Characterization Tests of IMUs for Small Satellite Implementation

A project present to
The Faculty of the Department of Aerospace Engineering
San Jose State University

in partial fulfillment of the requirements for the degree
Master of Science in Aerospace Engineering

By

Keith Young

December 2015

approved by

Dr. Marcus Murbach
Faculty Advisor

CHARACTERIZATION TESTS OF IMUs FOR
SMALL SATELLITE IMPLEMENTATION

by

Keith H. Young

APPROVED FOR THE DEPARTMENT OF AEROSPACE ENGINEERING

SAN JOSÉ STATE UNIVERSITY

December 2015

Marcus Murbach — Date

Marcus Murbach
Department of Aerospace Engineering

The Designated Project Committee Approves the Project Titled

Abstract

It is important to understand the fundamentals and devices behind dynamics and controls before venturing into the calculations and data analysis. This paper starts with an introduction and discussion into spacecraft dynamics, covering various that are involved. The second chapter goes into a more in-depth look into how these devices work and how the setup of such devices is vitally important. The next chapter starts getting into the inertial measurement unit discussion and how exactly they are used in a satellite's control system. A brief overview of the two IMUs are discussed here as well, the Epson and SBG Ellipse IMU. The following chapters delve into the actual testing of each device over various testing procedures including a vibration table test involving a sine sweep and random vibration. A comparison is made between the two devices to determine the viability on-board the International Space Station for use in an autonomous robot as well as their ability to work on a small satellite. Various analysis techniques are used including a look at the Allan Variance plots as well as a power spectral density (PSD) plot of each device. These processes allow a look into the properties of the IMUs and give insight into how they behave under real-life conditions. The Allan Variance plot gives deeper meaning into the intrinsic noise of the device while a PSD plot helps to understand at what frequencies cause the device problems. A comparison is also conducted between the devices' company-provided datasheet and the values obtained through testing. It is also worth noting that the Appendix provides all of the behind-the-scenes looks at how the data analysis was accomplished with use of MATLAB. An easy to use all-in-one function is described to provide all of the necessary tables and graphs. This will lead to improved evaluation techniques that will result in better IMU selection for applications, such as precision pointing, for next gen nano sats.

Contents

Chapter 1: Literature Review

1.1 Introduction of Spacecraft Dynamics

Spacecraft dynamics and control plays a huge role in spaceflight. One could argue that the beginning of spacecraft control research started in 1957 with the successful launch of Sputnik. Back then the only goal at the time was to put an object in orbit with no intention of actually controlling it. Since then, the main research laboratories have focused their attention to how a satellite behaves in space. This chapter will discuss the history and implementation of satellite guidance, navigation and control (GNC).

A spacecraft has many factors that must be taken into account when trying to control it. The main variables under investigation are its position, velocity and attitude. The attitude of a spacecraft is its orientation in space. To further explore these factors, we can separate the former two into orbital mechanics which deal with its motion relative to a larger body of mass. For Earth-based satellites it is usually easiest to relate the position and velocity to Earth. Attitude is concerned with motion relative to itself only and is measured using sensors placed on the satellite. These sensors include magnetometers and sun sensors

Attitude can also be explored more thoroughly by looking specifically at determination, prediction and control. Attitude determination involved the aforementioned sensors to rely data back to Earth regarding the satellite's orientation. Depending on the specific mission and cost of the satellite, multiple sensors can be used to gather more precise readings. Attitude prediction is the ability to simulate how a spacecraft will behave in the future. This process evolves as technology constantly advances because of the computational power required. Simulation is a

huge part of spacecraft dynamics and will be discussed later in this paper. Attitude control is the aspect of using devices on the satellite to control its movement through space. By using thrusters and gyroscopes, among other devices, a satellite can be oriented in any direction. Another use of control is to change orbits around whatever object it is orbiting. This is usually not needed for Earth-based satellites but nonetheless it still must be mentioned.

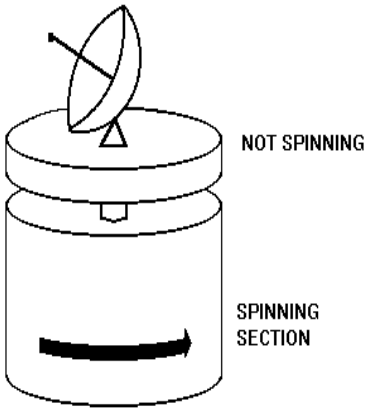


Figure 1: Basic setup of a spin-stabilized satellite.

Controlling a satellite can also be broken into two parts, spin-stabilized and three-axis stabilized. The former, shown in Figure 1, involves using a satellite's angular momentum to spin around its major axis in order to maintain its attitude[1]. The latter is a more complex system usually involving gyroscopes to align the satellite in any direction desired. Both setups have their advantages and disadvantages while cost and weight are factors as well. With the popularity of university-funded nano satellites, the goal is usually to devise a mission that is both inexpensive and lightweight. With that in mind, engineers and scientists must consider that every device they want to put on a satellite adds weight and money.

1.2 Historical Review of Air-Bearing Spacecraft Simulators

As mentioned previously, research into spacecraft dynamics and control started with the start of the space race. With the ability to simply put an object into orbit realized, future missions required a certain goal. Whatever these missions entailed, the ability to easily control a satellite was a must. A new type of laboratory test had to be implemented in order to successfully test future missions. Air bearing testbeds were the new type of Earth-based experiments. Simply put, air bearings virtually take away friction between two objects by releasing pressurized air. The ability to effectively control a satellite in a frictionless environment opened up the ability to conduct experiments that very much resembled their space-like counterparts.

The most common type of laboratory setup is using a spherical air bearing. The satellite's foundation would be a hollow hemisphere that fits on top of the sphere. This will allow freedom to experiment with rotational attitude. This setup can be seen in Figure 2[2]. Of course, more complicated setups can be built to allow translational movement as well.

Figure 2: This setup shows the hollowed-out section with the hemisphere foundation sitting on top.

The advantage of these new setups is that both the government and university level laboratories can use them. Spherical air bearings can easily test a nano satellite as well as a commercially built larger one that weighs as much as a few thousand pounds. A few of the universities

currently pursuing this research include, Stanford, University of Victory and MIT. The experiments conducted here have a variety of goals, from robotic arms to free-flying spherical robots. NASA Ames Research Center also has a testbed for experimenting with nano satellites. This is a field that has been around for for a while but is constantly growing as technology advances.

1.3 CubeSats

CubeSats are a relatively new type of satellite that allows small organizations to launch satellites into space. They were first introduced as a joint project between Cal Poly and Stanford University in 1999. As of now, large satellites are only available to national projects are extremely wealthy private companies. The current price per pound of placing an object in orbit is upwards of $30,000[3]. CubeSats are classified as nano satellites because of the small $10cm^3$ shape and weight limit of roughly 1.33kg. Figure 3 shows the size of a CubeSat[4] relative to a person's hand. Even though the size is the main limiting factor, this is extremely welcoming for universities as it allows students to conduct actual spaceflight experiments. In most cases, a handful of CubeSats are placed as extra

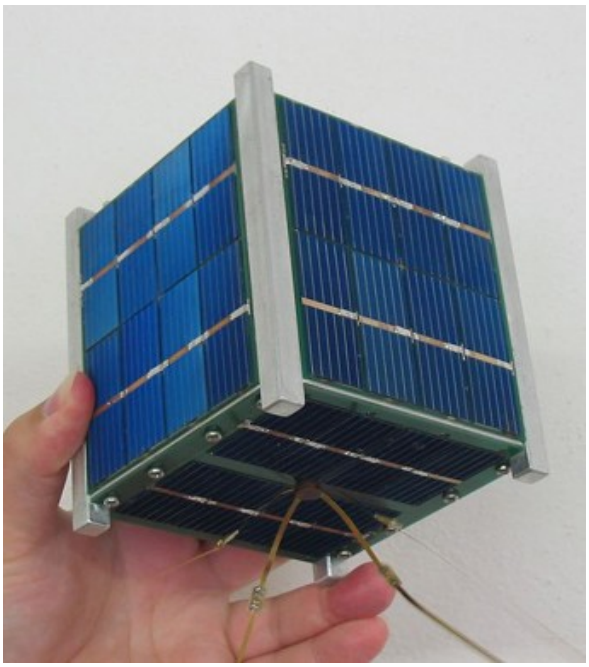


Figure 3: A CubeSat can easily fit in the palm of your hand.

payloads for an otherwise regular rocket launch. The technology used in CubeSats is rapidly evolving as smaller and lighter components are being manufactured. Even though the size is the main limiting factor, this is extremely welcoming for universities as it allows students to conduct actual spaceflight experiments. In most cases, a handful of CubeSats are placed as extra payloads for an otherwise regular rocket launch. The technology used in CubeSats is rapidly evolving as smaller and lighter components are being manufactured.

1.4 Magnetometers

One of the core sensors of a satellite are magnetometers. These devices are able to measure Earth's magnetic field, which varies from point to point. Since each point in space has a different measurement, the data that is collected can be converted into coordinates. This allows engineers to know the exact location and attitude of a satellite. An experiment conducted by Richard

Linhart[5] implemented two magnetometers in order to read values along all three axes (x, y and z). They developed an attitude stabilization algorithm that the satellite applies after it reads in the magnetic field data. The magnetometer is also able to utilize the magnetic field to properly orientate itself as needed. This type of propulsion is called electrodynamic tether.

The tether method only works correctly when turned on at the appropriate times during a satellite's orbit. In general, the magnetic field in the z-direction is greatest at the poles and weakest along the equator. This experiment was successful and they have plans to further their development of more efficient magnetometer stabilized satellites

Another experiment that dealt with magnetometers was put forth by Hee Jung and Mark Psiaki[6]. The main goal for their research is to develop a completely autonomous satellite using magnetometers and sun sensors. Their reason for pursuing this concept is to eventually lower costs of certain missions. This method will most likely have less precision than what is currently in play, however, missions that do not require such precision should be interested. This experiment builds on previous research in the area. Just and Psiaki wanted to test if a filter that is used with a magnetometer in order to cancel out some of the noise would be beneficial. If this works then it would allow for much greater measurement readings. They examined three previous satellite missions and used the actual flight data to improve on an already designed filter. In their conclusion they state that they have successfully implemented the filter.

1.5 Reaction Wheels

Reaction wheels are another commonly used device to control the attitude of a satellite. They use the conservation of angular momentum to spin the satellite in one direction. Taking advantage of three wheels allows for the aforementioned three-axis spin-stabilized spacecraft. The wheels are controlled by an electric motor that spins a rotor. This reaction creates a torque that can be controlled to vary how fast the satellite spins. Reaction wheels have been in use for some time, however, they are relatively new for use with CubeSats. A study done recently by Oluwatosin et al.[7] investigated attitude control of a CubeSat based on a PD controller. They successfully showed that the use of three or four wheels provide a means for fast and accurate attitude control. This was all done by simulation so some extra analysis is required before officially implementing it on a satellite. A reaction wheel setup can be seen in Figure 4[8].

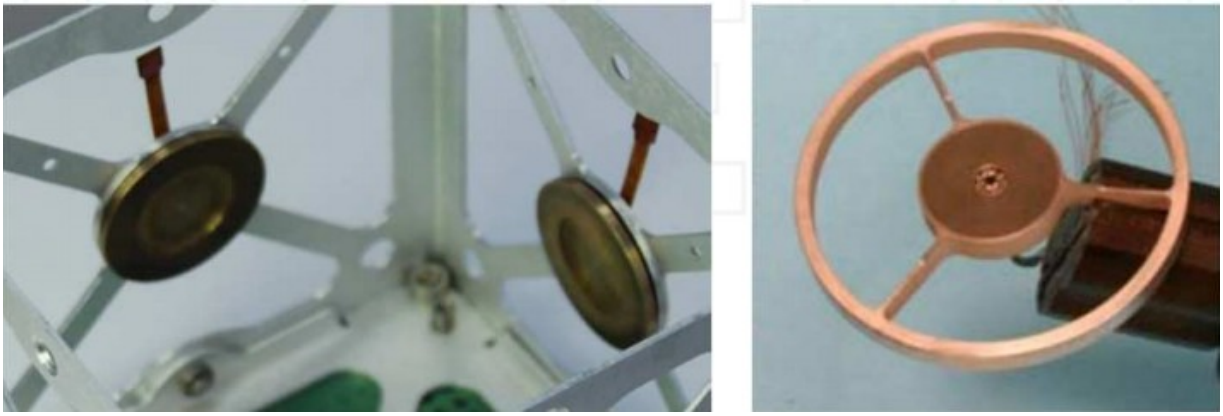


Figure 4: Two types of reaction wheels can be see here, inside (left) and outside (right) of a satellite.

1.6 Computer Processing

A large part of any engineering endeavor is the ability to understand the data. It is not too difficult for anyone to gather data. The hardest part is getting the data to make sense. With the increase in popularity of CubeSats, more and more open-source software is being developed. This allows anyone to freely use previously uploaded programs and alter them in any shape and form. The possibilities are endless when it comes to data processing and manipulation. CubeSats are of huge interest too because of their smaller size and thus, less room for processing power. A study conducted by Kaslow et al.[9] looked into a model-based simulation of a CubeSat mission.

Their model is comprised of three separate programs: MagicDraw, MatLab and STK. A program called ModelCenter was necessary to properly integrate these together. The main goal of this experiment was to show that a CubeSat is capable of processing large data sets with reliable results. They carried out various mission simulations to demonstrate how well their system could handle the data. They succeed in showcasing it in simulations and are ready to progress with their design. They also are making all of their models available to the academic community to allow students to learn and build from their design.

Any satellite requires a central controller to handle all of it's subsystems. With the limited size of CubeSats, such devices have been named microcontrollers. The challenge of developing such a controller is having enough room for everything that is required. This includes, but not limited to, LEDs, USB ports, serial ports, Ethernet and various input/output ports. The core of this board

will also require a processor. In recent years, the Arduino and BeagleBone boards have become extremely popular. The two main advantages to these boards are that they are inexpensive (~$50) and open-source. Anyone can search online for plenty of free programs to help them navigate their board.

These boards offer a great starting point for both electronic enthusiasts and university labs. The possibilities are also endless too. Everything about the board is programmable. With the right knowledge and tools you can use these in model airplanes, underwater submarines or even send one in orbit aboard a CubeSat.

A team at the University of Arizona led by Andy Eatchel[10] looked into developing their own microcontroller for future CubeSat missions. They mention that the main obstacles to overcome are the radiation effects of low-Earth orbit and providing enough system requirements for a mission. The following list shows the minimum requirements for a CubeSate microcontroller: microprocessor, FRAM, real time clock, AD converter, modem voltage and current reset. This list includes just enough to power on and maintain enough power to calculate basic data calculations. This is a good place to start when trying to understand just what goes into a CubeSat. The microcontroller is such a small but necessary component.

1.7 Thrusters

One of the main forms of active control of a satellite are thrusters. Simply put, these devices release an exhaust in order to move the satellite. This could involve small bursts to adjust the attitude or longer bursts to alter the spacecraft's orbit. Satellites in low-Earth orbit experience a

very small amount of atmospheric drag. Over time this drag lowers the orbit of the satellite until the inevitable crash back into Earth. Thrusters also provide a means to counter this drag and extend the life of the satellite. There are a variety of thrusters from ones that releas a gas to electric propulsion. Thrusters are widely used because their versatility but may require extra cargo to run. This obstacle may not be too big a problem for large satellites, but CubeSats offer the complexity of being very small.

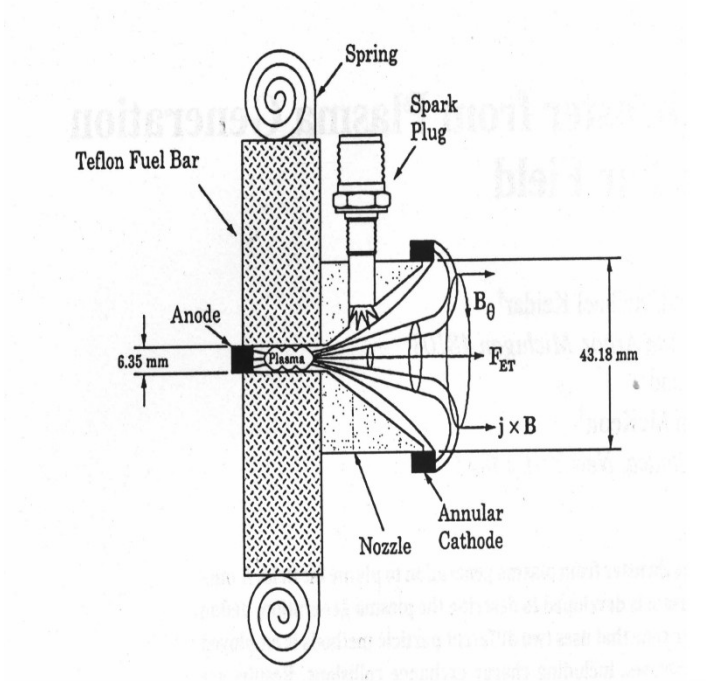


Figure 5: Example setup of a PPT propulsion system. The physical dimensions are of importance when regarding the efficiency.

Electric propulsion has been used since the 1960s but the technology is still being researched. Solar panels provide the means for a constant renewal of energy and could extend the life of certain satellites if it is applied to propulsion. The United States Air Force has put in a lot of research into electric propulsion. A paper published on pulsed-plasma thrusters (PPT) looked

into creating a micro PPT version for use on CubeSats[11]. PPT involves running an electric arc over a fuel source (usually Teflon) that produces a plasma, as seen in Figure 5[12]. The use of charged plates in this setup force the plasma out of the nozzle which propels the spacecraft forward. Their newly developed system provides a mean for precisely controlling micro-satellites.

The plasma emitted by these systems are extremely low in mass but are expelled at enormous speeds. A satellite created by NASA[13] in 2000 used a PPT system as an attitude control device. The actual thrust of the PPT is only 860 micro-Newtons but the plasma is traveling at over 13,000 meters per second. This is achieved by consuming only 70 watts of power.

Another team at the Osaka Institute of Technology[14] looked into a PPT system for controlling their 15kg satellite. They tried to redesign already designed systems in order to make a more efficient thruster. This included looking at the physical dimensions of the nozzle as well as the size and position of the ignitor. A team at VelTech College in India looked into creating a more powerful type of PPT to that could provide enough thrust to significantly raise the orbit of a satellite[15]. They claim that they can extend the life of a mission by over 8 months with their setup. Their system provides 1000 volts to produce the plasma and a thrust similar to that of NASA's satellite. The difference here is that the thruster is built to provide this amount of thrust for a much longer period of time.

There are many factors to consider when deciding on what thruster to use for a satellite. The primary limitations will be weight and size. Typical setups that involve burning a liquid fuel work well for larger satellites but may not be practical for smaller ones. Electric propulsion provides a means of excellent thruster precision at a fraction of the weight and size. The main

obstacle to overcome is the efficiency of an electric system. This area of development is of great interest and is constantly being researched.

Chapter 2: Types of Spacecraft Control

2.1 Momentum Devices

Inertial measurement units are essential in understanding the attitude of a satellite. As mentioned briefly in the first chapter, there are a multitude of measurement devices and a variety of methods to control a satellite. These control devices include, but are not limited to: IMUs, Sun sensors, Earth sensors, star trackers and magnetometers. The actual method of control can be any combination of thrusters, spin stabilization, magnetorquers or momentum wheels.
Momentum based control focusing on manipulating the angular momentum by storing and transferring this energy in order to control the satellite in specific ways[16]. To define this process better, Sidi[17] says "a symmetrical rotating body produces angular torque when accelerated about its axis of rotation." Momentum and reaction wheels both operate in the same way but have slightly different purposes. These wheels consist of a rotor attached to an electric motor that can be controlled remotely.

The difference is how the two wheels are operated. Momentum wheels are set to a high rotational speed in order to maintain stability across that axis, often times requiring three wheels for full control as shown in Figure 6. Reaction wheels are configured to change speed in order to create a torque around whichever axis, thus rotating the spacecraft[18]. Other devices, such as control moment gyros (CMGs) are a combination of both wheels that operate on a gimbal in order to allow for more freedom of control.

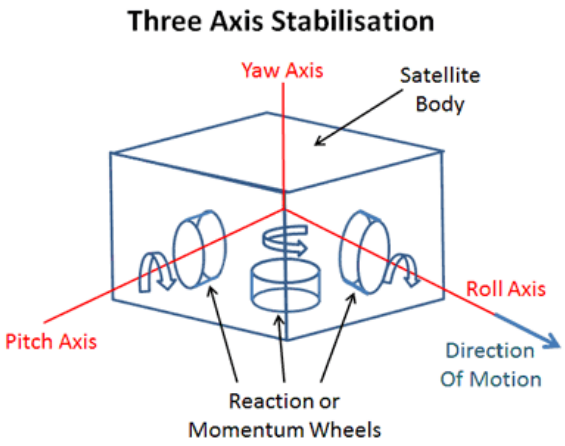


Figure 6: Illustration showing basic 3-axis stabiliation using momentum wheels[19].

2.2 Reaction Wheel Saturation

Reaction wheels are able to store and transfer angular momentum purely based on their rate of rotational speed. By changing this rate, a torque is generated which will alter the attitude of the satellite. If an outside torque is applied such as a gravity gradient or atmospheric drag, then the sensor will read this change and apply the necessary rate change for the reaction wheels. This rate change, whether it is a deceleration or acceleration causes the torque, which is clearly seen in the following formula, where torque $\vec{\tau}$ is equal to the change of angular momentum $\vec{H}$ ¿ *time*

$$\vec{\tau} = \frac{d\vec{H}}{dt} \tag{2.1}$$

However, a wheel can only generate so much torque before it reaches its maximum spin rate. At this point, the wheel can only decelerate which will cause a torque in the opposite direction of what is needed. This state of holding maximum momentum is called saturation, and certain mechanisms are put into place to desaturate the system, as explained below.

Another form of saturation is when a gimbal configuration is in place. The International Space Station uses four CMGs for attitude control, which spin at a constant rate of 6600 rpm[20]. Since the spin rate cannot be controlled, the only way to adjust their torque vector is to change their orientation. Figure 7 shows the null configuration where stability is maintained by having a net angular momentum of zero, since opposing wheels cancel out their torque.

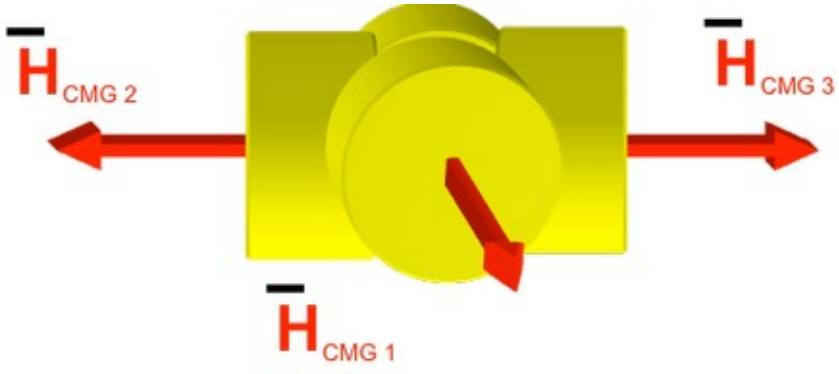


Figure 7: Null position of CMGs aboard the International Space Station.[19]

However, as these wheels are turned to face upwards, a torque is created. In a similar situation to accelerating a wheel to its maximum speed, once all four CMGs are oriented upwards, they reach their saturation point and thus cannot create any more torque, as seen in Figure 8.

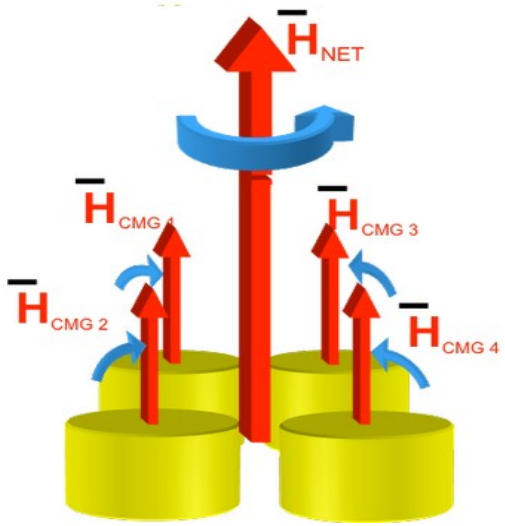


Figure 8: Once all CMGs are facing upwards, saturation is reached as no more torque can be generated.[19]

2.3 Three-Axis Stabilization

As mentioned above, one configuration to achieve full three-axis stabilization is placing a momentum wheel on each axis. Other configuration include using a momentum wheel and thruster system, a single-gimbal momentum wheel or a double-gimbal momentum wheel setup. Utilizing a momentum wheel along the pitch axis with thrusters along all three axes will yield full control over the spacecraft. Figure 9 shows this configuration while Figure 10 illustrated the basic concept of how the devices work together to control the attitude. This configuration takes advantage of roll error signals proposed by Dougherty, et al[21], which are how the control thrusters are operated. When certain attitude changes are detected, the signal initiates the thrusters to provide a torque along the roll and yaw axes to maintain stability.

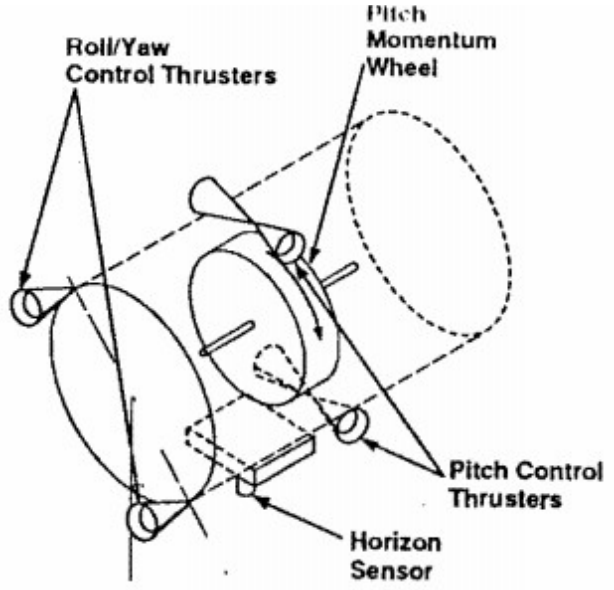


Figure 9: Single momentum-wheel and control thruster configurations can provide full 3-axis stabilization of a satellite.[21]

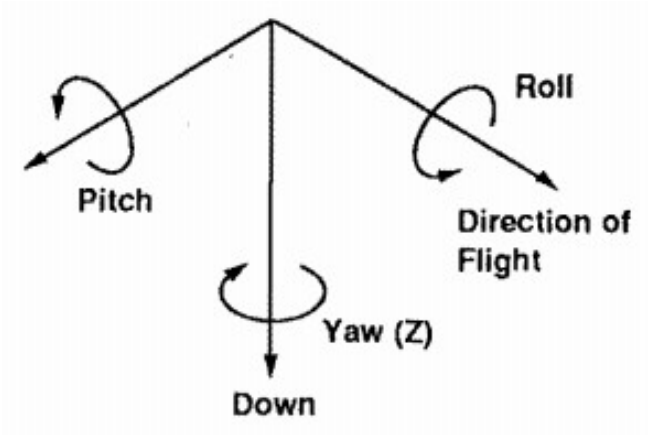


Figure 10: Simple illustration showing how the axes are aligned for the above configuration.[21]

The single-gimbal momentum wheel configuration uses its ability to rotate while spinning in order to maintain full control. A roll sensor is used as well in order to correctly adjust for any movement. The gimbal axis is aligned with the roll axis and its starting position, or null position,

is such that the momentum wheel's spin axis is aligned with the negative pitch axis, which can be seen in Figure 11. From null position, any rotation away from it will introduce a component of angular momentum along the z-axis. With this extra momentum, some form of desaturation mechanism is needed to correct for any imbalance. This mechanism can be a pair of thrusters or an electromagnet torquer as shown in Figure 11.

