Q1 Function Approximation with RBFN (10 Marks)

a) In the first case, RBFN is trained using the 'Exact Interpolation' method, and the fitting result is displayed in Figure 1.

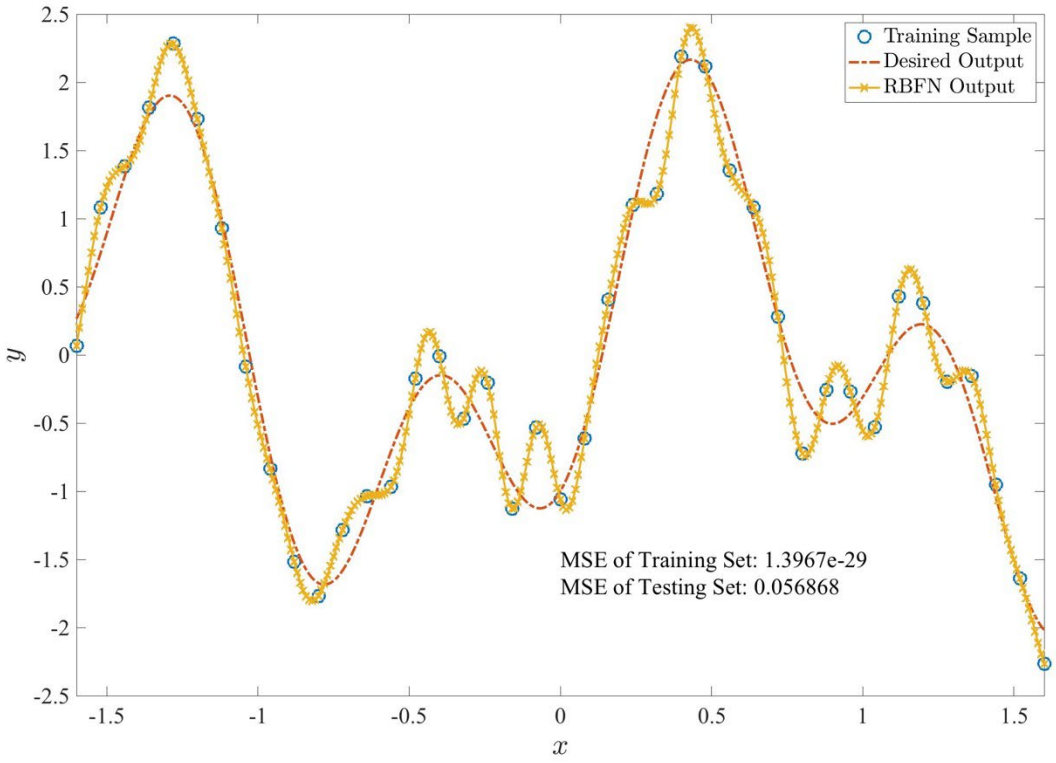


Figure 1 Fitting result obtained via the method of Exact Interpolation.

Mean Square Error (MSE) upon the training set is 1.3×10^{-29}, which is approximately 0, since the weights of RBFN are derived to perfectly fit all the training samples. In the meanwhile, MSE upon the test set is 0.057, which is due to the inability of RBFN to model the underlying function corrupted by noise.

b) In this case, only 20 training samples are randomly selected as the centers of RBFN. Moreover, the bias term is introduced following the fashion given in Page34 of Lecture5. The fitting result is shown in Figure 2. Compared with a), MSE drops to 0.013 for the test set while increases to 0.046 for the training set. Additionally, the fitting curve obtained in this case is much smoother than that obtained in section a).

