

THE WIENER FILTER*

Aditya Nag
Benjamin Weidman

This work is produced by OpenStax-CNX and licensed under the
Creative Commons Attribution License 1.0[†]

Abstract

Enter the Wiener filter, an adaptive filter. Derivations and applications.

The wiener filter is an adaptive filter. It tailors itself to be the “best possible filter” for a given dataset. Below is a simplistic version of the derivation for the wiener formula.

1

Consider our standard equation to model a signal with noise:

$$y[n] = x[n] + n[n] \quad (1)$$

We want to pass this $y[n]$ through a filter ‘h’ such that we get back something that very closely resembles our original signal x , i.e. x [U+F01A].

So basically we want to design a filter that minimizes the difference between x and x [U+F01A]. Lets start out by minimizing the least mean square error between x and x [U+F01A].

$$\| x - \tilde{x} \|_2 \quad (2)$$

But we know x [U+F01A] is $h*y$ (h convolved with y), so we have:

$$\| x - x * y \|_2 \quad (3)$$

We can expand this expression known algebraic rules. Then we can take the Fourier transform of the expression to find the power spectra.

$$\sum_j (X_j - H_j Y_j)^2 \quad (4)$$

$$\sum_j (X_j - H_j (X_j + N_j))^2 \quad (5)$$

*Version 1.1: Dec 16, 2004 12:16 am -0600

[†]<http://creativecommons.org/licenses/by/1.0>

We minimize this expression over H and in the end after all the simplification we get the following formula for H , our filter optimized to minimize the difference between x and x [U+F01A].

$$H(f) = \frac{(|X(f)|)^2}{(|X(f)|)^2 + (|N(f)|)^2} \quad (6)$$

Where $X(f)$ is the power of the signal and $N(f)$ is the power of the noise.

Just by inspection we can see that this filter will take care of the basics. For example when there is no noise, the filter response goes to just 1. When the signal is 0 the filter response goes to 0.

Notice to use this filter we will need to know the power spectrum of the actual signal. We also need to estimate the power spectrum of the noise. In real life this is done in numerous ways. Oftentimes out of convenience engineers will approximate the noise to Gaussian. This works well with varying results.

2 Building the Filter

One way to approximate the noise is “by inspection” or what is more commonly known as “guess and check”. In this method, you take a close look at your received signal and your pure signal.

