University California Santa Cruz
Electrical and Computer Engineering Deparment

Lab 3: 3D Sensor Calibration

Kyle Jeffrey

February 13, 2020

Contents

1. Summary from Lab Manual

"The purpose of this lab is to familiarize you with a modern 9DOF IMU sensor and the errors in the sensing elements. You will learn to communicate with the IMU, understand the error sources, and learn to apply modern linear algebra techniques to calibrate the errors out of the sensor. You will learn about bias drift, and the basics of gyro integration to get an angle from the raw rate sensor."

1.1 Background from Lab Manual

"With the advent of smart phones, the price of integrated low cost MEMs accelerometers, gyros, and magnetometers have dropped to an amazing degree. While these MEMs sensors are certainly not navigation (or even tactical) grade, they can be combined to give very good high bandwidth information about the orientation of the device in space (the attitude or pose of the device).
Like most MEMs sensor, however, these miniaturized devices have a plethora of error sources, including temperature drift, non-linearity, hysteresis, and some cross coupling between channels due to the manufacturing process. While more expensive sensors have much better spec's, these come at vastly higher cost (100K+),and often with severe restrictions on use due to ITAR[1] *restrictions.*
In this lab, we will ignore all of these error sources and focus merely on scale factor and bias errors for the sensors, and methods to measure and calibrate out these biases. We will also take a look at gyro bias drift, and use naïve methods to remove it and validate the effectiveness of such a method. Note that there is every reason to believe that in these low-cost sensors the bias and scale factor errors are a function of temperature (and most likely other factors as well). The Sparkfun sensor board has an on board temperature sensor for exactly this reason, however recalibrating the sensors over a temperature range is difficult, and requires extensive equipment unavailable in the lab.
Also note that the 9DOF IMU moniker from both Invensense and Sparkfun is quite misleading. The 9DOFcomes from the fact that there are 3 different 3-axis sensors on board (accelerometer, magnetometer, andgyro). Since each is measuring a different phenomenon, then there are 9 different measurements (10 withthe temperature) available at any point in time, thus 9DOF. However, these sensors are combine through theprocess of sensor fusion and attitude estimation to estimate (not measure) the inertial displacement (x,y,z)and attitude (roll, pitch, yaw) of the

[1]ITAR – International Traffic in Arms Regulations, passed in 1976, has severe restrictions on re-exportation of guidance devices. See:`https://en.wikipedia.org/wiki/International_Traffic_in_Arms_Regulations`

device. Thus in any real sense, they can be used to get a 6DOF solution(which is all that exists in inertial space). Further note that these specific devices are incapable of resolvingthe attitude to the accuracy required to remove gravity from the accelerometer output. That is to say thatthe residual error in gravity orientation will swamp the inertial acceleration of the device itself, and thusthe inertial position (double integration of the acceleration) will be completely unusable. Realistically, these devices are used to estimate attitude and not position. Technically, this is called an AHRS (Attitude HeadingReference System), not an IMU (Inertial Measurement Unit)."

2. Ellipsoid Calibration Using Simulated Data

From the lab manual:
In order to understand how we will be calibrating the 3-axis sensors, it is useful to understand the geometry of the problem. It is instructive to do this in 2D first. Begin with a hypothetical 2D sensor that measures the x and y components of some vector that is fixed. As you rotate the sensor, the sensing axes of the sensor measure the components of the fixed vector. If you plot the points from the sensor there will be a circle of radius equal to the length of the vector. However, the sensor is imperfect. There is a null shift on each of its axes, and the scale factors are different on both axes as well. Now if you rotate the sensor and plot out the points, you will get an ellipse that is centered on the biases, and whose semi-major and semi-minor axes lengths correspond to the scale factors. Note that if there were cross-sensitivity in the axes, it would be rotated as well. And lastly, there is wideband noise on the measurements.2 The task (finally) is given the noisy points on the ellipse; find the scale factor, cross-coupling, and null shifts such that you can reconstruct the circle centered at the origin with radius corresponding to your vector. And, of course, you don't get the complete ellipse, but only part of it. Unfortunately, you cannot do this with simple least squares. This is because the problem is not linear in the coefficients that you want. It is, however, linear in algebraic combinations of the coefficients that you want. So you do this in two steps: (1) Solve the LS for the funny quantities, and (2) Extract the desired coefficients from the solution to (1) algebraically. It sounds worse than it is.

Simulated data was provided for the class, and with some guidance from the TA's, two vectors were determined to represent the constants and the variables of the expanded ellipse equation.

$$\frac{x-x_0}{a^2} - \frac{y-y_0}{b^2} = R^2 \rightarrow x^2 = 2(x)(x_0) - (x_0^2) - (\frac{a^2}{b^2})(y^2) + 2(\frac{a^2}{b^2})(y_0)(y) - (y_0^2)(\frac{a^2}{b^2}) + (a^2)R^2$$

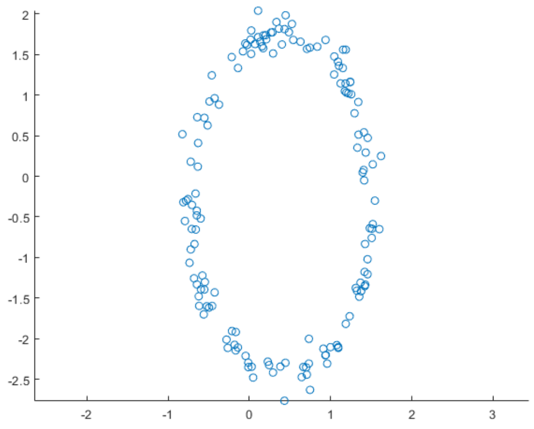
Split into Vectors:

$$x^2 = M * k$$

$$\text{M} = \begin{bmatrix} x & -y^2 & y & 1 \end{bmatrix}, \text{ k} = \begin{bmatrix} 2(x_0) \\ \frac{a^2}{b^2} \\ 2(\frac{a^2}{b^2})(y_0) \\ -(x_0^2) - (y_0^2)(\frac{a^2}{b^2}) + (a^2)R^2 \end{bmatrix}$$

Figure 2.1: Ellipsoid Equation Expansion

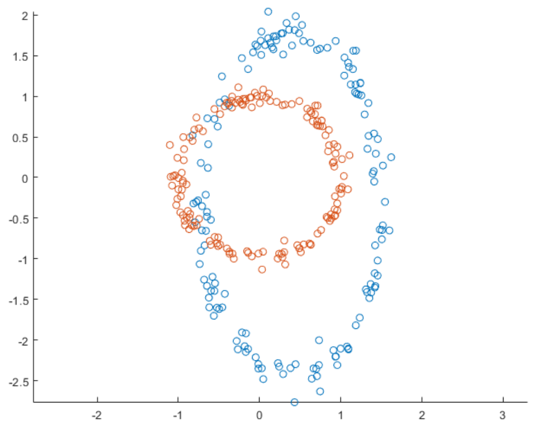
With these vector equations, the M matrix can be generated from the x measured and y measured values from the sample data. The sample data before any adjustments appears like this:



