Homework #1

1. **Simplified No-Free-Lunch Theorem**
   1. We only need to consider about the test inputs. The test inputs is . Now consider about the behavior of function, we can draw the figure below,

f(x) = 1

g(x) = 0

g(x) = 1

i = N+M

i = N+(M/3)

i = N+1

i = 1

Thus, we can get the OTS by formula (a). Because our data set is discrete, we use a ceiling function to compute the correct answer, where ceiling means the minimum integer that bigger than the input. We get,

* 1. Within a noiseless assumption in D set, we take the `training set’ as an one-to-one mapping function. The problem can be solved by estimating the numbers of mapping from `testing set’ to Y set. It is equal to
  2. Assume we fixed g as . We can draw the figure below,

f(x)

111…..………………………………………………………………….……….

g(x) = 1

111………………………111…………….……………………………………………………...111

i = N+M

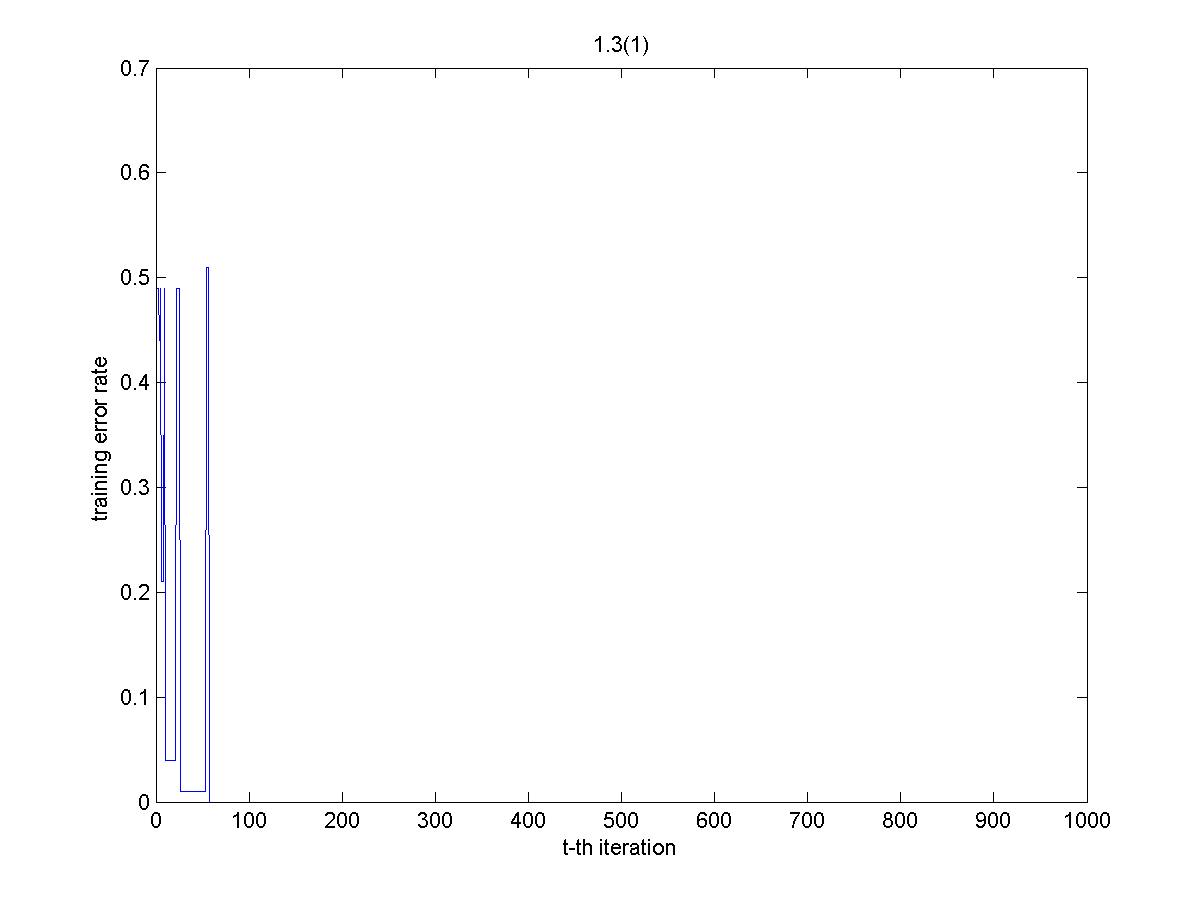
i = N+1

i = 1

My approach is choosing k bits from the area which i belong N+1 to N+M (denote ) in f(x) whether k bits should be 0, the area belong i = 1 to N is fixed. This combination would match those f satisfies . The combination is

