Interferometry and the Search For More Sources

Eduardo Herrera

April 8, 2014

Abstract

For this lab, we focused on using interferometers to track various sources across the sky from horizon to horizon using a tracking code that followed the source for the amount of time it was visible to the telescopes. From there data was manipulated from these sources, starting with the point source data from the Crab Nebula (3C144). We were able to derive values for the Declination and the Baseline (with different codes for each to make appropriate guesses for either the Declination or the Baseline) which were very close to their actual values to within one percent. With the Sun data taken for most of the day (a full observation was rendered short by other groups' scripts not finishing) we were able to get a value for ϕ and a create a wave that traced the envelope of the Sun data. As for the Moon, there was a similar process wielding relatively unstaisfactory results with the same method.

Introduction

In the study of sources in the sky it is important to understand what the components of the interferometer do, or be able to tell how one's data has been affected by the telescope and equipment and be able to correct for these affects so that as much data as possible comes from the source itself rather than having data give results that do not reflect past observations; in other words, noise from other sources.

Theory and Methods

Interferometry

The most vital part for our observations, interferometry involves using a two radio telescope array (there could be more in large arrays). The telescopes themselves are not the interferometer, but this array along with the electronics are. The two telescopes are separated by a baseline vector, **b**. With a source in the direction $\hat{s}$. Plane waves from the sources will go to one of the telescopes first with a short delay before the same plane wave reaches the second telescope, this delay is called the *geometric delay*, τ, defined by

$$\tau = \frac{\mathbf{b}\hat{s}}{c} \tag{1}$$

If there was another source, then it would cause another geometric delay and would be different from the delay of the first source and would also have a different $\hat{s}$. To differentiate from the sources they would need to correlate, which is what the correlator does to differentiate between the signals by plotting on a power spectrum, uncorrelated parts will average out to zero when correlated and the correlated data will appear as spikes for the signal from which it can be determined what their τ's are by looking at the time-axis.

The Rotation Matrices

Before making any measurements, a script has to be developed that takes into account our position on earth as well as the positions of the sources in the sky, especially the Sun and Moon, which have constantly changing positions in the sky as opposed to the relatively still point sources.

To start, most would start with longitude and latitude coordinates in degrees but to ultimately get to galactic coordinates (or any other type of coordinate system) via matrices that translate coordinates from one convention to another. If the longitude and latitude coordinates given were in degrees rather than radians this procedure takes the longitude, latitude and results in right ascension, and declination. So to start in this example we'd have to convert from degrees to radians using the code:

```
x = np.array([0.,0,0])
x[0] = np.cos(np.deg2rad(lat)) * np.cos(np.deg2rad(lon))
x[1] = np.cos(np.deg2rad(lat)) * np.sin(np.deg2rad(lon))
x[2] = np.sin(np.deg2rad(lat))
return x
```

after setting up this first array, to change from one coordinate system to another requires multiplying that array by a matrix resulting in another array that gives the new coordinates.

$$\vec{x}' = R * \vec{x} \tag{2}$$

as an example if we were to to change from (RA,DEC) to (Hour Angle, Declination) out matrix would look like

$$R_{(\alpha,\delta)\to(ha,\delta),1} = \begin{bmatrix} \cos(LST) & \sin(LST) & 0 \\ -\sin(LST) & \cos(LST) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Figure 1: an example of a matrix that changes an array of one coordinate system to another

For this lab, there were transformation matrices that started with our location in longitude and latitude, but we needed to correct for the objects in the sky in azimuth and altitude.

Use of Scripts

Data for this lab was taken using Kevin Yu's script.

Results

A majority of these these data sets were taken horizon to horizon, and resulted in varying results, from very good data for the Crab Nebula and the Sun to questionable data for moon measurements.

The Crab Nebula

Declination

To determine the declination angle of the crab nebula we needed to fit our smoothed crab nebula data (to reduce the effect of the DC offset of the data) with the equation

$$F(h_s) = Acos[2\pi(\frac{B}{\lambda}cos(\delta))sin(h_s)] - Bsin[2\pi(\frac{B}{\lambda}cos(\delta))sin(h_s)] \tag{3}$$

where we treat A and B as unknown and as the things we want to solve for using the least squares method.

The smoothing involved using the 'boxcar' method of averaging, where within an arbitrarily sized box, the median of two peaks would be created as a data point and then the procedure would move on to the next set of points eventually creating a trace of the original data with a smoother curve. This is meant to reduce the homing of the telescopes made during observation, as well as other anomalies that could have tampered with the data; unnecessary noise.

What we know is that this point source is at a fixed right ascension so for the data going into h_s we must subtract it's known right ascension from the local sidereal time (lst) data and convert to radians. We also assume that we know B, c, v and by extension l (λ in the equation), which are 10 meters for the baseline, the speed of light, 10.67 GHz and $\frac{c}{v}$ respectively. The most important part of this is to create a 'guess' for the value $sin(\delta)$ by picking values for q between 19 and 24 degrees and using least squares to generate a graph of χ^2 versus the guessed angle.

After running this code we got a value of about 22.34 degrees which was very close to the true declination of the crab nebula, 22 degrees, as seen by the following graph.

