Satellite Orbit Determination Using Payload-collected Observation Data

NICHOLAS H.J. BIJNENS

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

Graduate Program in
EARTH AND SPACE SCIENCE

York University
Toronto, Ontario
January 2018

Abstract

The rapid rise in small satellite deployment and miniaturization of communication technology will require a cheaper, leaner and more efficient way of tracking those satellites. Using the already-collected timing data from the payload observations means that no additional on-board equipment or processing will be required and that it could even be applied to existing missions, as well. This thesis provides a solid foundation and development analysis to support this new way of using satellite payload data. It shows how combining even the most basic form of observed data (the time-of-access and location) can provide deeper and more insightful knowledge. Each cost function used and explored combines this data in a different manner and, therefore, provides a different kind of insight pertaining to a different aspect of the satellite behaviour. This, combined with the power of machine learning, has proven to be an effective way of determining the position and velocity of the satellite with strong potential for future development in real-world Earth-observation or, perhaps even, interplanetary missions.

Acknowledgements

I would like to thank my co-supervisors Franz Newland and Jinjun Shan for their guidance and support throughout my research work. It was the perfect balance between pointing me in the right direction and allowing me to make my own mistakes that made this a rich, meaningful and enjoyable learning experience. I would also like to extend my thanks to exactEarth for providing guidance from an industry perspective and supplying additional data and insight, as well as NSERC and our provincial and federal governments for their financial support. Finally, I could not have done this without the unconditional love and support I can always count on from my partner, Joshua, my parents, Murielle and Frank, and my brother, Thomas. They were there to support me when things got a little tougher and to celebrate with me when I achieved important milestones. This has truly been an enriching and fulfilling experience that has added a new dimension to my professional and personal life.

Table of Contents

Abstract ii

Acknowledgements iii

Table of Contents iv

List of Tables viii

List of Figures ix

1 Introduction 1

1.1 Motivation 3

1.2 Research Objectives 5

1.3 Applications of Observation-based Orbit Determination Capabilities 6

1.4 Thesis Outline 7

2 Background 9

2.1 Satellite Orbit Determination 9
2.1.1 Keplerian Orbital Elements 10
2.1.2 The Two-Body Equation 11
2.1.3 Current Methods 12
2.2 Automatic Identification System 15
2.2.1 Operating Principle 16
2.2.2 Satellite Receivers 17
2.2.3 Challenges of Using AIS Signals 18

3 Methodology **20**

3.1 Input Data Simulation 20
3.2 Ground-track-based Estimation Method 21
3.2.1 Point Cluster Centroid 21
3.2.2 Least Squares Intersection 23
3.3 Simultaneous Perturbation Stochastic Approximation Method . . . 24
3.3.1 Background 25
3.3.2 The SPSA Algorithm 26
3.3.3 Cost-function Selection 28
3.3.4 Search-range Boundaries 30
3.3.5 Customized Algorithm Methodology 31

3.3.5.1 Segment 1: Reading in the "true" visibility information 31
3.3.5.2 Segment 2: Analysis window and search range constraints 32
3.3.5.3 Segment 3: Generate the "true" point set for analysis 33
3.3.5.4 Segment 4: Variable transmission interval settings . 33
3.3.5.5 Segment 5: Initializing the orbital elements 34
3.3.5.6 Segment 6: Applying the weighting factor 35
3.3.5.7 Segment 7: The optimization loop 36
3.3.5.8 Segment 8: Generating and applying the perturbation vectors 37
3.3.5.9 Segment 9: Generating the new best estimate . . . 38
3.4 Genetic Algorithm 39
3.4.1 Background 40
3.4.2 The Genetic Algorithm 42
3.4.3 Customized Algorithm Methodology 45
3.4.3.1 Segment 5: Generating the STK point file 45
3.4.3.2 Segment 6: Parent selection 46
3.4.3.3 Segment 7: Generating the new population of children 47
3.4.3.4 Segment 8: Close Neighbour Approximation 49

3.4.3.5 Segment 9: Determining Proximity to Convergence 51

4 Optimization Results **53**

4.1 Preliminary Gradient Descent-based Optimization 53

4.1.1 Varying Transmission Intervals 68

4.2 Genetic Algorithm Optimization 80

4.2.1 Simulated Environment 80

4.3 Real-world AIS Data 94

4.3.1 Optimization Results 94

4.3.2 Improving Real-world Convergence 98

5 Concluding Remarks **101**

5.1 Summary of Results 101

5.2 Usage and Applications 102

5.3 Future Work 103

5.4 Conclusion 104

Bibliography **105**

List of Tables

1.1 Orbit determination method comparison 3

2.1 AIS Reporting Intervals 17

3.1 Orbital Elements Search-range Boundaries 30

3.2 Variable Transmission Interval Ship Distribution Assumption 34

3.3 Close Neighbourhood Approximation Range Values for the Orbital Elements 49

4.1 Confidence Levels Used to Develop the Weighting Function 73

List of Figures

2.1 Overview of the Keplerian Orbital Elements - from ResearchGate . 11

2.2 Flow chart outlining the extended Kalman Filter 14

2.3 Ships transmitting their AIS signal with their ID, location information and timing information at different intervals 16

3.1 STK scenario showing the grid points and satellite object in the ideal case simulation 21

3.2 Sample point cluster at one time step, using evenly distributed point cluster 22

3.3 Plot of the centroids of each point cluster at every time step over a period of 5 hours 23

3.4 Plots comparing the coordinates of the estimated sub-satellite point on the surface using the centroid method and the LSI method with randomly distributed data points - true orbit ground track is shown in green 25

3.5 Figure showing the difference in visibility start and stop time for an arbitrary ship of interest 30

3.6 Outline of the general genetic algorithm 43

4.1 Modified SPSA algorithm results for the semi-major axis at Feb 1, 2016 00:30:00 UTC 54

4.2 Modified SPSA algorithm results for the eccentricity at Feb 1, 2016 00:30:00 UTC 55

4.3 Modified SPSA algorithm results for the inclination at Feb 1, 2016 00:30:00 UTC 56

4.4 Modified SPSA algorithm results for the argument of perigee at Feb 1, 2016 00:30:00 UTC 57

4.5 Modified SPSA algorithm results for the RAAN at Feb 1, 2016 00:30:00 UTC 58

4.6 Modified SPSA algorithm results for the true anomaly at Feb 1, 2016 00:30:00 UTC 59

4.7 Single cost function results for the semi-major axis at Feb 1, 2016 00:30:00 UTC 61

4.8 Single cost function results for the eccentricity at Feb 1, 2016 00:30:00 UTC 62

4.9 Single cost function results for the inclination at Feb 1, 2016 00:30:00 UTC 63
4.10 Single cost function results for the RAAN at Feb 1, 2016 00:30:00 UTC 63
4.11 Single cost function results for the argument of latitude at Feb 1, 2016 00:30:00 UTC 64
4.12 Single cost function results for the semi-major axis at Mar 29, 2016 00:00:00 UTC 65
4.13 Single cost function results for the eccentricity at Mar 29, 2016 00:00:00 UTC 66
4.14 Single cost function results for the inclination at Mar 29, 2016 00:00:00 UTC 67
4.15 Single cost function results for the RAAN at Mar 29, 2016 00:00:00 UTC 67
4.16 Single cost function results for the argument of latitude at Mar 29, 2016 00:00:00 UTC 68
4.17 Variable transmission interval results without weighting compensation for the semi-major axis at Mar 29, 2016 00:00:00 UTC 69
4.18 Variable transmission interval results without weighting compensation for the eccentricity at Mar 29, 2016 00:00:00 UTC 70

4.19 Variable transmission interval results without weighting compensation for the inclination at Mar 29, 2016 00:00:00 UTC 71
4.20 Variable transmission interval results without weighting compensation for the RAAN at Mar 29, 2016 00:00:00 UTC 72
4.21 Variable transmission interval results without weighting compensation for the argument of latitude at Mar 29, 2016 00:00:00 UTC . . 73
4.22 Variable transmission interval results with weighting compensation for the semi-major axis at Mar 29, 2016 00:00:00 UTC 75
4.23 Variable transmission interval results with weighting compensation for the eccentricity at Mar 29, 2016 00:00:00 UTC 76
4.24 Variable transmission interval results with weighting compensation for the inclination at Mar 29, 2016 00:00:00 UTC 77
4.25 Variable transmission interval results with weighting compensation for the RAAN at Mar 29, 2016 00:00:00 UTC 78
4.26 Variable transmission interval results with weighting compensation for the argument of latitude at Mar 29, 2016 00:00:00 UTC 79
4.27 Semi-major axis error results for the six cases 82
4.28 Eccentricity error results for the six cases 83
4.29 Inclination error results for the six cases 84
4.30 RAAN error results for the six cases 85

4.31 Argument of latitude error results for the six cases 86
4.32 Semi-major axis error results 88
4.33 Eccentricity error results 89
4.34 Inclination error results 89
4.35 RAAN error results 90
4.36 Argument of latitude error results 90
4.37 Semi-major axis error results 91
4.38 Eccentricity error results 92
4.39 Inclination error results 92
4.40 RAAN error results 93
4.41 Argument of latitude error results 93
4.42 Semi-major axis error results 95
4.43 Eccentricity error results 95
4.44 Inclination error results 96
4.45 RAAN error results 96
4.46 Argument of latitude error results 97
4.47 Cost function test cases results 99

1 Introduction

Nearly every satellite currently in-orbit or to be launched will require an ability to determine its orbital elements or state vector as a function of time. Presently, this is most often achieved through the use of a Global Navigation Satellite System (GNSS) receiver or NORAD-provided Two-Line Element (TLE) data. With today's significant increase of interest in nanosatellite development, as well as the more affordable alternative offered by hosted payloads, functionality and capability can often be limited. GNSS receivers can take up a relatively large amount of power, mass and space on-board nanosatellites but provide the most accurate means of satellite positioning. NORAD TLE data does not require any on-board equipment or processing as it is obtained through a third-party website but its update frequency can put severe limitation on its usability, especially when smaller beamwidth transmitters are used. In the case of a hosted payload, this payload often does not have access to the host telemetry for orbit information, yet such information may provide useful insight for the payload mission itself.

Furthermore, the nanosatellite industry is still very much an up-and-coming industry and has only been developed over the past approximately 15 years, with most of the developement occurring since 2012 [23]. They have opened the door to a new kind of big data collection through satellite constellations. However, because of this novelty, these types of missions are still left to use technology that was originally designed for larger, heavier satellites, such as communications satellites. There has been very little development so far in terms of developing a new way for these nanosatellites to determine their orbit, keeping in mind their restrictions in terms of mass, space and financial cost, meaning there was also a limited amount of background references for this research to build upon. The research in this thesis does just that; develop a solid backbone for developing a new way of determining the orbit of a nanosatellite, provided that it conforms to the payload data restrictions outlined in Chapter 5.

The research starts with a crude, geometric solution that attempts to estimate the satellite's orbit based on the position of the sub-satellite point on the surface through finding the centroid of the data point cluster at each time step, and the altitude through the apparent diameter of the satellite footprint. However, while working on this method, severe limitations were discovered that lead to the development of more advanced methods in cost function minimization through gradient descent and machine learning techniques.

1.1 Motivation

There are a number of commercial nanosatellites equipped with Automatic Identification System (AIS) receivers that can receive and track AIS signals transmitted by ships and land beacons on the surface of Earth. Up until this point, they are reliant upon the NORAD TLE data to determine the satellite's orbit since no GPS receiver is installed on-board. Given that the NORAD TLE data update frequency for commercial satellites can be as infrequent as once every 1 to 2 weeks, the known orbital state vector can grow outdated very quickly. While the satellite orbit can be propagated using software such as AGI System Tool Kit (STK) in between TLE updates, the satellite drift in between updates is significant enough over a 2-week timespan to cause connection loss for ground station downlinks, especially in the case of small beamwidth transmitters. It also limits knowledge of any events or unexpected perturbations that could have caused the satellite's orbit to be altered. Therefore, the ability to receive daily updates with TLE-level accuracy would be advantageous in terms of downlink uptime and orbit validation. Table 1.1 provides a comparison between using a GPS receiver, NORAD-provided TLE data and what would be an ideal solution for nanosatellite operations.

Table 1.1 Orbit determination method comparison

	GPS	NORAD	IDEAL
Power Cost	MED	NONE	NONE
Dollar Cost	MED	NONE	LOW
Mass Cost	LOW	NONE	NONE
Update Frequency	HOURS	WEEKS	DAYS
Accuracy	SUB-METER	KILOMETERS	KILOMETERS
On-board Computation	LOW	NONE	NONE

As can be seen from Table 1.1, NORAD TLE data corresponds to the orbit determination needs of the nanosatellite industry, or better, except for the update frequency. Unfortunately, as mentioned earlier, the update frequency is what causes downlink losses if left for more than 1 week, which is a high impact risk to take into consideration. Hence, if a method were to be developed that can meet all the criteria of the TLE data but also provide an update on the order of a few days, it would match the needs of the nanosatellite industry exactly in terms of cost, accuracy, on-board computation, and downlink uptime through a more appropriate update frequency.

Using the payload observation data that the satellite has already collected as part of its main mission has several unique advantages. First and foremost it allows for leaner, more efficient satellite missions as no on-board power, mass or downlink capacity needs to be dedicated to orbit determination information. Second, it also allows for near-instantaneous orbit determination using data that has already been downlinked to the ground station, without the computational restrictions that exist for on-board processing. And finally, developing algorithms for this observation-

based orbit determination method will also allow for other related data analyses to be conducted, revealing more insight about the satellite's orbit, its maneuvers and the observed data points themselves. AIS provides just one type of data that can be used with the algorithm developed in this thesis and will be used for testing the real-world applicability of that algorithm. However, the aim is that the algorithm can be used for many other types of large, observation data sets with similar characteristics to AIS data, such as ADS-B and the Internet of Things.

1.2 Research Objectives

At its essence, the purpose of this research is to develop a proof-of-concept for an orbit determination method that allows for near-instantaneous orbit determination of a satellite using payload-collected time-of-access data of a large number of known data points. This kind of orbit determination has never before been successfully implemented and would bring a new level of functionality to new and existing nanosatellites observing the body they orbit. As part of this thesis, the algorithm is developed using simulated satellite data based on real-world AIS data and ship locations, made progressively more realistic to properly incorporate many of the subtleties involved in this type of orbit determination. The algorithm can then later on, as a future project, be made more general beyond AIS-based applications.

The objective of this research is to consistently achieve TLE-level accuracy or

better, meaning 20km along-track and 10km cross-track and radially, using the most realistic simulation data to date. These simulations will take into account all constraints and challenges specific to surface data point observations mentioned in this thesis, such as the data point transmission interval, location distribution and altitude variations. The algorithm also needs to be sufficiently flexible, meaning that it needs to work reliably and with the same level of accuracy for any desirable date of interest and, thus, any realistic distribution of ships. In order to achieve this, data point restrictions and known challenges were kept in mind throughout the entire implementation process, with additional suggestions at the end of this thesis in order to facilitate the expansion of the algorithm to include functionality with real-world observation data. The focus of this thesis will be with specific application of AIS observation data.

1.3 Applications of Observation-based Orbit Determination Capabilities

The algorithm proposed in this thesis has a wide range of potential applications in the nanosatellite industry. Some examples of specific scenarios in which the method could applied are AIS, ADS-B, Internet of Things, BlueForce tracking and other remote sensing missions with a sufficiently large set of observed data

points. Earth observation tracking and data collection missions are increasingly more popular and these kinds of tracking missions are more effectively carried out by many small, low-cost satellites, rather than a few large satellites. The beauty of this technique then is that it will allow these nanosatellites to focus their resources on their specific mission, such that they can be designed lighter and cheaper, while maintaining critical orbit determination functionality. These types of space missions usually require a LEO space segment, a wide field-of-view, and will generally observe a large number of data points, making them ideal for the type of orbit determination algorithm discussed in this thesis. Furthermore, due to the fact that the statistical advantage of a large data set will be used to ensure stability and accuracy, any outliers in the observation data that do not agree with the estimation based on the large majority of the data points can then be analyzed in reverse sense.

1.4 Thesis Outline

Chapter 1 of this thesis has provided an introduction to the motivation behind this research and the objectives to be achieved. Chapter 2 will discuss the background behind current orbit determination techniques, as well as the satellite AIS infrastructure. In Chapter 3, the methodology will be explained in detail, with specific regard to the different attempts made at developing a successful algorithm. Chapter 4 then shows and analyses the results and improvements made to the al-

gorithm, finishing off with the latest version. Finally, Chapter 5 will conclude this thesis with improvement suggestions in terms of convergence accuracy and future development potential.

2 Background

2.1 Satellite Orbit Determination

Determining a satellite's orbit with high accuracy is a mission-critical function that can either be achieved through on-board or ground-based data collection. It involves determining or estimating the state vector (in this case, position-velocity-time (PVT) or orbital elements) of a satellite at a given moment in time. It is most often achieved by applying a form of statistical estimation or optimization to a set of observations, from sensors on-board or from ground-based stations. The current state vector of the satellite can then be used for observation scheduling, downlink scheduling, pointing requirements, as well as a basis for orbital and attitude maneuver requirements. However, the accuracy and effectiveness of these operations is dependent upon the accuracy of the state vector. While some methods, such as GPS-based orbit determination, can generate a very accurate and up-to-date state vector, other methods can cause the error in the propagated state vector to grow to a point where it causes ground stations to no longer be able to communicate

effectively with the satellite, prior to the next available update.

Jerome Vetter offers an in-depth look into the history of satellite orbit determination in "Fifty Years of Orbit Determination" [33]. Milani and Gronchi present general theoretical considerations involved in satellite orbit determination in the "Theory of Orbit Determination" [1]. "Statistical Orbit Determination" by Tapley, Schutz and Byron focuses more on the typical observations involved and the mathematical tools used for state estimation [3]. Finally, Vallado and Crawford offer an even more specialized view of satellite orbit determination based solely on TLE-data through SGP4 propagations in "SGP4 Orbit Determination" [12].

2.1.1 Keplerian Orbital Elements

A satellite's current orbit and position along that orbit can be represented through the six Keplerian orbital elements (see Figure 2.1 below, as taken from ResearchGate). The first of the six elements is the semi-major axis (a), defined as half the length of the line of apsides (the line connecting the closest and furthest points from the central body along the orbit). Second is the eccentricity (e), which ranges between 0 and 1 and describes the circularity of the orbit - 0 being perfectly circular and 1 being hyperbolic. Next is the inclination (i), indicating the angular offset of the orbit from the equatorial plane. Fourth is the argument of periapsis (ω), which describes the angular distance in the orbital plane between the ascending

node and the periapsis. The fifth element is the right ascension of the ascending node (Ω), defined as the angular distance in the equatorial plane between the Vernal Equinox and the ascending node. And finally, the true anomaly (Θ) describes the angular distance in the orbital plane between the periapsis and the current satellite position.

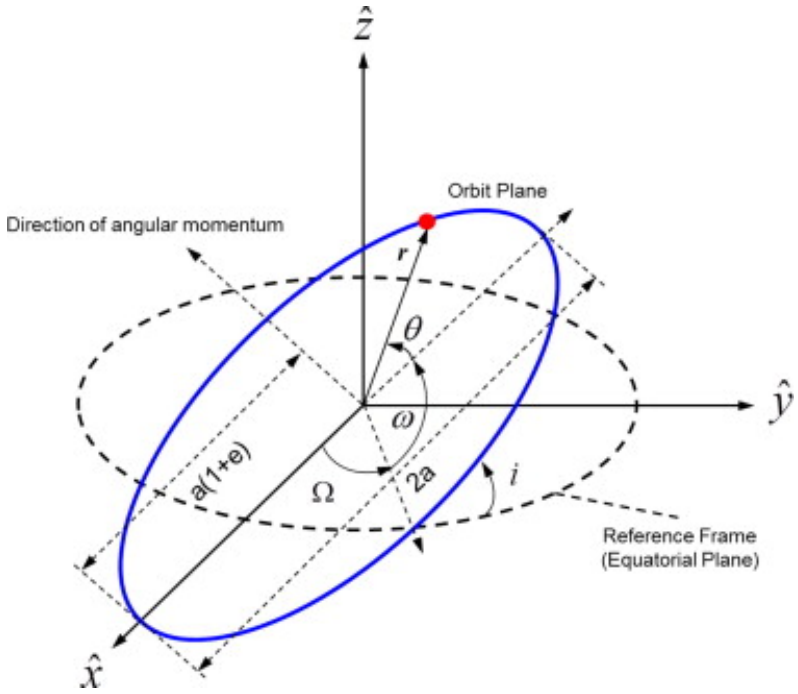

Fig. 2.1 Overview of the Keplerian Orbital Elements - from ResearchGate

2.1.2 The Two-Body Equation

The two-body problem is concerned with determining the motion of two gravitational bodies interacting with one another. Based on Newton's Law of Gravitation,

the motion of these two bodies can be written as:

$$\overrightarrow{F_{12}} = -\overrightarrow{F_{21}} = G\frac{m_1 m_2}{r^2}\frac{\overrightarrow{r}}{r}$$

Setting $\mu = Gm_1$ and $\overrightarrow{F_{12}} = m_2\ddot{\overrightarrow{r}}$ and rearranging, gives what is known as the two-body equation of orbital motion:

$$\ddot{\overrightarrow{r}} = \frac{-\mu}{r^3}\overrightarrow{r}$$

This equation only takes into account the pure gravitational interaction between the two bodies (such as, for example, a satellite and Earth) but not any of the perturbation affecting this motion. Since determining the orbit of a spacecraft requires an accurate of the spacecraft's motion, these perturbations need to be taken into account as well. This can include atmospheric drag, Earth's oblateness and solar radiation pressure. More information on how these perturbations can be accounted for is outlined in Section 2.1.3.

2.1.3 Current Methods

One of the most popular and accurate orbit determination methods is using the observations from an on-board GPS receiver. With the appropriate hardware, availability, and processing, accuracy of less than 15m in position and 0.05 m/s in velocity can be achieved. However, the major downside to using a GPS receiver to perform orbit determination is that its power, mass and cost specifications can

easily exceed the mission constraints. This often limits the allowable specifications of other on-board equipment, increases the launch and operational costs or simply makes using a GPS receiver infeasible, especially in terms of nanosatellites.

When using a GPS receiver is unrealistic given the mission constraints, satellite operators most often resort to using the NORAD-provided TLE data. The orbital elements contained in the TLE set for a specified date and time can be used as input to an SGP4 propagation to predict the satellite orbit at other time instances. The accuracy of such a propagation is on average around 10km cross-track and radially and 20km in-track. However, the accuracy depends on the update frequency of the TLE data, which, for commercial satellites, can be up to two weeks. Moreover, relying on a third party, like NORAD, for orbit information also makes the satellite operator vulnerable to any changes such provider may make to their services or policies. Satellite operators can also use a range of other methods such as using Doppler observations or radar tracking, but these methods are expensive and mostly used on large communications satellites or by organizations such as NORAD to provide input for the TLE data.

Two of the most common methods of determining the orbit of a satellite are through Least Squares (LSOD) or the extended Kalman Filter (EKF). The main difference between the two is that the LSOD method requires an entire set of measurements in order to output the satellite state vector, whereas the EKF can

be run with every new measurement taken. The EKF will be briefly explained below (see Fig 2.2).

