3.1 Radial-basis function networks

3.2 Computing the weight matrix

- What is the lower bound for the number of training examples, N?
- What happens with the error if $N = n$? Why?
- Under what conditions, if any, does (4) have a solution in this case?
- During training we use an error measure defined over the training examples. Is it good to use this measure when evaluating the performance of the network? Explain!

1) It needs to be larger than 1.
2) $rank(Phi) \leq n$ There will only one solution for equation (4). The training error will be zero.
3) $N \leq n$
4) No. We need to test the model on test set but not on the training set.

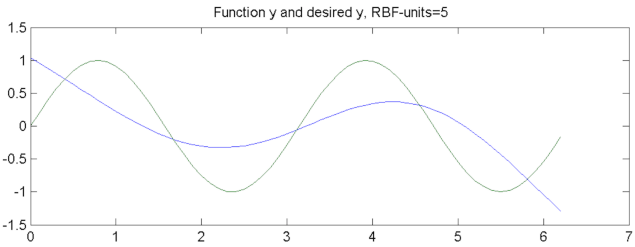
4.1 Batch mode training using least squares

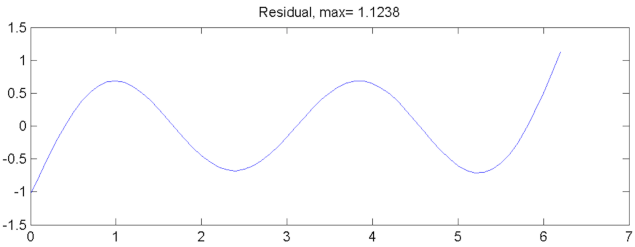
- How many units did you require to get down to a maximum (absolute) residual value of 0.1, 0.01 and 0.001?
- Give a good reason for the big difference in residual between 5 and 6 units for $\sin(2x)$.
- How many units did you require, when approximating square$(2 * x)$, to come down to residual values of 0.1, 0.01 and 0.001?
- Approximating square$(2 * x)$ is a somewhat special case of function approximation since it is similar to another area of use for artificial neural networks. Which?
- Can you, with a suitable action (e.g. transforming network output), easy get down (for training values) to a residual value=0? What action? How many units did you require?
- Can an RBF network solve the XOR problem? If not, explain why not. If yes, explain how.

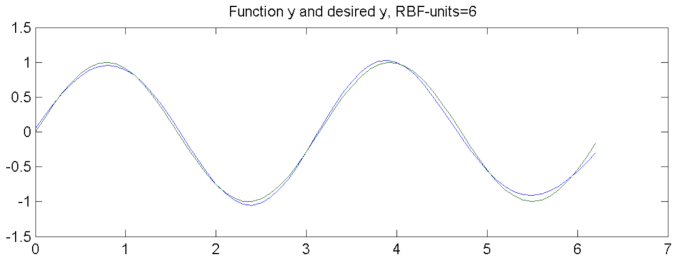
1) 3)

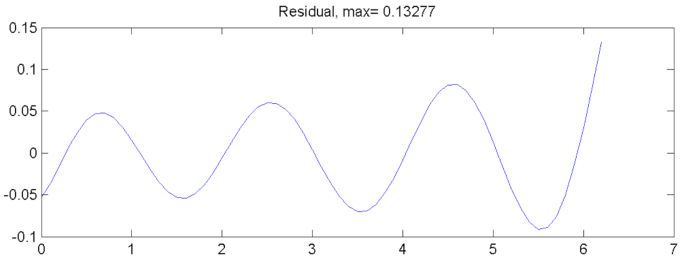
Sin	Square
unit = 7 25 56 residual=0.0941 0.0099 0.0009	unit = 44 59 61 residual=0.0974 0.0065 0.0004

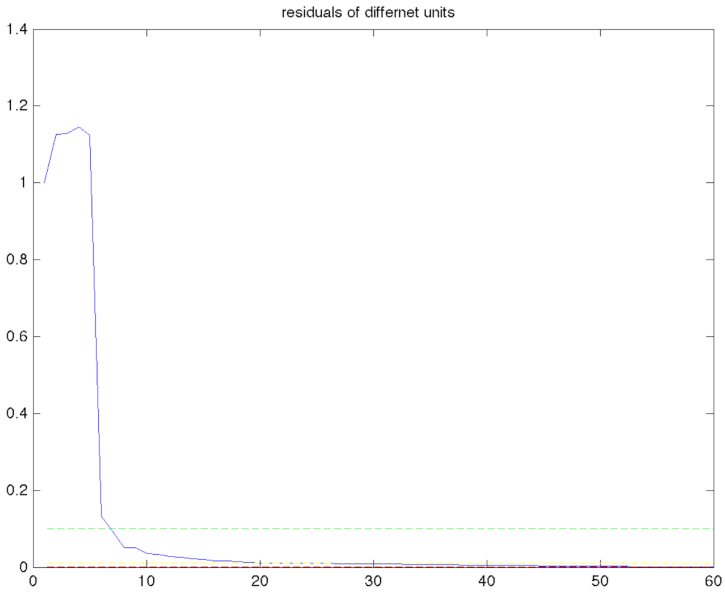
2) At some points, more RBF units give better approximation.











4) Using two layers of feed forward neural network with different hidden units to approximate the Gaussian functions (in lab1).

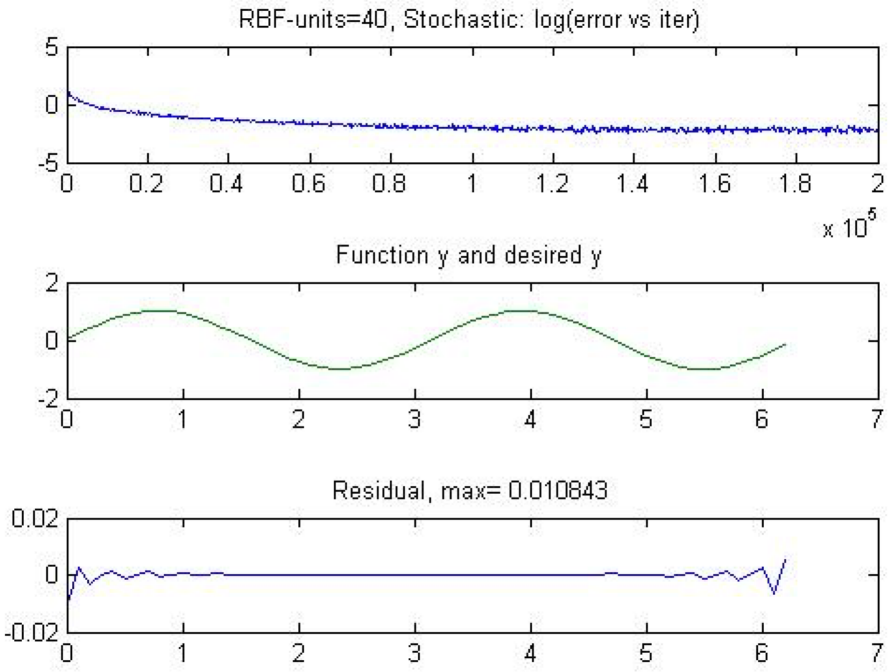
5) When the number of training samples $N \leq n$, the residual value=0.

6) The RBF can solve the XOR problem only when $N \leq n$, i.e. $n \geq 4$. n needs to be larger than the patterns or classes of the dataset.

