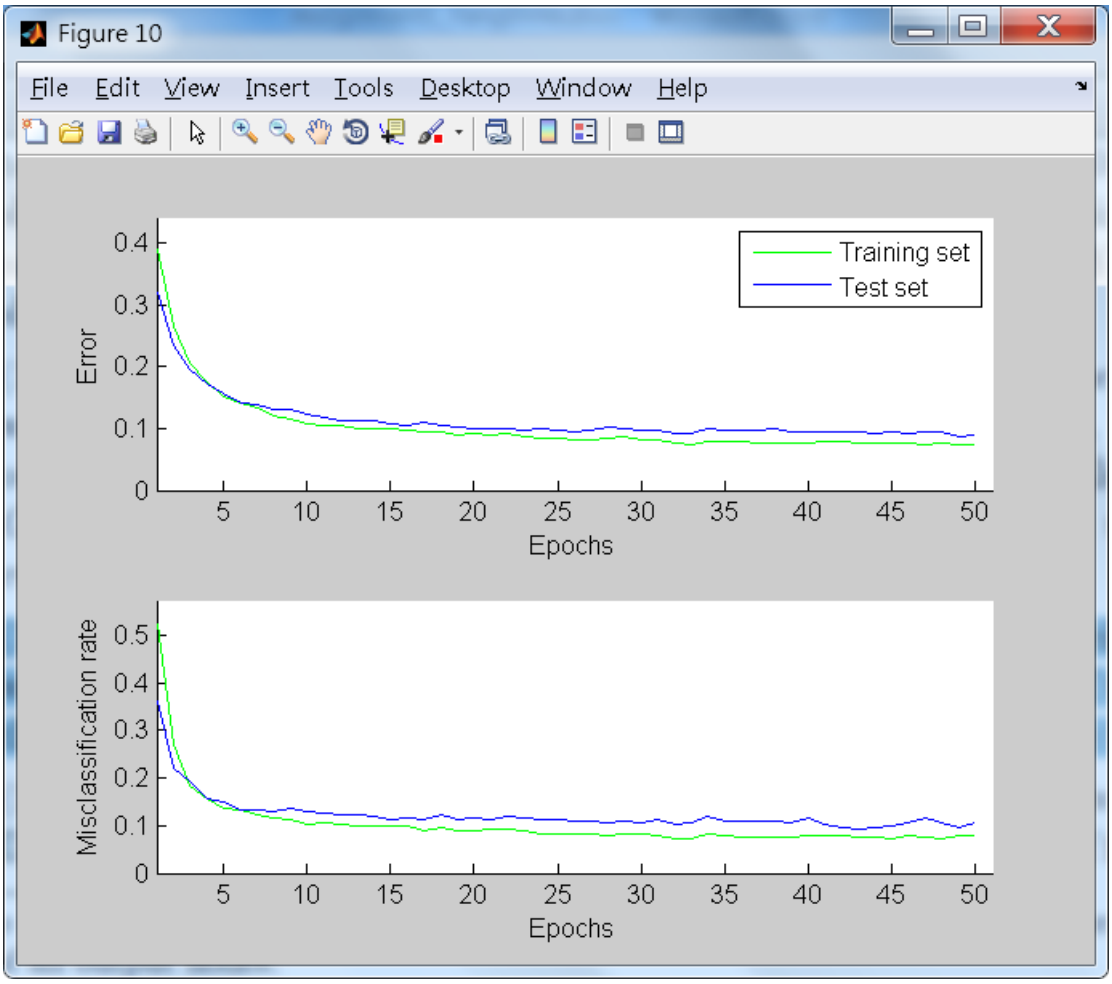
Intelligent Systems (IS Fall 2013)

Assignment 1 Bonus: MNIST

Student: Fang-Lin He

---

1. (45 bonus points total) Your mission, should you decide to accept it will be to train a neural network on the popular MNIST dataset. This dataset consists of 70.000 (60k train + 10k test set) images of handwritten digits, but we are going to use only 10% of it. Each 2828 pixel image is grayscale, and belongs to one of ten classes (the digits). Your network should have 28 x 28 = 784 input units, 100 hidden units and 10 output units.
   (a) (5 bonus points) Download the dataset from http://mldata.org/repository/data/viewslug/mnist-original/ and load it into your program. Then split the data into training and test set. The first 60k samples are training data, and the last 10k samples are test data. Then randomly pick a subset of 6000 images from the training set and one of 1000 images from the test set and shuffle each of them. Those two subsets will be our training and test data.
   >> Please load the dataset first (mnist-original.mat.m). The size of this file is too big, so I didn't attach this file. Then, run my script "bonus_1a.m".
   (b) (5 bonus points) Binarize the targets, so for example 6 would become the vector (0; 0; 0; 0; 0; 0; 1; 0; 0; 0) and 2 would become (0; 0; 1; 0; 0; 0; 0; 0; 0; 0). You should now have your training data in a matrix (2D array) of size 6000 x 784 and the corresponding targets in a matrix of size 6000 x 10. Your test data should be 1000 x 784 and 1000 x 10, respectively.
   >> Please run my script "bonus_1b.m".
   (c) (15 points) Implement forward- and backward-passes for a neural network with 784 input units, 100 hidden units and 10 output units. Hidden and output units should all use the sigmoid activation function. Also implement Online Gradient Descent for this network (that means weight updates after every image).
   (d) (10 points) Run your network for 50 epochs (that is 300k weight updates) with a learning rate of $\eta = 1/300$. For every epoch, keep track of the error and the number of misclassifications for both the training and the test set.
   >> (c) and (d) please run the script "bonus_1c.m". I didn't have time to rewrite the "multiple_neural.m" to a function, so… it's still a script. The figure of errors and the misclassification rate are shown as below:



(e) In fact, from these plots, I don't think this model is underfitting or overfitting.