Gravity Gradient Tensor Algorithm for Autonomous Satellite Navigation

José Ramón Millán Fernández

Delft University of Technology

TU Delft

Gravity Gradient Tensor Algorithm for Autonomous Satellite Navigation

by

José Ramón Millán Fernández

Student Number: 4818334
Supervisor: Joao De Teixeira da Encarnacao

TU Delft

Preface

This thesis marks the end of my master studies in Spaceflight. It has taken a lot of effort to finish it, and there have been struggles. But I would like to thank everyone who has helped me in this endeavour.

I would like to express my gratitude to TU Delft for providing me with all the resources I have needed in these years and especially in this thesis. I want to thank my lecturers for providing me with the knowledge I have needed and that will prove useful in the future. I want thank my thesis supervisor Joao De Teixeira da Encarnacao, for being patient and always incredibly helpful through this process, and I acknowledge Prof. Byron Tapley of the Center for Space Research of the University of Texas at Austin for the original idea for the topic of this thesis.

An immense thanks to everyone who has supported me. To my friends and loved ones, who no matter what, have always been there for me and kept me motivated even in the roughest of times. And lastly, to my family, specially my parents, who have been an inspiration to me, and a pillar in both this thesis and in my life. Thank you so much!

José Ramón Millán Fernández

Abstract

Satellite navigation is a system that makes use of satellites to provide geo-positioning. Global Navigation Satellite System (GNSS) is a global system, not only for ground-based users but also for satellites determining their own trajectories in Earth orbit. While GNSS determines position by making use of other satellites, gravity gradiometry is a technology that can potentially determine position with only on-board data. Gravity gradiometry is commonly used in missions such as ESA's GOCE to obtain a map of Earth's gravity field. By using a gravity field model, orbital position can be obtained.

This thesis investigates how gravity gradiometry and gravity field models can be used for autonomous satellite navigation. In an autonomous environment, the satellite can navigate independently from any data which is not its own. Specifically, it evaluates the accuracy of a least squares orbit determination algorithm. It calculates a gravity gradient vector in the geocentric spherical frame with the use of factorisation. It also includes the estimation of the scale coefficient and bias as calibration parameters, and includes white noise error. There are two versions of the algorithm. The corrected version includes the estimation of calibration parameters, while the uncorrected does not.

This algorithm estimates satellite position based solely on a gravity field model and gravity gradient measurements in orbit. This process is programmed in Rust, using an optimised generation of the normalised associated Legendre functions to evaluate the gravity gradient calculation for every degree and order. This process assumes the gravity field models are static and up to date, and the instruments are all perfectly aligned with most systematic errors negligible.

A sensitivity analysis is performed to better understand the algorithm. It is observed that higher deviations from the correct trajectory take the algorithm more iterations to converge, and deviations larger than $211[km]$ on one of the axes converge into incorrect positions. Another test indicates that higher altitude orbits estimate positions with a higher converged deviations, while lower altitude orbits take more iterations to converge. One last test demonstrates that the computation of associated Legendre functions is the main factor for the duration of each least squares iteration. Iteration duration shows an exponential growth with maximum degree and order.

To verify the functionality of the algorithm, the process is done by comparing the differences between gravity field models. The measured gravity gradient is calculated from a model and the estimated gradient is calculated with another. Both the differences between gradients and position estimation deviation show linearity, thus verifying the procedure.

To validate the algorithm, real data is used in the form of GOCE satellite information provided by ESA. Level 2 data is used to obtain the measured gravity gradient. The least squares algorithm estimates positions with an average deviation of up to 0.688254% relative

to GOCE's GNSS positions. Despite the low percentage, it is a difference in the scale of $10[km]$, caused by the GOCE gradiometer errors and misalignments.

Test results show how several parameters affect the accuracy of the algorithm. An increase in maximum degree and order used for the estimation of gravity gradient increases the computation time exponentially. On the other hand, it shows an exponential decay in deviation that converges from maximum degree and order 80 onwards. When introducing scale coefficient and bias errors, a large deviation of up to 3.14% in the estimations is seen. When introducing a randomly generated white error between $0.1[E]$ and $-0.1[E]$, a perpetual deviation is formed with an average of 10^{-10}%. This is done with an uncorrected version of the algorithm. If the corrected version of the algorithm is used with bias, scale coefficient and white noise, it only generates a radial deviation average in the scale of 10^{-5}%. The corrected deviation is significantly smaller than the uncorrected deviation for bias and scale coefficient errors. This shows that the corrected least squares method can be used for an improvement in navigation of approximately 10^{5}%.

A final accuracy test is performed in which an initial deviation bias, scale coefficient and white noise are introduced. An optimal combination for the algorithm settings is obtained by decreasing the number of least squares iterations per epoch to the feasible amount of 2 to perform the estimation within the epoch time. And the maximum D/O used for estimation is 80. When using the combination, it yields average deviations of $9.81658 \cdot 10^{-5}$%, $9.43886 \cdot 10^{-5}$%, $9.99120 \cdot 10^{-5}$% for radius, longitude and latitude respectively.

Although the test results are promising and verifiable. Comparison to GOCE data shows that there are more calibration parameters to estimate. The accelerometer parameters in the gradiometer are not perfectly aligned with the gradiometer reference frame. The star trackers that provide transformation matrices when retrieving data also contain their own calibration parameters and errors. The next step to evaluate the least squares algorithm is to research how these parameters can affect its accuracy. Future work should include refined frame transformations, multiple epoch combinations, and further testing on a wider variety of orbits and bodies.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
COM	Centre Of Mass
D/O	Degree and Order
EGG	Electrostatic Gravity Gradiometer
GGT	Gravity Gradient Tensor
GNSS	Global Navigation Satellite System
GOCE	Gravity field and Ocean Circulation Explorer
GRF	Gradiometer Reference Frame
GSF	Geocentric Spherical Frame
SGG	Satellite Gravity Gradiometry
SSRF	Star Sensor Reference Frame

Symbols

Symbol	Definition	Unit
$\vec{a}$	Acceleration	[m/s^2]
A	Design Matrix	[-]
$\vec{b}$	Accelerometer Bias	[m/s^2]
$\vec{C}$	Scale coefficient	[-]
C_{pq}, S_{pq}	Spherical Harmonic Stokes Coefficients	[-]
$\vec{d}$	Non Gravitational Accelerations on COM	[m/s^2]
E_{al}	Accelerometer Misalignment Error	[-]
E_{co}	Inter-Axis Coupling Error	[-]
$\vec{g}$	Arbitrary Gravity Acceleration vector	[-]
G	Gravitational Constant	[Nm^2/kg^2]
K	Scale Factor Error	[-]
K_2	Transfer Function Error	[s^2/m]
$\vec{n}$	White Noise	[m/s^2]
M	Mass of Celestial Body	[kg]
p	Degree	[-]
P_{pq}	Associated Legendre Function	[-]
P_q^p	Superscript Associated Legendre Function	[-]
q	Order	[-]
r	Radial position in GSF	[m]
$\vec{l}$	Distance from Accelerometer to COM	[m]
R	Equatorial Radius of Celestial Body	[m]
u	GSF reference frame	[-]
V	Gravity Potential	[m^2/s^2]
$\vec{V}_u$	Gravity Acceleration	[m/s^2]

Symbol	Definition	Unit
$\vec{V}_{uu}$	Gravity Gradient Vector	[s^{-2}]
$\boldsymbol{V}_{uu}$	Gravity Gradient Tensor	[s^{-2}]
w	Weight Factor	[-]
$\boldsymbol{W}$	Weight Matrix	[-]
$\vec{y}$	Position Estimate	[m]
ρ	Density	[kg/m^3]
ω	Angular Velocity Component	[rad/s]
$\boldsymbol{\Omega}$	Angular Velocity Matrix	[rad/s]
$\dot{\boldsymbol{\Omega}}$	Angular Acceleration Matrix	[rad/s^2]
λ	Longitude in GSF	[rad]
ϕ	Latitude in GSF	[rad]
τ	Dampening Factor	[-]
∇	Vector Differential Operator	[-]

1 Introduction

1.1. Motivation

Over the past decade, the space sector has experienced a resurgence, driven in part by increased involvement from private companies and rapid technological advancements. As a result, larger and more ambitious missions are becoming feasible. However, the current global economic climate places greater emphasis on cost-efficiency, pushing the need for alternative technologies and methods in space engineering. One of the areas with significant potential for improvement is satellite navigation.

Traditional navigation systems, such as the Global Navigation Satellite System (GNSS) for Earth or the Deep Space Network (DSN) for interplanetary missions, rely heavily on external infrastructure. These systems can be costly, mass-intensive, and less efficient, particularly for orbits around distant bodies from Earth, where communication delays and large antennas add to mission complexity. A promising alternative is satellite gravity gradient navigation. Gravity gradiometry first proposed in the 1950s and applied to find Earth's gravity field by R. Rummel and O.L. Colombo in 1985 [1].

Since then, gravity field modeling and gravity gradiometry have advanced considerably. Modern gradiometers have become smaller, lighter, and more efficient, enabling more practical deployment. These advantages make Satellite Gravity Gradiometry (SGG) based navigation a compelling solution, particularly for missions where reliance on GNSS systems is difficult such as Lunar orbits, or impossible such as Mars orbits. Each celestial body has a unique gravity gradient signature, which can be used to estimate a satellite's relative position to a known gravity field model. While Earth's gravity field is highly dynamic due to tides, tectonic shifts, and human activity, more geologically stable bodies such as the Moon or Mars often have more consistent gravity models.

1.2. State of the Art

A literature study is performed separately from this document, in which the actual advancement in gravity gradient navigation is researched.

The Gravity Field and Ocean Circulation Explorer (GOCE) is a satellite that has been used to

obtained a detailed of Earth's gravity field by making use of gravity gradiometry. A. Bobojc et al. [1] proposes using gravity gradiometry data from the future GOCE mission to enhance satellite orbit determination. They found improvements when combining GNSS and gradiometer data via a weighted method to reduce errors, though this compromises the goal of full self sufficiency. One of the primary challenge noted was bias due to limited gradiometer bandwidth.

By 2015, Chen et al. [2] introduced an eigen-decomposition approach using GOCE's star tracker, gravity gradient data, and Earth rotation models. Their method achieved position and velocity errors of $120[m]$ and $0.125[m/s]$ respectively but lacked bias correction, reducing its real-world applicability.

In 2016, Sun et al. [2] addressed this by incorporating accelerometer biases into an Unscented Kalman Filter (UKF), based on Park et al. [3] sigma point propagation. Position errors ranged from $10.8[m]$ to $1208.3[m]$, and velocity errors from $0.013[m/s]$ to $1.2[m/s]$. By adjusting the bias variance matrices, errors were reduced to $10.4[m]$ – $677.0[m]$ and $0.012[m/s]$ – $0.80[m/s]$. This showed that axis-specific tuning of bias parameters can improve estimation. In 2022, Chen et al. [4] implemented an Extended Kalman Filter (EKF) with least squares to estimate and correct bias.

Kalman filter methods provide more accuracy gravity gradient navigation, but they also add complexity and further computation time. Spherical harmonic gravity gradient estimation is already a time consuming and computationally demanding process. Additional processing extends this disadvantage.

To address this, there is a need to research if a simpler least squares algorithm can handle previous limitations such as bias estimation. On top of this, gravity gradient navigation shows promise for orbits around celestial bodies that don't have GNSS availability. Algorithms have not yet been tested around a non-Earth orbit, since there is not as much orbital data to assess their success. This can be addressed by an orbit simulation.

1.3. Research Proposal

1.3.1. Questions

There is a no navigation algorithms for satellites outside of Earth orbit (non-GNSS environment), which use least squares to estimate bias too. This leads to the main question of the thesis:

How accurately can a least squares algorithm estimate satellite position in an environment without GNSS?

This is then divided into multiple sub-question.

- *How can bias and the scale factor be implemented into the least squares algorithm?*
- *How do variables affect the accuracy of the gravity gradient navigation method?*
- *What is the accuracy of the algorithm when real data is used?*

1.3.2. Requirements

The requirement are in the form of assumptions. This helps in focusing the research towards a more specific point, to better quantify results, and to know the limitations of the project. Here they are stated:

- The gravity field models are up to date. When it comes to performing any procedures around Earth, it is important to note that the gravity field of Earth shifts. Tectonic movement and tides for example, change the distribution of the mass of Earth. The errors that could arise from this fact are not taken into account.
- The accelerometer calibration parameters K_2, E_{co}, E_{al} can be assumed to be calibrated, and thus not included. The two exceptions being the bias $\vec{b}$ and the scale factor K (formulated as scale coefficient $\vec{C}$) of transforming from voltage to acceleration, as defined in Section 2.3.1.
- The bias $\vec{b}$ is assumed to be 3 global values (1 per axis) instead of 3 per accelerometer. This assumption is taken to estimate bias as part of the state in the least squares algorithm in each orbital epoch. This simplification is further touched upon in Section 2.3.1 and later in Section 7.3, where the consequences of taking this assumption are explained.
- Bias $\vec{b}$ is assumed to be static, thus, non-changing as the satellite orbit progresses because the testing of this algorithm is done through shorter orbit periods to keep the testing feasible in terms of computation time. With a short time frame, bias can be assumed to not change.
- White noise $\vec{n}$ is assumed to be a random Gaussian value with mean 0 and a maximum amplitude obtained from modern gradiometer accuracy in literature.
- In Chapter 6, the high D/O estimation of gravity gradient from the Moon orbit simulation is assumed to act as the measured gravity gradient.
- The distance between the accelerometers of the gradiometer l is assumed to be known by design, making it a parameter in the measured gravity gradient equation that does not have to be estimated.

1.3.3. Planning

This planning shows how different sections of this thesis help in answering the questions proposed in Section 1.3.1 and add to the content itself.

- The methodology (Chapter 2) answers the first sub-question, specifically Section 2.3.1 and Section 2.3.2, where calibration parameters are included into the estimation of the least squares method. The rest of the sections in the methodology describe the construction of the least squares algorithm, as well as details in how they are programmed.
- The Sensitivity Analysis (Chapter 3) partially answers the 2nd sub-question, in which initial deviation, orbital altitude and D/O are assessed on how they affect the algorithm. Primarily, it also serves as a section to know the limitations of the algorithm and how different circumstances can yield different results.
- The Verification (Chapter 4) is a section that is used to make sure the algorithm works correctly, not only as it is intended, but physically too.
- The Validation (Chapter 5) answers the 3rd sub-question by making use of a real case scenario in the form of GOCE data. It also describes the process in which this data is retrieved and transformed to be used by the algorithm.

- The Results (Chapter 6) finalise the answer to the 2nd sub-question by testing algorithm variables that can affect accuracy by making use of data from a Moon orbit simulation. With the knowledge of the optimal algorithm variables for this orbit, a final test is conducted where all calibration parameters and errors are introduced to finally answer what the accuracy to answer the main question.
- The Conclusion (Chapter 7) discusses directly the answers to the thesis questions. It also tackles the limitations of the algorithm and recommends future work.

2

Algorithm Design

This chapter describes the design of the least squares estimation algorithm on which the accuracy assessment is based.

2.1. Gravity Gradient Formulation

The objective is to obtain the data needed to analyse gravity gradient satellite navigation. For this, knowledge of the gravity field is needed. This section explains how the gravity gradient is calculated, and what each of its components are.

Gravity field is conservative, which means that it is path independent, thus the potential energy differences between points in the field are a function of position only. This characteristic is relevant to using gravity field for navigation, as their position dependant values do not change because of the orbit path the satellite performs. The basic properties of the gravity field are associated with a linear potential field $V(x, y, z)$ (where x, y, z are coordinates of arbitrary axes) such that:

$$\begin{aligned} \nabla V &= -\vec{g} \\ \nabla \times V &= 0 \\ \frac{d^2}{dr^2}V + \frac{d^2}{d\lambda^2}V + \frac{d^2}{d\phi^2}V &= 0 \end{aligned}, \tag{2.1}$$

due to the gravity field being harmonic. Gravity gradient is obtained as a second spacial derivative of gravity potential V. Assuming the source is a point mass, the simplified equation for the gravity potential is

$$V = -G\frac{M}{r}, \tag{2.2}$$

where G is the gravitational parameter of the body, M is its mass, and r is the radial distance from its centre of mass (COM). Equation 2.2 is an ideal equation, and not true to asymmetrical bodies. Gravitational force, as seen in in the same equation, comes from the mass of a body. Bodies such as large as planets or natural satellites are more complex than a point mass. They vary in shape and density. This means, that gravity potential can be formulated as the integral of density over radius through volume:

$$\nabla^2 V = \mathbf{V} = -G \int\int\int \frac{\rho(x,y,z)}{r(z,y,z)} dxdydz \tag{2.3}$$

To be able to represent gravity field, spherical harmonics are used, which are a solution of Laplace's equation in spherical coordinates. They make use of the associated Legendre function P_{pq} and a pair of coefficients C_{pq} and S_{pq} called the Stokes coefficients:

$$\mathbf{V}_{uu} = \frac{GM}{r} \sum_{p=0}^{\infty} \sum_{q=0}^{p} \left(\frac{R}{r}\right)^{p+1} (C_{pq} \cos(q\lambda) + S_{pq} \sin(q\lambda)) P_{pq}(\sin(\phi)), \tag{2.4}$$

where $\boldsymbol{V}_{uu}$ is the gravity gradient tensor (GGT). Equation 2.4 depends on the spherical coordinates in the Geocentric Spherical Frame (GSF) $u(r, \lambda, \phi)$, where r is radius from the centre of the celestial body, λ is the longitude, and ϕ is the latitude, u i sued to represent all 3 spherical coordinates. The associated Legendre function makes use of degree p and order q. The larger the degree and order (D/O), the smaller the details it represents in the body. The last components that make this possible for mapping are the Stokes coefficients C_{pq} and S_{pq}. These determine how prominent each degree and order term in the summation is. These coefficients come from gravity field models obtained from measurements with gradiometers in missions such as GOCE. These are unique to each body, but vary from model to model due to different techniques used, measuring instrument accuracy, and time in which they are measured. In summary, with a spherical harmonic gravity field, by summing up all elements which vary in D/O, every point around the body becomes unique in its gravity potential. This will be the basis for the method used in this thesis.

2.1.1. Stokes Coefficients

One of the key components in representing a gravity field with spherical harmonics are the Stokes coefficients C_{pq}, S_{pq}. These coefficients are taken from gravity field models and encapsulate the gravitational characteristics of a celestial body. The maximum resolution of the gravity field model is determined by the maximum D/O that can be reliably used in the spherical harmonic expansion. These coefficients are commonly given in a normalised form. Section 2.1.2 shows more details on this normalisation.

The gravity field models used in this thesis are limited to those constructed solely from satellite data, without incorporating ground-based measurements. This restriction aligns with the main objective of the work: to assess navigation with satellite autonomous data in an environment where only gradiometer data is used. While some Earth gravity models achieve high resolution, like "$XGM2019e$" [5], reach up to D/O 5540, 2190 and 760. A high resolution is because of terrestrial gravity data. Instead, for Earth, a model like "$GOSG02S$" [6] reaches D/O 300 and it is purely formed from data obtained by GOCE.

In this thesis, the algorithm makes use of the "$GOSG02S$" gravity field model for Earth and the "$GRGM600PRIM$" model [7], which is a version of the "$GRGM1200A$" model truncated to D/O 660 for the Moon. Other models may be used for their respective bodies. The results in this case should yield similar patterns, with small changes to the results due to different measurements being used to create the models, a different maximum D/O, or different techniques used to formulate the gravity field.

2.1.2. Associated Legendre Function

The Associated Legendre Function, denoted as $P_p^q(x)$, is a variable function that arises in the solution of the Laplace equation in spherical coordinates. It plays an important role in the representation of the gravitational potential of a celestial body using spherical harmonics by providing a latitudinal dependence that forms the shape of the field. From Equation 2.4, it can be seen that it is a contributor to the shape of the gravity field, and thus the calculation of gravity potential and gradient. The value of the associated Legendre function can be obtained with the use of the Legendre polynomials:

$$P_p^q(x) = (-1)^q (1-x^2)^{\frac{q}{2}} \frac{d^q}{dx^q} P_p(x), \tag{2.5}$$

where P_p represents the Legendre polynomial, a formula dependent only on degree p and not order. It is an integer based solution to the Legendre differential equation. This is obtained by using Rodrigues' Formula:

$$P_p(x) = \frac{1}{2^p p!} \frac{d^p}{dx^p} \left((x^2-1)^p \right) \tag{2.6}$$

At lower D/O, the associated Legendre function is relatively straightforward to compute. However, as the D/O increase, the formulation becomes more complex and computationally expensive. The Legendre polynomial is unfortunately difficult and lengthy to derive for high degrees, particularly when using Equation 2.6. To address this, Dobrokhotov et al. [8] indicates that a more efficient alternative formulation is:

$$\begin{aligned} (p+1)P_{p+1}(x) &= (2p+1)P_p - pP_{p-1} \\ pP_p(x) &= (2p-1)P_{p-1} - (p-1)P_{p-2} \\ P_p &= \frac{2p-1}{p} P_{p-1} - \frac{p-1}{p} P_{p-2} \end{aligned} \tag{2.7}$$

Equation 2.7 encounters a second problem upon computation. This function is dependent on 2 lower degree functions. These lower degree functions subsequently are dependant on 2 lower degree functions each, and so on until reaching degree 0. When approached directly in a computer algorithm, the number of equations computed by this method scale with 2^n, where n is the degree computed at that point. Figure 2.1 shows how this happens.

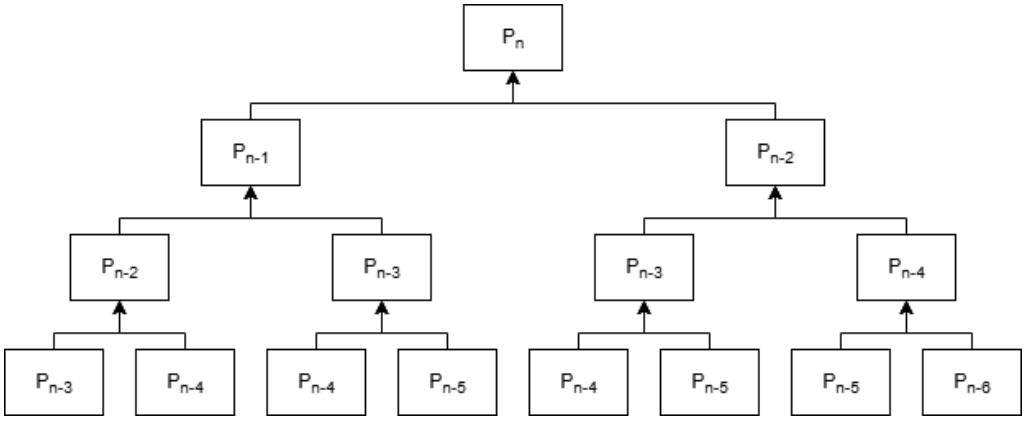

Figure 2.1: Flowchart of how Equation 2.7 computes the Legendre polynomial.

In high degrees such as 300, the number of equations needed to compute P_{300} is 2^{300}. Even though this bypasses having to make use of symbolic differentiation 300 times, it introduces a new problem of nested functions. When evaluating a gravity function, such as gravity acceleration, with spherical harmonics, all degrees must be computed. What this means is that number of equations computed increases now to $\sum_0^n 2^n$.

To bypass this next problem, a new method to make use of Equation 2.7 is proposed in this thesis. The process is initialised by setting the first two Legendre Polynomials to $P_0 = 1$ and $P_1 = x$. They are both saved into a list of computed functions. Then, the next one is computed by making use of the previous two by using Equation 2.7. It is saved to the list and used for the next one subsequently, and so on, until the maximum degree is obtained. Figure 2.2 shows a flowchart of how this works.

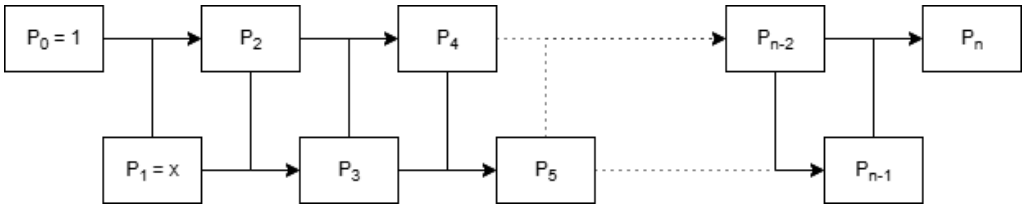

Figure 2.2: Flowchart of how Equation 2.7 can be used in this new more effective manner

The new method accomplishes 2 things:

1. Reduces the number of computations needed to form a Legendre function from 2^n to n, as there are no longer any nested functions.
2. Due to the fact that each computed function is saved before approaching the next degree, it means that during the computation of the highest degree, all others are saved too. The number of equations needed to be solved goes from $\sum_0^n n$ to n.

The overall improvement is immensely noticeable with a transition in the computations needed of $\sum_0^n 2^n \rightarrow n$. A test is done in Rust, to compare the time needed to compute Legendre polynomials up to and including degree $p = 60$. The nested method yields a computation time of $630.87[s]$. The new method yields a computation time of $9.0510 \cdot 10^{-4}[s]$. The change in computation time should be in the scale of 10^{-15}%, but the limitations of the hardware being used for this thesis form a change in the scale of 10^{-4}%. The change is still an extremely significant difference in computation time, and justifies the new method via which the associated Legendre function is computed.

To compute the associated Legendre Function, the notation used is used in spherical harmonics. Equation 2.4 shows P_{pq}. This is because P_{pq} cancels out the intermittent negative sign in the associated Legendre function:

$$\begin{aligned} P_{pq}(x) &= P_p^q(x)(-1)^q \\ P_{pq}(x) &= (1-x^2)^{\frac{q}{2}} \frac{d^q}{dx^q} P_p(x) \end{aligned} \tag{2.8}$$

The associated Legendre Function introduces order q, which scales from 0 up to the current degree p. An expanded version of Equation 2.7 can be used, which incorporates the order and is used to compute higher degrees within the same order, as shown in Equation 2.9.

$$P_{pq} = \frac{2p-1}{p-q} P_{p-1\,q} - \frac{p+q-1}{p-q} P_{p-2\,q} \tag{2.9}$$

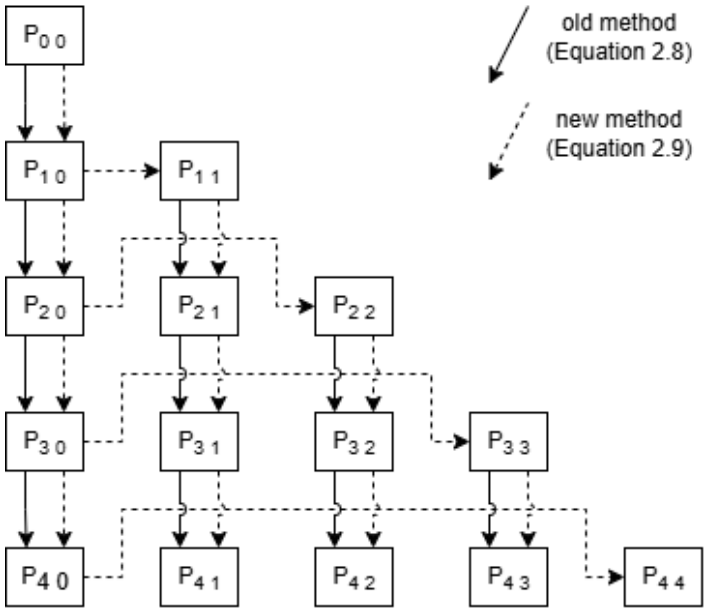

Figure 2.3: Flowchart of how the associated Legendre functions are computed. It shows how Equation 2.8 formulates the values, as represented by solid arrows. It also shows how adding the recurrence in Equation 2.9 streamlines the formulation, as represented by dashed arrows.

With this, all the associated Legendre functions can be pre-computed as a symbolic equation before the main algorithm runs, which can be later evaluated by substituting its x elements by the numerical values of sin (ϕ) when computing spherical harmonic gravity gradient. By pre-computing the equations in a symbolic state, all the process of calculating the associated Legendre functions in every computation is reduced to evaluating them, as seen in Equation 2.4.

To make the associated Legendre functions compute adequately in a functional gravity equation, it needs to be normalised. As mentioned in Section 2.1.1, the Stokes coefficients are given in a normalised form. Associated Legendre function values tend to get extremely high at high D/O. This makes it impossible to compute through regular means without overflowing numerical computations. To avoid this, the functions are normalised too with a specific coefficient N_{pq}.

$$N_{pq} = \sqrt{(2n+1)\frac{(p-q)!}{(p+q)!}} \tag{2.10}$$

This normalised form provides computational stability, and prevents high D/O solutions from diverging into values too large to calculate.

2.2. Gravity Gradient Tensor and Factorisation

To obtain the gravity gradient derivative, the gravity gradient estimate must firsts be calculated. For this, the second derivative of the spherical harmonic gravity potential equation

(Equation 2.4) is obtained. Appendix A contains the exact equations of all derivatives, while below there is an example the equation for the second spacial derivative of the gravity potential with respect to state r:

$$V_{rr} = \frac{GM}{R}\left(\frac{1}{r^2}\right)\sum_{p=0}^{p_{max}}\sum_{q=0}^{p}(p+1)(p+2)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda))P_{pq_s} \tag{2.11}$$

The computation of each element of the tensor $\boldsymbol{V}$ requires evaluating numerous terms, and it takes a significant amount of computations to calculate. This is especially due to the inclusion of the associated Legendre function. To make these computations more efficient, the factorisation mentioned below is done in order to build these equations more efficiently. Firstly, it must be observed in Equation 2.41, there are only 6 distinct forms of V_{uu}, due to the symmetrical nature of a gravity gradient tensor, in which $V_{r\lambda} = V_{\lambda r}$, $V_{r\phi} = V_{\phi r}$, and $V_{\lambda\phi} = V_{\phi\lambda}$. With the derived equations for the gravity gradient tensor elements (derivations shown in Section A.1.3), and knowing that each parameter is independent from each other, their factors can be isolated:

$$\vec{F}^{(r)} = \begin{bmatrix}\left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\end{bmatrix} \tag{2.12}$$

$$\vec{F}^{(\lambda)} = \begin{bmatrix}C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda)+S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda))\end{bmatrix} \tag{2.13}$$

$$\vec{F}^{(\lambda)} = \begin{bmatrix}P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2P''_{pq}(\sin(\phi))-\sin(\phi)P'_{pq}(\sin(\phi))\end{bmatrix} \tag{2.14}$$

A vector Q with only 1 is formed with the amount of distinct elements (6) in the gravity gradient tensor, since it is a symmetrical matrix.

$$\vec{Q} = \begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1\end{bmatrix}^T \tag{2.15}$$

By Multiplying these two vectors, a matrix with the number of columns being the number of factors in $F^{(r)}$ and the number of rows being the number of distinct elements in the GGT.

$$\vec{Q}\vec{F}^r = \begin{bmatrix}\left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}\end{bmatrix} \tag{2.16}$$

Finally, a new matrix is formed to represent which factors are on each of the GGT elements. In the case of r, the matrix is:

$$\mathbf{E}^{(r)} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{2.17}$$

The QF matrix is elevated element by element to the power of E. And then, all the elements per row are multiplied:

$$\Pi_{j=1}^{j^{(r)}}(\vec{Q}\vec{F}_j^{(r)})^{\mathbf{E}^{(r)}} = \begin{bmatrix} (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} \\ (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} \end{bmatrix} \tag{2.18}$$

This process is done for all parameters and then all the factors are multiplied as seen below:

$$\mathbf{V}_{uu_{pq}} = \frac{GM}{R}\Pi_{k=r,\lambda,\phi}\Pi_{j=1}^{j^k}(\vec{Q}\vec{F}_j^k)^{\mathbf{E}^k} \tag{2.19}$$

where p and q are the degree and order respectively, which are parameters present in the λ factors. With all 6 $\mathbf{V}_{uu_{pq}}$ elements obtained, the same process is done in Appendix A to obtain its Jacobian $\nabla^3 V_{pq} = \mathbf{V}_{uuu_{pq}}$, of which, out of the 18 possible values, only 10 are unique. This is used to construct both $\boldsymbol{V}_{uu}$ and $\boldsymbol{V}_{uuu}$ in the least squares algorithm. Figure 2.4 shows the process carried for the spherical harmonic gravity calculation.

