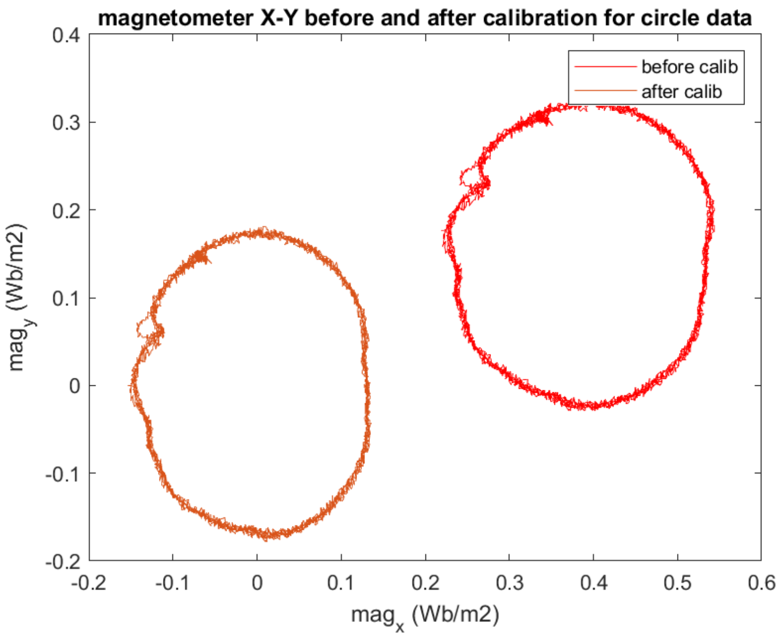
Driver code for my group is available on: goriparthi.t

We collected our circle data near ruggles station and the mini tour data near MFA. We drove the car around the circle for five times and I plotted the magnetic field about X vs magnetic field about Y for the circle data bag.



Then, we calibrated the data in such a way that makes the circle's origin coincident with the actual origin and transforms the ellipse into a circle. We apply translation, rotation, and scaling in that following order to calibrate the data. The exact formulae are given below:

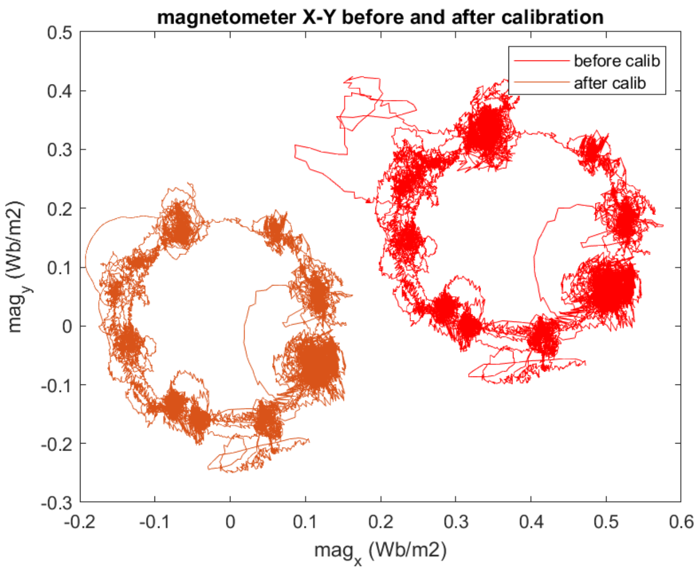
$$m_t = m - \begin{bmatrix} b_{H0} \\ b_{H1} \end{bmatrix}$$