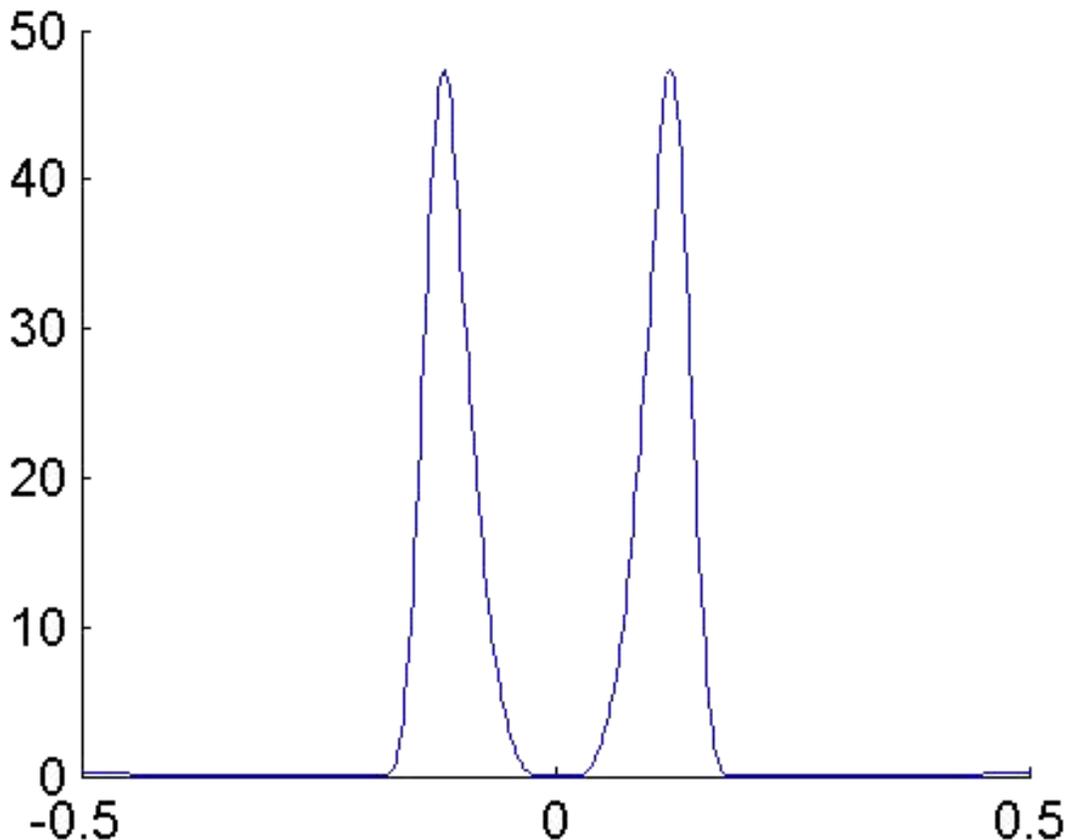


Figure 1: Pure signal spectrum.

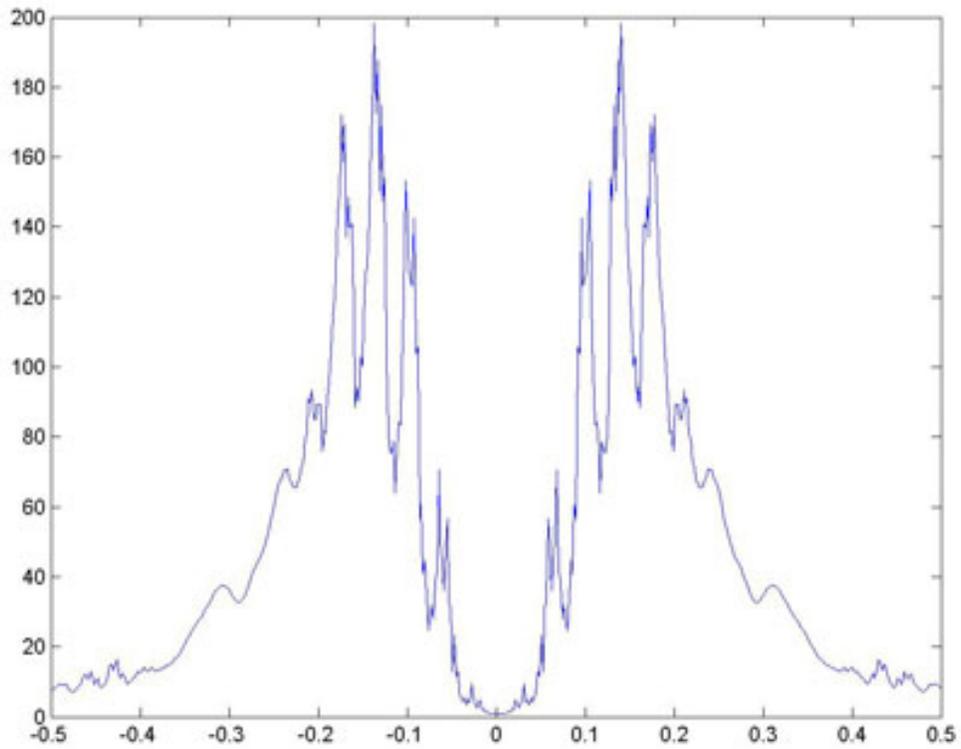


Figure 2: Received signal spectrum.

