8-9-2022

Three-axis magnetometer calibration with norm preservation

Seth Lichlyter
Mississippi State University, seth.lichlyter@gmail.com

Three-axis magnetometer calibration with norm preservation

By

Seth Lichlyter

Approved by:

Yang Cheng (Major Professor)
Shantia Yarahmadian
Calvin Walker
Adrian Sescu (Graduate Coordinator)
Jason M. Keith (Dean, Bagley College of Engineering)

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Aerospace Engineering
in the Bagley College of Engineering

Mississippi State, Mississippi

August 2022

Name: Seth Lichlyter

Date of Degree: August 9, 2022

Institution: Mississippi State University

Major Field: Aerospace Engineering

Major Professor: Cheng, Yang

Title of Study: Three-axis magnetometer calibration with norm preservation

Pages in Study: 70

Candidate for Degree of Master of Science

This thesis proposes a set of methods for the purpose of improving the calibration of three-axis magnetometers. Specifically, these methods aim to improve the accuracy of the bias estimation methods currently in use. The first proposed method utilizes a constrained optimization problem based on norm preserving. The second proposed method finds the same bias estimate as the first method, but in a computationally more efficient manner. The last proposed method tackles the case where the value of the local geomagnetic field is only imprecisely known. Computer simulations demonstrate the viability of the proposed methods.

TABLE OF CONTENTS

LIST OF TABLES iv

LIST OF FIGURES v

CHAPTER

I. INTRODUCTION 1

Literature Review 2
Organization of Thesis 9

II. METHODOLOGY 10

Simulation of Magnetometer Readings 10
Definition of an Attitude Matrix 11
Processing Raw Gyroscope Data 12
Conversion of Data into Quaternions 12
Methods of Measuring Attitude 12
Finding the Time Derivative of a Quaternion 13
Conversion of Rotation Vectors to Quaternions 13
An Expression for Local Perturbation 14
The Exponential Map 15
Time Derivative of a Quaternion 15
Conversion of Angular Velocity to Quaternions 16
Conversion of Quaternions to Attitude Matrix 18
Data Simulation Function 20
Output of the Data Simulation Function 23
Comparing the Calibration Methods 24
The TLS Method Function 26
The Method 1 Function 28
The Method 2 Function 30
The Method 3 Function 33
Deriving the Measurement of Constraint Error 38
The Optimization Function 39

III. RESULTS 41

Creating a Baseline Reading 41
Source of the Attitude Data 41

Value of the Local Magnetic Field 43
Baseline Readings 45
Realistic Range for Bias Testing 46
Verification of Bias Measurement..48
Constraint Error Measurements...51
Comparison of Algorithm Accuracy 52
SSE Function 53
SSE over Range of Noise 53
Verification of Bias Estimation 56
SSE over Range of Bias 58
Accuracy when H_k is not Constant 61
Comparison of Algorithm Speed...63
Conclusion 65
A Note Regarding Complexity 67

LIST OF TABLES

Table 3.1 Gyroscope Product Attributes 42

Table 3.2 Local Magnetic Field – Starkville, MS 45

Table 3.3 Estimated Bias of a Laboratory Sensor 47

Table 3.4 Estimated Bias of a Phone Sensor 48

LIST OF FIGURES

Figure 2.1 Euler axis/angle rotation....19

Figure 3.1 Simulated Magnetometer Readings (Zero-Bias, Noiseless)....46

Figure 3.2 Estimated and True Biases from TLS Method Function....49

Figure 3.3 Estimated and True Biases from Method 2 Function....50

Figure 3.4 Constraint Error from TLS Method....51

Figure 3.5 Constraint Error from Method 2....52

Figure 3.6 Comparison of Accuracy for Methods over Range of Noise....54

Figure 3.7 Difference between TLS Method SSE and Method 2 SSE....55

Figure 3.8 Sample of the Difference in SSE at Z-Axis....56

Figure 3.9 Verification of Method 2 Bias Estimation at Highest Magnitude of Noise....57

Figure 3.10 Comparison of Accuracy for Methods over Range of Bias....59

Figure 3.11 Difference between TLS Method SSE and Method 2 SSE....60

Figure 3.12 Estimated and True Biases from Method 3 Function....62

Figure 3.13 Comparison of Computational Speed between Method 2 and the TLS Method....64

CHAPTER I

INTRODUCTION

Three-Axis Magnetometers (TAM) are sensors that measure the strength and direction of a surrounding local magnetic field. A major application of magnetometers is the calculation of an object's attitude or orientation based on the data provided by magnetometers. Many platforms such as satellites, airplanes, and submarines are equipped with magnetometers because these platforms use attitude determination systems to assist in navigation, and magnetometers contribute heavily to those systems. [1] [2] [3] [4] [5] In multi-sensor systems, the data taken from TAM sensors can be used in conjunction with other sensors like accelerometers, gyroscopes, and GPS in order to improve the accuracy of an attitude determination system. In situations where other sensors are not performing properly, the presence of a magnetometer can become essential to the vehicle's ability to function. [1] [2] [3] [5] [6]

When analyzing the results taken from magnetometer readings, it is important to recognize that those results can be distorted by internal and environmental factors. The distortions caused by a sensor's internal makeup can appear as consistent biases, while the distortions caused by environmental factors can appear when there is a significant change in the natural electromagnetic environment around the sensor. [4] [6] [7] Those distorted readings can cause the attitude of the vehicle to be miscalculated, and an incorrect attitude determination can negatively impact the performance of a vehicle. [2] [8] The bias can be minimized by using sensors that maintain a very small error tolerance or by calibrating a sensor by applying an optimization algorithm that filters the bias from the measured data of a sensor.

In this thesis, new methods of sensor calibration are developed with the goal of improving the accuracy of sensor calibration as well as improving the speed with which the calibration is implemented. Similar to other well-established calibration methods, these methods utilize an optimization algorithm in order to estimate bias. Uniquely, these methods apply a specific norm-preservation as a constrain to the optimization algorithm. This constraint reduces the quantity of computations required and reduces the opportunity for error to enter the system.

Literature Review

The simplest way to add external sensors is to use a high-quality apparatus such as Inertial Navigation Systems (INS) or Attitude Heading Reference Systems (AHRS) that minimize the sensor bias created. However, this option is accompanied by two potential problems. As a product, magnetometers can become much more expensive and much heavier as their quality increases. The high cost of top-of-the-line magnetometers can deter researchers from investing in them without additional funding. The increased weight of high-quality magnetometers can disqualify them as an option in situations where space and weight is limited. For example, micro aerial vehicles (MAVs) and small satellites (CubeSats) both require magnetometers, but MAVs and CubeSats both have very strict weight requirements. [1] [3] [4] [6] .

Some hybrid attitude determination methods use optimization algorithms and utilize external sensors at the same time. They propose the installation of smaller external sensors in an attempt to replicate the accuracy of AHRS while avoiding the higher cost and weight of AHRS. A number of these methods attempt to combine Micro-electromechanical Systems (MEMS) based inertial sensors which provide short-term accuracy with a magnetometer and GPS which provide long-term accuracy. This setup has two major flaws. First, there is consistent inability to track the vector of acceleration due to gravity when maneuvers with high attitude dynamics are employed over an extended period of time. Second, reliance on GPS becomes a liability in locations that don't consistently provide a strong GPS signal. [3] [4] [5]

One method attempts to address these flaws by installing an accelerometer along with the other sensors. Another method uses visual sensors called Visual Odometry (VO) navigation systems to take relative rotation measurements which contribute to attitude determination. These methods can improve the overall accuracy of the system. However, every method that relies heavily on external sensors draws closer to sharing the same limitations of cost and weight associated with INS and AHRS. [3] [4] [5] The best way to avoid those limitation is to apply an optimization algorithm that calculates the bias from the measured data of a sensor without adding external sensors.

Because so many calibration algorithms have been developed, it is helpful to separate them into groups. Every method falls within one of the following two categories. The first category includes Attitude-dependent algorithms, and the second category includes Attitude-independent algorithms. Both categories require that measurements of the local magnetic field be taken, and both categories rely on a predetermined model of the magnetic field to be used as reference.

Attitude-dependent methods are distinctive in that they depend on additional attitude information such as attitude rates. [9] Attitude rates can be determined from the measurements taken from gyroscopes. This is why methods that rely on external sensors such as gyroscopes are often Attitude-dependent methods. For applications such as airplanes and cars, Attitudedependence can be beneficial because there is an assumed baseline attitude of zero roll angle and zero pitch angle. [4] For other applications, attitude information is not as easily accessible. [10]

Calibration algorithms have evolved from Attitude-dependent to Attitude-independent methods. This transition became possible when researchers realized that the magnitude of sensor data is constant no matter how the magnetometer is oriented within a specific location. [7] Attitude-independent methods maintain the advantage of versatility. While Attitude-dependent schemes require that additional sensors be installed on the platform in order for the attitude determination system to work properly, Attitude-independent schemes do not require any external sensors. If attitude determination is required in situations where external sensors are unavailable, malfunctioning, or prone to error, an Attitude-independent method is necessary. [10]

The use of attitude-independent methods also has disadvantages. Some Attitude-independent methods require the magnetometer sensor to be rotated in every direction within a uniform magnetic field in order for the calibration to work. [8] Also, depending on the application, results taken from Attitude-dependent methods can reach a higher degree of accuracy than results taken from Attitude-independent methods. [10] Finally, while this is not a weakness unique to Attitude-independent methods, in order to use attitude-independent calibration successfully, external resources such as geomagnetic models are required. [4]

Without the additional data taken from the visual sensors, any attempt to find the sensor bias becomes a minimization problem which necessitates an iterative solution. There are a number of valid iterative methods, but they require knowledge regarding the magnitude of the magnetic field in a specific geographical location (usually provided by a model) in order to work. If there are any disruptions in that magnetic field, and those disruptions are not shown in the model, significant error will occur during the bias estimation process. [6]

Within the category of Attitude-independent calibration algorithms, it is helpful to distinguish between two subcategories. Attitude-independent calibration algorithms can be applied through two methods. The first method is called the batch approach, and the second method is called the real-time approach. The two methods are distinguishable from each other based on how they process results. The batch approach analyzes an array of data representing measurements recorded over a length of time, and it produces an array of estimations representing a calibration over that length of time. The real-time approach analyzes a point of data representing a moment in time, it produces a point of estimation representing the calibration for that moment in time, and it repeats the process for the next relevant point in time.

The advantages of the batch approach are dependent on the requirements of its application. For example, the batch approach is effective when it is used to calibrate the sensors of a satellite because satellites do not require moment-to-moment attitude calibrations in order to function. Here, computational efficiency even increases when data is processed in batches rather than each individual packet of data processed separately. However, this long-term efficiency is negated in other situations. Vehicles like UAVs are capable of executing attitude dynamic maneuvers, and the success of those maneuvers can depend on maintaining constant attitude awareness. The batch-processing method would not be capable of providing continual sensor calibration in this case, as it requires additional time to process each batch of data.

The advantages of the real-time approach are simply the opposite of the advantages of the batch approach. The real-time approach is effective in situations that require maintaining constant attitude awareness. However, when constant attitude awareness is not required, the computational efficiency of the real-time approach is lower than the computational efficiency of the batch approach. Because our research focuses on the batch approach, it is important to look at the advantages and disadvantages of Attitude-independent calibration methods based on their ability to produce batch results rather than real-time results. [1] [6] [7]

Because our research focuses on Attitude-independent batch-processing methods, it is important to discuss some of the most common Attitude-independent methods capable of batch-processing. The first method presented is the Kalman Filter. The Kalman Filter is a linear quadratic estimation algorithm that employs a joint probability distribution to estimate results. This filter is often adjusted to one of two settings. The first setting is called an Extended Kalman Filter (EKF). This filter produces results quickly, but it employs a first order linearization which often oversimplifies those results. The second setting is called an Unscented Kalman Filter (UKF). The UKF produces a higher level of accuracy than the EKF, but the cost of that accuracy is a significantly longer computational timeframe. The delay makes the UKF infeasible for applications that require real-time attitude updates. [2] [10]

The next method presented is called an approximate Maximum Likelihood (ML) estimation. This method maximizes a likelihood function by using a centering approximation. The resulting approximation produces a reasonable value for a preliminary estimate, and this estimate is used to solve the resulting fourth order objective function. This algorithm also has notable flaws. Other methods regularly outperform the approximate ML estimation in situations where the range of attitude-affecting maneuvers is limited. Also, results taken from an approximate ML estimation are highly sensitive to changes in the initial conditions used. If those initial conditions include even small amounts of error, the results can be significantly altered. [7]

The final method presented is the Least Square (LS) method. Similar to the ML estimation, the LS method maximizes a likelihood function by using a centering approximation. When linearized, the LS method incorporates a pseudo-linear equation within intermediate variables in order to provide a preliminary estimate. [7] A new method is derived when a second step is added to the LS method which employs the Gauss-Newton iterative process. This new method is aptly named TWOSTEP and will be discussed in further detail. [2]

Here we focus on common Attitude-independent calibration models that specifically use numerical iterations to calculate the internal bias of Three-Axis Magnetometers. The first relevant calibration model is called Naïve Quartic Scoring (NQS). NQS is usually derived from a standard Newton-Raphson approximation. While NQS is a functional method of calibration, it also occasionally generates numerical inconsistencies. These inconsistencies can create situations where the calculated minimum is local rather than global. [1] The next relevant calibration model is called the Fixed-Point method. This method converts the calibration problem into a Fixed-Point problem where the sensor bias is defined as a fixed point within the loss function. The convergences calculated from this method are frequently inferior to the results from other models. [1]

One of the most accurate calibration models available is called TWOSTEP. As described previously, the TWOSTEP model calculates an initial approximation by using the centered estimate method. TWOSTEP then calculates an iterative approximation based first on the centered estimate and then on each iteration of the Gauss-Newton process. Before applying the Gauss-Newton process, TWOSTEP analyzes the centered estimate to determine if the optimization conditions have already been met. If not, that second step is implemented.

A flaw in the TWOSTEP model originates from the way that the effective measurement is calculated through a concept called scalar checking. This concept capitalizes on the understanding that when a local magnetic field is measured from the body coordinates as well as the reference coordinates, the magnitude of those two measurements are equal. However, this assumption is only correct in cases where the magnitude of noise is zero. When noise is added to the measurements, the variation of error for the effective measurement becomes a function of bias. That variation caused by bias is disregarded when the second step is implemented. This can lead to inaccurate measurements of bias.

The final relevant model is called Total Least Squares (TLS). The TLS method is based on the same assumptions as the TWOSTEP method, but TLS attempts to improve on the success of TWOSTEP by producing a precise Newton-Raphson algorithm that replaces the centered estimate from TWOSTEP. Based on the statistical evidence, resulting TLS convergences are more consistent that TWOSTEP convergences. [1]

While the TLS method is a significant improvement to other methods, the goal of our research is to explain a weakness within the TLS method, propose a solution to that weakness, and analyze the effectiveness of that solution. The issue with the Total Least Square method that we examine can be described as “Constraint-preservation error”. Through the formulation of the TLS algorithm in Reference [1], a particular mathematical relationship between three variables is derived. These three variables are the measurement of the magnetic field by the magnetometer, the constant bias vector of the sensor (including the estimate error of the bias), and the value of the geomagnetic field with respect to an Earth-fixed coordinate system.

While this relationship is discussed in Reference [1], it is not utilized as a constraint when the TLS loss function is optimized. When the TLS results are calculated, those three variables can be compared to each other to determine whether or not this mathematical relationship is maintained. As is demonstrated in our research, the results provide an approximation of the exact value predicted by this relationship, but there is a measurable error between this approximation and the exact value. Therein lies an opportunity to improve the TLS method by enforcing this relationship as a constraint rather than allowing the error to remain.

The method proposed within our research attempts to improve the effectiveness of the TLS method by using the mathematical relationship discussed in Reference [1] as a constraint that must be preserved while the loss function is being optimized. This strategy is implemented through our newly designed Norm-Preserving (NP) algorithm. This algorithm will ensure that the constraint error equals zero by applying the mathematical relationship as a nonlinear constraint function. The goal of our research is based on the premise that adding this constraint to the optimization function for the TLS problem improves the effectiveness of the optimization function and the accuracy of the calibration. Our results confirm that premise.

Additionally, we propose an improvement to our own NP algorithm with the explicit purpose of reducing the computational time needed to calculate sensor bias. This strategy is implemented through our Fast Norm-Preserving (FNP) algorithm. This algorithm will preserve the improved accuracy of the NP algorithm while reducing the time needed to optimize the TLS loss function. The goal of this FNP algorithm is based on the premise that by incorporating the mathematical relationship into the loss function rather than making it be a separate nonlinear constraint function will increase the speed of the optimization function. Our results confirm that premise.

Organization of Thesis

This thesis is organized as follows. Following the introduction and literature review in this chapter, the methodology behind our new approach is defined in Chapter 2. Next, our new methods are compared to the TLS method. The results of this comparison are presented in Chapter 3 as well as the conclusion based on those results.