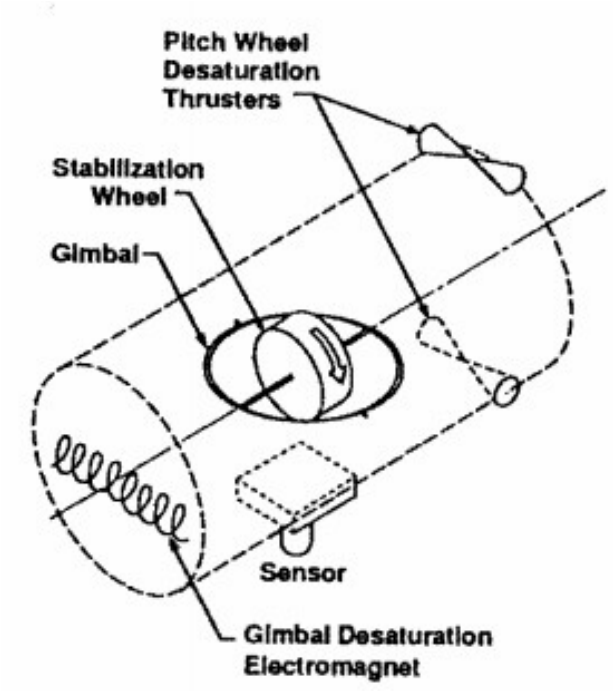


Figure 11: Single-gimbal configuration for a satellite.[21]

A similar configuration is the double-gimbal setup with one wheel as illustrated in Figure 12. The gimbal is setup to move with two-degrees-of-freedom with control thrusters along the roll and yaw axes. The main advantage of a double-gimbal configuration over a single-gimbal one is the decoupling of nutation and orbit rate. In other words, this allows the satellite to roll while maintaining the momentum wheel's orientation in inertial space. This results in a significantly less reaction in the yaw axis.

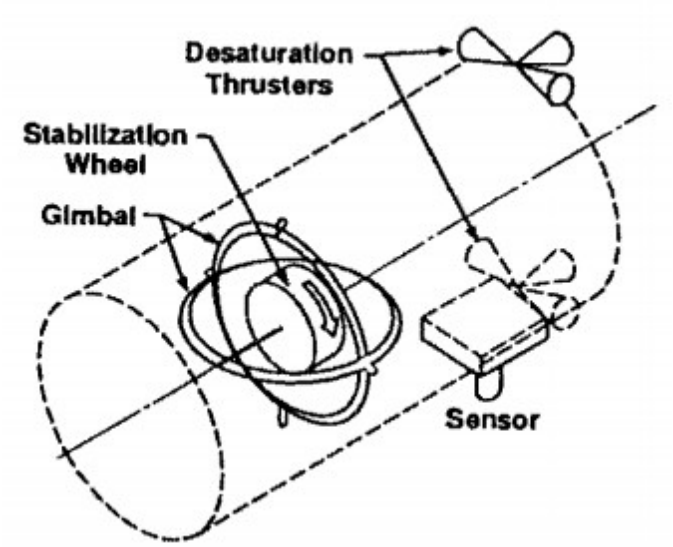


Figure 12: Double-gimbal setup with one momentum wheel.[21]

As mentioned earlier, the most basic idea for full three-axis control is placing a reaction wheel on each axis, as shown in Figure 13. This setup is easily visualized because each wheel provides independent control along its axis. For full stability, the sensor must constantly maintain a balance between each axis, constantly decreasing or increases the rate of the wheels. This transfer of momentum is setup as a closed-loop control system. However, even though the attitude remains balance, the overall angular momentum may slightly accumulate change. This is offset by the aforementioned desaturation techniques.

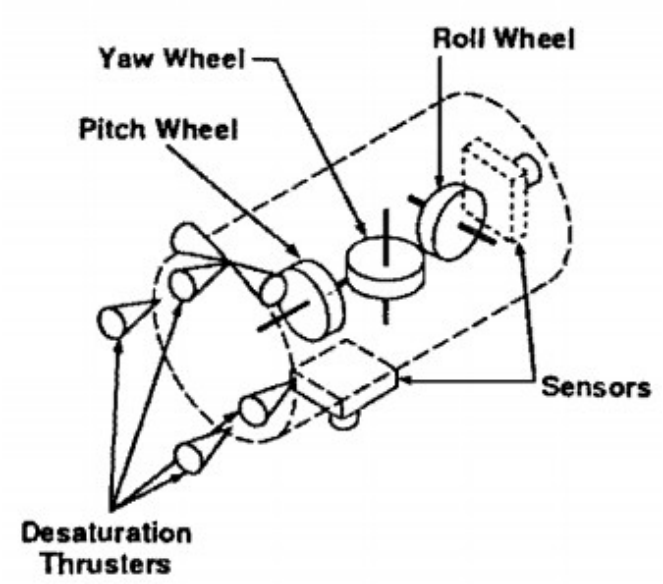


Figure 13: Three-wheel setup to provide full three-axis control and stabilization.[21]

2.4 Propulsion

For now, any propulsion system onboard a CubeSat has only one job, attitude control. A survey done by NASA in 2010[22] looked at the possibility of introducing a large enough propulsion system on a CubeSat in order to accomplish larger tasks, like orbit change or formation flying. Current technologies in this area include butane systems, pulsed plasma thrusters and vacuum arc thrusters but have not been used solely other than attitude control. However, with the huge popularity and demand of CubSats, more involved propulsions systems are under constant development.

The main constraints of adding a larger propulsion system are the mass, volume and power requirements. Ideally, these CubeSats are combinations of 10cm cube objects, which means creating a larger satellite defeats the purpose of the original idea. Before developing the actual propulsion system, it is first necessary to calculate initial delta-v requirements for simple plane

changes and raises in orbit. As a rough estimate, a delta-v of 135m/s is required per degree of orbit change and 0.6 m/s per km of altitude change in low-Earth orbit at an altitude of 250 km[23]. This can be seen below in Table 1:

Table 1: Delta-v Required for Orbit Raising and Plane Changes at Various Altitudes

Earth Orbital Altitude	Plane Change (m/s)/deg	1 km Altitude Change (m/s)
250	135.35	0.58
500	132.86	0.55
750	130.97	0.52

2.5 Chemical Propulsion

One of the most popular types of mono-propellant for conventional spacecraft is hydrazine. Unfortunately, current technology is too robust to fit on a CubeSat so research is underway to miniaturize this hardware. JPL has development a hydrazine milli-Newton thruster[24] for CubeSats but is still not ready for commercial use. Hydrazine is also extremely hazardous to work with due to its toxicity and flammable nature, which makes it not ideal in the small-scale, university development labs.

Hydrogen Peroxide is another monopropellant that has been seriously considered for use on CubeSats. Researchers at the Austrian Research Centres Seibersdoorf (ARCS) are underway developing hydrogen peroxide micro-thrusters for small spacecraft. Similar to the hydrazine thruster, this micro-thruster are still too large to provide efficient attitude control but may be used as main thrusters for orbit maneuvers. Unlike hydrazine, hydrogen peroxide is much safer to work with and can be handled by less experienced university groups.

2.6 Cold Gas Thrusters

This type of system is mechanically very simple and can use non-toxic materials as a source of fuel. Fundamentally, all a cold gas system does is vent high-pressured gas through a nozzle to produce thrust. This is basically the definition of Newton's Third Law and works well as a means of low-impulse propulsion. Cold gas systems are more suitable in terms of making mass and volume restrictions, however low-impulse is not necessarily ideal for quick orbit maneuvers.

2.7 Pulsed Plasma Thrusters

One group of researchers, Scharlemann, et al[25], have looked into creating a miniaturized Pulsed Plasma Thruster (μPPT) along with an advanced Field Emission Electric Propulsion (FEEP). A FEEP system works by using a strong electric field to strip electrons from a metal in order to accelerate these particles to create thrust. Regular sized PPTs have been used previously for their 'structural simplicity and low power requirements,' but have yet to be fully functional as a miniaturized version for CubeSats.

This group has managed to create a prototype system of four μPPTs with a total power requirement of less than 2 W and weight of 250 g, which is just under 20% of a single CubeSat's total weight requirement. This setup can provide an impulse of 7 μNs and offer a total delta-v of about 11m/s. The FEEP, which requires at least a double CubeSat configuration, can provide a thrust range between 0.03 – 0.1mN at a power requirement of 4 W and weighing roughly 415 g (31% of a single CubeSat's weight). This is a promising start into figuring out the most efficient propulsion system for a CubeSat that can provide both attitude control and orbit maneuver abilities.

2.8 Inertia Matrix and Rotational Stability

Stability of a spacecraft from rotation is dependent on the moment of inertia, specifically around its center of mass. When dealing with the angular momentum of a spacecraft, $h=[I]w$, the inertia tensor is defined as:

$$[I]=\begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \tag{2.2}$$

This matrix is derived from looking at the angular momentum of an object. When describing an object moving linearly through space with a rotation about its center of mass, the following equation is used:

$$\frac{d}{dt}A_I=\frac{d}{dt}A_B+\omega\times A \tag{2.3}$$

Which says that when observing from a fixed reference frame ('I' for inertial), the rate of change of a vector **A** is equal to the rate of change of **A** in the rotating body frame plus the vector product $\omega\times A$ [17]. This can be applied to deriving the angular momentum of a rigid body, as seen below.

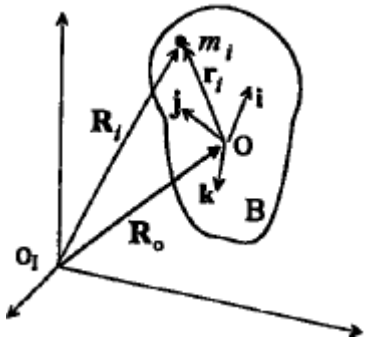


Figure 14: Angular motion of a rigid body[17].

This figure shows an object with center of mass at point *O* with unit vectors **i**, **j**, **k** along the body axis. It can be seen here that any point m_i inside the body can be represented by $R_i = R_0 + r_i$ with the derivative resulting in:

$$\dot{R}_i = \dot{R}_0 + \dot{r}_i + \omega \times r_i = v_0 + v_i + \omega \times r_i \tag{2.4}$$

Using the definition of angular momentum, $h = r \times (m v)$, and the previous equation, the angular momentum of m_i can be shown as:

$$h_i = r_i \times m_i (\dot{R}_0 + \dot{r}_i + \omega \times r_i) \tag{2.5}$$

This can be simplified by letting $\dot{r} = 0$ for a rigid body, and summing over the entire body to obtain:

$$h = \sum_{m_i} -v_0 \times m_i r_i + \sum_{m_i} r_i \times (\omega \times r_i) m_i \tag{2.6}$$

Recall that angular momentum about the center of mass is zero, resulting in the first summation disappearing. Working through the triple product, the result yields $h = [I]\omega$, which is the inertia matrix (eq. 2.1) times angular velocity.

Due to the complexity of product of inertia terms off-diagonal terms, it is ideal to choose a body axis frame that results in a purely diagonal matrix, which sets these other terms to zero. A satellite with zero or negligible products of inertia make it easier to design a control system. The previous equation can be expanded to show the full version of Euler's moment equation seen here

$$M = \dot{h}_b + \omega \times h \tag{2.7}$$

This equation shows the total moment acting on the satellite in the body frame reference. When dealing with the individual axes, the following equations are seen[17]:

$$M_x = I_x \dot{\omega}_x + \omega_y \omega_z (I_z - I_y)$$

$$M_y = I_y \dot{\omega}_y + \omega_x \omega_z (I_x - I_z)$$
$$M_z = I_z \dot{\omega}_z + \omega_x \omega_y (I_y - I_x) \quad (2.8)$$

To determine conditions for stability around any of the principle axes without external moments, the left-side can be set to zero, **M** = 0. For a satellite facing the Earth for communication purposes, it makes sense to find stability about the Z axis, letting $\omega_z = n + \epsilon$, where ϵ is a small disturbance[17]. Taking the derivative here yields $\dot{\omega}_z = \dot{n} + \dot{\epsilon}$ and setting $\epsilon = 0$ changes the equations shown above by substituting in the new value for ω and $\dot{\omega}$. In order to analyze the stability criteria, a second order differentiation followed be a Laplace transform must occur. The first looks like this:

$$\dot{\omega} + \frac{n^2 (I_z - I_y)}{I_y} \frac{(I_Z - I_x)}{I_x} \omega_x = 0 \quad (2.9)$$

The Laplace transform with no initial conditions is simply:

$$\left[s^2 + \frac{n^2 (I_z - I_y)}{I_y} \frac{I_z - I_x}{I_x} \right] \omega_x (s) = 0 \rightarrow s^2 + \beta^2 = 0 \quad (2.10)$$

Solving for β gives $\beta = n \left[\left(1 - \frac{I_z}{I_x} \right) \left(1 - \frac{I_z}{I_y} \right) \right]^{\frac{1}{2}}$ and stability requires that β must be real.

From here it is clearly seen that the criteria is thus $I_z > I_x, I_y$ or $I_z < I_x, I_y$. To put simply, for stability along the Z axis, the body must be spinning along its axis of minimum or maximum moment of inertia. In the case that the condition results in $I_x > I_z > I_y$ or $I_x < I_z < I_y$, then β will be imaginary and the resulting roots of the determinant will be positive and unstable.

Chapter 3: IMU Technology and Performance Grade

3.1 Performance Factors

There are five main performance factors when considering an IMU for a mission. Depending on the required mission specifications, finding the best balance between cost and performance is the goal. The five factors are Angle Random Walk (ARW), Bias Offset Error, Bias Instability, Temperature Sensitivity and Shock and Vibration Sensitivity.

ARW describes the bias in your signal after integrating the output. This is described more in depth in Appendix A. Bias offset error is simply the stationary noise that the device is reading. Ideally, a gyroscope should be displaying zero movement when placed in a non-moving environment, however, some noise will be registered no matter the technology used. Calibrating the output data by applying a passive filter is one way to correct for this. Bias Instability is a crucial characteristic of IMUs and is harder to control than bias offset. Instability can cause drastic sensor readings over a long period of time and is measured by looking at the Allan Variance plot described in Appendix A. Companies usually provide their own Allan Variance plot in their datasheet so it can be replicated in the lab and compared.

Temperature, shock and vibration sensitivity are explained in their name. Under temperature stress and long-term vibration, any device can falter. Assuming the temperature stays ideal throughout a mission, an IMU is more likely to experience harsher vibration conditions. The most common and forceful vibration comes from the rocket launch, where all of the components in the payload must survive the extreme acceleration and shaking. For long-term vibrations,

certain filters, including anti-aliasing and decimation, can be put in place to dampen the sensor output.

3.2 IMU Technology

IMUs are used in a variety of fields from commercial uses to advanced military applications. Depending on their exact applications, different technologies will be considered. As of now, the most commonly used IMUs are fiber optic gyros (FOG), ring laser gyros (RLG) and micro-electro-mechanical systems (MEMS) based IMUs[27]. Each of these technologies also include a wide range of performance grades and costs which allows any customer to find a device that suits their needs.

This type of device can be found in many devices that are used in everyday scenarios, from cell phones, GPS navigation and even video game controllers. The underlying principle of these devices is to measure small variations in mass displacement on an extremely small scale. This displacement is read and outputted as a voltage. Accelerometers use Hooke's Law to measure this displacement while gyroscopes take advantage of the Coriolis Effect.

In simple terms, as showing in Figure 15, two plates are positioned on either side of the mass. Once an acceleration occurs, the capacitance between the mass and plates are measured and from this, the displacement is calculated, seen below

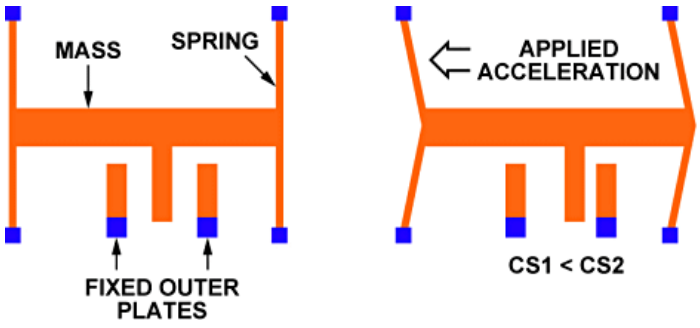


Figure 15: Illustration showing mass displacement in an accelerometer.[26]

$$C_1 = \frac{A}{d-x} \; C_2 = \frac{A}{d+x} \tag{3.1}$$

$$\Delta C \propto A\frac{x}{d} \tag{3.2}$$

As the mass moves a distance x during an acceleration, equation 3.2 comes into play. The value of x is calculated the following way:

$$F = kx \rightarrow F = ma \tag{3.3}$$

$$\omega_o = \sqrt{\frac{k}{m}} \quad \text{, where} \quad x = \frac{a}{\omega_o^2} \tag{3.4}$$

By using Hooke's Law and Newton's First, eq. 3.3, we can then calculate the distance x by measuring the resonant frequency ω_o in eq. 3.4

The change in capacitance is proportional to the area of the plate and the distance to the mass. This simple illustration shows just a single case, however, in real MEMs devices, a multitude of plates are used for better accuracy. This can be seen in Figure 16 and Figure 17 below.

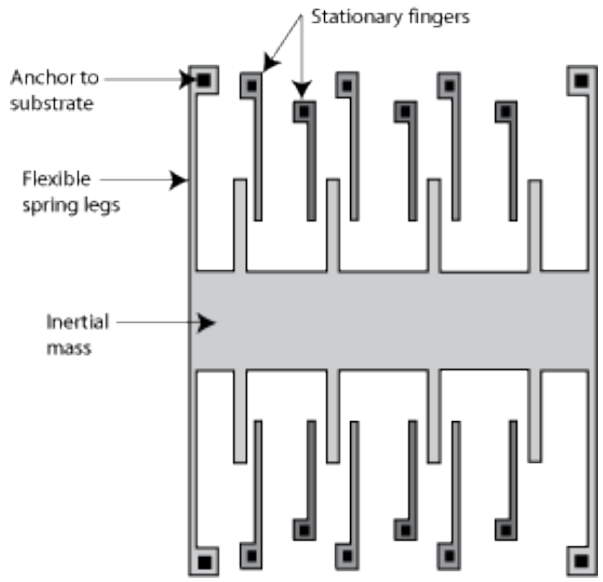


Figure 16: Illustration showing multiple masses aligned with parallel plates. More plates provide a higher degree of accuracy.[26]

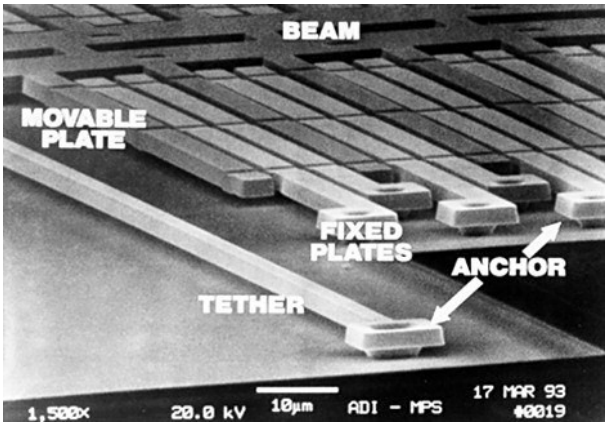


Figure 17: Zoomed-in picture of an actual accelerometer showing the plates and masses.[26]

Gyroscopes use the Coriolis force, eq. 3.5, in order to accurately measure the angular velocity.

$$F_c = -2m(\Omega x v) \tag{3.5}$$

This states that the force is perpendicular to the velocity of the mass v with respect to the rotating reference frame as shown by Ω .

One example, shown in Figure 18, shows that when an angular rate, Ω , is introduced to vibrating masses along the x-axis, a resultant force is seen along the y-axis. Figure 19 shows a

rotating wheel example. This works by spinning a disc along a certain axis, and when an angular rate is introduced along either of the remaining axes, the disc will experience a displacement which is read as a capacitance change.

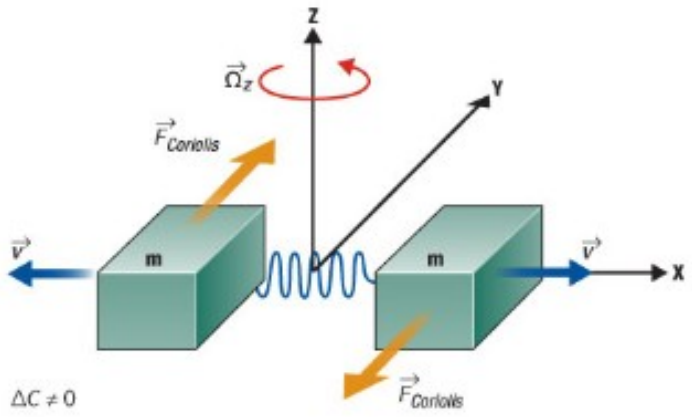


Figure 18: Illustration of the Coriolis effect in a MEMS gyroscope. Two masses vibrate in opposite direction and when an angular rate is introduced, a resulting force is seen perpendicular to the vibration.[26]

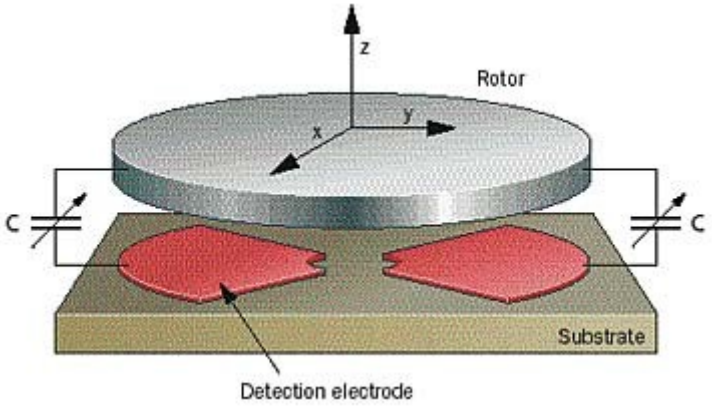


Figure 19: Rotating gyroscope configuration. Any external, off-axis force will result in a capacitance change detected by the electrode.[26]

3.3 IMU Cost and Performance Criteria

In any lab, from garage tinkering to a rendezvous mission with the ISS, a device's cost and performance must be considered. The various characteristics to consider are the device's range

of measurement, the interface and the number axes measured. There may be more specifications to look into but these are the standard characteristics to consider when conducting a trade study.

When considering the range of a potential IMU, it is important to understand exactly the mission's goals. For small-scale missions, a small range can be used (±2g for accelerometers, or ±50deg/s for gyroscopes) whereas a much higher precision may be required (±250g or ±500deg/s) for high-profile missions[27]. These values show the highest force or angular velocity that can be measured, which means your average smartphone user would not require a device range of ±250g. However, the catch to this is that using a higher range device is not always ideal for small-scale missions. For best sensitivity, the device's range should never be too much greater than what is expected from the mission.

The interface options for a device are important when considered how the data is sent and received between the device and a controller. The two main categories are analog and digital, and several subcategories within digital. Digital controllers are used when dealing with a larger system as it can be directly coded into the entire setup. These are serial devices that usually involve UART or I²C connection but may also require more complicated coding. Deciding on an interface may also be limited to the controller used, like a BeagleBone Black or an Arduino.

The number of axes measured is also quite important. Most IMUs will usually come with accelerometers and gyroscopes that measure on all three axes, making it 6 degrees-of-freedom (DOF). With the addition of magnetometers and temperature or pressure, an IMU can be capable of 10 DOF. Each accelerometer or gyroscope will measure their respective units in all three axes, however, in order to fully describe an object's position or velocity, an IMU is required on

each axis. Since each axis is coupled, knowing how the devices react in any situation is vital to how the output data is interpreted.

Pricing depends on many factors, like DOF, technology, weight and size as well as precision. For laboratory testing purposes, prices can range from $15, like the Ivensense MPU-6000, up to $26,000 like the Advanced Navigation's spatial fog IMU. However, when expensive missions are considered and require the most accurate readings, pricing can be much higher.

3.4 Epson and SBG Ellipse IMUs

When buying an off-the-shelf product, like the IMUs used for this project, it is necessary to read and understand the user manual. There are many settings that are dealt with that require attention in order to fully utilize the device. A good starting point is looking at the device's functional block diagram. These diagrams give a brief overview of what to expect in terms of internal filtering. Figure 20 shows block diagram of the Epson gyroscope and accelerometer, while Figure 21 shows the same for the SBG Ellipse.

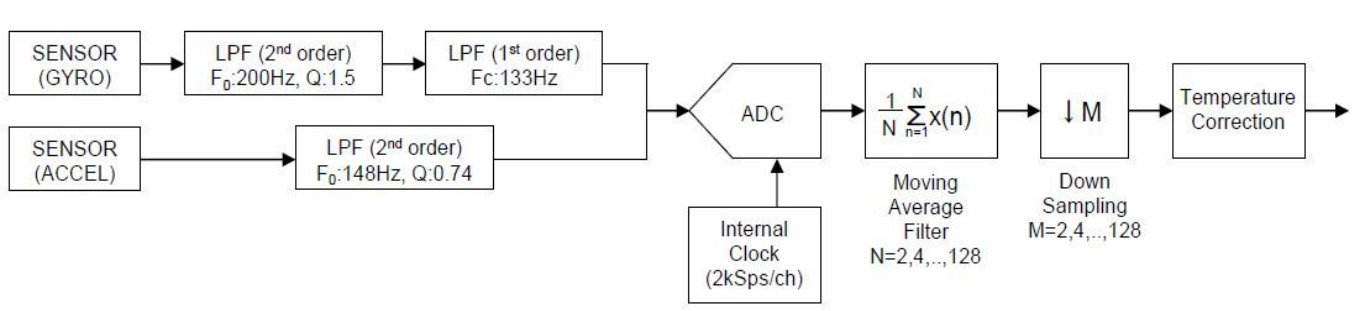


Figure 20: Functional block diagram showing internal filtering of the Epson IMU.