Notice the skew and null bias of the data along with some bandwith noise. We hope to adjust this to fit a normalized circle disribution. The k vector can be found using some matrix math such that $k = M\ x^2$ where '\' is a pseudo inverse operation i.e. $k = inv(M^T * M) * x^2 = M\backslash x^2$. With a k vector calculated, the values of each element are:

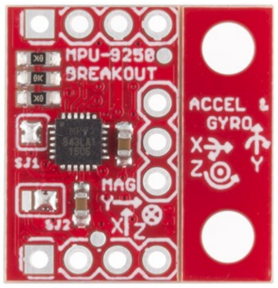
$x_0 = k(1)/2 = 0.3962$,
$y_0 = k(3)/(2 * k(2)) = -0.3388$,
$a = \sqrt{(k(4) + x_0^2 + (k(2) * y_0^2)/R^2)} = 1.1056$,
$b = sqrta^2/k(2) = 2.1445$

From this, the data can now be normalized to appear as such:



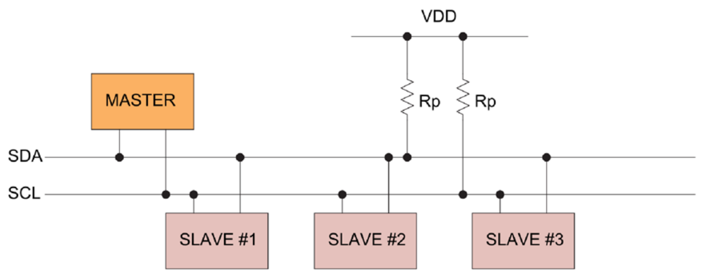
3. Naive Calibration of Magnetometer and Accelerometer

In this section of the lab, simpler forms of calibration were used. Experimental testing was done with the the MPU-9250 Motion Tracking device, shown below, to attempt simpler calibration. Firstly, the breakout board must be implemented using the I^2C communication protocol.



3.1 I2C Interface

I^2C is a synchronous serial communication protocol using two wires. One wire is SDA(The Data Line), and the other SCL(The Clock line). The Clock and data lines are driven by a "master" and wait to recieve response from the "slaves".

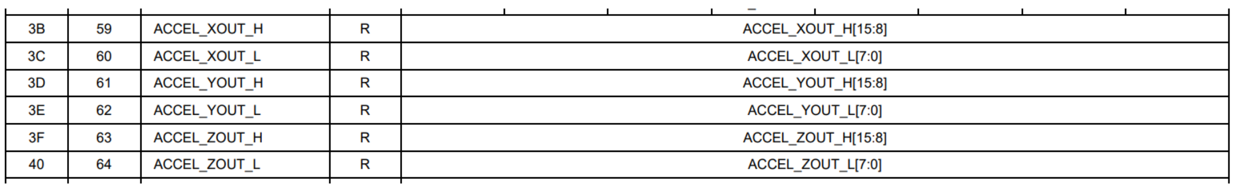


The datasheet for the MPU9250 outlines the protocol for communicating with the device which acts as a slave. The datasheet states in section *7.2*:

"I2C Interface I2C is a two-wire interface comprised of the signals serial data (SDA) and seri al clock (SCL). In general, the lines are open-drain and bi-directional. In a generalized I2C interface implementation, attached devices can be a

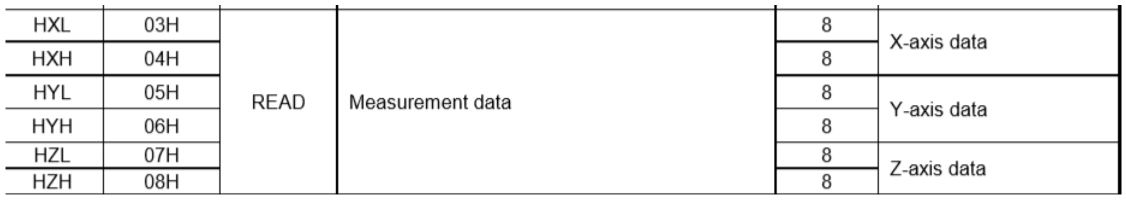
master or a slave. The master device puts the slave address on the bus, and the slave device with the matching address acknowledges the master. The MPU-9250 always operates as a slave device when communicating to the system processor, which thus acts as the master. SDA and SCL lines typically need pull-up resistors to VDD. The maxim um bus speed is 400 kHz. The slave address of the MPU-9250 is b110100X which is 7 bits long. The LSB bit of the 7 bit address is determined by the logic level on pin AD0. This allows two MPU-9250s to be connected to the same I2C bus. When used in this confi guration, the address of the one of the devices shoul d be b1101000 (pin A D0 is logic low) and the address of the other should be b1101001 (pin AD0 is logic high)."

So, the address of the slave is either 104 (b1101000) or 105(b1101001). This address is required to communicate with the sensor. The lab provides an I2C module for communication using this protocol. When writing or reading to a slave device from the master, an address for which slave device, and an address for what register on that slave device is required to write data. Here are the registers for the Accelerometer sensor on board, taken from the Register Map:



3B	59	ACCEL_XOUT_H	R	ACCEL_XOUT_H[15:8]
3C	60	ACCEL_XOUT_L	R	ACCEL_XOUT_L[7:0]
3D	61	ACCEL_YOUT_H	R	ACCEL_YOUT_H[15:8]
3E	62	ACCEL_YOUT_L	R	ACCEL_YOUT_L[7:0]
3F	63	ACCEL_ZOUT_H	R	ACCEL_ZOUT_H[15:8]
40	64	ACCEL_ZOUT_L	R	ACCEL_ZOUT_L[7:0]

Figure 3.1: Accelerometer Registers



HXL	03H	READ	Measurement data	8	X-axis data
HXH	04H			8	
HYL	05H			8	Y-axis data
HYH	06H			8	
HZL	07H			8	Z-axis data
HZH	08H			8	

Figure 3.2: Magnetometer Registers

Only a cursory understanding of I2C is required to use I2C as there is a provided MPU9250 library to read directly from the instruments. To read the accelerometer registers, simply use the functions provided by the MPU9250 library.
The datasheet for the MPU9250 states that pullup resistors are used to tie SDL and SDC to Vcc, otherwise the outputs could be floating for non zero values. Here's a schematic of the I2C communciation layout.

