A0177828L
Liu Xiaohan

EE4305 Fuzzy/Neural Systems for Intelligent Robotics

Mini-project for part I: Neural Networks

All codes are shown in file.

Q1. Rosenbrock's Valley Problem

a).

Let z = f (x, y)

$\frac{\partial z}{\partial x}$ = $400x^3 - 400yx + 2x - 2$ and $\frac{\partial z}{\partial y}$ = $200(y-x^2)$

Let $\frac{\partial z}{\partial x}$ = 0 and $\frac{\partial z}{\partial y}$ = 0, got x = 1 and y = 1.

So, the only saddle point (1, 1) is the global maximum or minimum of f (x, y).
When (x, y) = (1, 1), f (x, y) = 0. However, when (x, y) is any other value, the value of f (x, y) is greater than 0. For example, when (x, y) = (0, 0), f (x, y) = 1 > 0.
Therefore, f (x, y) has a global minimum at (x, y) = (1,1) where f(x, y) = 0

b).

For Steepest (Gradient) descent method, w(k +1) = w(k) −ηg(k).

$$g(k) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix} = \begin{pmatrix} 400x^3 - 400yx + 2x - 2 \\ 200(y - x^2) \end{pmatrix}$$

Set 1 x 10^{-8}as threshold to control stop.
Figures below shows details.

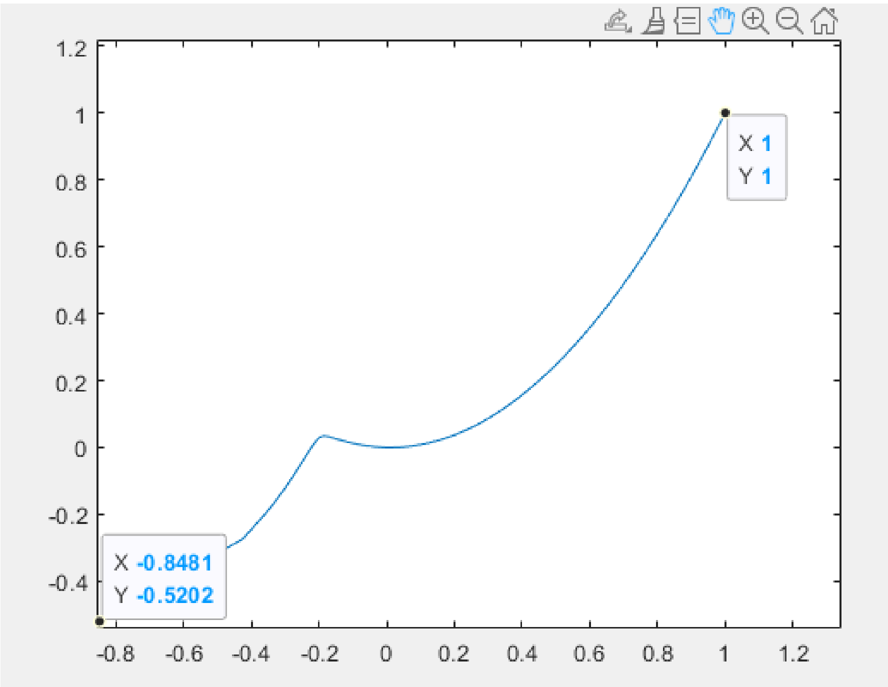


Fig.1 trajectory of (x, y) gradient descent method

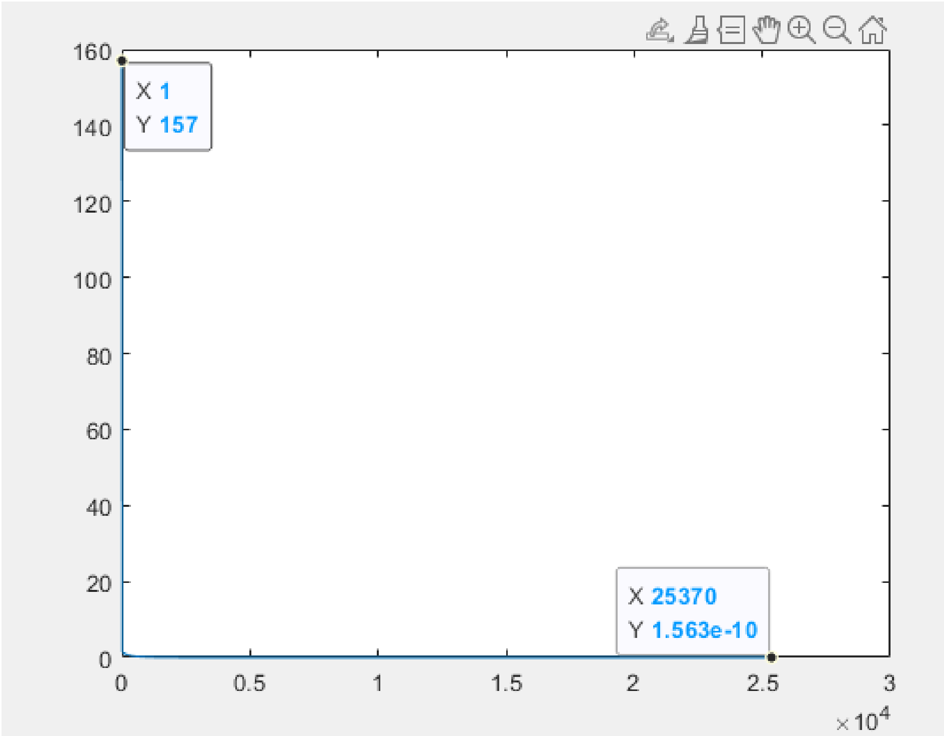


Fig.2 f(x,y) against iterations gradient descent method

When η = 0.001, about 25000 iterations needed to reach threshold and converge to global minimum. (x,y) stops at (1,1) and f(x,y) converge to 0. ($1.563X10^{-10} \approx 0$)

