# Proof of completion

# 

# Influence of learning factor

The learning factor of both file increases the learning rate of the net. When we uses variable learning rate, it improves the result of the net. However, the learning factor can be into harmonic oscillation where at that point, we need to change the value of the magic number in eta=eta\*magic number. If we set the magic number too high, we can have instability oscillation where every time it just overshoot. A magic number around 1.01~1.05 is recommended and I do see improvement on learning rate for the preset net file. The learning factor caused the eta value to be in an sine-ish form where when it is small, it will increase to larger, and when it is too large, it will drastically decrease to smaller steps.

# Accuracy achieved in training (e.g., vs number of interneurons and number of iterations)

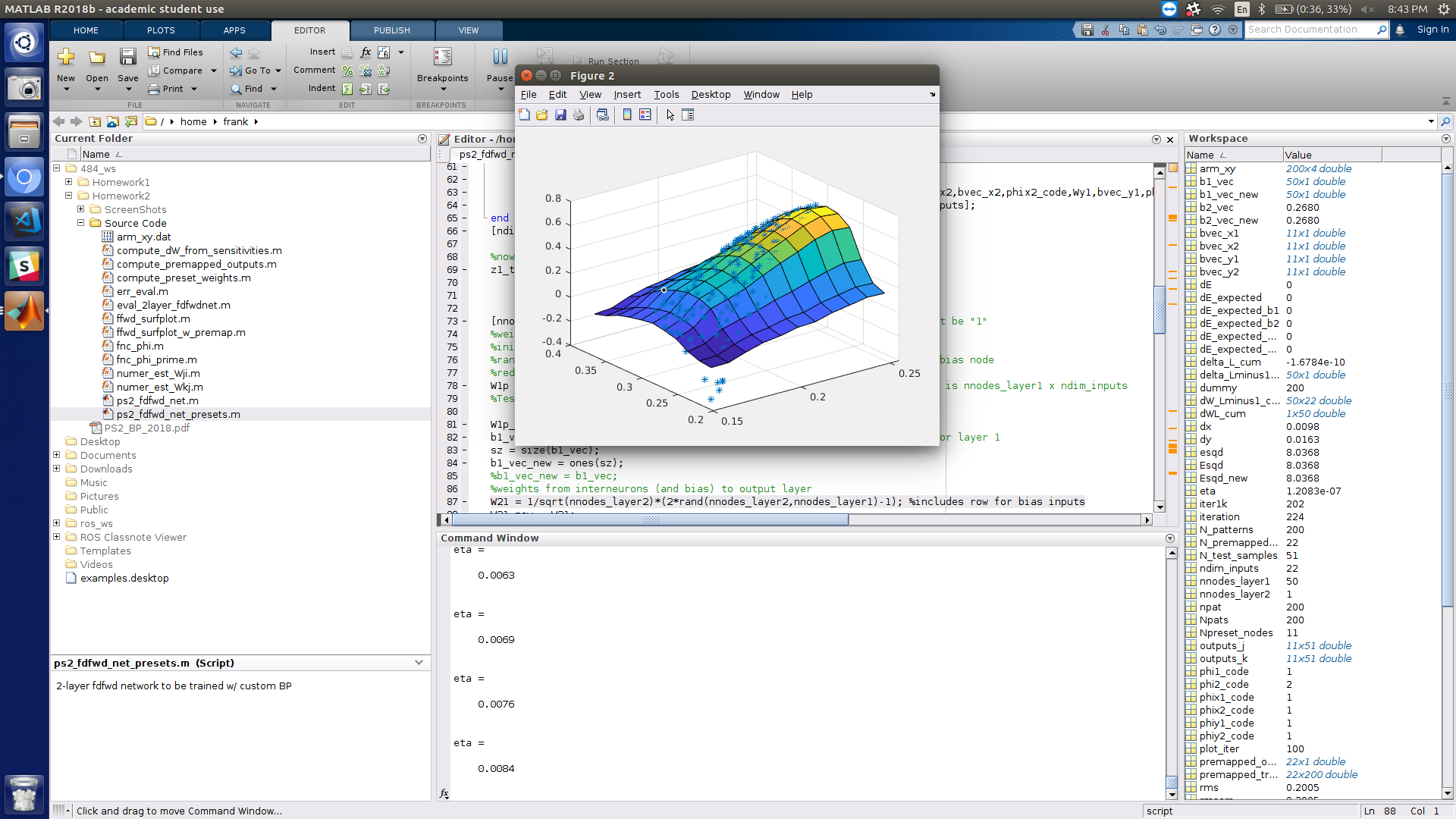
With the increase of interneurons, the net tends to converge faster. I found that if I change the net by the degree of 10, neuron at 500, I can save half of the iteration and be able to get the pretty smooth net around 500 iterations. At about 1000 neurons, I found that the network is no longer smooth but rather edgy and sharp. It do fit every point but I think we might have over trained the net. However, if we have too much neurons, we run into issue of the network will not converge.

