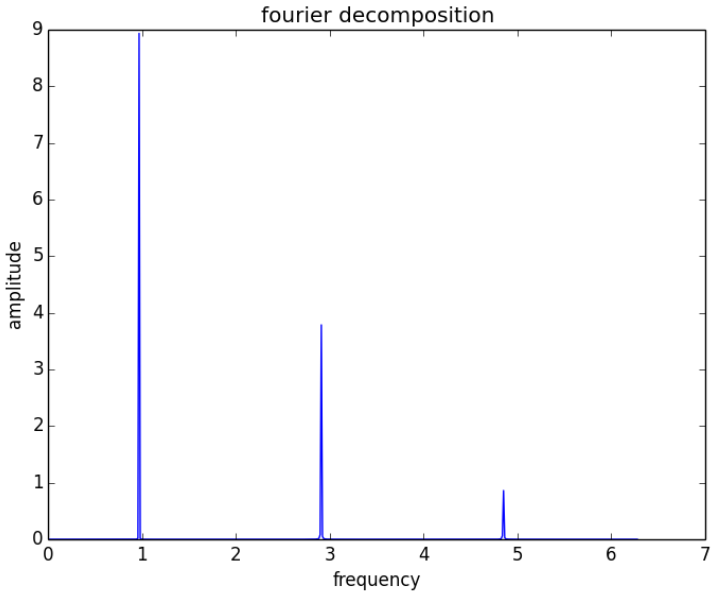
Computational Physics,
Roy Rinberg
Homework 9

12.53
Here is a plot of the power of the FFT of the first function



The Fourier is mostly real:

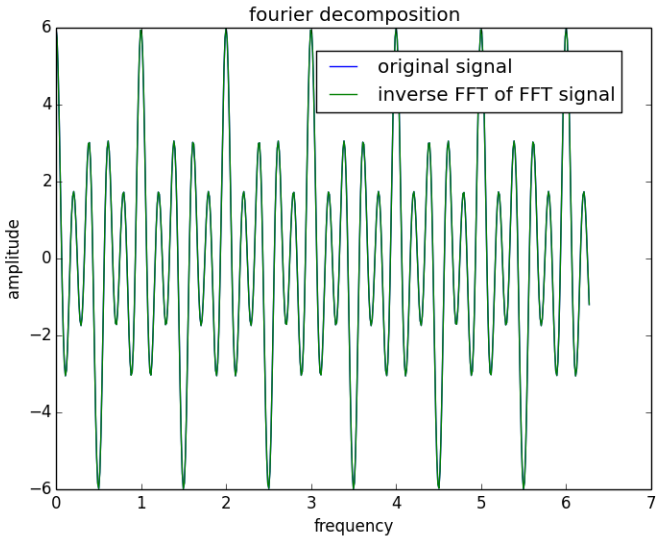
The sum of the absolute value of the reals:
25.3660988591
and of the imaginaries:
15.1932821402

however; the sum of the reals is :
12.0
and of the imaginaries:
1.4444825544e-15

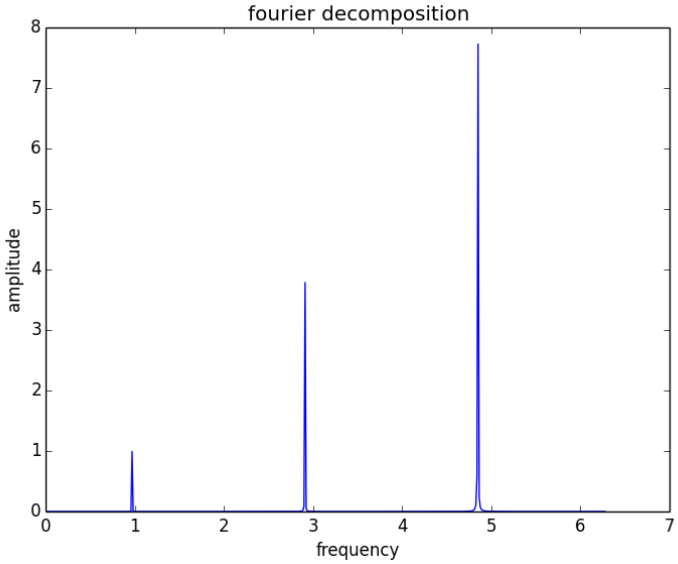
the sum of the imaginaries
-2.2440274465e-16

The number is more real than it is imaginary

This is a plot of the original signal, versus the Inverse Fourier of the Fourier. As you can see, they completely overlay one another