* 1. By the answer of 1-1(3). Because all those f in (2) are equally likely in probability, we can express the expected value of OTS(f, g) for a fixed g just by weighted all possible situation equal to 1,
  2. For any two algorithm and , both of them take the same inputs D. We can claim that any of g inducted by algorithm A is a fixed g to f. Thus, we use the conclusion of 1-1(4) and get this,

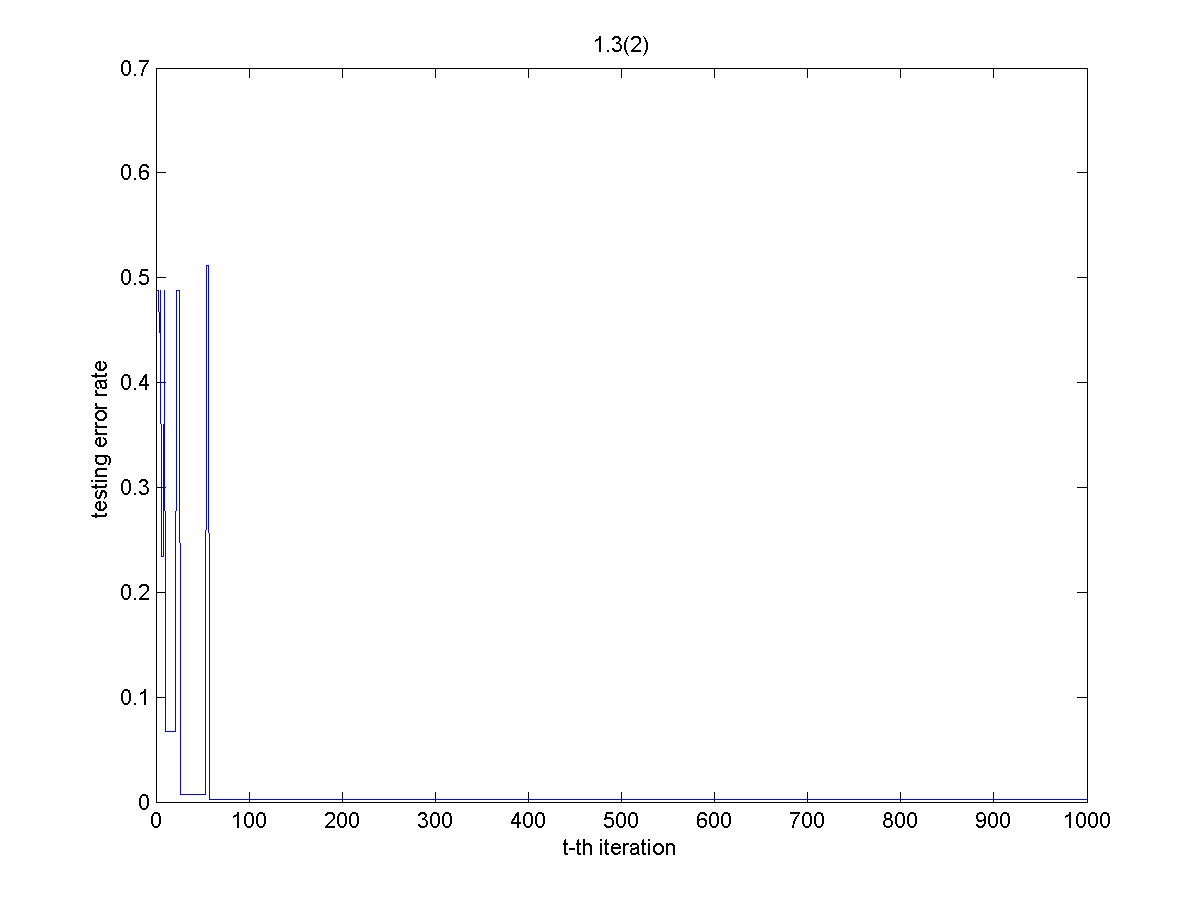
1. **Bins and Marbles**
   1. Draw one sample that has 10 marbles drawn from the bin without any red,
      1. Consider :
      2. Consider :
      3. Consider :
      4. Consider :
      5. Consider :
      6. Consider :
      7. Consider :
      8. Consider :
      9. Consider :
2. **Perceptron Learning (\*)**
   1. The figure 1.3(1) plot the t – training error below:



The training error rate lied at 0.49 at the beginning and lied around 0.01 to 0.51. It moves to 0 at t = 57, and won’t come up again. Because once it comes to 0, no matter which training data you chose, it won’t miss when you test the sign equal or not.

Brief conclusion:

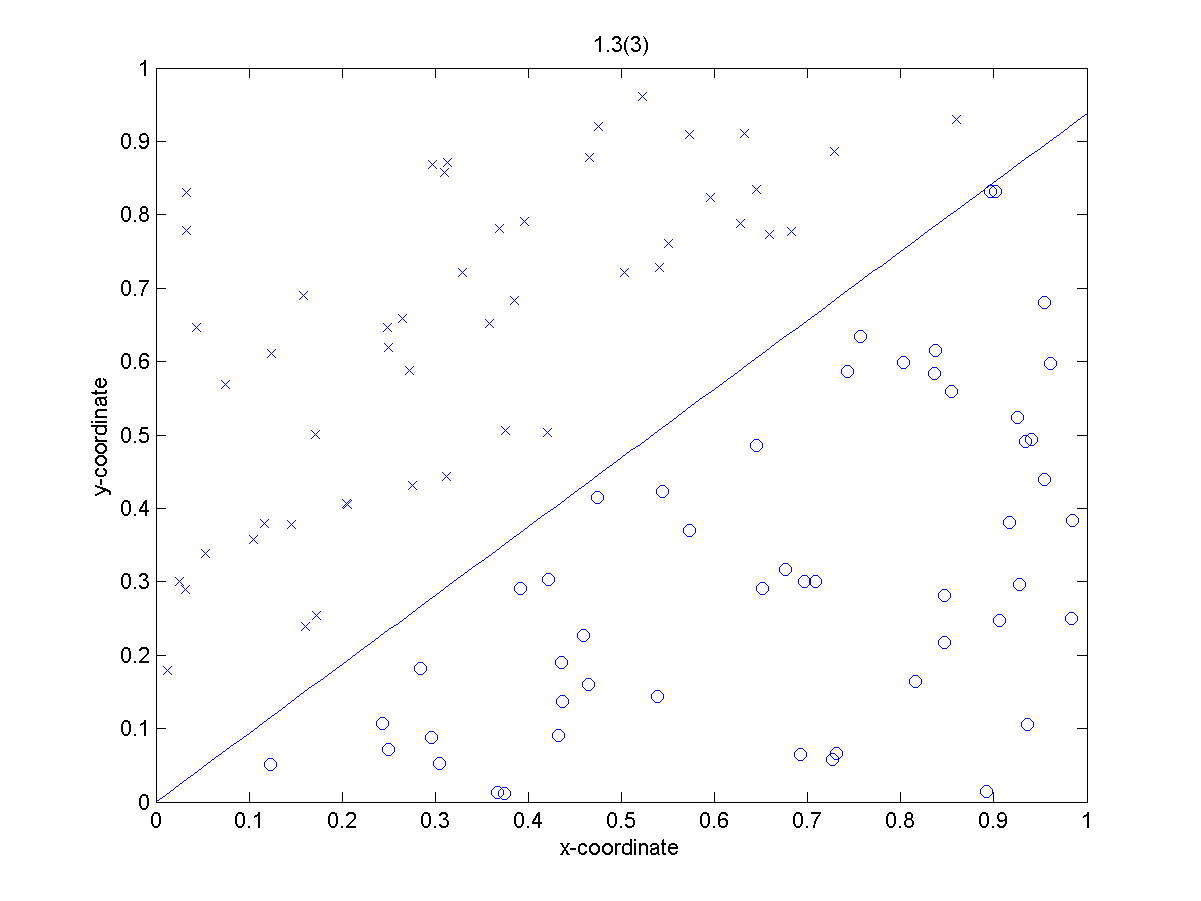
* The first decision function (t = 1) not always get the highest training error.
* The training error may fall by 0.5 only in several steps.
* The training error comes to a very lower rate, say 0.01 in our case. It still can come up.
* Once the training error becomes 0, it never comes up again.
  1. The figure 1.3(2) plot the t – testing error below:



At t = 57, the testing error rate went to 0.02. Compare with the figure 1.3(1), the training error become 0 imply the decision function g has no chance to improve anymore. Thus, the testing error can only stand at 0.02 since the training error comes to 0.