The orbital position state $\vec{y}$ at the present epoch is introduced. The current degree and order (initialised at 0 and 0 respectively) are introduced. With the D/O, the Stoke coefficients C_{pq} an S_{pq} are extracted from their list. Similarly, the associated Legendre function and its differentials P_{pq}, P'$_{pq}$, and P''$_{pq}$ are extracted from their symbolic list. The state $\vec{y}$ is used to evaluate the symbolic associated Legendre functions. Then, it is used to compute the separate factors that will be used in the factorisation process for $F^{(r)}$, $F^{(\lambda)}$ and $F^{(\phi)}$. With the factorisation vectors, the constructions of the actual components occurs as shown in Equation 2.19. Since the value of $V_{uu_{pq}}$ or $V_{uuu_{pq}}$ is for one D/O combination only, it is summed to a total $\boldsymbol{V}_{uu}$ or $\boldsymbol{V}_{uuu}$ and the process is repeated until all the D/O combinations are covered up to the maximum D/O. $\boldsymbol{V}_{uuu}$ is computed as the gradient of the GGT elements.

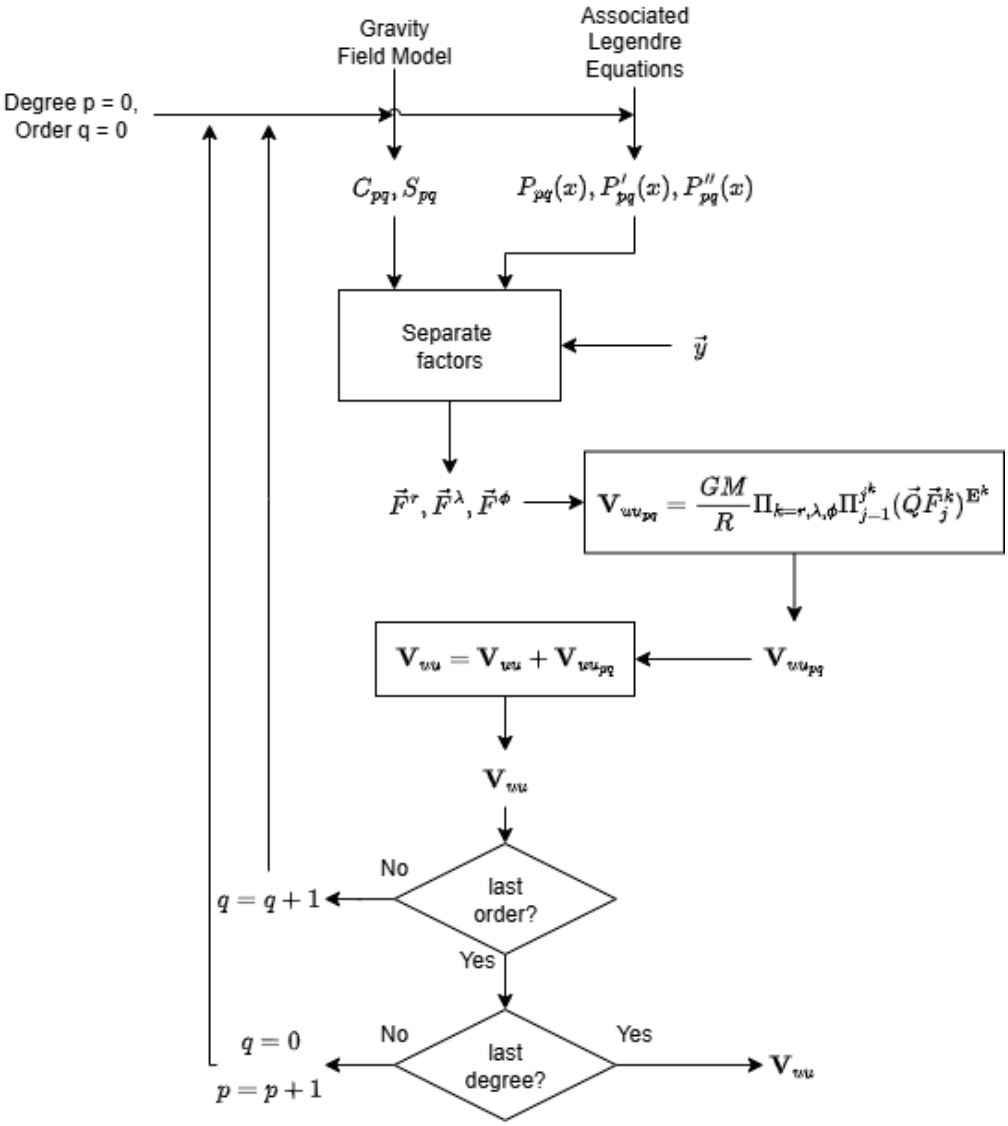

Figure 2.4: Diagram of how $\mathbf{V}_{uu}$ is computed in the algorithm.

Factorisation reduces the number of computations. Each element in the GGT has different factors, and the number of times their factors are computed are reduced from 6 to 3 in the case of $\mathbf{V}_{uu_{pq}}$ and from 10 to 4 in the case of $\mathbf{V}_{uuu_{pq}}$. This reduction of at least 50% means that the computational time for this part is halved, plus the time it would take to perform the matrix calculations in Equation 2.16 and Equation 2.18.

2.3. Least Squares Method

The gravity field is measured by making use of gradiometers to obtain the GGT, which is presented as $\boldsymbol{V}_{uu}$. In this case, the coordinates are represented as r, λ, ϕ, which means that the second derivatives can be written as:

$$\mathbf{V}_{uu} = \begin{bmatrix} V_{rr} & V_{r\lambda} & V_{r\phi} \\ V_{\lambda r} & V_{\lambda\lambda} & V_{\lambda\phi} \\ V_{\phi r} & V_{\phi\lambda} & V_{\phi\phi} \end{bmatrix} \tag{2.20}$$

In this thesis, this form is referred to as the uncorrected GGT, because it does not take into account any kind of calibration parameter correction. When performing derivatives, it can be seen in Equation 2.4 that each of the terms that contain the spherical coordinates are independent from each other. The derivatives can thus be done only for each parameter independently. This means that the matrix seen in Equation 2.20 is a symmetric matrix. Only 6 of the 9 parameters need to be obtained.

$\mathbf{V}_{uu}$ can be formulated in the uncorrected vector form $\vec{V}_{uu}$ as shown in Equation 2.21, where only its 6 distinct terms are used.

$$\vec{V}_{uu} = \begin{bmatrix} V_{rr} \\ V_{r\lambda} \\ V_{r\phi} \\ V_{\lambda\lambda} \\ V_{\lambda\phi} \\ V_{\phi\phi} \end{bmatrix} \tag{2.21}$$

With $\vec{V}_{uu}$ and observations the an uncorrected least squares approach can be used to estimate the position of the satellite. This can be represented as:

$$\vec{y}_{i+1} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T(\vec{V}_{uu}^{(measured)} - \vec{V}_{uu}(\vec{y}_i)) + \vec{y}_i \tag{2.22}$$

where $\vec{y}_0$ is an initial guess of the satellite position to start the algorithm. i represents the iteration number, since Equation 2.22 is performed multiple times in each orbital epoch while using the result of the previous calculation to converge into an estimated point. The Jacobian **A** represents the uncorrected design matrix of the least squares method. The measured gravity gradient vector is $\vec{V}_{uu}^{(measured)}$, which represents the measurement data. This, instead of coming from Equation 2.4, comes from the measurements of the instrumentation within the satellite. The satellite position state is represented by the uncorrected position vector $\vec{y}$ in GSF:

$$\vec{y} = \begin{bmatrix} r \\ \lambda \\ \phi \end{bmatrix} \tag{2.23}$$

Equation 2.22 is used in an iterative manner so that the position estimate converges towards a value.

Thus, the design matrix is the Jacobian of the gravity gradient vector and equivalent to the GGT gradient, thus $\mathbf{A} = \boldsymbol{V}_{uuu}$.

$$\mathbf{A} = \frac{d}{du}\vec{V}_{uu} = \begin{bmatrix} V_{rrr} & V_{rr\lambda} & V_{rr\phi} \\ V_{r\lambda r} & V_{r\lambda\lambda} & V_{r\lambda\phi} \\ V_{r\phi r} & V_{r\phi\lambda} & V_{r\phi\phi} \\ V_{\lambda\lambda r} & V_{\lambda\lambda\lambda} & V_{\lambda\lambda\phi} \\ V_{\lambda\phi r} & V_{\lambda\phi\lambda} & V_{\lambda\phi\phi} \\ V_{\phi\phi r} & V_{\phi\phi\lambda} & V_{\phi\phi\phi} \end{bmatrix} \tag{2.24}$$

2.3.1. Accelerometer equation

The acceleration measured by the accelerometers within a satellite can be formulated in the form of:

$$\vec{a} = (\mathbf{\Omega}^2 + \dot{\mathbf{\Omega}} - \mathbf{V}_{uu})\vec{l} + \vec{d} \tag{2.25}$$

where $\vec{a}$ is the measured acceleration, $\mathbf{\Omega}$ is the angular velocity of the satellite used for the centrifugal acceleration $\mathbf{\Omega}^2\vec{l}$, $\dot{\mathbf{\Omega}}$ is the angular acceleration of the satellite used for the Euler acceleration $\dot{\mathbf{\Omega}}\vec{l}$, $\vec{l}$ is the distance from the accelerometer to the centre of mass of the satellite, $\vec{d}$ represents all the combined non gravitational accelerations such as solar radiation or acceleration caused by attitude control devices within the satellite itself. The angular velocity $\mathbf{\Omega}$ is expressed in the form of the matrix as shown in:

$$\mathbf{\Omega} = \begin{bmatrix} 0 & -\omega_\phi & \omega_\lambda \\ \omega_\phi & 0 & -\omega_r \\ -\omega_\lambda & \omega_r & 0 \end{bmatrix} \tag{2.26}$$

The acceleration $\dot{\mathbf{\Omega}}$ is expressed similarly as:

$$\dot{\mathbf{\Omega}} = \begin{bmatrix} 0 & -\dot{\omega}_\phi & \dot{\omega}_\lambda \\ \dot{\omega}_\phi & 0 & -\dot{\omega}_r \\ -\dot{\omega}_\lambda & \dot{\omega}_r & 0 \end{bmatrix} \tag{2.27}$$

The square of the angular velocity $\mathbf{\Omega}^2$ used to find centrifugal acceleration is expressed as:.

$$\mathbf{\Omega}^2 = \begin{bmatrix} -\omega_\lambda - \omega_\phi & \omega_r\omega_\lambda & \omega_r\omega_\phi \\ \omega_r\omega_\lambda & -\omega_r - \omega_\phi & \omega_\lambda\omega_\phi \\ \omega_r\omega_\phi & \omega_\lambda\omega_\phi & -\omega_r - \omega_\lambda \end{bmatrix} \tag{2.28}$$

Equation 2.25 can be re-arranged in order to obtain $\boldsymbol{V}_{uu}$:

$$\boldsymbol{V}_{uu} = -\vec{a}\frac{1}{\vec{l}} + \mathbf{\Omega}^2 + \dot{\mathbf{\Omega}} + \vec{d}\frac{1}{\vec{l}} \tag{2.29}$$

From what can be seen in the equation, it is possible to obtain $\boldsymbol{V}_{uu}$ from the acceleration data $\vec{a}$ given by the accelerometers, while $\mathbf{\Omega}$ and $\dot{\mathbf{\Omega}}$ can be obtained by attitude determination instrumentation such as a star tracker, the non-gravitational accelerations $\vec{d}$ however, can be determined with gradiometer readings by using common mode, in which accelerations between two opposing accelerometers are averaged and $l = 0$, thus leaving only $\vec{d}$ from these readings. This is not the only unknown. The acceleration obtained from the accelerometer readings also contains systematic errors:

$$\vec{a} = (K + E_{\mathsf{al}} + E_{\mathsf{co}})\vec{a}^{(real)} + K_2\vec{a}^{(real)^2} + \vec{b} + \vec{n} \tag{2.30}$$

The real acceleration is represented by $\vec{a}^{(real)}$, The errors and calibration parameters are the following:

- The scale factor K, as mentioned by P. Touboul et al. [9] is an uncertainty in the electrostatic gain when converting voltage to acceleration within the accelerometers, thus it is a coefficient to acceleration $\vec{a}^{(real)}$. In this thesis, it is not ignored, as it is a very common systematic error.
- Accelerator misalignment E_{al} and inter-axis coupling E_{rmco} are two causes for calibration parameters in the equation. E_{al} occurs as a result of an angular deviation from the ideal accelerometer arrangement. E_{co} occurs as a result of accelerometer axes not being perfectly perpendicular to each other. In this thesis, accelerometers and all instrumentation is assumed to be aligned to make the processing of data easier, thus E_{al} and E_{co} are assumed to be 0.
- C. Stummer [10] writes about the quadratic factor error, which appears from the transfer function when converting the accelerometer's voltage into acceleration. It is represented by K_2, and it acts as a coefficient with $\vec{a}^{(real)^2}$. B. Frommknecht [11] states that non linear acceleration errors can be calibrated by making use of a control loop. In this thesis, it is assumed that this error is also 0.
- Systematic white noise, represented by $\vec{n}$, comes from the accelerometers measurements. The measurements have a degree of uncertainty, represented by noise. Noise happens in all 3 axes and here it is assumed to be white noise. It has a Gaussian probability distribution.
- Accelerometer bias is present in each of its 3 axes and is represented by $\vec{b}$. A bias is a specific value that each accelerometer has due to any imperfection or drift within its mechanism. Unlike noise, it is not a random error, but instead a starting value which may drift with time. In a 3 axis gradiometer, there are 18 different biases to take into account due to the 6 accelerometers that are utilised. In this thesis, the bias is assumed to be global to the gradiometer, instead of local to each accelerometer. This is done to have 3 bias elements corrected so that it can be done in 1 epoch. It is also assumed to be static.

With most of the errors assumed to not take place, the equation is left as:

$$\vec{a} = K\vec{a}^{(real)} + \vec{b} + \vec{n} \tag{2.31}$$

Substituting into Equation 2.29

$$\boldsymbol{V}'_{uu} = -(K\vec{a} + \vec{b} + \vec{n})\frac{1}{\vec{l}} + \boldsymbol{\Omega}^2 + \dot{\boldsymbol{\Omega}} + \vec{d}\frac{1}{\vec{l}} \tag{2.32}$$

$$\boldsymbol{V}'_{uu} = -K\vec{a}\frac{1}{\vec{l}} + \boldsymbol{\Omega}^2 + \dot{\boldsymbol{\Omega}} + \vec{d}\frac{1}{\vec{l}} - \vec{b}\frac{1}{\vec{l}} - \vec{n}\frac{1}{\vec{l}} \tag{2.33}$$

Here, the corrected GGT $\boldsymbol{V}'_{uu}$ is formulated to introduce calibration parameters and errors into the least squares algorithm. The prime superscript $'$ represents that this GGT considers accelerometer calibration parameters. It is known in this thesis as "corrected". The addition of the bias component can be represented in matrix form as:

$$\vec{b}\frac{1}{\vec{l}} = \begin{bmatrix} b_r \\ b_\lambda \\ b_\phi \end{bmatrix} \frac{1}{||\vec{l}||^2} \begin{bmatrix} l_r & l_\lambda & l_\phi \end{bmatrix} \tag{2.34}$$

The values of the $\vec{l}$ vector in Equation 2.34 are different for each satellite design and configuration, but they are assumed to always be known. This only leaves the terms b_r, b_λ, and b_ϕ as unknowns, which need to be estimated along with the position state of the satellite.

Equation 2.33 can be rearranged into:

$$\boldsymbol{V}'_{uu} = \boldsymbol{C}\boldsymbol{V}_{uu} - \vec{b}\frac{1}{\vec{l}} - \vec{n}\frac{1}{\vec{l}} \tag{2.35}$$

where $\boldsymbol{C}$ is a coefficient which acts as an unknown to represent K and $\vec{d}$ and is formulated as:

$$\boldsymbol{C} = \frac{-K\vec{a}\frac{1}{\vec{l}} + \boldsymbol{\Omega}^2 + \dot{\boldsymbol{\Omega}} + \vec{d}\frac{1}{\vec{l}}}{-\vec{a}\frac{1}{\vec{l}} + \boldsymbol{\Omega}^2 + \dot{\boldsymbol{\Omega}} + \vec{d}\frac{1}{\vec{l}}} \tag{2.36}$$

In this thesis, it is named the Scale Coefficient. The denominator of this coefficient corresponds to the previously set formulation of $\boldsymbol{V}_{uu}$ in Equation 2.29. This is done in order to keep $\vec{d}$ as part of $\boldsymbol{C}$, and thus a parameter which can be estimated by least squares. The matrix form for $K\vec{a}\frac{1}{\vec{l}}$ is:

$$K\vec{a}\frac{1}{\vec{l}} = \begin{bmatrix} K_r a_r \\ K_\lambda a_\lambda \\ K_\phi a_\phi \end{bmatrix} \frac{1}{||\vec{l}||^2} \begin{bmatrix} l_r & l_\lambda & l_\phi \end{bmatrix} \tag{2.37}$$

The representation of $\vec{d}$ in matrix form is very similar to Equation 2.34.

$$\vec{d}\frac{1}{\vec{l}} = \begin{bmatrix} d_r \\ d_\lambda \\ d_\phi \end{bmatrix} \frac{1}{||\vec{l}||^2} \begin{bmatrix} l_r & l_\lambda & l_\phi \end{bmatrix} \tag{2.38}$$

The top rows of unknowns all correspond to the r terms, the middle row to the λ terms, and the bottom row to the ϕ terms. This makes it possible to formulate $\boldsymbol{C}$ and subsequently $\boldsymbol{C}\boldsymbol{V}_{uu}$ as the following formula.

$$\boldsymbol{C}\boldsymbol{V}_{uu} = \begin{bmatrix} C_r & 0 & 0 \\ 0 & C_\lambda & 0 \\ 0 & 0 & C_\phi \end{bmatrix} \begin{bmatrix} V_{rr} & V_{r\lambda} & V_{r\phi} \\ V_{\lambda r} & V_{\lambda\lambda} & V_{\lambda\phi} \\ V_{\phi r} & V_{\phi\lambda} & V_{\phi\phi} \end{bmatrix} \tag{2.39}$$

White noise $\vec{n}$ is assumed to be a random Gaussian value with a mean of 0. The standard deviation depends on the accuracy of the instrumentation provided. The new GGT is formulated as:

$$\boldsymbol{V}'_{uu} = \begin{bmatrix} V'_{rr} & V'_{r\lambda} & V'_{r\phi} \\ V'_{\lambda r} & V'_{\lambda\lambda} & V'_{\lambda\phi} \\ V'_{\phi r} & V'_{\phi\lambda} & V'_{\phi\phi} \end{bmatrix} = \begin{bmatrix} C_r V_{rr} - \frac{b_r l_r}{||\vec{l}||^2} & C_r V_{r\lambda} - \frac{b_r l_\lambda}{||\vec{l}||^2} & C_r V_{r\phi} - \frac{b_r l_\phi}{||\vec{l}||^2} \\ C_\lambda V_{\lambda r} - \frac{b_\lambda l_r}{||\vec{l}||^2} & C_\lambda V_{\lambda\lambda} - \frac{b_\lambda l_\lambda}{||\vec{l}||^2} & C_\lambda V_{\lambda\phi} - \frac{b_\lambda l_\phi}{||\vec{l}||^2} \\ C_\phi V_{\phi r} - \frac{b_\phi l_r}{||\vec{l}||^2} & C_\phi V_{\phi\lambda} - \frac{b_\phi l_\lambda}{||\vec{l}||^2} & C_\phi V_{\phi\phi} - \frac{b_\phi l_\phi}{||\vec{l}||^2} \end{bmatrix}, \tag{2.40}$$

where it is shown how the corrected GGT V'_{uu} relates to the uncorrected GGT V_{uu}.

2.3.2. Corrected Design Matrix

With bias, scale coefficient to be estimated, the least squares method must be expanded to accommodate them. As observed from Equation 2.40, $\boldsymbol{V}'_{uu}$ is no longer a symmetric matrix, and all 9 terms need to be computed. The corrected vector form $\vec{V}'_{uu}$ for the least squares method is:

$$\vec{V}'_{uu} = \begin{bmatrix} V'_{rr} \\ V'_{r\lambda} \\ V'_{r\phi} \\ V'_{\lambda r} \\ V'_{\lambda\lambda} \\ V'_{\lambda\phi} \\ V'_{\phi r} \\ V'_{\phi\lambda} \\ V'_{\phi\phi} \end{bmatrix} = \begin{bmatrix} C_r V_{rr} - b_r \frac{l_r}{||\vec{l}||^2} \\ C_r V_{r\lambda} - b_r \frac{l_\lambda}{||\vec{l}||^2} \\ C_r V_{r\phi} - b_r \frac{l_\phi}{||\vec{l}||^2} \\ C_\lambda V_{r\lambda} - b_\lambda \frac{l_r}{||\vec{l}||^2} \\ C_\lambda V_{\lambda\lambda} - b_\lambda \frac{l_\lambda}{||\vec{l}||^2} \\ C_\lambda V_{\lambda\phi} - b_\lambda \frac{l_\phi}{||\vec{l}||^2} \\ C_\phi V_{r\phi} - b_\phi \frac{l_r}{||\vec{l}||^2} \\ C_\phi V_{\lambda\phi} - b_\phi \frac{l_\lambda}{||\vec{l}||^2} \\ C_\phi V_{\phi\phi} - b_\phi \frac{l_\phi}{||\vec{l}||^2} \end{bmatrix} \tag{2.41}$$

The design matrix is the differential of this vector with respect to the coordinates, the scale coefficients $\vec{C}$ and the biases $\vec{b}$. The corrected state $\vec{y}'$ to be estimated is:

$$\vec{y}' = \begin{bmatrix} r & y\lambda & y\phi & C_r & C_\lambda & C_\phi & b_r & b_\lambda & b_\phi \end{bmatrix}^T \tag{2.42}$$

The corrected design matrix $\boldsymbol{A}'$ contains 81 elements.

$$\begin{aligned} \boldsymbol{A}' = \frac{d}{d\vec{y}}\vec{V}'_{uu} &= \begin{bmatrix} V'_{rrr} & V'_{rr\lambda} & V'_{rr\phi} & V'_{rrC_r} & V'_{rrC_\lambda} & V'_{rrC_\phi} & V'_{rrb_r} & V'_{rrb_\lambda} & V'_{rrb_\phi} \\ V'_{r\lambda r} & V'_{r\lambda\lambda} & V'_{r\lambda\phi} & V'_{r\lambda C_r} & V'_{r\lambda C_\lambda} & V'_{r\lambda C_\phi} & V'_{r\lambda b_r} & V'_{r\lambda b_\lambda} & V'_{r\lambda b_\phi} \\ V'_{r\phi r} & V'_{r\phi\lambda} & V'_{r\phi\phi} & V'_{r\phi C_r} & V'_{r\phi C_\lambda} & V'_{r\phi C_\phi} & V'_{r\phi b_r} & V'_{r\phi b_\lambda} & V'_{r\phi b_\phi} \\ V'_{\lambda rr} & V'_{\lambda r\lambda} & V'_{\lambda r\phi} & V'_{\lambda rC_r} & V'_{\lambda rC_\lambda} & V'_{\lambda rC_\phi} & V'_{\lambda rb_r} & V'_{\lambda rb_\lambda} & V'_{\lambda rb_\phi} \\ V'_{\lambda\lambda r} & V'_{\lambda\lambda\lambda} & V'_{\lambda\lambda\phi} & V'_{\lambda\lambda C_r} & V'_{\lambda\lambda C_\lambda} & V'_{\lambda\lambda C_\phi} & V'_{\lambda\lambda b_r} & V'_{\lambda\lambda b_\lambda} & V'_{\lambda\lambda b_\phi} \\ V'_{\lambda\phi r} & V'_{\lambda\phi\lambda} & V'_{\lambda\phi\phi} & V'_{\lambda\phi C_r} & V'_{\lambda\phi C_\lambda} & V'_{\lambda\phi C_\phi} & V'_{\lambda\phi b_r} & V'_{\lambda\phi b_\lambda} & V'_{\lambda\phi b_\phi} \\ V'_{\phi rr} & V'_{\phi r\lambda} & V'_{\phi r\phi} & V'_{\phi rC_r} & V'_{\phi rC_\lambda} & V'_{\phi rC_\phi} & V'_{\phi rb_r} & V'_{\phi rb_\lambda} & V'_{\phi rb_\phi} \\ V'_{\phi\lambda r} & V'_{\phi\lambda\lambda} & V'_{\phi\lambda\phi} & V'_{\phi\lambda C_r} & V'_{\phi\lambda C_\lambda} & V'_{\phi\lambda C_\phi} & V'_{\phi\lambda b_r} & V'_{\phi\lambda b_\lambda} & V'_{\phi\lambda b_\phi} \\ V'_{\phi\phi r} & V'_{\phi\phi\lambda} & V'_{\phi\phi\phi} & V'_{\phi\phi C_r} & V'_{\phi\phi C_\lambda} & V'_{\phi\phi C_\phi} & V'_{\phi\phi b_r} & V'_{\phi\phi b_\lambda} & V'_{\phi\phi b_\phi} \end{bmatrix} \\ &= \begin{bmatrix} C_r V_{rrr} & C_r V_{rr\lambda} & C_r V_{rr\phi} & V_{rr} & 0 & 0 & -\frac{l_r}{||\vec{l}||^2} & 0 & 0 \\ C_r V_{rr\lambda} & C_r V_{r\lambda\lambda} & C_r V_{r\lambda\phi} & V_{r\lambda} & 0 & 0 & -\frac{l_\lambda}{||\vec{l}||^2} & 0 & 0 \\ C_r V_{rr\phi} & C_r V_{r\lambda\phi} & C_r V_{r\phi\phi} & V_{r\phi} & 0 & 0 & -\frac{l_\phi}{||\vec{l}||^2} & 0 & 0 \\ C_\lambda V_{rr\lambda} & C_\lambda V_{r\lambda\lambda} & C_\lambda V_{r\lambda\phi} & 0 & V_{r\lambda} & 0 & 0 & -\frac{l_r}{||\vec{l}||^2} & 0 \\ C_\lambda V_{r\lambda\lambda} & C_\lambda V_{\lambda\lambda\lambda} & C_\lambda V_{\lambda\lambda\phi} & 0 & V_{\lambda\lambda} & 0 & 0 & -\frac{l_\lambda}{||\vec{l}||^2} & 0 \\ C_\lambda V_{r\lambda\phi} & C_\lambda V_{\lambda\lambda\phi} & C_\lambda V_{\lambda\phi\phi} & 0 & V_{\lambda\phi} & 0 & 0 & -\frac{l_\phi}{||\vec{l}||^2} & 0 \\ C_\phi V_{rr\phi} & C_\phi V_{r\lambda\phi} & C_\phi V_{r\phi\phi} & 0 & 0 & V_{r\phi} & 0 & 0 & -\frac{l_r}{||\vec{l}||^2} \\ C_\phi V_{r\lambda\phi} & C_\phi V_{\lambda\lambda\phi} & C_\phi V_{\lambda\phi\phi} & 0 & 0 & V_{\lambda\phi} & 0 & 0 & -\frac{l_\lambda}{||\vec{l}||^2} \\ C_\phi V_{r\phi\phi} & C_\phi V_{\lambda\phi\phi} & C_\phi V_{\phi\phi\phi} & 0 & 0 & V_{\phi\phi} & 0 & 0 & -\frac{l_\phi}{||\vec{l}||^2} \end{bmatrix} \end{aligned} \tag{2.43}$$

$\vec{V}'_{uu}$ and design matrix $\boldsymbol{A}'$ are computed with the current estimated state $\vec{y}'_i$.

2.3.3. Scaling and Dampening

The state λ and ϕ are at a scale of ≈ 0 to 2π. The state r is at a scale factor of $\approx 10^6$ to 10^7. This can be a problem when computing least squares because a change in V_{rr} is not in the same scale as a change in $V_{r\lambda}$ or $V_{\phi\phi}$. As a result, a very large change in the state $\vec{y}$ on the first least-squares estimation of each epoch leads to an overshoot in the result. This happens specially for the parameter r.

To partially mitigate this problem, the weighed least squares method is used. It makes use of the weighed matrix $\boldsymbol{W}$, which is a diagonal matrix where each term is $\frac{1}{w_{ij}}$, where w_{ij} normalises the corresponding $\vec{V}'_{uu}$ values. i and j are the corresponding rows and columns respectively.

To prevent the least-squares method from spiking in the first iteration of each epoch, the Levenberg–Marquardt damping method is used. Gavin, H. P. [12] shows that a diagonal matrix with all its terms being the dampening factor τ can prevent these spikes. Both of these methods are done within the least squares equation in such way that:

$$\vec{y}_{i+1} = (\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A} - \tau\boldsymbol{I})^{-1}\boldsymbol{A}^T\boldsymbol{W}(\vec{V}_{uu}^{(measured)} - \vec{V}_{uu}(\vec{y}_i)) + \vec{y}_i\,, \tag{2.44}$$

where i is the iteration number within each epoch. To let the subsequent iterations calculate a change large enough to converge, the default dampening factor is attenuated with each iteration by $\tau = (1 \cdot 10^{-7})^i$.

Figure 2.5 shows how the iterative process is carried out in the algorithm. The initial state $\vec{y}'_0$ is either the initially guessed state (if it's the first epoch being calculated) or final estimation of the state on the previous epoch. With the current state, the gravity gradient tensor $\boldsymbol{V}_{uu}$ and its derivative $\boldsymbol{V}_{uuu}$ are calculated as shown in Section 2.2. With these, they are converted into the gravity gradient vector $\vec{V}$'$_{uu}$ and the design matrix $\boldsymbol{A}$' are formulated by including the calibration parameter corrections as seen in Section 2.3.2. From the measurement data, the gravity gradient vector $\vec{V}_{uu}^{(measured)}$ is extracted. With this, the least squares method from Figure 2.5 is used to compute $d\vec{y}_i$. This is the process for an iteration, in which $d\vec{y}_i$ is added to $\vec{y}_i$ to form $\vec{y_{i+1}}$, which shall serve as the state for computing the next iteration. The state computed at the end of the final iteration, is the final estimation for the epoch. After which, the iteration number i is reset to 0, and the iterative process restarts-

2.4. Iterative Process

There are two criteria to stop the iterative process. One option is to set an convergence threshold which has to be crossed. The change in state $d\vec{y}$ must be equal or inferior to this limit set. This criteria evaluates the speed of the process i.e., how many iterations it takes to converge. The second criteria is to set a finite amount of iterations. This can be especially used for missions with a limited time and thus number of iterations to compute position, or for testing, where the real point is known and the final error for each coordinate compared to this point can be computed. This criteria evaluates how potentially accurate the process can end up being regardless of the number of iterations, while arriving at an estimated point.

The following two sections will review each of the types of criterion.

1. **Error Threshold**: For the error threshold, a limit to how much of a deviation from the objective point is set. To be consistent, the error is set to be in the form a of a distance (in $[m]$). The results, however, are in spherical co-ordinates. This means that for the angular co-ordinates λ and ϕ, the error has to be converted into radians. For the radius r this is not needed. The radius threshold error is set as $\epsilon_{\text{thr}_\text{r}}$, it is implemented as

$$\epsilon_{\text{thr}_\lambda} = \epsilon_{\text{thr}_\phi} = \arcsin\left(\frac{\epsilon_{\text{thr}_\text{r}}}{r}\right) \tag{2.45}$$

   where $\epsilon_{\text{thr}_\lambda}$ and $\epsilon_{\text{thr}_\phi}$ correspond to λ and ϕ respectively. As a result, the higher the altitude of the satellite, the higher the accuracy is needed in terms of spherical co-ordinate angles.
2. **Finite Iterations**: A set amount of iterations is assigned to the process. Upon reaching this number, the iterative loop stops. This setting is not efficient at all for practical processes because it is most likely that for most epochs, there will be insufficient iterations. This approach is used for analytical and testing purposes. It can truncate iterations to only 1 in order to analyse how initialisation and details within a single iteration works. It can truncate iterations to beyond the usual needed to reach the error threshold to evaluate if solutions converge and to evaluate the final error.

