SYDE 556

Assignment 1: Representations in Populations of Neurons

Author: *Jonathan Johnston*

Course Instructor: *Professor C. Eliasmith*

This assignment is an introduction into the basic behaviours of a set of neuron models. It describes some ways in which neuron populations may hold a representation of a given stimuli, as well as the error associated in representation.

The assignment corresponds to the document hosted at:

http://nbviewer.ipython.org/github/celiasmith/syde556/blob/master/Assignment%201.ipynb

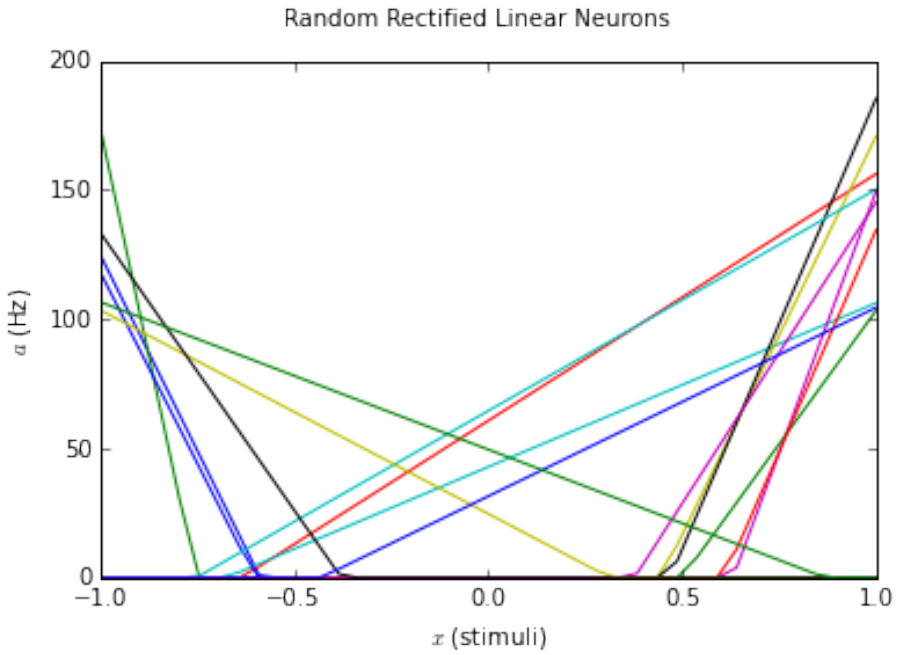
Section 1: Representation of Scalars

1.1 Basic Encoding and Decoding

In this section we define a rectified linear neuron model which will be used in subsequent parts of the assignment.

Part A In this section we generate a set of random rectified linear neurons and plot them together.

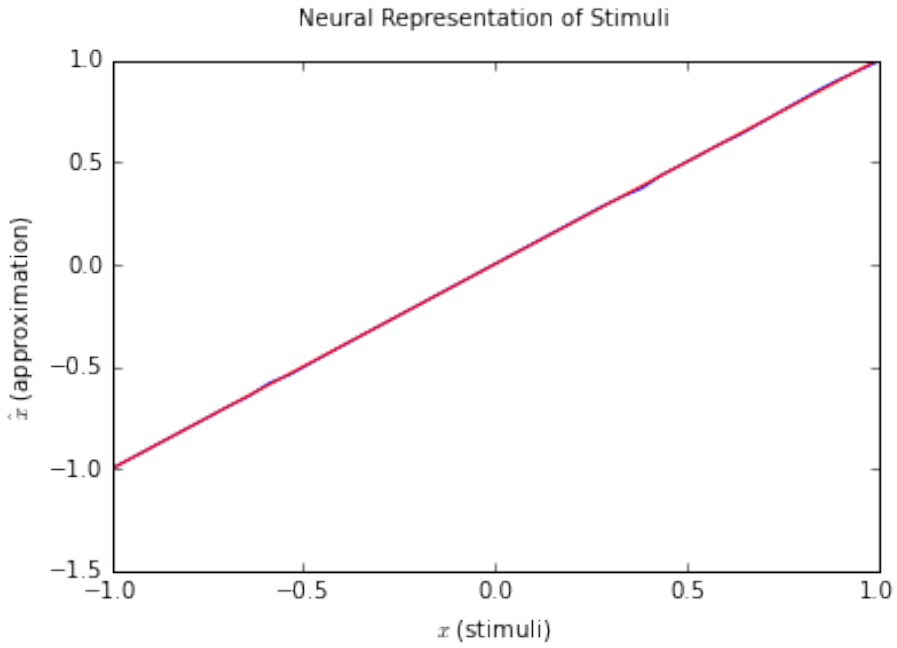
```
{'alpha': 302.7486688044811, 'x_int': -0.5863175132062743, 'max_fire_rate': 125.241822184527
```

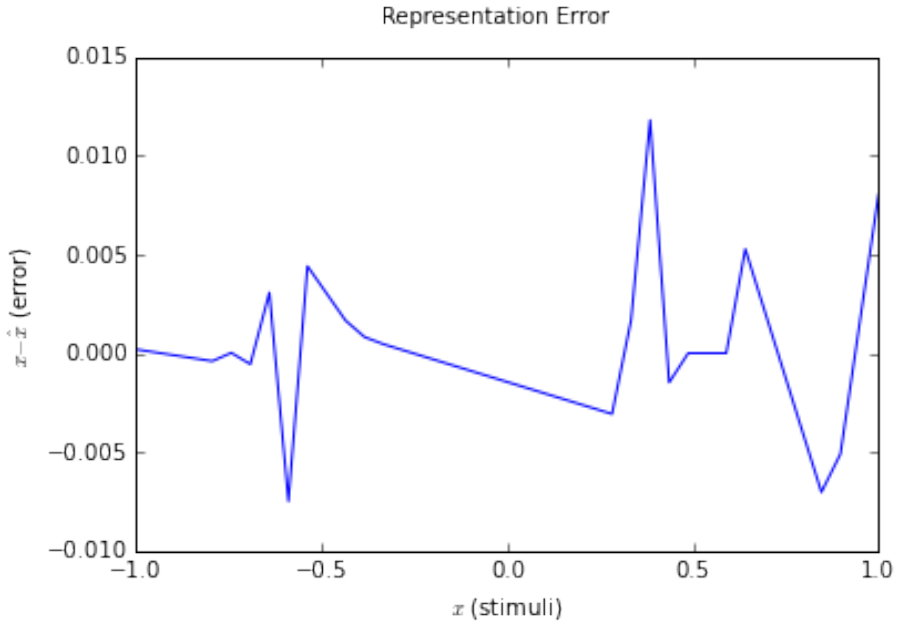


Part B In this section we find the optimal decoders for the generated neuron population.

```
Decoders:  [  1.27868286e-02   1.06104688e-03   1.47608793e-02  -1.92313556e-03
   7.89726219e-04  -1.45547825e-03   5.45000057e-04  -1.74013763e-02
   7.56167497e-14  -7.56554289e-04  -8.60648740e-03   2.28423639e-03
  -3.54890845e-03  -4.45060577e-05  -1.57825721e-04  -3.29282909e-03]
```

