 0.0008° and 0.0029°, over time as their coefficients are zero. All of the other elements are increasing in values. The semiparameter increases 0.0004 km each day. The

RAAN increases 0.0359° of its values every day and for the AoP, the values increases 0.0178° every day. and for the true anomaly, it increases 0.0381° each day. The only one that has strong correlation between the error and the propagation time is the semiparameter, while the others have very low correlation value.

Lastly, according to the graph and the information of linear regression gathered from the Table. 4.7 for two LAPAN-A3, only the eccentricity that does not have any changes of value, at 0.0006°, over time. While almost all of the rest of the orbital elements' error increases with time. The inclination error increases 0.0001° each day, the RAAN error increases 0.0327° each day, the AoP error increases 0.0231° each day, the true anomaly error increases 0.0294° each day. While the semiparameter error decreases 1 km each day. Those values are only according to the regression equations, but then the correlation should always be in consideration as well, in this case only the inclination error has a strong correlation to the propagation period, while the others are virtually does not correlate whatsoever.

State Vector Error Analysis
The graphs in Fig. 4.31(b), Fig. 4.32(b), and Fig. 4.33(b) shows behavior of error as well as the regression analysis of each state vector error with respect to the propagation time. Table. 4.6 gives a more detailed sense of information of the correlation between each state vectors error with the propagation time. Linear regression equations are given in the Table. 4.8. Looking at the state vector error graphs, it might gives a lot of confusion on what does it mean or how does the state vector behave, that is why the author will only describe them as described with the linear regression analysis that the author had made.

For LAPAN-A1, the position vectors error change over time, while the velocity vector does not. The only error value that is increasing over time is the rx error values at 0.0146 km every day, while the other position vectors are decreasing their values. for ry and rz, they increased their value of error by 0.0069 km and 0.0074 km, respectively. The velocity vectors, as stated before and according to the equation of regression on the stated table, does not change its value from it initial intercept at 5.927 $\frac{km}{s}$, 5.8694 $\frac{km}{s}$, and 3.5995 $\frac{km}{s}$ respectively. Now this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

For LAPAN-A2 case, All of the vectors error changes except the vx and vz value for which it stays at its intercept value of 5.927 $\frac{km}{s}$ and 3.5995 $\frac{km}{s}$ over time. The only vector that is increasing with time is the ry value at which it increasing by 187 km per day. All of the other vectors are decreasing with time. The rx error decreases by 0.0653 km, rz error decreases by 0.0125 km, vy error decreases by 0.0001 $\frac{km}{s}$. Just like in LAPAN-A1 case, this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value.

For LAPAN-A3 case, the values of the state vectors error are changing throughout the propagation period. All of the error values are increasing but the rz where it decreases its value by 0.1148 km per day. the rx and ry values are increasing by 0.2592 km and 0.43431 km. As for the velocity vectors error, vx, vy, and vz, their values are increasing by 0.0001 $\frac{km}{s}$ per day, 0.0002 $\frac{km}{s}$ per day, and 0.0005 $\frac{km}{s}$ per day, respectively.

Again this does not mean that the values are in a constant "line" of dots. The value varies, but since most of the values are cancelling out each other fluctuation effects, the equation seemed to give a non changing value. The author explicitly separate the correlation values analysis for the state vector because of the fact that there are very little correlation between the state vector errors with the propagation time. That might explain why does the graphs seem so complicated yet the equation does not really give any changes of value over the propagation time.

4.4 Propagation Error Analysis (Fourier Analysis)

For this section, The same principle applies, all of the errors are in their absolute values, for a better view of the errors. As stated in the last part of the last chapter, the seemingly periodic graphs / function were taken and being treated further for more analysis. The analysis that were done for them are in a form of Fourier analysis where the author needed to "guess" the period and also the value of the coefficient or how high the order is to achieve convergence for the particular function / graph.

	LAPAN-A1	LAPAN-A2	LAPAN-A3
Two-Body	Inclination	-	-
Two-Body + J_2	Inclination	-	-
	RAAN	-	-
SGP4	Inclination	Semiparameter	-
	RAAN	-	-

TABLE 4.10: The seemingly periodic graphs

4.4.1 Two-Body Propagation Error

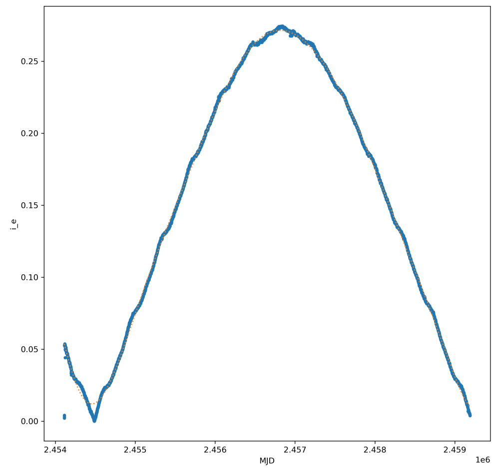


FIGURE 4.34: Inclination error of LAPAN-A1 satellite using Two-Body propagation

As depicted in the Table. 4.10, the only one that was seemingly periodic was the Inclination error for the LAPAN-A1 satellite. As shown also in the Fig. 4.34, the Inclination error does represent some sort of periodical pattern to it. Taking the Fourier analysis the number orders that was taken was six (6), due to the fact that the value of R^2 has achieved a satisfactory or convergence value.

$$\begin{aligned} y = \; & a0 + a1\cos(wx) + a2\cos(2wx) + a3\cos(3wx) \\ & + a4\cos(4wx) + a5\cos(5wx) + a6\cos(6wx) \\ & + b1\sin(wx) + b2\sin(2wx) + b3\sin(3wx) \\ & + b4\sin(4wx) + b5\sin(5wx) + b6\sin(6wx) \end{aligned} \tag{4.1}$$

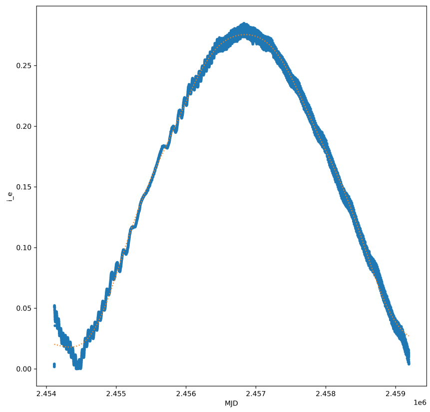
	Values
a0	-4.846593e+01
a1	-1.102166e+02
a2	-3.682966e+01
a3	6.328400e+01
a4	4.193391e+01
a5	2.037737e+00

a6	-1.382274e+00
b1	-4.828093e+01
b2	-1.230913e+02
b3	-6.739144e+01
b4	1.229214e+01
b5	1.238742e+01
b6	9.663160e-01
ω	-4.100101e-04
R^2	0.9992023578051531

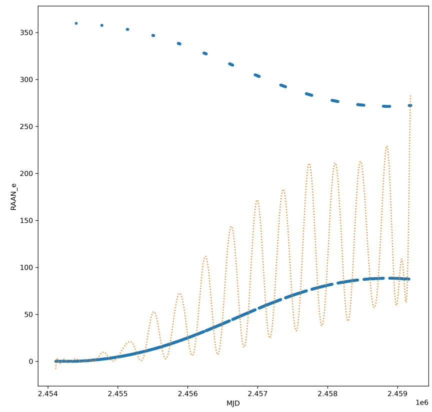
TABLE 4.11: Coefficient, frequency, and R^2 values of the Inclination error Fourier analysis

From both of the graph, equation, and the table above, it is clear that by using the Fourier analysis on the graph shows more convergence and shows a higher relation between the error and the propagation period. Showed by the value of R^2 that almost touch the value of 1, or 100% for that matter, a satisfactory relation were achieved.

4.4.2 Two-Body + J_2 Propagation Error



(a) Inclination error of LAPAN-A1 satellite using Two-Body + J_2 propagation



(b) RAAN error of LAPAN-A1 satellite using Two-Body + J_2 propagation

FIGURE 4.35: LAPAN-A1 Inclination and RAAN error using Two-Body + J_2 propagation

As depicted in the Table. 4.10, the only ones that were seemingly periodic were the Inclination error and the RAAN error for the LAPAN-A1 satellite. As shown also in the Fig. 4.35, the Inclination error does represent some sort of periodical pattern to it. Taking the Fourier analysis for the Inclination error, the number orders that was taken was five (5), due to the fact that the value of R^2 has achieved a satisfactory or convergence value. As for the RAAN error the number of order that was taken was fifty (50), due to the fact that the highest value of R^2 were given by the order.

$$
\begin{aligned}
y = \; & a0 + a1\cos(wx) + a2\cos(2wx) \\
& + a3\cos(3wx) + a4\cos(4wx) \\
& + a5\cos(5wx) + b1\sin(wx) \\
& + b2\sin(2wx) + b3\sin(3wx) \\
& + b4\sin(4wx) + b5\sin(5wx)
\end{aligned} \tag{4.2}
$$

	Values
a0	1.489364e-01

a1	-1.243990e-01
a2	-4.479891e-03
a3	-1.191218e-03
a4	2.628061e-04
a5	-5.782975e-04
b1	-3.999076e-02
b2	-1.543241e-03
b3	7.908187e-05
b4	3.293784e-03
b5	3.377461e-03
ω	-1.167335e-03
R^2	0.9957478752479194

TABLE 4.12: Coefficient, frequency, and R^2 values of the Inclination error Fourier analysis

$$
\begin{aligned}
y = \; & a0 + a1\cos(wx) + a2\cos(2wx) + a3\cos(3wx) \\
& + \cdots + a48\cos(2wx) + a49\cos(2wx) + a50\cos(2wx) \\
& + b1\sin(wx) + b2\sin(2wx) + b3\sin(3wx) \\
& + \cdots + b48\cos(2wx) + b49\sin(2wx) + b50\sin(2wx)
\end{aligned}
\tag{4.3}
$$

	Values
a0	-4.006370e+04
a1	-1.437767e+04
a2	6.378647e+04
a3	4.174249e+04
a48	-1.063557e+05
a49	3.275605e+04
a50	-4.374729e+03
b1	-2.153780e+04
b2	-1.668072e+04
b3	1.575393e+04
b48	8.152135e+04
b49	-2.946996e+04
b50	4.512614e+03

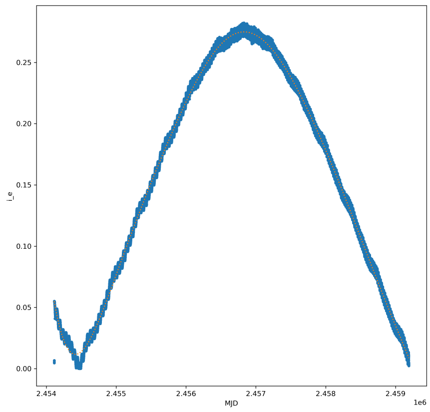
ω	4.450230e-04
R^2	0.5765167666801361

TABLE 4.13: Coefficient, frequency, and R^2 values of the RAAN error Fourier analysis

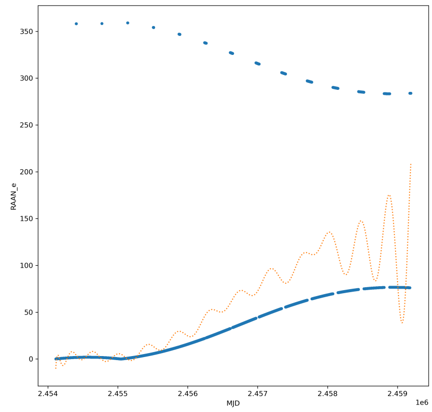
From both of the Fig. 4.35(a), Eq. 4.2, and the Table. 4.12, it is clear that by using the Fourier analysis on the graph shows more convergence and shows a higher relation between the error and the propagation period. Showed by the value of R^2 that almost touch the value of 1, or 100% for that matter, a satisfactory relation were achieved.

On the other hand, the Fig. 4.35(b), Eq. 4.3, and the Table. 4.13, it is clear that in comparison to the linear analysis of the error, the Fourier analysis shows a more promising / satisfactory results on the behavior of the error with respect to time. The R^2 value of the data fir shows a relation of 0.5 or 50%.

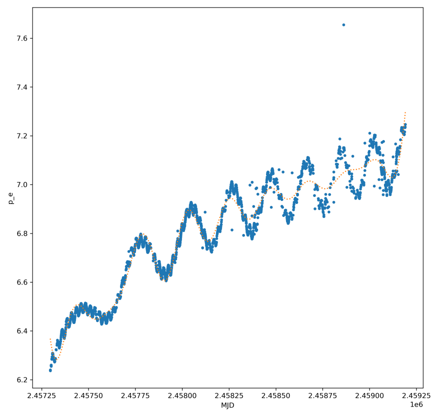
4.4.3 SGP4 Propagation Error



(a) Inclination error of LAPAN-A1 satellite using Two-Body + J_2 propagation



(b) RAAN error of LAPAN-A1 satellite using Two-Body + J_2 propagation



(c) RAAN error of LAPAN-A2 satellite using Two-Body + J_2 propagation

FIGURE 4.36: LAPAN-A1 and LAPAN-A2 Inclination, RAAN, and Semiparameter error using Two-Body + J_2 propagation

As depicted in the Table. 4.10, the only ones that were seemingly periodic were the Inclination error and the RAAN error for the LAPAN-A1 and the LAPAN-A2 satellites. As shown also in the Fig. 4.35, the Inclination error does represent some sort of periodical pattern to it. Taking the Fourier analysis for the Inclination error, the number orders that was taken was five (5), due to the fact that the value of R^2 has achieved a satisfactory or convergence value. As for the RAAN error the number of order that was taken was twenty seven (27), due to the fact that the highest value of R^2 were given by the order. As for the Semiparameter error the number of order that was taken was ten (10), due to the fact that the value of R^2 has achieved a satisfactory or convergence value.

$$\begin{aligned} y = \; & a0 + a1\cos(wx) + a2\cos(2wx) \\ & + a3\cos(3wx) + a4\cos(4wx) \\ & + a5\cos(5wx) + b1\sin(wx) \\ & + b2\sin(2wx) + b3\sin(3wx) \\ & + b4\sin(4wx) + b5\sin(5wx) \end{aligned} \tag{4.4}$$

	Values
a0	-5.307368e+00
a1	-9.753827e+00
a2	-6.842628e+00
a3	-3.757338e+00
a4	-1.324095e+00
a5	-2.375144e-01
b1	-3.711659e+00
b2	-4.580396e+00
b3	-2.909188e+00
b4	-1.025766e+00
b5	-1.600369e-01
ω	-4.437255e-04
R^2	0.9975433364084535

Table 4.14: Coefficient, frequency, and R^2 values of the Inclination error Fourier analysis

$$\begin{aligned} y = \; & a0 + a1\cos(wx) + a2\cos(2wx) + a3\cos(3wx) \\ & + \cdots + a25\cos(2wx) + a26\cos(2wx) + a27\cos(2wx) \\ & + b1\sin(wx) + b2\sin(2wx) + b3\sin(3wx) \\ & + \cdots + b25\sin(2wx) + b26\cos(2wx) + b27\sin(2wx) \end{aligned} \tag{4.5}$$

	Values
a0	8.001953e+04
a1	2.597930e+05
a2	8.736742e+04
a3	2.119655e+05
a25	1.269937e+05
a26	2.137609e+05
a27	5.073635e+03
b1	2.586211e+04
b2	1.098418e+05
b3	1.328319e+05
b25	7.102846e+05
b26	9.851845e+02
b27	-2.792145e+04
ω	4.176050e-04
R^2	0.35168788501264137

TABLE 4.15: Coefficient, frequency, and R^2 values of the RAAN error Fourier analysis

$$\begin{aligned} y = \ & a0 + a1\cos(wx) + a10\cos(10wx) + a2\cos(2wx) \\ & + a3\cos(3wx) + a8\cos(8wx) + a9\cos(9wx) \\ & + b1\sin(wx) + b10\sin(10wx) + b2\sin(2wx) \\ & + b3\sin(3wx) + b8\sin(8wx) + b9\sin(9wx) \end{aligned} \tag{4.6}$$

	Values
a0	6.878166e+00
a1	-2.686397e-01
a2	-1.905744e-02
a3	1.591948e-01
a8	-6.105143e-02
a9	1.568347e-02
a10	5.589840e-02
b1	1.874879e-02
b2	1.937725e-01

b3	2.924714e-03
b8	-4.549744e-02
b9	2.305028e-02
b10	-5.776247e-03
ω	2.939877e-03
R^2	0.9632764548800795

TABLE 4.16: Coefficient, frequency, and R^2 values of the Semiparameter error Fourier analysis

From both of the Fig. 4.36(a), Eq. 4.4, and the Table. 4.14, it is clear that by using the Fourier analysis on the error data shows more convergence and shows a higher relation between the error and the propagation period. Showed by the value of R^2 that almost touch the value of 1, or 100% for that matter, a satisfactory relation were achieved.