4.2 On-line training using the delta rule

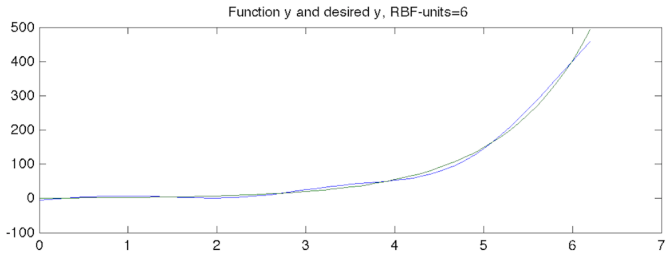
- How many *units* and *iterations* did you require to come down to a maximum residual value of 0.01? What value(s) of η did you use?
- Now try approximating some function of your own choice. Use least squares or the delta rule as you wish.

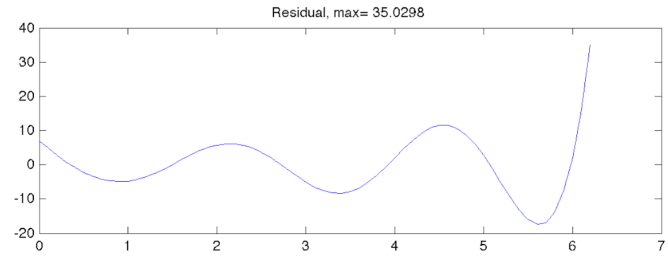
1) units=40; itersub =20; itermax =20000;eta=1;
   units larger: better but slower (more units to train);
   more iterations: better but slower;
   smaller eta: need more time to be optimal (converge slowly).



2) Use least squares to approximate exponential function.

Possible reason: Square function and exponential are both monotonous



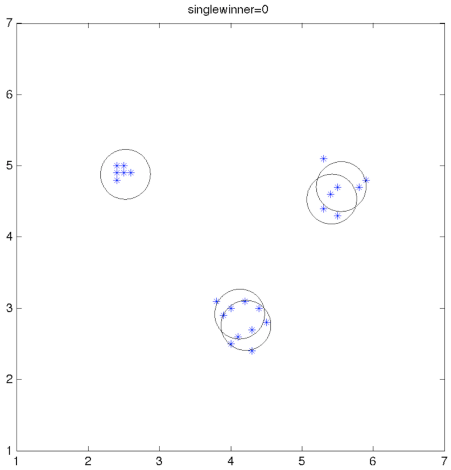


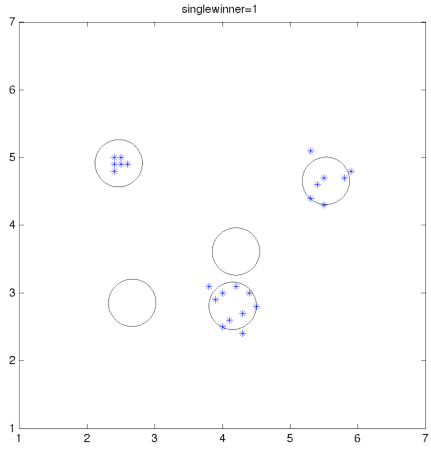
Sin	Square
unit = 7 25 56 residual=0.0941 0.0099 0.0009	unit = 44 59 61 residual=0.0974 0.0065 0.0004
exponential	
unit =59 61 62 residual=0.0690 0.0043 0.0000	

5 RBF Placement by Self Organization

- What problem could be seen when using the single winner strategy (singlewinner=1)?
- What is the advantage of using this strategy?

1) When using single winner strategy, only the winner units can get update.



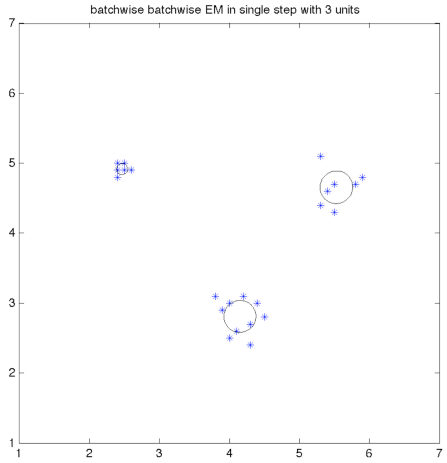


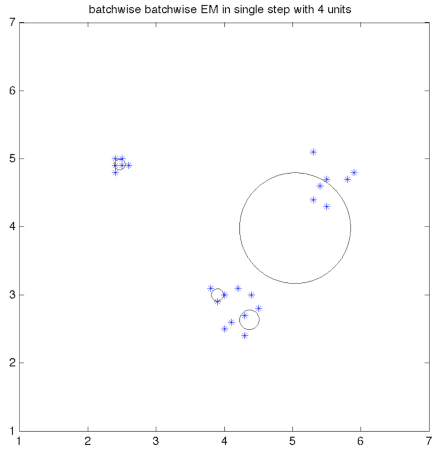
2) It converges faster and results in less useful RBF units (we can throw away those useless RBF units that do not represent any cluster.)

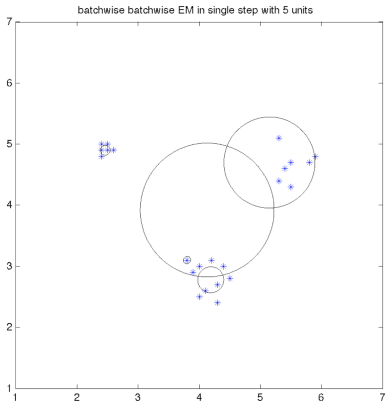
EM algorithm

1) Batchwise EM (single winner strategy: singlewinner=1)

Units = 3,4,5



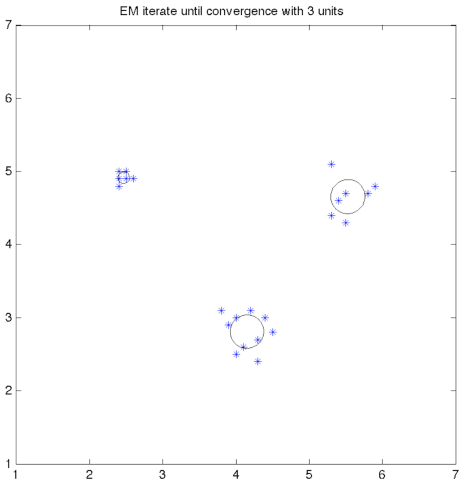


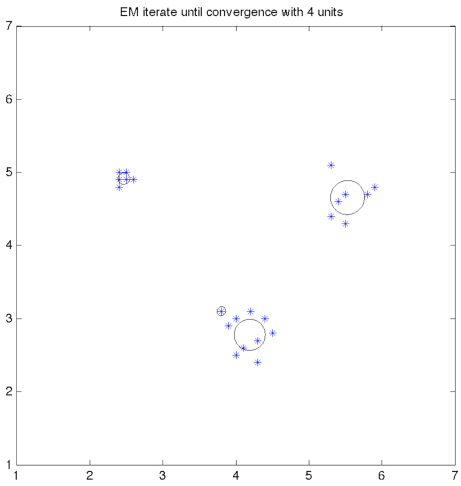


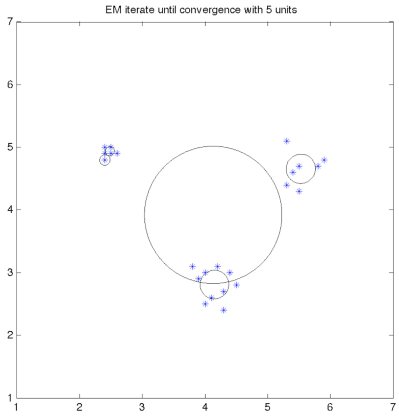
When using single winner strategy, only when units=3, the Batchwise EM can give good clustering result.