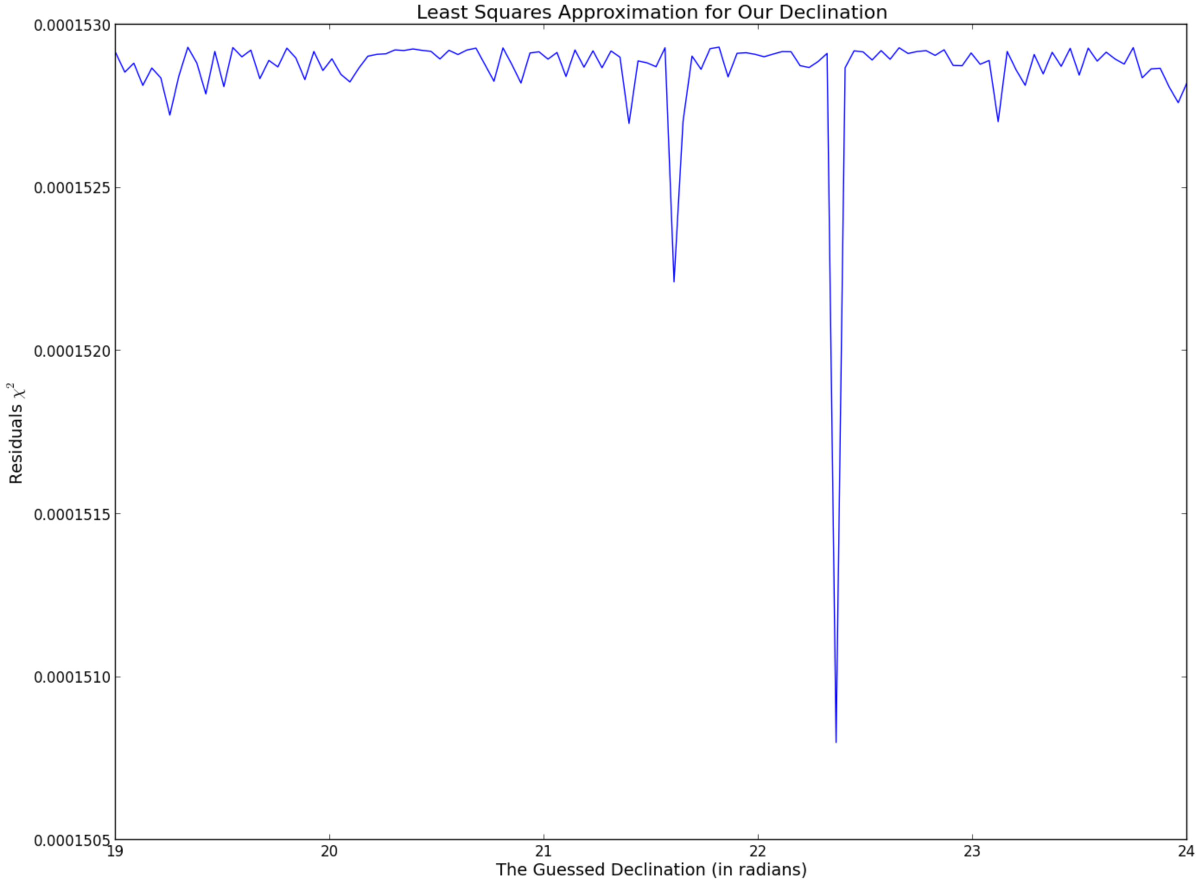

Figure 2: A graph of χ^2 versus declination in degrees for the crab nebula, the importance of χ is that it represents an error that when squared, the closer it is to 0, the more certain we can be that the value at minimum χ^2 is the correct one to a higher degree of confidence, and the model agrees closely to the true declination

We see a sharp decrease in χ^2 near where the observed declination angle should be, and there are no other points nearly as steep as the one at about 22.3 degrees, which closely agrees with the true declination angle of 22 degrees; a calculation within a percent of the observed value.

Cable Length

From the code, The matrix a could be called which housed the values of A and B which are equal to $cos(2\pi * v\tau)$ and $sin(2\pi * v\tau)$ respectively. For both, we get a value of the order 10^-6 which after plugging in and solving for τ and multiplying that by the speed of light, we get a value of about .007 meters, which comes out to about a 1 cm cable difference length.

Baseline

As a way to make sure that the previous method of least squares is correct(a failsafe), the code was modified so that δ could now be assumed to be what is in the lab packet, and instead make guesses for B, the baseline. Again the data for the crab nebula was smoothed out via the boxcar method again, and much of the code was left alone except now the declination was assumed instead of the baseline. Now the 'guesses' made ran between 7 meters and 12 meters, with a guess being made every .05 meters. After going through all the guesses, the code came out with a best guess of 10.027 meters, which is less than a percent within the actual (assumed) value of 10 meters. Below is the graph for this run.