Brief conclusion:

* The first decision function (t = 1) not always get the highest testing error.
* The testing error may fall by 0.5 only in several steps.
* The testing error comes to a very lower rate, say 0.05 in our case. It still can come up.
* The testing error converge to a rate, it may not 0.
* The testing error converge when the training error become 0, because the decision function won’t be changed by training set.
* Even we have 1000 iteration here, the decision function stop upload at t = 57.
  1. The figure 1.3(3) plot the t – testing error below:

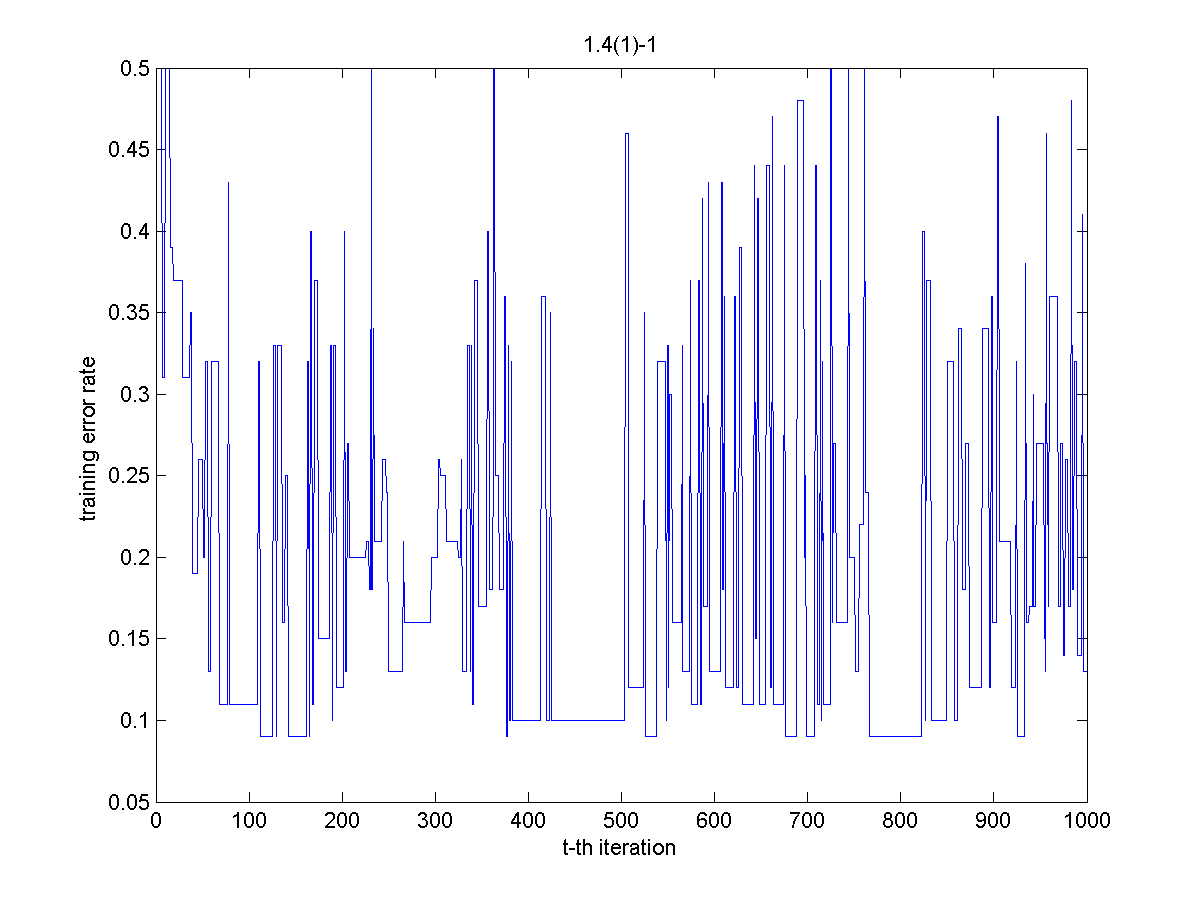


We markup the `-1‘ label as `x’ and the `1’ label as `o’. Hence our training error become 0 at the end, our decision function can separate the different label clearly.