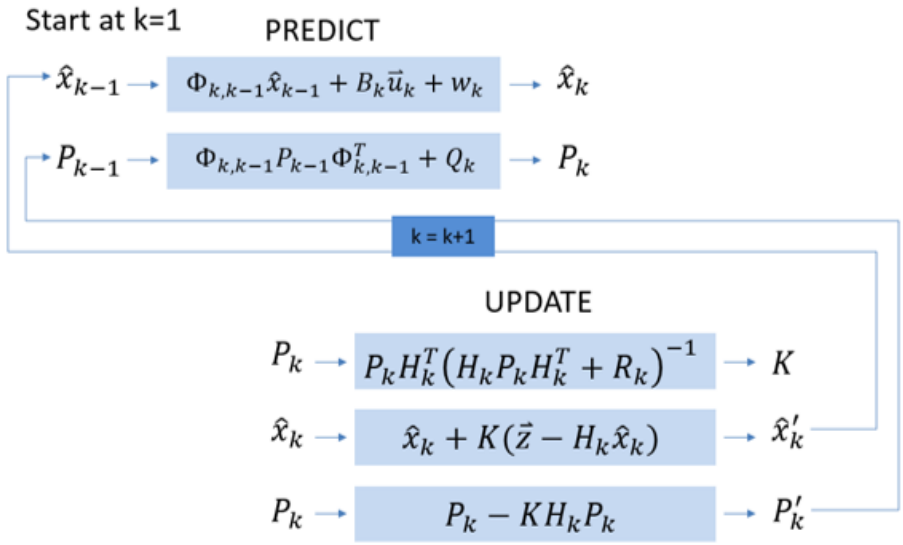

Fig. 2.2 Flow chart outlining the extended Kalman Filter

At its essence, the EKF contains two main phases, the prediction phase and the update phase. During the prediction phase, the satellite's state vector, which includes information on satellite's current estimated position and velocity OR orbital elements, is propagated in time by means of the state transition matrix $\Phi_{k,k-1}$. This matrix, representing how the satellite's position and velocity change over time, is obtained through integration of the two-body equation of motion. Any perturbations such as atmospheric drag, Earth's oblateness and solar radiation pressure, are contained in the control input matrix B_k. Similarly the covariance matrix P_{k-1} is

also time-updated.

During the update phase, the state vector and covariance matrix are updated with the observation measurements z. In order to do so, a transformation matrix H_k needs to be constructed that relates the state vector parameters to the measurements themselves. This allows for the computation of the Kalman gain K, which is in turn used to update the state vector and covariance matrix.

In summary, the EKF allows one to start with an estimated satellite orbit state vector (such as from TLE data), and update this state vector with each new measurement coming in (such as from a GPS receiver on-board).

2.2 Automatic Identification System

According to Chapter V of the Safety of Life at Sea Convention, all ships over 300 gross tonnage on international routes, all cargo ships over 500 gross tonnage not on international routes and all passenger ships must be equipped with an AIS transponder to send and receive AIS signals [25]. These signals contain the ship's course, position, speed and rate of turn. Due to the curvature of Earth, ships and coastal marine stations can receive signals from any ship or ground station within an 80km radius [4]. The information received can then be used to avoid colliding with other vessels as well as to monitor maritime activity.

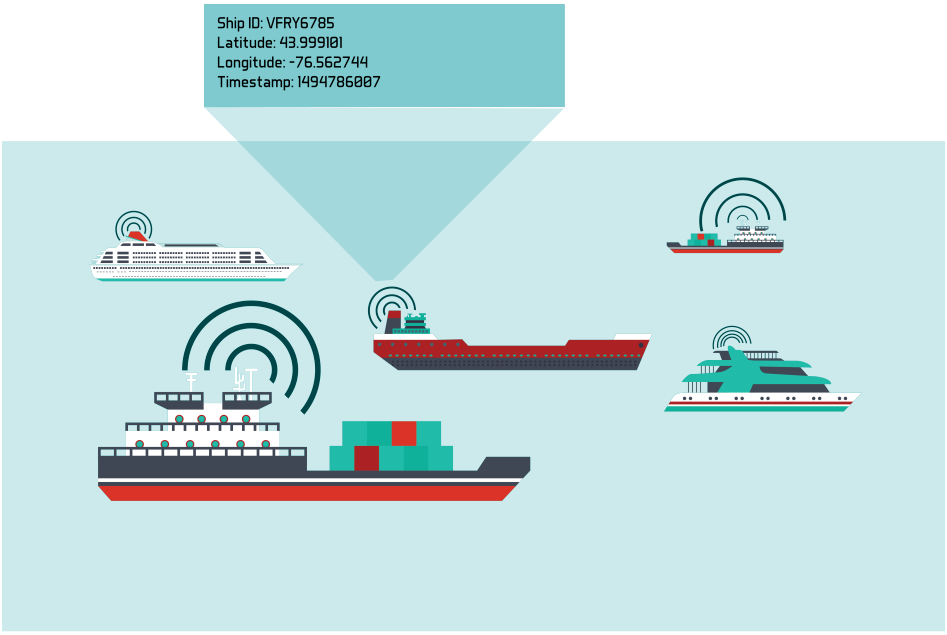

Fig. 2.3 Ships transmitting their AIS signal with their ID, location information and timing information at different intervals

2.2.1 Operating Principle

The AIS system retrieves the necessary information from on-board instruments and a GNSS receiver, transmits that information through a VHF signal and relies on Self-Organizing Time Division Multiple Access (SOTDMA) to avoid interference of signals transmitting at the same time [16]. Before transmitting the signal, each AIS transponder checks and announces its transmission time slot to other AIS transponders in the immediate vicinity - these are referred to as cells of ships.

Using SOTDMA ensures that there is very little to no interference amongst signals in the same cell. Furthermore, each ship transmits its signal at a variable time interval. This means that, depending on the ship's turning rate and speed, the interval at which it transmits its AIS signal can range from every second to once every 3 minutes (see Table 2.1 below [2])

Table 2.1 AIS Reporting Intervals

Ship Condition	Not Changing Course	Changing Course
At anchor or moored and speed <3 knots	3 min	3 min
At anchor or moored and speed >3 knots	10 s	10 s
Speed 0 to 14 knots	10 s	3.33 s
Speed 14 to 23 knots	6 s	2 s
Speed >23 knots	2 s	2 s

2.2.2 Satellite Receivers

While being able to receive signals from ships and land beacons within an 80km radius is sufficient for ship navigation and safety requirements, when it comes to global marine traffic monitoring it significantly limits the amount of information available. In order to break the 80km barrier, satellites were launched with their own AIS receivers in order to receive signals from ships worldwide. Commercial satellite operators, such as exactEarth and Orbcomm, maintain and monitor these satellites to receive available AIS signals around the clock. They have also worked to guarantee signal reception despite interference caused by being able to see multiple

ship cells at any one time instant.

The satellites to be studied are exactView satellites in various orbits. At an altitude of around 700km, their field-of-view spans approximately 50 degrees in latitude and 50 degrees in longitude on Earth's surface, assuming visibility out to the limb. The exactView satellites can receive the AIS data from ships and land beacons worldwide, meaning that at any given time anywhere from less than ten to many thousands of ships could be visible. Most of the time, there will be an abundance of data points to be used for computations, especially if the analysis time period extends over an entire orbit, for instance.

2.2.3 Challenges of Using AIS Signals

While the algorithm was developed with the goal of it being a general application of the orbit determination method proposed, the initial objective here was to apply it successfully to AIS signal data. These signals and their inherent operating principles also come with their own challenges; some applicable to other data forms as well and some unique to AIS itself. The most significant of these challenges is believed to be the transmission interval. As mentioned previously in this Chapter, each ship's transmission interval is a function of that individual ship's speed and turning rate. Given the large number of ships observed on average within the satellite footprint, it can be assumed that the vast majority of ships will be detected

at least once during the pass. It was, furthermore, initially assumed that all ships transmit their signal continuously in order to develop the base algorithm. However, since this aspect of the AIS transmissions was expected to have the largest impact on the performance of the algorithm, it was taken into account early on. The results can be found in Section 4.3 of this report.

Another challenge of AIS is the potential for missed messages. In satellite AIS operations, the focus is placed on ship detection rather than message detection, which could result in a message detection rate of 10% or less. This means that nearly all ships will be detected at least once but it is highly likely that many of the ships messages will not be detected. This can place an additional uncertainty on the ship visibility timing. However, statistics play an advantage here, once again, as many hundreds to thousands of ships will be within the satellite footprint on average at any given time.

Finally, the unknown altitude of the AIS transceiver (in terms of the transceiver location relative to mean sea level, as well as extreme weather conditions) and the potential for ionospheric bounce effects could potentially affect the overall performance of the algorithm.

3 Methodology

3.1 Input Data Simulation

As part of the initial stages of this research, STK by AGI Inc. was used to create a uniform grid of simulated ships and ground stations on a simulated Earth's surface orbited by a simulated LEO satellite (see Fig.3.1). This way, any transmission variations, signal source uncertainties and interference sources were eliminated such that a baseline proof-of-concept could be constructed. The information extracted from STK is the coordinates of each ship (assuming zero altitude at mean sea level) and the time information for each instant that each ship is visible. This way, the expected "true" visibility information is known for each ship beneath the satellite during a pre-determined time interval. In a real-world scenario, this data would come from the satellite payload rather than the simulation. In the later stages, the simulation was made increasingly realistic, such that it resembles the real-world AIS data scenario much more closely. However, the type of data collected remained the same.

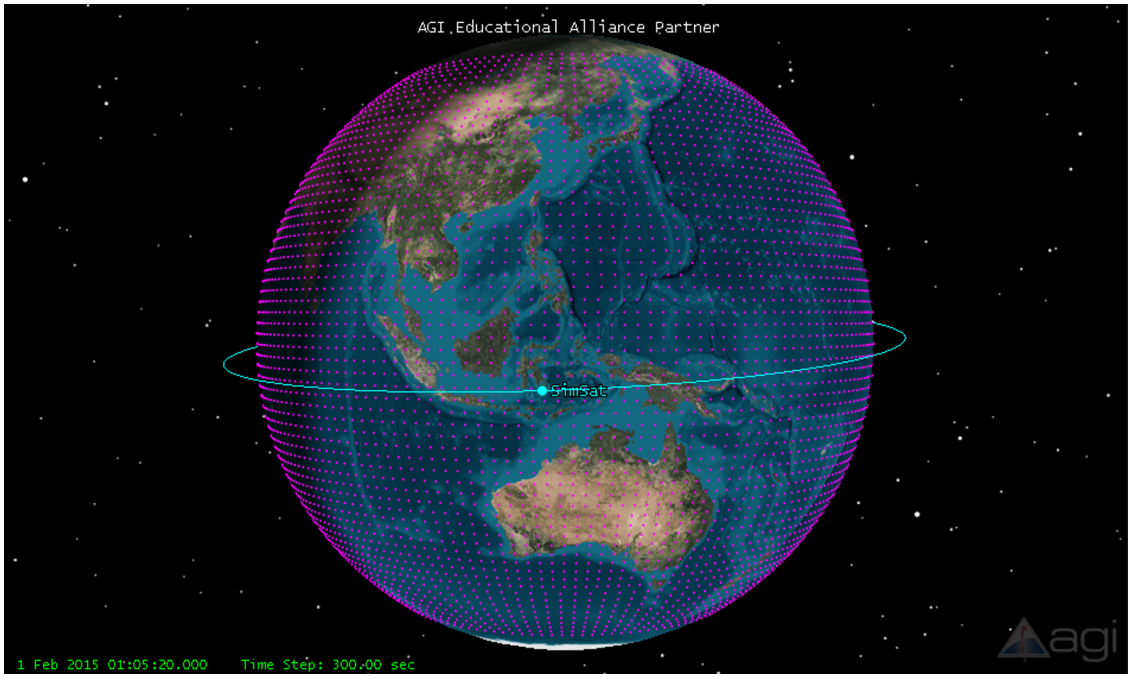

Fig. 3.1 STK scenario showing the grid points and satellite object in the ideal case simulation

3.2 Ground-track-based Estimation Method

The first attempt at using this visibility information to determine the satellite's orbit was to estimate the sub-satellite point on the surface and altitude and every time step over the course of one or more orbits.

3.2.1 Point Cluster Centroid

One way of estimating the sub-satellite path is by determining the location of the centroid of each point cluster (see Fig.3.2). Plotting these centroids over the

course of three orbits will give the result in Fig. 3.3, assuming a two-body point mass scenario. However, this method heavily relies on an evenly distributed and densely populated point cluster, which is unlikely to occur in a real-world setting.

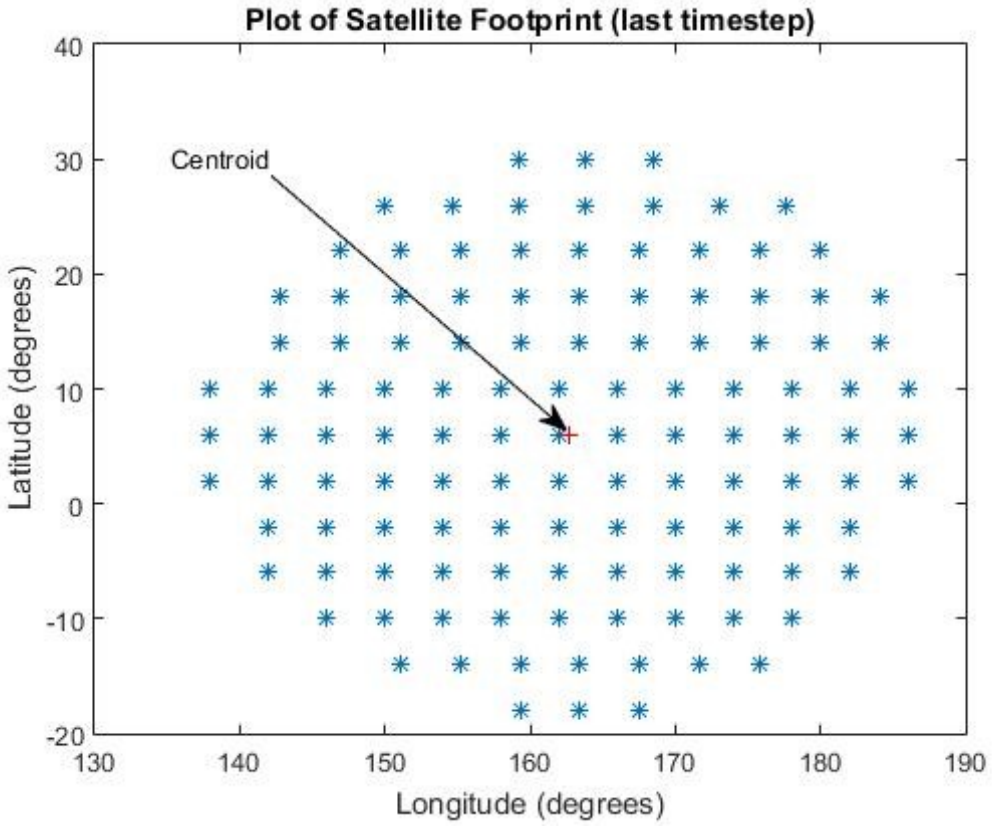

Fig. 3.2 Sample point cluster at one time step, using evenly distributed point cluster

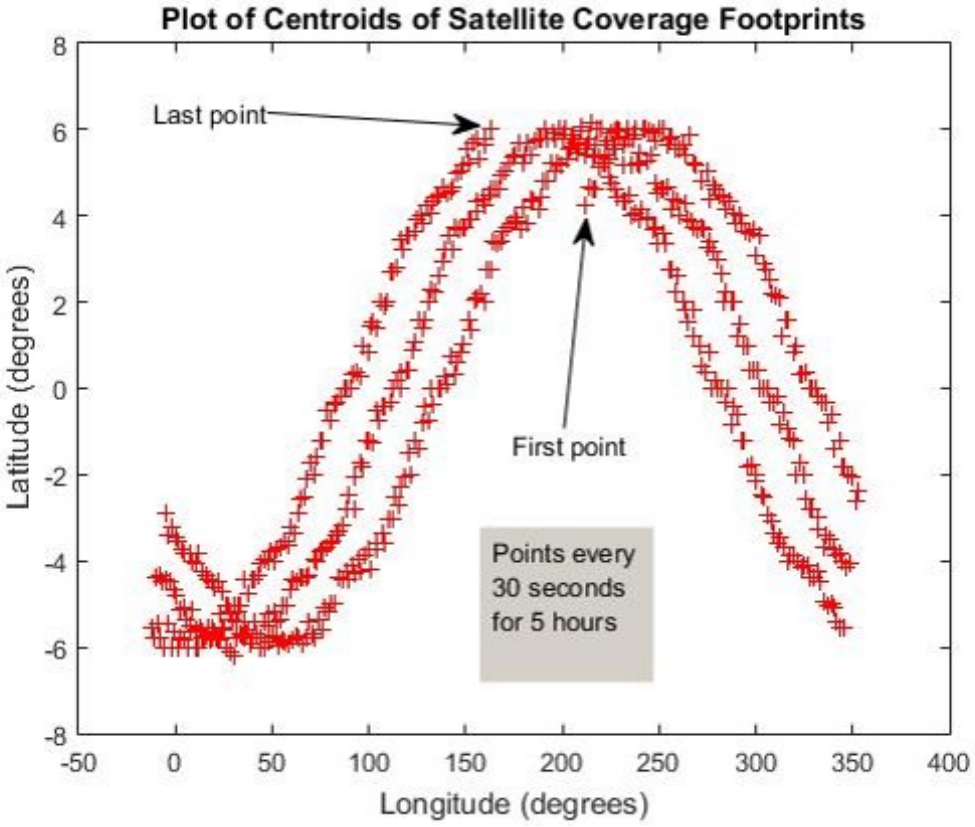

Fig. 3.3 Plot of the centroids of each point cluster at every time step over a period of 5 hours

3.2.2 Least Squares Intersection

In order to try and mitigate this, a Least Squares Intersection (LSI) method was designed such that the subsatellite point can be computed more accurately and more reliably. The centroid method and LSI method are compared in Fig. 3.4 for 300 randomly distributed data points. While the LSI method significantly

improved the result, the sub-satellite path estimation was still significantly noisy (with 1 degree of error in the sub-satellite point translating to over 10km of error on-orbit), even in the ideal two-body point mass scenario, which does not yet account for gravitational and other orbital perturbations.

Moreover, the satellite altitude estimation, which was based on geometry, is inherently very noisy and becomes significantly more complex when considering the variations in Earth's gravitational field. An accuracy better than 60km in altitude could not be achieved.

3.3 Simultaneous Perturbation Stochastic Approximation Method

After considering and testing multiple other orbit estimation techniques, such as a ground-track-based method, it was determined that using a gradient descent-based Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm would be a much more accurate, reliable and adaptable solution than the geometric ground-track-based estimation method.

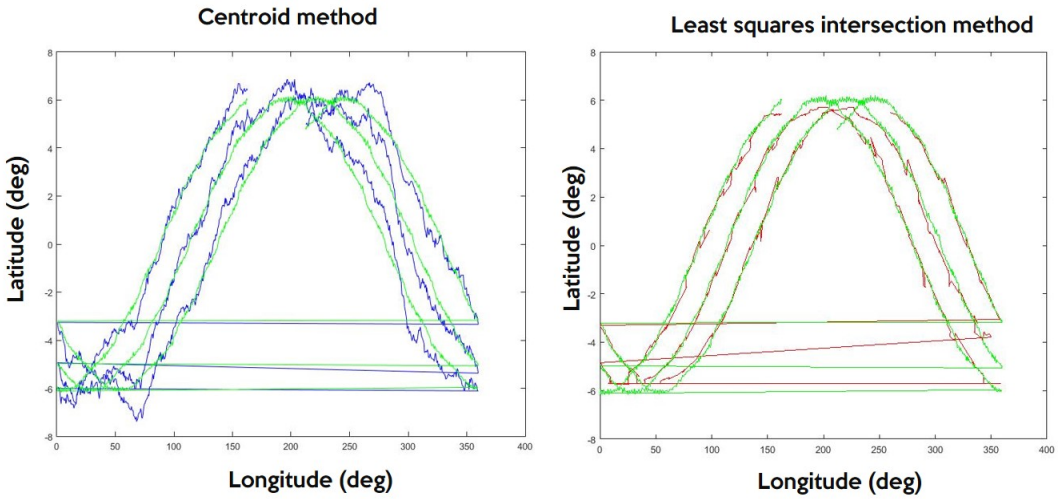

Fig. 3.4 Plots comparing the coordinates of the estimated sub-satellite point on the surface using the centroid method and the LSI method with randomly distributed data points - true orbit ground track is shown in green

3.3.1 Background

The original SPSA algorithm was developed by James C. Spall at Johns Hopkins University in the late 1990's. Its objective was to provide the ability to apply an optimization algorithm when a detailed model of the relationship between the optimization parameters is not available or easily determined. Rather than assuming that the user can compute the gradient numerically, the SPSA algorithm makes use of "perturbations" to the original functions to analytically determine the gradient. At its core, it is a gradient descent-based optimization algorithm, with added functionality for cases where there is no explicit gradient of the cost

function available. The algorithm developed in this paper uses the SPSA algorithm as a backbone but is fully customized, with added and removed elements, for this particular application.

3.3.2 The SPSA Algorithm

The original SPSA algorithm as developed by James C. Spall is very theoretical in nature and makes use of a variety of constant coefficients. The algorithm [32] thus starts with the "Initialization and Coefficient Selection" phase, followed by generating the perturbation vector, evaluating the cost function values, updating the parameter guess and iterating until the termination conditions are reached.

1. Initialize the parameters to be optimized to an initial guess Θ_0 and set $k = 0$

2. Select non-negative coefficients a, c, A, α and γ which will constitute the SPSA gain sequences:

$$a_k = \frac{a}{(A + k + 1)^\alpha}$$

$$c_k = \frac{c}{(k + 1)^\gamma}$$

   For more guidelines on how to determine such coefficients according to the original SPSA algorithm, please consult [32].

3. Generate a perturbation vector Δ_k of the same dimension p as the number of parameters to be optimized. Each of its p components are determined by

Monte Carlo and should be independently generated from a zero-mean distribution (for example, a Bernoulli distribution with a $\frac{1}{2}$ likelihood of assigning a ± 1 value to each component)

4. Evaluate the cost function measurements $y(\Theta_k)$ at:

$$y(\Theta_k + c_k\Delta_k)$$

$$y(\Theta_k - c_k\Delta_k)$$

5. Approximate the local gradient $g_k(\Theta_k)$ as:

$$g_k(\Theta_k) = \frac{y(\Theta_k + c_k\Delta_k) - y(\Theta_k - c_k\Delta_k)}{2c_k} \begin{bmatrix} \Delta_{k1}^{-1} \\ \Delta_{k2}^{-1} \\ \vdots \\ \Delta_{kp}^{-1} \end{bmatrix}$$

6. Update the parameter estimate Θ as:

$$\Theta_{k+1} = \Theta_k - a_k g_k(\Theta_k)$$

7. Increment k to $k+1$ and continue to iterate until the difference in the optimization parameters between iterations is sufficiently small.

The SPSA-based algorithm developed and used in this research is largely identical to the one outlined above. However, upon experimentation with a variety of different values and combinations of the gain sequence coefficients, no values within the ranges suggested in [32] resulted in proper convergence. Therefore, rather than

determining the coefficients individually, the gain sequences a_k and c_k themselves were determined experimentally. These two gain sequences represent the gradient step size (how far to move down the gradient) and perturbation step size (by how much to perturb the current best estimate), respectively.