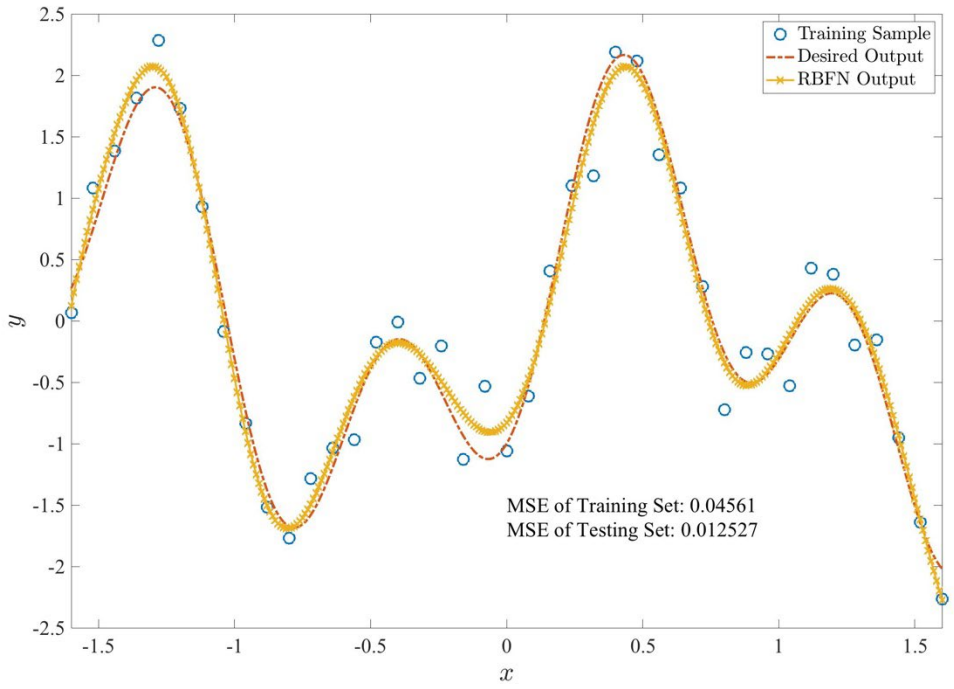
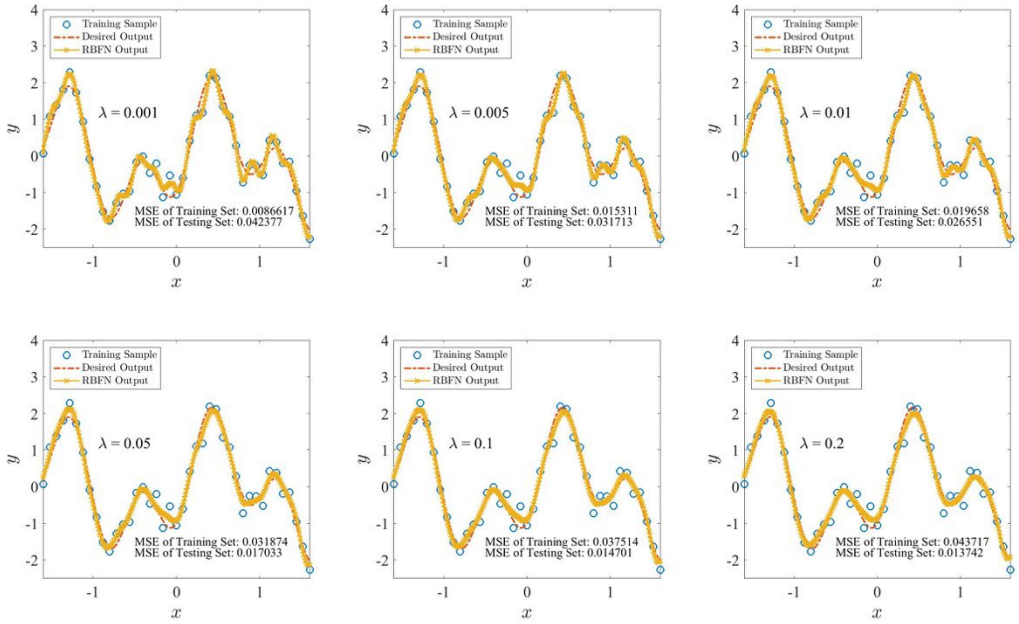


Figure 2 Fitting result obtained via the strategy of Fixed Centers (20) Selected at Random.

c) Use the same centers and widths as those determined in a) and apply the regularization with varying strength (λ) to RBFN. The obtained fitting results are given in Figure 3, and the relation between regularization strength and the fitting performance of RBFN is illustrated in Figure 4.