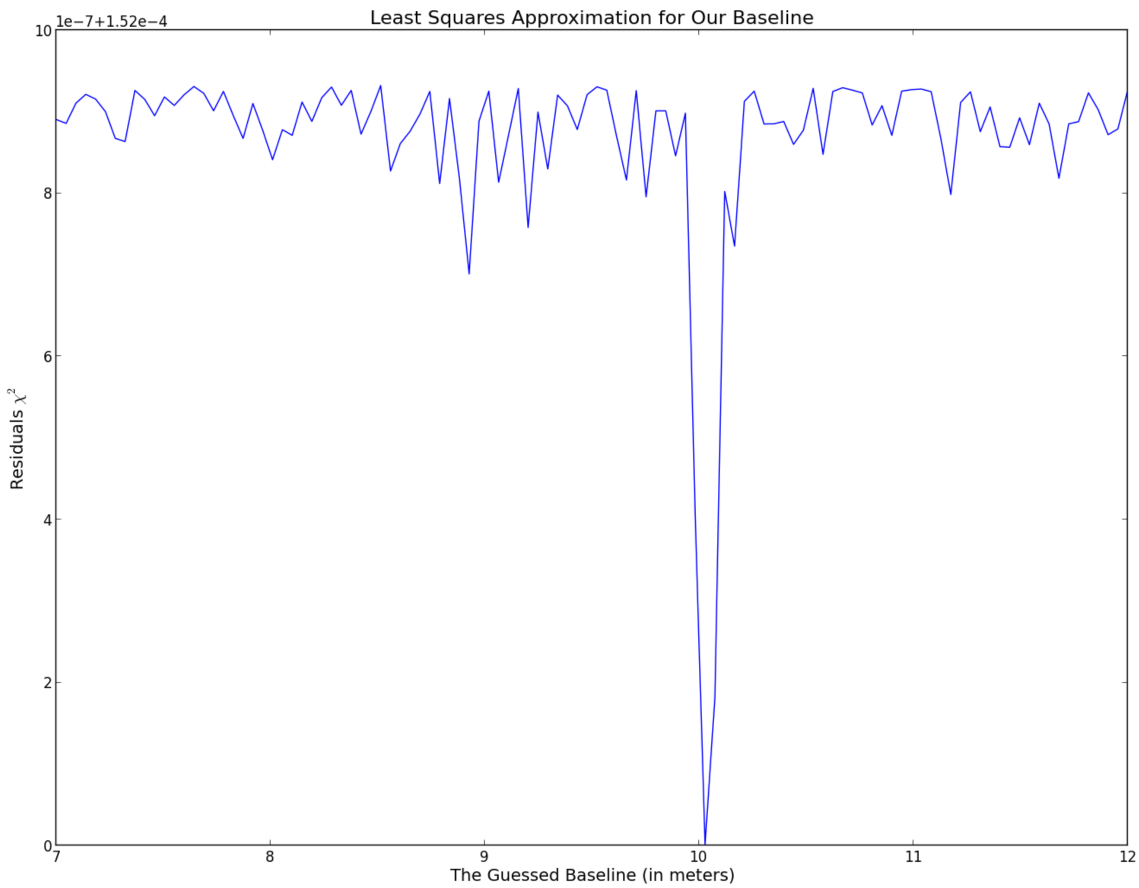

Figure 3: A graph of χ^2 versus the guessed baseline length in meters, again there is a single large dip to near zero for χ^2 meaning that the value of 10.027 meters was the best fit for this data

This baseline data together with the declination data indicates that the data gotten from the telescope is accurate(after smoothing), because our code was accurately written to track this source.

Previous Attempts and their Solution

On previous attempts, nearly all of the data was not viable for data analysis because the code was at the time set to chage position to continue tracking the object every 2 minutes. Because of this, our sources easily got out of range within a few hours, resulting in a lot of noise in our data. Our tracking beam was not concentrated on the source but would be instead on tangent to the source after a while, losing power that the telescope would have collected. With an update to track every thirty seconds made, the data came out better than before.

Least Squares Fit

Using the same code to guess a baseline, a graph was generated plotting the smoothened data set with YBAR, the best fit line for the data set, our observing time was relatively short so the whole set rather than a window was fitted. Below is the result