On the other hand, the Fig. 4.36(b), Eq. 4.5, and the Table. 4.15, it is clear that in comparison to the linear analysis of the error, the Fourier analysis shows a more promising / satisfactory results on the behavior of the error with respect to time. The R^2 value of the data fir shows a relation of 0.35 or 35%.

Lastly, the Fig. 4.36(c), Eq. 4.6, and the Table. **??**, it is clear that by using the Fourier analysis on the error data shows more convergence and shows a higher relation between the error and the propagation period. The R^2 value of the data fir shows a relation of almost 1 or 100%.

CHAPTER 5
SUMMARY, CONCLUSION, RECOMMENDATION

5.1 Summary

Based on what have been shown and described in this thesis, this thesis can be summarized as the following:

1. The author had successfully acquired and extracted the needed reference values from the TLE historical data from the stated website.

2. The extractor and convertor tools that were used for the "actual" data, TLE historical data, worked well. There are a few consideration to be made for the extractor, but is sufficient enough as for the author to conduct this thesis.

3. The author has "successfully" build the conversion tools and also the numerical modeling fo the propagator for this research. The quotation remarks that they are in fact working as they intended to, but there are rooms for improvements in utilizing them.

4. The author has successfully asses how the propagated orbital elements and state vectors behaves over the period of time.

5. The author has successfully compares the deviation or error of values between the propagation values and the actual TLE values.

6. Evaluation of the errors of propagation of each propagation method has been done in the Chapter 4, the orbital elements are relatively has more correlation in comparison to the state vectors, although this is just the matter of relativistic view on both results.

7. The evaluation shows the need of different treatment on how the results are analyzed, whether it is by using the regression analysis or Fourier series.

5.2 Conclusion

Based on what have been shown and described in this thesis, especially in the result of the error analysis, this thesis can be concluded as the following:

1. Generally, based on the equations of regression for all of the propagation methods of each satellites, the values of coefficient shows a promising numbers on how the errors are behaving over the propagation period. But if the author would only look at the equations all by themselves, then the author would be lying to say that they are relatively a neat "analysis". The author need also to consider about each orbital elements and state vectors R^2 values, for it determines how the variables correlates with the propagation time. Turns out that the values of R^2 for almost all of the variable are sadly unsatisfying to see. The enumerated list after this would give a little more detail on the linear regression stand point.

   Two-Body Propagation

   - The semiparameter errors for all of the satellites gave much of linear sense to all of them. From the correlation perspective the value of the R^2, all of them gave 0.99 value. This implies that the errors were analyzed best with this method of analysis.
   - The other one that was best suited with this method of analysis was the Inclination error of LAPAN-A3 satellite. From the correlation perspective the value of the R^2, it gave the value of 0.99.

   Two-Body + J_2 Propagation

   - The semiparameter error of LAPAN-A2 satellite was best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.99.

- The Inclination error of LAPAN-A2 satellite was also best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.99.

SGP4 Propagation

- The Inclination error of LAPAN-A3 satellite was best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.99.

2. As seen not all of the errors were best suited with the linear regression analysis. This is due to some of the graphs exhibited some quasi-periodic pattern to them. To actually see the behavior over time of these quasi-periodic graphs, the author took them to be further analyzed by using Fourier analysis. The R^2 values gave some better values of correlation after they went through this analysis. This gave some satisfactory results in analyzing the errors as well. The enumerated list after this would give a little more detail on the Fourier analysis stand point.

Two-Body Propagation

- The only one that was best fitted with this method of analysis was the Inclination error of LAPAN-A1 satellite. From the correlation perspective the value of the R^2, it gave the value of 0.99.

Two-Body + J_2 Propagation

- The Inclination error of LAPAN-A1 satellite was best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.99.
- The RAAN error of LAPAN-A1 satellite was also best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.57. This might not seem like much of an improvement, but in comparison to the linear regression stand point, this method gave better fiting.

SGP4 Propagation

- The Inclination error of LAPAN-A1 satellite was best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.99.
- The RAAN error of LAPAN-A1 satellite was best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.35. This might not seem like much, but in comparison to the linear regression, this method gave the best fit.
- The semiparameter error of LAPAN-A2 satellite was best fitted with this method of analysis. From the correlation perspective the value of the R^2, it gave the value of 0.96. The semiparameter error was also done with the linear regression, and gave a value of 0.84 of R^2, but as can be seen Fourier analysis gave the best fit.

3. From the additional analysis that were done for the seemingly periodic values of errors in the propagations using the Fourier analysis, it shows that the periodic values are more sufficient to be analyzed using this method of analysis in comparison to the linear regression analysis. This due to the nature of the Fourier analysis that replicates the periodical functions as the sum of simpler trigonometrical functions. Although there are some exceptions to be considered also when using the Fourier analysis.

4. The consideration that can be stated now is that because Fourier analysis is good for periodic functions. Although any functions can virtually be proven by Fourier analysis, but the order used to prove the functions as simple sum of sines and cosines would lead to a very lengthy iteration. in addition, due to the same nature, the analysis would only be accurate for some period of time. Say there is this linear function that is represented with the Fourier analysis, but since there is a need to state the period of the function, the analysis would only be valid for "that" span of period, and then reset back to the initial coefficient value.

5. A line that can be drawn with this analysis is that, Each and every satellites gave different behavior in their state. Different propagation method also gave different behavior on the prediction of the state. Thus the errors would

also be different for each and every propagation and also for each and every satellites. This intrigues a sense that there should be different kind of analysis to be implemented for each and every elements for each and every satellites.

6. Another line that can be drawn from this event is that there might be some uncertainties in the simulation model of the propagation method, rather than the base mathematical model of the propagation. What the author mean is that, since the author propagated the state from the very first value of the historical data and evaluate the values at the same time as the other updates, there has got to be some accumulated errors in the propagation.

5.3 Recommendation

Based on the results of error analysis, the author can draw some lines for future works that can be developed from this research:

1. The mathematical modeling of the two-body and the two-body + J_2 propagation by the author should be robust enough to be used. But, there is still a need to have some consideration on how to use it.

2. The simulation regimes or the propagation regimes that were used in this thesis only takes the very first data of the historical data as the initial state and propagate it through the whole time, which presumably causes some errors to accumulate bigger as the time goes. There might need a consideration to update the initial state of the propagation. What it means is that, after the propagation of the next evaluated time, the value of the initial state would become that of the value in the time of evaluation.

3. In regards to the propagation method, the author uses the time difference of every epochs with respect to the first epoch of the historical data. That might also be the cause of why the error analysis looked the way as depicted in the previous chapter. There should be a consideration on taking the time difference between each epoch rather than what the author did. This method should also be in conjunction with the previous point of recommendation to further reduces of the accumulated errors.

4. As this thesis was intended for LAPAN's satellites use only as the subject of interest, this might be too biased for different kind of propagation methods, let alone the modeling of the simulation of the propagation. There need to be an assessment for other kind of orbit based on the elevation or eccentricity for example. That might get a little confusing, but that way the errors might be analyzed better since there are many samples of satellites, rather than many samples of propagation time.

5. In regards to the error analysis, the use of the two methods that were explained and implemented should selectively be used for each parameters. As proven also by the analysis, the errors gave pattern to them, either periodical patter, linear, or chaotic pattern. Meaning that the analysis cannot be done explicitly by one method only.

6. There should be an improvement of the error analysis regime, especially in the modeling of the fit. In addition there should be other method of error analysis to be taken into considerations. Sure linear regression is one of the most straight forward one to use, but the other kinds should also be used for a more robust result of error behavior of each propagation method.

7. In addition there is a need to consider taking different approach on the method of analysis for each and every parameters that might provide better fitting in comparison to the already used ones in this thesis. e.g., Kalman Filter, Spectral Method, and Machine Learning.

8. There might be some consideration to use the analysis right off the bat after the extraction of the TLE data. This is because the data itself shows some kind of pattern to them. Again, using the Fourier transform for the seemingly periodical values would sufficiently give some neat results. Also the use of linear regression for the more linear patterned values would also give relatively neat predictions to the data. Again these two methods, or maybe with the use of other methods, would be sufficient enough to guess the future state right off the bat.

9. Lastly, for practical recommendation, it is highly recommended to of course use the SGP4 propagation as the method of prediction. But, there is also a

need to use the errors analysis, in the predictions, using selective method of analysis to give an idea regarding the predictions while left in the blindspots of updates.

References

1.6. scipy : high-level scientific computing. (n.d.). https://scipy-lectures.org/intro/scipy.html. (2020-11-08 20:31:29)

About us. (n.d.). https://numpy.org/about/. (2020-11-08 20:08:35)

Anaconda individual edition. (2020). https://docs.anaconda.com/anaconda/. (2020-11-07 13:20:33)

community, T. S. (2020). *scipy.integrate.solve_ivp.* https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html. (2020-11-20 00:22:21)

Curtis, H. D. (2014). *Orbital mechanics for engineering students* (Third Edition ed.). Butterworth-Heinemann.

cysgp4 0.3.3. (n.d.). https://pypi.org/project/cysgp4/. (2020-11-08 21:23:21)

Definition of two-line element set coordinate system. (2011). https://spaceflight.nasa.gov/realdata/sightings/SSapplications/Post/JavaSSOP/SSOP_Help/tle_def.html. (2020-07-06 21:16:31)

Describing orbits. (2018). https://www.faa.gov/about/office_org/headquarters_offices/avs/offices/aam/cami/library/online_libraries/aerospace_medicine/tutorial/media/III.4.1.4_Describing_Orbits.pdf. Retrieved from https://www.faa.gov/about/office_org/headquarters_offices/avs/offices/aam/cami/library/online_libraries/aerospace_medicine/tutorial/media/III.4.1.4_Describing_Orbits.pdf (2020-11-07 22:21:42)

Frost, J. (n.d.). *How to interpret r-squared in regression analysis.* https://statisticsbyjim.com/regression/interpret-r-squared-regression/#:~:text=R%2Dsquared%20is%20a%20goodness,measure%20for%20linear%20regression%20models.&text=R%2Dsquared%20measures%20the%20strength,the%20model%20fits%20the%20data.

Greene, M. R. (2009). Ssc09-x-7 - increasing the accuracy of orbital position information from norad sgp4 using intermittent gps readings - m. greene.

Howell, K. B. (2016). *Principles of fourier analysis.* CRC Press.

Kelso, T. (n.d.-a). *Frequently asked questions: Two-line element set format.* https://celestrak.com/columns/v04n03/. (2020-09-13 18:36:58)

Kelso, T. (n.d.-b). *Orbital coordinate systems, part i.* https://celestrak.com/columns/v02n01/.

Kelso, T. S. (n.d.). *Space surveillance.* https://celestrak.com/columns/v04n01/. (2020-09-13 18:37:27)

Kelso, T. S. (2007). Validation of sgp4 and is-gps-200d against gps precision ephemerides.

Kepler's laws of planetary motion. (2020). https://www.ck12.org/c/physics/keplers-laws-of-planetary-motion/lesson/Kepler\T1\textquoterights-Laws-of-Planetary-Motion-PHYS/. (2020-11-13 16:03:21)

Lapan-a1 / lapan-tubsat. (n.d.). http://pusteksat.lapan.go.id/page/lapan-a1. (2020-08-06 21:16:31)

Lapan-a2 / lapan-orari. (n.d.). http://pusteksat.lapan.go.id/page/lapan-a2. (2020-08-06 21:16:51)

Lapan-a3 / lapan-ipb. (n.d.). http://pusteksat.lapan.go.id/page/lapan-a3. (2020-08-06 21:16:58)

Li, L. C. X.-Z. B. Y.-G. L. . K.-B. (2017). *Orbital data applications for space objects: Conjunction assessment and situation analysis.* Springer.

Linear models. (2020). https://scikit-learn.org/stable/modules/linear_model.html. (2020-12-14 18:20:44)

Matplotlib: Visualization with python. (n.d.). https://matplotlib.org/. (2020-11-08 20:50:12)

McClain, D. A. V. W. D. (2013). *Fundamentals of astrodynamics and applications 4th edition* (Fourth Edition ed.). Microcosm Press.

McKinney, W., & the Pandas Development Team. (2020). *pandas: powerful python data analysis toolkit.*

Miura, N. Z. (2009). Comparison and design of simplified general perturbation models (sgp4) and code for nasa johnson space center, orbital debris program office.

Newton's universal law of gravity. (n.d.). https://sites.google.com/site/oakastroflash/Home/laws-of-physics/newton-s-universal-law-of-gravity. (2020-11-12 17:03:21)

Regression analysis. (n.d.). https://corporatefinanceinstitute.com/resources/knowledge/finance/regression-analysis/.

Satcat boxscore. (n.d.). https://celestrak.com/satcat/boxscore.phphttps://celestrak.com/satcat/boxscore.php. (2020-09-13 19:05:09)

Sejarah lapan. (n.d.). https://www.lapan.go.id/page/sejarah. (2020-08-05 21:57:39)

symfit 0.5.3. (2020). https://pypi.org/project/symfit/.

Taylor, J. R. (1996). *Introduction to error analysis.* University Science Books.

Tombasco, J. (2011). Orbit estimation of geosynchronous objects via ground-based and space-based optical tracking.

Tugas dan fungsi. (n.d.). https://www.lapan.go.id/page/tugas-dan-fungsi. (2020-08-05 21:54:06)

Vallado, D. A. (2006). Revisiting spacetrack report 3: Rev 2.

What is an orbit propagator? (2000). https://satspy.com/orbprop.html. (2020-10-06 02:05:43)

l Varoquaux; Alexandre Gramfort, F. P. G. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research 12 (2011) 2825-2830*.