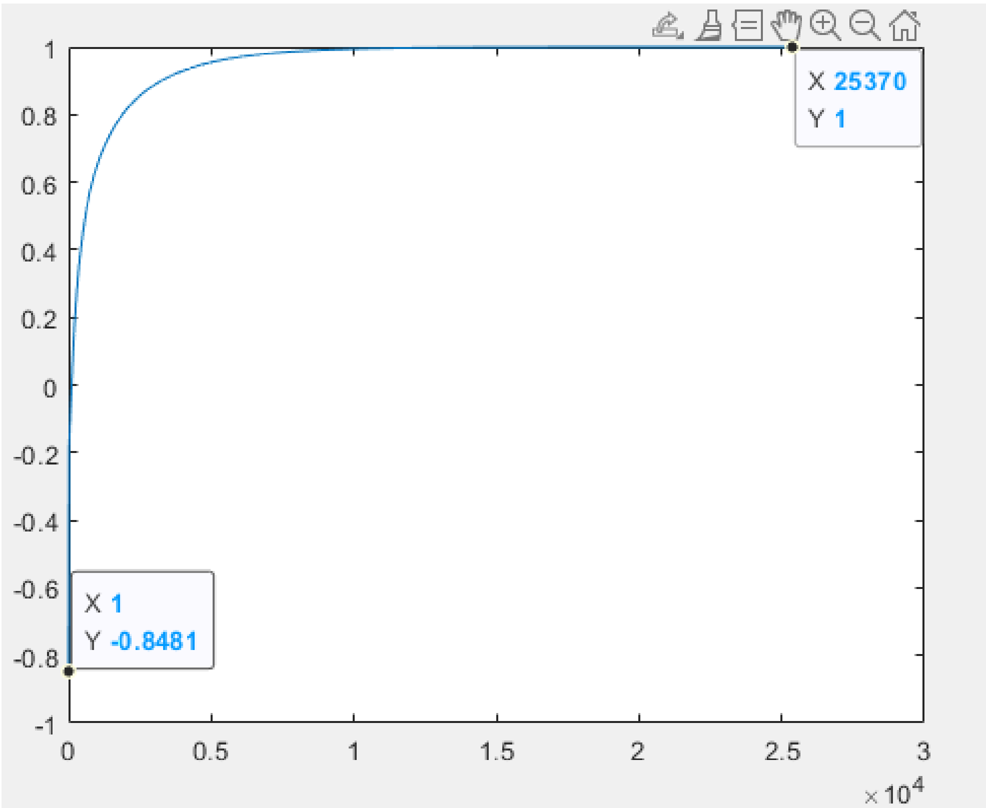


Fig.3 x against iterations gradient descent method

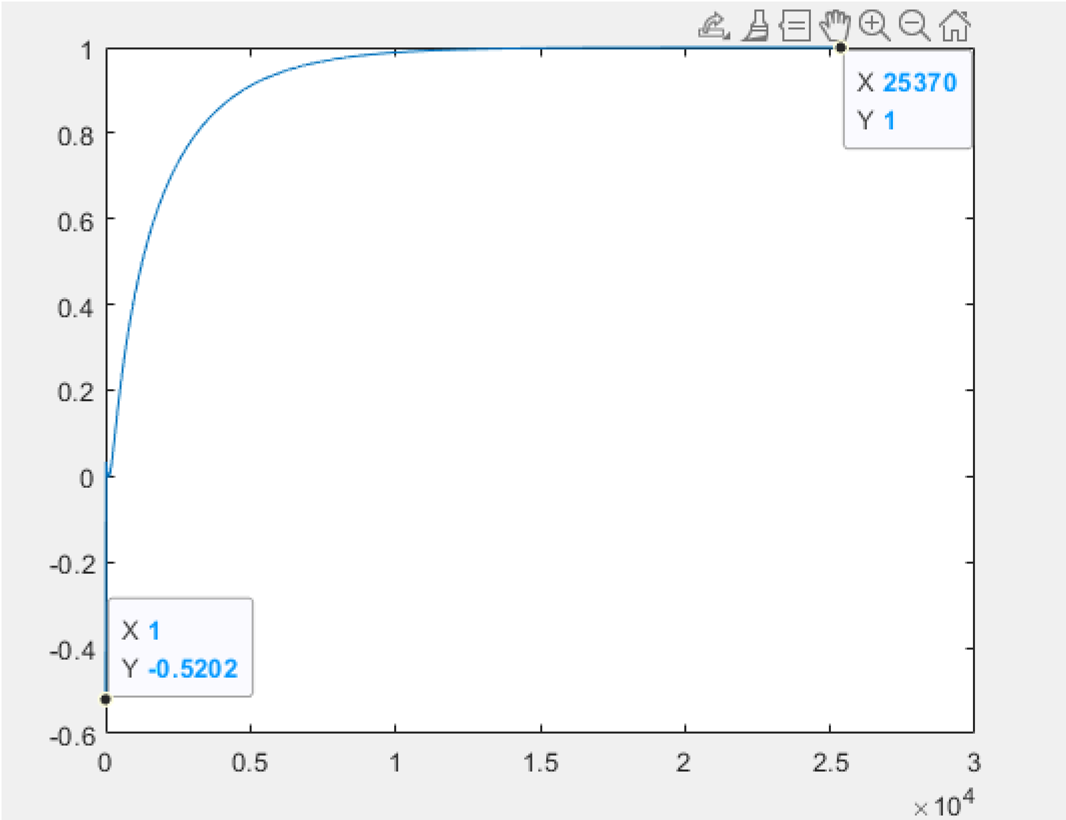


Fig.4 y against iterations gradient descent method

When η = 1, all values show NaN, which means the while loop run infinitely. Because of Taylor's Series, when η is not small enough, the gradient descent algorithm cannot converge to a certain point. η is too big for this situation.

c).

For Newton's method, $\Delta w(n) = -H^{-1}(n)g(n)$

Hessian Matrix: H(n) = $\begin{pmatrix} \frac{\partial^2 f(x,y)}{\partial x^2} & \frac{\partial^2 f(x,y)}{\partial x \partial y} \\ \frac{\partial^2 f(x,y)}{\partial y \partial x} & \frac{\partial^2 f(x,y)}{\partial y^2} \end{pmatrix} = \begin{pmatrix} 1200x^2 - 400y + 2 & -400x \\ -400x & 200 \end{pmatrix}$

Set 1 x 10^{-8}as threshold to control stop.

Figures below shows details.