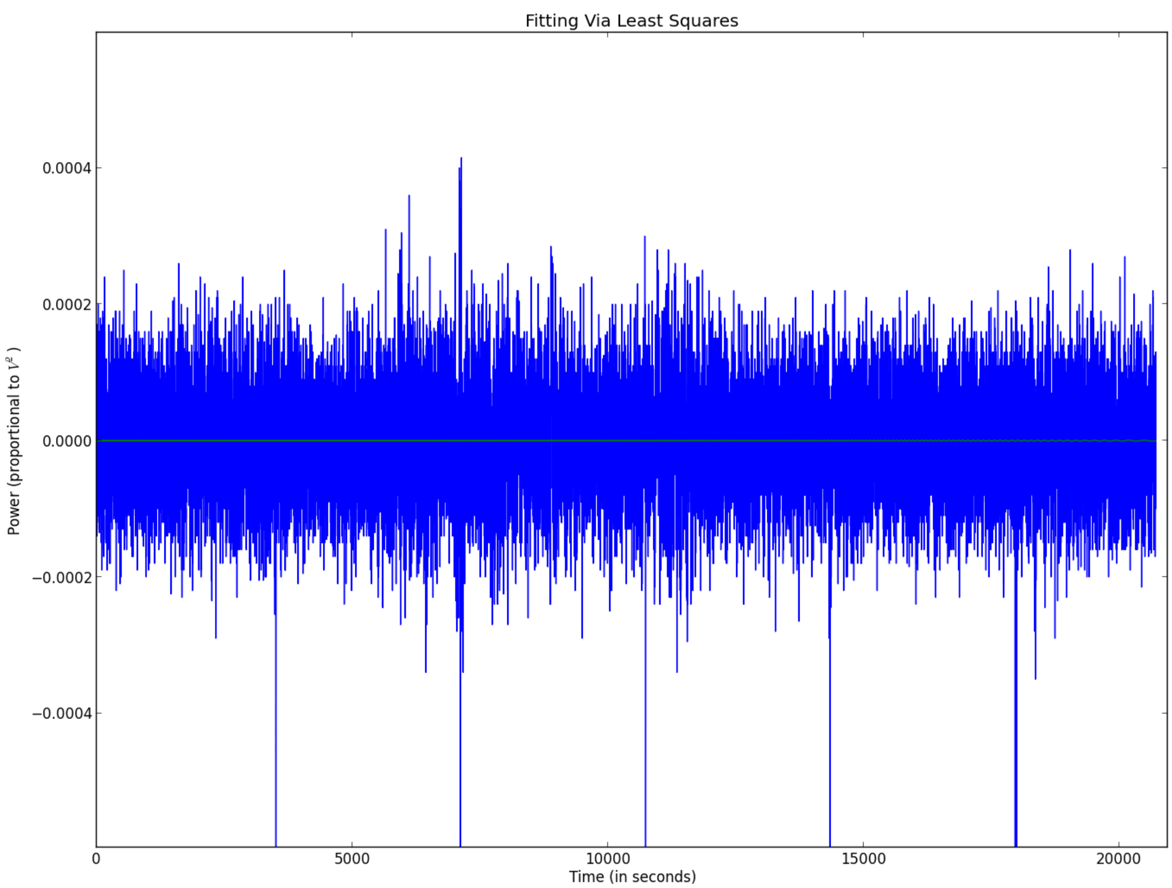

Figure 4: The least squares fit for the smoothened crab nebula data, from this vantage point, the best fit line can only barely be seen

We do not see much from here, the least square fit line seems to emulate a straight line since the data set doesn't seem to have a sinusoidal form, it's more like a straight line. The reason it could look like this is because it is so far away that the Power Spectrum appears to be constant, especially over a relatively short time interval (we did not take a whole horizon to horizon observation). Now, if we were to zoom in on the best fit line

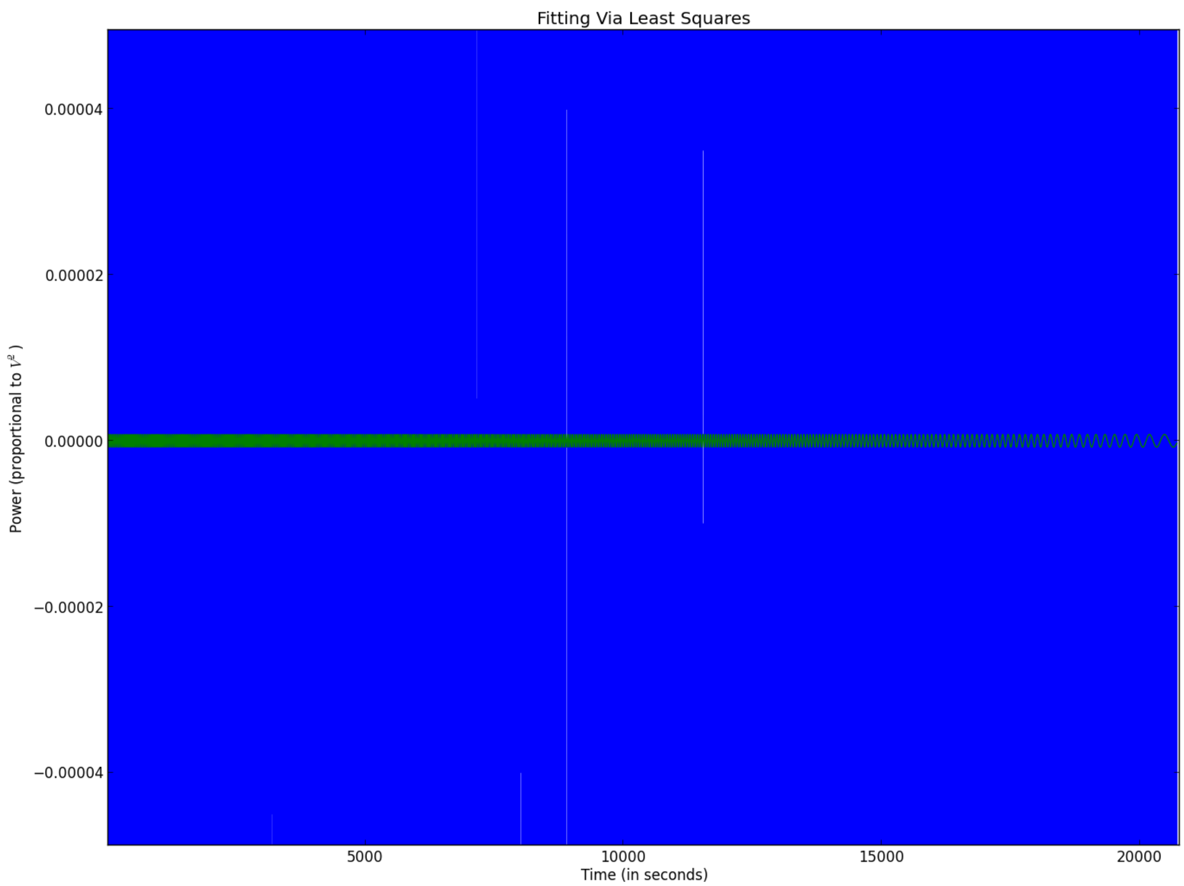

Figure 5: The same graph as before but where we zoom in on the best fit line

In this zoomed in picture, we see that the best fit line is a sine wave whose wavelength increases as time passes. This can be explained by the time when we chose to observe it. We started observations when the Nebula was already at its peak and so the signal was at its strongest, and as it neared and passed the horizon, it's wavelength was lengthened and the signal gradually became weaker.