2) Iterative EM (single winner strategy: singlewinner=1)

Units = 3,4,5





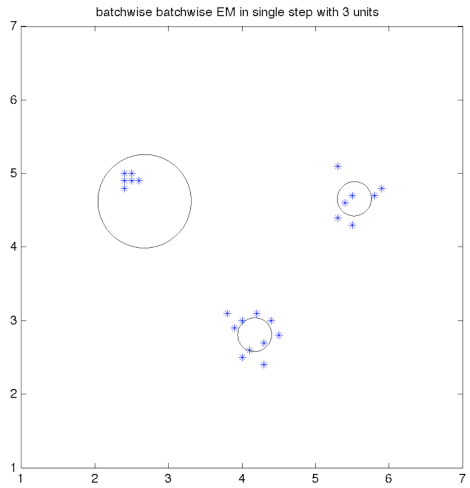


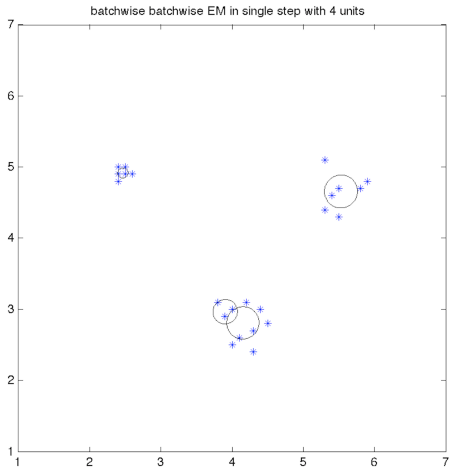
When using single winner strategy, when units=3 or 4, the iterative EM can give good clustering results.

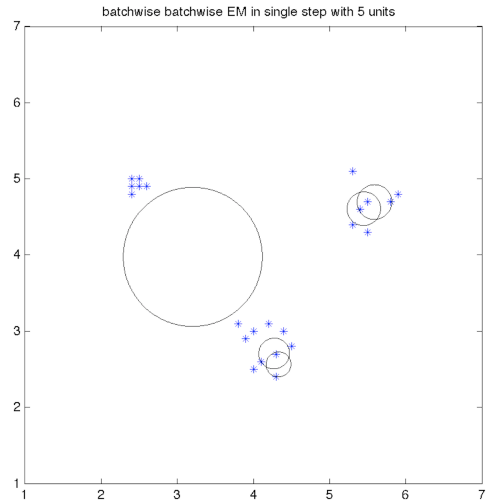
Therefore, iterative EM is slightly better than Batchwise EM. It gives more stable convergence.

3) Batchwise EM (do not use single winner strategy: singlewinner=0)

Units = 3,4,5





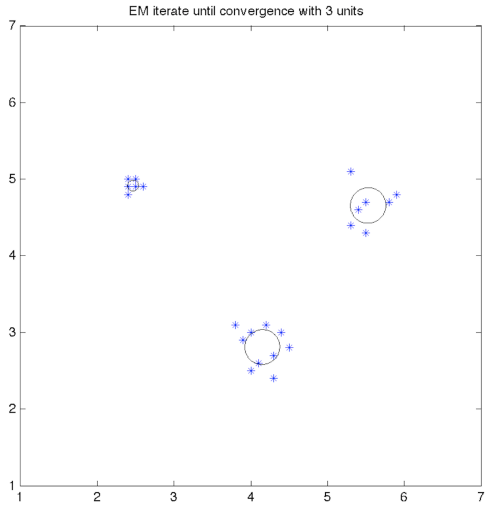


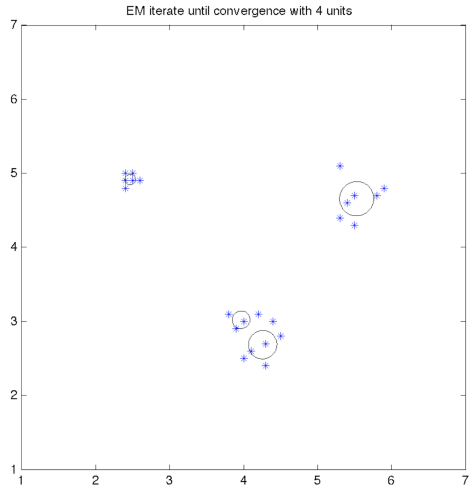
When not using single winner strategy, when units=3 or 4, the Batchwise EM can approximate the clusters. But it is bad on some certain that it is likely to

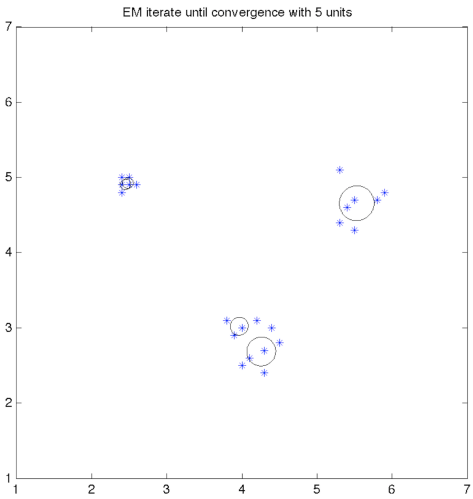
estimate more than two clusters to approximate a cluster!

4) Iterative EM(do not use single winner strategy: singlewinner=0)

Units = 3,4,5







When not using single winner strategy, when units=3 or 4 or 5, the iterative EM can approximate the clusters. But it is bad on some certain that it is likely to estimate more than two clusters to approximate a cluster!

- Describe differences between using the single winner strategy (singlewinner=1) and allowing all units to move (singlewinner=0) for the batchwise algorithm.

When using the single winner strategy, only winning clusters get update. We need to know how many clusters are there in advance. Otherwise, there will be some useless estimated Gaussians that do not approximate any true cluster.

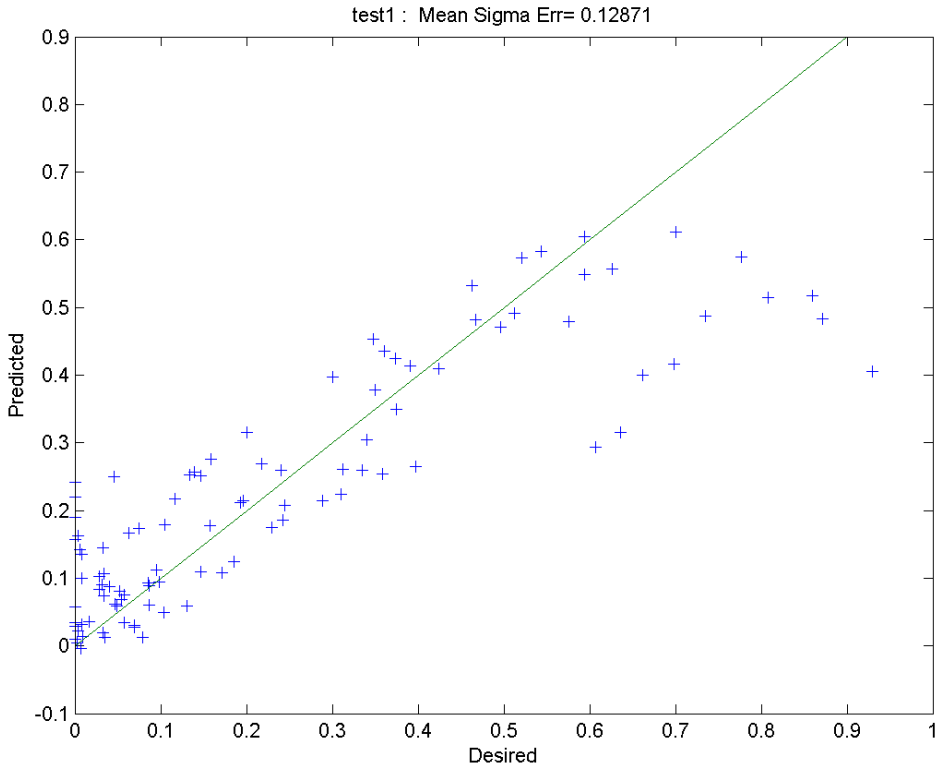
When allowing all units to move, all the estimated clusters get update in each iteration. We do not need to know how many clusters are there in advance. But there will be more than one estimated Gaussians that approximate a true cluster.