Brief conclusion:

* The test data of 1-3 is clear, so our training result is clear too.

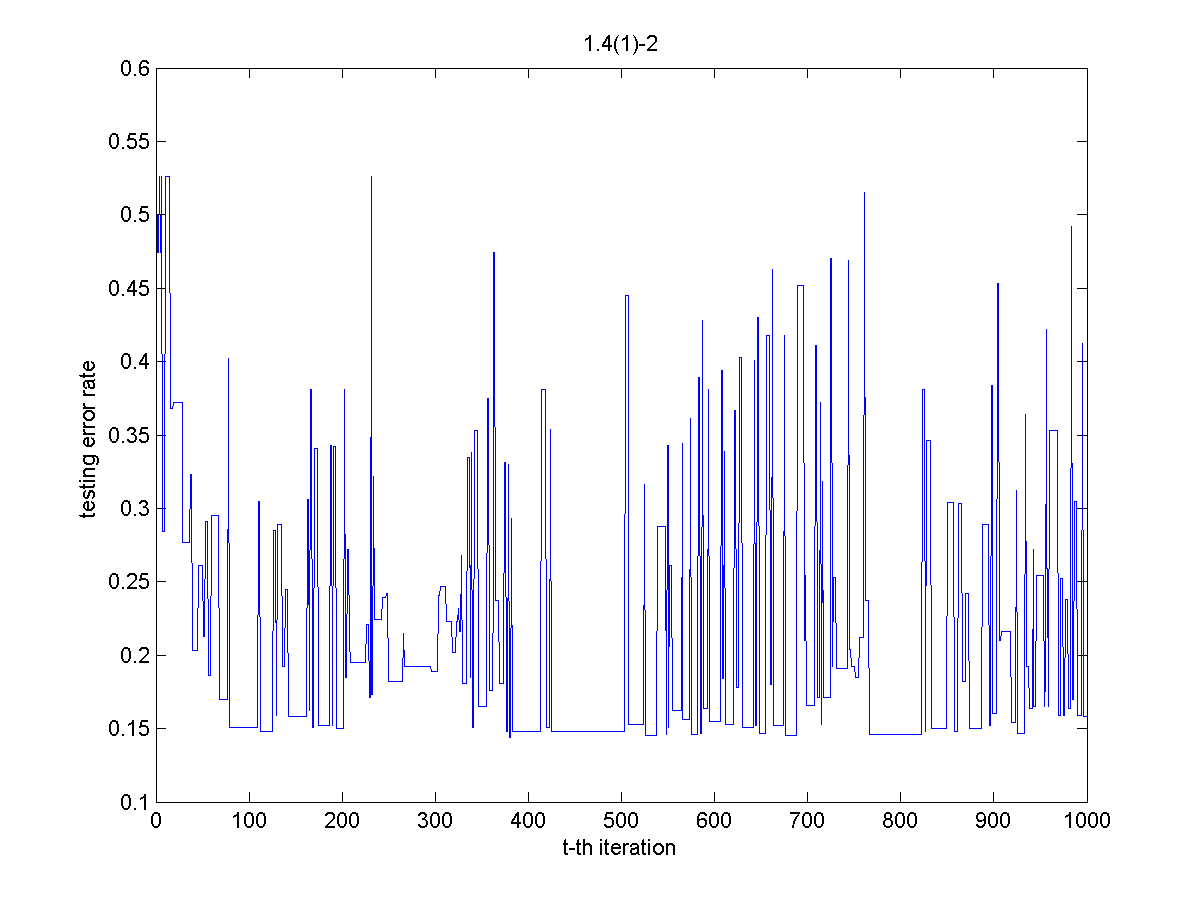
1. **Perceptron Learning on Noisy Examples (\*)**
   1. 1. The figure 1.4(1)-1 plot the t-training error below:



The training error rate lied at 0.50 at the beginning and lies around 0.09 to 0.50. It never moves to 0 and it didn’t converge.

