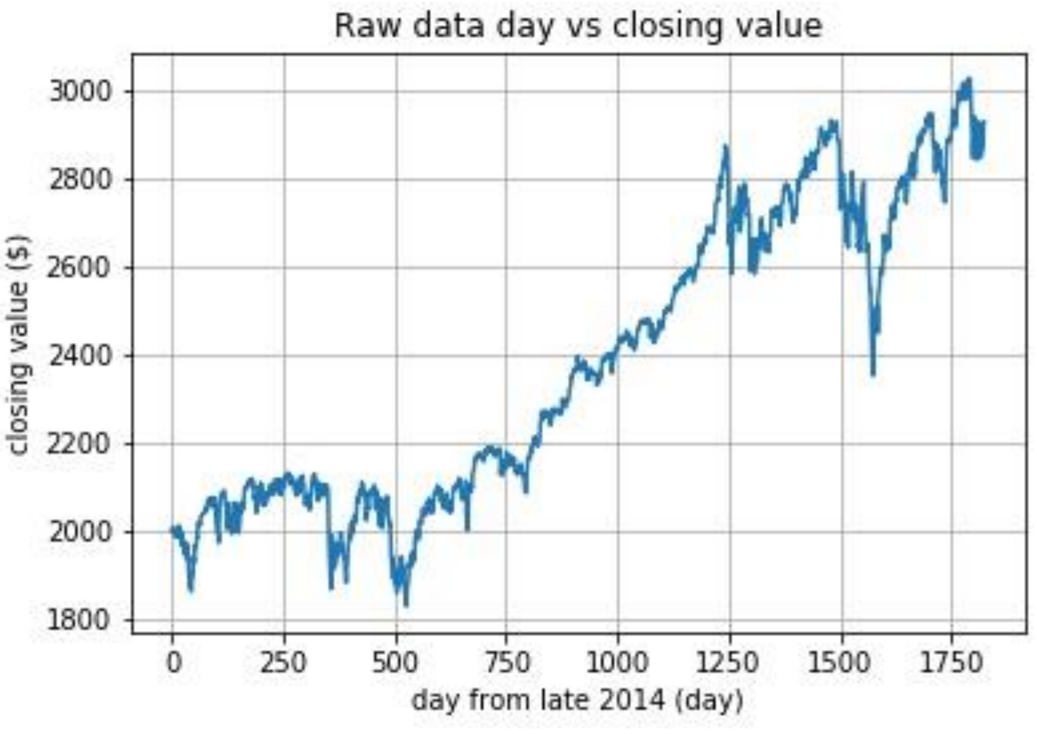
PHY407 Lab Assignment 5

Vincent Fan 1002563380
Satchel Page 1001695686

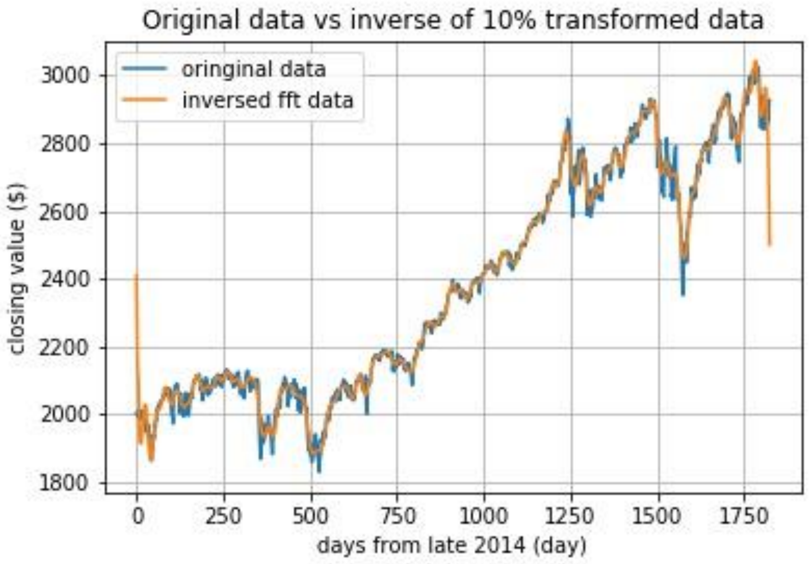
Question 1: Fourier Filtering and Smoothing

a) For this question, the goal was solely to import the data provided in the sp500 file and plot it on a graph. Because data wasn't provided for weekends and holidays, and therefore skipped over some days, we had to be careful to define other periods of time accordingly. For example, a month was calculated to be approximately 22 days by averaging the amount of days per month, and subtracting the average number of weekend or holidays from there. The plot of the raw data was the following:



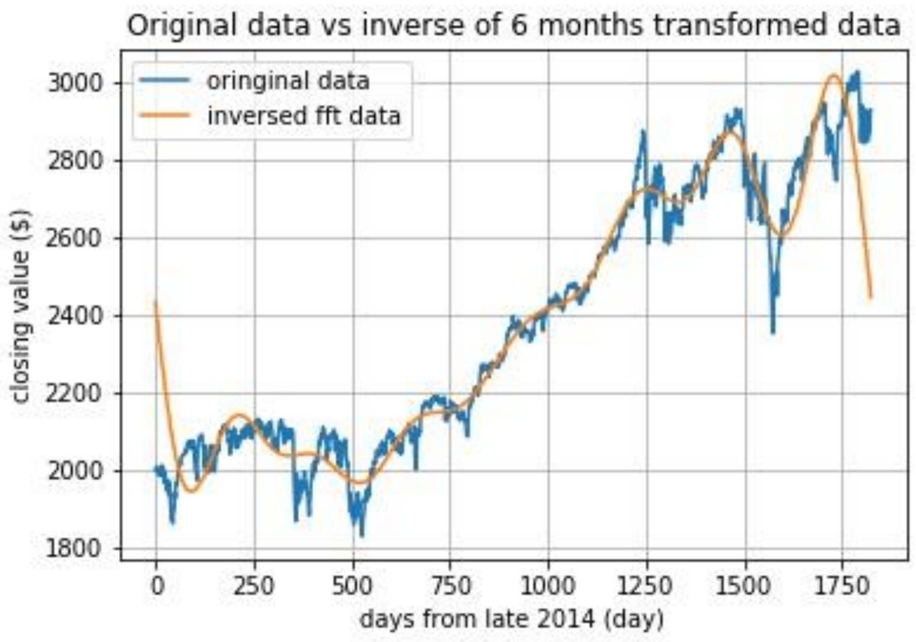
b) Here we just wanted to calculate the coefficients of the discrete/fast Fourier transform of the data with the rfft function. We included an inverse function to ensure we end up with the initial data set when applied to the transformed coefficient data set. The array of coefficients is labelled close_fft in the code. **Nothing more to submit here.**

c) For this question, the idea was first to zero out all but the first 10% of our values for the Fourier coefficient array, then inverse the FFT and plot the new data set on the same graph as the one in part a to compare. Doing so yielded the following plot:



Contrasting the two, the data that's been transformed and inverted fits the original pretty well, but we notice a much smoother trace. Oscillations are far less sharp or defined, and the values reached by the 10% curve are less extreme this way.

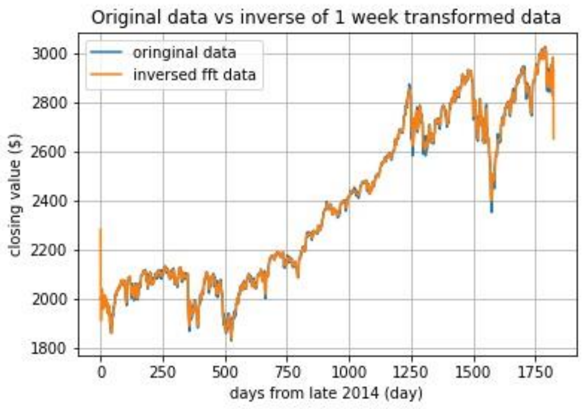
d) The idea for this question is to modify our function again by implementing a low-pass filter, which filters out variation up to a period of 6 months. Abstracting 6 months t on average be around 180 days, and removing holidays and weekends we have a period of around 130 days total. This gives a frequency of 1/130 = 0.0077 d^{-1}. Plotting our low pass filtered function against the original yields the following graph:

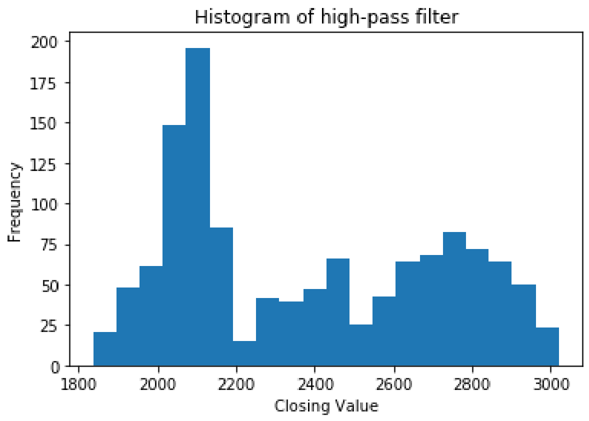


The reason for the far smoother curve generated by the filtered function is simply because the variation through periods of 130 days or more has been filtered out. This means that while the general path is conserved, the trace lacks any variation between these frequencies, which eliminates most of the local maximums and minimums in the original data through the curve. The reason for the initial spike at the left boundary is that the first value of our FFT array is the sum of our FFT values, meaning it starts quite high. The dip at the edge boundary is just the continuation of the trend that was happening a period of around 130 days before. Since there is no final point to trace, it just continues along its path until the boundary because the variation has been minimized to ignore a fair amount of end points.

e) For this next question, we want to do a similar modification to our function to the one done in the previous question, only this time were going to filter out variations which are **below** a period of 1 week. This means keeping all frequencies above a defined minimum of 1/week. From our previous estimates, given we have no data for weekends, we define a week to be 5 days, and have a value of 0.20 d^{-1}.

The plot of this new filtered data, as well as a histogram representation are shown in the following plots:





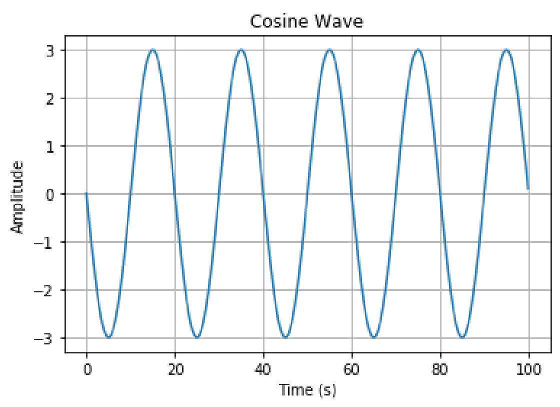
We can clearly see that this high-pass filter traces the data much more precisely than the previous low-pass filter, which makes sense because here we are only filtering out large variation, while leaving room for small fluctuations, allowing for a very accurate trace.

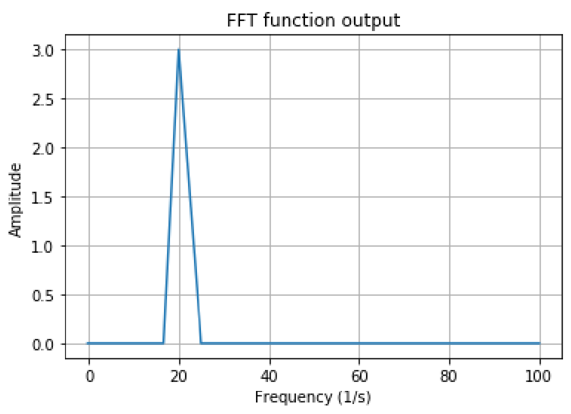
The histogram shows a modal value around 2100, and a positive skew around that point. This suggests that the stock index is increasing past some consistent average. Over short time periods, we can also see few jumps where there aren't many values in between two significant ones (ie 2100 -2300). This means that over small time intervals, the increases are steady rather than extreme.

Question 2: Analysing air pressure measurements on Mars
For this question, we are going to be using data from the Mars Science Laboratory about surface air pressure measurements over 2 Mars years (2014-2018).

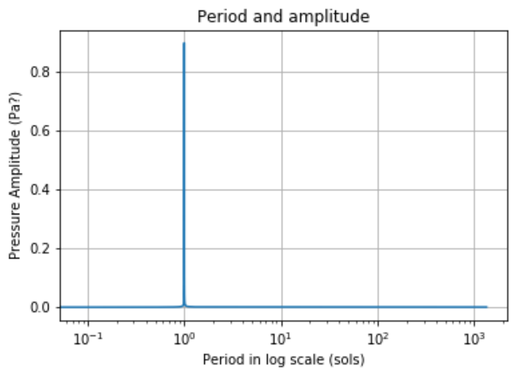
a) The goal for this question is to write a function which takes a data series with constant time intervals and performs a Fourier analysis on the data. We want it to calculate the FFT of the inputted dataset, and return the amplitude, period and phase of waves within it.