Once the iterative process is finished, the next epoch of the satellite's orbit will be processed. An important note is that all the final estimations of the elements in y are kept as the initial guess for the next epoch. There are two reason for doing this.

- Systematic errors such as b and K are almost static and do not change too much during one orbital revolution.
- Position r, λ and ϕ change every epoch, as well as non-gravitational accelerations $\vec{d}$. Nevertheless, It is much more efficient to take the previous epoch as an initial guess, rather than to start the process from scratch every single epoch. By taking the last epoch's final guess, there is a potential of having a better performance when more epochs are computed, specially at last for the initial ones.

A diagram of how this iterative process can be observed in Figure 2.5

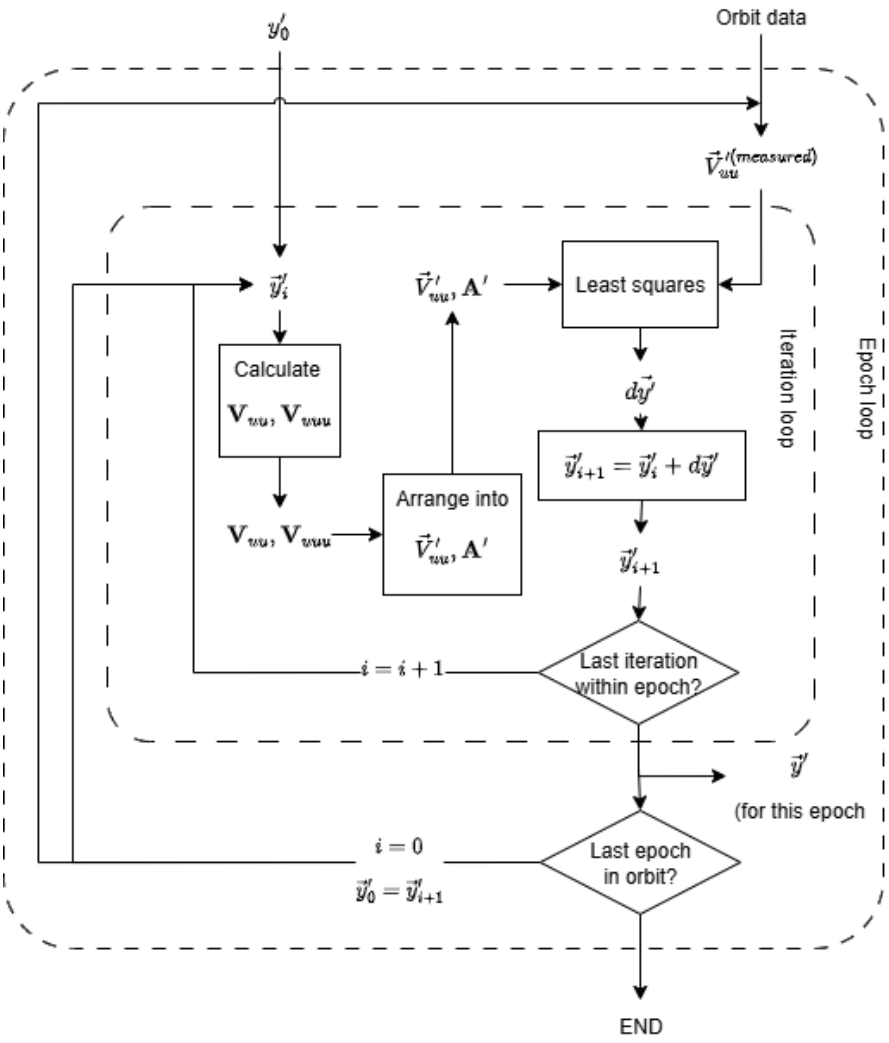

Figure 2.5: Diagram of how the least squares iterative process works, where the initial state $\vec{y}'_0$ and the gradiometer measurements $\vec{V}'_{uu}$ are the inputs and the estimated state $\vec{y}'$ is the output for each orbital epoch.

2.5. Programming

Two programming languages are used in this thesis. Python is an easy language to program and its large amount of modules and libraries. The main problem is that Python is a very high level programming language, and very slow. For processes which need a substantial amount of computational power, it is not effective. To adress this, Rust is used because it is extremely fast and can be used efficiently.

- **Python**: Used for more specific functions which can be done outside of the main process:
  - Graphs for results and main display.
  - Extracting data from GOCE gradiometer data files.
  - Generating data for the Lunar orbit simulation in Chapter 6.
- **Rust**: It is mainly used for the bulk of the computations:
  - Reading the files containing Stokes coefficients from gravity field models.

- Forming all associated Legendre functions in a symbolic state, as shown in Equation 2.8 and Figure 2.3. This process is one of the biggest notable differences with Python. In Rust, forming the functions takes about $300[s]$, while in Python, it takes more than $72[hr]$, especially when symbolic differentiation's need to be done.
- Preforming the least squares iterations shown in Figure 2.5, which is another part in which Rust shines over python. At D/O 100, 20604 computations of V_{pq}^{uu} and V_{pq}^{uuu} need to be done in each iteration. Speed and efficiency are needed for this, since Python computes a maximum D/O of 100 in $15[hr]$, while Rust does this same process within $10[s]$

The details on how the software is installed can be seen in Appendix B.

2.6. Simulation

The Lunar orbit simulation is generated in Tudat. Tudat is an astrodynamics tool and library provided by TU Delft which can be used in C++ and Python. in this case, Python is used along with this tool to generate all the data needed within the orbit to obtain $\vec{V}_{uu}^{\prime(measured)}$. The accuracy of the simulation is not what is being tested in this thesis. The most important points to take from the simulation is the position data needed to obtain $\vec{V}_{uu}^{\prime(measured)}$. Bias and scale coefficient which are afterwards added to it, including white noise too. The least squares estimation does not interact with the actual position of the satellite, and is only used to measure the accuracy of the prediction itself. The biases and the scale coefficient are not known to the least squares algorithm either and are used only to construct $\vec{V}_{uu}^{\prime(measured)}$. Figure 2.6 shows the version of Figure 2.5 that is used to run the algorithm with the simulated data

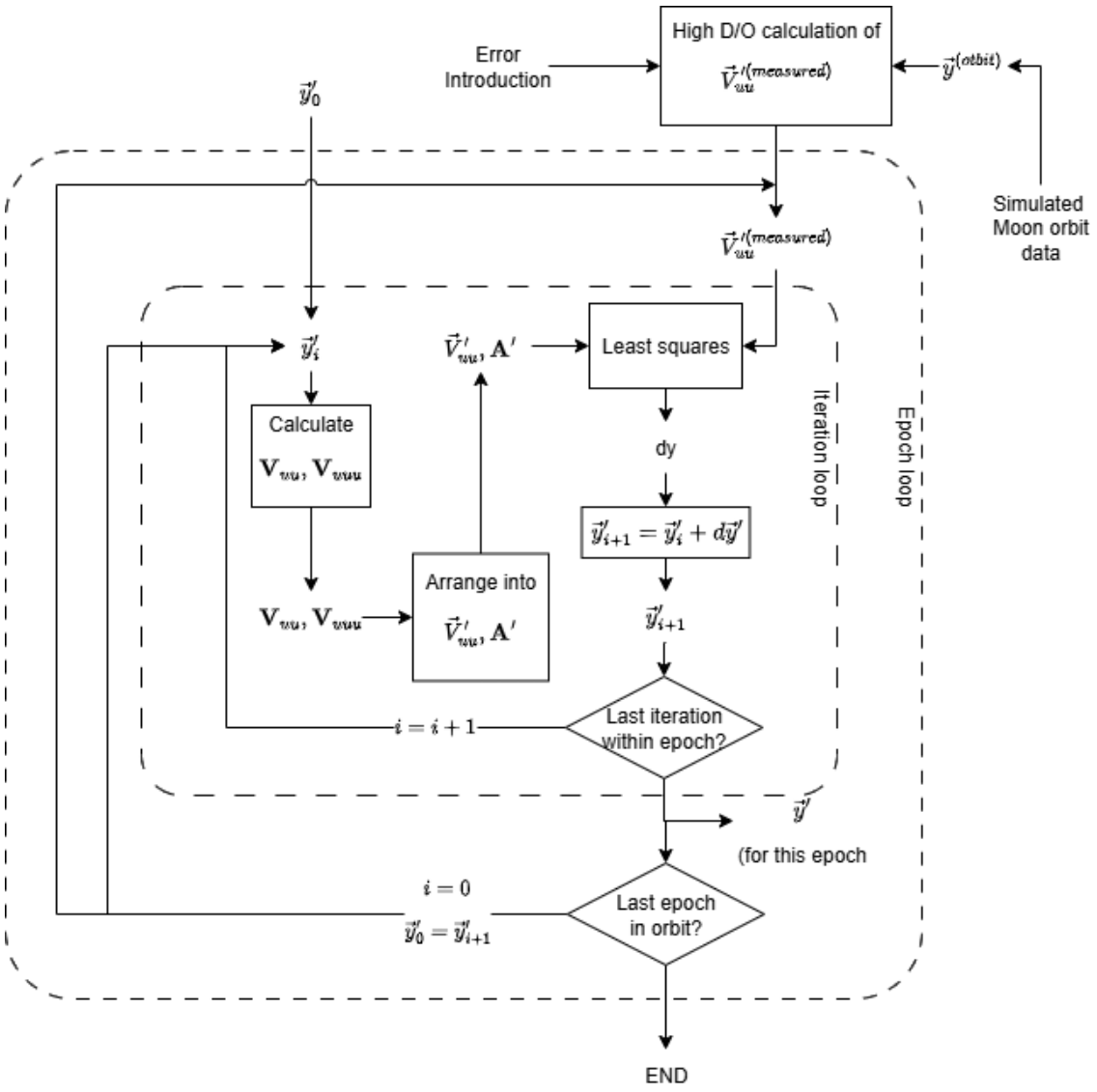

Figure 2.6: Diagram of how the least squares iterative process works for the Moon orbit simulation data.

3

Sensitivity Analysis

A sensitivity analysis intends to evaluate the least squares computational method. While the results of this thesis are an analysis of similar kind, this section is dedicated to a sensitivity analysis on initial deviation, orbital altitude, degree and order to assess how they affect the results and efficiency of the least squares algorithm

3.1. Initial Deviation

An undeviated initial position $\vec{y}_{obj}$ is set. From this, a deviated initial guess is assumed to observe the amount of iterations it takes to approach the undeviated position. If the process is done correctly, there should be a pattern in which a larger deviation correlates with a larger amount of iterations to converge, since there is more distance to correct from at higher initial deviation. This test uses the uncorrected form of the least squares method to have a better perspective on how initial deviation affects the algorithm without error correction being made. To generate deviations, the scalar (in $[m]/[km]$) forms are obtained for all axis. After this, the deviations corresponding to the latitude λ and the longitude ϕ are converted with the use of Equation 2.45 into angular deviations. The deviations are randomly generated in a range between $0.577[km]$ and $300[km]$ for each axis. The lower limit is set so that the total combined minimum deviation is of $1[km]$. The higher limit is set in this particular number so that there are enough correctly converged results to find a pattern. Higher values for this limit generate more values which converge away from the undeviated point.

This test is carried out in a lunar environment, with an undeviated position of $r = 3000000[m], \lambda = 1.1[rad], \phi = 0.1[rad]$. These are arbitrarily chosen values, as this test is only focused on deviation. If other angular GSF coordinates are used, they yield a similar pattern nevertheless. It makes use of a maximum D/O of 100 to obtain high accuracy without needing a long computation time. The algorithm is undampened to not constrain the progress of any iteration towards the converged value. It is assumed that the convergence occurs when $d\vec{y}$ is less than the threshold, which is set to $1[km]$ (or 10^{-6} times the undeviated position). A total of 30 different deviations are generated, with a random seed of 42 in the random number generator.

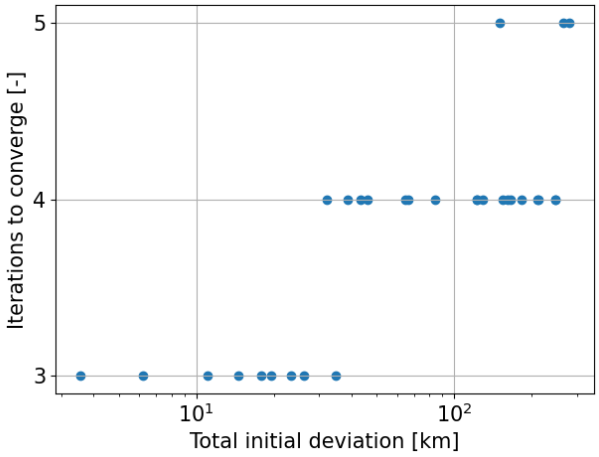

Figure 3.1: Amount of uncorrected least squares iterations for different initial deviations at a threshold of 10^{-6}.

Figure 3.1 shows an overall pattern in which the number of iterations to converge increases as the initial deviation increases, thus displaying positive correlation between an increase in iterations to converge and the initial deviation. Positive correlation is proof of a working algorithm, because the least squares takes longer to converge with a larger initial deviation. These are not all 30 results, because 4 of the randomly generated results converge into incorrect points. This is due to the deviation being too large. The deviations of these 4 mentioned points are seen in the following table:

Total Deviation $[m]$	r Deviation $[m]$	λ Deviation $[deg]$	Φ Deviation $[deg]$
264910	-3974.62	$1.17208 \cdot 10^{-1}$	$\mathbf{5.06409}$
211437	$\mathbf{-211285}$	$-1.10562 \cdot 10^{-1}$	$1.05496 \cdot 10^{-1}$
280504	$\mathbf{-280395}$	$-2.33517 \cdot 10^{-2}$	$1.47028 \cdot 10^{-1}$
246238	620.556	$\mathbf{4.69575}$	$3.39922 \cdot 10^{-1}$

Table 3.1: Initial deviations that lead to converging into a different point, with the bold values being the main contributor.

By observing the total combined deviations in Table 3.1, and comparing them to the ones shown in Figure 3.1, it can be deduced that total deviation is not the cause of the incorrect convergences. Instead, it is because of the individual axis deviations. Table 3.1 highlights the axes that are higher than any of the correctly converged points. The angular main contributors seen in this table are equivalent to a deviation of $264810[m]$ for ϕ in the first row of the table and $245594[m]$ for λ in the 4th row of the table. Least squares resolves navigation with each axis values separate, and all of them contributing to $d\vec{y}$. A deviation of at least $211.285[km]$ in any axis leads to convergence into points which are too significantly deviated from the actual orbit to be considered a valid estimate. This is additional insightful knowledge of the limitations of the algorithm presented in this thesis.

3.2. Orbit Altitude Test

Spherical harmonic models offer a detailed map of the gravity field based on the Stokes coefficients. The shape of the gravity field is formed by using associated Legendre functions. The level of detail of this shape is based on the level of D/O that is used, but it is

not the only variable. The orbital altitude affects greatly the intensity of gravity gradient, as seen in Section A.1.3. At higher altitudes, the gravity potential, acceleration and gradient become less intense, thus the gravity field map (and thus gravity gradient maps) become less detailed.

In this test, an arbitrary initial estimate deviated state of $1 \cdot 10^4[m]$ is set away from the real position state. The real position state is what the estimated state is compared to. When testing the effects of orbital altitude, instrument measurement sensitivity needs to be taken into account too. Spero, R. [13] indicates that the signal-to-noise ratio of a gradiometer is:

$$SNR = \frac{V_{uu}}{n_{uu}}, \tag{3.1}$$

where V_{uu} is a component of GGT to be measured by the gradiometer in GSF, and n_{uu} is the corresponding noise to the component. In gravity gradient equations, the $\frac{d^2}{dr^2}$ component is proportional to altitude by $\frac{1}{r^3}$, the $\frac{d^2}{drd\lambda}$, $\frac{d^2}{drd\phi}$ components are proportional by $\frac{1}{r^2}$, and the rest of the components are proportional by $\frac{1}{r}$, hence it can be predicted that there should be a larger deviation of the estimated state at higher altitudes, since the signal to noise ratio decreases as much.

A noise of $0.1[E] = 1 \cdot 10^{-10}[1/s^2]$ is added to the elements of $\vec{V}_{uu}^{(measured)}$. It is assumed that for this test, the noise remains constant instead of random to act as a constant variable, since the independent variable is orbital radius. An arbitrary maximum D/O of 50 is taken. The iterative process is set to stop after a convergence threshold (mentioned in Section 2.4) of $2[m]$. In this test, the orbital radius r is changed to represent a change in the orbital altitude. This same test is also conducted with an increased initial deviation. Table 3.2 shows the outcome of this test.

Initial r $[m]$	Initial Deviation $[m]$	Error $[m]$	Iterations to converge $[-]$
$2 \cdot 10^6$	$1 \cdot 10^3$	$3.72529 \cdot 10^{-8}$	3
	$1 \cdot 10^4$	$3.74857 \cdot 10^{-8}$	4
$2 \cdot 10^7$	$1 \cdot 10^3$	$9.27210 \cdot 10^{-3}$	2
	$1 \cdot 10^4$	$9.27210 \cdot 10^{-3}$	3
	$1 \cdot 10^5$	$9.27211 \cdot 10^{-3}$	4
$2 \cdot 10^8$	$1 \cdot 10^3$	$6.05539 \cdot 10^{1}$	2
	$1 \cdot 10^4$	$6.05539 \cdot 10^{1}$	3
	$1 \cdot 10^5$	$6.05538 \cdot 10^{1}$	3
$2 \cdot 10^9$	$1 \cdot 10^3$	$5.74359 \cdot 10^{5}$	3
	$1 \cdot 10^4$	$5.74359 \cdot 10^{5}$	3
	$1 \cdot 10^5$	$5.74359 \cdot 10^{5}$	3
$2 \cdot 10^{10}$	$1 \cdot 10^3$	$4.34124 \cdot 10^{9}$	20

Table 3.2: Least squares accuracy difference affected by the orbital altitude.

The results show 3 ways in which altitude affects the results of the least squares algorithm:

- **Converged error**: As predicted, the error shown in Table 3.2 indicates that a low orbital altitude of $2 \cdot 10^6[m]$ shows an error in the scale of $10^{-8}[m] = 5 \cdot 10^{-13}$%, while

a high orbital altitude of $2 \cdot 10^{10}[m]$ shows an error in the scale of $10^{9}[m] = 5\%$. The pattern seen is that the higher the orbital altitude, the higher the error of the converged state.

- **Convergence speed**: By having the same maximum D/O on all tests conducted in this section, each iteration takes a similar amount of time, as demonstrated in Section 3.3. Convergence speed is instead measured by the number of iterations to converge. Table 3.2 shows that higher altitudes take less iterations to converge. This is explained by the fact that at higher altitudes, the map of the gravity gradients loses detail and becomes smoother. The lack of detail facilitates a quicker route towards the estimated state. More importantly, this is also because of the scale of the initial deviation compared to the orbital altitude. In higher altitudes, the same initial deviation is proportionally smaller.
- **Maximum converging deviation**: When the error becomes large enough, convergence becomes more unstable. Table 3.2 shows that at $r = 2 \cdot 10^{9}[m]$, it needs more iterations to converge than previous lower altitudes. At $r = 2 \cdot 10^{10}[m]$, the number of iterations to converge spikes up to 20. This can be explained buy the fact that at an altitude high enough, the error becomes larger than the initial deviation to correct, thus the convergence having to cover more distance.

3.3. Computation Performance test

When evaluating the least squares algorithm, the performance and efficiency has to be measured. If the computation time does not follow the predicted pattern, the algorithm does not process the calculations accordingly.

The majority of the computational cost is attributed to the evaluation of the associated Legendre functions. Each evaluation corresponds to substituting a specific value into the variable x. In the context of this test, a computation is the calculation of a D/O combination, while an evaluation is the substitution of a value into x. From a mathematical perspective, higher D/O require significantly more computations than lower ones. In this context, an ”equation computation” refers to the complete evaluation of the Legendre function for a specific D/O. The initial estimate of the total number of such computations performed during a single iteration of the least squares process is:

$$N_{\text{comp}} = \frac{1}{2} p_{\text{max}}^2 = \sum_{p=0}^{p_{\text{max}}} \sum_{q=0}^{p} 1, \tag{3.2}$$

where N_{max} is the number of computations, and p_{max} is the maximum degree. This is the equivalent to the amount of Stokes coefficients, or associated Legendre equations being used. The number of computations is not the only contributor to the computation time. An evaluation is every time a value is substituted into a symbolic Associated Legendre function to get the result for a given point. To really know the performance in a more detailed manner, the number of evaluations must be known. The number of evaluations that have to be done within each iteration of the least squares method is:

$$T_{\text{iter}} \propto N_{\text{eval}} = \sum_{p=0}^{p_{\text{max}}} \sum_{q=0}^{p} \frac{1}{2}(p + \frac{1}{2}) \tag{3.3}$$

When looking at the Equation 3.3, the number of evaluations is an approximate average, since the amount of evaluations does not scale gradually, but in jumps every 2 degrees. The table below shows this detail.

Degree p [-]	Legendre Polynomials	N_{eval} [-]	N_{eval} w.r.t. degree p
0	1	0	$\frac{1}{2}p$
1	x	1	$\frac{1}{2}(p+1)$
2	$\frac{1}{2}(3x^2-1)$	1	$\frac{1}{2}p$
3	$\frac{1}{2}(5x^3-3x)$	2	$\frac{1}{2}(p+1)$
4	$\frac{1}{8}(35x^4-30x^2+3)$	2	$\frac{1}{2}p$
5	$\frac{1}{8}(63x^5-70x^3+15x)$	3	$\frac{1}{2}(p+1)$
6	$\frac{1}{16}(231x^6-315x^4+105x^2-5)$	3	$\frac{1}{2}p$

Table 3.3: Proof of the number of evaluations needed for each Legendre Polynomial.

In Table 3.3, the $\frac{1}{2}$ represents that Legendre polynomials make use of of 1 out of every 2 powers of x in $P_p(x)$. The most important data is N_{eval} with respect to degree p, which formulates how many evaluations there are in a certain degree. The table shows that they alternate in a 1:1 ratio between $\frac{1}{2}p$ and $\frac{1}{2}(p+1)$. The average between the two can be made to be $\frac{1}{2}(l+\frac{1}{2})$. The numbers seen in the table are not the only Legendre polynomials being computed during the iterations, but additionally there are the 1st, 2nd and 3rd degree derivatives: $P'_{pm}(x)$, $P''_{pm}(x)$, $P'''_{pm}(x)$ respectively. For these, their Legendre polynomials are different in terms of number of evaluations, as shown in Table 3.4 for the 1st derivatives.

Degree p [-]	Legendre Polynomials	N_{eval} [-]	N_{eval} w.r.t. degree p
0	0	0	-
1	1	0	$\frac{1}{2}(p-1)$
2	$3x$	1	$\frac{1}{2}p$
3	$\frac{1}{2}(15x^2-3)$	1	$\frac{1}{2}(p-1)$
4	$\frac{1}{8}(140x^3-60x)$	2	$\frac{1}{2}p$
5	$\frac{1}{8}(315x^4-210x^2+15)$	2	$\frac{1}{2}(p-1)$
6	$\frac{1}{16}(1386x^5-1260x^3+210x)$	3	$\frac{1}{2}p$

Table 3.4: Proof of the number of evaluations needed for each 1st derivative of Legendre Polynomial.

Table 3.4 shows that this data now alternated between $\frac{1}{2}(p-1)$ and $\frac{1}{2}p$. This makes the average $\frac{1}{2}(p-\frac{1}{2})$ for the amount of evaluations per degree in the first derivative of the Legendre polynomials. When calculating a low maximum D/O of 30, the amount of computations is 1984. The amount of evaluations in the Legendre polynomial are 0 for degree 0, thus making it more negligible towards the pattern as maximum D/O increases.

Degree p [-]	Legendre Polynomials	N_{eval} [-]	N_{eval} w.r.t. degree l
0	0	0	-
1	0	0	-
2	3	0	$\frac{1}{2}(p-2)$
3	$15x$	1	$\frac{1}{2}(p-1)$
4	$\frac{1}{8}(420x^2-60)$	1	$\frac{1}{2}(p-2)$
5	$\frac{1}{8}(1260x^3-420x)$	2	$\frac{1}{2}(p-1)$
6	$\frac{1}{16}(6930x^4-3780x^2+210)$	2	$\frac{1}{2}(p-2)$

Table 3.5: Proof of the number of evaluations needed for each 2nd derivative of Legendre Polynomial.

Table 3.5 shows the data for the 2nd derivative. where it is also assumed that the degree 1 computations (2 out of 1984 at max D/O f 30) are also negligible towards the global average. The average is between $\frac{1}{2}(p-2)$ and $\frac{1}{2}(p-1)$, making it $\frac{1}{2}(p-\frac{3}{2})$.

Degree p [-]	Legendre Polynomials	N_{eval} [-]	N_{eval} w.r.t. degree p
0	0	0	-
1	0	0	-
2	0	0	-
3	15	0	$\frac{1}{2}(p-3)$
4	$105x$	1	$\frac{1}{2}(p-2)$
5	$\frac{1}{8}(3780x^2-420)$	1	$\frac{1}{2}(p-3)$
6	$\frac{1}{16}(27720x^3-7560x)$	2	$\frac{1}{2}(p-2)$

Table 3.6: Proof of the number of evaluations needed for each 3rd derivative of Legendre Polynomial.

Table 3.6 shows the data for the 3rd derivative, where it is assumed that the degree 2 computations (3 out o 1984 at max D/O of 30) are also negligible towards the global average. The average for the 3rd derivative is $\frac{1}{2}(p-\frac{5}{2})$. With this, the total amount of iterations for computing al the associated Legendre functions is derived with:

$$N_{\text{eval}} = \sum_{p=0}^{p_{\text{max}}}\sum_{q=0}^{p}\frac{1}{2}(p+\frac{1}{2}) + \sum_{p=1}^{p_{\text{max}}}\sum_{q=0}^{p}\frac{1}{2}(p-\frac{1}{2}) + \sum_{p=2}^{p_{\text{max}}}\sum_{q=0}^{p}\frac{1}{2}(p-\frac{3}{2}) + \sum_{p=3}^{p_{\text{max}}}\sum_{q=0}^{p}\frac{1}{2}(p-\frac{5}{2}) \quad (3.4)$$

where N is not the duration, but instead the number of iterations related the max D/O.

The algorithm is executed with models with different maximum D/O to get a good idea of how the relation of N correlates with actual computation time. Table 3.7 shows the results for computation time.

Model	Maximum D/O	$N^{(eval)}$	Time for iterations $[s]$
GOSG02S (GOCE)	300	$1.80895 \cdot 10^7$	276.80
GGM03S (Grace)	180	$3.92011 \cdot 10^6$	56.252
ITG_Champ01K (Champ)	70	$2.33463 \cdot 10^5$	3.6162
GEM9	30	$1.8863 \cdot 10^4$	0.38927

Table 3.7: Computation time taken for a single iteration for different gravity field models.

Table 3.7 shows the an increase in the form of $T^{(comp)} = (2 \cdot 10^{-5})D/O^{2.8509}$, where $T^{(comp)}$ is the computation time. Due to the significant increase in computation time, more efficient testing can be done by truncating high D/O models.

4

Verification

The contents of the algorithm are verified in this chapter. Due to the sensitivity of the results depending on the correct formulation of associated Legendre functions, unit testing is performed on them too.

4.1. Unit Testing

Unit testing for associated Legendre functions is done to ensure that their formulation works adequately.

Associated Legendre functions all converge into a common value for $P_{pq}(1)$. Figure 4.1 shows the pattern of various associated Legendre Functions at order $q = 0$.The pattern indicates that all associated Legendre functions at order 0 (Legendre polynomials) converge to 1 when $x = 1$. Figure 4.2 shows that all associated Legendre functions of order 1 or above converge to 0 when $x = 1$. In this figure, the associated Legendre functions are noted as P_p^q, which is the way it is expressed in Equation 2.5, with the only notable differences being a negative sign in half of the computed associated Legendre functions, thus converging in 0 as well.

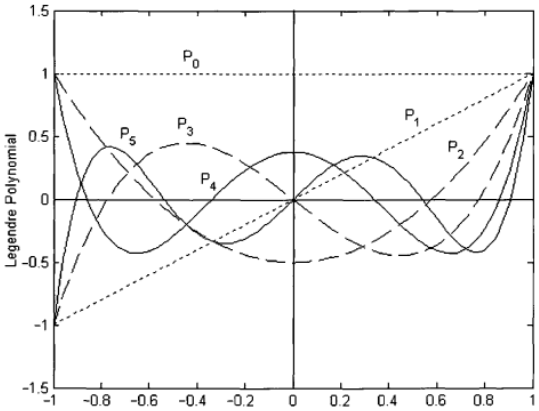

Figure 4.1: Legendre functions as a function of x at order $q = 0$. [14]

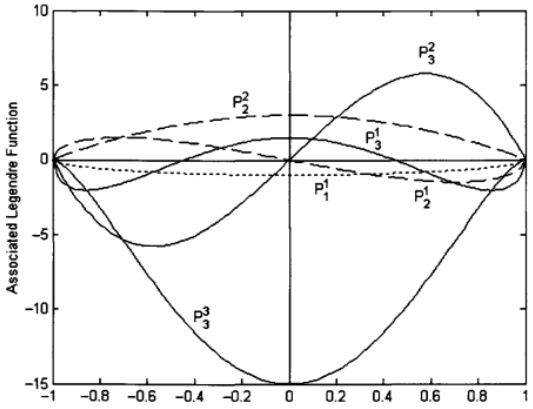

Figure 4.2: Legendre functions as a function of x at orders $q = 1, 2, 3$. [14]

To test if the associated Legendre function generation is computed correctly, a maximum

D/O of 200 of symbolically generated equations is set to perform an evaluation of their function at 1 instead of at sin (ϕ). Indeed, all functions generated yielded 1 for order 0 and 0 for any other orders.

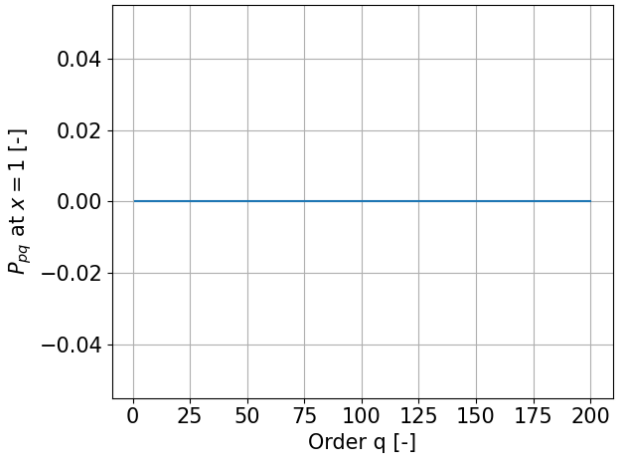

Figure 4.3: Value of the Associated Legendre functions evaluated at $x = 1$ for all orders q in degree $p = 200$.

