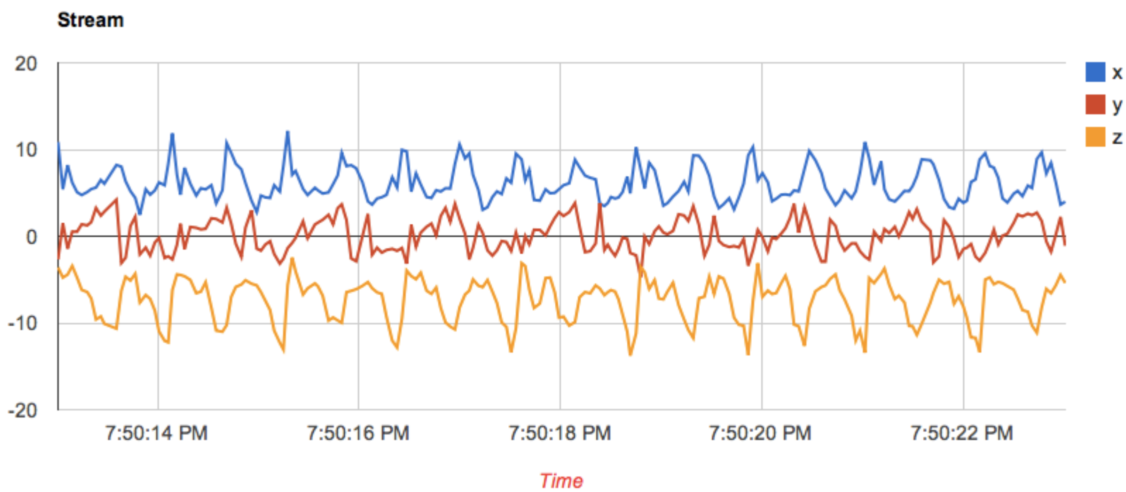
Ze He, Nam Phan, Andrew Wang

CS 390MB Data Analytics 1 Part A

Group 6

Five different filter scenarios

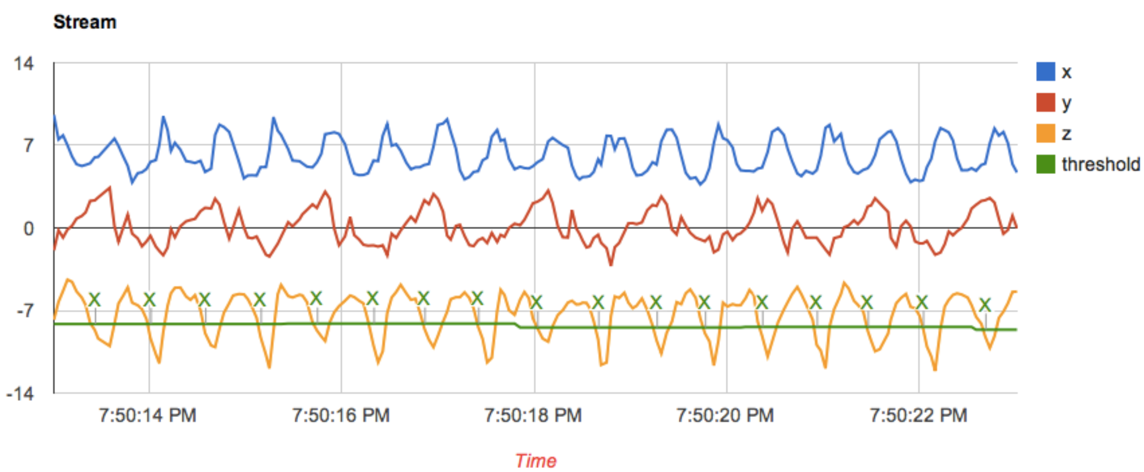
The raw data visualization



Filter type: Smoothing

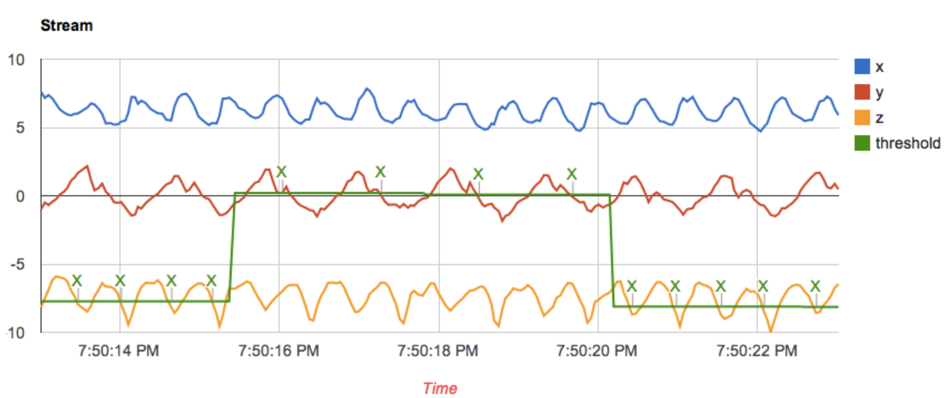
Smoothing factor: 2

Step Count: 3897

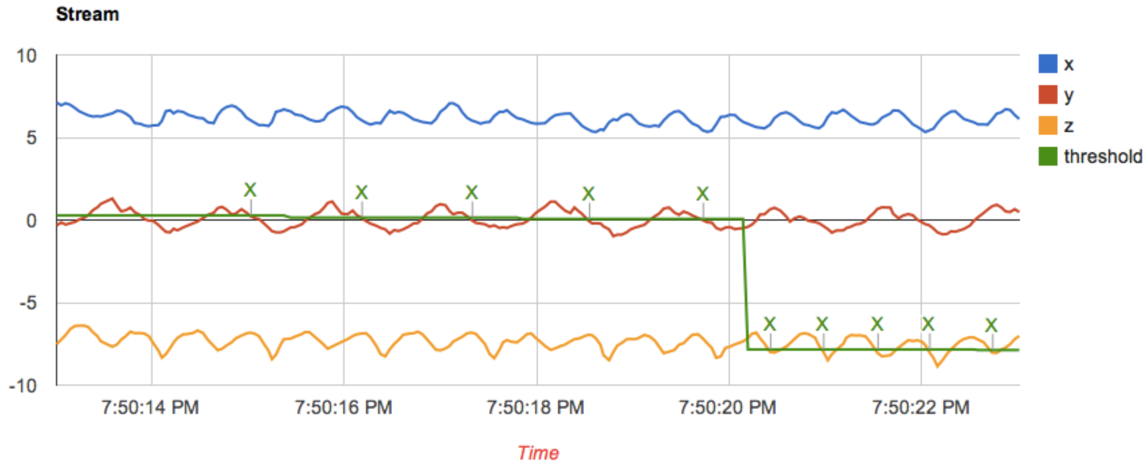


Smoothing factor: 5

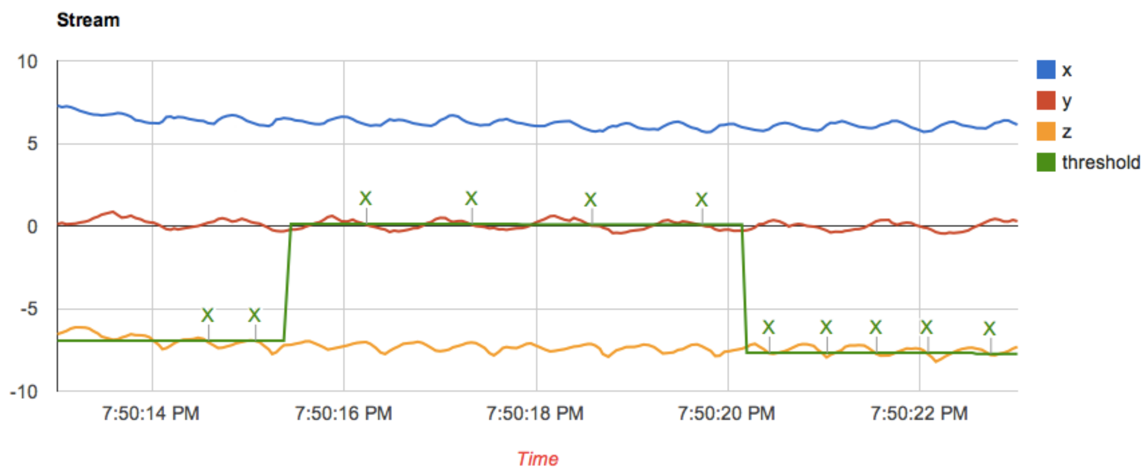
Step Count: 3779



Smoothing factor: 10
Step Count: 3711

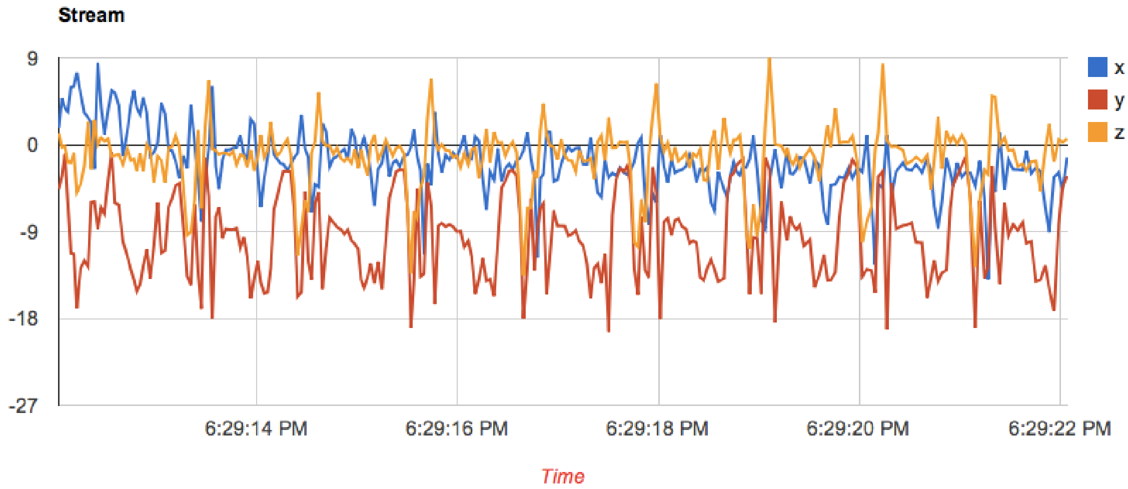