Appendices

Appendix A: Constants for the Codes

```
"""
This python file contains the necessary constants for astrodynamics
computational tool project
"""

MIU_N = 1.0  # normalized gravitational constant
ER_N = 1.0

# Astrodynamic Constants (JGM-2) in SI-Units

# Geocentric
ER = 6378.1363  # km
TU = 806.81099130673  # s
VU = 7.905366149846  # km/s
MIU = 3.986004415e5  # km3/s2
J2 = 0.0010826269

from datetime import datetime, timedelta
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from Constants import *


def KepEqtnE(M, e):
    """
    Description
    -----------
    This function is intended for calculating the Eccentric Anomaly (E_nu) from Mean Anomaly and Eccentricity

    Parameters
    ----------
    M : Mean Anomaly taken from the TLE.
    e : Eccentricity taken from the TLE.

    Returns
    -------
    E_nu : returns the Eccentric Anomaly with a float type.

    """
    if (M > -np.pi and M < 0) or M > np.pi:
        E0 = M - e
```

```
    else:
        E0 = M + e

    dE = 0.1
    while dE > 1e-8:
        E_nu = E0 + (M - E0 + e * np.sin(E0)) / (1 - e * np.cos(E0))
        dE = np.abs(E_nu - E0)
        E0 = E_nu

    return E_nu


def E2nu(E, e):
    """
    Description
    -----------
    This function is intended for calculating the True Anomaly (nu) from Eccentric Anomaly and Eccentricity

    Parameters
    ----------
    E : Eccentric Anomaly from the KepEqtnE function.
    e : Eccentricity taken from the TLE.

    Returns
    -------
    nu : returns the True Anomaly with a float type.

    """
    sinnu = (np.sin(E) * np.sqrt(1 - e ** 2)) / (1 - e * np.cos(E))
    cosnu = (np.cos(E) - e) / (1 - e * np.cos(E))

    nu = np.arctan2(sinnu, cosnu)

    return nu


def M2nu(M, e):
    """
    Description
    -----------
    This function is intended for calculating the True Anomaly (nu) from Mean Anomaly

    Parameters
    ----------
    M : Mean Anomaly taken from the TLE.
    e : Eccentricity taken from the TLE.

    Returns
    -------
    nu : returns the True Anomaly with a float type.
```

```
    """
    E = KepEqtnE(M, e)
    nu = E2nu(E, e)

    return nu


def testnu(E, e):

    above = 1 + e
    below = 1 - e
    root = np.sqrt(above / below)

    last = E / 2

    half = root * np.tan(last)

    nu = np.arctan(half) * 2

    return nu


def tle_ephemeris(TLE):
    """
    Description
    -----------
    This function is for extracting the TLE ephemeris into a list of COEs of the particular satellite

    Parameters
    ----------
    TLE : a txt file that contains a satellite's TLE ephemeris

    Returns
    -------
    returns a list containing the satellite's COEs (p, e, i, omega, w, nu)

    """
    df = pd.read_table(TLE, header=None).values

    line1 = df[1, 0]
    line2 = df[2, 0]

    year = datetime(int("20" + line1[18:20]), month=1, day=1)
    day = timedelta(float(line1[20:32]))
    epoch = year + day

    time = epoch
    Inclination = float(line2[8:16])
    RAAN = float(line2[17:25])
    Eccentricity = float("." + line2[26:33])
```

```
    AOP = float(line2[34:42])

    MeanAnomaly = float(line2[43:51])

    nu = np.rad2deg(M2nu(MeanAnomaly, Eccentricity))

    n = float(line2[52:63])
    maj_axis = (MIU ** (1 / 3) / ((2 * n * np.pi / 86400) ** (2 / 3))) / ER

    semiparameter = maj_axis * (1 - (Eccentricity ** 2))

    coes_t = [time, semiparameter, Eccentricity, Inclination, RAAN, AOP, nu]

    return coes_t


def tle_ephemerides(TLE):
    """
    Descriptions
    ------------
    This function is for extracting the TLE ephemerides into a list of dictionaries that contains the COEs of the parti

    Parameters
    ----------
    TLE : a txt file that contains a satellite's TLE ephemerides

    Returns
    -------
    Returns a list of dictionaries containing the satellite's COEs (p, e, i, omega, w, nu)
    """
    df = pd.read_table(TLE, header=None).values

    coes = []

    for i in range(df.shape[0]):
        if i % 2 == 0:
            line1 = df[i, 0]
            line2 = df[i + 1, 0]

            year = datetime(int("20" + line1[18:20]), month=1, day=1)
            day = timedelta(float(line1[20:32]))
            epoch = year + day

            x0 = {"Epoch": epoch}
            x1 = {"Inclination": float(line2[8:16])}
            x2 = {"Right Ascension of the Ascending Node": float(line2[17:25])}
            x3 = {"Eccentricity": float("." + line2[26:33])}
            x4 = {"Argument of Perigee": float(line2[34:42])}

            MeanAnomaly = float(line2[43:51])
```

```
            nu = np.rad2deg(M2nu(MeanAnomaly, float("." + line2[26:33])))
            if nu < 0:
                nu = nu + 360
            else:
                nu = nu

            x5 = {"True Anomaly": nu}

            n = float(line2[52:63])
            maj_axis = (MIU ** (1 / 3) / ((2 * n * np.pi / 86400) ** (2 / 3))) / ER

            x6 = {"Semiparameter": (maj_axis * (1 - (float("." + line2[26:33]) ** 2)))}

            coes.append({**x0, **x6, **x3, **x1, **x2, **x4, **x5})

    coes_df = pd.DataFrame(coes)
    coes_df.drop_duplicates(subset=["Epoch"], inplace=True)
    coes_np = coes_df.to_numpy()

    return coes_np

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp
from Constants import ER, ER_N, MIU_N, TU, VU, MIU, J2

# unit vectors
I = np.array([1, 0, 0])
J = np.array([0, 1, 0])
K = np.array([0, 0, 1])


def rv2coe(state):
    """
    Description
    -----------
    This function converts the state vector of a satellite, with respect to a celestial body, to the classical orbital

    Parameters
    ----------
    state : it is an array of state vector or the position vector and the velocity vector of the satellite (rx, ry, rz,

    Returns
    -------
    coe : Returns as a list of orbital elements

        p       = semiparameter
        e       = eccentricity
        i       = inclination
        omega   = right ascension of the ascending node
```

```
        w       = argument of periapsis
        nu      = true anomaly
    """
    rx, ry, rz, vx, vy, vz = state

    R = np.array([rx, ry, rz])
    V = np.array([vx, vy, vz])

    R_mag = np.linalg.norm(R)
    V_mag = np.linalg.norm(V)

    # defining specific angular momentum
    h_vec = np.cross(R, V)
    h_mag = np.linalg.norm(h_vec)

    # defining node vector
    n_vec = np.cross(K, h_vec)
    n_mag = np.linalg.norm(n_vec)

    # defining eccentricity
    e_vec = (1 / MIU_N) * (((V_mag ** 2 - (MIU_N / R_mag)) * R) - ((np.dot(R, V)) * V))
    e_mag = np.linalg.norm(e_vec)

    # defining specific mechanical energy
    xi = ((V_mag ** 2) / 2) - (MIU_N / R_mag)

    # defining the semiparameter and the semimajor axis
    if e_mag != 1:
        a = -(MIU_N / ((2 * xi)))
        p = a * (1 - e_mag ** 2)
    else:
        p = h_mag ** 2 / MIU_N
        a = np.inf

    # defining inclination
    cos_i = h_vec[2] / h_mag
    i = np.arccos(cos_i) * (180 / np.pi)

    # defining the longitude of the ascending node
    cos_omega = (n_vec[0]) / (n_mag)
    omega = np.arccos(cos_omega) * (180 / np.pi)
    if n_vec[1] < 0:
        omega = 360 - omega
    else:
        omega = omega

    # defining the argument of perigee
    cos_w = (np.dot(n_vec, e_vec)) / (n_mag * e_mag)
    w = np.arccos(cos_w) * (180 / np.pi)
    if e_vec[2] < 0.0:
```

```
        w = 360 - w
    else:
        w = w

    # defining the true anomaly
    cos_nu = (np.dot(e_vec, R)) / (e_mag * R_mag)
    nu = np.arccos(cos_nu) * (180 / np.pi)
    if np.dot(R, V) < 0:
        nu = 360 - nu
    else:
        nu = nu

    coe = [p, e_mag, i, omega, w, nu]

    return coe


def coe2rv(coe):
    """
    Description
    -----------
    This function definition converts the classical orbital elemnts to the state vector, with respect to a celestial bo

    Parameters
    ----------
    coe : a list of orbital elements that consists of:

        p       = semiparameter
        e       = eccentricity
        i       = inclination
        omega   = right ascension of the ascending node
        w       = argument of periapsis
        nu      = true anomaly

    Returns
    -------
    state : Returns as a list of state vector
    """
    [p, e, i, omega, w, nu] = coe
    i_rad = np.radians(i)
    omega_rad = np.radians(omega)
    w_rad = np.radians(w)
    nu_rad = np.radians(nu)

    r_p = (p * np.cos(nu_rad)) / (1 + (e * np.cos(nu_rad)))
    r_q = (p * np.sin(nu_rad)) / (1 + (e * np.cos(nu_rad)))
    r_w = 0
    r_pqw = np.array([r_p, r_q, r_w])

    v_p = -(np.sqrt(MIU_N / p)) * (np.sin(nu_rad))
```

```
    v_q = (np.sqrt(MIU_N / p)) * (e + np.cos(nu_rad))
    v_w = 0
    v_pqw = np.array([v_p, v_q, v_w])

    m11 = (np.cos(omega_rad) * np.cos(w_rad)) - (
        np.sin(omega_rad) * np.sin(w_rad) * np.cos(i_rad)
    )
    m12 = (-np.cos(omega_rad) * np.sin(w_rad)) - (
        np.sin(omega_rad) * np.cos(w_rad) * np.cos(i_rad)
    )
    m13 = np.sin(omega_rad) * np.sin(i_rad)

    m21 = (np.sin(omega_rad) * np.cos(w_rad)) + (
        np.cos(omega_rad) * np.sin(w_rad) * np.cos(i_rad)
    )
    m22 = (-np.sin(omega_rad) * np.sin(w_rad)) + (
        np.cos(omega_rad) * np.cos(w_rad) * np.cos(i_rad)
    )
    m23 = -np.cos(omega_rad) * np.sin(i_rad)

    m31 = np.sin(w_rad) * np.sin(i_rad)
    m32 = np.cos(w_rad) * np.sin(i_rad)
    m33 = np.cos(i_rad)

    rot_matrix = np.array([[m11, m12, m13], [m21, m22, m23], [m31, m32, m33]])

    R_ijk = np.dot(rot_matrix, r_pqw)
    V_ijk = np.dot(rot_matrix, v_pqw)

    state = np.hstack([R_ijk, V_ijk])

    return state


def twobody(time, state):
    """
    Description
    -----------
    This function is to find the gradient of two-body poblem that will be used in the solve_ivp solver, of the propagat

    Parameters
    ----------
    time    : time parameter (starting, end) used in the solve_ivp solver
    state   : the state vector (rx,ry,rz,vx,vy,vz) that will be used in the solve_ivp solver

    Returns
    -------
    state_dot : the gradient of the twobody problem to be used in the solve_ivp solver
    """
    rx, ry, rz, vx, vy, vz = state
```

```
    r_vec = np.array([rx, ry, rz])

    r_mag = np.linalg.norm(r_vec)

    ax, ay, az = ((-MIU_N) / (r_mag ** 3)) * r_vec

    return [vx, vy, vz, ax, ay, az]


def twobody_j2(time, state):
    """
    Description
    -----------
    This function is to find the gradient of two-body + J2 poblem that will be used in the solve_ivp solver, of the pro

    Parameters
    ----------
    time     : time parameter (starting, end) used in the solve_ivp solver
    state    : the state vector (rx,ry,rz,vx,vy,vz) that will be used in the solve_ivp solver

    Returns
    -------
    state_dot : the gradient of the twobody problem to be used in the solve_ivp solver

    """
    rx, ry, rz, vx, vy, vz = state

    r_vec = np.array([rx, ry, rz])

    r_mag = np.linalg.norm(r_vec)

    k = 1.5 * MIU_N * J2 * (ER_N / r_mag) ** 2
    ax = -MIU_N * r_vec[0] / r_mag ** 3 * (1 - k * (5 * r_vec[2] ** 2 / r_mag ** 3 - 1))
    ay = -MIU_N * r_vec[1] / r_mag ** 3 * (1 - k * (5 * r_vec[2] ** 2 / r_mag ** 3 - 1))
    az = -MIU_N * r_vec[2] / r_mag ** 3 * (1 - k * (5 * r_vec[2] ** 2 / r_mag ** 3 - 3))

    return (vx, vy, vz, ax, ay, az)


def prop2body(T, Y):
    """
    Descriptions
    ------------
    This function is intended for the propagtion method of the stated mathematical model of the 2B perturbation model /

    Parameters
    ----------
    T : Timestamp (epoch) from the TLE data.
    Y : Initial state vector.
```

```
    Returns
    -------
    Returns an array of propagated state vectors according to timestamps.
    """
    period = np.array([])

    for i in range(len(T)):
        stamp2date = T[i].to_pydatetime()
        date2float = stamp2date.timestamp()
        period = np.append(period, date2float / TU)

    difference = np.array([])

    ts = 0
    tf = period[-1] - period[0]
    p = 1

    while p in range(len(period)):
        haiya = period[p] - period[0]
        difference = np.append(difference, haiya)
        p = p + 1

    list_dt = difference.tolist()
    dt = list(set(list_dt))
    dt.sort()

    atol = 1e-08
    rtol = 1e-13

    sol = solve_ivp(
        twobody, [ts, tf], Y, method="RK45", atol=atol, rtol=rtol, t_eval=dt
    )

    return sol


def propj2(T, Y):
    """
    Descriptions
    ------------
    This function is intended for the propagtion method of the stated mathematical model of the 2BJ2 perturbation model

    Parameters
    ----------
    T : Timestamp (epoch) from the TLE data.
    Y : Initial state vector.

    Returns
    -------
```

```
    Returns an array of propagated state vectors according to timestamps.
    """
    period = np.array([])

    for i in range(len(T)):
        stamp2date = T[i].to_pydatetime()
        date2float = stamp2date.timestamp()
        period = np.append(period, date2float / TU)

    difference = np.array([])
    p = 1
    while p in range(len(period)):
        haiya = period[p] - period[0]
        difference = np.append(difference, haiya)
        p = p + 1

    list_dt = difference.tolist()
    dt = list(set(list_dt))
    dt.sort()

    atol = 1e-08
    rtol = 1e-13
    ts = 0
    tf = period[-1] - period[0]

    sol = solve_ivp(
        twobody_j2, [ts, tf], Y, method="RK45", atol=atol, rtol=rtol, t_eval=dt
    )

    return sol


def coes2rvs(coes):
    """
    Descriptions
    ------------
    Turning an array of list of COEs into an array of list of State Vector

    Parameters
    ----------
    coes : an array of list containg COEs

    Returns
    -------
    Returnin state vectors in a form of array

    """
    coes = coes[:, 1:]
    rvs = np.array([])
```

```
    for q in range(len(coes)):
        tc = coe2rv(coes[q])
        rvs = np.append(rvs, tc)

    return rvs.reshape(-1, 6, order="F")


def rvs2coes(Y):
    """
    Descriptions
    ------------
    Turning an array of list of State Vector into an array of list of COEs