3.2 Data Capture

Between 100 - 1000 values for all of the Accelerometer axises, and all of the Magnetometer axises were captured to create a simple scaling and offset calibration for both instruments. The lab manual states there are two ways to calibrate using a "naive calibration":

3.2.1 Magnetometer

To measure and calibrate the magnetometer, the lab manual states:

"(1) Rotate the sensor in the horizontal plane until the x-axis peaks and the y-axis reads very close to zero (this should correspond to the point of "Horizontal Intensity" in the image below (or verify it yourself using simplev ector math). Mark that direction on your table, and use that to set your value. This will allow you to do both horizontal axes. Analogously you can use the "Vertical Comp" to do the z-axis. (2) The second way is to try to match the magnetic field vector itself, and tilt and swing the magnetometer until you max out the x-axis while having zero on both the y-and z-axes. You will need to use jigs and ramps to ensure that you can consistently reach this point,and then proceed as you did with the accelerometer."
The magnetometer is very sensitive to nearby electronics. When data was measured on a table in the lab vs. on a chair in the lab, there would be up to a 100 point difference in measurement and, the first technique for calibrating the sensor could not be done. The x and y axis of the magnetometer had significant biases when measured, shown here:

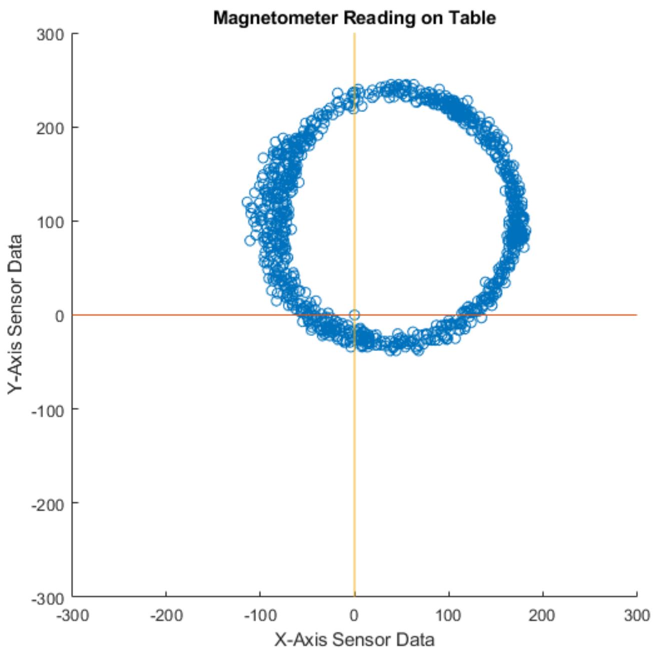


Figure 3.3: Magnetometer Data on Table

Notice the significant bias off zero of the magnetometer. Luckily, there isn't any skew to the shape of the readings, and is circular. The bias for axises is the offset from zero. For this set of readings, x offset = 40 and y offset = 110. Take note though that the readings can be very skewed by what surface the mag is near.
With this offset, consider max and min values of both axes to correspond to the horizontal intensity. The described "naive calibration technique can now be used. Find the point where y axis reads zero and the x - axis peaks. Experimentally, the max value for x when y was zero was 150. This corresponds to the horizontal intensity of 23,454 nT. Now instead of doing the same technique for the Z-axis against the Y-axis, the second naive calibration technique was executed i.e. finding where x and y are zero and where z is a peak value. A 3d dimensional plot of the mag shows the z to have a null bias of -550. Shown here:

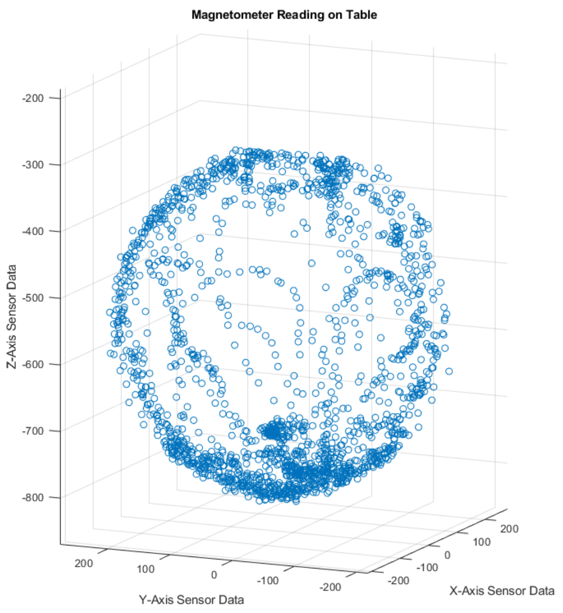


Figure 3.4: Magnetometer Data on Table

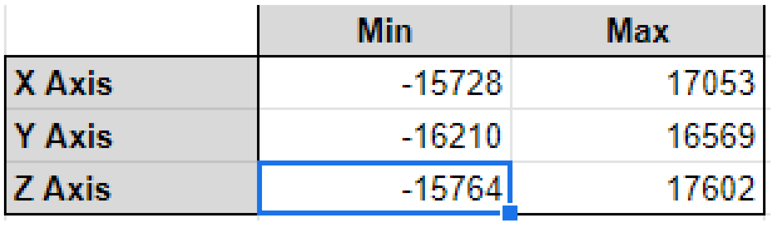
With all of the offsets accounted for, the magnetometer's max value was found for when two axes were zero and one axes was maxed out. Using matlab, *the max value was shown to be 250.*

3.2.2 Accelerometer

To calibrate the Accelerometer using simple techniques the lab manual states:

The accelerometer measures specific force, not acceleration. That is: $a = (a_s - g)$ ***If you put an accelerometer on a table, it measures*** $-g$***. If you measure it while it is falling, it will read zero until impact. You are going to use this to your advantage. First, calibrate the z-axis of the accelerometer by measuring the value with the sensor flat on the table, then flip the sensor upside down and measure again. These points should correspond to +/- 1g. Note, to minimize the effect of noise on your calibration, take at least 100-1000 samples at 50Hz in each orientation and average them to improve SNR. With those two points, you should be able to extract both scale factor and null shift for the z-axis of the accelerometer. Repeat the experiment for the x- and y-axes. Note that this means you will have to hold the accelerometer at 90 degrees to your bench and flip it over. Make yourselves a jig from foamcore or anything else to help you maintain the alignment. Record***

your values for null shifts and offsets for each axis. From this, the experimental data shown here was acquired:



	Min	Max
X Axis	-15728	17053
Y Axis	-16210	16569
Z Axis	-15764	17602

Figure 3.5: Accelerometer Data

Thus, the null shift bias for x, y, and z was 662, 178, and 919 respectively.