Smoothing factor: 20
Step Count: 3369

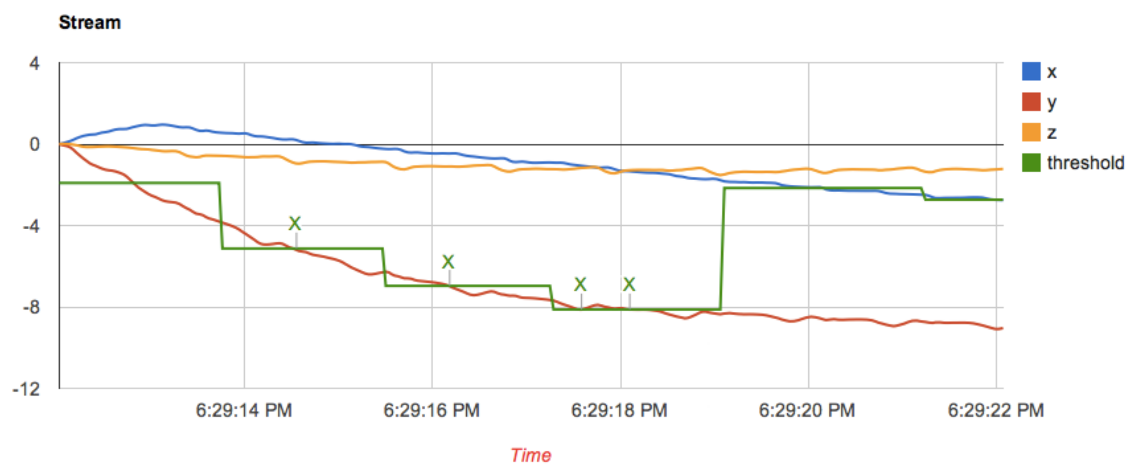


Comparing across all scenarios, it seems that for smoothing filter, when the factor is set at a relatively low level (<5), the filter will not be able to remove high frequency noise. However, if the factor is too high (>10), the time domain signal may get lost and become overly flat. The “high” and “low” is relative, and the number is only applicable to this specific case.
Therefore, for smoothing filter, it may be good to set the factor between 5 – 10.