# Differences between 1 vs. 3 hidden layers

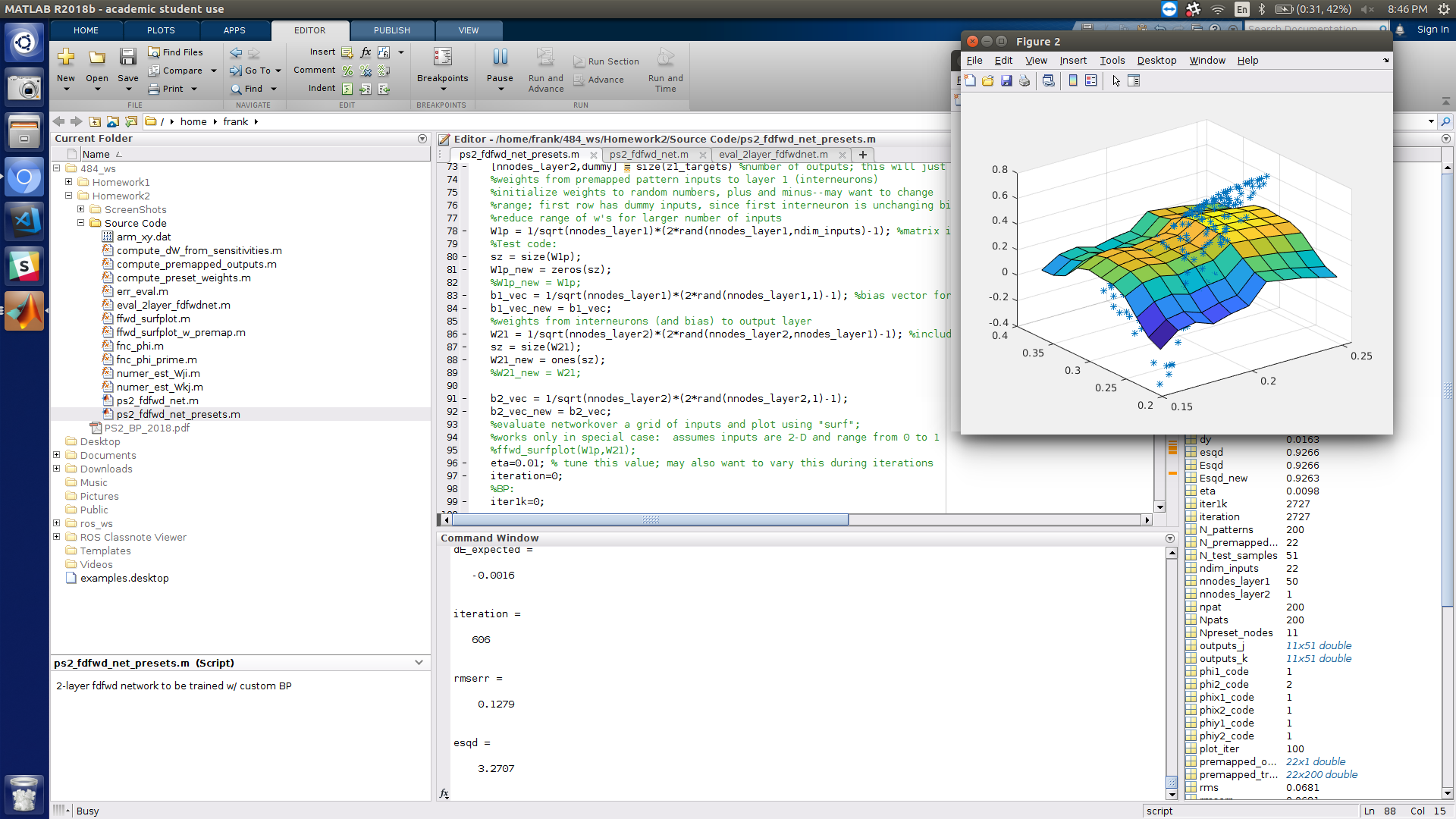
The biggest difference of 1 vs 3 hidden layers is speed of training. For three hidden layer net, the convergence speed is to the degree of 100 times faster. What’s more, the net result is more “fitted” to the data sets and more data are fitted to the net. For 1 layer network, it took about 5 hours of running to finally agree with the datasets. With 3 hidden layer, we can have a faster converging rate about 1k iteration later, we can see the result.