The Epson IMU runs several low-pass filters (LPF) and its own moving average filter before the data can be measured. The 'N' value for the filter relates to the filter order and the number of taps, as seen here:

$$y[n]=\sum_{i=0}^{n} b_i * x[n-i]$$

Where 'x' and 'y' are the input and output signal, respectively. N is the filter order which means that there will be N+1 terms total. The b_i variable is related to the impulse response and its value is $0 \leq b_i \leq N$.

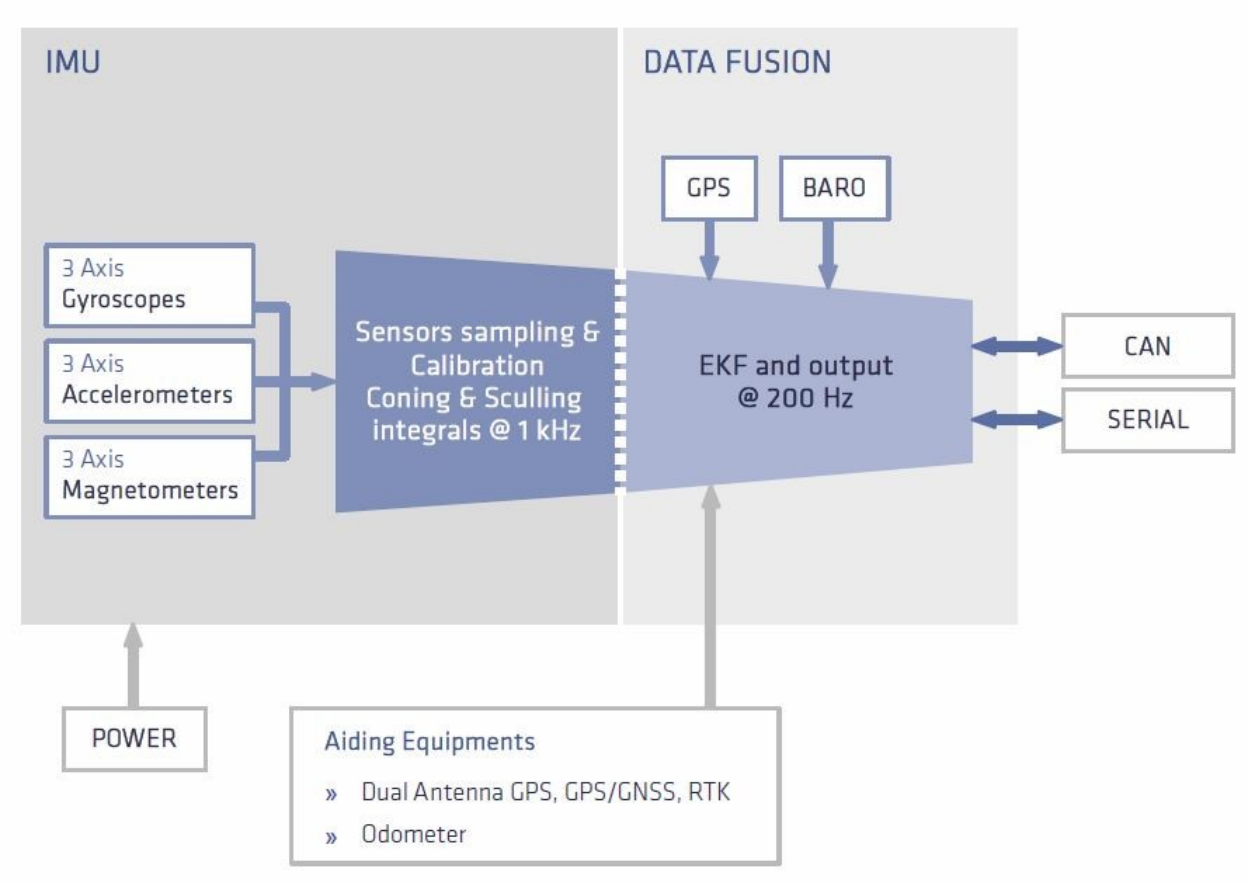


Figure 21: Simplified block diagram showing the internal filtering of the SBG Ellipse IMU.

This report doesn't deal too much with coning and sculling but a brief description is explained here. Coning is used to describe errors in measuring angular rate when integrating the measurement is not conducted fast enough. These errors must be dealt with before any meaningful calculation can be performed. Sculling is analogous to coning but for accelerometers and measuring acceleration. For the SBG Ellipse IMU, the data is then ran through an extended Kalman filter (EKF) before the data is read by the microcontroller.

Chapter 4: IMU Static Test Comparison

4.1 Bias and Standard Deviation

Static tests are the first method used in determining the bias of a sensor. This process consists of letting the sensor run on a stationary, flat table for several hours. Once completed, you can then graph the sensor output over time and determine the bias, which is the average value. At first glance, this shows how much background noise is measured when no other forces are involved. Shown below are the results of a five hour test conducted with the Epson and SBG Ellipse IMUs. The first two plots, Figure 22 and Figure 23, show the raw data measured (blue) from the gyroscopes along with the data plotted with the bias subtracted out (red). These tests were conducted at NASA Ames in Mountain View, CA

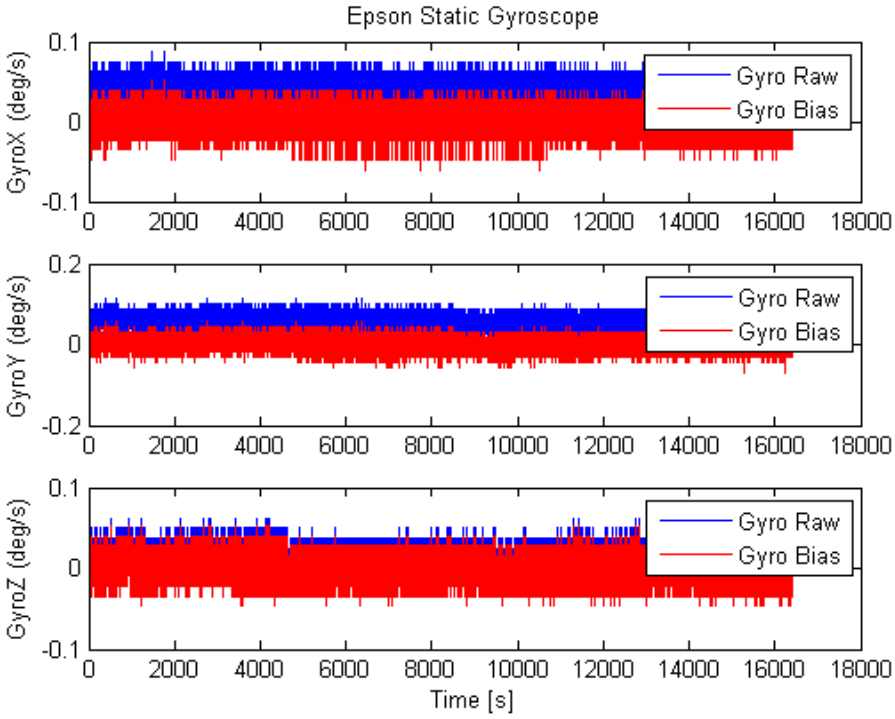


Figure 22: Static measurements from Epson gyroscope over a five hour test.

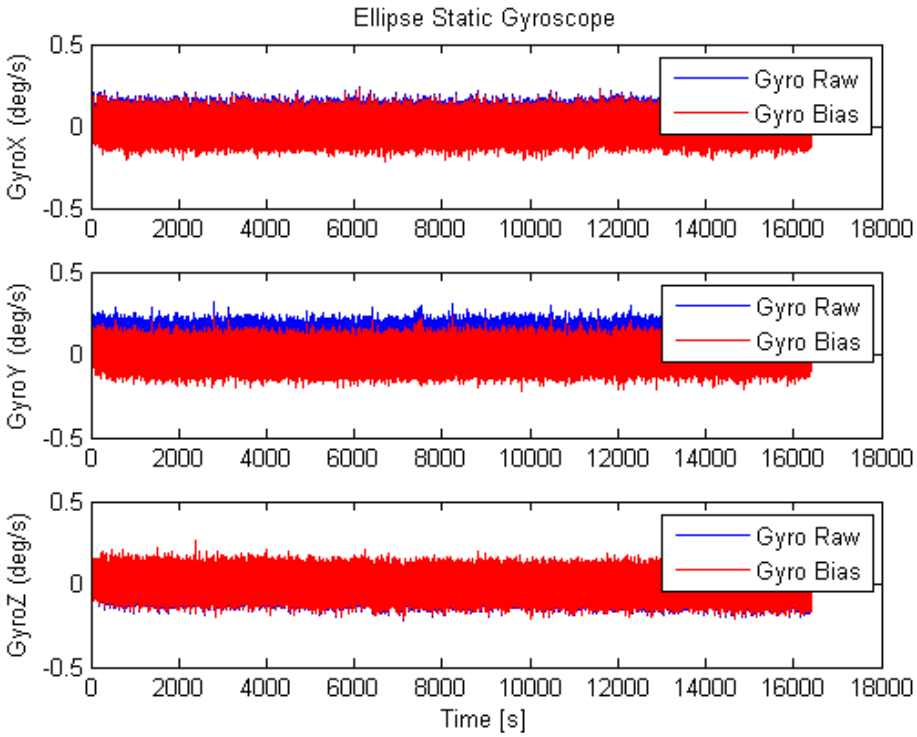


Figure 23: Static measurements from Ellipse gyroscope over a five hour test.

The accelerometers for both devices were also measured, with the Epson accelerometer in Figure 24 below. These results are from our first static test conducted using an Epson IMU.

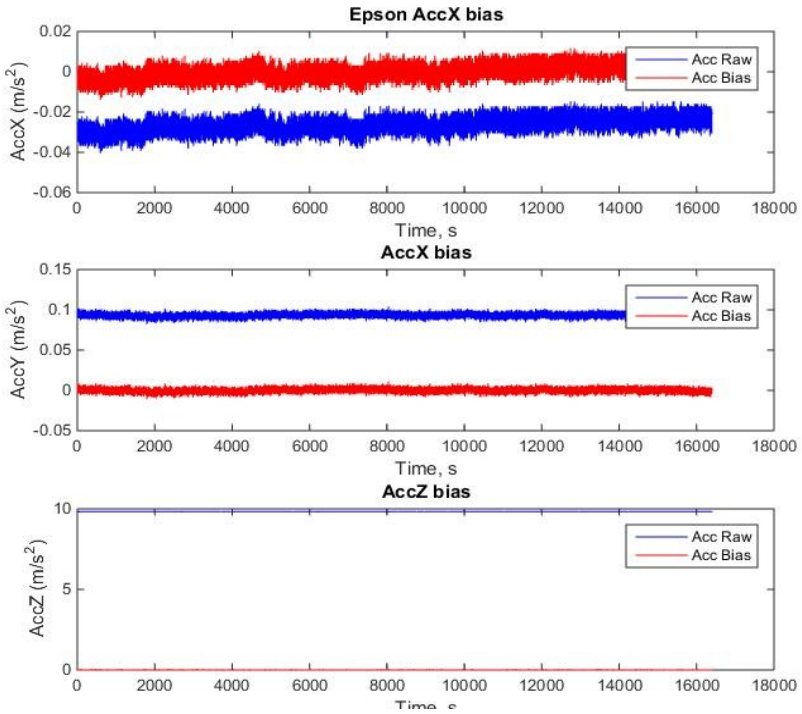


Figure 24: Static tests of Epson Accelerometer showing the measured data (blue line) and the data with the bias subtracted out (red line).

It makes sense that the bias line hovers around zero because taking out the background noise should result in no other recorded measurements. The same data for the SBG Ellipse accelerometer is shown below in Figure 25. Please note that the Ellipse accelerometer outputs the z-axis data in negative units. Table 1 shows a side-by-side comparison of the average and standard deviation values for each IMU's gyroscope and accelerometer.

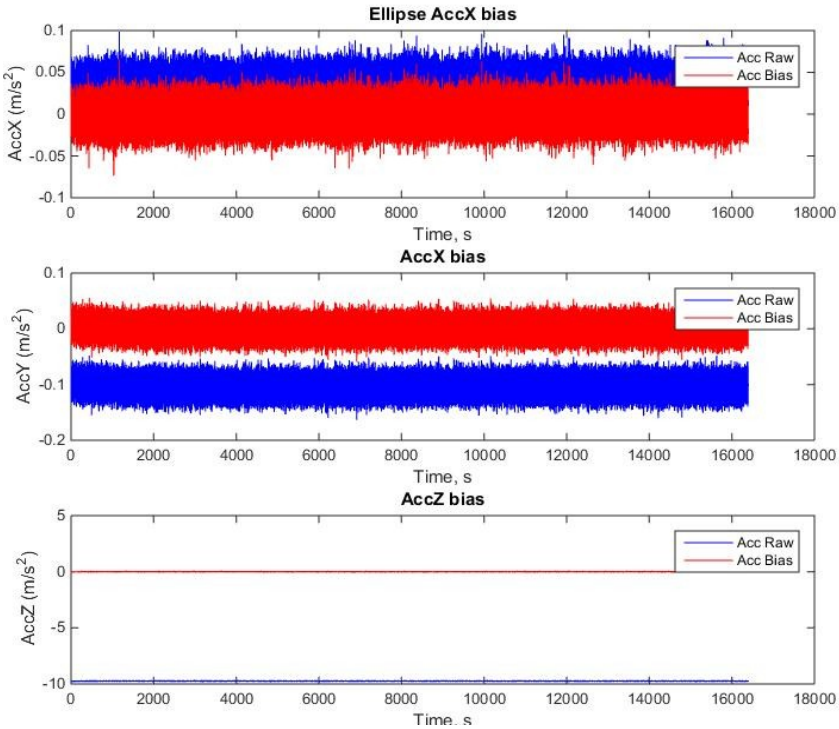


Figure 25: Static tests of Ellipse Accelerometer showing the measured data (blue line) and the data with the bias subtracted out (red line).

Table 1: Comparison of average and standard deviation values for Epson and Ellipse gyroscope and accelerometer.

Epson Gyro **Epson Accelerometer**

	avg[deg/s]	**std**		**avg[m/s]**	**std**
gyroX	0.03554	0.01197	**accelX**	-0. 02644	0. 00330
gyroY	0.05621	0.01313	**accelY**	0. 09312	0. 00195
gyroZ	0.00967	0.01312	**accelZ**	9.82571	0. 00173

Ellipse Gyro **Ellipse Accelerometer**

	avg[deg/s]	std		avg[m/s]	std
gyroX	0.01763	0.04857	**accelX**	0. 03338	0. 01287
gyroY	0.07806	0.04877	**accelY**	-0.10378	0. 01300
gyroZ	-0.00626	0.04994	**accelZ**	-9.73795	0. 01950

These tables show that the Epson gyroscope and accelerometer clearly haave less noise than the SBG Ellipse. The standard deviation along all three axes is roughly 4 times lower in the Epson which is a significant amount. Of course, this is the reason for measuring the quiescent bias. Once this value is measured, it can easily be subtracted out to obtain a static test measurement closer to zero.

4.2 Allan Variance Analysis

Another way of analyzing the sensor data is looking at the Allan Variance plot, shown in equation form as[30]:

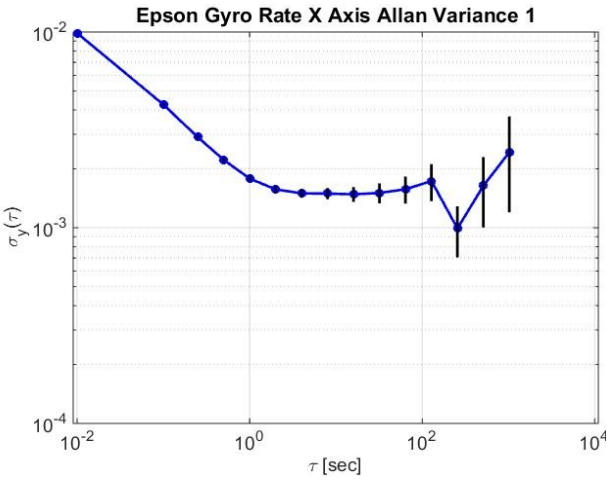
$$AVA\,R^2(\tau)=\frac{1}{2\cdot(n-1)}\sum_i\left(y(\tau)_{i+1}-y(\tau)_i\right)^2 \tag{4.1}$$

Which states that the Allan Variance, $AVAR(\tau)$, is a function of the averaging time, τ. To put simply, when dealing with a large pool of data, it is required to separate it into bins based on the averaging time[30] (ranging anywhere from $\tau=0.1¿\tau=10^3¿$. For a tau of one second, the data is looked at over intervals of one second. Following the equation, it is necessary to take the average of all the values in each of these one-second bins. The next step is to take the difference between each successive bin's values, square this value and take the overall sum before dividing

by the scaling factor. The last step is taking the square root. This process will yield one value in the graphs shown below.

To obtain the full Allan Variance plot, the process must be conducted at least 9 times[30] with an increase in tau each time. The plots used here use a total of 15 bins. For equation 4.1, the number of bins used is displayed as 'n' while y_i is the averaged value in each time interval. The Allan Variance plot is described more in Appendix A.

Shown in Figure 26, the Allan Variance plot of the gyroscope z-axis sensor gives a more in-depth look into what noise is actually measured. This plot is also helpful in comparing the results with the given datasheet of the sensor. Figure 27 shows the Allan Variance plot of the accelerometer along the z-axis.



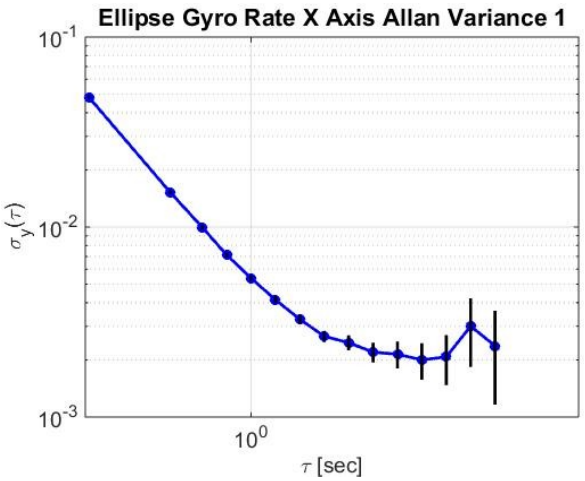
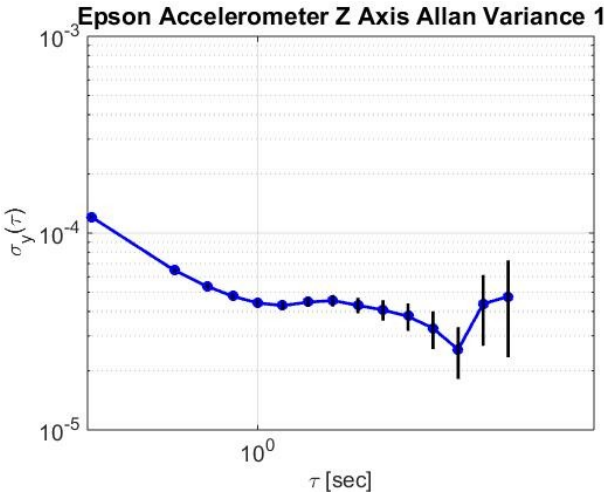


Figure 26: Side-by-side comparison of Epson and Ellipse Allan Variance plots of their x-axis gyroscopes



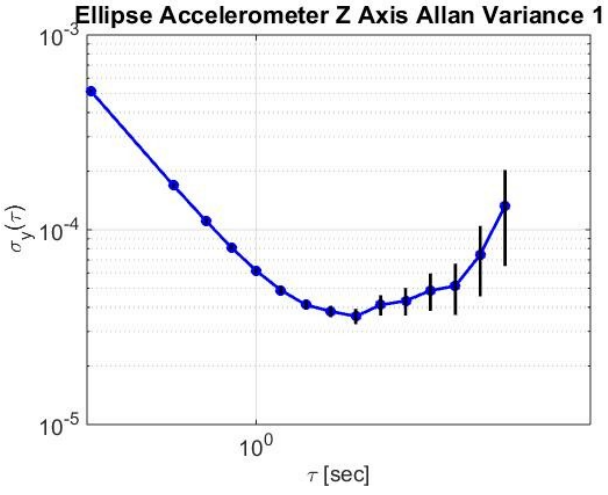


Figure 27: Side-by-side comparison of Epson and Ellipse Allan Variance plots of their z-axis accelerometers

Certain values, like angular random walk (ARW) and bias stability can be calculated from the plot and compared. Please refer to Appendix A for a more in-depth explanation. Table 2 shows these values for all axes of the gyroscope:

Table 2: Angular Random Walk and Bias Instability of Epson and Ellipse Gyroscope

Epson Gyroscope	**x-axis**	**y-axis**	**z-axis**	**Epson Datasheet x-axis**
ARW ($\frac{deg}{\sqrt{hr}}$)	0.0710	0.0917	0.8686	0.2
Bias Instability ($\frac{deg}{hr}$)	8.63	8.297	9.72	6

Ellipse Gyroscope	**x-axis**	**y-axis**	**z-axis**	**Ellipse Datasheet x-axis**
ARW ($\frac{deg}{\sqrt{hr}}$)	0.329	0.3136	0.3240	0.16
Bias Instability ($\frac{deg}{hr}$)	8.67	14.91	11.39	8

4.3 Running Standard Deviation from Static Test

Figure 28 shows plots from data obtained through a 5 hour static test in the lab. The three subplots show the three axes for both of the IMUs. This is to show how the long-term integration behaves.

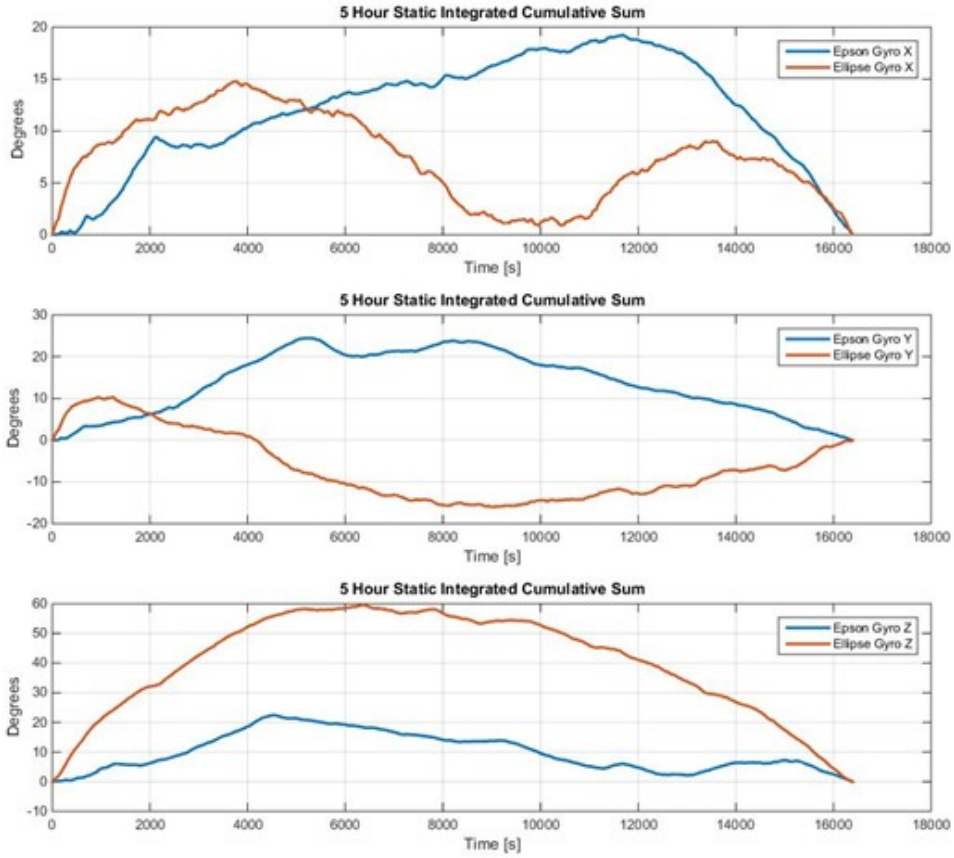


Figure 28: Cumulative sum from previous 5 hour static test.

4.4 Gyroscope Simulation in Simulink

Simulating control devices is an important part of creating a full GNC system. A simple gyroscope model can be represented by its initial rotation rate, ω , a constant bias, N_b ,

angular random walk (sometimes referred to as white-band noise), N_w , and a sensitivity, N_s , which is represented as a simple linear gain. This model is shown below

$$g = \omega + N_b + N_w + N_s \tag{4.2}$$

These values should be readily available in the device's datasheet and can be easily verified after an initial measurement is taken over sometime. The constant bias is just the mean value of the measurement over time, so once this value is calculated it can be easily subtracted out. The angular random walk is calculated with use of an Allan Variance plot and is described in detail in Appendix A while the sensitivity is just a value found on the data sheet.

To set this up in Simulink as a stationary test, meaning the device would just be sitting on a table with no rotation, the following block diagram (Figure 29) is used:

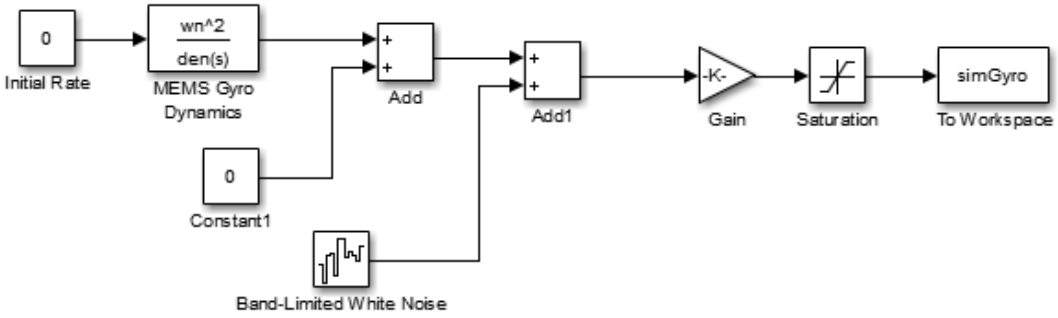


Figure 29: Simulink block diagram of the Epson gyroscope.