Then we try to figure out what the noise could look like based on this information. For this data set, we have traced out our “best guess” for the noise spectrum below

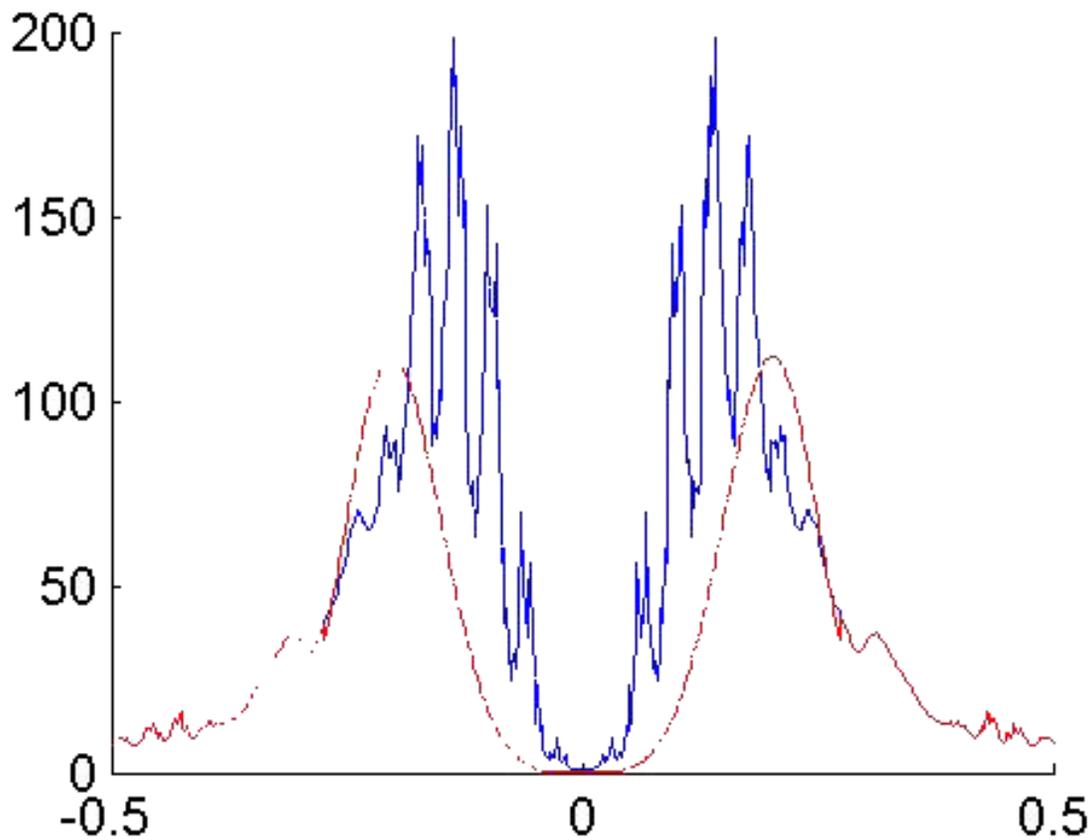


Figure 3: Noise trace.

Just by looking at the noise spectrum we want, we look for patterns we could fit a mathematical formula to. In this example you can see that the function decays like a polynomial on one side and exponentially on the other.

So our general noise toggling function becomes:

(7)

And so we fix our parameters alpha, beta and gamma until we get something that resembles the function we drew in red earlier.

NOTE: This function is particular to this data set. For other types of noise, you would have to fit a new appropriate noise function.

So now using the values we calculated for $X(f)$ and $N(f)$ we create our wiener filter.