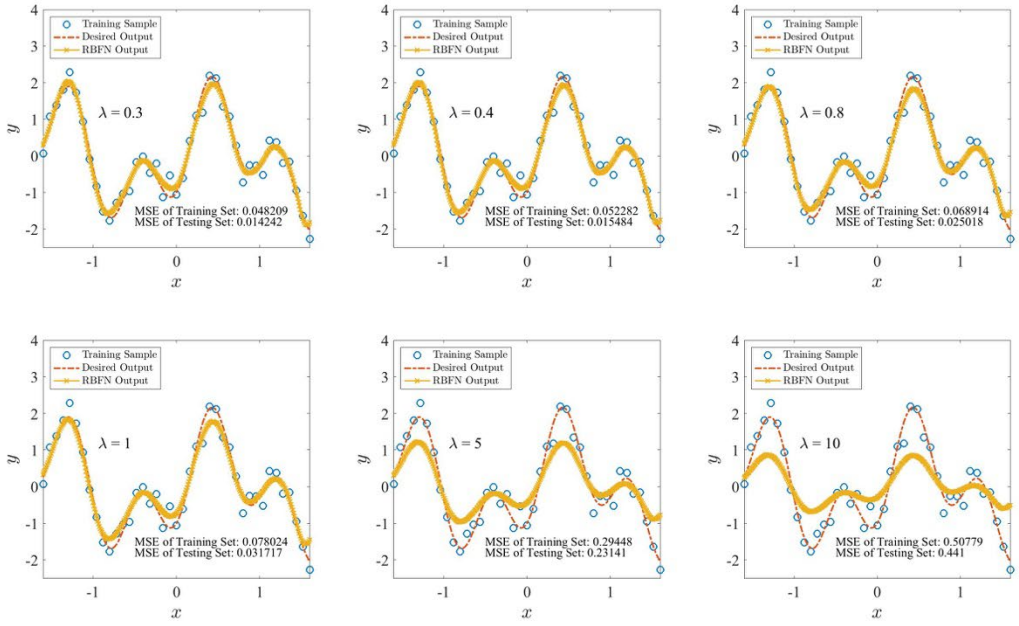


Figure 3 Fitting results obtained via the method of Exact Interpolation with different regularization strength.

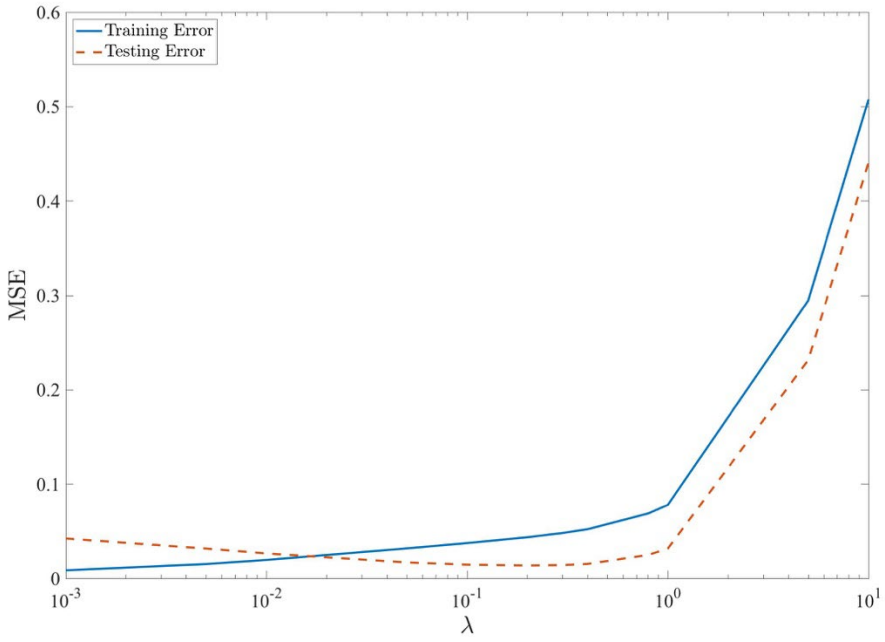


Figure 4 MSE upon the training/test set versus regularization strength.

It can be found from Figure 3-4 that the fitting curve becomes smoother as the regularization strength increases. In particular, when $\lambda = 0.2$, the performance of resulted RBFN reaches its best. However, if the regularization strength continues to grow, slopes of the obtained curve would become too flat to properly fit the underlying function and the MSE starts to rise.

Q2 Handwritten Digits Classification using RBFN (20 Marks)

The following experiments select class 7 and class 8 as label '1' and all the remaining classes as label '0' for demonstration.

a) Use the 'Exact Interpolation' method and apply regularization with different strength to determine the weights of RBFN. Classification performance of the obtained RBFNs is compared in the following figures.

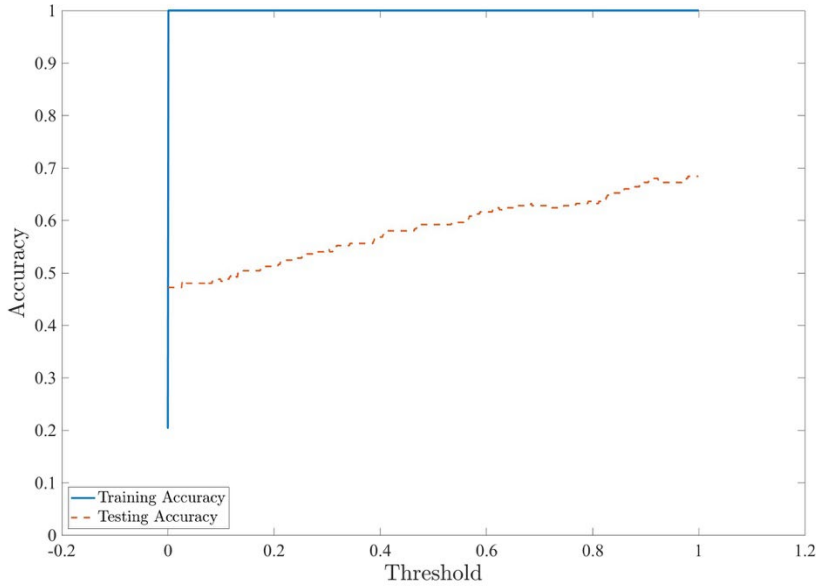


Figure 5 The performance of RBFNs using exact interpolation method.