3.3.3 Cost-function Selection

The choice of cost function, perturbation amount and step size are extremely important. After careful analysis of the timing information available and the effect each orbital element has on this information, it was determined that two types of cost functions can be identified. The start and stop time (SST) error is computed in two parts: the start time error for each ship by taking the difference between when it is first seen during the estimated orbit and when it is first seen during the "true" orbit, as well as a similar procedure with the time it was last seen for the stop time error. The SST error then is the RMS value of both the start and stop time errors combined over the observation period.

Fig. 3.5 illustrates the concept behind the SST error. As is evident in the figure, the ship of interest will enter the footprint along the estimated satellite orbit prior to entering the footprint along the "true" orbit. For the sake of argument, let us assume the estimated satellite first observed the ship at 1750 seconds. Since the "true" satellite would only start seeing the ship at the current time step (1800

seconds), the start time error would be -50 seconds. Similarly, the ship would pass out of view from the estimated satellite approximately 45 seconds from now at 1845 seconds, whereas it would pass out of view from the "true" satellite approximately 100 seconds from now at 1900 seconds. The stop time error would then be -55 seconds. This would be repeated for each ship of interest, after which the RMS value will be taken of the start time errors and stop time errors all at once.

The total visibility length (TVL) error is computed by taking the difference between the total time a ship is observed by the estimated satellite and the "true" satellite, then taking the RMS value over the entire observation period. For the ship in Fig. 3.5, the TVL error would be $95seconds - 100seconds = -5seconds$. The TVL error points to an offset predominantly in the radial and/or cross-track direction.

This means that a and i have a negligible effect on the SST error (as they do not cause any along-track offset) and can only be estimated using the TVL error cost function. However, ω and θ produce almost entirely along-track variations and are therefore most accurately estimated using the SST error function. Finally, e and Ω produce significant effects on both error metrics and can thus be estimated using both cost functions.

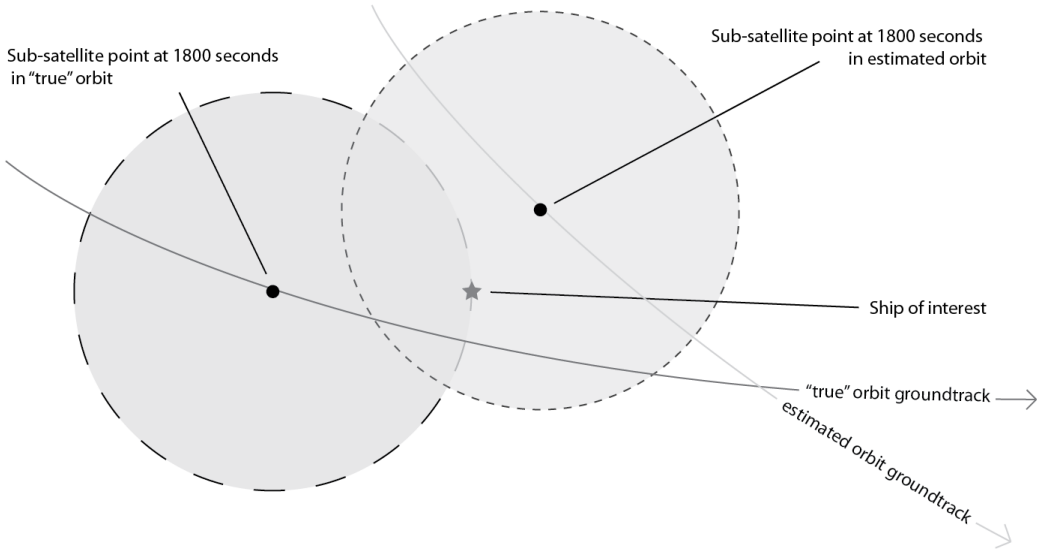

Fig. 3.5 Figure showing the difference in visibility start and stop time for an arbitrary ship of interest

3.3.4 Search-range Boundaries

It is assumed that the most accurate first guess we have to start the optimization would either be based on the latest available TLE data or a previous estimate, both no older than 2 weeks. Given this assumption, the search range for each of the orbital elements can be constrained as presented in Table 3.1.

Table 3.1 Orbital Elements Search-range Boundaries

a	15km in each direction
e	1 order of magnitude in each direction
i	1.5 degrees in each direction
Ω	0 to 360 degrees
ω	0 to 360 degrees
θ	0 to 360 degrees

These boundaries significantly decrease the computation time, and can be assumed to do so without loss of generality.

3.3.5 Customized Algorithm Methodology

3.3.5.1 Segment 1: Reading in the "true" visibility information

To start, the algorithm takes as input the simulation-generated coverage file. This will contain each observed data point and note its identification number, latitude, longitude, number of accesses (including if it is 0) and a list of each access time interval in seconds from epoch. This information is similar to what would be available from the AIS data, except that the time information will not be an interval in that case but a UNIX timestamp. This is because the AIS signal is not transmitted continuously. However, the AIS-based UNIX timestamps could be converted to time intervals in a separate function.

3.3.5.2 Segment 2: Analysis window and search range constraints

This section will define the start and stop time of the analysis window and the upper and lower bounds on the orbital element search range. The variable *firstinview* specifies when the analysis will start, in terms of seconds from epoch. In other words, what is the earliest visibility start time any of the analyzed points can have? Similarly, *lastinview* specifies the latest visibility start time. For instance, the STK scenario that is used to generate the "true" visibility data has a length of 4 hours - approximately 1.5 hours before the time of interest and 2.5 hours after the time of interest. That way it can be assured that all analyzed points will have a complete visibility interval, not cut-off by the starting and stopping of the STK simulation. The algorithm then uses these two variables to specify what portion of this STK-generated data will actually be used for the optimization.

Important to note is the "time of interest". This refers to the exact time instance (in UTC time) for which orbital elements of the satellite will be estimated. This will also be the equivalent value (in seconds from epoch instead of UTC) of *firstinview*. Given that an average orbit for the simulated satellite is just over an hour long, *lastinview* was set to 3600 seconds after *firstinview*. However, the algorithm can also handle analysis windows of longer than one orbit and distinguish between the multiple times a single point is seen over multiple orbits. This can prove to be useful

in estimating slow-changing orbital elements more accurately later on by capturing more of their effect in the data collection.

As mentioned previously, knowing the upper bound on the age of the initial guess (for instance TLE data of at most 2 weeks old) allows us to impose boundaries on the search range. These are defined as *LBounds* and *UBounds*. The bounds are defined as outlined in section 3.3.3 above.

3.3.5.3 Segment 3: Generate the "true" point set for analysis

Based on the defined *firstinview* and *lastinview*, this section generates the matrix containing all of the points to be analyzed that fall within this interval, referred to as *compPointSet*. It will store their point ID, latitude, longitude, visibility start time, visibility stop time and total time of visibility. Later on in the code, the estimated orbit will generate a similar matrix *adjPointSet* with the same format for comparison.

3.3.5.4 Segment 4: Variable transmission interval settings

The initial assumption of a continuous transmission interval for each individual ship was expected to be the one with the largest impact on the performance of the algorithm. Hence, it was one of the first improvements made to the algorithm to bring it closer to reality. This segment of the code sets the percentage of ships

transmitting at each respective interval (see Table 3.2) and then adjusts the visibility interval values for each respective population of ships with randomly assigned point IDs. This is done by setting the visibility start time and stop time of each ship to a multiple of its assigned transmission interval. Since only the real-world ships are affected by the variable transmission interval and not our simulations of the estimated orbits, this is only done once and only to *compPointSet*.

Table 3.2 Variable Transmission Interval Ship Distribution Assumption

Transmission Interval	Ship Percentage
180 s	5% of ships
30 s	10% of ships
10 s	46% of ships
2 s	30% of ships
1 s	9% of ships

3.3.5.5 Segment 5: Initializing the orbital elements

The initial orbital element guess would ideally be either the latest available TLE data for the satellite (which would be at most 2 weeks old) or the last estimation, whichever is newer. Currently, the initial guess contains the orbital elements based on the TLE data set. These orbital elements form the Orbital Elements Guess (OEG) and will be updated throughout the optimization.

Once initialized, the function *genSTKScript* produces the script that will be used to run the STK simulation for the orbit estimate. The function *runSTKScript* will then take as input the script generated in the previous step and execute it. This

will open up a new instance of STK, run the simulation, save the new *.cvaa* file and then exit the STK instance. This leaves behind the *.cvaa* file for *readCVAAfile* to read through and output a matrix of point ID's and their respective latitude, longitude, visibility start time, visibility stop time and total visibility time called *adjPointSet*. In other words, it will have the exact same format as *compPointSet* such that both matrices can be compared. This is done through the function *compErrors* which takes as input both matrices and computes the SST and TVL errors for each analyzed data point. As a means for stabilizing the algorithm, all errors are filtered such that only those within 2 standard deviations are retained. All others are considered outliers and have an adverse effect on the error metric computation.

3.3.5.6 Segment 6: Applying the weighting factor

Applying a weighting factor to the error metrics of each individual point/ship compensates for the misinterpretation due to the variable transmission interval. Initially a linear weighting factor was applied where each low-confidence error metric was multiplied by 90% and the high-confidence error metrics were left untouched. While this worked well in some cases, when the date of interest was changed and the initial error values were significantly larger, the algorithm failed to converge. This was avoided by changing the weighting factor to an exponential function, raising

the low-confidence error metrics to the power of 0.985. This value was determined experimentally to have the highest reliability when testing different dates of interest but should be refined and studied further.

3.3.5.7 Segment 7: The optimization loop

This segment starts by initializing the *termTVL* variable to 0. This variable will track the amount of consequent iterations in which the TVL error (which is the only error metric being used as the cost function) difference is less than 0.1 seconds. Once this amount exceeds 3 iterations, the optimization loop is terminated. This was found to be superior to setting a limit on each orbital element step size between iterations or terminating when the TVL error difference falls below a pre-determined value just once. The reason for the latter being that in order for this to be a viable termination condition, the error difference would have to be sufficiently small, but when set this small it might sometimes never be reached. When then set to a larger value, like the current 0.1 seconds, this may by chance occur somewhere earlier on in the optimization just once and should, therefore, not yet be a reason for termination.

3.3.5.8 Segment 8: Generating and applying the perturbation vectors

The next step is to generate the negative and positive perturbation vectors. First, a vector of 6 values (this was eventually changed to 5 values to reflect the incorporation of the argument of perigee into the argument of latitude - see section 4.1) is created, with each value being randomly assigned a direction coefficient of +1 or -1 with equal probability. Next the step sizes are defined for each orbital element, as a function of the TVL error value at the end of the previous iteration. These values were determined by studying the behaviour of the optimization for a wide range of intial guesses and dates of interest such that the algorithm can converge within reasonable error and in a timely fashion.

Once the direction coefficients and step sizes have been determined, they are combined to form the perturbation vectors. In order to get a good approximation of the gradient, the perturbations are set to be twice the step size when the TVL error is greater than 100 seconds and 1.5 times the step size when the TVL error is less than or equal to 100 seconds. The perturbation vector is then added to and subtracted from the previously determined OEG to create a positive perturbation *OEG_PP* and a negative perturbation *OEG_PN* to the orbital element guess. Please note that as part of these perturbations, each orbital element will be increased and decreased by their own respective amounts. For instance, the semi-major axis may

be changed by 2000km, whereas the inclination would only be changed by 1 degree during the same perturbation. Each perturbation is then checked to make sure it falls within the pre-determined bounds and the semi-major axis is checked so that the perigee altitude does not fall below 100km.

Once again, an STK script is generated and executed, the *.cvaa* file is read and used to generate a new *adjPointSet*, the error metrics are computed, and the appropriate weighting factors are applied. This is done for each of the two perturbations.

3.3.5.9 Segment 9: Generating the new best estimate

Based on the results from the positive and negative perturbations, the gradient can be determined based on the difference between the two TVL errors. The OEG vector elements are then adjusted along the downward direction of the gradient (towards zero) by the same step size as determined in the beginning of Segment 9. This forms the new best estimate for the satellite orbit.

Finally, the elements for the new best estimate are checked to be within bounds and to satisfy the perigee altitude requirement. Once again, an STK script is generated and executed, the *.cvaa* file is read and used to generate a new *adjPointSet*, the error metrics are computed, and the appropriate weighting factors are applied. The final TVL error value for this iteration is then computed so that it can be used to determine the step sizes in the next iteration. This TVL error value is compared

to that of the previous iteration and $termTVL$ is incremented if this difference falls below 0.1; otherwise, $termTVL$ is reset back to 0.

This concludes the iteration and the process is repeated until $termTVL$ reaches a value of 3, giving us confidence that the algorithm has converged properly and sufficiently.

3.4 Genetic Algorithm

The Genetic Algorithm (GA) has the inherent advantage of being able to entirely traverse a large search space, efficiently. Following some of the issues encountered with the gradient-descent based algorithm, as explained further in section 4.1, it was determined that designing a GA would overcome those challenges and increase the stability and consistency of the algorithm. As opposed to searching a narrow range along the gradient, risking being caught in local minima as with the gradient descent approach, a GA introduces randomness in order to search the entire space and increase flexibility of the algorithm. The GA has been used as a successful method of orbit determination in various other applications before, such as ground-based or space-based optical streak observations of a satellite [24] and optical observations of a GEO satellite [17]. However, these solutions focus on observations of the satellite itself and not its payload observations.

3.4.1 Background

The GA, as the name suggests, is derived from natural genetic reproduction processes. It, generally, starts off with a random initial population of individuals and chooses a set of parents from these individuals. The parents are chosen based on their cost function and represent a large enough variety so as to avoid local optimization. From these parents, children are then created through three different reproduction techniques: direct copy, mutation and recombination. The key lies in balancing the rates of reproduction of each technique such that enough randomness is introduced but not so much that the algorithm fails to converge.

The implementation of a GA introduces some distinct advantages over the previous gradient descent-based approach. For one, it does not require any initial orbit guess to seed the optimization. This is a significant advantage as it removes any bias the user might inject with the initial guess and generalizes the algorithm to any initial guess, within the boundaries specified in section 3.3.4. The GA does this by initializing the optimization with a population of randomly generated initial guesses within those pre-determined boundaries. The boundaries are sufficiently large so as to guarantee that the correct orbit lies within those boundaries but sufficiently small such as to avoid searching any unnecessary orbital element values.

The GA also eliminates any need for a pre-defined step size function, for both the

gradient estimation and the perturbation size since both are not longer required operations. This also significantly generalizes the algorithm as it works towards convergence and chooses its own step sizes based on the reproduction method definitions. The algorithm then determines independently which step sizes generate more suitable results as it explores different orbital element combinations. As will be explored in section 4.1.2 as well, the need for a step size function in the gradient descent-based optimization is a limiting factor in the robustness and consistency of the algorithm. The particular requirements of the step size function actually change with the date of interest, so choosing a date of interest sufficiently far into the future, say 6 months further than tested with the current step size function, will cause the algorithm to be extremely unlikely to converge.

Additionally, the GA no longer requires a weighting function to be implemented in order to compensate for the varying transmission intervals. Whereas the gradient descent-based relied on the actual cost function values in order to determine the progress of the optimization and proximity to convergence, the GA only considers the relative cost function values throughout the optimization. This allows the GA to determine the proximity of a given estimation to the "true" orbit without needing a weighting function to normalize the cost function value to a similar range as when all the ships transmit continuously.

Finally, the inherent ability of the GA to search the entire defined search space

and minimize the risk of converging on a local minimum is extremely desirable for this particular research application. The estimation of satellite orbital elements based on its observations is one with a highly irregular cost function slope, meaning there are many local minima that the optimization could consider the correct estimate if not handled appropriately. By introducing a sufficient amount of randomness into the search process, as will be explored in more detail in section 3.4.2, the GA can effectively eliminate estimates corresponding to local minima and continue its search along a path of global optimization. Refining the estimate locally once the global minimum has been found can then be achieved through employing a different search technique in combination with the GA. However, the key here is that this should only be done once the area in the global space containing the global minimum has been located.

3.4.2 The Genetic Algorithm

The GA is in itself a highly customizable type of algorithm. While a general procedure is always followed, the internal operations of a genetic algorithm can vary widely.

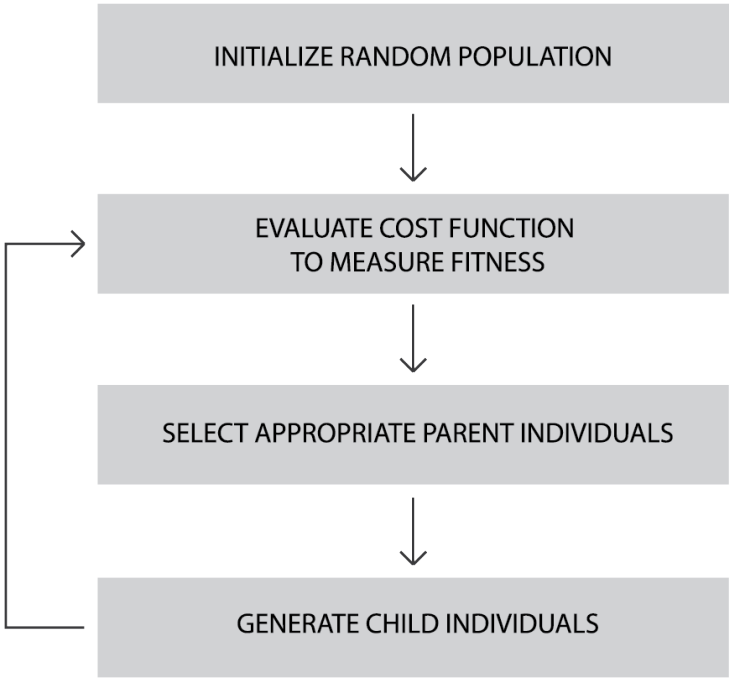

Fig. 3.6 Outline of the general genetic algorithm

In general, the following steps are always present in a genetic algorithm (see Fig. 3.6):

1. Initialize a random population of individuals. Each individual constitutes a valid guess of the optimization parameters. That is, they must have the same dimension as the optimization parameters and contain appropriate values that can be evaluated without error

2. Measure the fitness (i.e. cost function value) of each individual in the population

3. Select the parent individuals

4. Generate the child individuals from the parents individuals through direct copy, recombination or mutation

5. Continue to iterate from Step 2 until the termination conditions have been reached

The next section will describe the customized GA that was developed for this research from the ground up, and the changes that were made in order to make it most effective for the optimization at hand. These custom changes include:

1. The manner in which parent individuals are select to maintain enough variety while at the same time working towards convergence

2. The manner in which child individuals are generated and when to resort to each of the reproduction techniques based on the behaviour of the optimization

3. The addition of the simulated annealing-based minimization at the end of each iteration in order to refine the orbital element guesses further, but still avoid landing in a local minimum.

4. The manner in which proximity to convergence is determined

3.4.3 Customized Algorithm Methodology

This subsection will explore the details of the GA as implemented for this particular research project. The algorithm was designed and implemented completely from the ground up, with many customizations to make is as effective as possible in determining the correct orbital elements estimate for a simulated satellite. Segments 1 through 4 in the GA are identical to Segments 1 through 4 in the gradient descent-based optimization algorithm.

3.4.3.1 Segment 5: Generating the STK point file

In order to speed up the algorithm and avoid computing the visibility of points the satellite won't pass over, the algorithm creates an STK point file with all the points seen during the "true" orbit. This was not implemented in the gradient descent-based algorithm as it does not allow the algorithm to compare data points that are not seen during the true orbit but may be seen during the estimated orbit. The gradient descent-based algorithm made use of this additional data to estimate the gradient of the cost function. However, the GA does not require this additional data and it was decided that significantly increased performance warranted implementing the point file generation. It is also important to note that the point file will assume an altitude of 0 meters for each data point. Even if the

actual data points vary in altitude during the "true" orbit, this assumption does not change for the points created for the estimate orbit to observe.

3.4.3.2 Segment 6: Parent selection

The parent selection process is crucial to guaranteeing a sufficiently wide, yet converging, traversal of the search space. In order to have an appropriate variety of parents, the particular selection process in this algorithm first looks at the four individuals with the lowest Total RMS (TRMS) error value. This error value is computed by first taking the RMS value of the start time error, the stop time error and the TVL error for each data point and then taking the RMS of all those individual RMS values. Next it selects the 2 individuals with the lowest start time error RMS, the lowest stop time error RMS and the lowest TVL error RMS, respectively. It is important to note, again, that the start and stop time errors are predominantly used for estimating orbital elements that represent along-track motion of the satellite. The TVL error, on the other hand, relates to the radial and cross-track components. Finally, the TRMS error combines all three error metrics in one cost function and will therefore be used as the main cost function to measure the overall progress and success of the optimization. By selecting a subset of parent individuals with low error values in each of the aforementioned cost functions - start error, stop error, TVL error and TRMS error - guarantees that each parent will

have at least one strength in the orbital elements, along with a set of parents (those based on the TRMS error) that should have an overall good representation of the true orbital elements as the algorithm approaches the global minimum.

In order to introduce additional randomness to the GA, a number of random individuals are selected to be part of the parent pool based on the lowest TRMS error value of the previous iteration. If that value was larger than 200 seconds, 5 random parents individuals are selected. This decreases by 1 parents individual for every 50 second decrease in the lowest TRMS error value of the previous iteration. When that minimum TRMS value is less than 50 seconds, only 1 random parent individual will be selected.

The algorithm then generates a matrix of all possible unique combinations of parent individuals. If more than 20 possible combinations result, random combinations will be removed in order to limit the number of combinations to 20. This does not affect the overall accuracy of the final estimate as there is still a large variety in the combinations, and these combinations change randomly at every iteration, but it significantly decreases running time.

3.4.3.3 Segment 7: Generating the new population of children

The population of children is generated through three main reproduction techniques: direct copy, recombination and mutation. Each parent couple is assigned

a reproduction sequence, meaning that each orbital element of the child will be assigned one of the three reproduction techniques. The percentage at which each technique is assigned is determined based on the current state of the algorithm. If the algorithm has detected an early convergence (more on this later), it will assign 50% of the orbital elements to recombination and 50% to mutation in order to increase the random behaviour. If the algorithm has detected that it is close to convergence (more on this later), it will increase the direct copy percentage and decrease the mutation percentage. This ends up being divided as 30% direct copy, 65% recombination and 5% mutation. It is important to still maintain a minimal level of randomness, hence why the mutation percentage is left at 5%. Any other situation will prompt a direct copy percentage of 0%, recombination of 80% and mutation of 20%.

Each parent couple is set to create two children. Each orbital element of each of the two children from the same parents will be generated using the same reproduction technique, but the values will be different. To explain this idea further, in the case of direct copy, child #1 with get a copy of the respective orbital element of parent #1 and child #2 will get a copy of the orbital element of parent #2. In the case of recombination, the recombination will happen with respect to each parent respectively. Finally, when the orbital element is assigned a mutation, the orbital element will be mutated by a relatively small amount for the first child, and by a

relatively large amount for the second child.

The recombination technique operates by taking the difference between the values of the two parents for the particular orbital element that was assigned to be reproduced by recombination. It then assigns a random recombination percentage between 10% and 90%. This percentage is multiplied by the parental difference and added to the respective parents' orbital element value. For the first child it will be added to parent #1 and for the second child it will be added to parent #2.

The mutation technique operates similarly to the recombination technique. However, at the end of the recombination technique, a random mutation is applied to the orbital element in question.

Once the child orbital elements have been generated, the cost function values are computed for each child.