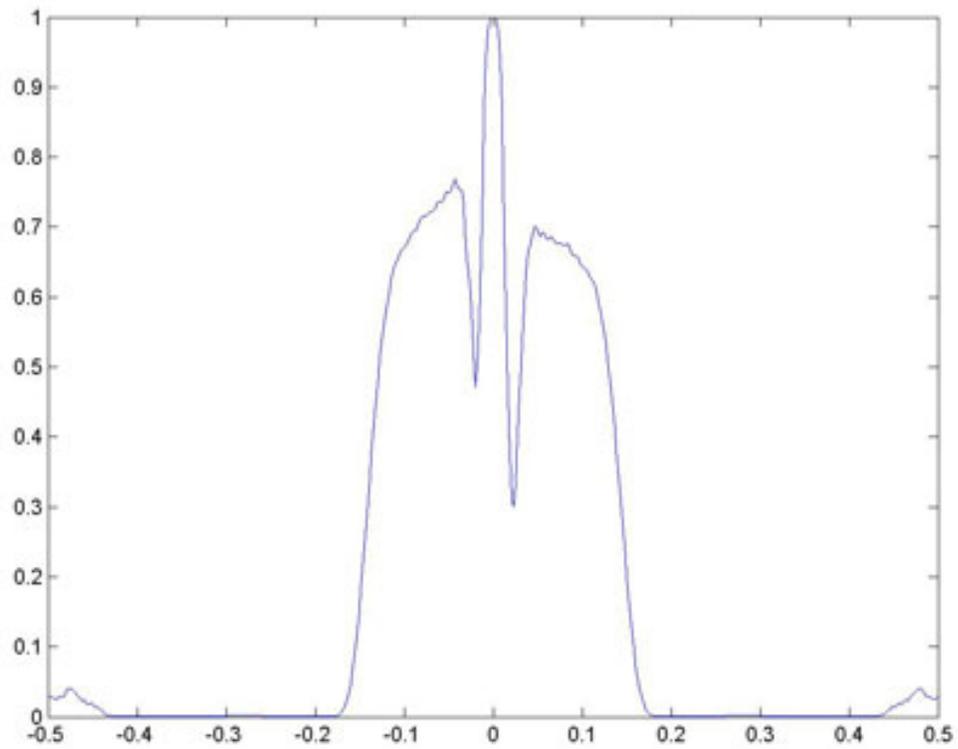


Figure 4: Magnitude Plot of the Wiener filter.

So lets try the filter on the data shall we?

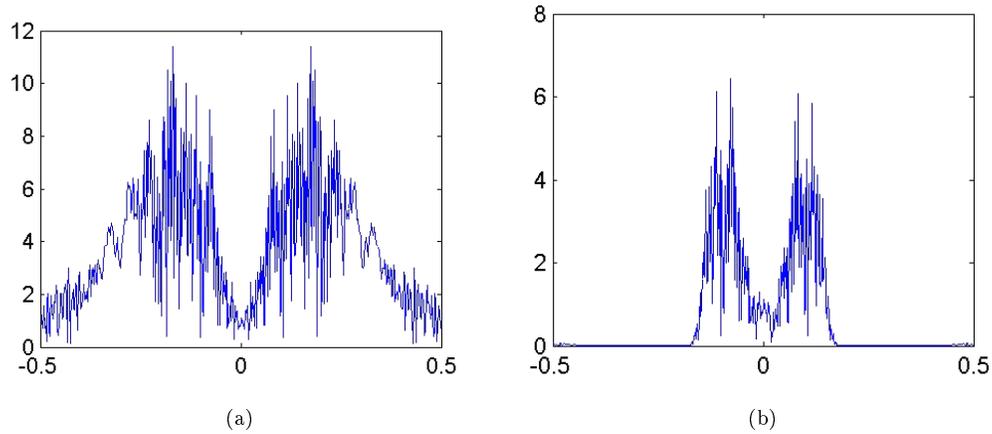


Figure 5: (a) Unfiltered data. (b) Filtered data.

We see that the wiener filter does its job pretty well. It even wins over the bandpass filter in the simpler example since it even removes the excess amplitude added by the noise.

Lets take a look at the image of all the time series vectors.

