**Jonathan Samson**

**Brandon Chin**

**CMSC 409 – Project 3**

**1) What are the inputs and outputs for this problem.**

There is only one input and one output for this problem. The one input is the time in hours, and the one output is the hourly energy consumption in kW. However, the goal for this project is to find a polynomial regression that approximately matches the data. So, the inputs for the neuron will be slightly different than just the time of day. A polynomial function of degree 3 would look like *a+bx+cx2+dx3.* In this problem, we treat the time of day as *x*, and look for the different weights *a* through *d*, where *a* is the bias and b through c are weights 1 to 3. We have multiple inputs in this case, *x*, *x2*, and *x3*, where x is the time of day. We also create models for 1 degree, 2 degree, and 3 degree polynomial regression functions, and the inputs vary accordingly.

**2) What should be the activation function of your decision unit? Why did you choose it?**

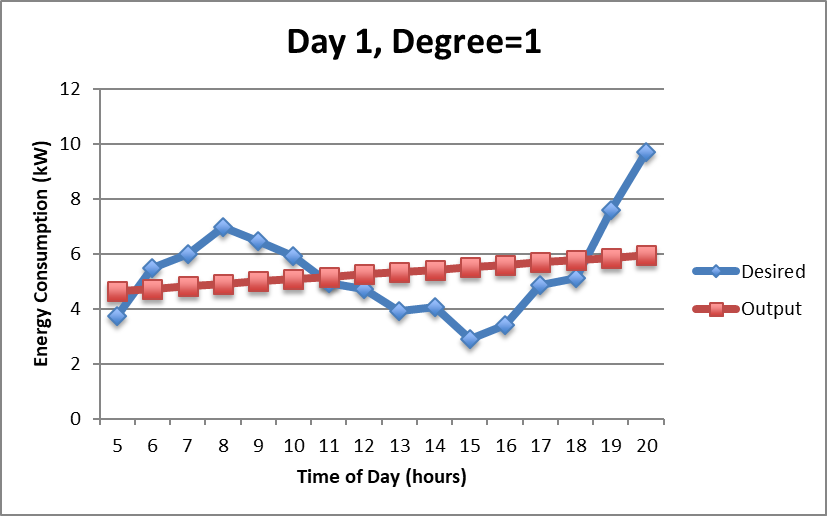
The activation function of the decision unit should be a linear activation function. This is because we are looking for continuous real values, rather than a decision function. In a normal decision problem, we would want values between 0 and 1 to classify the data, and would measure error as the distance from output to the correct answer. In this problem, the correct answer is not simply 0 or 1, but could be on a wide range of values measuring energy consumption. So we use a linear function which can go from negative infinity to positive infinity, in hopes our output will be close to the energy consumption.

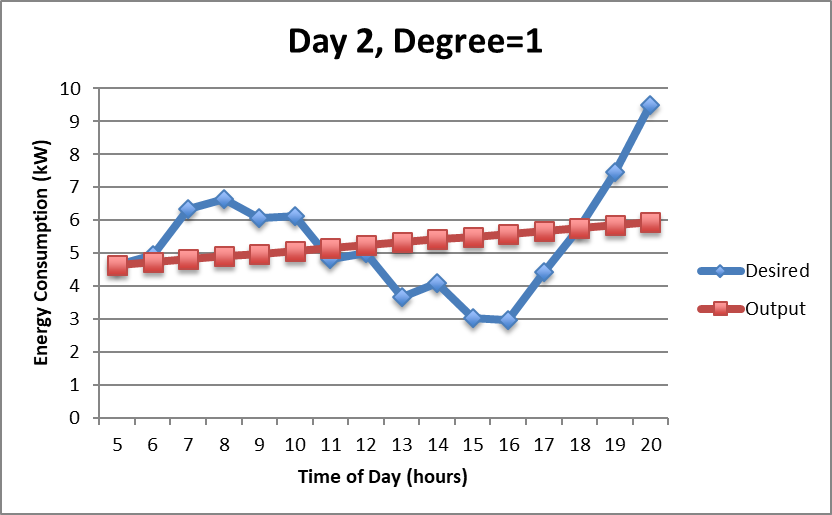
**3) Compare the training and testing errors obtained using the architectures on Figure 1**

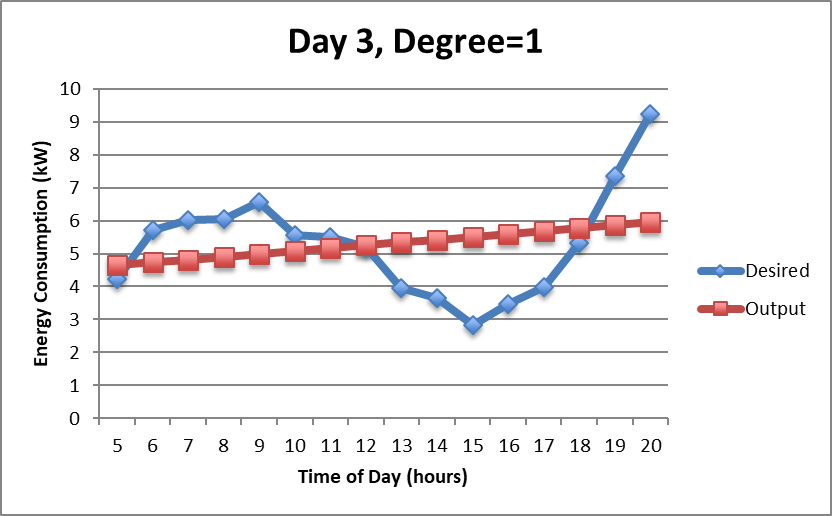
**a. Train the decision unit on the data from the first 3 days. Report training error for each of the three days. Present a graph (original data vs. trained model), similar to the Figure 2.**

We created graphs of the output and desired results for days 1 to 3, and models of degree 1 to 3. Total error is calculated as the square root of the sum of squared errors.

**Degree 1**

****



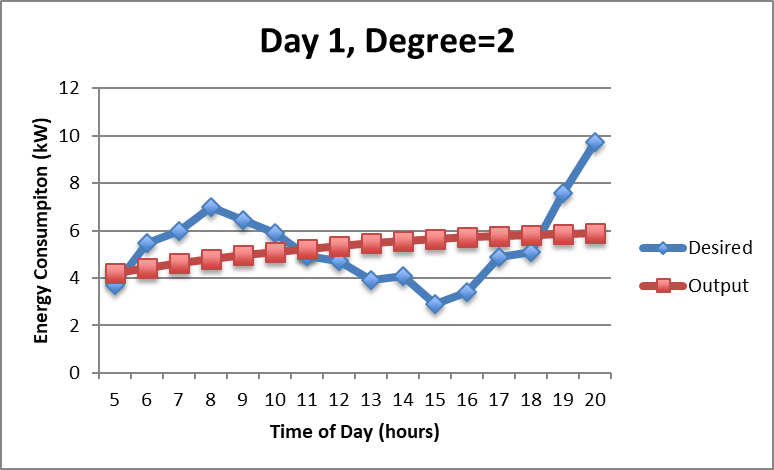
****

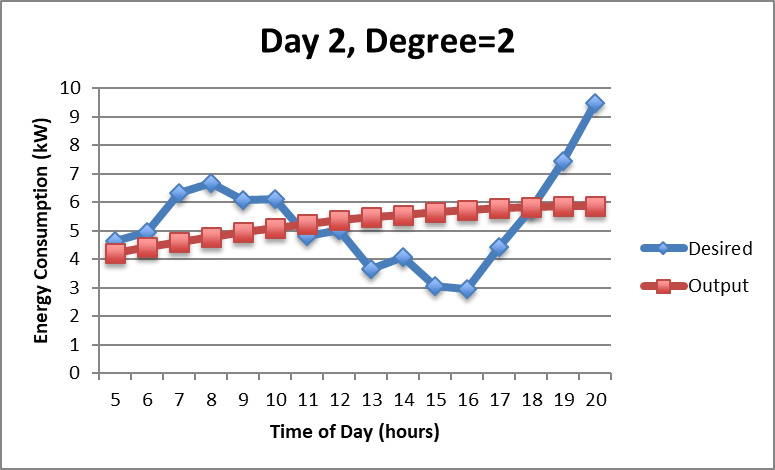
**Total Error for Day 1:** 6.649214787984489