$$m_{rot} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} m_t$$

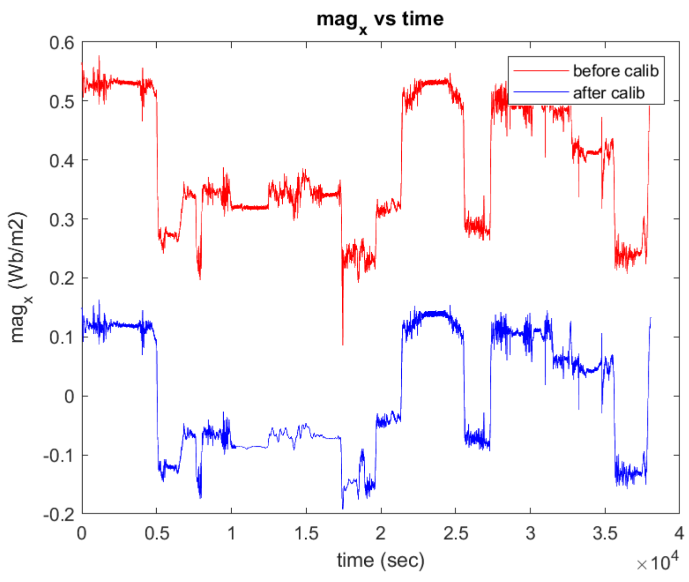
$$\sigma = \frac{b}{a}$$

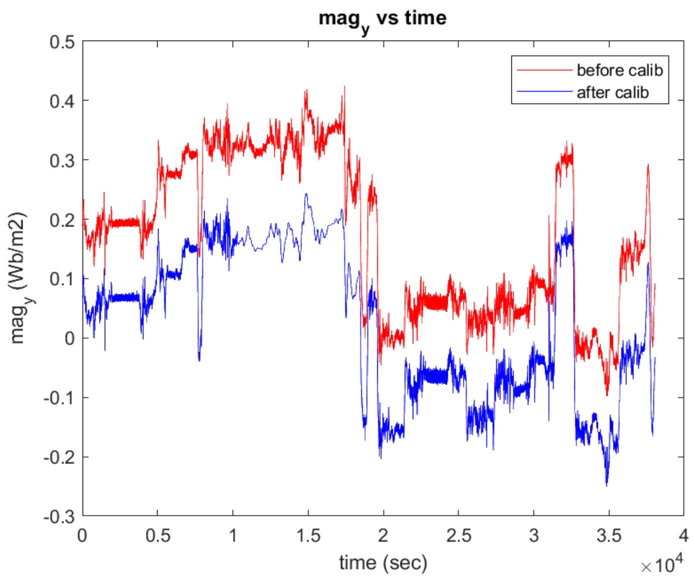
$$m_{scale} = \begin{bmatrix} \sigma & 0 \\ 0 & 1 \end{bmatrix} m_{rot}$$

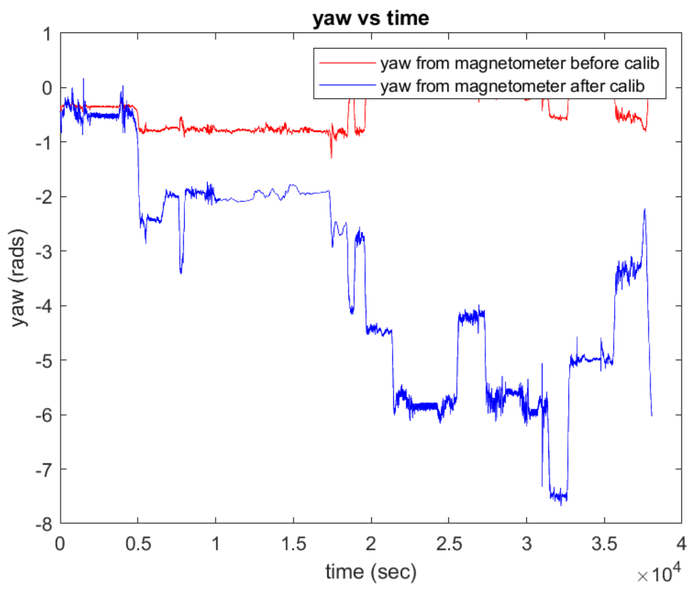
The figure above shows the calibrated and uncalibrated magnetic field data for circle dataset. This data contains errors because of soft-iron and hard-iron distortions. Soft-iron distortions are caused by iron which has low carbon content and can be easily magnetized and demagnetized. Whereas, Hard-iron distortions are caused by iron that can be difficult to demagnetize once it is magnetized. Soft-iron distortions distort the shape of the circle and hard-iron distortions shift the center of the circle from the origin.