4.2. Errors in Gravity Field Models

As stated in Section 1.3.2, the gravity field models are assumed to be up to date. In reality, Earth's gravity field constantly changes, even in small quantities. Additionally, as technology advances, the gravity field map details and accuracy increases. To demonstrate the effect this assumption has on the results of the least squares algorithm, 3 different gravity field models for Earth are compared:

- GOSG02S [6] is a 2023 gravity field model made from GOCE satellite data only, with no prior gravity field data used in this its creation. It uses data from GOCE's electrostatic gravity gradiometer (EGG), and high-low satellite-to-satellite tracker (hl-SST) (which makes use of GNSS). Normal equations from both are combined. Tikhonov regularisation is used for orders $q < 20$ in degrees $d <= 200$, and for all D/O with degrees $d > 200$.
- GO_CONS_GCF_2_TIM_R6 (TIM_R6) [15] is a 2019 model that makes use of the GOCE satellite data only. It also makes use of GOCE's EGG and hl-SST data by combining their normal equations. Kaula regularisation is a variation of Tikhonov regularization and it is used for high D/O $201 < p, q < 300$.
- GO_CONS_GCF_2_TIM_R6e (TIM_R6e) [16] is a an extension of the 2019 TIM_R6 model. In addition to the satellite data used by the previous model, it also uses terrestrial observations (TER) for angles approximating the poles. The hl-SST, SGG and TER normal equations are combined. It uses the same regularisation as the 2019 TIM_R6 model.

This test is conducted by setting an initial orbital position for the satellite. Instead of making use of $\vec{V}_{uu}^{(measured)}$ (as shown in Equation 2.44) as an external value to the algorithm, it is obtained from the same type of calculation used for $\vec{V}_{uu}$ seen in Section 2.2 and Section 2.3.2. $\vec{V}_{uu}$ is calculated with the Stokes coefficients from the "TIM_R6e" model, while $\vec{V}_{uu}^{(measured)}$

is calculated with the coefficients from the ”GOSG02S” model. Both are started from the same initial position $\vec{y}_0$. $\vec{V}_{uu}$ and $\vec{V}_{uu}^{(measured)}$ are applied to the least squares equation as:

$$\vec{y}_{i+1} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T(\vec{V}_{uu}^{(\text{GOSG02S})}(\vec{y}_0) - \vec{V}_{uu}^{(\text{TIM\_R6e})}(\vec{y}_i)) + \vec{y}_i \tag{4.1}$$

where the uncorrected design matrix $\boldsymbol{A}$ is computed with the ”GOSG02S” model. The difference between $\vec{V}_{uu}^{(\text{GOSG02S})}$ and $\vec{V}_{uu}^{(\text{TIM\_R6e})}$ is related to the difference between the two models. With each iteration, the position that the ”TIM_R6e ” model would estimate moves towards the position that the ”GOSG02S” model would estimate.

This results in deviations of $34.4919[m]$, $5.95340 \cdot 10^{-4}[deg]$, $6.81139 \cdot 10^{-4}[deg]$ for r, λ, ϕ respectively. The deviation is because of the different values of their Stokes coefficients. ”TIM_R6e” makes use of additional ground data and a different regularisation. Figure 4.4 shows a line representing the estimated position of $\vec{y}$ in the three GSF axes when compared to the initial position. This shows the deviation there is between the points estimated with different models.

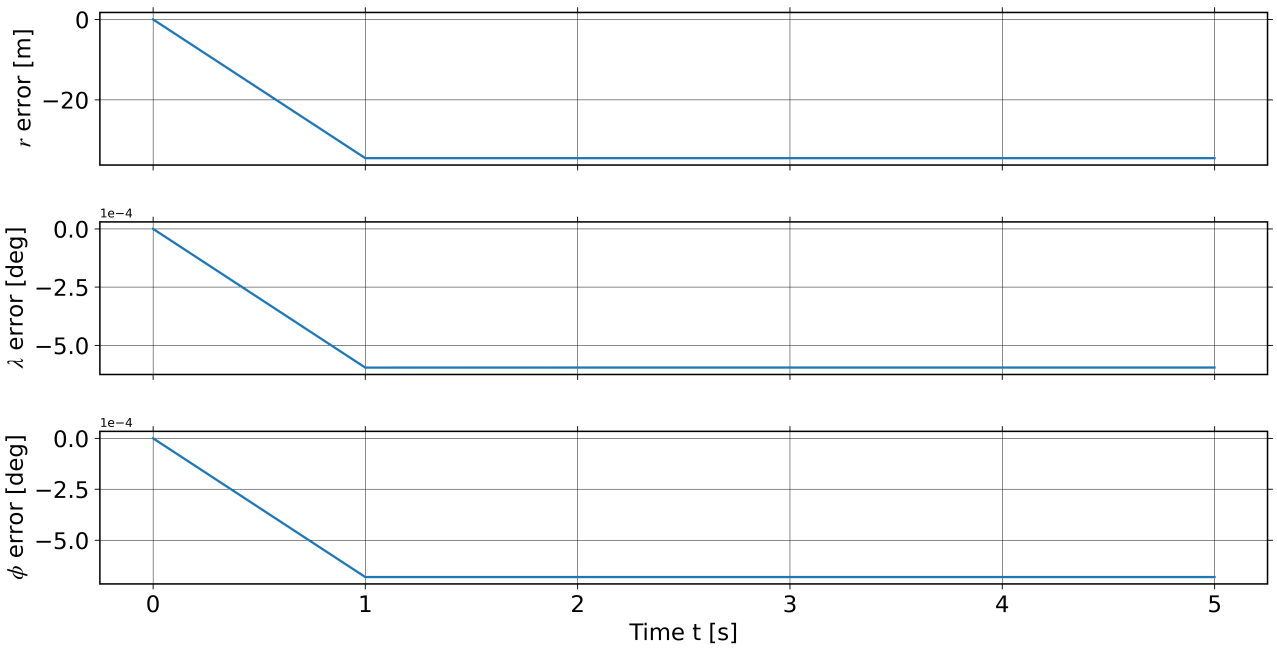

Figure 4.4: Deviation progression with each iteration of the least squares method with a comparison between the ”GOSG02S” model and the ”TIM_R6e” model.

A new test is conducted using the ”TIM_R6” model instead of its ”TIM_R6e” variant. The output for the estimate deviations are $79.5654[m]$, $8.28032 \cdot 10{-}4[deg]$, $7.90466 \cdot 10^{-4}[deg]$ for r, λ, ϕ respectively. These differences are because of the different in regularisation techniques used, as well as the D/O in which they are used each. This difference is larger than the results when using ”TIM_R6e”, even though ”TIM_R6e” makes use of terrestrial measurements that ”$GOSG02S$” does not use. Terrestrial data may provide additional detail that approximates ”TIM_R6e to ”$GOSG02S$” closer than ”TIM_R6”. Similarly to Figure 4.4, Figure 4.5 shows a line representing the deviation of the estimated state $\vec{y}$ compared to the initial guess through multiple iterations of the least squares algorithm in Equation 4.1.

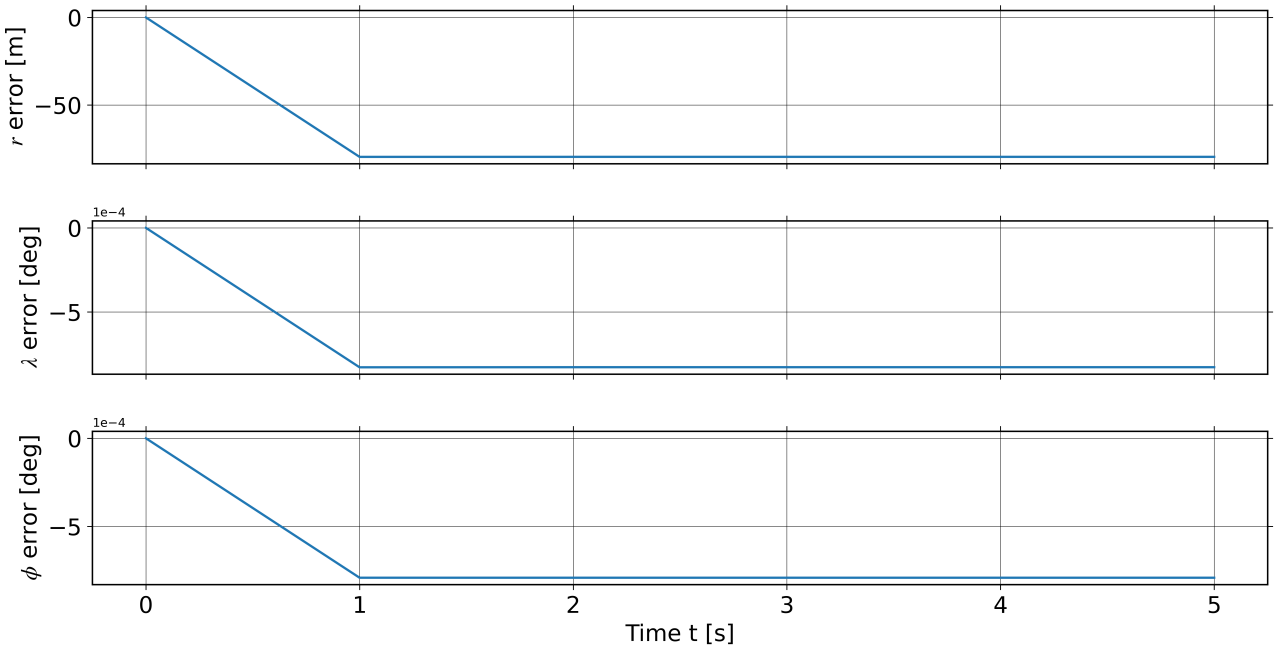

Figure 4.5: Deviation progression with each iteration of the least squares method with a comparison between the ”GOSG02S” model and the ”TIM_R6” model.

A final test is conducted, with ”TIM_R6e” used to compute $\vec{V}_{uu}^{(measured)}$ and ”TIM_R6” representing $\vec{V}_{uu}$. The deviations are $45.0862[m]$, $2.33100 \cdot 10^{-4}[deg]$, $1.09167 \cdot 10^{-4}[deg]$ for r, λ, ϕ respectively. The deviation is caused from the $R6e$ model making use of ground data on top of the $R6$ model and including it into the sum of their normal equations.

The comparison between the $R6$ and $R6e$ models is a result which verifies this process further; the difference between ”GOSG02S” and ”TIM_R6e”, and the difference between ”GOSG02S” and ”TIM_R6” differ to approximately the difference between ”TIM_R6e” and ”TIM_R6e”. A visual representation of how this works is shown in Figure 4.6.

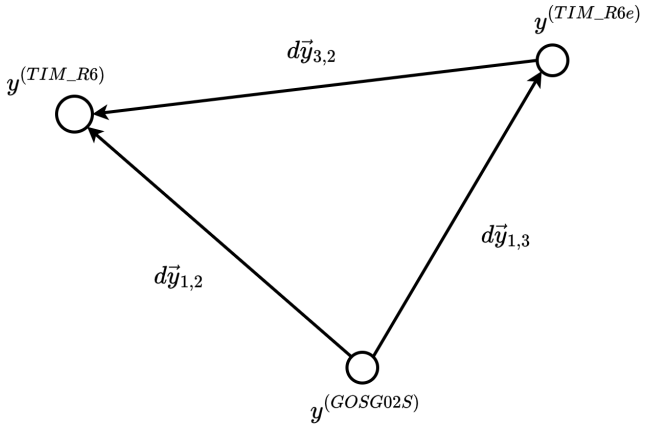

Figure 4.6: Visual representation of how $d\vec{y}$ are related in a vector space.

The difference in estimated states is:

$$d\vec{y}_{1,2} - d\vec{y}_{1,3} = d\vec{y}_{3,2}$$
$$\begin{bmatrix} 79.5654 \\ 8.28032 \cdot 10^{-4} \\ 7.90466 \cdot 10^{-4} \end{bmatrix} - \begin{bmatrix} 34.4919 \\ 5.95340 \cdot 10^{-4} \\ 6.81139 \cdot 10^{-4} \end{bmatrix} = \begin{bmatrix} 45.0735 \\ 2.32692 \cdot 10^{-4} \\ 1.09326 \cdot 10^{-4} \end{bmatrix} \approx \begin{bmatrix} 45.0862 \\ 2.33100 \cdot 10^{-4} \\ 1.09167 \cdot 10^{-4} \end{bmatrix}, \tag{4.2}$$

with the sub 1 representing "GOSG02S", the sub 2 representing "TIM_R6" and the sub 3 representing "TIM_R6e". The expression for $d\vec{y}$ is equivalent to a linear expression:

$$d\vec{y} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\Delta\vec{V}_{uu} \tag{4.3}$$

To prove linearity with the gravity field models, the gravity gradient vector differences $\Delta\vec{V}_{uu}$ are extracted, and are compared similarly.

$$\Delta\vec{V}_{uu_{1,2}} - \Delta\vec{V}_{uu_{1,3}} = \Delta\vec{V}_{uu_{3,2}}$$
$$\begin{bmatrix} -9.84927 \cdot 10^{-11} \\ 9.54011 \cdot 10^{-7} \\ -6.92491 \cdot 10^{-7} \\ 3.06139 \cdot 10^{-7} \\ 4.84468 \cdot 10^{-6} \\ -7.30911 \cdot 10^{-6} \end{bmatrix} - \begin{bmatrix} -4.27804 \cdot 10^{-11} \\ 7.67171 \cdot 10^{-7} \\ -1.63340 \cdot 10^{-7} \\ 1.61833 \cdot 10^{-7} \\ 1.83039 \cdot 10^{-6} \\ -3.02335 \cdot 10^{6} \end{bmatrix} = \begin{bmatrix} -5.57123 \cdot 10^{-11} \\ 1.86684 \cdot 10^{-7} \\ -5.29151 \cdot 10^{-7} \\ 1.44306 \cdot 10^{-7} \\ 3.01429 \cdot 10^{-6} \\ -4.28576 \cdot 10^{-6} \end{bmatrix} \approx \begin{bmatrix} -5.57270 \cdot 10^{-11} \\ 1.86763 \cdot 10^{-7} \\ -5.29438 \cdot 10^{-7} \\ 1.44954 \cdot 10^{-7} \\ 3.01624 \cdot 10^{-6} \\ -4.28865 \cdot 10^{-6} \end{bmatrix} \tag{4.4}$$

In Equation 4.4, it is observed how $\Delta\vec{V}_{uu}$ shows a linear pattern. The linearity is due to the 2 elements in the gravity gradient equations (as shown in Section A.1.3) that are taken from gravity field models are the Stokes coefficients C_{pq} and S_{pq}. All elements of the GGT show a linear proportionality to the Stokes coefficients, meaning that a change in the coefficients is a direct change in the value of the GGT, and subsequently $\Delta\vec{V}_{uu}$. The values displayed are verified with manual calculations.

One last point to highlight from the results in both Equation 4.4 and Equation 4.2, is that the differences (shown after the $=$ sign) are not exactly the value computed (shown $\approx$ sign). Instead, they contain a small deviation from the actual values that they should display. The deviation is explained by the fact that the models taken for this test are truncated at lower D/O than 300 (their maximum D/O) to efficiently run the algorithm.

5

Validation With GOCE Data.

Validation is essential to prove that this method can work in a practical sense. In this chapter, the least squares algorithm is tested using GOCE gradiometer data as the measured gravity gradient introduced into the process. The chapter is divided into a section which describes the process of retrieving and processing information from GOCE, and a section where the results of this test are analysed.

5.1. Gravity Gradient in Different Reference Frames

This section shows a difference between the frame in which the gradiometer data is given, and the frame in which the least squares algorithm works, and how to transform from one frame to the other. The gradiometer data is taken from GOCE level 2 data [17], which, unlike the level 1 data, is calibrated.

Data is given in two frames: the Gradiometer Reference Frame (GRF), which is the gradiometer's own local frame, and the Local North Oriented Frame (LNOF), which is a local frame to the satellite that aligns directionally with GSF, despite one being geocentric and the other one being local. GSF is formulated in spherical coordinates. The Earth Centred Reference Frame (EFRF) is the Cartesian form of GSF. The relation between EFRF, LNOF and GRF is shown in Figure 5.1.

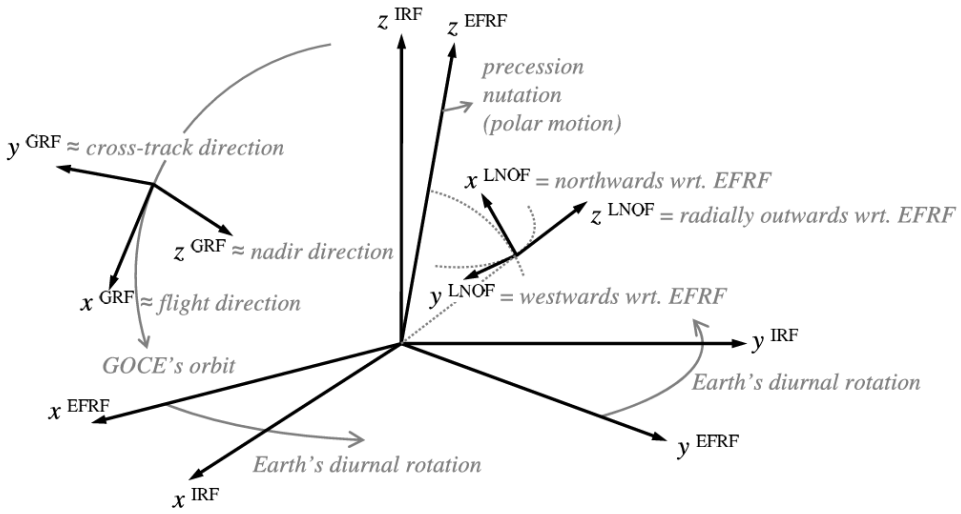

Figure 5.1: Graph to show the relation between the frames used by ESA by Th. Gruber, R. Rummel et al [18].

The angles that the LNOF alignment uses relates to the ECRF. r and $z^{(LNOF)}$ are in the same direction, ϕ and $x^{(LNOF)}$ are in the same direction, and λ and $y^{(LNOF)}$ are in opposite direction.

Despite the alignment, the results estimated by the code are in GSF. This means that to properly compare, they need to be transformed to LNOF. To know the relation between GSF and LNOF for the GGT, a transformation matrix is not enough. Tensors are the second differential with respect to each frame, and so there is a curve difference between both tensors which has to be calculated. To achieve this, the Christoffel symbols of the second kind are used. The fundamental formula for this is:

$$\Gamma_{ij}^{k} = \frac{1}{2} g^{kl} \left(\frac{\delta g_{lj}}{\delta u^{i}} + \frac{\delta g_{il}}{\delta u^{j}} - \frac{\delta g_{ij}}{\delta u^{l}} \right) \tag{5.1}$$

Appendix A shows all the related derivations. As calculated and confirmed by Reed, G. B. [19], LNOF gravity gradient relates to GSF as:

$$\begin{aligned}
V_{xx} &= \frac{1}{r^2} V_{\phi\phi} + \frac{1}{r} V_r \\
V_{yy} &= \frac{1}{r^2 \cos(\phi)^2} V_{\lambda\lambda} - \frac{\sin(\phi)}{r^2 \cos(\phi)} V_{\phi} + \frac{1}{r} V_r \\
V_{zz} &= V_{rr} \\
V_{xy} &= -\frac{1}{r^2 \cos(\phi)} V_{\lambda\phi} - \frac{\sin(\phi)}{r^2 \cos(\phi)^2} V_{\lambda} \\
V_{xz} &= \frac{1}{r} V_{r\phi} - \frac{1}{r^2} V_{\phi} \\
V_{yz} &= -\frac{1}{r \cos\phi} V_{r\lambda} + \frac{1}{r^2 \cos(\phi)} V_{\lambda}
\end{aligned} \quad , \tag{5.2}$$

where the entire derivation process can be found in **??**.

By applying Equation 5.2, the estimated values are in Cartesian form, meaning that they are linear instead of angular. With this, the Laplace second order partial differential equation can be applied as follows:

$$\frac{d^2V}{dx^2} + \frac{d^2V}{dy^y} + \frac{d^2V}{dz^2} = V_{xx} + V_{yy} + V_{zz} = 0 \tag{5.3}$$

Level 2 GOCE data is retrieved from the ESA dissemination service by the name of "EGG_TRF_2_" [20]. By using the Laplace second order partial differential equation results from the gradients retrieved from the data, in values in scales ranging between $10^{-11} - 10^{-10}[1/s^2]$, with an RMS of $1.57620 \cdot 10^{-11}[1/s^2]$. Figure 5.2 shows the range of results through 150 epochs of the GOCE level 2 gradiometer data. This scale is close to 0, but it is important to point out that it is not completely 0. By taking results displayed by Siemes, C. et al. [21], it can be seen that there is a discrepancy between gravity gradient elements on a scale of $10^{-10}[1/s^2]$ at low frequencies and $10^{-11}[1/s^2]$ at high frequencies. This is an indicator that there are errors in the level 2 data, even if these results have been calibrated and corrected. Measurement errors, transformation matrices and numerical errors contribute to this deviation from 0.

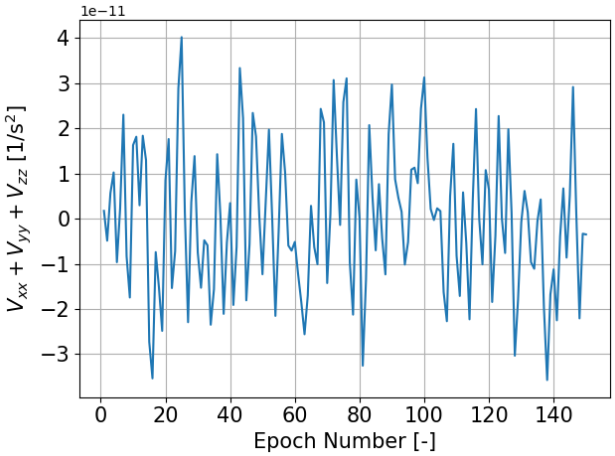

Figure 5.2: Result of the Laplace equation shown in Equation 5.3 when using level 2 GOCE data in LNOF for 150 epochs.

When performing the simulation for the analysis shown in Section 5.2, the GGT elements in each epoch can be changed into the LNOF Cartesian form with Equation 5.2. With these relations known, the gravity gradients provided by GOCE data can be transformed into GSF used in the estimation, shown in:

$$
\begin{aligned}
V_{rr} &= V_{zz} \\
V_{r\lambda} &= -r\cos(\phi)V_{yz} + \frac{1}{r}V_{\lambda} \\
V_{r\phi} &= rV_{xz} + \frac{1}{r}V_{\phi} \\
V_{\lambda\lambda} &= r^2\cos(\phi)^2V_{yy} + \sin(\phi)\cos(\phi)V_{\phi} - r\cos(\phi)^2V_r \\
V_{\lambda\phi} &= -r^2\cos(\phi)V_{xy} - \frac{\sin(\phi)}{\cos(\phi)}V_{\lambda} \\
V_{\phi\phi} &= r^2V_{xx} - rV_r
\end{aligned}
\tag{5.4}
$$

5.2. Analysis

GOCE's orbital position data is taken from the level 2 mentioned in Section 5.1, specifically the file corresponding to the orbit data between the 1st of February and the 12th of February, 2010, with the _0100 suffix. The data needed in the file is found under "$GG\_spatial\_record$", where the GPS time, the GNSS provided GOCE position and the gravity gradient elements are found. The orbital position data is already in GSF, so there is no need to transform it. With the gravity gradient $\vec{V}_{uu}^{(measured)}$ calculated from the transformation in Equation 5.4, the corrected least squares algorithm estimates the positions as shown in Figure 5.4.

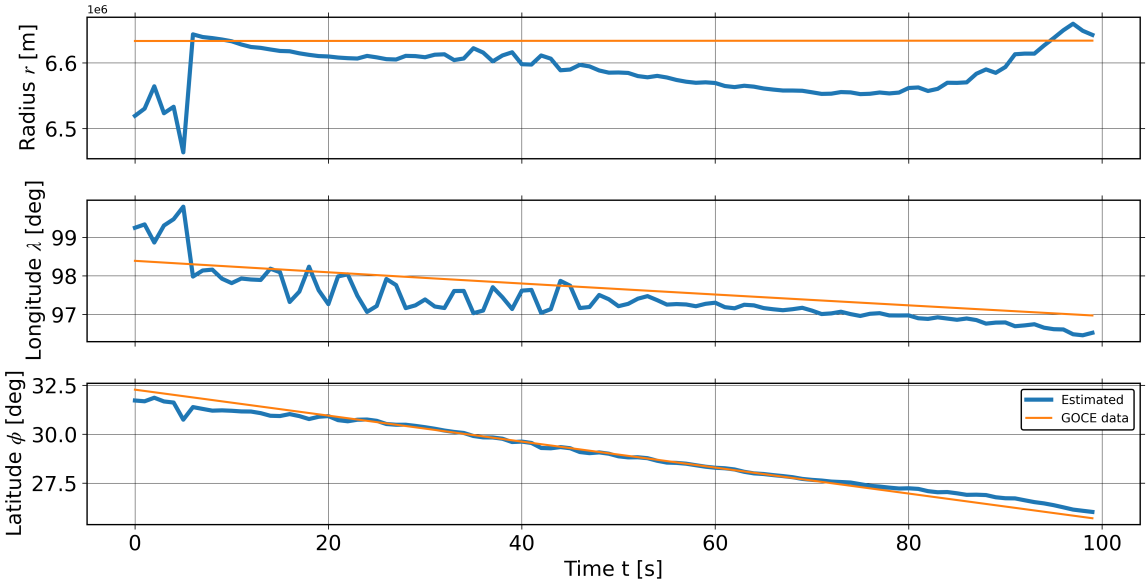

Figure 5.3: Orbital state (r, λ, ϕ) of the estimated position (max D/O of 100) compared to the actual GOCE positions provided by ESA.

Figure 5.3 displays a plot for each of the coordinates in satellite's position. The estimated line and the GOCE line follow a close pattern for both the second and third plot, which correspond to the longitude λ and latitude ϕ. Latitude ϕ shows the closest pattern. Most of the deviation occurring in the first 20 seconds, and has an average deviation of $0.276142[deg] = 31970.4[m] \approx 0.969\%$. Longitude λ shows a turbulent set of initial deviations from the true value before converging to the GOCE data with a deviation of $0.473453[deg] = 54813.4[m] \approx 0.486\%$. Figure 5.4 shows the deviations between the estimated state and the GNSS provided state of GOCE.

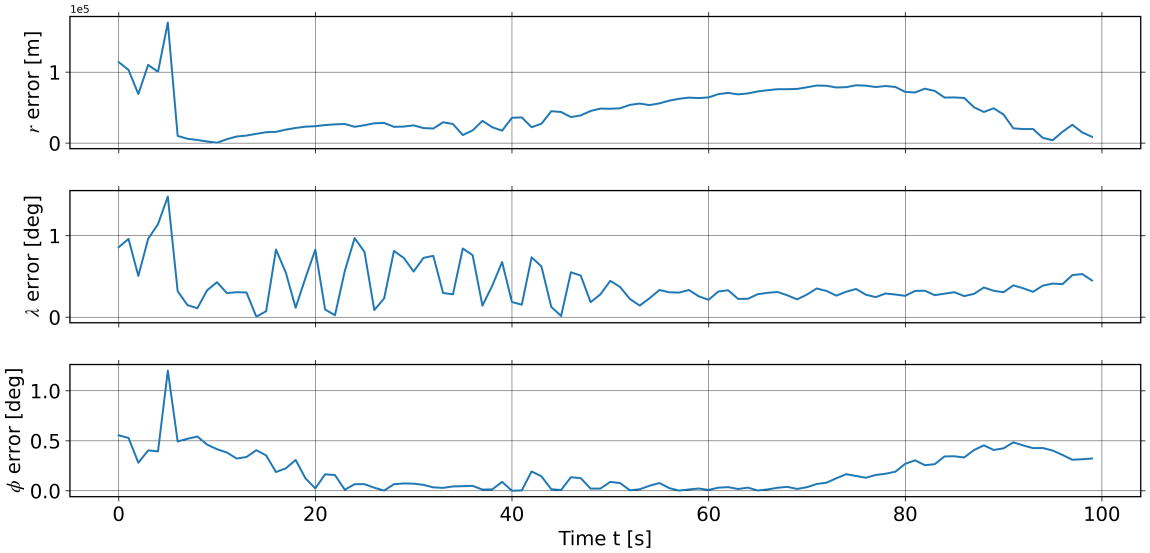

Figure 5.4: Orbital state (r, λ, ϕ) deviation from the estimated position (max D/O of 100) compared to the GNSS estimated GOCE positions.

The largest visual deviation from the pattern is for the estimation of radius r, with an RMS r error of $54862.3[m] \approx 0.827\%$. This may not look like a significant amount of deviation, but it is far larger than most other tests in this Thesis. There are multiple factors that cause this significant deviation.

- The problem arises when transforming in frames, as seen in Equation 5.4, because the first derivatives (gravity acceleration) (V_r, V_λ, V_ϕ) are required. Even though they are obtained at high accuracy, it is still part of the estimator, and not the real data. Thus, there is a discrepancy between the actual GSF measured gravity gradient and the one calculated for this test. The gravity acceleration measured by the instruments can be obtained by using common mode calculations, or by integrating the gravity gradient tensor, and then performing an estimation of each integration arbitrary constant.
- The GOCE data from ESA is not a perfect match to the real position and gravity gradients experienced by the satellite. There are errors and calibration parameters in the gradiometer, as seen in Section 2.3.1 which are not assumed to not be present (such as K_2, E_{al} or E_{co}. The GNSS data is also an estimation of the position, which, despite its accuracy, does not ensure a perfect measurement.
- Bias $\vec{b}$ and scale coefficient $\vec{C}$ are not only 3 values for the entire gradiometer, as assumed in this thesis. Instead, there are 3 separate values local to each accelerometers. By observing data from what is shown by Visser, P.N.A.M· et al. [22], there is average bias of $(-187.426,\ 301.097,\ -67.1223)\ [nm/s^2]$ in the GRF, but each accelerometer has very different values between each others, with also different drifts. Scale factors (and by extension, scale coefficients) are also seen to have distinct values for each accelerometer.

Both of these factors are the cause for the significant amount of deviation between the state value estimated by the least squares algorithm, and the data state.

6

Results

In this chapter, the least squares algorithm is applied to the data of a simulated Lunar orbit. Several parameters are tested to find the optimal algorithm settings for the simulated orbit. With this combination, the main thesis question is answered by running the algorithm while including errors, a realistic number of iterations per epoch, and calibration parameters.

6.1. Procedure

The primary objective of this thesis is to assess the feasibility and accuracy of satellite navigation based solely on gravity gradiometry. This approach relies exclusively on gravity gradient data, without the use of GNSS signals or ground-based information. To accomplish this, the accuracy is evaluated in an orbit around the Moon, in which GNSS is not as readily available. In a real scenario, the data used for estimation is the measured gravity gradient from the satellite and gravity field model data of the Moon. To generate the results in the thesis, a simulation of a lunar orbit is performed.

As described in Section 2.6, the orbital simulation is performed by using only the spherical harmonic acceleration as a result of the Moon's mass and shape. The settings for the initial state are: semi-major axis of $2.99279 \cdot 10^{6}[m]$, an eccentricity of $4.03294 \cdot 10^{-3}$, an inclination of $1.1[rad]$, an argument of periapsis of $1.31226[rad]$, and a longitude of ascending node of $3.82958 \cdot 10^{-1}[rad]$. Figure 6.1 shows the position parameters.

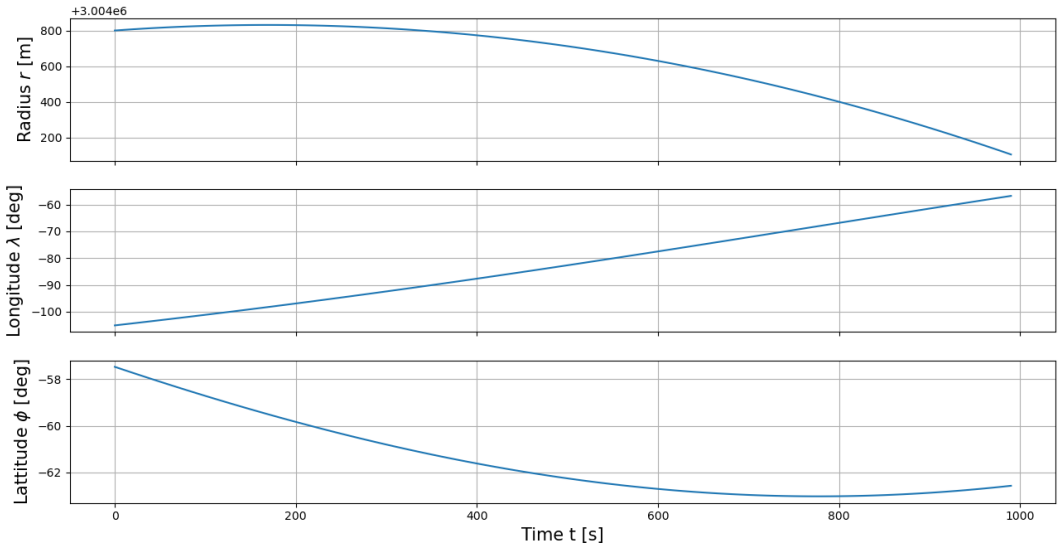

Figure 6.1: Spherical coordinates of the satellite position in the simulated Lunar orbit.