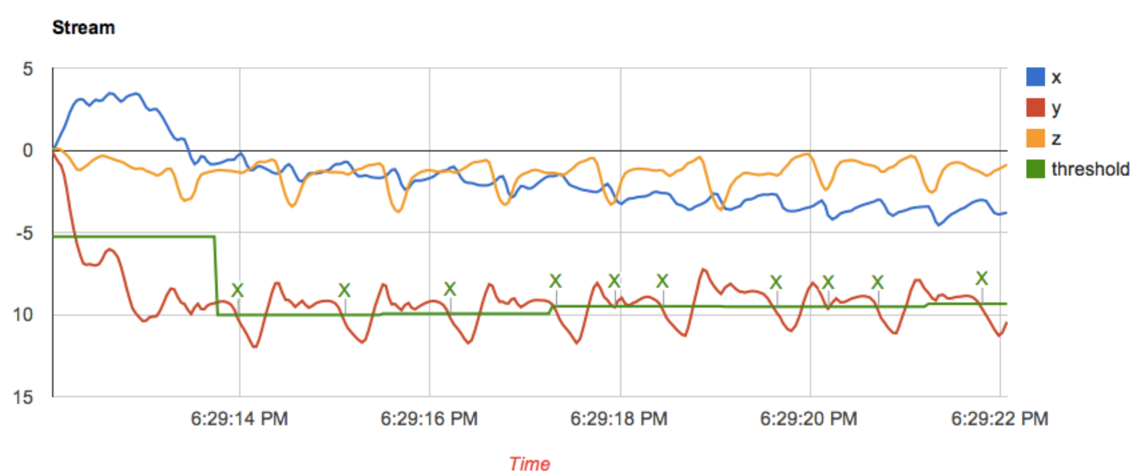
Filter type: Butterworth
Raw data visualization



