# Sparse Coding with Integrate and Fire Neurons

**Thomas Pinella**

5.8.16

Professor Haefner
Computational Neuroscience

# Abstract

Interpreting the responses of neurons to sensory stimuli as stochastic samples of a probability distribution has yielded positive results with predictive power (Hoyer 2005). Here, we sought to achieve a similar feat of getting a sparse coding of sensory stimuli, but using integrate and fire neurons. With stimuli in the form of 1-dimensional Gabor functions, we succeeded in representing it as a sparse coding amongst a set of neurons and, using the firing rates of the neurons, we were able to regenerate the stimulus function based solely off the neurons' internal representation.

# Introduction

Using integrate and fire neurons, the change in a neuron's voltage depends on two factors: 1) the excitatory aspect -- how similar its own projective field is to the stimuli and 2) the inhibitory aspect -- how similar its own projective field is to every other neuron that is firing. We used half-wave squaring in both instances to avoid negatives. Based off this formula for calculating neuron voltage, we were able to sparsely encode sensory stimuli as firing rates across all the neurons.

# Gabor

Although we experimented with a couple other options, we primarily performed our tests using a 1-dimensional Gabor function as our projective fields and stimuli. For the projective fields of each neuron, we assigned it a vector of length m, where the contents of the vector represented the output of the 1-dimensional Gabor function. The higher m, the higher a resolution of the Gabor we get. We varied phi for each projective field for each neuron such that over n neurons we covered the range $[0, 2\pi)$. The more neurons, the more projective fields and therefore the higher the resolution.

## Results with Gabor Stimuli

When passing in a 1-dimensional Gabor function with phi equal to 0 as our stimuli, we saw exactly what we would expect: a high firing rate for the neuron with a projective field of phi equal to 0, and low firing rates for all other neurons. Below is an output of the firing rates of all 256 neurons. The first neuron's projective field has a phi of 0 and the last has one approaching $2\pi$, with every other neuron somewhere in between.

```
[ 34.24657534  18.83239171  11.93317422   0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
   0.          0.          0.          0.          0.          0.          0.
```

```
0.          0.          0.          0.          0.          0.          0.
0.          0.         11.93317422  18.83239171]
```

This is what we expect to see: the highest firing rate is the first neuron since it has a phi equal to 0, and then it drops off to 0 until we get to the end where the neurons have phi value close to $2\pi$, meaning their projective fields are similar enough to the stimuli to cause some spiking.

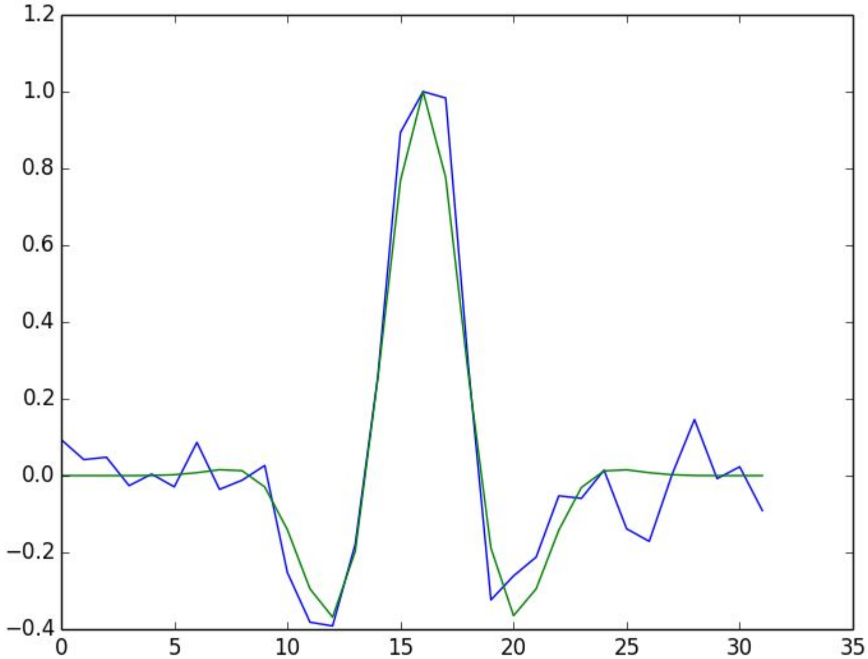
To prove this encoding's accuracy, we take the dot product of A -- the matrix of all the projective fields where each column is a neuron's projective field -- and S, which is the vector of firing neuron firing rates printed out above. This dot product should return an approximation of the stimuli, and indeed it does. In the above example, we had a perfect recreation of the Gabor function with phi equal to 0.