3.4.3.4 Segment 8: Close Neighbour Approximation

In order to refine the estimation results further, a close neighbour approximation is performed on the current parent population. First, a step size is set depending on the lowest cost function value of the previous iteration in accordance with ranges presented in Table 3.3:

Table 3.3 Close Neighbourhood Approximation Range Values for the Orbital Elements

a	5km to 50m
e	5E-4 to 5E-6
i	0.5 to 0.005 degrees
Ω	5 to 0.05 degrees
ω	5 to 0.05 degrees
θ	5 to 0.05 degrees

Subsequently, each parent individual of orbital elements is perturbed by that step size in the positive and negative direction. The cost function values for each of these perturbations are then compared to one another and the lower one of the two, along with its associated perturbation direction, is chosen. The refining process continues in this direction and with the same step size as long as the new cost function value is lower than the original parent cost function value or 5 evaluations have been performed.

A process similar to simulated annealing is then used to determine which orbital element estimate will continue as a parent individual to the next iteration: the original parent or the slightly perturbed parent with a lower cost function value. Inherent to this method, early on in the optimization process the probability is higher for the original parent to be chosen, even though its cost function value may be higher, if the two cost function values are close together. If the cost function value of the perturbed parent is significantly lower than that of the original parent, the probability of it being chosen increases. The reason why this is useful for the particular optimization in question is that, early on, there should be enough variety in the parent pool, and, unless a significant improvement can be made,

there should be no close-neighbour optimization yet. However, later on in the optimization process, there is more confidence that the algorithm is getting close to the "true" value, and therefore close neighbour approximations are more useful at this stage.

3.4.3.5 Segment 9: Determining Proximity to Convergence

The final step is to determine whether or not the algorithm is approaching convergence. This proximity to convergence can be estimated by two different metrics: the difference between the parental cost function values and the difference between the cost function values of subsequent iterations. In order for the algorithm to detect full convergence, it must encounter two conditions:

1. the maximum and minimum of the parental cost functions values must be within 1 second from each other for 5 subsequent iterations

2. the minimum cost function values of two subsequent iterations must be within 1% of each other 10 times in a row

One might notice that the actual cost function value is not used as a convergence condition. The reason for this is that, while in theory the cost function value should approach zero closely, all the noise introduced to the measurements and cost function error metrics, such as the transmission interval, the cost function may

not necessarily be at zero when the algorithm converges to an estimate close to the "true" value. Moreover, the cost function value at which this happens cannot confidently be predicted. Hence, convergence is instead determined by the relative cost function values between parents and between iterations.

4 Optimization Results

4.1 Preliminary Gradient Descent-based Optimization

In order to test the conclusions metioned above regarding the cost function usage, the algorithm was generated such that all orbital elements were estimated first using the TVL error cost function. While doing this, it was known that ω and θ would not be estimated accurately and would need to be refined. Hence, a second phase of optimization then keeps a, e, i constant at the estimated value, and refines ω, θ and even Ω. The Ω element would likely be within reasonable error after the first phase but can be further refined using the SST error cost function as well. For testing purposes and based on available TLE data, a date of interest of February 1, 2016 00:30:00.000 UTC was used in the results below. Please note that the "RMS value" takes the RMS value of the 10 last iterations.

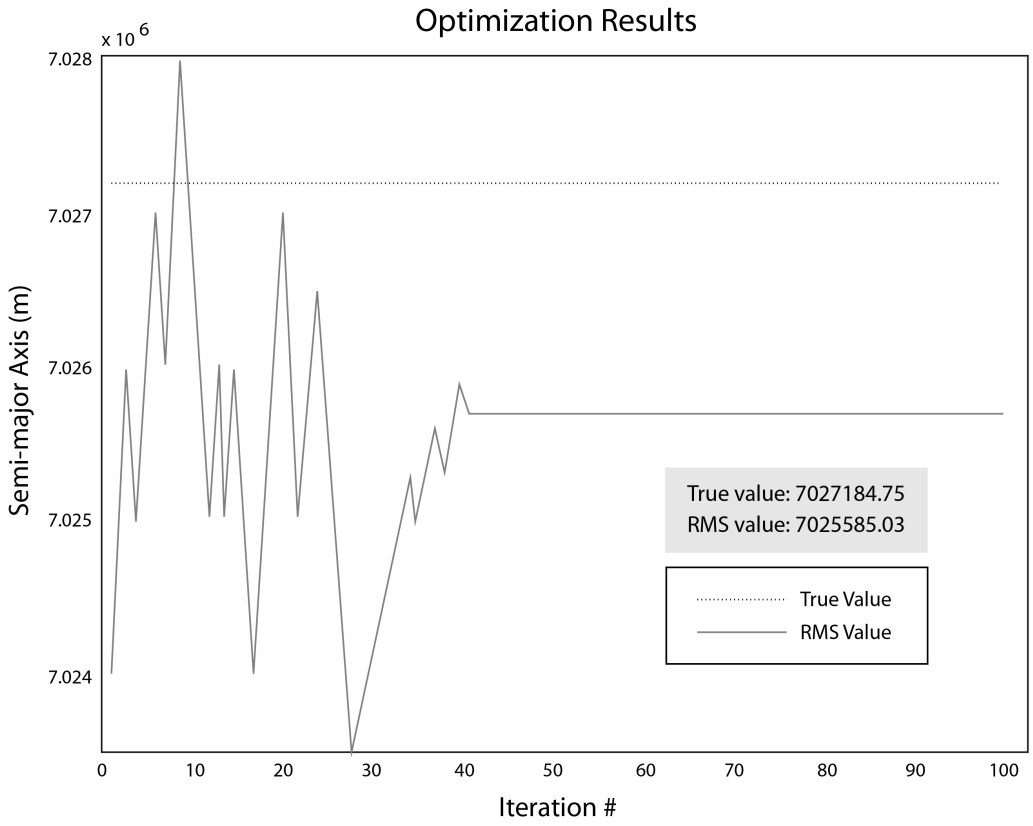

Fig. 4.1 Modified SPSA algorithm results for the semi-major axis at Feb 1, 2016 00:30:00 UTC

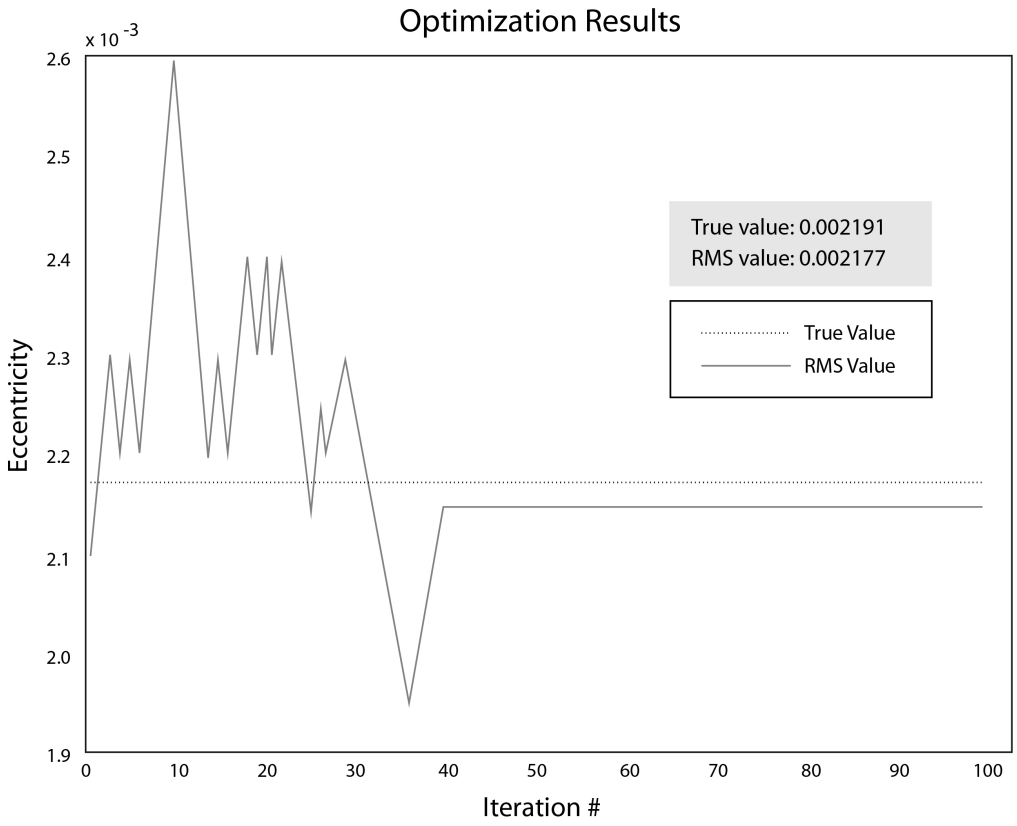

Fig. 4.2 Modified SPSA algorithm results for the eccentricity at Feb 1, 2016 00:30:00 UTC

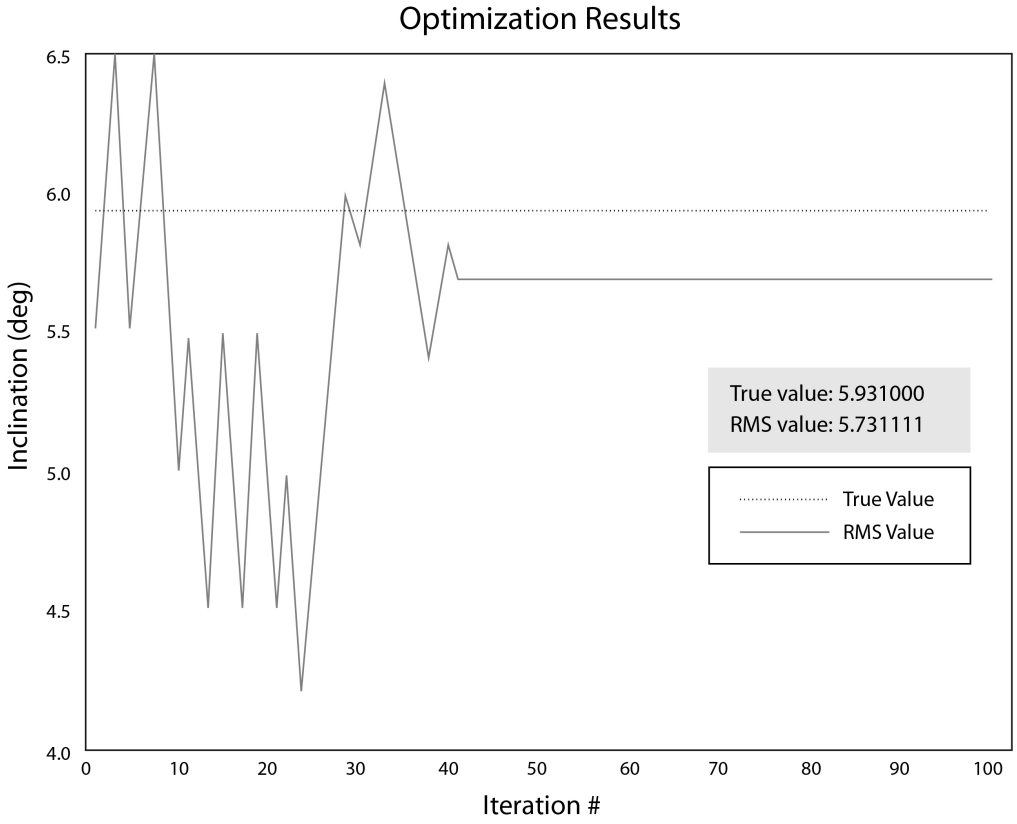

Fig. 4.3 Modified SPSA algorithm results for the inclination at Feb 1, 2016 00:30:00 UTC

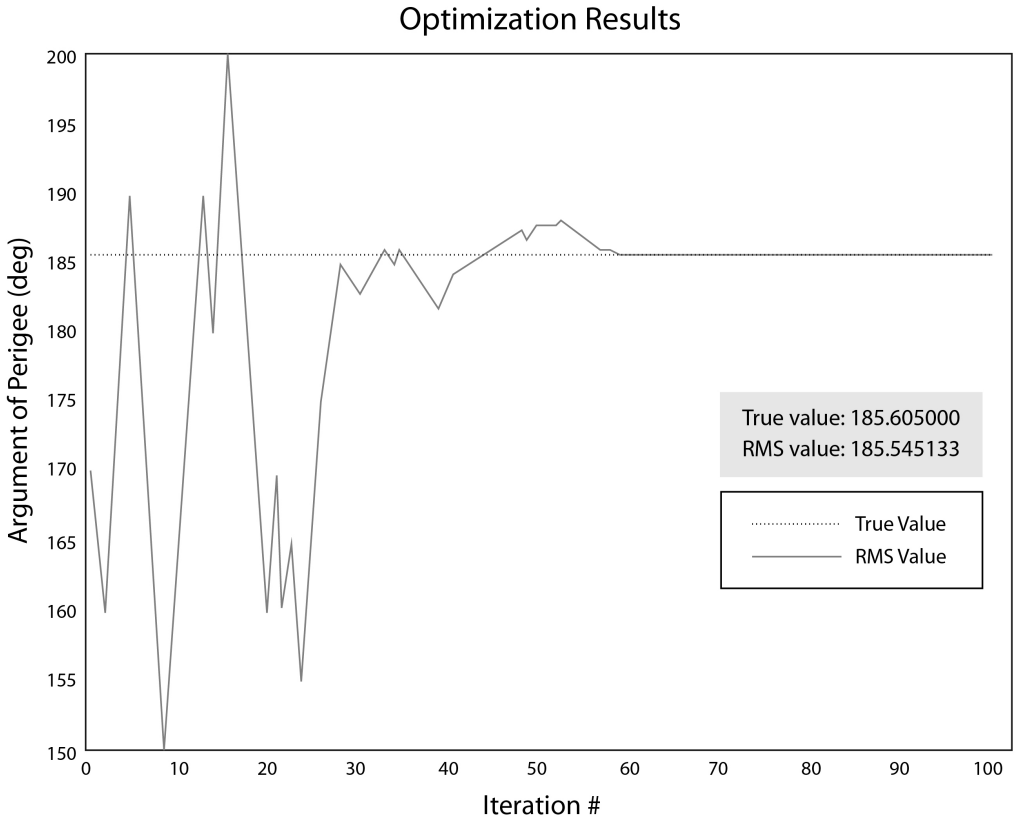

Fig. 4.4 Modified SPSA algorithm results for the argument of perigee at Feb 1, 2016 00:30:00 UTC

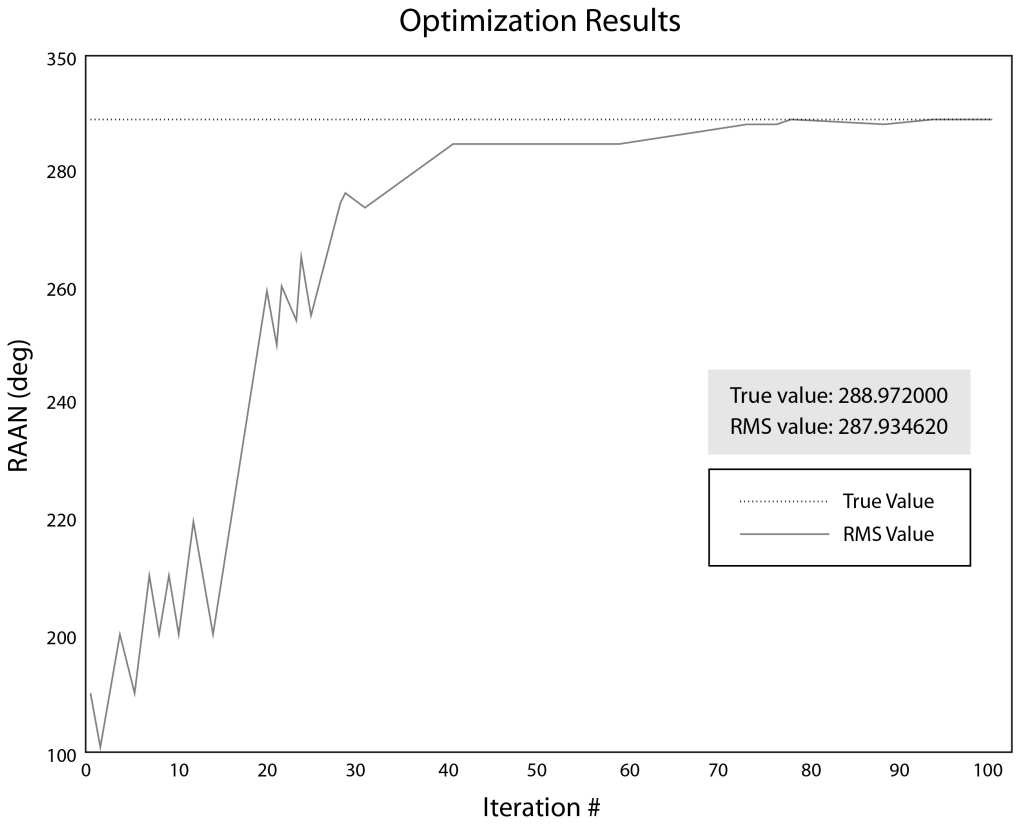

Fig. 4.5 Modified SPSA algorithm results for the RAAN at Feb 1, 2016 00:30:00 UTC

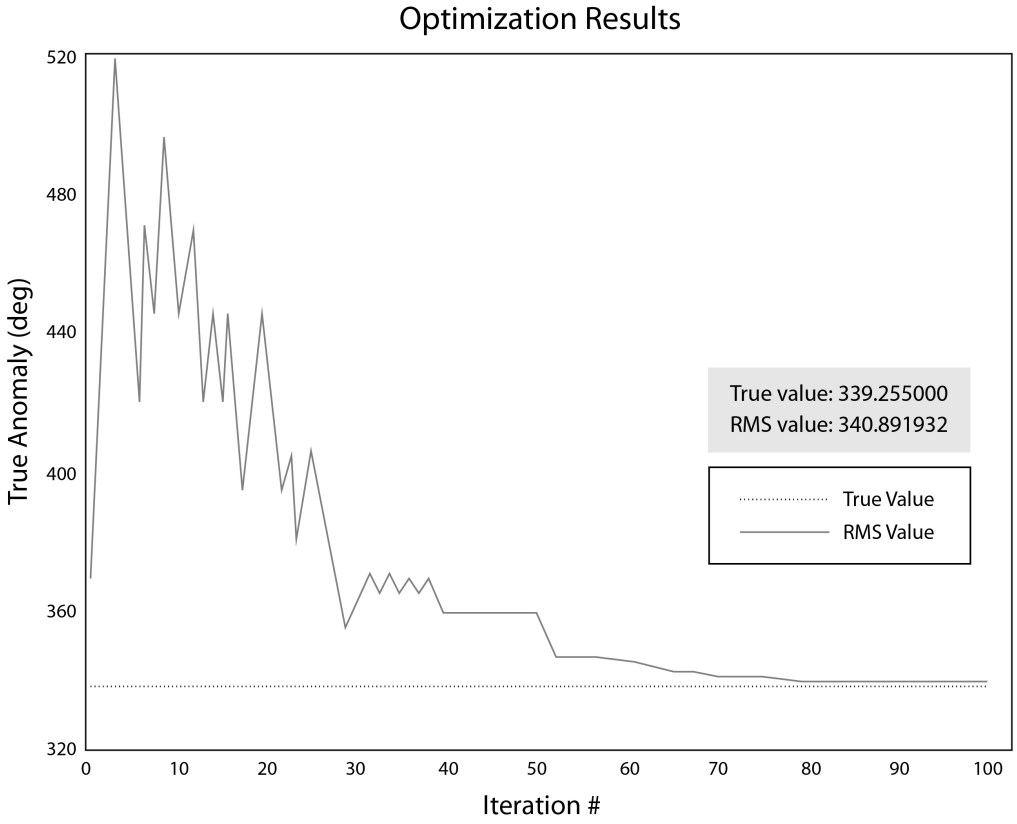

Fig. 4.6 Modified SPSA algorithm results for the true anomaly at Feb 1, 2016 00:30:00 UTC

The technique converges to the correct orbital elements, using the simulated evenly-distributed data points (see Figs 4.1-4.6). The initial guess was based on a typical two-week-old TLE set orbit estimate. It can be seen that, as expected, the first three orbital elements converge very closely after around 40 iterations using the TVL error cost function. The ω, Ω and θ are then further refined using the SST error cost function until they too converge.

It is important to note that the cost functions used in this optimization consider only the ships seen during the estimated orbit and compare those to the same ships as seen during the "true" orbit. Additionally, due to the mostly low eccentricity of the exactView satellite orbits, the argument of perigee is highly unstable and changes very rapidly. Hence, it was determined that estimating the argument of perigee is rather meaningless and, instead, the combination angle of the true anomaly and argument of perigee will be considered. This will be referred to as the argument of latitude, u.

In order to improve running time and the overall accuracy of the optimization, a method to estimate u using only the TVL error cost function was developed. Whereas ω and θ could initially only be estimated using the SST error cost function, by expanding the range of ships being compared, the sum of the two elements can be highly accurately determined using the TVL error cost function. This way, the estimation of u can be combined with the other orbital elements all at once. Instead of only comparing the ships seen during both the estimated orbit and the "true" orbit, we now also include the ships that are seen but should not be seen and the ships that are not seen but should have been seen. This cuts down the running time almost in half and makes sure that the error metrics are more accurate and complete.

First, the optimization was run (see Fig.4.7-4.11) for the same date of interest

as the previous test, set to February 1, 2016 00:30:00 UTC. The perturbation size P was also adjusted prior to this optimization such that it can be used on a wider range of values and is applicable to all dates of interested without alteration. Please note that the "RMS value" takes the RMS value of the 10 last iterations.