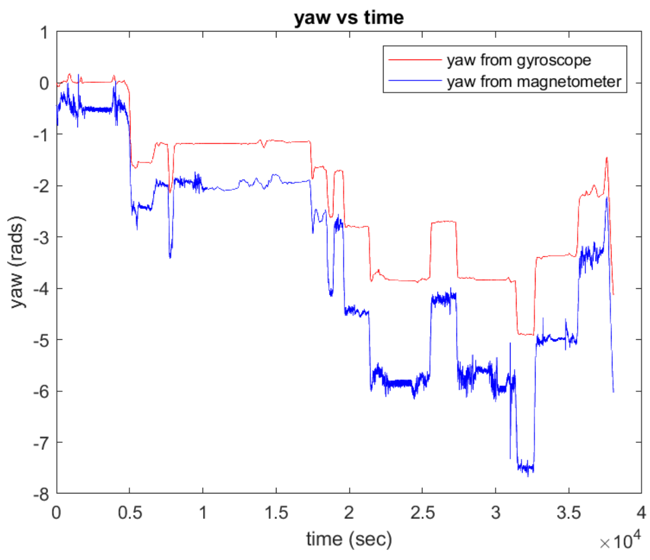
The above figure shows the magnetometer data before and after calibration for driving data. The below figures show the magnetometer data about X and Y axes before and after calibration.