The 'MEMS Gyro Dynamics' block is a transfer function that simulates a MEMS system and the 'Saturation' block is there to limit the input as the gyro only reads up to 16 bits worth of data. This block diagram is setup to simulate the Epson gyroscope that was tested for this report but with an initial bias of 0 in order to verify this value after we run a test. Using the walk bias of $0.2\frac{deg}{\sqrt{hr}}$ and gain sensitivity 0.125, the following data is outputted for a five hour simulation (Figure 30):

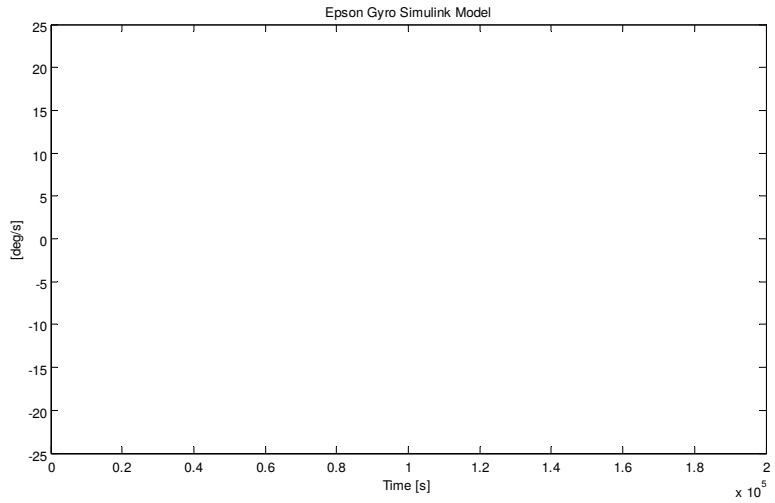


Figure 30: Data generated from Simulink model of a gyroscope.

This data has a mean value, or bias, of -0.045 deg/s. This does appear to be slightly different than the measured data but the model used is very simplistic and does not implement any low pass filters. This is just a short demonstration to show how easy it is to use Simulink to get a quick simulation of a gyroscope. Running this data through an Allan Variance plot yields the following graph (Figure 31). The key aspect to take from this is that the general shape is similar to the actual data even if the values are off.

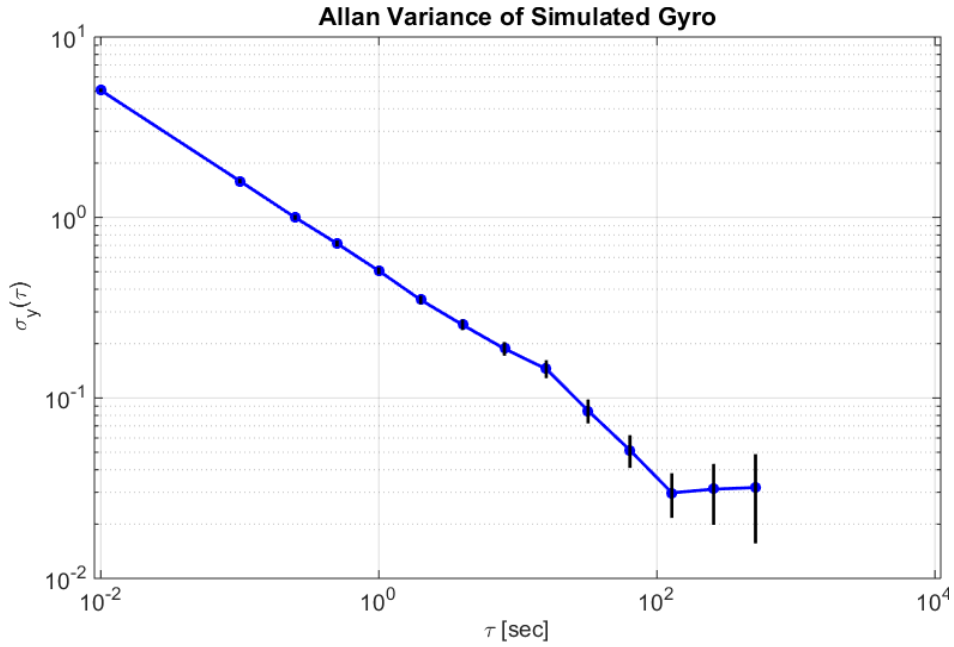


Figure 31: Allan Variance plot of the simulated gyroscope.

.

Chapter 5: Vibration Table Setup and Analysis

5.1 Vibration Test Preparation

The goal of our vibration test is to see how the IMUs react when vibrated at similar frequencies to that experienced on a real mission. Our current set of IMUs will be used for a project named AstroBEE, which will consist of a small, automated robot flying around the International Space Station. This robot will be propelled by a number of fans (still to be determined) which means our vibration test should mimic similar conditions.

5.2 Types of Vibration Tests

Vibration testing over a broad range of frequencies is important to understand how a device naturally vibrates. This report primarily looks at sinusoidal testing and random vibration testing. Sinusoidal testing is a good starting point and gives a good idea of how the device behaves over a range of frequencies. The goal of covering a large range of frequencies increases the chance that the device will vibrate at its maximum amplitude, or resonance frequency. This frequency is of great important as it could cause the most harm to the device if left to linger too long. Further tests could be conducted to measure fatigue life of the device by vibrating at the resonant frequency until damage occurs[28]. Sinusoidal vibrations are more of an ideal situation and are thus not found in every-day situations because they only focus on one frequency and amplitude, as shown in Figure 32.

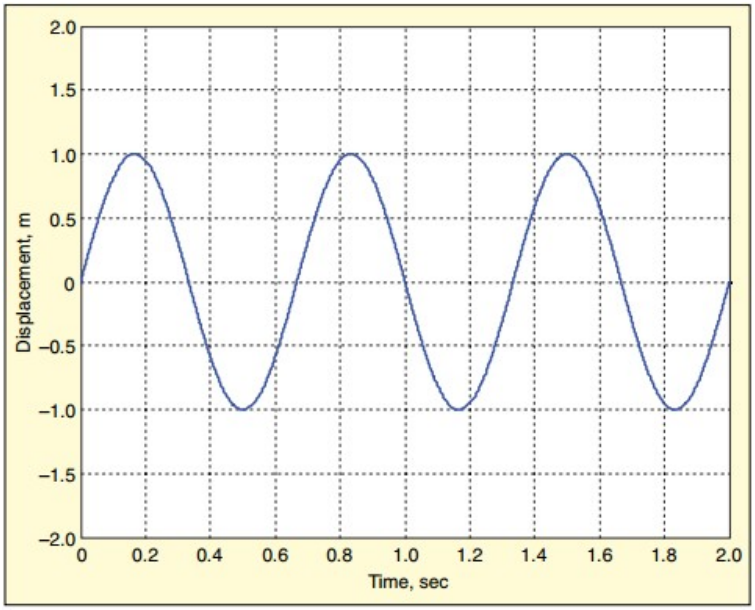


Figure 32: Sample sinusoidal time history showing a predictable pattern.[26]

It is also important to repeat the sine sweep after any other tests are conducted in order to repeat the same output. Any shift of resonant frequencies may be a sign of damage. To better simulate these scenarios, random vibration testing is used.

Normal vibrations usually do not contain predictable patterns which means only relying on a sinusoidal test is not enough. Random vibrations are used to simulate a more realistic scenario because "random [testing] simultaneously includes all of the forcing frequencies and simultaneously excites all our product's resonances."[26] A typical random vibration spectrum is shown in Figure 33 through a range of 1Hz to 200Hz. The shape of this power spectral density (PSD) plot shows the average acceleration of the random signal at each frequency.

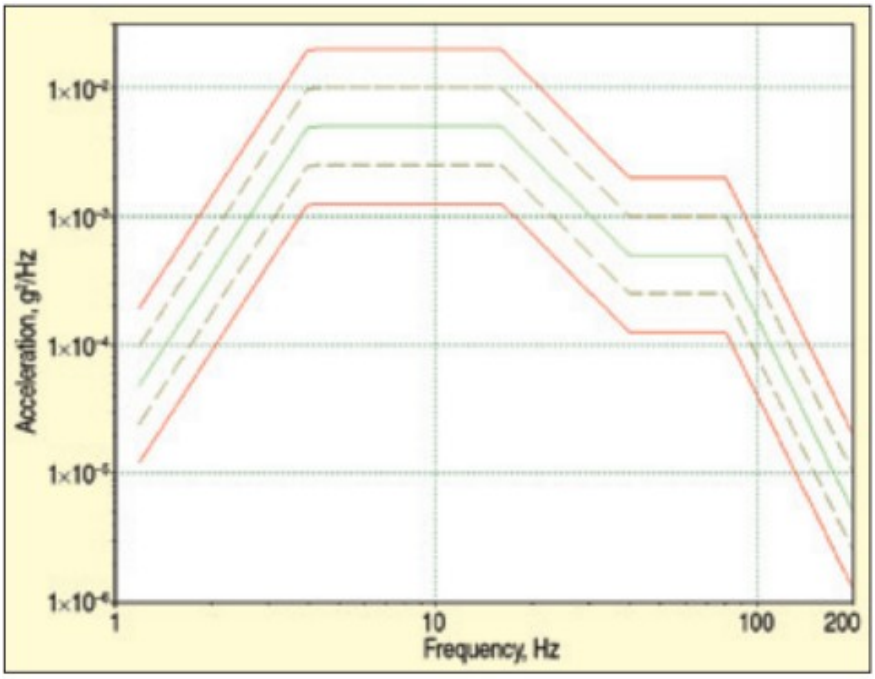


Figure 33: Sample random vibration PSD spectrum.[26]

The y-axis is a statistical measurement of the overall vibration energy, in units of g^2/Hz. This means that the area under the curve is g^2, the signal's mean square, while its square root is the root-mean-square (RMS) of the signal. Since this vibration is a random test, there is an infinite number of possible signals that can be averaged to achieve the specific magnitude at each frequency. This is the main advantage of a random vibration over its sinusoidal counterpart, real word randomness can be achieved through this process.

In preparation for the vibration test, a ¾ inch plate of aluminum (Figure 35) was cut to a size of 16 by 12 inches. Before any permanent marks and holes were made, a drawing was made in AutoCAD in order to plan out how the devices would fit (Figure 34).

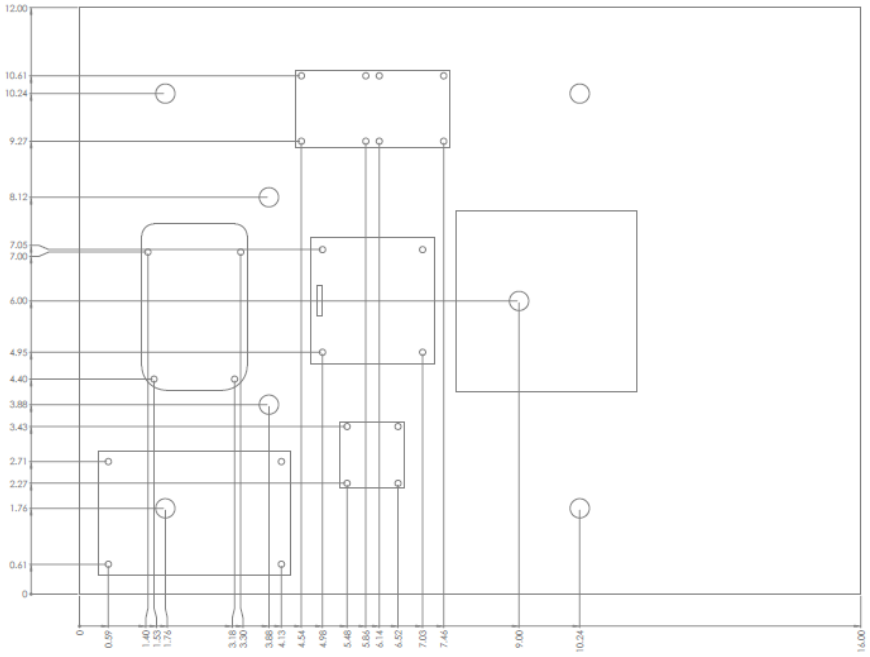


Figure 34: AutoCAD drawing showing measurements and placement of the testbed devices.

This allowed enough area to place the three IMUs, BeagleBone Black and the battery pack on with plenty of space to spare.

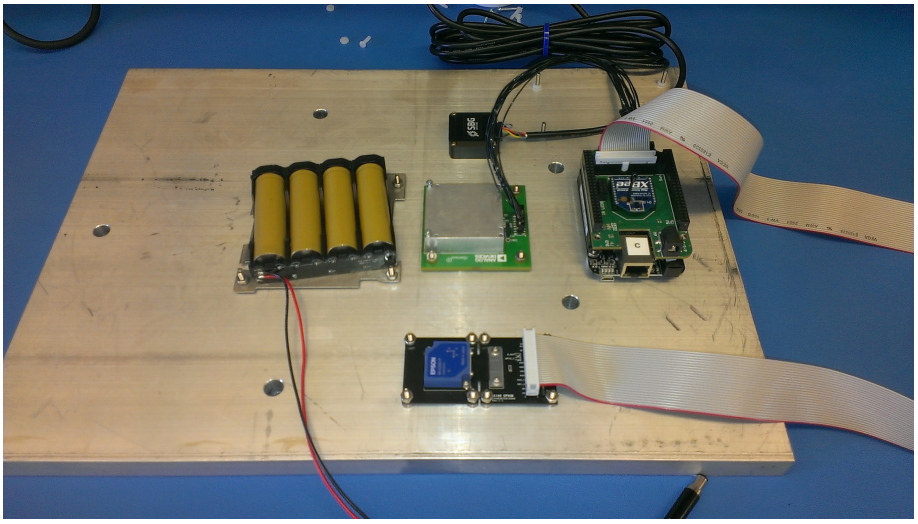


Figure 35: Test bed components screwed into aluminum plate ready for vibration test

The larger holes are matched up with the hole pattern of the actual vibration table as shown in Figure 36.

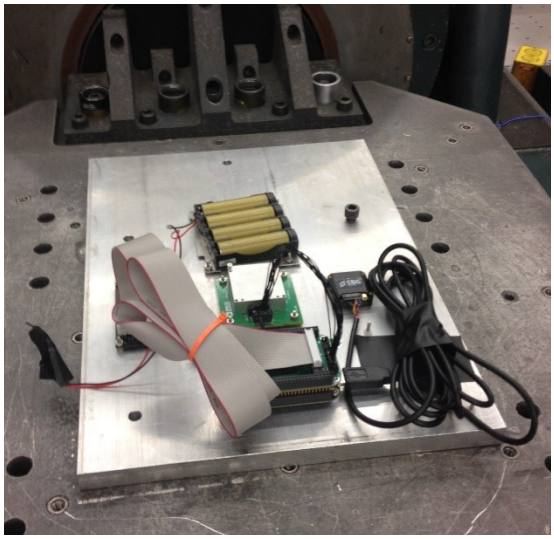


Figure 36: Testbed screwed into vibration table.

5.3 Vibration Test Parameters

The vibration table was set as followed, with frequency ranging from 2kHz to 20kHz. The PSD spectrum plots for the first Random test and first Sine sweep test are located below, while the remaining plots can be found in Appendix B. Recall that these tests were only conducted along one axis of the vibration table, aligned with the x-axis for the IMUs. The PSD shown in Figure 37 is for the first Sine sweep test conducted with a total magnitude of ⅛ g_{rms}. The graph shows a straight line because the Sine function is kept at a constant amplitude as the frequency increases from 2 Hz to 2kHz. Safety precautions are in effect as seen in the red lines, however with such a small amplitude there was no worry of causing damage to the devices.

As for the Random test (Figure 38), the maximum amplitude desired was $0.67g_{rms}$ with higher vibrations during the middle of the test. The shape of this plot is similar to the sample test shown in Figure 33.

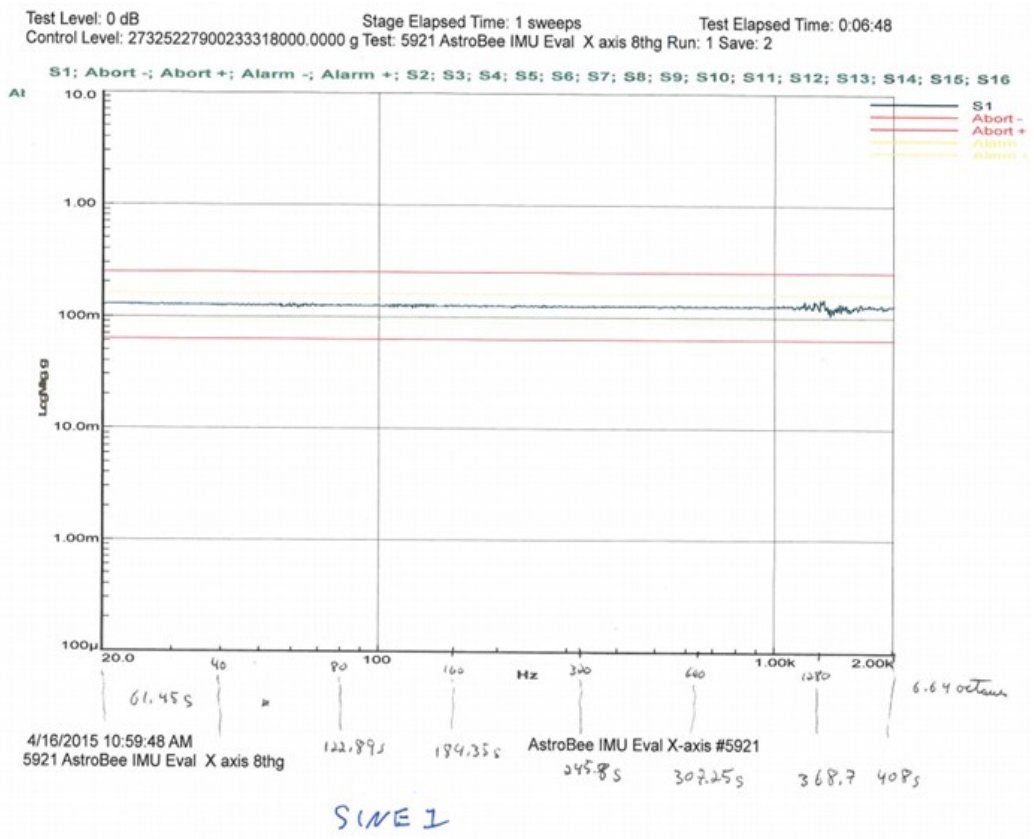


Figure 37: Sine sweep Test 1 of amplitude ⅛ grms

There are minor fluctuations seen at higher frequencies, however these are insignificant as it falls within the boundaries shown.

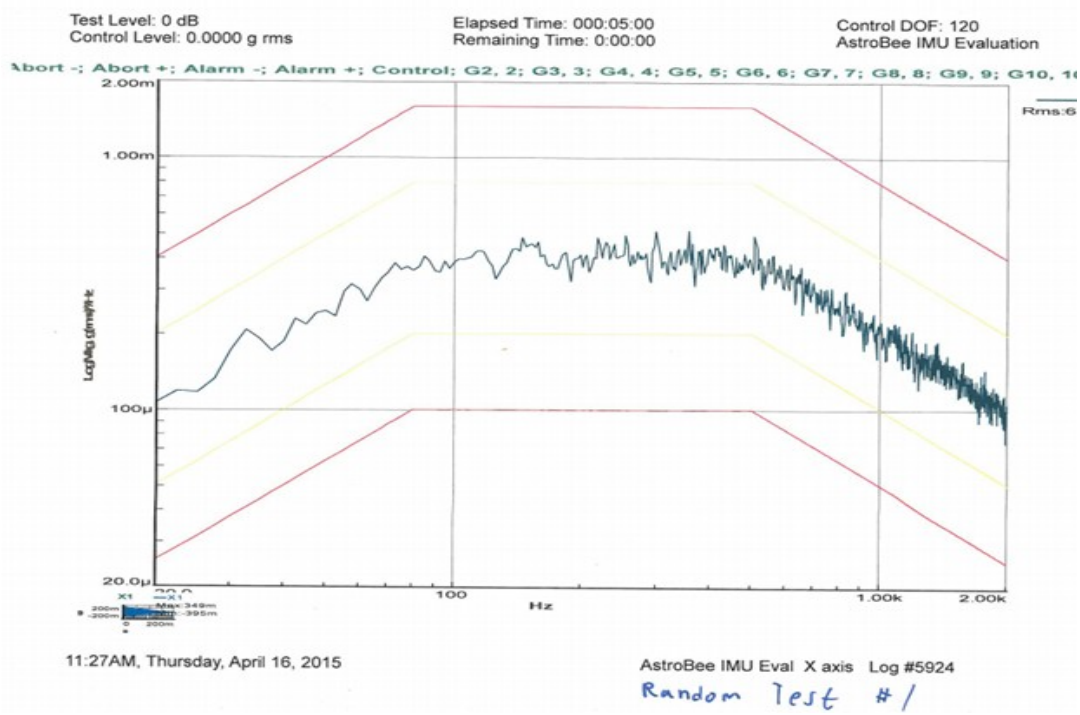


Figure 38: Random test of amplitude 0.67 grms.

- **Sine Sweep along x-axis:**

1. **1/8 G_{rms} amplitude test for 408s (6min 48s)**
2. **1/4 G_{rms} amplitude test for 408s (6min 48s)**
3. **1/2 G_{rms} amplitude test for 408s (6min 48s)**

- **Random test along x-axis:**
    1. **0.67G_{rms} amplitude test for 300s (5min)**
    2. **0.48G_{rms} amplitude test for 300s (5min)**
    3. **0.21G_{rms} amplitude test for 300s (5min)**

The test was only conducted in the x direction because the goal for this test was to locate any glaring problems so one axis was sufficient. Another reason for this is because the IMUs use identical sensors for each axis.

The rate at which the data was collected was approximately 20 Hz. This value was measured by looking at the timestep increment of each data point. The average increment is 51ms but with a standard deviation of 12ms. The reason for this uncertainty is the way the code is written in order to collect data from the three IMUs at the same rate.

The original plan was to include several minutes of static data before and after the entire test but that did not occur. To account for this, the downtime data between tests was concatenated together to provide the necessary data. Figure 39 shows the raw data taken from the Ellipse accelerometer along the x-axis during the sine sweeps.

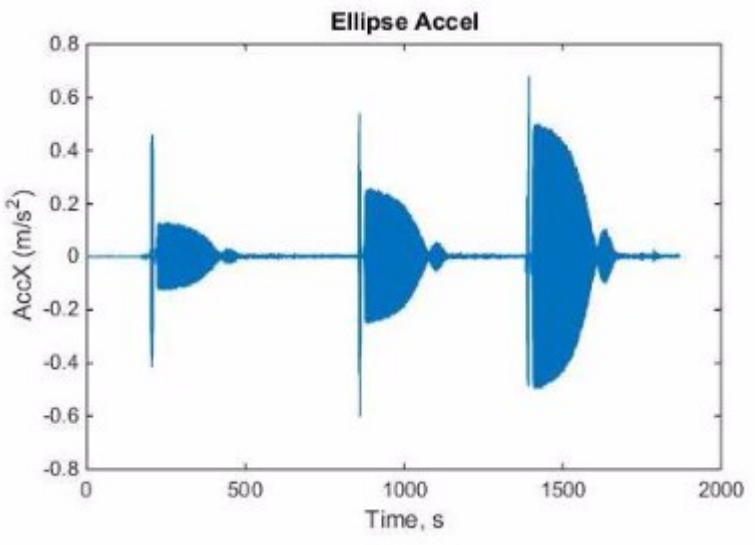


Figure 39: Raw accelerometer data from vibration test showing the sine sweep results along the x-axis. The start/end points of the three tests are easily seen here.

Using the data in Figure 39, MATLAB was used to pick apart the quiescent areas and concatenate them into one array. This yielded the necessary data that allowed a fair comparison between the vibration data and quiescent data. These results can be seen below in

and Table 4. The plot also shows that each test had increasing vibration magnitude, with a short start-up vibration as seen by the vertical line.

5.4 MATLAB Data Analysis

The testbed is programmed to read in the raw data from the three gyroscopes as well as the timestamp each time a piece of data is received and place it all in a comma separated file (CSV). This allows for a check of the frequency of data collection. In total, this file has 36 columns of data which contain the gyroscope, accelerometer, magnetometer, temperature and various other measurement devices' data. It is also important to note that this file contains the entire vibration data, including all three tests and quiescent data. With this large CSV file, it was necessary to create a simple and easy-to-use MATLAB script file that can organize and analyze the data in just a few steps. More of the MATLAB code is located in Appendix D.

5.4.1 'GyroRaw' Function

This script file starts off by reading in the CSV file and making one variable contain all of the information as seen below. Since the Sine Sweep and Random tests are separate files, the user must indicate what data will be analyzed.

```
%% Start by calling Sine Sweep data
clear all, clc, close all
raw = csvread('sineSweep.csv');
```

With this 'raw' variable, it is then necessary to separate the data into their respective tests so the data can then be broken down more. This function separates the three tests into their own

variables while also keeping the three axes distinct from each other. This is easily accomplished in MATLAB by creating a cell array of the variables seen here (only part of the code is shown):

```
RawData = {[Ep1stX] [SBG1stX] [Ep2ndX] [SBG2ndX] [Ep3rdX] [SBG3rdX]... }
```

This function then returns this cell array containing variables for all axes of all three tests for both the Epson and SBG Ellipse IMU. This cell variable is then deconstructed in each of the following functions by use of the embedded 'cell2mat' function. Once deconstructed, each individual variable can be easily accessed and manipulated.

5.4.2 'GyroRawPlots' and 'GyroStdAvg' Functions

The next function, 'GryoRawPlots,' simply plots the raw data as shown in Figure 40. This function is set up to plot either the Sine sweep test or Random test data depending on what the user asks for. It is set up in a way that can be easily modified if needed for any reason. The function takes the necessary data cells and plots them in a 4x1 subplot.

The standard deviations and averages shown in Table 1 and Table 2 in Chapter 6 are calculated in the function 'GyroStdAvg.' This function is pretty straightforward as it just prints out the data in table format. This table is then reformatted and shown in Chapter 6.

5.4.3 'GyroMoveStd' and 'VibeCumSum' Functions

This first function will plot the moving, or running, standard deviation of the desired test. This process is explained in detail below in section 6.2. The latter function is described in section 6.3 and displayed the cumulative sum plots for the data.

Chapter 6: Vibration Table Sine Sweep Analysis

6.1 Raw Data Plots from Sine Sweep

The following figure shows the plot from graphing the output data obtained directly from the vibration test. Each subplot contains plots for each axis during the first test. It is clearly seen that the SBG IMU (shown in red) has more noise compared to the Epson (blue line).