CHAPTER II

METHODOLOGY

This chapter describes the methodology of our experiment. This includes the setup and the reasoning behind our setup. We will first describe the process of creating simulated magnetometer readings. Next, we will present the derivation of the TLS function as well as our new functions. Finally, we will provide a proof for the measurement of our new constraint. The application of this constraint is what will differentiate our new calibration functions from the TLS calibration function.

Before discussing the formulation and comparison of the calibration methods, those methods will be assigned specific nomenclature. The previous Total Least Squares method from which our new methods are derived will continue to be called the TLS Method. The first new method which seeks to improve upon the accuracy of the TLS Method through the preservation of a nonlinear constraint will be simply called Method 1. The second new method which seeks to improve upon the computation speed of Method 1 by reducing the size and number of parameters required within the optimization function will be called Method 2. The final new method assumes that the value of the geomagnetic field with respect to earth is unknown. This method will be labeled Method 3.

Simulation of Magnetometer Readings

For the purposes of testing our new calibration methods, it is necessary for us to have access to true or simulated sensor readings. The origin of those readings is not relevant to the testing phase of our research. It only matters that the data used reasonably represents the possible motion of an object through space without the distortion of magnetic bias caused by an actual magnetometer already included in the data. By using a gyroscope to help simulate readings, we can control the “true” bias of the readings and compare the accuracy of different bias filtering methods. For the purposes of our experiment, gyroscopic sensor data was recorded from the arbitrary circular motion of a smartphone. That gyroscopic data was then converted to a variable called the Unknown Attitude Matrix with respect to Earth-fixed coordinates. [11]

From there, a local geomagnetic reference value, sensor bias values, and external noise values were artificially introduced to the data. The output of this process was then used as simulated raw magnetometer input. This input was then calibrated by the TLS method, method 1, method 2 and method 3. While this is the approach chosen for our research, it is by no means the only way to test our new methods of calibration. Whatever approach is used must provide a researcher with the ability to regulate the true simulated bias and compare it to the bias measured by a calibration method. As long as this requirement is met, it should not matter if the simulated magnetometer readings are derived from real data or if they are generated from computer code. The results of our comparison will be repeatable. The next subheading provides details of our approach to create simulated magnetometer readings, beginning with the formulation of the attitude matrix.

Definition of an Attitude Matrix

Before attempting to calculate the attitude matrix, it is important to understand what attitude means. In the realm of navigation, the attitude of an object is its orientation in space. For an aircraft, that is often described by the angles of roll, pitch, and yaw. Later on, we will discuss different conventions that are used to quantify the attitude of an object such as quaternions and

Euler’s angles. In order to calculate an attitude matrix, we chose to record data from a gyroscope. A gyroscope is a sensor that measures the angular velocity of an object.

Processing Raw Gyroscope Data

The purpose of this subheading is to provide background on gyroscopes, describe how our gyroscopic data is collected and explain how that data is then used. Essentially all satellites utilize gyroscopes, and the vast majority of aerial vehicles have gyroscopes built into them. [12] They often have an apparatus called a Micro-electromechanical system (MEMS). MEMS based sensors are utilized as a way to reduce the size and cost compared to more complex Inertial Navigation Systems (INS) or Attitude Heading Reference Systems (AHRS). [3] While gyroscopes can have their own errors and biases, these inaccuracies do not affect how we are using gyroscopic data for our research. Our data is not meant to be a perfectly accurate sample, but rather it is meant to be a way to produce a realistic simulated attitude matrix that can later be applied to an optimization algorithm.

Conversion of Data into Quaternions

Here, we convert the angular velocities found in the last section to quaternions. Before describing the steps involved in that conversion, it’s important to explain what a quaternion is and why it is used here.

Methods of Measuring Attitude

There are a couple common methods that can be used to describe the attitude of an object. One is the set of Euler angles. There are three Euler angles, and as mentioned earlier, they are easy to visualize on an aircraft with roll, pitch, and yaw. However, the Euler angles have flaws. First, the way that Euler angles are set up with three angles in 3-d space allows for errors to occur when one of those angles gets close to a specific value. This error is called singularity. Second, because Euler angles are used to integrate incremental changes of attitude over time, they do not have the highest levels of accuracy. [13]

Another method we can use is the quaternions. This parameter uses four values: a scalar (real) value that is the first or last term depending on notation, and the three vector (imaginary) values. An initial quaternion that shows zero rotation would have a scalar value of 1, and a vector value of $0i + 0j + 0k$. This method avoids the drawbacks of the Euler angles because it never reaches the singularity error, and it has a higher level of accuracy when integrating an object's angular velocity over time. For these reasons, the quaternion is used in our research.

Finding the Time Derivative of a Quaternion

The next step is to find a method for converting angular velocities into quaternions. We develop zeroth and first order integration methods based on the Tayler series of quaternions (**q**) over time, but we first provide a proof for a quaternion representing incremental rotation as shown in this subheading. Our formulations here are derived from the work presented in Joan Sola's "Quaternion kinematics for the error-state Kalman filter." [14]

Conversion of Rotation Vectors to Quaternions

As shown in the Equation 2.1 below, a quaternion can be defined as a scalar value and three vector values bounded within the space of quaternions ℍ. [14, p. 4] Additionally, Equation 2.2 defines the following constraints on the vector values. [14, p. 4] This is illustrated practically in Equation 2.3 in the form of a quaternion vector.

$$Q = a + bi + cj + dk \in \mathbb{H} \tag{2.1}$$

$$i^2 = j^2 = k^2 = ijk = -1 \tag{2.2}$$

$$\mathbf{q} \triangleq \begin{bmatrix} q_w \\ \mathbf{q}_v \end{bmatrix} = \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} \tag{2.3}$$

From here we proceed to the topic of Quaternions with respect to rotation vectors. Reference [14] provides a proof of the following: If ϕ represents a rotation around an axis $\mathbf{u}$, a quaternion translates only one half of that rotation. By accepting the proof for this statement, we derive a formula for the conversion from rotation vectors to quaternions that is presented in Equation 2.4 below. [14, p. 22]

$$\mathbf{q} \triangleq \mathrm{Exp}(\phi\mathbf{u}) = e^{\phi\mathbf{u}/2} = \cos\frac{\phi}{2} + \mathbf{u}\sin\frac{\phi}{2} = \begin{bmatrix} \cos(\phi/2) \\ \mathbf{u}\sin(\phi/2) \end{bmatrix} \tag{2.4}$$

An Expression for Local Perturbation

We will apply Equation 2.4 shortly, but first we must present the topic of time-integration of rotation rates. The rate of time-integration can be measured from the differential equation that is applied to our specific definition of the rotation rate. This definition can be either local or global. Based on the fact that our experimental method is based on local measurements, we will be focusing only on the local definition. This is denoted by the cursive L subscript.

The variable $\tilde{\mathbf{q}}$ represents the perturbed orientation of a quaternion. This variable can be defined as the quaternion product of an unperturbed orientation $\mathbf{q}$ and a small local perturbation. The small local perturbation angle $\Delta\phi$ can be calculated through a relationship between the perturbed orientation of a quaternion and the conjugate of the unperturbed orientation **q*** as shown in Equation 2.5 below. [14, p. 44]

$$\Delta\phi_{\mathcal{L}} = \mathrm{Log}(\mathbf{q}_{\mathcal{L}}^{*} \otimes \tilde{\mathbf{q}}_{\mathcal{L}}) \tag{2.5}$$

The Exponential Map

Because we assume that the perturbation angle $\Delta\phi$ is small, then we can approximate the small perturbation quaternion $\Delta\mathbf{q}$ using a Taylor expansion. Here, we introduce the exponential function of the skew-symmetric 3x3 matrix $e^{[\phi]_\times}$. This matrix is constrained by its Taylor series. The relationship between a rotation matrix **R** and the skew-symmetric matrix is known as the exponential map, which is shown in Equation 2.6 below. [14, p. 17]

$$\mathbf{R} = e^{[\phi]_\times} \tag{2.6}$$

Time Derivative of a Quaternion

At this point we can apply the Taylor expansions of quaternions and rotation matrices of Equation 2.6 and Equation 2.4 to the small perturbation angle of Equation 2.5 and end up with the following linear approximations in Equation 2.7. [14, p. 45]

$$\Delta\mathbf{q}_{\mathcal{L}} \approx \begin{bmatrix} 1 \\ \frac{1}{2}\Delta\phi_{\mathcal{L}} \end{bmatrix} \tag{2.7}$$

Now we can introduce time derivatives in our efforts to derive an equation for time-integration of rotation. By starting with the definition of a time derivative, we perform a number of steps including the substitution of Equation 2.7. All of these steps are illustrated in Equation 2.8 shown below. [14, p. 46]

$$
\begin{aligned}
\dot{\boldsymbol{q}} &\triangleq \lim_{\Delta t \to 0} \frac{\boldsymbol{q}(t+\Delta t) - \boldsymbol{q}(t)}{\Delta t} \\
&= \lim_{\Delta t \to 0} \frac{\boldsymbol{q} \otimes \Delta\, \boldsymbol{q}_{\mathcal{L}} - \boldsymbol{q}}{\Delta t} \\
&= \lim_{\Delta t \to 0} \frac{\boldsymbol{q} \otimes \left(\begin{bmatrix} 1 \\ \frac{1}{2} \Delta \phi_{\mathcal{L}} \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right)}{\Delta t} \\
&= \lim_{\Delta t \to 0} \frac{\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \frac{1}{2} \Delta \phi_{\mathcal{L}} \end{bmatrix}}{\Delta t} \\
&= \frac{1}{2} \boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega}_{\mathcal{L}} \end{bmatrix}
\end{aligned}
\qquad (2.8)
$$

Finally, we have the information necessary to create the differential equation that will be used to integrate local rotation rates over time. This is shown through Equation 2.9 below.

$$
\dot{\boldsymbol{q}}(t) = \frac{1}{2} \boldsymbol{q}(t) \otimes \boldsymbol{\omega}(t) \qquad (2.9)
$$

Conversion of Angular Velocity to Quaternions

As shown in section 4.6 of Reference [14], two different approximations of time-integration of rotation rates are given. The zeroth order integration assumes that the angular velocity is constant over the period [t_n, t_n+1], while the first order integration assumes that the angular velocity is linear over the period [t_n, t_n+1]. For our experiment, we will be using this more accurate first order integration. Here, the approximation we make is that for each time step, the angular rate $\boldsymbol{\omega}_n$ is linear, meaning that the first derivative of the angular rate is constant, and that the second (or any higher) derivative is equal to zero. If we define a median angular rate in terms of $\boldsymbol{\omega}_n$ and the first time-derivative of $\boldsymbol{\omega}$, we get Equation 2.10.

$$\bar{\boldsymbol{\omega}} = \frac{\boldsymbol{\omega}_{n+1} + \boldsymbol{\omega}_n}{2} = \boldsymbol{\omega}_n + \frac{1}{2}\dot{\boldsymbol{\omega}}\Delta t \tag{2.10}$$

Based on Equation 2.4 (discussed earlier), the right-hand side can be further calculated (also shown below). This relationship will be helpful later on.

$$\mathbf{q}\{\bar{\boldsymbol{\omega}}\Delta t\} = \begin{bmatrix} \cos(\|\bar{\boldsymbol{\omega}}\|\Delta t/2) \\ \frac{\bar{\boldsymbol{\omega}}}{\|\bar{\boldsymbol{\omega}}\|}\sin(\|\bar{\boldsymbol{\omega}}\|\Delta t/2) \end{bmatrix} \tag{2.11}$$

By substituting the variables of the first order time-derivative of angular velocity and the median angular rate for their definitions, we reach Equation 2.12. Based on the commuter properties of the quaternion, we can present Equation 2.12 in a more concise form. This finally leads us to Equation 2.13 shown below.

$$\mathbf{q}_{n+1} = \mathbf{q}_n \otimes \mathbf{q}\{\bar{\boldsymbol{\omega}}\Delta t\} + \frac{\Delta t^2}{48}\mathbf{q}_n \otimes (\boldsymbol{\omega}_n \otimes \boldsymbol{\omega}_{n+1} - \boldsymbol{\omega}_{n+1} \otimes \boldsymbol{\omega}_n) + \cdots \tag{2.12}$$

$$\mathbf{q}_{n+1} \approx \mathbf{q}_n \otimes \left(\mathbf{q}\{\bar{\boldsymbol{\omega}}\Delta t\} + \frac{\Delta t^2}{24}\begin{bmatrix} 0 \\ \boldsymbol{\omega}_n \times \boldsymbol{\omega}_{n+1} \end{bmatrix}\right) \tag{2.13}$$

The summation equation 2.13 can be broken into two parts. The first term represents the zeroth order midward integration. The second term represents a second-order correction which disappears when there is no change between $\boldsymbol{\omega}_n$ and $\boldsymbol{\omega}_{n+1}$. Once those two terms of the summation have been calculated, the next step is to find the cross-product of two quaternions. Reference [15] provides that formulation.

However, before we present the equation, the topic of notation needs to be discussed. There are two conventions for presenting a quaternion’s order of elements. Either the real element is listed first, or it is listed last. Neither of the notations has an effect on the fundamental principles of quaternions, but the formulations derived from each notation are different. It is very important for us to not use one notation and then attempt to apply it to an equation based on the other notation without first completing a notation conversion.

This exact situation is relevant to our next step. The equation from reference [14] lists the scalar value first, while the equation from reference [15] lists the scalar value last. This complication is fixed by adjusting the arrays in equation 2.13 to list the first term fourth, as well as insuring that the initial quaternion also lists its scalar value last. With this completed we can now apply the quaternion cross-product as shown in the next section. This gives us a list of quaternions in matrix form with a number of columns (each column a quaternion in chronological order) equal to the number of angular velocity data points used for the experiment.

Conversion of Quaternions to Attitude Matrix

In this section we will present the formula for converting a quaternion to an attitude matrix. This formula returns the direction cosine matrix in terms of the 4x1 Euler parameter vector **q**. The first element is the non-dimensional Euler parameter, while the remaining three elements form the Euler-parameter vector. Here, we explain how that function is derived.

Reference [15] builds the foundation for this proof when discussing vectors in Three Dimensions. At this point we discuss the Euler Axis/Angle Representation of Attitude. Figure 2.1 will be an important visualization of the process discussed in the equations below.

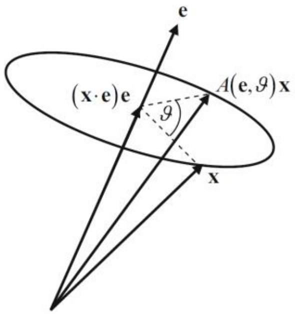

Figure 2.1 Euler axis/angle rotation

Reference [15, p. 41] Figure 2.6

$$\begin{aligned} &\mathbf{x} \equiv An\ arbitrary\ vector \\ &\mathbf{e} \equiv Euler\ axis\ (unit\ vector) \\ &\vartheta \equiv Euler\ angle\ of\ rotation \\ &A \equiv Rotated\ reference\ frame \end{aligned} \tag{2.14}$$

Finally, Reference [15] defines the attitude matrix in Equation 2.15, with the following trigonometric terms labeled:

$$A(\mathbf{e},\vartheta) = \begin{bmatrix} c + (1-c)e_1^2 & (1-c)e_1e_2 - se_3 & (1-c)e_1e_3 - se_2 \\ (1-c)e_2e_1 - se_3 & c + (1-c)e_2^2 & (1-c)e_2e_3 - se_1 \\ (1-c)e_3e_1 - se_2 & (1-c)e_3e_2 - se_1 & c + (1-c)e_3^2 \end{bmatrix} \tag{2.15}$$

$$c \equiv \cos\vartheta \quad \text{and} \quad s \equiv \sin\vartheta$$

Here it appears that there are four independent variables that affect the attitude matrix (e_1, e_2, e_3, and the Euler angle of rotation), but because the magnitude of e is constrained to equal one, this equation only depends on three parameters. This combined with the definition of the unit quaternion, finally gives the quaternion representation of the attitude matrix. This is shown below in equation 2.16.

$$\begin{aligned} A(\boldsymbol{q}) &= (q_4^2 - \|\boldsymbol{q}_{1:3}\|^2)\boldsymbol{I}_3 - 2q_4[\boldsymbol{q}_{1:3}\times] + 2\boldsymbol{q}_{1:3}\boldsymbol{q}_{1:3}^{\mathrm{T}} \\ &= \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_2q_1 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_3q_1 + q_2q_4) & 2(q_3q_2 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \end{aligned} \qquad (2.16)$$

This is the equation that is used to convert quaternions into attitude matrices in our conversion function.