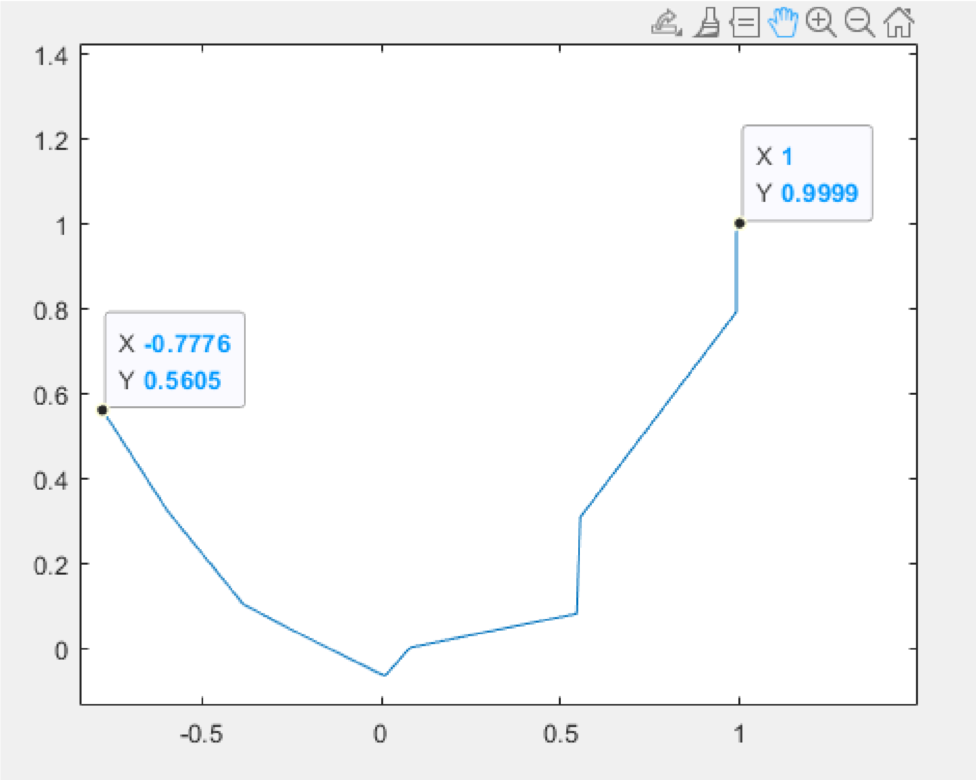


Fig.5 trajectory of (x, y) Newton's method

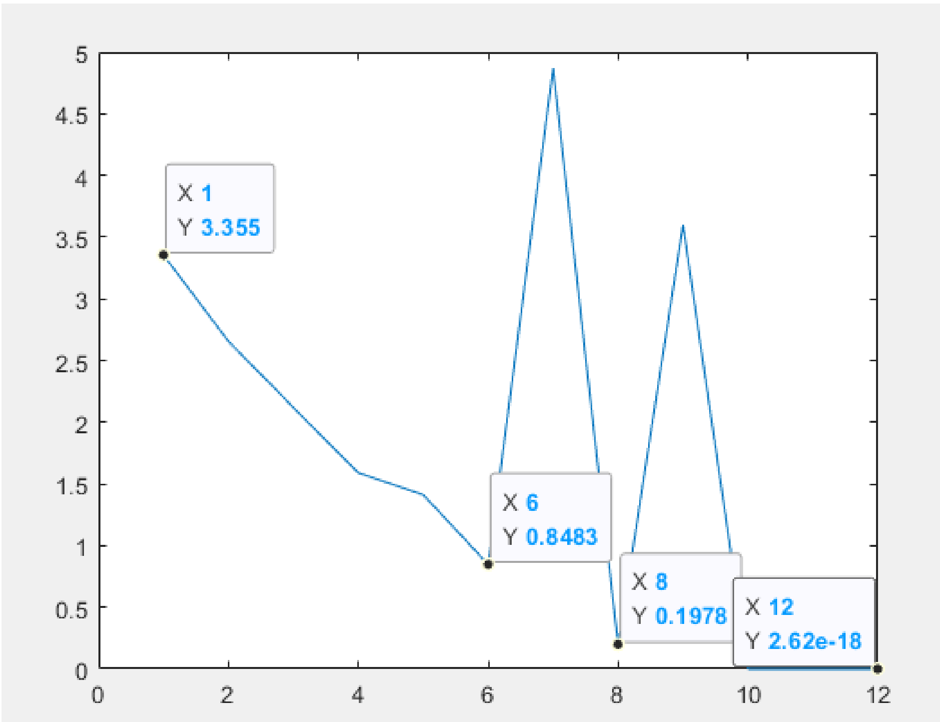


Fig.6 f(x,y) against iterations Newton's method

For Newton's method, only 12 iterations needed to reach global minimum which is much less than gradient descent method. (x,y) stops at (1,1) and f(x,y) reach 0. ($2.62\text{X}10^{-18} \approx 0$)

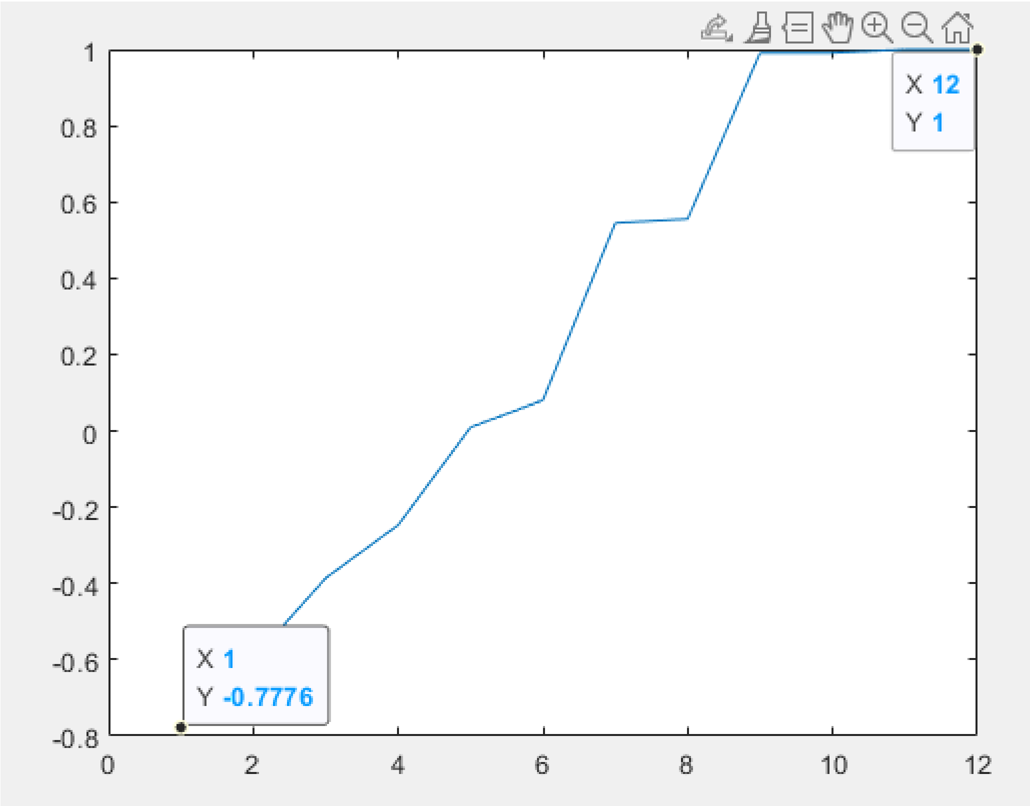


Fig.7 x against iterations Newton's method

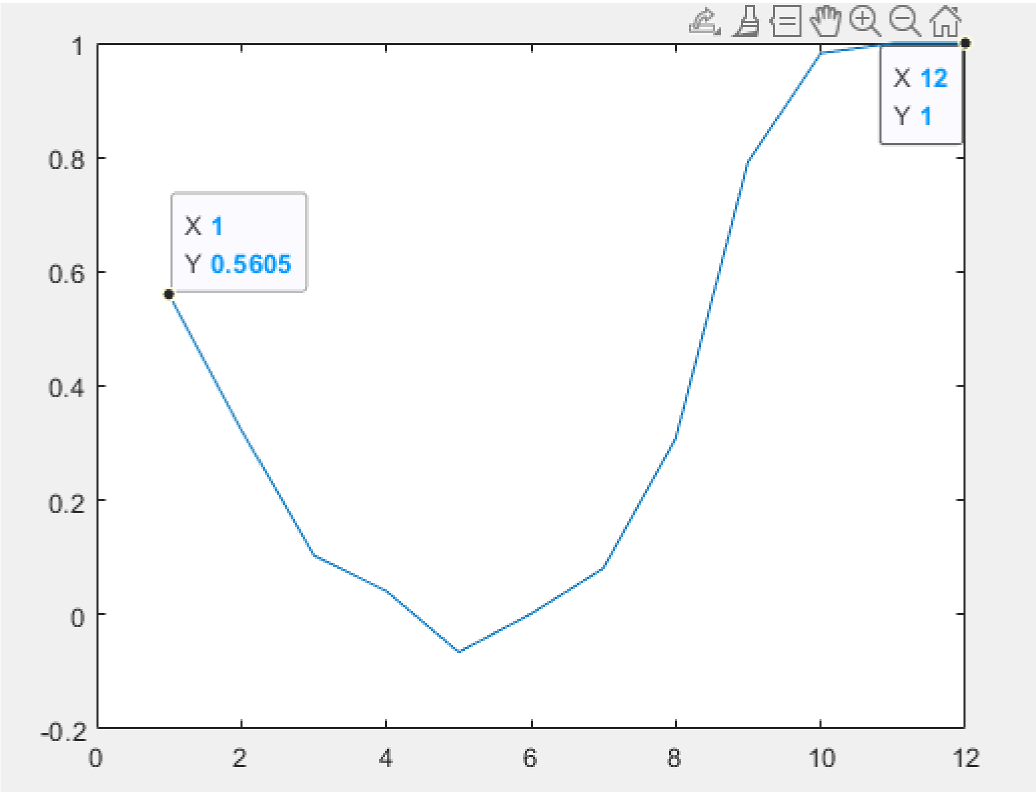


Fig.8 y against iterations Newton's method

Q2. Function Approximation

The plot of objective function is shown as Fig.9 below.