12.5.3 – 2
Here is a plot of the power of the FFT of the 2nd function



For some reason, the last "blip" only reaches 8 and not 9, this is not the case for the other FFT I did on the odd function

Additionally, we were told to check that this is mostly imaginary, but I did not get that that was true:
But instead, the same thing as for the one before:
Sum of reals:
12.0
sum of imaginaries:

-2.44162329244e-15

and
The sum of the absolute value of the reals:
25.5288448794
and of the imaginaries:
25.4477568369

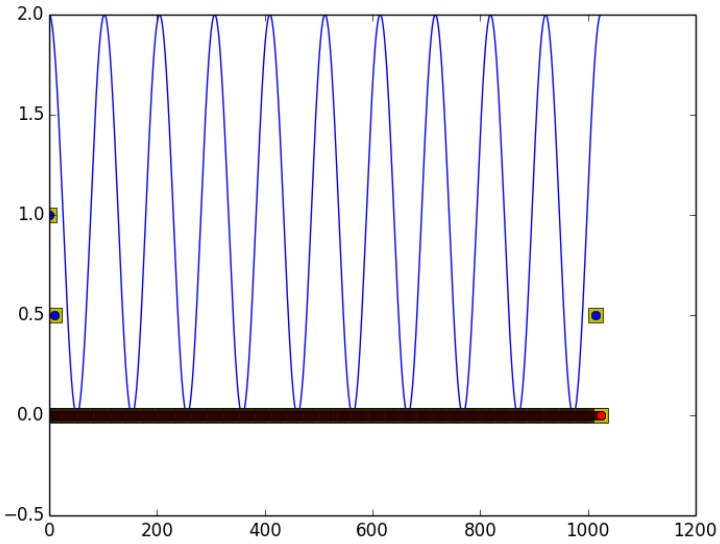
Certainly it is more significantly imaginary than before, but it is still more real than imaginary.

12.11- 3

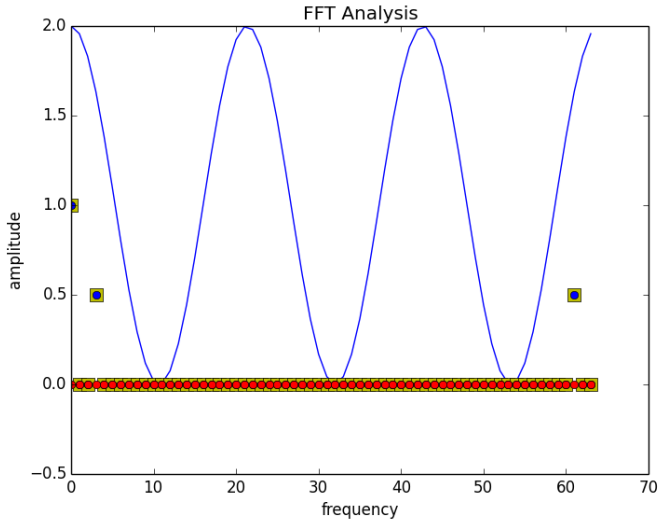
I ran fft.py a good number of times, and am pretty comfortable with the answers

The output contains both a plot and a time analysis of the operations:
FFT took 0.00102000000001 seconds but DFT would take 0.0130560000001 seconds numpy FFT took 2.10000000038e-05 seconds

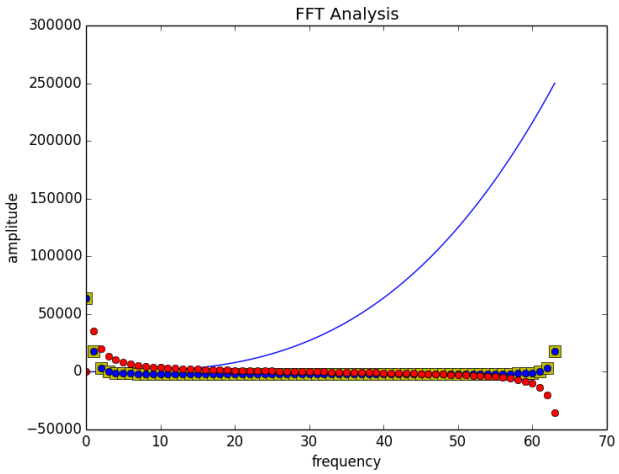
The plot represents the real and imaginary components of an FFT of the signal, using the python method and the numpy method (to compare precision and time). I don't know how to measure the precision, since they can only be measured relative to one another, however, the results appear directly superimposed on itself.



though perhaps this is easier to see:



I ran the program with a non-sinusoidal function x^3



Which gets me an answer that makes sense to me.