Data Simulation Function

At this point we discuss the process of designing a data simulation function. This is the function that takes the calculated quaternion data, applies that data to the conversion function discussed earlier, and builds an equation in Refence [11] which calculates the simulation-based (or data-based) measurement of the magnetic field by a magnetometer, otherwise known as the variable $\tilde{\boldsymbol{B}}_k$. But before we can reach that point, there are a number of intermediate steps that we must cover. First, we analyze the variable labeled as $\boldsymbol{H}_k$ in the Equation 2.17 shown below.

$$\widetilde{\boldsymbol{B}}_k = \boldsymbol{A}_k\boldsymbol{H}_k + \boldsymbol{b} + \boldsymbol{\epsilon}_k, \quad k = 1, 2, \dots, m \qquad (2.17)$$

Using reference [11], we can define $\boldsymbol{H}_k$ as the value of the geomagnetic field with respect to the Earth-fixed coordinate system. This geomagnetic field can be assumed to be approximately constant for certain types of measurements where the range of motion of the sensor is limited to a local area. This geomagnetic field can also be assumed to fluctuate significantly over time. This assumption can be most easily validated by the application of satellites orbiting Earth. The values of $\boldsymbol{H}_k$ used for our research will be provided in Chapter 3.

Once the value of $\boldsymbol{H}_k$ is determined, the next piece of step of the data simulation function is the incorporation of a norm preserving redundancy. As explained earlier, one of the attributes of a correctly calculated quaternion is that the magnitude of its norm is always equal to 1. The equations used to calculate the unit quaternions are based on that premise. However, there are opportunities for error. If the computer calculations give a quaternion a norm that is very slightly different than 1 because of rounding, and if that quaternion is used to help calculate the next quaternion, their individual errors have the ability to compound. This increases the chances for future error in the code. However, there is a simple solution to this opportunity for error. Reference [16] explains that the easiest way to make sure that all of the attitude quaternions have a magnitude of 1 is to normalize each quaternion before they are used to calculate another variable. Normalizing a quaternion converts it into a unit quaternion. Reference [16] shows that this is accomplished by dividing the quaternion by the norm of itself as shown in Equation 2.18 below.

$$\mathbf{q}_{unit} = \frac{\mathbf{q}}{\|\mathbf{q}\|} \tag{2.18}$$

Now we can be confident that every quaternion that is used to determine an attitude matrix is a unit quaternion.

The next step within the data simulation function is to convert the quaternions to the attitude matrix. This is achieved by using the conversion function that was described earlier. After determining the attitude matrices for each moment in time, we then define the following variable $\boldsymbol{B}_k$ using an equation from Reference [11] shown below. $\boldsymbol{B}_k$ requires three variables in order to find its value. First it needs the attitude matrix $\boldsymbol{A}_k$ which we just calculated using the conversion function. Next it needs the value of the geomagnetic field $\boldsymbol{H}_k$ which we will approximate in Chapter 3. Finally, it needs a value for the magnetometer's bias $\boldsymbol{b}$. In another section we approximate the bias of the smartphone's magnetometer that was used in our local experiment. However, for the purpose of this data simulation function, we must be able to use this function as a template no matter what the value of the bias is.

$$\boldsymbol{B}_k \triangleq \boldsymbol{A}_k \boldsymbol{H}_k + \boldsymbol{b} \tag{2.19}$$

So, similarly to the way that the variable $\boldsymbol{H}_k$ can be calculated we can now use a random value generator and multiply it by an order of magnitude scaling value for bias which can be found as values within the range of ±0.8 as shown in the equation below. This proof for this estimated range will be provided in the following subheading. The main difference between the random value function used to determine $\boldsymbol{H}_k$ and the random value function used to find $\boldsymbol{b}$ is that $\boldsymbol{b}$ is defined to have the same random number generated at every iteration of an experiment, while $\boldsymbol{H}_k$ is defined so that different random numbers can be generated at every iteration of the experiment. This is important because while both variables need to be able to take any value based on the needs of the experiment, over the course of the experiment, the value for $\boldsymbol{b}$ should always remain constant, while the value for $\boldsymbol{H}_k$ must be able to change for each iteration. This is because aside from any large changes in external bias (such as solar flares), the bias of a magnetometer is relatively constant, while the value of the geomagnetic field can have the potential to change very quickly over small changes in distance. An approximation for the bias will be presented in Chapter 3.

Output of the Data Simulation Function

Once we determine a range of values for $\boldsymbol{B}_k$ in Chapter 3, we can use a relationship provided by Reference [11] to find the variable $\tilde{\boldsymbol{B}}_k$ discussed at the beginning of this section. This relationship is derived from Equation 2.17 and Equation 2.19. It is illustrated in Equation 2.20 shown below. The only additional variable is the measurement noise vector $\boldsymbol{\epsilon}_k$. The measurement noise vector is also calculated by using the random value function.

$$\widetilde{\boldsymbol{B}}_k = \boldsymbol{B}_k + \boldsymbol{\epsilon}_k \tag{2.20}$$

According to Reference [11], the measurement noise vector is calculated with the assumption that it is zero-mean Gaussian. The term Gaussian is simply another way of saying the distribution should be normal, and the term zero-mean indicates that the average of the distribution should be zero. Fortunately, the random value function meets both of those requirements perfectly. The other factor in determining the measurement noise vector is the scaling value that the random value function is multiplied by. In our experiment we assign a value of 5 mG to the scaling value, as that provides a reasonable resulting noise vector. However, the average scale for magnetic noise can vary significantly on the location of measurement. Because of this, it is important to be able to quickly modify the scale value of the measurement noise based on the needs of the experiment.

At this point the process for completing an iteration within the data simulation function is finished. The number of iterations chosen to run in the function is equal to the number of quaternion vectors in the run, also known as the variable *n*. This means that with the input of the number of quaternion vectors *n*, the values of the quaternion vectors, the value of the constant bias $\boldsymbol{b}$, and the scaling values for both the value of the geomagnetic field and the measurement noise, we end up with the output of the value of the geomagnetic field $\boldsymbol{H}_k$ and the measurement of the magnetic field $\tilde{\boldsymbol{B}}_k$.

Comparing the Calibration Methods

This is the section that discusses the calibration methods that optimize the bias measurements. The accuracy of these functions will be compared to one another in the results chapter. As far as similarities go, all of the functions list the same inputs of $\tilde{\boldsymbol{B}}_k$, $\boldsymbol{H}_k$, σ, and *n*. Each of these methods use a Matlab optimization function called *fmincon*.

Before applying the minimization function, we define the option settings of this function. First, we define the algorithm that will be used. Here we pick the Interior Point Algorithm. This algorithm is discussed in detail in Reference [17]. Next, we define the maximum number of iterations allowed before termination, the maximum number of function evaluations before termination, and the termination tolerance on the function value. Here we set the maximum number of iterations at 5,000. We set the maximum number of function evaluations at 50,000. We set the termination tolerance at 1e-6.

The next step is to set up the optimization function. Before we describe the optimization function, we should derive those algorithms built specifically for the TLS method, the Method 1 function, and the Method 2 function. This begins by defining the measurement model as shown in Equation 2.17. It should be noted here that bias and noise are not the only factors relevant to sensor calibration. There are other elements as well, including scale factors and nonorthogonality corrections. These misalignment matrix elements are not considered in our comparison of the

TLS method and our new methods. For our purposes, the bias vector $\boldsymbol{b}$ and the measurement noise vector $\boldsymbol{\epsilon}_k$ are the only calibration parameters.

Next, we present a simplified model for the noise variable. As discussed previously, the first characteristic of the noise variable is that it is zero mean. This property is conveyed by the equation below.

$$E\{\boldsymbol{\epsilon_k}\} = 0 \tag{2.21}$$

The second characteristic of the noise variable is that it has a covariance as shown in the equation below.

$$E\{\boldsymbol{\epsilon_k}\boldsymbol{\epsilon}_{\boldsymbol{k}}^{T}\} = \sigma_{\boldsymbol{k}}^{2} I \tag{2.22}$$

The final characteristic of the noise variable is that it is classified as white noise. This means that the variables contributing to the noise are independent and there is no correlation between the magnitude of the noise at one point and the magnitude of the noise at any other point. This concept is illustrated in the equation below.

$$E\{\boldsymbol{\epsilon}_j\boldsymbol{\epsilon}_k^{\boldsymbol{T}}\} = \boldsymbol{0} \tag{2.23}$$

It should be noted that the literature supporting the TLS method in Reference [11] assumes a more general form of the covariance matrix than the one used here. Also, an important distinction regarding notation must be made. The noiseless measurement of the magnetic field is labeled $\boldsymbol{B_k}$. The definition of this variable was presented in Equation 2.19. Depending on the convention used, the variable $\boldsymbol{B_k}$ can also represent the “True” measurement of the magnetic field, that is, what the value of the measurement would be without noise or bias. This alternative definition is shown below.

$$\boldsymbol{B}_k(\mathrm{Alt}) \triangleq \boldsymbol{A}_k\boldsymbol{H}_k \tag{2.24}$$

This different definition is acceptable to use, but it is important to be consistent and not to start with one definition and then switch to the other definition when comparing the different algorithms. Based on our convention, there are two resulting estimated quantities. The first estimated quantity is the bias itself, and the second estimated quantity is the measurement of the magnetic field. The presence of an estimated variable is denoted by a hat symbol above the variable names $\hat{\boldsymbol{b}}$ and $\hat{\boldsymbol{B}}_k$.

Next, it should be noted that the proofs of the TLS Method, Method 1, Method 2 and Method 3 all assume that the value of the geomagnetic field defined by the Earth-based coordinate system $\boldsymbol{H}_k$ is known. This is not necessarily true in the real world, but it is a useful approximation when comparing calibration methods. Finally, it is helpful view each algorithm as a corresponding optimization problem. In the next subsections, we provide the proofs for those optimization problems.

The TLS Method Function

This first function has already been derived in Reference [11]. It uses the Total Least Square method. The cost function is shown in the Equation below.

$$J(\boldsymbol{b}, \boldsymbol{B}_k) = \frac{1}{2}\sum_{k=1}^{m}\begin{bmatrix} 2(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k) \\ \|\widetilde{\boldsymbol{B}}_k\|^2 - \|\boldsymbol{B}_k\|^2 - 2\boldsymbol{B}_k^{\mathrm{T}}\boldsymbol{b} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{b} - \mu_k \end{bmatrix}^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1} \begin{bmatrix} 2(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k) \\ \|\widetilde{\boldsymbol{B}}_k\|^2 - \|\boldsymbol{B}_k\|^2 - 2\boldsymbol{B}_k^{\mathrm{T}}\boldsymbol{b} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{b} - \mu_k \end{bmatrix} \tag{2.25}$$

The variables μ_k and $\boldsymbol{\Sigma}_k$ are defined in Reference [11]. In this thesis we are using a simplified version of these variables because we have assumed the covariance of Equation 2.22. This optimization problem does not have any constraint. In all attitude-independent algorithms, the following relationship is used to eliminate the variable “$\boldsymbol{A}_k$” from the calculations.

$$\|\boldsymbol{A}_k\boldsymbol{H}_k\| = \|\boldsymbol{H}_k\| \tag{2.26}$$

The TLS algorithm uses this relationship as an effective measurement through the following interpretation of Equation 2.17.

$$\boldsymbol{A}_k\boldsymbol{H}_k = \widetilde{\boldsymbol{B}}_k - \boldsymbol{b} - \boldsymbol{\epsilon}_k \tag{2.27}$$

$$\|\boldsymbol{A}_k\boldsymbol{H}_k\| = \|\widetilde{\boldsymbol{B}}_k - \boldsymbol{b} - \boldsymbol{\epsilon}_k\| \tag{2.28}$$

$$\|\boldsymbol{H}_k\| = \|\widetilde{\boldsymbol{B}}_k - \boldsymbol{b} - \boldsymbol{\epsilon}_k\| \tag{2.29}$$

By continuing to apply Equation 17 from Reference [11], Equation 2.29 can be used to derive the following equation.

$$\|\widetilde{\boldsymbol{B}}_k\|^2 - \|\boldsymbol{H}_k\|^2 = 2\boldsymbol{B}_k^{\mathrm{T}}\boldsymbol{b} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{b} + (2\widetilde{\boldsymbol{B}}_k^{\mathrm{T}}\boldsymbol{\epsilon}_k - \boldsymbol{\epsilon}_k^{\mathrm{T}}\boldsymbol{\epsilon}_k) \tag{2.30}$$

The TLS estimates $\hat{\boldsymbol{b}}$ and $\hat{\boldsymbol{B}}_k$ are what minimize the cost function. The minimization problem can be made simpler based on the relationship between $\hat{\boldsymbol{B}}_k$ and $\hat{\boldsymbol{b}}$, which is obtained from the necessary condition for optimality. Next, we apply the estimate for $\boldsymbol{B}_k$ from Reference [11] to Equation 2.30 which is presented here.

$$\widehat{\boldsymbol{B}}_k = \widetilde{\boldsymbol{B}}_k - \frac{\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k - \mu_k}{2\left(\boldsymbol{B}_k - \widehat{\boldsymbol{b}}\right)^{\mathrm{T}}\boldsymbol{\Sigma}_k(\boldsymbol{B}_k - \boldsymbol{b}) + \mathrm{tr}(\boldsymbol{\Sigma}_k)^2} \times \boldsymbol{\Sigma}_k\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) \tag{2.31}$$

Every value of the variable $\hat{\boldsymbol{B}}_k$ can be determined once the value of the variable $\hat{\boldsymbol{b}}$ has been found. With this relationship, we can have an equivalent way of finding the estimates. At this point, we solve the equivalent optimization problem with the following cost function for the variable $\hat{\boldsymbol{b}}$. From there, we calculate $\hat{\boldsymbol{B}}_k$. This equation is important because it will be used later to derive a formula which will calculate the constraint error of the different functions.

$$J(\boldsymbol{b}) = \frac{1}{2}\sum_{k=1}^{m} \frac{\left[\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{b}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{b}\right) - \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k - \mu_k\right]^2}{4\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{b}\right)^{\mathrm{T}}\boldsymbol{\Sigma}_k\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{b}\right) + 2\mathrm{tr}(\boldsymbol{\Sigma}_k)^2} \tag{2.32}$$

The Method 1 Function

The goal of this new function, as presented in this thesis, is to be a norm-preserving function. What this means is that when the constraint error discussed in a previous section is measured after the bias is estimated, that constraint error should not simply approximately equal zero (as is the case with the TLS estimate) but that constraint error should exactly equal zero to the smallest order of magnitude available to the computing device used. The purpose of including this constraint is to improve the accuracy of the bias estimation by limiting the opportunity for the minimization function to converge to a point further away from the true bias.

It should be pointed out that for this method there is not only one cost function but a nonlinear constraint function as well. The first difference between the TLS method and Method 1 is that Method 1 does not utilize the effective measurement in Equation 2.30. Instead, it solves a constrained optimization problem which ensures the following relationship.

$$\|\boldsymbol{H}_k\| = \left\|\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\| \tag{2.33}$$

The cost function of Method 1 is shown in the equation below.

$$J(\boldsymbol{b}, \boldsymbol{B}_k) = \frac{1}{2}\sum_{k=1}^{m} \frac{1}{\boldsymbol{\sigma}_k^2}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right) \tag{2.34}$$

Here, we present the nonlinear constraint and equivalent forms of the same constraint. The last equation is the form that will be used within the optimization algorithm.

$$\|\boldsymbol{B}_k - \boldsymbol{b}\| = \|\boldsymbol{H}_k\| \tag{2.35}$$

$$\|\boldsymbol{B}_k - \boldsymbol{b}\|^2 = \|\boldsymbol{H}_k\|^2 \tag{2.36}$$

$$(\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b}) = \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k \tag{2.37}$$

The Method 2 Function

The third function that we will discuss is the Method 2 function. The goal of this new function is to maintain the norm-preserving attribute of the Method 1 function while producing those resulting estimates within an objectively shorter period of time. With Method 2, we only build one cost function before applying it to the optimization function. Doing this will allow Method 2 to maintain simplicity and hopefully achieve results faster than Method 1. It should also be noted that Method 2 yields the exact same estimates as Method 1

The goal of Method 2 is to find the estimated bias $\hat{\boldsymbol{b}}$ that minimizes the cost function $J(\boldsymbol{b})$ and calculate the estimated measurement $\hat{\boldsymbol{B}}_k$ from the estimated bias $\hat{\boldsymbol{b}}$. Here we use the method of Lagrange multipliers as shown in the equation below.

$$\begin{aligned} J(\boldsymbol{b}, \boldsymbol{B}_k) = \frac{1}{2}\sum_{k=1}^{m}\left[\frac{1}{\boldsymbol{\sigma}_k^2}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right)\right] \\ + \frac{1}{2}\sum_{k=1}^{m}\lambda_k\left[(\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b}) - \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k\right] \end{aligned} \tag{2.38}$$

In this case there is a set of three optimality conditions based on the following three partial derivatives.

$$\frac{\partial J}{\partial \boldsymbol{B}_k} = 0 \tag{2.39}$$

$$\frac{\partial J}{\partial \boldsymbol{b}} = 0 \tag{2.40}$$

$$\frac{\partial J}{\partial \lambda_k} = 0 \tag{2.41}$$

Based on those three partial derivatives, the three optimality conditions are as follows.

$$-\boldsymbol{\sigma}_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}_k\right) + \lambda_k\left(\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) = 0 \tag{2.42}$$

$$\sum_{k=1}^{m} \lambda_k\left(\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) = 0 \tag{2.43}$$

$$\left(\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right)^{\mathrm{T}}\left(\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k = 0 \tag{2.44}$$

Based on the first optimality condition, we derive the following equation.