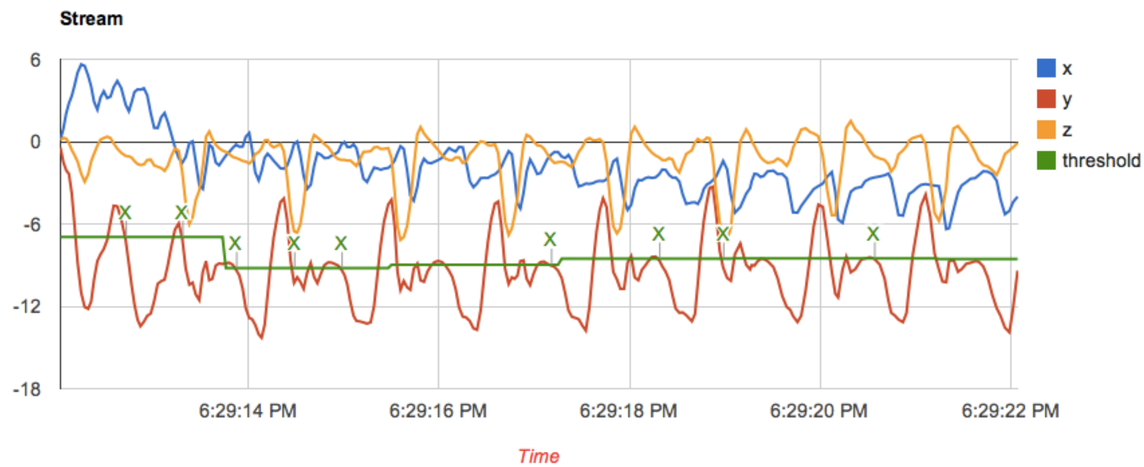
Cutoff Frequency: 0.3
Step Count: 2471



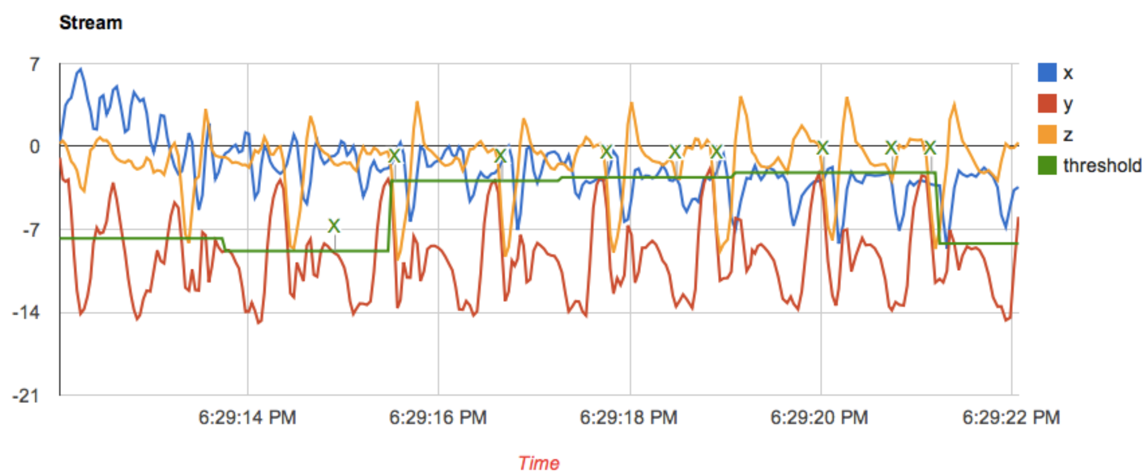
Cutoff frequency: 1
Step Count: 3641



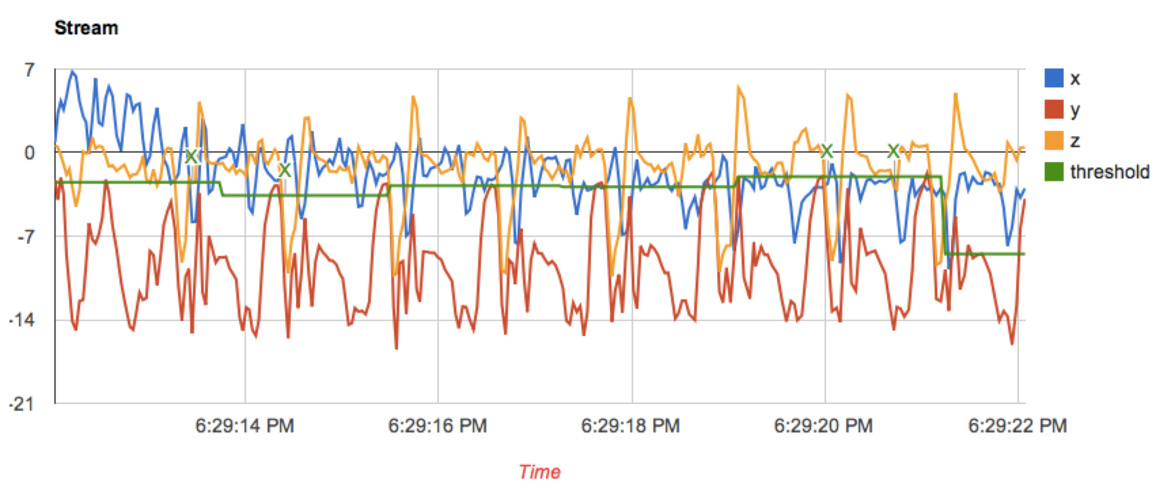
Cutoff frequency: 2
Step Count: 4324



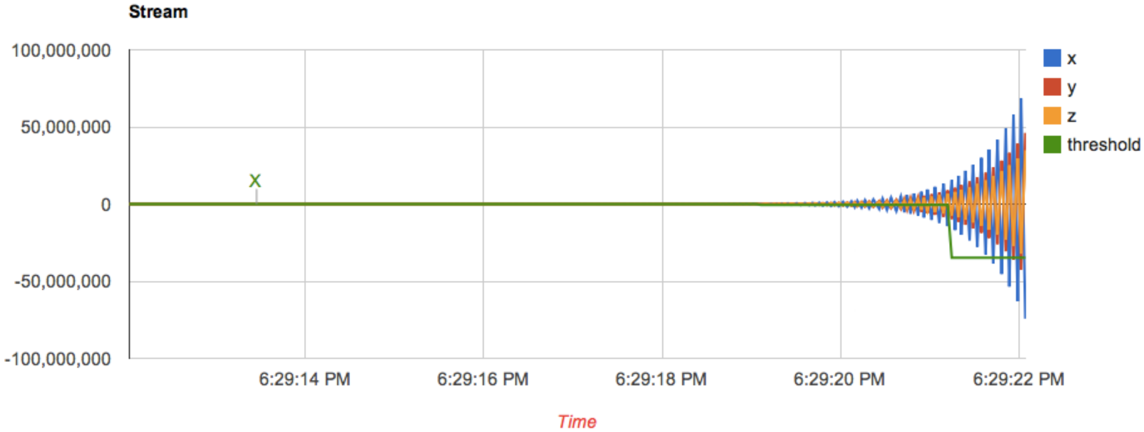
Cutoff frequency: 3
Step Count: 4104



cutoff frequency: 4.6
Step Count: 3928

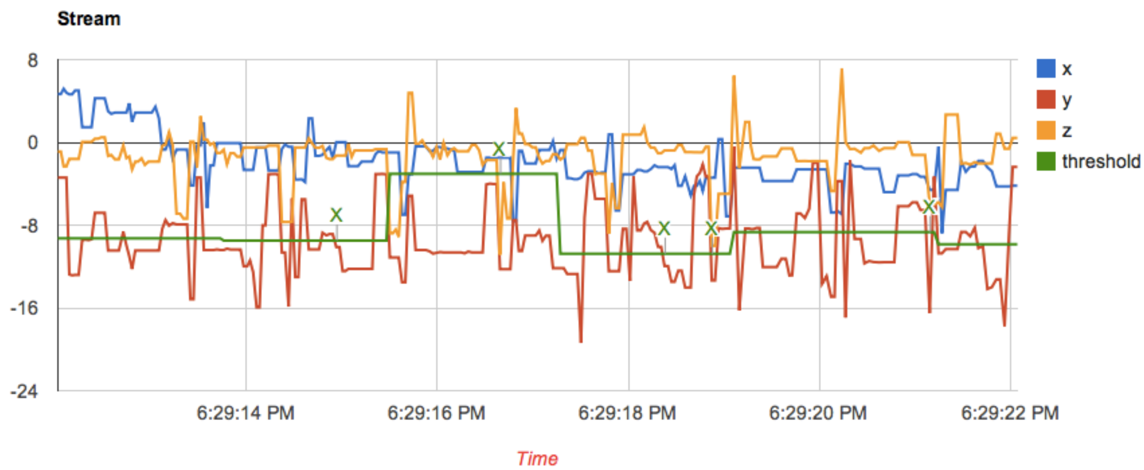


Cutoff frequency: 4.7
Step Count: 2



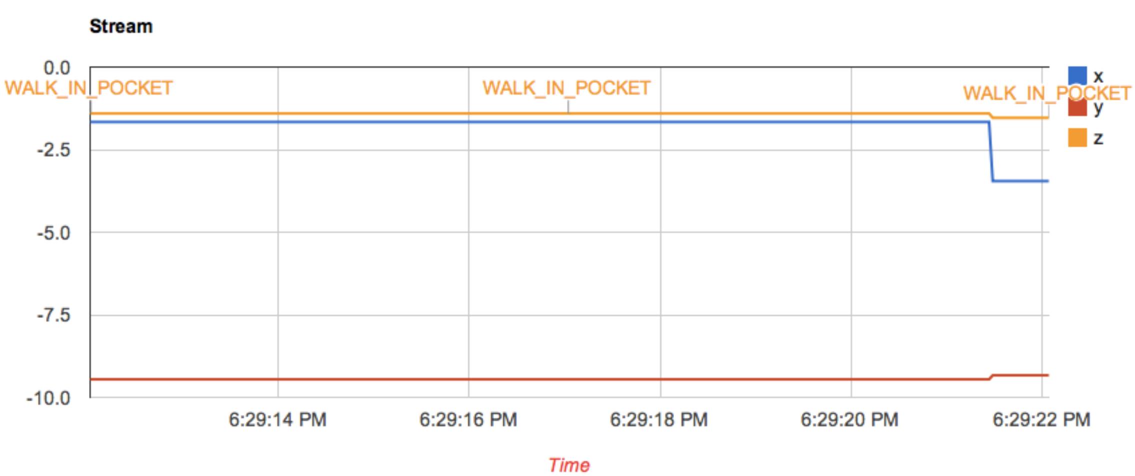
When the cutoff frequency is too low, the signal is overly flattened, and step information gets lost.
However, if the frequency is set to high (>3), then the results become noise and may be meaningless (>4.7).
Therefore, the best results may be produced with frequency set between 1-3.