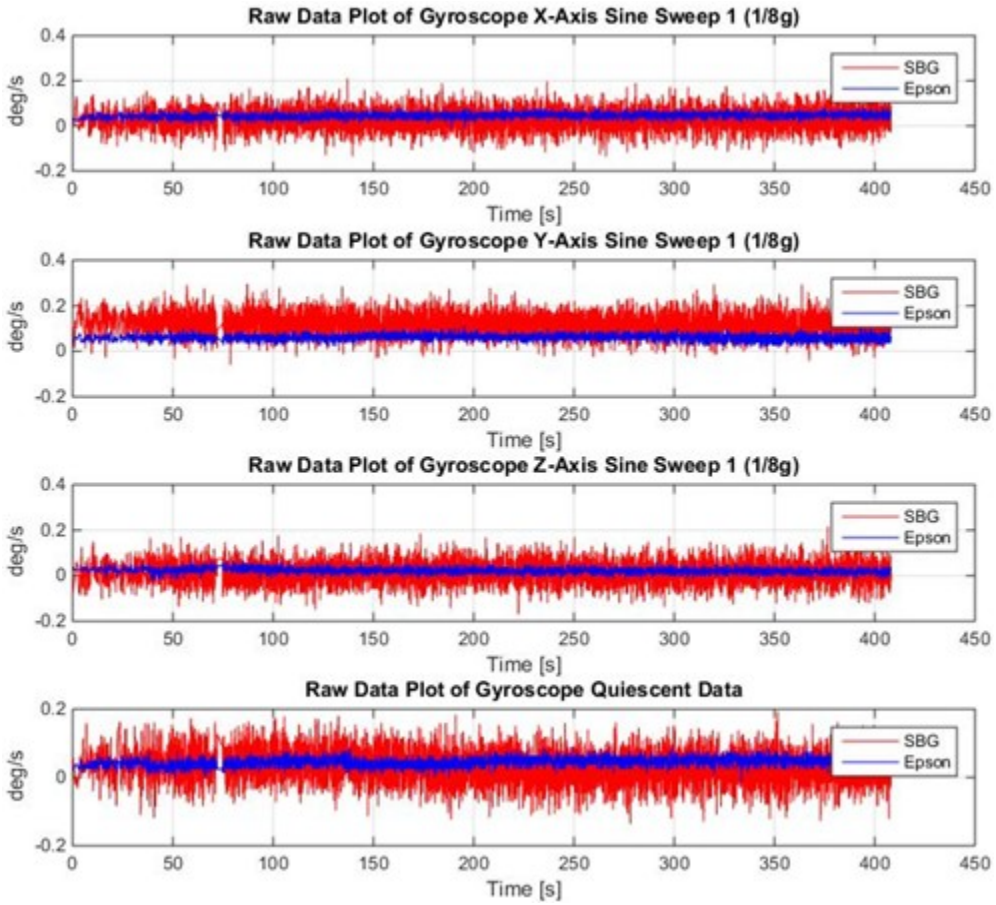


Figure 40: Raw data showing the results of the first Sine Sweep for the SBG (red) and Epson (blue) gyroscope across all three axes.

6.2 Running Standard Deviation from Sine Sweep

Calculating the running standard deviation is an analytical method to look at long-term bias in the device. It is a fairly straightforward process where the standard deviation is calculated over a set of data points, known as a window. This window then jumps forward and calculates the standard deviation again, and so on throughout the length of the test. These data points are then plotted in order to see how the standard deviation changes over time. In MATLAB, the following code does just that:

```
for i = 1:20:length(a)-50
      s(i) = std(a(i:i+49));
```

```
end
```

which sets the window size to 50 data points while jumping forward by 20 for each measurement. This allows for some overlap in order to utilize as much data points as possible. The following plots, shown in Figure 41, look at a direct comparison of a moving standard deviation between the Epson and SBG Ellipse Gyroscope. A plot of quiescent data is shown as well. A forward moving window of 50 points was used while increasing in steps of 20 points. This allows some overlap to occur to achieve the best results. Additional plots for the other axes can be seen in Appendix C.

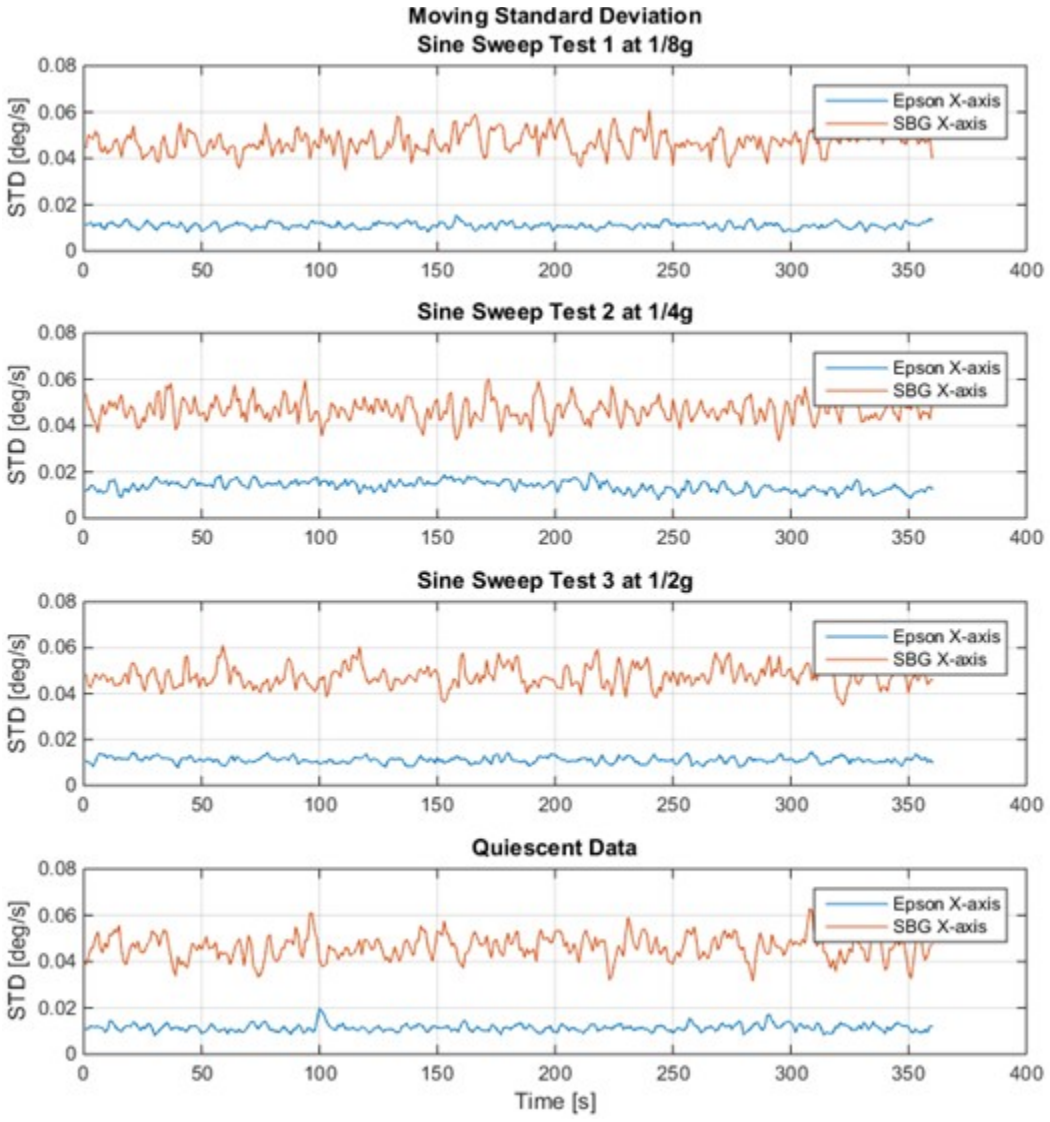


Figure 41: Running Standard Deviation plot shown for both Epson and SBG x-axis for each of the three Sine Sweep tests.

6.3 Integrated Cumulative Sum from Sine Sweep

The following plots, shown in Figure 42, compare the integrated cumulative sums of the Epson and SBG Ellipse gyroscopes for the various tests. The first step was finding the average value of each measurement in order to find the bias. This value was then subtracted out of each point get a more reliable output. With this newly constructed data (bias subtracted out), each point was

then multiplied by the change in time between each point (50ms), which effectively integrated the plot. This integrated data was then run through a cumulative sum function on MATLAB to obtain the figures shown below. The two plots show all three axes for the first test of the Random and Sine Sweep, respectively. The plots for the other tests can be found in Appendix C.

The reason that all of the plots are summing to zero is because the bias was subtracted out, thus leaving the same amount of points greater than and less than 0.

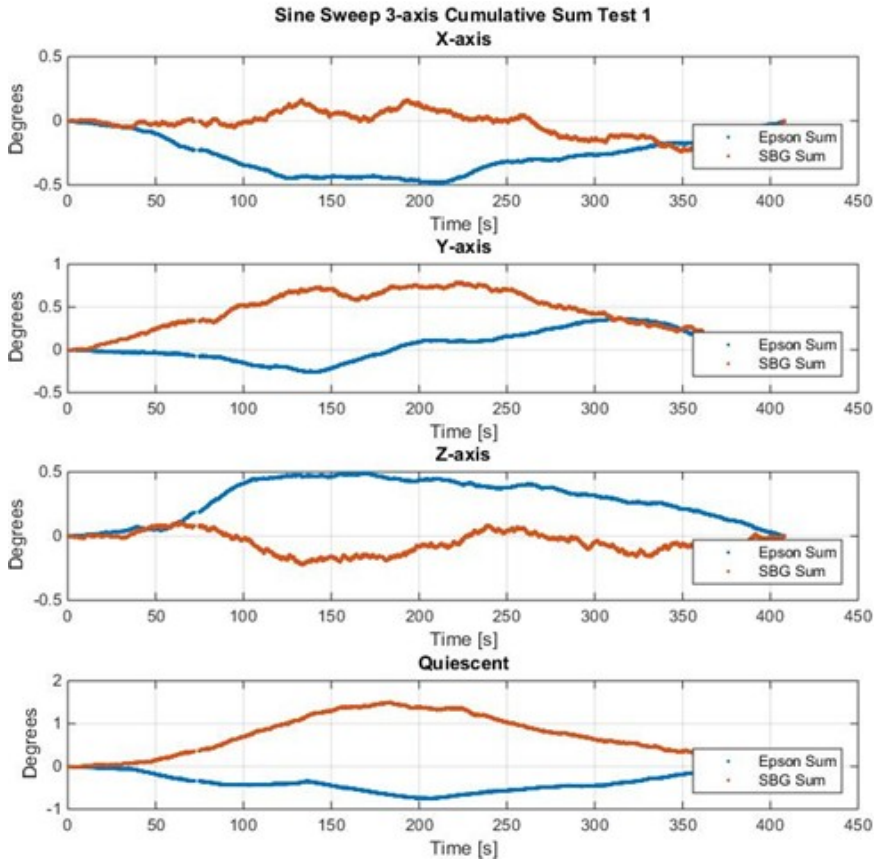


Figure 42: Cumulative sum of gyro rates for the first Sine Sweep Test at 1/8g for all three axes.

6.4 Power Spectral Density

With the help of MATLAB's built-in function, 'fft,' a PSD plot can be created. In short, a PSD tells you the strength of the energy, in this case the vibrations, across the tested frequencies. A quick glance will show where the high energy peaks and low energy troughs are. PSDs are also useful in determining any unwanted oscillatory frequencies in your system. Certain spikes in the plot may offer insight into certain vibrational frequencies to avoid and may require further investigation. A deeper analysis would involve integrating a specific frequency range to calculate the total energy within[29]

PSD plots for the Epson and SBG Ellipse gyroscope x-axis are plotted below. These plots allow investigation into the oscillatory signals of the data, specifically variations in frequency ranges that are worth taking a closer look at. The key takeaway from these plots is that the magnitude of the vibrations increases for higher frequencies. It is also worth noting that the vibration magnitude slightly decreases for each test, which makes sense as the overall energy put in is less.

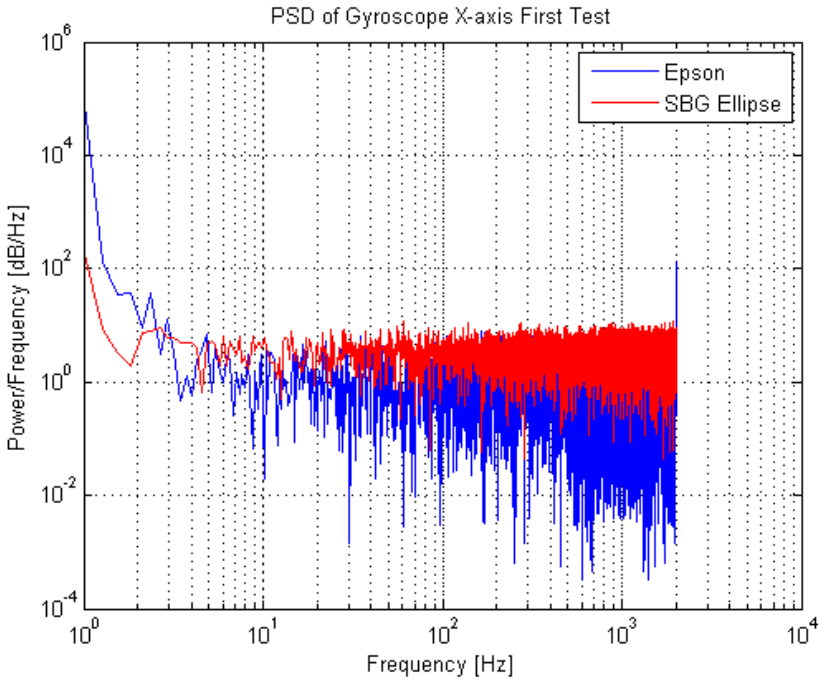


Figure 43: PSD plot of gyroscopes' x-axis for the Sine Sweep 1st test.

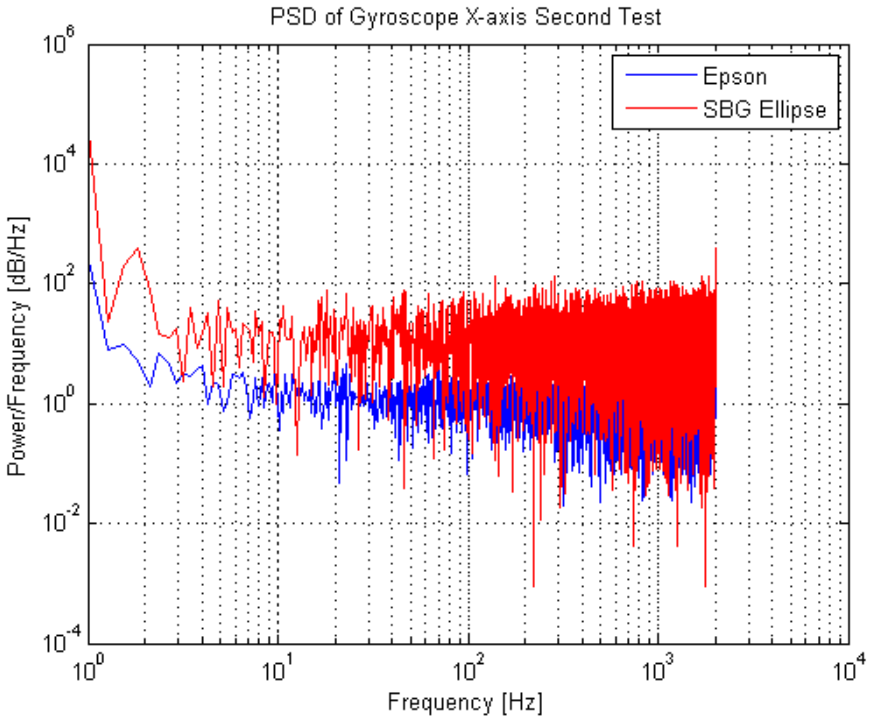


Figure 44: PSD plot of gyroscopes' x-axis for the Sine Sweep 2nd test.

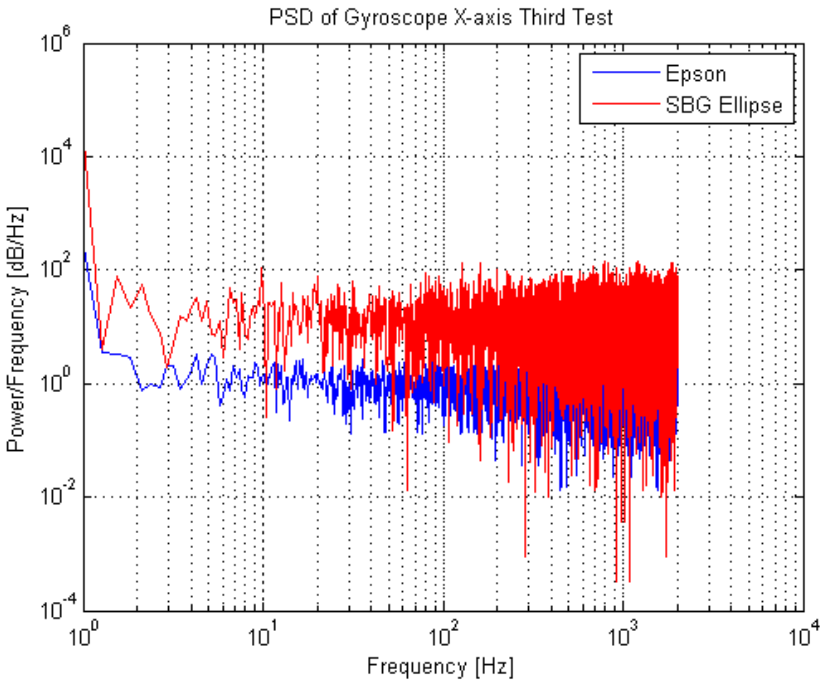


Figure 45: PSD plot of gyroscopes' x-axis for the Sine Sweep 3rd test.

6.5 Standard Deviation of Sine Sweep

The tables presented below show all of the values for standard deviation and average of each devices' axis across all three tests for both the Random and Sine Sweep.

The most notable trend here is that the standard deviation of the SBG device is roughly 4 times that of the Epson. This is thought to be caused by the internal filtering differences between the two devices.

Table 3: Standard deviation and average values are shown for both devices for the Sine Sweep test. A column showing the quiescent data is also shown for comparative reasons.

Standard Deviation Values

Sine Sweep	Gyro	1/8G	1/4G	1/2G	Quiescent
Epson	X-axis	0.0116	0.0146	0.0113	0.0126
	Y-axis	0.0139	0.012	0.0121	0.013
	Z-axis	0.0118	0.0112	0.0114	0.0117
SBG	X-axis	0.0476	0.0474	0.048	0.0475
	Y-axis	0.0478	0.0473	0.0482	0.0496
	Z-axis	0.0487	0.0481	0.0492	0.0478

Average Values

Sine Sweep	Gyro	1/8G	1/4G	1/2G	Quiescent
Epson	X-axis	0.0452	0.0401	0.0404	0.0444
	Y-axis	0.0607	0.0532	0.0549	0.0533
	Z-axis	0.0227	0.025	0.0244	0.0217
SBG	X-axis	0.0298	0.0297	0.0222	0.029
	Y-axis	0.1194	0.1127	0.1101	0.1223
	Z-axis	0.0142	0.0095	0.0083	0.0133

Chapter 7: Vibration Table Random Sweep Analysis

This section, like Chapter 5, will show the data results from each IMU but from the Random Sweep.

7.1 Raw Data Plots from Random Sweep

The following figure shows the plot from graphing the output data obtained directly from the vibration test. Each subplot contains plots for each axis during the first test. It is clearly seen that the SBG IMU (shown in red) has more noise compared to the Epson (blue line).

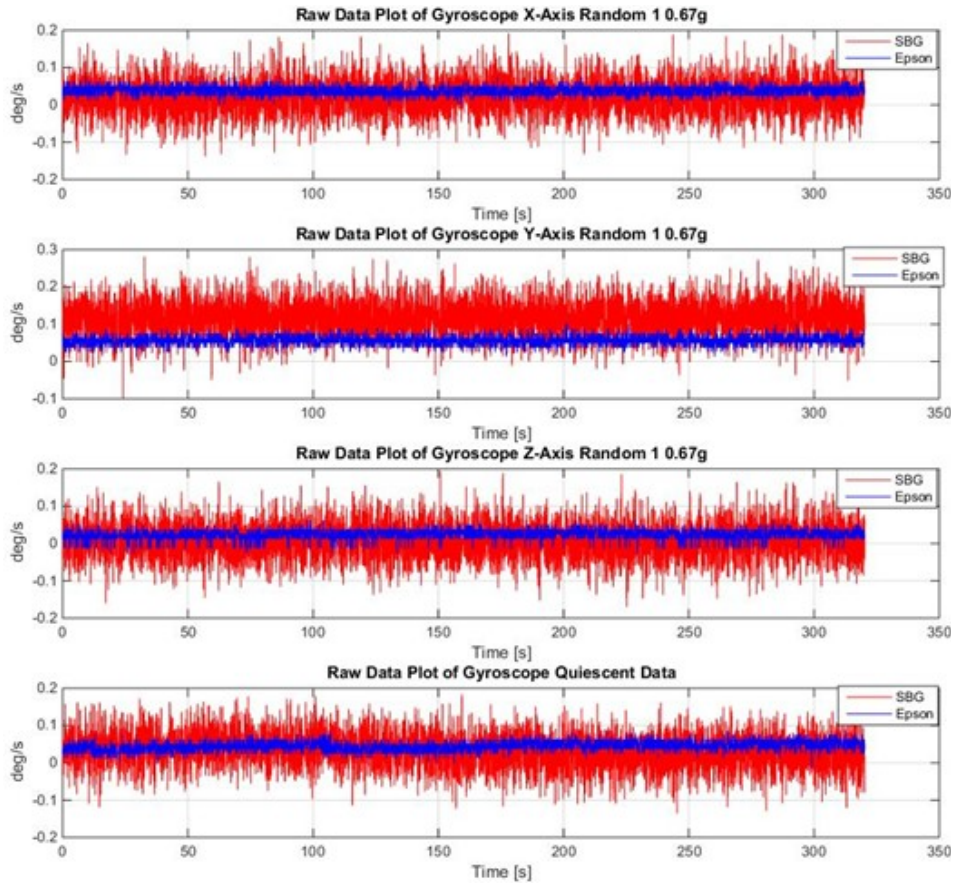


Figure 46: Raw data showing the results of the first Random Test for the Epson and SBG gyroscope across all three axes.

7.2 Running Standard Deviation from Random Sweep

The following plots will look at a direct comparison of a moving standard deviation between the Epson and SBG Ellipse Gyroscope. A plot of quiescent data is shown as well. A forward moving window of 50 points was used while increasing in steps of 20 points. This allows some overlap to occur to achieve the best results. Additional plots showing the results from the y-axis and z-axis are shown in Appendix C.

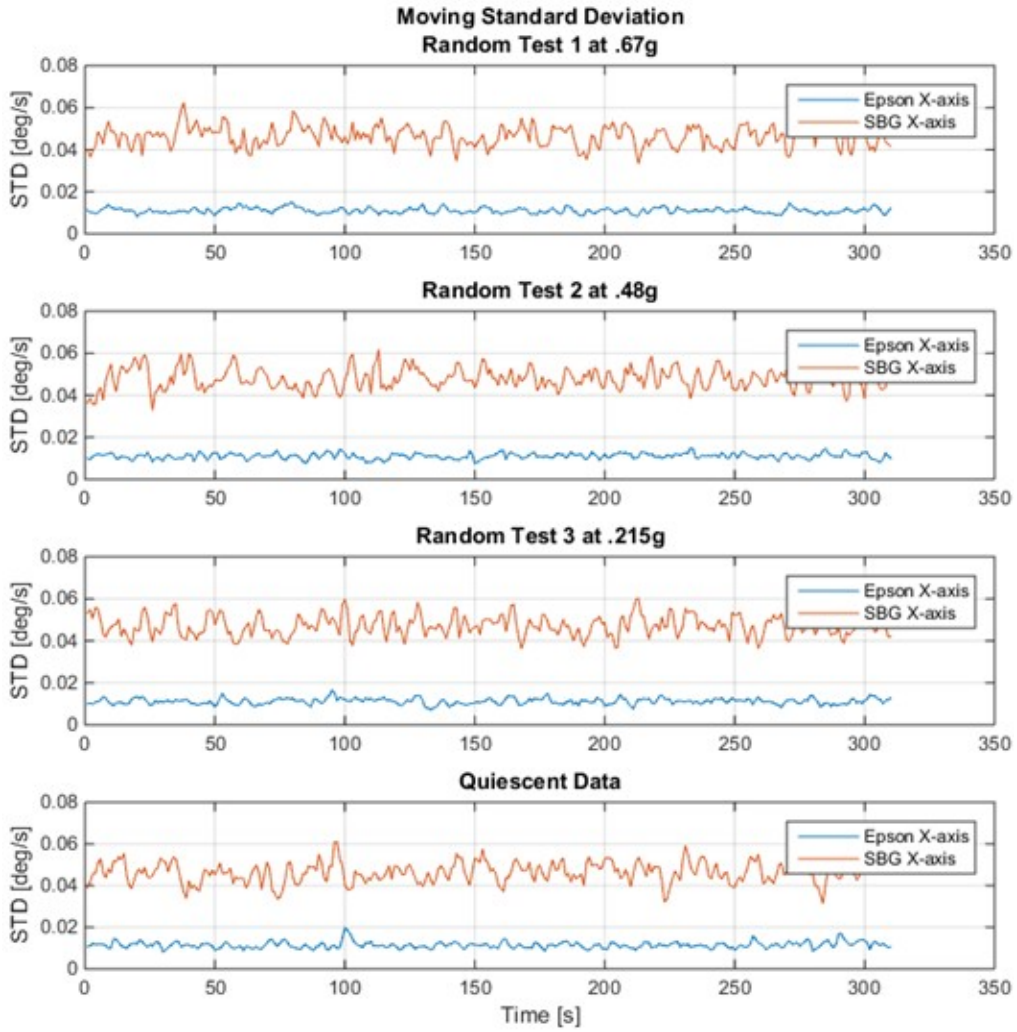


Figure 47: Running Standard Deviation plot shown for both Epson and SBG x-axis for each of the three Random Vibration tests.

These are related to the standard deviation values shown in chapter 4. The SBG Ellipse device clearly has more noise in the system, and without further compensation could be a problem for long-term activity.