The Sun

We observed the Sun again after updating our tracking time for the majority of the day, missing out on the first hour or so of data.

Phi

With the sun data, it was important to choose an area of the data that resembles a parabola so that the appropriate value of ϕ could be chosen and so that a envelope could determine how well our choice of ϕ was. For this sun data, the window of data points 1700 to 3300 (we took one data point per second) contained an area that had a parabolic shape so between these points. With this, we begin to find a ϕ according to the fitting equation

$$F(t) = cos((B/l)cos(\delta)cos(hour_{angle} + \phi)) \tag{4}$$

for which we have to minimize for ϕ. The hour angle is not like the point source, it is always changing since the Sun is an extended point source so it had to be adjusted by subtracting the right ascension array from the lst array to give us an hour-angle array, which then has to be converted to radians. In addition, as mentioned, there was focus on a parabolic area of the Sun Data. Another detail that had to be dealt with was the DC offset of our data, to deal with this, the mean value of the 'volts' data was subtracted off of the original 'volts' data, this is not the most effective way to account for DC offset but for fitting purposes it is acceptable. After making sure to have all arrays match in dimension; all arrays must cover the same place with the same lengths, we made a guess in ϕ and we see that from the graph, we get a ϕ value of 2.06266 radians.

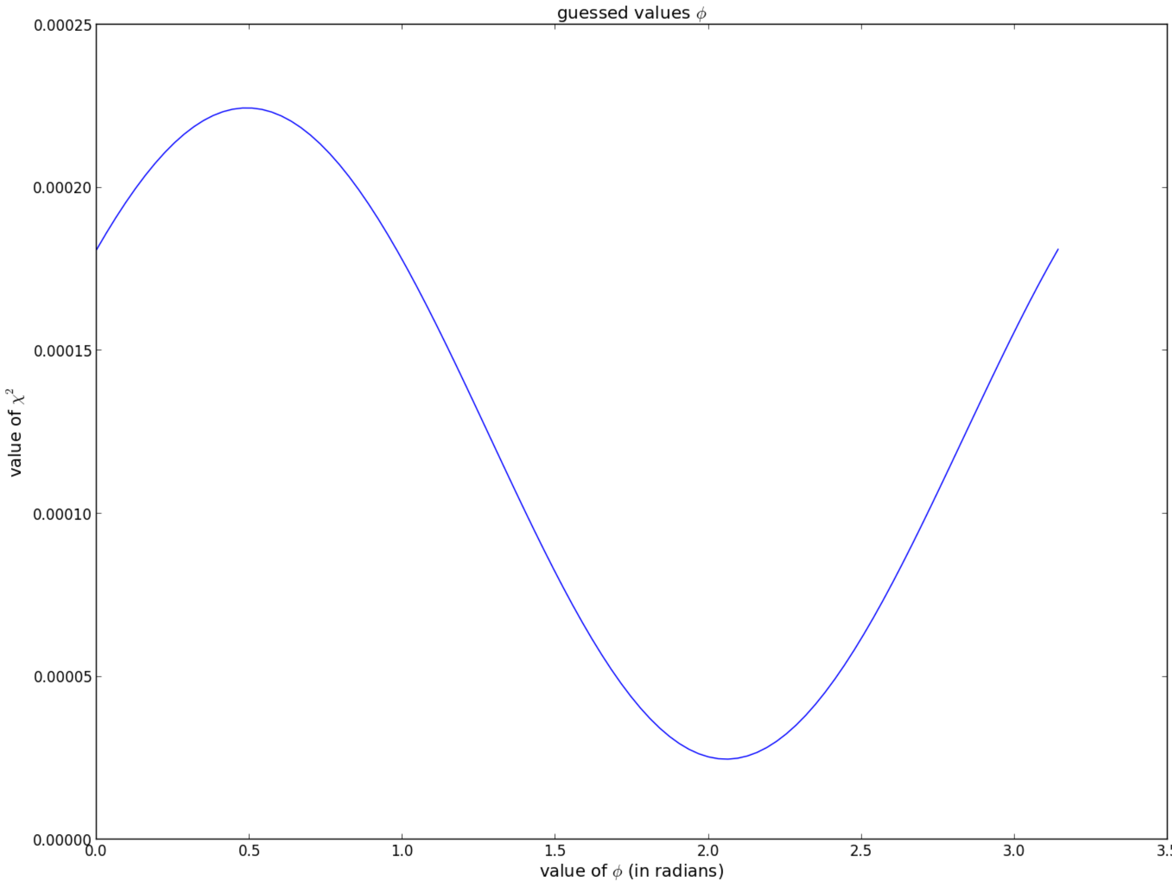

Figure 6: A graph of χ^2 versus guesses in ϕ, we see a strong peak towards zero at the value of about2.062.

As we see, the graph looks like a sine wave, the reason perhaps being that for ϕ this trend is periodic since it the ϕ is in a cosine function, and since it's in radians, its behavior must be periodic.