Wavelet-Gaussian

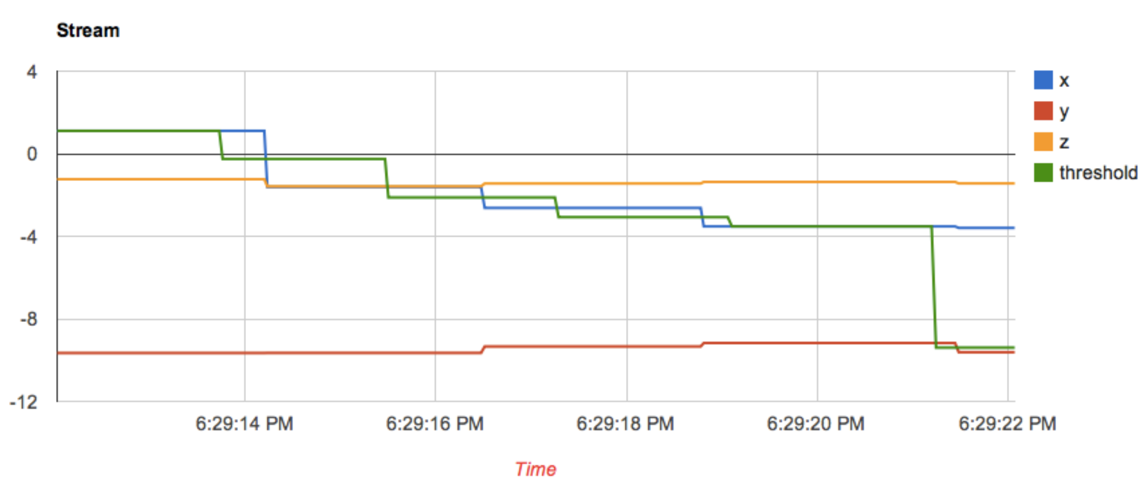


It seems that Wavelet-Gaussian filters out more signals, and misses many step detections. It may not be ideal for this task.

Wavelet Compression
Level: 2
Step Count: 0

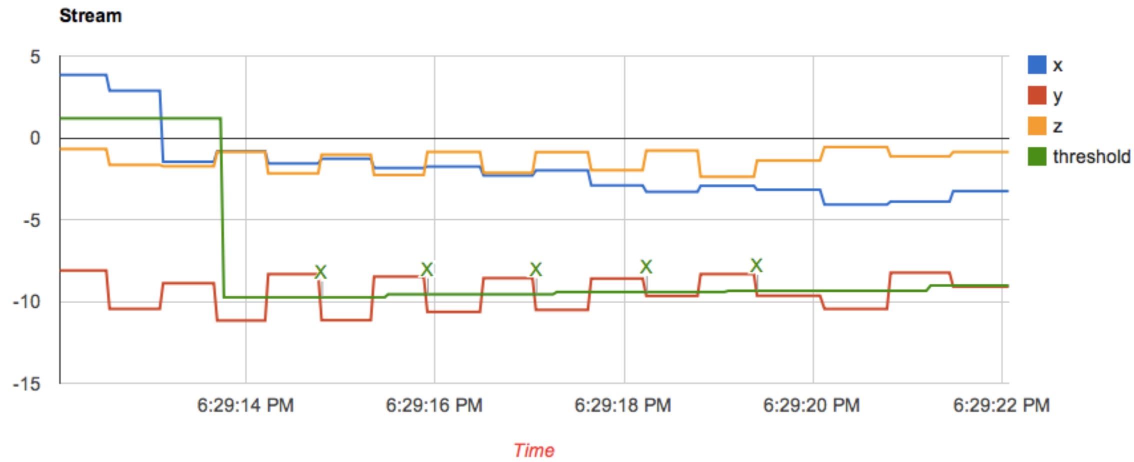


Level: 4
Step Count:3



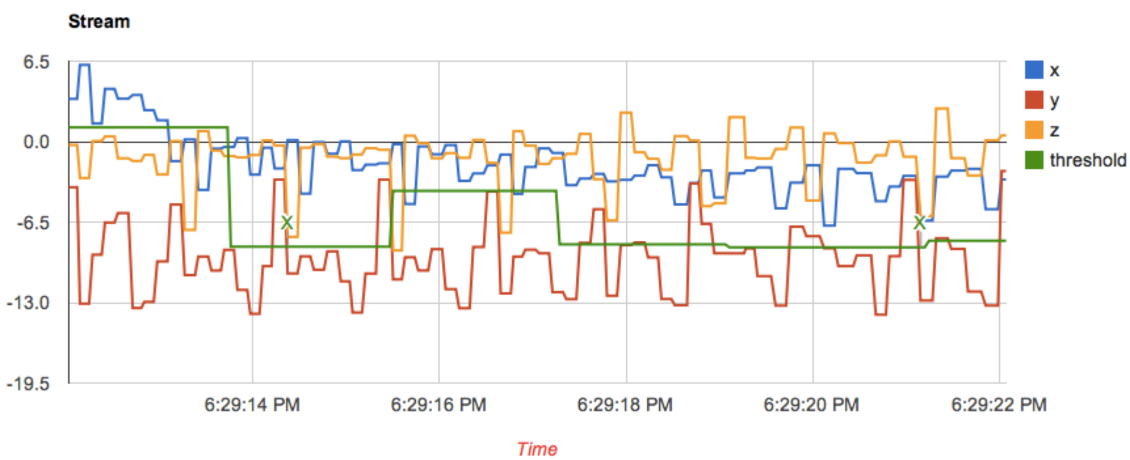
Level 6

Step Count: 867



Level 8:

Step Count: 4463



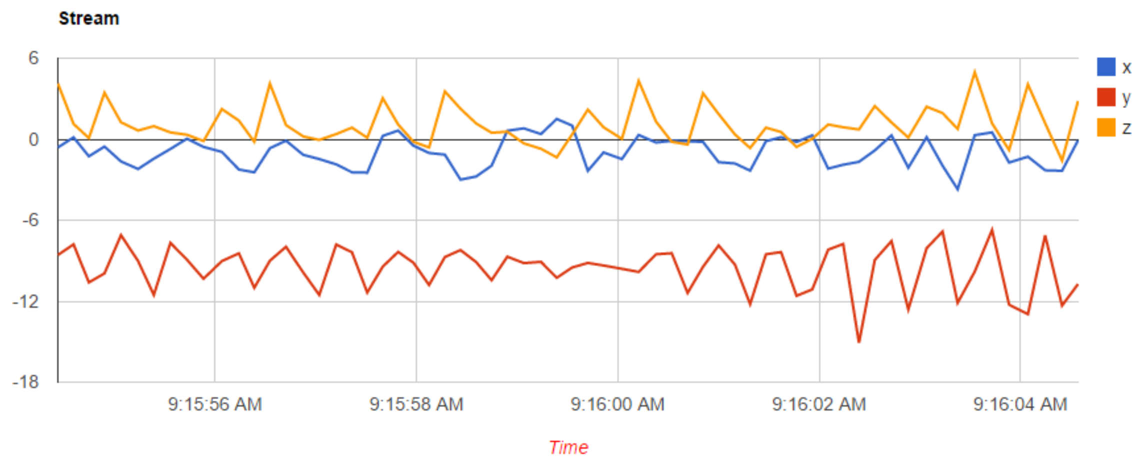
When the level is set too small (<4), the signal is overly flattened. Therefore, the level should be set above a certain level (>6).