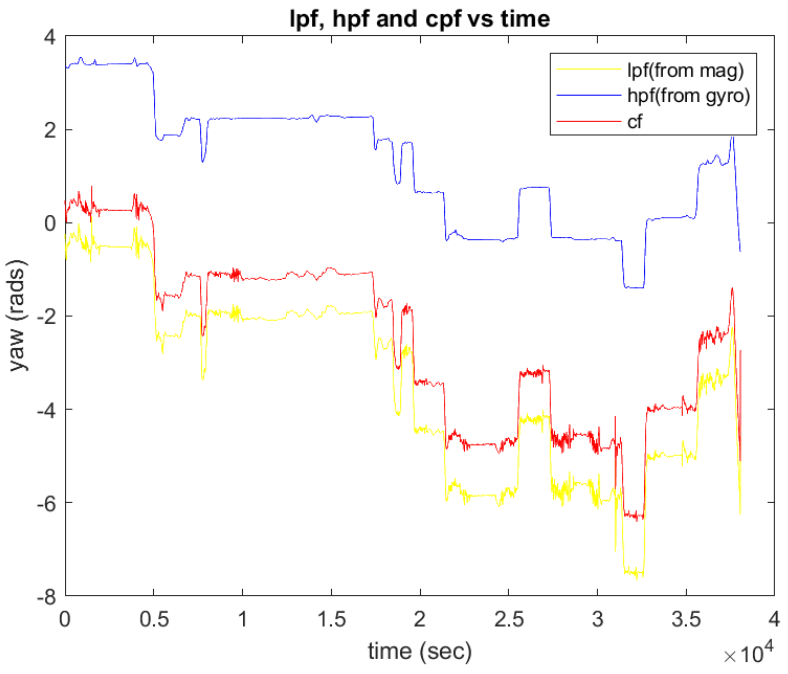




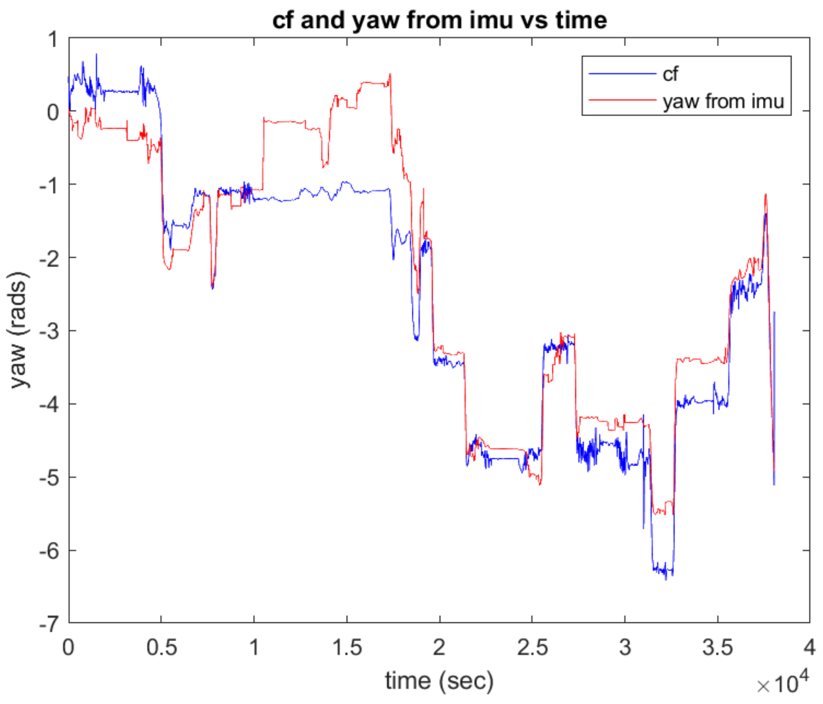
And then, we find out the yaw from mag_x and mag_y using the following formula: yaw = atan2(-mag_y / mag_x). We also get yaw from gyroscope data by integrating angular velocity about Z axis with respect to time. The below figure shows the yaw estimates from gyroscope and magnetometer together.



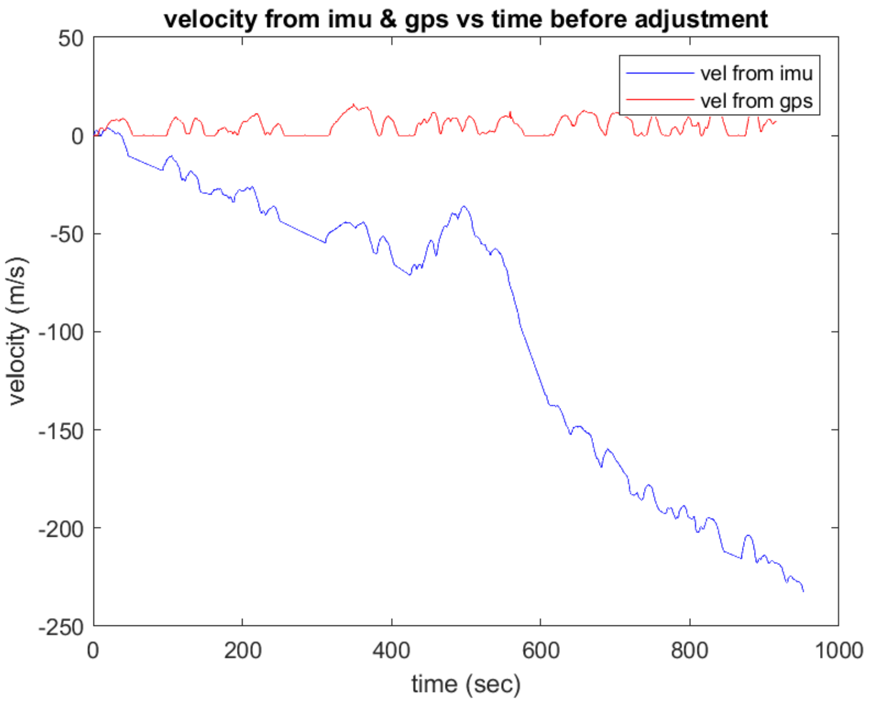
Then, I applied a high pass filter to gyroscope data and a low pass filter to magnetometer data. A low pass filter smoothens the curve and a high pass filter sharpens the curve. The cutoff frequency for the low pass filter is 0.01* pi rad/sample and for high pass filter is 0.00000001* pi rad/sample. After filtering both the yaw estimates, I applied a complementary filter to both the estimates to find the final yaw estimate. The formula for the complementary filter that I used is: yaw_estimate = (1 – alpha) * yaw_from_mag + alpha * yaw_from_gyro. The value of alpha is 0.2 in my case.