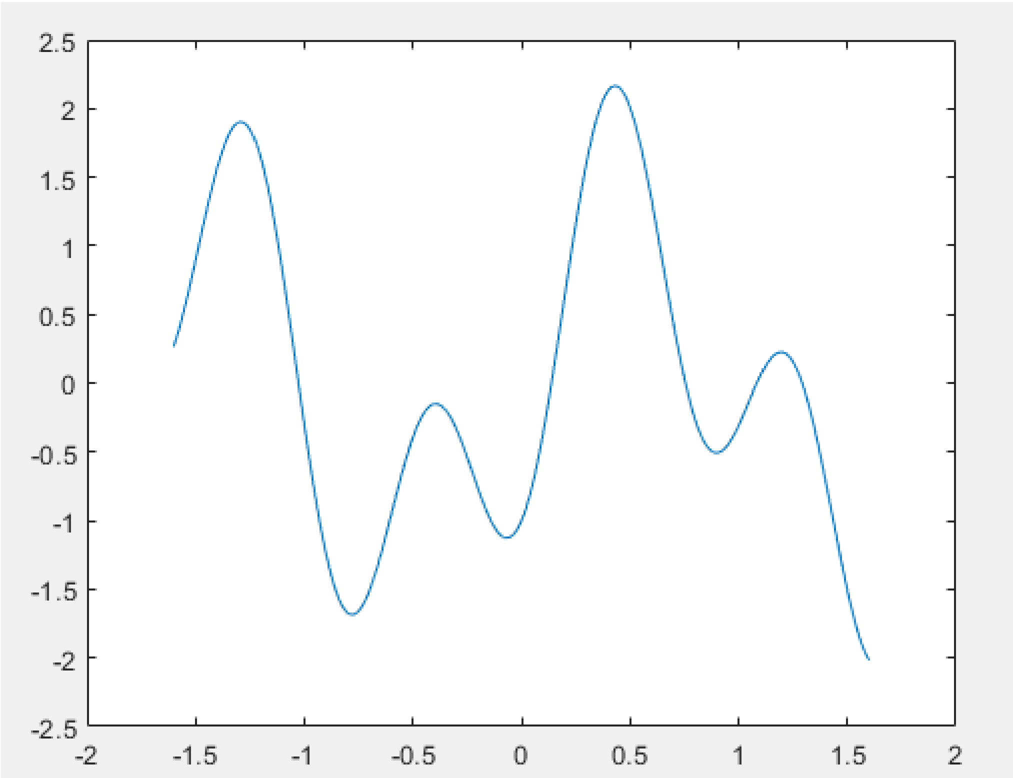


Fig.9 plot of objective function

Based on Guideline in lecture4, P24, there are 8 times line changes in Fig.9. Therefore, at least 8 hidden neurons are needed to construct the basic shape.

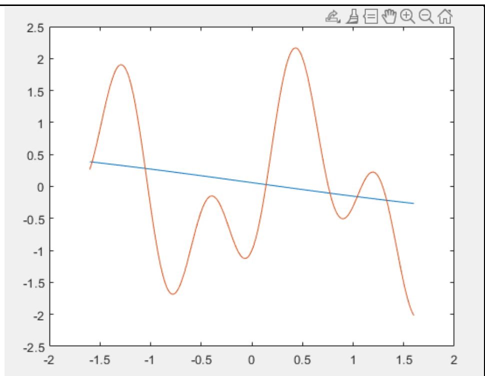
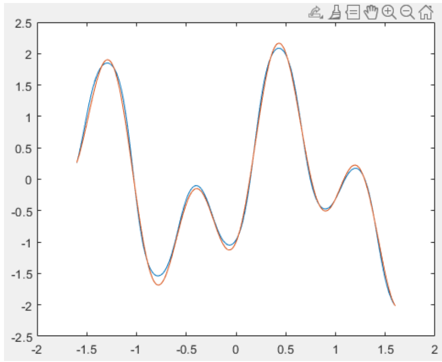
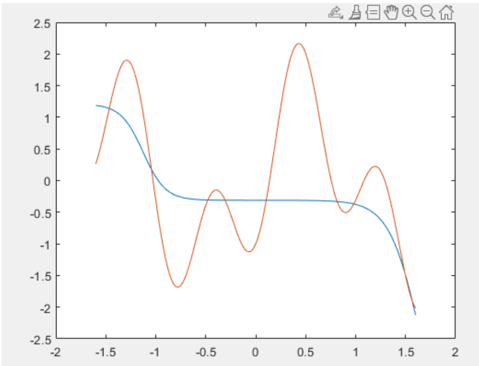
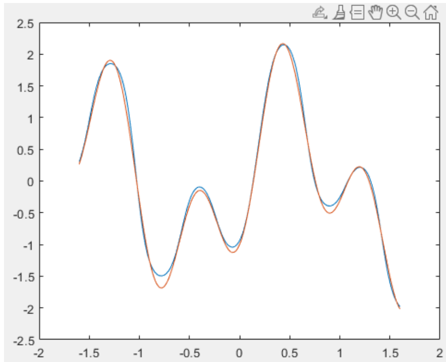
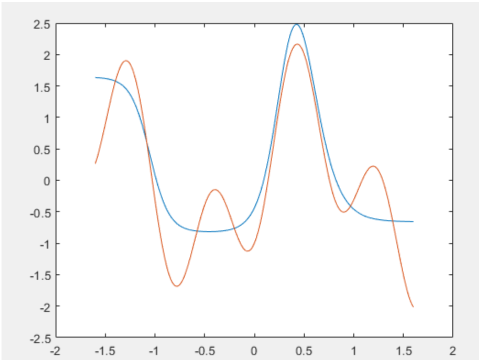
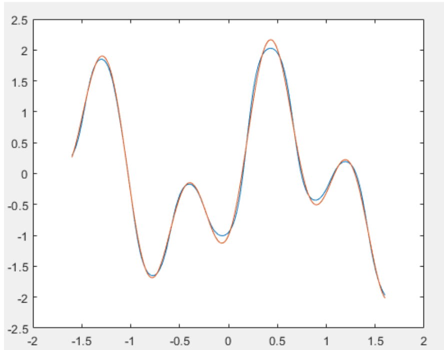
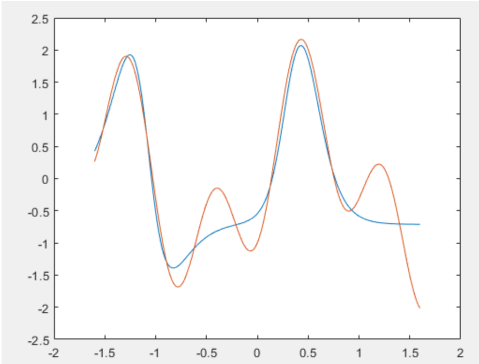
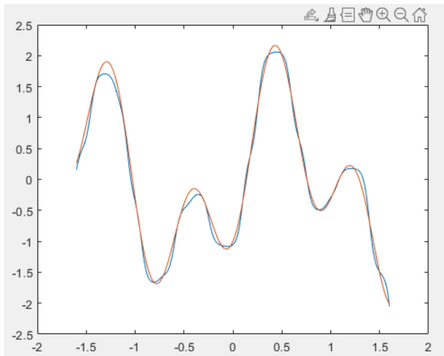
a).