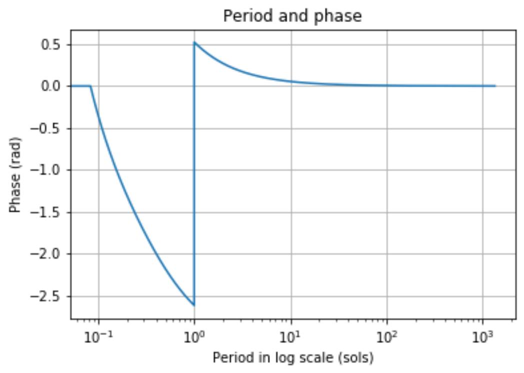
After creating the function, we defined a cosine wave to test the function against. The values for amplitude, period and phase are known for our cosine wave, the comparisons with the outputted values of the FFT and the originally created cosine wave are the following:
fft amplitude: 3.0 original amplitude: 3
fft period: 20.0 original period: 20 sol
fft phase: 0.25 original phase: 0.25 rad





b) **Nothing to submit for this part. But we calculated the amplitude and phase for set A:**





c) In part b), we performed a Fourier analysis of the data found in pressure field A. Using the same analysis method, we now want to analyze pressure field B, and find the Amplitude, Phase and Period of the 3 waves in the dataset.

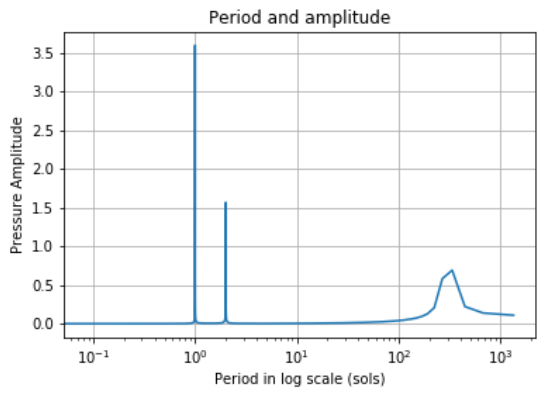
The values obtained by the code for the amplitude, period, and phase of the three waves are displayed in the following output:

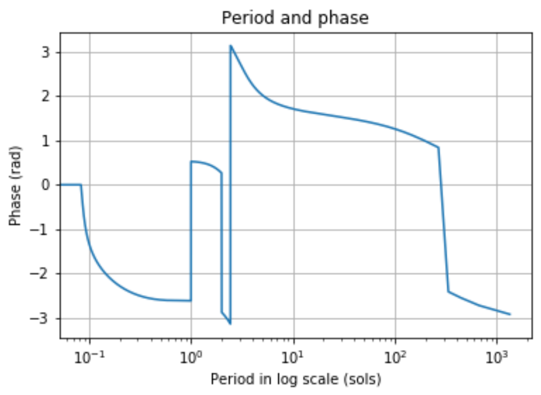
```
Peak [1]:
amplitudes: 3.602253180062382
period: 1.00018698578908 sol
phase: 0.5235902876188429 rad

Peak [2]:
amplitudes: 1.5698512761872043
period: 1.998878923766816 sol
phase: 0.26236588497491725 rad

Peak [3]:
amplitudes: 0.6912258309483749
period: 334.3125 sol
phase: -2.4142071518308907 rad
```

Additionally, we have the following outputs for period in log scale vs Amplitude as well as period in log scale vs Phase for Set B:

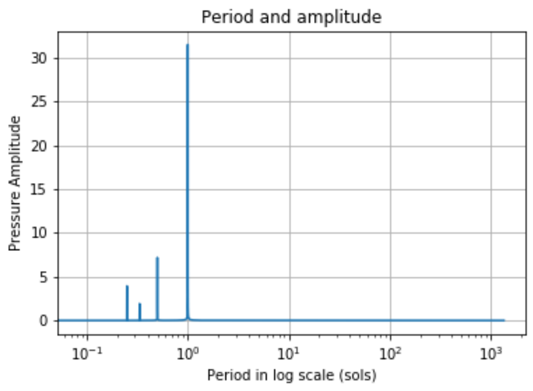


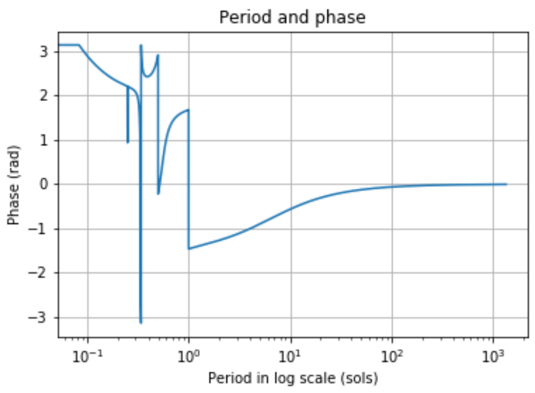


d) We now want to apply similar techniques to dataset C, which contains cleaned pressure data which uses only the diurnal and sub-diurnal waves, in an attempt to find the periods and amplitudes of these waves as well. Using a similar code to the last few questions, we yield the following output for the 4 waves.

```
Peak [1]:
amplitudes: 3.944800576915364
period: 0.25 sol
phase: 0.9352007680700093 rad
```