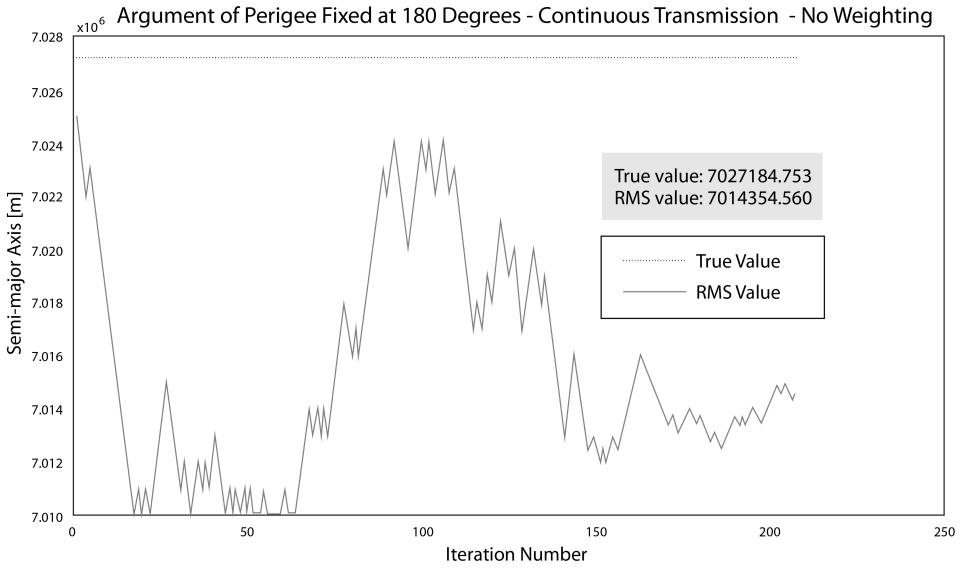

Fig. 4.7 Single cost function results for the semi-major axis at Feb 1, 2016 00:30:00 UTC

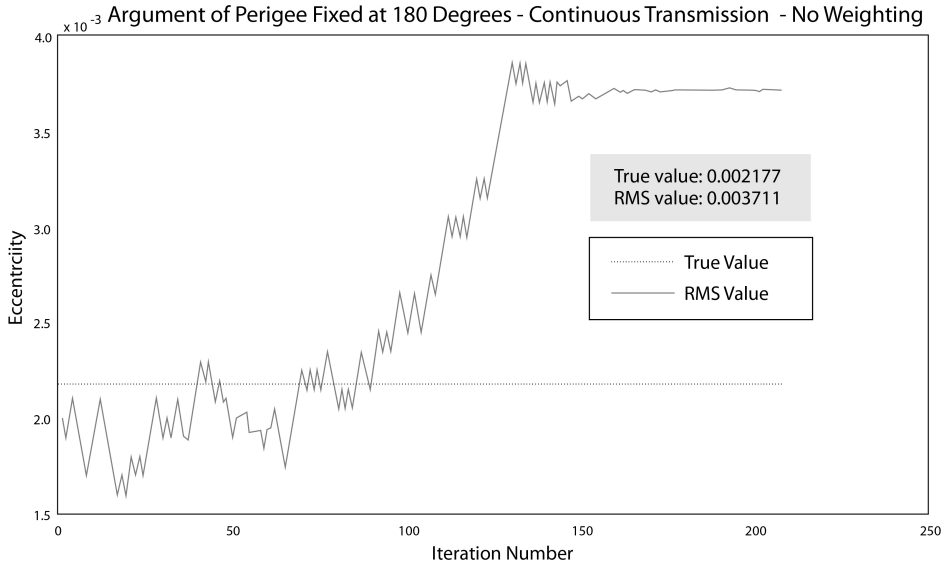

Fig. 4.8 Single cost function results for the eccentricity at Feb 1, 2016 00:30:00 UTC

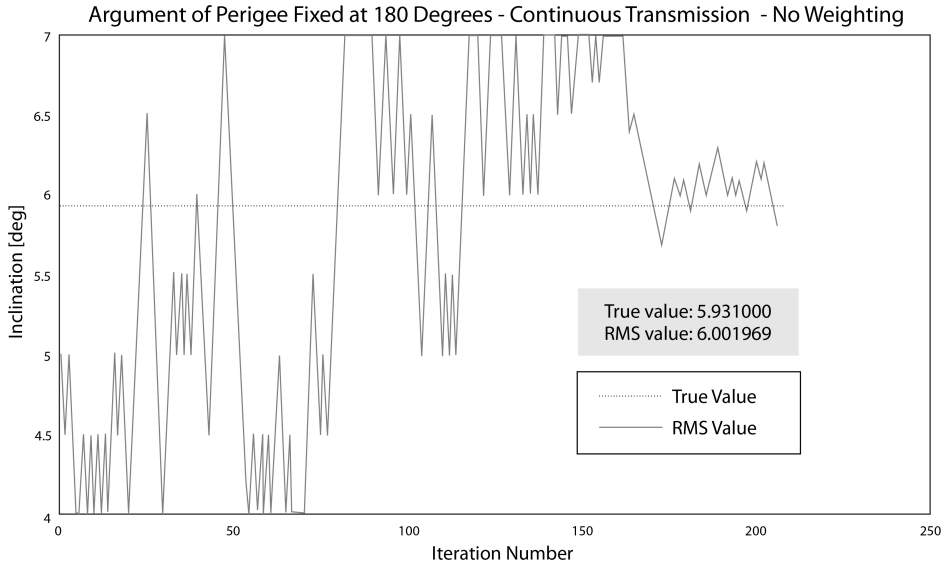

Fig. 4.9 Single cost function results for the inclination at Feb 1, 2016 00:30:00 UTC

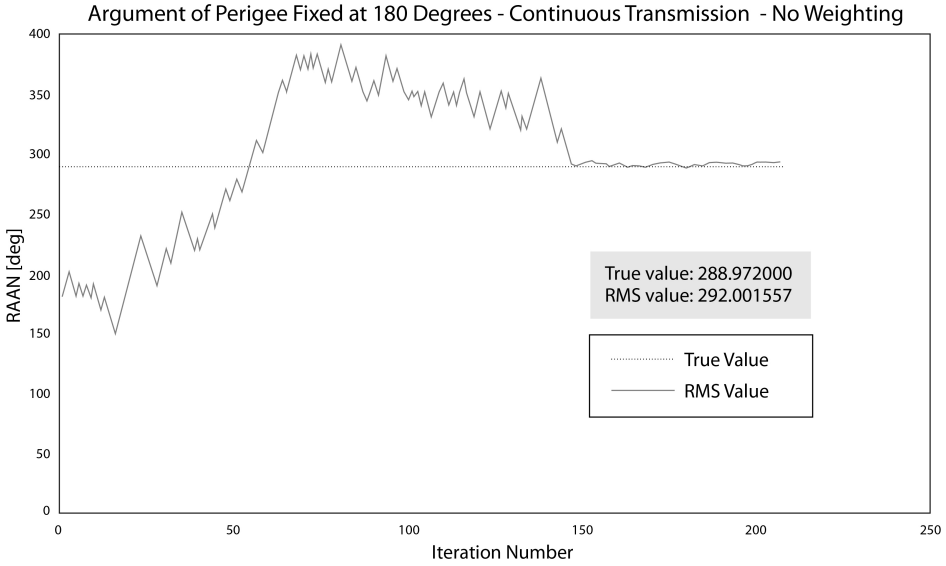

Fig. 4.10 Single cost function results for the RAAN at Feb 1, 2016 00:30:00 UTC

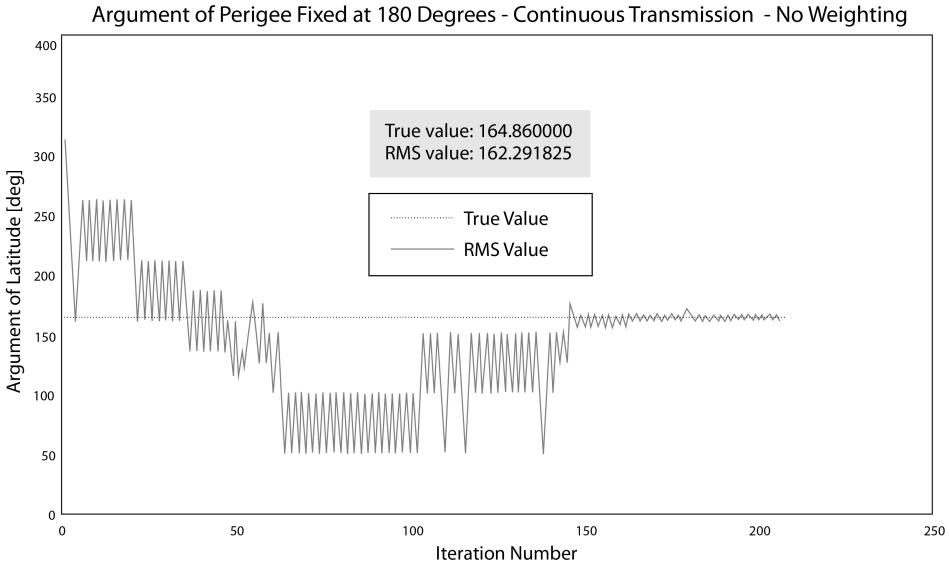

Fig. 4.11 Single cost function results for the argument of latitude at Feb 1, 2016 00:30:00 UTC

All orbital elements converged closely to their expected values after approximately 200 iterations. Based on these results, the ground station misshaping would be less than 5 degrees.

In order to confirm that the algorithm can indeed also be used for a different date of interest, it was run for March 29, 2016 00:00:00 UTC (see Fig.4.12-4.16). No changes were applied to the algorithm, other than the input file with the new "true" visibility times from STK.

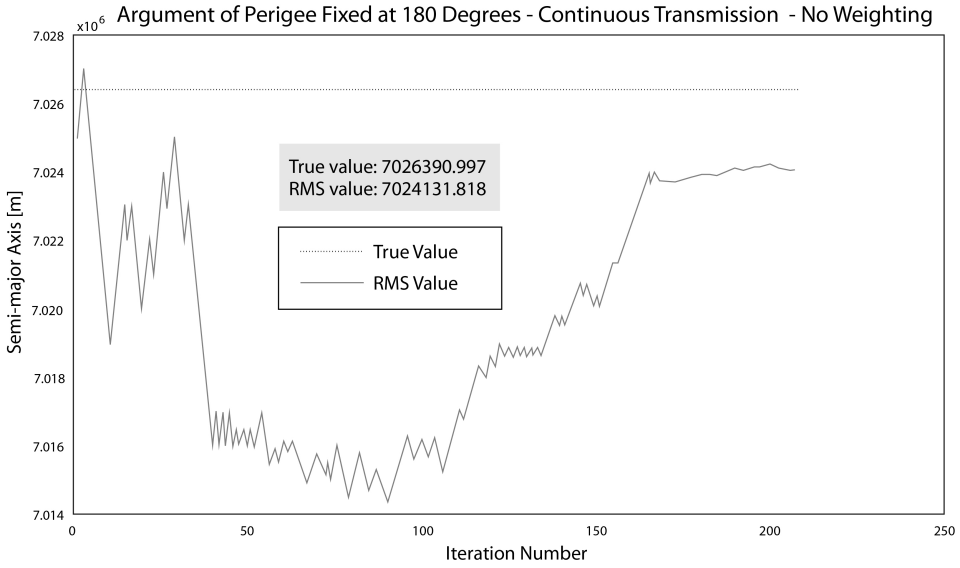

Fig. 4.12 Single cost function results for the semi-major axis at Mar 29, 2016 00:00:00 UTC

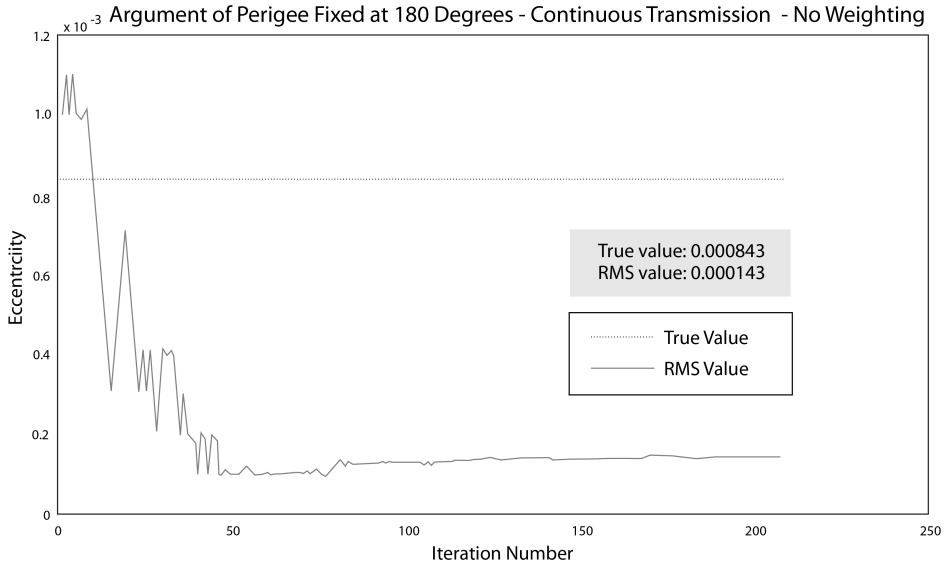

Fig. 4.13 Single cost function results for the eccentricity at Mar 29, 2016 00:00:00 UTC

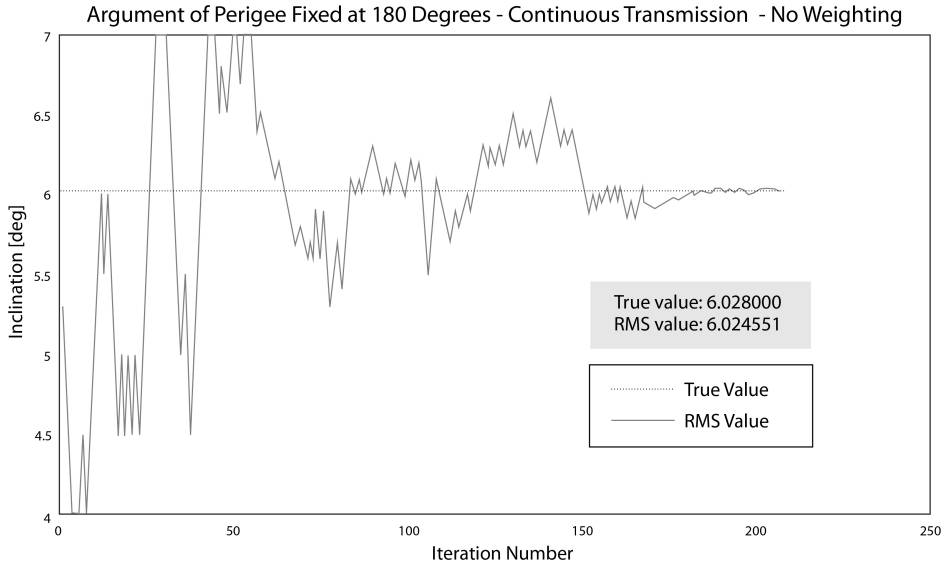

Fig. 4.14 Single cost function results for the inclination at Mar 29, 2016 00:00:00 UTC

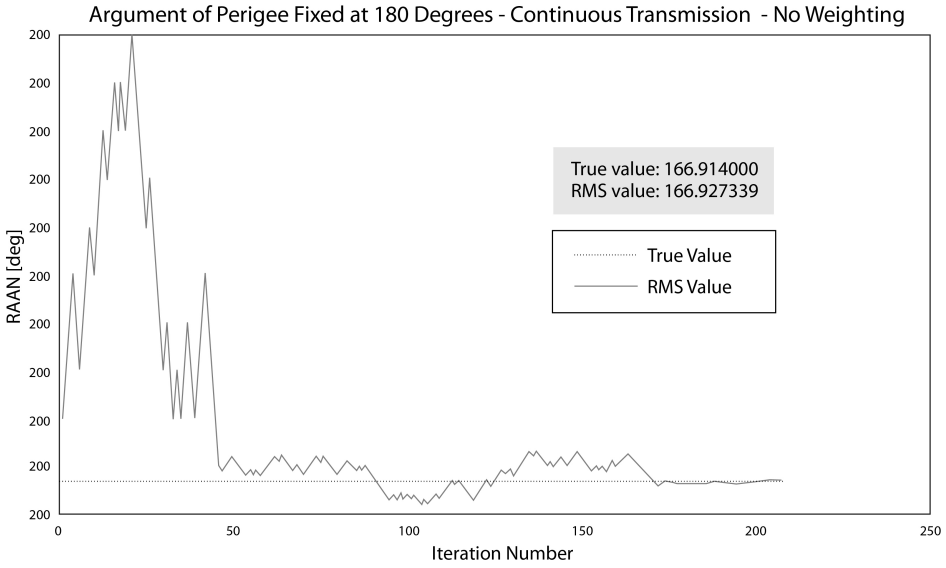

Fig. 4.15 Single cost function results for the RAAN at Mar 29, 2016 00:00:00 UTC

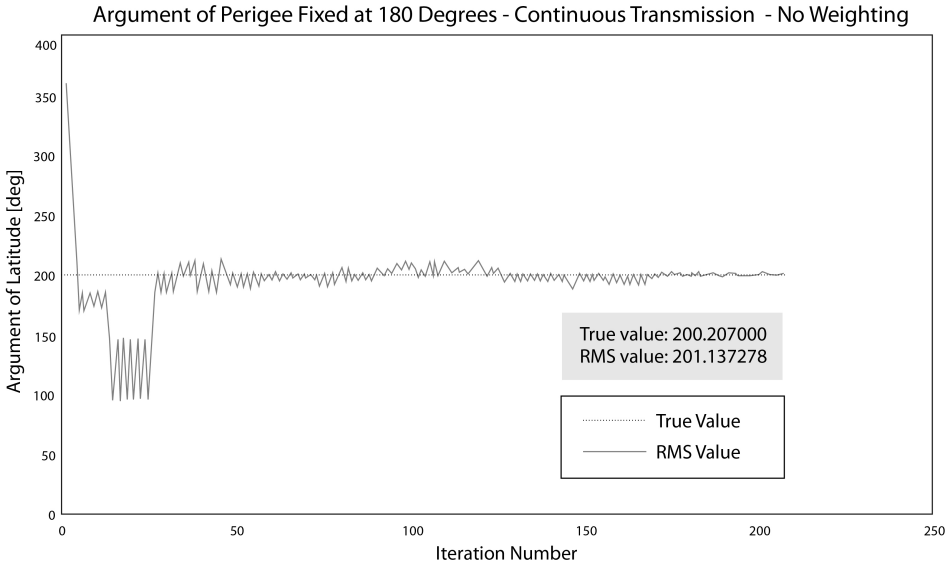

Fig. 4.16 Single cost function results for the argument of latitude at Mar 29, 2016 00:00:00 UTC

Once again, all orbital elements converged very closely to their expected values after 200 iterations. The optimization was also run for July 1, 2016 00:00:00 UTC and December 1, 2016 00:00:00 UTC with similar results.

4.1.1 Varying Transmission Intervals

Changing the rate at which each of the ships transmits its signal, changes when they can be observed by the satellite passing by from above. While a continuously transmitting ship will be seen from the moment the satellite is within range and right up until the satellite passes out of view, this is not true for variably trans-

mitting ships. The higher the transmission interval, the higher the uncertainty of whether a ship was seen when it first came into view or whether it had been in view for some time before the signal was received. This can significantly impact the start and stop time of the visibility and the associated error metrics for both the SST error and the TVL error. The algorithm may detect a high error and believe the orbital elements must be off, while in fact the only issue may be a late reception of the signal. In order to test how the algorithm will handle these cases, it was run for March 29, 2016 00:00:00 UTC (the significance of this date is that we have real-world AIS data from exactEarth for testing in the future) without any changes (see Fig.4.17-4.21).

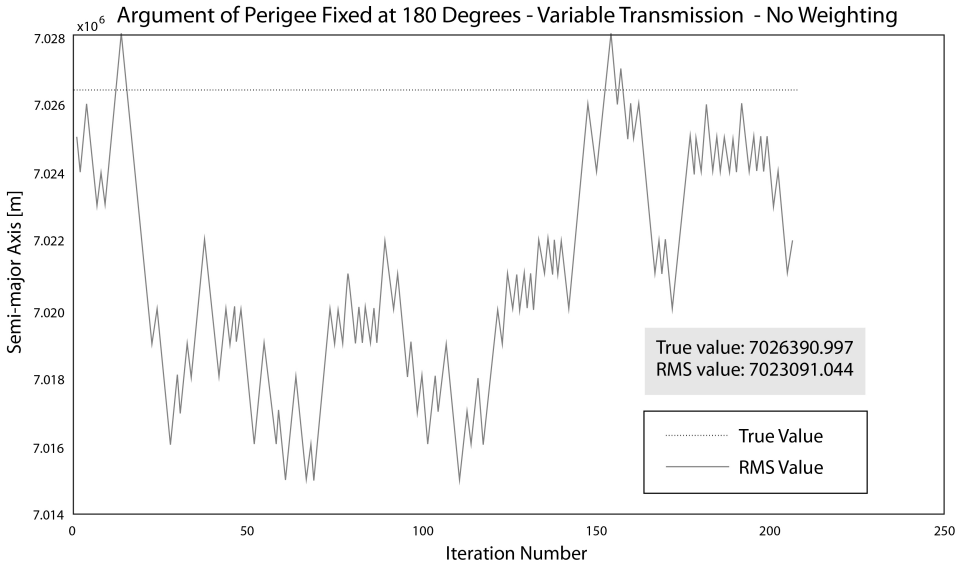

Fig. 4.17 Variable transmission interval results without weighting compensation for the semi-major axis at Mar 29, 2016 00:00:00 UTC

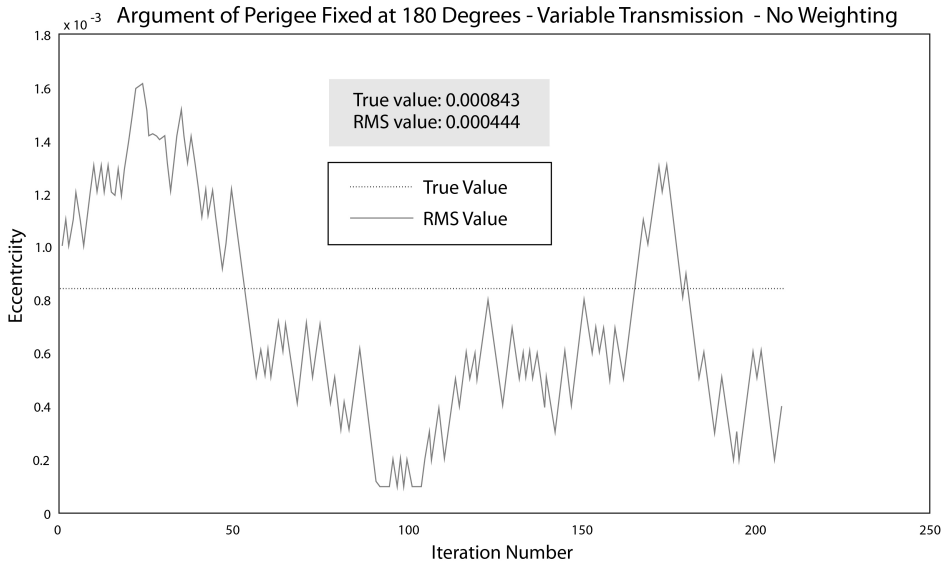

Fig. 4.18 Variable transmission interval results without weighting compensation for the eccentricity at Mar 29, 2016 00:00:00 UTC

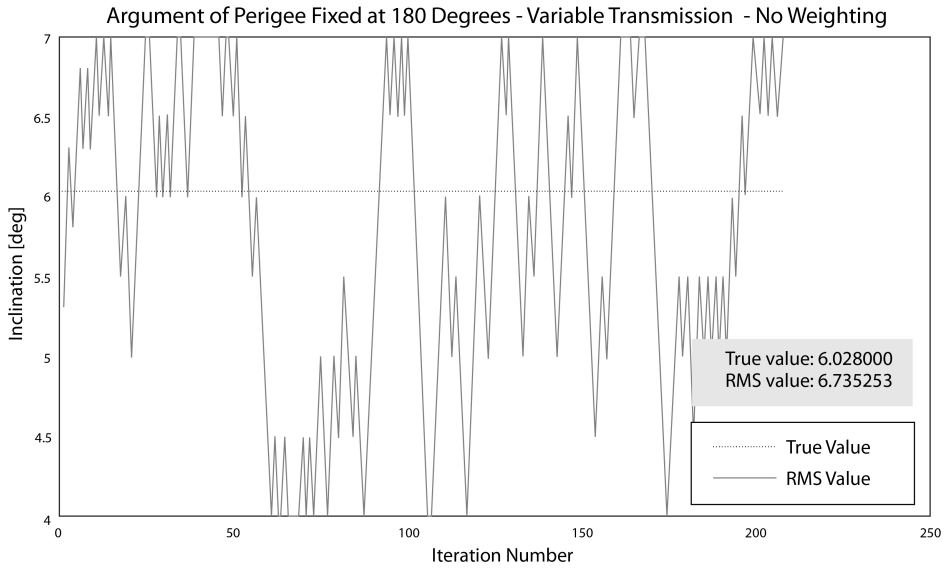

Fig. 4.19 Variable transmission interval results without weighting compensation for the inclination at Mar 29, 2016 00:00:00 UTC