4. Gyro Bias and Bias Drift

The Gyro sensor on board the MPU9250 detects the amount of rotation in degrees/sec. Though, the sensor has a null bias and a drift. Here is a graph with the null biases, and a graph of the adjusted data. Both of these graphs present the raw data before any conversion to degrees per second.

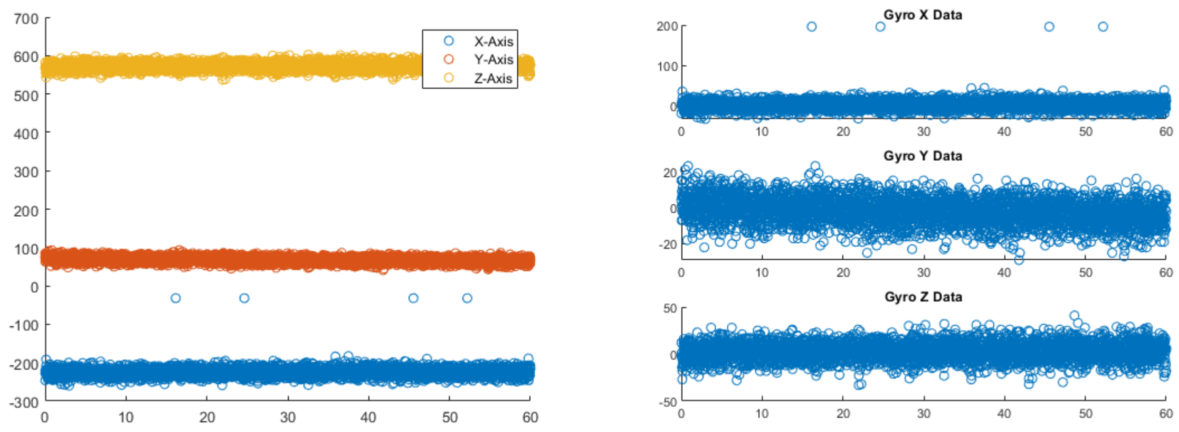
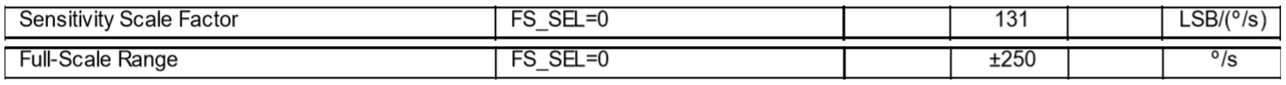


Figure 4.1: Gyro Bias Unadjusted and Adjusted Over Hour

In the next part, the data was converted to degrees per second. The datasheets provide the conversion from bits to engineering units. In this mode, the gyro has a max detection of 250 degrees per second.



Sensitivity Scale Factor	FS_SEL=0		131		LSB/(°/s)
Full-Scale Range	FS_SEL=0		±250		°/s

Figure 4.2: Gyro Specs

It is not apparent that there is any drift in the data based off visual inspection of the data, but to determine the drift of the gyro in absolute rotation over the hour, integrate the data as such: $\int_0^t d(t)dt$ where d(t) represets the value of the gyro at time t. Remember that the value of the gyro is in degrees per second. An integration can't be done on raw data so, a discrete integration was done to get values for the total rotation of the Gyro axes over the hour. Call D(t) the function of absolute rotation in degrees at time t in seconds. By inspection, it's apparent that d(t) is a derivative of D(t). So $D(t) = \Sigma_{k=0}^{t} d(k)\Delta t$. The data was taken with a polling rate of 1Hz so $\Delta t = 1$.

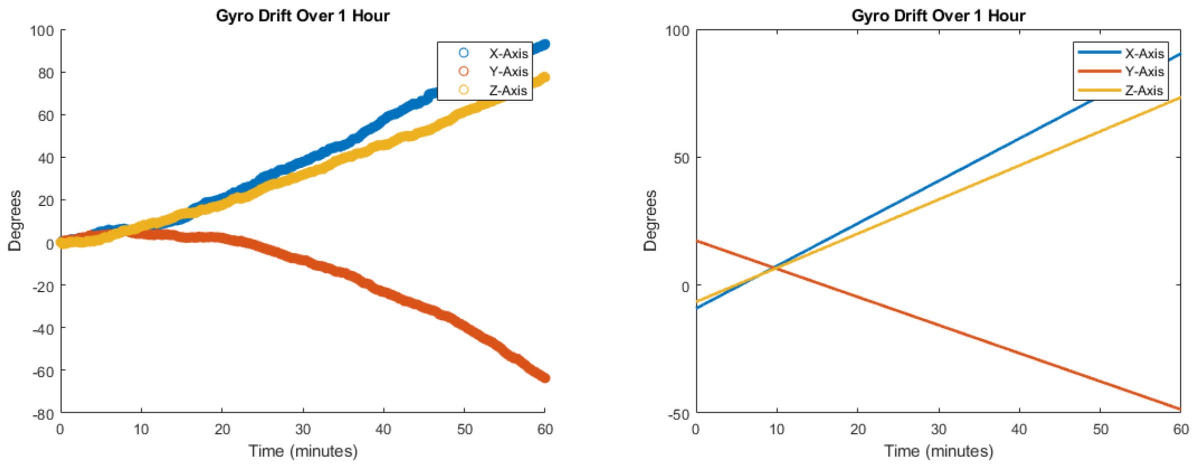


Figure 4.3: Gyro Drift Over Hour

The X, Y-axis drift between 80-90 degrees per hour, while the Y-axis drifts about 70 degrees per hour. The Y-axis does seem to drift at more of a polynomial rate based on this data set but, to reduce complexity of callibration, a first order best fit line simulated the data. Some more investigation might be required if more time was allotted.

5. Gyro Scale Factor via Integration

The lab manual gives an introduction into finding the scale factor of the gyro:

"The proper way to get scale factors for a gyro is to use something called a "rate table" which will spin the sensor at a constant rate on each axis and still allow you to get the data out. They are expensive pieces of equipment and we don't have one.5 Instead, we are going to calibrate the scale factor on the gyros by integrating the rotation rate through a constant set of angles. Begin by setting up your sensor and taking 10 seconds of data to establish your biases on each axis. Next, set up a set of stops on your bench that you can rotate through a defined angle (180 degrees works well). Rotate the sensor so that each axis (sequentially, not simultaneously) rotates through the angle, and then back. Integrate the gyro rates (subtracting out your bias from each raw measurement) and see how close to the correct angle you got. Adjust your scale factors until you get it. Experiment with doing this at slower and faster speeds. See if your scale factors change with speed."