For parameter initial, epochs = 1000 is set. The train function is traingd -- Gradient descent backpropagation is used. About activation functions, based on Lecture4, tansig is chosen in hidden neurons and purelin is chosen in output neurons.

The outputs of the MLP for the test samples after training and the desired outputs plot is shown as Table.1 below.

# of hidden neurons	plot	# of hidden neurons	plot

A0177828L
Liu Xiaohan

1		8	
2		9	
3		10	
4		20	

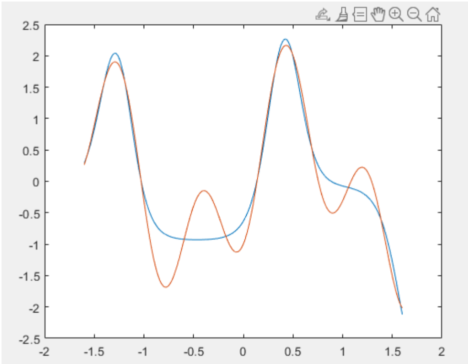
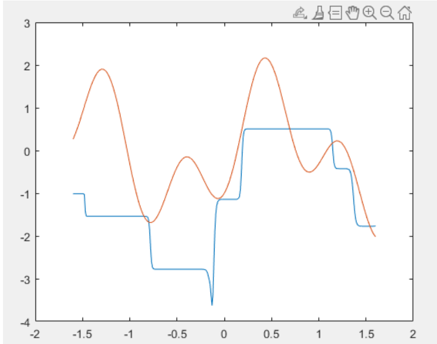
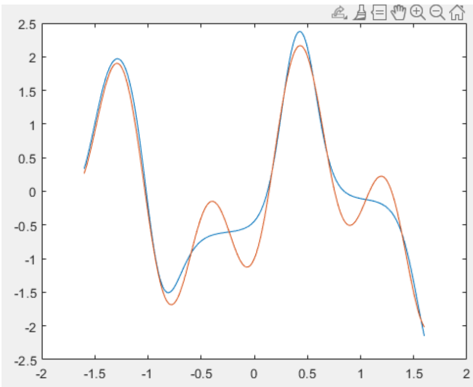
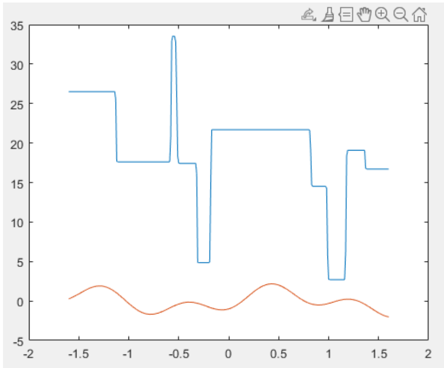
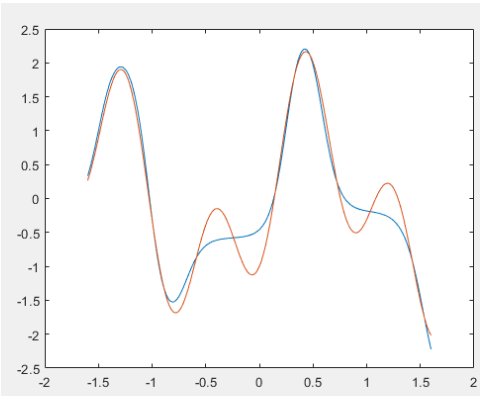
5		50	
6		100	
7			

Table.1 Outputs of the MLP for the test samples after training and the desired outputs plot with different number of hidden neurons (traingd)

- From table 1, when number of hidden neurons < 8, the shape of output one is simpler than desired one and not accurate. Therefore, under-fitting occurs.
- When 8<= number of hidden neurons <=10, the shape of output one is similar to the desired one. Therefore, proper-fitting occurs.
- When number of hidden neurons = 20, 50, 100, more shapes than the original curves appeared. Therefore, over-fitting occurs.

The experiments shows that minimal number of hidden neurons is consistent with the guideline, which is mentioned as before, minimal number of hidden neurons = 8.

When x=-3 and +3, The result is shown as Table.2 below.

Ground truth y = 0.8090 and 0.8090		
# of hidden neurons	Output for x=+3	Output for =-3
1	-0.469955060087000	0.590052734264638
2	-4.69365610992332	1.20880412788973
3	-0.658523992657022	1.64218477553055
4	-0.711484808267674	-0.307762382200991
5	-3.83622651138909	-0.176301622157303
6	-3.09682606942276	-0.313679527109884
7	-2.48425487451409	-0.168532022208418
8	-2.18043773851126	-0.282747210700202
9	-2.09215987464426	-0.000152845414580494
10	-2.13454773113606	0.129957033730232
20	-4.28649099232765	-0.721362134286125
50	-48.2663289450782	581.187456030468
100	-2414.35842114294	1512.53456387184

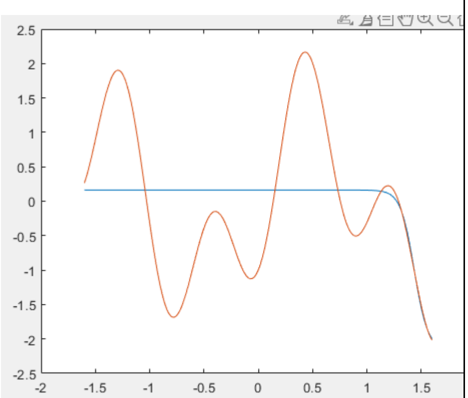
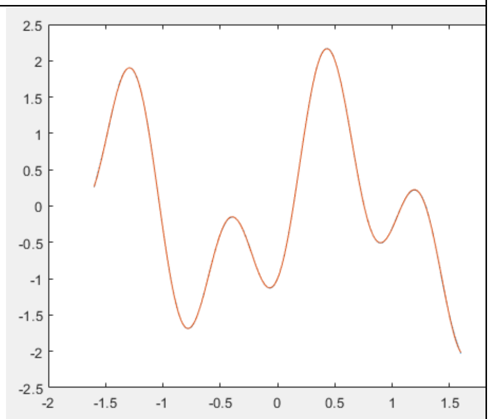
Table2. Result for x=-3 and +3 (traingd)