Comparing across all filters, if the parameters are set properly, Butterworth and Smoothing may produce similarly good results using this step detection algorithm, because the steps detected after applying other filters have higher miss rate even with the seemingly best case.

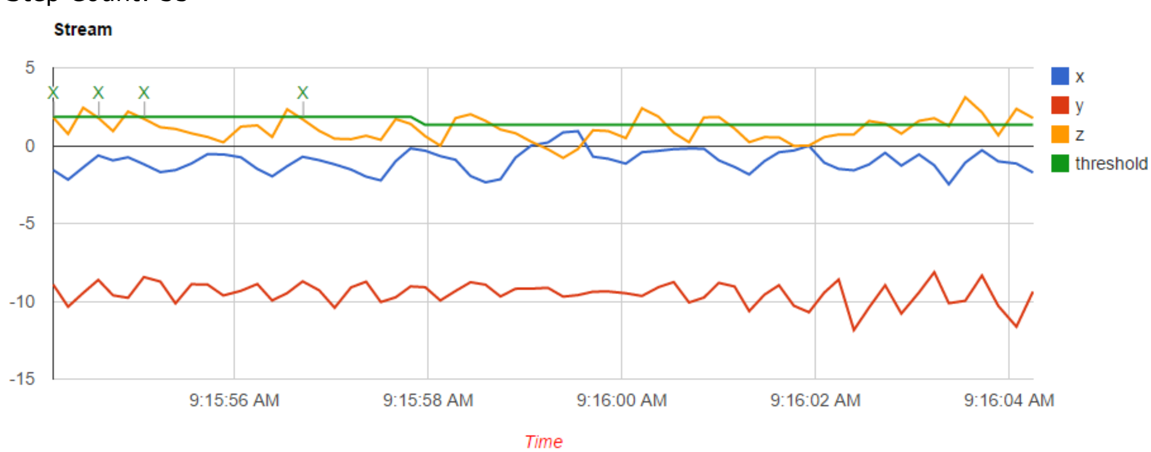
Part B

Phone in Bag:
Importer: Nam Phan
Phone config: Phone in Bag

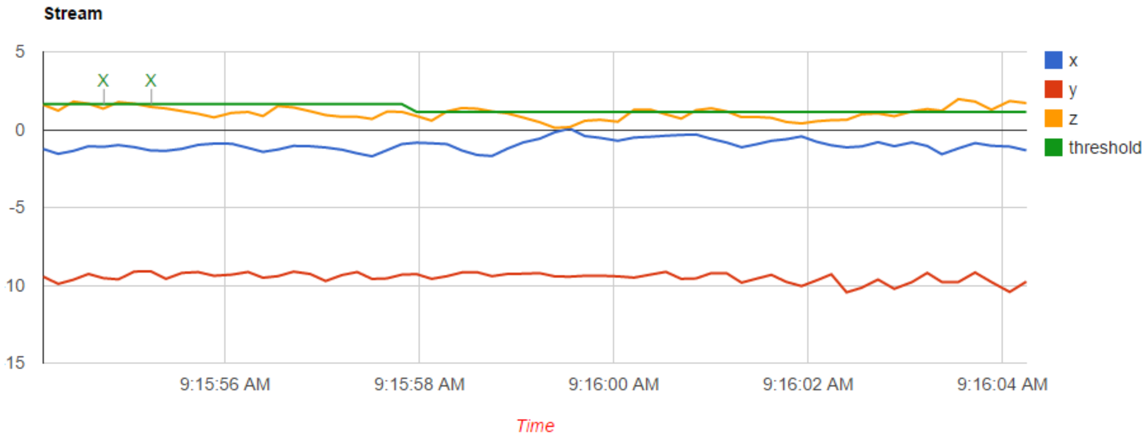
Raw:



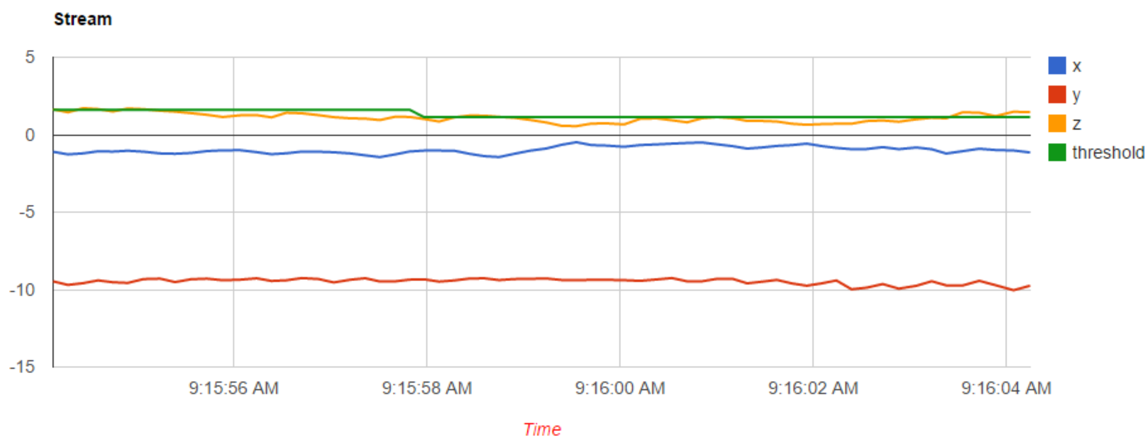
Smoothing Factor: 2
Step Count: 35



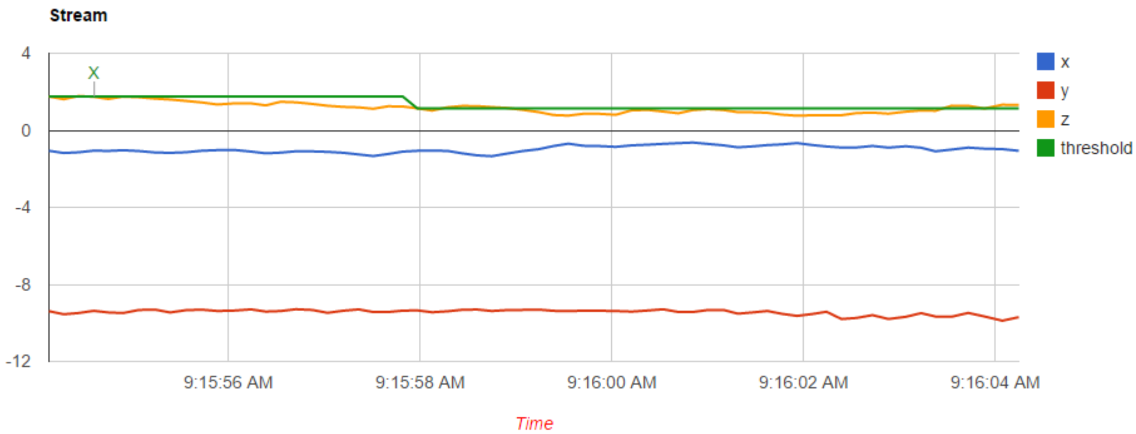
Smoothing Factor: 5
Step Count: 15



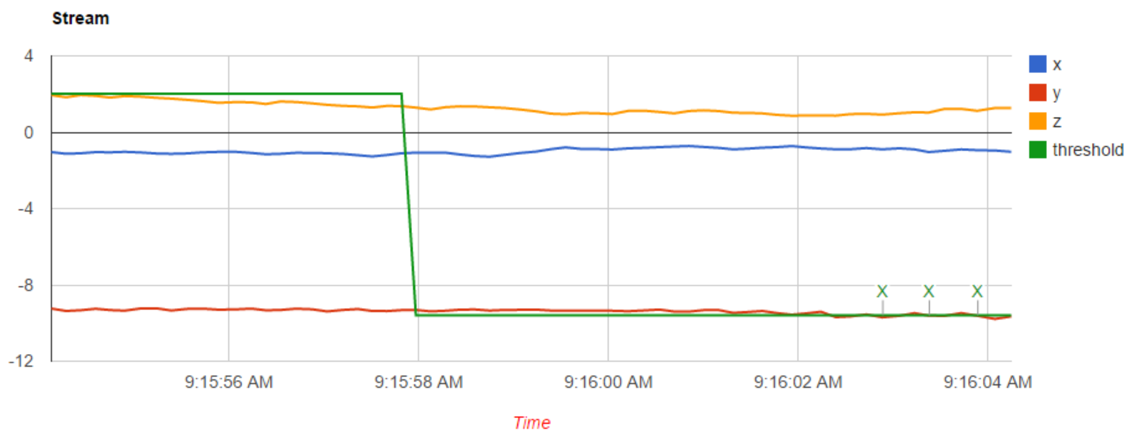
Smoothing Factor: 10
Step Count: 10



Smoothing Factor: 15
Step Count: 5



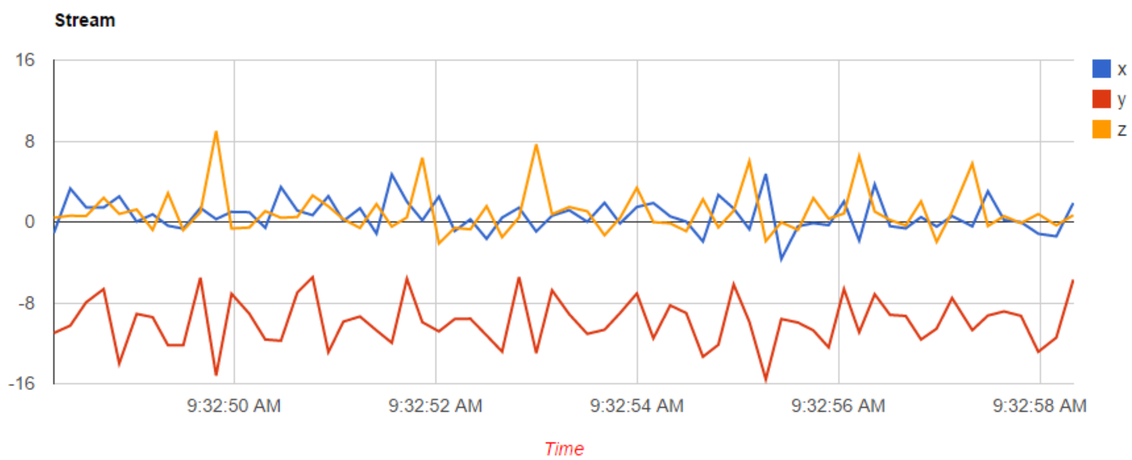
Smoothing Factor: 20
Step Count: 5



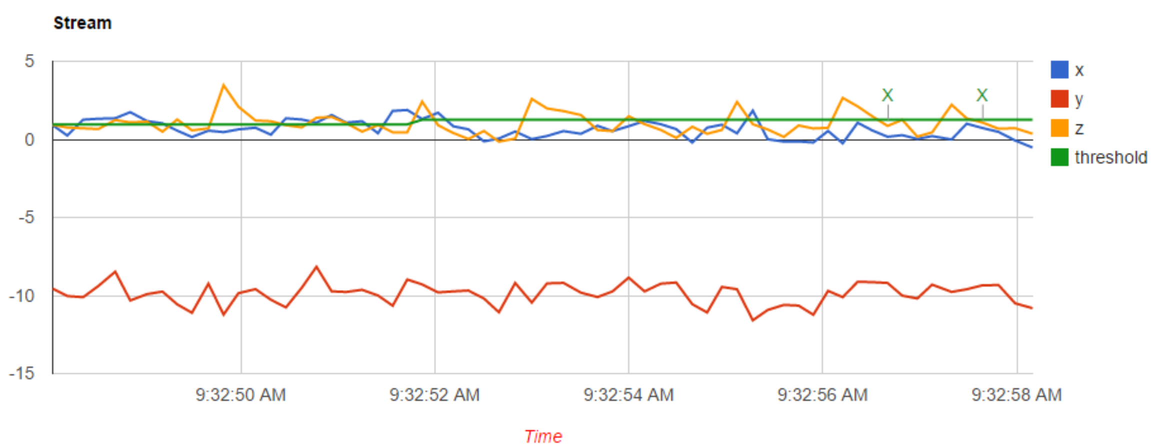
The Phone in bag configuration was very interesting compared to a phone in pocket configuration, because physically the phone synchronized with the bag movement, which means that unlike phone in pocket(front pocket) it did not exhibit leg movement. Rather it exhibited more up and down movements than right and left movement, respectively more X than Y. This allowed us to focus on a person's vertical movement, and ignoring over influencing Y movements of the leg. With this configuration, steps can be detected by setting a threshold for X and counting every negative intercept. However, this doesn't mean that Y values are not important, because they can provide more indepth intuition on detecting steps.