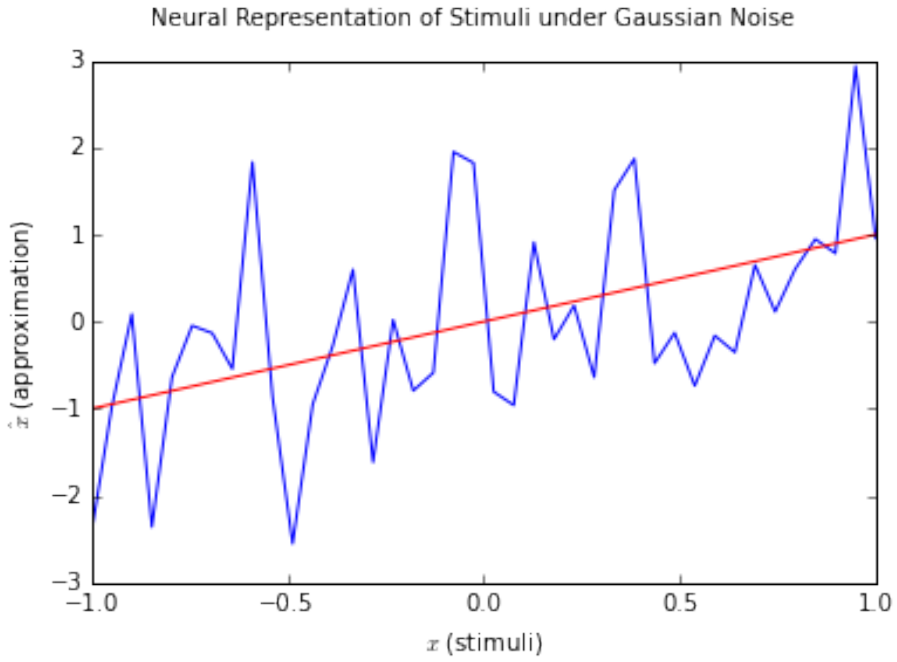
Part C In this section, we decode firing rates into an approximation of the stimuli, which describes how the neurons represent the stimuli.

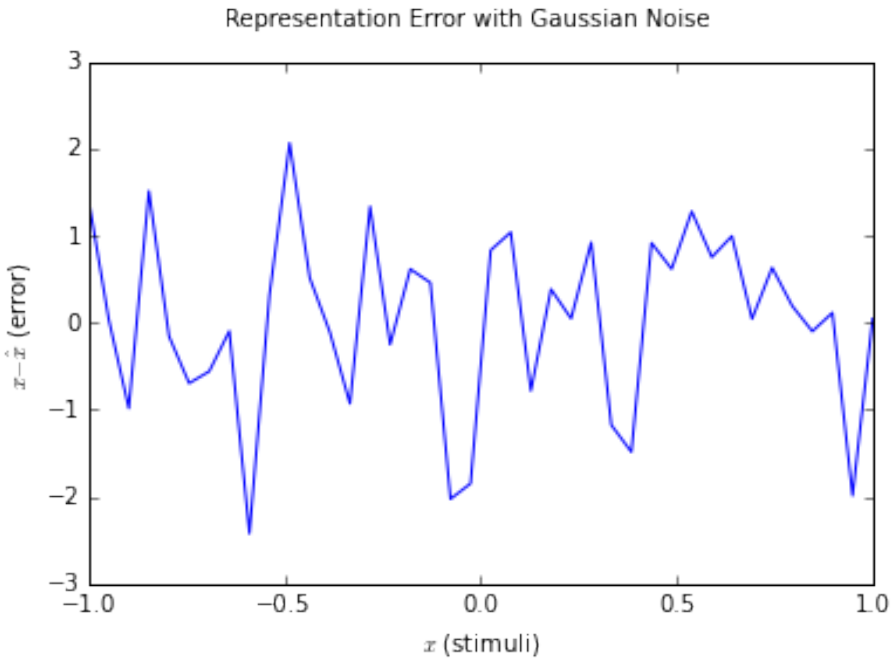





```
Root Mean Squared Error (RMSE):  0.00343041803281
```

Part D We decode under Gaussian noise proportional to the highest firing rate in the population.