A null bias reading was taken over a 10 second period to get these new null offsets
X-Axis = 46
Y-Axis = 97
Z-Axis = -564

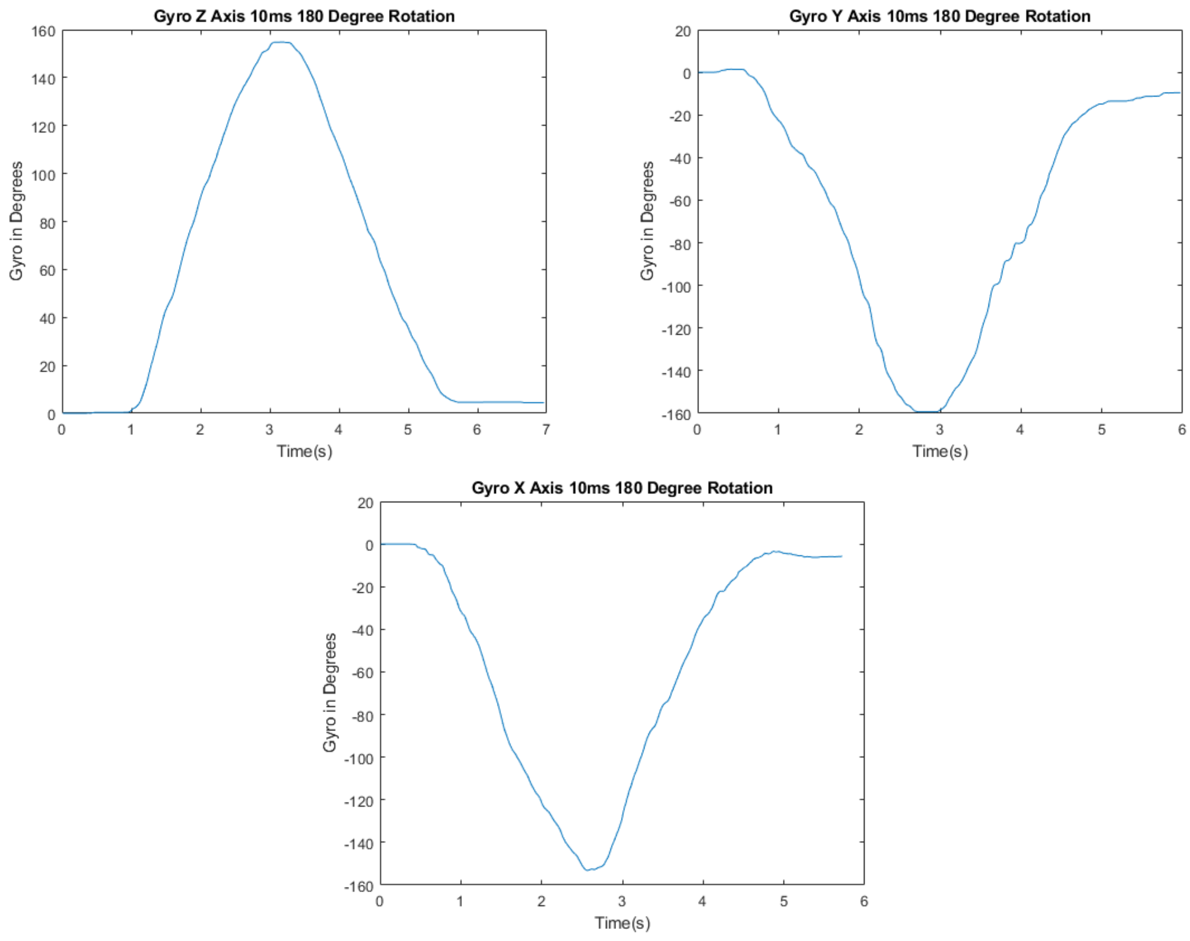


Figure 5.1: Gyro Integration of Rotation

The gyro was then rotated about several axes and integrated to get an absolute angle of rotation over a period of time. Each axis was rotated 180 degrees and then rotated back to the origin. Experimentally, the data was cleaner if the polling rate was set higher for the data output and the gyro was rotated quickly.

Notice all of the axes fall 20 degrees short of the 180 degree rotation. This means the scale is slightly off, but, the null biases appear effective as each axis returns to it's zero point of reference. **The scale was experimentally found to be 113 LSB / degree vs. the datasheets 131 LSB / degree**

6. Tumble Test Simulation

For this section, the lab provides matlab code to generate simulated Magnetometer and Accelerometer data. The data is going to be calibrated using the naive technique of visual inspection, and a provided library for Least Squares calibration.

6.1 Naive Calibration

The random tumble data generated these two graphs for Accelerometer and Magnetometer:

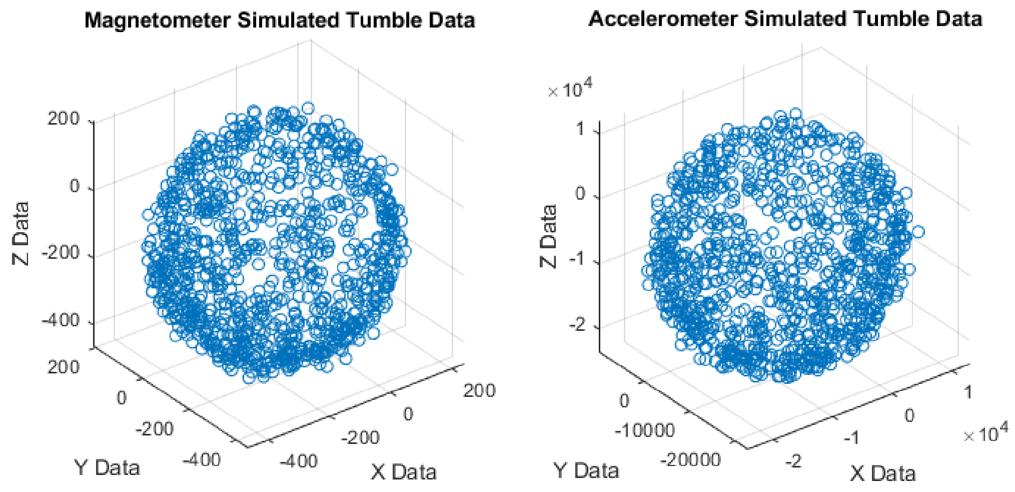
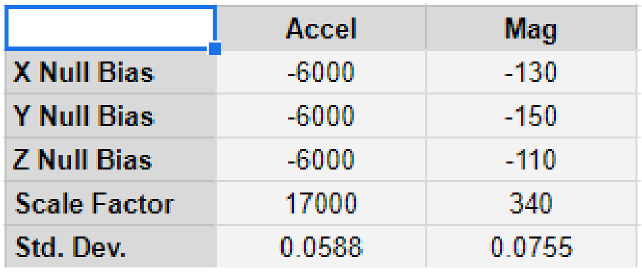