    Parameters
    ----------
    Y : an array of list containg State Vectors

    Returns
    -------
    Returnin COEs in a form of array

    """
    state = Y
    coes = np.array([])

    for q in range(len(state)):
        tc = rv2coe(state[q])
        coes = np.append(coes, tc)

    return coes.reshape(6, -1, order="F")

import numpy as np
import pandas as pd
from numpy import linalg as LA
from Constants import TU, VU, ER


def TLE_COES(tle):
    """
    Parameters
    ----------
    tle : a numpy array of epochs and coes

    Returns
    -------
    tlecoes : a dataframe of epochs and coes
    """
    headers = ["Epoch", "p_i", "e_i", "i_i", "RAAN_i", "AoP_i", "nu_i"]
    contents = tle

    tlecoes = pd.DataFrame(data=contents, columns=headers)
```

```
    tlecoes["p_i"] = tlecoes["p_i"] * ER
    tlecoes[["p_i", "e_i", "i_i", "RAAN_i", "AoP_i", "nu_i"]] = (
        tlecoes[["p_i", "e_i", "i_i", "RAAN_i", "AoP_i", "nu_i"]].astype(float).round(4)
    )

    return tlecoes


def TLE_RVS(tle, tle_rvs):
    """
    Parameters
    ----------
    tle : a numpy array of epochs and coes
    tle_rvs : a numpy array of converted rvs

    Returns
    -------
    tlervs : a dataframe of epochs and rvs
    """
    epochs = tle[:, 0].reshape(-1, 1)

    headers = ["Epoch", "rx_i", "ry_i", "rz_i", "vx_i", "vy_i", "vz_i"]
    contents = np.hstack((epochs, tle_rvs))

    tlervs = pd.DataFrame(data=contents, columns=headers)

    tlervs[["rx_i", "ry_i", "rz_i"]] = (
        (tlervs[["rx_i", "ry_i", "rz_i"]] * ER).astype(float).round(2)
    )
    tlervs[["vx_i", "vy_i", "vz_i"]] = (
        (tlervs[["vx_i", "vy_i", "vz_i"]] * VU).astype(float).round(3)
    )

    return tlervs


def PROP_COES(tle, coes):
    """
    Parameters
    ----------
    tle : a numpy array of epochs and coes
    coes : a numpy array of propagated coes

    Returns
    -------
    propcoes : a dataframe of epochs and propagated coes
    """
    epochs = tle[:, 0].reshape(-1, 1)
    coes_i = tle[0, 1:]
```

```
    coes = np.vstack((coes_i, coes.transpose()))

    headers = ["Epoch", "p_f", "e_f", "i_f", "RAAN_f", "AoP_f", "nu_f"]
    contents = np.hstack((epochs, coes))

    propcoes = pd.DataFrame(data=contents, columns=headers)

    propcoes["p_f"] = propcoes["p_f"] * ER
    propcoes[["p_f", "e_f", "i_f", "RAAN_f", "AoP_f", "nu_f"]] = (
        propcoes[["p_f", "e_f", "i_f", "RAAN_f", "AoP_f", "nu_f"]]
        .astype(float)
        .round(4)
    )

    return propcoes


def PROP_RVS(tle, tle_rvs, rvs):
    """
    Parameters
    ----------
    tle : a numpy array of epochs and coes
    tle_rvs : a numpy array of rvs
    rvs : a numpy array of propagated rvs

    Returns
    -------
    proprvs : a dataframe of epochs and propagated rvs
    """
    epochs = tle[:, 0].reshape(-1, 1)
    rvs_i = tle_rvs[0]

    rvs = np.vstack((rvs_i, rvs.transpose()))

    headers = ["Epoch", "rx_f", "ry_f", "rz_f", "vx_f", "vy_f", "vz_f"]
    contents = np.hstack((epochs, rvs))

    proprvs = pd.DataFrame(data=contents, columns=headers)

    proprvs[["rx_f", "ry_f", "rz_f"]] = (
        (proprvs[["rx_f", "ry_f", "rz_f"]] * ER).astype(float).round(2)
    )
    proprvs[["vx_f", "vy_f", "vz_f"]] = (
        (proprvs[["vx_f", "vy_f", "vz_f"]] * VU).astype(float).round(3)
    )

    return proprvs
```

```python
def ERROR_COES(tle, coes):
    """
    Parameters
    ----------
    tle : a numpy array of epochs and coes
    coes : a numpy array of propagated coes

    Returns
    -------
    errorcoes : a dataframe of coes propagation errors
    """
    epochs = tle[1:, 0].reshape(-1, 1)
    tle = tle[1:, 1:]
    coes = coes.transpose()

    diff = abs(coes - tle)

    headers = ["Epoch", "p_e", "e_e", "i_e", "RAAN_e", "AoP_e", "nu_e"]
    contents = np.hstack((epochs, diff))

    errorcoes = pd.DataFrame(data=contents, columns=headers)

    errorcoes["p_e"] = errorcoes["p_e"] * ER
    errorcoes[["p_e", "e_e", "i_e", "RAAN_e", "AoP_e", "nu_e"]] = (
        errorcoes[["p_e", "e_e", "i_e", "RAAN_e", "AoP_e", "nu_e"]]
        .astype(float)
        .round(4)
    )

    return errorcoes


def ERROR_RVS(tle, tle_rvs, rvs):
    """
    Parameters
    ----------
    tle : a numpy array of epochs
    tle_rvs : a numpy array of epochs and rvs
    coes : a numpy array of propagated rvs

    Returns
    -------
    errorrvs : a dataframe of rvs propagation errors
    """
    epochs = tle[1:, 0].reshape(-1, 1)
    rvs = rvs.transpose()

    diff = abs(rvs - tle_rvs[1:])

    headers = ["Epoch", "rx_e", "ry_e", "rz_e", "vx_e", "vy_e", "vz_e"]
```

```
    contents = np.hstack((epochs, diff))

    errorrvs = pd.DataFrame(data=contents, columns=headers)

    errorrvs[["rx_e", "ry_e", "rz_e"]] = (
        (errorrvs[["rx_e", "ry_e", "rz_e"]] * ER).astype(float).round(2)
    )
    errorrvs[["vx_e", "vy_e", "vz_e"]] = (
        (errorrvs[["vx_e", "vy_e", "vz_e"]] * VU).astype(float).round(3)
    )

    return errorrvs


def ALL_IN1(coes, rvs):
    rvs_drop = rvs.drop(["Epoch"], axis=1)
    tleall = pd.concat([coes, rvs_drop], axis=1)

    return tleall

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.datasets import load_boston
from Main_File import *
from Constants import *
from sgp4 import *


def coes_analysis(reg):
    ylabels_ = [
        "error (km)",
        "error (deg)",
        "error (deg)",
        "error (deg)",
        "error (deg)",
        "error (deg)",
    ]
    titles = [
        "Semiparameter",
        "Eccentricity",
        "Inclination",
        "Right Ascension of the Ascending Node",
        "Argument of Perigee",
        "True Anomaly",
    ]
    x = reg[0]

    n_rows = 2
    n_cols = 3
```

```
    FIG, axs = plt.subplots(n_rows, n_cols, sharex=True, figsize=(18, 10))

    axs[0, 0].scatter(x, reg[1], color="b")
    axs[0, 1].scatter(x, reg[2], color="g")
    axs[0, 2].scatter(x, reg[3], color="r")
    axs[1, 0].scatter(x, reg[4], color="c")
    axs[1, 1].scatter(x, reg[5], color="m")
    axs[1, 2].scatter(x, reg[6], color="y")

    axs[0, 0].plot(x, reg[7].predict(x), color="black", linewidth=3)
    axs[0, 1].plot(x, reg[8].predict(x), color="black", linewidth=3)
    axs[0, 2].plot(x, reg[9].predict(x), color="black", linewidth=3)
    axs[1, 0].plot(x, reg[10].predict(x), color="black", linewidth=3)
    axs[1, 1].plot(x, reg[11].predict(x), color="black", linewidth=3)
    axs[1, 2].plot(x, reg[12].predict(x), color="black", linewidth=3)

    index_coe = 0
    index_ylabel = 0
    for m in range(n_rows):
        for n in range(n_cols):
            axs[m, n].set_title(titles[index_coe])
            axs[m, n].set_xlabel("Time (Day)")
            axs[m, n].set_ylabel(ylabels_[index_ylabel])
            axs[m, n].grid("minor")
            index_coe = index_coe + 1
            index_ylabel = index_ylabel + 1

            for tick in axs[m, n].get_xticklabels():
                tick.set_rotation(45)

    plt.subplots_adjust(wspace=(0.24))
    plt.show()

    return FIG


def rvs_analysis(reg):
    ylabels_ = [
        "error (km)",
        "error (km)",
        "error (km)",
        "error (km/s)",
        "error (km/s)",
        "error (km/s)",
    ]
    titles = ["rx", "ry", "rz", "vx", "vy", "vz"]
    x = reg[0]

    n_rows = 2
    n_cols = 3
```

```
    FIG, axs = plt.subplots(n_rows, n_cols, sharex=True, figsize=(18, 10))

    axs[0, 0].scatter(x, reg[1], color="b")
    axs[0, 1].scatter(x, reg[2], color="g")
    axs[0, 2].scatter(x, reg[3], color="r")
    axs[1, 0].scatter(x, reg[4], color="c")
    axs[1, 1].scatter(x, reg[5], color="m")
    axs[1, 2].scatter(x, reg[6], color="y")

    axs[0, 0].plot(x, reg[7].predict(x), color="black", linewidth=3)
    axs[0, 1].plot(x, reg[8].predict(x), color="black", linewidth=3)
    axs[0, 2].plot(x, reg[9].predict(x), color="black", linewidth=3)
    axs[1, 0].plot(x, reg[10].predict(x), color="black", linewidth=3)
    axs[1, 1].plot(x, reg[11].predict(x), color="black", linewidth=3)
    axs[1, 2].plot(x, reg[12].predict(x), color="black", linewidth=3)

    index_coe = 0
    index_ylabel = 0
    for m in range(n_rows):
        for n in range(n_cols):
            axs[m, n].set_title(titles[index_coe])
            axs[m, n].set_xlabel("Time (Day)")
            axs[m, n].set_ylabel(ylabels_[index_ylabel])
            axs[m, n].grid("minor")
            index_coe = index_coe + 1
            index_ylabel = index_ylabel + 1

            for tick in axs[m, n].get_xticklabels():
                tick.set_rotation(45)

    plt.subplots_adjust(wspace=(0.24))
    plt.show()

    return FIG


def coes_regres(data, time):
    x = time * TU / (86400)
    y = data.values.transpose()

    x = x.reshape(len(x), 1)
    y1 = y[1].reshape(len(y[1]), 1)
    y2 = y[2].reshape(len(y[2]), 1)
    y3 = y[3].reshape(len(y[3]), 1)
    y4 = y[4].reshape(len(y[4]), 1)
    y5 = y[5].reshape(len(y[5]), 1)
    y6 = y[6].reshape(len(y[6]), 1)

    regr1 = LinearRegression().fit(x, y1)
    regr2 = LinearRegression().fit(x, y2)
```

```
    regr3 = LinearRegression().fit(x, y3)
    regr4 = LinearRegression().fit(x, y4)
    regr5 = LinearRegression().fit(x, y5)
    regr6 = LinearRegression().fit(x, y6)

    score1 = regr1.score(x, y1)
    score2 = regr2.score(x, y2)
    score3 = regr3.score(x, y3)
    score4 = regr4.score(x, y4)
    score5 = regr5.score(x, y5)
    score6 = regr6.score(x, y6)

    return (
        x,
        y1,
        y2,
        y3,
        y4,
        y5,
        y6,
        regr1,
        regr2,
        regr3,
        regr4,
        regr5,
        regr6,
        score1,
        score2,
        score3,
        score4,
        score5,
        score6,
    )


def coes_information(q, w, e, r, t, y, u, i, o):
    regr_error = pd.DataFrame()
    regr_error["Legend"] = [
        "p_Error_Intercept",
        "e_Error_Intercept",
        "i_Error_Intercept",
        "RAAN_Error_Intercept",
        "AoP_Error_Intercept",
        "nu_Error_Intercept",
        "p_Error_Coef",
        "e_Error_Coef",
        "i_Error_Coef",
        "RAAN_Error_Coef",
        "AoP_Error_Coef",
        "nu_Error_Coef",
```

```
        "p_Error_Score",
        "e_Error_Score",
        "i_Error_Score",
        "RAAN_Error_Score",
        "AoP_Error_Score",
        "nu_Error_Score",
    ]

    regr_error["A1_2B"] = [
        q[7].intercept_[0],
        q[8].intercept_[0],
        q[9].intercept_[0],
        q[10].intercept_[0],
        q[11].intercept_[0],
        q[12].intercept_[0],
        q[7].coef_[0, 0],
        q[8].coef_[0, 0],
        q[9].coef_[0, 0],
        q[10].coef_[0, 0],
        q[11].coef_[0, 0],
        q[12].coef_[0, 0],
        q[13],
        q[14],
        q[15],
        q[16],
        q[17],
        q[18],
    ]
    regr_error["A2_2B"] = [
        w[7].intercept_[0],
        w[8].intercept_[0],
        w[9].intercept_[0],
        w[10].intercept_[0],
        w[11].intercept_[0],
        w[12].intercept_[0],
        w[7].coef_[0, 0],
        w[8].coef_[0, 0],
        w[9].coef_[0, 0],
        w[10].coef_[0, 0],
        w[11].coef_[0, 0],
        w[12].coef_[0, 0],
        w[13],
        w[14],
        w[15],
        w[16],
        w[17],
        w[18],
    ]
    regr_error["A3_2B"] = [
        e[7].intercept_[0],
```

```
        e[8].intercept_[0],
        e[9].intercept_[0],
        e[10].intercept_[0],
        e[11].intercept_[0],
        e[12].intercept_[0],
        e[7].coef_[0, 0],
        e[8].coef_[0, 0],
        e[9].coef_[0, 0],
        e[10].coef_[0, 0],
        e[11].coef_[0, 0],
        e[12].coef_[0, 0],
        e[13],
        e[14],
        e[15],
        e[16],
        e[17],
        e[18],
    ]
    regr_error["A1_J2"] = [
        r[7].intercept_[0],
        r[8].intercept_[0],
        r[9].intercept_[0],
        r[10].intercept_[0],
        r[11].intercept_[0],
        r[12].intercept_[0],
        r[7].coef_[0, 0],
        r[8].coef_[0, 0],
        r[9].coef_[0, 0],
        r[10].coef_[0, 0],
        r[11].coef_[0, 0],
        r[12].coef_[0, 0],
        r[13],
        r[14],
        r[15],
        r[16],
        r[17],
        r[18],
    ]
    regr_error["A2_J2"] = [
        t[7].intercept_[0],
        t[8].intercept_[0],
        t[9].intercept_[0],
        t[10].intercept_[0],
        t[11].intercept_[0],
        t[12].intercept_[0],
        t[7].coef_[0, 0],
        t[8].coef_[0, 0],
        t[9].coef_[0, 0],
        t[10].coef_[0, 0],
        t[11].coef_[0, 0],
```

```
    t[12].coef_[0, 0],
    t[13],
    t[14],
    t[15],
    t[16],
    t[17],
    t[18],
]
regr_error["A3_J2"] = [
    y[7].intercept_[0],
    y[8].intercept_[0],
    y[9].intercept_[0],
    y[10].intercept_[0],
    y[11].intercept_[0],
    y[12].intercept_[0],
    y[7].coef_[0, 0],
    y[8].coef_[0, 0],
    y[9].coef_[0, 0],
    y[10].coef_[0, 0],
    y[11].coef_[0, 0],
    y[12].coef_[0, 0],
    y[13],
    y[14],
    y[15],
    y[16],
    y[17],
    y[18],
]
regr_error["A1_SG"] = [
    u[7].intercept_[0],
    u[8].intercept_[0],
    u[9].intercept_[0],
    u[10].intercept_[0],
    u[11].intercept_[0],
    u[12].intercept_[0],
    u[7].coef_[0, 0],
    u[8].coef_[0, 0],
    u[9].coef_[0, 0],
    u[10].coef_[0, 0],
    u[11].coef_[0, 0],
    u[12].coef_[0, 0],
    u[13],
    u[14],
    u[15],
    u[16],
    u[17],
    u[18],
]
regr_error["A2_SG"] = [
    i[7].intercept_[0],
```

```
        i[8].intercept_[0],
        i[9].intercept_[0],
        i[10].intercept_[0],
        i[11].intercept_[0],
        i[12].intercept_[0],
        i[7].coef_[0, 0],
        i[8].coef_[0, 0],
        i[9].coef_[0, 0],
        i[10].coef_[0, 0],
        i[11].coef_[0, 0],
        i[12].coef_[0, 0],
        i[13],
        i[14],
        i[15],
        i[16],
        i[17],
        i[18],
    ]
    regr_error["A3_SG"] = [
        o[7].intercept_[0],
        o[8].intercept_[0],
        o[9].intercept_[0],
        o[10].intercept_[0],
        o[11].intercept_[0],
        o[12].intercept_[0],