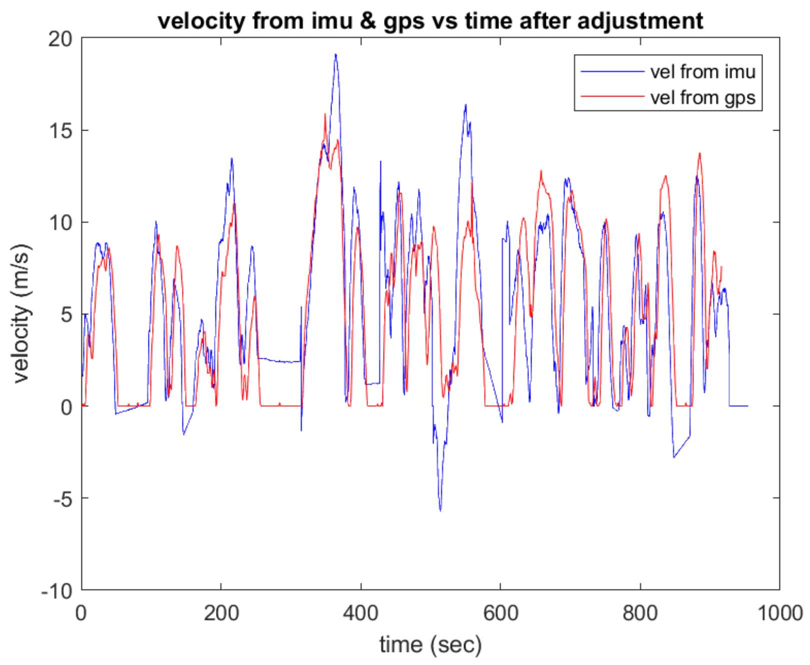
The above graph shows yaw estimates from magnetometer and gyroscope after low and high pass filtering respectively and complementary filter together. I compared this yaw estimate with the actual yaw we obtain from the IMU and the result is given below in the figure.



I would trust the yaw from the magnetometer more than the yaw computed from the gyroscope because the errors in the magnetometer can be calibrated easily by removing soft-iron and hard-iron distortions (and we did it in the first part) but it is way more difficult to do that with gyroscope because it has got angle random walk, rate random walk and bias instability. So, I chose the alpha value in such a way that the yaw estimate from complementary filter is close to the yaw estimate from the magnetometer data. Then, I estimated forward velocity from accelerometer data by integrating the linear acceleration about x. I estimated forward velocity from UTM northing and UTM easting values by differentiating them with respect to time and taking L2 norm of them.