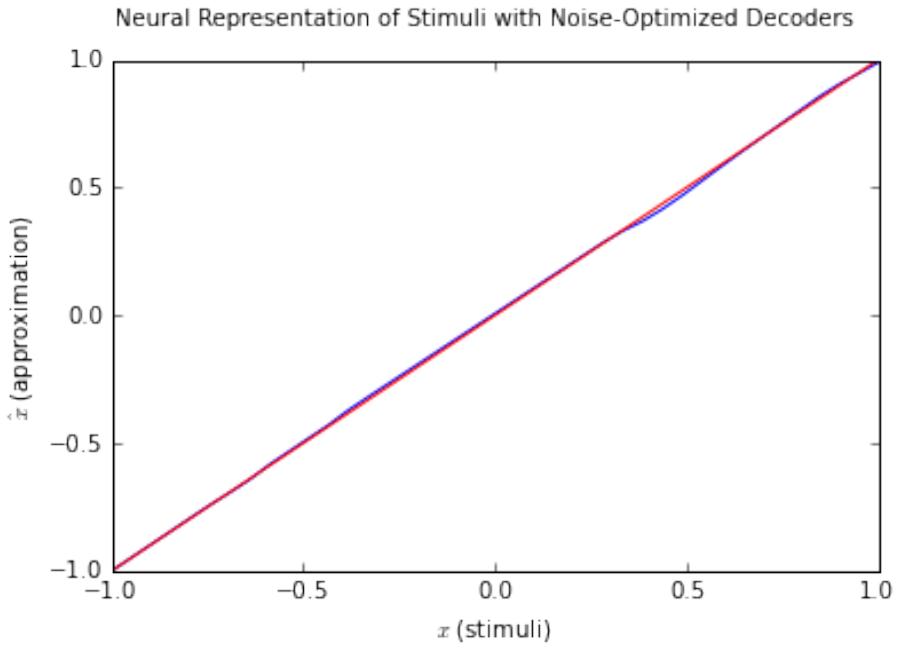


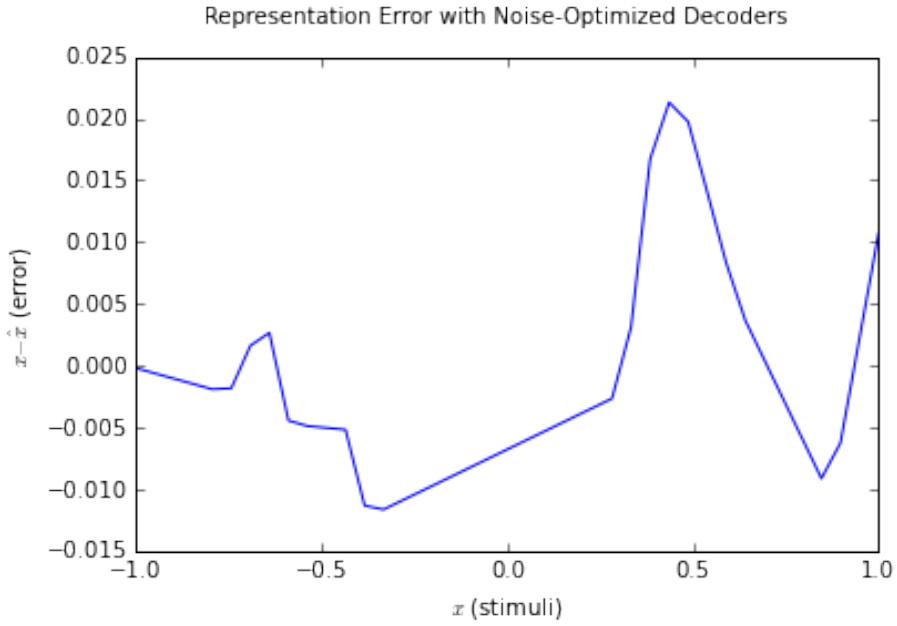



```
Root Mean Squared Error under Gaussian Noise (RMSE):  1.03866767172
```

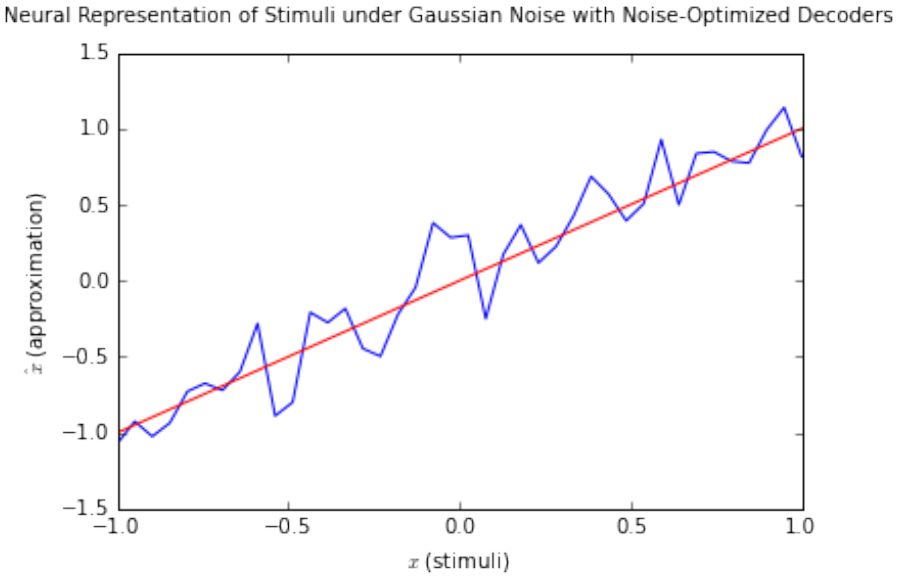
Part E We recompute optimal decoders to take Gaussian noise into account.

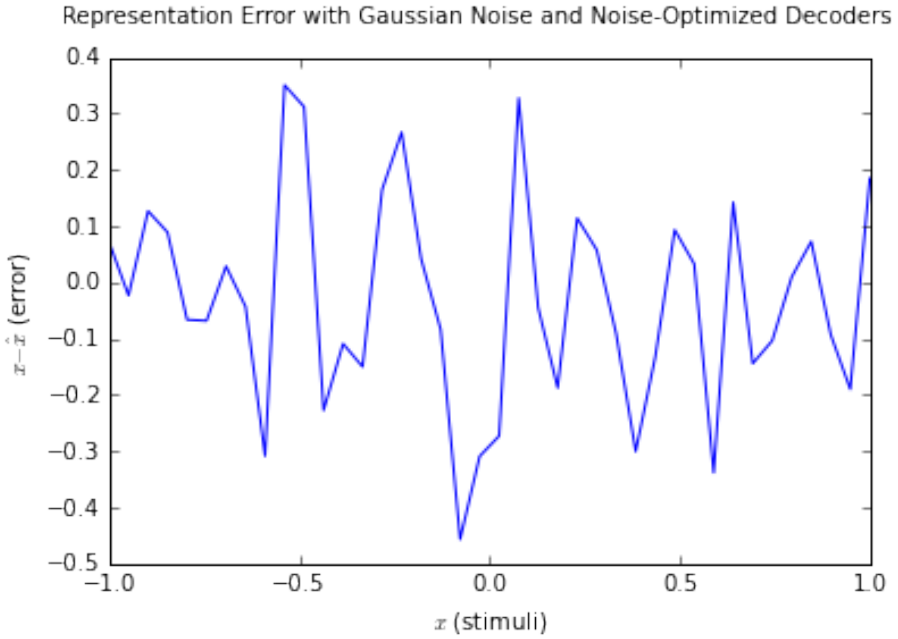
```
Decoders under noise:  [ -2.91049538e-04  -2.64506519e-04   1.77250765e-03   1.09927847e-03
  -5.32878835e-05   3.71258339e-04   2.35937776e-04  -2.73721939e-04
   2.29395810e-05  -8.18836806e-05   1.28951530e-03   8.78549601e-04
  -3.55041961e-03  -6.79393517e-04   1.75979242e-03  -3.99152543e-03]
```






```
Root Mean Squared Error with Noise-Optimized Decoders (RMSE):  0.00822719565029
```






```
Root Mean Squared Error under Gaussian Noise with Noise-Optimized Decoders (RMSE):  0.191346
```

Part F ‘Based on the RMSE of the variations of neuron noise and decoder noise ’ ‘compensation, it appears that the noise introduces a hundred-fold ’ ‘increase

in the RMSE that the normal decoder produces. Compensating ' 'for the noise by altering the decoder accordingly results in a better ' 'result. However, in the non-noisy case, the compensation results in ' 'higher error. In essence, using the decoder to compensate for noisy ' 'neurons results in a lower variability in RMSE, which means a better ' 'error in the worst case, but a worse error in the best case.'