Peak [2]:
amplitudes: 1.9260042300971532
period: 0.33331256231306083 sol
phase: 0.9616063685780324 rad
Peak [3]:
amplitudes: 7.1943459941742836
period: 0.49990654205607477 sol
phase: 2.9159009063889325 rad
Peak [4]:
amplitudes: 31.547820723558637
period: 1.00018698578908 sol
phase: -1.4637120531847327 rad
The corresponding log(period) vs Amplitude and log(period) vs Phase plots are shown below:





e) Using Dataset D now, which is similar though with less clean pressure data than C, we want to repeat the Fourier analysis once again, and see whether or not we are able to extract the same information that we took from dataset C.

It is more difficult to extract values from this set. Because we knew it was similar to C, the method used to find the peaks this time were through referencing the peaks we found in part C. Using C as a reference, we found the following 6 points:

Peak [1]:
amplitudes: 3.7854989091157787
period: 0.25 sol
phase: 0.9713903439477437 rad

Peak [2]:
amplitudes: 1.8411042822048556
period: 1.0031882970742685 sol
phase: -1.0676096138790432 rad

Peak [3]:
amplitudes: 1.9240872171238592
period: 0.49953305939484494 sol
phase: -2.694199509112788 rad

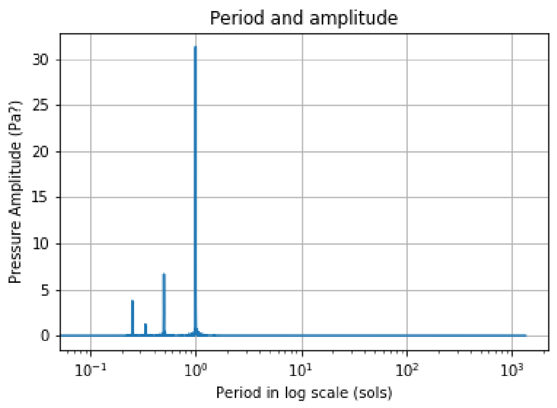
Peak [4]:
amplitudes: 6.638448993675578
period: 0.50009349289454 sol
phase: -0.17981641373018847 rad

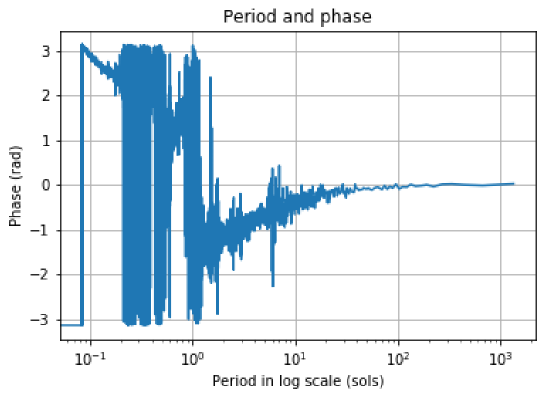
Peak [5]:
amplitudes: 6.484532518187595
period: 0.49990654205607477 sol
phase: 2.8288496772011182 rad

Peak [6]:
amplitudes: 31.3065803313626
period: 1.00018698578908 sol
phase: -1.4245489918578353 rad