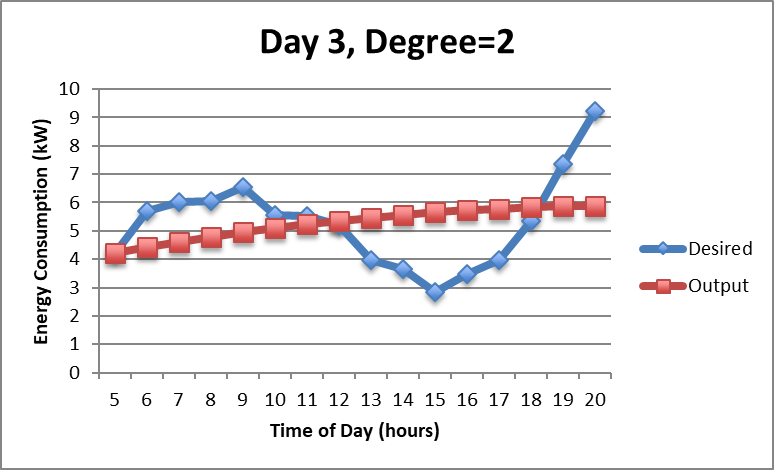
**Total Error for Day 2:** 6.5122696869325445

**Total Error for Day 3:** 6.318587571614721

**Degree 2**

****



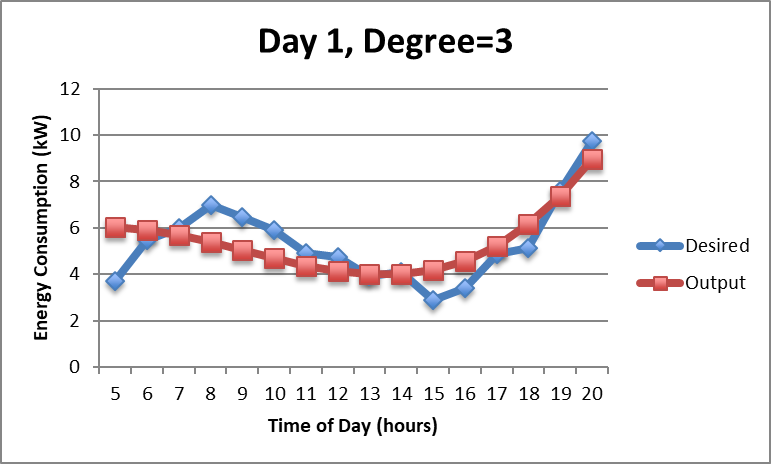


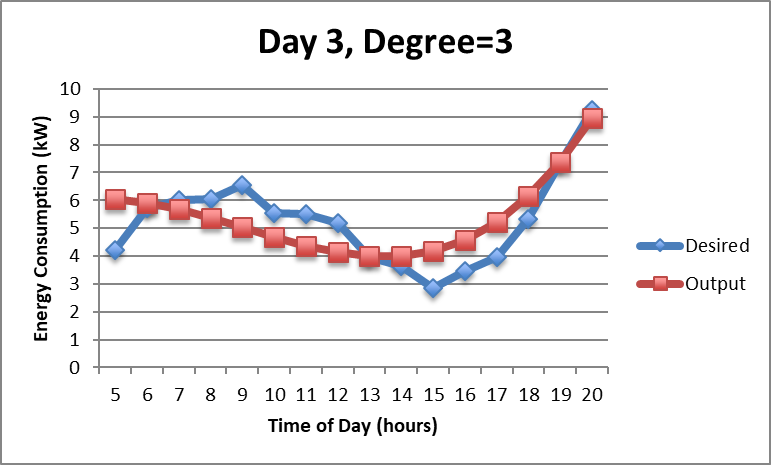
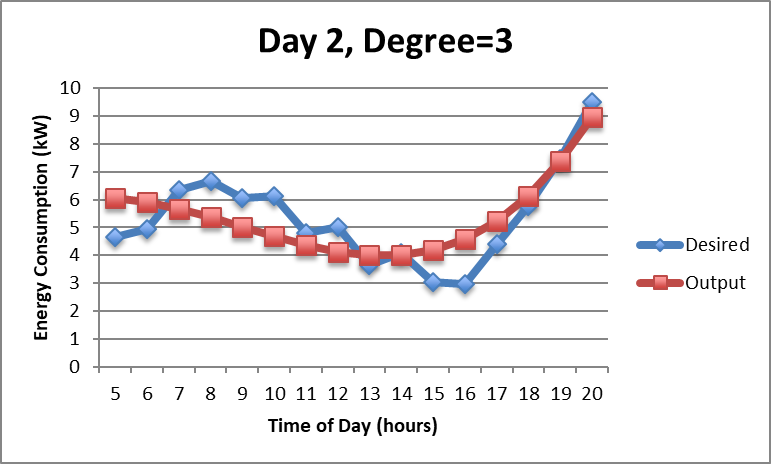
**Total Error for Day 1:** 7.3437976265969676

**Total Error for Day 2:** 7.283992162956873

**Total Error for Day 3:** 7.068096708100044

**Degree 3**

****

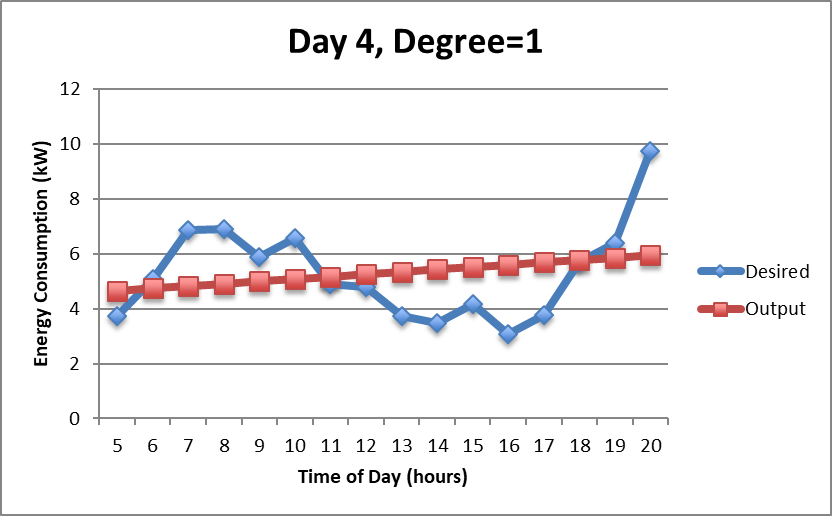
****

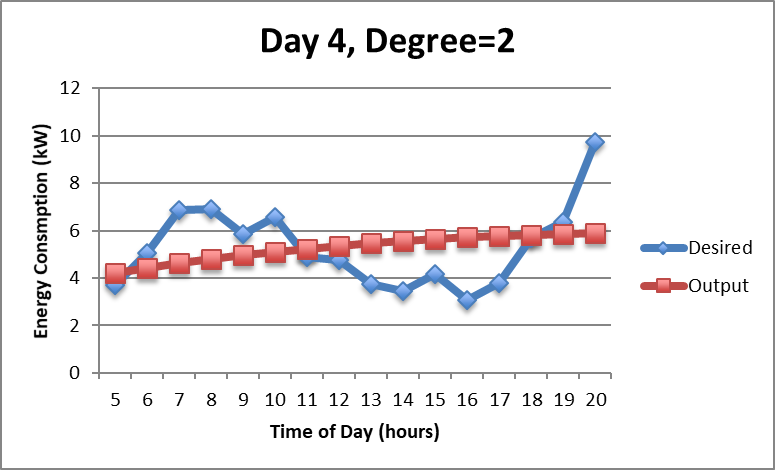
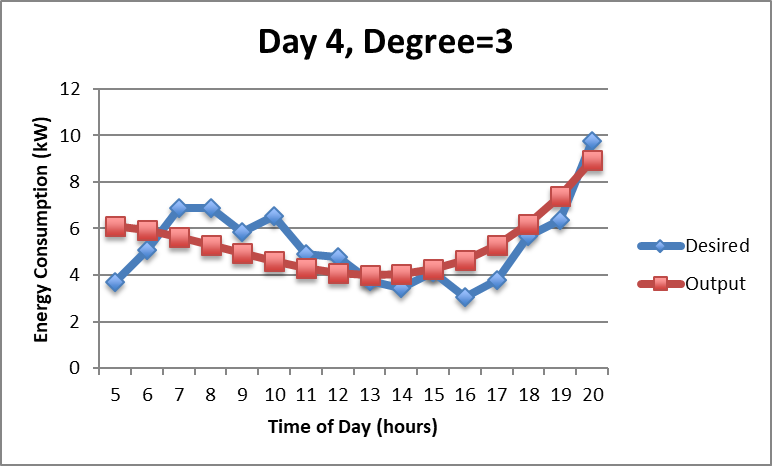
**Total Error for Day 1:** 4.1492729726042255