Least Squares Fitting

To fit a least squares fit to our data at the parabolic area, all we do is plot our Y array which is defined as the volts data of the sun minus the mean of that same data, again to compenstae for the DC offset. And plot that against the YBAR dta, which is the best fit line for the data acquired, plotting these, we get,

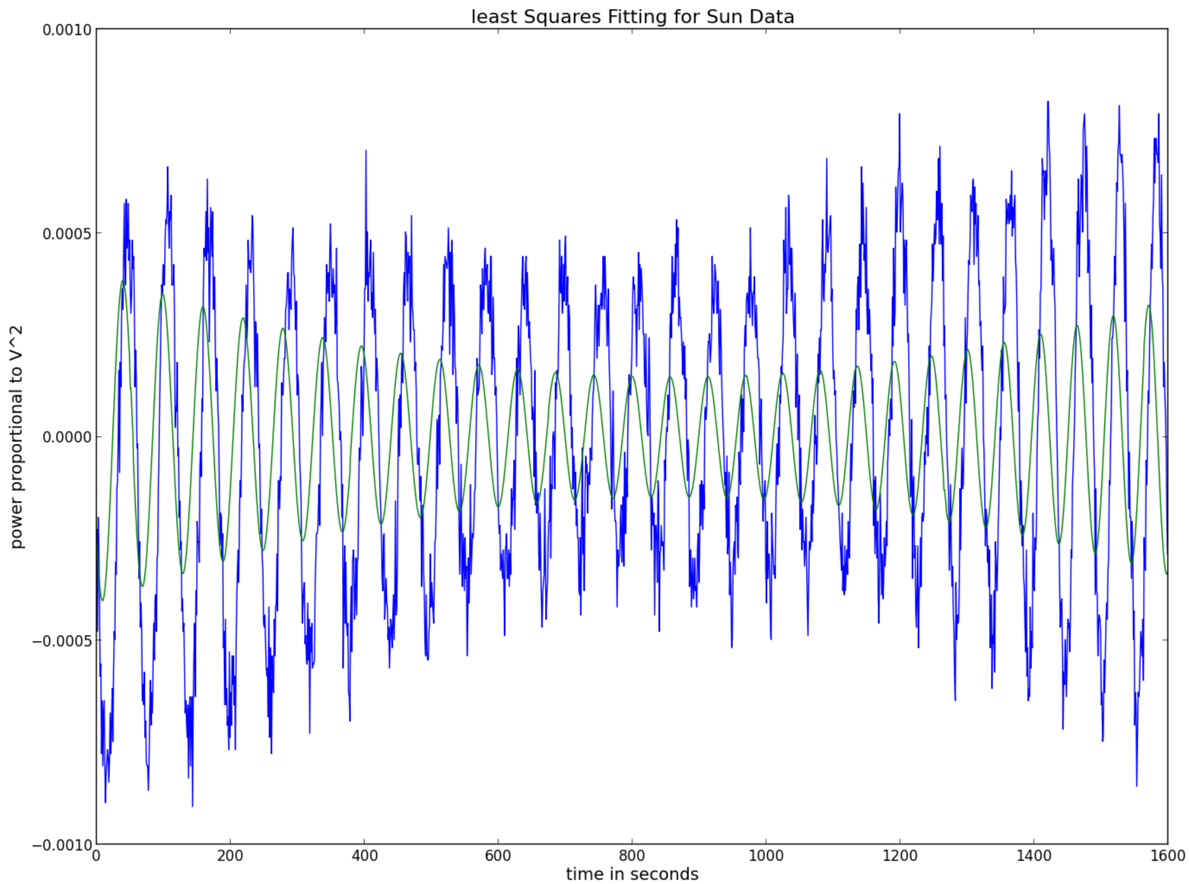

Figure 7: Fitting for our Sun Data, the best fit data fits well with the sun data.

This least squares fit line does an excellent job in approximating the wavefront of this window of values, so that a pattern can be seen more easily.

Envelope

To fit for an envelope around our data, that is where our minimized ϕ comes in when plugging it into our modulation function F(t). After plugging this value into our equation, we can plot YBAR as well as a negative version of it too, after doing both of these our result is