Figure 6.1 shows the orbit truncated to the first 100 epochs (1000 seconds). The orbit data is shortened to this time length in order to produce the accuracy tests in a feasible time frame.

Figure 2.6 is used to compute the gravity gradient component vector $\vec{V}_{uu}^{(measured)}$ of maximum D/O 200 in every epoch for this section. The high D/O is to simulate highly accurate readings so that errors and calibration parameters can be introduced separately. .

6.2. Accuracy Analysis

Several parameters influence the accuracy of the estimation process and must be considered when evaluating its performance. These parameters affect the reliability and precision of the navigation solution and are therefore critical to the assessment.

6.2.1. Degree and Order

The level of detail in a gravity field model is determined not only by its accuracy but also by the maximum D/O used in its definition. A higher D/O enables the representation of more complex gravitational variations, thereby capturing finer details of the celestial body's gravity field. In gravity gradiometry, the accuracy of the GGT, as seen in Equation 2.4, also depends on the chosen maximum D/O. To evaluate the effect of D/O on navigation performance, position estimates are computed using varying maximum D/O values. These tests are performed with an uncorrected algorithm to better show exactly how maximum D/O affects the estimation without error correction involved, and with 5 iterations per epoch to make sure the estimations converge. Figure 6.2 and Figure 6.3 illustrate the effect of varying the maximum D/O on the estimated geocentric radius r because among all estimated parameters, r is the one that visually shows best the deviation from the simulated position.

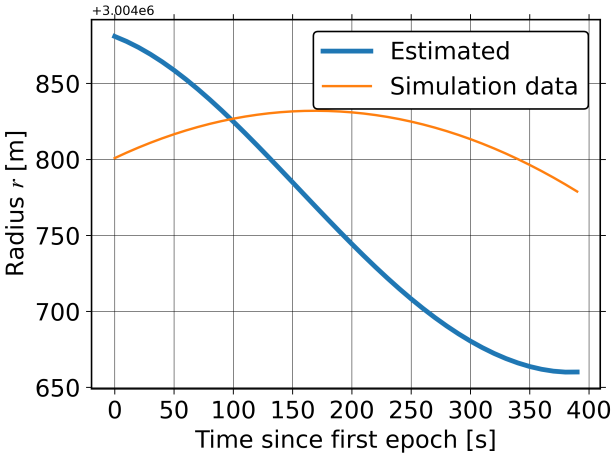

Figure 6.2: Estimated position compared to the simulation for 40 epochs at a maximum D/O of 20.

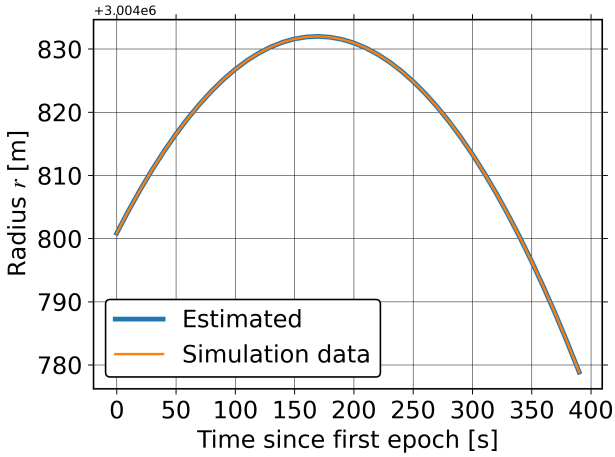

Figure 6.3: Estimated position compared to the simulation for 40 epochs at a maximum D/O of 180.

The most notable result from Figure 6.2 and Figure 6.3 is that position estimations using lower maximum D/O of 20 exhibit significantly larger deviations from the actual trajectory compared to those using higher D/O values of 180, with an RMS radial r deviation of $88.3560[m]$. A maximum D/O of 180 generates an RMS radial r deviation of $2.85158 \cdot 10^{-10}[m]$. Lower degrees provide the main contributions to the gravity gradient with higher scale values and less detailed shapes. Higher degrees provide the smaller details that can lead to more accurate estimations. The impact of D/O combinations can be observed in:

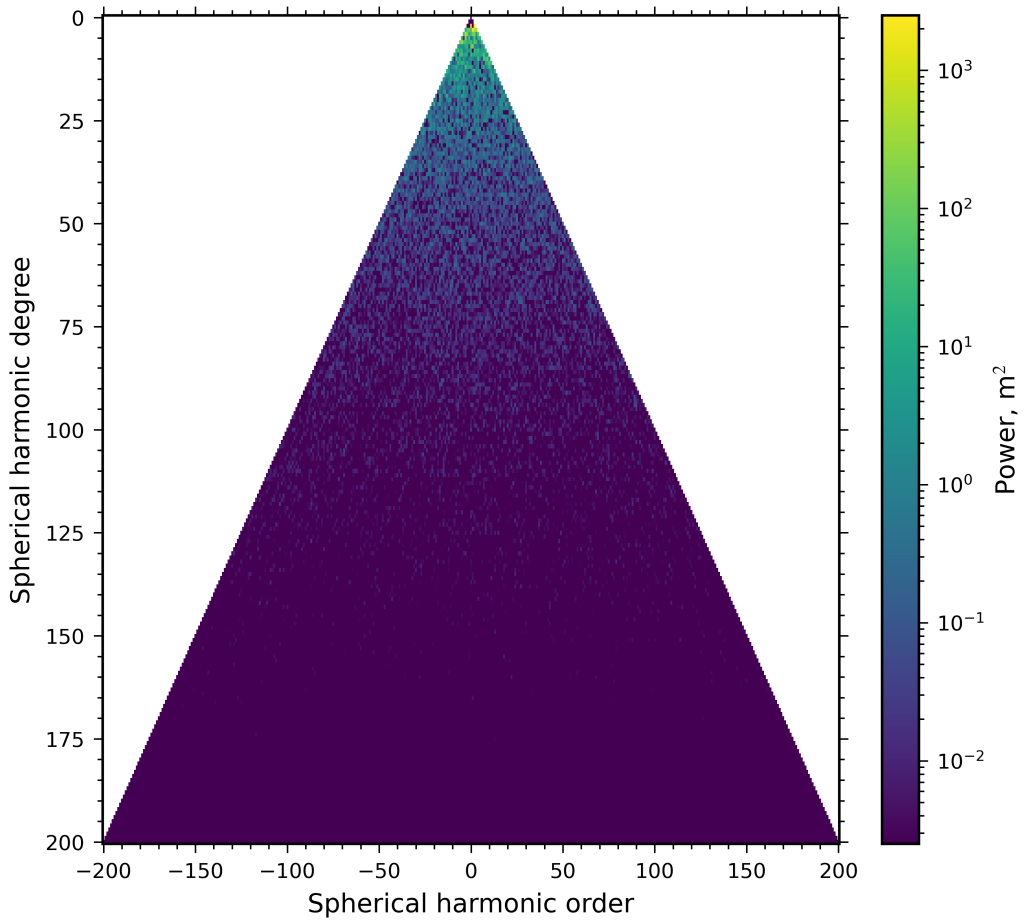

Figure 6.4: Fraction of relevance (power) each D/O combination has within the formulation of gravity gradients for the GRGM1200B model for the Moon.

This graph shows the power and thus significance each Stokes Coefficient has for each D/O combination. Lighter coloured D/O combinations have more of an impact on the overall value of the gravity potential. This is also applied to its second spatial derivative, the gravity gradient. The graph is made with the use of a Moon gravity field model. It can be seen that, for the Moon's gravity field, the first 100 D/O combinations are the strongest, making up for 99.9391% of the final value. The change on gravity gradient quantities after D/O 100 can be assumed to be almost negligible, since they contribute 0.0608953%.

Figure 6.5 shows scattered points representing the average r deviation when the least squares algorithm is run at a different maximum D/O, which indicates the maximum D/O impacts the overall accuracy of the results, where the curves of best fit are found by exponential regression.

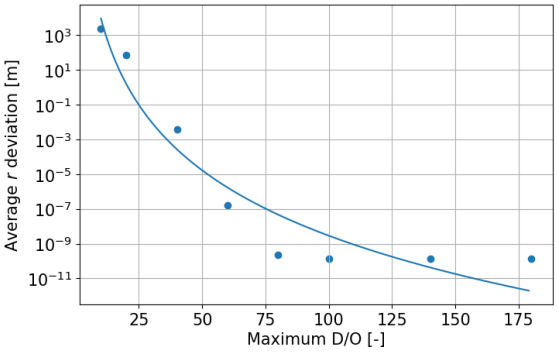

Figure 6.5: Average deviation of the estimated position for calculations using different levels of maximum D/O.

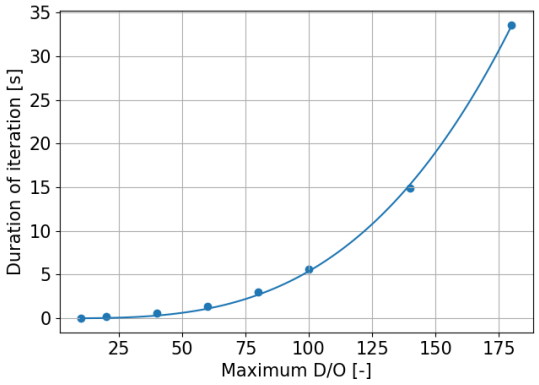

Figure 6.6: Average duration for calculations using different levels of maximum D/O.

The position deviations converge with points with a larger maximum D/O of 75 to $1.0458145 \cdot 10^{-5}$ [m]. The largest improvements to the deviations reach a plateau between from D/O of 80. This shows the correlation between the converged deviation from the maximum D/O and the power of each D/O combination shown in Figure 6.4. Another point of notice is the time it takes to compute the estimated positions, which increases significantly with the maximum D/O. Figure 6.6 shows a scatter of points indicating the computation time for a different maximum D/O. This is the same behaviour as the one found in Section 3.3. The best D/O to maximise accuracy and minimise computation time is 80.

6.2.2. Instrument Scale and Biases

The accuracy of the instruments measuring the GGT affect the accuracy of the navigation estimates. The errors in measurement are included in the $\vec{V}_{uu}^{\prime(measured)}$ observation vector. A change in $\vec{V}_{uu}^{\prime(measured)}$ affects not only the position in which each least squares epoch converges, but the entire corrected state $\vec{y}$ that it estimates. To deal with this, parameter estimations from the corrected least squares method (as seen in Section 2.3.2) can be utilised to deal with inaccuracies caused. Figure 6.7 and Figure 6.8 shows how a scale coefficient and bias can affect the estimations respectively. In these tests, a large scale coefficient $\vec{C}$ of 1.2 in all GSF directions or a large bias $\vec{b}$ of $0.2[1/s^2]$ in all GSF directions are introduced respectively. An arbitrary maximum D/O of 100 is considered for the estimation of $\vec{V}_{uu}$ for a fast but detailed computation, and the uncorrected least squares algorithm (as seen in Equation 2.23, Equation 2.21 and Equation 2.24) is utilised to compare the results to how the corrected algorithm deals with this.

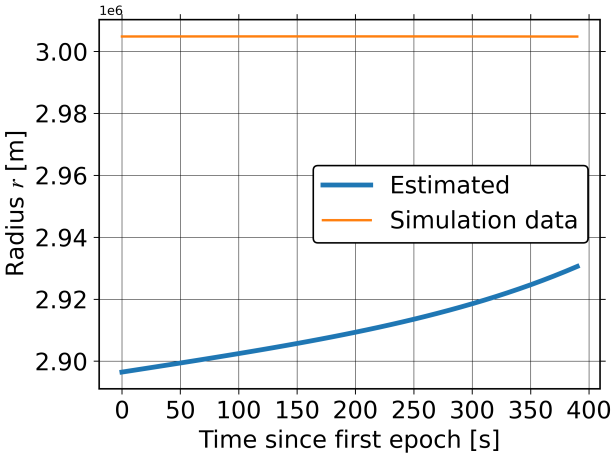

Figure 6.7: Estimated position compared to the simulation with uncorrected errors $\vec{C} = (1.2, 1.2, 1.2)$. RMS r deviation of $94824.6[m]$.

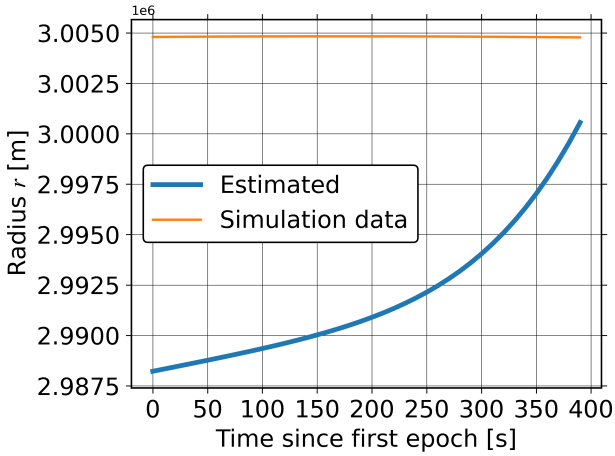

Figure 6.8: Estimated position compared to the simulation with uncorrected biases $\vec{b} = (0.2, 0.2, 0.2)$. RMS r deviation of $13313.0[m]$.

In Figure 6.7 and Figure 6.8, the first epoch estimated position (orange line) deviates the most, by $\approx 107500[m]$ and $\approx 17000[m]$ respectively, from the simulated path (blue line). Both the scale coefficient $\vec{C}$ and bias $\vec{b}$ have a very significant influence on the accuracy of the least squares method when uncorrected. The values of the gravity gradient $\vec{V}_{uu}^{(measured)}$ affected by the calibration parameters lead the least squares algorithm into a different point to estimate, thus the large jump for the first epoch's estimate in both figures. Larger parameters cause a larger deviation in measured gravity gradient, hence causing a larger deviation in the estimated points by the least squares algorithm.

6.2.3. Instrument white noise

In addition to scale factor and bias, white noise is also a relevant factor, as seen in Equation 2.35. Gradiometers can have white noise $\vec{n}$ error of $0.1[E] = 1 \cdot 10^{-10}[1/s^2]$ in the form of white noise [23]. For this test, white noise is randomly generated in a Gaussian distribution between $0.1[E]$ and $-0.1[E]$ and added to $\vec{V}_{uu}^{(measured)}$ in an uncorrected algorithm. Figure 6.9 shows the deviation, which is very small compared to the deviation caused by the scale coefficient and bias. The pattern shows the random nature of deviations caused by random white noise, and creates an RMS error of $4.26065 \cdot 10^{-6}[m]$ that can not be minimised by this algorithm.

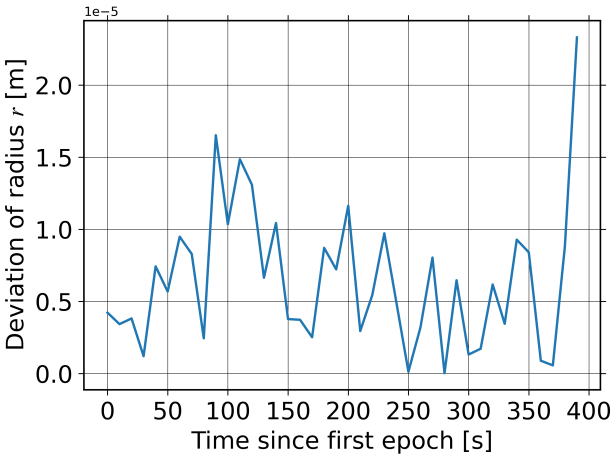

Figure 6.9: Estimated position deviation compared to the simulation with white noise $0.1[1/s^2]$. RMS r deviation $= 8.10972 \cdot 10^{-6}[m]$.

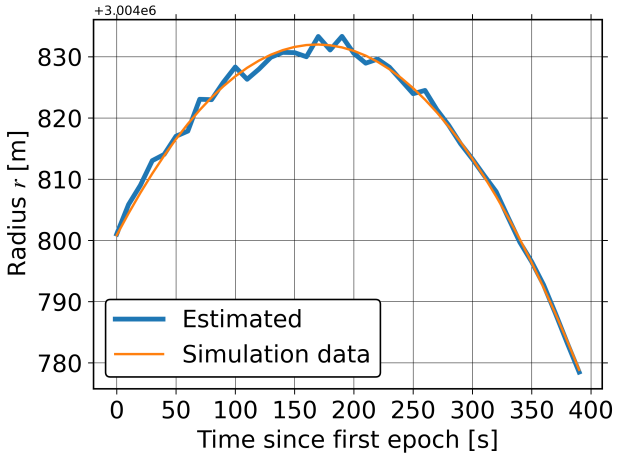

Figure 6.10: Estimated position compared to the simulation with corrected scale coefficients $\vec{C} = (1.2, 1.2, 1.2)$, biases $\vec{b} = (0.2, 0.2, 0.2)$, and white noise $0.1[1/s^2]$. Average r deviation $= 9.56970 \cdot 10^{-1}[m]$.

6.2.4. All Instrument Errors

This test evaluates the accuracy of the least squares algorithm with all errors considered during estimation. Instead of the uncorrected algorithm, this test considers the corrected version, as seen in Equation 2.42, Equation 2.41 and Equation 2.43. Figure 6.10 shows the estimated position (blue line) following a close pattern to the simulated position (orange line), which shows a significant improvement in state estimation compared to the uncorrected versions seen in Figure 6.8 and Figure 6.7. Figure 6.11 shows the deviation between the estimated position and the simulated position in full detail and how all combined measurement errors affect the corrected estimated positions.

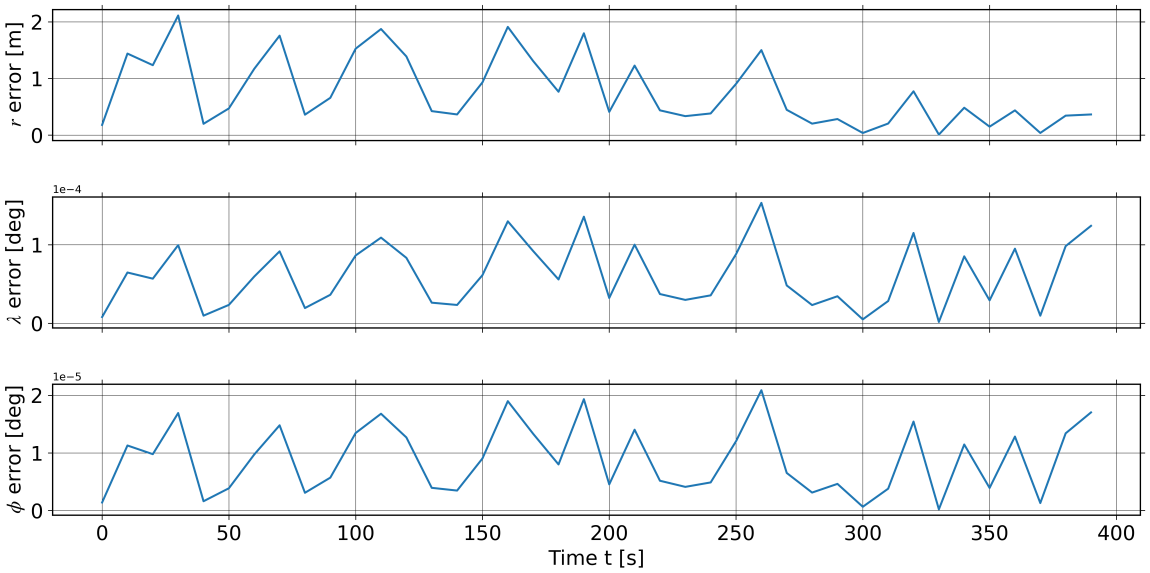

Figure 6.11: Position deviation of the least squares algorithm from the actual simulated positions while correcting for scale factor $\vec{C}$, bias $\vec{b}$ and white noise $\vec{n}$.

The RMS errors of the estimations are $9.56970 \cdot 10^{-1}[m]$ for r, $4.17161[m]$ for λ, and $5.94129 \cdot 10^{-1}[m]$. The corrected algorithm significantly reduces the converged deviation from the simulated position, making it reliable to use to estimate calibration parameters. Figure 6.11

shows similar paters for all parameters, but with a larger error in the λ direction. This is attributed to λ being the parameter with rate of change during the orbit, and thus having more distance between epochs to converge.

6.2.5. Number of Iterations and Dampening Factor

All the previous figures in Section 6.2.2 Section 6.2.3 and Section 6.2.4 show the estimation process done with 5 least squares iterations per orbit epoch. The number of iterations is done within each epoch and allows the estimation to converge on a value before moving on to the next epoch, which is estimated independently as seen in Section 2.4, while carrying the estimate for the previous epoch as an initial guess. Reducing the number of iterations reduces the computation time proportionally, while also reducing the accuracy. This section reports how many iterations are enough to converge at an accurate estimate with less than $10[m]$ of deviation from the simulated position. Since the dampening factor τ, as shown in Section 2.3.3, changes with each iteration, the initial dampening factor is a variable to take into account in this test.

All tests in this section are done using the corrected least squares algorithm. For the first test, the number of iterations is reduced to 2 without changing the dampening factor. The expected effect is that a high dampening factor slows down the convergence rate by iteration, while a low dampening factor, even though it is more unstable, yields a faster convergence rate by iteration. Figure 6.12 shows how only 2 iterations per epoch affect the estimation with a dampening factor of $\tau = 1 \cdot 10^{-7}$.

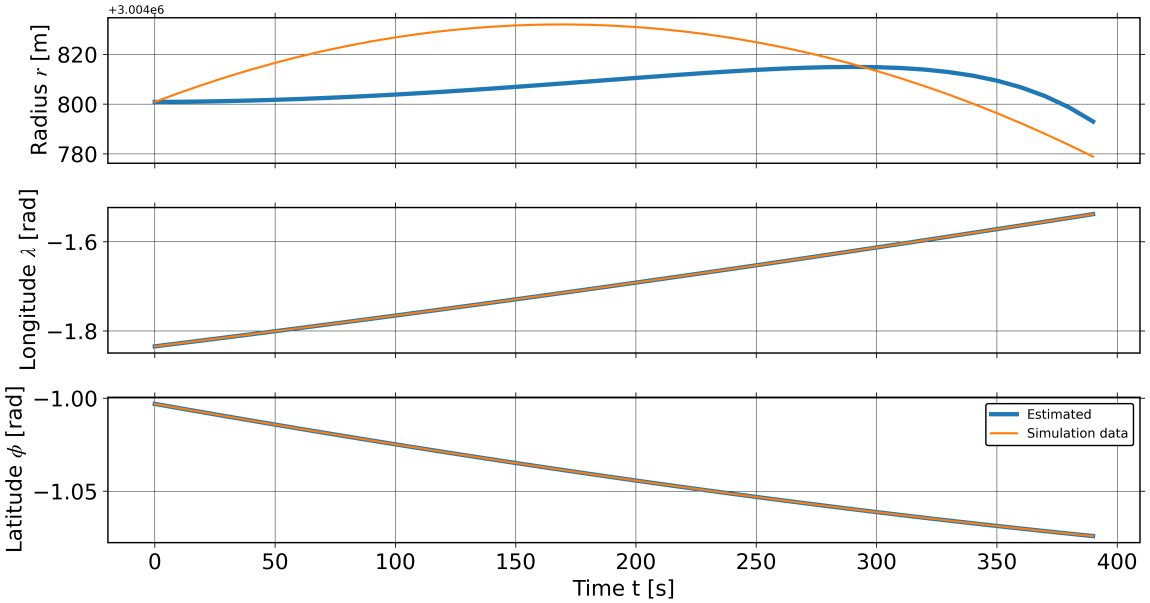

Figure 6.12: Position of the least squares algorithm compared to the actual simulated position. 2 iterations per epoch, initial $\tau = 1 \cdot 10^{-7}$.

The largest visual deviation from the simulated pattern is shown to be associated with the estimated radius r, as depicted by the distance between the yellow and blue lines in the top plot of Figure 6.12 with an RMS r deviation of $16.4411[m]$. On the other hand, the largest total deviation of $88.1222[m]$ for λ. The graph also displays a pattern in shows that the blue line moves towards the orange one with a delay, which means that the least squares algorithm estimations are directed towards the simulated state, but have too much dampening to attain a value large enough to converge closer.

For a new test, instead of changing the number of iterations, the dampening factor tau is reduced, and thus the convergence per iteration becomes larger. The initial dampening factor is reduced to $\tau = 1 \cdot 10^{-9}$ to increase convergence by 10^2 fo the first iteration and by 10^4 for the second iteration when compared to $\tau = 1 \cdot 10^{-7}$ (as seen how dampening functions, as shown in Section 2.3.3. Figure 6.13 shows how this change has affected the estimated r, which was previously shown to be the visually deviated from the pattern.

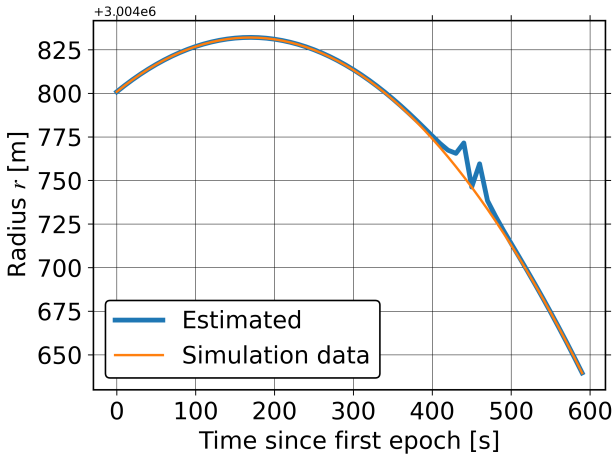

Figure 6.13: Radius r of the least squares algorithm compared to the actual simulated r. 2 iterations per epoch, initial $\tau = 1 \cdot 10^{-9}$.

In these results, the RMS r deviation is reduced to $3.83494[m]$, while the λ error is reduced to $2.21244[m]$. With less dampening, each iteration converges faster to the estimated value, making the estimations significantly closer to the correct values. The reduction in dampening factor also leads to more extreme first iterations per epoch.

A third test is conducted in which the dampening factor reduced to $\tau = 0.0$ to test how no dampening affects the iterations.

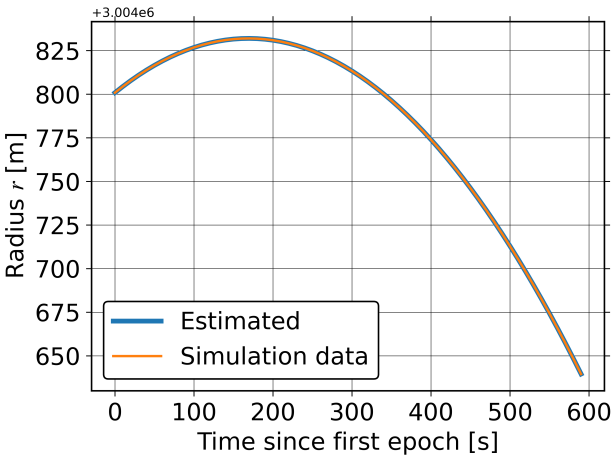

Figure 6.14: Radius r of the least squares algorithm compared to the actual simulated r. 2 iterations per epoch, initial $\tau = 0$.

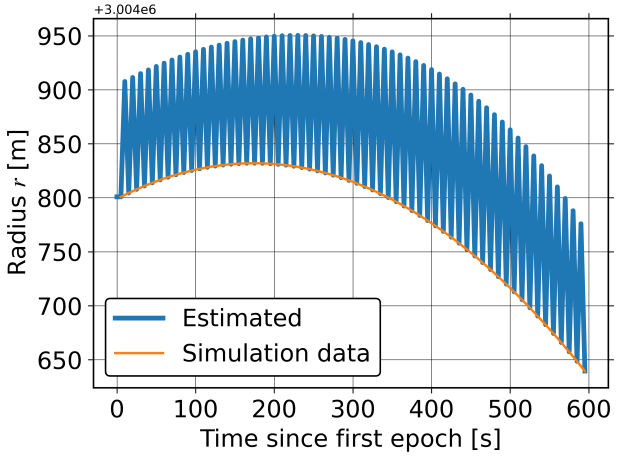

Figure 6.15: Radius r of the least squares algorithm compared to the actual simulated r including the first iteration result. 2 iterations per epoch, initial $\tau = 0$.

Figure 6.14 shows a further improvement in the final iteration of the estimation, with the

blue estimated line following an even closer pattern to the orange line and an r deviation of $1.45132 \cdot 10^{-2}[m]$. Figure 6.15 shows the same test, but includes the estimation data from the first of the 2 iterations in each epoch. The first iterations in each epoch can be seen creating spike patterns on the blue line. With no dampening, the first iteration overshoots by estimating a $d\vec{y}$ that is too large. Even though the second iteration results are improved in every epoch, there are large spikes in the estimations for the first iterations. In this test, these spikes do not affect the end result. In an orbit with a higher altitude, spikes may overshoot enough to reach the wrong convergence threshold seen in Section 3.1, and yield an incorrect estimation.

6.3. Combined Test

Realistically, all of the factors that affect the accuracy of the estimator happen at the same time. For a final test, all errors are introduced as white noise of $0.1[E]$, $\vec{C} = (1.15, 0.95, 1.05)$, $\vec{b} = (0.05, 0.10, -0.20)[m/s^2]$. The biases and errors are arbitrarily chosen values that represent a more extreme case when compared to the ones provided in the GOCE data in Chapter 5. An arbitrary initial estimation deviation of $1000[m]$ is set to observe how the algorithm also deals with deviation in a combined setting. When it comes to iterations, dampening factor and maximum D/O, the most optimal combination is chosen. A maximum D/O of 80 is close to the error convergence seen in Figure 6.5 so it is chosen as part of the optimal combination for this test. In the device (Intel(R) Core(TM) i7-13700H CPU, 16 GB RAM) in which tests are conducted in this thesis, a D/O of 80 iteration takes $\approx 4[s]$ to compute. The Sampling rate of the gravity gradient observations, and, consequently, of the orbital positions considered in the simulation is $10[s]$. With an iteration time of $4[s]$, it leaves room to perform 2 iterations. For these iterations, the initial dampening factor is $\tau = 1 \cdot 10^{-9}$, as it is shown in Section 6.2.5 to be an optimal dampening factor with 2 iteration, but is still high enough to be safe against sudden divergence. Figure 6.16 shows the final position results, in which the blue line that represents the estimated state follows a close pattern to the simulated line, while having deviations, especially observed in the pattern of the r state.