Brief conclusion:

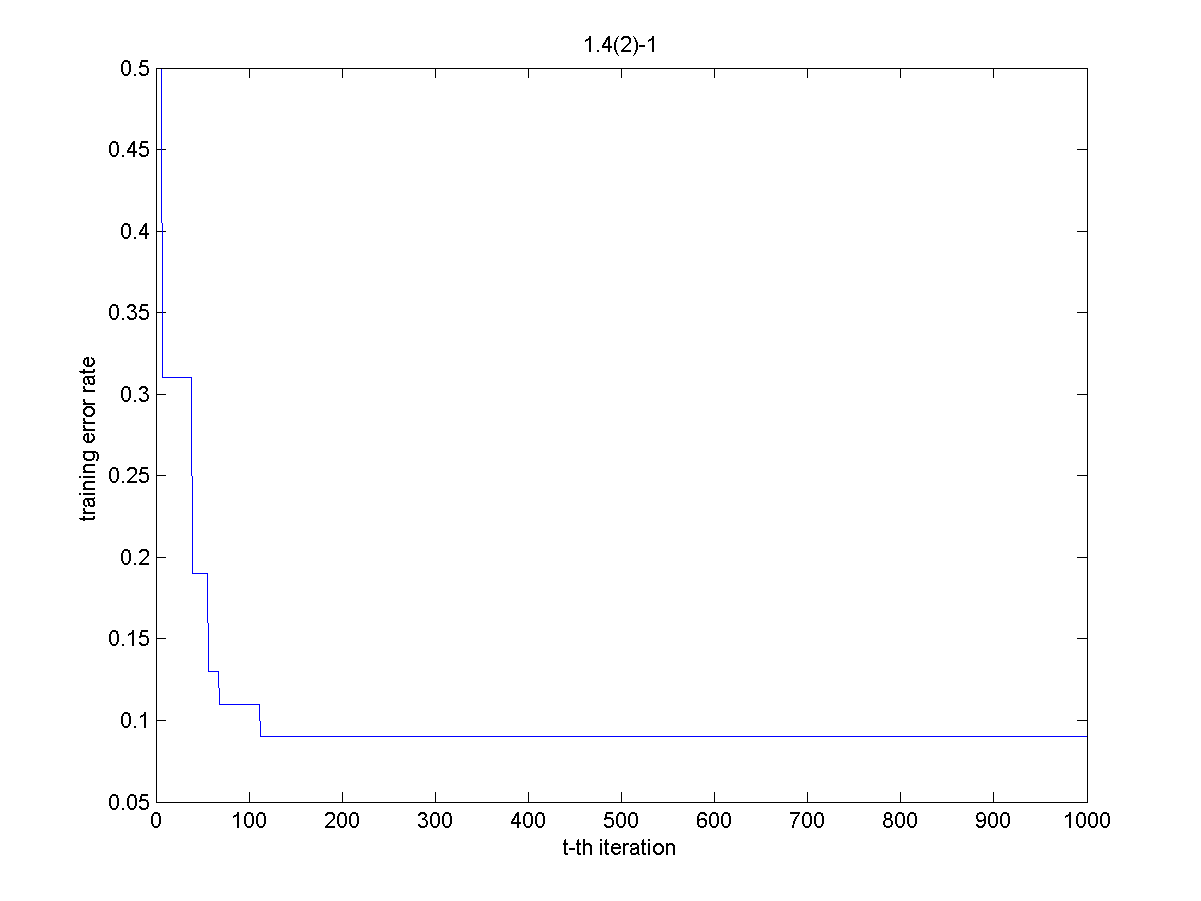
* The training error cannot converge at all.
* The training error won’t be better when the iteration be bigger.
* The noise of training data set made the decision function even cannot work at training set.
  + 1. The figure 1.4(1)-2 plot the t-testing error below:



The testing error lied on 0.526 at the beginning and lied around 0.15 to 0.526. It never came to 0.