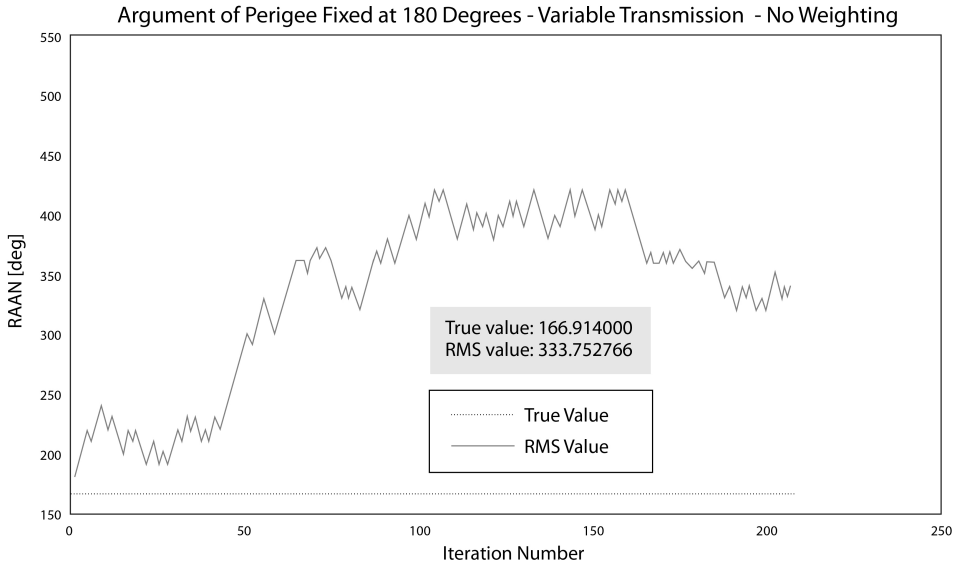

Fig. 4.20 Variable transmission interval results without weighting compensation for the RAAN at Mar 29, 2016 00:00:00 UTC

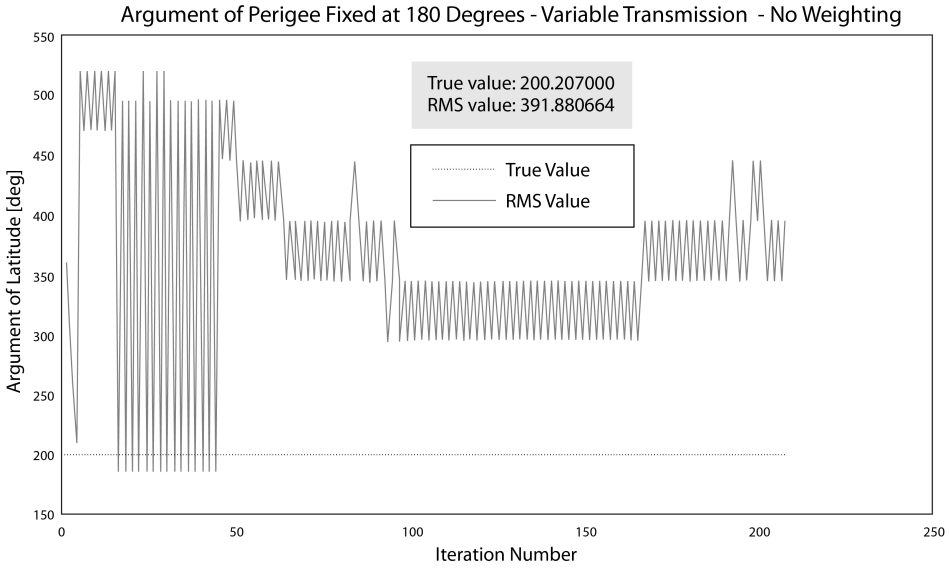

Fig. 4.21 Variable transmission interval results without weighting compensation for the argument of latitude at Mar 29, 2016 00:00:00 UTC

From these results it is clear that changing the transmission rate highly impacts the RAAN and argument of latitude, whereas the other orbital elements still converge relatively well. In order to compensate for the potential misinterpretations of the error metrics, a weighting function was applied to the SST error values based on a few characteristics. These characteristics determine whether we have high or low confidence in the error value (see Table 4.1).

Table 4.1 Confidence Levels Used to Develop the Weighting Function

Error Type	Confidence Level
negative start time error	low
positive start time error	high
negative stop time error	high
positive stop time error	low

For those instances with a negative start time error, it means the ship was seen earlier than it should in the estimated orbit compared to the "true" orbit. This could be because the satellite was in fact offset from its true orbit and therefore received the signal from the ship at a different time; it could be that the ship itself had a high transmission interval and therefore did not transmit a signal until the satellite had already been in view for some time; or it could be that the ship had a high transmission interval but transmitted its signal just as the satellite came into view and we are, once again, dealing with an orbit offset. Hence, we assign this error metric a low confidence rating. A similar rationale is valid for those ships with a positive stop time error. Each error value with a low confidence rating was weighted using an exponential weighting function, such that the weighting gets lighter as the error approaches zero. The optimization was run, with the weighting function compensation for March 29, 2016 00:00:00 UTC (see Fig.4.22-4.26).

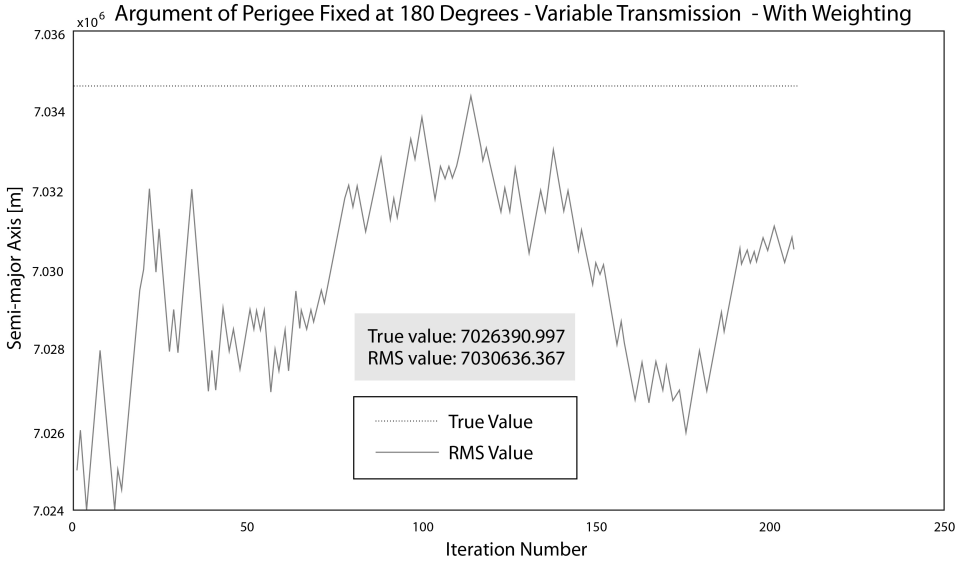

Fig. 4.22 Variable transmission interval results with weighting compensation for the semi-major axis at Mar 29, 2016 00:00:00 UTC

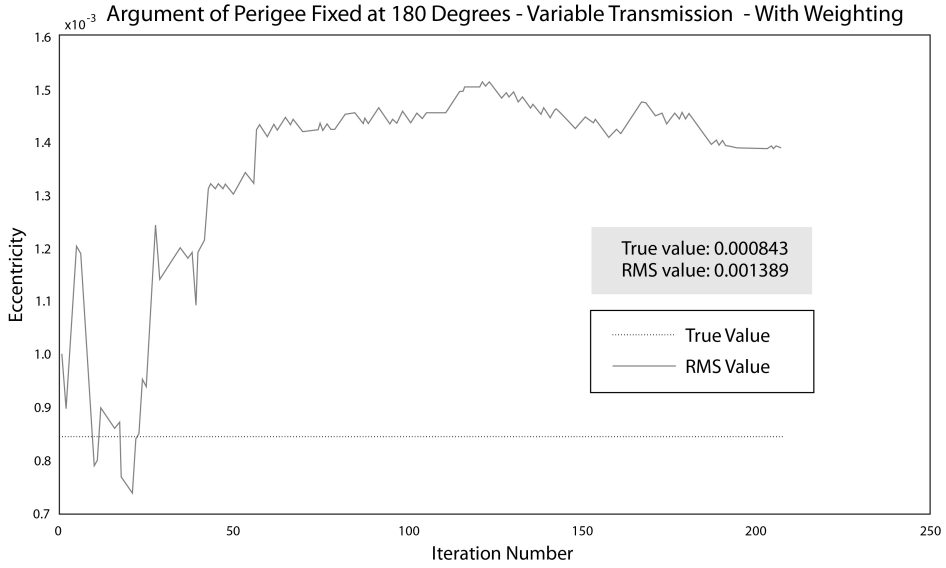

Fig. 4.23 Variable transmission interval results with weighting compensation for the eccentricity at Mar 29, 2016 00:00:00 UTC

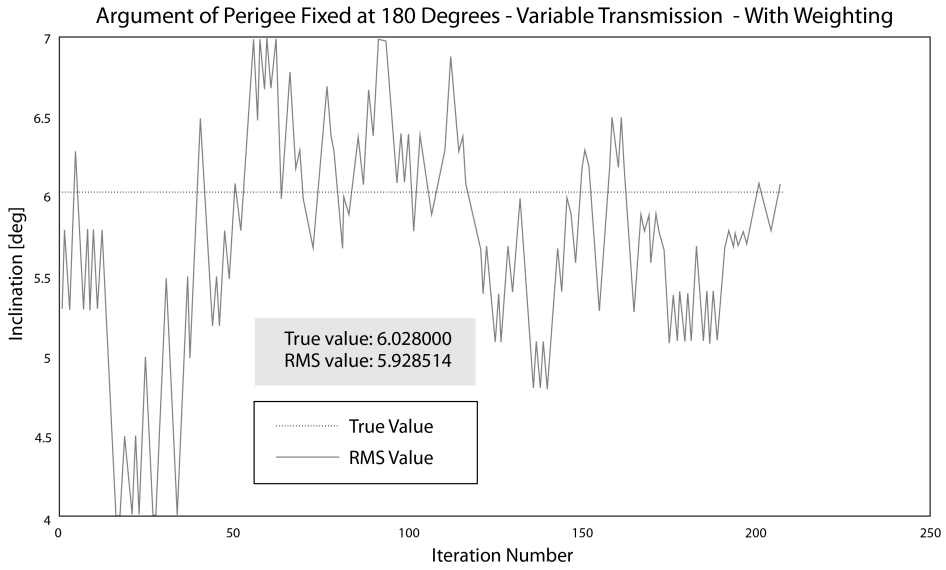

Fig. 4.24 Variable transmission interval results with weighting compensation for the inclination at Mar 29, 2016 00:00:00 UTC

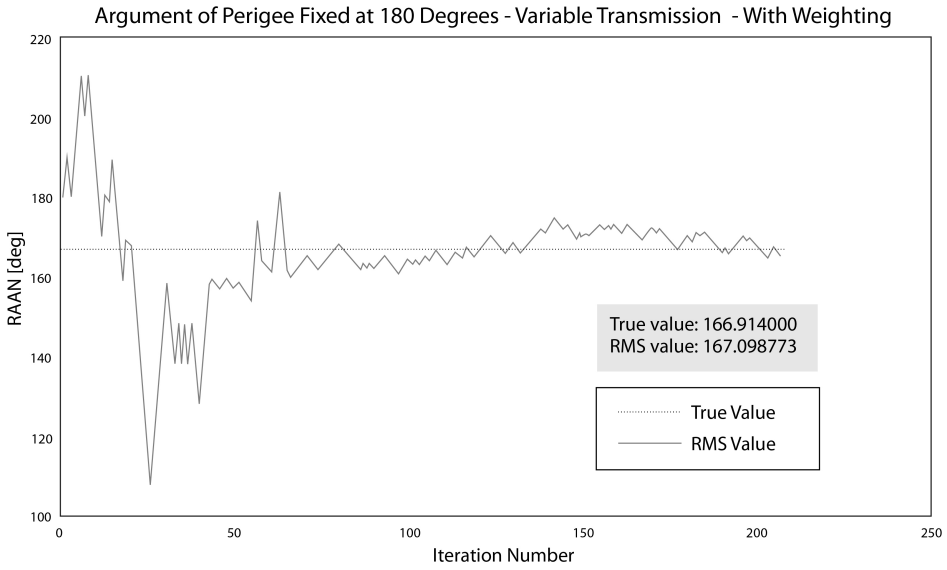

Fig. 4.25 Variable transmission interval results with weighting compensation for the RAAN at Mar 29, 2016 00:00:00 UTC

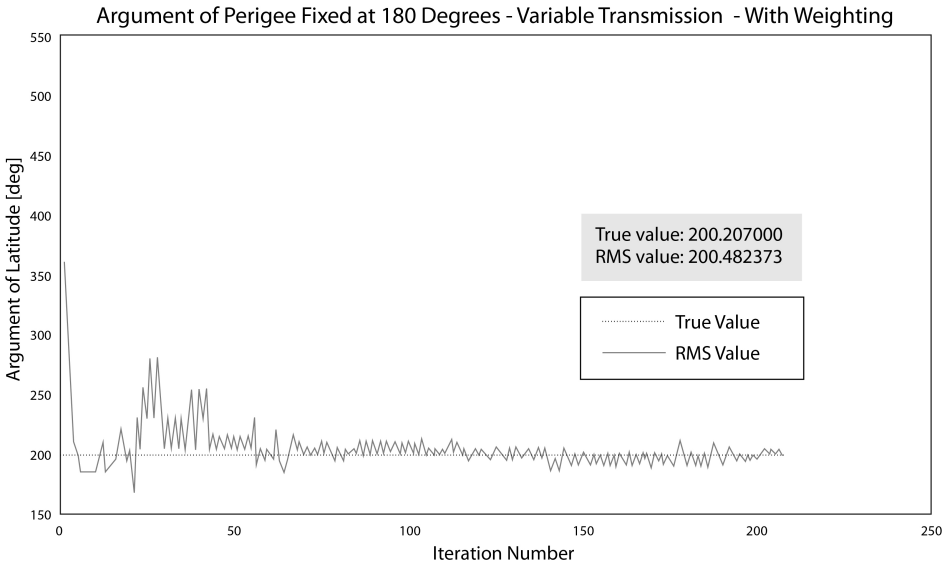

Fig. 4.26 Variable transmission interval results with weighting compensation for the argument of latitude at Mar 29, 2016 00:00:00 UTC

This effectively eliminates the misinterpretation of the error metrics due to the variable transmission interval and allows all orbital elements to converge very closely to their true value.

However, it was discovered that while these results are acceptable in terms of accuracy, they are not in terms of reliability. The parameters in this optimization, including the weighting function, only worked well when the date of interest was relatively close to Feb 1, 2016 and grew increasingly less reliable as this date of interest was set futher into the future.

4.2 Genetic Algorithm Optimization

4.2.1 Simulated Environment

The genetic algorithm was designed to overcome many of the constraints and limitations of the gradient descent-based optimization. In order to test the GA optimization more extensively, six test cases were designed:

1. Evenly-spaced grid of observation points (e.g. ships in the case of an AIS payload) with a constant transmission/visibility interval and zero altitude (E-CT-ZA in results plots)

2. Real point location distribution (based on real AIS payload data) with a constant transmission/visibility interval and zero altitude (R-CT-ZA in results plots)

3. Real point location distribution with variable transmission/visibility, no weighting function, and zero altitude (R-VTNW-ZA in results plots)

4. Real point location distribution with variable transmission/visibility, with weighting function, and zero altitude (R-VTWW-ZA in results plots)

5. Real point location distribution with variable transmission/visibility, no weighting function, and variable altitude. (R-VTNW-VA in results plots)

6. Real point location distribution with variable transmission/visibility, with weighting function, and variable altitude (R-VTWW-VA in results plots)

Each of these test cases was tested using a basic from-scratch GA without any of the aforementioned customizations on the parent selection (it only selected the 10 best TRMS parents), reproduction techniques (it only created one child per parent couple, with large random values for the mutation technique), cost function combinations (it only considered the TRMS error metric) and close-neighbour approximation (it did not perform any close-neighbour approximations).

The results in Fig. 4.27 - 4.31 show that the semi-major axis, eccentricity and inclination have relatively high accuracy levels for all six simulation cases. All three convergence errors satisfy TLE-level accuracy conditions. However, the RAAN and argument of latitude estimations degrade with increasing realism. Especially the varying altitude simulations fall significantly outside of the TLE-level accuracy conditions, which would require each of these two orbital to converge at less than 0.05 degree error. The results also show that, given that the amount of points observed in the real AIS data set is more than twice that of the evenly spaced grid, an increase in observation points yields a significant increase in estimation accuracy for the RAAN and argument of latitude. This is signified by the jump between the E-CT-ZA and R-CT-ZA curves in Fig. 4.30 and Fig. 4.31 below. All orbital elements, except for currently the semi-major axis, are also positively affected by

the use of the weighting function when considering variable transmission intervals. This is signified by the jump between the R-VTNW-ZA and R-VTWW-ZA curves, and the R-VTNW-VA and R-VTWW-VA curves.