**Total Error for Day 2:** 3.758825869971608

**Total Error for Day 3:** 3.849389354715179

**b. Predict the energy consumption of the 4th date. Calculate the error of your prediction using the data for the 4th date. Report testing error. Present a graph (original data vs. trained model), similar to the Figure 2.**

**** 

**Total Error for Testing Data Degree 1:** 6.74502

**Total Error for Testing Data Degree 2:** 7.0456291

**Total Error for Testing Data Degree 3:** 4.84638328

**4) Report the number of iterations, the learning rate, data pre-processing steps you have chosen (such as normalization of input data). Clearly explain why you selected these values.**

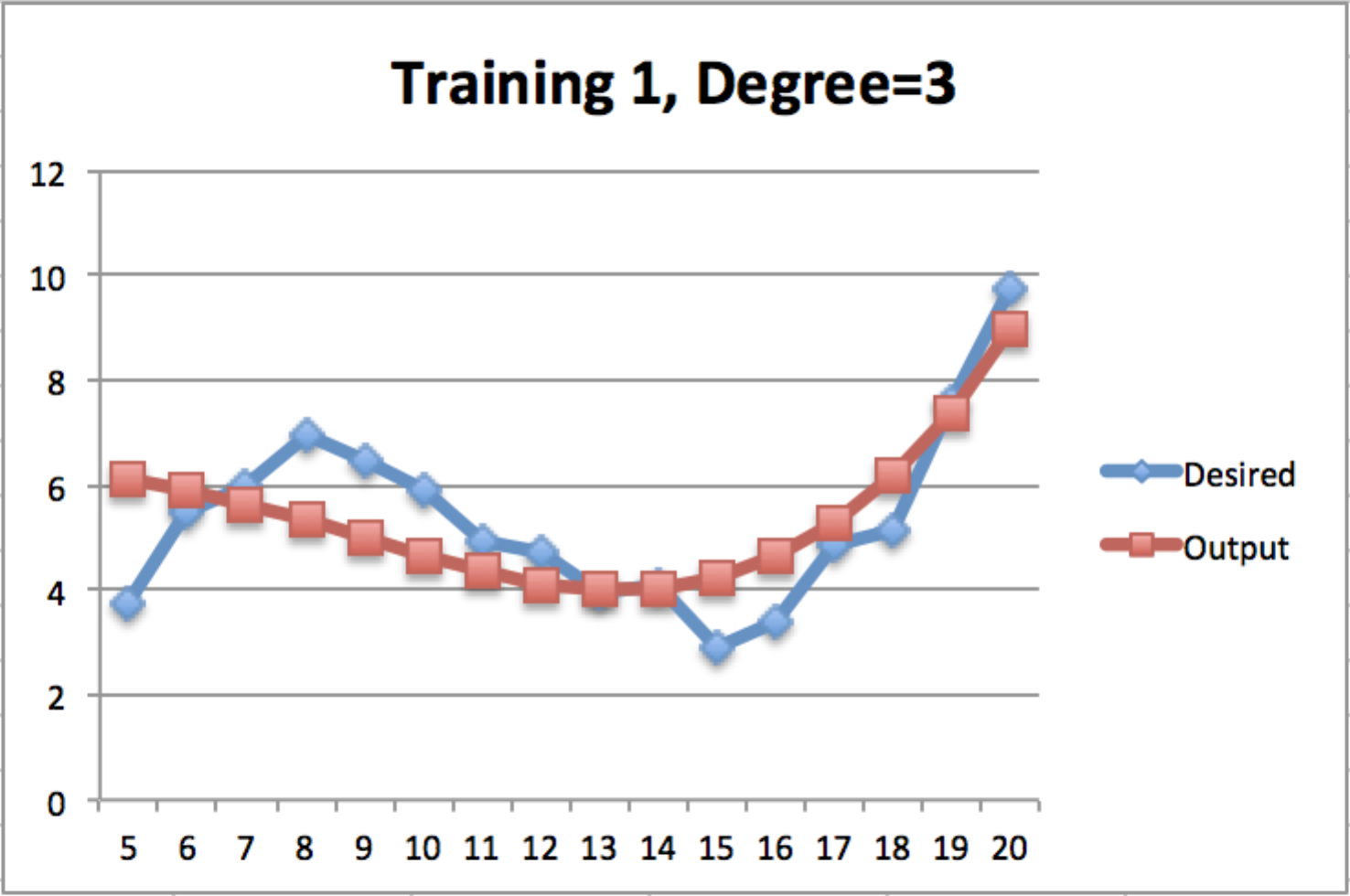
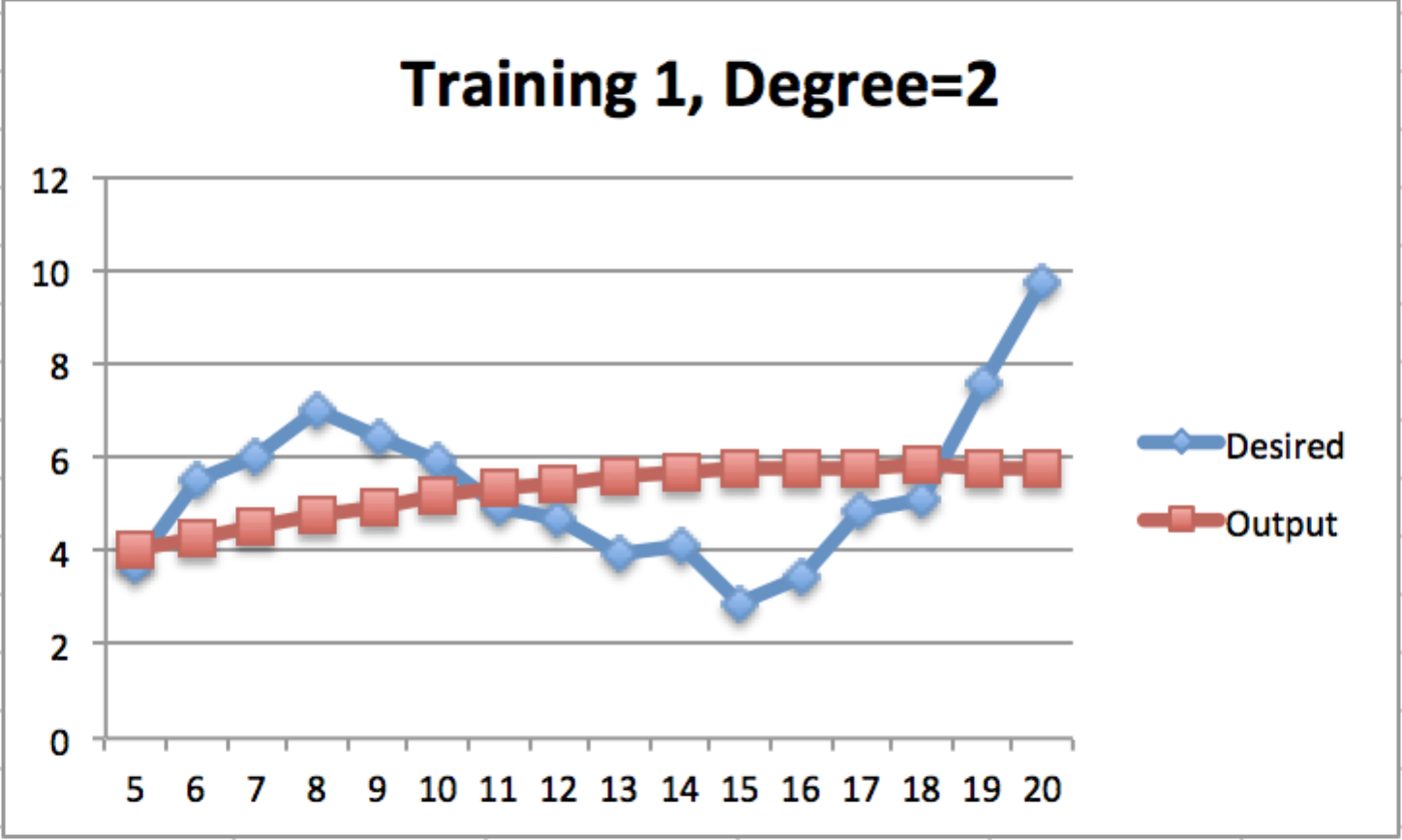
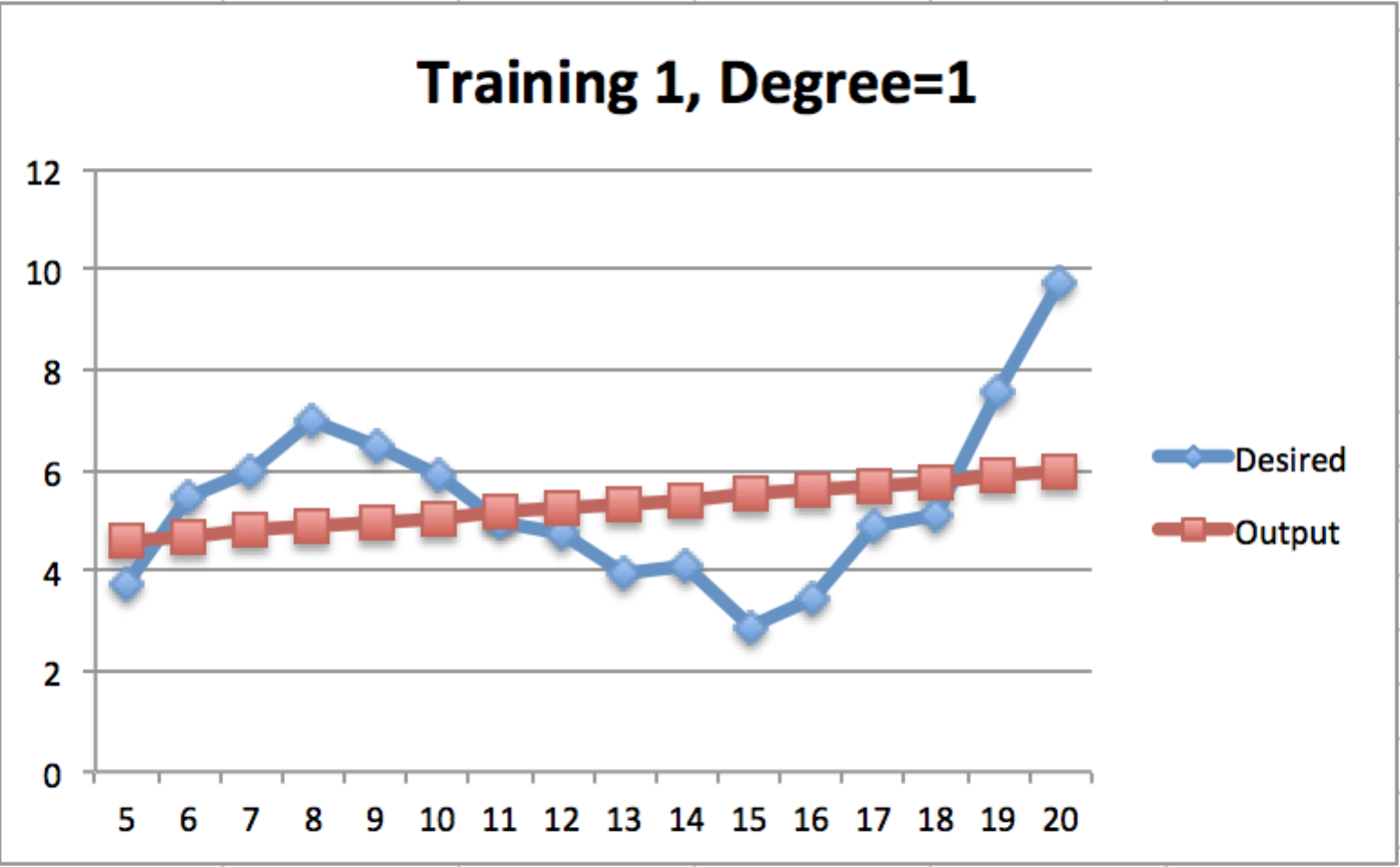
The number of iterations we decided to have is 1,000,000, and our learning rate we decided to be .00000001. We needed to make sure the training constant is low, since the inputs can get very large, for example 203 is 8000. After testing both the training constant and the number of iterations at smaller and larger numbers, we found that too few iterations or too large of a training constant, and the output would be wildly inaccurate and exceedingly large. Using our knowledge from the previous assignments as well as significant guessing and checking, we decided that the numbers we chose resulted in the most accurate output. We tried with only 100 iterations as well as with a learning rate of 0.5, and we saw that the learning rate needed to be much lower, and the iterations needed to be much higher in order to achieve a more precise output.