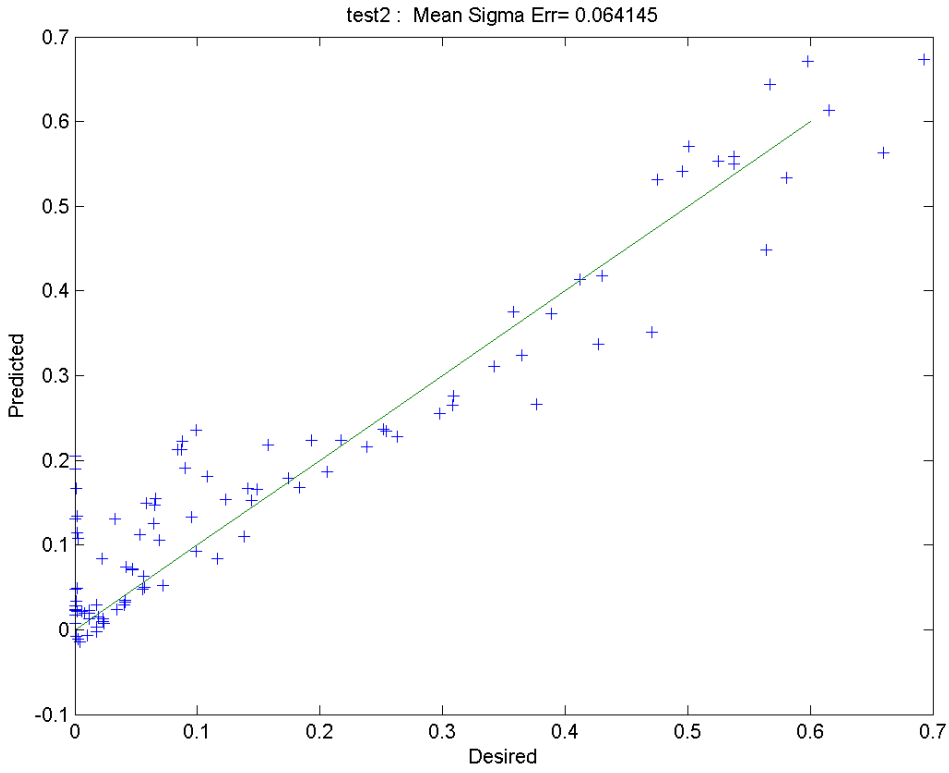
6 Function Approximation for Noisy Data

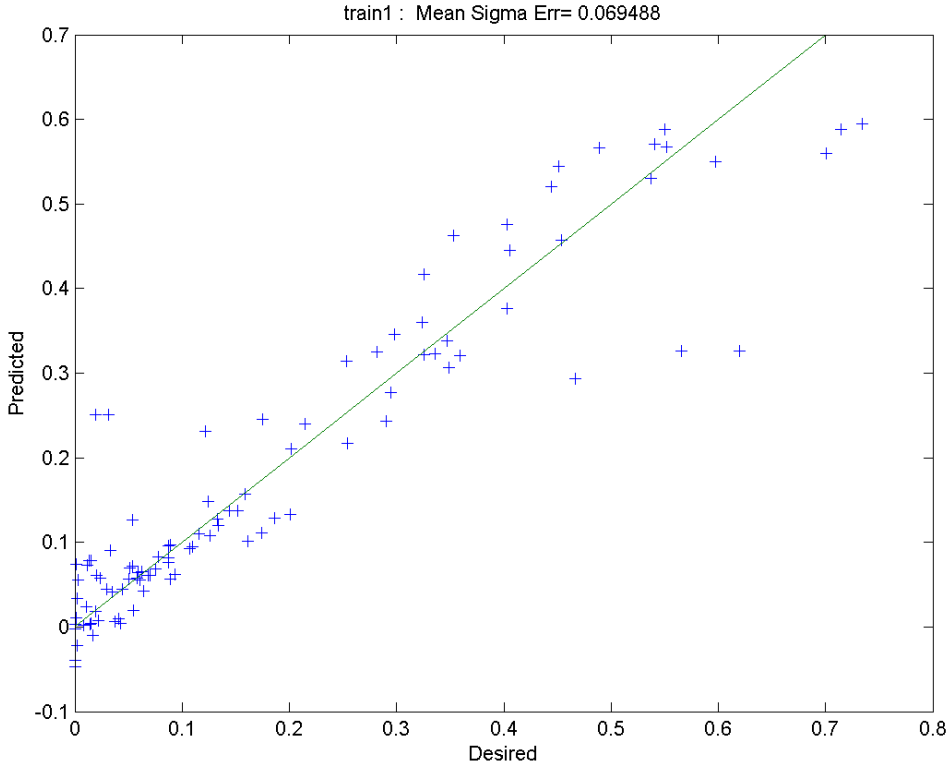
When the network har a large learning capacity, there is always the risk of overlearning. If you have time, try reducing the number of units to see if it gets better or worse. You can also test what happens if you low-pass filter the input to get rid of some noise during training.

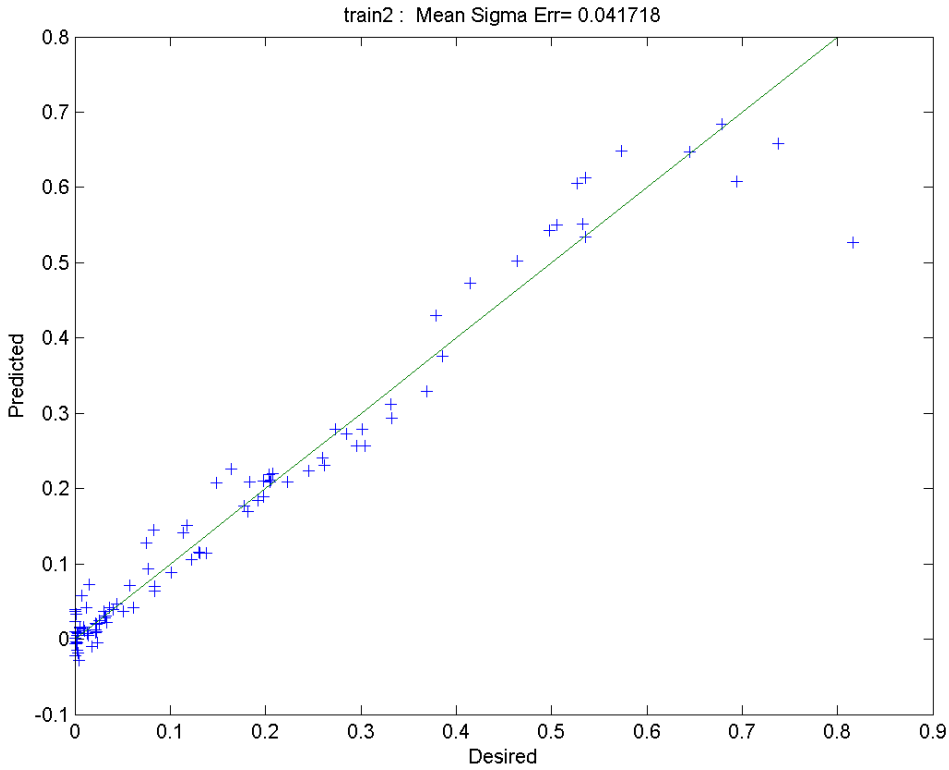
- Did you try these improvements? Did it help?