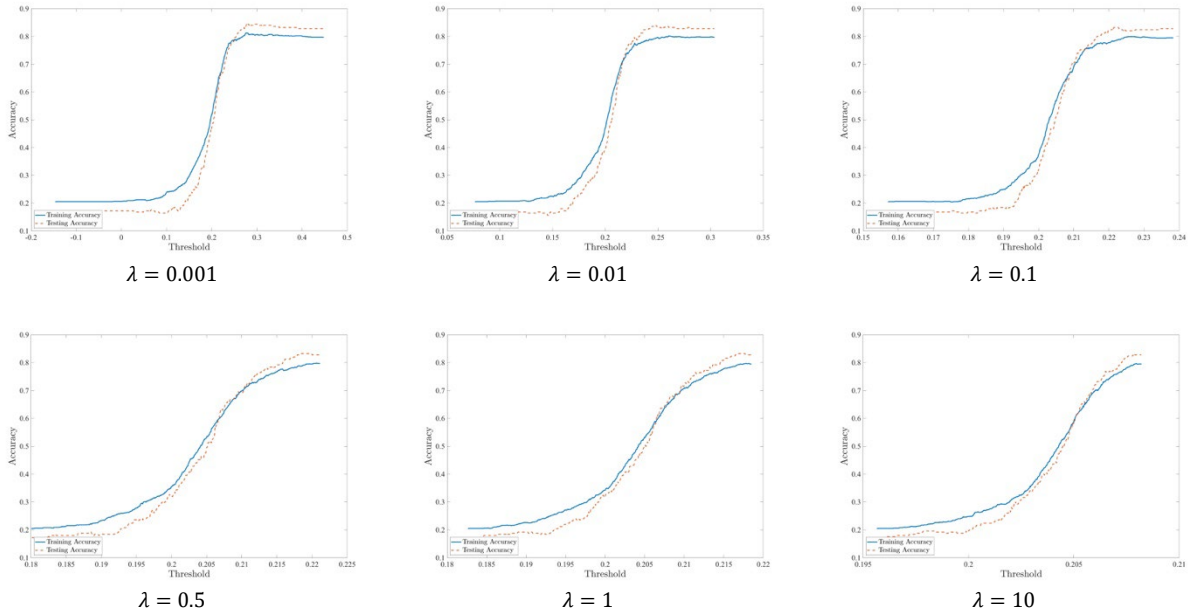


Figure 6 Comparison of the performance of RBFNs that are yield under different regularization strength.

It can be found from the above figures that, in the case of Exact Interpolation, the training accuracy jumps to and remains at 100% once the threshold passes 0. This confirms the fact that RBFN obtained via Exact Interpolation does pass through each individual training sample exactly. However, such RBFN performs poorly on the test set with the test accuracy around 68%, i.e. RBFN overfits on the training set.

After applying regularization, the training accuracy cannot be 100% anymore which is natural since regularization will smoothen the output and RBFN is not purely exact interpolation for the training set, i.e. reduce the overfitting. In this case, for both training and test set, the accuracy rate are around 80%. Also, as the regularization factor λ increases, the output range becomes narrower, which means the deviation in the prediction output is smaller.

b) When randomly select 100 training samples as the centers of RBFN and fix the widths according to $\sigma = d_{max}/\sqrt{2M}$, the classification performance of the yield RBFN is displayed in Figure 7.

