Johnson Le

02/28/2019

CMPE 167

Lab 3: 3D Sensor Calibration

Introduction:

The focus of this lab is on the IMU MPU-9250 which integrates a MEMs accelerometer, gyro, and magnetometer into a single module. The problem with this module is it is cheap and therefore is prone to a lot of error sources. The IMU will need to be calibrated. In this lab, we achieve this by discovering a scale factor and bias drift and analyzing the reliability of this method.

Part 1: Hardware Setup, Materials Needed, and Configuration

Method:

Following the instructions provided in the lab manual, I was able to get some data readings from the IMU. There wasn't much to be done from this part. I had sets of jumper cables laying around and was able to connect the IMU with just wires as well as a breadboard if needed. I wrapped the ends of the wires to ensure I don't connect the IMU incorrectly. This proved to be very useful as disconnecting and reconnecting the IMU became a simple task.

Part2: Ellipsoid Calibration Using Simulated Data

Method:

From the provided equation of an ellipse $\frac{(x-x_0)^2}{a^2}+\frac{(y-y_0)^2}{b^2}=R^2$, solve for x^2 as seen in figure 1. Do not take the extra step taking the square root because we want to keep terms separate to be able to solve for the constants attached to x and y. Utilizing matlab and the provided code, create a matrix of the variables using the provided Xmeas and Ymeas data and putting column of ones for the constant part. Using matlab's backslash or mldivide function, we can solve the system of equations to get values for the constants. From there, it is just simple algebra and matlab can be used for this as well. Once the unknown varaibles are solved for, we can use the provided functions to plot the calibrated and corrected ellipse data.

$$\frac{x^2}{a^2} - \frac{2xx_0}{a^2} + \frac{x_0^2}{a^2} + \frac{y^2}{b^2} - \frac{2yy_0}{b^2} + \frac{y_0^2}{b^2} = R$$

$$x^2 = a^2\left[R^2 + \frac{2xx_0}{a^2} - \frac{x_0^2}{a^2} - \frac{y^2}{b^2} + \frac{2yy_0}{b^2} - \frac{y_0^2}{b^2}\right]$$

$$x^2 = a^2R^2 + 2xx_0 - x_0^2 - \frac{a^2}{b^2}y^2 + \frac{2a^2y_0}{b^2}y - \frac{a^2y_0^2}{b^2}$$

$$= 2xx_0 - \frac{a^2}{b^2}y^2 + \frac{2a^2y_0}{b^2}y - \frac{a^2y_0^2}{b^2} + a^2R^2 - x_0^2$$

Fig 1: Ellipse Equation Algebra

Results:

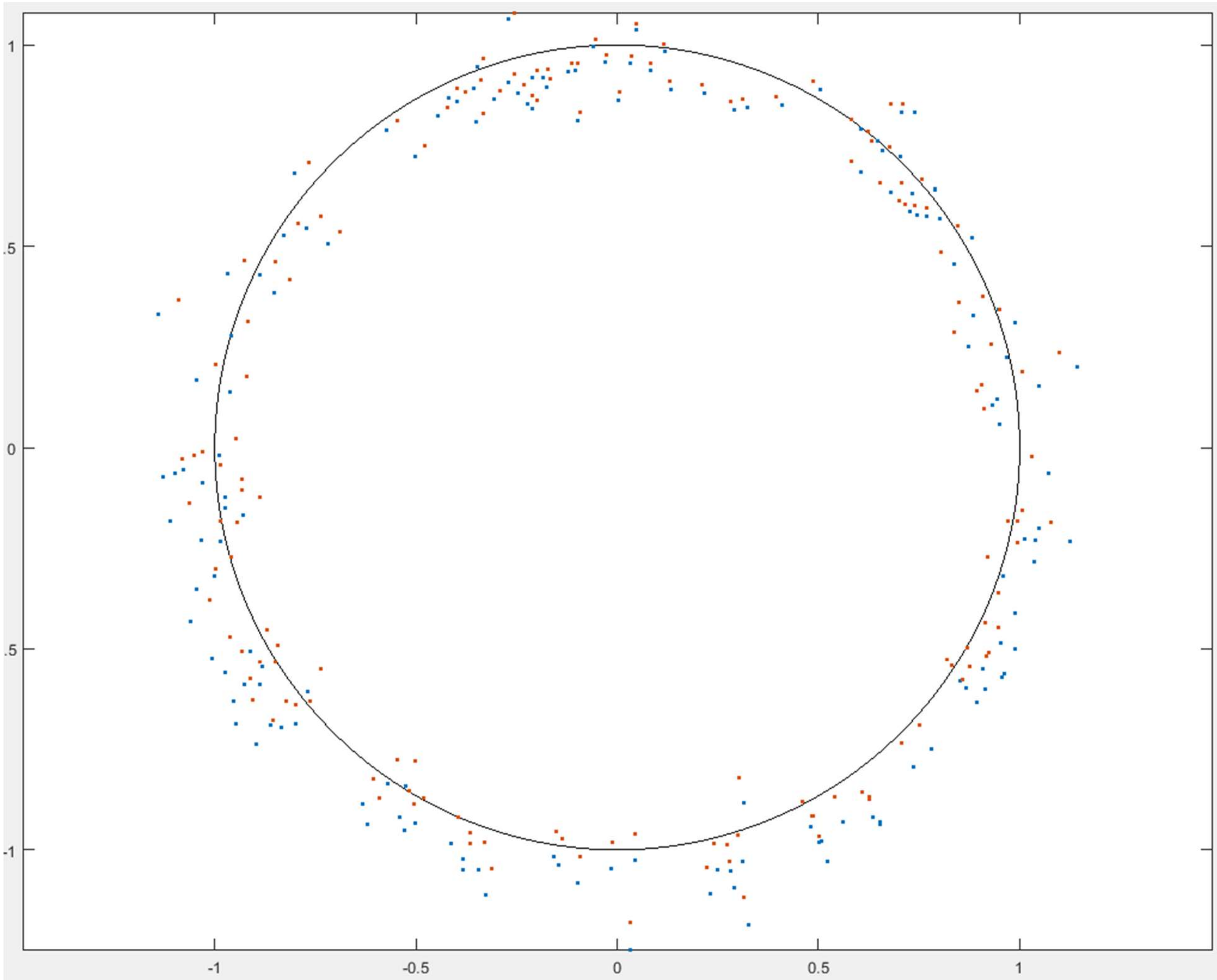


Fig 2: Calibrated Ellipse Data

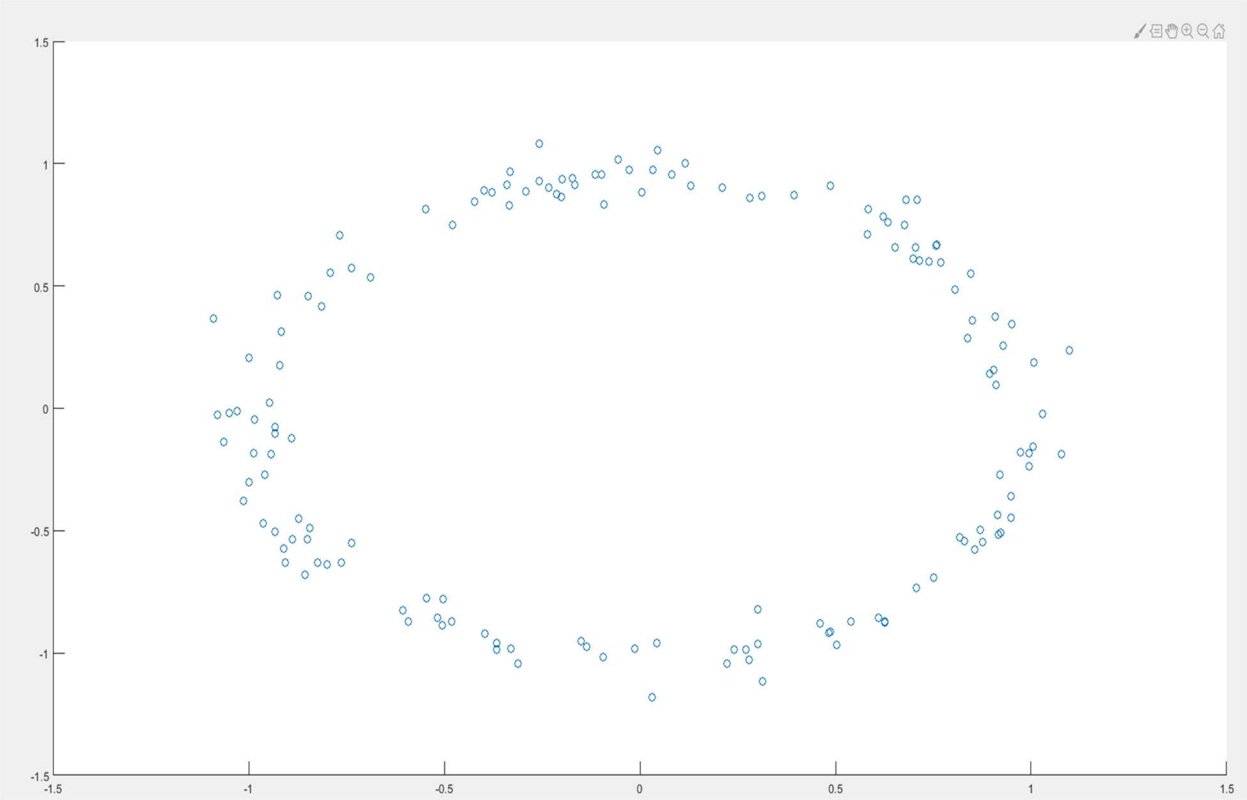


Fig 3: Corrected Ellipse Data

Part 3: Naïve Calibration of Magnetometer and Accelerometer

<u>Method:</u>

The first step is to set up a way to log data into a file. In the appendix section of the manual, it is recommended that putty is used which is what we will use but other methods should be fine. This is useful because we can later import the data into excel or matlab.

To set up taking data, we utilize the provided MUP9250 library to be able to interface the IMU data. The library updates the data every 100hz which is quite fast, and we can slow it down grabbing data every 50hz. It is important to print easily extractable data for putty to print data. A simple '"x,y,z" format can be used. For this naïve calibration, we don't have to take data for too long.

For the accelerometer, grab data for each axis at both the negative and positive poles. Once that is done, we need to find out what the sensitivity scale. The provided library and the MPU-9250 datasheet, we can determine the sensitivity scale is 16384 LSB/g. We know that the min and max of each axis should represent -1g and 1g. Averaging the data given and then dividing by the sensitivity scale, we are able to plot experimental vs expected providing us with a null shift and offset for each axis.

The magnetometer is calibrated a bit differently. The manual provides us with two methods. We will opt for the second method. Once an orientation is set up, some data can be taken. There will be min and max for each axis when the other axes are zero. The null shift can be calculated by getting the average between the min and max value. Plotting expected vs experimental, we can extract the slope which will be the scale factor.