Improvement1: reducing numbers of units
Units=20

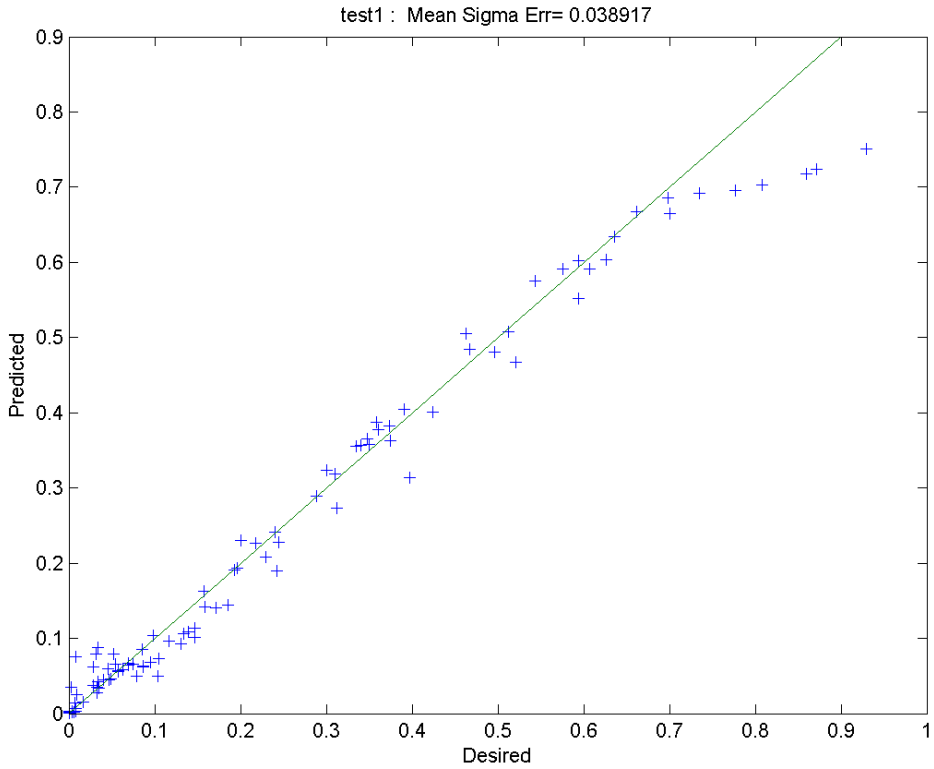


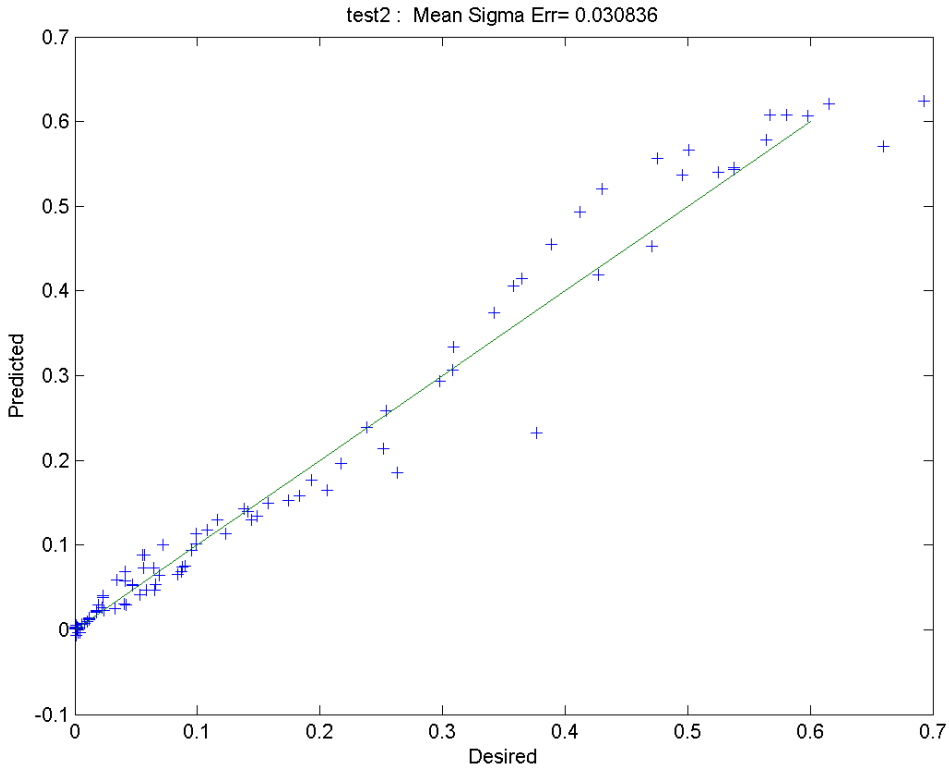

test2 : Mean Sigma Err= 0.064145
Predicted
Desired
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

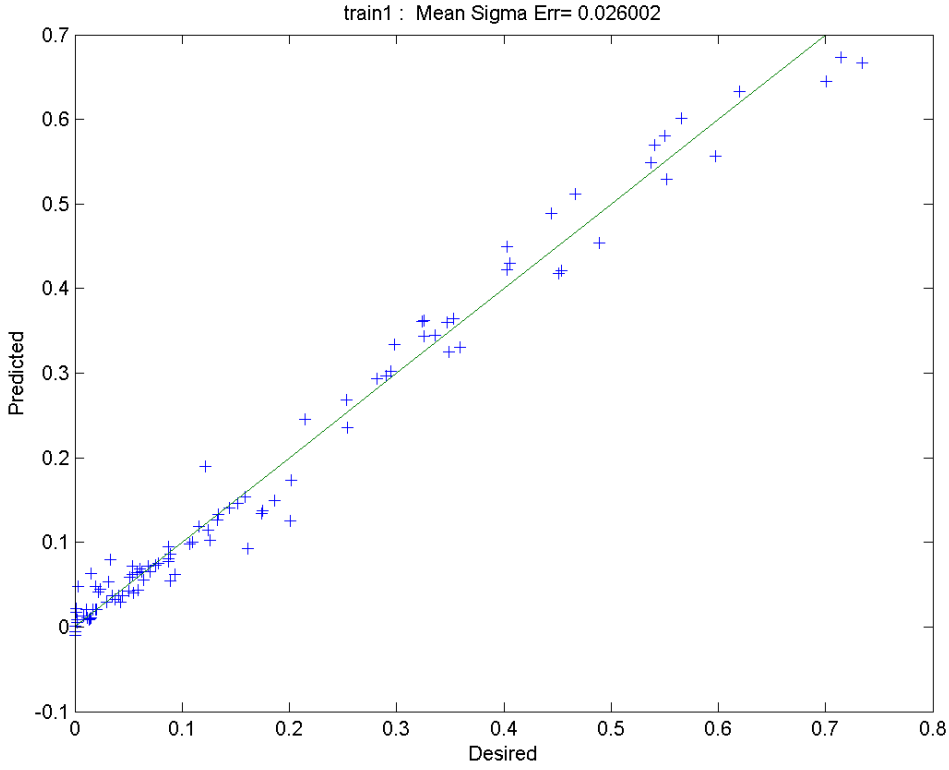

train1 : Mean Sigma Err= 0.069488
Predicted
Desired
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8