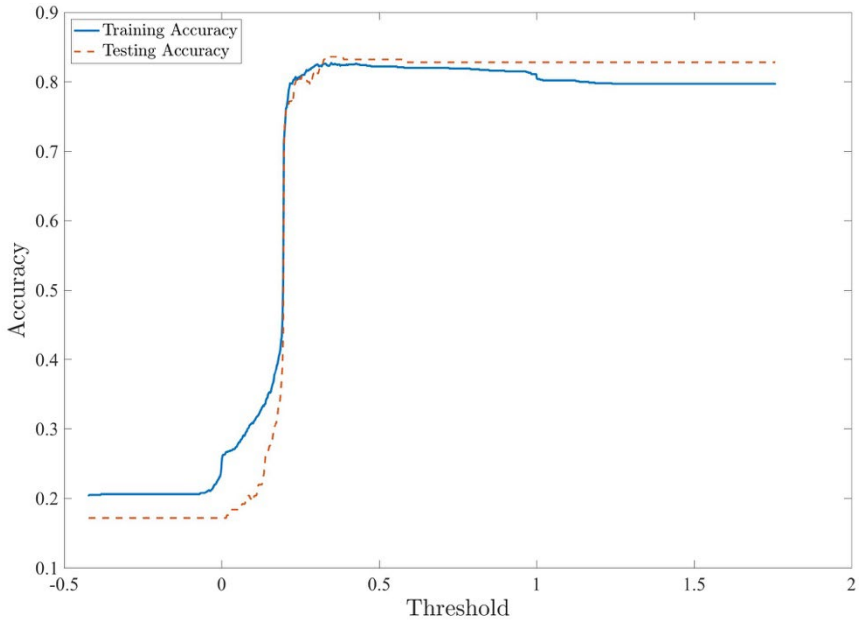
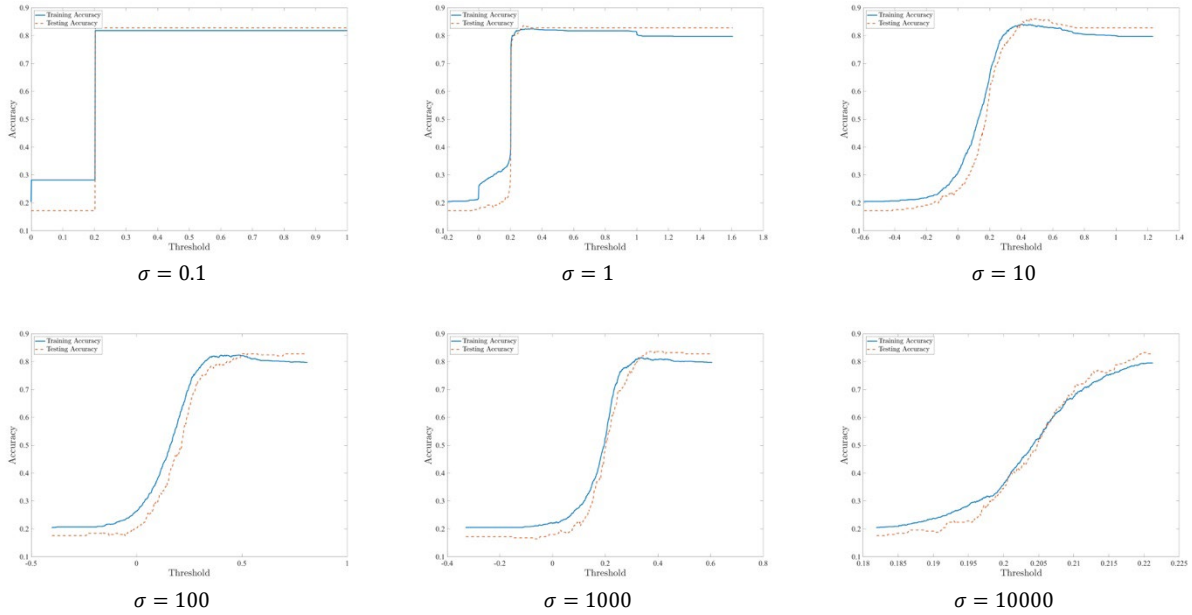


Figure 7 Performance of the RBFN with 100 centers and width $\sigma = 1.47$.

We then vary the value of width from 0.1 to 10000 as shown in Figure 8 and plot the resulted best classification accuracy under each width in Figure 9. It is found that the classification results are directly affected by the selection of σ: either a small or large σ would deteriorate the performance greatly. Furthermore, $d_{max}/\sqrt{2M}$ might not be a good universal width estimation especially when the data is of high dimension.



$\sigma = 0.1$ $\sigma = 1$ $\sigma = 10$

$\sigma = 100$ $\sigma = 1000$ $\sigma = 10000$

Figure 8 Comparison of the performance of RBFNs that are yield under different regularization strength.

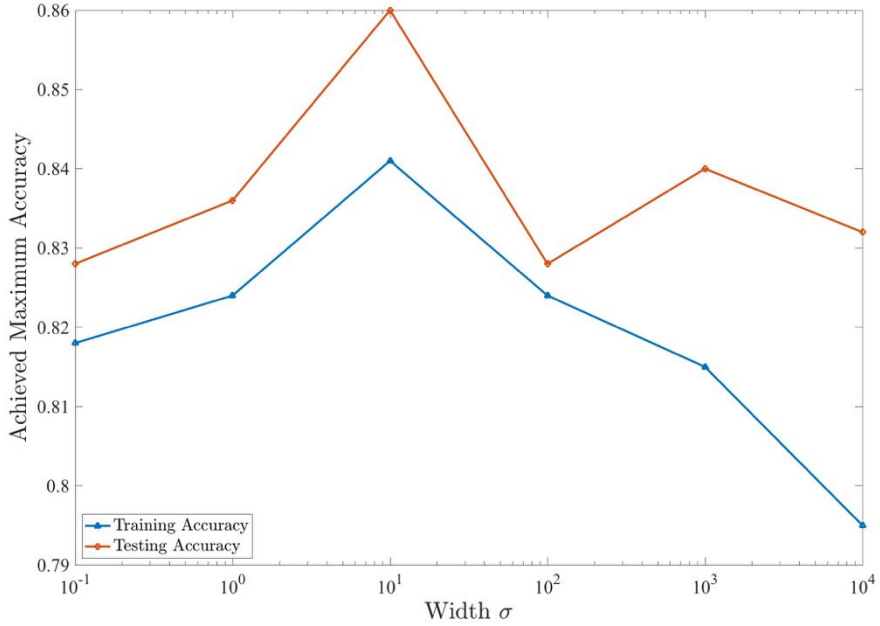


Figure 9 The performance of RBFN is sensitive to the selection of width.