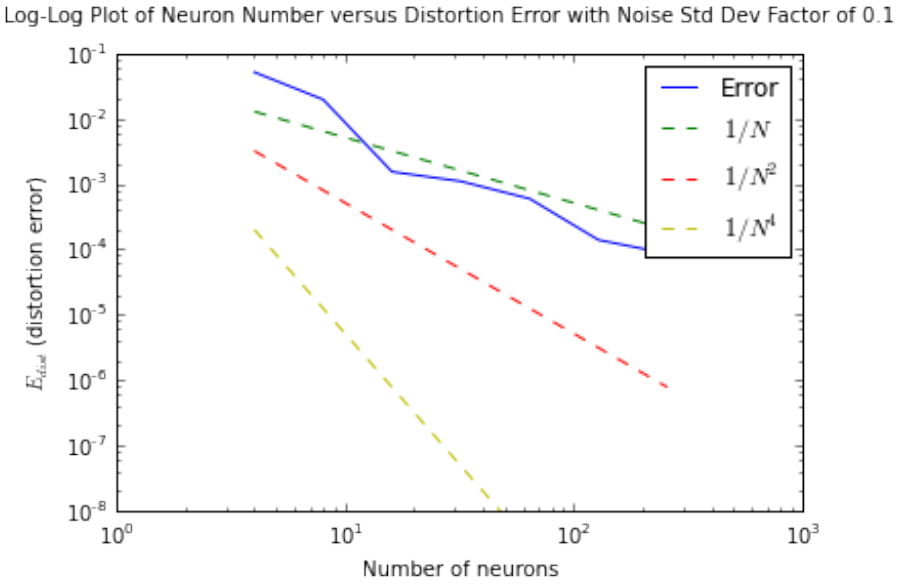
Comparison of RMSE for variations of neuron noise and noise compensation

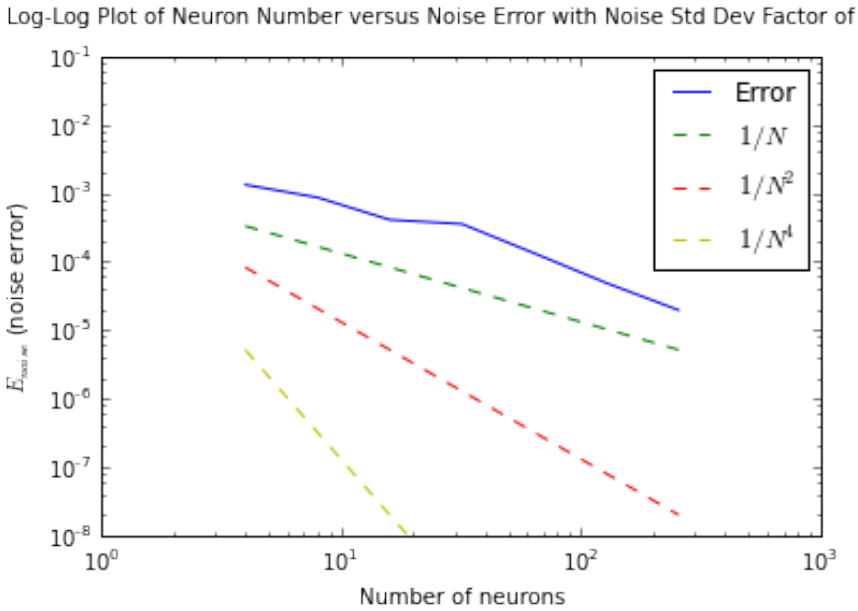
	Normal Decoder	Noise-Optimized Decoder
No Noise	0.00343041803281	0.00822719565029
Gaussian Noise	1.03866767172	0.191346117467

1.2 Exploring Sources of Error

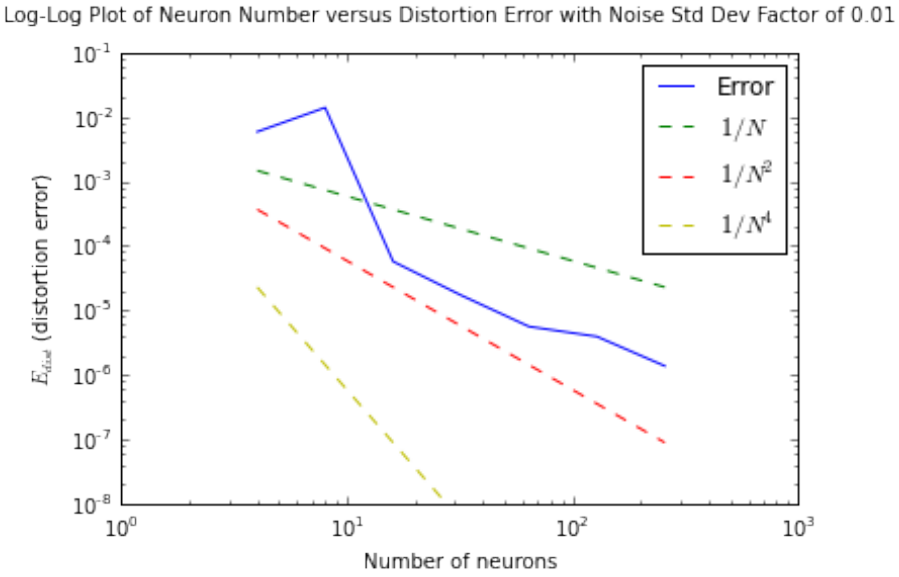
Part A and B Here, we explore various error associated with various numbers of neurons.