There are several data preprocessing steps that can be taken including: data cleaning, data integration, data transformation, data reduction, and data discretization. Data cleaning is filling in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies. Given the data sets that were given to us, we did not have to do any data cleaning with the data. Data integration is using multiple databases, data cubes, or files. We are using data integration with pulling training data from multiple files as well as using a separate test data file. We do not have multiple databases or data cubes. Data transformation is normalization and aggregation. We considered normalizing the data, and for a few normalize. Though we tried to normalize the data, we felt that given the effects of normalizing the data to one and the resulting effects that it had on the bias, it was better to not normalize the data. Data reduction is reducing the volume but producing the same or similar analytical results. Data discretization is part of data reduction, but it is replacing numerical attributes with nominal ones.

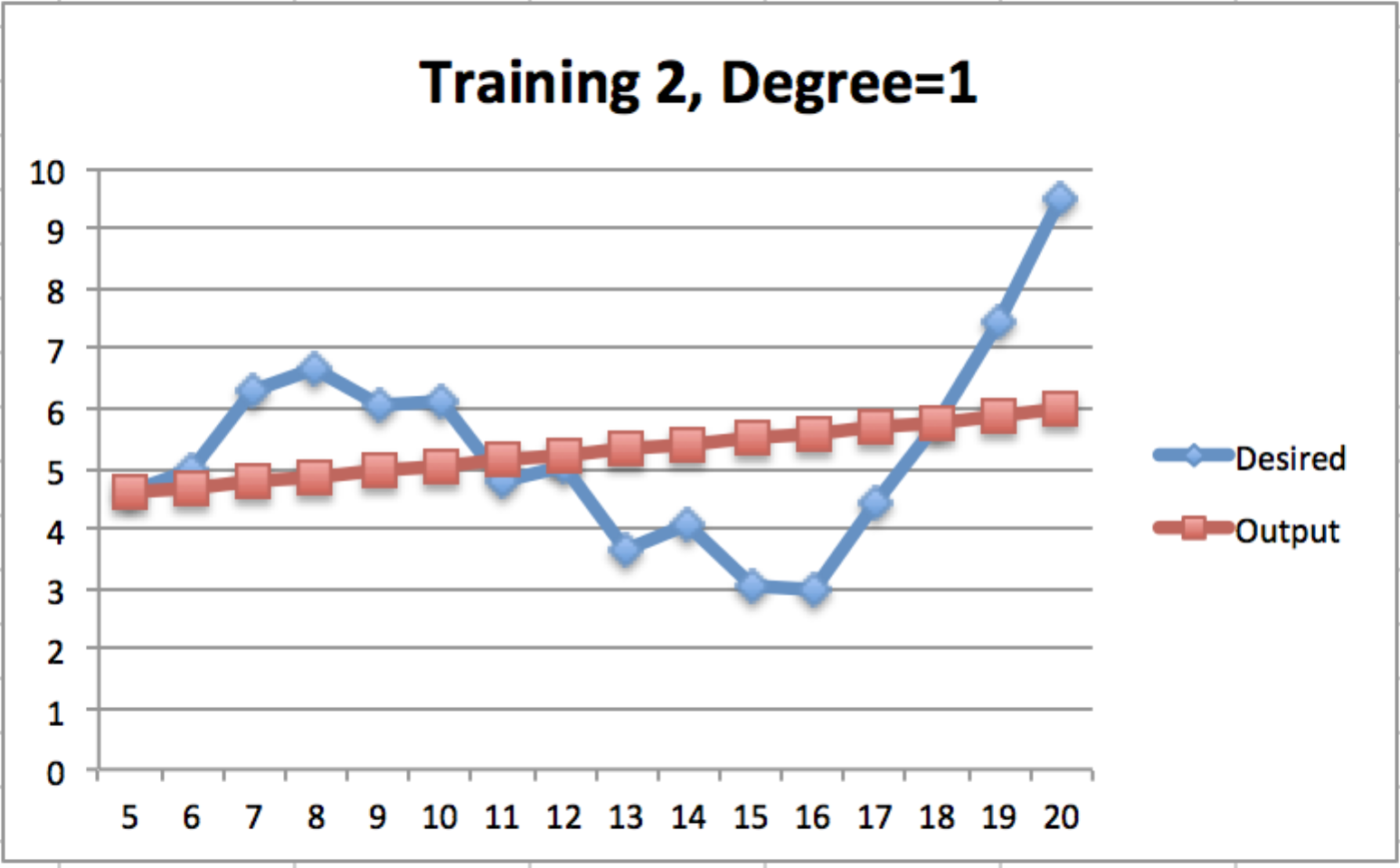
Could the error be further reduced using a network of neurons opposed to a single decision unit? If so, discuss how and why these methods would reduce the error. Yes, as the system gets more complicated, a network of neurons would be able to predict the trend more precisely. However, there would be more training required in order to reduce the error, and this would be more expensive as you would have more neurons. This is due to an increase in the number of neurons and the need for each one to make a decision, which would mean that each one would need to be trained on the data set, and you would need more data in order to efficiently train the network of neurons. However, there is a potential with too many neurons that it will not be more efficient and actually cause more problems.