c)

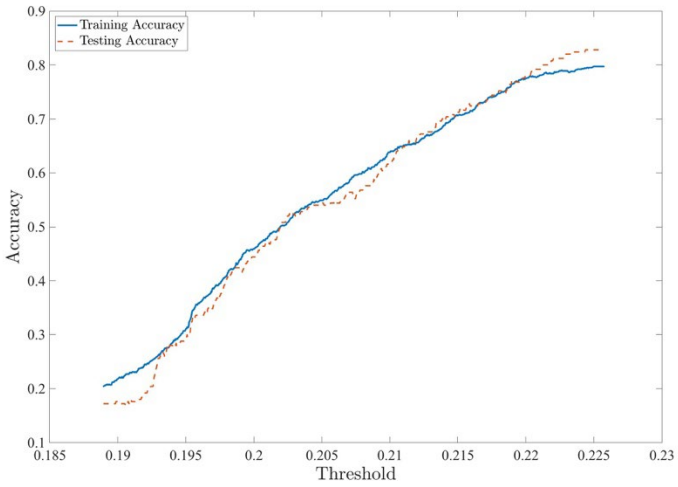


Figure 10 Performance of the RBFN with 2 centers and width $\sigma = d_{max}/\sqrt{2M}$.

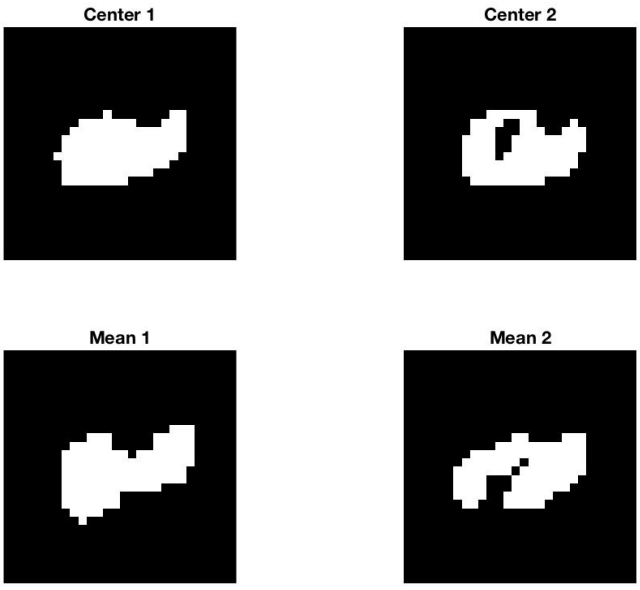


Figure 11 Comparison between the centers discovered by K-mean clustering and the mean of training samples.

Comparing Figure 10 with Figure 7, the resulted performance RBFN with 2 centers determined by 'k-means clustering' is comparable with that of 100-center RBFN in section b) but not superior obviously. But as we know in b) the 100 centers are randomly selected and here the 2 centers are determined in a meaningful manner which may be helpful to improve the performance. However, in our dataset there is large noise and the image of the digits undergo different transformations, which makes the classification task more challenging. Also the way we choose the label for two

classes lead to unbalanced data. What's more, as shown in Figure 11, it is hard to observe the clear underlying pattern of the clustering center and the mean of the image but we should know that in the simple dataset where the digits are not under rotation and reflection, the obtained 2 centers will be close to the mean of the training images.

Q3 Self-Organizing Map (SOM) (20 Marks)