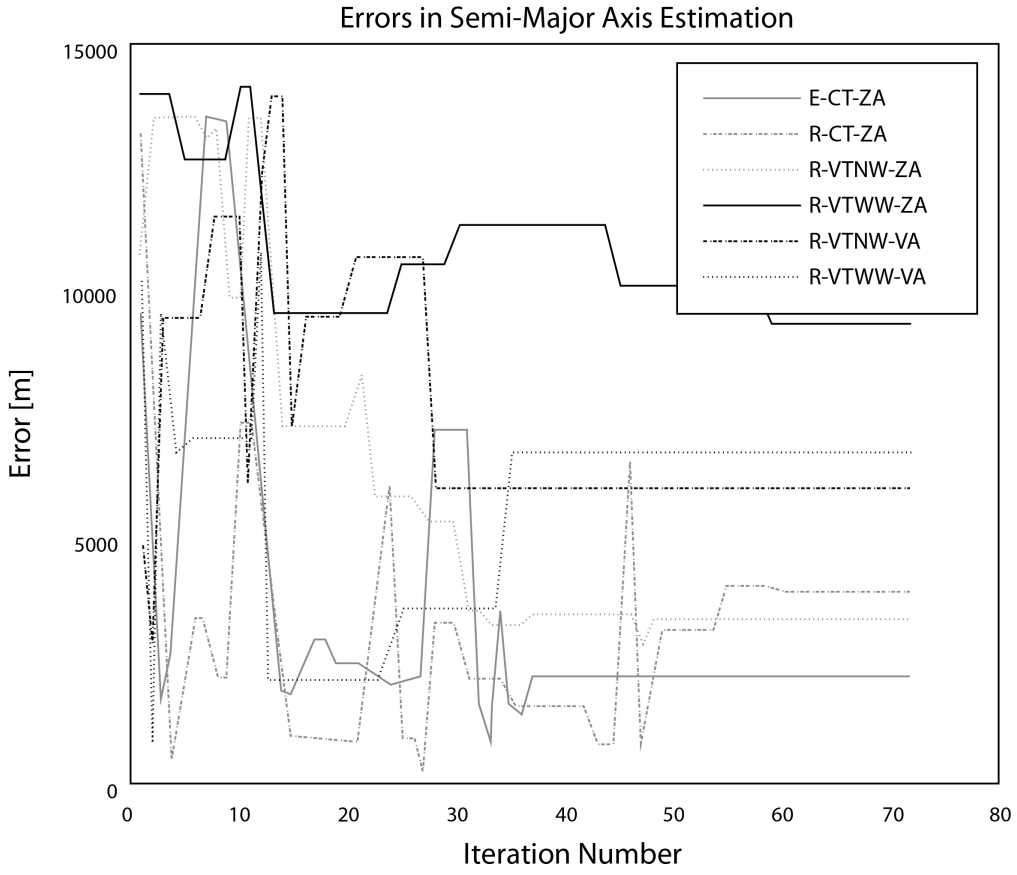

Fig. 4.27 Semi-major axis error results for the six cases

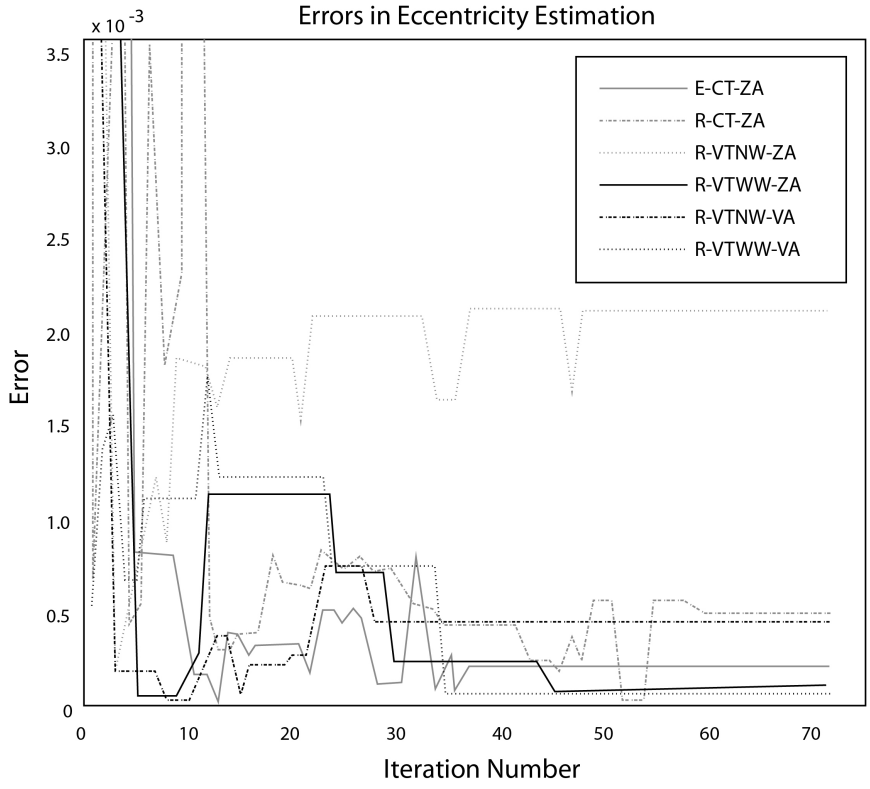

Fig. 4.28 Eccentricity error results for the six cases

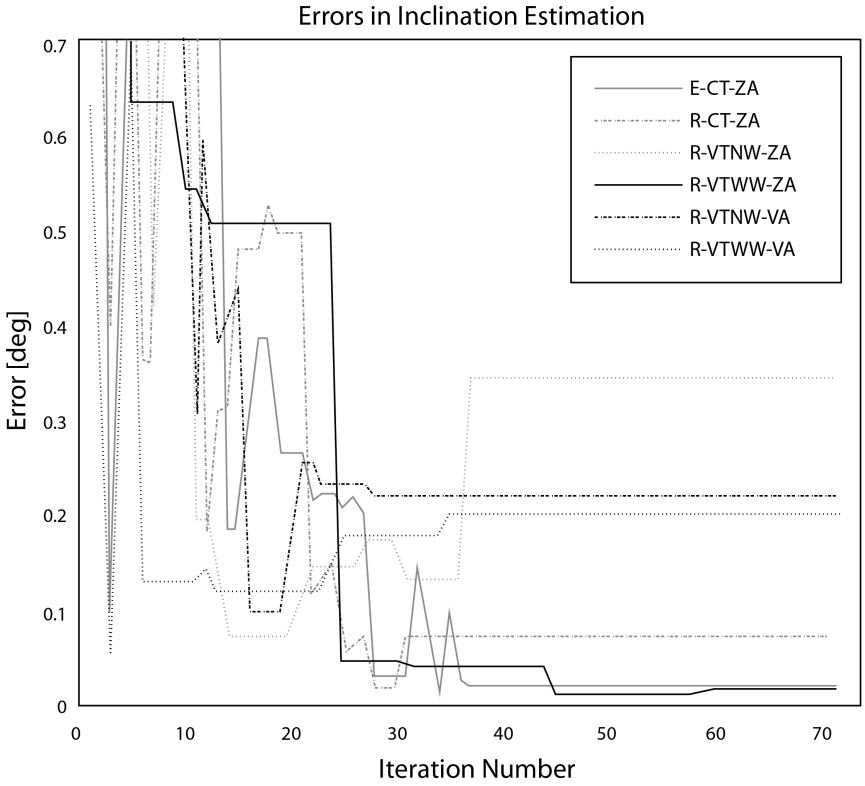

Fig. 4.29 Inclination error results for the six cases

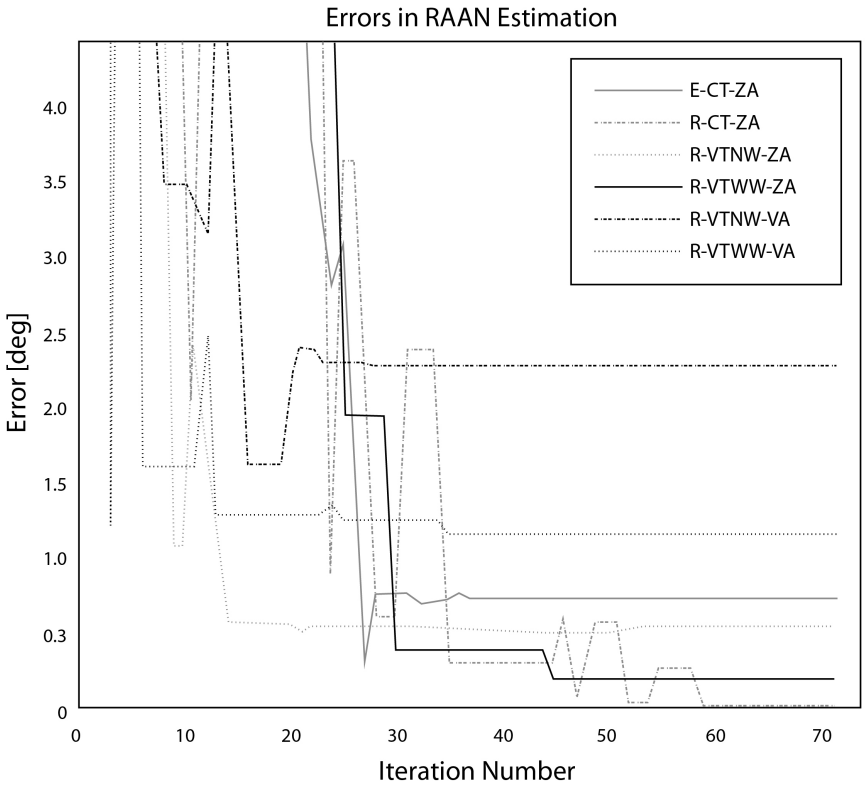

Fig. 4.30 RAAN error results for the six cases

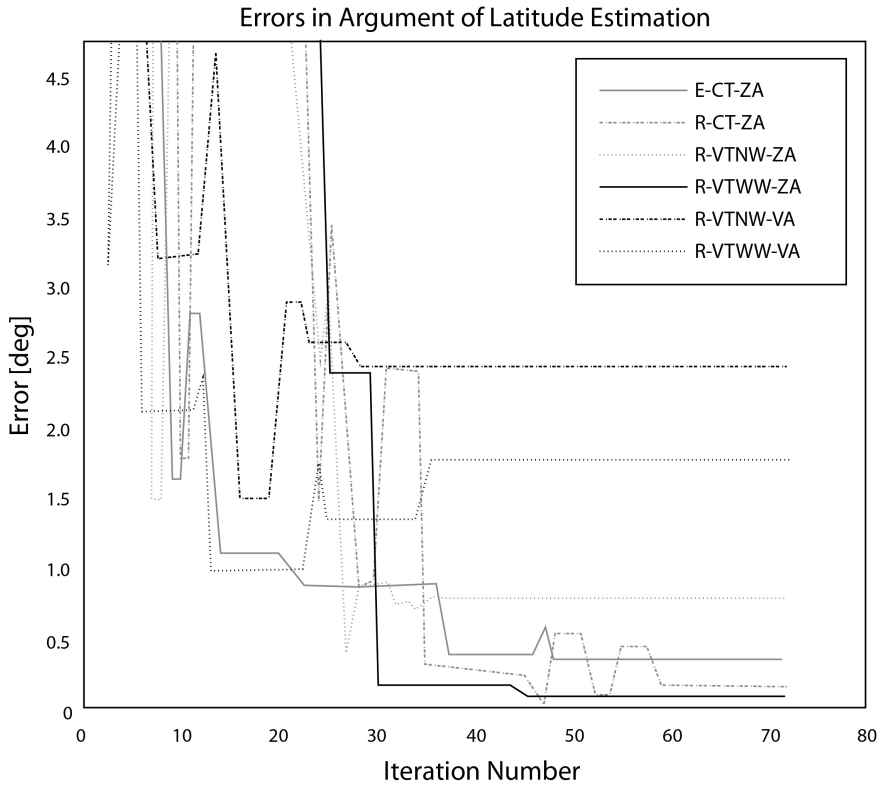

Fig. 4.31 Argument of latitude error results for the six cases

The GA is an ideal method for the orbit determination problem at hand as it caters directly to the inherent issues present in the search space, as described before. The results discussed prior in this section also prove it to be successful in terms of accurately estimating the orbital elements and maintaining reliability over different dates of interest and orbital conditions. In order to further increase accuracy and reliability, some further algorithmic changes were implemented on the GA algorithm:

1. Increasing the observation time frame from one orbit worth of data to three orbits worth of data, in order to make the effect of an offset in the RAAN and argument of latitude more pronounced and more easily detected by the algorithm.

2. Adding more variety to the parent pool in order to help avoid premature convergence.

3. Fine tuning the reproduction technique probabilities and their respective methods.

4. Creating a Genetic Algorithm - Simulated Annealing (GASA) hybrid algorithm that includes a process similar to Simulated Annealing at the end of each iteration to explore the close neighbourhoods around the parent individuals, especially as the algorithm nears convergence and only small steps are needed.

The algorithm was first tested on the same date of interest as in the other tests, March 29 2016 00:00:00.000 UTC. As can be seen in Fig. 4.32 - 4.36, the algorithm consistently converges to orbital element values close to the "true" orbit. It is important to note that the figures show the error between the estimated orbit and the "true" orbit and not the actual orbital element values. In order to verify consistency, 5 algorithm iterations were performed without making any changes to

the algorithm.

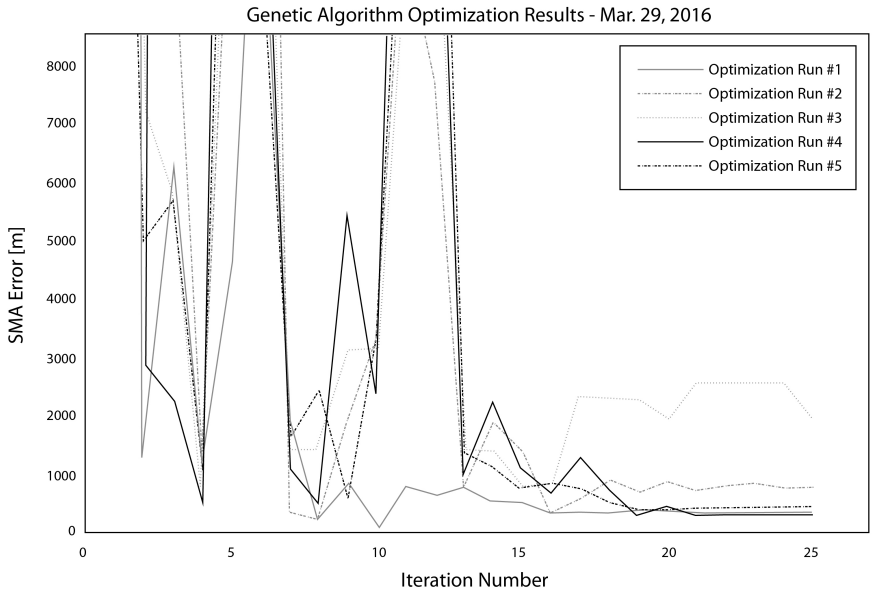

Fig. 4.32 Semi-major axis error results

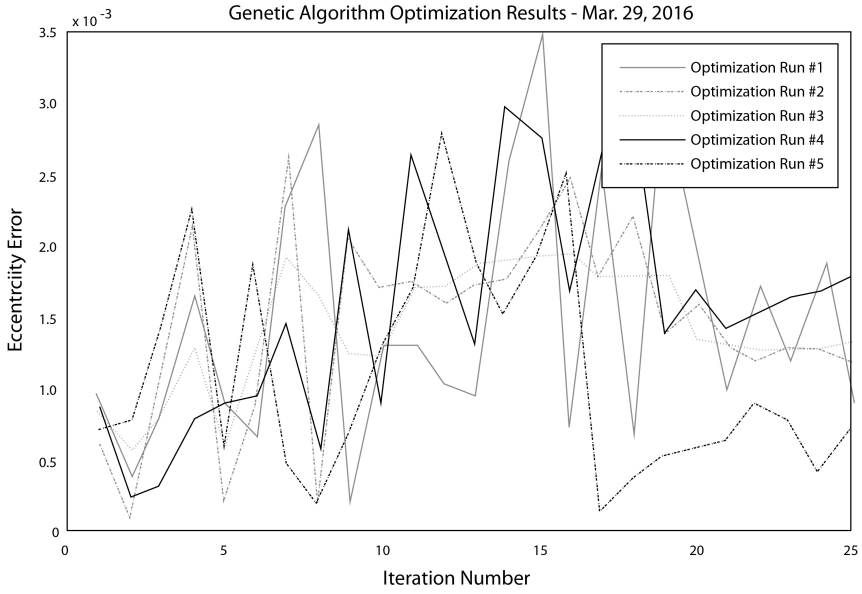

Fig. 4.33 Eccentricity error results

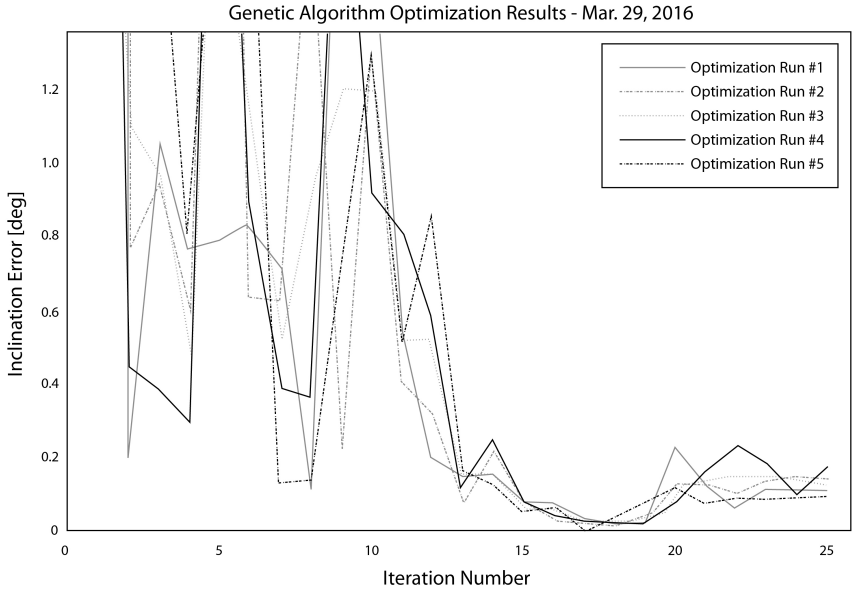

Fig. 4.34 Inclination error results

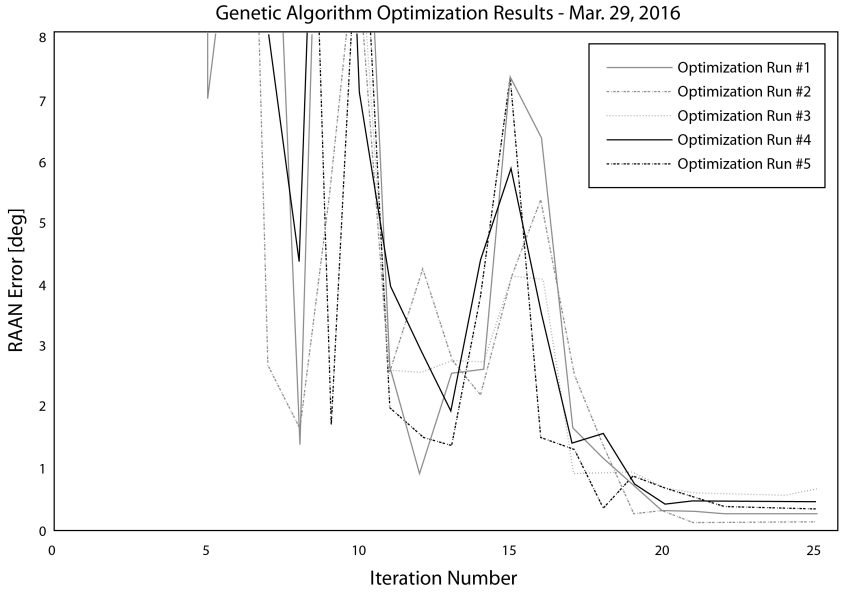

Fig. 4.35 RAAN error results

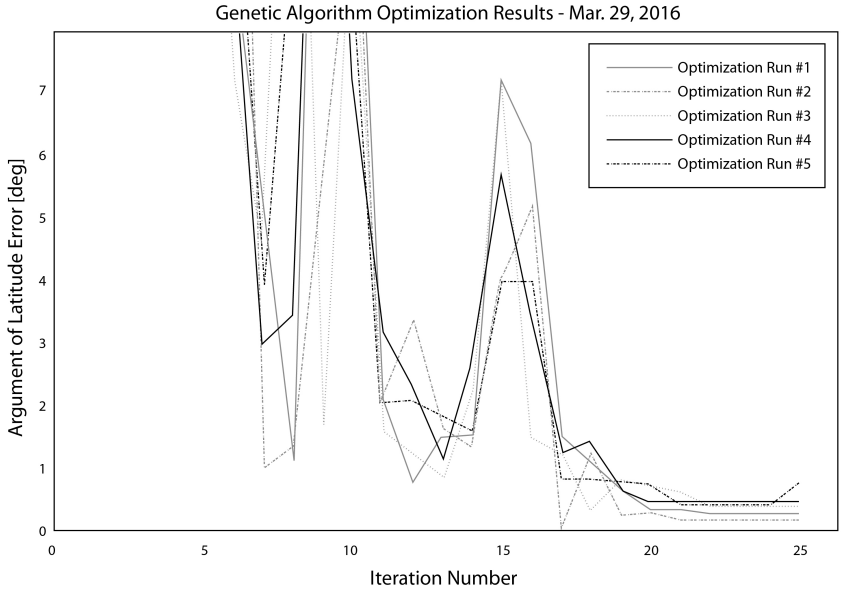

Fig. 4.36 Argument of latitude error results

Next, the algorithm was also tested for a date of interest exactly one year later on March 29 2017 00:00:00.000 UTC. This was an important test since the original gradient descent-based algorithm was not able to consistently converge to an accurate estimate at a date of interest this far from the initial date of interest. Once again, the algorithm was run 5 consecutive times without making any changes whatsoever. Fig. 4.37 - 4.41 show that the GA-based algorithm now has no problem dealing with a different date of interest, and thus a different "true" orbit and a different initial state.

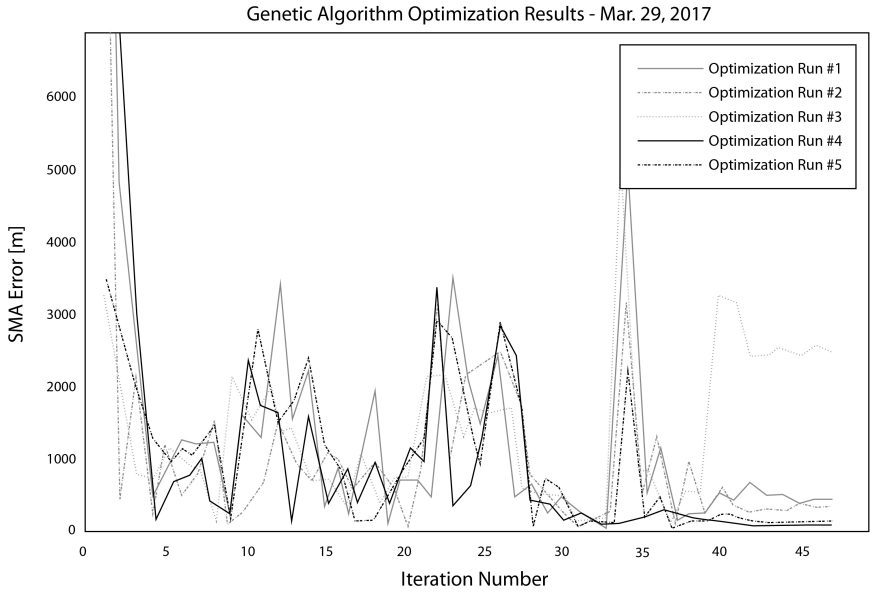

Fig. 4.37 Semi-major axis error results

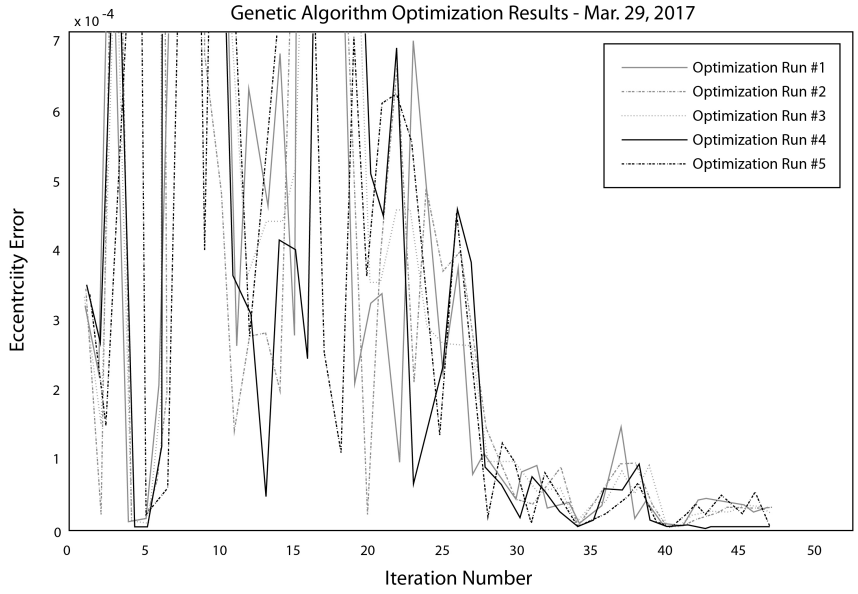

Fig. 4.38 Eccentricity error results

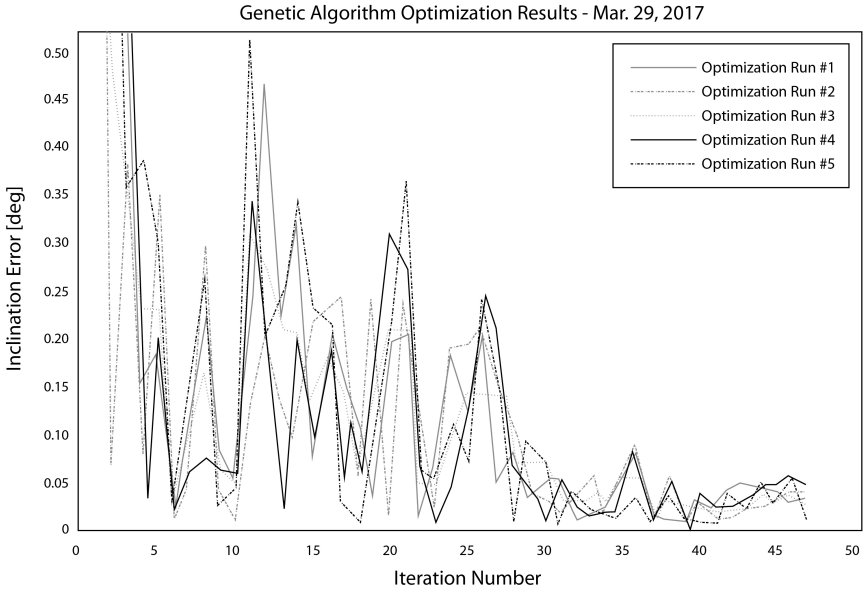

Fig. 4.39 Inclination error results

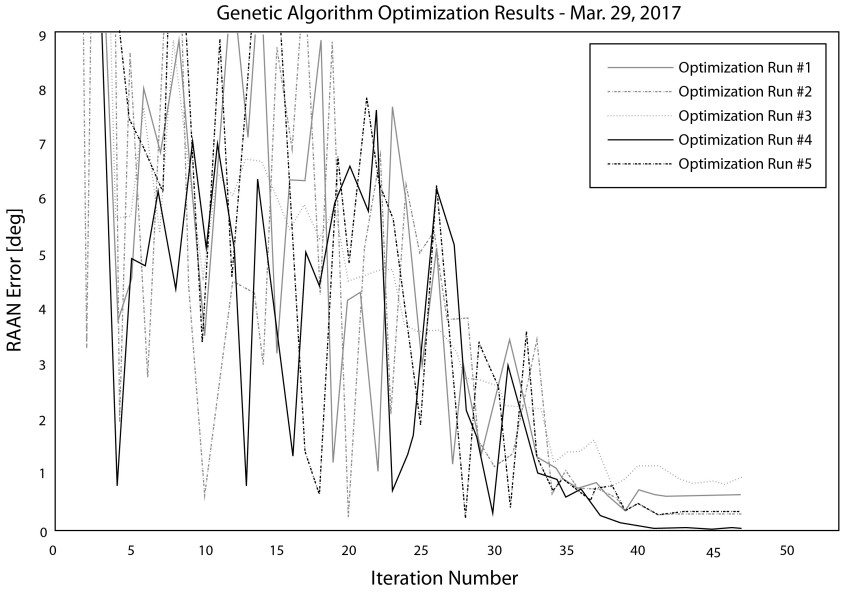

Fig. 4.40 RAAN error results

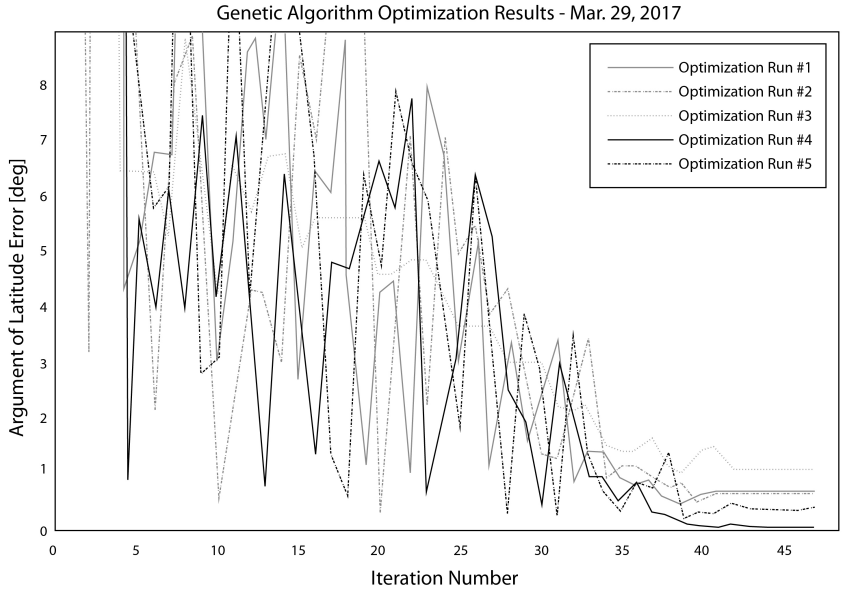

Fig. 4.41 Argument of latitude error results

4.3 Real-world AIS Data

Upon successful testing of the simulated AIS data from real-world ship positions, the algorithm was applied to real-world AIS data from the exactView 9 (EV9) satellite, provided by exactEarth Ltd.