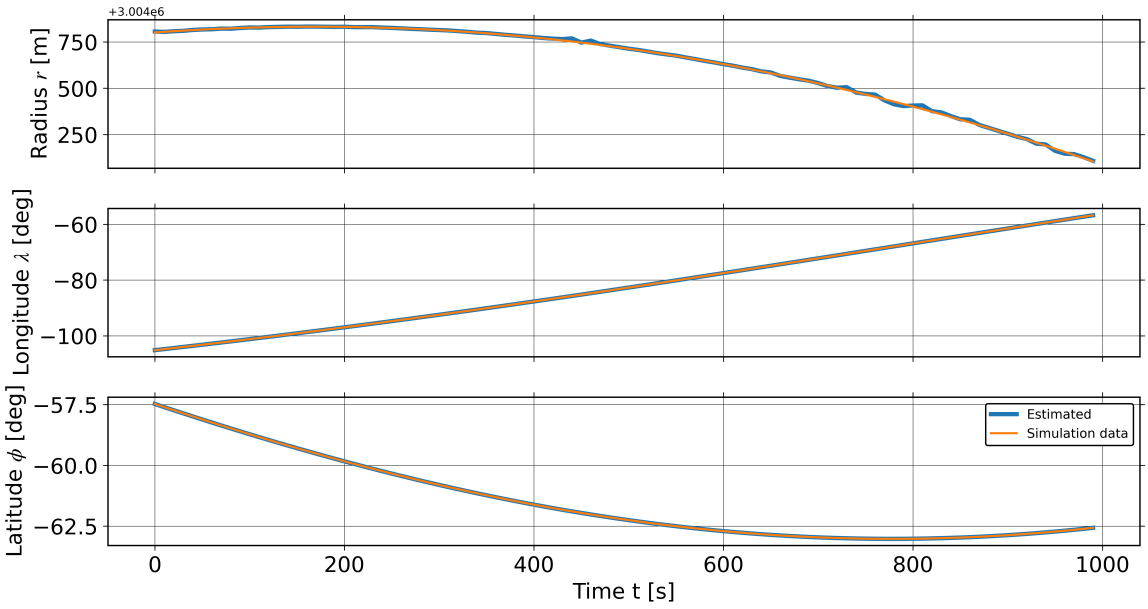

Figure 6.16: Orbital state $\vec{y}(r, \lambda, \phi)$ of the estimated position (max D/O of 80) compared to the simulated positions with all errors introduced.

The final result for this test shows an RMS radius r deviation of $6.89922[m] \approx 2.29630 \cdot 10^{-4}$%,

λ deviation of $1.78146 \cdot 10^{-4}[deg] \approx 2.22683 \cdot 10^{-4}$%, and ϕ deviation of $1.56926 \cdot 10^{-4}[deg] \approx 2.57256 \cdot 10^{-4}$%. Figure 6.17 shows the deviations of the estimated position when compared to the simulated position.

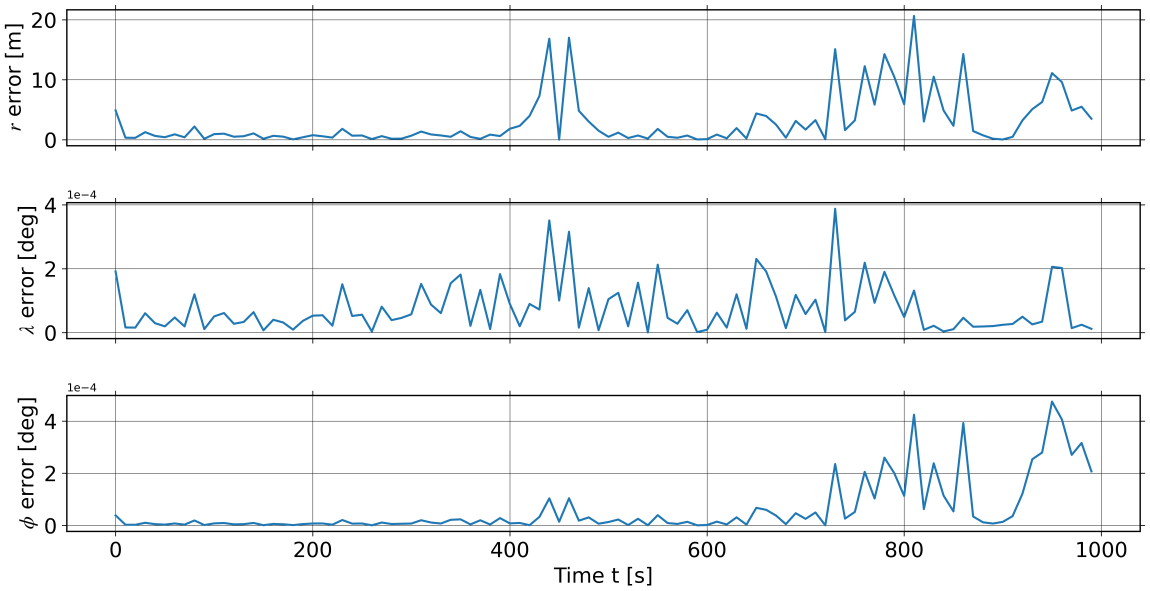

Figure 6.17: Orbital state r, λ, ϕ deviation of the estimated position (max D/O of 80) from the simulated positions with all errors introduced.

In Figure 6.17, it can be observed that the longitude λ shows more unstable patterns in the form of larger spikes. Longitude is the state that changes the most during an orbit, from going through a range of $-100[deg]$ to $-60[deg]$, while r and ϕ don't change as significantly as λ. In between each epoch, λ is the state that changes the most, as seen in Figure 6.1. Thus, the least squares algorithm converges in values of λ slower than the other 2 states.

6.4. Additional Observations

A polar orbit is taken as reference to assess how extreme values of latitude ϕ are processed. Figure 6.18 shows a severe sudden deviation after $350[s]$ of orbit in the estimations. This is generated by running the algorithm on a polar orbit with 40 epochs, 5 iterations, a dampening factor of $1 \cdot 10^{-7}$, uncorrected, no errors, and no initial deviations. .

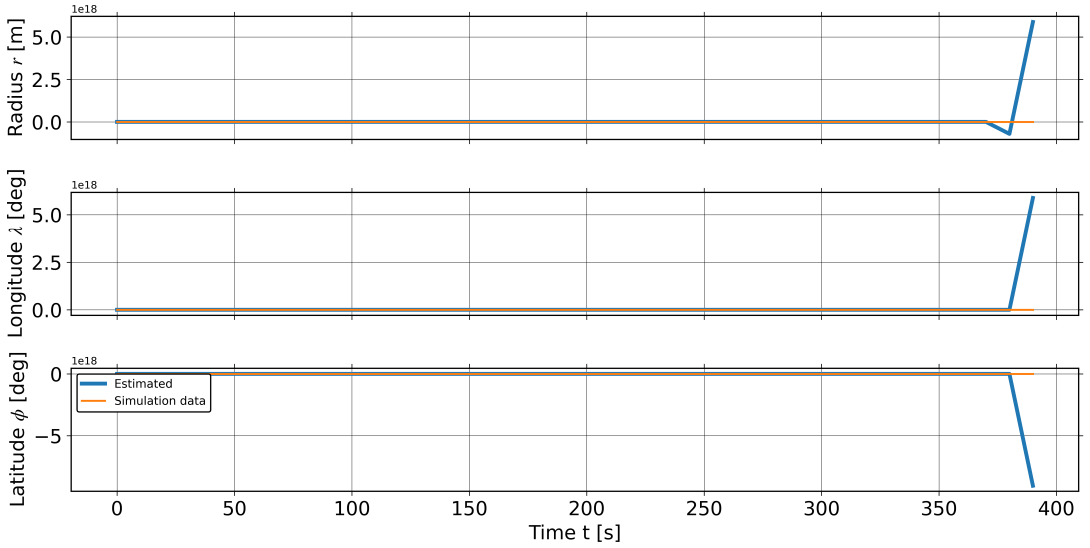

Figure 6.18: Spherical coordinate estimation of the satellite position in the Tudat Lunar polar orbit simulation.

The sudden spike is caused by the latitude angle ϕ approaching $\pm 90[deg]$. The calculations during the least squares estimation involve involve terms of the form of $\frac{1}{\cos(\phi)}$. As ϕ approaches this value, $\cos(\phi)$ approaches 0, causing the expression to diverge toward infinity and resulting in a sharp numerical instability. By taking the results of this orbit, the divergence starts when $\phi = -90.08[deg]$. With this, it can be deduced that the range for this singularity point is $\pm 89.92 - 90.08[deg]$.

This behaviour highlights one of the primary drawbacks of using spherical coordinates: the presence of singularities at specific angles, particularly near the poles, which can introduce significant errors in estimation.

7

Conclusion

GNSS data, as proven during the verification process, is overall more accurate and faster than a self reliant gravity gradient method. Thus, the purpose this thesis is best suited for other celestial bodies. Such as the Moon or Mars, which have been very accurately mapped in terms of gravity field. Additionally, there is the benefit that their gravity field is less dynamic than Earth's, which means that such missions have the potential to rely on accurate navigation without having to communicate with other satellites or Earth-bound positioning techniques. On top of this, the mission will be much more reliable and able to operate autonomously for longer durations of time.

7.1. Assumptions

Before a conclusive discussions, the assumptions made during this thesis project need to be mentioned, which are:

- The gravity field models are up to date. When it comes to performing any procedures around Earth, it is important to note that the gravity field of Earth shifts. Tectonic movement and tides for example, change the distribution of the mass of Earth. The errors that could arise from this fact are not taken into account.
- The accelerometer calibration parameters K_2, E_{co}, E_{al} can be assumed to be calibrated, and thus not included. The two exceptions being the bias $\vec{b}$ and the scale factor K (formulated as scale coefficient $\vec{C}$) of transforming from voltage to acceleration, as defined in Section 2.3.1.
- The bias $\vec{b}$ is assumed to be 3 global values (1 per axis) instead of 3 per accelerometer. This assumption is taken to estimate bias as part of the state in the least squares algorithm in each orbital epoch. This simplification is further touched upon in Section 2.3.1 and later in Section 7.3, where the consequences of taking this assumption are explained.
- Bias $\vec{b}$ is assumed to be static, thus, non-changing as the satellite orbit progresses because the testing of this algorithm is done through shorter orbit periods to keep the testing feasible in terms of computation time. With a short time frame, bias can be assumed to not change.
- White noise $\vec{n}$ is assumed to be a random Gaussian value with mean 0 and a maximum amplitude obtained from modern gradiometer accuracy in literature.

- In Chapter 6, the high D/O estimation of gravity gradient from the Moon orbit simulation is assumed to act as the measured gravity gradient.
- The distance between the accelerometers of the gradiometer l is assumed to be known by design, making it a parameter in the measured gravity gradient equation that does not have to be estimated.

7.2. Thesis Question Discussion

After analysing the tests carried during this thesis, the questions proposed at the start can now be answered.

- *How can bias and the scale factor be implemented into the least squares algorithm?*

The assumption that all measurement devices are aligned along the same coordinate axes allows for the simplification of sensor modelling by grouping all instrument biases and scale factors into a single value per frame axis. Systematic errors can be incorporated into the gravity gradient equations in the form of a scale coefficient $\vec{C}$ (which substitutes the scale factor) and bias $\vec{b}$. As a result, the gravity gradient tensor (GGT) is no longer symmetrical, since every term becomes different, making them 9 distinct values.

The coefficients and biases are introduced into the state vector as $(C_r, C_\lambda, C_\phi, b_r, b_\lambda, b_\phi)$, corresponding to scale and bias terms in the radial, longitudinal, and latitudinal directions, respectively. Incorporating the additional coefficient and bias states modifies the Jacobian of the GGT, which in turn affects the structure of the design matrix used in the least squares estimation process. With a total of 9 observations and 9 unknowns, it becomes possible to simultaneously estimate the position, scale coefficients at the same time while using least squares.

- *How do variables affect the accuracy of the gravity gradient navigation method?*

Orbit estimation is a process that requires a large amount of variables which change the outcome, and thus the accuracy of the results.

During sensitivity analysis tests, it is observed that a larger deviation in the starting position estimate causes a slower convergence towards the actual value. Deviations of around $1[km]$ need 3 iterations to converge. Deviations around $100[km]$ need 5 iterations to converge. Deviations of more than $211.285[km]$ on each axis converge into incorrect points. It is also observed on an orbital altitude test that orbits at higher altitudes converge at a faster rate, needing 1 less iteration to arrive at the final estimation. On the other hand, higher altitudes but display a larger deviation after convergence, with an error of up to $10^5[m]$ or even $10^9[m]$ at an altitude of $2 \cdot 10^9[m]$ and $2 \cdot 10^{10}[m]$ respectively. The convergence rate may be caused by the smoother pattern of the gravity field models at higher altitudes, making a faster convergence towards a value. The larger error is caused by the significant drop in intensity of gravity gradient at higher altitudes, at which the noise is far more significant relative to the signal detected by the gradiometer.

Testing different gravity field models also proves the impact of small variations in gravity field. When 3 models derived from GOCE data are compared, they yield a difference in position in the order of 10 to $100[m]$. Although this is a small difference; it shows that different which gravity field models impact this method. It also proves that around celestial bodies such as the Moon or Mars, in which the gravity field is less dynamic, this method is potentially more accurate.

Testing the algorithm with GOCE data shows that transformations between frames require a large volume of data, most of which comes from gradiometer and star tracker data, which means that there is a significant amount of errors that can be introduced from the moment the gradiometer measures the gravity gradient, to introducing the gravity gradient into the least squares algorithm. Because of this, there is a notable discrepancy between the estimated and the observed state position values.

During the final testing, multiple other variables are found to affect accuracy, notably the maximum degree and order (D/O) used in the estimations. A low maximum D/O of 20 can result in estimations different from the actual orbital positions, and which don't even follow its pattern, with an RMS radial r deviation of $88.3560[m]$ and a computation time of $0.22[s]$. High maximum D/O of 180 display more accuracy, with an RMS r deviation of $2.85158 \cdot 10^{-10}[m]$, at the cost of a severe increase in computation time of up to $33[s]$ per iteration. When introducing scale factors (scale coefficient $\vec{C}$), biases $\vec{b}$ and white noise $\vec{n}$, the results are also affected, with RMS r deviations of even $94824.6[m]$ and $13313.0[m]$. When introducing a white noise with an amplitude of $0.1[E] = 10^{-10}[1/s]$, a randomly fluctuating deviation from the simulated position with an RMS of $8.10972 \cdot 10^{-6}[m]$. When using the corrected least squares algorithm designed to estimate scale coefficient and bias instead, the error is reduced to 10^{-1} to $10^{0}[m]$. Each epoch is independently processed over multiple iterations of the least squares method. The number of iterations has a significant impact on the accuracy, with a lower amount of iterations leaving less chances for the least squares algorithm to converge. Although this can be compensated by lowering the dampening factor τ, it can also lead to more unstable estimation, specially when the there is a large deviation in the initial estimate.

- *What is the accuracy of the algorithm when real data is used?*

The algorithm is applied to real data from GOCE orbital positions and gravity gradients. The gravity gradient tensor (GGT) is the measured gravity gradient used as input for the least squares algorithm, and the positions serve as a reference to determine the error. The RMS accuacy is of $54862.3[m]$, or about 0.688254% of orbital radius r. The relative change seems small, but a deviation of more than $10000[m]$ is very significant, and for autonomous satellites basing their navigation on this algorithm, it can lead to incorrect attitude or orbital control responses. A cause for this deviation is related to the systematic errors which are not estimated in this algorithm. The gradiometer data needs to be transformed from the body to the LNOF which requires star tracker data that carries additional errors. On top of this, the provided in a local north oriented frame (LNOF), while the algorithm is implemented in the geocentric spherical frame (GSF). To transform from LNOF to GSF, the first derivative of gravity potential is needed. This is computed by the implemented algorithm for the GOCE position. Since the gravity accelerations are computed while the gravity gradients are taken from the data, the discrepancy between these two forms causes further errors.

How accurately can a least squares algorithm estimate satellite position in an environment without GNSS?

The last test carried during this thesis evaluates the accuracy of the algorithm in a Lunar orbit. Event hough recent development shows that some GNSS signals can reach satellites in orbits around the Moon [24], it is still an environment in which an autonomous navigation system can be an advantage. For the final test, white noise, bias and scale factor errors are added into the measured GGT. It also includes an initial deviation of $1000[m]$. It shows that, under the assumptions made in this thesis, a least squares algorithm can estimate with a deviation of $6.89922[m] \approx 2.29630 \cdot 10^{-4}$%, $1.78146 \cdot 10^{-4}[deg] \approx 2.22683 \cdot 10^{-4}$%,

$1.56926 \cdot 10^{-4}[deg] \approx 2.57256 \cdot 10^{-4}$% for radius r, longitude λ and latitude ϕ respectively.

7.3. Limitations

Despite its demonstrated feasibility, the proposed algorithm presents several limitations that must be acknowledged.

A primary limitation stems from the use of GSF as the reference frame for the estimation process. While this frame aligns naturally with the representation of gravity fields with spherical harmonics, it introduces several computational and numerical challenges. One such challenge is that two of the parameters in GSF, latitude (ϕ) and longitude (λ), are angular rather than linear. This complicates the error propagation and comparison process within the least squares estimation, as angular quantities require a wight factor in the least squares to ensure consistent and meaningful results. Small changes in angular values can result in disproportionately large errors.

Another significant drawback of the GSF is the presence of coordinate singularities, particularly when ϕ approaches $\pm 89.92 - 90.08[deg]$. The mathematical formulation involves the term $\frac{1}{\cos(\phi)}$. At these points, the term diverges toward infinity, leading to instability and sharp spikes in the estimated values. Singularities can severely affect the navigation performance, meaning that the algorithm is not suitable for polar orbits.

Additionally, the algorithm assumes that all noise and estimation errors are aligned with the axes of the selected coordinate frame. This simplification makes error handling within the algorithm more straightforward. However, it does not reflect the true nature of measurement errors in practical systems. In reality, sensor errors are axis-dependent and vary based on the orientation and characteristics of each individual instrument. By combining and treating errors as if they were aligned, the algorithm doesn't estimate the individual calibration parameters of each accelerometer. This causes a deviation from the actual position when compared to real-world measurements, reducing the physical fidelity of the results. This is one of the causes for the large deviation between GOCE data and estimations.

An important limitation to also note is the maximum deviation possible to converge into the correct position. With a deviation of at least $211285[m]$ in one of the axes, the least squares algorithm converges into an incorrect position. Thus, the initial estimation should not deviate more than this value. As a result, it is imperative that the weighed method is used to compensate for the scale differences between the linear and angular components of the algorithm.

The last limitation worth mentioning is the computational cost. Although the algorithm developed in this thesis runs on a personal computer, the use of rust and the proposed Legendre function implementation greatly reduces the computation time. It starts being specially noticeable at a maximum D/O of 80. It also means that to obtain accurate results feasibly, the tests conducted in this thesis are shortened to orbital epochs between 40 and 100 depending on the test.

The limitations above highlight the trade off between computational convenience and physical accuracy. Addressing them would require alternative coordinate representations, better handling of angular quantities, and a more realistic error modelling framework.

7.4. Research Plan Assessment

This section reiterated the research plan, the complications during the thesis project, and how it affected the process. First, a recap of the plan is layed out:

- The methodology (Chapter 2) answers the first sub-question, specifically Section 2.3.1 and Section 2.3.2, where calibration parameters are included into the estimation of the least squares method. The rest of the sections in the methodology describe the construction of the least squares algorithm, as well as details in how they are programmed.
- The Sensitivity Analysis (Chapter 3) partially answers the 2nd sub-question, in which initial deviation, orbital altitude and D/O are assessed on how they affect the algorithm. Primarily, it also serves as a section to know the limitations of the algorithm and how different circumstances can yield different results.
- The Verification (Chapter 4) is a section that is used to make sure the algorithm works correctly, not only as it is intended, but physically too.
- The Validation (Chapter 5) answers the 3rd sub-question by making use of a real case scenario in the form of GOCE data. It also describes the process in which this data is retrieved and transformed to be used by the algorithm.
- The Results (Chapter 6) finalise the answer to the 2nd sub-question by testing algorithm variables that can affect accuracy by making use of data from a Moon orbit simulation. With the knowledge of the optimal algorithm variables for this orbit, a final test is conducted where all calibration parameters and errors are introduced to finally answer what the accuracy to answer the main question.
- The Conclusion (Chapter 7) discusses directly the answers to the thesis questions. It also tackles the limitations of the algorithm and recommends future work.

The complications and their effects are addressed here:

- When testing the algorithm, it showed erratic and unstable behaviour when it comes to converging into the correct position states. This was found to be due to the difference in scale between the linear and angular components in the gravity gradient vector $\vec{V}_{uu}$. Because of this difference, as part of the methodology, the weighed method is implemented into the least squares algorithm, and a dampening factor is included which were not previously planned.
- When testing with GOCE data, the frame differences between the data provided and the data being used in the algorithm made it not possible to directly use it. This presented a large roadblock to the thesis process until the frame transformation was studied by using the Chistoffel symbols of the second kind, as shown in **??**, which extended the thesis time.

7.5. Future Work

There are several aspects of this research that can be explored in future work. First, there is the co-ordinate frame limitation. From the previous section, it can be seen that there is a very significant drawback from using GSF for the least squares estimation. Future work on this aspect should focus on finding the Cartesian form of GGT, its derivative, and applying it to the least squares algorithm.

The next step in for this algorithm is to research how to obtain, transform and use raw data from star trackers and accelerometers. This is a step forward to implementing a more accurate algorithm that considers real data, frames, as well as inaccuracies, errors and transformation matrices as a starting reference point. Even though GOCE data is for Earth orbit, it provides the methods to process data and generate a measured gravity gradient

parameter for the least squares process.

With the retrieval of instrument data, it is possible to have a more detailed overview of errors. This thesis assumes all instrument errors are aligned in the same axis. If this assumption is not taken, all these errors become separate. It allows for a more accurate estimation of measurement errors, and thus, a more accurate estimation of the orbital position state.

The proposed algorithm makes use of 1 epoch to estimate 9 different elements. If instead, several epochs are used at a time, more elements can be estimated, such as the individual accelerometer and star tracker calibration and data combination parameters can be estimated.

A

Derivations

A.1. Gravity Gradient Factorisation Derivation

To simplify and make it easier to follow, the associated Legendre functions with $x = \sin(\phi)$ are simplified as:

$$P_{pq}(\sin(\phi)) = P_{pq_s} \tag{A.1}$$

The gravity potential equation is used as a reference in the appendix to follow the derivations that stem from it:

$$V_{pq} = \frac{GM}{R}\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s} \tag{A.2}$$

A.1.1. Gravity Acceleration derivation

Gravity acceleration is obtained by differentiating the gravity potential shown in Equation A.2 over the 3 different coordinates. In this case, where the spherical coordinates are $u(r, \lambda, \phi)$, it is derived into $V_{r_{pq}}$, $V_{\lambda_{pq}}$ and $V_{\phi_{pq}}$ respectively as:

$$\begin{aligned} V_{r_{pq}} = \frac{d}{dr}V_{pq} &= \frac{GM}{R}(p+1)\left(\frac{R}{r}\right)^{p}\left(-\frac{R}{r^2}\right)(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s} \\ &= \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s} \end{aligned} \tag{A.3}$$

$$\begin{aligned} V_{\lambda_{pq}} = \frac{d}{d\lambda}V_{pq} &= \frac{GM}{R}\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda)(q) + S_{pq}\cos(q\lambda)(q))P_{pq_s} \\ &= \frac{GM}{R}q\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda))P_{pq_s} \end{aligned} \tag{A.4}$$

$$
\begin{aligned}
V_{\phi_{pq}} = \frac{d}{d\phi}V_{pq} &= \frac{GM}{R}\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))\frac{d}{d\phi}P_{pq_s} \\
&= \frac{GM}{R}\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P'_{pq_s}\cos(\phi) \\
&= \frac{GM}{R}\cos(\phi)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P'_{pq_s}
\end{aligned}
\tag{A.5}
$$

A.1.2. Gravity Acceleration

Gravity acceleration can be arranged in the form of a vector $\vec{V}_{u_{pq}}$, where each component corresponds to each axial direction:

$$
\vec{V}_{u_{pq}} = \begin{bmatrix} V_{r_{pq}} \\ V_{\lambda_{pq}} \\ V_{\phi_{pq}} \end{bmatrix}^T
\tag{A.6}
$$

A.1.3. Gravity Gradient Derivation

Gravity gradient is obtained by taking the second differential of the gravity potential V_{pq}, or the first differential of gravity acceleration $\vec{V}_{u_{pq}}$ with respect to the coordinate parameters $u(r, \lambda, \phi)$:

$$
\begin{aligned}
\frac{d^2}{dr^2}V_{pq} &= \frac{GM}{R}\left(\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}(p+1) + \left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}(p+1)^2\right)(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s} \\
&= \frac{GM}{R}\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1}(p^2+3p+2)(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s} \\
V_{rr_{pq}} &= \frac{GM}{R}\left(\frac{1}{r^2}\right)(p+1)(p+2)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s}
\end{aligned}
\tag{A.7}
$$

$$
\begin{aligned}
\frac{d^2}{dr\,d\lambda}V_{pq} = \frac{d}{d\lambda}V_{r_{pq}} &= \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda)(q) + S_{pq}\cos(q\lambda)(q))P_{pq_s} \\
V_{r\lambda_{pq}} = V_{\lambda r_{pq}} &= \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)q\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)P_{pq_s}
\end{aligned}
\tag{A.8}
$$

$$
\begin{aligned}
\frac{d^2}{dr\,d\phi}V_{pq} = \frac{d}{d\phi}V_{r_{pq}} &= \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))\frac{d}{d\phi}P_{pq_s} \\
&= \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P'_{pq_s}\cos(\phi) \\
V_{r\phi_{pq}} = V_{\phi r_{pq}} &= \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)\cos(\phi)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P'_{pq_s}
\end{aligned}
\tag{A.9}
$$

$$\begin{aligned}\frac{d^2}{d\lambda^2}V_{pq} = \frac{d}{d\lambda}V_{\lambda_{pq}} &= \frac{GM}{R}q\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\cos(q\lambda)(q) - S_{pq}\sin(q\lambda)(q))\frac{d}{d\phi}P_{pq_s} \\ V_{\lambda\lambda_{pq}} &= \frac{GM}{R}(-q^2)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq_s}\end{aligned} \tag{A.10}$$

$$\begin{aligned}\frac{d^2}{d\lambda d\phi}V_{pq} = \frac{d}{d\lambda}V_{\phi_{pq}} &= \frac{GM}{R}\cos(\phi)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda)(q) + S_{pq}\cos(q\lambda)(q))P'_{pq_s} \\ V_{\lambda\phi_{pq}} = V_{\phi\lambda_{pq}} &= \frac{GM}{R}q\cos(\phi)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda))P'_{pq_s}\end{aligned} \tag{A.11}$$

$$\begin{aligned}\frac{d^2}{d\phi^2}V_{pq} = \frac{d}{d\phi}V_{\phi_{pq}} &= \frac{GM}{R}\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))\frac{d}{d\phi}(\cos(\phi)P'_{pq_s}) \\ V_{\phi\phi_{pq}} &= \frac{GM}{R}\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))(P''_{pq_s}\cos(\phi)^2 - P'_{pq_s}\sin(\phi))\end{aligned} \tag{A.12}$$

A.1.4. Gravity Gradient Tensor and Vector

Gravity gradient is commonly arranged in the form of a matrix tensor $\boldsymbol{V}_{uu_{pq}}$ as:

$$\boldsymbol{V}_{uu_{pq}} = \begin{bmatrix} V_{rr_{pq}} & V_{r\lambda_{pq}} & V_{r\phi_{pq}} \\ V_{\lambda r_{pq}} & V_{\lambda\lambda_{pq}} & V_{\lambda\phi_{pq}} \\ V_{\phi r_{pq}} & V_{\phi\lambda_{pq}} & V_{\phi\phi_{pq}} \end{bmatrix} \tag{A.13}$$

where it is important to note that, as shown by the derived equations for each individual element in the matrix, the gravity gradient tensor is a symmetric matrix. This means that there are only 6 unique elements. These elements can be arranged into a gravity gradient vector $\vec{V}_{uu_{pq}}$, as:

$$\vec{V}_{uu_{pq}} = \begin{bmatrix} V_{rr_{pq}} \\ V_{r\lambda_{pq}} \\ V_{r\phi_{pq}} \\ V_{\lambda\lambda_{pq}} \\ V_{\lambda\phi_{pq}} \\ V_{\phi\phi_{pq}} \end{bmatrix}^T \tag{A.14}$$

A.1.5. Gravity Gradient Derivatives

Differentiation is conducted to find the third derivatives of gravity potential. In this case, the gravity gradient spatial derivative vector $\vec{V}_{uu_{pq}}$ is:

$$\vec{V}_{uuu_{pq}} = \begin{bmatrix} \frac{GM}{R}\left(-\frac{1}{r^3}\right)(p+1)(p+2)(p+3)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda))P_{pq_s} \\ \frac{GM}{R}q\left(\frac{1}{r^2}\right)(p+1)(p+2)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda)+S_{pq}\cos(q\lambda))P_{pq_s} \\ \frac{GM}{R}\cos(\phi)\left(\frac{1}{r^2}\right)(p+1)(p+2)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda))P'_{pq_s} \\ \frac{GM}{R}q^2\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\cos(q\lambda)-S_{pq}\sin(q\lambda))P_{qr_s} \\ \frac{GM}{R}q\cos(\phi)\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda)+S_{pq}\cos(q\lambda))P'_{qr_s} \\ \frac{GM}{R}\left(-\frac{1}{r}\right)(p+1)\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda))(\cos(\phi)^2P''_{pq_s}-\sin(\phi)P'_{pq_s}) \\ \frac{GM}{R}q^3\left(\frac{R}{r}\right)^{p+1}(C_{pq}\sin(q\lambda)-S_{pq}\cos(p\lambda))P_{pq_s} \\ \frac{GM}{R}q^2\cos(\phi)\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\cos(q\lambda)-S_{pq}\sin(p\lambda))P'_{pq_s} \\ \frac{GM}{R}q\left(\frac{R}{r}\right)^{p+1}(-C_{pq}\sin(q\lambda)+S_{pq}\cos(q\lambda))(\cos(\phi)^2P''_{pq_s}-\sin(\phi)P'_{pq_s}) \\ \frac{GM}{R}q\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda))(\cos(\phi)^2P''_{pq_s}-\sin(\phi)P'_{pq_s}) \end{bmatrix}^T \tag{A.15}$$

A.1.6. Factorisation

As shown in Section 2.2, each element in the derivations for $\vec{V}_{uu}$ and $\vec{V}_{uuu}$ is independent to r, λ and ϕ. This means that it can be divided into separate common factors. The factors corresponding to each coordinate state are arranged into vectors $\vec{F}$:

$$\vec{F}^r = \begin{bmatrix} \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \end{bmatrix} \tag{A.16}$$

$$\vec{F}^\lambda = \begin{bmatrix} C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda)+S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda)+S_{pq}\sin(q\lambda)) \end{bmatrix} \tag{A.17}$$

$$\vec{F}^\phi = \begin{bmatrix} P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2P''_{pq}(\sin(\phi))-\sin(\phi)P'_{pq}(\sin(\phi)) \end{bmatrix} \tag{A.18}$$

The $\vec{F}$ vectors are repeated into a matrix for each element in the gravity gradient vector $\vec{V}_{uu}$ (which is 6). This is useful for steps that come after this one. The following equations show the steps to accomplish this:

$$\vec{Q} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T \tag{A.19}$$

$$\vec{Q}\vec{F}^r = \begin{bmatrix} \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \end{bmatrix}, \tag{A.20}$$

$$\vec{Q}\vec{F}^{\lambda} = \begin{bmatrix} C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \end{bmatrix}, \tag{A.21}$$

$$\vec{Q}\vec{F}^{\phi} = \begin{bmatrix} P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) & \cos(\phi)P'_{pq}(\sin(\phi)) & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \end{bmatrix} \tag{A.22}$$

The exponential matrices determine which version of each factor is used for each gravity gradient component, for example, V_{rr}, which has the second differential wit respect to r, has an E^r segment of $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$, but has $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ as segment for E^{λ} and E^{ϕ}.

$$\boldsymbol{E}^r = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{A.23}$$

$$\boldsymbol{E}^{\lambda} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{A.24}$$

$$\boldsymbol{E}^{\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.25}$$

$$(\vec{Q}\vec{F}^r)^{E^r} = \begin{bmatrix} 1 & 1 & (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ 1 & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & 1 \\ 1 & (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} & 1 \\ \left(\frac{R}{r}\right)^{p+1} & 1 & 1 \\ \left(\frac{R}{r}\right)^{p+1} & 1 & 1 \\ \left(\frac{R}{r}\right)^{p+1} & 1 & 1 \end{bmatrix} \tag{A.26}$$

$$(\vec{Q}\vec{F}^{\lambda})^{E^{\lambda}} = \begin{bmatrix} C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & 1 & 1 \\ 1 & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & 1 \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & 1 & 1 \\ 1 & 1 & -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ 1 & q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) & 1 \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) & 1 & 1 \end{bmatrix} \tag{A.27}$$

$$(\vec{Q}\vec{F}^{\phi})^{E^{\phi}} = \begin{bmatrix} P_{pq}(\sin(\phi)) & 1 & 1 \\ P_{pq}(\sin(\phi)) & 1 & 1 \\ 1 & \cos(\phi)P'_{pq}(\sin(\phi)) & 1 \\ P_{pq}(\sin(\phi)) & 1 & 1 \\ 1 & \cos(\phi)P'_{pq}(\sin(\phi)) & 1 \\ 1 & 1 & \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \end{bmatrix} \tag{A.28}$$

With the QF^E matrices, the individual parts for each gravity gradient component are computed:

$$\Pi_{j=1}^{j^r}(\vec{Q}\vec{F}_j^r)^{\mathbf{E}^r} = \begin{bmatrix} (p+1)(p+2)\left(\frac{1}{r^2}\right)\left(\frac{R}{r}\right)^{p+1} \\ (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} \\ (p+1)\left(-\frac{1}{r}\right)\left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} \\ \left(\frac{R}{r}\right)^{p+1} \end{bmatrix} \tag{A.29}$$

$$\Pi_{j=1}^{j^{\lambda}}(\vec{Q}\vec{F}_j^{\lambda})^{\mathbf{E}^{\lambda}} = \begin{bmatrix} C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) \\ q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) \\ -q^2(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda)) \\ q(-C_{pq}\sin(q\lambda) + S_{pq}\cos(q\lambda)) \\ C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda) \end{bmatrix} \tag{A.30}$$

$$\Pi_{j=1}^{j^{\phi}}(\vec{Q}\vec{F}_j^{\phi})^{\mathbf{E}^{\phi}} = \begin{bmatrix} P_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) \\ \cos(\phi)P'_{pq}(\sin(\phi)) \\ P_{pq}(\sin(\phi)) \\ \cos(\phi)P'_{pq}(\sin(\phi)) \\ \cos(\phi)^2 P''_{pq}(\sin(\phi)) - \sin(\phi)P'_{pq}(\sin(\phi)) \end{bmatrix} \tag{A.31}$$

A.2. GSF and LNOF relation

??