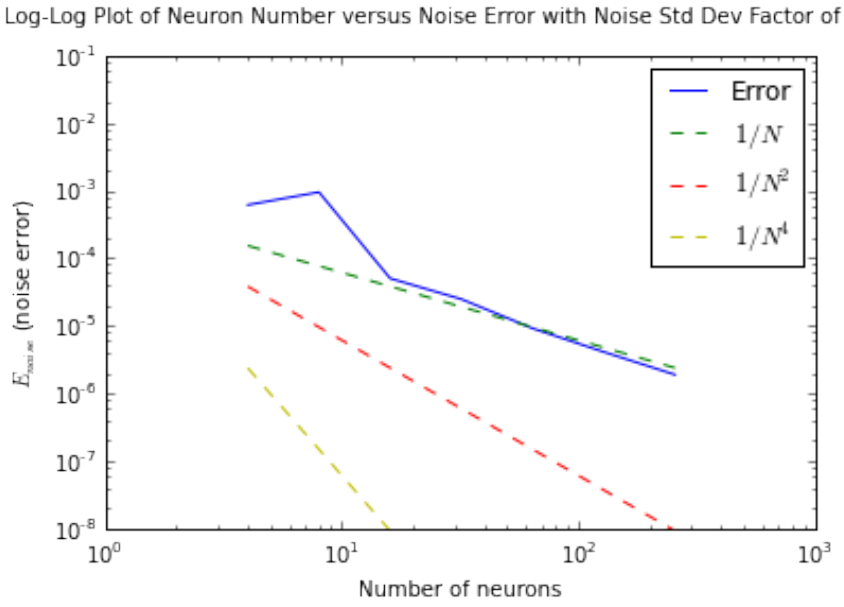
```
Number of neurons [4, 8, 16, 32, 64, 128, 256]
```






```
Standard deviation factor of noise:  0.1
Error due to distortion:  [0.050942052924142278, 0.019439286294447734, 0.00152847593754535
Error due to noise:  [0.0013214195808122174, 0.00086141188679938715, 0.0004073501777461629,
```






```
Standard deviation factor of noise:  0.01
Error due to distortion:  [0.0058457465175674132, 0.013862739790768162, 5.7198260071924361e-
Error due to noise:  [0.00061203493346832707, 0.00094586151683884493, 5.0210594997922611e-05
```

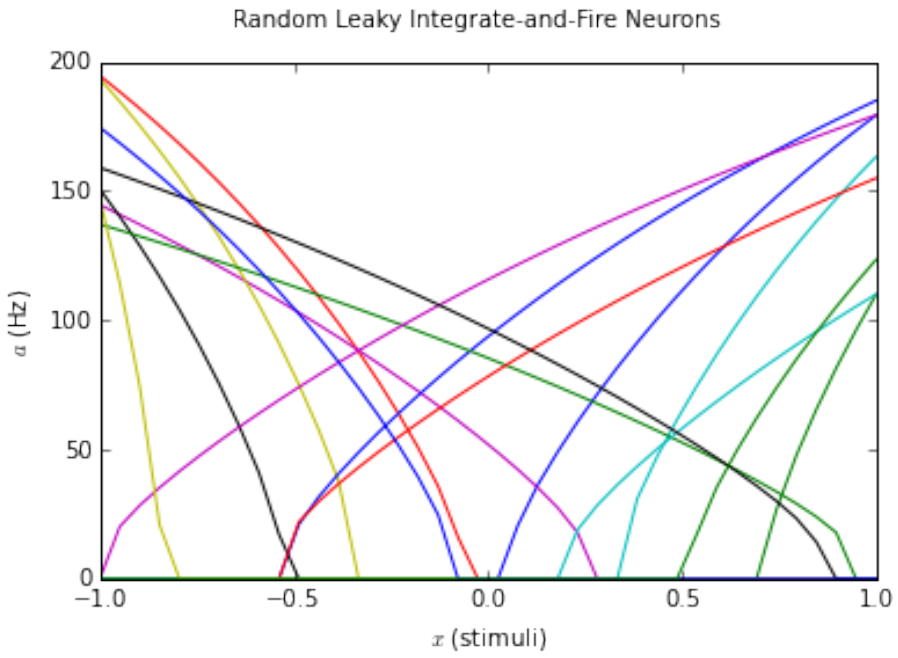
Part C Comment on differences in parts A and B

While there is a significant difference in the total error resuling from a change in the standard deviation of the neuron noise, the proportionality between the number of neurons and the total error from distortion and noise is constant. The error due to noise is related to the number of neurons with a 1/N relationship, and the error due to distortion is related to the number of neurons with a 1/N^2 relationship. A massive decrease in error (10^6 fold) can be seen, for example, by increasing neuron population from 4 to 256. Only a modest improvement in error can be seen with a decrease in standard deviation of the noise. Also, at low neuron populations, distortion error is greater but at high neuron populations noise error is higher. Therefore more neurons will encode a stimuli better than fewer neurons, but noise error will not be overcome as easily with more neurons.

1.3 Leaky Integrate-and-Fire Neurons

Part A Here we define a leaky integrate-and-fire neuron model.

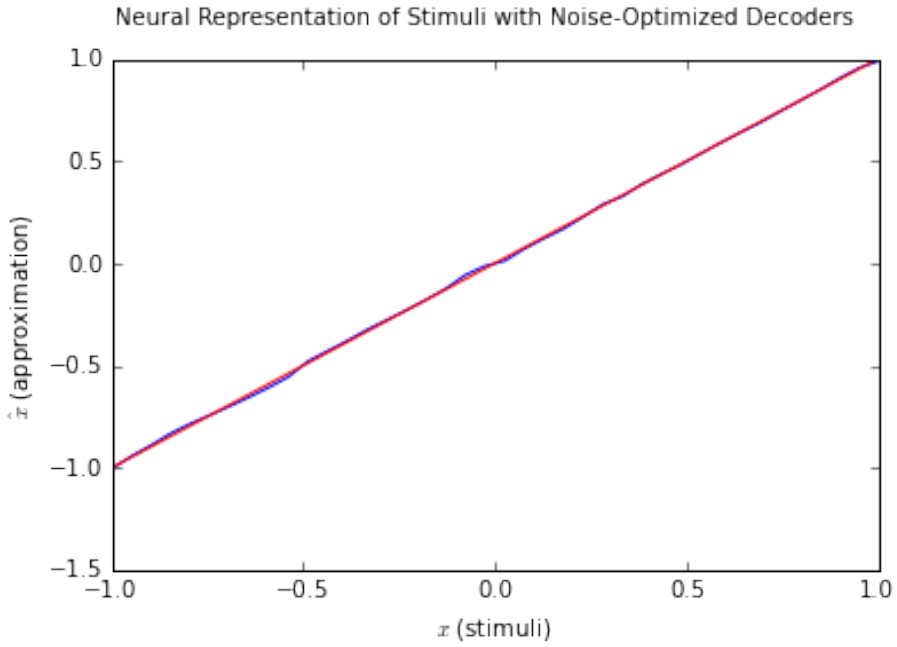
```
{'alpha': 5.4086794376427898, 'x_int': 0.056894806132190556, 'tau_rc': 0.02, 'max_fire_rate'
```