4.3.1 Optimization Results

The real-world data included the time-of-access timestamp, ship ID, latitude and longitude of all messages received by EV9 from March 29, 2016 00:00:00.000 UTC until March 29, 2016 21:00:00.000 UTC. While accurate convergence was not expected without any changes to the algorithm tested on simulated data, it was nevertheless important to test the performance of the algorithm on this real-world data, analyze the behaviour and hypothesize potential improvements.

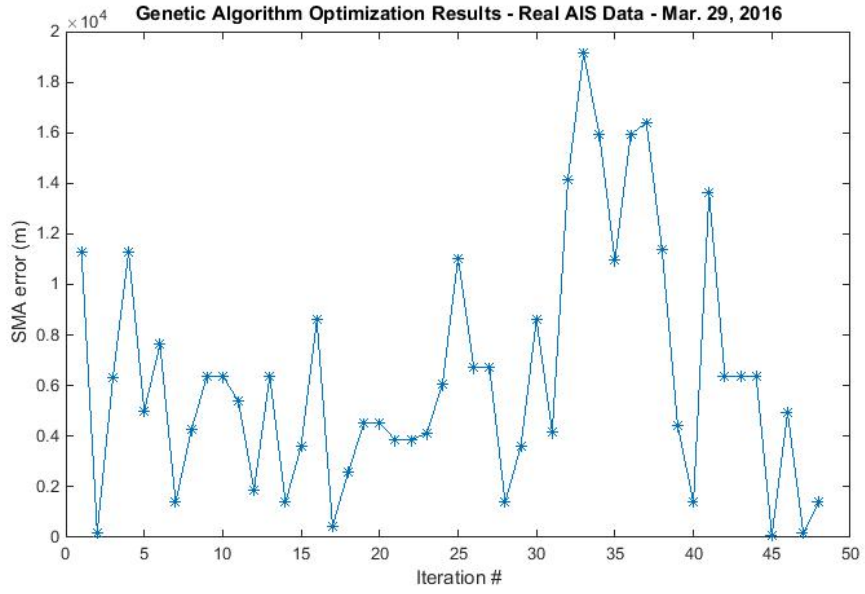

Fig. 4.42 Semi-major axis error results

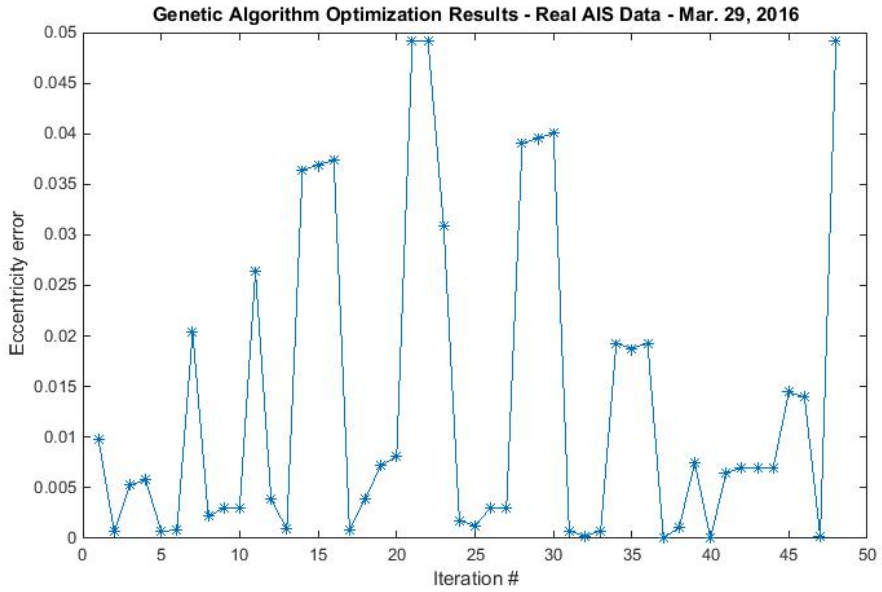

Fig. 4.43 Eccentricity error results

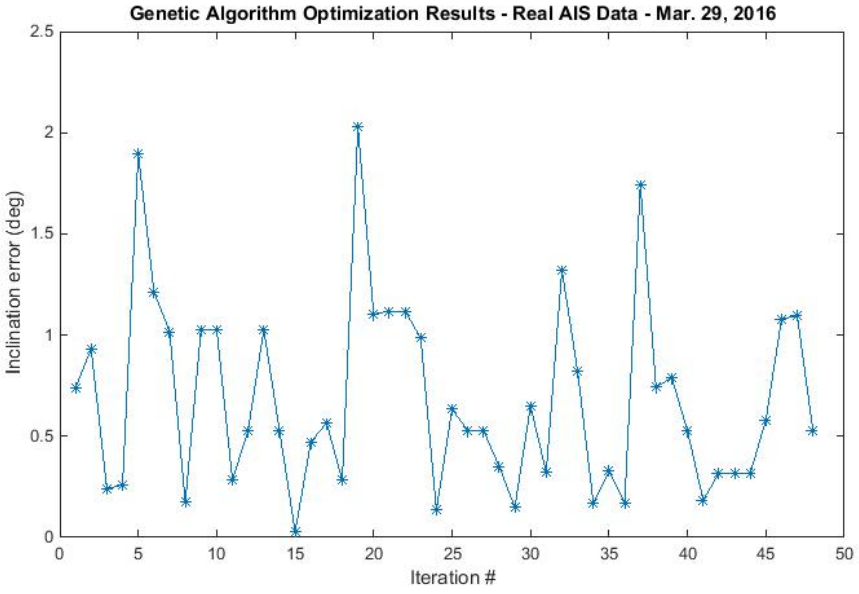

Fig. 4.44 Inclination error results

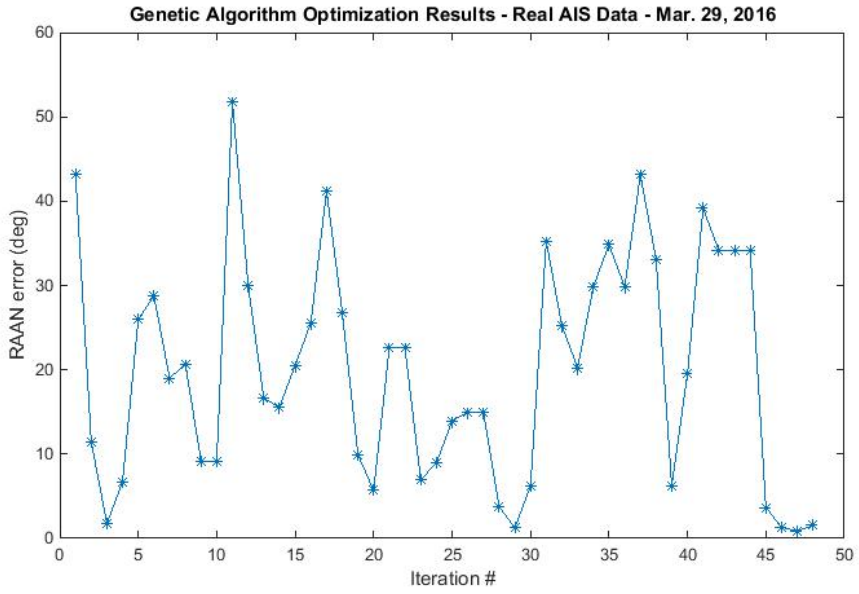

Fig. 4.45 RAAN error results

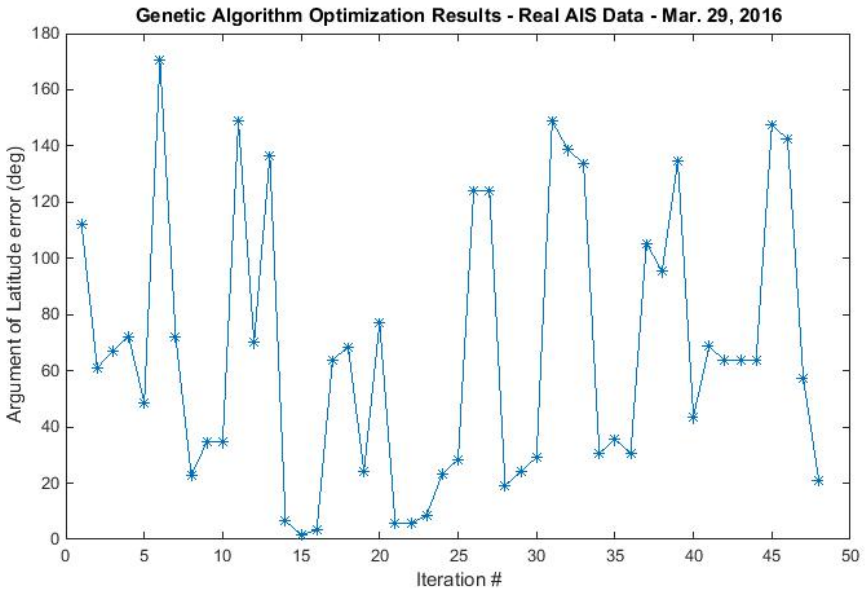

Fig. 4.46 Argument of latitude error results

As Fig. 4.42 - 4.46 show, the algorithm does not converge properly to any particular estimated orbital elements. While the semi-major axis, eccentricity and inclination estimations still stay within relatively error bounds, it is, once again, the RAAN and argument of latitude that prove to be the most unstable in their estimations. For these latter two orbital elements, the error remains at level of 10s of degrees for the RAAN to over 150 degrees for the argument of latitude. This points to a similar issue as experienced before in terms of those two orbital elements, but to a higher degree, not yet properly taken care of by the adjusted cost functions described earlier.

While the true cause of this inability to converge on real-world AIS data is still unknown, it is suspected that this is largely due to the cost function that is used, especially with regard to estimating the RAAN and the argument of latitude. The effect of the type of cost function on the error values was explored further.

4.3.2 Improving Real-world Convergence

While the results obtained by the optimization based on simulated data are acceptable for this study, especially given the novelty and consistency achieved with simulated data, some tests were done in order to determine how convergence could be improved in terms of the real-world AIS data. Since the first three orbital elements are slow-changing elements and can already be estimated with high accuracy, these tests focused on added sensitivity in terms of the RAAN and argument of latitude.

The errors of each of the two orbital elements were studied for a test starting with the true orbit. The first 40 tests increased the error in both the RAAN and argument of perigee as a function of the iteration number, and set a small constant error on the other elements. Tests 41 to 80 maintained that small constant error in the first three elements and decreased the error in the RAAN and argument of perigee, starting at the truth values. Finally, the last 40 tests looked at the effect of increasing the error in the RAAN and decreasing the error in the argument of

latitude.

Many combinations for the error metrics were explored, but only three were found to collectively represent the errors introduced on the orbital elements accurately. These were the TRMS error (which was used as a main cost function in the previous optimizations), the STSP difference metric and STSP error metric. The STSP difference metric takes the absolute value of the difference between the RMS of the start time error for all data points and the RMS of the stop time error for all data points. The STSP error metric, on the other hand, is quite different from all previous metrics used. It counts the sum of all instances where the start time error or stop time error were larger than 2 sigma.

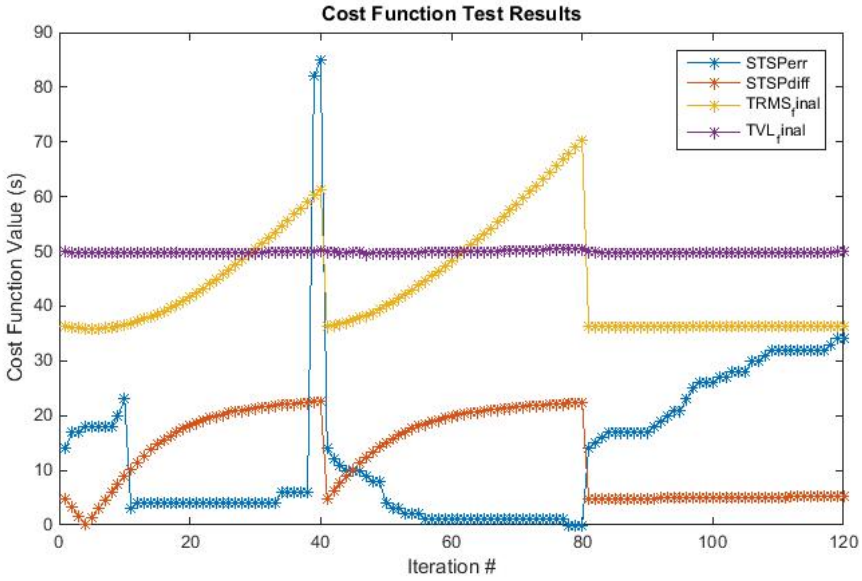

Fig. 4.47 Cost function test cases results

The results of these tests showed that the TRMS error does not accurately reflect an error in the RAAN and argument of latitude when their errors are opposite in direction. In a sense, these errors will cancel one another out and still produce a TRMS error equivalent to a very small error in these two orbital elements, even though they might in fact be very large but opposite in direction from each other. The hypothesis to be tested in future related work is if a cost function were to be used that incorporates features from both the TRMS error metric and the STSP error metric, then the RAAN and argument of latitude error will be represented much more effectively and increase the response of the GA to such errors.

5 Concluding Remarks

5.1 Summary of Results

While the methodology of this research covered a variety of techniques, from a purely geometric solution to a gradient descent-based optimization to a machine learning GA, it is the GA that produced the most accurate, consistent and reliable results with the simulated data. There are still a number of steps to take in order to make the algorithm perform with the intricacies of real-world AIS data, but the GA performed close to TLE-level with the simulated data, including modifications to make it as realistic as possible. The most realistic results from the GA have an accuracy of around 5-10km cross-track and radially and 25-35km along-track. Hence, referring back to Table 1.1, the solution developed here has no additional on-board power cost, it has no additional on-board dollar cost (it will require ground-based computing but this can be achieved on any ordinary computer), it has no additional mass cost, it can provide an orbit update daily, it provides an accuracy similar to TLE on the order of 10 kilometers and it requires no on-board computation. The

algorithm meets all of the aforementioned "ideal" needs for nanosatellite missions.

5.2 Usage and Applications

The usage of the algorithm depends on a variety of factors. First of all, the algorithm needs to be customized and improved further to make sure that it can perform accurately and reliably with real-world data. However, the data itself also needs to conform to the following conditions:

- Data points need to be available for most of the satellite's orbital path
- The satellite needs to be observing a large number of data points
- The satellite needs to have a wide field-of-view (out to the horizon)
- The data needs to include each point's position coordinates and this position needs to be known on the order of every few minutes.
- The data points need to be moving very slowly, relative to the satellite, such that there speeds can be considered negligible (though, with modifications in the future this could be expanded to faster moving objects, such as ADS-B on aircraft)
- There needs to be a regular downlink of data, since data processing will be done on the ground

- The operator needs to have access to a computer for data processing and running the algorithm
- The algorithm in its current state takes about 20-24 hours to run

In addition to being used as the primary orbit determination method, the algorithm also has a great potential in terms of orbit validation. This can be done in tandem with other orbit determination techniques to verify the current orbit estimate, as well as generate estimates in between updates from the primary OD method.

5.3 Future Work

The accomplishments and developments in this thesis provide a solid backbone for an orbit determination algorithm based on the time-of-access information of a large number of observed data points in real-world scenarios, AIS-based or otherwise. However, as shown, more research and development will be necessary in order to bring it to a level where it can operate on real-world data. The most obvious next step is to adjust the cost function(s) so that they can more accurately reflect errors in the RAAN and argument of latitude. However, a more in-depth study on the assumptions made by the current algorithm and where these fail would perhaps give a more complete overview of the types of changes that will need to be made in order to guarantee convergence, and moreover accurate convergence, with

real-world data.

Another up-and-coming area of development is the Internet of Things (IoT) and connecting such devices through satellite constellations. Many sensors in today's consumer and industrial environments are already connected to the internet but it is widely believed by companies in the industry that satellite constellations are the next step in making such IoT systems more scalable and accessible. Adapting the algorithm discussed here to be compatible with such systems and provide near-real-time feedback in a low-power environment will require changes to the efficiency and computing requirements of the GA. Furthermore, the algorithm could also combine knowledge of the data from multiple satellites to increase accuracy and convergence speed.

5.4 Conclusion

The algorithm developed in this research provides a novel way of preparing the space industry for the future. The rapid rise in nanosatellite deployment and miniaturization of communication technology will require a cheaper, leaner and more efficient way of tracking those satellites. Using the already-collected timing data from the payload observations means that no additional on-board equipment or processing will be required and that it could even be applied to existing missions, as well. This thesis provides a solid foundation and development analysis to

support this new way of using satellite payload data. It has shown how combining even the most basic form of observed data (the time-of-access and location) can provide deeper and more insightful knowledge. Each cost function used and explored combines this data in a different manner and, therefore, provides a different kind of insight pertaining to a different aspect of the satellite behaviour. This, combined with the power of machine learning, has proven to be an effective way of determining the position and velocity of the satellite with strong potential for future development in real-world Earth-observation or, perhaps even, interplanetary missions.

Bibliography

[1] G. Gronchi A. Milani. Theory of orbit determination. *Theory of Orbit Determination*, 2009.

[2] Neil Arundale. Ais reporting interva. *Web*.

[3] G. Born B. Tapley, B. Schutz. Statistical orbit determination. *Statistical Orbit Determination*, 2004.

[4] Heather Ball. *Satellite AIS for Dummies*. John Wiley & Sons Canada, Ltd., 2013.

[5] Nicholas Bijnens. Gps-based satellite orbit determination using ekf. *York University, ESS5410 - Advanced Satellite Positioning*, 2016.

[6] W. Boyce. Examination of norad tle accuracy using the iridium constellation. *Advances in the Astronautical Sciences*, 2005.

[7] W. Marshall C. Levit. Improved orbit predictions using two-line elements. *Advances in Space Research*, 2011.

[8] General Dynamics Canada. Spaceborne gps receivers. Web.

[9] J. Carson-Jackson. Satellite ais - developing technology or existing capability? *Journal of Navigation*, 2012.

[10] H. Curtis. Preliminary orbit determination. *Orbital Mechanics for Engineering Students*, 2010.

[11] P. Bohn D. Hobbs. Precise orbit determination for low earth orbit satellites. *Annals of the Marie Curie Fellowship Association*, 2006.

[12] P. Crawford D. Vallado. Sgp4 orbit determination. *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2008.

[13] David Bull David Beasley and Ralph Martin. An overview of genetic algorithms: Part 1 fundamentals. *University of Cardiff*, 1993.

[14] O. Montenbruck E. Gill. Comparison of gps-based orbit determination strategies. *European Space Agency, (Special Publication)*, 2004.

[15] D. Finkleman. "tle or not tle?" that is the question. *Advances in the Astronautical Sciences*, 2007.

[16] U.S. Coast Guard. How ais works. *U.S. Coast Guard Navigation Centre*.

[17] T. Hanada H. Hinagawa, H. Yamaoka. Orbit determination by genetic algorithm and application to geo observation. *Advances in Space Research*, 2014.

[18] C. Ryoo J. Hong, W. Park. Tle data-based precise estimation of satellite's orbital parameters. *Proceeding of the 16th International Conference on Control, Automation and Systems*, 2016.

[19] Kim S et.al. J. Hong, Kim J. Tle data based precise estimation of satellite's orbital parameters. *International Conference on Control, Automation and Systems*, 2017.

[20] D. Mortari Karimi R. Initial orbit determination using multiple observations. *Celestial Mechanics and Dynamical Astronomy*, 2011.

[21] Elizabeth Keil. Kalman filter implementation to determine orbit and attitude of a satellite in a molniya orbit. *Virginia Polytechnic Institute and State University*, 2014.

[22] Govindarajan Kothandaraman and Mario Rotea. Spsa algorithm for parachute parameter estimation. *17th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, 2003.

[23] Otto Koudelka. Nanosatellites for technological and science missions. *Institute of Communication Networks and Satellite Communications*, 2017.

[24] F. Curti L. Ansalone. A genetic algorithm for initial orbit determination from a too short arc optical observation. *Advances in Space Research*, 2013.

[25] International Maritime Organization. *SOLAS - International Convention for the Safety of Life at Sea*. Lloyd's Register, 2005.

[26] Q. Zhang R. Wang, J. Liu. Propagation errors analysis of tle data. *Advances in Space Research*, 2009.

[27] K. Riesing. Orbit determination from two line element sets of iss-deployed cubesats. *29th Annual AIAA/USU Conference on Small Satellites*, 2015.

[28] Kathleen Riesing. Two line element sets of cubesats in leo: Accuracy assessment and estimation techniques for improvement. *29th Annual AIAA/USU Conference on Small Satellites*, 2015.

[29] Franz Newland Ruben Yousuf and Thia Kirubarajan. Satellite orbit determination using ground-based navigation data. *European Navigation Conference*, 2011.

[30] S. Gangal S. Aghav. Simplified orbit determination algorithm for low earth orbit satellites using spaceborne gps navigation sensor. *Artificial Satellites*, 2014.

[31] Payman Sadegh. Constrained optimization via stochastic approximation with a simultaneous perturbation gradient approximation. *Automatica*, pages 889–892, 1998.

[32] James C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, pages 817–823, 1998.

[33] Jerome R. Vetter. Fifty years of orbit determination: Development of modern astrodynamics methods. *Johns Hopkins APL Technical Digest*, 27(3):239–252, 2007.

[34] I.-Jeng Wang and James C. Spall. Stochastic optimisation with inequality constraints using simultaneous perturbations and penalty functions. *International Journal of Control*, pages 1232–1238, 2008.

[35] D. Whitley. A genetic algorithm tutorial. *Statistics and Computing*, 1994.

[36] J. Wright. Optimal orbit determination. *Advances in the Astronautical Sciences*, 2002.