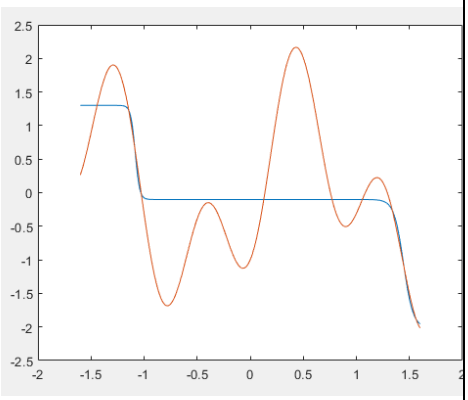
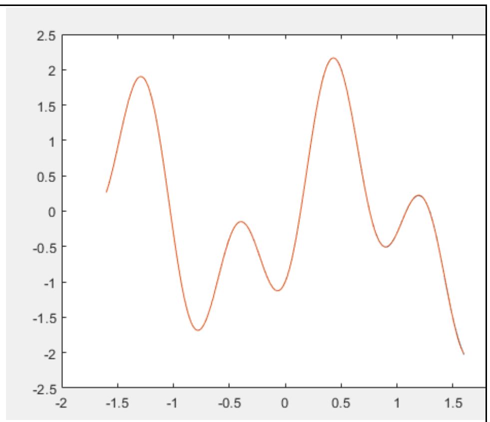
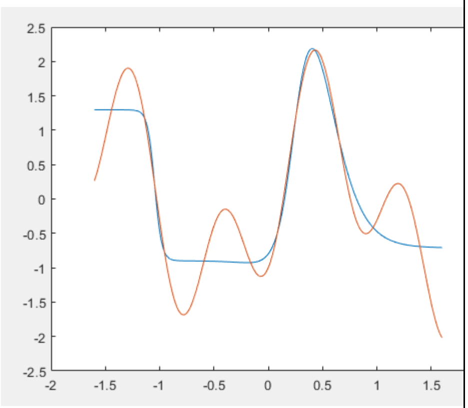
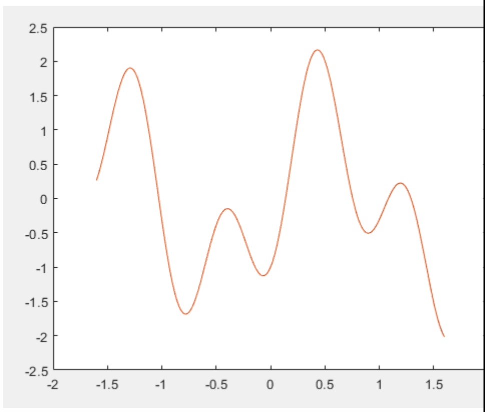
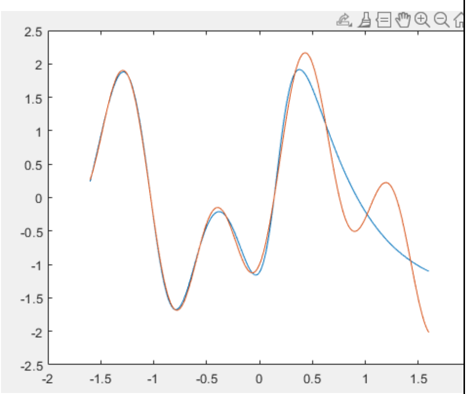
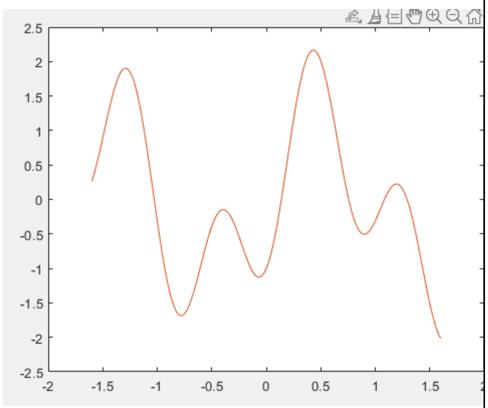
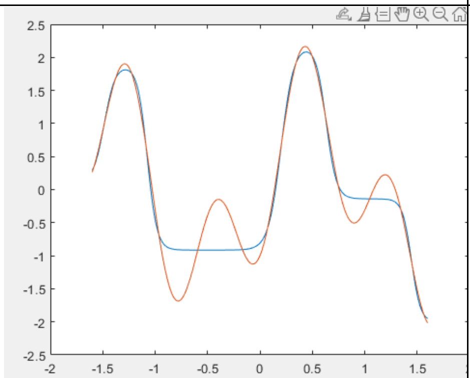
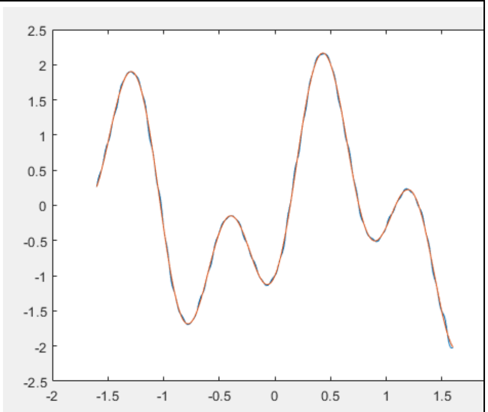
From Table.2, since ground truth = 0.8090, all the results for x=-3 and +3 seems inaccurate. Therefore, the MLP cannot make reasonable predictions outside of the domain of the input limited by the training set.

b).

For parameter initial, upper epochs = 1000 is set. The train function is trainlm -- Levenberg-Marquardt backpropagation. About activation functions, based on Lecture4, tansig is chosen in hidden neurons and purelin is chosen in output neurons.

The outputs of the MLP for the test samples after training and the desired outputs plot is shown as Table.3 below.

# of hidden neurons	plot	# of hidden neurons	plot
1		8	

2		9	
3		10	
4		20	
5		50	

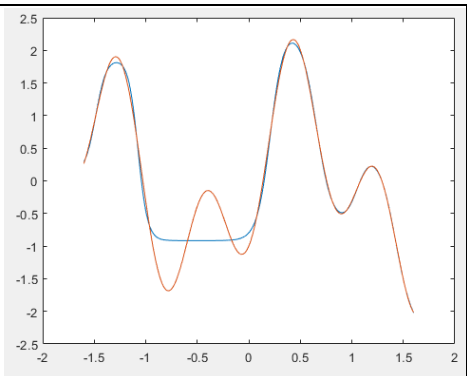
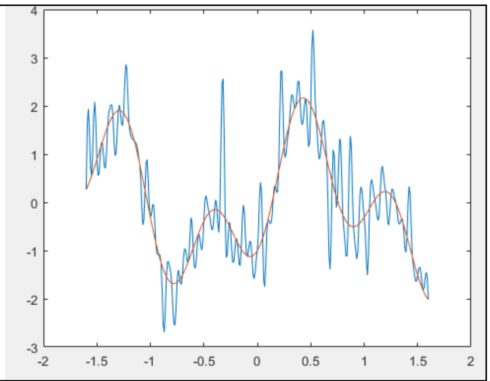
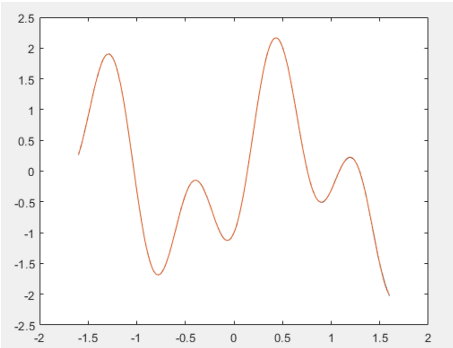
6		100	
7			

Table.3 Outputs of the MLP for the test samples after training and the desired outputs plot with different number of hidden neurons (trainlm)

- From table 3, when number of hidden neurons < 7, the shape of output one is simpler than desired one and not accurate. Therefore, under-fitting occurs.
- When 7<= number of hidden neurons <=20, the shape of output one is similar to the desired one. Therefore, proper-fitting occurs.
- When number of hidden neurons = 50, 100, more shapes than the original curves appeared. Therefore, over-fitting occurs.

The experiments shows that minimal number of hidden neurons is one less than the guideline which is mentioned as before, minimal number of hidden neurons = 7 in this experiment.

When x=-3 and +3, The result is shown as Table.4 below.