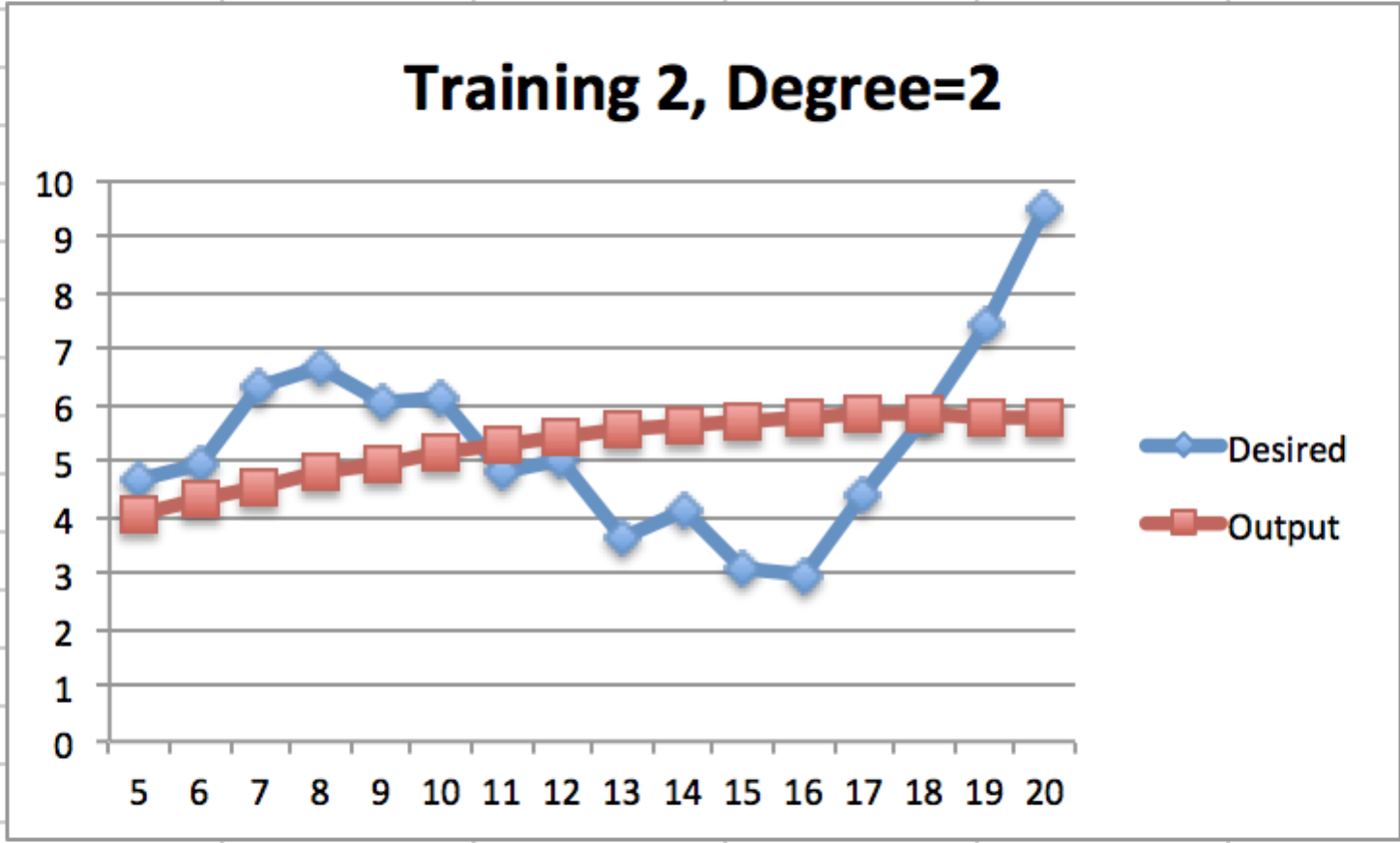
**Graphs**

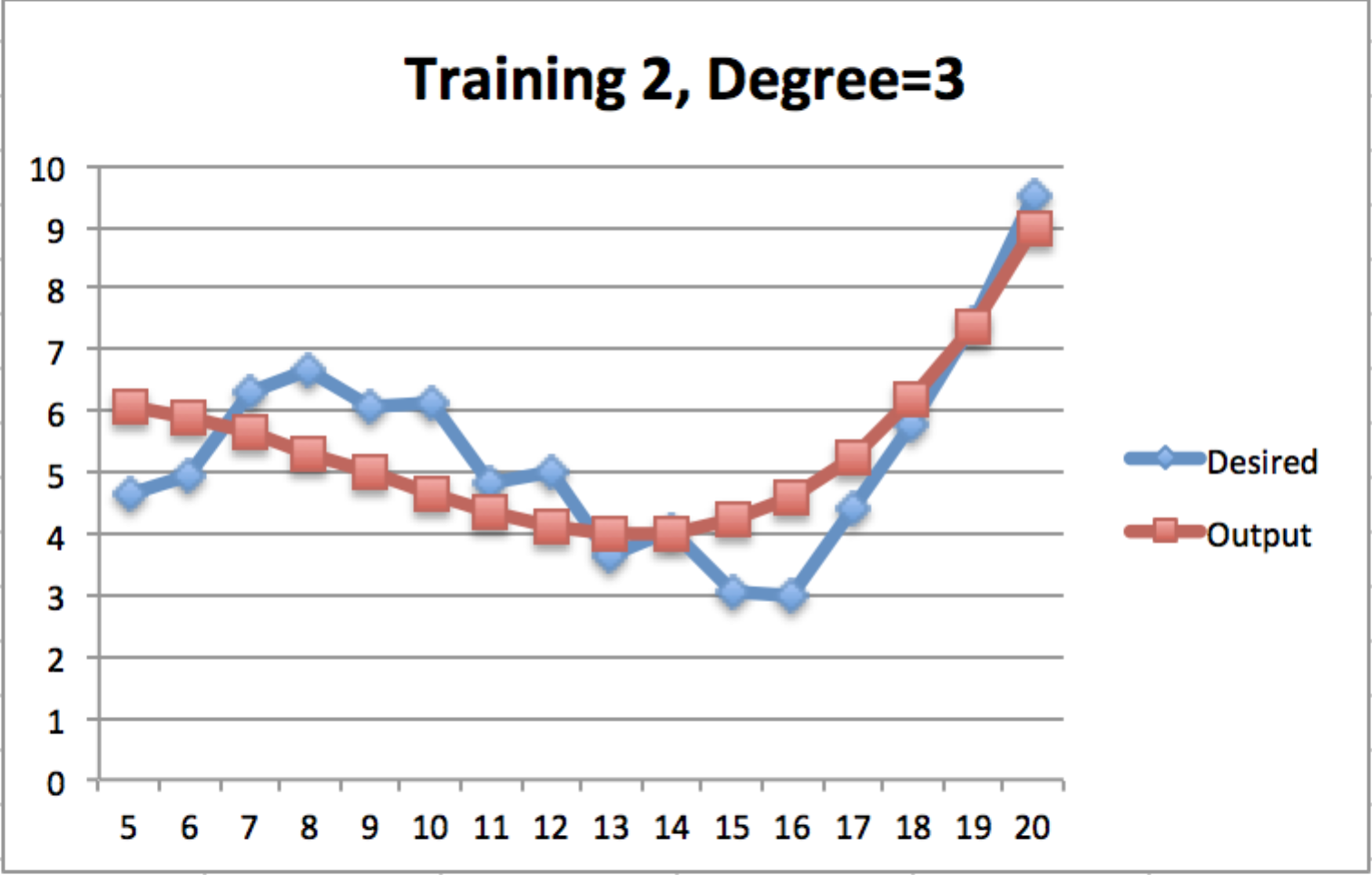
Day 1 (Training 1)