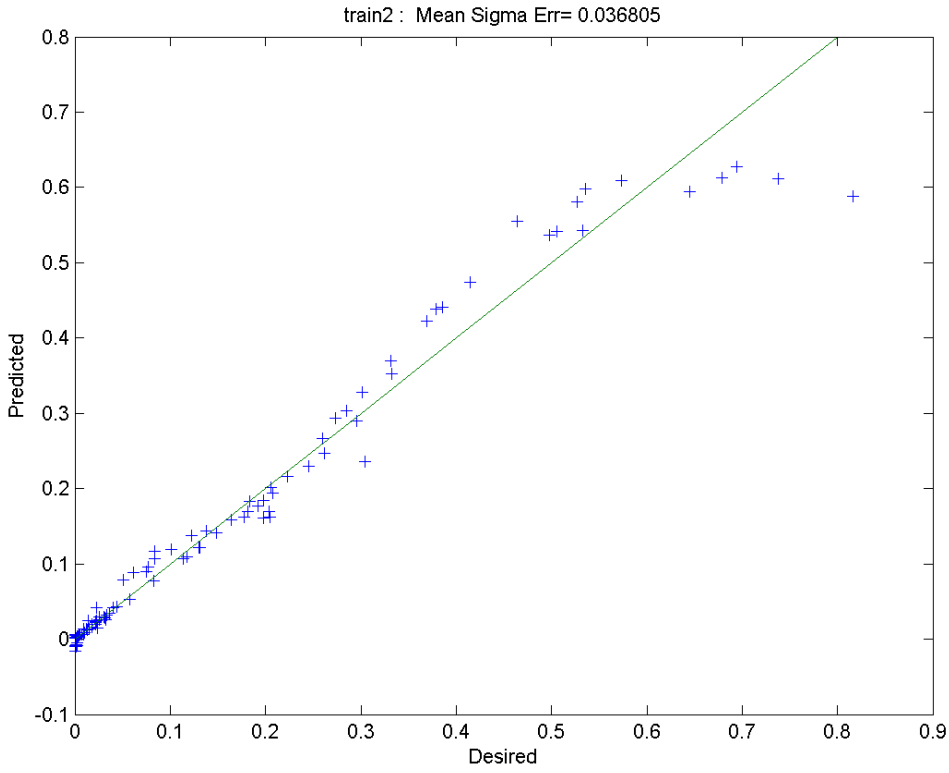

train2 : Mean Sigma Err= 0.041718
Predicted
Desired
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

Units=15:

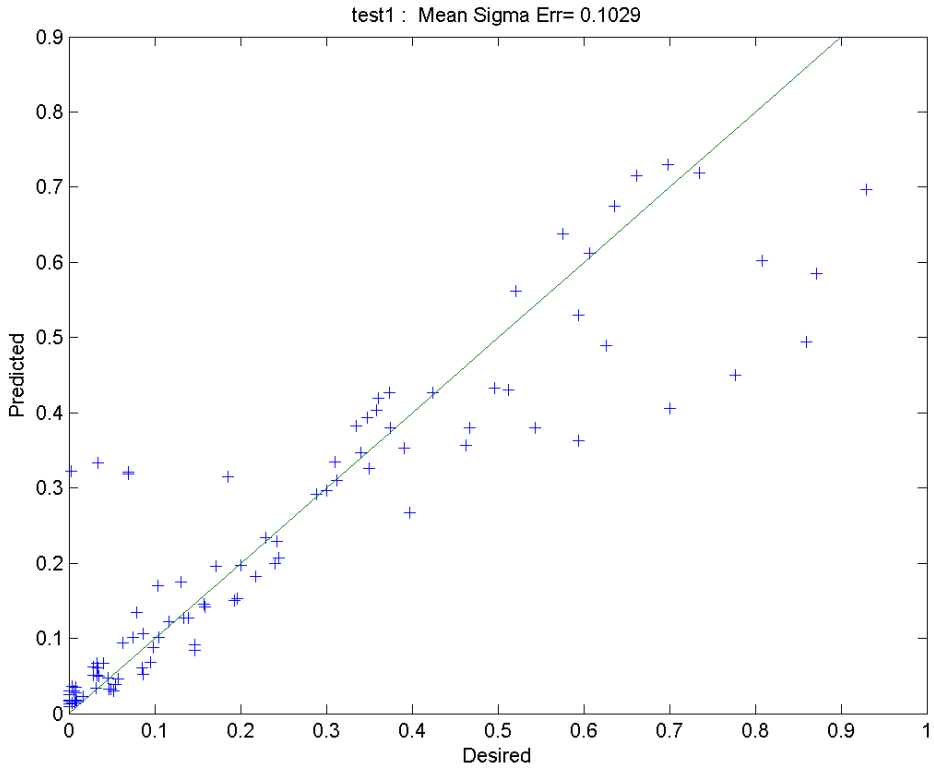

test1 : Mean Sigma Err= 0.038917
Predicted
Desired
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

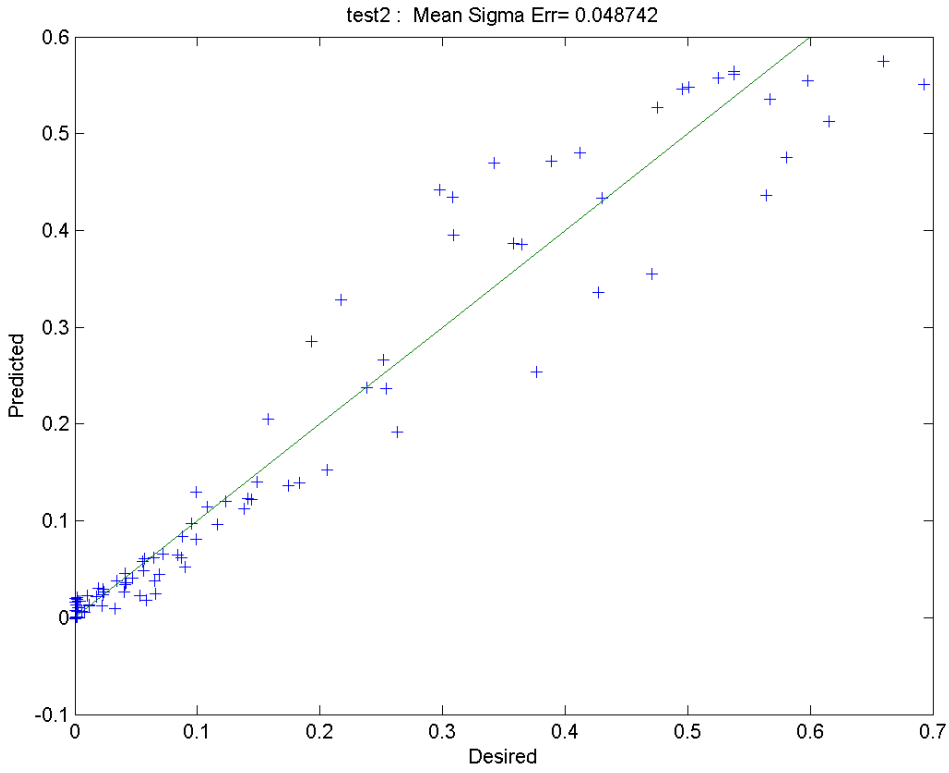

test2 : Mean Sigma Err= 0.030836
Predicted
Desired
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