# Alternative values for the fixed weights

By changing the weight, I do see the result changes. With W1p set to zeros and W21 set to ones, I can achieve a very fast learning rate. However, if I used all ones and all zeros for both fixed weight, it doesn’t seems to have too much of an effect. The net is slower compared to the default configuration.

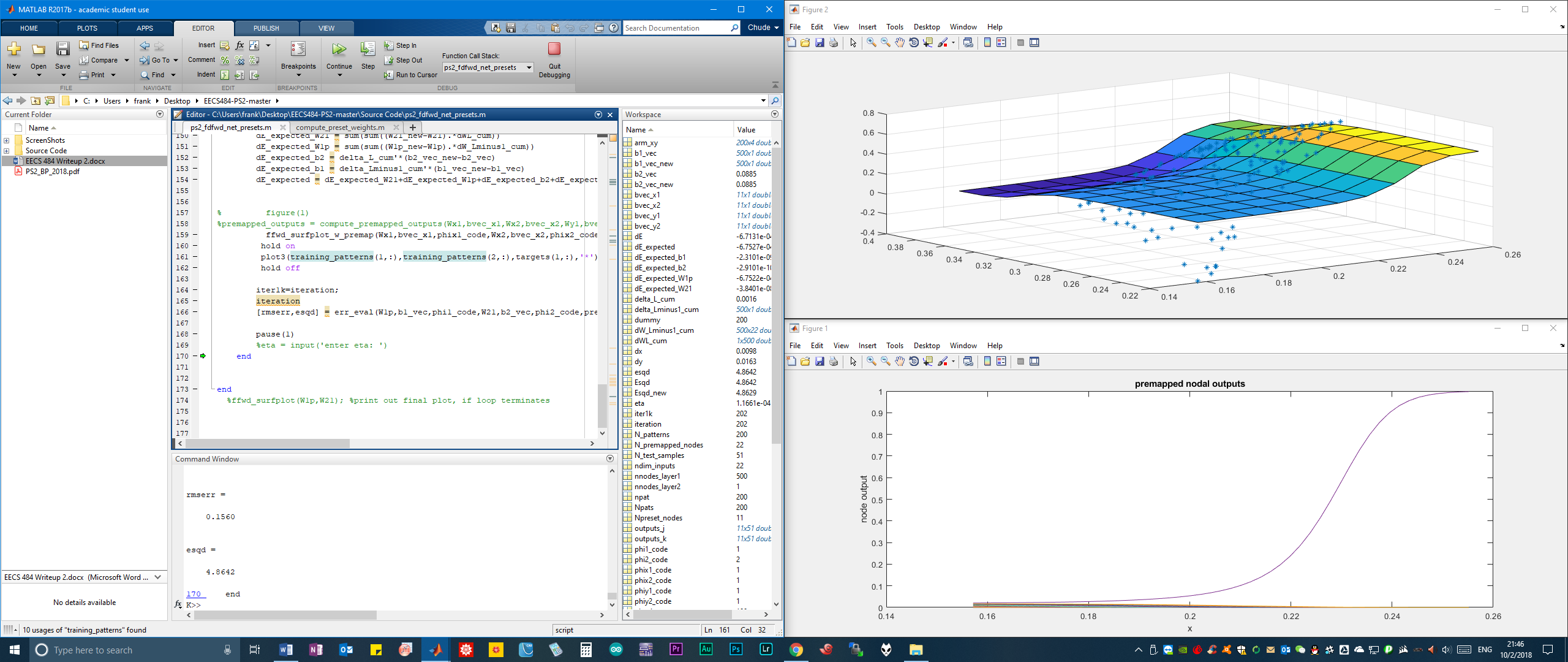


Beta value to zeros...