Brief conclusion:

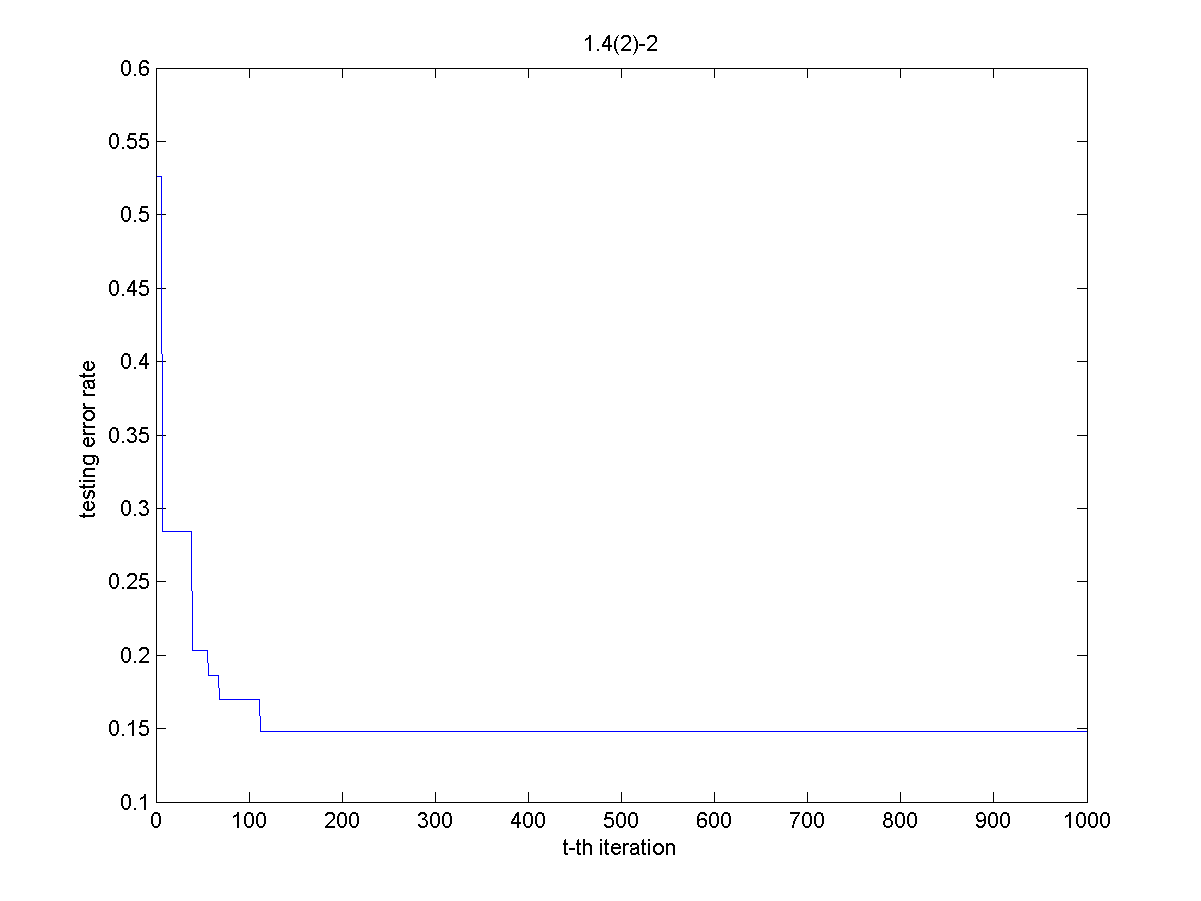
* The testing error cannot converge at all.
* The maximum testing error is bigger than the maximum training error.
* The minimum testing error is bigger than the minimum training error too.
* The testing error won’t be better when the iteration get bigger.
  1. 1. The figure 1.4(2)-1 plot the t-training error below:



The training error rate lied on 0.5 at the beginning and it is non-increased. The minimum of training error is 0.09.

Brief conclusion:

* The training error won’t be increased because we always picked out the best training function g for g\*. The best means that function hold the minimum training error. So g\* won’t get a training error bigger than training error we got before.
  + 1. The figure 1.4(2)-1 plot the t-training error below:



The training error rate lied on 0.5 at the beginning and it is non-increased.

Brief conclusion:

* The testing error won’t be increased because we are lucky enough, maybe we climb to the global best solution.
* We always picked out the best training function g for g\*. The best means that function has the minimum training error. However, the best training error (the local best) cannot guarantee the best testing error (the global best).