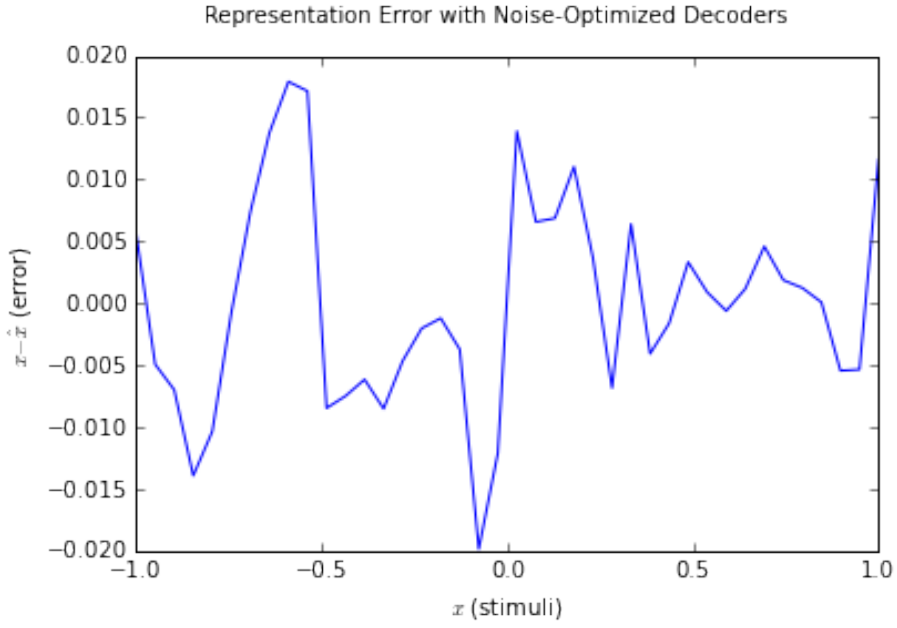


Part B Now we find the error due to noise.

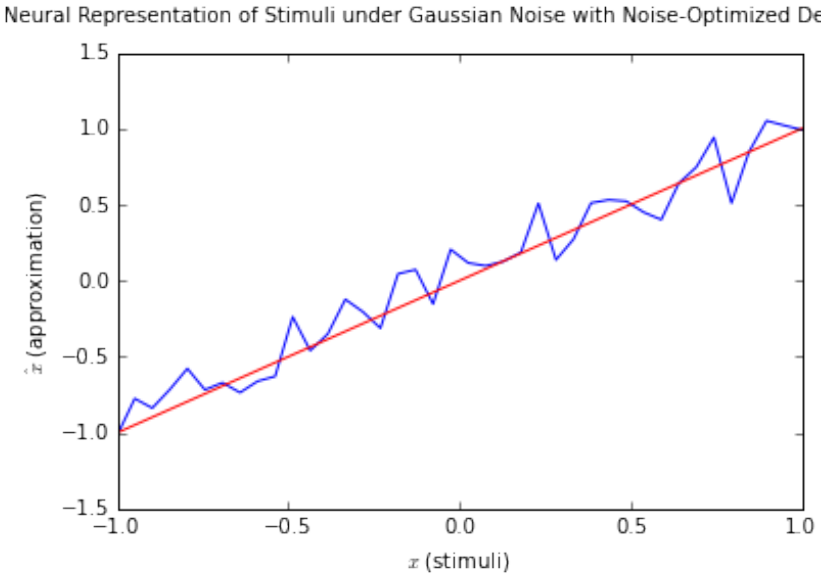
```
Noise-optimized decoders:  [ 0.00160684  0.00027159 -0.0011432   0.00080915 -0.00120326 -0.0
```

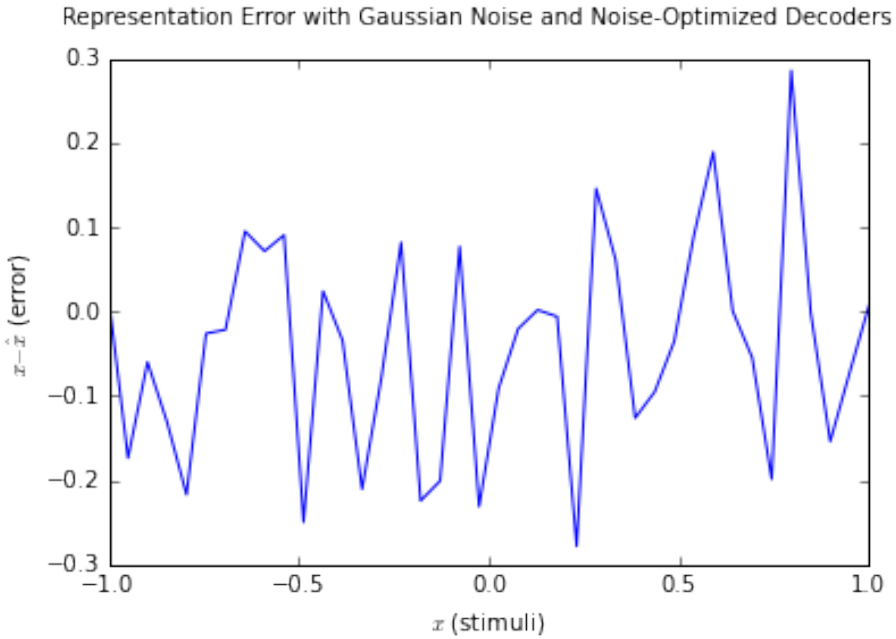
```
-0.00070381  0.00076376 -0.00086382  0.00055182  0.00059563  0.00101251
-0.00034474 -0.00099616 -0.00079977  0.00052491]
```





```
Root Mean Squared Error with Noise-Optimized Decoders (RMSE):  0.00843094365499
```