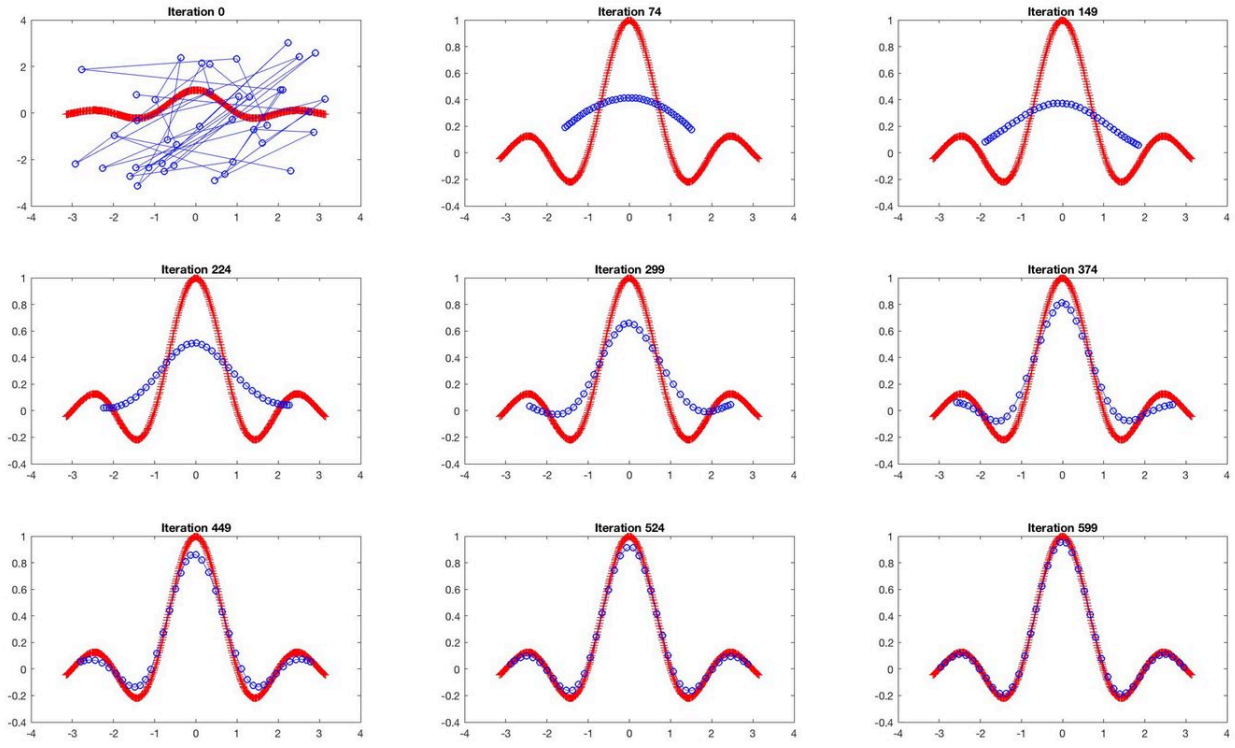
For n-dimensional input space (n=2 in a & b), n=784 in c)) and m (m=1×40 in a), m=8×8 in b), m=10×10 in c)) output neurons:

1. Sampling: choose an input vector x from the training set;
2. Determine winner neuron $i(x)$: $i(x) = \text{argmin}_k ||w_k - x||$ (Euclidean distance)
3. Update the weights of all the neighborhood neurons j of the winner neuron $i(x)$:

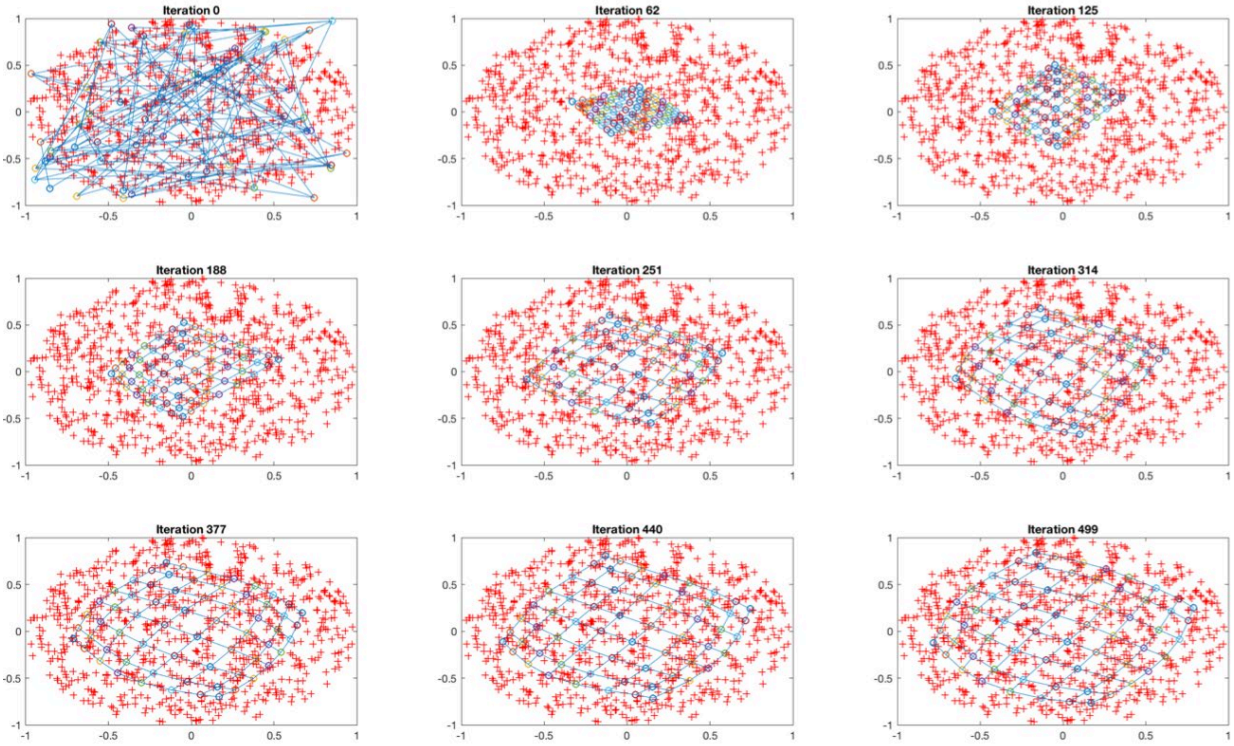
$$w_j(n+1) = w_j(n) + \eta(n) h_{j,i(x)}(n)\big(x - w_j(n)\big), j = 1,2,\dots,m$$