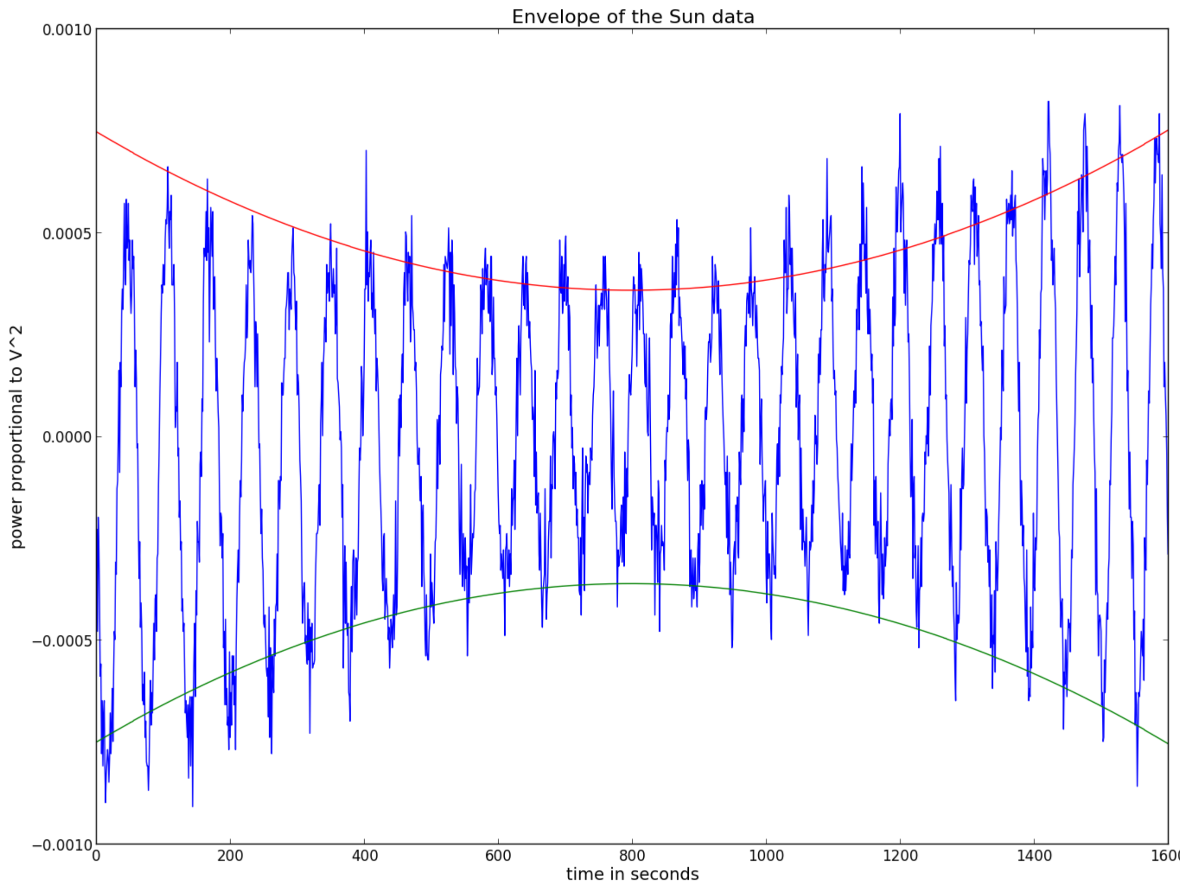

Figure 8: A good approximation of the envelope encasing the data set to a good degree, the envelope correctly approximates and averages over the peaks of the individual spikes to create a single wave describing the data.

This enveloping line is the result of graphing YBAR with the Sun data, the importance of YBAR becomes apparent when trying to measure the radius of the sun.

The Radius of the Sun

To find the radius of the sun, it is necessary to plot the MF_{theory} versus the product of fringe frequency and R; fR. A combination that minimizes MF has to be a zero cross section and so we can use that to find an R, by looking at our YBAR array's minimum value and using indexes to find the concurrent values of declinations, right ascensions, lst's and using those values to find a value of f_f, known as the fringe frequency. With this fringe frequency we can find the value of R by dividing the value crossing zero on a bessel function by the fringe frequency to get R, the radius of the Sun. Shown below is the bessel function for the Sun.

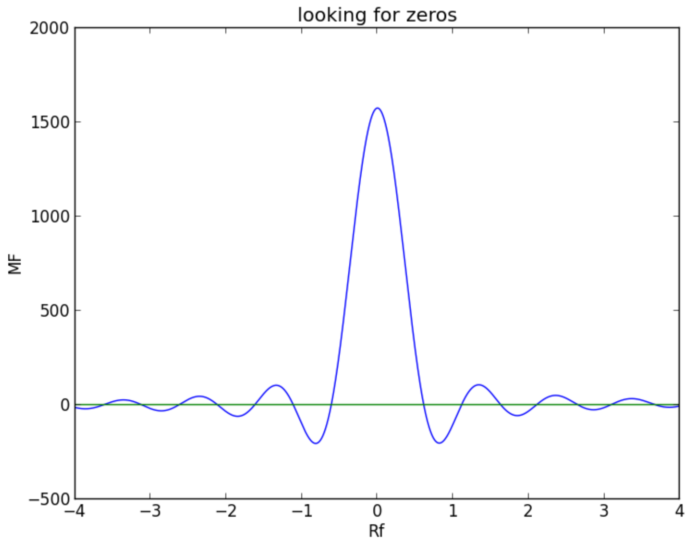

Figure 9: The Bessel Function for the Sun

Another detail in the parabolas we try to find is their cause. When the dishes point toward the sun, some of the waves-fronts may experience destructive interference, which we see on the graphs as parabolas or dips. These are actually related to the fringe frequency. There can be different values of R in the combination fR but when at a local minimum 9 we can get the lowest values of the angles needed in the fringe frequency so that we can just divide the value at which the x-axis crosses 0 by the fringe fraction to get the Radius of the Sun in radians

The only unfortunate part that while calculating R, we came across a value of .025 radians which is a factor of 10 larger than the true radian radius of the Sun. Attempts to fix this radius measurement wielded nothing since choosing a larger R only madethe radius larger than it already should be. The error could have been in the code run to get the values for δ and h_s, or more likely, human error while attempting to compute the radius by hand.

The Moon

This data is treated identically to the sun data save for some modifications and regions of interest. Unfortunately this data did not yield any good data to use least squares on, so the data presented here may not be favorable.

Phi

Again, using the same procedure that was used on the sun, get a ϕ of 2.79 radians, since data was taken from an area that had a 'half-parabola' shape so this ϕ may not be all that reliable. A note should be made that the moon data was smoothened out, so this data is not useless since the DC offset has been at least partially dealt with.