7.3 Integrated Cumulative Sum from Random Sweep

A description of how the following plots were obtained can be found in section 5.3

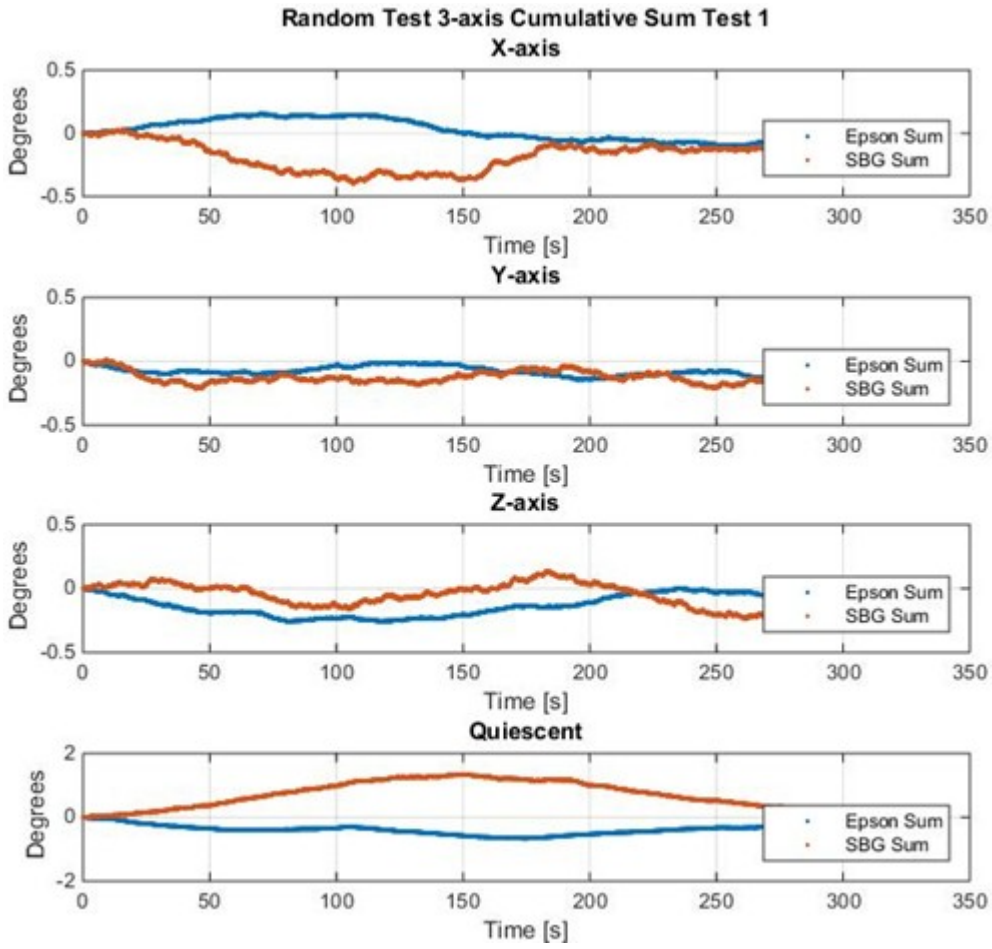


Figure 48: Cumulative sum of gyro rates for the first Random Test at 0.67g for all three axes

7.4 Power Spectral Density

Similar to section 6.4, below are the PSD plots of the gyroscopes during the random vibration test. Only the x-axis measurements are shown as that was the only axis vibrated.

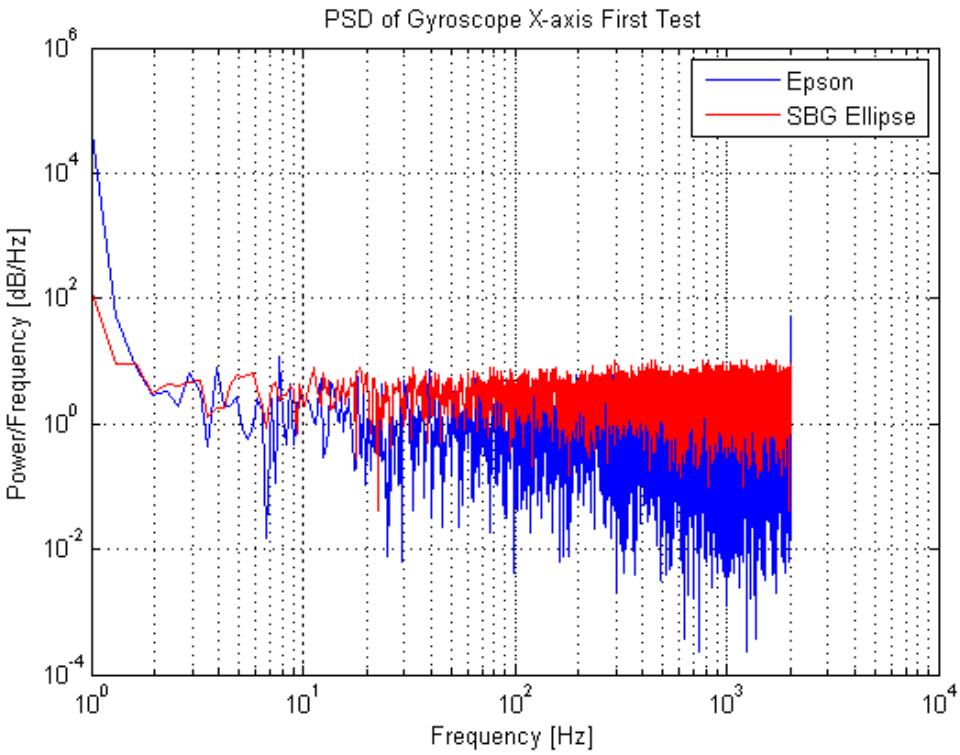


Figure 49: PSD plot of gyroscopes' x-axis for the Random vibration 1st test.

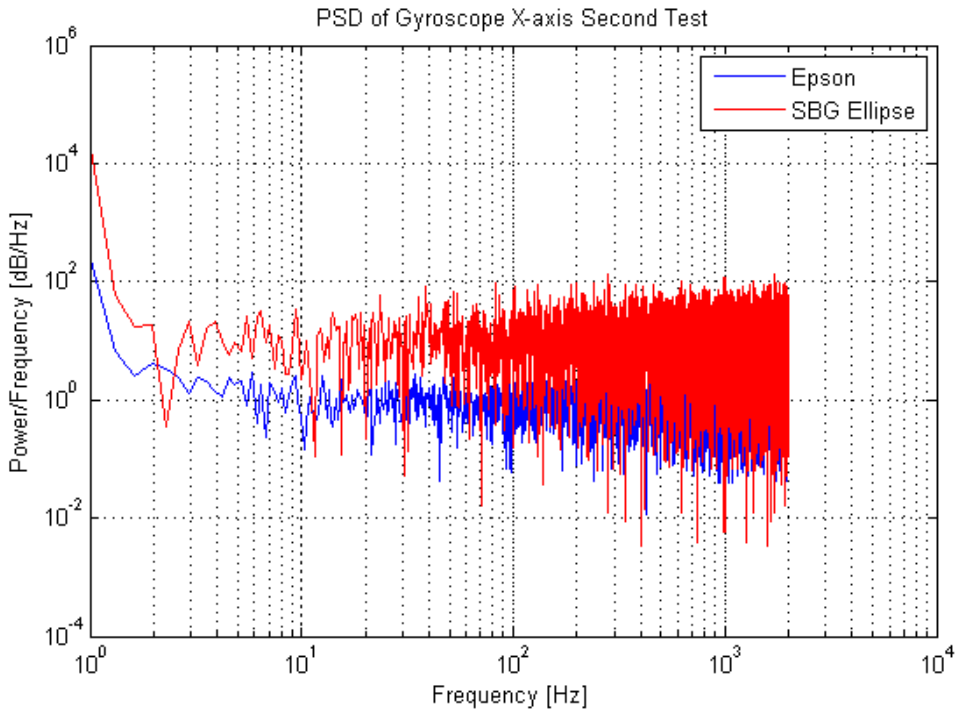


Figure 50: PSD plot of gyroscopes' x-axis for the Random vibration 2nd test.

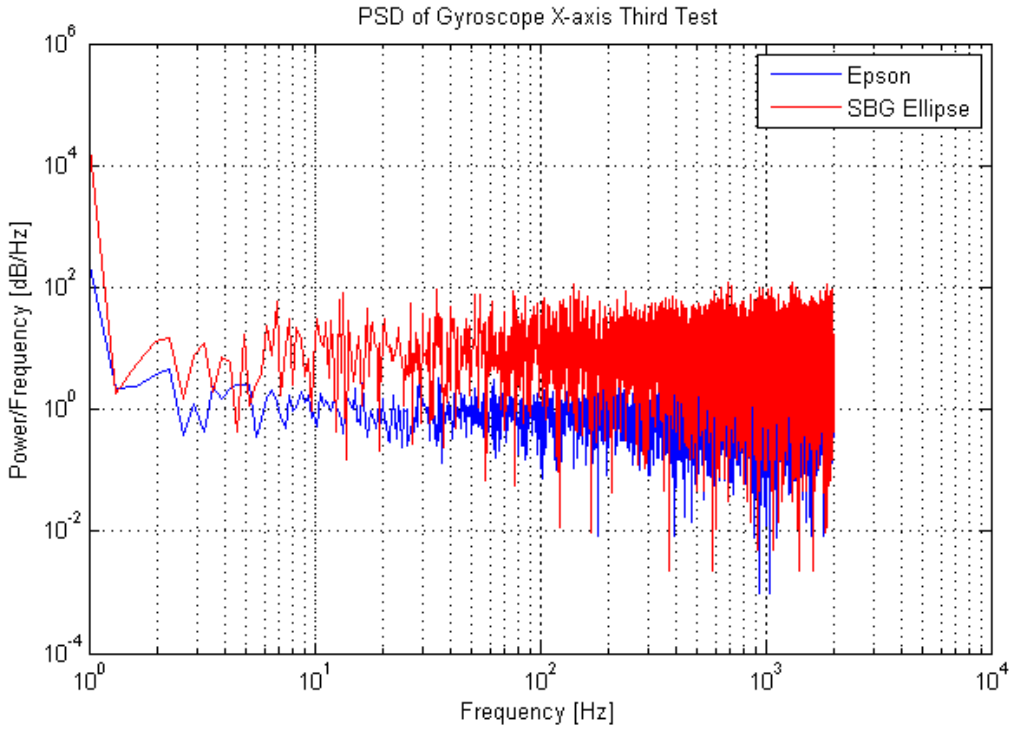


Figure 51: PSD plot of gyroscopes' x-axis for the Random vibration 3rd test

7.5 Standard Deviation of Random Test

The tables presented below show all of the values for standard deviation and average of each devices' axis across all three tests for both the Random and Sine Sweep.

The most notable trend here is that the standard deviation of the SBG device is roughly 4 times that of the Epson. This is thought to be caused by the internal filtering differences between the two devices.

Table 4: Standard deviation and average values are shown for both devices for the Random test. A column showing the quiescent data is also shown for comparative reasons.

Standard Deviation Values

Random Test	Gyro	1/8G	1/4G	1/2G	Quiescent
Epson	X-axis	0.0114	0.0115	0.0115	0.0126
	Y-axis	0.0121	0.0121	0.0123	0.013
	Z-axis	0.0124	0.0141	0.011	0.0117
SBG	X-axis	0.0469	0.0481	0.0473	0.0475
	Y-axis	0.0474	0.0478	0.0468	0.0496
	Z-axis	0.0489	0.0483	0.048	0.0478

Average Values

Random Test	Gyro	.067G	.48G	.215G	Quiescent
Epson	X-axis	0.0379	0.0419	0.0433	0.0444
	Y-axis	0.0564	0.0556	0.0569	0.0533
	Z-axis	0.0244	0.0174	0.0136	0.0217
SBG	X-axis	0.0224	0.0236	0.0261	0.029
	Y-axis	0.1162	0.1176	0.1112	0.1223
	Z-axis	0.0074	0.0048	0.0081	0.0133

Chapter 8: Conclusion

8.1 Summary

With the advancement of research and development into smaller and smaller satellites, it is important to find the most efficient, yet price-conscious inertial measurement unit. This report gave an overview of a couple of devices, the Epson and SBG Ellipse IMU, in regards to static and vibration testing and a way to use data analysis software like MATLAB to easily produce post-processing analysis. From the calculations, it is clearly seen that the Epson IMU initially performs better than the SBG Ellipse in both the static and vibration test. The word initially is used here because more tests need to be conducted which is explained in the section below. Having said that, these first glance measurements do favor the Epson IMU in all of the tests conducted. This could result in a number of reasons, from internal settings to intrinsic noise within the device itself.

Several MATLAB functions have been shared to allow any user to easily generate the necessary tables and plots to ascertain the validity of one of these devices. The initial setup may require some tinkering but once a spreadsheet of data is collected, the functions readily work from there.

8.2 Future Work

This report provides the first steps in understanding which IMU will work best for long-term use onboard a satellite. It would be ideal to run more characteristic tests, including, but not limited to linear velocity tests, thermal tests and a more intense vibration test. It is also necessary to understand the inner workings of the actual device, knowing how the actual circuitry and internal settings are set up. One solution for this would be to create a better system for how the data is read from the IMU to the microcontroller. As mentioned in the report, the current system is not the best, however, it is sufficient to determine any large problems with the devices.

Appendix A - Allan Variance Analysis

A.1 Averaging Time

To better understand the meaning of using averaging time intervals, the following graphs are provided by *Stockwell*[30]. Figures A1, A2 and A3 help visualize what it means when $\tau = 1, 10,$ and 500 . The Allan Variance plot is dominated by sensor noise when looking at short averaging times. However, over larger averaging times, the variance decreases. This can be seen in the graphs below that as you increase the averaging time, a better representation of the bias is obtained and thus this value is closer to the initial measurement.

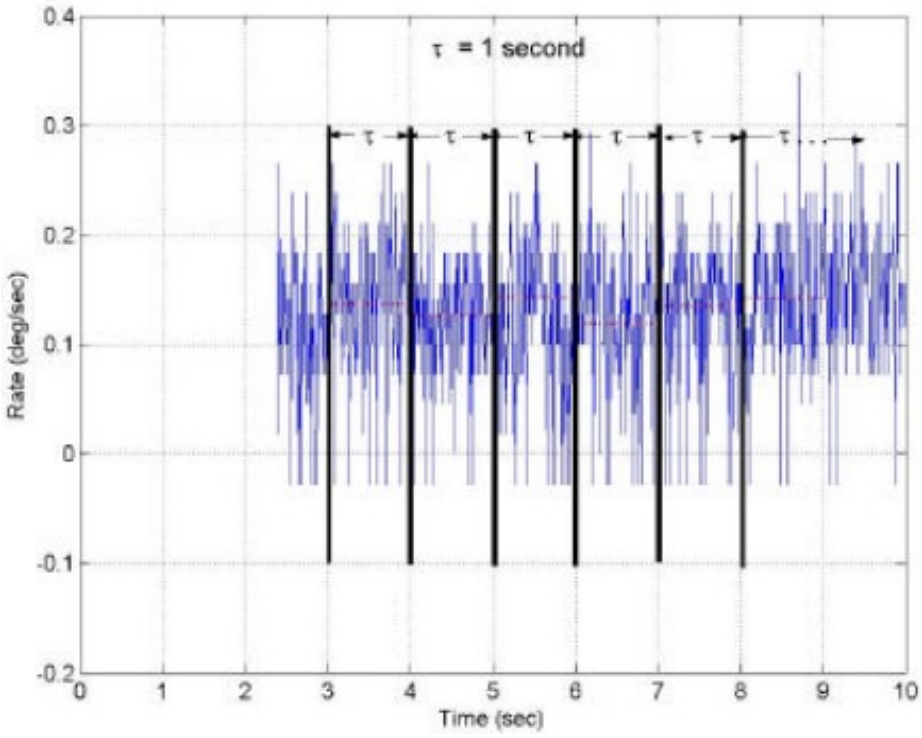


Figure A1: Sample data showing time intervals for τ = 1

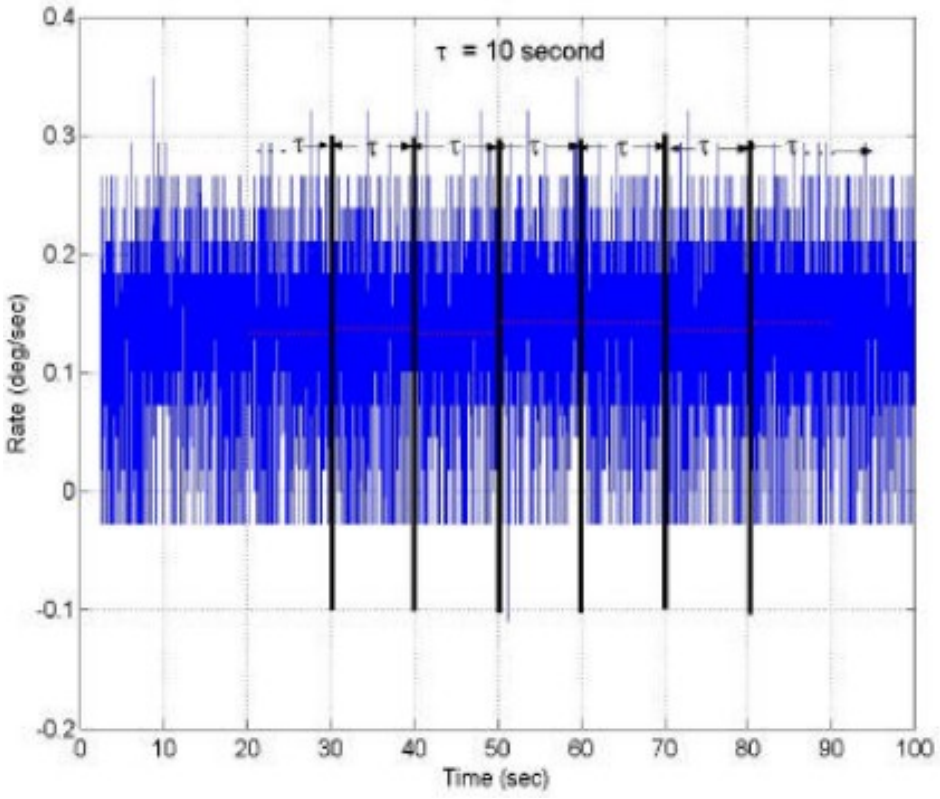


Figure A2: Sample data showing time intervals for tau = 10

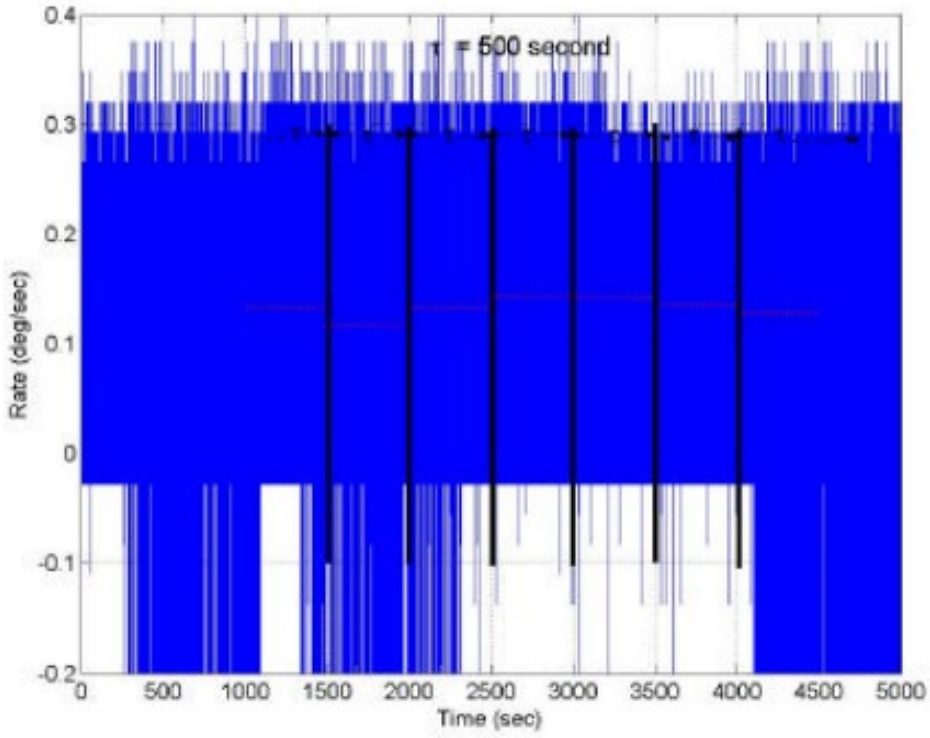


Figure A3: Sample data showing time intervals for tau = 500

A.2 Understanding Noise

Understanding the hardware of a CubeSat is only part of the requirement when working in this field. Care must be taken when dealing with the actual data that is received from the various attitude determination components. A huge part of collecting correct data is understanding the drawbacks of each sensor. When it comes to gyroscopes and accelerometers, bias and noise must be compensated for before the data becomes reliable. One method of this is the implementation of a Kalman filter[7]. Bias is defined as the averaged data collected over a long period during a static test. Noise is considered a short-term variation of the output or the standard deviation of

the output while the sensor is at rest[30]. In other words, it could be a slight error in how the sensor measures data that must be corrected for.

A well-known method of characterizing the noise and stability in sensors is the Allan Variance plot developed by Dr. David Allan[30]. This method plots the intrinsic noise of a system (y-axis) as a function of the averaging time (x-axis)[31].

It can also be described as representing the root-mean-square random-drift errors as a function of the averaging time[22]. The plot can be broken down into five parts- angle random walk, rate random walk, bias instability, quantization noise and rate ramp [2].

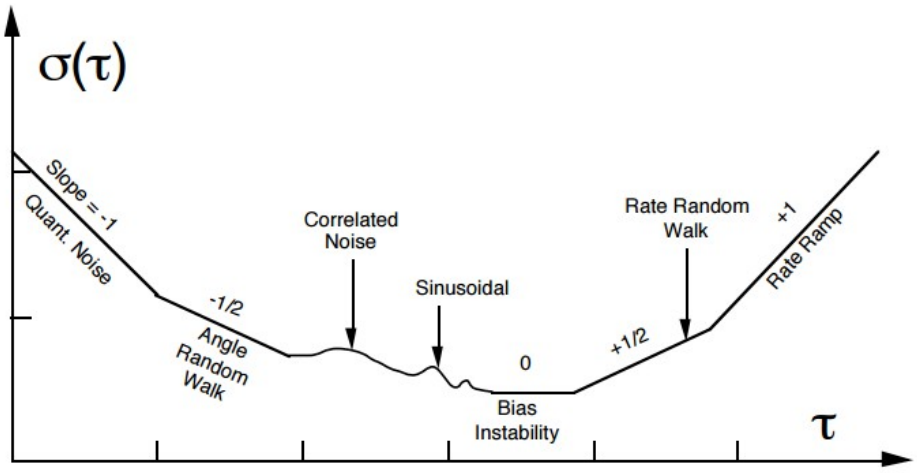


Figure A4: General shape of Allan Variance for a gyro sensor showing the various forms of noise

From the graph we see that intrinsic noise dominates for small τ while increasing τ will produce the five properties listed above. Angle random walk (ARW) is commonly used as a specification for rate sensors and in gyroscopes it is used to understand angle measurements[32].

Gyro sensors measure in deg/sec but can be integrated to measure the actual angle. ARW describes the 'average deviation or error that will occur when you integrate the signal[31].' When a sensor is at rest, one would expect the output data to be zero as well as the resulting integration. However, random noise will cause disruptions in the data and will result in a non-zero integration. Consequently, integration of faulty data will lead to a significant growth of error and cause a drift in the position and velocity outputs[30]. An Allan Variance analysis is useful when determining why types of noise is most active in a device. It is most useful as a starting point to determine what areas of the device should be further investigated.

A.3 Calculating Angular Random Walk and Bias Stability

Looking at figure A4, we see that bias instability is calculated when the plot has a slope of zero, and that ARW is calculated at a slope of -½ . The process is straight-forward for calculating these values. Looking at the left image of figure 3, we choose a point that is along a slope of 0 for bias instability. This point has coordinates of (8, 0.00157) where the y-value is called sigma, σ. From the IEEE paper on bias measurements[31], the bias instability can be calculated from the following equation,

$$B = \frac{\sigma}{0.6642} * 3600 \qquad [A1]$$

which will give units of *deg/hr.* ARW is calculated when the slope is -½ so we choose the coordinate (0.25, 0.002876) for this calculation:

$$ARW = \sqrt{\tau * 60} \qquad [A2]$$

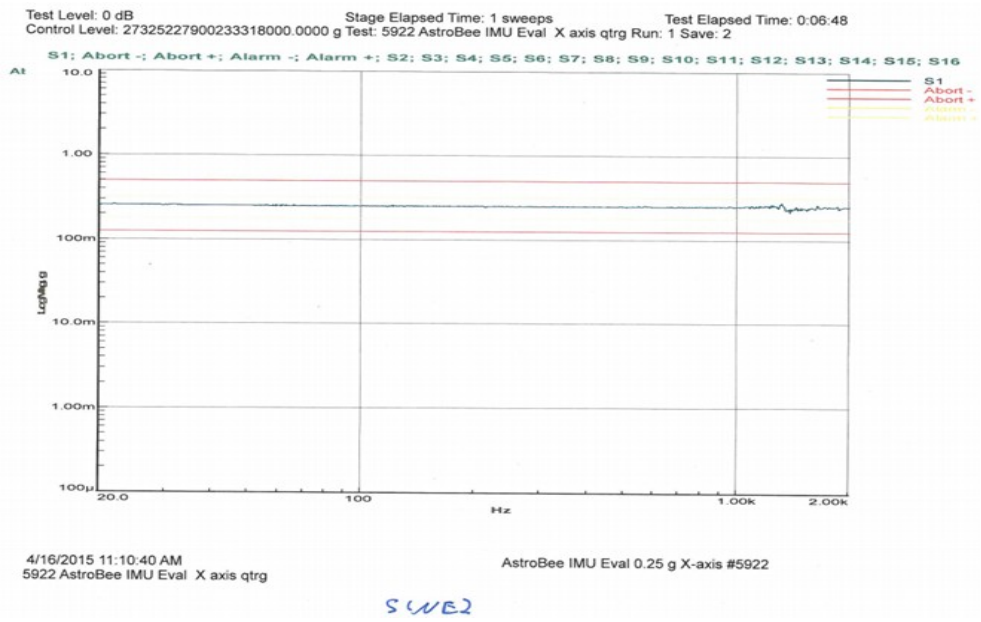
where τ is the averaging time, which is simply the x-coordinate.