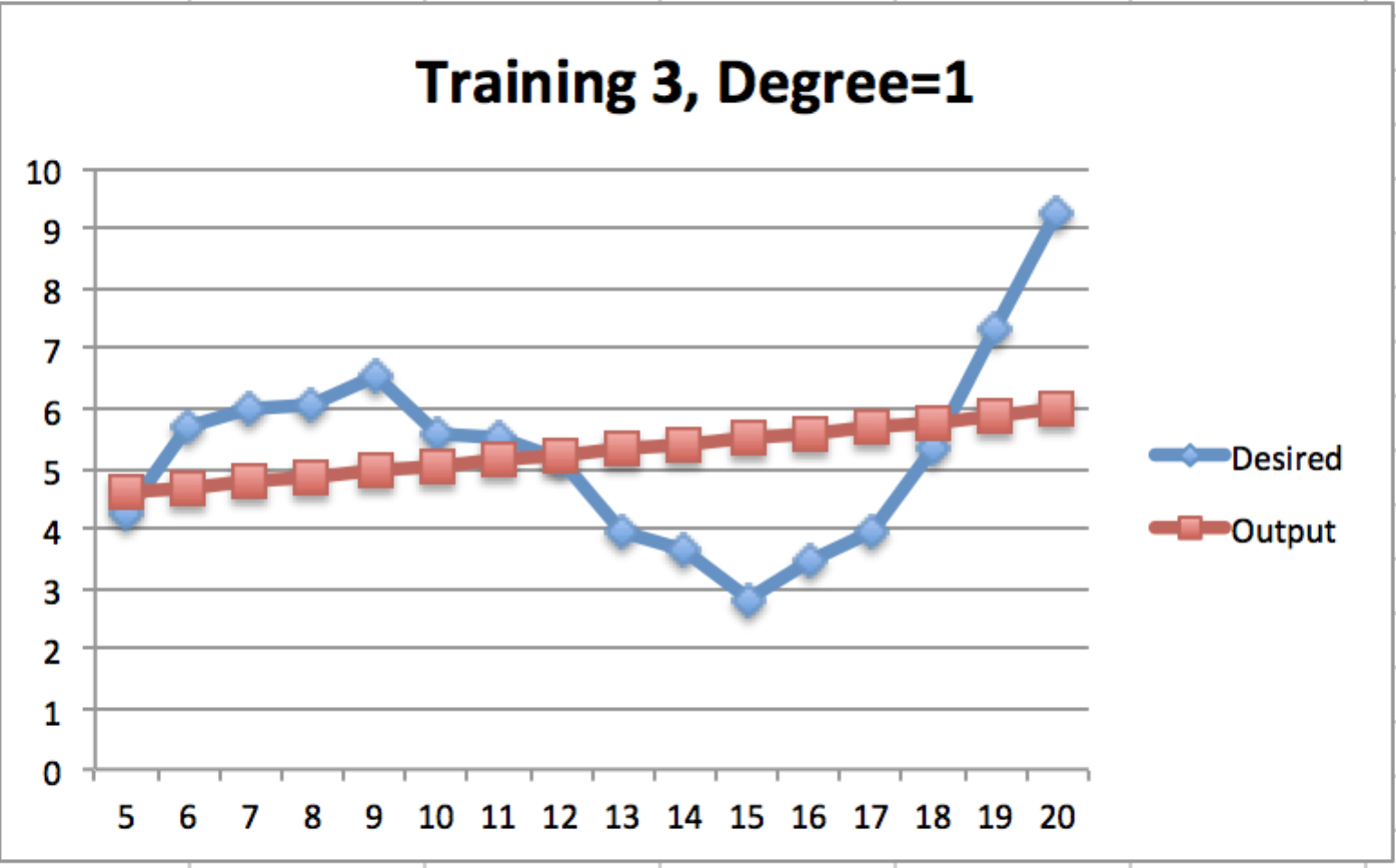
Day 2 (Training 2)