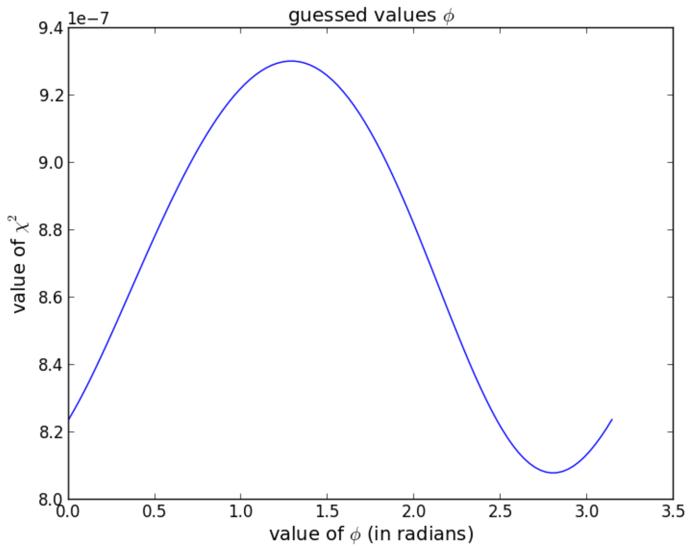

Figure 10: A graph of the the most favorable guess for ϕ based on the value of χ^2

Least Squares Fitting

Using this value of ϕ, we again calculted the array for the best fit line of our data interval and resulted in

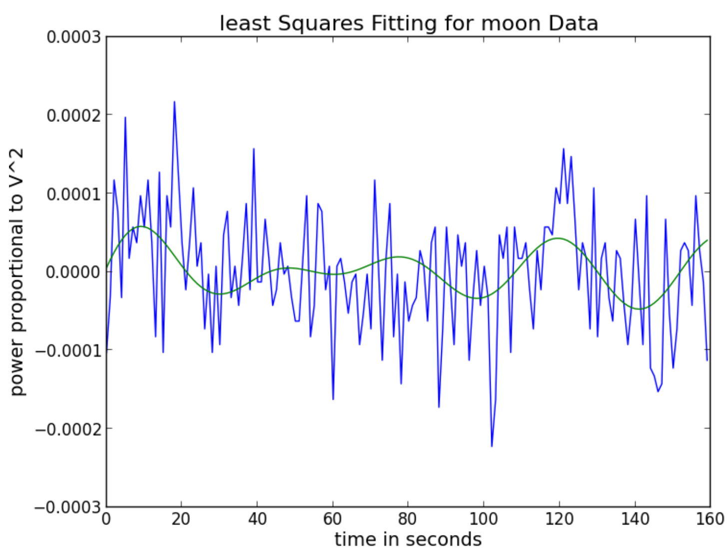

Figure 11: The least square fit line for the data on some interval of the whole set

we see that the least squares fit line does accurately fit the data, so perhaps the ϕ from the previous section is more accurate than previously thought.

Envelope

As for the envelope, when it is overliad over the smoothed data, it does not look right, mostly because the enevelope does not go over the peaks of the data points, the data range selected is not parabolic.

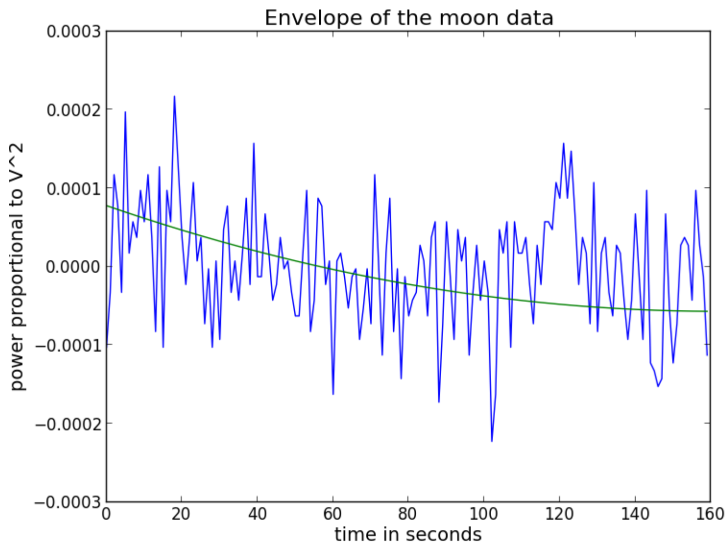

Figure 12: The supposed envelope of the moon data, it is wrong mostly because of the bad data

Since we did not choose a window with a minimum, ϕ could not be accurately measured, so that will affect the radius measured.

Radius

Once again we would have some minimumparabola to look at, but from the moon data gathered there did not seem to be any good areas to pick a minima.

Conclusion/Discussion

Overall, the results were satisfactory, being able to get a measurement that worked with least squares approximations with two different sources did at least affirm that tracking codes were working. The problem was mainly with the moon data that did not seem to be good data to work with, which may be because it wasn't reflecting fully as it was in previous weeks. Even after smoothing the moon data, it still weilded bad results for the envelope because of its irregular form over larger windows. Though there was the start of being able to calculate the Radius of the Sun, it could not be finished due to a lack of understanding the locations of the minimum values in the arrays, Radii couldn't be fully completed.