$$\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}} = \frac{\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}}{(1 + \lambda_k \boldsymbol{\sigma}_k^2)} \tag{2.45}$$

The Lagrange Multiplier can be calculated based on this newly defined relationship in Equation 2.46 below. By setting Equation 2.45 and 2.46 equal to each other, we derive Equation 2.47. At this point, we can keep the Lagrange Multiplier on the left side of the equation and move all the other terms to the right side of the equation. This leaves us with a solution for the Lagrange Multiplier in Equation 2.48 below.

$$\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}} = \frac{\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}}{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|}\left\|\boldsymbol{H}_k\right\| \tag{2.46}$$

$$1 + \lambda_k \boldsymbol{\sigma}_k^2 = \frac{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|}{\|\boldsymbol{H}_k\|} \tag{2.47}$$

$$\lambda_k = \boldsymbol{\sigma}_k^{-2} \left(\frac{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|}{\|\boldsymbol{H}_k\|} - 1 \right) \tag{2.48}$$

At this point, we calculate the cost function *J(**b**)*. Our first step is to rearrange the variables in Equation 2.46 and perform a number of algebraic operations. Through this process we arrive at Equation 2.51.

$$\widehat{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}} = \frac{\|\boldsymbol{H}_k\|}{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|} \left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) \tag{2.49}$$

$$\widehat{\boldsymbol{B}}_k = \frac{\|\boldsymbol{H}_k\|}{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|} \left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) + \widehat{\boldsymbol{b}} \tag{2.50}$$

$$\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}_k = \frac{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\| - \|\boldsymbol{H}_k\|}{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|} \left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) \tag{2.51}$$

The next step is to include the variance of the noise on both sides of the equation. This is accomplished in Equation 2.52 and that equation is further simplified as shown in Equation 2.53 below.

$$\boldsymbol{\sigma}_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}_k\right) = \boldsymbol{\sigma}_k^{-2}\frac{\left(\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\| - \|\boldsymbol{H}_k\|\right)^2}{\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\|^2}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) \qquad (2.52)$$

$$\boldsymbol{\sigma}_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}_k\right) = \boldsymbol{\sigma}_k^{-2}\left(\left\|\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right\| - \|\boldsymbol{H}_k\|\right)^2 \qquad (2.53)$$

This means that when the estimated measurement $\hat{\boldsymbol{B}}_k$ and the estimated bias $\hat{\boldsymbol{b}}$ are related through the newly defined Equation 2.46, the cost functions can be determined as shown below. The cost function as a function of both measurement and bias is determined by the left side of Equation 2.53. The cost function as a function of only bias is determined by the right side of Equation 2.53 as shown below.

$$J(\boldsymbol{b}, \boldsymbol{B}_k) = \frac{1}{2}\sum_{k=1}^{m}\boldsymbol{\sigma}_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right) \qquad (2.54)$$

$$J(\boldsymbol{b}) \triangleq J\left(\boldsymbol{b}, \boldsymbol{B}_k(\boldsymbol{b}_k)\right) = \frac{1}{2}\sum_{k=1}^{m}\boldsymbol{\sigma}_k^{-2}\left(\left\|\widetilde{\boldsymbol{B}}_k - \boldsymbol{b}\right\| - \|\boldsymbol{H}_k\|\right)^2 \qquad (2.55)$$

The Method 3 Function

With the new Method 3 function, there is a very important distinction that must be made when compared to the other methods. The goal of this method is not to improve the general accuracy or computation speed of the other methods. Here, we attempt to solve a specific minimization problem where we assume that the value of the geomagnetic field with respect to the Earth-based coordinate system is unavailable. What is available is a measurement of that field $\tilde{\boldsymbol{H}}_k$. This measurement varies over time regardless of the location being measured, and it is defined in the equation below. Here, the geomagnetic field noise is defined as zero-mean Gaussian with a variance also identified below.

$$\boldsymbol{\epsilon}_{\boldsymbol{H},k} = \text{Geomagnetic Field Noise} \tag{2.56}$$

$$\boldsymbol{\sigma}_{\boldsymbol{H},k}^{2} = \text{Variance of } \boldsymbol{\epsilon}_{\boldsymbol{H},k} \tag{2.57}$$

$$\widetilde{\boldsymbol{H}}_k = \boldsymbol{H}_k + \boldsymbol{\epsilon}_{\boldsymbol{H},k} \tag{2.58}$$

Aside from addition of the relationship above, the measurement model for Method 3 is the same as the measurement model for the other methods presented in Equation 2.17. Additionally, the noiseless measurement $\boldsymbol{B}_k$ has the same definition as previously shown in Equation 2.19. The norm constraint for Method 3 is defined by Equation 2.59 below.

$$\|\boldsymbol{B}_k - \boldsymbol{b}\| = \|\boldsymbol{H}_k\| \tag{2.59}$$

Based on Equation 2.58, the following identity can be derived and algebraically manipulated as shown in the equations below.

$$\widetilde{\boldsymbol{H}}_k^{\mathrm{T}} \widetilde{\boldsymbol{H}}_k = \left(\boldsymbol{H}_k + \boldsymbol{\epsilon}_{\boldsymbol{H},k}\right)^{\mathrm{T}} \left(\boldsymbol{H}_k + \boldsymbol{\epsilon}_{\boldsymbol{H},k}\right) \tag{2.60}$$

$$\widetilde{\boldsymbol{H}}_k^{\mathrm{T}}\widetilde{\boldsymbol{H}}_k = \boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k + \left[2\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k} + \boldsymbol{\epsilon}_{\boldsymbol{H},k}^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k}\right] \tag{2.61}$$

Based on the constraint defined in Equation 2.59, the following relationship can also be derived through Equation 2.61.

$$\widetilde{\boldsymbol{H}}_k\widetilde{\boldsymbol{H}}_k = (\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b}) + \left[2\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k} + \boldsymbol{\epsilon}_{\boldsymbol{H},k}^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k}\right] \tag{2.62}$$

Next, we define the mean and the variance of the noise as shown in the equations below.

$$\mu_{N,k} \triangleq E\left\{2\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k} + \boldsymbol{\epsilon}_{\boldsymbol{H},k}^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k}\right\} = 3\boldsymbol{\sigma}_{H,k}^2 \tag{2.63}$$

$$\boldsymbol{\sigma}_{N,k}^2 \triangleq E\left\{\left(2\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k} + \boldsymbol{\epsilon}_{\boldsymbol{H},k}^{\mathrm{T}}\boldsymbol{\epsilon}_{\boldsymbol{H},k}\right)^2\right\} = 4\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k\boldsymbol{\sigma}_{H,k}^2 + 6\boldsymbol{\sigma}_{H,k}^4 \tag{2.64}$$

$$\boldsymbol{\sigma}_{N,k}^2 = 4(\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b})\boldsymbol{\sigma}_{H,k}^2 + 6\boldsymbol{\sigma}_{H,k}^4 \tag{2.65}$$

The equations for the mean and the variance of the noise can now be approximated as shown in the equations below.

$$\mu_{N,k} \approx 0 \tag{2.66}$$

$$\boldsymbol{\sigma}_{N,k}^2 \approx 4(\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b})\boldsymbol{\sigma}_{H,k}^2 \tag{2.67}$$

The cost function as a function of both measurement and bias is set equal to the following equation.

$$\begin{aligned} J(\boldsymbol{b}, \boldsymbol{B}_k) = &\frac{1}{2}\sum_{k=1}^{m} \boldsymbol{\sigma}_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right) \\ &+\frac{1}{2}\sum_{k=1}^{m} \boldsymbol{\sigma}_{N,k}^{-2}\left[\widetilde{\boldsymbol{H}}_k\widetilde{\boldsymbol{H}}_k - (\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b}) - \mu_{N,k}\right]^2 \end{aligned} \tag{2.68}$$

This cost function can be simplified by applying the approximations for the mean and variance of the noise as shown in the equation below.

$$\begin{aligned} J(\boldsymbol{b}, \boldsymbol{B}_k) = &\frac{1}{2}\sum_{k=1}^{m} \boldsymbol{\sigma}_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}_k\right) \\ &+\frac{1}{2}\sum_{k=1}^{m} \boldsymbol{\sigma}_{N,k}^{-2}\left[\widetilde{\boldsymbol{H}}_k^{\mathrm{T}}\widetilde{\boldsymbol{H}}_k - (\boldsymbol{B}_k - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{B}_k - \boldsymbol{b})\right]^2 \end{aligned} \tag{2.69}$$

For the sake of simplicity, the inverse of the variance is being treated as a constant. No simple solution is found because the cost function *J* is now quartic in $\boldsymbol{B}_k$ and $\boldsymbol{b}$. For the purpose of finding an approximate solution to this quartic problem, we introduce the following equation and estimate the variables $\boldsymbol{B}'_k$ and $\boldsymbol{b}$.

$$\boldsymbol{B}'_k = \boldsymbol{B}_k - \boldsymbol{b} \tag{2.70}$$

The cost function J is quartic with respect to $\boldsymbol{B}'_k$, but it is quadratic with respect to $\boldsymbol{b}$. This brings us to a new cost function that is a function of $\boldsymbol{B}'_k$ and $\boldsymbol{b}$.

$$\begin{aligned} J(\boldsymbol{b}, \boldsymbol{B}'_k) = & \frac{1}{2}\sum_{k=1}^{m} \sigma_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}'_k - \boldsymbol{b}\right)^{\mathrm{T}}\left(\widetilde{\boldsymbol{B}}_k - \boldsymbol{B}'_k - \boldsymbol{b}\right) \\ & + \frac{1}{2}\sum_{k=1}^{m} \sigma_{N,k}^{-2}\left[\widetilde{\boldsymbol{H}}_k \widetilde{\boldsymbol{H}}_k - (\boldsymbol{B}'_k)^{\mathrm{T}}(\boldsymbol{B}'_k)\right]^2 \end{aligned} \tag{2.71}$$

Finally, we construct a loop which produces updated values for $\hat{\boldsymbol{B}}'_k$ and $\hat{\boldsymbol{b}}$ alternately. In the first step of the loop, we define the following relationship with respect to the cost function in Equation 2.102. We then use the optimization algorithm to find the $\hat{\boldsymbol{B}}'_k$ that minimizes the cost function $J(\hat{\boldsymbol{b}}, \boldsymbol{B}'_k)$.

$$\boldsymbol{b} = \widehat{\boldsymbol{b}} \tag{2.72}$$

In the second step of the loop, we solve for the estimated bias $\hat{\boldsymbol{b}}$ with a fixed value of $\hat{\boldsymbol{B}}'_k$ by using the equation below. We apply the loop continuously until a stable value for the estimated bias has been achieved.

$$\widehat{\boldsymbol{b}} = \frac{\left(\sum_{k=1}^{m} \sigma_k^{-2}\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{B}}'_k\right)\right)}{\sum_{k=1}^{m} \sigma_k^{-2}} \tag{2.73}$$

Deriving the Measurement of Constraint Error

This is the section that covers the derivation of the constraint error equation after the calibration methods are implemented. This section largely discusses work completed in Reference [11]. Specifically, we analyze how the constraint error of the TLS method is calculated and compare the resulting constraint error with the constraint error of our other methods. The TLS estimate is found by minimizing a loss function. Here we may apply an unconstrained optimization algorithm that is standard in order to calculate the optimal $\hat{\boldsymbol{b}}$. This loss function is derived from Equation 2.32.

$$J(\hat{\boldsymbol{b}}) = \frac{1}{2}\sum_{k=1}^{\boldsymbol{m}} \frac{\left[\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)^T\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^T\boldsymbol{H}_k - \mu_k\right]^2}{4\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)^T\boldsymbol{\Sigma}_k\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) + 2\mathrm{tr}(\boldsymbol{\Sigma}_k^2)} \tag{2.74}$$

Next, the partial derivative with respect to $\hat{\boldsymbol{b}}$ of Equation 2.74 is presented as Equation 2.75 shown below. Here the variable s_k is also defined below as one half of the trace operator applied to the covariance squared in the Equation below.

$$\begin{aligned}\boldsymbol{g} \triangleq \frac{\partial J}{\partial \hat{\boldsymbol{b}}} = \frac{1}{4}\sum_{k=1}^{m} & -\frac{2\left[\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)^T\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^T\boldsymbol{H}_k - \mu_k\right]}{\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)^T\boldsymbol{\Sigma}_k\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) + s_k}\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) \\ & + \frac{\left[\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)^T\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^T\boldsymbol{H}_k - \mu_k\right]^2}{\left[\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)^T\boldsymbol{\Sigma}_k\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right) + s_k\right]^2}\boldsymbol{\Sigma}_k\left(\tilde{\boldsymbol{B}}_k - \hat{\boldsymbol{b}}\right)\end{aligned} \tag{2.75}$$

$$s_k \triangleq 0.5\mathrm{tr}(\boldsymbol{\Sigma}_k^2) \tag{2.76}$$

Here, Reference [11] provides the estimate of $\boldsymbol{B}_k$ as shown in Equation 2.31 earlier. below. This is based upon the results taken from the partial derivative in Equation 2.75. Next, we reiterate the definition of the error-model for the bias estimate. This is provided in Equations 2.77 shown below.

$$\widehat{\boldsymbol{b}} = \boldsymbol{b} + \boldsymbol{\Delta b} \tag{2.77}$$

If we substitute the definitions of $\tilde{\boldsymbol{B}}_k$ and $\hat{\boldsymbol{b}}$ from Equations 2.20 and 2.78 respectively into the numerator of Equation 2.31, and if we approximate the result by ignore any higher order terms, we arrive at Equation 2.78 shown below.

$$\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right)^T \left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^T \boldsymbol{H}_k - \mu_k \approx 2(\boldsymbol{B}_k - \boldsymbol{b})^T (\boldsymbol{\epsilon}_k - \boldsymbol{\Delta b}) \tag{2.78}$$

This finally brings us to the equation we have been looking for. Here we ignore the smallest terms and set the rest equal to zero. The equation shown below is a measurement of the constraint approximation results used in the TLS method. Here we will also measure the constraint results of our new method and compare those results to the TLS method.

$$\left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right)^T \left(\widetilde{\boldsymbol{B}}_k - \widehat{\boldsymbol{b}}\right) - \boldsymbol{H}_k^T \boldsymbol{H}_k = 0 \tag{2.79}$$

The Optimization Function

In this subheading we discuss MATLAB’s optimization function. As explained in Reference [17], the function is a nonlinear capable solver built into the Matlab optimization toolbox. It can serve a number of purposes, as well as process a number of inputs. The first input that it reads is the function being minimized. For our purposes, the functions to minimize are the cost functions derived from the TLS Method, Method 1, and Method 2 functions. The next input that the function reads is the Initial point. For our new Method 1 and Method 2 functions, this initial point is set at an array of zeros. For the TLS Method function, the initial point is set at an array with the first three values represented by zero, and the remaining values represented by the variable $\tilde{\boldsymbol{B}}_k$.

After the input of the initial point, there are a number of inputs available but most of them will be left unused by our functions. These include inputs for linear inequality constraints, linear equality constraints, lower bounds, and upper bounds. The last input that the optimization function reads is the nonlinear constraints. For the TLS Method and our Method 2 functions, we use no nonlinear constraints. For the Method 1 function, we do have a nonlinear constraint input as described earlier. With all the inputs filled, the Optimization function provides an output of the variable $\hat{\boldsymbol{b}}$. The variable $\hat{\boldsymbol{b}}$ transposed translates to the estimated bias. At this point, we have reached the end of our setup, and we are ready to produce results.

CHAPTER III

RESULTS

Here we discuss the results. First, we will reproduce baseline magnetometer readings in order to verify our initial setup, and we will approximate a range of sensor biases that we will be applying to our optimization algorithms. Then, we will test our measurement verification formula. Next, we will compare the accuracy of our optimization functions. Finally, we will compare the speed in which those functions complete their calculations. Before providing results, we should note that all measurements of magnetic field are in units of milligauss (mG) unless otherwise specified.

Creating a Baseline Reading

The purpose of this first subheading within Chapter 3 is to present the results that lead to a simulated "true" magnetometer reading that will be used to compare the different calibration methods. The first factor to be presented will be the gyroscopic data that will be converted to quaternions. The second factor to be presented will be the magnitude of the local magnetic field. Together, these two factors will provide a zero-bias, noiseless simulation of magnetometer data.

Source of the Attitude Data

Our team decided to use the gyroscope built into a smartphone. The specifications of this gyroscope are as follows: Its manufacturer is STMicroelectronics. Its product number is

L3G4200D. It is described as a 3-Axis MEMS Gyroscope, and its axis, range and sensitivity are shown in the figure below. [18]

Table 3.1 Gyroscope Product Attributes

Axis	X (Pitch), Y (Roll), Z (Yaw)
Range °/s	±250, 500, 2000
Sensitivity (LSB/(°/s))	8.75 ~ 70

Reference [18]

The name of the smartphone application used to record the gyroscopic data from the sensor is PHYPHOX. It was developed by RWTH Aachen University for the following function: to document the raw data from the phone's gyroscope. While the specs are given in units of degrees/s, the output of the application is a rotation rate given in units of radians/s. [19] Here, we applied a rotational motion to the phone and recorded the angular velocity for a period of time sufficient to generate and capture regular periodic motion. Within our sample, just under two thousand points of data were recorded at a rate of 0.01 seconds between each gyroscopic measurement. A link to the raw gyroscopic data used for our research is provided in Reference [20].