        o[7].coef_[0, 0],
        o[8].coef_[0, 0],
        o[9].coef_[0, 0],
        o[10].coef_[0, 0],
        o[11].coef_[0, 0],
        o[12].coef_[0, 0],
        o[13],
        o[14],
        o[15],
        o[16],
        o[17],
        o[18],
    ]

    regr_error[
        [
            "A1_2B",
            "A2_2B",
            "A3_2B",
            "A1_J2",
            "A2_J2",
            "A3_J2",
            "A1_SG",
            "A2_SG",
            "A3_SG",
```

```
        ]
    ] = (
        regr_error[
            [
                "A1_2B",
                "A2_2B",
                "A3_2B",
                "A1_J2",
                "A2_J2",
                "A3_J2",
                "A1_SG",
                "A2_SG",
                "A3_SG",
            ]
        ]
        .astype(float)
        .round(4)
    )

    return regr_error


def rvs_regres(data, time):
    x = time * TU / (86400)
    y = data.values.transpose()

    x = x.reshape(len(x), 1)
    y1 = y[7].reshape(len(y[7]), 1)
    y2 = y[8].reshape(len(y[8]), 1)
    y3 = y[9].reshape(len(y[9]), 1)
    y4 = y[10].reshape(len(y[10]), 1)
    y5 = y[11].reshape(len(y[11]), 1)
    y6 = y[12].reshape(len(y[12]), 1)

    regr1 = LinearRegression().fit(x, y1)
    regr2 = LinearRegression().fit(x, y2)
    regr3 = LinearRegression().fit(x, y3)
    regr4 = LinearRegression().fit(x, y4)
    regr5 = LinearRegression().fit(x, y5)
    regr6 = LinearRegression().fit(x, y6)

    score1 = regr1.score(x, y1)
    score2 = regr2.score(x, y2)
    score3 = regr3.score(x, y3)
    score4 = regr4.score(x, y4)
    score5 = regr5.score(x, y5)
    score6 = regr6.score(x, y6)

    return (
        x,
```

```
        y1,
        y2,
        y3,
        y4,
        y5,
        y6,
        regr1,
        regr2,
        regr3,
        regr4,
        regr5,
        regr6,
        score1,
        score2,
        score3,
        score4,
        score5,
        score6,
    )


def rvs_information(q, w, e, r, t, y, u, i, o):
    regr_error = pd.DataFrame()
    regr_error["Legend"] = [
        "rx_Error_Inter",
        "ry_Error_Inter",
        "rz_Error_Inter",
        "vx_Error_Inter",
        "vy_Error_Inter",
        "vz_Error_Inter",
        "rx_Error_Coeff",
        "ry_Error_Coeff",
        "rz_Error_Coeff",
        "vx_Error_Coeff",
        "vy_Error_Coeff",
        "vz_Error_Coeff",
        "rx_Error_Score",
        "ry_Error_Score",
        "rz_Error_Score",
        "vx_Error_Score",
        "vy_Error_Score",
        "vz_Error_Score",
    ]

    regr_error["A1_2B"] = [
        q[7].intercept_[0],
        q[8].intercept_[0],
        q[9].intercept_[0],
        q[10].intercept_[0],
        q[11].intercept_[0],
```

```
        q[12].intercept_[0],
        q[7].coef_[0, 0],
        q[8].coef_[0, 0],
        q[9].coef_[0, 0],
        q[10].coef_[0, 0],
        q[11].coef_[0, 0],
        q[12].coef_[0, 0],
        q[13],
        q[14],
        q[15],
        q[16],
        q[17],
        q[18],
    ]
    regr_error["A2_2B"] = [
        w[7].intercept_[0],
        w[8].intercept_[0],
        w[9].intercept_[0],
        w[10].intercept_[0],
        w[11].intercept_[0],
        w[12].intercept_[0],
        w[7].coef_[0, 0],
        w[8].coef_[0, 0],
        w[9].coef_[0, 0],
        w[10].coef_[0, 0],
        w[11].coef_[0, 0],
        w[12].coef_[0, 0],
        w[13],
        w[14],
        w[15],
        w[16],
        w[17],
        w[18],
    ]
    regr_error["A3_2B"] = [
        e[7].intercept_[0],
        e[8].intercept_[0],
        e[9].intercept_[0],
        e[10].intercept_[0],
        e[11].intercept_[0],
        e[12].intercept_[0],
        e[7].coef_[0, 0],
        e[8].coef_[0, 0],
        e[9].coef_[0, 0],
        e[10].coef_[0, 0],
        e[11].coef_[0, 0],
        e[12].coef_[0, 0],
        e[13],
        e[14],
        e[15],
```

```
        e[16],
        e[17],
        e[18],
    ]
    regr_error["A1_J2"] = [
        r[7].intercept_[0],
        r[8].intercept_[0],
        r[9].intercept_[0],
        r[10].intercept_[0],
        r[11].intercept_[0],
        r[12].intercept_[0],
        r[7].coef_[0, 0],
        r[8].coef_[0, 0],
        r[9].coef_[0, 0],
        r[10].coef_[0, 0],
        r[11].coef_[0, 0],
        r[12].coef_[0, 0],
        r[13],
        r[14],
        r[15],
        r[16],
        r[17],
        r[18],
    ]
    regr_error["A2_J2"] = [
        t[7].intercept_[0],
        t[8].intercept_[0],
        t[9].intercept_[0],
        t[10].intercept_[0],
        t[11].intercept_[0],
        t[12].intercept_[0],
        t[7].coef_[0, 0],
        t[8].coef_[0, 0],
        t[9].coef_[0, 0],
        t[10].coef_[0, 0],
        t[11].coef_[0, 0],
        r[12].coef_[0, 0],
        t[13],
        t[14],
        t[15],
        t[16],
        t[17],
        t[18],
    ]
    regr_error["A3_J2"] = [
        y[7].intercept_[0],
        y[8].intercept_[0],
        y[9].intercept_[0],
        y[10].intercept_[0],
        y[11].intercept_[0],
```

```
        y[12].intercept_[0],
        y[7].coef_[0, 0],
        y[8].coef_[0, 0],
        y[9].coef_[0, 0],
        y[10].coef_[0, 0],
        y[11].coef_[0, 0],
        r[12].coef_[0, 0],
        y[13],
        y[14],
        y[15],
        y[16],
        y[17],
        y[18],
    ]
    regr_error["A1_SG"] = [
        u[7].intercept_[0],
        u[8].intercept_[0],
        u[9].intercept_[0],
        u[10].intercept_[0],
        u[11].intercept_[0],
        u[12].intercept_[0],
        u[7].coef_[0, 0],
        u[8].coef_[0, 0],
        u[9].coef_[0, 0],
        u[10].coef_[0, 0],
        u[11].coef_[0, 0],
        u[12].coef_[0, 0],
        u[13],
        u[14],
        u[15],
        u[16],
        u[17],
        u[18],
    ]
    regr_error["A2_SG"] = [
        i[7].intercept_[0],
        i[8].intercept_[0],
        i[9].intercept_[0],
        i[10].intercept_[0],
        i[11].intercept_[0],
        i[12].intercept_[0],
        i[7].coef_[0, 0],
        i[8].coef_[0, 0],
        i[9].coef_[0, 0],
        i[10].coef_[0, 0],
        i[11].coef_[0, 0],
        i[12].coef_[0, 0],
        i[13],
        i[14],
        i[15],
```

```
    i[16],
    i[17],
    i[18],
]
regr_error["A3_SG"] = [
    o[7].intercept_[0],
    o[8].intercept_[0],
    o[9].intercept_[0],
    o[10].intercept_[0],
    o[11].intercept_[0],
    o[12].intercept_[0],
    o[7].coef_[0, 0],
    o[8].coef_[0, 0],
    o[9].coef_[0, 0],
    o[10].coef_[0, 0],
    o[11].coef_[0, 0],
    o[12].coef_[0, 0],
    o[13],
    o[14],
    o[15],
    o[16],
    o[17],
    o[18],
]

regr_error[
    [
        "A1_2B",
        "A2_2B",
        "A3_2B",
        "A1_J2",
        "A2_J2",
        "A3_J2",
        "A1_SG",
        "A2_SG",
        "A3_SG",
    ]
] = (
    regr_error[
        [
            "A1_2B",
            "A2_2B",
            "A3_2B",
            "A1_J2",
            "A2_J2",
            "A3_J2",
            "A1_SG",
            "A2_SG",
            "A3_SG",
        ]
```

```
        ]
        .astype(float)
        .round(4)
    )

    return regr_error

import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
from Constants import TU, ER, VU
from pandas.plotting import register_matplotlib_converters

register_matplotlib_converters()


def plot_inplane(sol):
    """
    Description
    -----------
    This function is used to plot the propagated positions of the satellite in the [X,Y] plane.

    Parameters
    ----------
    sol : it is the solver of the twobody gradient, the X and Y coordinates are needed.

    Returns
    -------
    None.
    """
    y_states = sol

    xs = y_states[0, :] * ER
    ys = y_states[1, :] * ER

    fig, ax = plt.subplots(figsize=(9, 9))
    ax.plot(xs, ys, ".")
    ax.set(xlabel="x (KM)", ylabel="y (KM)", aspect="equal")
    plt.grid("major")

    plt.show()

    return fig


def plot_outplane(sol):
    """
    Description
    -----------
    This function is used to plot the propagated positions of the satellite in the [Y,Z] plane.
```

```
    Parameters
    ----------
    sol : it is the solver of the twobody gradient, the Y and Z coordinates are needed.

    Returns
    -------
    None.
    """
    y_states = sol

    ys = y_states[1, :] * ER
    zs = y_states[2, :] * ER

    fig, ax = plt.subplots(figsize=(9, 9))
    ax.plot(ys, zs, ".")
    ax.set(xlabel="y (KM)", ylabel="z (KM)", aspect="equal")
    plt.grid("major")

    plt.show()

    return fig


def plot_3d(sol):
    """
    Description
    -----------
    This function is used to plot the propagated positions of the satellite in a 3D space.

    Parameters
    ----------
    sol : it is the solver of the twobody gradient, the X, Y, and Z coordinates are needed.

    Returns
    -------
    None.
    """
    y_states = sol

    xs = y_states[0, :] * ER
    ys = y_states[1, :] * ER
    zs = y_states[2, :] * ER

    fig = plt.figure(figsize=(10, 6))
    ax = Axes3D(fig)

    ax.plot(xs, ys, zs, ".")
    ax.set(xlabel="x (KM)", ylabel="y (KM)", zlabel="z (KM)")
    plt.show()
```

```
    plt.show()

    return fig


def plot_coes(coes_array):
    """
    Plot COEs from array parsed from TLE

    Keyword Arguments:
    coes_array -- List of COEs from TLE
    """
    ts = coes_array[:, 0]

    ylabels_ = [
        "Semiparameter (km)",
        "Eccentricity (deg)",
        "Inclination (deg)",
        "RAAN (deg)",
        "AoP (deg)",
        "True Anomaly (deg)",
    ]
    titles = [
        "Semiparameter",
        "Eccentricity",
        "Inclination",
        "Right Ascension of the Ascending Node",
        "Argument of Perigee",
        "True Anomaly",
    ]

    n_rows = 2
    n_cols = 3
    FIG, axs = plt.subplots(n_rows, n_cols, sharex=True, figsize=(18, 10))

    axs[0, 0].plot(ts, coes_array[:, 1] * ER, ".", color="b")
    axs[0, 1].plot(ts, coes_array[:, 2], ".", color="g")
    axs[0, 2].plot(ts, coes_array[:, 3], ".", color="r")
    axs[1, 0].plot(ts, coes_array[:, 4], ".", color="c")
    axs[1, 1].plot(ts, coes_array[:, 5], ".", color="m")
    axs[1, 2].plot(ts, coes_array[:, 6], ".", color="y")

    index_coe = 0
    index_ylabel = 0
    for m in range(n_rows):
        for n in range(n_cols):
            axs[m, n].set_title(titles[index_coe])
            axs[m, n].set_xlabel("Time (Day)")
            axs[m, n].set_ylabel(ylabels_[index_ylabel])
            # axs[m, n].set_ylim(ylims[index_coe])
```

```
            axs[m, n].grid("minor")
            index_coe = index_coe + 1
            index_ylabel = index_ylabel + 1

            for tick in axs[m, n].get_xticklabels():
                tick.set_rotation(45)

    plt.subplots_adjust(wspace=(0.3))
    plt.show()

    return FIG


def Coes_plot_2D(coes_array, solver):
    """
    Description
    -----------
    This function is intended for plotting all of the propagated COEs.
    All of which are separated according to its own elements though time.

    Parameters
    ----------
    coes_array  : it is the COEs that was cnverted from the propagated state vectors.
    solver      : it is the solver of the twobody gradient, the only thing that is used in this function is the solver.

    Returns
    -------
    None.
    """
    ts = solver

    ylabels_ = [
        "Semiparameter (km)",
        "Eccentricity (deg)",
        "Inclination (deg)",
        "RAAN (deg)",
        "AoP (deg)",
        "True Anomaly (deg)",
    ]
    titles = [
        "Semiparameter",
        "Eccentricity",
        "Inclination",
        "Right Ascension of the Ascending Node",
        "Argument of Perigee",
        "True Anomaly",
    ]

    mins_maxs = []
    for row in coes_array:
```

```
        min = row.min()
        max = row.max()
        min_max = (min, max)
        mins_maxs.append(min_max)

    # Setting ylim values for the matplotlib
    p_ylim = (0, (mins_maxs[0][1] + 2) * ER)
    ecc_ylim = (0, mins_maxs[1][1] + 0.1)
    inc_ylim = tuple(np.add(mins_maxs[2], (-20, 20)))
    Omeg_ylim = tuple(np.add(mins_maxs[3], (-20, 20)))
    omeg_ylim = tuple(np.add(mins_maxs[4], (-20, 20)))
    nu_ylim = (-5, 365)

    ylims = [p_ylim, ecc_ylim, inc_ylim, Omeg_ylim, omeg_ylim, nu_ylim]

    n_rows = 2
    n_cols = 3
    FIG, axs = plt.subplots(n_rows, n_cols, sharex=True, figsize=(18, 10))

    axs[0, 0].plot((ts * TU / 86400), coes_array[0, :] * ER, ".", color="b")
    axs[0, 1].plot((ts * TU / 86400), coes_array[1, :], ".", color="g")
    axs[0, 2].plot((ts * TU / 86400), coes_array[2, :], ".", color="r")
    axs[1, 0].plot((ts * TU / 86400), coes_array[3, :], ".", color="c")
    axs[1, 1].plot((ts * TU / 86400), coes_array[4, :], ".", color="m")
    axs[1, 2].plot((ts * TU / 86400), coes_array[5, :], ".", color="y")

    index_coe = 0
    index_ylabel = 0
    for m in range(n_rows):
        for n in range(n_cols):
            axs[m, n].set_title(titles[index_coe])
            axs[m, n].set_xlabel("Time (Day)")
            axs[m, n].set_ylabel(ylabels_[index_ylabel])
            axs[m, n].set_ylim(ylims[index_coe])
            axs[m, n].grid("minor")
            index_coe = index_coe + 1
            index_ylabel = index_ylabel + 1

            for tick in axs[m, n].get_xticklabels():
                tick.set_rotation(45)

    plt.subplots_adjust(wspace=(0.3))
    plt.show()

    return FIG


def plot_rvs(coes_array, rvs):
    ts = coes_array[:, 0]
```

```
    ylabels_ = ["rx (km)", "ry (km)", "rz (km)", "vx (km/s)", "vy (km/s)", "vz (km/s)"]
    titles = [
        "X position",
        "Y position",
        "Z position",
        "X velocity",
        "Y velocity",
        "Z velocity",
    ]

    n_rows = 2
    n_cols = 3
    FIG, axs = plt.subplots(n_rows, n_cols, sharex=True, figsize=(18, 10))

    axs[0, 0].plot(ts, rvs[:, 0] * ER / 1000, ".", color="b")
    axs[0, 1].plot(ts, rvs[:, 1] * ER / 1000, ".", color="g")
    axs[0, 2].plot(ts, rvs[:, 2] * ER / 1000, ".", color="r")
    axs[1, 0].plot(ts, rvs[:, 3] * VU, ".", color="c")