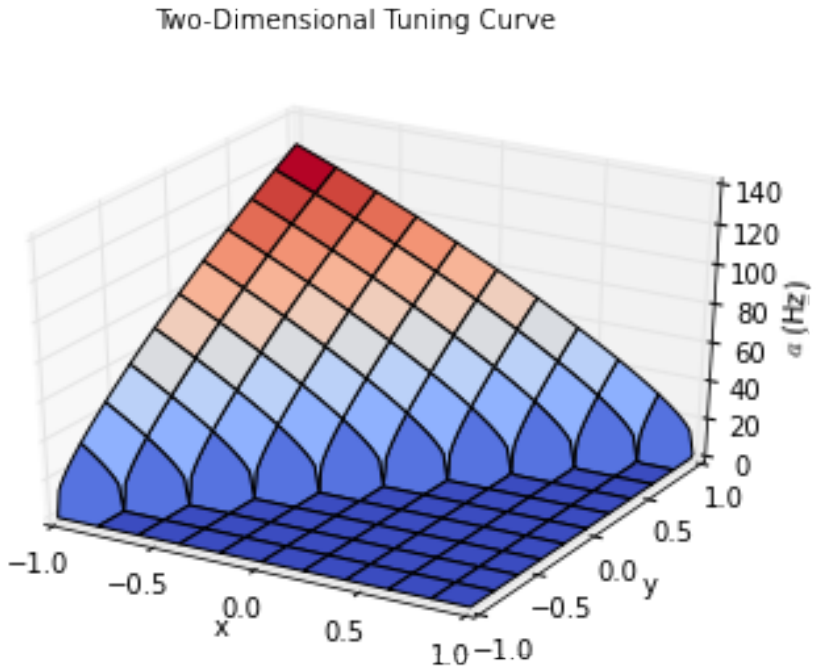



```
Root Mean Squared Error under Gaussian Noise with Noise-Optimized Decoders (RMSE):  0.134718
```

Section 2: Representation of Vectors

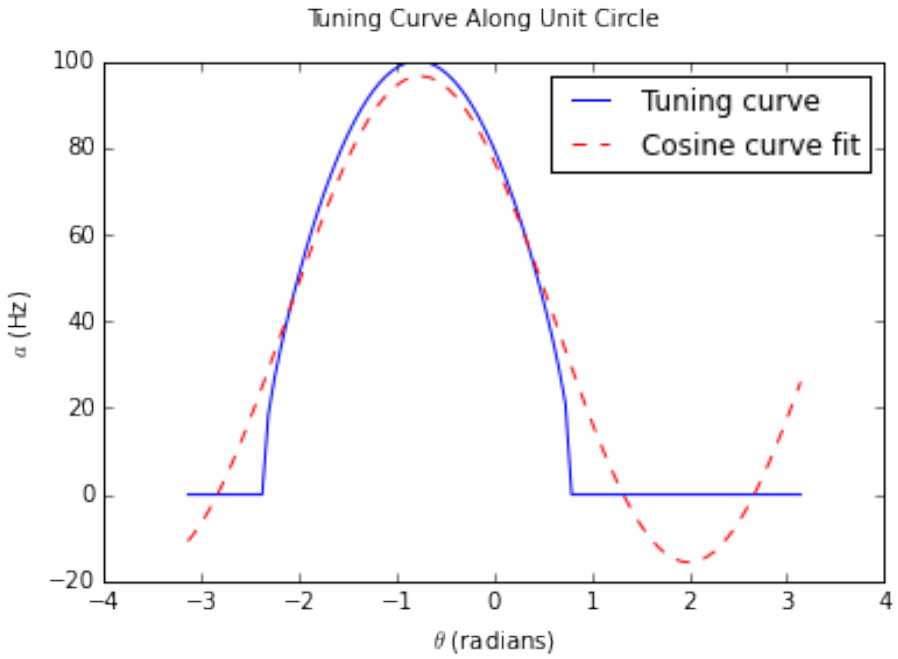
2.1 Vector Tuning Curves

Part A We now bring the previous LIF model into two dimensions. It is worth noting that the plot is representing the entire grid of -1 to 1 on both the x and y axes. This means that the tuning curve appears as if it hits a higher frequency than is defined in its max firing rate. This is not the case, however, because the activity should stop at the unit circle defined around the origin (0,0).



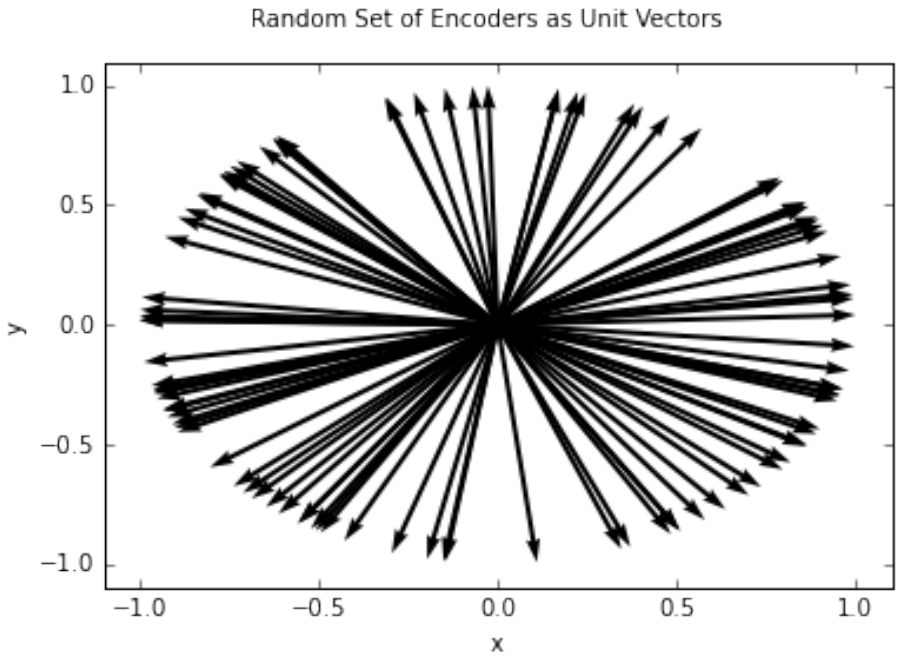
Part B In this section we use a set of points distributed evenly around the unit circle as stimuli to find the corresponding firing rates.

We also fit a cosine on the resulting curve. The cosine is a good fit because it approximates the nature of taking values of a sort of exponential function along a circular curve. Where it fails is in its periodicity. The tuning curve is only active in a defined region, and is zero elsewhere, whereas the cosine curve is periodic and has no discontinuities. The cosine is also not ideal because the function being approximated is exponential, and therefore many cosines would be necessary to approximate it perfectly.