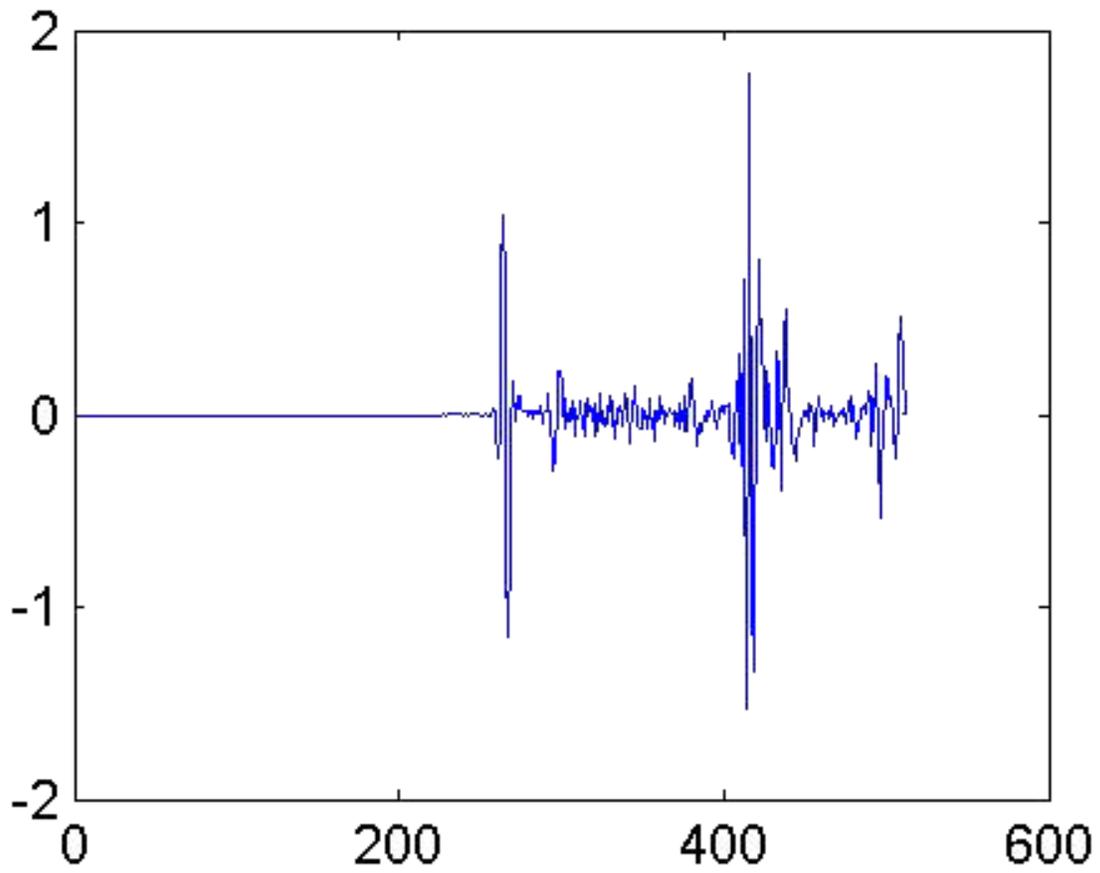


Figure 6: A single unfiltered time series vector.

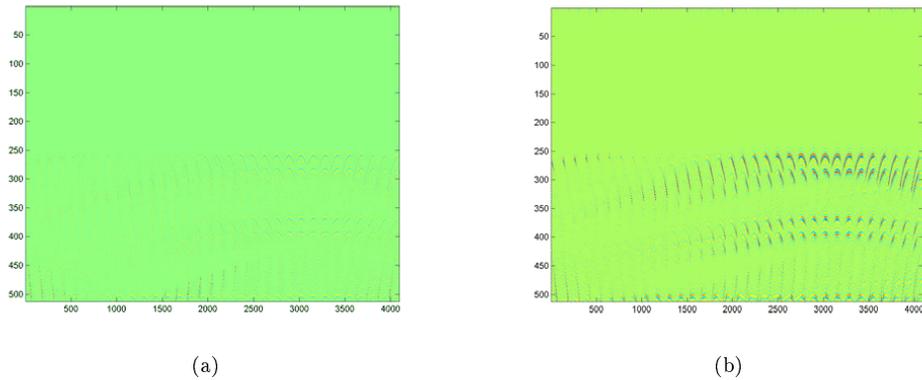


Figure 7: (a) Image of unfiltered time vectors. (b) Image of filtered time vectors.

We can see that the unfiltered image is pretty blurry with not much distinguishable from the background. On the other hand the picture of the filtered version comes out pretty well and we can see distinct values for every time series.

So that concludes the section on filtering. Hopefully you have a better understanding on how to design adaptive filters, the wiener filter in particular. Now its time to move the data to the next stage of the process: Imaging.