An important note is that there are two constants -- alpha and sigma -- that we can tweak to give us different firing rates and thus different sparse codings. Alpha is a constant that we multiply the difference of our excitatory and inhibitory voltage changes by, so by increasing it we see not only higher firing rates, but more neurons firing. By increasing sigma squared, or our variance, we see less neurons firing. Therefore, by changing these constants around, we can control how sparse we want our coding to be. The above example was calculated with alpha set to 250,000,000 and sigma set to 0.001. As long as alpha is large enough such that neurons are able to fire, it's value will not upset the accuracy of the model, but rather only control how sparse the coding will be. We raised alpha even higher and found that even though more neurons were firing, we were still able to perfectly retrieve the stimuli through taking the dot product of A and S.

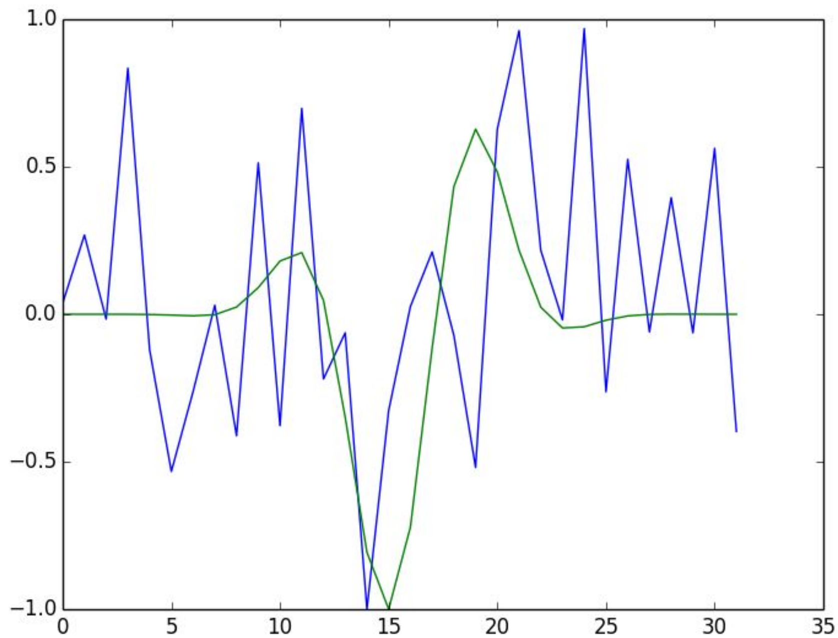
We also tested adding Gabor functions together as our stimuli and those were similarly successful.