Figure 6.1: Accelerometer and Magnetometer Tumbler Data

Using the naive calibration techniques from the previous sections, these values were found:



	Accel	Mag
X Null Bias	-6000	-130
Y Null Bias	-6000	-150
Z Null Bias	-6000	-110
Scale Factor	17000	340
Std. Dev.	0.0588	0.0755

Figure 6.2: Magnetometer Histogram

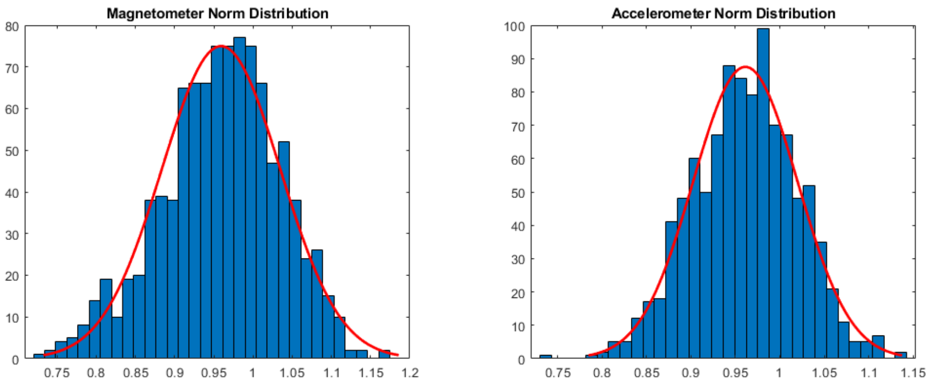


Figure 6.3: Magnetometer Histogram

z

6.2 Least Squares Calibration

The objective now is to forget about naive calibration and use the lab's provided least squares matlab function to calibrate the tumble data. The raw data is first converted to engineering units from the datasheet specs. The Accelerometer has a 16,384 LSB/g scale and the Magnetometer has a .15 uT/LSB scale for 16/bit mode. This technique was only done on the magnetometer in this part.

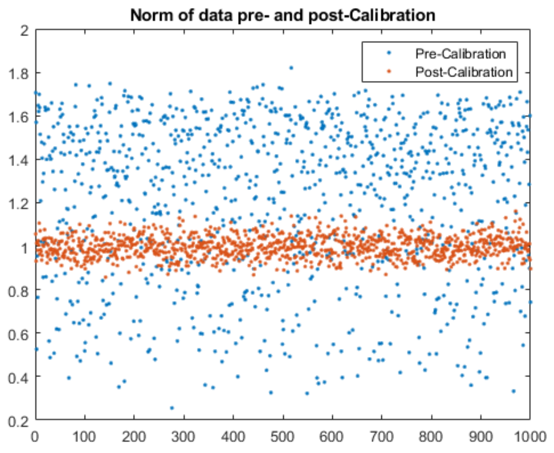


Figure 6.4: Least Squares Magnetometer Calibration

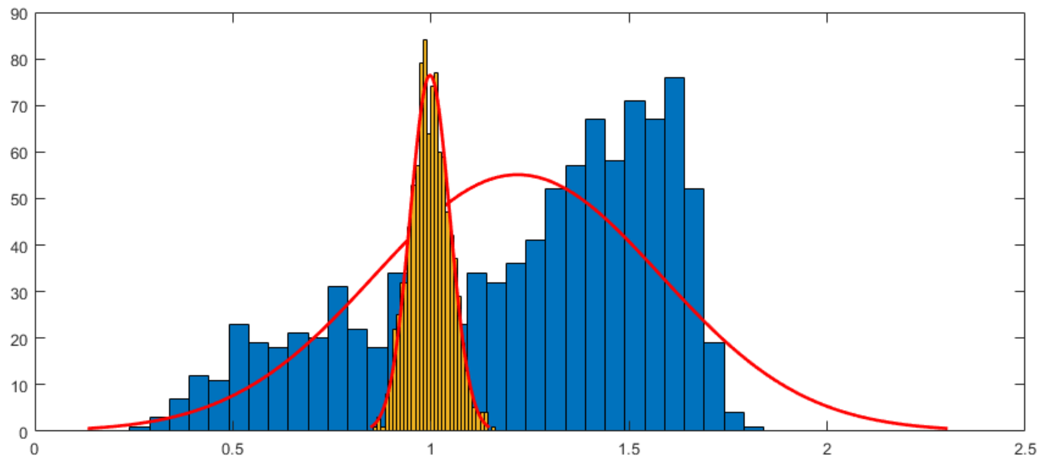


Figure 6.5: Least Squares Magnetometer Histogram

The standard deviation for the adjusted Magnetometer norm data was 0.0501.

7. Tumble Test for Accelerometer and Magnetometer

Now that multiple techniques of calibration have been explored, 3D ellipsoid sensor data was taken for both the Magnetometer and Accelerometer. The data was scaled to engineering units and the vector magnitude distribution is analyzed here:

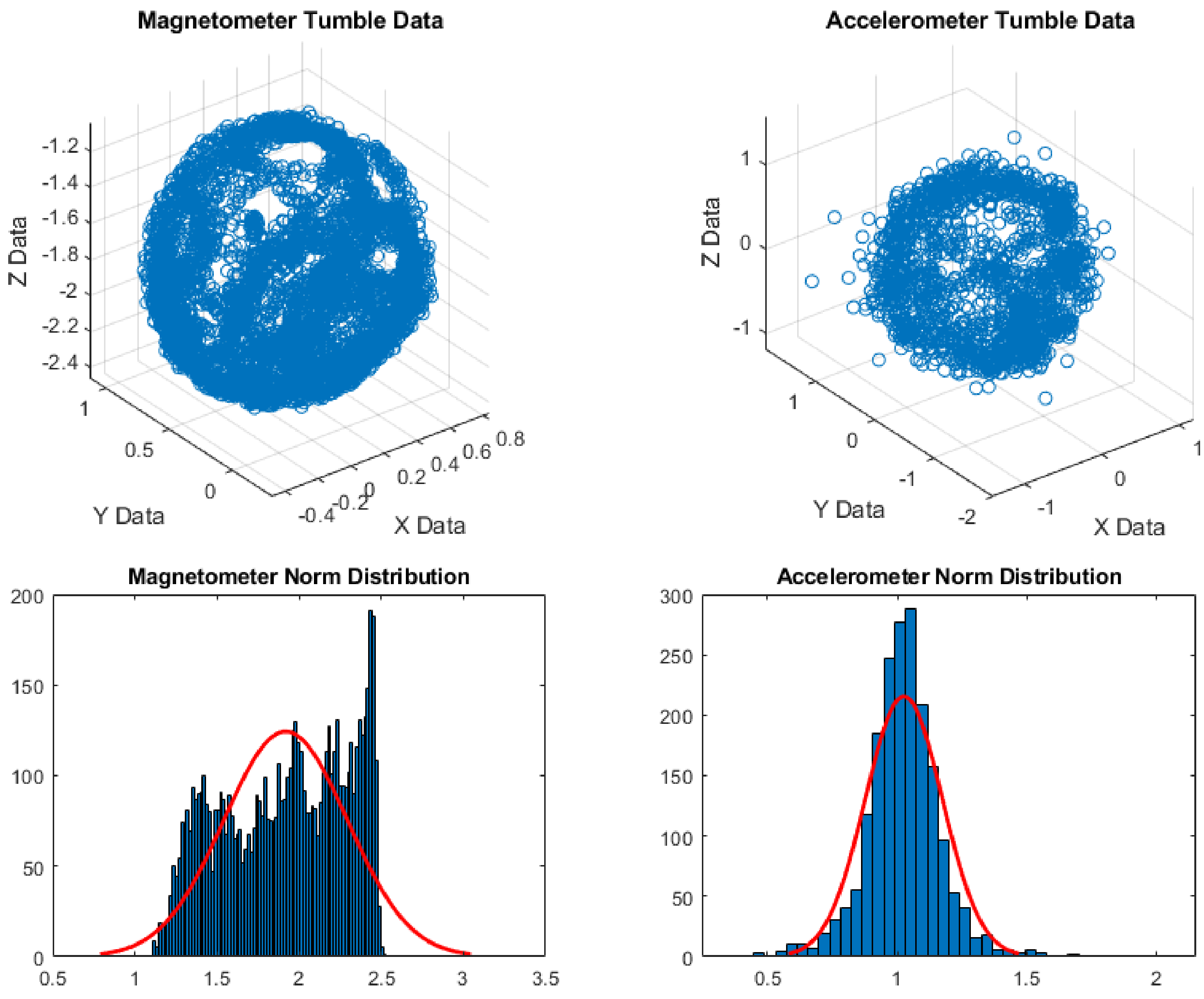
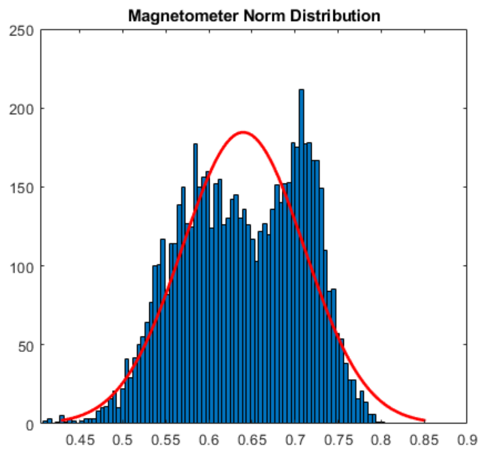


Figure 7.1: Experimental Tumble Data

The standard deviation for the data sets were .3760 and .1284 for the Magnetometer and Accelerometer respectively. The means were 1.9178 for the mag and 1.0258 for the Accelerometer.

7.1 Naive Calibration

Using the offsets from part 4 that were experimentally determined, the sensors are calibrated. That is, *Null Bias: 32, 115, -550 for x, y, z axis of Magnetometer. 662, 178, 919 for Accelerometer.*:



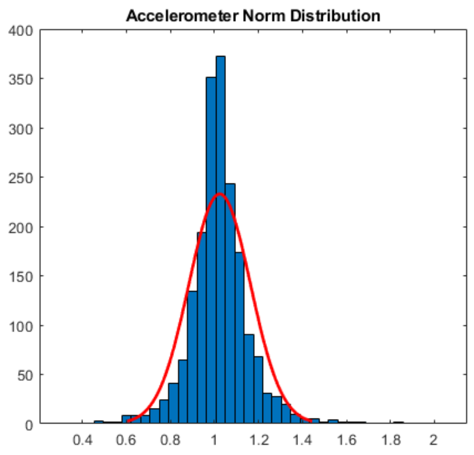


Figure 7.2: Experimental Tumble Norm Distributions

The standard deviation for the data sets were .0706 and .1400 for the Magnetometer and Accelerometer respectively. The means were .6399 for the mag and 1.0243 for the Accelerometer.