<u>Results:</u>

Accelerometer

Sensitivity Scale =	16384					
	x averages		y averages		z averages	
	-0.970458984	-1	-0.994873047	-1	-0.988769531	-1
	1.025390625	1	1.007080078	1	1.031494141	1

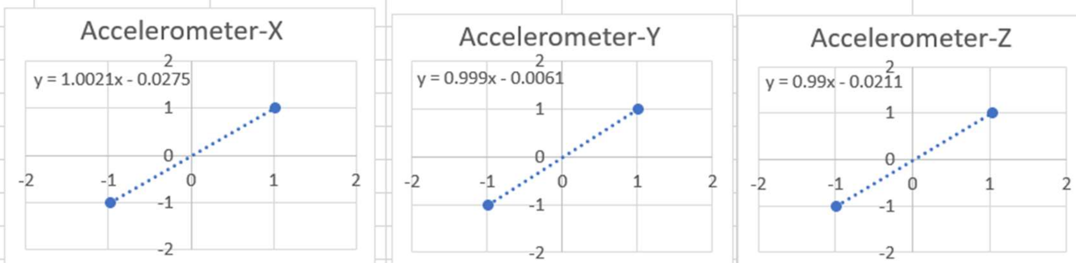


Fig 4: Part 3 Accelerometer Data

Magnetometer

	x	y	z
min	-70	-55	-430
max	97	140	1
avg			
min	-59	-49	-390
max	86	131	-7
null	13.5	42.5	-214.5
	0.011976	0.010256	0.00464
scale	1.0705	1.0833	1.1254
out-	-0.70659	-0.50256	-1.80974
out+	1.02994	1.34359	-0.03248

Fig 5.1 Part 3 Magnetometer Data

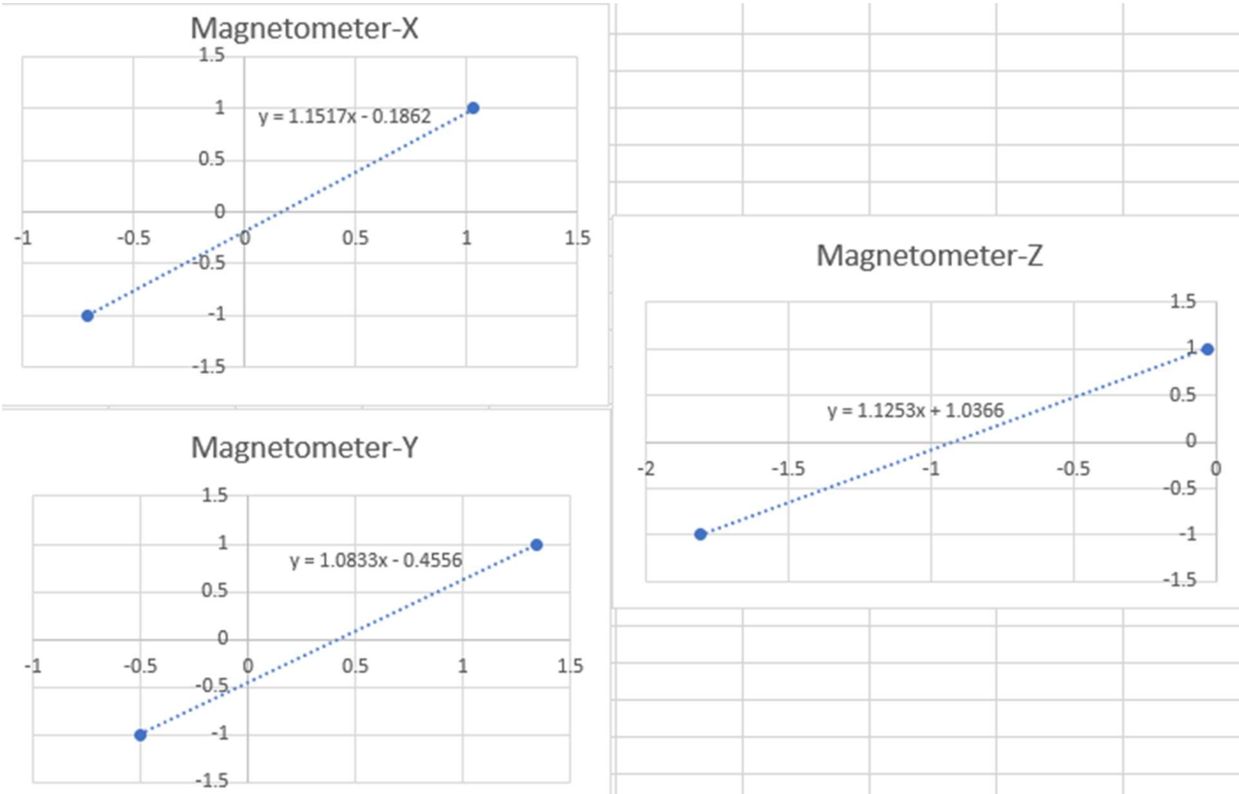


Fig 5.2 Part 3 Magnetometer Data

Part4: Gyro Bias and Bias Drift

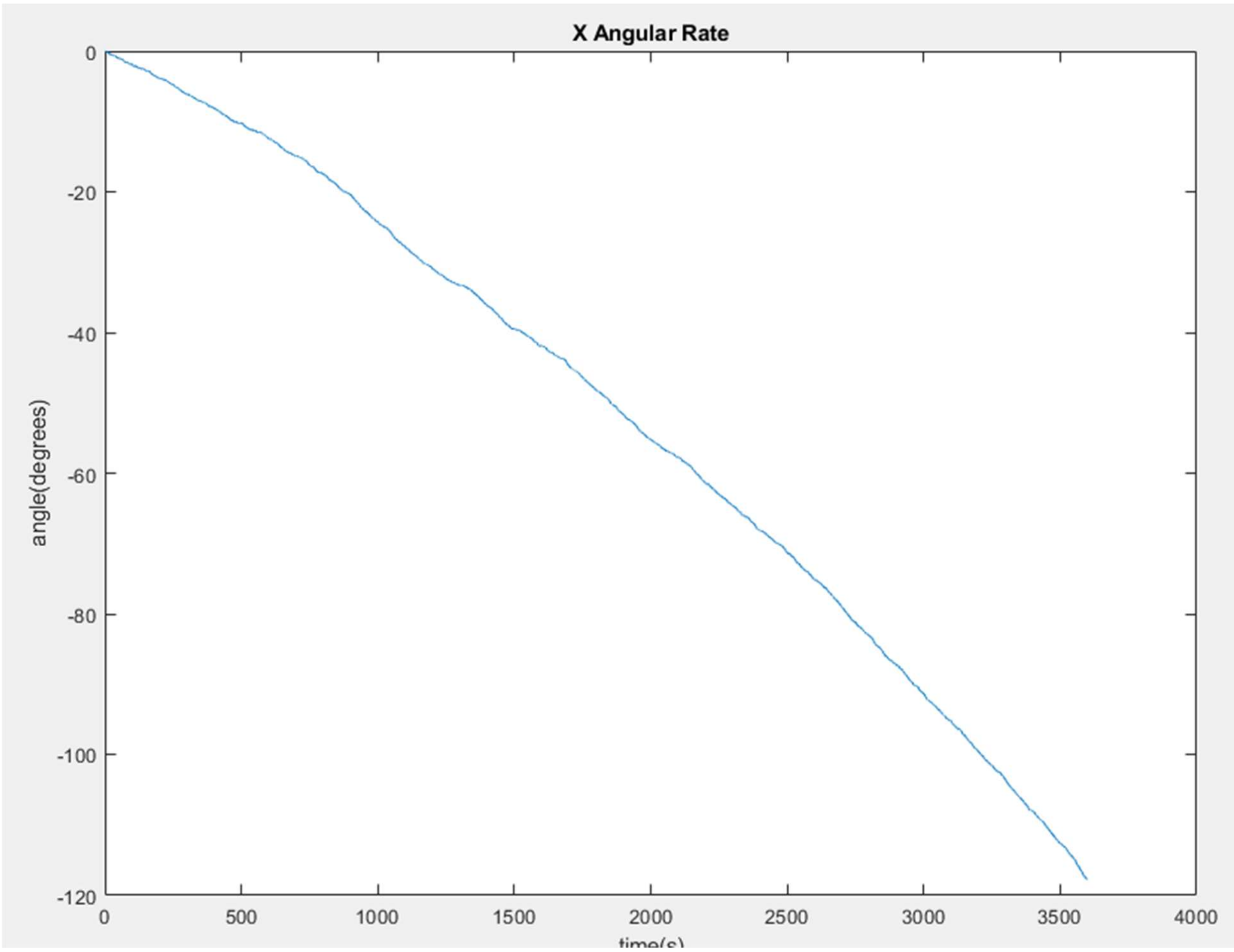
Now that we have some naïve calibration values for accelerometer and gyroscope, we move into getting some gyroscope data. Gyros have a bias drift that drifts over time. We will need to find the drift rate, so we can accommodate for that drift later.

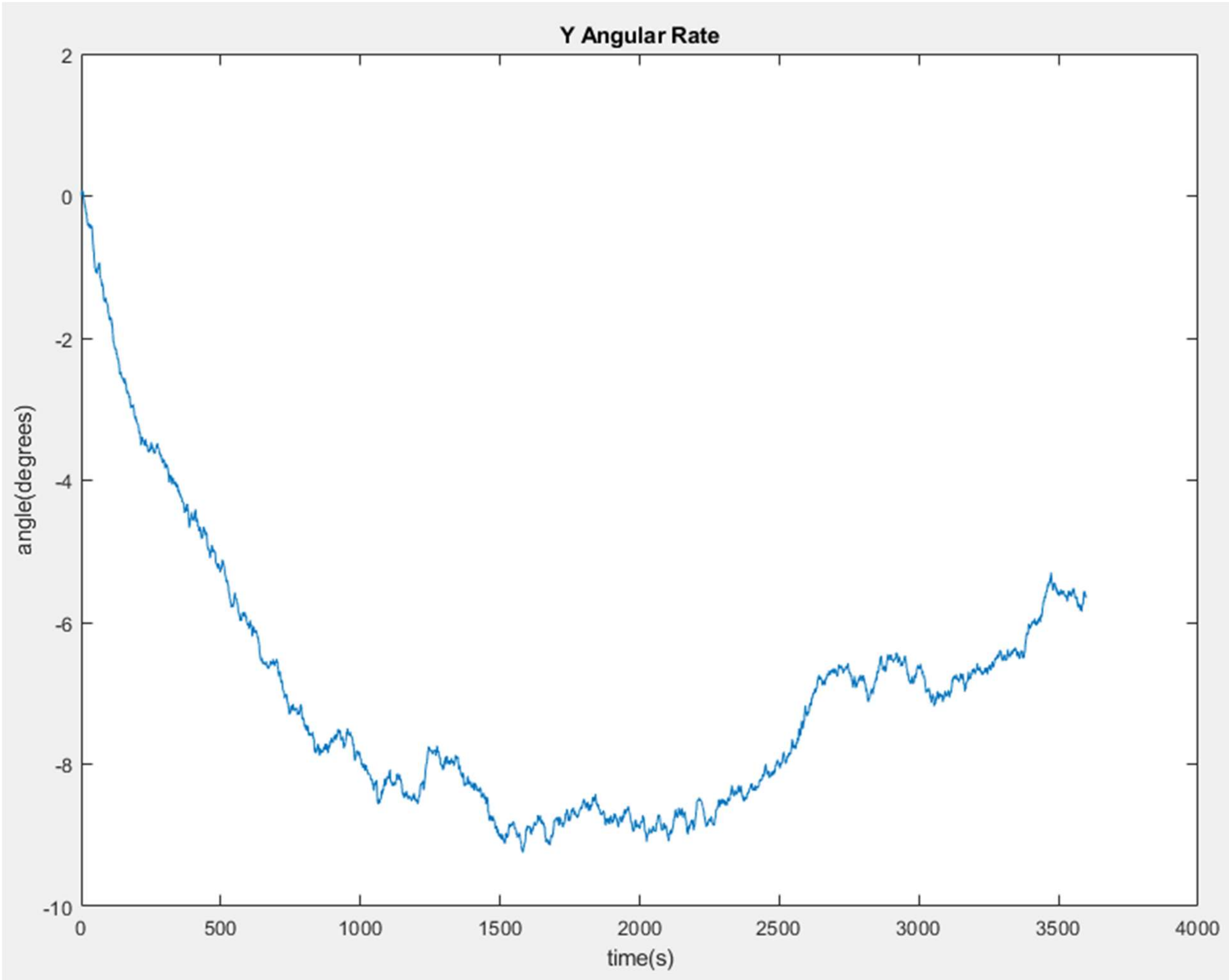
<u>Method:</u>

An ideal gyroscope will display will output 0 degrees per second while stationary indicating that there is no bias drift. We will be determining bias drift by monitoring the gyroscopic data over a long period of time. This is best done in a calm environment to minimize any disturbances. It is recommended that data be taken at a reasonable frequency instead of constantly. Although this reduces resolution of data, it can significantly reduce the amount of data which is useful for reducing computation load for matlab. For example, taking data in 0.5hz which is slow will provide us with 2 data per second. In one hour, that is 7200 data which is already quite a bit and can add up. The data provided is proportional to the angular rate.