GSF is a geocentric frame and LNOF is a local frame, with basis (i_1, i_2, i_3) and (e_1, e_2, e_3) respectively. Assuming Cartesian state vector $\vec{x} = [x, y, z]^T$, and spherical state vector $\vec{u} = [\phi, \lambda, r]$, their coordinates relate as:

$$\begin{aligned} x &= r\cos(\phi)\sin(\lambda) \\ y &= r\cos(\phi)\cos(\lambda) \\ z &= r\sin(\phi) \end{aligned} \tag{A.32}$$

The covariant of the local basis e_n is given in terms of the geocentric basis:

$$\vec{e}_n = i_m \frac{x^m}{u^n} \tag{A.33}$$

The composition of $\frac{dx}{du}$ is shown in:

$$\frac{d\vec{x}}{d\vec{u}} = \begin{bmatrix} -r\sin(\phi)\cos(\lambda) & -r\cos(\phi)\sin(\lambda) & \cos(\phi)\cos(\lambda) \\ -r\sin(\phi)\sin(\lambda) & r\cos(\phi)\cos(\lambda) & \cos(\phi)\sin(\lambda) \\ r\cos(\phi) & 0 & \sin(\phi) \end{bmatrix} \tag{A.34}$$

The inner products of the basis expressed this way is defined as:

$$g_{nm} = \vec{e}_n \cdot \vec{e}_m \tag{A.35}$$

$$\begin{aligned} e_n &= i_m \frac{dx^m}{du^n} \\ e_1 &= -r\sin(\phi)\cos(\lambda)i_1, -r\sin(\phi)\sin(\lambda)i_2, r\cos(\phi)i_3 \\ e_2 &= -r\cos(\phi)\sin(\lambda)i_1, r\cos(\phi)\sin(\lambda)i_2, 0i_3 \\ e_3 &= \cos(\phi)\sin(\lambda)i_1, \cos(\phi)\sin(\lambda)i_2, \sin(\phi)i_3 \end{aligned} \tag{A.36}$$

$$g_{nm} = \begin{bmatrix} r^2 & 0 & 0 \\ 0 & r^2\cos(\phi)^2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.37}$$

With the inverse being:

$$g^{nm} = g_{nm}^{-1} = \begin{bmatrix} \frac{1}{r^2} & 0 & 0 \\ 0 & \frac{1}{r^2\cos(\phi)^2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.38}$$

The Chistoffel symbols of the second kind are:

$$
\begin{aligned}
\Gamma^k_{ij} &= \frac{1}{2} g^{kl} \left(\frac{dg_{jl}}{du^i} + \frac{dg_{il}}{du^j} - \frac{dg_{ij}}{du^l} \right) \\
\Gamma^1_{11} &= \frac{1}{2}\frac{1}{r^2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^1_{12} &= \frac{1}{2}\frac{1}{r^2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^1_{13} &= \frac{1}{2}\frac{1}{r^2}(2r) + \frac{1}{2}(0) + \frac{1}{2}(0) = \frac{1}{r} \\
\Gamma^1_{22} &= \frac{1}{2}\frac{1}{r^2}(2r^2 \cos(\phi) \sin(\phi)) + \frac{1}{2}(0) + \frac{1}{2}(0) = \cos(\phi) \sin(\phi) \\
\Gamma^1_{23} &= \frac{1}{2}\frac{1}{r^2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^1_{33} &= \frac{1}{2}\frac{1}{r^2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^2_{11} &= \frac{1}{2}(0) + \frac{1}{2}\left(\frac{1}{r^2 \cos(\phi)^2} \right)(0) + \frac{1}{2}(0) = 0 \\
\Gamma^2_{12} &= \frac{1}{2}(0) + \frac{1}{2}\left(\frac{1}{r^2 \cos(\phi)^2} \right)(-2r^2 \cos(\phi) \sin(\phi)) + \frac{1}{2}(0) = -\frac{\sin(\phi)}{\cos(\phi)} \\
\Gamma^2_{13} &= \frac{1}{2}(0) + \frac{1}{2}\left(\frac{1}{r^2 \cos(\phi)^2} \right)(0) + \frac{1}{2}(0) = 0 \\
\Gamma^2_{22} &= \frac{1}{2}(0) + \frac{1}{2}\left(\frac{1}{r^2 \cos(\phi)^2} \right)(0) + \frac{1}{2}(0) = 0 \\
\Gamma^2_{23} &= \frac{1}{2}(0) + \frac{1}{2}\left(\frac{1}{r^2 \cos(\phi)^2} \right)(2r \cos(\phi)^2) + \frac{1}{2}(0) = \frac{1}{r} \\
\Gamma^2_{33} &= \frac{1}{2}(0) + \frac{1}{2}\left(\frac{1}{r^2 \cos(\phi)^2} \right)(0) + \frac{1}{2}(0) = 0 \\
\Gamma^3_{11} &= \frac{1}{2}(0) + \frac{1}{2}(0) + \frac{1}{2}(-2r) = -r \\
\Gamma^3_{12} &= \frac{1}{2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^3_{13} &= \frac{1}{2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^3_{22} &= \frac{1}{2}(0) + \frac{1}{2}(0) + \frac{1}{2}(-2r \cos(\phi)^2) = -r \cos(\phi)^2 \\
\Gamma^3_{23} &= \frac{1}{2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0 \\
\Gamma^3_{33} &= \frac{1}{2}(0) + \frac{1}{2}(0) + \frac{1}{2}(0) = 0
\end{aligned}
\tag{A.39}
$$

$$\begin{aligned}
\Gamma^1_{mn} &= \begin{bmatrix} 0 & 0 & \frac{1}{r} \\ 0 & \cos(\phi)\sin(\phi) & 0 \\ \frac{1}{r} & 0 & 0 \end{bmatrix} \\
\Gamma^2_{mn} &= \begin{bmatrix} 0 & -\frac{\sin(\phi)}{\cos(\phi)} & 0 \\ -\frac{\sin(\phi)}{\cos(\phi)} & 0 & \frac{1}{r} \\ 0 & \frac{1}{r} & 0 \end{bmatrix} \\
\Gamma^3_{mn} &= \begin{bmatrix} -r & 0 & 0 \\ 0 & -r\cos(\phi)^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\end{aligned} \tag{A.40}$$

With the use of the Equation A.40, the derivative of the local basis e_m are given by:

$$\frac{de_m}{du^n} = \Gamma^s_{mn} e_s \tag{A.41}$$

The fist partial derivatives of gravity potential (gravity acceleration) are expressed as:

$$\begin{aligned}
\nabla V &= e^m \frac{dV}{du^m} \\
\nabla &= e^m \frac{d}{du^m}
\end{aligned} \tag{A.42}$$

Knowing the expression for the operator ∇, the second derivatives of gravity potential (gravity gradient) are expressed as:

$$\begin{aligned}
\nabla V &= \frac{dV}{du^n} e^m \\
\nabla &= e^m \frac{d}{du^m} \\
\nabla\nabla V &= e^m \frac{d}{du^m}\left(\frac{dV}{du^n} e^n\right) \\
&= \left(\frac{d^2V}{du^m du^n} + \frac{dV}{du^n}\frac{de^m}{du^m}\right) e^m \\
&= \left(V_{mn} - \Gamma^s_{mn} V_s\right) e^m e^n
\end{aligned} \tag{A.43}$$

Each local basis e^m has a normalised equivalent $\hat{e}^m$ where $\frac{1}{r^2}$, $\frac{1}{r^2\cos(\phi)^2}$, 1 are the normaliser factors for $\hat{e}^1$, $\hat{e}^2$ and $\hat{e}^3$ respectively. With this, the transition from GSF gravity gradients to LNOF gravity gradients is:

$$\begin{aligned}
V_{xx} &= \frac{1}{r^2}\left(V_{\phi\phi} - \left(\Gamma^1_{11} V_\phi + \Gamma^2_{11} V_\lambda + \Gamma^3_{11} V_r\right)\right) \hat{e}^1 \hat{e}^1 \\
&= \frac{1}{r^2} V_{\phi\phi} + \frac{1}{r} V_r
\end{aligned} \tag{A.44}$$

$$\begin{aligned} V_{yy} &= \frac{1}{r^2 \cos(\phi)^2} \left(V_{\lambda\lambda} - \left(\Gamma^1_{22} V_\phi + \Gamma^2_{22} V_\lambda + \Gamma^3_{22} V_r \right) \right) \hat{e}^2 \hat{e}^2 \\ &= \frac{1}{r^2 \cos(\phi)^2} V_{\lambda\lambda} - \frac{\sin(\phi)}{r^2 \cos(\phi)} V_\phi + \frac{1}{r} V_r \end{aligned} \tag{A.45}$$

$$\begin{aligned} V_{zz} &= \left(V_{rr} - \left(\Gamma^1_{33} V_\phi + \Gamma^2_{33} V_\lambda + \Gamma^3_{33} V_r \right) \right) \hat{e}^3 \hat{e}^3 \\ &= V_{rr} \end{aligned} \tag{A.46}$$

$$\begin{aligned} V_{xy} &= \frac{1}{r^2 \cos(\phi)} \left(V_{\lambda\phi} - \left(\Gamma^1_{12} V_\phi + \Gamma^2_{12} V_\lambda + \Gamma^3_{12} V_r \right) \right) \hat{e}^1 \hat{e}^2 \\ &= \frac{1}{r^2 \cos(\phi)} V_{\lambda\phi} + \frac{\sin(\phi)}{r^2 \cos(\phi)^2} V_\lambda \end{aligned} \tag{A.47}$$

$$\begin{aligned} V_{xz} &= \frac{1}{r} \left(V_{r\phi} - \left(\Gamma^1_{13} V_\phi + \Gamma^2_{13} V_\lambda + \Gamma^3_{13} V_r \right) \right) \hat{e}^1 \hat{e}^3 \\ &= \frac{1}{r} V_{r\phi} + \frac{1}{r^2} V_\phi \end{aligned} \tag{A.48}$$

$$\begin{aligned} V_{yz} &= \frac{1}{r \cos(\phi)} \left(V_{r\lambda} - \left(\Gamma^1_{23} V_\phi + \Gamma^2_{23} V_\lambda + \Gamma^3_{23} V_r \right) \right) \hat{e}^2 \hat{e}^3 \\ &= \frac{1}{r \cos(\phi)} V_{r\lambda} + \frac{1}{r^2 \cos(\phi)} V_\lambda \end{aligned} \tag{A.49}$$

A.3. Star Tracker Quaternion Processing

Quaternions are 4 numbers which represent the satellite attitude, and are the result of data collected by star trackers. Rotation in the GOCE data files is represented by quaternions, since GOCE makes use of star trackers, and they are used to derive a rotation matrix as:

$$\boldsymbol{R}(q) = \begin{bmatrix} 2(q_0^2 + q_1^2) - 1 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & 2(q_o^2 + q_2^2) - 1 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & 2(q_0^2 + q_3^2) - 1 \end{bmatrix} \tag{A.50}$$

Equation A.50 forms part of the transformation from the gradiometer reference frame (GSF) to the inertial reference frame (IRF), as seen in Figure 5.1. The actual transformation for the GOCE data is provided in the EGG_NOM_1b levell 1 data file. To transform from IRF to the Earth centred Earth fixed frame (ECEF), Earth's rotation, precession, polar motion and nutation are needed. These transformations are not covered in this thesis, but they are a significant aspect of an autonomous navigation system when using gravity gradiometry.

B

Programming Details

The program can be found in 2 parts. A python part, which is used for orbit simulation, data generation and plotting. A rust part, which is used for the main algorithm and all the tests. The rust code is provided in a repository [25]
Each one of them has a README to be used for instructions on the installation and execution of the code.

C

Literature Study

C.1. Introduction

The contents of the literature study introduction are discussed in Section 1.1.

C.2. Literature Research

Traditionally, gravity gradient orbit determination has been aided by external sources, but as stated by Anthony DeGregoria [26], these are inconvenient, slower and much less reliable than the potential that gravity gradiometry has for autonomous guidance. It also shows a great potential for navigation algorithms fully dependant on gravity gradients. It deals with map matching, but it shows the improvement it has when a Kalman filter is applied to the algorithm. Low altitude bodies such as aeroplanes or satellites can greatly benefit from this concept, which is limited by the accuracy for making Earth gravity models and measuring gravity gradient.

Further research also reveals that gravity gradiometry aided navigation can yield results with the same accuracy as GPS aided navigation. Justin A. Richeson et al. [27] shows that a gravity gradiometry independent method can yield errors smaller even than GPS. The only limitation with this method, is the size and mass of the gradiometers available when published, which were of a mass of approximately $280[kg]$. Upon more investigation, it is also found out that according to a design by S. Weiner et al. [28], new gradiometers are capable of being less than $70[kg]$, while still providing optimal results. This statement is done based on a cold-atom gradiometer design, of which its flaws will be commented later. Nevertheless, it is clear that the shrinking of technology will inevitably make this a viable option.

An important addition to the literature research is that most of this field has recently been highly researched by Chinese sources. Most of them are extremely useful and serve as a source of inspiration for the goal of this research. An example of these methods is given by X. Sun et al. [23], in which an extended version of the Kalman filter, the unscented least squares (USL) filter is developed, which will most likely be used as a basis for this research later.

Gravity Field Map

The concept of a gravity field has been part of physics for a long time. Isaac Newton published the law of gravitational attraction in 1687, in which a simple equation describes what was then then considered to be the gravitational potential of Earth. This can be seen in Equation C.1, where V represents gravitational potential, G represents Earth's gravitational constant, M represents its mass, and r the radius from the centre of mass at which the potential is being measured.

$$V = -G\frac{M}{r} \tag{C.1}$$

The problem with this simple equation was that it could not work unless Earth was perfectly uniform in density and a perfect spherical shape. This is not the case. Taking the previous equation, it can be turned into Equation C.2, where x, y, z the dimensions of volume, and ρ is density as a function of the coordinates. This is to show that density does differ, thus gravity field is not uniform.

$$V = -G\int\int\int \frac{\rho(x,y,z)}{r(z,y,z}) dxdydz \tag{C.2}$$

T. M. MacRobert [29] explains how in 1785, Laplace published a memoir his own concept for gravitational potential. Laplace described potential as a spherical harmonic function. A function composed of layers of functions with degree and order. Legendre, with who he had been in contact with, was thus led to the discovery of what we now know today as the associated Legendre functions, by which gravitational spherical harmonics is expressed nowadays. Equation C.3 shows how gravity is expressed in terms of spherical harmonics, with p being degree and q being order.

$$V = \frac{GM}{R}\sum_{p=0}^{\infty}\sum_{q=0}^{p}\left(\frac{R}{r}\right)^{p+1}(C_{pq}\cos(q\lambda) + S_{pq}\sin(q\lambda))P_{pq}(\sin(\phi)) \tag{C.3}$$

Degree and power is what defines every component of gravity field. The associated Legendre function P_{pq}, the sine coefficient S_{pq} and the cosine coefficient C_{pq} define the shape of each component and the values corresponding to the specific spherical coordinate. Figure C.1 shows how degree and order components work.

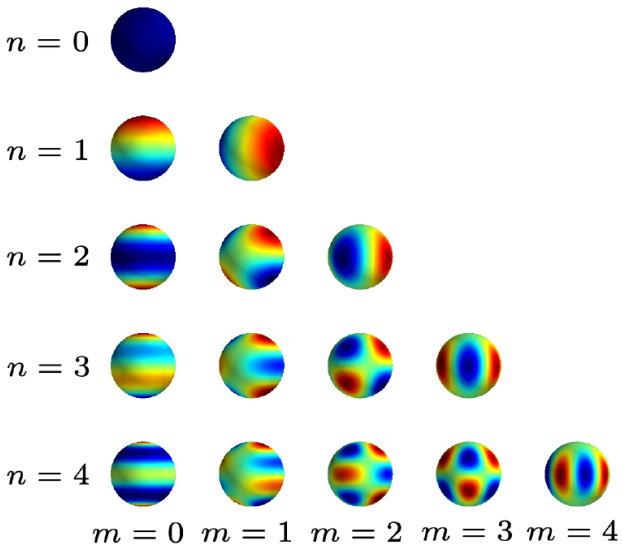

Figure C.1: Visualisation of how degree and order works on gravity field as shown by J. Hollebon et al. [30], in which n is degree and m is order.

It is also vital to know about the best gravity field models of Earth there are. This is in order to know to what extent is gravity gradiometry reliable at this time, and thus feasible. It is important to learn what are the most advanced sources for these models. the accuracy of spherical harmonic models is based on the accuracy of the coefficients C_{pq} and S_{pq} and how much degree and power can be attained.

The latest verified model computed by Philipp Zingerle et al. [31] goes up to degree and order 5540. This is obtained by a combined calculation of data taken by satellite, as well as ground stations. Despite this, the pure satellite data with combinations only goes up to degree and order 716, with GOCE data alone not surpassing degree and order 300.

GOCE (Gravity field and Ocean Circulation Explorer) [32] was a satellite mission in Low Earth Orbit (LEO) with the purpose of mapping Earth's gravity field and getting insights into its interior structure and ocean depths. This mission is a very significant contribution to gravity field models and the functionality of gravity gradiometers as a viable instrument in space.

A way to verify that GOCE data does not surpass degree and order 300 is by looking at a report written by the the European GOCE Gravity Consortium [33]. It can be observed that data processed from GOCE is limited to degree and order 300. Because the models observed are only to be used as a reference of what a gradiometer measures, taking degree and order 300 is the best way to evaluate the present state. On the other hand, it is also convenient to study higher degree and orders such as 716 and 5540 to demonstrate the feasibility of orbit determination by gravity gradiometry in the future.

It is also important to point out that the gravity model of Earth is constantly changing. Srinivas Bettadpur [34] shows the dependency of the gravity model, and how it dynamically changes depending on factors present on our planet such as Earth tides, ocean tides, atmospheric variation, and rotational deformation. This is a possible set-back when verifying a gravity gradiometry method in real time, as the field would be constantly shifting. But as a

way to develop and test a method, this doesn't have to be taken into account. The assumption is made that a static model of Earth is sufficient. When looking at further possibilities like the Moon and Mars, their field is not dynamic. Thus, it does not face these problems. On top of this, the Moon and Mars cannot rely on GNSS as much as Earth. Relay satellites are a possibility, but Mars orbiters would have a significant time delay from all the signals which have to be sent back and forth and its distance. Lunar orbiters are unable to use navigation on the dark side of the moon.

To validate this, it can be seen from official NASA sources [35] [36], that the Moon's gravity field model goes up to degree and order 1200 and Mars' model goes up to degree and order 110, all while both are static and reliable gravity field maps.

This is a vital piece of information. It indicates the best usage for gravity gradient satellite navigation. With it's usage being a navigation tool for bodies in which GNSS navigation is not available or reliable enough.

Gradiometers

In this section, the functionality and management of accelerometers and their subsequent usage as gradiometers will be discussed.

The original concept of a gradiometer was a torsion balance. It was proposed ad made by R. Eotvos [37]. It consists on 2 test masses hanging on a beam. Their acceleration is measured by the torsion of the fibre the beam is being carried on. In 1981, the first practical application was proposed by G. Balmino et al. [38]. It suggests the usage of gradiometry in opposition to satellite-to-satellite tracking. It states it to be cheaper and less complex. And thus, with the advancement of technology, gradiometers are improving their accuracy every year too.

Accuracy is key to know the position of a satellite in orbit. To determine the degree of accuracy in which the position can be obtained, the accuracy of the gradiometers must be known too. This is because the magnitude of errors in these devices greatly affects the computed state.

According to C. Siemes et al. [39], the gravity gradiometry data can be calibrated and improved over GOCE's data. This yields errors in gravity gradient ranging from $6[mE]$ for V_{xx} and V_{yy} to $12[mE]$ for V_{zz} and V_{xz} on band-passed readings from $1 - 10[mHz]$ (E stands for the unit of measurement Eotvos, which is equivalent to $10^{-9}[s^{-2}]$). Despite these being optimal results, these do not represent reality, as they have already been calibrated to a preset data collected by GOCE. The accuracy that needs to be known is the inherent accuracy of the gradiometers at their moment of use, to be able to calculate the position while in orbit.

Electrical Gradiometers

Electrical gradiometers have been the most commonly used devices to measure gravity field. They work by making use of electrostatic accelerometers. According to P. Touboul et al. [40], the principle of an electrostatic accelerometer lies on the movement of a proof mass within a sensor cage. When making use of accelerometers within Earth, the proof mass is linked to the cage by a mechanical stiffness. But for space applications, the proof mass is

not linked in any of the 3 dimensions. This makes it possible to make the accelerometer work in 3 dimensions and yield higher resolution results. This proof mass is kept passively motionless within the cage. On each axis, a pair of electrodes is kept at either side of the proof mass in order to sense position by the use of a capacitive sensor. The difference in signal can later be computed in order to get the position of the poof mass and subsequently, its acceleration. This can be seen visually in Figure C.2.

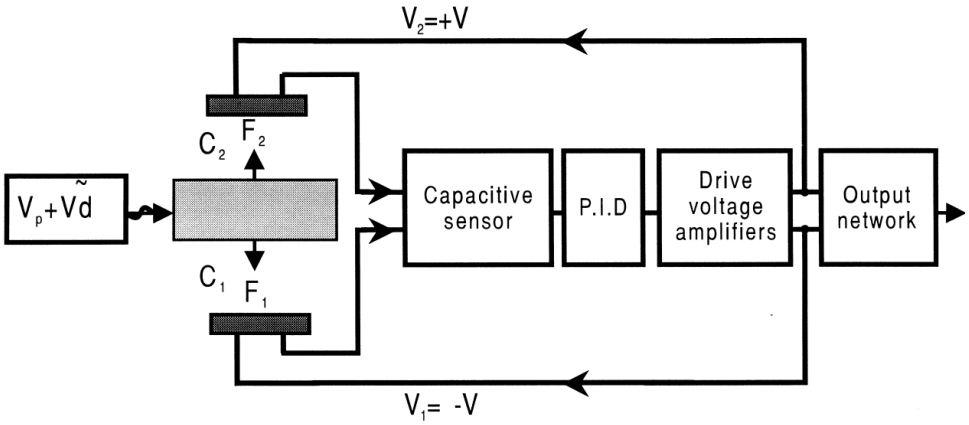

Figure C.2: Graph of how an electrostatic accelerometer works according to P. Touboul et al. [40].

There has been a steady advancement in gradiometer technology within the last 25 years alone. D. DiFrancesco et al. [21] shows major gradiometer advancements up to 2009. It indicates how, gradiometers for functions that require a much lower resolution (ranging within $1[E/Hz^{0.5}]$) are less prone to error. Any small change and shift in mass on Earth is a change and shift in the gravity field. This means that The larger the resolution, the more dynamic and small errors are detected. A satellite mission such as GOCE, which is made for the purpose of satellite gradient readings, has a gradiometer of an accuracy ranging in $0.004[E/Hz^{0.5}]$. The gradiometer in GOCE is used explicitly for this purpose and thus has its data processed after. This implies that readings with this amount of accuracy may not be as reliable when depending on the gradiometer readings alone.

University of Maryland has a design for Gradiometer, as seen from C. E. Griggs et al. [41], which can reach an accurate as low as $1.4 \cdot 10^{-4}[E/Hz^{0.5}]$ between $1 - 50[mHz]$ and $2 \cdot 10^{-5}[E/Hz^{0.5}]$ between $0.1 - 1[mHz]$. According to X. Sun et al [23] gradiometers can have a noise of down to $0.1[E]$. This was found at a specific frequency from J. A. Richeson et al. [27]

The most useful example for an electrical gradiometer is the Electrostatic Gravity Gradiometer (EGG), developed for the GOCE mission [42]. M. R. Drinkwater et al. [43] specify the performance characteristics of this gradiometer. It makes use of a measurement bandwidth of $5 \cdot 10^{-3}[Hz]$ to $0.1[Hz]$ with an error margin of $3 \cdot 10^{-3}[E/Hz^{0.5}]$. Z. Zhu et al. [44] states that GOCE's accuracy was of $1 \cdot 10^{-2}[E/Hz^{0-5}]$ to $2 \cdot 10^{-2}[E/Hz^{0-5}]$.

D. DiFrancesco et al. [21] also touches upon possible error and noise sources for gradiometers. It mentions standard instrument noise such as electronic noise, eddy currents, temperature, pressure and humidity variations and geologic noise. These come from the set up of the equipment and the vehicle itself, as well as other external accelerations. But the most vital types of errors which affect gradiometers are external inputs. Errors will be

discussed later on.

Quantum Gradiometers
There is a second way o measuring the gravity field of Earth. This is by making use of quantum gradiometers. This is technology which has a substantial amount of potential promise. It was first suggested by H. A. Chan and H. J. Paik [45]. The model proposed makes use of a Superconducting Quantum Interference Device (SQUID). The principle by which it works is a superconductive mass being attached to a weak spring. This spring lets the mass move enough with acceleration changes. The conversion of acceleration into a signal is caused by the transducer, made from sensing coils which generate current and magnetic flux due to the mass movement. The current is detected by its the SQUID and its voltage amplified. Figure C.3 shows how this works. This version, however, is plagued with calibration parameters and noises.

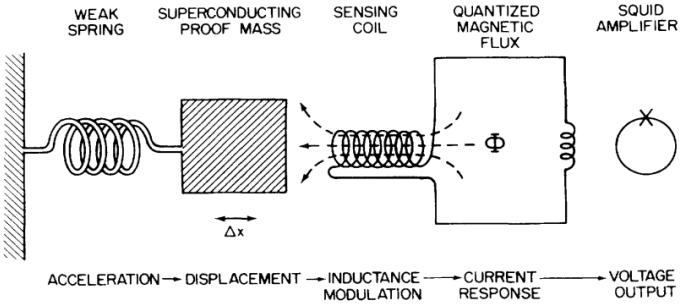

Figure C.3: Graph showing how a quantum gradiometer works based on SQUID technology [45].

J. Zhong et al. [46] suggests a more modern design called Atom Gravity Gradiometers (AGG). It states that they are highly immune to any other acceleration noises. W. Lyu et al. [47] explains how the design works through the use of Light-Pulsed Atom Interferometer (LPAI). Within each interferometer, a cold atom cloud is split, moved and recombined by laser pulses. Excited atoms can be imprinted by the phase of the laser pulses. This phase changes with gravity, hence giving signals when there is any acceleration change.

However, according to the latest advancements in these devices according to B. Stray et al. [48], the precision is $20[E]$, which is way too high to even consider it viable in comparison to the $0.1[E]$ observed from the electronic gradiometers. AGGs also seem to be mostly used for other purposes other than space. As seen from J. Vovrosh et al. [49], they are mostly used for Earth mapping, archaeology, hydrology and earthquake detection among many others. This unfortunately makes them less likely to be used in the near future. Despite this, it is a topic to be encouraged to advance in, as their designs show promise.

Errors and Calibration Parameters
Both internal and external factors have a significant effect on the results of orbit position determination. Because of this, it is vital to have knowledge on what these factors are, how they function ad how large they are.

External Noises
From X. Sun et al. [50], the most significant external factors are deduced. These can potentially alter the measurements.

- **The centrifugal acceleration**: represented by $\Omega^2\vec{r}_i$. It is the acceleration caused by the angular velocity in a radial direction to the motion. Ω is the angular velocity of the satellite and r_i is the distance between the accelerometer i and the COM (centre of mass) of the gradiomter.
- **The linear angular acceleration**: caused by an a change in angular velocity. This one is tangential to the rotating motion, and therefore normal to the centrifugal acceleration in direction. It is represented by $\dot{\Omega}\vec{r}_i$. $\dot{\Omega}$ is the rate of change of angular velocity.
- **Linear accelerations**: These accelerations affect the body without rotation. They do not depend on the positioning of the accelerometers. Thus, it can be assumed that they act on the COM of the gradiometer. This means that there is no need to multiply it by $\vec{r}_i$ or any other factor. They are represented by $\vec{d}$.
- **Tidal effect and others**: It affects the reading of the gravity gradient by only $0.1 \cdot 10^{-4}[E]$ according to X. Sun et al. [50]. It is thus not considered to be large enough to make any significant difference on the gravity gradiometry, and thus it can be assumed to be ignored. Other forces such as Coriolis are weaker than the tidal effect, and thus can be assumed out too.

These are the 3 main sources for noise in a gradiometer when wanting to evaluate the gravity field. The measurement perceived by the accelerometers within the gradiometer involves the gravity gradient tensor (GGT) $\boldsymbol{V}_{uu}$. The GGT is the second derivative of the gravity potential V with respect to the coordinates.
This, in turn creates the following equation (Equation C.4) for the perceived acceleration caught by an accelerometer

$$\vec{a}_i = \boldsymbol{\Omega}^2\vec{r}_i + \dot{\boldsymbol{\Omega}}\vec{r}_i - \boldsymbol{V}_{uu}\vec{r}_i + \vec{d} \tag{C.4}$$

A it can be observed in this equation, the only components exclusive to the accelerometers are the measured acceleration $\vec{a}_i$ and the distance from the gradiometer COM $\vec{r}_i$. $\boldsymbol{\Omega}$ and $\dot{\boldsymbol{\Omega}}$ can be obtained through attitude determination devices. This means that it is possible to find out $\boldsymbol{V}_{uu}$ through only 2 accelerometers or even 1 if the linear accelerations $\vec{d}$ are assumed to not be in place. This can be shown in Equation C.5.

$$\boldsymbol{V}_{uu} = \boldsymbol{\Omega}^2 + \dot{\boldsymbol{\Omega}} + \frac{\vec{d} + \vec{a}_i}{\vec{r}_i} \tag{C.5}$$

Calibration parameters and white noise
It can also be seen, from this same paper, that gradiometers contain systematic errors and parameters too.