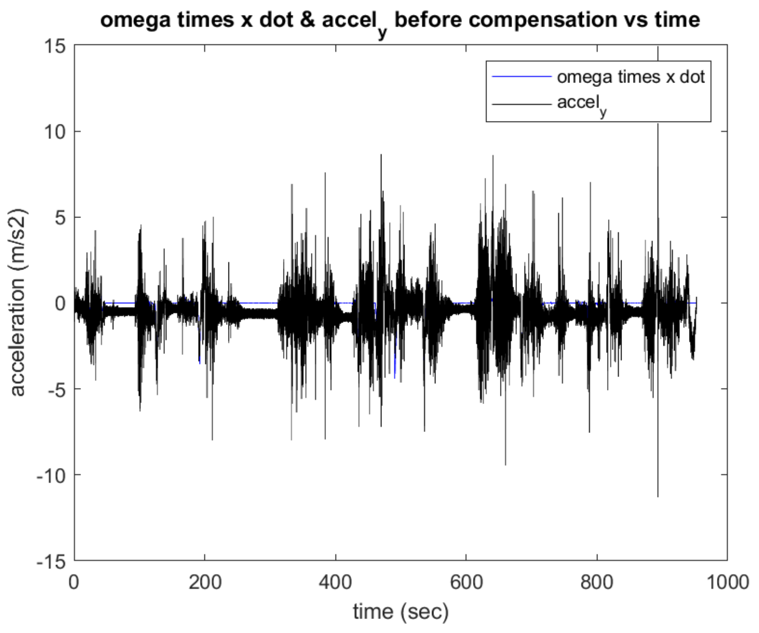
The velocity computed from IMU has got a lot of bias when compared to velocity computed from GPS. One of the reasons for this is because the linear acceleration about Y and Z axes have some contribution to the linear acceleration about X axis due to inclination i.e. pitch and roll. To remove this bias, I used the best fit line to estimate how the bias is varying and I subtracted this bias from the velocity curve. After removing the bias from the curve one time, there was still some bias left in the curve. So, I split the curves into six groups based on the change in slope(as in, each group of data has a monotonic increment or monotonic decrement in slope) and applied the best fit line method again for all the groups and subtracted the biases from their respective groups. Finally, the velocity of the IMU and GPS after adjustment is shown in the below figure.



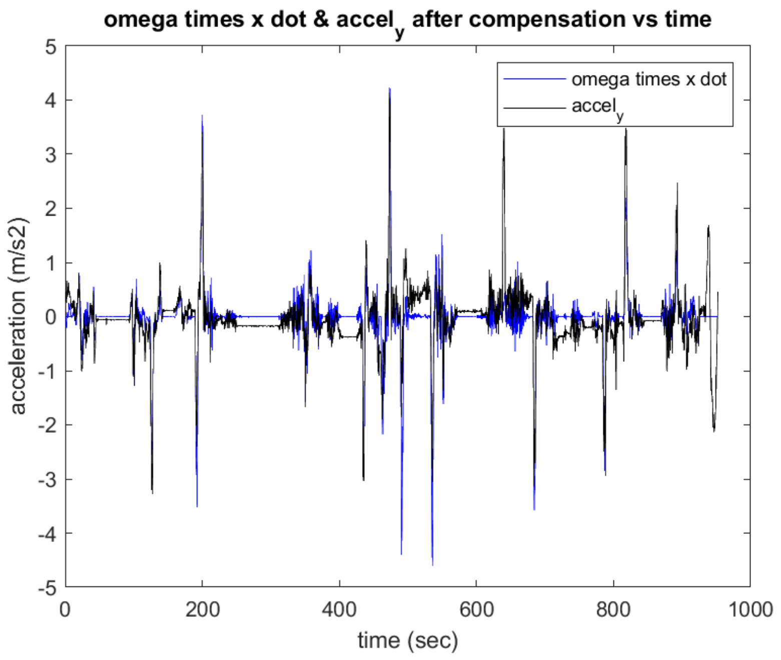
Then, I calculated the linear acceleration about X and Y axes from the angular velocity about Z axis and forward velocity using the following formulae:

$$\ddot{x}_{obs} = \ddot{X} - \omega\dot{Y} - \omega^2 x_c$$
$$\ddot{y}_{obs} = \ddot{Y} + \omega\dot{X} + \dot{\omega}x_c$$

Where, Y_dot = 0, x_c = 0 and Y_double_dot = 0. Then, I compared the calculated acceleration about Y axis against the obtained acceleration about Y axis.