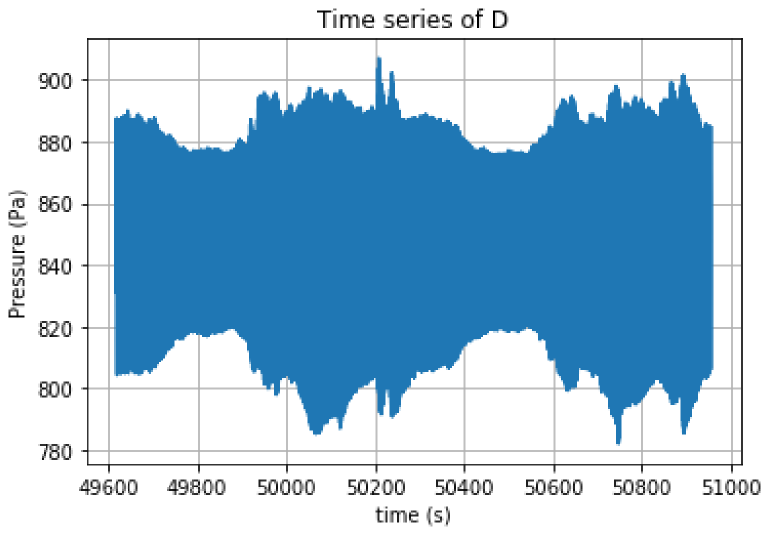
It's important to notice that peaks 4&5, as well as peaks 2&3 are very similar to each other. While there should truly only be 4, we extracted 6 because of the uncertainty of extracting these points purely from reference. This happens because the peaks we observe is actually "flat-tops" if we zoom in, thus it gives us lots of near identical peaks. In theory, the best way to remove the noise would be by using a Gaussian filter with a larger kernel size since we observed the peak region is quite big, where we find the average of a neighbourhood of data, putting more weights on closer neighbours, which would average out these peaks at 4&5 and 2&3 to be 1 peak each, giving us the desired 4 values rather than 6.

Plotting the phase and amplitude vs log(period) as done previously yields:





Plotting the time series of dataset D, we find:



Clearly, we can see fluctuations in the amplitude of pressure throughout the time series distribution, which are also apparent in the phase plot.

f) Finally, looking at dataset E, we now have the real pressure data including the seasonal pressure cycle, short term weather cycles and diurnal tides. Our objectives are to find the 4 largest waves with periods longer than 100 sols, which are seasonal cycles corresponding to summer and winter on Mars. Next we want to find the 4 largest waves with periods shorter than 1 sol (we took 1.1 just to account for values close to 1 sol), which are tides responding to the rotation of Mars and the heating from the Sun.

The two sets of data we got were the following:

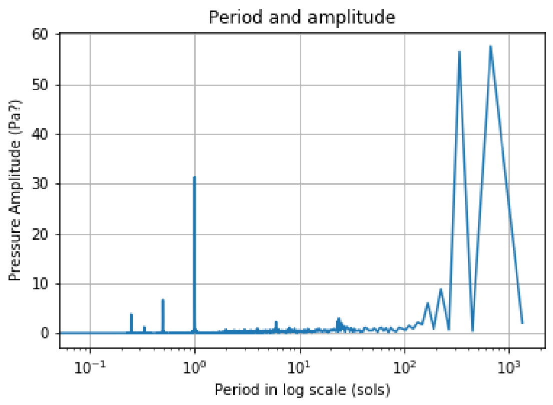
largest waves with periods longer than 100 sols:
Peak [1]:
amplitudes: 57.68172060693979
period: 668.625 sol
phase: 0.24524524310991014 rad
Peak [2]:
amplitudes: 56.53725838691755
period: 334.3125 sol
phase: -2.7642483913863773 rad
Peak [3]:
amplitudes: 8.803024739808025
period: 222.875 sol
phase: -1.0214370189444806 rad
Peak [4]:
amplitudes: 6.015890470292224
period: 167.15625 sol
phase: 0.8072130778579304 rad

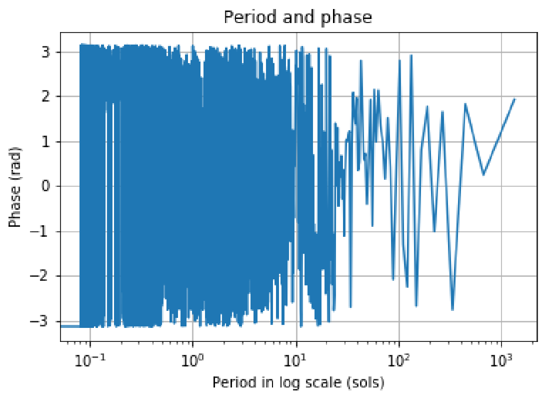
largest waves with periods shorted than 1 sol:
Peak [1]:
amplitudes: 31.30838735993113
period: 1.00018698578908 sol
phase: -1.424560869673691 rad
Peak [2]:
amplitudes: 6.638871408097484
period: 0.50009349289454 sol
phase: -0.1799528271570876 rad
Peak [3]:
amplitudes: 3.785269248224737
period: 0.25 sol
phase: 0.9712703012739636 rad
Peak [4]:
amplitudes: 1.617167805098776
period: 0.49916013437849943 sol
phase: 2.540518817210139 rad

The periods have the interesting relationship of being direct fractions, or multiples of each other on the order of ¼, ½, 1/3, or 1. This is very clear in the less than 1 sol set of values, but we can also see that the period of peak 2 (334) is half that of peak 1 (668), peak 3 is a third (223, and peak 4 is a quarter (167), they're completely cyclical.