Here we observe that there are two factors pertaining to the gyroscopic data that will affect the accuracy of our results. First, there is the size of our sample. This is obvious for a couple reasons. The more data we record, the less likely that error such as bias will have a significant impact on the data. Also, we can observe that the quantities of vectors ***n*** used should reflect a change in the direction of the vectors. If a change in direction is not visible, that would indicate that the size of the sample is not sufficiently large. Second, there is the rate of our sample. The affect of this variable on accuracy is dependent on the angular acceleration of the object itself. For example, in the case of a phone sensor completing many revolutions in a minute, the angular acceleration is high, and the sampling rate must also be relatively high. If it is not, the accurate motion of the sensor will not be accurately captured. However, in the case of a satellite completing few revolutions in a day, the angular acceleration is lower, and the sampling rate can remain relatively low. The accurate motion of the satellite will still be captured. At higher resolutions, the sample rate can effectively remove noise from the results.

The potential impact of this sample size and sample rate that is noted in this section can be addressed in further detail through future research. With this data from the gyroscope saved, the next step was to convert the data to quaternions through the computation programing platform called Matlab that we use for the duration of the experiment. The remaining steps in our experiment can be found within our Matlab code. [21]

Value of the Local Magnetic Field

For the purposes of estimating the magnetic values of our smartphone-sensor experiment, we used value of $\boldsymbol{H}_k$ found in reference [23]. In order to calculate the estimated magnetic field through this reference, we had to first provide the latitude, longitude, and elevation of the experiment’s location. After that we had to pick a specific model that would calculate the results. There were three options for modeling to choose from: the World Magnetic Model (WMM), the International Geomagnetic Reference Field (IGRF), and the Enhanced Magnetic Model (EMM).

According to Reference [24], there is notable differences between the IGRF model and the other models. The WMM model and the EMM model are what are called predictive-only models. This means that those models take measurements at a specific time and calculate predictions for what the future measurements would be. These predictions can be very accurate, but the longer amount of time the calculations predict from the original measurements, the more inaccurate those predictions become. Because of this, the WMM model has an expiration date for each set of predictions, in this case 5 years. Five years after each set of measurements and predictions, the WMM model has to complete a new set of measurements and predictions.

So, what makes the IGRF model different? In terms of future predictions, the IGRF model has a comparable level of accuracy to the WMM model. The main difference is that the IGRF model is retrospectively updated. This means that once the prediction becomes outdated, the IGRF researchers go back and replace the prediction value with a calculated approximation of the actual magnetic field in a given location at that given time. It should be noted that IGRF update covers any time between the years 1900 and 2000. This is not necessarily applicable to an experiment done in the Fall of 2021, but the IGRF method's usefulness extends to an algorithm that should be versatile if a future researcher needs it to use recorded values of past magnetic field calculations. For this reason, we will be using the IGRF method. After picking the IGRF model, the menu of Reference [23] asks for a start date, an end date the step size between points.

However, for the purposes of building the data simulation function, it should not matter what the value of $\boldsymbol{H}_k$ is as long as the magnitudes are a reasonable size. Regardless of the location of an experiment relative to the Earth's magnetic field, part of the goal of our research is to create a data simulation function that will be a versatile template for any researcher in any situation. For this reason, we created an option to test results with the variable $\boldsymbol{H}_k$ by using a function that generates random values. This function is described in Reference [21] as a random scalar value taken from a standard normal distribution. The output of this function is multiplied by the normal order of magnitude for values of the geomagnetic field with respect to Earth. This magnitude can be approximated to equal 45 micro-tesla as shown in Reference [24]. For the purposes of our experiment, we use the approximated values of $\boldsymbol{H}_k$ in the x, y, and z directions from Reference [23] as an array equal to north, east, and vertical components of the magnetic field in the table shown below at the date 2022-01-01.

Table 3.2 Local Magnetic Field – Starkville, MS

Model Used:	WMM-2020						
Latitude: **Longitude:** **Elevation:**	33° 26’ 52” N 88° 48’ 40” W 0.0 km Mean Sea Level						
Date	**Declination**	**Inclination**	**Horizontal Intensity**	**North Comp**	**East Comp**	**Vertical Comp**	**Total Field**
2022-01-01	-2° 19’ 22”	62° 1’ 22”	22,873.6 nT	22,854.8 nT	-927.0 nT	43,060.4 nT	48,758.6 nT
Change/ Year	-0° 5’ 11” /yr	-0° 4’ 41” /yr	5.5 nT/yr	4.1 nT/yr	-34.7 nT/yr	-131.0 nT/yr	-113.1 nT/yr
Uncertainty	0° 22’	0° 13’	128 nT	131 nT	94 nT	157 nT	145 nT

Reference [23]

This magnitude of the magnetic field can be converted from Nanotesla (nT) to Milligauss (mG) through a toolset provided by Reference [25]. The value of the total magnetic field is now presented as approximately 490 mG. This will be important information when we confirm the baseline readings and apply the optimization algorithms.

Baseline Readings

Now that we have the necessary information, we can simulate the baseline readings. The first results are the baseline magnetometer readings taken from the initial gyroscopic data. The purpose of these readings is not to replicate realistic magnetometer data, but rather to verify the initial conditions, and to allow us to control variables such as the true bias. The experimental magnetometer readings are shown in Figure 3.1 below. The result reflects the circular motion of the sensor during testing.

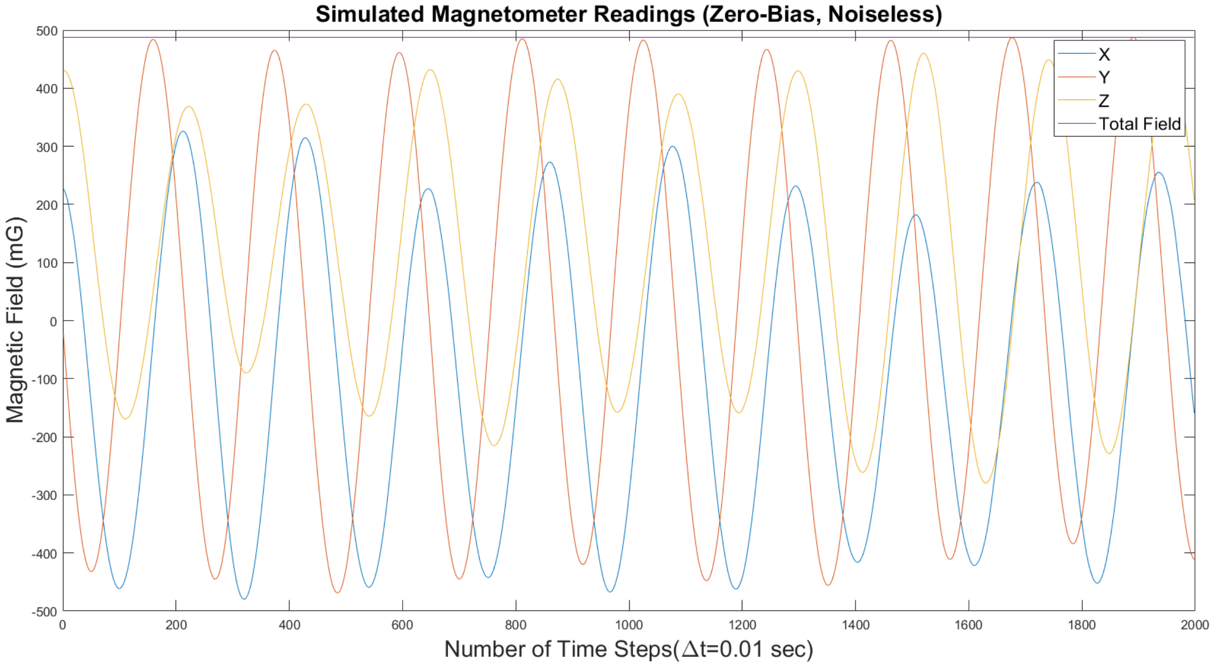

Figure 3.1 Simulated Magnetometer Readings (Zero-Bias, Noiseless)

Realistic Range for Bias Testing

In this section we will determine a proper approximation for the bias of the magnetometer used in our local experiment. The sensor used to measure the magnetic field of our experiment is a component included in Apple's iPhone 6. While much of the information regarding internal technology of the iPhone is proprietary, based on the research done, we have determined that it is likely an AK8963 3-axis magnetometer device. This device is developed by the Asahi Kasei Microdevices Corporation (AKM), and as shown in Reference [26], it is used by the smartphone's operating system as an electronic compass IC with high sensitivity Hall sensor technology. It has an output data resolution of 0.15 μT/LSB (least-significant bit). It has a measurement range of ±4900 μT.

Since the documentation within Reference [26] does not provide any specific details regarding the bias of the smartphone sensor, we want to find a way to approximate the bias through analogy. The closest comparison that we were able to find is found in the research completed in Reference [6]. Here is used the smartphone called an LG Nexus 4. It precedes the iPhone used in our local experiment by only a couple years. It should be noted that while it is helpful to our experiment to find a reasonable approximation for the magnetometer's bias, it is not required for our goal of creating a templet to measure bias in any situation. The equations should be designed to run successfully regardless of the expected bias.

As shown in the Tables 2.3 and 2.4 from Reference [6], when the value of the heading estimation σ is lowest, the results for the bias estimation are the most accurate. By analyzing the columns of data under the EXP1L and EXP2S headings, we can find a consistent approximation for the bias of a laboratory sensor. By analyzing the columns of data under the EXP3L and EXP4S headings, we can find a consistent approximation for the bias of a smartphone sensor.

Table 3.3 Estimated Bias of a Laboratory Sensor

	EXP1L - LABORATORY				EXP2S - LABORATORY			
	$\hat{\boldsymbol{b}}_x$ [mG]	$\hat{\boldsymbol{b}}_y$ [mG]	$\hat{\boldsymbol{b}}_z$ [mG]	$\sigma(h_E)$ [deg]	$\hat{\boldsymbol{b}}_x$ [mG]	$\hat{\boldsymbol{b}}_y$ [mG]	$\hat{\boldsymbol{b}}_z$ [mG]	$\sigma(h_E)$ [deg]
Raw	-0.0	-0.0	-0.0	8.5	-0.0	-0.0	-0.0	8.5
Centered	-35.1	-45.3	-59.7	3.1	-11.5	-61.7	-61.2	4.0
TWOSTEP	-34.8	-43.2	-59.8	3.1	26.0	32.5	-57.6	8.0
TWOSTEP*	-35.1	-45.0	-59.7	3.1	-4.2	-43.4	-60.5	4.3
VMC-LS	**-24.6**	**-33.7**	**-49.7**	**2.5**	**-17.0**	**-33.2**	**-49.5**	**2.8**
AI-EKF	-20.2	62.8	-63.0	6.2	26.6	39.6	-56.7	8.2
VMC-KF	-30.3	-43.1	-43.1	2.7	-11.7	-30.9	-41.1	3.4

Calculated bias of laboratory sensor in Reference [6].

Table 3.4 Estimated Bias of a Phone Sensor

	EXP3L - PHONE				EXP4S - PHONE			
	$\hat{\boldsymbol{b}}_x$	$\hat{\boldsymbol{b}}_y$	$\hat{\boldsymbol{b}}_z$	$\sigma(h_E)$	$\hat{\boldsymbol{b}}_x$	$\hat{\boldsymbol{b}}_y$	$\hat{\boldsymbol{b}}_z$	$\sigma(h_E)$
	[mG]	[mG]	[mG]	[deg]	[mG]	[mG]	[mG]	[deg]
Raw	0.0	0.0	-0.0	100.5	0.0	0.0	-0.0	100.5
Centered	67.0	801.5	-622.0	4.2	79.8	760.8	-584.8	6.0
TWOSTEP	67.7	817.6	-713.4	6.2	12.7	949.5	-762.5	28.4
TWOSTEP*	67.1	802.8	-629.1	4.3	59.9	817.2	-637.9	5.3
VMC-LS	85.7	800.3	-597.8	4.2	86.6	789.4	-603.8	4.5
AI-EKF	228.2	427.9	194.5	206.8	192.3	102.5	24.7	111.8
VMC-KF	**84.6**	**804.3**	**-599.7**	**4.2**	**87.4**	**795.5**	**-592.6**	**4.3**

Calculated bias of phone sensor in Reference [6].

The magnitudes of these estimated biases range from 0 mG to almost 1,000 mG. When measuring the accuracy of the optimization methods, it will be important to compare the methods over a range of biases to see how changing the magnitude of bias affects the accuracy of the optimization differently for each method. This table will be helpful when determining what realistic range of bias should be used. It is also important to note that depending on the local value of the magnetic field $\boldsymbol{H}_k$, when the magnitude of the bias becomes too large, the results of the optimization algorithms will not provide useful information. We must be careful to verify that the range of bias chosen does not exceed that threshold. This will be tested in the Observability Section of Chapter 3.

Verification of Bias Measurement

Baseline examples of the TLS Method and Method 2 function readings are shown in the figures below. Here we run the full sample size of data point vectors *n* at one thousand Monte Carlo runs *M* with a fixed value of the geomagnetic field $\boldsymbol{H}_k$, the reasonably assigned bias $\boldsymbol{b}$, and the noise $\boldsymbol{\sigma}$. The purpose of these readings is to confirm the successful function of these methods under normal conditions before comparing these methods over a wide range of conditions

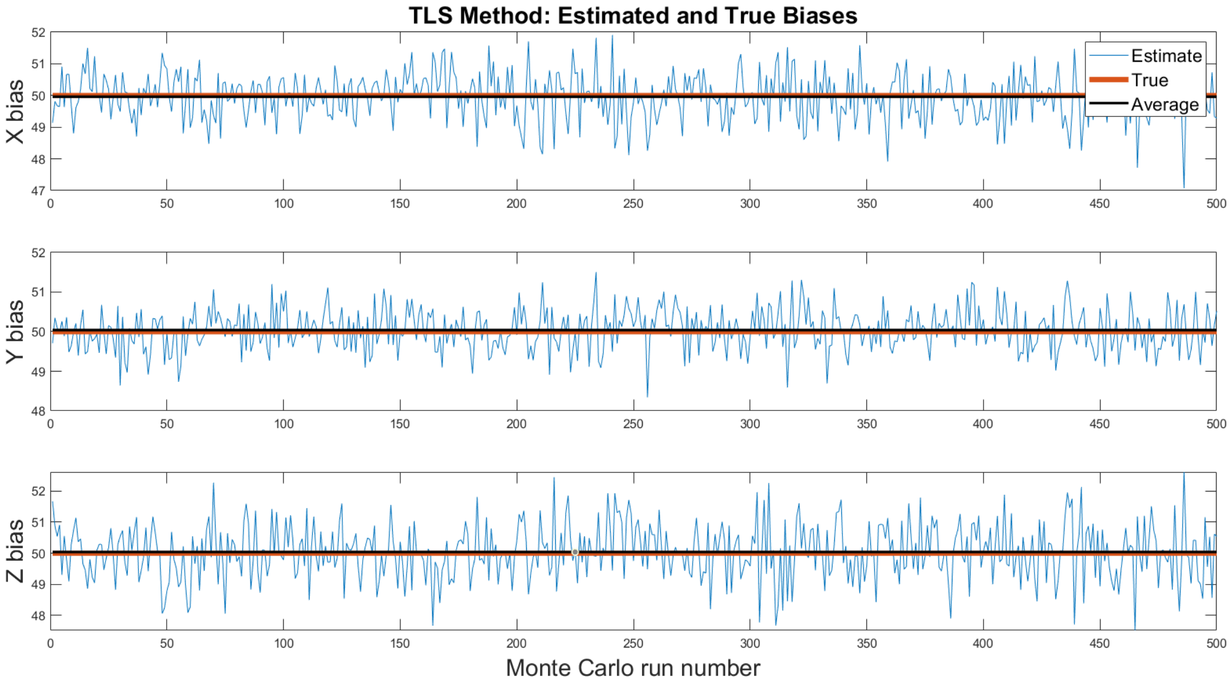

Figure 3.2 Estimated and True Biases from TLS Method Function

Number of vectors (*n*) is 226; Number of Monte Carlo simulations (*M*) is 500; Value of the geomagnetic field ($\boldsymbol{H}_k$) is [228.55;-9.27;430.60] mG; Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned value of noise ($\boldsymbol{\sigma}$) is 5 mG.