Appendix B - Vibration Table PSD Plots

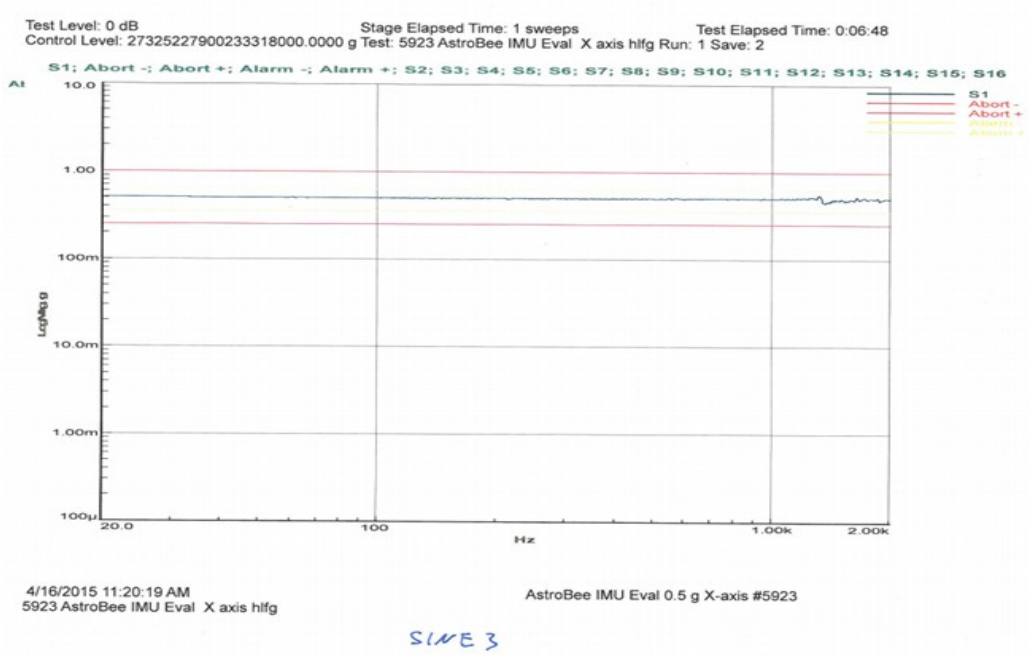
The following plots are an extension of the plots shown in Section 5.3 These are the remaining vibration table PSD outputs for the second and third test of the Sine Sweep and Random Test. They are similar in nature with a slight fluctuation towards the higher frequency range.

B.1 Sine Sweep

Sine Sweep Test 2 of amplitude ¼ g_{rms}

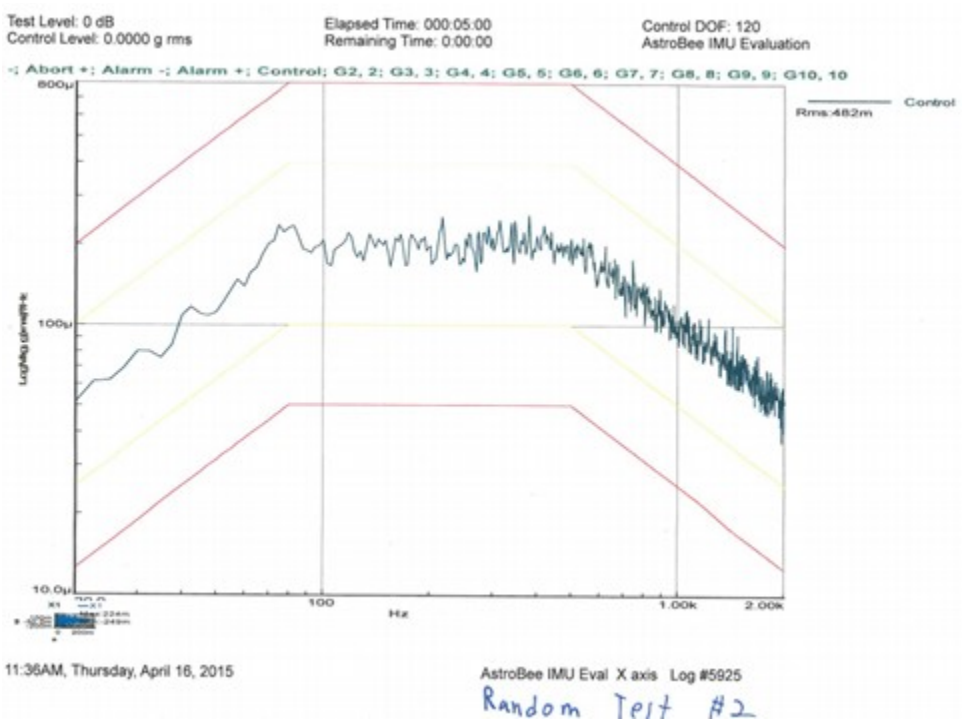


Sine Sweep Test 3 of amplitude ½ g_{rms}

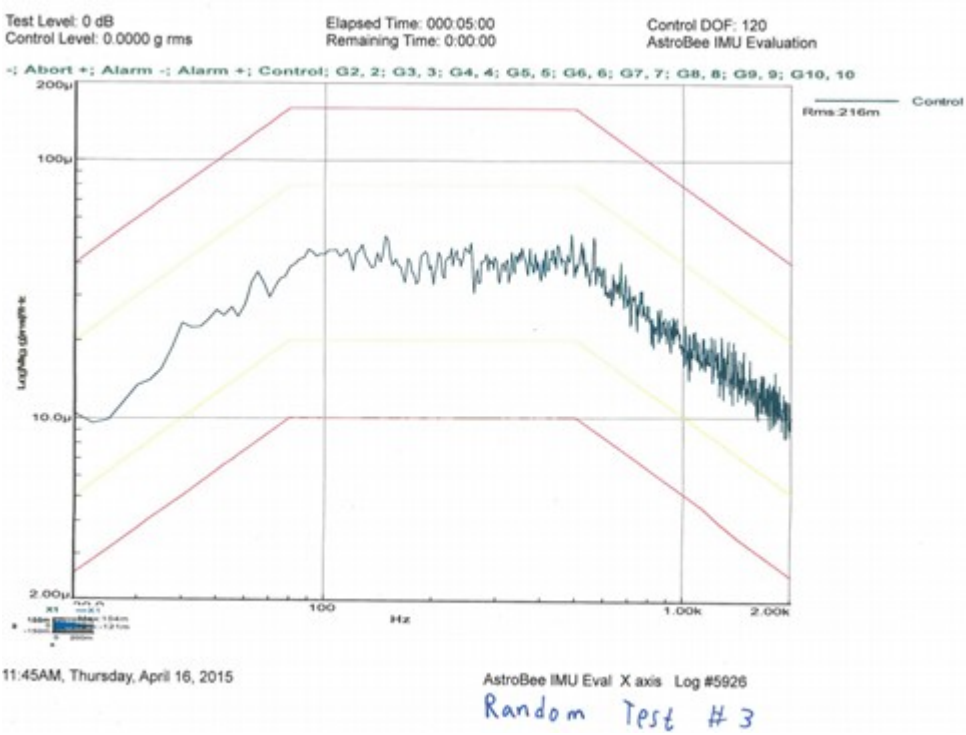


B.1 Random Test

Random test of amplitude 0.48 g_{rms}



Random test of amplitude 0.21 g_{rms}



Appendix C – Additional Vibration Plots

C.1 Running Standard Deviation Plots

The following plots show the running standard deviation for the y-axis and z-axis of both the Sine Sweep and Random vibration. The main paper focuses on the x-axis but the other axes are shown here for comparison purposes.

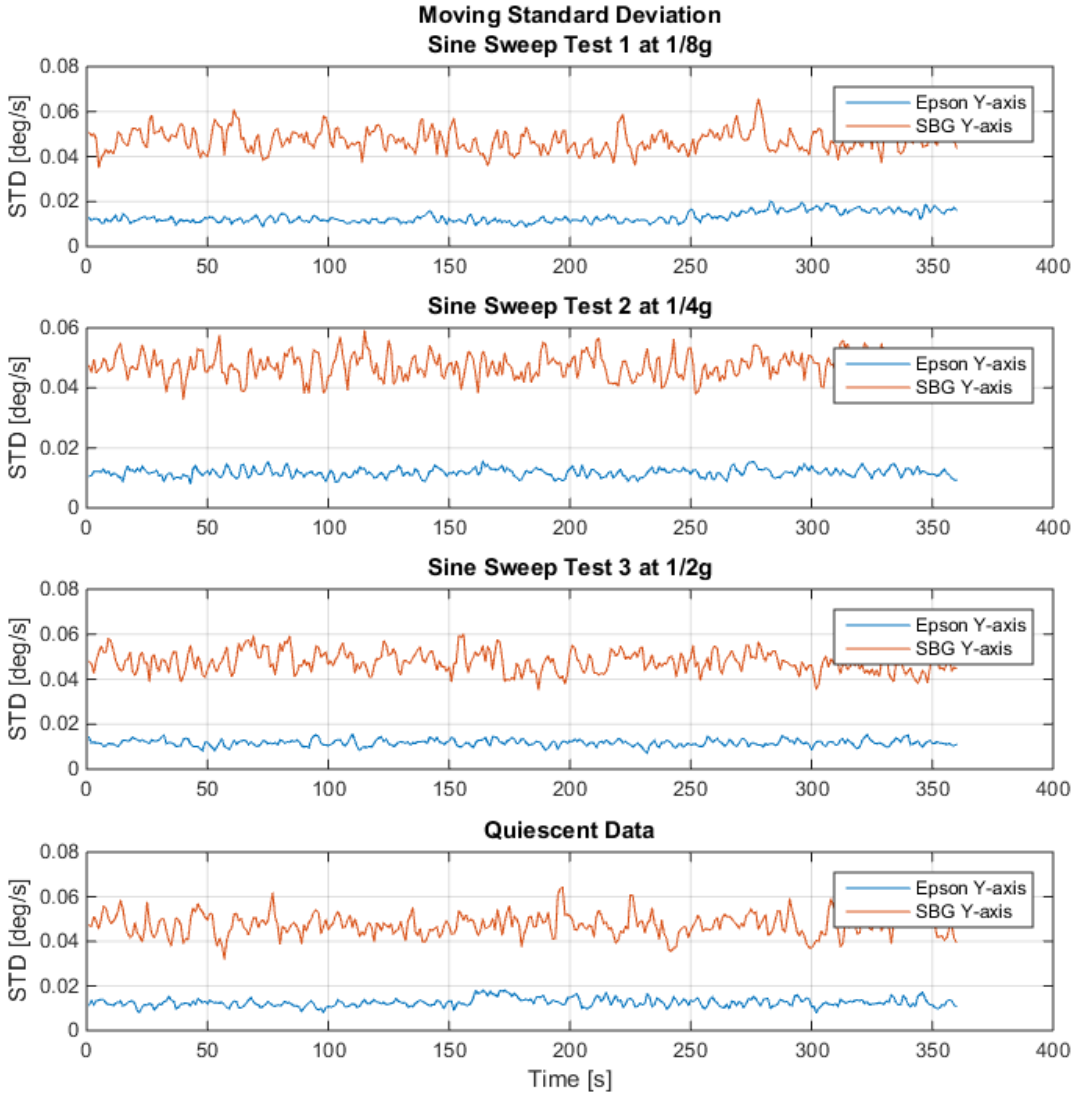


Figure C1: Running Standard Deviation plot shown for both Epson and SBG y-axis for each of the three Sine Sweep tests.

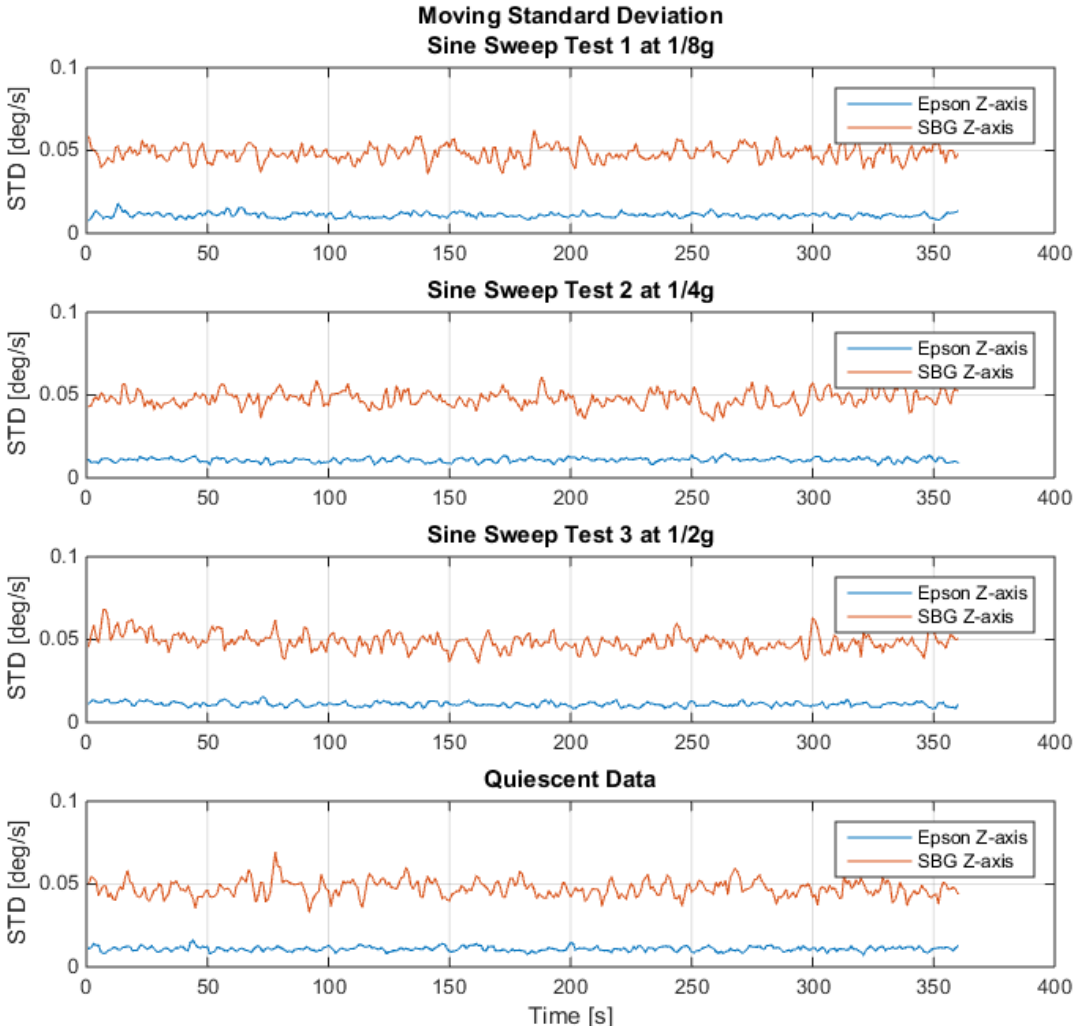


Figure C2: Running Standard Deviation plot shown for both Epson and SBG z-axis for each of the three Sine Sweep tests.

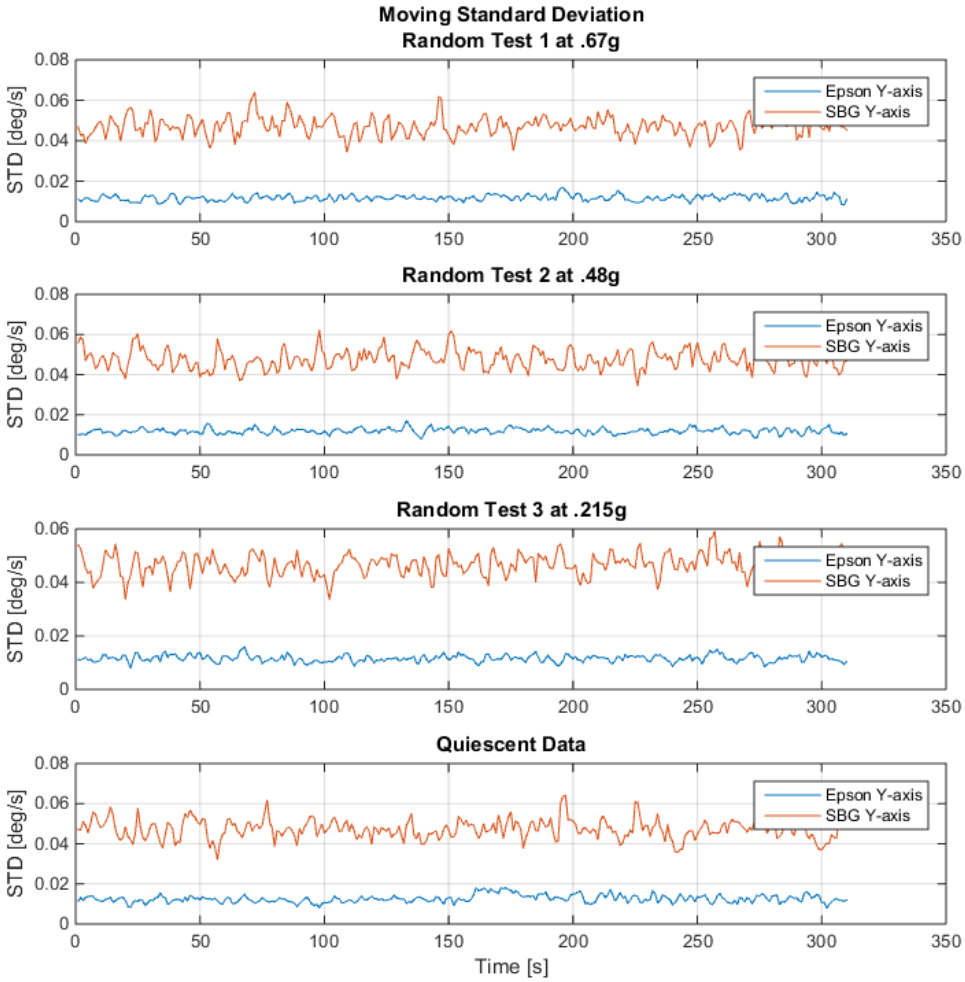


Figure C3: Running Standard Deviation plot shown for both Epson and SBG y-axis for each of the three Random vibration tests.

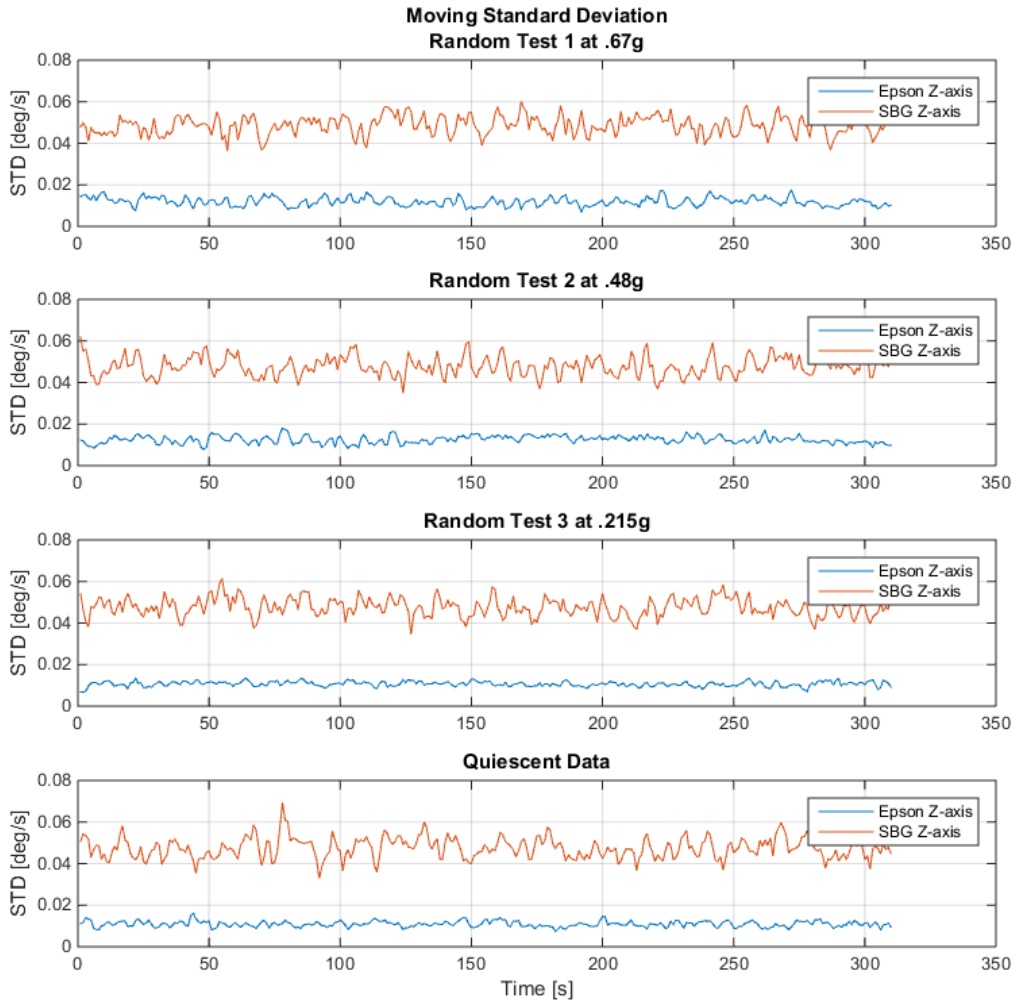


Figure C4: Running Standard Deviation plot shown for both Epson and SBG z-axis for each of the three Random vibration tests.

C.2 Integrated Cumulative Sum Plots

Shown below are additional plots for the integrated cumulative sum data. Each plot contains data for the subsequent tests for the Sine sweep and Random vibration showing each axis and the quiescent data. This quiescent data is the same throughout and shown for comparison.

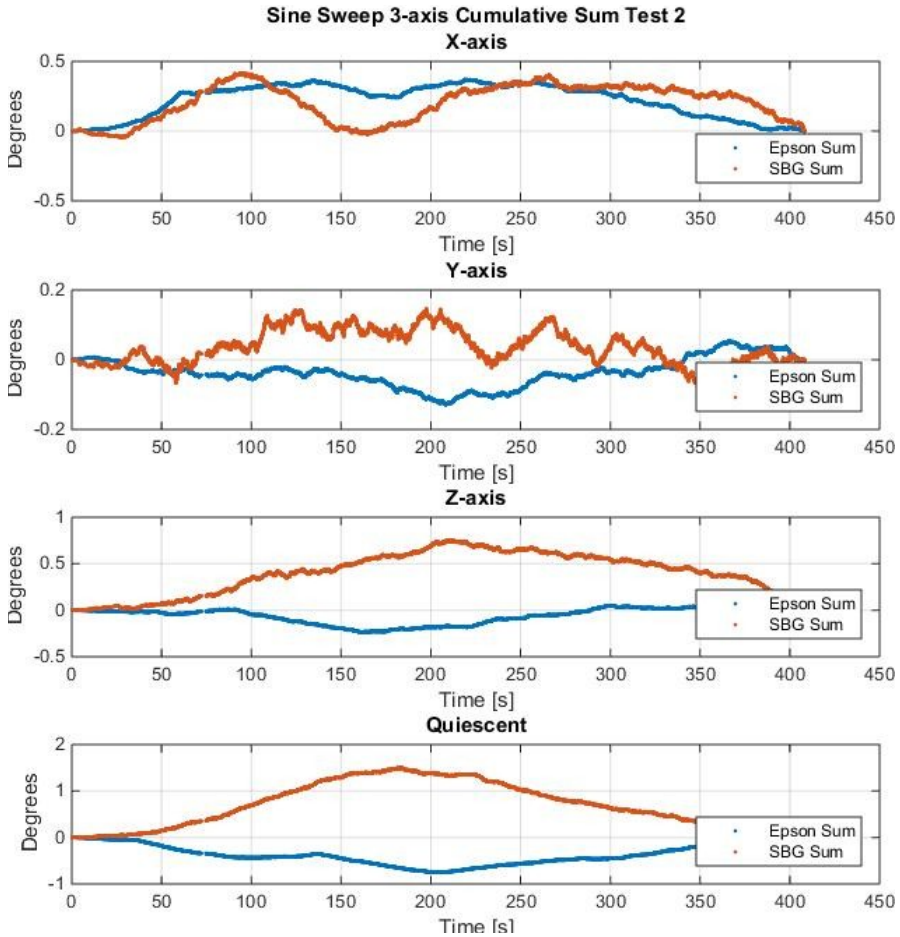


Figure C5: Integrated cumulative sum plots shown for both gyroscopes. The four plots here contain data from all three axes for the 2nd Sine sweep test along with the quiescent data for comparison.

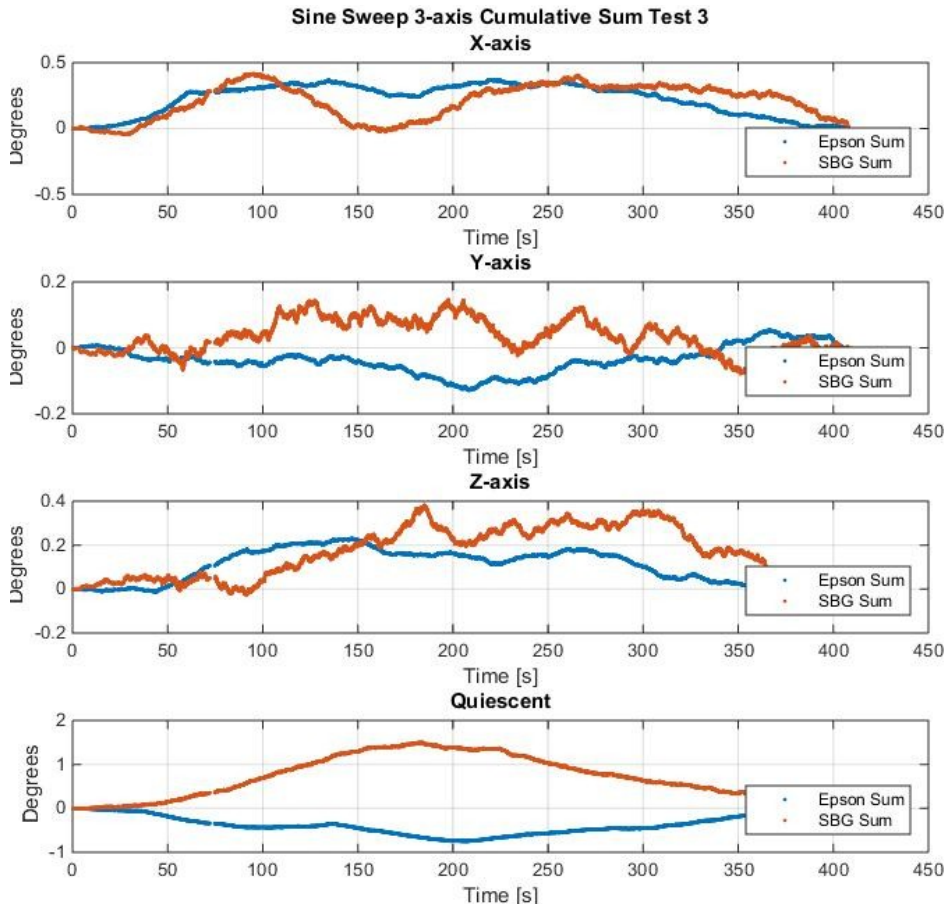


Figure C6: Integrated cumulative sum plots shown for both gyroscopes. The four plots here contain data from all three axes for the 3rd Sine sweep test along with the quiescent data for comparison.