Further, the periods where we find peaks in the dataset for periods of less than a Sol have local times of midnight (1 sol), noon (1/2 a sol) and 6am (1/4 of a sol).

Finally, the log(period) vs amplitude and log(period) vs phase plots for this data set were outputted to provide the following:





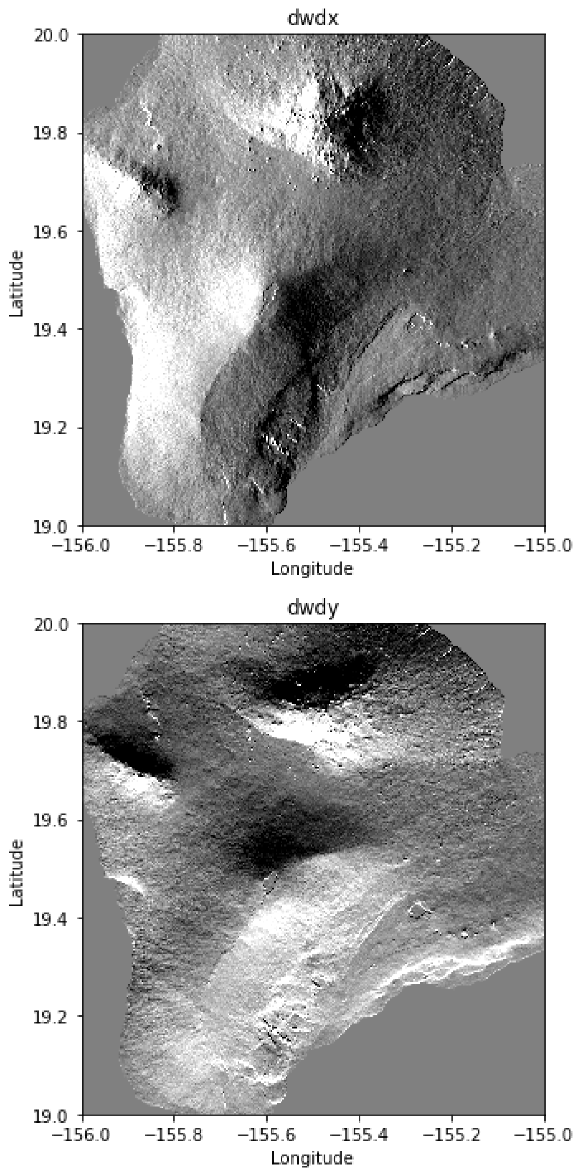
Question 3: Gradients using Fourier transforms and profiling FFT times