Config 2: Phone in Pocket
Imported by Ze He

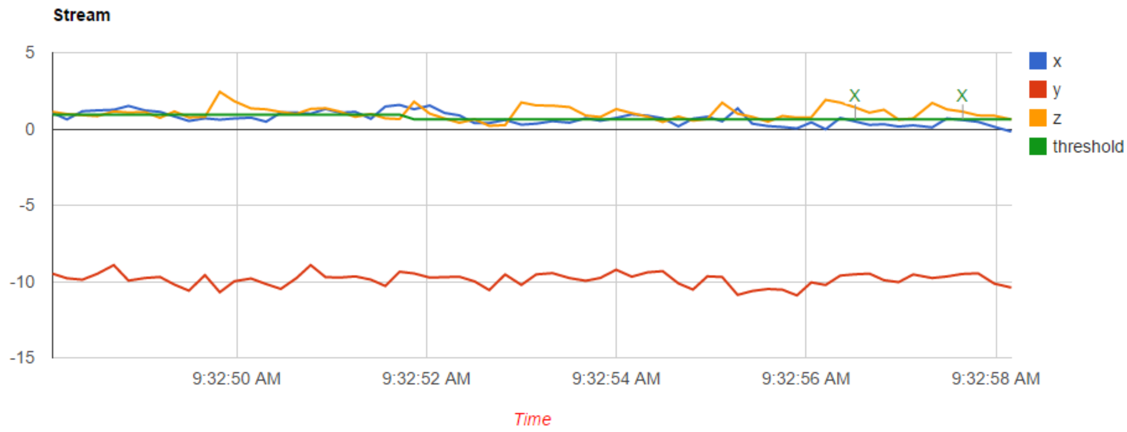
Raw



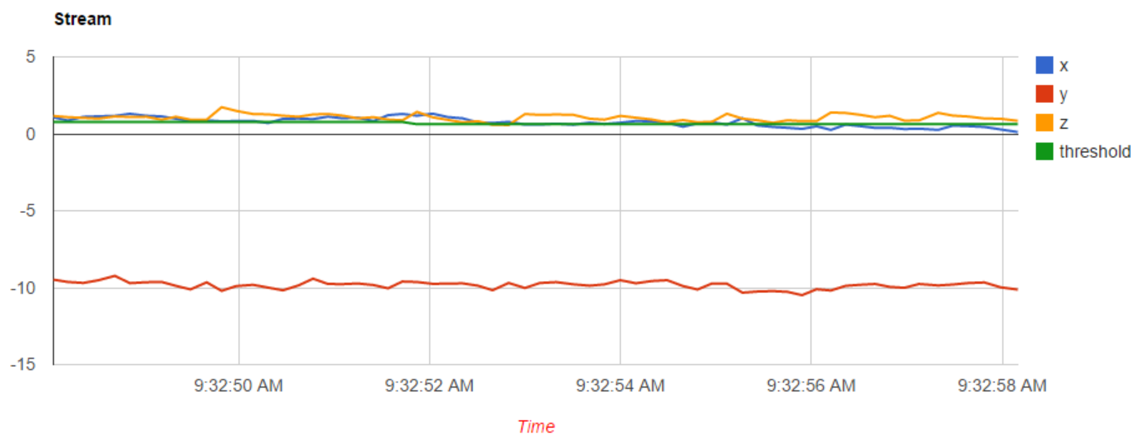
Smoothing Factor: 3
Step Count: 44



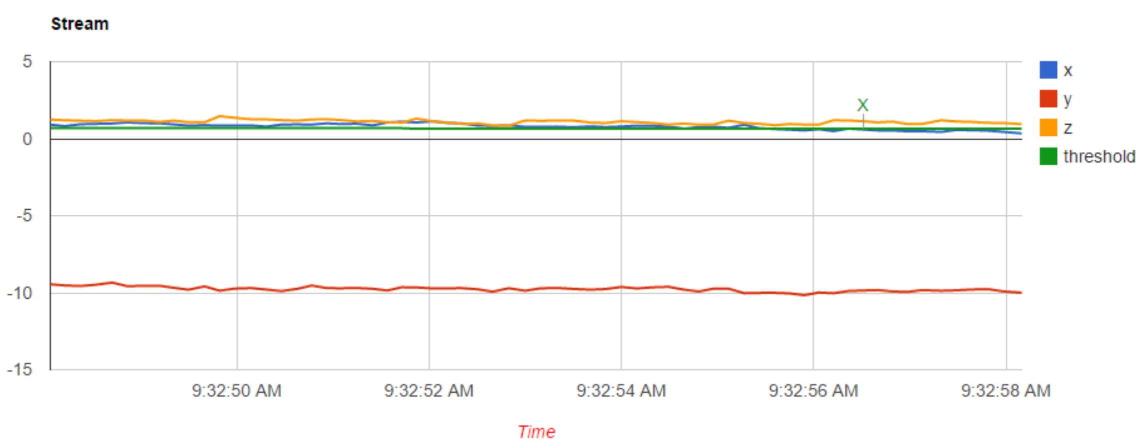
Smoothing Factor: 5
Step Count: 31



Smoothing Factor: 10
Step Count: 9



Smoothing Factor: 20
Step Count: 5



The Phone in pocket configuration differs very much from phone in bag configuration in that the frequency of peaks from the Y value is much more often. This is probably due to the back and forth movement of the leg. Analyzing the data, we notice that the step count becomes extremely unreliable with a smoothing factor of 5 or higher. The graph becomes too smoothed out to give meaningful data.