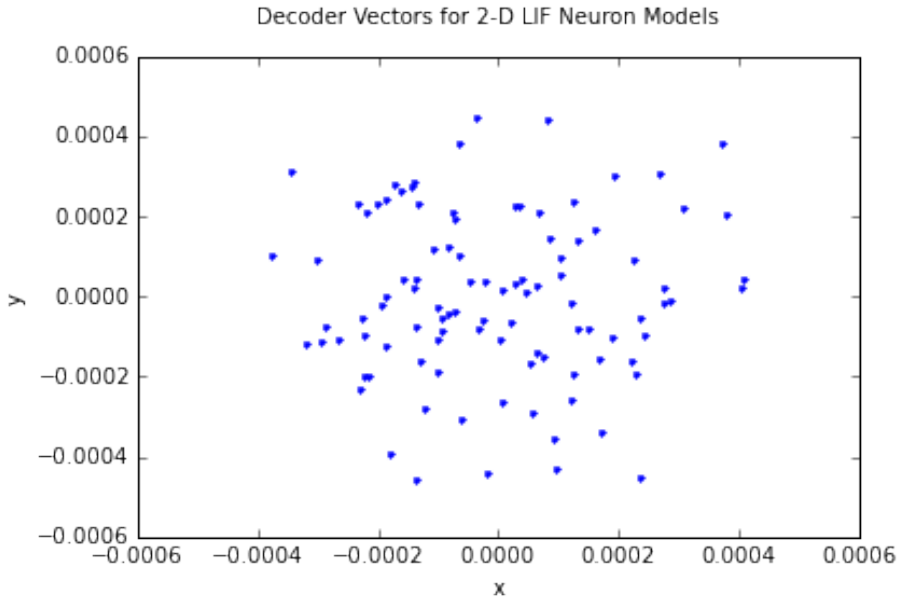


2.2 Vector Representation

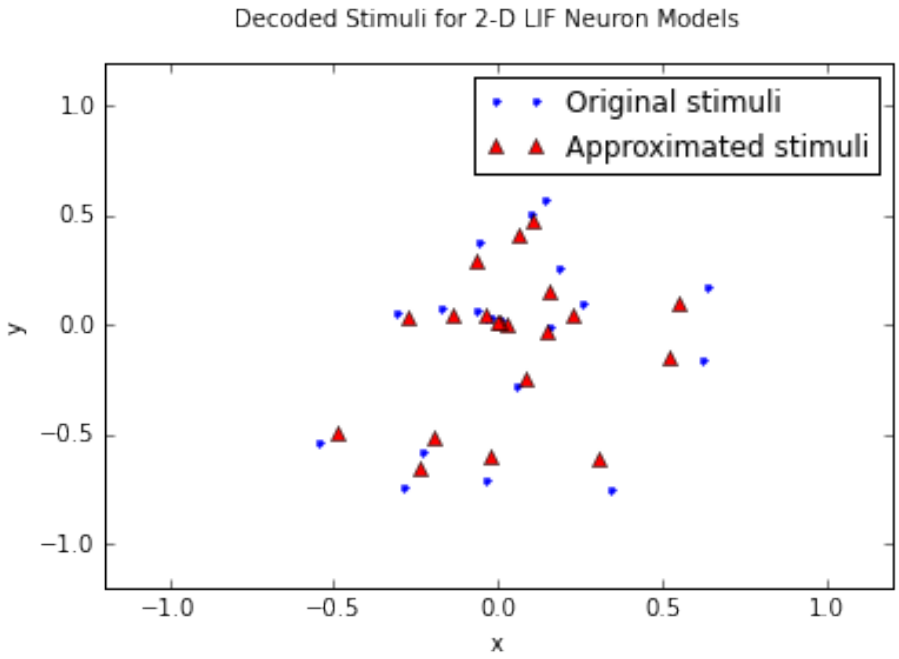
Part A We generate a set of 100 random, uniformly distributed unit vectors as encoders and plot them.



Part B The encoders all lie on the unit circle, and therefore have magnitude one. The magnitudes of the decoders, on the other hand, are extremely small in comparison. It follows reason that in representing a stimuli that will have at maximum a value of one, a weighted sum of the firing rates of a large number of neurons will result in each neuron's weight being extremely small. In addition, the angles of the decoder vectors are distributed somewhat evenly, as are the angles of the encoder vectors.



Part C Now we look at the stimuli as it is represented by the neuron population, and compare with the actual stimuli. As we can see, the accuracy of using the decoders to represent the stimuli is very good.



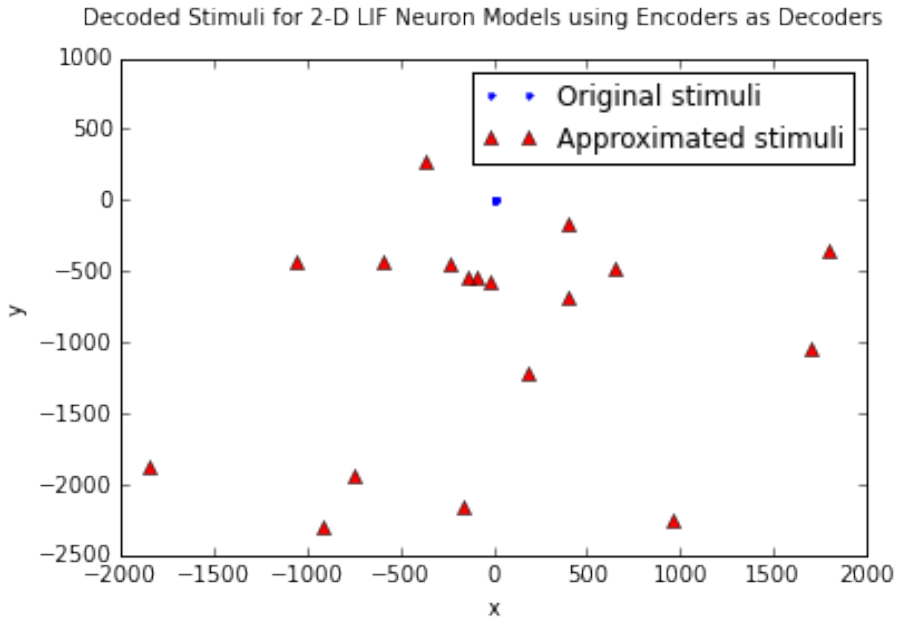

```
RMSE:  0.0580468379652
```

Part D Using the encoders as decoders, we repeat Part C normally. As well, we repeat C and D whilst ignoring the magnitude of the representation and stimuli vectors.

As we can see, without normalizing the result, the error corresponding with the use of encoders as decoders is extremely high. Using the decoders themselves results in a much lower error.

When we normalize the result of decoding with the encoders, however, the RMSE is much lower than previously, and is only about 7 orders of magnitude greater than using the decoders as intended. The interesting part is that when we normalize the result of the **decoders** being used as intended, the RMSE is much higher than is acceptable.

This result is interesting because it implies that the computational effort expended when determining decoders is not entirely necessary if a slightly higher error is acceptable. This means that there exists a speed-accuracy trade-off when considering the use of encoders or decoders to decode the firing rates of neural populations in order to represent a stimulus.




```
RMSE:  1031.52758096
RMSE, normalizing vectors:  17.0257207443
RMSE for use of encoder as decoder, normalizing vectors:  0.346794253054
```