CS 229 ASSIGNMENT 6:

Neural Network

NAME: Xiaopeng Xu KAUST ID: 129052

Answer to Question 1 and question 2 are in attachment 1.

Question 3 **Implementation of NN (using Back-Propagation)**

1) (5pts) **Describe all the parameters** you chose, including the number of inputs, outputs, and hidden neurons, the sizes of the initial random weights, learning rate etc.

D = 2

M = 5

K = 1

All initial weights are set to be 1,

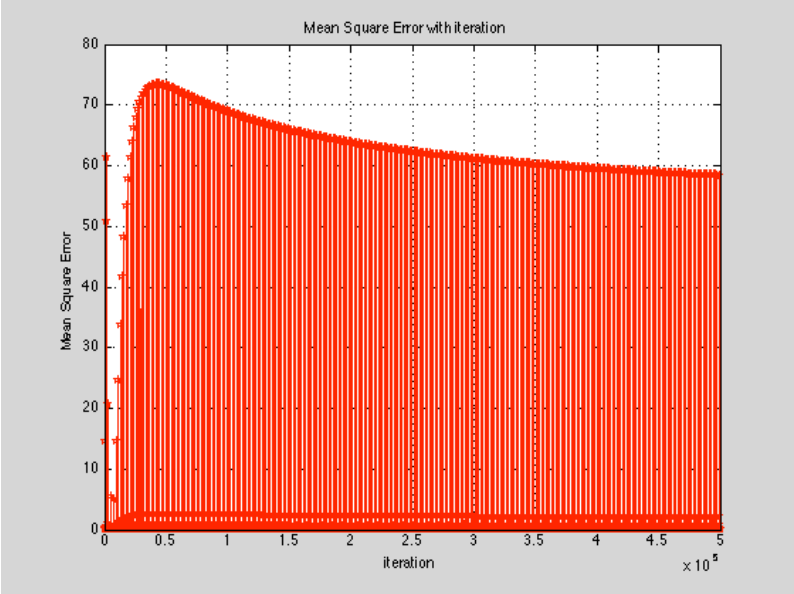
Using stochastic gradient descent method: learning rate = 0.2, max_iter = 500000

Using batch gradient descent method: learning rate = 0.2, max_iter = 5000

2) (20 pts) Find a learning rate that allows it to learn to a small mean squared error. **Plot a figure of how the error decreases during learning.**

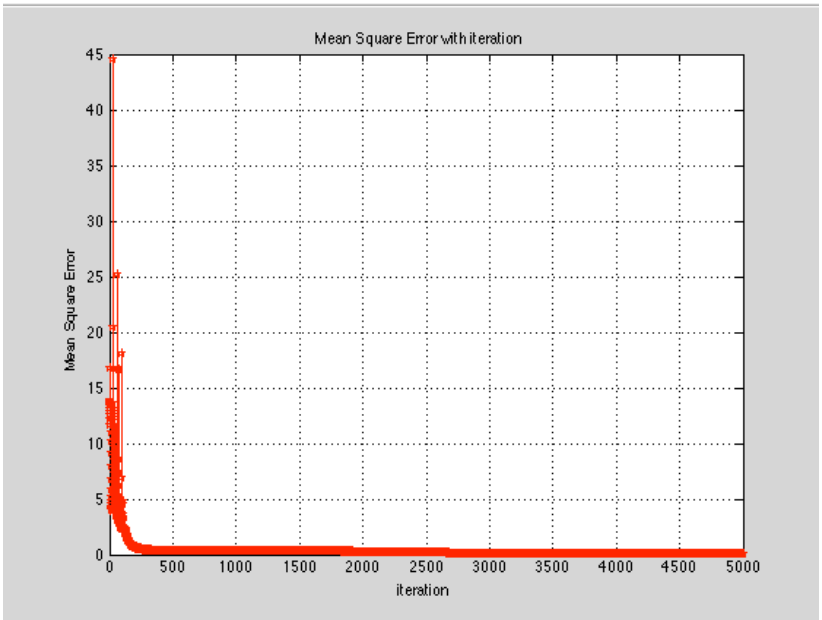
Using stochastic gradient descent method:

Learning rate = 0.2



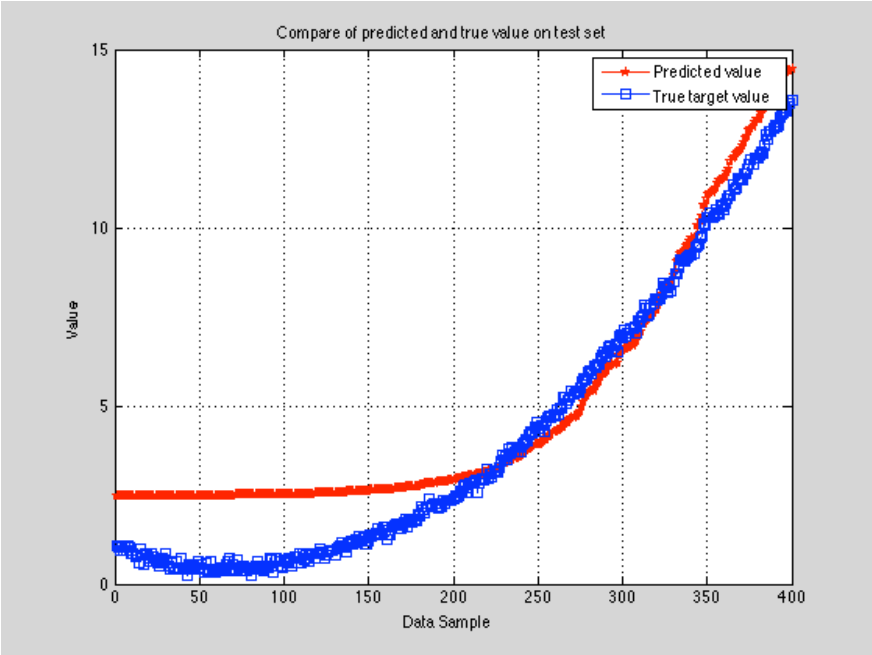
Using batch gradient descent method:

Learning rate = 0.2



3) (20 pts) **Test** the NN you learned by a **different** set of data **points (x,y)** (different from training set, but y is still generated by f(x)+noise), what's the error when comparing the **predicted y_n** with the **true target y**? Give the mean and standard deviation of errors.

Using stochastic gradient descent method:



Mean error on test set:

test_err_mean =

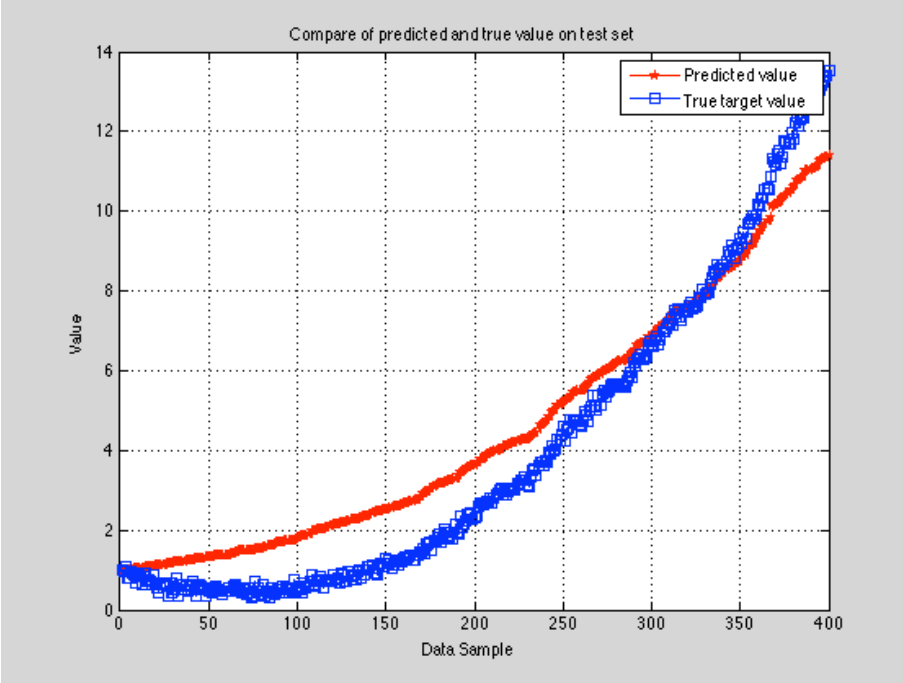
0.8471

Standard deviation of prediction of test set:

test_err_std =

0.9112

Using batch gradient descent method:



Mean error on test set:

test_err_mean =

0.5807

Standard deviation of prediction of test set:

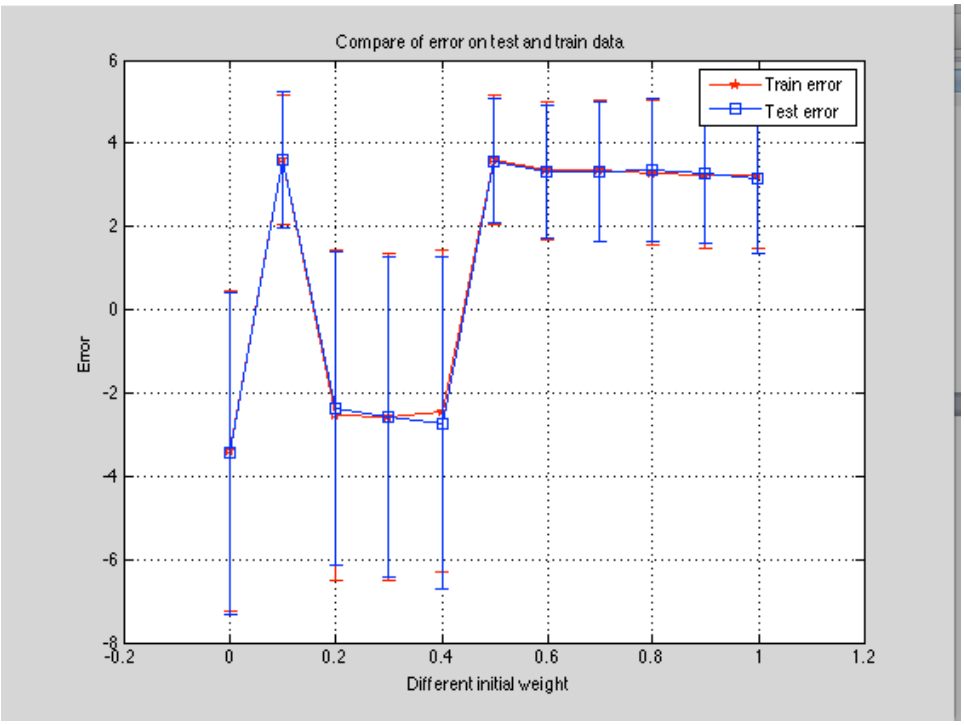
test_err_std =

0.8522

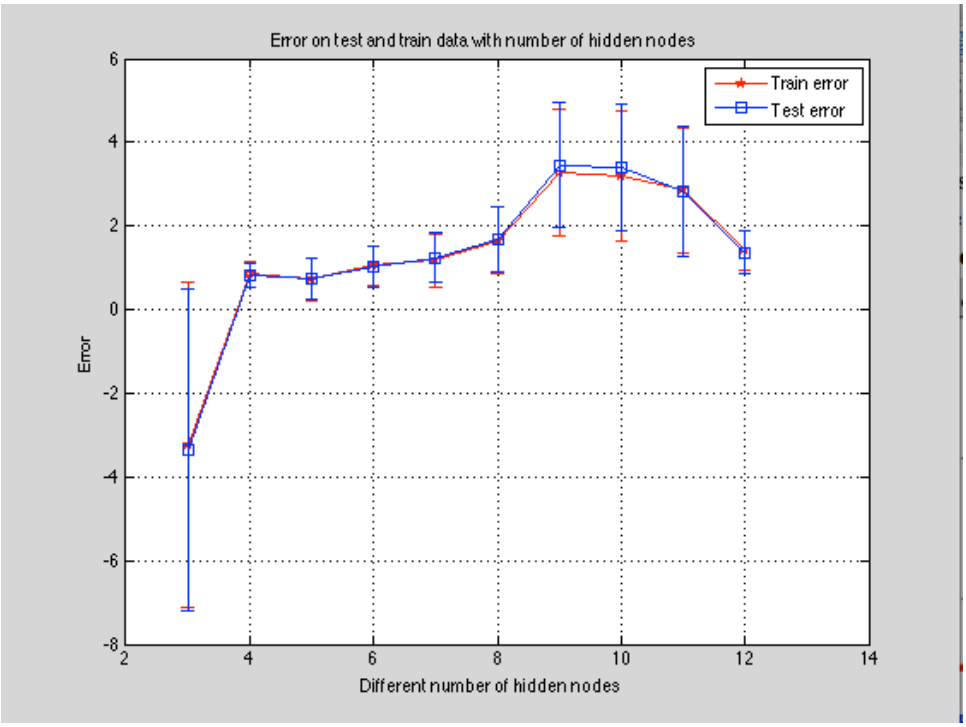
4) (15 pts) How will the training error and testing error be different if you re-train the NN by different initializations of weights? And how will they change if you set ***M*** (the number of hidden units) to be different values? **Plot the error bars** (as examples of http://www.mathworks.com/help/techdoc/ref/errorbar.html) **for both training and testing.**

Here I used stochastic gradient descent for this question.

Error on test and train data with different initial weights:



Error on test and train data with different number of hidden nodes:



An Additional Exercise (**a bonus** for those who still have time, energy and interest to work)

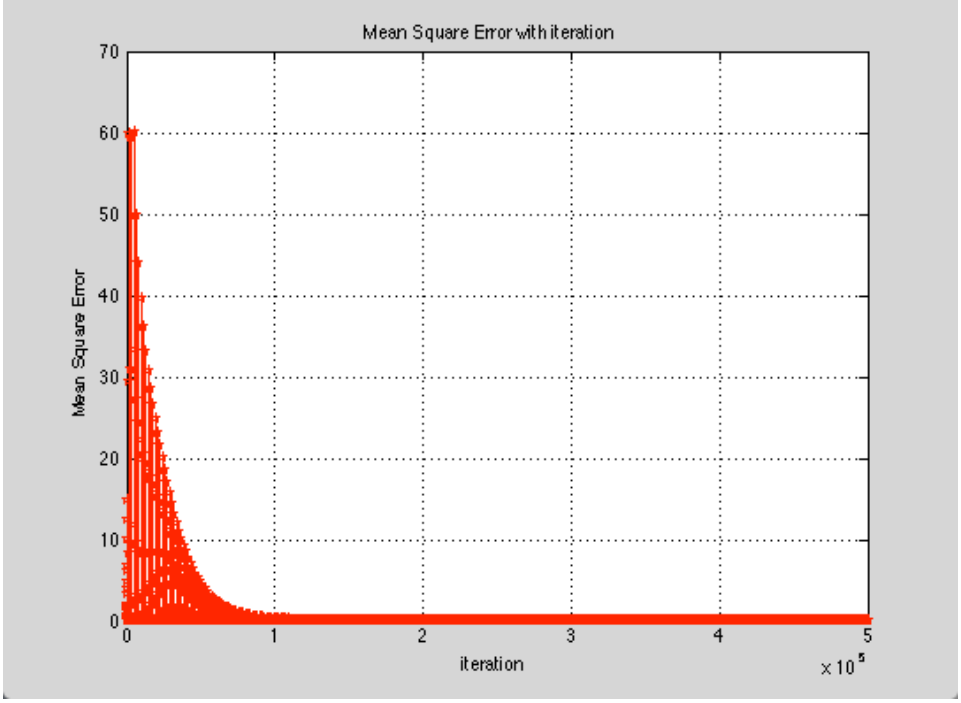
(20 points) Implement the **momentum** discussed in class.