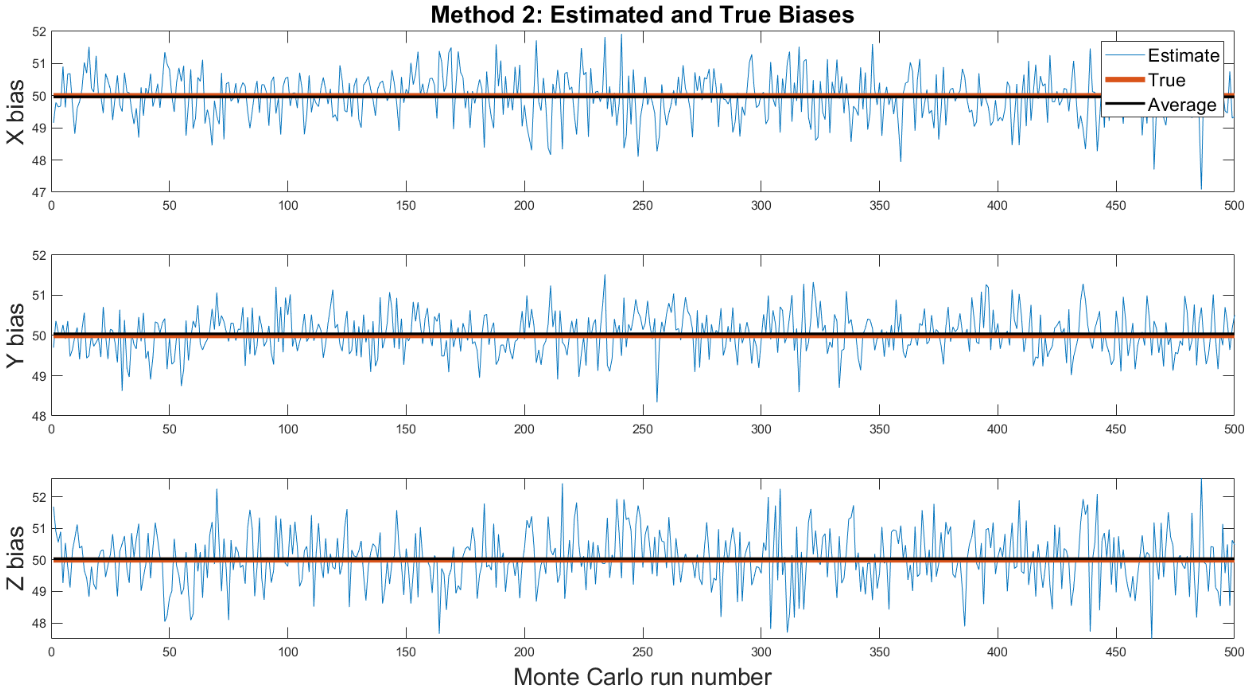

Figure 3.3 Estimated and True Biases from Method 2 Function

Number of vectors (n) is 226; Number of Monte Carlo simulations (M) is 500; Value of the geomagnetic field ($\boldsymbol{H}_k$) is [228.55;-9.27;430.60] mG; Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned value of noise ($\boldsymbol{\sigma}$) is 5 mG.

As shown above, the estimated bias from the TLS Method and the estimated bias from Method 2 both accurately calculate the true bias. The error between the true bias and average estimated bias for the TLS Method is [0.0374;-0.0411;-0.0397] mG. The error between the true bias and average estimated bias for Method 2 is [0.0344;-0.0407;-0.0363] mG. These results are important because they support our assumption that the number of vectors (n) and the number of Monte Carlo simulations (M) are sufficiently high to continue testing. However, more verification tests will be conducted. It should be noted that the estimated bias from Method 1 was found to be equal to the estimated bias from Method 2. The results from Method 2 were chosen to be representative of the results from these two new methods. The results from Method 3 will be discussed in a later section.

Constraint Error Measurements

In Chapter 2 we presented the optimization constraint that would be enforced through Method 1, 2, and 3. In this section we will verify enforcement of that constraint by comparing the constraint errors. The expected value of the constraint error for the TLS calibration method should be small, but it should be noticeably larger than the constraint error found within the other calibration methods.

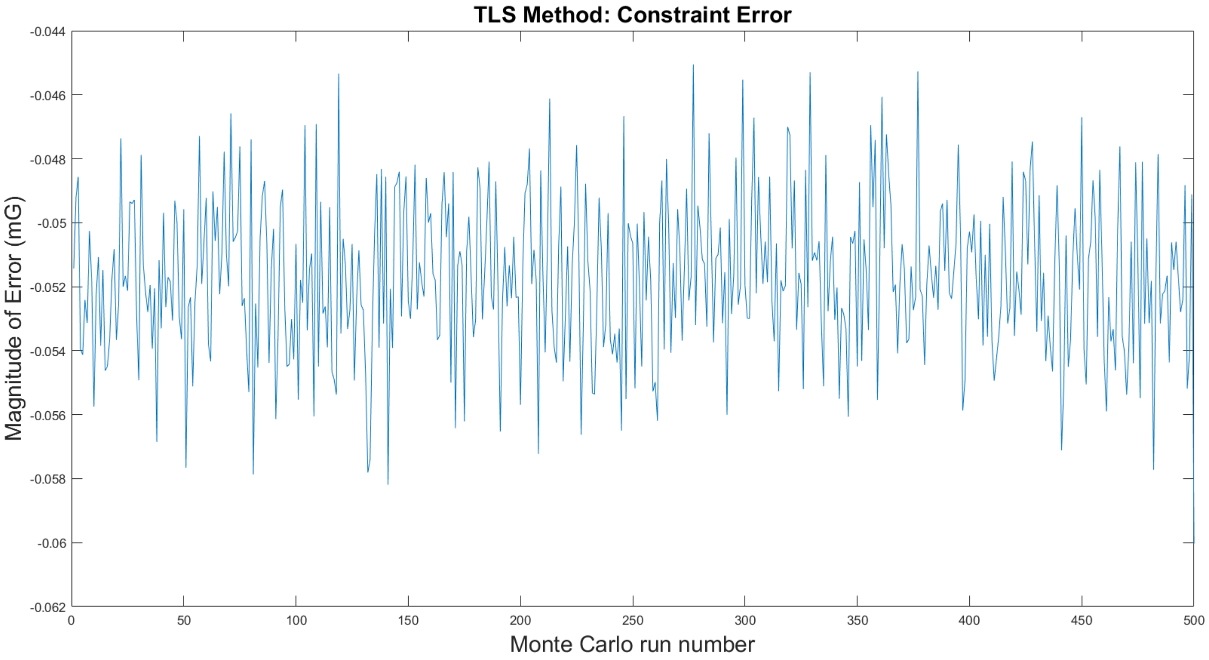

Figure 3.4 Constraint Error from TLS Method

Number of vectors (n) is 226; Number of Monte Carlo simulations (M) is 500; Value of the geomagnetic field ($\boldsymbol{H}_k$) is [228.55;-9.27;430.60] mG; Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned value of noise ($\boldsymbol{\sigma}$) is 5 mG.

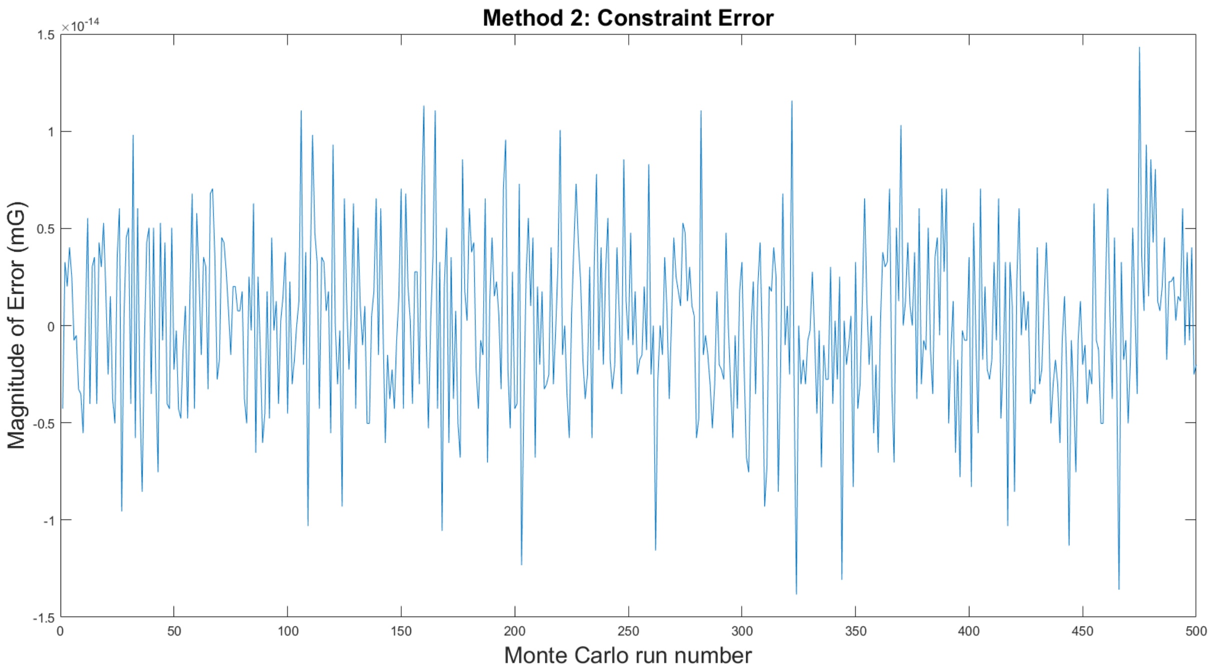

Figure 3.5 Constraint Error from Method 2

Number of vectors (n) is 226; Number of Monte Carlo simulations (M) is 500; Value of the geomagnetic field ($\boldsymbol{H}_k$) is [228.55;-9.27;430.60] mG; Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned value of noise ($\boldsymbol{\sigma}$) is 5 mG.

As calculated, the constraint error from the TLS Method calibration is significantly higher than the constraint error from the Method 2 calibration. The order of magnitude of the TLS Method constraint error is 10^{-2}. The order of magnitude of Method 2 constraint error is 10^{-14}. This result was expected. It should be noted that the constraint errors from Method 1 was found to be equal to the constraint error from Method 2. The results from Method 2 were chosen to be representative of the results from these two new methods.

Comparison of Algorithm Accuracy

The variables that we will be changing through each test are as follows. The TLS Method and the new methods are compared over a range of magnitudes of biases and noise. We will observe whether or not there are trends of higher, lower, or constant accuracy as noise increases and as bias increases for each of these methods. Additionally, we must verify that the bias estimation at the worst cases is still accurate. However, before presenting these comparisons, it is important to define the standard that will be used to determine which method is superior.

SSE Function

This heading explains the formula used to compare algorithms. The formula we are using is call the Sum of Squared Errors (SSE). At each axis, we will measure the difference between the points of estimated bias and the true bias. The resulting error can be a combination of positive and negative values. In order to create the consistency of absolute values at every point, we will square the values of error. Finally, we add the squared values together into a single sum at each point along the range of noise or bias. This is the SSE formula. The calibration method with the lower SSE error is the more accurate method.

SSE over Range of Noise

We will start by presenting the SSE results over a range of noise as shown in the figure below. Again, it should be noted that the resulting accuracy of the three new methods was found to be equivalent. The most relevant differences between the three new methods will be highlighted when comparing the speed of calibration. In this case, we will use the results from Method 2 to again represent the accuracy of all three new methods.

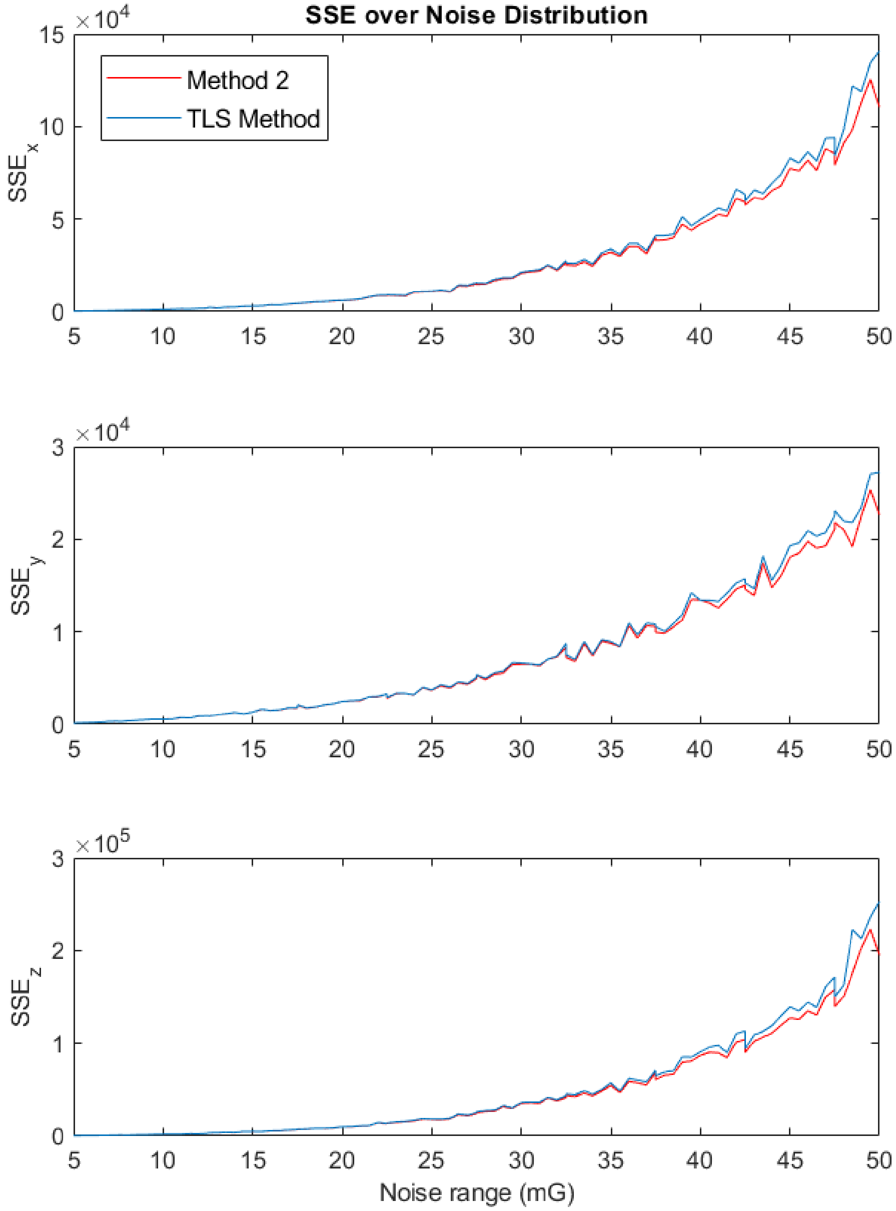

Figure 3.6 Comparison of Accuracy for Methods over Range of Noise

Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned range of noise (**σ**) is 5 mG to 50 mG.

As shown above, the accuracy of the TLS Method and Method 2 both decrease as the magnitude of the noise increase. At lower values of noise, the difference between the TLS Method SSE and the Method 2 SSE are very small. However, as the magnitude of the noise grows, the TLS Method SSE grows higher than the Method 2 SSE. This would indicate that Method 2 becomes more accurate than the TLS Method as the magnitude of noise increases. In order to verify that conclusion, we present the difference in SSE between the TLS Method and Method 2 in the figure below.

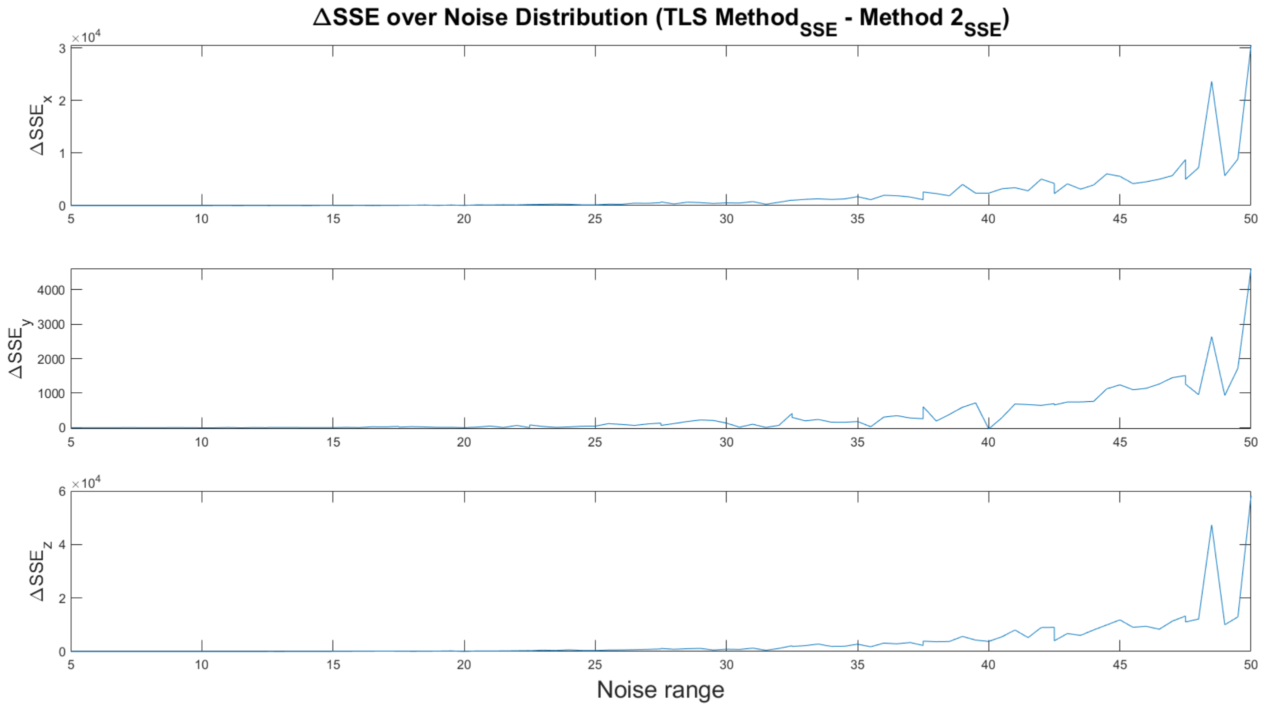

Figure 3.7 Difference between TLS Method SSE and Method 2 SSE

Calculated from data in Figure 3.6.