    axs[1, 1].plot(ts, rvs[:, 4] * VU, ".", color="m")
    axs[1, 2].plot(ts, rvs[:, 5] * VU, ".", color="y")

    index_rv = 0
    index_ylabel = 0
    for m in range(n_rows):
        for n in range(n_cols):
            axs[m, n].set_title(titles[index_rv])
            axs[m, n].set_xlabel("Time (Day)")
            axs[m, n].set_ylabel(ylabels_[index_ylabel])
            axs[m, n].grid("minor")
            index_coe = index_rv + 1
            index_ylabel = index_ylabel + 1

            for tick in axs[m, n].get_xticklabels():
                tick.set_rotation(45)

    plt.subplots_adjust(wspace=(0.3))
    plt.show()

    return FIG

import pandas as pd
import numpy as np
from numpy import savetx
from scipy.integrate import solve_ivp
from OrbitalElements import *
from TLE_Extractor import *
from Constants import *
from Plot_Func import *
from DataFrame import *

"""defining the LAPAN satellites ephemerides"""
```

```
LAPAN_A1_txt = "from launch/A1.txt"
LAPAN_A2_txt = "from launch/A2.txt"
LAPAN_A3_txt = "from launch/A3.txt"

A1DF = pd.read_table(LAPAN_A1_txt, header=None).values
A2DF = pd.read_table(LAPAN_A2_txt, header=None).values
A3DF = pd.read_table(LAPAN_A3_txt, header=None).values

"""extracting the COEs from the defined ephemerides"""
A1_ephemerides = tle_ephemerides(LAPAN_A1_txt)
A2_ephemerides = tle_ephemerides(LAPAN_A2_txt)
A3_ephemerides = tle_ephemerides(LAPAN_A3_txt)

"""Converting the ephemerides into state vectors"""
A1_RVs = coes2rvs(A1_ephemerides)
A2_RVs = coes2rvs(A2_ephemerides)
A3_RVs = coes2rvs(A3_ephemerides)

"""Taking the first state vector of each ephemerides"""
coe_rv_a1 = coe2rv(A1_ephemerides[0][1:])
coe_rv_a2 = coe2rv(A2_ephemerides[0][1:])
coe_rv_a3 = coe2rv(A3_ephemerides[0][1:])

"""Propagating LAPAN-A1/A2/A3 using solve_ivp then converting the RVs into COEs"""
# LAPAN-A1
solver1 = prop2body(A1_ephemerides[:, 0], coe_rv_a1)
prop_coes_A1 = rvs2coes(solver1.y.transpose())

# LAPAN-A2
solver2 = prop2body(A2_ephemerides[:, 0], coe_rv_a2)
prop_coes_A2 = rvs2coes(solver2.y.transpose())

# LAPAN-A3
solver3 = prop2body(A3_ephemerides[:, 0], coe_rv_a3)
prop_coes_A3 = rvs2coes(solver3.y.transpose())

"""Propagating LAPAN-A1/A2/A3 using solve_ivp then converting the RVs into COEs"""
# LAPAN-A1
solver1_J2 = propj2(A1_ephemerides[:, 0], coe_rv_a1)
prop_coes_A1_J2 = rvs2coes(solver1_J2.y.transpose())

# LAPAN-A2
solver2_J2 = propj2(A2_ephemerides[:, 0], coe_rv_a2)
prop_coes_A2_J2 = rvs2coes(solver2_J2.y.transpose())

# LAPAN-A3
solver3_J2 = propj2(A3_ephemerides[:, 0], coe_rv_a3)
prop_coes_A3_J2 = rvs2coes(solver3_J2.y.transpose())

################################################################################
```

```
#######################Below here are for convenience################################
####################################################################################

"""saving the propagation results for convenience"""
np.savetxt("A1_prop_t.csv", solver1.t, delimiter=",")
np.savetxt("A2_prop_t.csv", solver2.t, delimiter=",")
np.savetxt("A3_prop_t.csv", solver3.t, delimiter=",")

np.savetxt("A1_2B_RVs.csv", solver1.y, delimiter=",")
np.savetxt("A2_2B_RVs.csv", solver2.y, delimiter=",")
np.savetxt("A3_2B_RVs.csv", solver3.y, delimiter=",")
np.savetxt("A1_2B_Coes.csv", prop_coes_A1, delimiter=",")
np.savetxt("A2_2B_Coes.csv", prop_coes_A2, delimiter=",")
np.savetxt("A3_2B_Coes.csv", prop_coes_A3, delimiter=",")

np.savetxt("A1_J2_RVs.csv", solver1_J2.y, delimiter=",")
np.savetxt("A2_J2_RVs.csv", solver2_J2.y, delimiter=",")
np.savetxt("A3_J2_RVs.csv", solver3_J2.y, delimiter=",")
np.savetxt("A1_J2_Coes.csv", prop_coes_A1_J2, delimiter=",")
np.savetxt("A2_J2_Coes.csv", prop_coes_A2_J2, delimiter=",")
np.savetxt("A3_J2_Coes.csv", prop_coes_A3_J2, delimiter=",")

"""Aftermath for calling the propagated results"""
A1_t = np.genfromtxt("BuildingBlock/A1/A1_prop_t.csv", delimiter=",")
A2_t = np.genfromtxt("BuildingBlock/A2/A2_prop_t.csv", delimiter=",")
A3_t = np.genfromtxt("BuildingBlock/A3/A3_prop_t.csv", delimiter=",")

A1_2BRVs = np.genfromtxt("BuildingBlock/A1/A1_2B_RVs.csv", delimiter=",")
A2_2BRVs = np.genfromtxt("BuildingBlock/A2/A2_2B_RVs.csv", delimiter=",")
A3_2BRVs = np.genfromtxt("BuildingBlock/A3/A3_2B_RVs.csv", delimiter=",")
A1_2BCOEs = np.genfromtxt("BuildingBlock/A1/A1_2B_Coes.csv", delimiter=",")
A2_2BCOEs = np.genfromtxt("BuildingBlock/A2/A2_2B_Coes.csv", delimiter=",")
A3_2BCOEs = np.genfromtxt("BuildingBlock/A3/A3_2B_Coes.csv", delimiter=",")


A1_J2RVs = np.genfromtxt("BuildingBlock/A1/A1_J2_RVs.csv", delimiter=",")
A2_J2RVs = np.genfromtxt("BuildingBlock/A2/A2_J2_RVs.csv", delimiter=",")
A3_J2RVs = np.genfromtxt("BuildingBlock/A3/A3_J2_RVs.csv", delimiter=",")
A1_J2COEs = np.genfromtxt("BuildingBlock/A1/A1_J2_Coes.csv", delimiter=",")
A2_J2COEs = np.genfromtxt("BuildingBlock/A2/A2_J2_Coes.csv", delimiter=",")
A3_J2COEs = np.genfromtxt("BuildingBlock/A3/A3_J2_Coes.csv", delimiter=",")

from Main_File import *

# """TLE"""
# A1_inplane = plot_inplane(A1_RVs.transpose())
# A1_outplane = plot_outplane(A1_RVs.transpose())
# A1_3D = plot_3d(A1_RVs.transpose())
# TLE_plot_A1 = plot_coes(A1_ephemerides)

# A2_inplane = plot_inplane(A2_RVs.transpose())
```

```
# A2_outplane = plot_outplane(A2_RVs.transpose())
# A2_3D = plot_3d(A2_RVs.transpose())
# TLE_plot_A2 = plot_coes(A2_ephemerides)

# A3_inplane = plot_inplane(A3_RVs.transpose())
# A3_outplane = plot_outplane(A3_RVs.transpose())
# A3_3D = plot_3d(A3_RVs.transpose())
# TLE_plot_A3 = plot_coes(A3_ephemerides)

# A1_RVS_plot = plot_rvs(A1_ephemerides, A1_RVs)
# A2_RVS_plot = plot_rvs(A2_ephemerides, A2_RVs)
# A3_RVS_plot = plot_rvs(A3_ephemerides, A3_RVs)

# TLE_plot_A1.savefig("LAPAN-A1 COEs from TLE datasets.png", dpi=600)
# TLE_plot_A2.savefig("LAPAN-A2 COEs from TLE datasets.png", dpi=600)
# TLE_plot_A3.savefig("LAPAN-A3 COEs from TLE datasets.png", dpi=600)

# A1_inplane.savefig("LAPAN-A1_X&Y-Plane_Position.png", dpi=600)
# A1_outplane.savefig("LAPAN-A1_Y&Z-Plane_Position.png", dpi=600)
# A1_3D.savefig("LAPAN-A1_3D_Position.png", dpi=600)

# A2_inplane.savefig("LAPAN-A2_X&Y-Plane_Position.png", dpi=600)
# A2_outplane.savefig("LAPAN-A2_Y&Z-Plane_Position.png", dpi=600)
# A2_3D.savefig("LAPAN-A2_3D_Position.png", dpi=600)

# A3_inplane.savefig("LAPAN-A3_X&Y-Plane_Position.png", dpi=600)
# A3_outplane.savefig("LAPAN-A3_Y&Z-Plane_Position.png", dpi=600)
# A3_3D.savefig("LAPAN-A3_3D_Position.png", dpi=600)

# A1_RVS_plot.savefig("LAPAN-A1_State_Vector.png", dpi = 600)
# A2_RVS_plot.savefig("LAPAN-A2_State_Vector.png", dpi = 600)
# A3_RVS_plot.savefig("LAPAN-A3_State_Vector.png", dpi = 600)

# """2Body"""
# # LAPAN-A1
# inplane_A1 = plot_inplane(A1_2BRVs)
# outplane_A1 = plot_outplane(A1_2BRVs)
# plot3D_A1 = plot_3d(A1_2BRVs)
# coesplot_A1 = Coes_plot_2D(A1_2BCOEs, A1_t)

# # LAPAN-A2
# inplane_A2 = plot_inplane(A2_2BRVs)
# outplane_A2 = plot_outplane(A2_2BRVs)
# plot3D_A2 = plot_3d(A2_2BRVs)
# coesplot_A2 = Coes_plot_2D(A2_2BCOEs, A2_t)

# # LAPAN-A3
# inplane_A3 = plot_inplane(A3_2BRVs)
# outplane_A3 = plot_outplane(A3_2BRVs)
# plot3D_A3 = plot_3d(A3_2BRVs)
```

```
# coesplot_A3 = Coes_plot_2D(A3_2BCOEs, A3_t)

# # LAPAN-A1
# inplane_A1.savefig("LAPAN-A1_X&Y-Plane_Position-Two-Body.png", dpi = 600)
# outplane_A1.savefig("LAPAN-A1_Y&Z-Plane_Position-Two-Body.png", dpi = 600)
# plot3D_A1.savefig("LAPAN-A1_3D_Position-Two-Body.png", dpi = 600)
# coesplot_A1.savefig("LAPAN-A1_1Day_Propagated-COEs-Two-Body.png", dpi = 600)

# # LAPAN-A2
# inplane_A2.savefig("LAPAN-A2_X&Y-Plane_Position-Two-Body.png", dpi = 600)
# outplane_A2.savefig("LAPAN-A2_Y&Z-Plane_Position-Two-Body.png", dpi = 600)
# plot3D_A2.savefig("LAPAN-A2_3D_Position-Two-Body.png", dpi = 600)
# coesplot_A2.savefig("LAPAN-A2_1Day_Propagated-COEs-Two-Body.png", dpi = 600)

# # LAPAN-A3
# inplane_A3.savefig("LAPAN-A3_X&Y-Plane_Position-Two-Body.png", dpi = 600)
# outplane_A3.savefig("LAPAN-A3_Y&Z-Plane_Position-Two-Body.png", dpi = 600)
# plot3D_A3.savefig("LAPAN-A3_3D_Position-Two-Body.png", dpi = 600)
# coesplot_A3.savefig("LAPAN-A3_1Day_Propagated-COEs-Two-Body.png", dpi = 600)

# """2Body+J2"""
# # LAPAN-A1
# inplane_A1_J2 = plot_inplane(A1_J2RVs)
# outplane_A1_J2 = plot_outplane(A1_J2RVs)
# plot3D_A1_J2 = plot_3d(A1_J2RVs)
# coesplot_A1_J2 = Coes_plot_2D(A1_J2COEs, A1_t)

# # LAPAN-A2
# inplane_A2_J2 = plot_inplane(A2_J2RVs)
# outplane_A2_J2 = plot_outplane(A2_J2RVs)
# plot3D_A2_J2 = plot_3d(A2_J2RVs)
# coesplot_A2_J2 = Coes_plot_2D(A2_J2COEs, A2_t)

# # LAPAN-A3
# inplane_A3_J2 = plot_inplane(A3_J2RVs)
# outplane_A3_J2 = plot_outplane(A3_J2RVs)
# plot3D_A3_J2 = plot_3d(A3_J2RVs)
# coesplot_A3_J2 = Coes_plot_2D(A3_J2COEs, A3_t)

# # LAPAN-A1
# inplane_A1_J2.savefig("LAPAN-A1_X&Y-Plane_Position-J2.png", dpi = 600)
# outplane_A1_J2.savefig("LAPAN-A1_Y&Z-Plane_Position-J2.png", dpi = 600)
# plot3D_A1_J2.savefig("LAPAN-A1_3D_Position-J2.png", dpi = 600)
# coesplot_A1_J2.savefig("LAPAN-A1_Propagated-COEs-J2.png", dpi = 600)

# # LAPAN-A2
# inplane_A2_J2.savefig("LAPAN-A2_X&Y-Plane_Position-J2.png", dpi = 600)
# outplane_A2_J2.savefig("LAPAN-A2_Y&Z-Plane_Position-J2.png", dpi = 600)
# plot3D_A2_J2.savefig("LAPAN-A2_3D_Position-J2.png", dpi = 600)
# coesplot_A2_J2.savefig("LAPAN-A2_Propagated-COEs-J2.png", dpi = 600)
```

```python
# # LAPAN-A3
# inplane_A3_J2.savefig("LAPAN-A3_X&Y-Plane_Position-J2.png", dpi = 600)
# outplane_A3_J2.savefig("LAPAN-A3_Y&Z-Plane_Position-J2.png", dpi = 600)
# plot3D_A3_J2.savefig("LAPAN-A3_3D_Position-J2.png", dpi = 600)
# coesplot_A3_J2.savefig("LAPAN-A3_Propagated-COEs-J2.png", dpi = 600)

from DataFrame import *
from Main_File import *

"""TWO-BODY"""
A1_2Body_RVS_DF = PROP_RVS(A1_ephemerides, A1_RVs, A1_2BRVs)
A1_2Body_COES_DF = PROP_COES(A1_ephemerides, A1_2BCOEs)

A2_2Body_RVS_DF = PROP_RVS(A2_ephemerides, A2_RVs, A2_2BRVs)
A2_2Body_COES_DF = PROP_COES(A2_ephemerides, A2_2BCOEs)

A3_2Body_RVS_DF = PROP_RVS(A3_ephemerides, A3_RVs, A3_2BRVs)
A3_2Body_COES_DF = PROP_COES(A3_ephemerides, A3_2BCOEs)

"""Error Two-Body"""
A1_2Body_RVS_ERROR = ERROR_RVS(A1_ephemerides, A1_RVs, A1_2BRVs)
A1_2Body_COES_ERROR = ERROR_COES(A1_ephemerides, A1_2BCOEs)

A2_2Body_RVS_ERROR = ERROR_RVS(A2_ephemerides, A2_RVs, A2_2BRVs)
A2_2Body_COES_ERROR = ERROR_COES(A2_ephemerides, A2_2BCOEs)

A3_2Body_RVS_ERROR = ERROR_RVS(A3_ephemerides, A3_RVs, A3_2BRVs)
A3_2Body_COES_ERROR = ERROR_COES(A3_ephemerides, A3_2BCOEs)

"""All In One Two-Body"""
A1_2Body = ALL_IN1(A1_2Body_COES_DF, A1_2Body_RVS_DF)
A2_2Body = ALL_IN1(A2_2Body_COES_DF, A2_2Body_RVS_DF)
A3_2Body = ALL_IN1(A3_2Body_COES_DF, A3_2Body_RVS_DF)

A1_2Body_Error = ALL_IN1(A1_2Body_COES_ERROR, A1_2Body_RVS_ERROR)
A2_2Body_Error = ALL_IN1(A2_2Body_COES_ERROR, A2_2Body_RVS_ERROR)
A3_2Body_Error = ALL_IN1(A3_2Body_COES_ERROR, A3_2Body_RVS_ERROR)

################################################################################

"""TWO-BODY + J2"""
A1_J2_RVS_DF = PROP_RVS(A1_ephemerides, A1_RVs, A1_J2RVs)
A1_J2_COES_DF = PROP_COES(A1_ephemerides, A1_J2COEs)

A2_J2_RVS_DF = PROP_RVS(A2_ephemerides, A2_RVs, A2_J2RVs)
A2_J2_COES_DF = PROP_COES(A2_ephemerides, A2_J2COEs)

A3_J2_RVS_DF = PROP_RVS(A3_ephemerides, A3_RVs, A3_J2RVs)
A3_J2_COES_DF = PROP_COES(A3_ephemerides, A3_J2COEs)
```

```
"""Error Two-Body + J2"""
A1_J2_RVS_ERROR = ERROR_RVS(A1_ephemerides, A1_RVs, A1_J2RVs)
A1_J2_COES_ERROR = ERROR_COES(A1_ephemerides, A1_J2COEs)

A2_J2_RVS_ERROR = ERROR_RVS(A2_ephemerides, A2_RVs, A2_J2RVs)
A2_J2_COES_ERROR = ERROR_COES(A2_ephemerides, A2_J2COEs)

A3_J2_RVS_ERROR = ERROR_RVS(A3_ephemerides, A3_RVs, A3_J2RVs)
A3_J2_COES_ERROR = ERROR_COES(A3_ephemerides, A3_J2COEs)

"""All In One Two-Body + J2"""
A1_J2 = ALL_IN1(A1_J2_COES_DF, A1_J2_RVS_DF)
A2_J2 = ALL_IN1(A2_J2_COES_DF, A2_J2_RVS_DF)
A3_J2 = ALL_IN1(A3_J2_COES_DF, A3_J2_RVS_DF)

A1_J2_Error = ALL_IN1(A1_J2_COES_ERROR, A1_J2_RVS_ERROR)
A2_J2_Error = ALL_IN1(A2_J2_COES_ERROR, A2_J2_RVS_ERROR)
A3_J2_Error = ALL_IN1(A3_J2_COES_ERROR, A3_J2_RVS_ERROR)

################################################################################

"""Saving to .csv"""
A1_2Body_Error.to_csv("A1_TwoBody_Error_All.csv", index=False, header=True)
A2_2Body_Error.to_csv("A2_TwoBody_Error_All.csv", index=False, header=True)
A3_2Body_Error.to_csv("A3_TwoBody_Error_All.csv", index=False, header=True)

A1_J2_Error.to_csv("A1_TwoBody_J2_Error_All.csv", index=False, header=True)
A2_J2_Error.to_csv("A2_TwoBody_J2_Error_All.csv", index=False, header=True)
A3_J2_Error.to_csv("A3_TwoBody_J2_Error_All.csv", index=False, header=True)

import datetime
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from astropy.time import Time
from cysgp4 import *
from Constants import *
from Main_File import *
from OrbitalElements import *

def readtle(txtstr):
    dr = open(txtstr, "r").read()
    lines = dr.split('\n')
    list_txt = np.array(list(zip(*tuple(lines[idx::3] for idx in range(3)))))

    listed = []
    for i in range(len(list_txt)):
        listed.append(PyTle(*list_txt[i]))

    return listed
```

```
def epochs(listed):
    epoch = np.array([])
    for i in range(len(listed)):
        epoch = np.append(epoch, listed[i].epoch.mjd)

    return epoch

def sgp4_prop(time, satellite):
    results = []
    for i in range(len(satellite)):
        propagated = propagate_many(time[i], satellite[0], do_geo=False, do_topo=False)
        results.append(propagated)

    pos = np.array([])
    vel = np.array([])

    for j in range(len(results)):
        pos = np.append(pos, results[j]["eci_pos"])
        vel = np.append(vel, results[j]["eci_vel"])

    pos = pos.reshape(int(len(pos)/3),3) / ER
    vel = vel.reshape(int(len(vel)/3),3) / VU

    states = np.hstack((pos,vel))

    return states

def striptime(time):
    period = np.array([])

    for i in range(len(time)):
        less = datetime.strptime(time.isot[i], '%Y-%m-%dT%H:%M:%S.%f')
        low = less.timestamp()
        period = np.append(period, low/TU)