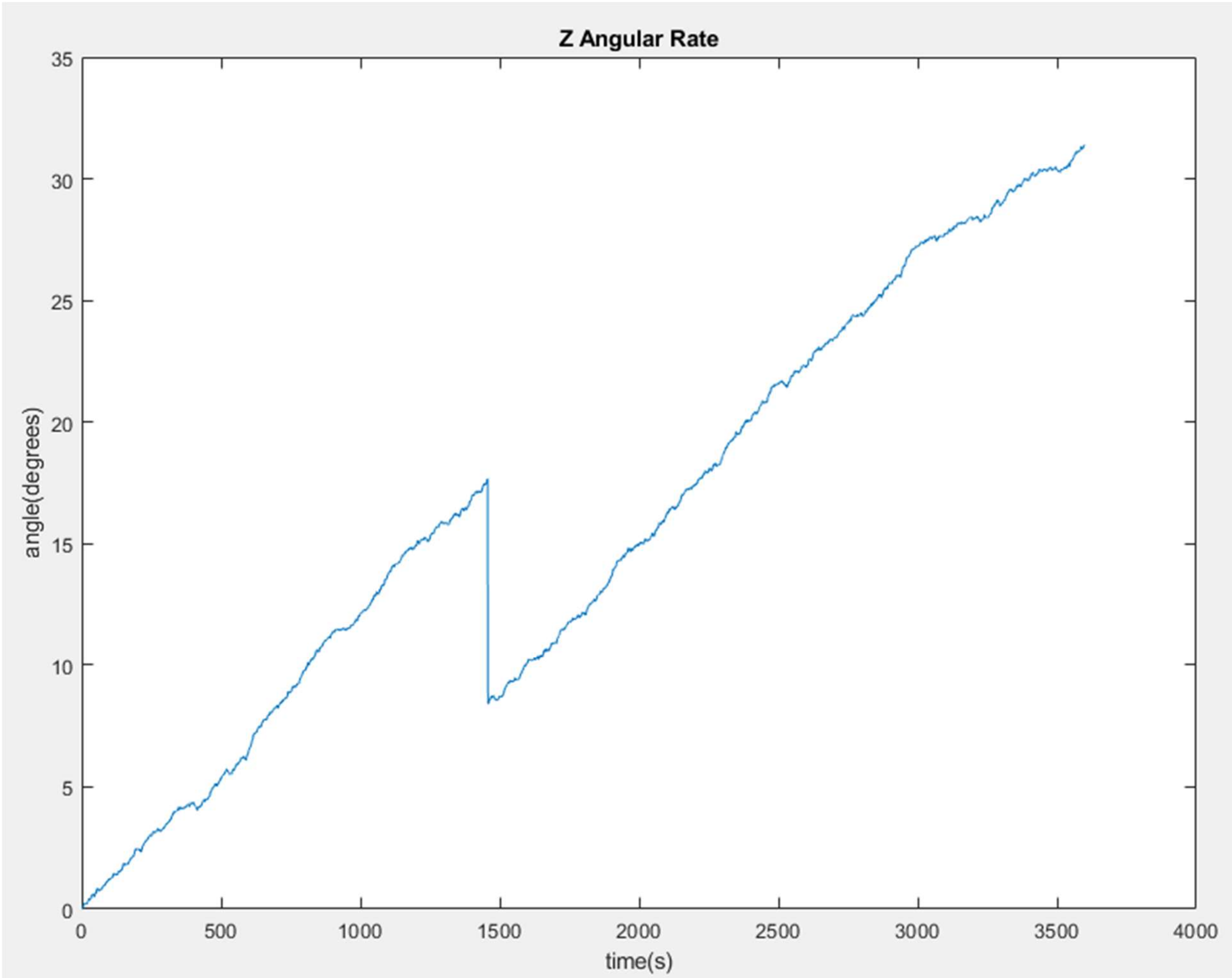
Results:

Around 20 minutes in, I accidently swatted the gyro and it is mostly seen on the z-axis.





Y Angular Rate
angle(degrees)
time(s)
2
0
-2
-4
-6
-8
-10
0
500
1000
1500
2000
2500
3000
3500
4000

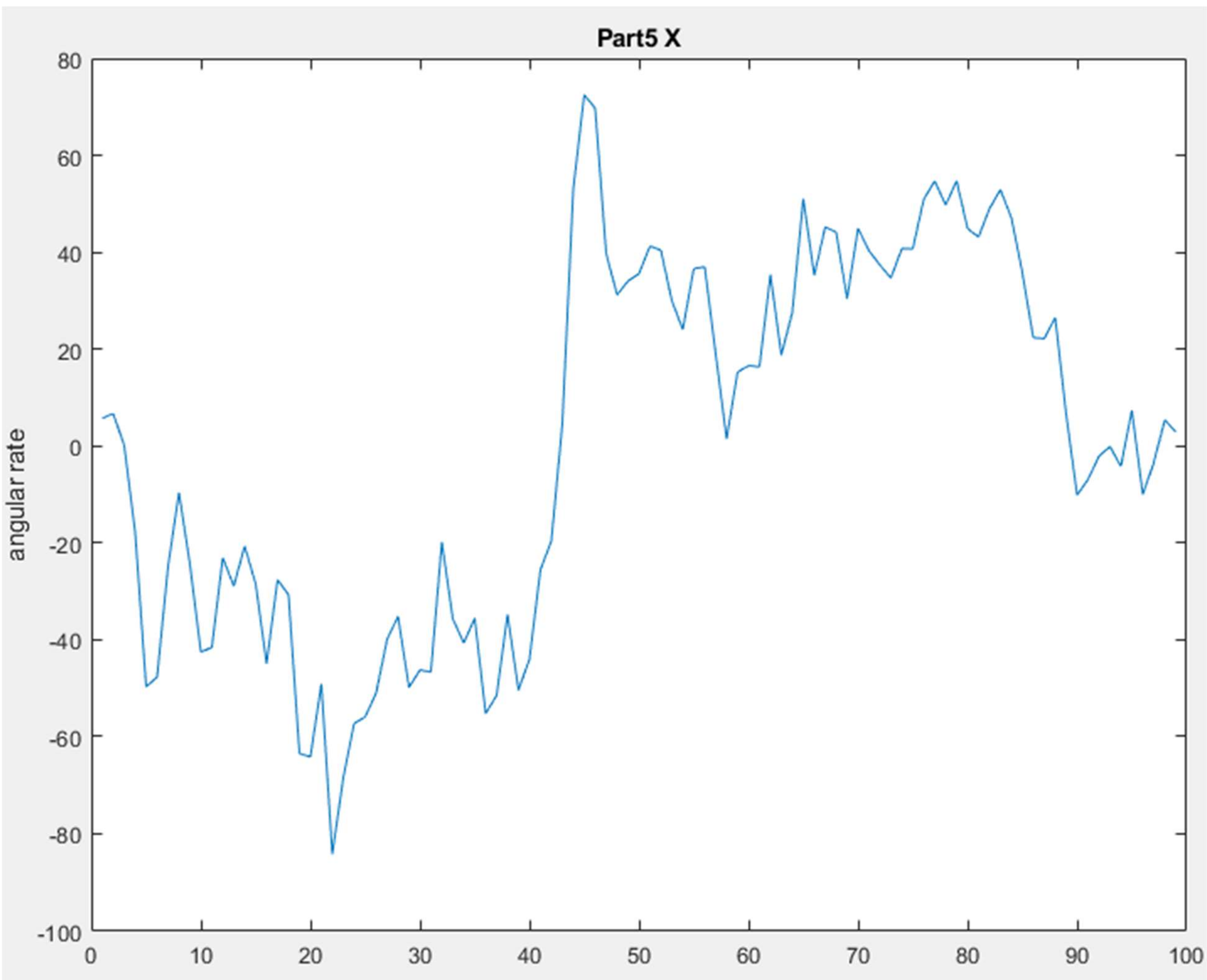

Z Angular Rate
angle(degrees)
time(s)
0
5
10
15
20
25
30
35
0
500
1000
1500
2000
2500
3000
3500
4000

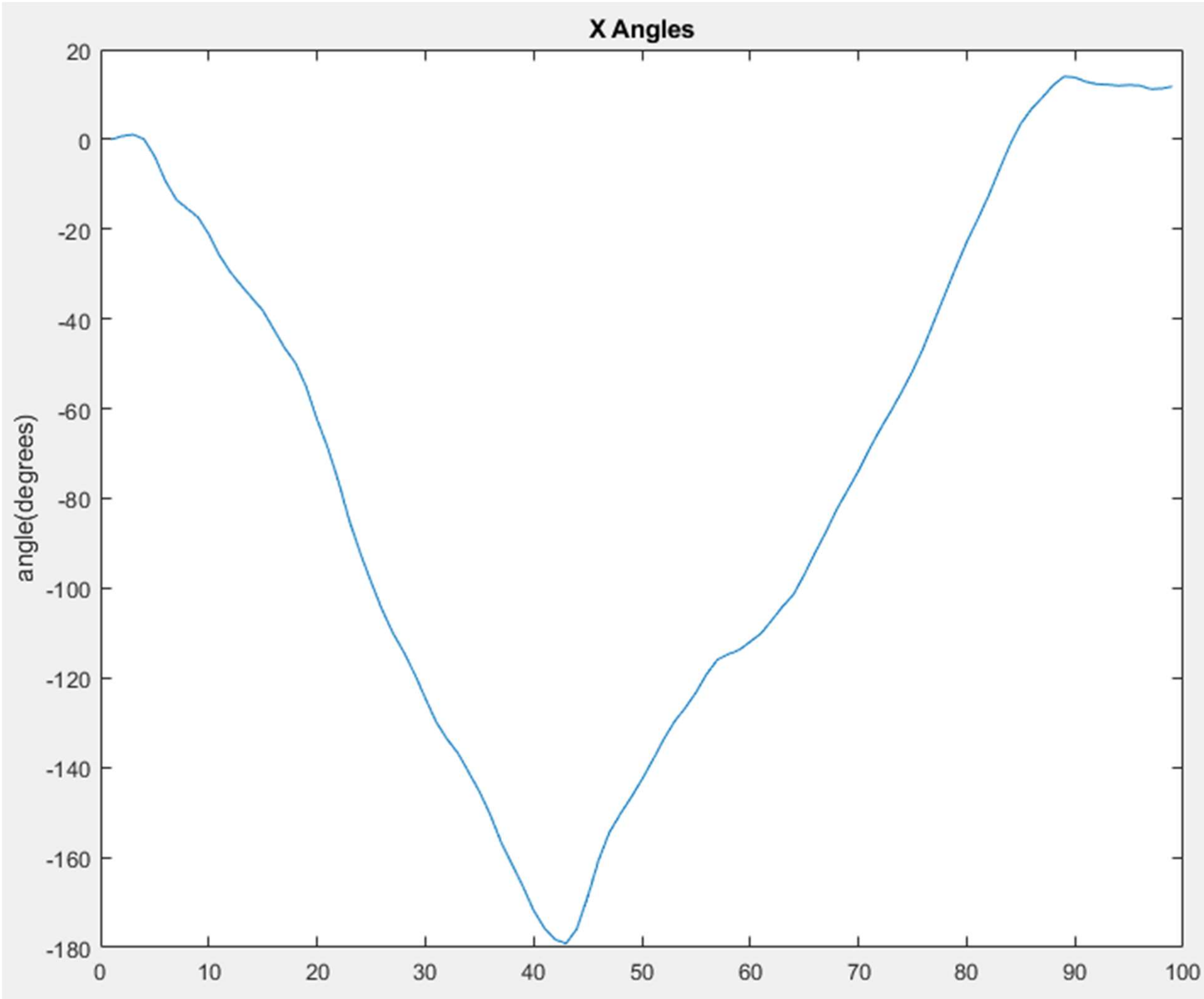
Part 5: Gyro Scale Factor via Angle Integration