As you can see, the calculated acceleration is very different from the observed acceleration because the calculated acceleration (omega times x dot) is compensated for the Y component of the acceleration about X and Z axes but the observed acceleration about Y axis contains the effects of acceleration due to gravity. So, I mitigated the effects of acceleration due to gravity by using a low pass filter on the observed acceleration and I got the following graph:



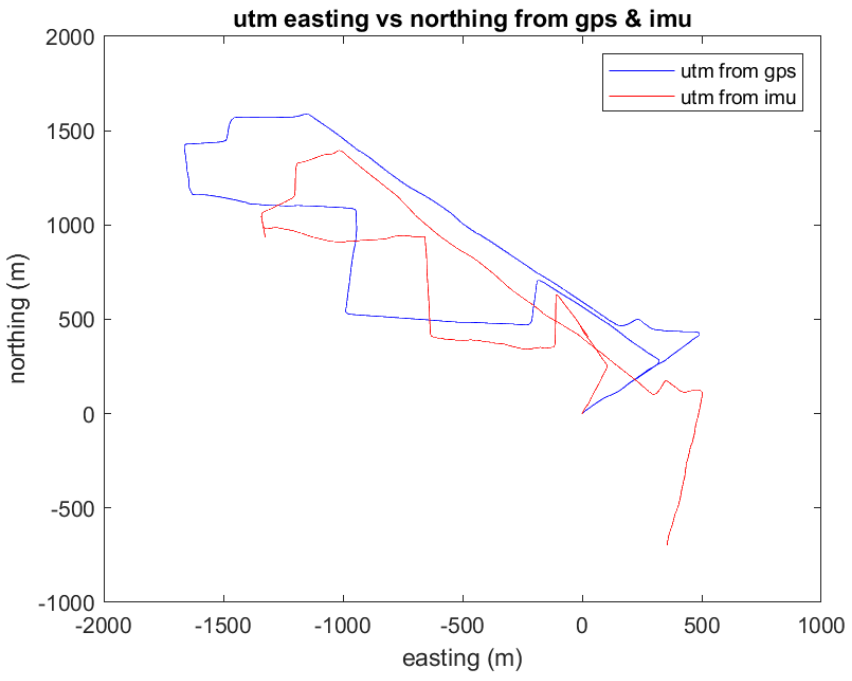
The observed acceleration with compensation is closer to the calculated acceleration. Later, I am calculating horizontal and vertical components of velocities from estimated yaw and velocities from part 1 and 2 using the following formulae:

```
v_e = vel_imu .* cos(yaw_compl);
v_n = vel_imu .* sin(yaw_compl);
```

Then, I integrated the velocities with respect to time to obtain the easting and northing displacements and I am comparing the calculated displacements with the displacements from GPS data. I used the following scaling factors:

```
x_e = x_e * min(utm_northing) / min(x_e);
x_n = x_n * min(utm_easting) / min(x_n);

utm_easting = utm_easting * 1.5;
utm_northing = utm_northing * 1.5;
```



As you can see, the displacements obtained from the IMU are slightly different from the ground truth displacements. This is because I used estimated yaw and velocities from IMU data to calculate the UTM displacements. And, these estimated yaw and velocity have deviations from the ground truth velocity and yaw. These calculations are subject to compounding errors, as in, even if there is a small bias at one of the timeframes, that bias will continue to propagate and become larger as the time goes on.

It is given in the vn100 vectornav datasheet that the magnetic heading accuracy is 2 degrees and gyroscope's noise density is 0.0035 degrees/sec/(Hz)^0.5. We would expect the UTM values from IMU to match the UTM values from the GPS for at least half of the path travelled without a position fix. But it is not case because there are deviations in velocity and estimates from the ground truth values. They deviated right from the beginning (t = 0s). But we see that the UTM from IMU is following the same trend as the UTM from GPS. The stated performance for dead reckoning did not match actual measurements because of the deviations in estimated velocity and yaw discussed above.