    difference = np.array([0])
    p = 1
    while p in range(len(period)):
        haiya = period[p] - period[0]
        difference = np.append(difference, haiya)
        p = p + 1

    return difference

################################################################################

"""Fetching the TLEs"""
LAPAN_A1 = "from launch/3LE/A1.txt"
LAPAN_A2 = "from launch/3LE/A2.txt"
LAPAN_A3 = "from launch/3LE/A3.txt"
```

```
A1 = tle_ephemerides("from launch/A1.txt")
A2 = tle_ephemerides("from launch/A2.txt")
A3 = tle_ephemerides("from launch/A3.txt")

RVs1 = coes2rvs(A1)
RVs2 = coes2rvs(A2)
RVs3 = coes2rvs(A3)

read_a1 = readtle(LAPAN_A1)
epoch_a1 = epochs(read_a1)
sgp4_a1 = sgp4_prop(epoch_a1[1:], read_a1[1:])
sgp4_a1_coes = rvs2coes(sgp4_a1).transpose()

read_a2 = readtle(LAPAN_A2)
epoch_a2 = epochs(read_a2)
sgp4_a2 = sgp4_prop(epoch_a2[1:], read_a2[1:])
sgp4_a2_coes = rvs2coes(sgp4_a2).transpose()

read_a3 = readtle(LAPAN_A3)
epoch_a3 = epochs(read_a3)
sgp4_a3 = sgp4_prop(epoch_a3[1:], read_a3[1:])
sgp4_a3_coes = rvs2coes(sgp4_a3).transpose()

period_a1 = Time(epoch_a1[1:], format='mjd', scale='utc')
period_a2 = Time(epoch_a2[1:], format='mjd', scale='utc')
period_a3 = Time(epoch_a3[1:], format='mjd', scale='utc')

A1_T = striptime(Time(epoch_a1[1:], format='mjd', scale='utc'))
A2_T = striptime(Time(epoch_a2[1:], format='mjd', scale='utc'))
A3_T = striptime(Time(epoch_a3[1:], format='mjd', scale='utc'))

################################################################################

"""DataFrame"""
# A1_SGP4_RVS_DF = PROP_RVS(A1, RVs1, sgp4_a1.transpose())
# A1_SGP4_COES_DF = PROP_COES(A1, sgp4_a1_coes.transpose())

# A2_SGP4_RVS_DF = PROP_RVS(A2, RVs2, sgp4_a2.transpose())
# A2_SGP4_COES_DF = PROP_COES(A2, sgp4_a2_coes.transpose())

# A3_SGP4_RVS_DF = PROP_RVS(A3, RVs3, sgp4_a3.transpose())
# A3_SGP4_COES_DF = PROP_COES(A3, sgp4_a3_coes.transpose())

"""Error DataFrame"""
# A1_SGP4_RVS_ERROR = ERROR_RVS(A1, RVs1, sgp4_a1.transpose())
# A1_SGP4_COES_ERROR = ERROR_COES(A1, sgp4_a1_coes.transpose())

# A2_SGP4_RVS_ERROR = ERROR_RVS(A2, RVs2, sgp4_a2.transpose())
# A2_SGP4_COES_ERROR = ERROR_COES(A2, sgp4_a2_coes.transpose())
```

```
# A3_SGP4_RVS_ERROR = ERROR_RVS(A3, RVs3, sgp4_a3.transpose())
# A3_SGP4_COES_ERROR = ERROR_COES(A3, sgp4_a3_coes.transpose())

"""ALLINON DataFrame"""
# A1_SGP4 = ALL_IN1(A1_SGP4_COES_DF, A1_SGP4_RVS_DF)
# A2_SGP4 = ALL_IN1(A2_SGP4_COES_DF, A2_SGP4_RVS_DF)
# A3_SGP4 = ALL_IN1(A3_SGP4_COES_DF, A3_SGP4_RVS_DF)

# A1_SGP4_Error = ALL_IN1(A1_SGP4_COES_ERROR, A1_SGP4_RVS_ERROR)
# A2_SGP4_Error = ALL_IN1(A2_SGP4_COES_ERROR, A2_SGP4_RVS_ERROR)
# A3_SGP4_Error = ALL_IN1(A3_SGP4_COES_ERROR, A3_SGP4_RVS_ERROR)

################################################################################

"""saving in .csv"""
# A1_SGP4_RVS_DF.to_csv('A1_SGP4_RVs.csv', index = False, header=True)
# A1_SGP4_COES_DF.to_csv('A1_SGP4_COEs.csv', index = False, header=True)
# A1_SGP4_RVS_ERROR.to_csv('A1_SGP4_RVs_Error.csv', index = False, header=True)
# A1_SGP4_COES_ERROR.to_csv('A1_SGP4_COEs_Error.csv', index = False, header=True)

# A2_SGP4_RVS_DF.to_csv('A2_SGP4_RVs.csv', index = False, header=True)
# A2_SGP4_COES_DF.to_csv('A2_SGP4_COEs.csv', index = False, header=True)
# A2_SGP4_RVS_ERROR.to_csv('A2_SGP4_RVs_Error.csv', index = False, header=True)
# A2_SGP4_COES_ERROR.to_csv('A2_SGP4_COEs_Error.csv', index = False, header=True)

# A3_SGP4_RVS_DF.to_csv('A3_SGP4_RVs.csv', index = False, header=True)
# A3_SGP4_COES_DF.to_csv('A3_SGP4_COEs.csv', index = False, header=True)
# A3_SGP4_RVS_ERROR.to_csv('A3_SGP4_RVs_Error.csv', index = False, header=True)
# A3_SGP4_COES_ERROR.to_csv('A3_SGP4_COEs_Error.csv', index = False, header=True)

# A1_SGP4.to_csv('A1_SGP4_All.csv', index = False, header=True)
# A2_SGP4.to_csv('A2_SGP4_All.csv', index = False, header=True)
# A3_SGP4.to_csv('A3_SGP4_All.csv', index = False, header=True)
# A1_SGP4_Error.to_csv('A1_SGP4_Error_All.csv', index = False, header=True)
# A2_SGP4_Error.to_csv('A2_SGP4_Error_All.csv', index = False, header=True)
# A3_SGP4_Error.to_csv('A3_SGP4_Error_All.csv', index = False, header=True)

################################################################################

"""Plotting"""
# plot_a1 = Coes_plot_2D(sgp4_a1_coes.transpose(), striptime(period_a1))
# A1_inplane  = plot_inplane(sgp4_a1[:,0:3].transpose())
# A1_outplane = plot_outplane(sgp4_a1[:,0:3].transpose())
# A1_spatial  = plot_3d(sgp4_a1[:,0:3].transpose())

# plot_a2 = Coes_plot_2D(sgp4_a2_coes.transpose(), striptime(period_a2))
# A2_inplane  = plot_inplane(sgp4_a2[:,0:3].transpose())
# A2_outplane = plot_outplane(sgp4_a2[:,0:3].transpose())
# A2_spatial  = plot_3d(sgp4_a2[:,0:3].transpose())
```

```
# plot_a3 = Coes_plot_2D(sgp4_a3_coes.transpose(), striptime(period_a3))
# A3_inplane  = plot_inplane(sgp4_a3[:,0:3].transpose())
# A3_outplane = plot_outplane(sgp4_a3[:,0:3].transpose())
# A3_spatial  = plot_3d(sgp4_a3[:,0:3].transpose())

# plot_a1.savefig("LAPAN-A1_Propagated-COEs-SGP4.png", dpi=600)
# plot_a2.savefig("LAPAN-A2_Propagated-COEs-SGP4.png", dpi=600)
# plot_a3.savefig("LAPAN-A3_Propagated-COEs-SGP4.png", dpi=600)

# A1_inplane.savefig("LAPAN-A1_X&Y-Plane_Position-SGP4.png", dpi=600)
# A2_inplane.savefig("LAPAN-A2_X&Y-Plane_Position-SGP4.png", dpi=600)
# A3_inplane.savefig("LAPAN-A3_X&Y-Plane_Position-SGP4.png", dpi=600)

# A1_outplane.savefig("LAPAN-A1_Y&Z-Plane_Position-SGP4.png", dpi=600)
# A2_outplane.savefig("LAPAN-A2_Y&Z-Plane_Position-SGP4.png", dpi=600)
# A3_outplane.savefig("LAPAN-A3_Y&Z-Plane_Position-SGP4.png", dpi=600)

# A1_spatial.savefig("LAPAN-A1_3D_Position--SGP4.png", dpi=600)
# A2_spatial.savefig("LAPAN-A2_3D_Position--SGP4.png", dpi=600)
# A3_spatial.savefig("LAPAN-A3_3D_Position--SGP4.png", dpi=600)

from Error_Func import *

"""Fetching the necessary data"""
data_A1_2B = pd.read_csv(
    "Tables-csv/A1/2B/A1_TwoBody_Error_All.csv", index_col=False, header=0
)
data_A2_2B = pd.read_csv(
    "Tables-csv/A2/2B/A2_TwoBody_Error_All.csv", index_col=False, header=0
)
data_A3_2B = pd.read_csv(
    "Tables-csv/A3/2B/A3_TwoBody_Error_All.csv", index_col=False, header=0
)

data_A1_J2 = pd.read_csv(
    "Tables-csv/A1/J2/A1_TwoBody_J2_Error_All.csv", index_col=False, header=0
)
data_A2_J2 = pd.read_csv(
    "Tables-csv/A2/J2/A2_TwoBody_J2_Error_All.csv", index_col=False, header=0
)
data_A3_J2 = pd.read_csv(
    "Tables-csv/A3/J2/A3_TwoBody_J2_Error_All.csv", index_col=False, header=0
)

data_A1_SG = pd.read_csv(
    "Tables-csv/A1/SGP4/A1_SGP4_Error_All.csv", index_col=False, header=0
)
data_A2_SG = pd.read_csv(
    "Tables-csv/A2/SGP4/A2_SGP4_Error_All.csv", index_col=False, header=0
)
```

```
data_A3_SG = pd.read_csv(
    "Tables-csv/A3/SGP4/A3_SGP4_Error_All.csv", index_col=False, header=0
)

"""Applying the scikit linear regression model from the function definition"""
# 2B
A1_2B = coes_regres(data_A1_2B, A1_t)
A2_2B = coes_regres(data_A2_2B, A2_t)
A3_2B = coes_regres(data_A3_2B, A3_t)
A1_2B_rvs = rvs_regres(data_A1_2B, A1_t)
A2_2B_rvs = rvs_regres(data_A2_2B, A2_t)
A3_2B_rvs = rvs_regres(data_A3_2B, A3_t)

# 2BJ2
A1_J2 = coes_regres(data_A1_J2, A1_t)
A2_J2 = coes_regres(data_A2_J2, A2_t)
A3_J2 = coes_regres(data_A3_J2, A3_t)
A1_J2_rvs = rvs_regres(data_A1_J2, A1_t)
A2_J2_rvs = rvs_regres(data_A2_J2, A2_t)
A3_J2_rvs = rvs_regres(data_A3_J2, A3_t)

# SGP4
A1_SG = coes_regres(data_A1_SG, A1_T)
A2_SG = coes_regres(data_A2_SG, A2_T)
A3_SG = coes_regres(data_A3_SG, A3_T)
A1_SG_rvs = rvs_regres(data_A1_SG, A1_T)
A2_SG_rvs = rvs_regres(data_A2_SG, A2_T)
A3_SG_rvs = rvs_regres(data_A3_SG, A3_T)

"""Graphing"""
# 2B
graphs_A1_2B = coes_analysis(A1_2B)
graphs_A2_2B = coes_analysis(A2_2B)
graphs_A3_2B = coes_analysis(A3_2B)
graphs_A1_2B_rvs = rvs_analysis(A1_2B_rvs)
graphs_A2_2B_rvs = rvs_analysis(A2_2B_rvs)
graphs_A3_2B_rvs = rvs_analysis(A3_2B_rvs)

# 2BJ2
graphs_A1_J2 = coes_analysis(A1_J2)
graphs_A2_J2 = coes_analysis(A2_J2)
graphs_A3_J2 = coes_analysis(A3_J2)
graphs_A1_J2_rvs = rvs_analysis(A1_J2_rvs)
graphs_A2_J2_rvs = rvs_analysis(A2_J2_rvs)
graphs_A3_J2_rvs = rvs_analysis(A3_J2_rvs)

# SGP4
graphs_A1_SG = coes_analysis(A1_SG)
graphs_A2_SG = coes_analysis(A2_SG)
graphs_A3_SG = coes_analysis(A3_SG)
```

```
graphs_A1_SG_rvs = rvs_analysis(A1_SG_rvs)
graphs_A2_SG_rvs = rvs_analysis(A2_SG_rvs)
graphs_A3_SG_rvs = rvs_analysis(A3_SG_rvs)

"""Errors DataFrame Information"""
COES_Error_DF = coes_information(
    A1_2B, A2_2B, A3_2B, A1_J2, A2_J2, A3_J2, A1_SG, A2_SG, A3_SG
)
RVS_Error_DF = rvs_information(
    A1_2B_rvs,
    A2_2B_rvs,
    A3_2B_rvs,
    A1_J2_rvs,
    A2_J2_rvs,
    A3_J2_rvs,
    A1_SG_rvs,
    A2_SG_rvs,
    A3_SG_rvs,
)
COES_Error_DF.to_csv("COES_Propagation_Error_Info.csv", index=False, header=True)
RVS_Error_DF.to_csv("RVS_Propagation_Error_Info.csv", index=False, header=True)

from symfit import parameters, variables, sin, cos, Fit
from Main_File import A1_ephemerides, A2_ephemerides, A3_ephemerides
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import julian

# Getting the needed data
data_A1_2B = pd.read_csv('Tables-csv/A1/2B/A1_TwoBody_All.csv', index_col=False, header=0)
data_A2_2B = pd.read_csv('Tables-csv/A2/2B/A2_TwoBody_All.csv', index_col=False, header=0)
data_A3_2B = pd.read_csv('Tables-csv/A3/2B/A3_TwoBody_All.csv', index_col=False, header=0)

data_A1_J2 = pd.read_csv('Tables-csv/A1/J2/A1_TwoBody_J2_All.csv', index_col=False, header=0)
data_A2_J2 = pd.read_csv('Tables-csv/A2/J2/A2_TwoBody_J2_All.csv', index_col=False, header=0)
data_A3_J2 = pd.read_csv('Tables-csv/A3/J2/A3_TwoBody_J2_All.csv', index_col=False, header=0)

data_A1_SG = pd.read_csv('Tables-csv/A1/SGP4/A1_SGP4_All.csv', index_col=False, header=0)
data_A2_SG = pd.read_csv('Tables-csv/A2/SGP4/A2_SGP4_All.csv', index_col=False, header=0)
data_A3_SG = pd.read_csv('Tables-csv/A3/SGP4/A3_SGP4_All.csv', index_col=False, header=0)

data_A1_2B_Error = pd.read_csv('Tables-csv/A1/2B/A1_TwoBody_Error_All.csv', index_col=False, header=0)
data_A2_2B_Error = pd.read_csv('Tables-csv/A2/2B/A2_TwoBody_Error_All.csv', index_col=False, header=0)
data_A3_2B_Error = pd.read_csv('Tables-csv/A3/2B/A3_TwoBody_Error_All.csv', index_col=False, header=0)

data_A1_J2_Error = pd.read_csv('Tables-csv/A1/J2/A1_TwoBody_J2_Error_All.csv', index_col=False, header=0)
data_A2_J2_Error = pd.read_csv('Tables-csv/A2/J2/A2_TwoBody_J2_Error_All.csv', index_col=False, header=0)
data_A3_J2_Error = pd.read_csv('Tables-csv/A3/J2/A3_TwoBody_J2_Error_All.csv', index_col=False, header=0)

data_A1_SG_Error = pd.read_csv('Tables-csv/A1/SGP4/A1_SGP4_Error_All.csv', index_col=False, header=0)
```

```
data_A2_SG_Error = pd.read_csv('Tables-csv/A2/SGP4/A2_SGP4_Error_All.csv', index_col=False, header=0)
data_A3_SG_Error = pd.read_csv('Tables-csv/A3/SGP4/A3_SGP4_Error_All.csv', index_col=False, header=0)

A1_t = data_A1_2B["Epoch"]
A2_t = data_A2_2B["Epoch"]
A3_t = data_A3_2B["Epoch"]

A1_t_2B = data_A1_2B_Error["Epoch"]
A2_t_2B = data_A2_2B_Error["Epoch"]
A3_t_2B = data_A3_2B_Error["Epoch"]

A1_t_J2 = data_A1_J2_Error["Epoch"]
A2_t_J2 = data_A2_J2_Error["Epoch"]
A3_t_J2 = data_A3_J2_Error["Epoch"]

A1_t_s = data_A1_SG["Epoch"]
A2_t_s = data_A2_SG["Epoch"]
A3_t_s = data_A3_SG["Epoch"]

# extracting the ones that seemingly has periodic pattern
A1_TLE_SP = A1_ephemerides[:,1].astype(float)
A1_TLE_Ec = A1_ephemerides[:,2].astype(float)
A1_TLE_In = A1_ephemerides[:,3].astype(float)
A2_TLE_SP = A2_ephemerides[:,1].astype(float)
A3_TLE_Ec = A3_ephemerides[:,2].astype(float)

A1_2B_error_In = data_A1_2B_Error["i_e"].values

A1_J2_Ec = data_A1_J2["e_f"].values
A2_J2_Ec = data_A2_J2["e_f"].values
A1_J2_error_In = data_A1_J2_Error["i_e"].values
A1_J2_error_RN = data_A1_J2_Error["RAAN_e"].values

A1_SG_Ec = data_A1_SG["e_f"].values
A2_SG_Ec = data_A2_SG["e_f"].values
A3_SG_Ec = data_A3_SG["e_f"].values
A1_SG_error_In = data_A1_SG_Error["i_e"].values
A1_SG_error_RN = data_A1_SG_Error["RAAN_e"].values
A2_SG_error_SP = data_A2_SG_Error["p_e"].values

# Guessing the period of the periodical graph
A1_TLE_SP_guess = 1/(2725*2)
A1_TLE_Ec_guess = 1/111
A1_TLE_In_guess = 1/(2725*4)
A2_TLE_SP_guess = 1/1896
A3_TLE_Ec_guess = 1/105

A1_2B_error_In_guess = 1/4717

A1_J2_Ec_guess = 1/40.17
```

```
A2_J2_Ec_guess = 1/49
A1_J2_error_In_guess = 1/4770
A1_J2_error_RN_guess = 1/4742.5

A1_SG_Ec_guess = 1/82.3
A2_SG_Ec_guess = 1/25.2
A3_SG_Ec_guess = 1/105.6
A1_SG_error_In_guess = 1/4720
A1_SG_error_RN_guess = 1/4803.1
A2_SG_error_SP_guess = 1/270

# Definig the dourier series
def fourier_series(x, f, n=0):
    """
    Returns a symbolic fourier series of order `n`.

    :param n: Order of the fourier series.
    :param x: Independent variable
    :param f: Frequency of the fourier series
    """
    # Make the parameter objects for all the terms
    a0, *cos_a = parameters(','.join(['a{}'.format(i) for i in range(0, n + 1)]))
    sin_b = parameters(','.join(['b{}'.format(i) for i in range(1, n + 1)]))
    # Construct the series
    series = a0 + sum(ai * cos(i * f * x) + bi * sin(i * f * x)
                      for i, (ai, bi) in enumerate(zip(cos_a, sin_b), start=1))
    return series

x, y = variables('x, y')
w, k, l = parameters("w, k, l")
w.value = A2_TLE_SP_guess ##### change according to the needed parameters
model_dict = {y: fourier_series(x, f=w, n=5)}
print(model_dict)

# Make step function data
epoch = pd.to_datetime(A2_t) ##### change according to the needed parameters
epoch_conv = epoch.apply(julian.to_jd)
xdata = epoch_conv
ydata = A2_TLE_SP ##### change according to the needed parameters

# Define a Fit object for this model and data
fit = Fit(model_dict, x=xdata, y=ydata)
fit_result = fit.execute()
print(fit_result)

# Plot the result
fig, ax = plt.subplots(figsize=(10, 10))
ax.plot(xdata, ydata, ".")
ax.plot(xdata, fit.model(x=xdata, **fit_result.params).y, ls=':')
ax.set_xlabel('MJD')
```

```
ax.set_ylabel("p_i")
plt.show()
```