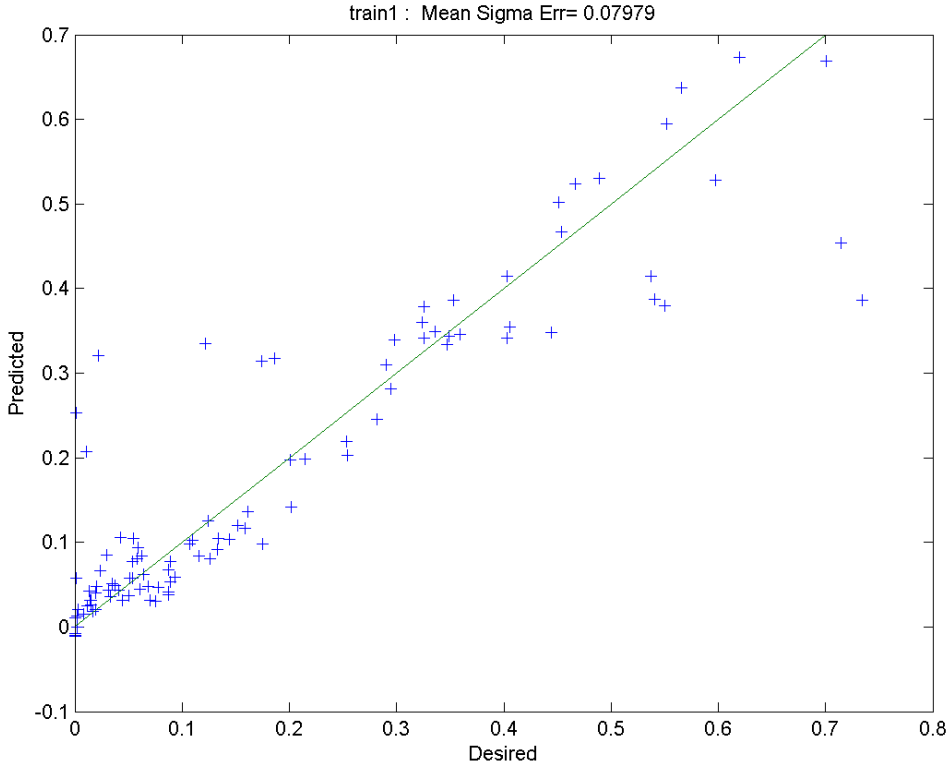

train1 : Mean Sigma Err= 0.026002
Predicted
Desired
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8

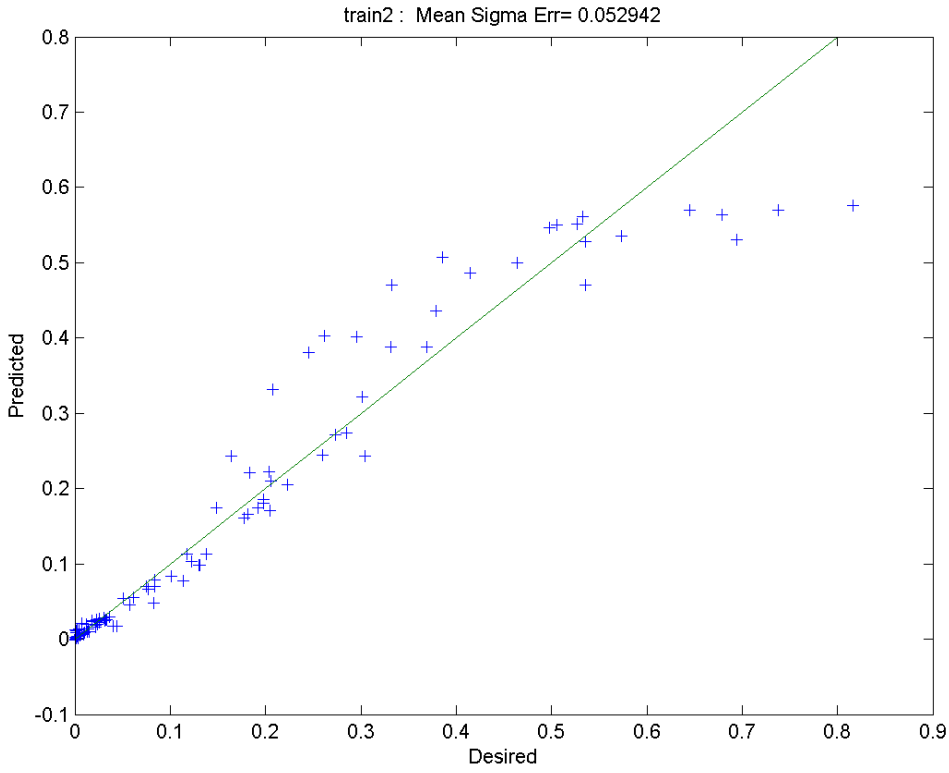

train2 : Mean Sigma Err= 0.036805
Predicted
Desired
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

Units=10:

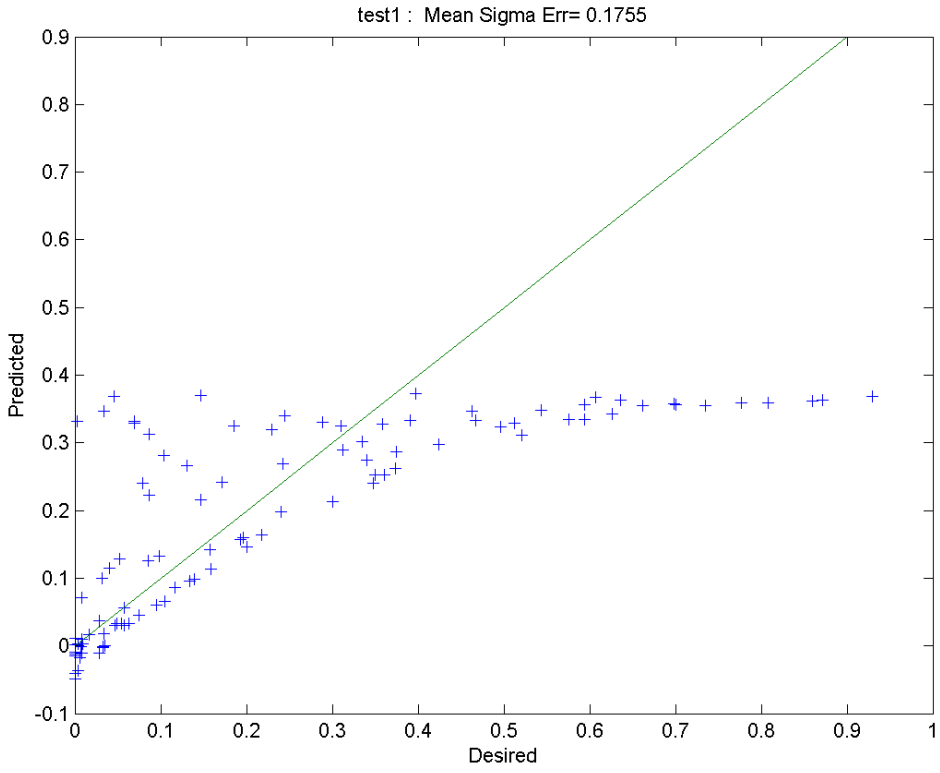

test1 : Mean Sigma Err= 0.1029
Predicted
Desired
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