4. If stopping criterion met, STOP; Otherwise, go to (1).

a)



b)



c) Generating the conceptual/semantic map:

- Choose one neuron from the output layer;
- Search the training set and determine which sample is closest to the particular neuron;
- Label the neuron using the class label of the training sample;
- Repeat the above procedure for all output neurons.

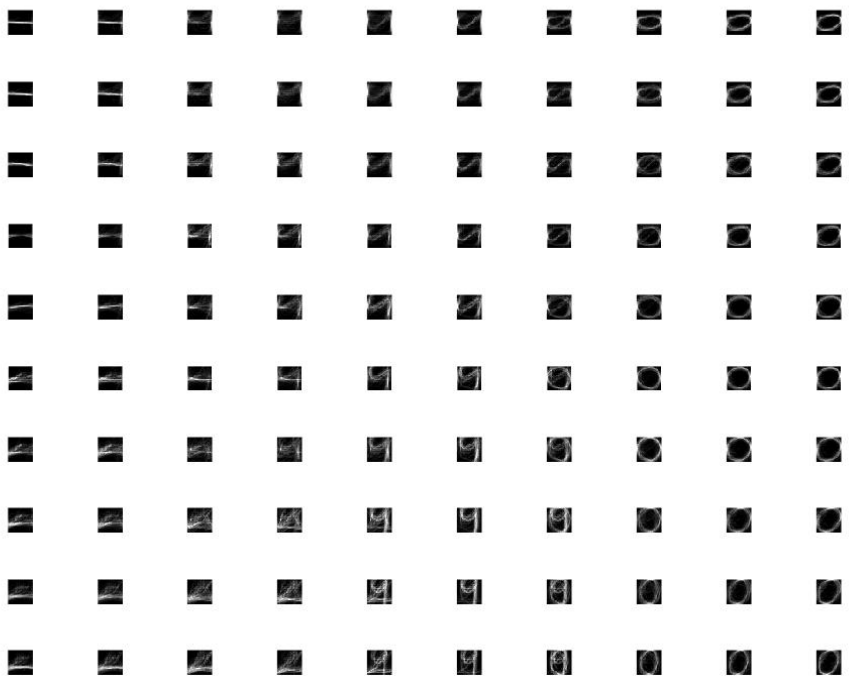
sem_map

10x10 double

	1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	2	2	2	0	0	0
2	1	1	1	1	1	2	2	1	0	0
3	1	1	1	1	1	2	2	2	1	0
4	1	1	1	1	1	2	2	2	1	0
5	1	1	1	1	1	1	1	1	0	0
6	1	1	1	1	1	1	1	1	0	0
7	1	1	1	1	1	1	1	1	0	0
8	1	1	1	1	1	1	1	1	1	0
9	1	1	1	1	1	1	1	1	1	0
10	1	1	1	1	1	1	1	1	1	0

Clustered regions can be seen clearly from this map (class 3 and class 4 are omitted in this demonstration). Digits ‘0’, ‘1’, ‘2’ can be compactly clustered, while several samples ‘1’ are relatively messed with ‘0’, ‘2’. Due to random initialization, the results are different for different runs.

The corresponding weights are visualized in the following figure. There is a clear connection between the weight of a neuron and the label assigned to it. It is especially illuminating to observe the vagueness of neurons at the boundaries between clusters.



Test the SOM:

- Choose one sample form the test set;
- Find out the winner neuron in the SOM which is closest to that particular sample;
- Label the test sample using the label of the winner neuron;
- Repeat the above procedure for all test samples.

Accuracy on training set: 62.83%

Accuracy on test set: 63.33%

Increase the epochs can enhance the average accuracy. Tuning other parameters like the SOM lattice size, learning rate, etc. would also yield different results. You are encouraged to conduct more trials and discuss possible findings.