Ground truth y = 0.8090 and 0.8090		
# of hidden neurons	Output for x=+3	Output for =-3
1	-0.233030978870586	1.29914070346049
2	-2.01020188577156	1.30042319951081
3	-0.712036469302263	1.29701165834951
4	-2.00280537973991	1.29698786778219
5	-2.33809892641387	-0.157138368672323
6	-2.09868877099677	0.115900588427756
7	-2.57466769617229	2.42303897213398
8	-2.40174317444111	-0.475986028930376

9	-2.40355431026055	-0.357507696107118
10	-2.25303172591565	-0.463087484223442
20	-2.19660244300982	-0.347491077126364
50	-2.14378220545705	0.226455825137963
100	-2.10936345271537	0.150323275242922

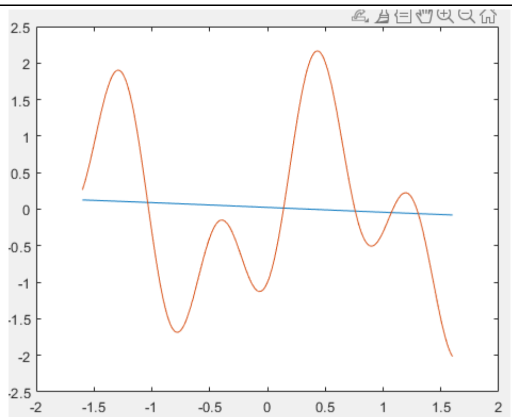
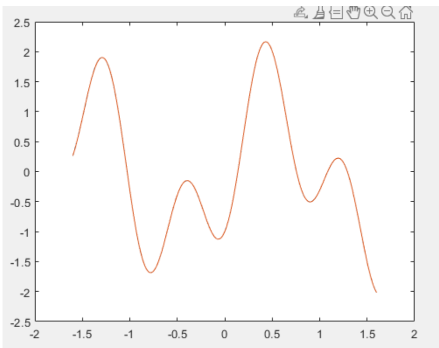
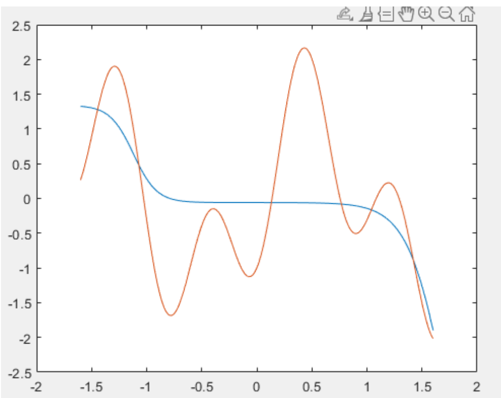
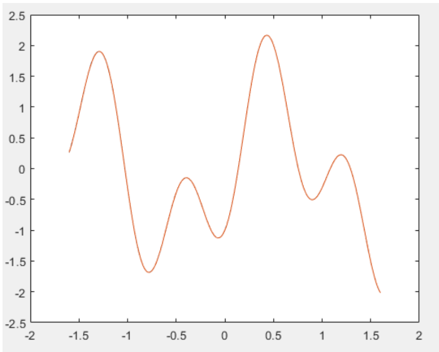
Table4. Result for x=-3 and +3(trainlm)

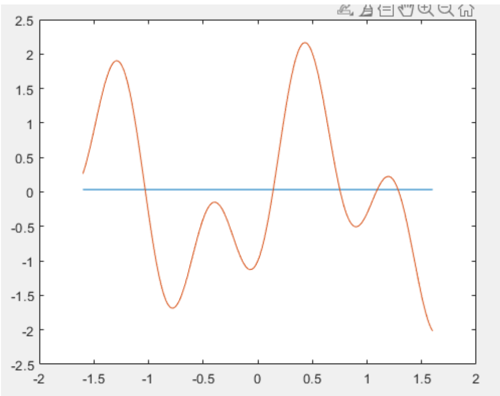
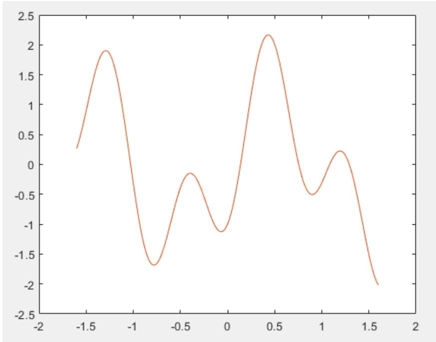
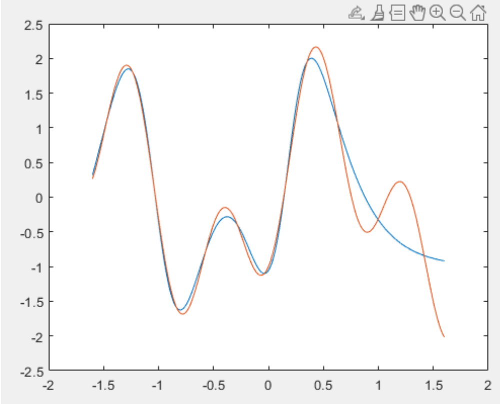
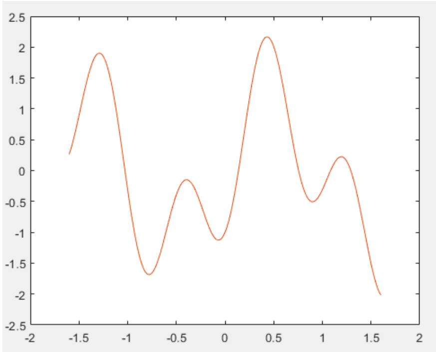
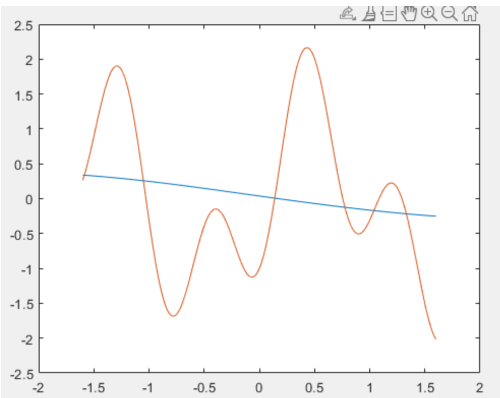
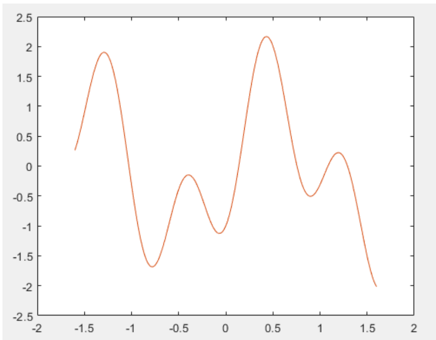
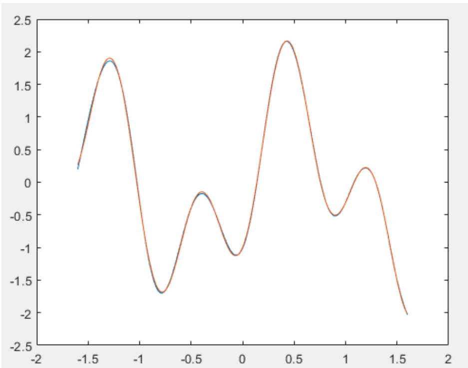
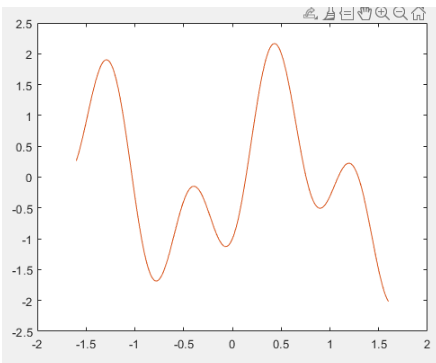
From Table.4, since ground truth = 0.8090, all the results for x=-3 and +3 seems inaccurate. But some point nearer to the true one than traingd. Therefore, the MLP cannot make reasonable predictions outside of the domain of the input limited by the training set.

c).