Once again, at lower levels of noise, it is difficult to distinguish between the SSE value of the TLS Method and the SSE value of Method 2. To this end, we present a sample of the same plot at the z-axis with the SSE label formatted to track logarithmic growth. Here we see the difference in SSE more clearly at all points along the range of noise.

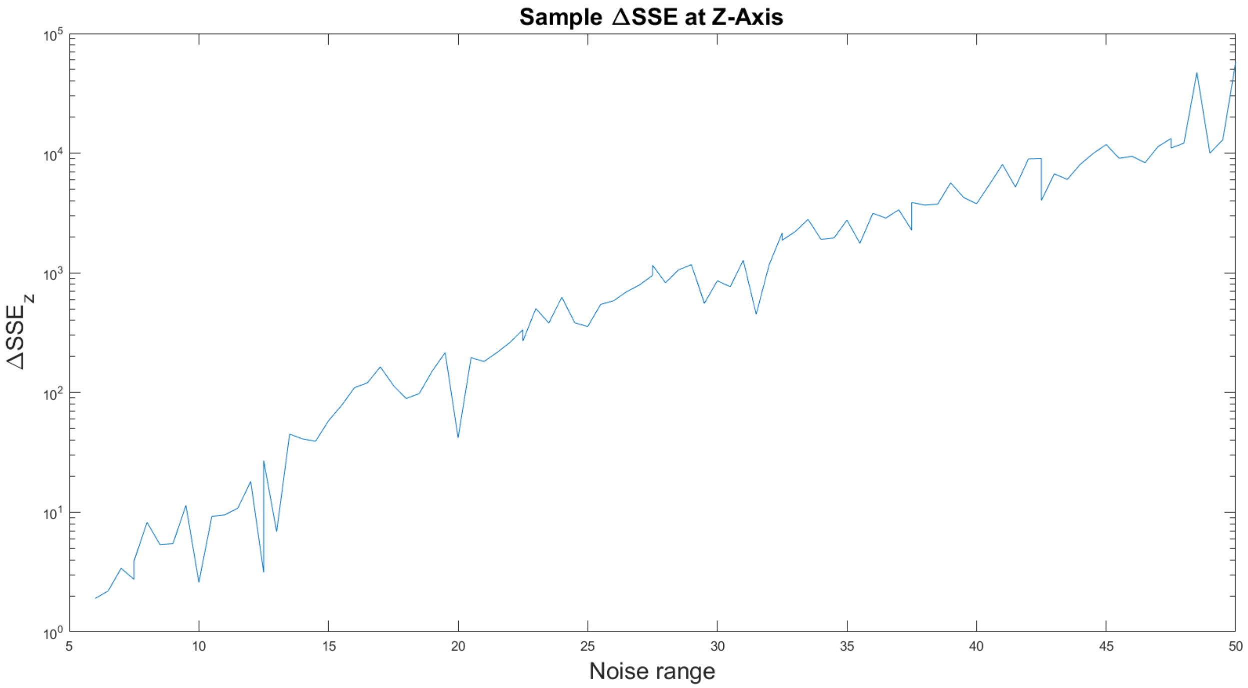

Figure 3.8 Sample of the Difference in SSE at Z-Axis

Calculated from data in Figure 3.7

Before definitively stating that Method 2 is more accurate than the TLS Method under this set of conditions, there is one more step that must be taken. We must verify that the resulting bias estimation is reasonably accurate at the highest value of noise within the range.

Verification of Bias Estimation

We have shown that Method 2 is more accurate than the TLS Method at higher levels of noise. However, this conclusion is irrelevant if neither method can produce a reasonably accurate estimation of the bias at all points along the range of noise. Here, we test the bias estimation of Method 2 when the magnitude of noise is 50 mG. The result of this test is presented in the figure below.

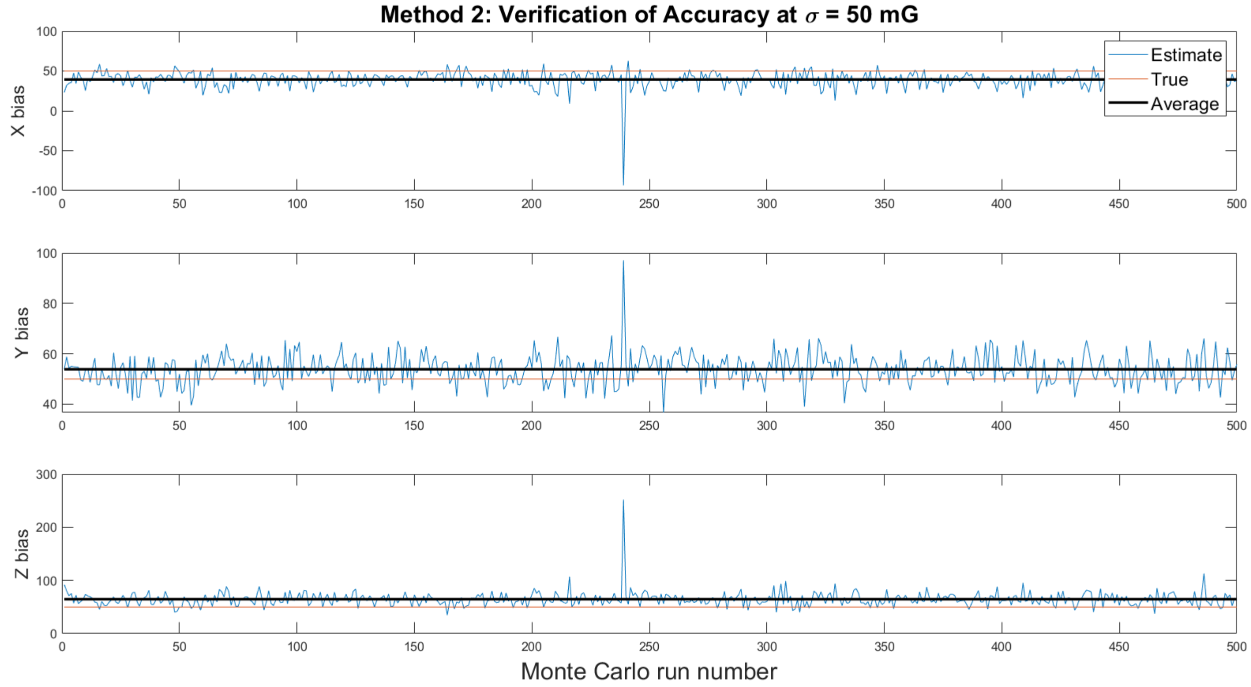

Figure 3.9 Verification of Method 2 Bias Estimation at Highest Magnitude of Noise

Number of vectors (n) is 226; Number of Monte Carlo simulations (M) is 500; Value of the geomagnetic field ($\boldsymbol{H}_k$) is [228.55;-9.27;430.60] mG; Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned value of noise ($\boldsymbol{\sigma}$) is 50 mG

As shown above, the estimated bias at this magnitude of noise does not perfectly reflect the true bias. However, over the course of Monte Carlo simulations, the bias estimations fall on both sides of the true bias. This ensures that the average bias estimation is still relatively accurate. Compared to the average bias estimation from the TLS Method at this magnitude of noise we calculated the following results. At the x-axis Method 2 was 7.2% more accurate. At the y-axis, Method 2 was 6.0% more accurate. At the z-axis. Method 2 was 6.7% more accurate. At this point, we contend that under the set conditions, Method 2 is more accurate than the TLS Method.

SSE over Range of Bias

We will now present the SSE results over a range of bias as shown in the Figure 3.10 below. Here we come across an unanticipated result. While the change in SSE was gradual for all methods over the range of noise, the change in SSE is much more abrupt at specific points over the range of bias. This outcome requires more analysis. To that end, we present the difference in SSE between the TLS Method and Method 2 in the Figure 3.11 below.

Once again, at most of the points with lower levels of noise, the SSE value of the TLS Method and the SSE value of Method 2 are virtually identical, while at specific points, either the TLS Method, Method 2, or both methods show a sharp increase in SSE error. The expected result was that the estimated bias of all methods would be very close to each other with a gradual increase in SSE as bias increases. We instead see large disparity between estimations of different methods and abrupt rather than gradual increases in SSE. Because of this, it would be inadvisable for use to make a judgement of accuracy between the methods based on the data found.

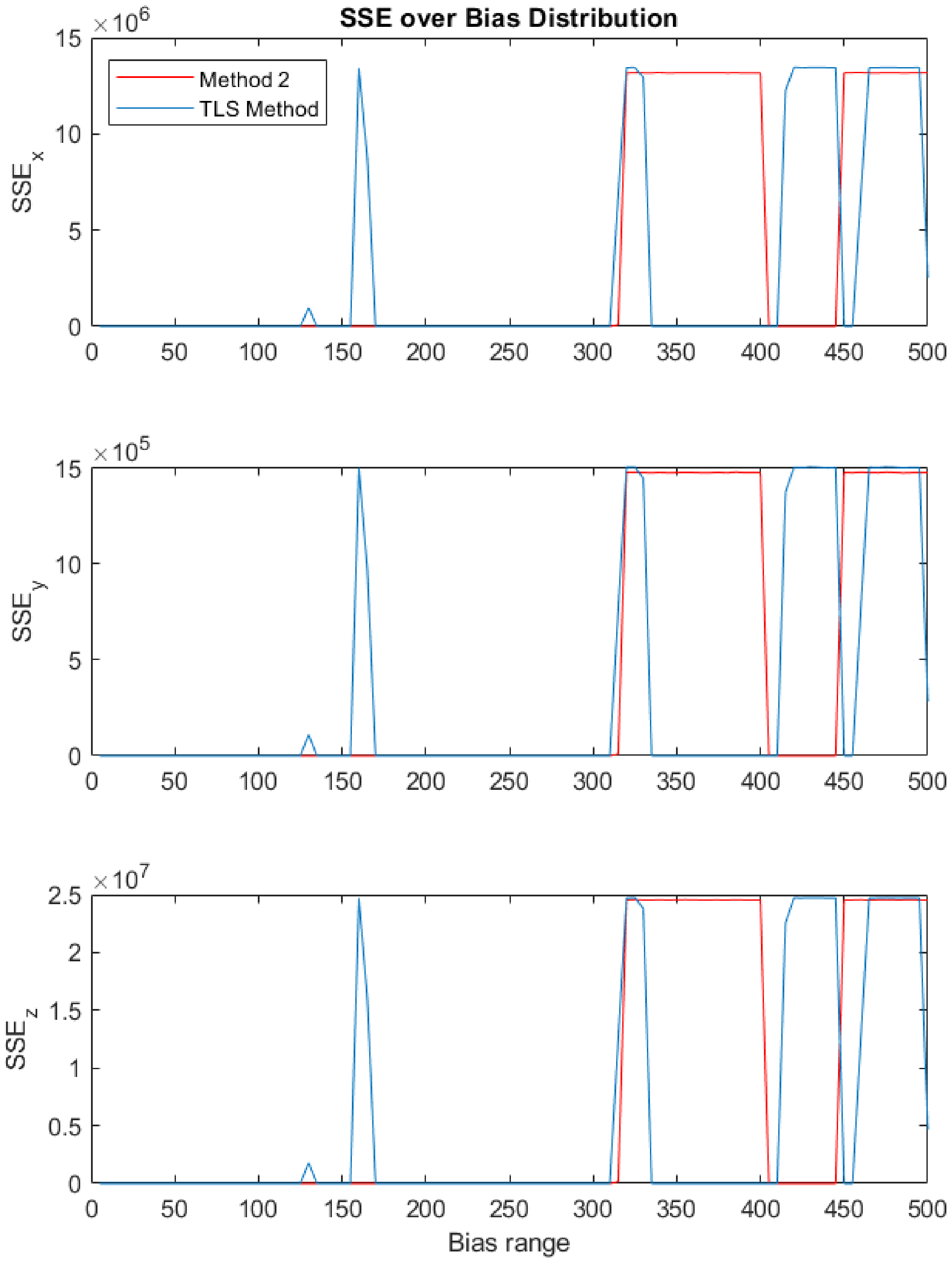

Figure 3.10 Comparison of Accuracy for Methods over Range of Bias

Assigned range of bias ($\boldsymbol{b}_{x,y,z}$) is 5 mG to 50 mG; Assigned magnitude of noise ($\boldsymbol{\sigma}$) is 5 mG

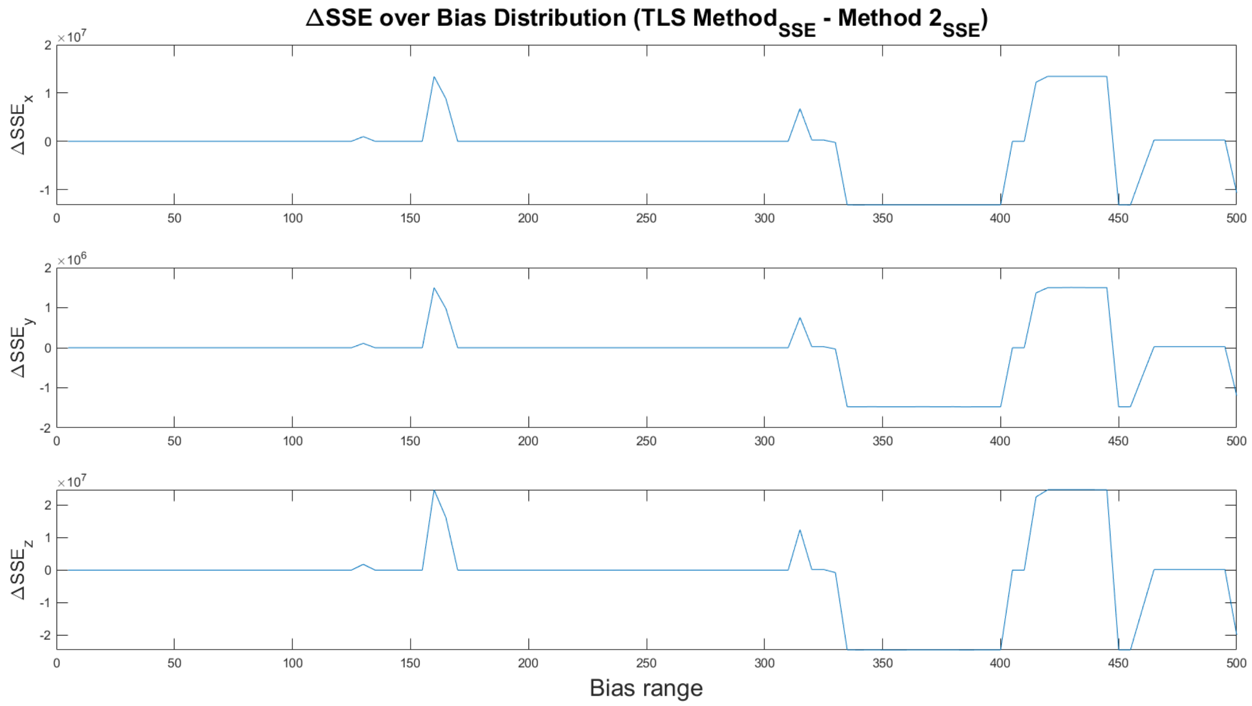

Figure 3.11 Difference between TLS Method SSE and Method 2 SSE

Calculated from data in Figure 3.10

One could argue that SSE spikes at lower values of bias only occur within the TLS Method, making Method 2 more reliable within the range of 5 mG to 314 mG of bias. One could also argue that the average SSE value over the total range of bias is higher within Method 2, making the TLS Method more reliable on average. We hold that these claims are premature based on the irregular data. These spikes in SSE error could more likely be attributed to complications within MATLAB's *fmincon* optimization function. Where the SSE spikes, our research shows that the bad estimates have a lower cost function. This could indicate an error within the dataset. At this point, the comparison of accuracy between the methods over the range of bias is inconclusive. These large, non-uniform spikes in SSE should be investigated further.