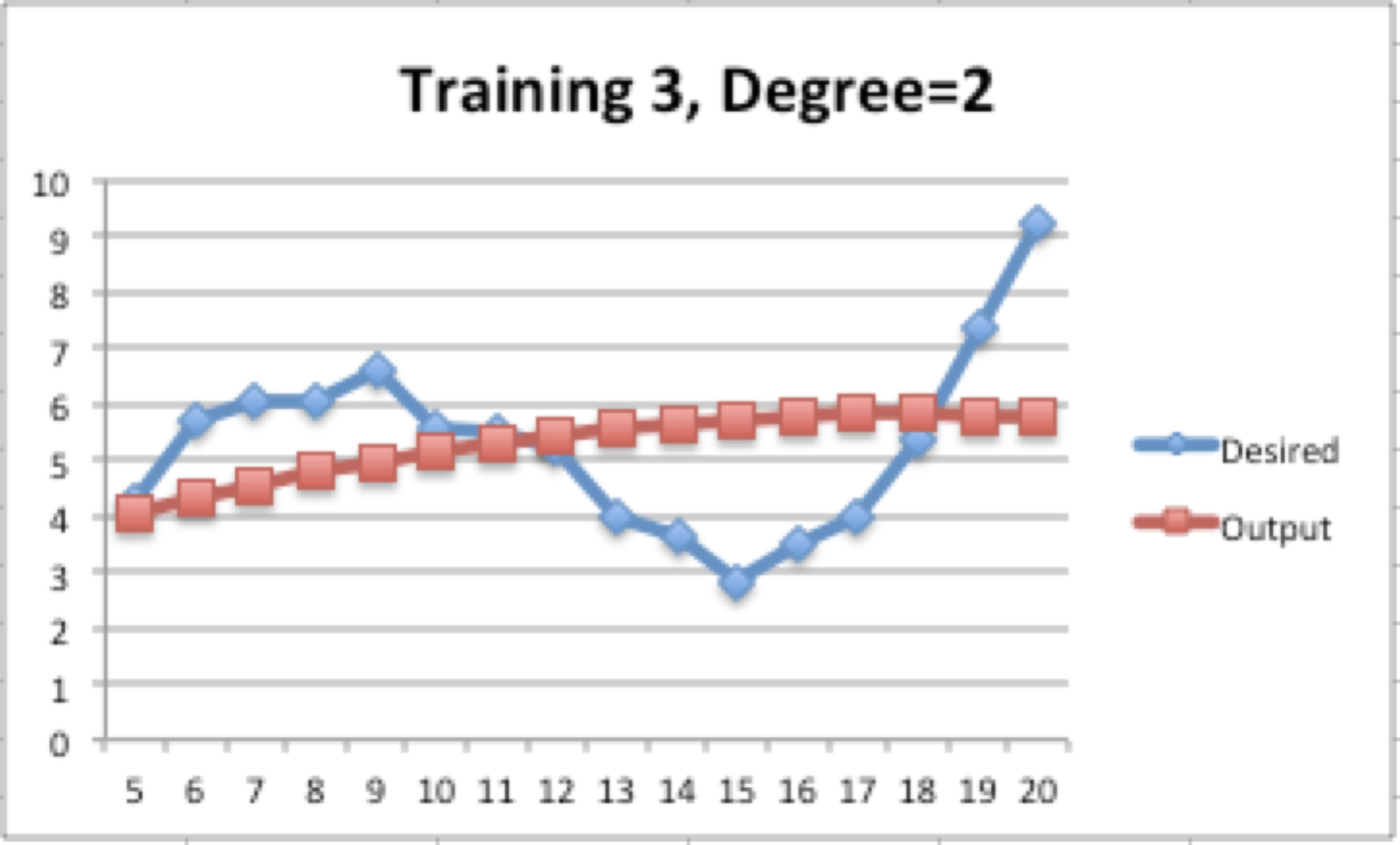


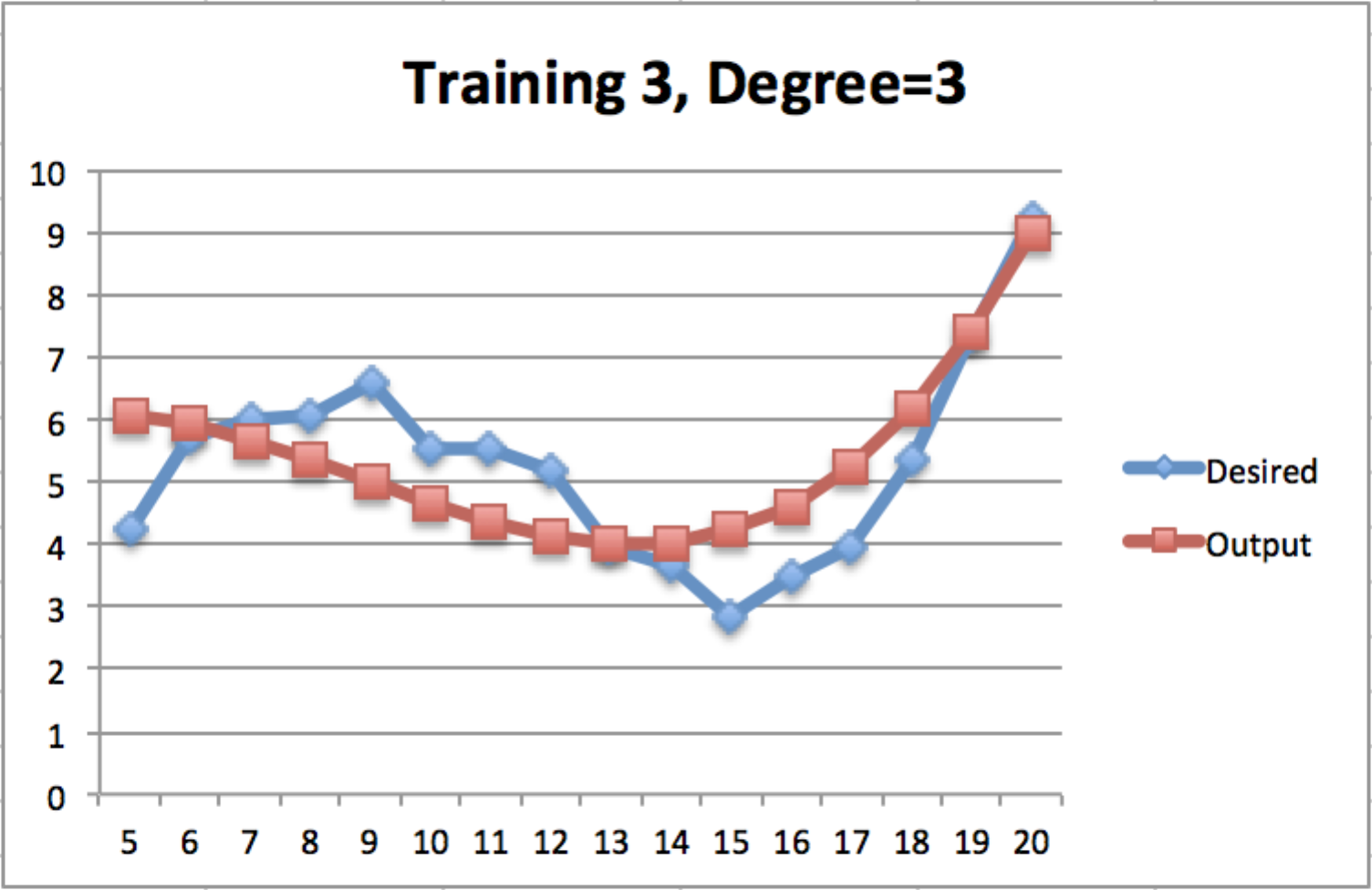




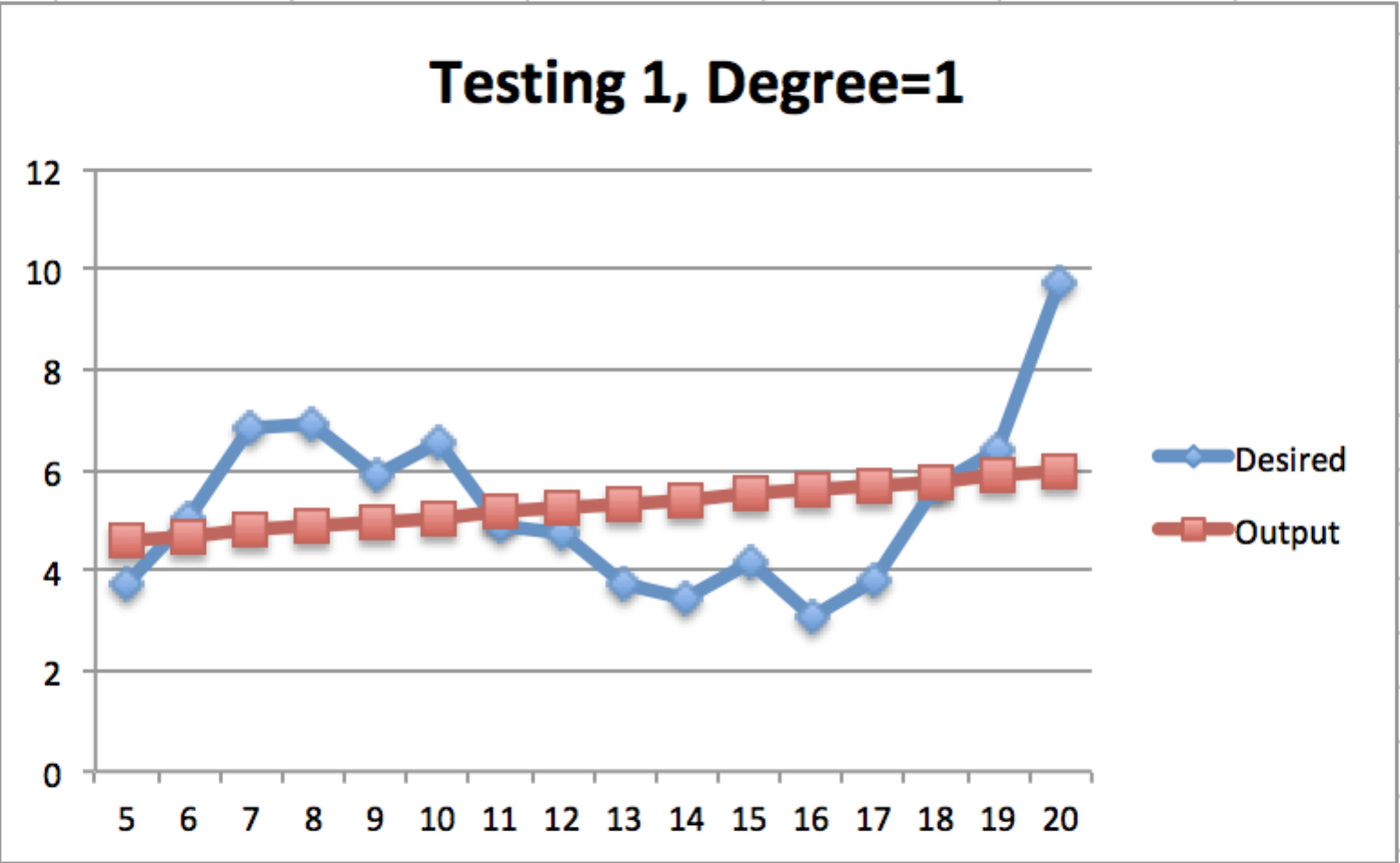
Day 3 (Training 3)