Turnitin Report

Checked

ORIGINALITY REPORT

9%	8%	5%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Montana State University, Bozeman Student Paper	1%
2	docplayer.net Internet Source	1%
3	pui.ristekbrin.go.id Internet Source	<1%
4	idoc.pub Internet Source	<1%
5	celestrak.com Internet Source	<1%
6	hdl.handle.net Internet Source	<1%
7	B TAPLEY. "The Orbit Problem", Statistical Orbit Determination, 2004 Publication	<1%
8	Lei Chen, Xian-Zong Bai, Yan-Gang Liang, Ke-Bo Li. "Orbital Data Applications for Space Objects", Springer Nature, 2017	<1%

	Publication	
9	pt.scribd.com Internet Source	<1%
10	Submitted to University of Glamorgan Student Paper	<1%
11	Submitted to University of Strathclyde Student Paper	<1%
12	vibdoc.com Internet Source	<1%
13	depositonce.tu-berlin.de Internet Source	<1%
14	moam.info Internet Source	<1%
15	en.wikipedia.org Internet Source	<1%
16	epdf.pub Internet Source	<1%
17	Submitted to University of Northumbria at Newcastle Student Paper	<1%
18	Michael Huemer. "Serious theories and skeptical theories: Why you are probably not a brain in a vat", Philosophical Studies, 2015 Publication	<1%

19	silo.pub Internet Source	<1%
20	Jaenisch, Holger, James Handley, and Peter Chin. "Performance comparison of the Prophecy (forecasting) Algorithm in FFT form for unseen feature and time-series prediction", Cyber Sensing 2013, 2013. Publication	<1%
21	www.docme.ru Internet Source	<1%
22	Submitted to Franklin University Student Paper	<1%
23	Submitted to University of Surrey Student Paper	<1%
24	www.arthra.ugal.ro Internet Source	<1%
25	www.crn2.inpe.br Internet Source	<1%
26	space.skyrocket.de Internet Source	<1%
27	www.cdeagle.com Internet Source	<1%
28	www.ilr.tu-berlin.de Internet Source	<1%

29	eprints.soton.ac.uk Internet Source	<1%
30	www.garada.unsw.edu.au Internet Source	<1%
31	dokumen.pub Internet Source	<1%
32	www.thieme-connect.de Internet Source	<1%
33	krishikosh.egranth.ac.in Internet Source	<1%
34	aerospaceresearch.net Internet Source	<1%
35	sevima.com Internet Source	<1%
36	Weiduo Hu. "Fundamental Spacecraft Dynamics and Control", Wiley, 2015 Publication	<1%
37	www.dtic.mil Internet Source	<1%
38	Submitted to Harrisburg University of Science and Technology Student Paper	<1%
39	Submitted to University of Kent at Canterbury Student Paper	<1%

40	M. Vázquez, E. Pallé, P. Montañés Rodríguez. "Chapter 8 Extrasolar Planetary Systems", Springer Science and Business Media LLC, 2010 Publication	<1 %
41	www.ccd.edu Internet Source	<1 %
42	Submitted to University of Sussex Student Paper	<1 %
43	www.math.ucdenver.edu Internet Source	<1 %
44	www.trylam.com Internet Source	<1 %
45	help.agi.com Internet Source	<1 %
46	aaltodoc.aalto.fi Internet Source	<1 %
47	www.thoughtco.com Internet Source	<1 %
48	P. Szabo, K. Gombikova, M. Ferencova, M. Kosuda. "Keplerian Orbit and Satellite skCUBE", 2019 New Trends in Aviation Development (NTAD), 2019 Publication	<1 %
	Submitted to Cranfield University	

49	Student Paper	<1%
50	Klaus Wittmann. "Fundamentals", Handbook of Space Technology, 04/10/2009 Publication	<1%
51	Nasution, Aulia, Wiweka Adiatma, Apriani Kusumawardhani, and Hendra Cordova. "Mapping system for the surface temperature in the Ijen Crater by Landsat-8 imagery with supervised image segmentation method based on fuzzy logic", International Seminar on Photonics Optics and Its Applications (ISPhOA 2014), 2015. Publication	<1%
52	Maximo Valentinuzzi. "An analysis of the mechanical forces in each semicircular canal of the cat under single and combined rotations", The Bulletin of Mathematical Biophysics, 1967 Publication	<1%
53	statisticsbyjim.com Internet Source	<1%
54	BEONG IN YUN. "SIGMOIDAL-TYPE SERIES EXPANSION", The ANZIAM Journal, 01/2008 Publication	<1%
55	Submitted to Brunel University Student Paper	<1%

56	Submitted to Commack High School Student Paper	<1%
57	Submitted to The University of Manchester Student Paper	<1%
58	Submitted to University of Southampton Student Paper	<1%
59	Submitted to Middle East Technical University Student Paper	<1%
60	Curtis, Howard D.. "Orbits in Three Dimensions", Orbital Mechanics for Engineering Students, 2014. Publication	<1%
61	Michel Capderou. "Handbook of Satellite Orbits", Springer Science and Business Media LLC, 2014 Publication	<1%
62	Submitted to University of Bristol Student Paper	<1%
63	Ashish Tewari. "Foundations of Space Dynamics", Wiley, 2021 Publication	<1%
64	Marco Lovera. "Control-oriented modelling and simulation of spacecraft attitude and orbit dynamics", Mathematical and Computer Modelling of Dynamical Systems, 2006	<1%

Publication

65	Submitted to Universita' Telematica Internazionale Nettuno Student Paper	<1%
66	repository.dl.itc.u-tokyo.ac.jp Internet Source	<1%
67	www.iosrjournals.org Internet Source	<1%
68	Submitted to Indian Institute of Technology, Madras Student Paper	<1%
69	drugg.fgg.uni-lj.si Internet Source	<1%
70	Submitted to UC, Boulder Student Paper	<1%
71	eoportal.org Internet Source	<1%
72	Submitted to University College London Student Paper	<1%
73	Submitted to University of Nottingham Student Paper	<1%
74	www.ukessays.com Internet Source	<1%
75	Claudia C. Celestino, Cristina T. Sousa, Wilson	

	Yamaguti, Helio Koiti Kuga. "The Impact on Geographic Location Accuracy due to Different Satellite Orbit Ephemerides", Mathematical Problems in Engineering, 2009 Publication	<1 %
76	eoivisa.com Internet Source	<1 %
77	"The Two- and the Three-Body Problems", Astronomy and Astrophysics Library, 2005 Publication	<1 %
78	tel.archives-ouvertes.fr Internet Source	<1 %
79	Widya Roza, Deddy El Amin, Eriko Nasemuddin Nasser. "Design and implementation of payload data handling based on field programmable gate array", 2014 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology, 2014 Publication	<1 %
80	Mazin Ahmed Elhag, Ahmed Abdelkarim Yassin, Mahmoud Esawi Babiker, Elhussein Abd Elrahman Abd Elmageed. "The unscented Kalman filter applied to satellite orbit determination using only publically available two-line element sets", 2013 INTERNATIONAL CONFERENCE ON COMPUTING, ELECTRICAL AND ELECTRONIC	<1 %

ENGINEERING (ICCEEE), 2013
Publication

81	Submitted to Stuyvesant High School Student Paper	<1%
82	elearning-icse.blogspot.com Internet Source	<1%
83	pdfs.semanticscholar.org Internet Source	<1%
84	www.oak.go.kr Internet Source	<1%
85	Seong-Hyeon Park, Hae-Dong Kim, Gisu Park. "Orbit, orbital lifetime, and reentry survivability estimation for orbiting objects", Advances in Space Research, 2018 Publication	<1%
86	Ahmed Abdelkarim Yassin Abdelkarim, Mahmoud Esawi Babikir Osman, Sharief F. Babiker. "Satellite Orbit Determination and Tracking Model Using Two-Line Elements Set", 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), 2018 Publication	<1%
87	Youngkwang Kim, Sang-Young Park. "Perturbative Analysis on Orbital Kinematics of Flybys and Applications to Doppler	<1%

Observation", Journal of Guidance, Control, and Dynamics, 2015
Publication

---

88 text-id.123dok.com
Internet Source
<1%

---

89 Brian Levine, Cory Cameron, Philip Freidin, Izan Peris Marti, Michael Reindl, Jason Swenson, John Teuber, Ernest Company Vallet. "Conceptual Design for a Space Based Solar Power System-SolPower", AIAA SPACE 2013 Conference and Exposition, 2013
Publication
<1%

---

90 Roberto Lupo, Carlo Albanese, Daniele Bettinelli, Marco Brancati, Giovanni Minei, Claudio Pernechele. "Lighthouse: A spacebased mission concept for the surveillance of geosynchronous space debris from low earth orbit", Advances in Space Research, 2018
Publication
<1%

---

91 scholar.lib.vt.edu
Internet Source
<1%

---

92 www.scribd.com
Internet Source
<1%

---

93 "Global Positioning System: Theory and Applications, Volume I", American Institute of Aeronautics and Astronautics (AIAA), 1996
Publication
<1%

94	Zhao-Yu Sun, Yu-Yin Wu, Jian Xu, Hai-Lin Huang, Bi-Fu Zhan, Bo Wang, Cheng-Bo Duan. " Characterization of quantum phase transition in the model with multipartite correlations and Bell-type inequalities ", Physical Review A, 2014 Publication	<1%
95	diva-portal.org Internet Source	<1%
96	T.A. Lovell. "Application of genetic algorithms to state estimation of tethered systems", Computer Methods in Applied Mechanics and Engineering, 2003 Publication	<1%
97	Thomas V. Peters, Ron Noomen, Pablo Colmenarejo. "Analytical Solutions to Two-Impulse Nondrifting Transfer Problems for Rendezvous in Elliptical Orbits", Journal of Guidance, Control, and Dynamics, 2014 Publication	<1%
98	David A. Vallado. "Orbital Mechanics Fundamentals", Wiley, 2010 Publication	<1%
99	J.F. Moreno, J. Melia. "A method for accurate geometric correction of NOAA AVHRR HRPT data", IEEE Transactions on Geoscience and Remote Sensing, 1993 Publication	<1%

100	directory.eoportal.org Internet Source	<1%
101	www.swe.uni-linz.ac.at Internet Source	<1%
102	Steven D. Deines. "Time Dilation and the Length of the Second: Why Timescales Diverge", The Astronomical Journal, 07/2007 Publication	<1%
103	digitalcommons.calpoly.edu Internet Source	<1%
104	www.star.le.ac.uk Internet Source	<1%
105	users.aalto.fi Internet Source	<1%
106	arc.aiaa.org Internet Source	<1%
107	ssl.mit.edu Internet Source	<1%
108	es.slideshare.net Internet Source	<1%
109	epdf.tips Internet Source	<1%
110	es.scribd.com Internet Source	<1%

111	mafiadoc.com Internet Source	<1%
112	"Mission Design & Implementation of Satellite Constellations", Springer Science and Business Media LLC, 1998 Publication	<1%
113	Muhammad Arif Saifudin, Abdul Karim, Mujtahid. "LAPAN-A4 Concept and Design for Earth Observation and Maritime Monitoring Missions", 2018 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES), 2018 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

Curriculum Vitae

	Basic Information
Name	Prana Paramartha Rao
Place of Birth	Jakarta, Indonesia
Date of Birth	19 August, 1997
Address	Jl. Palem Putri 2 No. 3, Taman Yasmin 5, Bogor
Email	pranaprao@gmail.com
Year	Education
2016 - present	International University Liaison Indonesia (IULI)
2012 - 2015	Bina Insani Senior High School
Year	Work Experiences
08/2019 - 12/2019	Student intern at FL Technics Indonesia, Indonesia
01/2018 - 01/2018	Student intern at Garuda Maintenance Facility, Indonesia
Language	Proficiency
Bahasa Indonesia	First Language
English	Professionally Spoken and Written
Year	Descriptions
08/2018 - 06/2019	Marasha Noir - Crops Sprayer Quadcopter
Year	Organization Experiences
2018 - 2019	Head of Aviation Engineering Student Association's Academic Division
2018 - 2019	Committee Member of PN Indo Jakarta Region
2018 - 2019	Head of IULI's Aeromodelling Club
2017 - 2018	Head of IULI's Choir Club
Year	Activities
10/2018	Innovation Tour Staff at Food Ingredients Asia 2018
03/2018 - 04/2018	Storage - Sorter Staff at Big Bad Wolf Indonesia