# Methodology

Two different structures of ANN were tested. One had 1 hidden layer, the other had 2. Both were fully linked and implemented in the latest python 2.7 using the pybrain package **(REF1)**. The function implemented in the hidden layer nodes was chosen to be a sigmoid function, with a linear function in the output layer. This choice comes from recommendations in the literature that suggest such a combination is robust and flexible **(REF2)**.

In all cases the ANN received 5 inputs, in this order

1. Illumination
2. DCW Concentration
3. Nitrate Concentration
4. Lutein Concentration
5. Nitrate flowrate

Of these, Inputs 1 and 5 are controllable and known at all times in a realistic production setting. Inputs fed to the ANN were rescaled, as the variables in the system under analysis are of very different size and ANNs require similarly scaled inputs, as their training benefits in terms of speed and numerical performance by having a search space that is evenly scaled on all axes. This rescaling was done through one of the following techniques.

Consider a list X of variables x\_1,x\_2,x\_3, x\_i etc. These are the inputs into the system.

They were standardized. This centers the data range around the mean and reduces the standard deviation to 1.

x\_i\_Standardized=(x\_i-median(X))/(std\_deviation(X))

The outputs were concentration differences for DCW, Nitrate and Lutein (which we will abbreviate as DeltaX). These come out scaled from the ANN, and in simulations they are added to the current scaled system variables to predict future points in the trajectory. Trajectories are rescaled successively.

Through literature correlations an initial number of nodes was initially estimated to be around 20 **(REF5-6)**. These however were tested through a simple hyper-parameter, together with the number of training epochs. Experimental data sets, each with 12 points, were employed in a hyper-parameter selection step. The choices for the number of nodes in the hidden layers was [5,10,15,20,30]. The choice for the number of training epochs was [15,30,50,100,150,200,300]. The following graphs show the 3d landscapes representing the intersection of these two choices and the error reported as in Figures **FIG**. For the number of epochs, the error decreases rapidly at the beginning and seems to flatten in either cases long before the maximum of 600 training epochs. As a rounding up, 400 epochs were chosen. As for the number of nodes, in the single hidden layer case adding more than 20 nodes doesn't seem to carry an improvement in performance, while in the case with two hidden layers a significant increase in the number of nodes beyond 10 seems to even contribute to a decrease in performance. In the end, 20 nodes were picked for the first case, and 15 nodes per hidden layer in the second case.

For training, 5 sets of experimental data points, each containing 12 data points, were used. To enhance the data, 50 replications were produced of each point with random noise added of 3 % of the variable size and a further 50 with a 5 % noise. These proportions were selected based on a realistic assessment of the accuracy of the original experimental measurements. An example of a fully simulated trajectory when compared to one of the training sets is visible in Figures **FIG.**

After being trained all types were then used to check on the two test sets, which will be named here TestSet 1 and TestSet 2, never seen before by the system. Although the trends in concentrations were the same, these differed in critical quantities such as the nitrogen inlet, as reported in the data. The former was 127.5, the latter flowrate was 25.5.

In the type of simulation presented here, the trajectory was fully simulated. Starting only from the initial experimental point, a simulated point was computed as in the previous case. However at the next step, instead of using the next experimental point as input to the ANN, the computed simulated point was used. This means that, except for the first point at t=0, the ANN never sees again an experimental point, thus any error is eventually accumulated and the fidelity of the ANN to the test set trajectory is solely the merit of good general.

# Discussion

Across the whole variation, performance seems robust, reproducing mostly reliably the trajectories and showing few points with any substantial deviation from the original trajectory.

This section will concentrate on what differences are most relevant in this case. The results for the 2 hidden layer test set have been shown, as one hidden layer turns out to be . Additionally, of the test sets used, Test Set 1 and Test Set 2, the latter shows the largest deviations in both 1 and 2 hidden layers, though still with good predictive capabilities.

The tendency for a slightly larger amount of error consistently seen in one of the two test sets (Test Set 2) seem to indicate a problem with bias in the training sets, mostly in the concentration of nitrogen, where the increase in production influenced by the nitrogen injection is not well represented. This can be attributed to a relative lack of variety in the training set, as the nitrogen flow input has only 2 different quantities used as input. This seems to be further compounded by the fact that many differences seem to start around *t=60* , which is when nitrogen flow starts. An input that does not vary much might cause more error than the addition of a feature would correct **(REF3)**. This could be solved by having a few more data sets with different nitrogen flowrates as the main point of change.

As the number of data points for each set is relatively low, problems in variance are harder to diagnose as the definition of the curve is less defined, although the proposed ANN seem to capture the variance well enough from the shape of the trajectories for nitrogen obtained. Differences between simulated and experimental trajectories are mostly related to the already mentioned start of the nitrogen flow.

Furthermore, one aspect that should also be considered is how Standardization and the centering around the mean seem to have worked best in this case in scaling the points and obtaining a better prediction power. This can indicate that the ANN, when trained, becomes more flexible where items that maximize the variance are present. A min-max normalization would have not captured this characteristic as well, and in the presence of larger outlier would have forced many of the scaled points to lie very close to each other.

Connected to this is the fact that a sigmoid function was chosen for the hidden layer. Sigmoids have a tendency to saturate and have a worse learning if the inputs are very large for either sign, as the gradients inside the ANN are flattened, which can lead to serious problems in learning **(REF4)**. Standardization solves this issue by concentrating inputs in a limited range around a mean. Furthermore, since this mean is 0 and the inputs are transformed so that they can be either positive or negative, this also means that the input will not concentrate on one sign and the optimization process will be smoother as the gradient descent will be more precise.

Finally, the most important intuition that should come from this simulation is that the well fitting aspect of the two hidden layers ANN, when compared with the the single hidden layer type as in Figure **FIG**, suggests something else. A larger number of neurons obviously helps by enhancing the number of parameters available, however this is not the only way it can improve the quality of prediction. Having a second layer rather helps describing “higher level” features of the system under analysis, which are further composites from the features of the first layer. This can be used to further understand the underlying dynamics of the real system investigated, meaning that it is probable that the relation between concentrations is non linear **(REF2)**.

**REF1**

Schaul, T., Bayer, J., Wierstra, D., Sun, Y., 2010. PyBrain. J. Mach. Learn. 11, 743–746.

**REF2**

Nielsen, M., 2014. Neural Networks and Deep Learning [WWW Document]. URL http://neuralnetworksanddeeplearning.com (accessed 6.24.15).

**REF3**

Hagan, M.T., Demuth, H.B., Beale, M.H., 2002. Neural Network Design.

**REF4**

Andrej Karpathy CS231n: Convolutional Neural Networks for Visual Recognition. Neural Networks in http://cs231n.github.io/neural-networks-1/

**REF5**

Lawrence, S., Lee Giles, C., Chung Tsoi, A., 1996. What Size Neural Network Gives Optimal Generalization? Convergence Properties of Backpropagation.

**REF6**

Elisseeff, A., Paugam-Moisy, H., 1997. Size of Multilayer Networks for Exact Learning: Analytic Approach. Adv. Neural Inf. Process. Syst. 9 162–168.