7.2 Least Squares Calibration

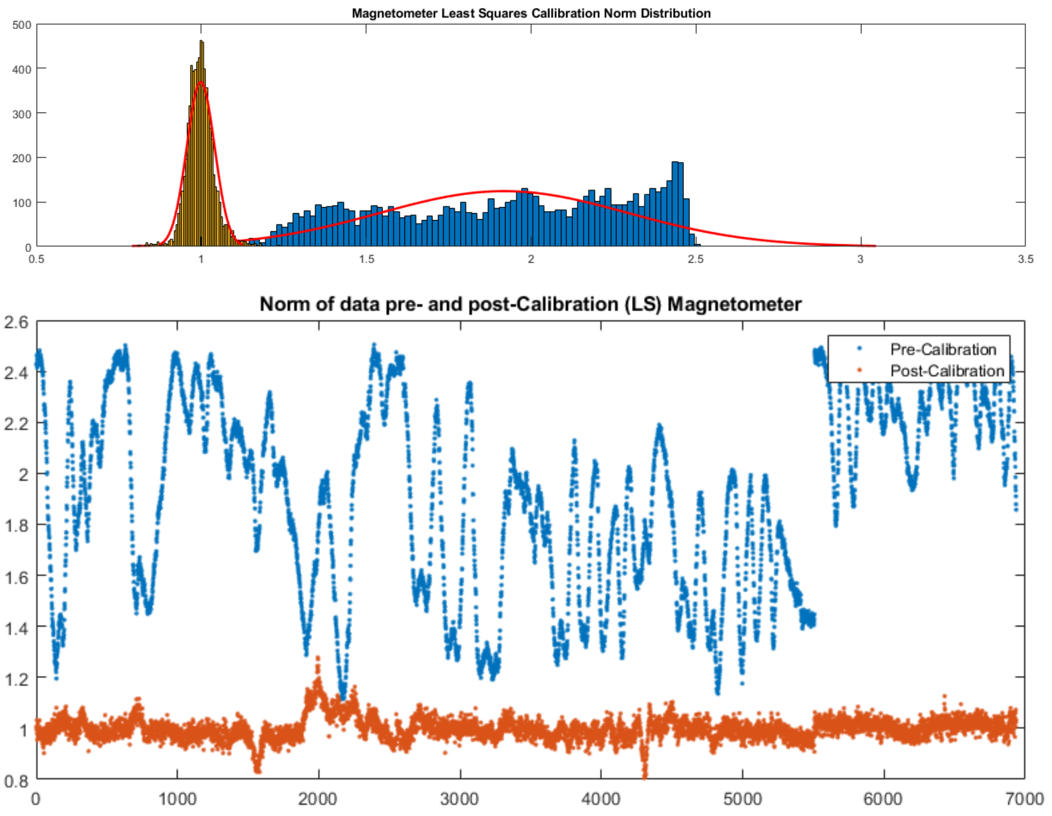


Figure 7.3: Experimental Tumble Magnetometer LS Calibration

The standard deviation for the Accelerometer after a Least Squares calibration is .1306 and has a norm mean of .9827.

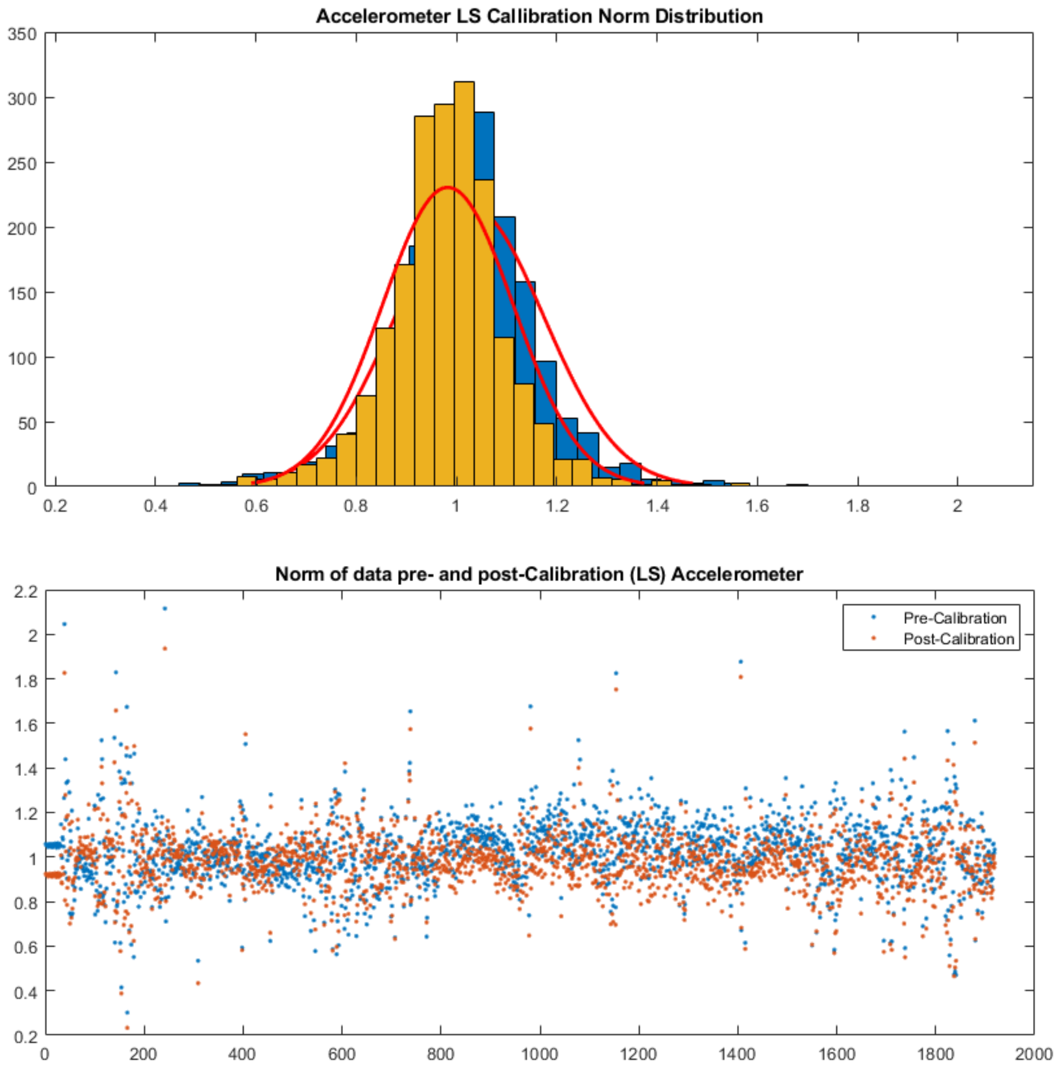
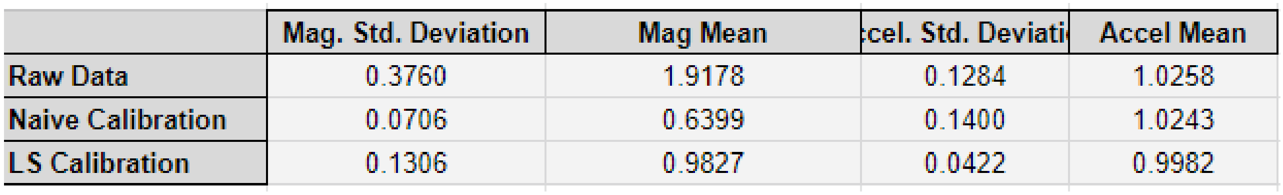


Figure 7.4: Experimental Tumble Accelerometer LS Callibration

The standard deviation for the Accelerometer after a Least Squares callibration is .0422 and has a norm mean of .9982.

7.3 Comparison of Values



	Mag. Std. Deviation	Mag Mean	cel. Std. Deviati	Accel Mean
Raw Data	0.3760	1.9178	0.1284	1.0258
Naive Calibration	0.0706	0.6399	0.1400	1.0243
LS Calibration	0.1306	0.9827	0.0422	0.9982

Figure 7.5: Experimental Tumble Callibration Statistics

8. Conclusion

In conclusion, this lab explored calibration techniques for IMU systems. These techniques included least squares solutions as well as naive techniques like calibration through visual inspection. The LS technique was more accurate but he naive calibration technique wasn't entirely worthless. The datasheets were off in both sensors by a factor of 10-20 percent. Calibration through data collecction and analysis is very much required before any use of the sensor, but Least Squares isn't necessarily the only option. Data analysis through data visualization was the most helpful part of gaining intuition on these systems.