## Results with Other Stimuli

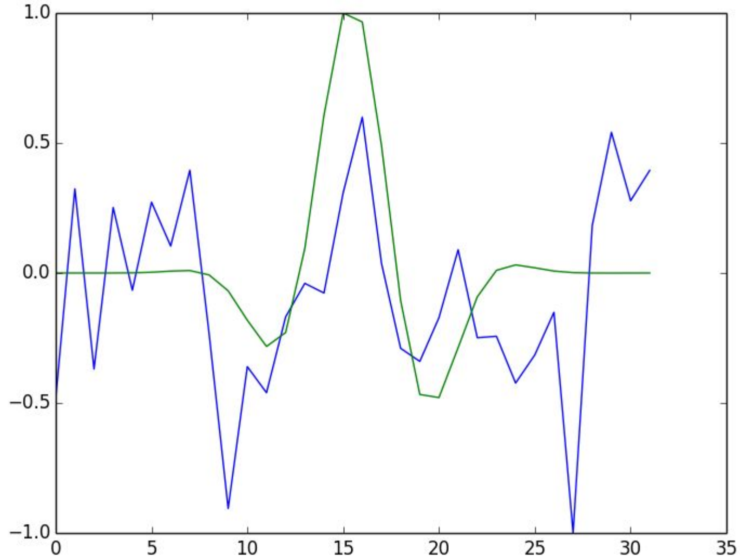
We tested our model with noisy Gabor functions as well, where we took the regular 1-dimensional Gabor stimuli and added some level of Gaussian noise to it. We found that we were still able to get rather good approximations of it. Below is graphed both the incoming noisy Gabor stimuli and the approximation of it.

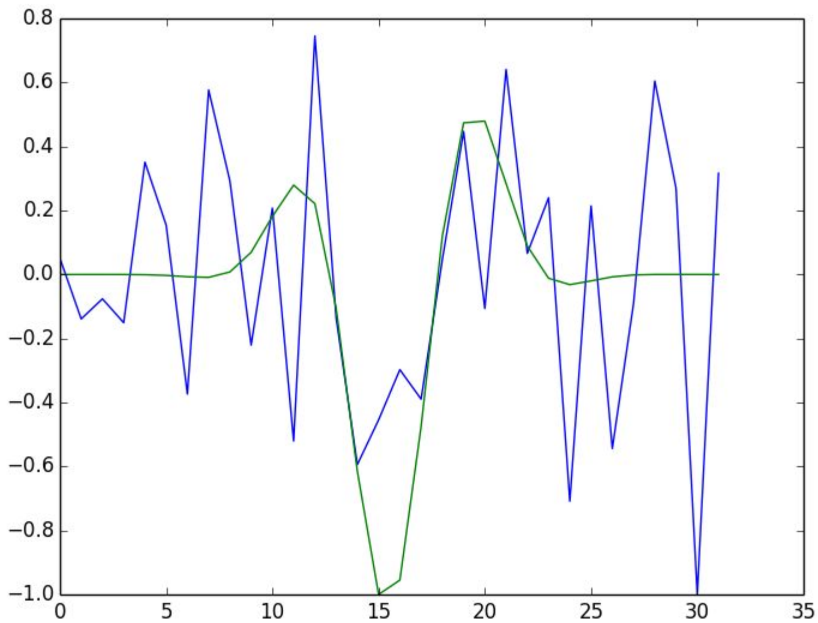



We tried inputting an even noisier Gabor:



With stimuli of straight Gaussian noise we got these resulting approximations:

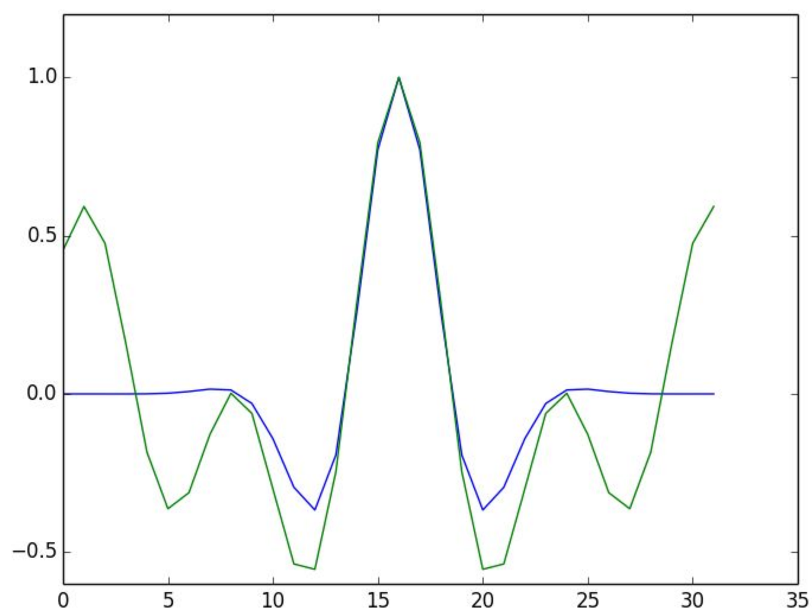


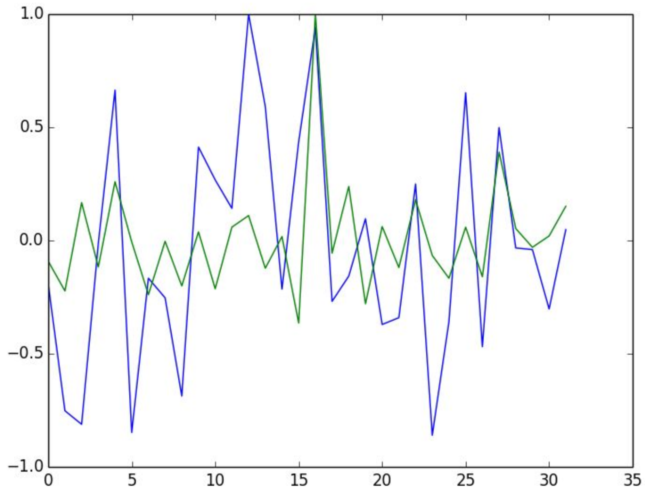
As it can be observed, the generative model created by the model's sparse coding does its best to match the given stimuli with a Gabor function. And it is somewhat successful in this endeavor. It correctly matches the highest peaks and lowest pits of the noise. However, since every projective field is a Gabor function with a different phi, the generative functions of the system are limited to Gabor functions, creating a clear limitation of this model.

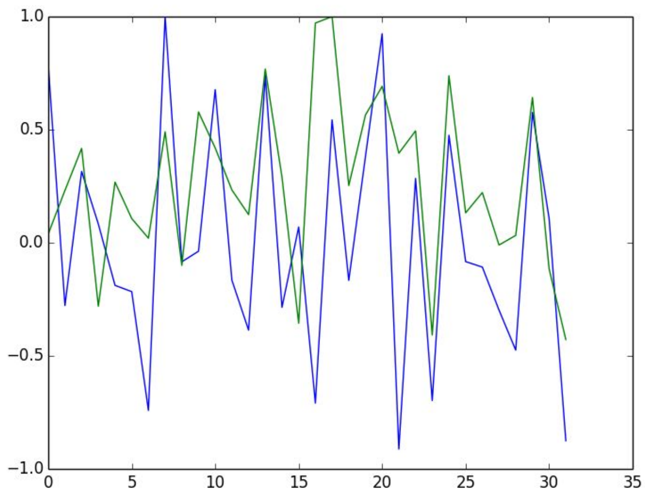
## Sine and Cosine Projective Fields

Seeing the limitation of Gabor-only approximations, we tested our model using a different projective field function: sine and cosine. We used the cosine function on the first half of the neuron projective fields and the sine function on the second half. First, we tested to see how it would approximate a Gabor function stimuli with phi set to 0.




We found this to be a fairly good approximation. Like before, we then tested it on some Gaussian noise:




Unlike before where we could only approximate with Gabor functions, by using sine and cosine as our projective fields, we were able to more accurately recreate the Gaussian noise.

## Conclusion and Next Steps

Much of the focus of this project revolved around the novelty of using integrate and fire neurons to achieve a sparse coding of sensory stimuli, which we proved to be possible. Through the use of Gabor projective fields, the accuracy of our generative model was limited to stimuli in the form of Gabor functions. However, as we saw with sine and cosine projective fields, it is possible to have more universal prediction power. In the future, we'd like to do more testing to discover what the optimal projective field function(s) would be.

## References

Hoyer, Patrik O., and Aapo Hyvarinen. "Interpreting Neural Response Variability as Monte Carlo Sampling of the Posterior." *Neural Information Processing Systems* (2003): n. pag. Web.