- The first of them is covered by P. Touboul et a. [9] and is an uncertainty in the electrostatic gains/scale factors used in accelerometers to convert voltage to acceleration. It is represented by K in this case and it is a coefficient with acceleration. There are ways to avoid this but it is better not to assume this out, as it is a very common systematic error.
- There are two other errors which can also be a coefficient with acceleration. These two are errors caused by the accelerometer misalignment and the inter-axis coupling.

They are represented by E_{al} and E_{co} respectively in this case. E_{al} occurs as a product of an angular deviation from the ideal accelerometer arrangement. E_{co} occurs as a product of a deviation from the arranged axes the accelerometers would ideally be placed at.

- C. Stummer [10] talks about the quadratic factor error. This one appears from the transfer function when converting the accelerometer's voltage into acceleration to be measured. It is represented by $K2$, and it acts as a coefficient with a^2 (the real acceleration) to give a value.
- Accelerometer bias. Each accelerometer has a bias in each of its 3 axes. It is represented by b. This means that in a 3 axis gradiometer, there are 18 different biases to take into account. It is possible, to reduce it to 3, since ideally all accelerometers have the same orientation, and thus all biases facing the same axis can be summed up into a total gradiometer bias. This, on top of the fact that any misalignment and inter-axis coupling errors are already taken into account. A bias is a specific value that each accelerometer has due to any imperfection or drift within its mechanism. Unlike noise (explained after), it is not a random error, but instead a starting value which may drift with time. It is possible to assume a fixed value bias in theory, but it must be stated before yielding results.
- Systematic noise. This may come from the accelerometers, their measurements, their information transfer or any system which operates in between. Systematic noise happens in all 3 axes and it can be treated as white noise. It is represented by n. It is treated as a Gaussian Random variable (GRV) in order to implemented within calculations.

Equation C.6 shows the arrangement in which an accelerometer calculates acceleration.

$$\vec{a} = (K + E_{al} + E_{co}\vec{a}^{(real)} + K2\vec{a}^{(real)^2} + \vec{b} + \vec{n} \tag{C.6}$$

In here, a is the real acceleration, $\tilde{a}$ is the measured accelerometer acceleration.

From C. Siemes [51], it is seen that these factors can be easily cancelled out in real scenarios by making use of a proof-mass shaking and science mode procedure. During the science mode procedure according to C. Siemes et al. [52], a closed loop is implemented in which accelerometer data is read and ion thrusters are used to counteract the anomalies. Shaking mode procedure is different.

As said by B. Frommknecht [11], the first step is to get rid of non-linear errors, in this case $K2$. Figure C.4 shows how this signal is fed into the closed loop, with a gain G and a transfer function H. If the gain is affected by the quadratic term, a signal of a frequency of the sum and differences of the frequencies present in the input will appear. This way both the shaking input and this frequency are filtered. After this, a modulation appears as a result of high frequency pulses. The amplitude of this modulation is proportional to $K2$.

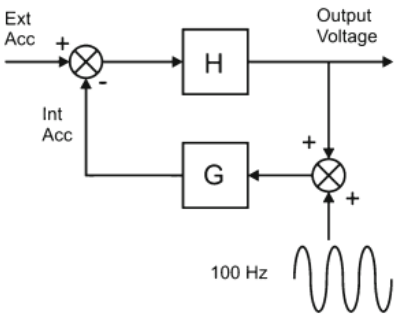

Figure C.4: Diagram of the high frequency signal fed on the control loop to determine the $K2$ error, given by B. Frommknecht et al. [11].

Secondly, cold gas thrusters are used to shake the spacecraft at high levels of linear and angular acceleration. This way it is far more significant than the gravity gradient signals. To explain this briefly, this allows to determine the parameters of the gradiometers without the shaking and helps distinguish each of the accelerations from each other by making them weakly correlated.

With this done, the proof-masses are positioned relative to their cages in order to zero out this error. The error which appears from this procedure is insignificant, and thus irrelevant, making it possible to ignore it. This procedure however, is done periodically throughout the early part of the spacecraft's orbit. This means that objects with a short flight time cannot apply this process. This will not be a problem for functioning satellites and long running vessels.

This leaves the vital and most important calibration parameters. The scale factor, unknown bias, and white noise from the equipment. By only taking these into account, the accelerometer measurement equation should look as shown in Equation C.7

$$\vec{a} = K\vec{a}^{(real)} + \vec{b} + \vec{n} \tag{C.7}$$

Attitude Determination Devices

Electric accelerometers have an output that depends on not only the gravity gradient, but other elements too. This can be seen in Equation C.4, now re-formulated as Equation C.8.

$$\vec{a}_i = (\boldsymbol{\Omega}^2 + \dot{\boldsymbol{\Omega}} - \boldsymbol{V}_{uu})\vec{r}_i + \vec{d} \tag{C.8}$$

In order to obtain the GGT, Ω needs to be known.

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{C.9}$$

The angular velocities ω are obtained by an attitude. In this case, the most commonly used technology and most accurate are Star Trackers. There is a large variation in star trackers. AI-based real-time star trackers are a relatively new and potential method to make use of star tracking. According to G. Carmeli et al. [53], the accuracy can be as low as

$0.01[deg]$ which is equivalent to $36[arcsec]$. On the other hand, much more developed and tested form of star trackers, such as the one tested by R. W. Bezooijen et al. [54], can have an accuracy of even $1[arcsec]$. For now, this seems like the most viable option to go with. The same can be seen with the APS autonomous star sensor from the Jena-Optronik company [55], in which random errors are smaller than $1[arcsec]$ and biases are smaller than $5[arcsec]$. This knowledge can also be used to know the angular velocity accuracy of a star tracker. T. Sun et al. [56] indicates that for an accuracy of $5[arcsec]$, the error in angular velocity would be of $0.0014[deg/s]$. Star trackers have an additional disadvantage of needing to be oriented towards the exterior of the satellite. This means that it is possible that the Star Sensor Reference Frame (SSRF) is different from the Gradiometer Reference Frame. This is the case for GOCE as shown by C. Siemes [51], in which each of the 3 star trackers had a reference frame of its own which needed to be rotated into the gradiometer frame.

Other devices exist which can provide attitude measurements. Magnetometers are devices most commonly used on Earth orbit. Using a similar method of measuring and mapping as we are trying to accomplish, but with the magnetic field of Earth. These subsequently then obtained the orientation of the satellite through this process. Magnetometers have been used to attitude determination in combination with attitude control. However, its accuracy is not as high as a star tracker. S. Carletta et al. [57] show in a recent article that it has an angular accuracy of $10[deg]$ and angular velocity accuracy of $0.2[deg/s]$. These values are too high in comparison with star trackers, and to even be able to get good data from gradiometers. On top of this, the article also states that by using an extended Kalman filter, it can reach an accuracy of $1[deg]$ and $0.01[deg/s]$, which are still not comparable values to a star tracker.

Magnetometers can also be coupled with solar sensors. According to the article by Q. Zhang et al. [58], while the angular rate accuracy does not change, the angular accuracy can be reduced to $0.3[deg]$. Despite this, angular rate is what this research is most interested on and thus it means that star trackers are still the best option.

By taking a look at the level of accuracy of each of these sensors, it is perfectly clear that star trackers hold a very significant advantage when it comes to measuring attitude. They were also used in GOCE, making them even more tested for complementing these kinds of missions.

Previous Work

In 2003, A. Bobojc e al. [1] made an initial suggestion for this satellite orbit determination using gradiometry. It was by what was then the future GOCE mission's gravity gradiometry observations. The paper shows an improvement in the orbit determination by means of the the use of gravity gradient. This is not enough and suggests that it is best to make this a joint process between the GPS observations of an orbit and the gradiometer readings. This would be done via a weighted method in order to optimise the errors. Of curse, this defeats the purpose of self sufficiency, which is the actual potential of gravity gradiometry in this case. It states the main issue for error is bias due to the limited gradiometer bandwidth.

In 2015, P. Chen et al. [2] proposes a method based on eigen-decomposition for GOCE data, in which errors are reduced. The overall process can be seen in Figure C.5. The data

taken is that of the star tracker, in order to get knowledge of the attitude of GOCE via star trackers, the gravity gradiometer, to obtain the gravity gradient and the Earth rotation model to be able to perform the correct rotations and handling data within axis.

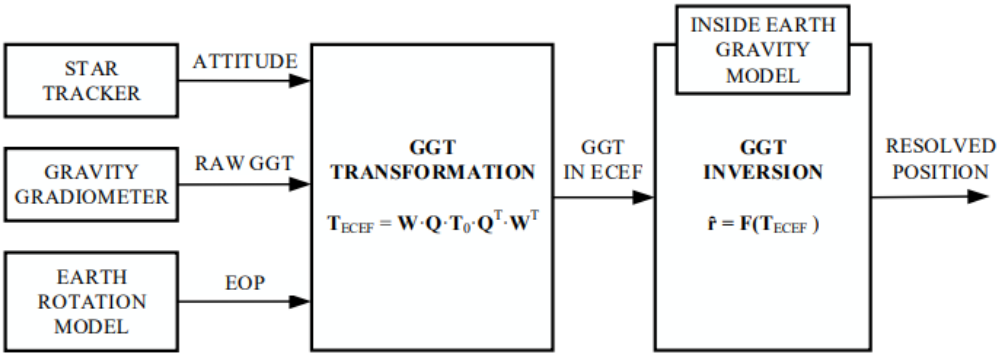

Figure C.5: Process by which results from P. Chen et al. [2] are obtained.

The results show an average position error of $120[m]$ and a velocity error of $0.125[m/s]$. While this is a method proven to work decently, it still cannot be applied to any practical cases. This is due to the fact that it does not take into account the accelerometer biases. Without this, when comparing to real world counterparts, errors are much larger.

In 2016, X. Sun et al. [50] do take into account calibration parameters such a bias, and make use of a batch least squares algorithm as a version of a Kalman filter design called the Unscented Kalman Filter (UKF) to perform orbit estimation. The algorithm used is one proposed by E. Park et al. [3] which propagates sigma points. Sigma points χ, according to S. J. Julier [59], are arbitrary state points used for a non-linear transformation of a vector. Instead of using a random Gaussian distribution for these points, they are chosen so that their sample weighted mean is the augmented state. The augmented state is a vector which contains both the state vector and the process noise vector. For a better insight on this process, Figure C.6 shows a full description of the process.

The results from this method yield position errors in the range of $10.8[m]$ to $1208.3[m]$ and velocity errors in the range of $0.013[m/s]$ to $1.2[m/s]$. While the lower range of errors is an accomplishment for a more realistic scenario like this one, in which calibration parameters are taken into account, the higher values indicate that one of the axes much more affected by errors than the other two. This is later changed by modifying the variance matrices of the biases for each accelerometer device by increasing the values corresponding to to the devices with the highest errors. The yy component is increased by a factor of 10 and the xy, yz components are increased by a factor of 100. This reduces the range to $10.4[m]$ to $677.0[m]$ and $0.012[m/s]$ to $0.80[m/s]$. This method to reduce the bias estimation errors is done in this case by knowing the results. Nevertheless, this could be effectively used by modifying the variance matrix components by accelerometer part depending on how large the errors are relative to each other after a certain epoch.

The 2016 publication also compared different gravity field models to EGM2008, the gravity model they used. They measured the differences between this one and EGM96, JGM3, EIGEN-6C4 and GGM05G. It shows great consistency in between them, with JGM3 having more significant errors on the along track and radial axes fro either position of velocity.

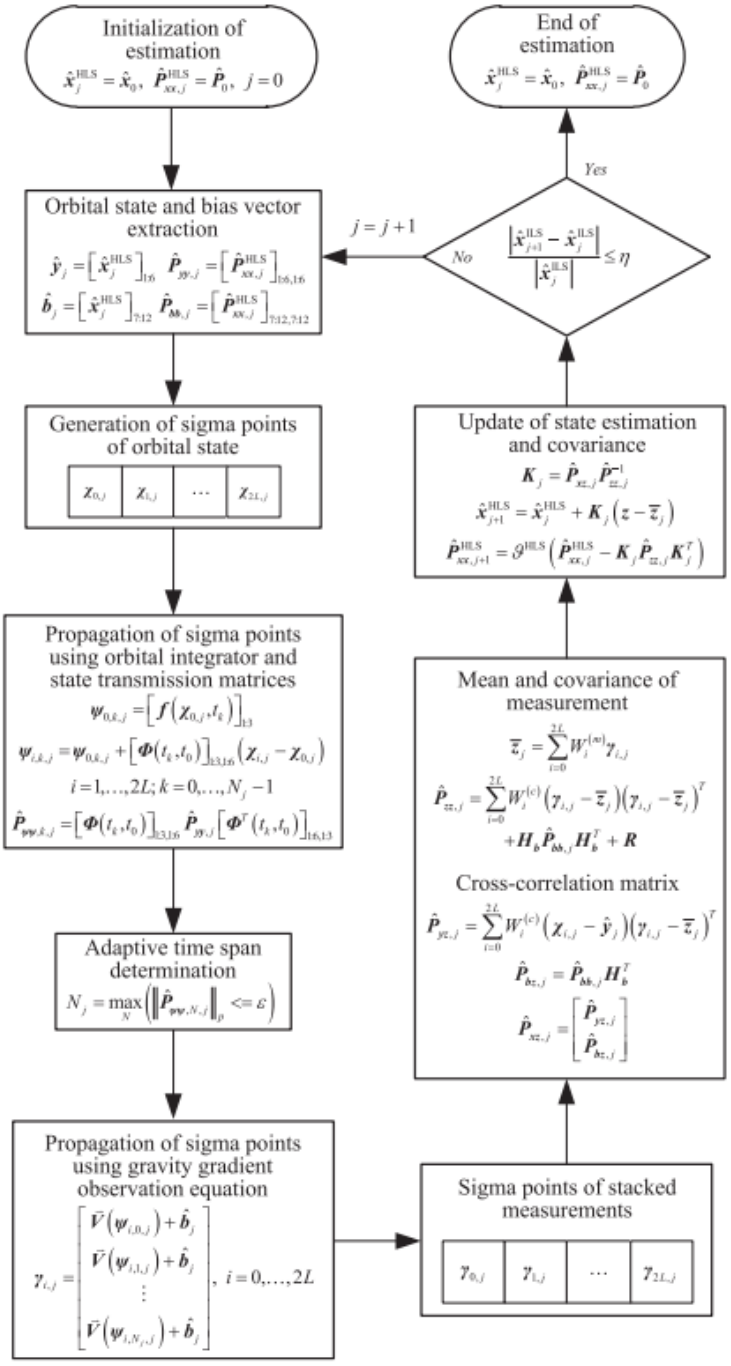

Figure C.6: Process by which results from X. Sun et al. [50] are obtained.

In 2022, P. Chen et al. [4] a process in which the bias is taken into account is done. in this process, an extended Kalman filter (EKF) is used along with a least squares solution to figure out the bias of the gradiometers. Most importantly, this research also gives a better perspective on how different variables can affect the results of orbit determination by this method.

- Orbit inclination seems to severely affect the ”along track” position error and ”radial” velocity error the most. With the more inclination, the smaller the error gets.
- Orbital altitude is a significant variable that changes results. The lower the altitude, the smaller the errors are all across the board. This is as expected, as accelerometers will be able to pick up gravitational gradients more accurately closer they are to Earth.

- Gravity field models also affect the results. When using a J_2 assumption model, the errors are large throughout all parts. Later, the $EGM2008$ gravity model is used. one truncated at degree and order 10 and the other at 50. The higher degree and order truncation yielded the smaller errors.
- Gradiometer noise is also observed. errors when using a noise of $0.01[E]$ and $0.1[E]$ seem to be very close and within the same range. Despite this, the $1[E]$ noise data appears to have a much larger jump in error increase. This leads to the conclusion that the current technology of gradiometers (with an accuracy or noise of $0.1[E]$) already yield satisfactory results, without taking into account how these affect biases. It also further solidifies the fact that quantum gradiometers (with a accuracy of $20[E]$) are not enough to yield good results.

This data is very useful for knowing what to expect when performing validation and verification, as well as knowing what different variables are expected to cause.

This, however, is done with data known already after the mission about all the orbit. While it gives a every accurate set of information, it is all retroactively obtained. The objective of this research is to find a way to execute this but as the mission is ongoing and autonomous from ground procedures for positioning.

The method which will be used this time is of a least squares computation. By making use of $\vec{V}_{uu}$ and its derivative with respect to position $\boldsymbol{V}_{uuu}$, the objective will be to measure how accurate this can be done with an initial position estimate of $\vec{y}_0$. Equation C.10 and Equation C.11 show a simplified overlook of how the principal behind the least squares works.

$$\frac{d\vec{V}_{uu}}{du} = \boldsymbol{V}_{uuu} = \boldsymbol{A} \tag{C.10}$$

$$\vec{y}_{i+1} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T(\vec{V}_{uu} - \vec{V}_{uu}^{(measured)}) + \vec{y}_i \tag{C.11}$$

This process, if taking bias into account, has not yet been seen within the literature. Accuracy will not be the only parameter measured. The speed at which each position is calculated will also be measured and compared.

C.3. Conclusion

The conclusion to this literature study is a re-formulation of the sub-questions which are eventually used in this thesis report, and which can be found in Section 1.3.1, as well as a plan for this thesis, found in Section 1.3.3.

References

[1] A. Bobojć and A. Drożyner. “Satellite orbit determination using satellite gravity gradiometry observations in GOCE mission perspective”. In: *Advances in Geosciences* 1 (2003), pp. 109–112. DOI: 10.5194/adgeo-1-109-2003. URL: https://adgeo.copernicus.org/articles/1/109/2003/.

[2] P. Chen, X. Sun, and C. Han. “Gravity Gradient Tensor Eigendecomposition for Spacecraft Positioning”. In: *Journal of Guidance, Control and Dynamics* 38.11 (Nov. 2015). DOI: https://doi.org/10.2514/1.G001195.

[3] Eun-Seo Park et al. “Satellite orbit determination using a batch filter based on the unscented transformation”. In: *Aerospace Science and Technology* 14.6 (2010), pp. 387–396. ISSN: 1270-9638. DOI: https://doi.org/10.1016/j.ast.2010.03.007. URL: https://www.sciencedirect.com/science/article/pii/S1270963810000428.

[4] P. Chen et al. “Observability Analysis for Orbit Determination Using Spaceborne Gradiometer”. In: *Journal of Aerospace Engineering* 36.2 (2023), p. 04022122. DOI: 10.1061/JAEEEZ.ASENG-4619. URL: https://ascelibrary.org/doi/abs/10.1061/JAEEEZ.ASENG-4619.

[5] P. Zingerle et al. “The combined global gravity field model XGM2019e”. In: *Journal of Geodesy* 66 (2020). DOI: https://doi-org.tudelft.idm.oclc.org/10.1007/s00190-020-01398-0.

[6] X. Xu et al. *A GOCE only gravity model GOSG02S based on the SGG and SST observations*. DOI: https://doi.org/10.5880/icgem.2023.002. URL: https://dataservices.gfz-potsdam.de/icgem/showshort.php?id=088e9b19-271b-11ee-95b8-f851ad6d1e4b.

[7] E. Mazarico. *Lunar Gravity Field: GRGM1200A*. URL: https://pgda.gsfc.nasa.gov/products/50 (visited on 01/07/2025).

[8] S. Yu. Dobrokhotov and A. V. Tsvetkova. “An Approach to Finding the Asymptotics of Polynomials Given by Recurrence Relations”. In: *Russian Journal of Mathematical Physics* 2 (2021), pp. 198–223. ISSN: 1061-9208. DOI: https://doi.org/10.1134/S1061920821020060.

[9] P. Touboul et al. “Gravitation and Geodesy with Inertial Sensors, from Ground to Space”. In: *Onera Aerospacelab journal* (Dec. 2016). URL: https://hal.science/hal-01512942.

[10] Claudia Stummer. *Gradiometer Data Processing and Analysis for the GOCE Mission*. 2012. URL: https://mediatum.ub.tum.de/doc/1111698/362238.pdf.

[11] B. Frommknecht et al. “GOCE level 1b data processing”. In: *Journal of Geodesy* 85 (Oct. 2011), pp. 759–775. DOI: https://doi-org.tudelft.idm.oclc.org/10.1007/s00190-011-0497-4.

[12] H. P. Gavin. *The Levenberg-Marquardt algorithm for nonlinear least squares curve-fitting problems*. May 2024. URL: https://people.duke.edu/~hpgavin/lm.pdf.

[13] Robert Spero. “Point-mass sensitivity of gravimetric satellites”. In: *Advances in Space Research* 67.5 (2021), pp. 1656–1664. DOI: https://doi.org/10.1016/j.asr.

2020.12.019. URL: https://www.sciencedirect.com/science/article/pii/S0273117720308759.

[14] N. A. Gumerov and R. Duraiswami. “CHAPTER 2 - Elementary Solutions”. In: *Fast Multipole Methods for the Helmholtz Equation in Three Dimensions*. Elsevier Series in Electromagnetism. Amsterdam: Elsevier Science, 2004, pp. 39–87. DOI: https://doi.org/10.1016/B978-008044371-3/50006-5. URL: https://www.sciencedirect.com/science/article/pii/B9780080443713500065.

[15] J.M. Brockmann et al. *The Earth’s gravity field as seen by the GOCE satellite - an improved sixth release derived with the time-wise approach ($GO\_CONS\_GCF\_2\_TIM\_R6$)*. GFZ Data Services. URL: https://dataservices.gfz-potsdam.de/icgem/showshort.php?id=escidoc:4315891 (visited on 04/25/2025).

[16] P. Zingerle et al. *The polar extended gravity field model TIM_R6e*. GFZ Data Services. URL: https://dataservices.gfz-potsdam.de/icgem/showshort.php?id=escidoc:4365907 (visited on 04/25/2025).

[17] *GOCE Level 2*. URL: https://earth.esa.int/eogateway/catalog/goce-level-2 (visited on 03/13/2025).

[18] R. Rummel Th. Gruber, O. Abrikosov, and R. van Hees. *GOCE High Level Processing Facility: GOCE Level 2 Product Data Handbook*. Aug. 2014.

[19] G. B. Reed. *Application of Kinematical Geodesy for Determining the Short Wave Length Components of the Gravity Field by Satellite Gradiometry*. 1973.

[20] *GOCE Level 2*. URL: https://goce-ds.eo.esa.int/oads/access/collection/GOCE_Level_2 (visited on 04/25/2025).

[21] D. DiFrancesco et al. “Gravity gradiometer systems – advances and challenges”. In: *Geophysical Prospecting* 57 (4 June 2009), pp. 615–623. DOI: https://doi.org/10.1111/j.1365-2478.2008.00764.x.

[22] P. N. A. M. Visser and J. A. A. van den IJssel. “Calibration and validation of individual GOCE accelerometers by precise orbit determination”. In: *Journal of Geodesy* (2016), pp. 1–13. DOI: https://doi.org/10.1007/s00190-015-0850-0.

[23] X. Sun et al. “Low-Earth Orbit Determination from Gravity Gradient Measurements”. In: *Acta Astronautica* 123 (2016), pp. 350–362. DOI: https://doi.org/10.1016/j.actaastro.2016.03.012.

[24] *GNSS works on the Moon*. UESPA. URL: https://www.euspa.europa.eu/newsroom-events/news/gnss-works-moon (visited on 07/18/2025).

[25] J.R. Millán Fernández. *Least Squares Algorithm for Gravity Gradient Navigation*. 2025. DOI: 10.4121/3ff164ad-582f-4af7-bfce-46aefaba2af9.

[26] A. DeGregoria. “Gravity Gradiometry and Map Matching: An Aid to Aircraft Inertial Navigation Systems”. 2010. URL: https://apps.dtic.mil/sti/pdfs/ADA517387.pdf.

[27] J. A. Richeson and D. J. Pines. “GPS Denied Inertial Navigation using Gravity Gradiometry”. In: *AIAA Guidance, Navigation and Control Conference and Exhibit* (2007), pp. 20–23. DOI: https://doi.org/10.2514/6.2007-6791.

[28] S. Weiner et al. “A Flight Capable Atomic Gravity Gradiometer With a Single Laser”. In: *2020 IEEE International Symposium on Inertial Sensors and Systems* (2020), pp. 1–3. DOI: https://doi.org/10.1109/INERTIAL48129.2020.9090014.

[29] T. M. MacRobert. *Spherical harmonics; an elementary treatise on harmonic functions, with applications.* Oxford, 1967. URL: https://archive.org/details/sphericalharmoni0000macr/page/66/mode/2up.

[30] Jacob Hollebon and Filippo Maria Fazi. “Efficient HRTF Representation Using Compact Mode HRTFs”. In: *Audio Engineering Society Convention 149*. Oct. 2020. URL: https://www.aes.org/e-lib/browse.cfm?elib=20941.

[31] P. Zingerle et al. “The combined global gravity feld model XGM2019e”. In: *Journal of Geodesy* 94.66 (2020). DOI: https://doi.org/10.1007/s00190-020-01398-0.

[32] *About GOCE*. URL: https://earth.esa.int/eogateway/missions/goce (visited on 05/04/2023).

[33] EGG-C. *Release 6 GOCE Gravity Field Models Validation Report*. 2019. URL: https://earth.esa.int/eogateway/documents/20142/37627/Release-6-gravity-model-validation-report-GO-TN-HPF-GS-0337-1.0.pdf.

[34] S. Bettadpur. *GRAVITY RECOVERY AND CLIMATE EXPERIMENT, UTCSR Level-2 Processing Standards Document*. 2018. URL: http://icgem.gfz-potsdam.de/GRACE_CSR_L2_Processing_Standards_Document_for_RL06.pdf.

[35] E. Mazarico. *Lunar Gravity Field: GRGM1200A*. URL: https://pgda.gsfc.nasa.gov/products/50. (accessed: 23/10/2022).

[36] A. Genova. *Mars Gravity Field: GMM-3*. URL: https://pgda.gsfc.nasa.gov/products/57. (accessed: 23/10/2022).

[37] Ronald Eotvos. “Bestimmung der Gradienten der Schwerkraft und ihrer Niveauflächen mit Hilfe der Drehwaage”. In: *Verhandlungen der 15. allgemeinen Konferenz der Internationalen Erdmessung* (1954), pp. 337–396. URL: https://realeod.mtak.hu/8220/1/bestimmung_der_gradienten_der_schwerkraft%20%282%29.pdf.

[38] G. Balmino et al. *GRADIO: Project Proposal for Satellite Gradiometry*. 1981. URL: https://ntrs.nasa.gov/api/citations/19820012732/downloads/19820012732.pdf.

[39] C. Siemes et al. “GOCE gradiometer data calibration”. In: *Journal of Geodesy* 93 (2019), pp. 1603–1630. DOI: https://doi.org/10.1007/s00190-019-01271-9.

[40] P. Touboul, B. Foulon, and E. Willemenot. “Electrostatic space accelerometers for present and future missions”. In: *Acta Astronautica* 45.10 (1999), pp. 605–617. ISSN: 0094-5765. DOI: https://doi.org/10.1016/S0094-5765(99)00132-0. URL: https://www.sciencedirect.com/science/article/pii/S0094576599001320.

[41] C. E. Griggs et al. “Sensitive Superconducting Gravity Gradiometer Constructed with Levitated Test Masses”. In: *Phys. Rev. Appl.* 8 (6 Dec. 2017), pp. 064–024. DOI: 10.1103/PhysRevApplied.8.064024. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.8.064024.

[42] *EGG Overview*. URL: https://earth.esa.int/eogateway/instruments/egg/description (visited on 06/13/2023).

[43] M. R. Drinkwater et al. “GOCE: ESA’s First Earth Explorer Core Mission”. In: *Space Science Reviews* 108 (2013), pp. 419–432. DOI: https://doi-org.tudelft.idm.oclc.org/10.1023/A:1026104216284.

[44] Z. Zhu et al. “Electrostatic gravity gradiometer design for the future mission”. In: *Advances in Space Research* 51.12 (2013), pp. 2269–2276. ISSN: 0273-1177. DOI: https://doi.org/10.1016/j.asr.2013.01.031. URL: https://www.sciencedirect.com/science/article/pii/S0273117713000689.

[45] H. A. Chan and H. J. Paik. “Superconducting gravity gradiometer for sensitive gravity measurements. I. Theory”. In: *Phys. Rev. D* 35 (12 June 1987), pp. 3551–3571. DOI: 10.1103/PhysRevD.35.3551. URL: https://link.aps.org/doi/10.1103/PhysRevD.35.3551.

[46] J. Zhong et al. “Quantum gravimetry going toward real applications”. In: *The Innovation* 3.3 (May 2022). DOI: https://doi.org/10.1016/j.xinn.2022.100230.

[47] Wei Lyu et al. “Compact High-Resolution Absolute-Gravity Gradiometer Based on Atom Interferometers”. In: *Phys. Rev. Appl.* 18 (5 Nov. 2022), p. 054091. DOI: 10.1103/PhysRevApplied.18.054091. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.18.054091.

[48] B. Stray et al. “Quantum sensing for gravity cartography”. In: *Nature* 602 (Feb. 2022), pp. 590–594. DOI: https://doi.org/10.1038/s41586-021-04315-3.

[49] Jamie Vovrosh et al. “Advances in Portable Atom Interferometry-Based Gravity Sensing”. In: *Sensors* 23.17 (2023). DOI: 10.3390/s23177651. URL: https://www.mdpi.com/1424-8220/23/17/7651.

[50] Xiucong Sun et al. “Low-Earth Orbit Determination from Gravity Gradient Measurements”. In: *Acta Astronautica* 123 (2016). Special Section: Selected Papers from the International Workshop on Satellite Constellations and Formation Flying 2015, pp. 350–362. ISSN: 0094-5765. DOI: https://doi.org/10.1016/j.actaastro.2016.03.012. URL: https://www.sciencedirect.com/science/article/pii/S0094576515301193.

[51] C. Siemes. *GOCE gradiometer calibration and Level 1b data processing*. Nov. 2011. URL: https://earth.esa.int/eogateway/documents/20142/37627/GOCE-gradiometer-calibration-and-level-1b-data-processing.pdf.

[52] C. Siemes et al. “GOCE gradiometer data calibration”. In: *Journal of Geodesy* 93 (June 2019), pp. 1603–1630. DOI: https://doi.org/10.1007/s00190-019-01271-9.

[53] G. Carmeli and B. Ben-Moshe. “AI-Based Real-Time Star Tracker”. In: *Electronics* 12.9 (2023). DOI: 10.3390/electronics12092084. URL: https://www.mdpi.com/2079-9292/12/9/2084.

[54] Roelof W. H. van Bezooijen. “True-sky demonstration of an autonomous star tracker”. In: *Acquisition, Tracking, and Pointing VIII*. Ed. by Michael K. Masten et al. Vol. 2221. International Society for Optics and Photonics. SPIE, 1994, pp. 156–168. DOI: 10.1117/12.178979. URL: https://doi.org/10.1117/12.178979.

[55] *Autonomous Star Sensor ASTRO APS*. 2013. URL: https://www.jena-optronik.de/products/star-sensors/astro-aps.html.

[56] T. Sun et al. “Effective star tracking method based on optical flow analysis for star trackers”. In: *Applied Optics* 55.36 (Dec. 2016), pp. 10335–10340. DOI: https://doi.org/10.1364/AO.55.010335.

[57] S. Carletta, P. Teofilatto, and M. Salim Farissi. “A Magnetometer-Only Attitude Determination Strategy for Small Satellites: Design of the Algorithm and Hardware-in-the-Loop Testing”. In: *Aerospace* 7.1 (2020). DOI: 10.3390/aerospace7010003. URL: https://www.mdpi.com/2226-4310/7/1/3.

[58] Qi Zhang and Yulin Zhang. “Design and Verification of an Integrated Panoramic Sun Sensor atop a Small Spherical Satellite”. In: *Sensors* 22.21 (2022). DOI: 10.3390/s22218130. URL: https://www.mdpi.com/1424-8220/22/21/8130.

[59] S.J. Julier, J.K. Uhlmann, and H.F. Durrant-Whyte. “A new approach for filtering nonlinear systems”. In: *Proceedings of 1995 American Control Conference - ACC’95*. Vol. 3. 1995, 1628–1632 vol.3. DOI: 10.1109/ACC.1995.529783.