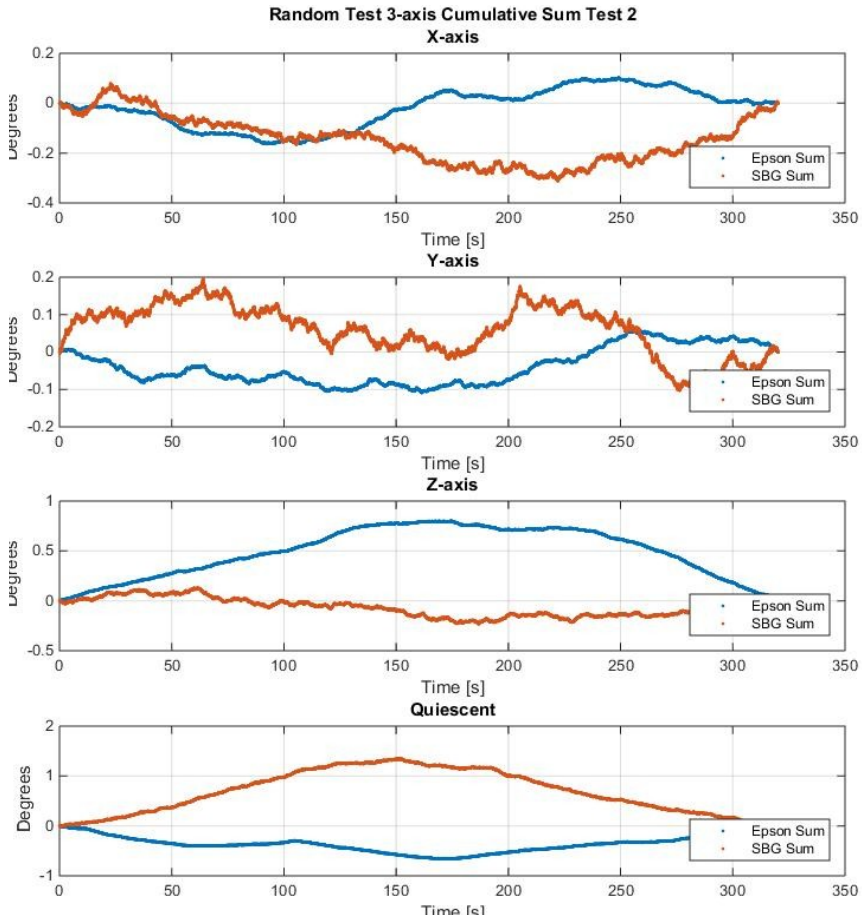


Figure C7: Integrated cumulative sum plots shown for both gyroscopes. The four plots here contain data from all three axes for the 2nd Random vibration test along with the quiescent data for comparison.

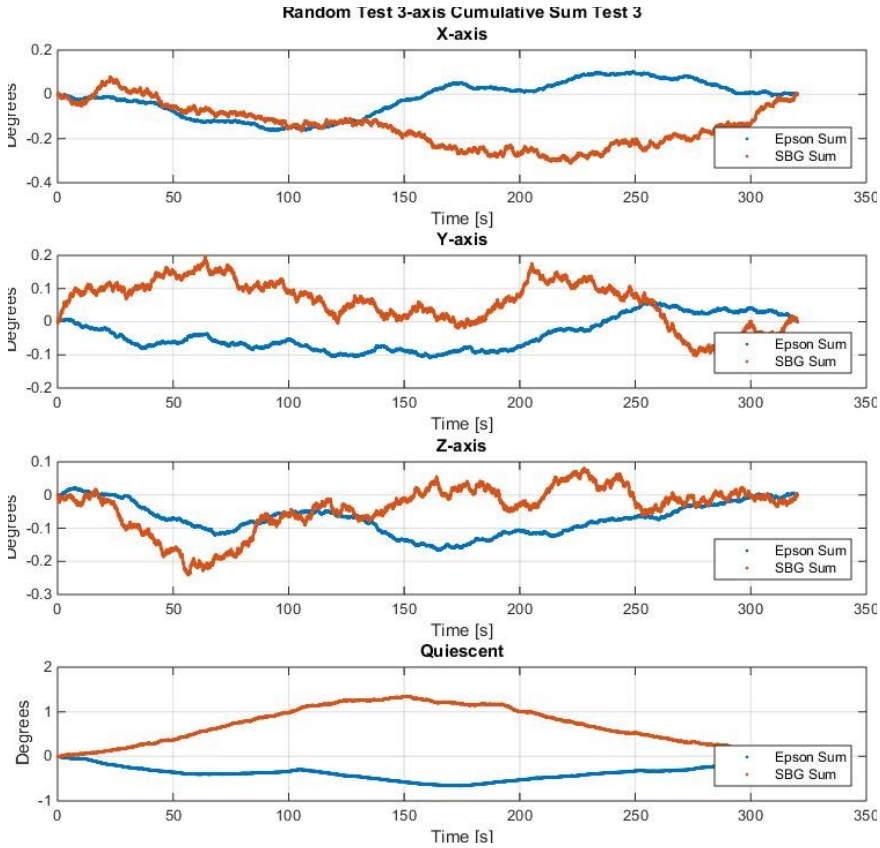


Figure C8: Integrated cumulative sum plots shown for both gyroscopes. The four plots here contain data from all three axes for the 3rd Random vibration test along with the quiescent data for comparison.

Appendix D – Extra MATLAB Code

D.1 Master File

In order to make the data analysis as streamlined as possible, a simple master script was created, called 'VibeTestMaster.m.' This script allows any CSV file generated by the testbed to be thoroughly analyzed while generating the required tables and graphs required for full investigation.

```
%% Start by calling Sine Sweep data
clear all, clc, close all
raw = csvread('sineSweep.csv');
flag = 'sine'
%% Separate raw test data into each Gyro axis for Epson and SBG
[Gyro,t] = GyroRaw(raw,flag);

%% Plot raw test data for either 'Sine' or 'Random' data
GyroRawPlots(Gyro,t,flag)

%% Calculate Averages and Standard Deviation
GyroAvgStd(Gyro,flag)

%% Plot Running Standard Deviation
GyroMoveStd(Gyro,flag)

%% Calculate Cumulative Sum
VibeCumSum(Gyro,t)

%% Plot in Frequency Domain
PlotFreq(Gyro,t)
```

All the user has to do here is load in the CSV file as indicated in the third line and then rename the 'Flag' variable to designate what test was used (sine sweep or random). Also contained in this script is the same setup but for the accelerometer data.

D.2 GyroRaw

The GyroRaw function takes in the raw CSV data and a string to designate which vibration test was used. The first few lines just setup basic variables like max size and the time vector. The 'if' statements are used in conjunction with the string parameter. Since the sine sweep and random tests were varying in length, different vectors needed to be setup depending on which parameter the user called for. The code after is where each IMU axis and test is separated into its own variable. Recall that both vibration tests each had 3 separate tests. The final line takes all of the variables and combines it into one cell variable.

```
function [RawData,time] = GyroRaw(raw,st)
%Accel function takes in the raw testbed data (Epson, Analog and SBG)
and
%outputs relevent Gyroscope data for the Epson and SBG.

n = max(size(raw));
end_pt = n;
time = raw(1:end_pt,1)/1000; %ms -> s
% Separate the raw data for Epson and SBG
% Both of these sets will be in deg/s
% X axis values

if strcmp(st,'sine')
      Sine1st = raw(3623:10855,:);
      Sine2nd = raw(16307:23539,:);
      Sine3rd = raw(26609:33841,:);
elseif strcmp(st,'random')
      Sine1st = raw(2356:8589,:);
      Sine2nd = raw(13645:19878,:);
      Sine3rd = raw(24163:30396,:);
end

Ep1stX = Sine1st(:,3); % X-axis [deg/s]
SBG1stX = Sine1st(:,33)*180/pi; % [deg/s];% X-axis


Ep2ndX = Sine2nd(:,3); % X-axis [deg/s]
SBG2ndX = Sine2nd(:,33)*180/pi; % [deg/s];% X-axis


Ep3rdX = Sine3rd(:,3); % X-axis [deg/s]
SBG3rdX = Sine3rd(:,33)*180/pi; % [deg/s];% X-axis


% Y axis values
Ep1stY = Sine1st(:,4); % Y-axis [deg/s]
```

```
SBG1stY = Sine1st(:,34)*180/pi; % [deg/s];% Y-axis

Ep2ndY = Sine2nd(:,4); % X-axis [deg/s]
SBG2ndY = Sine2nd(:,34)*180/pi; % [deg/s];% X-axis

Ep3rdY = Sine3rd(:,4); % X-axis [deg/s]
SBG3rdY = Sine3rd(:,34)*180/pi; % [deg/s];% X-axis

% Z axis values

Ep1stZ = Sine1st(:,5); % Y-axis [deg/s]
SBG1stZ = Sine1st(:,35)*180/pi; % [deg/s];% Y-axis

Ep2ndZ = Sine2nd(:,5); % X-axis [deg/s]
SBG2ndZ = Sine2nd(:,35)*180/pi; % [deg/s];% X-axis

Ep3rdZ = Sine3rd(:,5); % X-axis [deg/s]
SBG3rdZ = Sine3rd(:,35)*180/pi; % [deg/s];% X-axis

RawData = {[Ep1stX] [SBG1stX] [Ep2ndX] [SBG2ndX] [Ep3rdX] [SBG3rdX]
[Ep1stY] [SBG1stY]...
[Ep2ndY] [SBG2ndY] [Ep3rdY] [SBG3rdY] [Ep1stZ] [SBG1stZ] [Ep2ndZ]
[SBG2ndZ] [Ep3rdZ] [SBG3rdZ]};

end
```

D.3 GyroRawPlots

GyroRawPlots plots the raw data for the gyroscopes. This function takes in the cell variable created previously as well as the time vector and the string to designate random or sinusoidal vibration. The cell is then decomposed back into its original matrix format and separated back into variables so they can be analyzed. The function then determines if it is the sine or random vibration and plots the graphs accordingly.

```
function GyroRawPlots(RawDataCell,time,st)

%GyroRawPlots takes in the raw data from GyroRaw.m and plots the raw
plots from the Sine or Random tests in all 3 axes

load('StaticRaw.mat')
%Convert the Cell data from RawDataCell into a matrix
RawData = cell2mat(RawDataCell);

% X axis values
Ep1stX = RawData(:,1); % X-axis [deg/s]
SBG1stX = RawData(:,2);
```

```
Ep2ndX = RawData(:,3); % X-axis [deg/s]
SBG2ndX = RawData(:,4);
Ep3rdX = RawData(:,5); % X-axis [deg/s]
SBG3rdX = RawData(:,6);
% Y axis values
Ep1stY = RawData(:,7); % Y-axis [deg/s]
SBG1stY = RawData(:,8);
Ep2ndY = RawData(:,9); % X-axis [deg/s]
SBG2ndY = RawData(:,10);
Ep3rdY = RawData(:,11); % X-axis [deg/s]
SBG3rdY = RawData(:,12);
% Z axis values
Ep1stZ = RawData(:,13); % Y-axis [deg/s]
SBG1stZ = RawData(:,14);
Ep2ndZ = RawData(:,15); % X-axis [deg/s]
SBG2ndZ = RawData(:,16);
Ep3rdZ = RawData(:,17); % X-axis [deg/s]
SBG3rdZ = RawData(:,18);

if strcmp(st,'sine')

    % Plot comparison graphs
    % Start with subplot of X Y Z for Sine Sweep 1st test
    figure, subplot(4,1,1)
    plot(time(1:length(SBG1stX)),SBG1stX,'r'), hold on
    plot(time(1:length(Ep1stX)),Ep1stX,'b'), legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope X-Axis Sine Sweep 1 (1/8g)')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

    subplot(4,1,2),
    plot(time(1:length(SBG1stY)),SBG1stY,'r'), hold on
    plot(time(1:length(Ep1stY)),Ep1stY,'b'), legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope Y-Axis Sine Sweep 1 (1/8g)')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

    subplot(4,1,3),
    plot(time(1:length(SBG1stY)),SBG1stZ,'r'), hold on
    plot(time(1:length(Ep1stY)),Ep1stZ,'b'), legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope Z-Axis Sine Sweep 1 (1/8g)')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

    subplot(4,1,4),
    plot(time(1:length(SBG1stY)),SBGgyroX(1:length(SBG1stY)),'r'), hold
on
    plot(time(1:length(Ep1stY)),gyroX(1:length(SBG1stY)),'b'),
legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope Quiescent Data')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

end

if strcmp(st,'random')

    % Plot comparison graphs
    % Start with subplot of X Y Z for Sine Sweep 1st test
    figure, subplot(4,1,1),
```

```
    plot(time(1:length(SBG1stX)),SBG1stX,'r'), hold on
    plot(time(1:length(Ep1stX)),Ep1stX,'b'), legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope X-Axis Random 1 0.67g')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

    subplot(4,1,2),
    plot(time(1:length(SBG1stY)),SBG1stY,'r'), hold on
    plot(time(1:length(Ep1stY)),Ep1stY,'b'), legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope Y-Axis Random 1 0.67g')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

    subplot(4,1,3),
    plot(time(1:length(SBG1stY)),SBG1stZ,'r'), hold on
    plot(time(1:length(Ep1stY)),Ep1stZ,'b'), legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope Z-Axis Random 1 0.67g')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

    subplot(4,1,4),
    plot(time(1:length(SBG1stY)),SBGgyroX(1:length(SBG1stY)),'r'), hold
on
    plot(time(1:length(Ep1stY)),gyroX(1:length(SBG1stY)),'b'),
legend('SBG','Epson')
    title('Raw Data Plot of Gyroscope Quiescent Data')
    xlabel('Time [s]'),ylabel('deg/s'), grid on

end
end
```

D.4 GyroAvgStd

GyroAvgStd uses the same method above by taking in the raw data from the cell variable and separating it back into the individual variable names (this process has been omitted here for brevity). Once the variables are separated, it is easy enough to use MATLAB's built in 'mean' and 'std' function to calculate the average and standard deviation of each data set. With these values calculated, two simple tables are setup shown in the code below, which just offer two different orientations of the same table.

The data is presented in the table (Figure C1) to show the relationship between each axis for all three vibration subtests. Similar to the previous functions, indicating 'sine' or 'random' will display the relative data for that test. To generate a table in MATLAB, vectors are created to act

as the rows and columns. Once these are set up, the function 'printmat' is used to display the table as a matrix with the title as a string parameter. Only the data for the first two tests are shown below as the code is mirrored for the remaining tables.

```
function GyroAvgStd(RawDataCell,st)
% This function will take in the raw data and output a table of the
% averages and standard deviations

...
% Compute averages for 3 axes of Epson for the 3 tests
Ep1avgX = mean(Ep1stX);
Ep2avgX = mean(Ep2ndX);

% Compute std for 3 axes of SBG for the 3 tests
SBG1stdX = std(SBG1stX);
SBG2stdX = std(SBG2ndX);

...

% Create output table
% Start with Epson 3 axis table for test 1
output = {'EpGyroX';'EpGyroY';'EpGyroZ'};
EpAvgX1 = [Ep1avgX; Ep1avgY; Ep1avgZ];
EpStdX1 = [Ep1stdX; Ep1stdY; Ep1stdZ];
T = [EpAvgX1 EpStdX1]';
if strcmp(st,'sine')
    printmat(T,'Epson 3 Axis Gyro Printout of Sine Test 1
(1/8g)','avg[deg/s] std','gyroX gyroY gyroZ')
elseif strcmp(st,'random')
    printmat(T,'Epson 3 Axis Gyro Printout of Random Test 1
(0.67g)','avg[deg/s] std','gyroX gyroY gyroZ')
end

rnames = {'GyroX';'GyroY';'GyroZ'};
T1 = table(EpAvgX1,EpStdX1,'RowNames',rnames)
writetable(T1,'AvgStd\Epson Test1.xlsx','WriteRowNames',true)


% Output SBG data
% Test 1
output = {'SBGGyroX';'SBGGyroY';'SBGGyroZ'};
SBGAvgX1 = [SBG1avgX; SBG1avgY; SBG1avgZ];
SBGStdX1 = [SBG1stdX; SBG1stdY; SBG1stdZ];
T = [SBGAvgX1 SBGStdX1]';
if strcmp(st,'sine')
    printmat(T,'SBG 3 Axis Gyro Printout of Sine Test 1
(1/8g)','avg[deg/s] std','gyroX gyroY gyroZ')
elseif strcmp(st,'random')
```

```
    printmat(T,'SBG 3 Axis Gyro Printout of Random Test 1
(0.67g)','avg[deg/s] std','gyroX gyroY gyroZ')
end

rnames = {'GyroX';'GyroY';'GyroZ'};
T1 = table(SBGAvgX1,SBGStdX1,'RowNames',rnames)
writetable(T1,'AvgStd\SBG Test1.xlsx','WriteRowNames',true)

% Start with Epson 3 axis table for test 2
output = {'EpGyroX';'EpGyroY';'EpGyroZ'};
EpAvgX2 = [Ep2avgX; Ep2avgY; Ep2avgZ];
EpStdX2 = [Ep2stdX; Ep2stdY; Ep2stdZ];
T = [EpAvgX2 EpStdX2]';
if strcmp(st,'sine')
    printmat(T,'Epson 3 Axis Gyro Printout of Sine Test 2
(1/4g)','avg[deg/s] std','gyroX gyroY gyroZ')
elseif strcmp(st,'random')
    printmat(T,'Epson 3 Axis Gyro Printout of Random Test 2
(0.48g)','avg[deg/s] std','gyroX gyroY gyroZ')
end

rnames = {'GyroX';'GyroY';'GyroZ'};
T1 = table(EpAvgX2,EpStdX2,'RowNames',rnames)
writetable(T1,'AvgStd\Epson Test2.xlsx','WriteRowNames',true)
```

```
SBG 3 Axis Gyro Printout of Sine Test 3 (1/2g) =
                    gyroX        gyroY        gyroZ
   avg[deg/s]      0.02222      0.11013      0.00832
          std      0.04801      0.04818      0.04921


T1 =

              SBGAvgX3      SBGStdX3
              _________     _________

    GyroX       0.02222       0.04801
    GyroY       0.11013      0.048178
    GyroZ     0.0083151      0.049211
```

Figure D1: Table output from GyroAvgStd function.

D.5 VibeCumSum

As explained in section 6.3, the integrated cumulative sum gives another method of measuring long-term bias in IMU devices. Below is the code used in the function'VibeCumSum.m.' The first process is to subtract out the mean value from the raw data.

```
Ep1stX = Ep1stX - mean(Ep1stX);
SBG1stX = SBG1stX - mean(SBG1stX);
```

Next, integration is accomplished by multiplying each data point by 'dt,' which in this case is 0.05s (50ms). This is completed in MATLAB by running through a for loop the length of the data. Only part of the code is shown.

```
for i = 1:length(Ep1stX)
    Ep1stX(i) = Ep1stX(i) * dt;
    Ep2ndX(i) = Ep2ndX(i) * dt;
    Ep3rdX(i) = Ep3rdX(i) * dt;
```

The next step is to calculate the cumulative sum of each variable using MATLAB's function 'cumsum.' These variables are then organized into vectors for better organization.

```
EpSum1 = [cumsum(Ep1stX) cumsum(Ep1stY) cumsum(Ep1stZ)];
EpSum2 = [cumsum(Ep2ndX) cumsum(Ep2ndY) cumsum(Ep2ndZ)];
EpSum3 = [cumsum(Ep3rdX) cumsum(Ep3rdY) cumsum(Ep3rdZ)];
```

With the cumulative sum measured, it is then important to check that the process makes sense by looking at the final value. Since the bias was subtracted out before this process, the total integration should result in zero.

```
% Find final cumulative sum values
SumFinX = [EpSum1(end,1) EpSum2(end,1) EpSum3(end,1) SBGSum1(end,1)...
           SBGSum2(end,1) SBGSum3(end,1)]';

SumFinY = [EpSum1(end,2) EpSum2(end,2) EpSum3(end,2) SBGSum1(end,2)...
           SBGSum2(end,2) SBGSum3(end,2)]';
```

With the check confirmed, the last step is to output the data in a table and plot the measurements.

```
%% Output table of final values of each sum
rnames = {'1';'2';'3';'4';'5';'6'};
T1 = table(SumFinX, SumFinY, SumFinZ,'RowNames',rnames)
writetable(T1,'CumSumFinal.xlsx','WriteRowNames',true)

%% Plot test 1 all 3 axes
figure, subplot(4,1,1)
plot(time(1:length(EpSum1(:,1))),EpSum1(:,1),'.'), hold on
plot(time(1:length(SBGSum1(:,1))),SBGSum1(:,1),'.')
title({'Sine Sweep 3-axis Cumulative Sum Test 1';'X-
axis'}),legend('Epson Sum','SBG Sum','Location','SouthEast')
xlabel('Time [s]'),ylabel('Degrees'),grid on
```

References

1 *Satellites*. Digital image.*Space Science & Technology*. N.p., Mar. 2004. Web. Nov. 2012. <https://www.courses.psu.edu/aersp/aersp055_r81/satellites/satellites.html>.

2 *Spherical Air Bearings*. Digital image.*Nelson Air Corp.*N.p., n.d. Web. <http://www.nelsonair.com/NA_prods_spherical.htm>.

3 Trusculescu, M., and Balan, M., "Nanosatellites: The Tool for Earth Observation and Near Earth Environment Monitoring,"*Earth Observation,*edited by R. Rustamov and S. Salahova, InTech, 2012, pp. 25-37.

4 Taisen Zhuang, Shashurin, A., and Keidar, M., "Micro-cathode thruster for cube satellite propulsion,"*Plasma Science (ICOPS), 2012 Abstracts IEEE International Conference on,*2012, pp. 5E-8-5E-8.

5 Richard, L., "A Three Axis Magnetometer For use in a Small Satellite,"*Applied Electronics, 2006. AE 2006. International Conference on,*2006, pp. 113-116.

6 Jung, H., and Psiaki, M.L., "Tests of Magnetometer/Sun-Sensor Orbit Determination Using Flight Data,"*Journal of Guidance, Control, and Dynamics,*Vol. 25, No. 3, 2002, pp. 582-590.

7 Oluwatosin, A., Hamam, Y., and Djouani, K., "Attitude control of a CubeSat in a Circular Orbit using Reaction Wheels,"*AFRICON, 2013,*2013, pp. 1-8.

8 *In Space*. Digital image.*CubeSat Kit*. N.p., 2013. Web. Nov. 2014. <http://www.cubesatkit.com/content/space.html>.

9 Kaslow, D., Soremekun, G., Hongman Kim, "Integrated model-based systems engineering (MBSE) applied to the Simulation of a CubeSat mission,"*Aerospace Conference, 2014 IEEE,*2014, pp. 1-14.

10 Andy Eatchel, Ronald Fevig, et al., “Development of a Baseline Telemetry System for the Cubesat Program at the University of Arizona”. Internatiofnal Telemetry Conference. (2002)

11 Selstrom, Jeremy J. *Thrust and Performance Study of Micro Pulsed Plasma Thrusters*. No. AFIT/GAE/ENY/10-M21. AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING AND MANAGEMENT, 2010.

12 *Pulsed Plasma Thrusters*. Digital image.*Nonequilibrium Gas and Plasma Dynamics Laboratory*. University of Michigan, Sept. 2014. Web. Nov. 2014.

13 Pencil, E., "Pulsed Plasma Thrusters," 2008. <http://www.nasa.gov/centers/glenn/about/fs23grc.html>.

14 Tanaka, M., Kisaki, S., Ikeda, T., "Research and development of pulsed plasma thruster systems for nano-satellites at Osaka Institute of Technology,"*Vehicle Power and Propulsion Conference (VPPC), 2012 IEEE,*2012, pp. 517-522.

15 Bromaghim, D.R., Spanjers, G.G., Spores, R.A., "A proposed on-orbit demonstration of an advanced pulsed-plasma thruster for small satellite applications,"*Aerospace Conference Proceedings, 2000 IEEE,*Vol. 4, 2000, pp. 75-84 vol.4.

16 Schwartz, J., "- Historical Review of Air-Bearing Spacecraft Simulators,"- *Journal of Guidance, Control, and Dynamics,*No. - 4, 2003, pp. - 513.

17 Sidi, Marcel J. *Spacecraft Dynamics and Control: A Practical Engineering Approach*. Cambridge: Cambridge UP, 1997. Print.

18 Agrawal, B., "- Air-bearing-based satellite attitude dynamics simulator for control software research and development,"*SPIE 4366, Technologies for Synthetic Environments,* Vol. 4366, No. -, 2001, pp. - 204.

19 Lawson, Barry. "Battery and Energy Technologies." *Satellite Technology Challenges*. N.p., n.d. Web. 16 Oct. 2015.

20 Gurrisi, Charles, et al. "Space Station Control Moment Gyroscope Lessons Learned." *Proceedings of the 40th Aerospace Mechanisms Symposium*. 2010.

21 H. J. Dougherty, K. L. Lebsock, and J. J. Rodden, "Attitude Stabilization of Synchronous Communications Satellites Employing Narrow-Beam Antennas," Journal of Spacecraft, Vol. 8, No.8, August 1971.

22 Mueller, Juergen, Richard Hofer, and John Ziemer. "Survey of propulsion technologies applicable to cubesats." (2010).

23 Wertz, J. and Larson, W., "Space Mission Analysis and Design", 3rd edition, Micocosm Press/ Springer, 2007.

24 Parker, J.M., Thunnissen, D., Blandino, J., and ganapathi, G., "The Preliminary Design and Status of a Hydrazine MilliNewton Thruster Development", AIAA Paper 99-2596, 35th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, June 20-24, 1999, Los Angeles, CA.

25 Scharlemann, C., et al. "Propulsion for Nanosatellites." *The 32nd International Electric Propulsion Conference*. 2011.

26 Bernstein, Jonathon. "An Overview of MEMS Inertial Sensing Technology."*SensorsOnline*. Sensormag, n.d. Web. 15 Dec. 2015. <http://www.sensorsmag.com/sensors/acceleration-vibration/an-overview-mems-inertial-sensing-technology-970>.

27 Douxchamps, Damien. "A Small List of IMU / INS / INU." *A Small List of IMU / INS / INU*. N.p., n.d. Web. 03 Aug. 2015.

28 Von Buren, John. "What Is Random Vibration Testing." *Dissolution Technologies* 5.4 (1998): 16-18. Web. 27 Oct. 2015.

29 "Power Spectral Density Function." *Power Spectral Density Function*. Cygnus Research International, n.d. Web. 24 Oct. 2015.

30 Allan, Sterling. "The Allan Variance." *Allan Variance*. N.p., 27 June 2012. <http://www.allanstime.com/AllanVariance/index.html. 16 Nov. 2015>.

31 Stockwell, Walter. *Bias Stability Measurement: Allan Variance* (n.d.): n. pag. <http://www.moog-crossbow.com/Literature/Application_Notes_Papers/Gyro_Bias_Stability_Measurement_using_Allan_Variance.pdf>. 15 Sept. 2015.

32 IEEE Std 952-1997, "Guide and Test Procedure for Single Axis Interferometric Fiber Optic Gyros," IEEE, 1997, p.63.