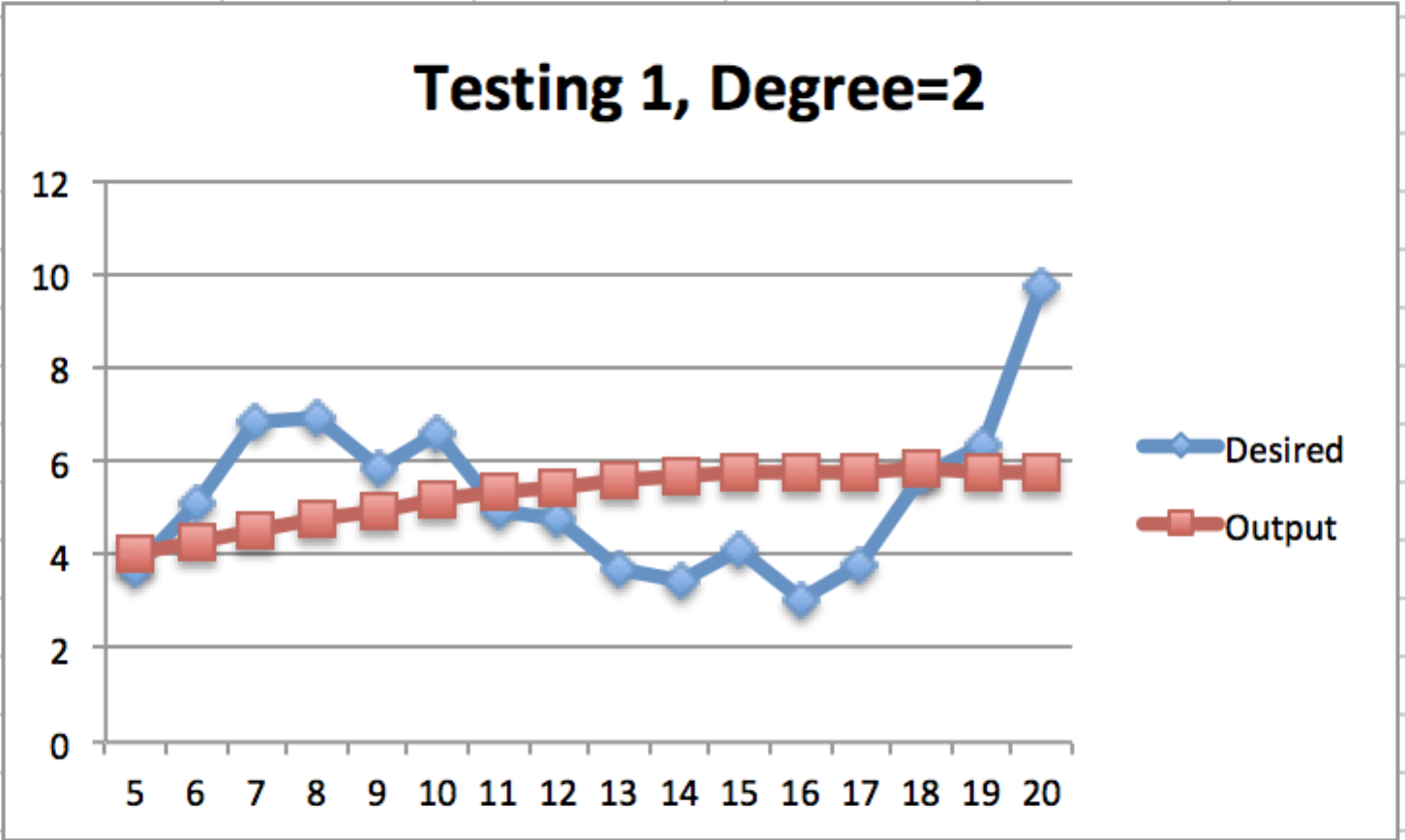


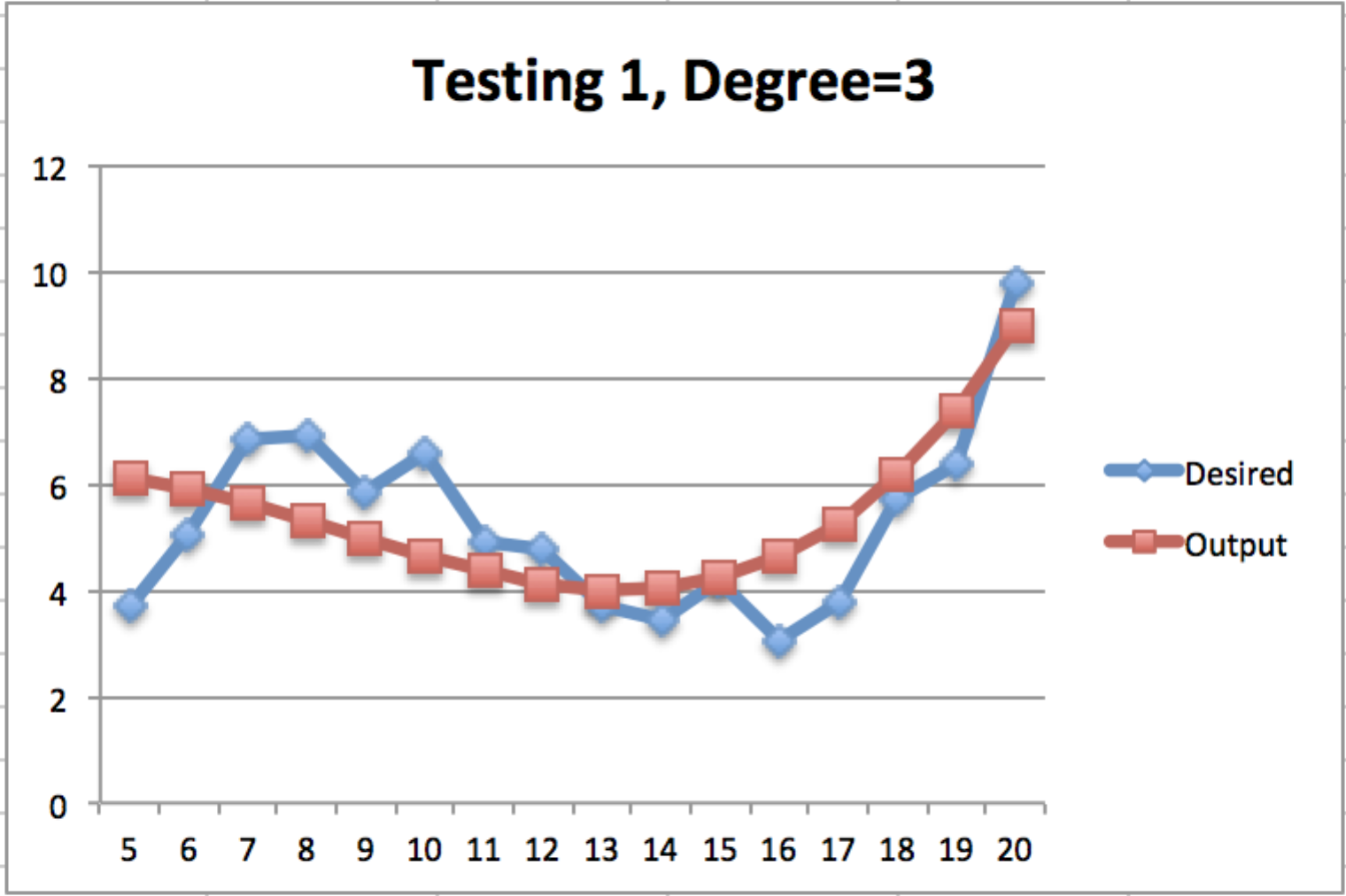




Day 4 (Testing 1)







**Total Error for Training Data Degree 1:** 11.2296014

**Total Error for Testing Data Degree 1:** 6.74502

**Total Error for Day 1:** 6.649214787984489

**Total Error for Day 2:** 6.5122696869325445

**Total Error for Day 3:** 6.318587571614721

**Total Error for Training Data Degree 2:** 12.15826526

**Total Error for Testing Data Degree 2:** 7.0456291

**Total Error for Day 1:** 7.3437976265969676

**Total Error for Day 2:** 7.283992162956873

**Total Error for Day 3:** 7.068096708100044

**Total Error for Training Data Degree 3:** 6.99409024

**Total Error for Testing Data Degree 3:** 4.84638328

**Total Error for Day 1:** 4.1492729726042255

**Total Error for Day 2:** 3.758825869971608

**Total Error for Day 3:** 3.849389354715179

**Total Error for Testing Data Degree 1:** 6.74502

**Total Error for Testing Data Degree 2:** 7.0456291

**Total Error for Testing Data Degree 3:** 4.84638328