Method:

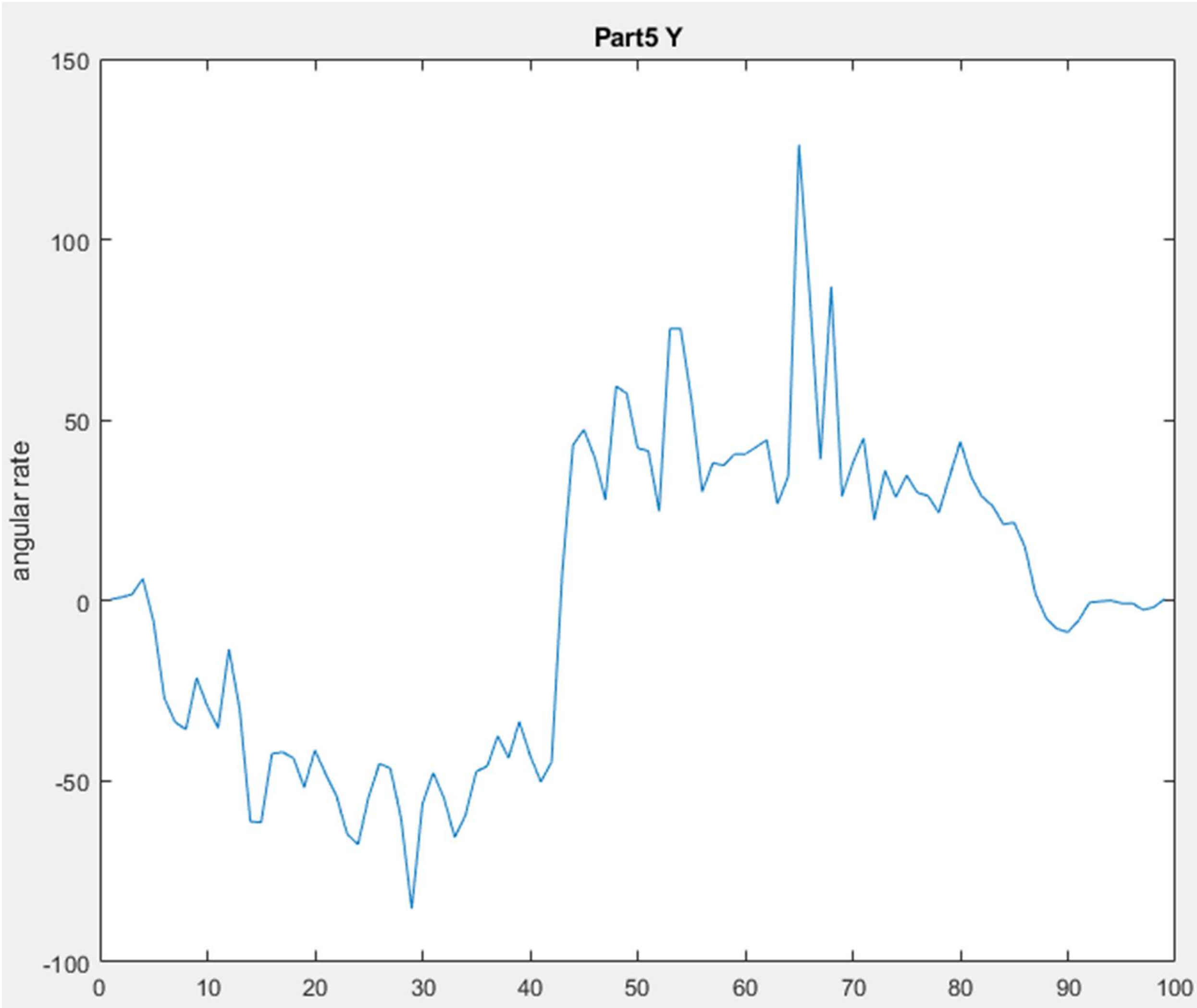
Part 5 will involve taking some controlled data to get actual angles. Take 10 seconds of stationary readings for a bias. This is done before and after our experiment and the values will be averaged for a bias. We will be rotating the IMU 180 degrees and back. It helps to mark some line on a table used to indicate the starting and ending point of rotation. Log data on each axis. Using matlab, subtract out the bias and then integrate using cumtrapz on the data and plot. Observe the change and note down the min or max depending on which way the IMU was rotated. Scale the min/max down to -+180. The plot should end up roughly going from 0 degrees to -+180 then back.

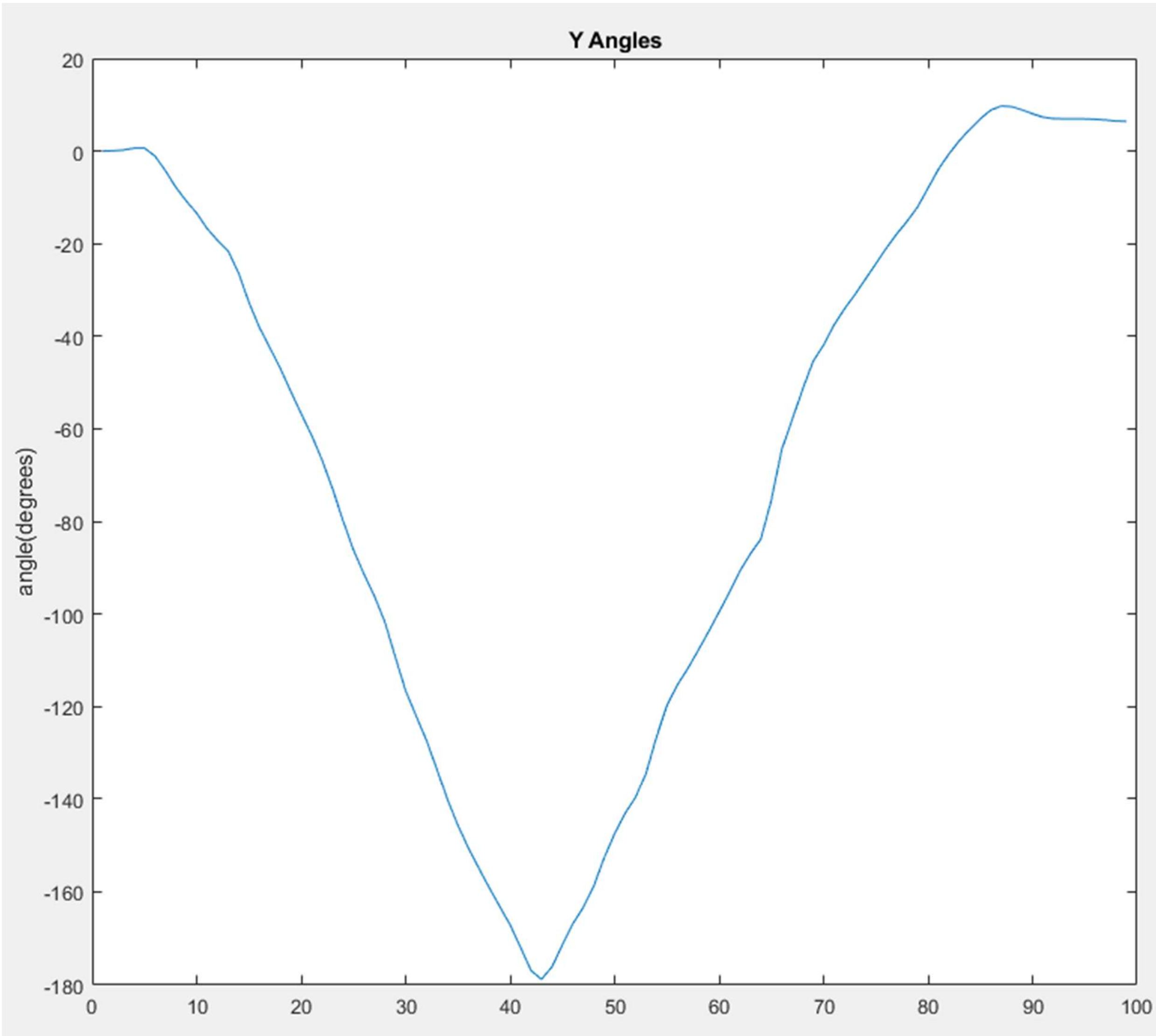
Result:

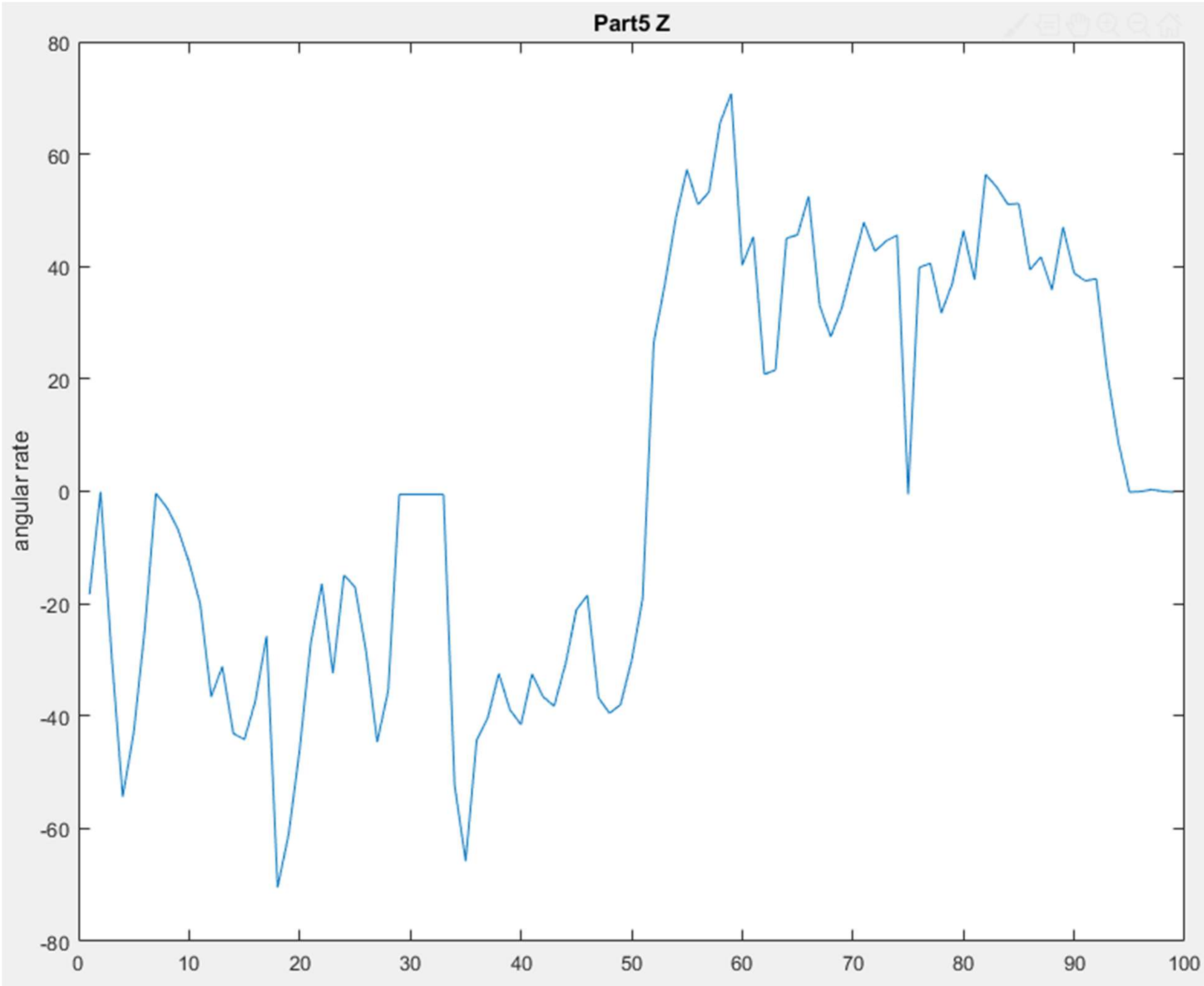


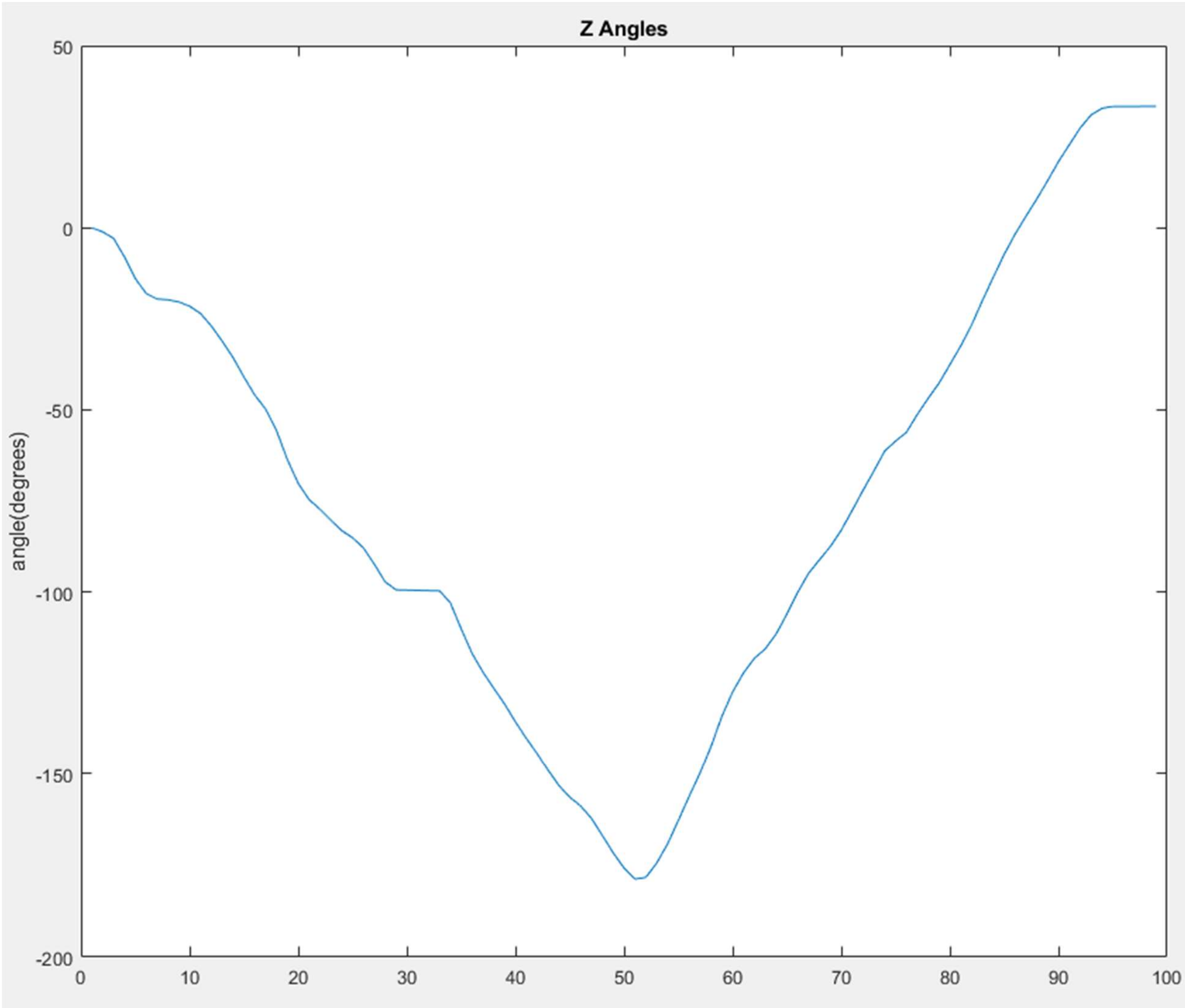


X Angles
angle(degrees)
20
0
-20
-40
-60
-80
-100
-120
-140
-160
-180
0
10
20
30
40
50
60
70
80
90
100


Part5 Y
angular rate
150
100
50
0
-50
-100
0
10
20
30
40
50
60
70
80
90
100


Y Angles
angle(degrees)
20
0
-20
-40
-60
-80
-100
-120
-140
-160
-180
0
10
20
30
40
50
60
70
80
90
100


Part5 Z
angular rate
80
60
40
20
0
-20
-40
-60
-80
0
10
20
30
40
50
60
70
80
90
100



Z Angles
angle(degrees)
50
0
-50
-100
-150
-200
0
10
20
30
40
50
60
70
80
90
100

Part 6: Tumble Test for Accelerometer and Magnetometer

Method:

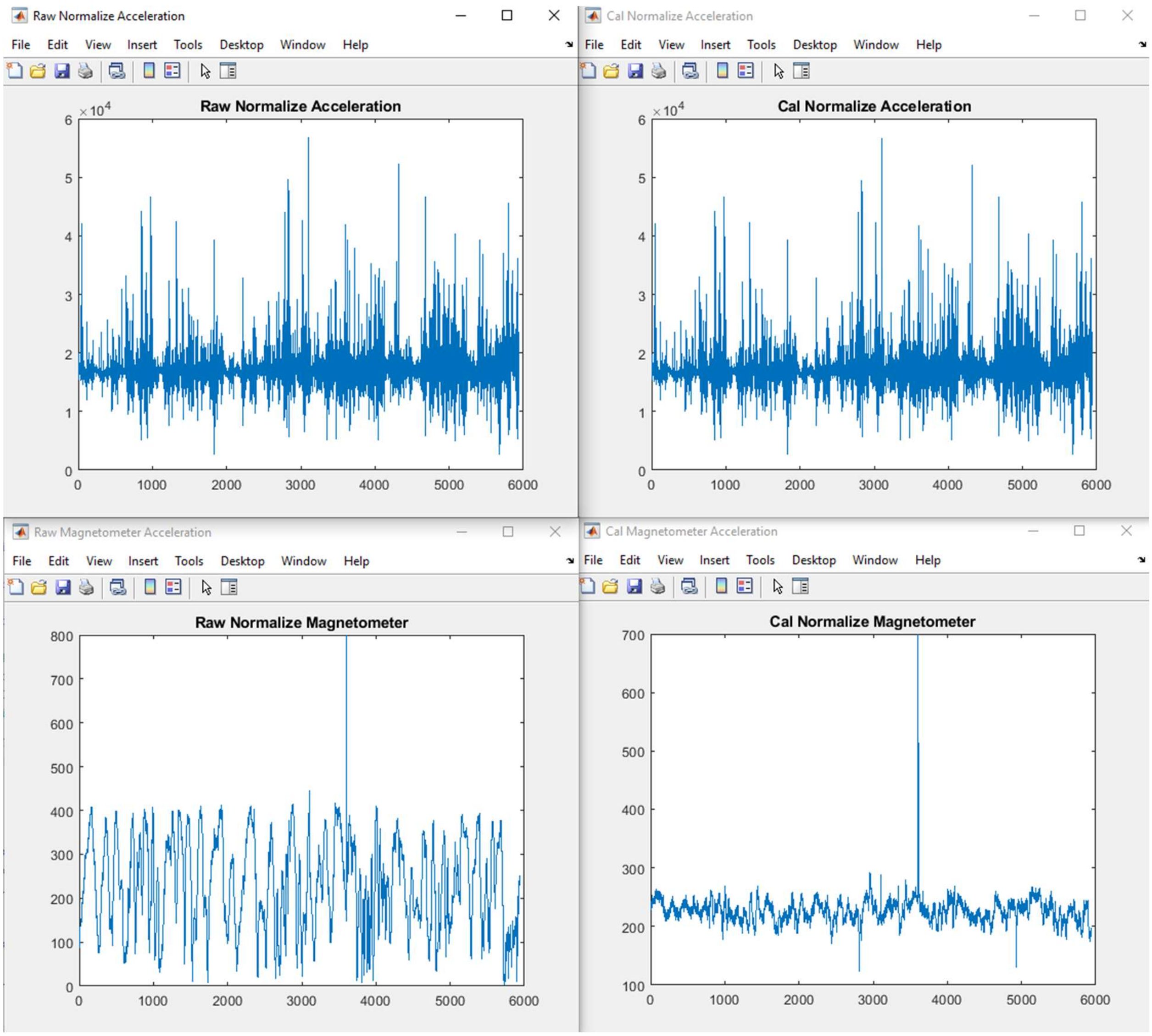
Now that we have explored a little on ellipsoid calibration using provided 2D data, we will be taking our own data in 3D and calibrating that. We will be needing a lot of data to fill out a "spherical" shape of points. Be a painter and paint a 3D sphere with the IMU while logging accelerometer and magnetometer data over some time. The more data taken, the "fuller" the sphere will be.