a) Imported the code from lab 3, for the gradient mapping of Hawaii

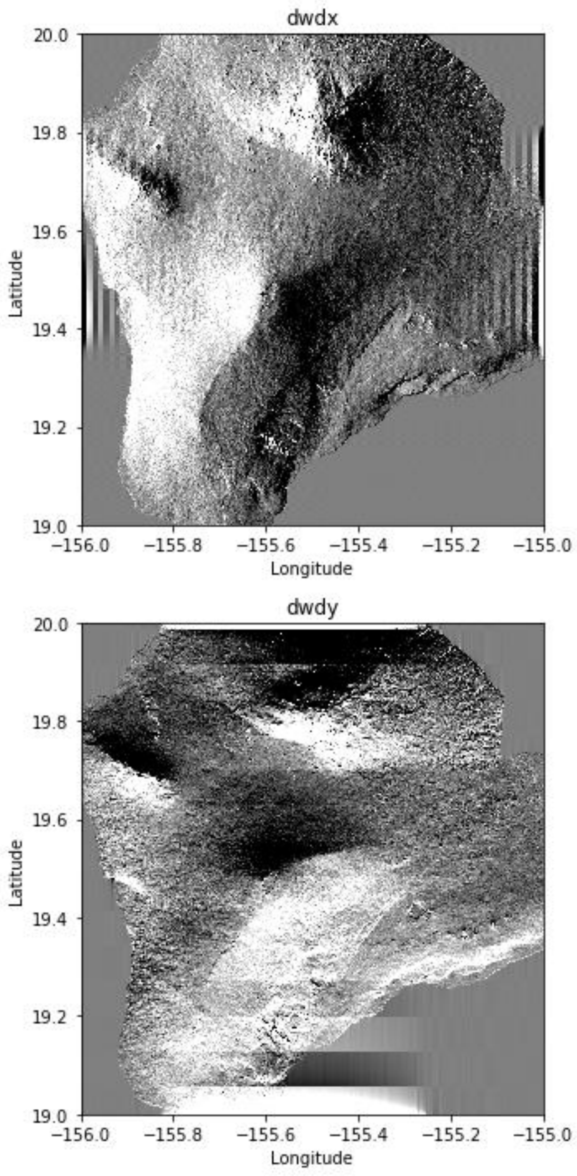
b) **Nothing to submit here**

c) For this question, the goal is first to write a function to calculate the gradient of a 2-dimensional data set using a Fourier transform to calculate the first derivative. Using that function, we want to calculate the longitudinal *(dw/dx)* and latitudinal *(dw/dy)* derivatives as in the previous lab, and then compare the results with the gradients we calculated in lab 3.
We used the formula (right)which is the derivative of Fourier.

$$\sum_{k=-\infty}^{\infty} \gamma_k \frac{i2\pi k}{L} \exp\left(i\frac{2\pi kx}{L}\right).$$



(Gradients from Lab 3)

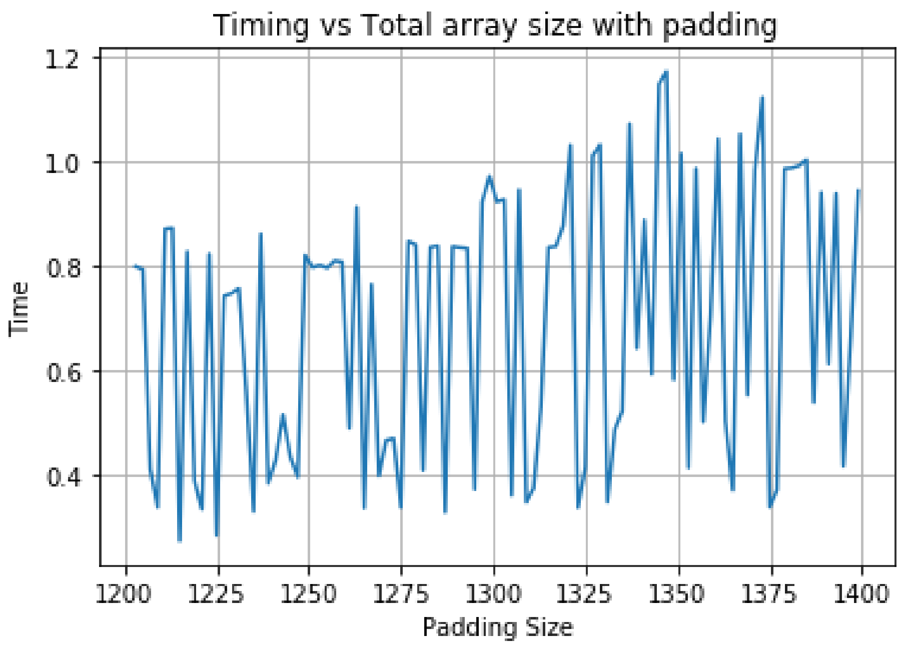


(Gradients from FFT)

The plots of the original gradients from lab 3 can be seen on the left, while the results from the FFT method of calculation can be seen on the right. Two immediate differences stand out, one being the stripe pattern we see on the Fourier transformed gradients which don't appear on the original. Second, and more pressingly, the Fourier transformed output appears far grainier than the Lab 3 outputs. This difference is likely a consequence of the indirectness of using the FFT method. The original method was a direct calculation of gradient whereas in the FFT version, we have to transform the data and then invert it back to normal. The cause of the stripes and graininess of the FFT version is likely caused by information lost during the process of transforming the data and then inverting it back again.

d) This question now turns towards the actual speed of our programs. The program speed is rather slow, so to investigate this, we want to create a function which pads our original data set prior to the gradient calculation, while still reversing it to the proper size. We then want to time the function for each value of padding, from sizes of 1-100 with step sizes of 1.

After adjusting our code to implement this adjustment, we outputted the following relationship:



Beyond the slight upward trend, the variation we see in the timing can be explained by numpys algorithm favouring particular values over others. A little research informed us that numpy tends to work at maximum efficiency for FFT n values that are powers of 2, and minimum efficiency for prime numbers, with of course some variation throughout the other values.