Accuracy when $\boldsymbol{H_k}$ is not Constant

There is one final case in which we compare the accuracy of the calibration methods. There are many situations where it is not reasonable to assume that the value of the local magnetic field $\boldsymbol{H_k}$ is constant over time. For example, the magnetometer within a satellite does not remain in a specific locality for an extended period of time, and the magnetic field surrounding satellites is constantly changing. In the case of the TLS Method, Method 1, and Method 2, these calibration methods are not equipped to estimate the bias under the condition of an $\boldsymbol{H_k}$ that is not constant. However, Method 3 has been specifically designed to estimate the bias under this condition while preserving the accuracy of Method 1 and Method 2. For this reason, a comparison of the calibration methods under this condition is not competitive.

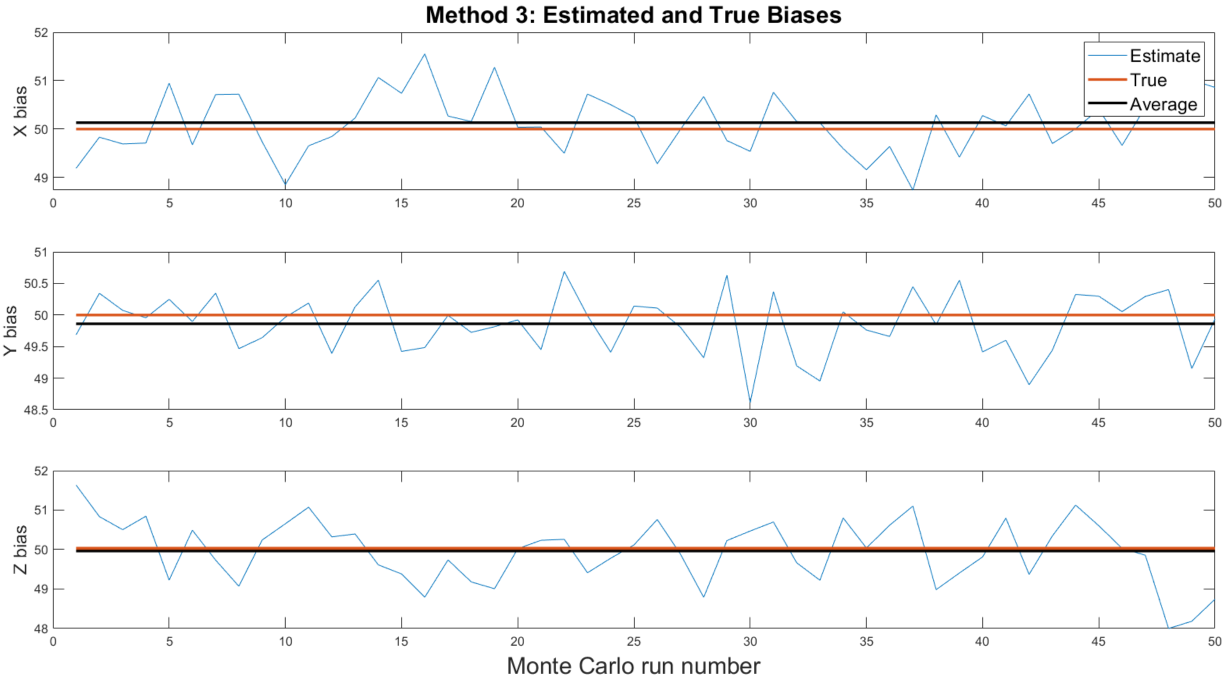

Figure 3.12 Estimated and True Biases from Method 3 Function

Number of vectors (*n*) is 226; Number of Monte Carlo simulations (*M*) is 50; Value of the geomagnetic field ($\boldsymbol{H}_k$) is [228.55;-9.27;430.60] mG; Assigned values of bias ($\boldsymbol{b}_{x,y,z}$) [50;50;50] mG; Assigned value of noise ($\boldsymbol{\sigma}$) is 5 mG.

Method 3 is not the most accurate method when the local value of the magnetic field is consistent, but it is the most accurate method here because it is simply the only method that is capable of estimating the bias under this condition.

Comparison of Algorithm Speed

The final comparison within this thesis is a comparison of computation speed Between the TLS Method, and our newly developed Method 1, Method 2, and Method 3. Here it should be noted that Method 2 was specifically designed to improve upon the computation speed of Method 1 while retaining the accuracy of Method 1. In this regard, the development of Method 2 was completely successful. Over the course of testing, simulations of Method 1 could take hours to resolve while simulations of Method 2 under the same condition would be completed within 2 minutes. Additionally, simulations of Method 3 were recorded as taking even longer than Method 2. This should not be surprising as the goal of Method 3 is to meet the requirement of estimating bias under the condition of a changing value of $\boldsymbol{H}_k$. Method 3 sacrificed efficiency in order to meet this requirement. It should be noted that the possibility remains that Method 3 could be implemented more effectively than the manner conducted in our experiment, and we must give allowance for the possibility of differing results based on a more effective implementation of Method 3. Based on our tests, the two most important speeds to compare are the speed of the TLS Method and the speed of Method 2. This is what will be presented in the figure below.

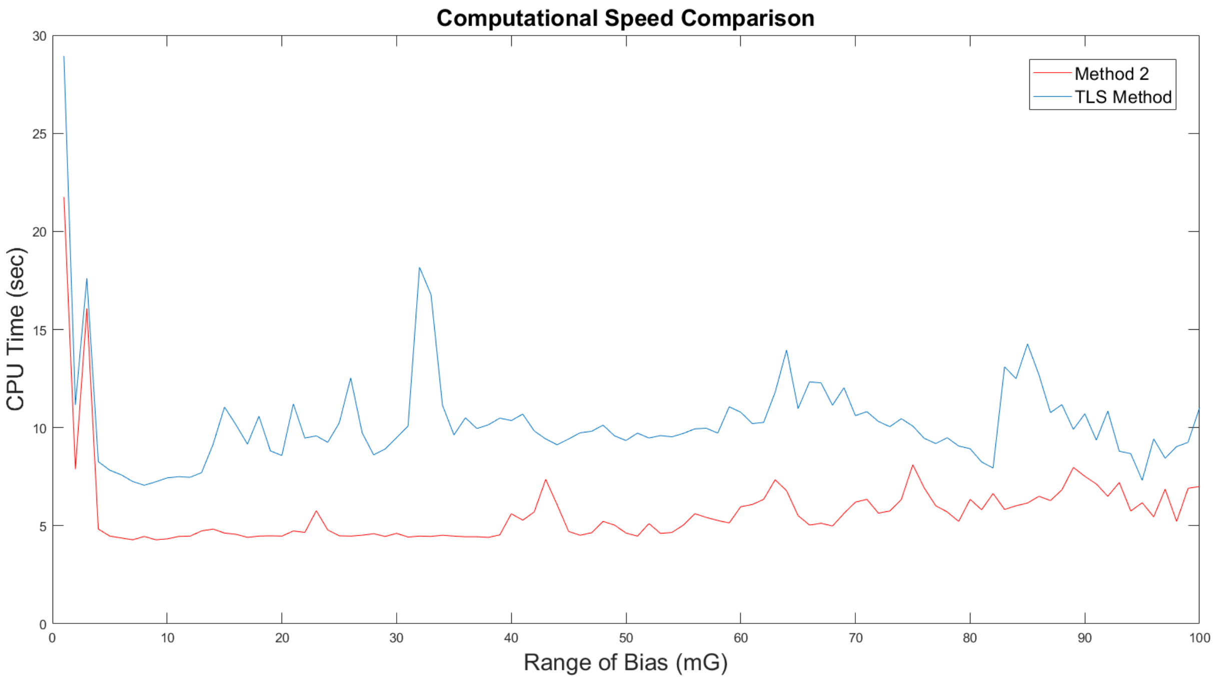

Figure 3.13 Comparison of Computational Speed between Method 2 and the TLS Method

Assigned range of bias ($\boldsymbol{b}_{x,y,z}$) is 5 mG to 50 mG; Assigned magnitude of noise ($\boldsymbol{\sigma}$) is 5 mG

As shown in the experiment above, Method 2 is on average at least 30% faster than the TLS Method. It should be noted that the MATLAB unit of measurement *cputime* is not perfectly consistent depending on the environment of the computer calculating the speed. The TLS Method and Method 2 appeared to have competitive computational expenses, and the possibility remains that the TLS Method could be implemented more effectively than the manner conducted in our experiment. There are also external factors built into the testing computer itself that could change the speed of any of the methods on any given day. However, with some external factors removed, and repetitive testing conducted, Method 2 still appears to consistently outperform the TLS method in terms of speed. Based on this data, we assert that the new Calibration Method 2 is likely more efficient and therefor computationally faster than the TLS Calibration Method.

Conclusion

A simulation of a zero-bias, noiseless magnetometer reading was developed through the use of gyroscopic data and the local value of the geomagnetic field with respect to Earth. This simulation allowed us to test different calibration methods and their ability to estimate a chosen "true" bias. Based on the literature review, the Total Least Square (TLS) Method showed improvement when compared to other methods of calibration. Over the course of this thesis, we developed a method with the goal of improving the accuracy of the TLS Method by defining a constraint within the optimization algorithm that reflects a norm preserving relationship. We named this development Method 1.

Next, we developed a method with the goal of improving the computational speed of Method 1 while maintaining the accuracy based on the norm-preserving constraint of Method 1. We accomplish this by only building one cost function before applying it to the optimization function, rather than forcing the optimization function to build the cost function at every iteration. We named this development Method 2.

Next, we developed a method with a unique goal. Rather than attempting to improve the general accuracy or computational speed of the other methods, we attempted to solve a specific minimization problem where the exact value of the geomagnetic field with respect to the Earth-based coordinate system is inconsistent. We accomplished this by taking a quartic cost function and composing a variation in which the cost function is quadratic with respect to bias. We named this development Method 3.

We then compared the accuracy of the TLS Method and the new methods represented by Method 2. We based this comparison on the variability of noise and the variability of sensor bias. In the case of variable noise, Method 2 showed improved accuracy in comparison to the

TLS Method as the magnitude of the noise increased. At the highest level of noise, Method 2 provided at least 6% improvement in accuracy when compared to the TLS Method.

In the case of the variable bias, both the TLS Method and Method 2 produced inconsistent results. Over a wide range of bias, both methods performed with equally consistent levels of accuracy. However, at specific points of bias, one or both of the methods produced very high levels of SSE error. This magnitude of error caused any resulting bias estimations to become entirely useless. The TLS Method showed high levels of SSE error at an earlier point in the bias range than Method 2. Method 2 produces higher average levels of SSE error over the bias range than the TLS Method. These are reasons to consider either method to be more accurate. However, because the results are so inconsistent, more research should be completed before a definitive statement is made regarding the accuracy of these methods over a bias range.

In the specific case where the value of the local magnetic field $\boldsymbol{H}_k$ with respect to Earth is not constant over time, there is only one method that produces accurate results. That method is Method 3. Here, there is no comparison between Method 3 and the other methods, because the other methods cannot function properly without either the $\boldsymbol{H}_k$ value or some other modification.

Finally, we compared the computational speed of the TLS Method to Method 2. Method 2 performed over 30% faster than the TLS Method over a range of tests. Based on this result and Method 2's higher accuracy over a range of noise, we assert that a substantial goal of our thesis, to create a calibration method that improves the TLS Method, has been achieved. As far as Method 1 and 3 are concerned, their accuracy is equivalent to Method 2, and their computational speeds are significantly lower. Method 2 is superior to the TLS Method and Method 1. Unless dealing with the specific case of a non-constant value of $\boldsymbol{H}_k$, Method 2 is also superior to Method 3.

A Note Regarding Complexity

Based on the results of previous sections, it is important reiterate an important point. There are a number of variables that contribute to the accuracy of the calibration methods. Our thesis tests changes in the magnitude of the noise and changes in the magnitude of the bias. There are opportunities outside of our thesis for further testing of other variables. These include researching the exit flags of the *fmincon* function to find an explanation for the SSE error over the range of bias, changing to an optimization function besides *fmincon*, increasing the number of Monte Carlo simulations to improve accuracy, and increasing the number of vectors (n) to improve accuracy. Due to this complexity, there are a number of paths that could be taken to build upon the research completed in our thesis.

REFERENCES

[1] J. L. Crassidis and Y. Cheng, "Three-Axis Magnetometer Calibration".

[2] B. Grandvallet, A. Zemouche, M. Boutayeb and S. Changey, "Real-Time Attitude-Independent Three-Axis Magnetometer Calibration for Spinning Projectiles: A Sliding Window Approach," *IEEE Transactions on Control Systems Technology,* 2014.

[3] L. Wang, S. Xiong, Z. Zhou, Q. Wei and J. Lan, "Constrained Filtering Method for MAV Attitude Determination," *IMTC 2005 - Instrumentation and Measurement,* 2005.

[4] L. Zhiping and X. Jinqiang, "New Calibration and Error Compensation for Strapdown Magnetometer," *Proceedings of the 34th Chinese Control Conference,* 2015.

[5] P. Tortora, Y. Oshman and F. Santoni, "Attitude Independent Estimation of Spacecraft Angular Rate Using Geomagnetic Field Observations," *Universita di Bologna; Israel Institute of Technology; Universita di Roma,* Vol. 6-2637.

[6] G. Troni and R. M. Eustice, "Magnetometer Bias Calibration Based on Relative Angular Position: Theory and Experimental Comparative Evaluation," *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 2014.

[7] Y. Wu and W. Shi, "On Calibration of Three-Axis Magnetometer," *IEEE Sensors Journal,* 2015.

[8] W. Zeng, Q. Bian, J. Gao, L. Chang and Y. Tong, "Attitude-Independent Magnetometer Calibration Based on Adaptive Filtering," *IEEE Sensors Journal,* 2022.

[9] J. Hashmall and J. Deutschmann, "An Evaluation of Attitude-Independent Magnetometer-Bias Determination Methods," *National Aeronautics and Space Administration / Goddard Space Flight Center,* Vols. Contract NAS 5-31500.

[10] L. Huang and W. Jing, "Attitude-Independent Geomagnetic Navigation Using Onboard Complete Three-Axis Magnetometer Calibration," *Harbin Institute of Technology.*

[11] J. L. Crassidis and Y. Cheng, "Three-Axis Magnetometer Calibration Using Total Least Squares," *AIAA Scitech 2020 Forum,* 2020.

[12] F. Corrigan, "Drone Gyro Stabilization, IMU And Flight Controllers Explained," 7 May 2020. [Online]. [Accessed 20 April 2022].

[13] J. Diebel, "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors," *Stanford University,* 2006.

[14] J. Sola, "Quaternion kinematics for the error-state Kalman filter," 2017.

[15] F. L. Markley and J. L. Crassidis, Fundamentals of Spacecraft Attitude Determination and Control, Microcosm Press and Springer: Space Technology Library, 2014.

[16] K. Groÿekatthöfer and Z. Yoon, "Introduction into quaternions for spacecraft attitude representation," *Technical University of Berlin,* 31 May 2012.

[17] Matlab, "Constrained Nonlinear Optimization Algorithms," The MathWorks, Inc., [Online]. Available: https://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html. [Accessed 19 May 2022].

[18] Digi-Key Electronics, "STMicroelectronics L3G4200D," [Online]. Available: https://www.digikey.com/en/products/detail/stmicroelectronics/L3G4200D/2352584. [Accessed 20 April 2022].

[19] RWTH Aachen University, "Gyroscope (Raw data)," [Online]. Available: https://phyphox.org/experiment/gyroscope/. [Accessed 20 April 2022].

[20] S. Lichlyter, "Gyro1019," Google, 18 June 2022. [Online]. Available: https://docs.google.com/spreadsheets/d/1kQd-K4Hhe5HPKV_K4R0TRHizHRMTjBc4/edit?usp=sharing&ouid=108312829471708942640&rtpof=true&sd=true.

[21] The MathWorks, Inc., "Matlab," [Online]. Available: https://www.mathworks.com/help/matlab/. [Accessed 20 April 2022].

[22] National Oceanic and Atmospheric Administration, "Magnetic Field Calculators," [Online]. Available: https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml#igrfwmm. [Accessed 11 May 2022].

[23] National Oceanic And Atmospheric Administration, "Geomagnetic Frequently Asked Questions," [Online]. Available: https://www.ngdc.noaa.gov/geomag/faqgeom.shtml#What_is_the_difference_between_IGRF_and_WMM_models. [Accessed 17 May 2022].

[24] DataQualityApps e. K., "Conversion Calculator," [Online]. Available: https://www.convert-measurement-units.com/conversion-calculator.php?type=magnetische-flussdichte. [Accessed 7 June 2022].

[25] Asahi Kasei Microdevices, "AK8963 3-axis Electronic Compass," October 2013. [Online]. Available: https://download.mikroe.com/documents/datasheets/ak8963c-datasheet.pdf.

[26] V. N. Krishnachandran, "Where Do the Terms of the Power Series Expansions of Sine and Cosine Functions Come from? Involutes!," *Vidya Academy of Science & Technology,* 2016.

[27] S. Lichlyter, "Thesis Appendix Matlab Code," Google, 20 June 2022. [Online]. Available: https://docs.google.com/document/d/1mbtCco7HVMb1Yx-ZadADCT4FuyK6lTDH/edit?usp=sharing&ouid=108312829471708942640&rtpof=true&sd=true.