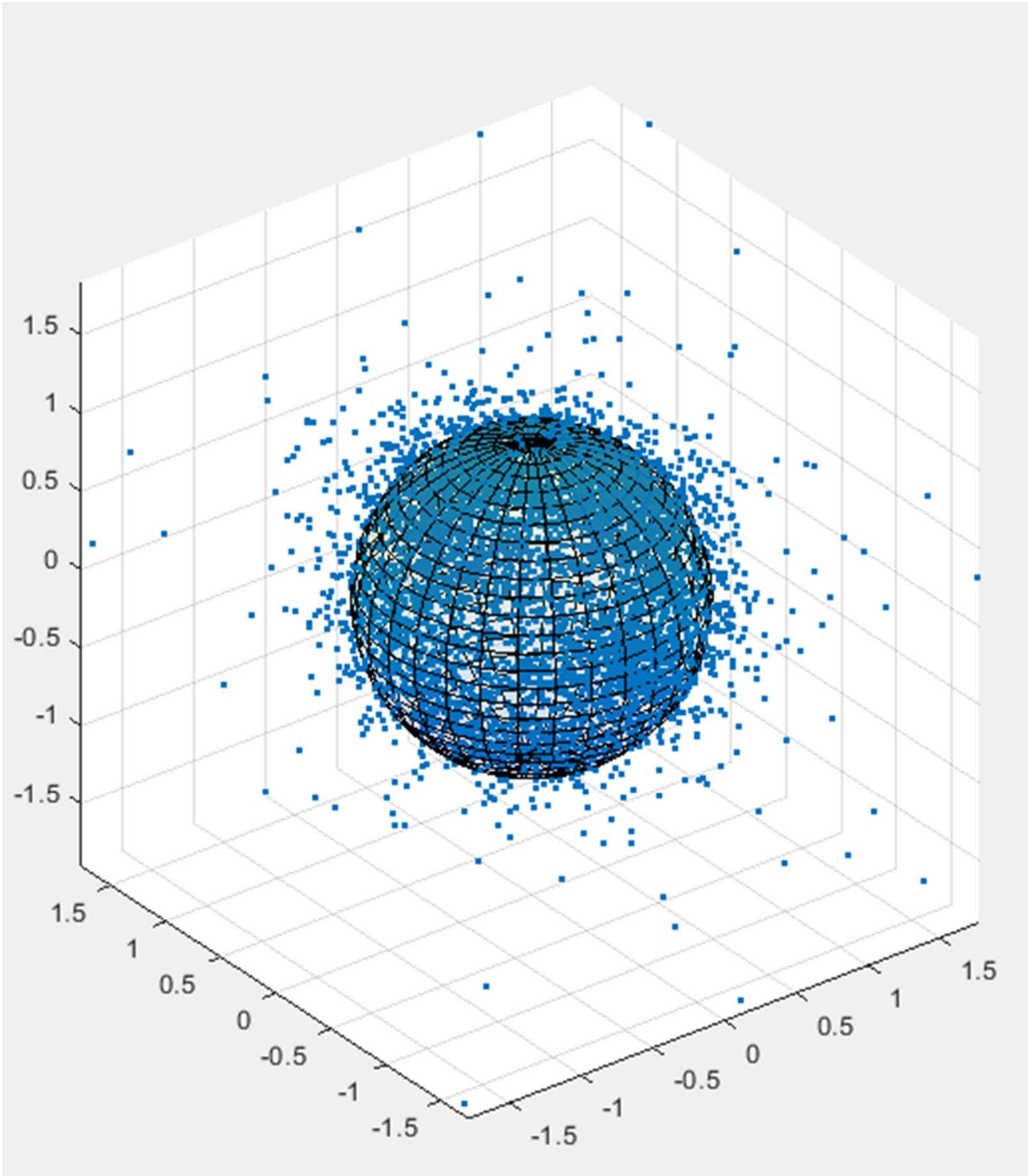
Once data has taken, use matlab to plot the raw and normalized data. This is for observing the changes needed to normalize the data. To further observe, we can utilize the provided matlab code to plot in 3D with a reference sphere. The calibration/normalization is done by using the shift and scale factors found in part 3.

Further calibration can be done by observing the sphere. More than likely, the data can be aligned more to the reference sphere. Add some offset and scaling on the axises that need it to try and fit it onto the reference sphere. These scale and shift values are closer to the actual values.

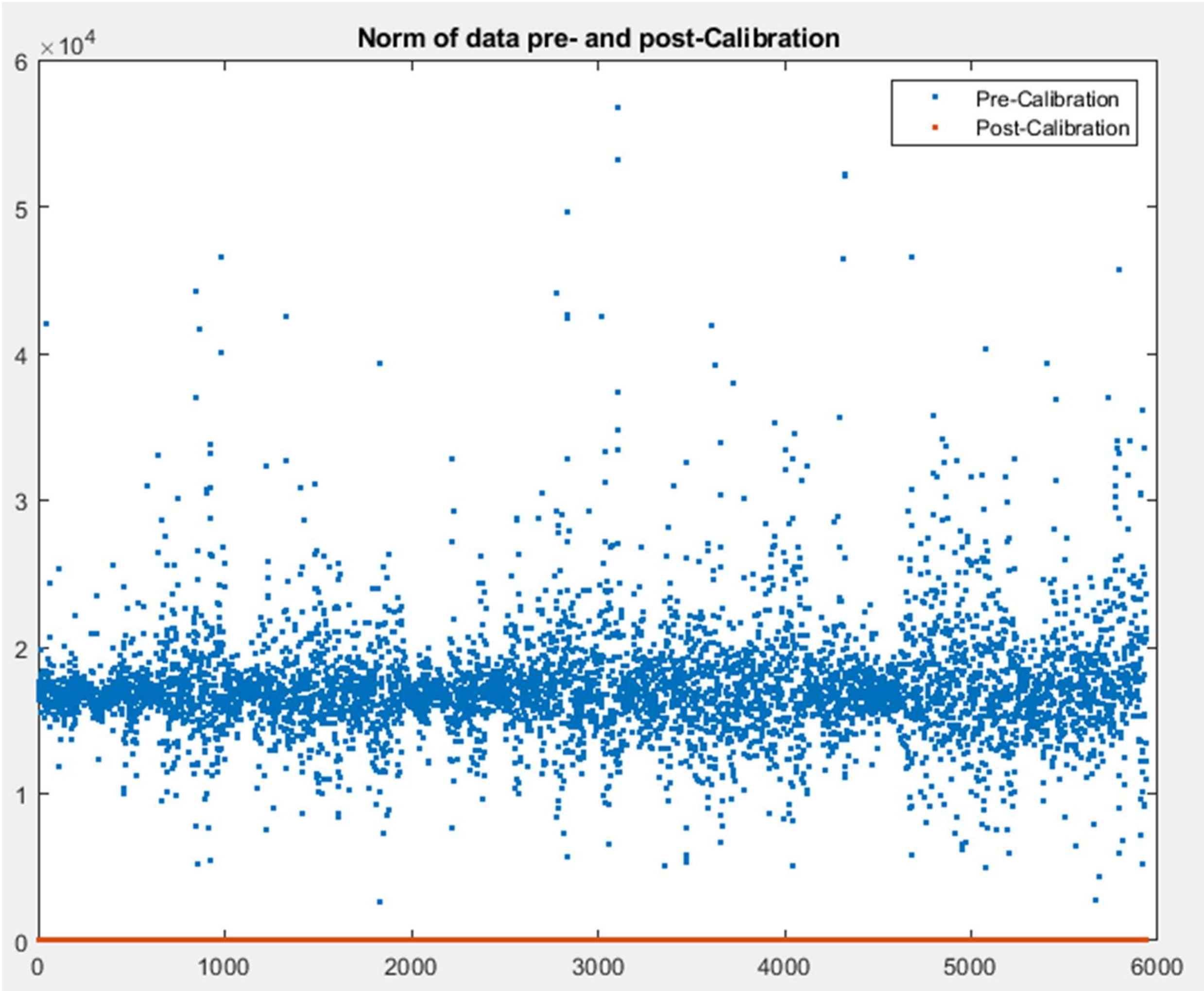
Results:

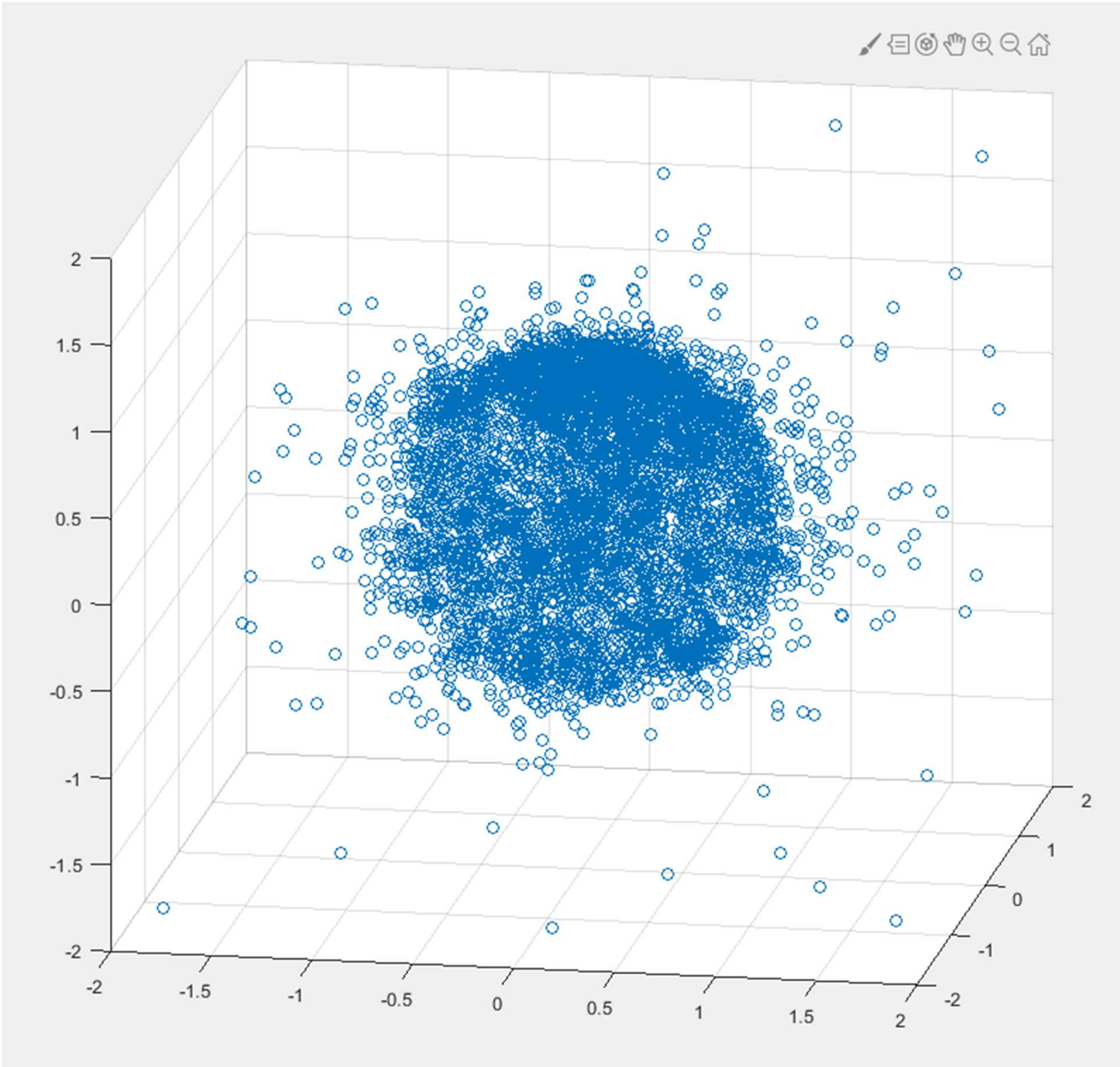
For the 3D part of accelerometer, I only provided the post calibration because there was very little change.