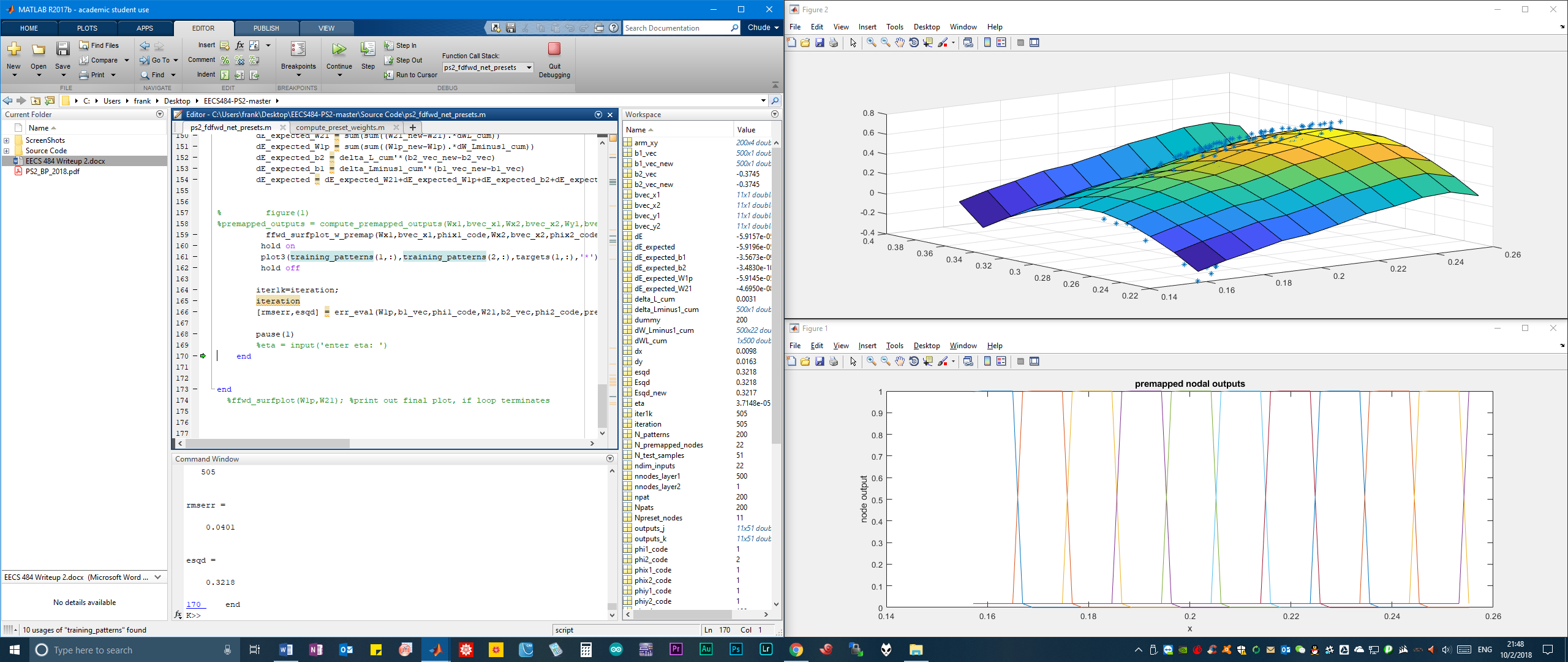


zeros and ones for the two weights

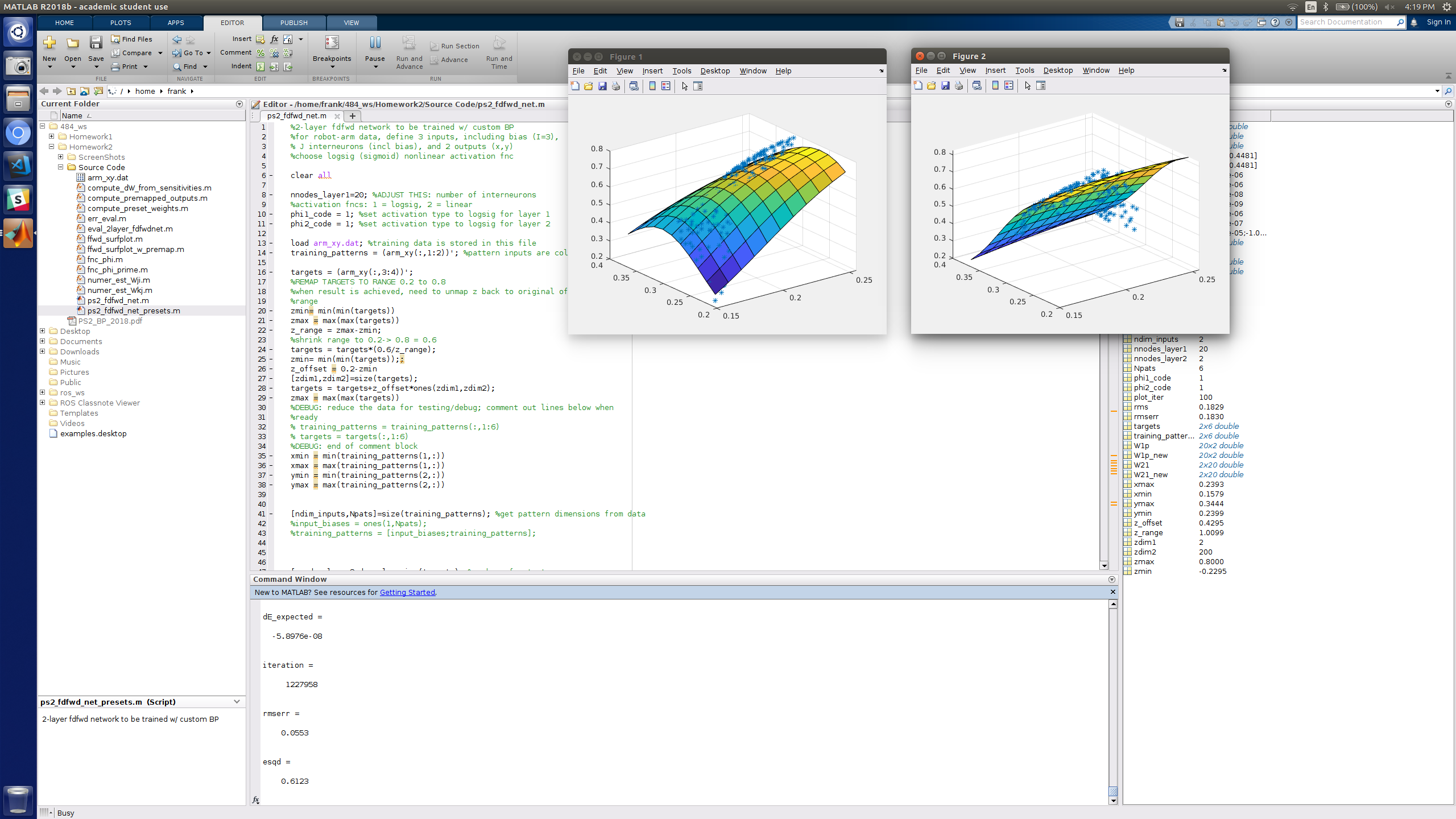
If we change the weigh scale value to a very small value, we can see the system reacts slower:



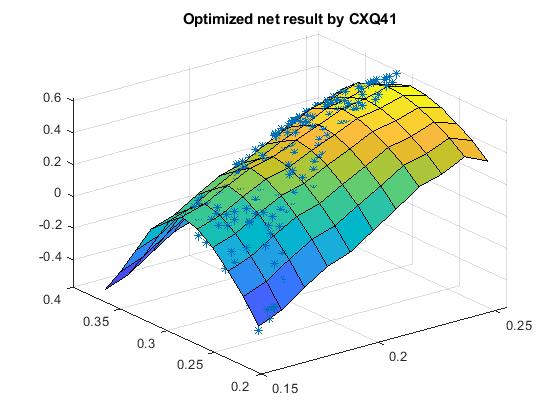
If we increase the weight scale to an arbitrary large number, the net will operate marginally faster, just a little bit. However, the result seems to be smoother compared to the previous ones.



# Show plots of your results



one-layer result after 5 hours operation



three-layer net after 2k iteration

# Report on any other observations or variations

The pre-defined weight for 3 hidden layer has a large effect on the learning of the net. If we get “lucky” and get a relatively close weight, we will see the convergence of the net very fast. However, if we have choose bad pre-defined weight, it will take longer for the system to converge.

If we change the final layer activation function for the “preset” file into logsig, we will have a smoother learning result, however, some of the points at the bottom corner will not get covered well. The learning rate also get decreased.