For parameter initial, upper epochs = 1000 is set. The train function is trainbr -- Bayesian Regulation backpropagation About activation functions, based on Lecture4, tansig is chosen in hidden neurons and purelin is chosen in output neurons.

The outputs of the MLP for the test samples after training and the desired outputs plot is shown as Table.5 below.

<table>
<tr><th># of hidden neurons</th><th>plot</th><th># of hidden neurons</th><th>plot</th></tr>
<tr><td>1</td><td></td><td>8</td><td></td></tr>
<tr><td>2</td><td></td><td>9</td><td></td></tr>
</table>

3		10	
4		20	
5		50	
6		100	

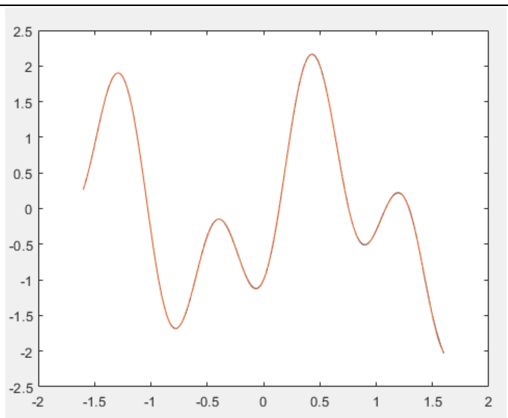
7			

Table.5 Outputs of the MLP for the test samples after training and the desired outputs plot with different number of hidden neurons (trainbr)

- From table 5, when number of hidden neurons < 6, the shape of output one is simpler than desired one and not accurate. Therefore, under-fitting occurs.
- When 6<= number of hidden neurons, the shape of output one is similar to the desired one. Therefore, proper-fitting occurs.
- When 8<= number of hidden neurons, the line almost perfectly approximate.

The experiments shows that minimal number of hidden neurons is two less than the guideline which is mentioned as before, minimal number of hidden neurons = 6 in this experiment. But perfectly approximate when minimal number of hidden neurons = 8.

When x=-3 and +3, The result is shown as Table.6 below.

Ground truth y = 0.8090 and 0.8090		
# of hidden neurons	Output for x=+3	Output for =-3
1	-0.153983393244907	0.198428623372432
2	-5.66537860341988	1.34076852608285
3	0.0350991872749660	0.0346905892333012
4	-1.00334746959124	-2.08630215679945
5	-0.280245543924274	0.410294671109673
6	-3.16421554793635	-3.49016199472266
7	-3.17690806683954	-0.314729987643747
8	11.0654077836396	-0.515633262553104
9	5.18794816187557	-0.480877559889672
10	3.13308096789287	1.68178036066801
20	2.43125847061462	1.37291725120011
50	-1.05035507890005	-0.0571989488300253
100	-0.747072557857184	0.0892769291344520

Table.6 Result for x=-3 and +3(trainbr)

From Table.6, since ground truth = 0.8090, all the results for x=-3 and +3 seems inaccurate. But some point nearer to the true one than traingd and trainlm. Therefore, the MLP cannot make reasonable predictions outside of the domain of the input limited by the training set.

Q3. Image Classification

For accuracy calculation, when predict result >= 0.5, it is regarded as 1, when predict result < 0.5, it is regarded as 0.

a).

To realize Rosenblatt's perceptron algorithm, net = perceptron is using here. Different number of epochs give different accuracy which is shown as Table.7 below.

Number of epochs	Training Accuracy	Testing Accuracy
1	0.577	0.550
10	0.737	0.600
50	0.819	0.580
100	0.759	0.590
200	0.957	0.540
500	1	0.580

Table.7 Train and Test accuracy in Rosenblatt's perceptron algorithm

From Table.7, more epochs will increase training accuracy more but not testing accuracy. And testing accuracy is poor in Table.7, which means Rosenblatt's perceptron is too simple for this question. One disadvantage of single-layer perceptron is that they can only classify linearly separable datasets. Multi layers perceptron might perform better in this problem.

b).

To set epochs as question3a, after subtracting the mean value from each image and dividing each image by the standard deviation, the train and test accuracy is shown as Table.8 below.

Number of epochs	Train Accuracy	Test Accuracy
1	0.703	0.510
10	0.838	0.620
50	0.937	0.610
100	0.979	0.580
200	0.980	0.550
500	1	0.560

Table.8 Train and Test accuracy in Rosenblatt's perceptron algorithm after process data

Compare Table.7 to Table.8, training accuracy is higher than before. Except when epochs = 1, testing accuracy is slightly increasing or similar. After reducing the influence of global mean and variance, the data becomes more concentrated and stable, and the influence of some outlier data is reduced. The performance of the model when the number of epochs is small is more stable than before, but the improvement in testing accuracy is not significant. A multi-layer perceptual model might be more suitable for this problem.

c).

For parameter initial, upper epochs = 1000 is set. The train function is traincgb -- Conjugate gradient backpropagation with Powell-Beale restarts, which is chosen based on *Choose a Multilayer Neural Network Training Function - MATLAB & Simulinkbased* cancer data set which is a pattern recognition problem. on Lecture4, tansig is chosen in hidden neurons and logsig is chosen in output neurons. Different size of hidden layer gives different accuracy which is shown as Table.9 below.

Size of hidden layer	Train Accuracy	Test Accuracy
10	1	0.650
20	1	0.680
30	1	0.680
40	1	0.710
50	1	0.680
100	1	0.740
150	1	0.760
200	1	0.750

Table.9 Train and Test accuracy in MLP after process data (batch)

Compared to Table.8, the performance in testing data is much higher than before and performance of training data is much more stabler and better than before. In this question, MLP in batch mode performant better than single layer perceptron.

d).

For parameter initial, epochs = 200 is set, because this algorithm always got result within 200 epochs in batch mode. The train function is traincgb -- Conjugate gradient backpropagation with Powell-Beale restarts, which is chosen based on *Choose a Multilayer Neural Network Training Function - MATLAB & Simulinkbased* cancer data set which is a pattern recognition problem. on Lecture4, tansig is chosen in hidden neurons and logsig is chosen in output neurons. Since processing data gives better result, data processing applies in question c and d as well. Different size of hidden layer gives different accuracy which is shown as Table.10 below.

Size of hidden layer	Train Accuracy	Test Accuracy
10	1	0.720
20	1	0.710
30	1	0.730
40	1	0.700
50	1	0.710
100	1	0.750
150	1	0.770
200	1	0.770

Table.10 Train and Test accuracy in MLP after process data (sequential)

Compared to single layer perceptron, MLP in sequential mode costs much more time for train. And speed of MLP in sequential mode is much lower than batch mode. Compared to Table.9, testing accuracy increase slightly. For this question, considering both speed and performance, MLP in batch mode is more suitable than sequential mode. However, if aiming for higher accuracy, sequential mode is more suitable.

e).

- Image background clean-up

Although the pictures in the dataset have been cut according to the objects, for 32x32 images, the background of different colors and textures occupies a large part of the picture, which will interfere with the performance of the algorithm. If boundary cutting is used to separate out accurate objects and process the background into a uniform color, it will greatly reduce the interference of the background to the algorithm.

- Consider RGB Image instead of Grey-level Image

Since only Grey-level image is considered here, some coloring features lose in the processing. If involving the RGB information, most animals may not have bright colors, which can be easily distinguished with some manmade objects.

- Use PCA to reduce number of features

Since 1024 features is too much for training. PCA may help to reduce number of features without losing information to improve performance.