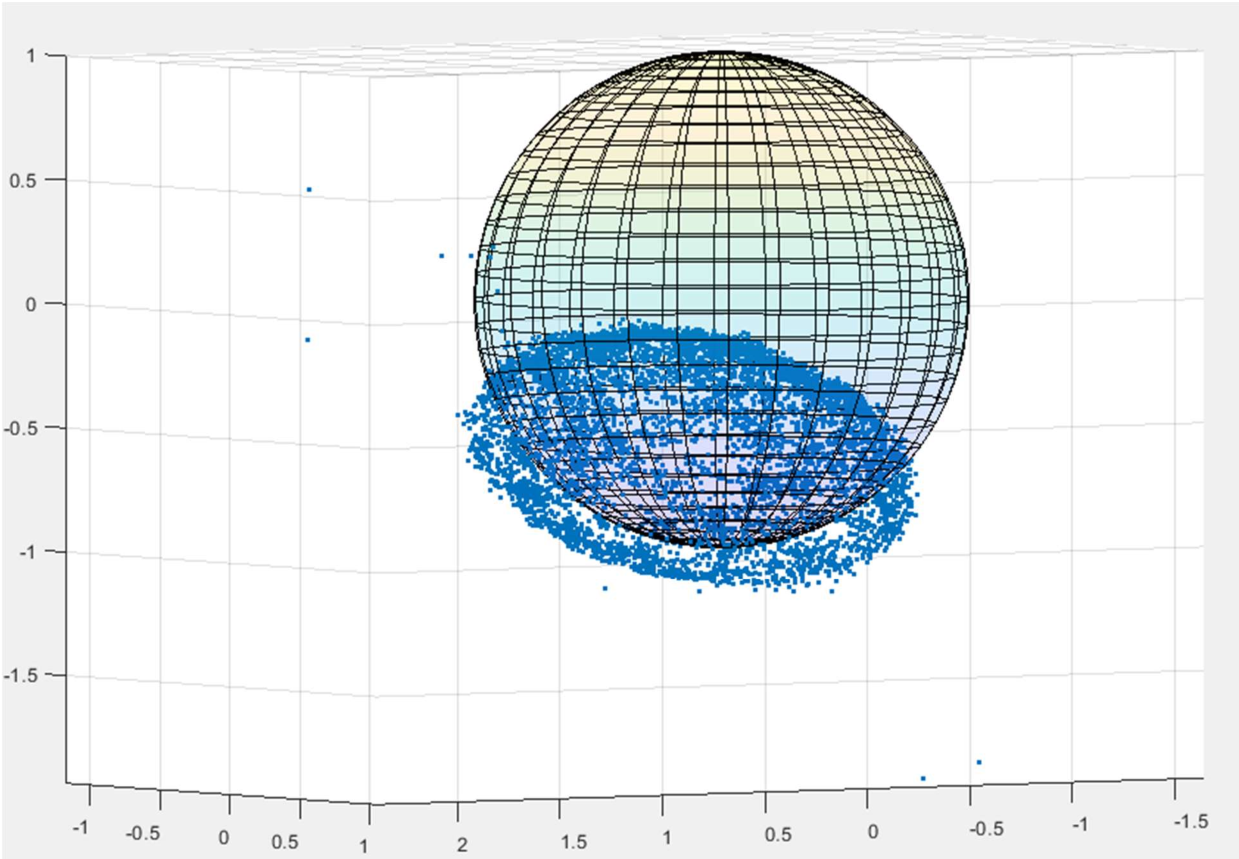
1.5
1
0.5
0
-0.5
-1
-1.5
1.5
1
0.5
0
-0.5
-1
-1.5
1.5
1
0.5
0
-0.5
-1
-1.5

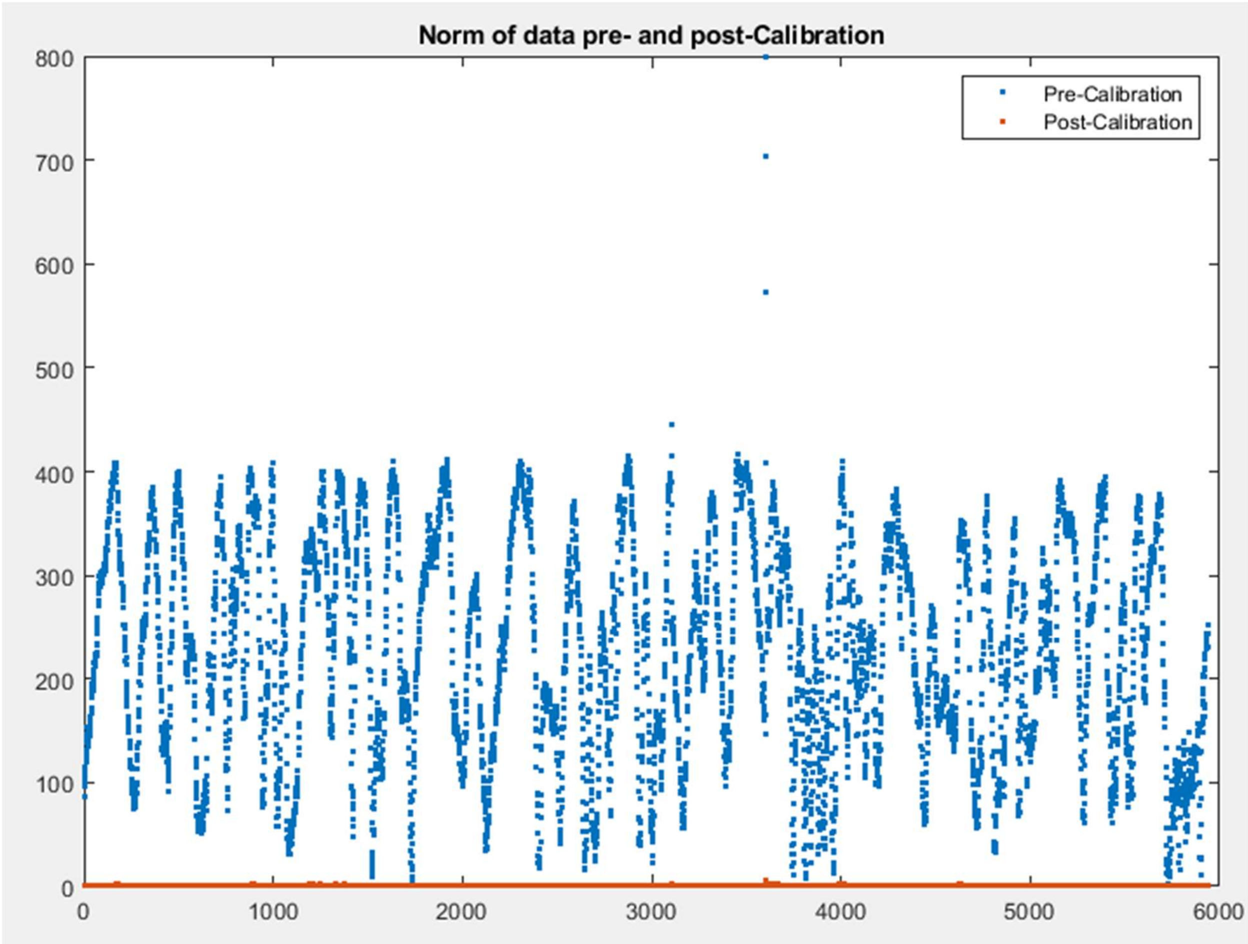

Norm of data pre- and post-Calibration
×10^4
6
5
4
3
2
1
0
0
1000
2000
3000
4000
5000
6000
Pre-Calibration
Post-Calibration