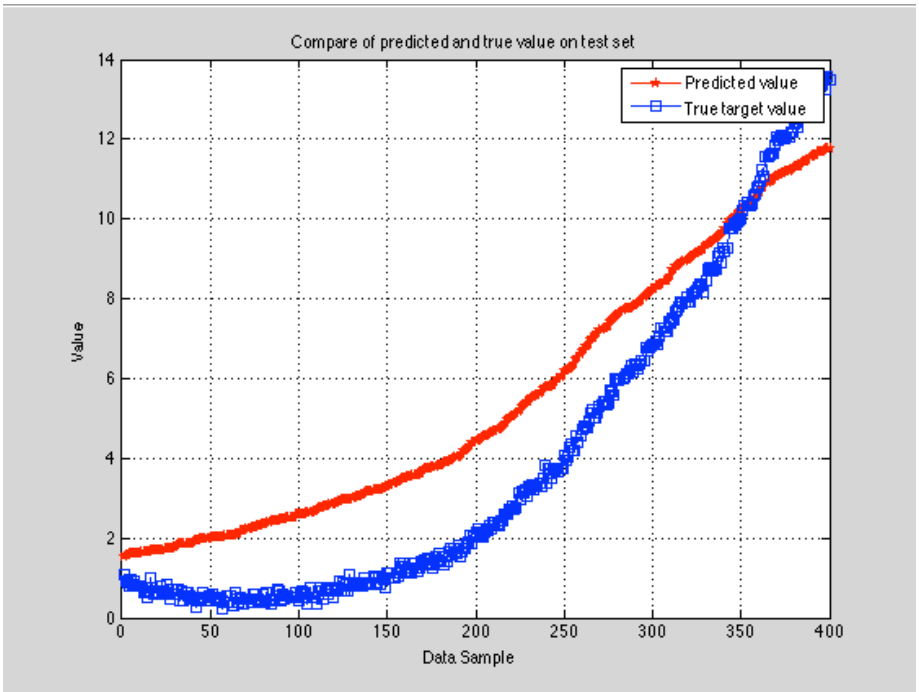
Run it and see if it learns faster, or learns a better solution (in terms of testing error). Describe whether it improved speed, accuracy, or both, and why you think that occurred. (It's OK if you discover your "improvement" had no effect or even made it worse. The important thing is to test it and explain the results).

Here I also used stochastic gradient descent for this question. Below is my initial parameters.

```
%% Neuro Network Parameters
D = 2; % No. of input nodes
K = 1; % No. of output nodes
M = 5; % M is No. of hidden layer nodes
Kfold = 5; % Folder number of K-fold cross-validation
max_iter = 500000; % Maximum iteration for training
eta = 0.2; % Learning rate
mom = 0.9;  % Momemtum for learning
W1=ones(D+1,M);
W2=ones(M+1,K);
```

This is my results:





I find that at the beginning, momentum learns very fast and go to optimal quickly that traditional stochastic gradient descent. However, when it goes near the optimal it learns very slowly. Given the same number of iterations, momentum gives me a lower accuracy.

Mean error on test set:
test_err_mean =
1.4402

Standard deviation of prediction of test set:
test_err_std =
1.0507