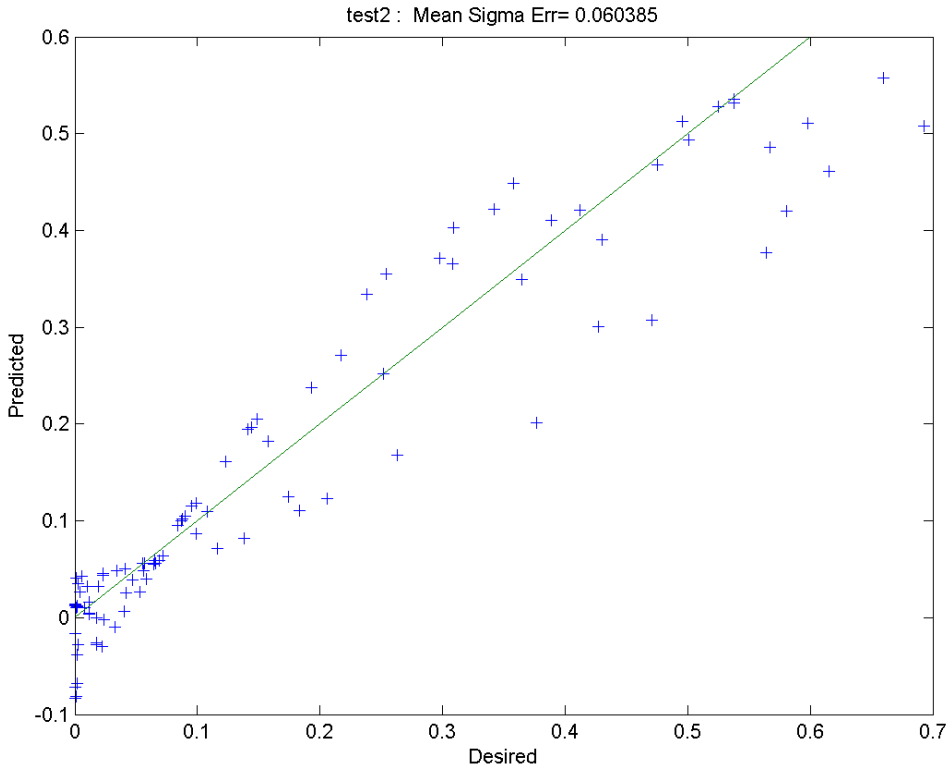

test2 : Mean Sigma Err= 0.048742
Predicted
Desired
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

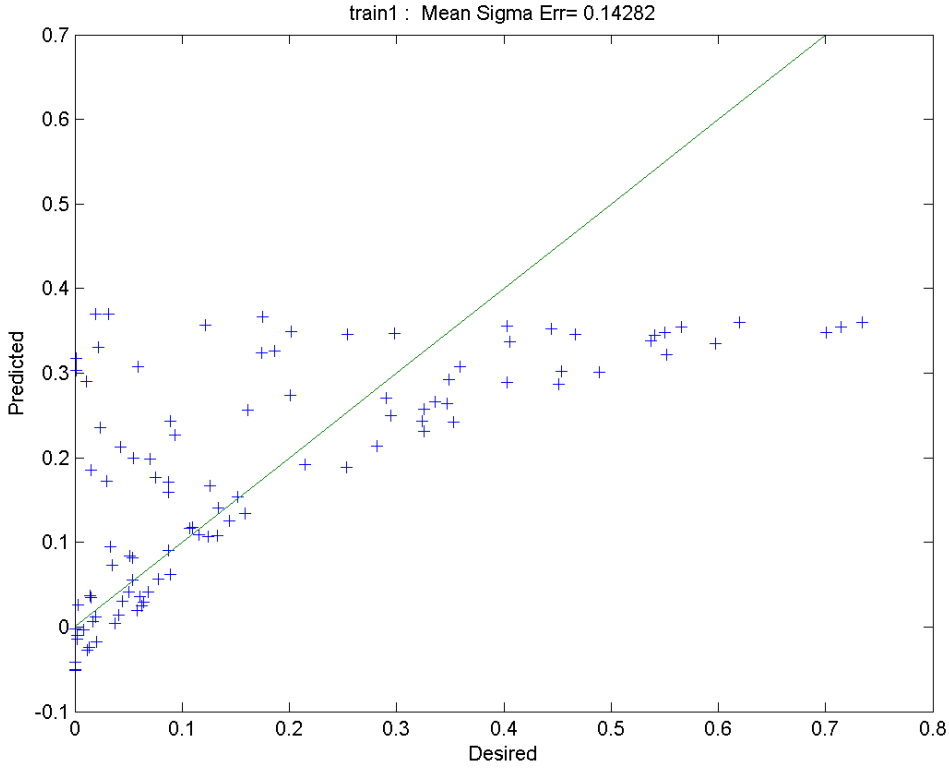

train1 : Mean Sigma Err= 0.07979
Predicted
Desired
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8

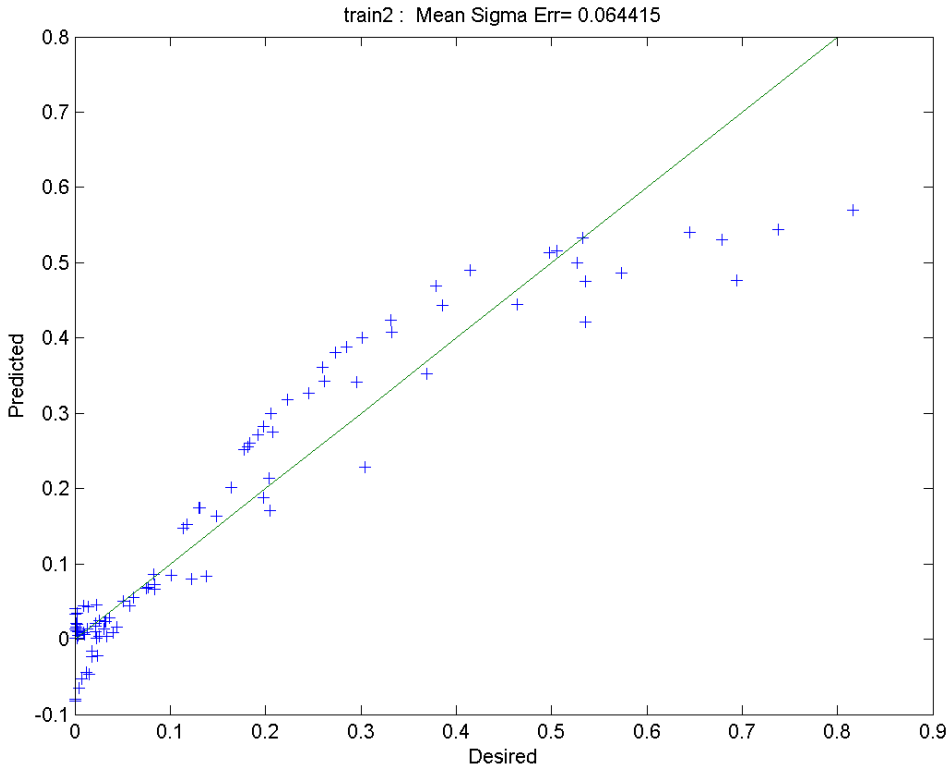

train2 : Mean Sigma Err= 0.052942
Predicted
Desired
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

Units=5:


test1 : Mean Sigma Err= 0.1755
Predicted
Desired
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


test2 : Mean Sigma Err= 0.060385
Predicted
Desired
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7


train1 : Mean Sigma Err= 0.14282
Predicted
Desired
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8



Both training error and test error will reduce when units change from 20 to 15. When the units further decrease, the error will increase. The following table can show this process.

Units	20	15	10	5
Train1	0.069488	0.026002	0.07979	0.14282
Train2	0.04178	0.036805	0.052942	0.064415
Test1	0.12891	0.03817	0.1029	0.1755
Test2	0.064145	0.030836	0.04874	0.060383

The original positions of units and derivation are same.

Improvement2: using low pass filter for the input to get rid of noise

We try with different filters (low pass filter, median filter, moving averaging filter, see link:

http://se.mathworks.com/help/signal/examples/signal-smoothing.html#zmw57dd0e232)

But it does not help to improve the results. We find it hard to choose an appropriate filter.

We consider that the main reason is that the dataset is not appropriate to be processed with low pass filter.