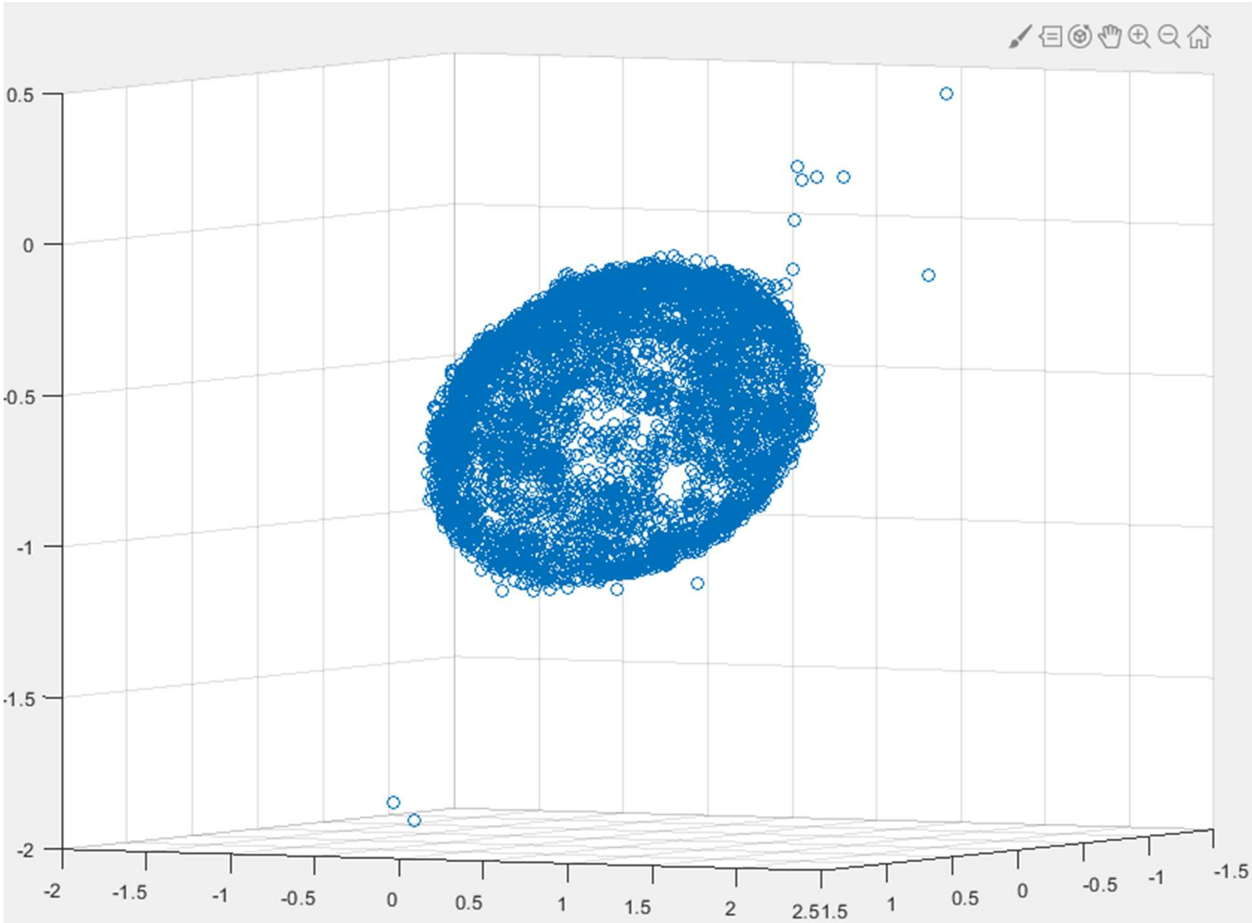


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2
1
0
-1
-2

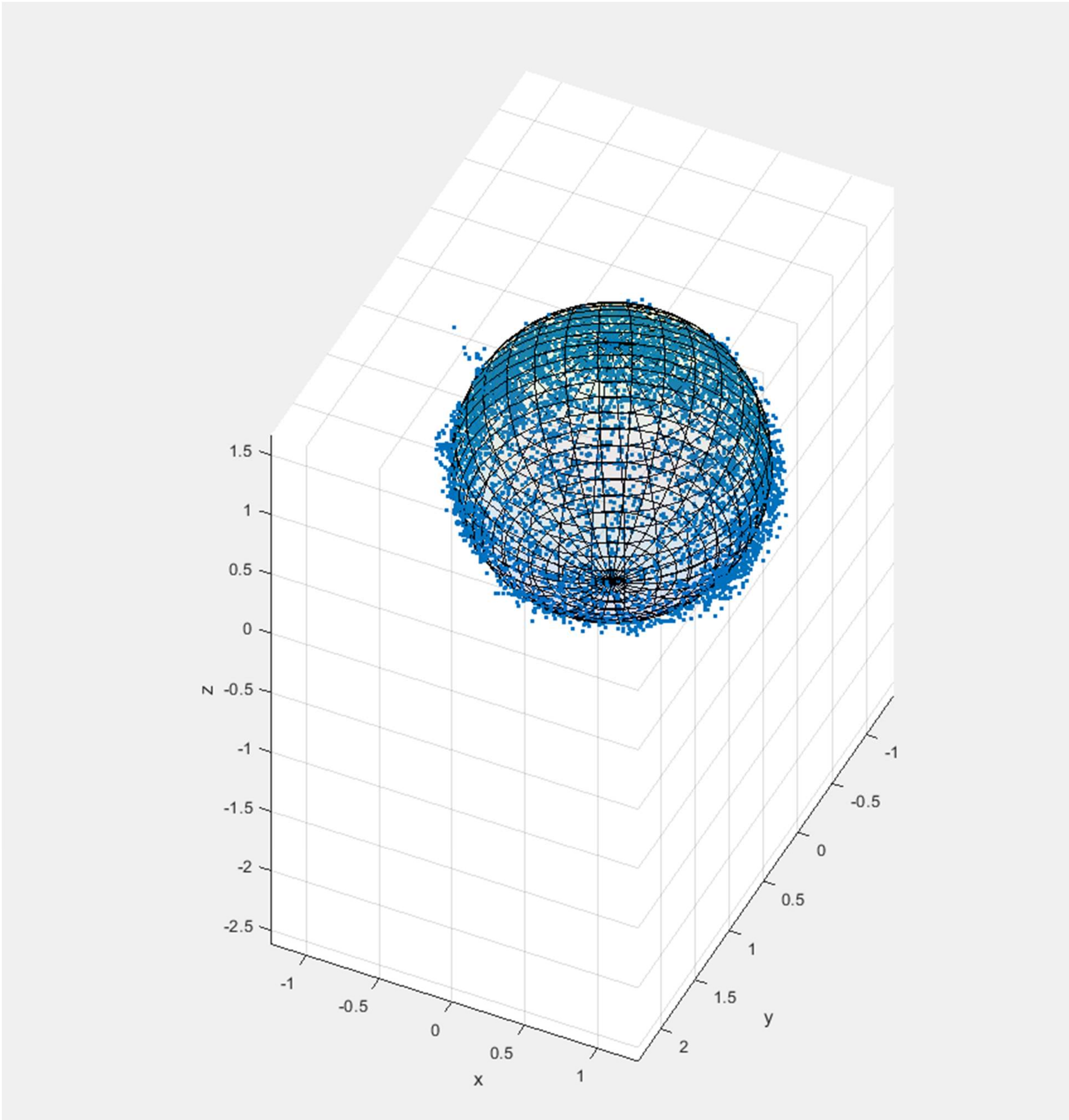
MAGNETOMETER

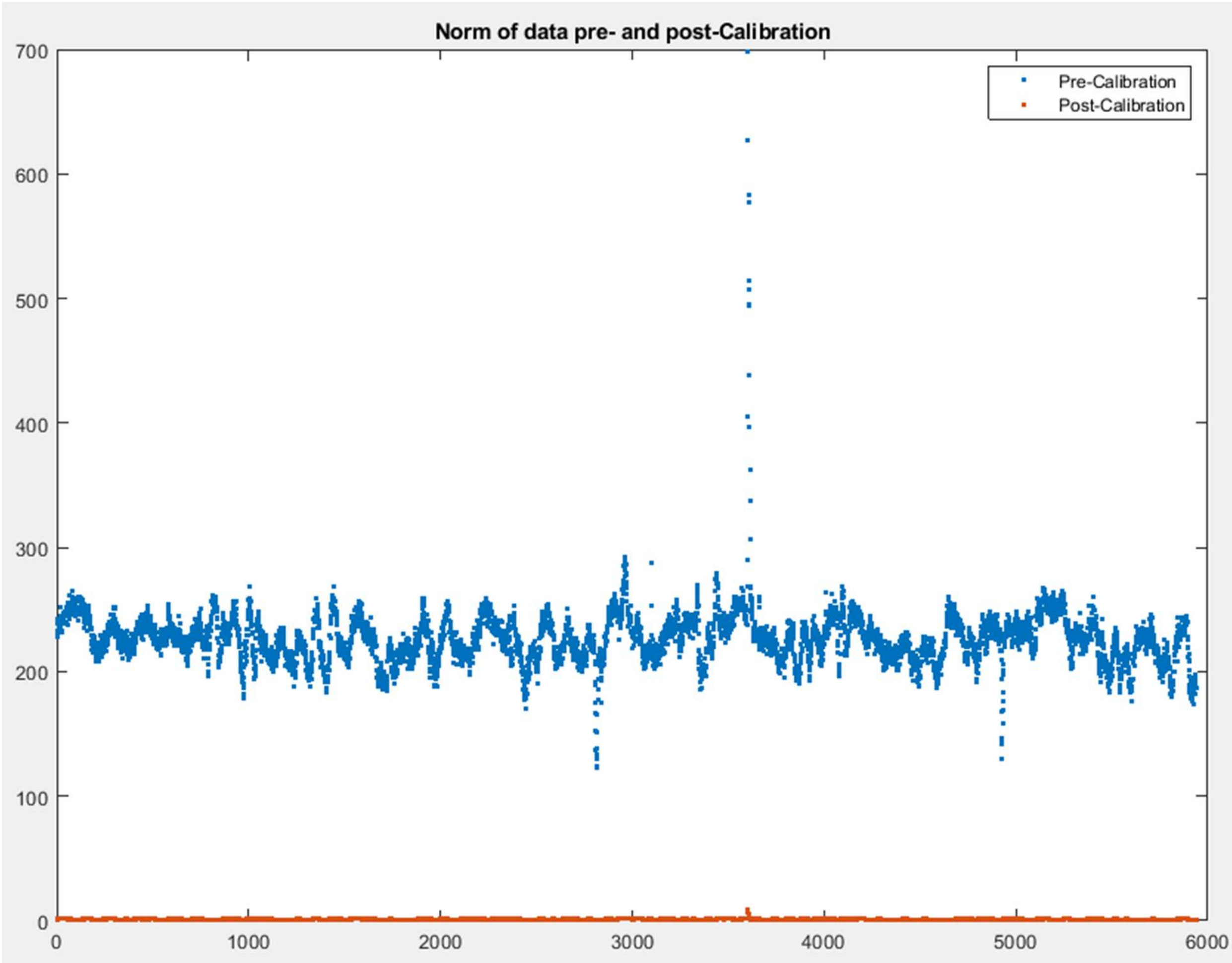


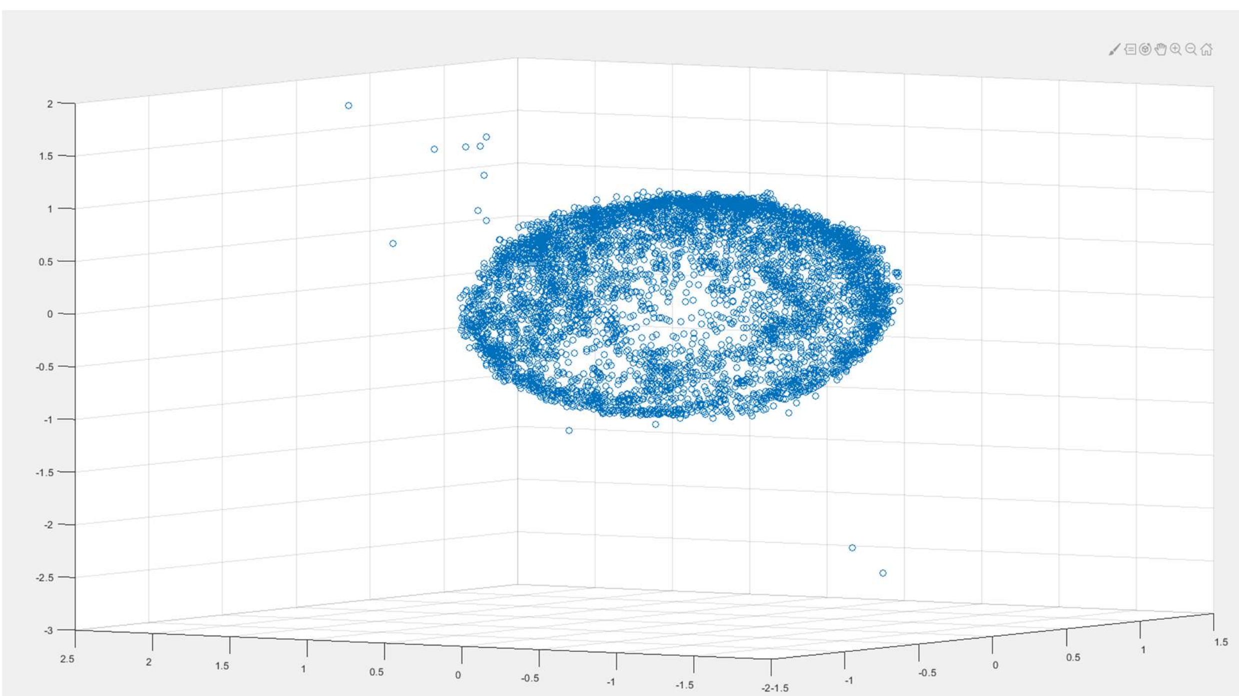

Norm of data pre- and post-Calibration
Pre-Calibration
Post-Calibration
800
700
600
500
400
300
200
100
0
0
1000
2000
3000
4000
5000
6000


0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
1.5
1
0.5
0
-0.5
-1
-1.5

With scale and shift




Norm of data pre- and post-Calibration
Pre-Calibration
Post-Calibration
700
600
500
400
300
200
100
0
0
1000
2000
3000
4000
5000
6000


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2 -1.5
-1
-0.5
0
0.5
1
1.5

Conclusion:

This lab was focused on calibrating each individual component of the MPU-9250 translating raw data with errors to useful normalized data. Each component had different levels of accuracy and error and I believe they can be used to calibrate each other for further accuracy because they all measure different things. Although there was not much work to be done in the lab, there was a lot of understanding and research needed. A lot of time also went into taking data. Overall, it was confusing lab but it was cool to see the end result in part 6 that built off the other parts.