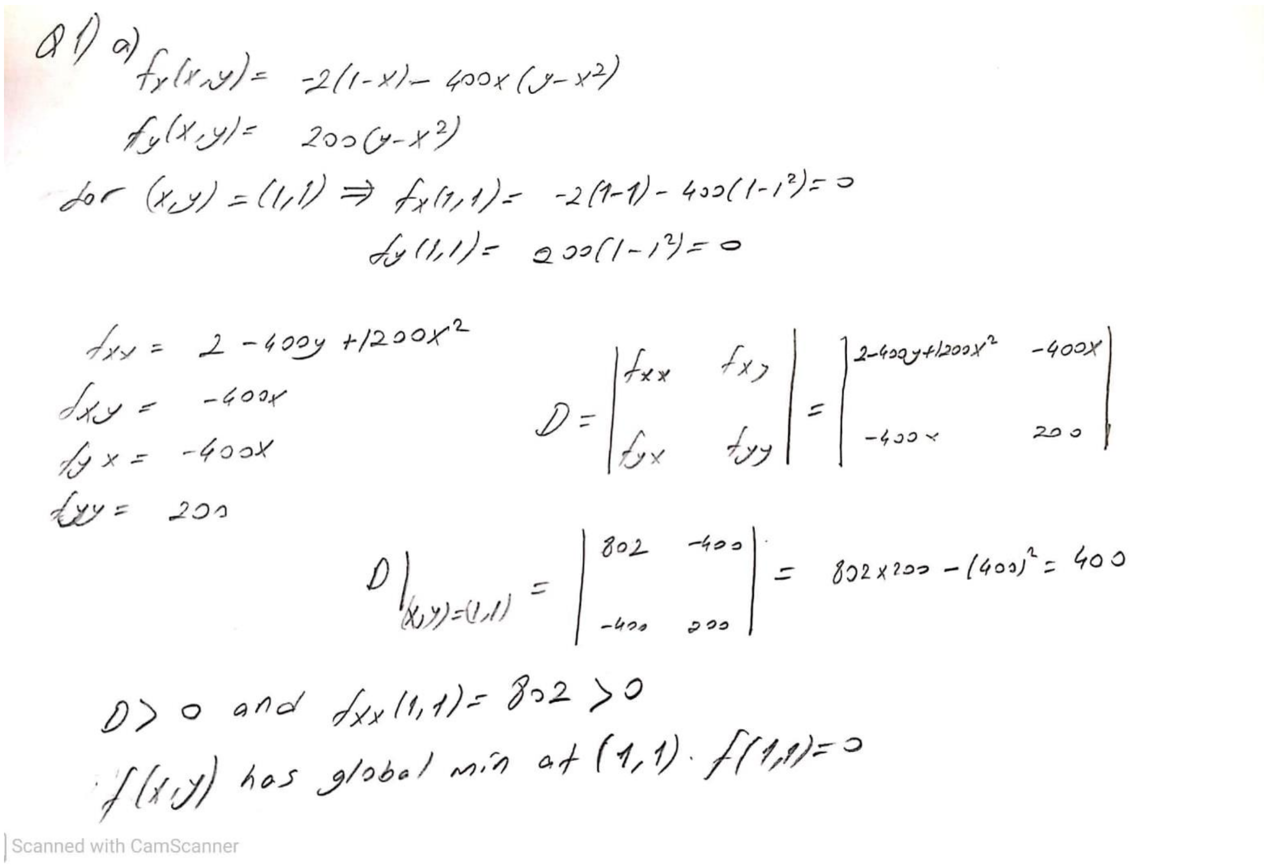
EE5904 NEURAL NETWORKS WH2

Rosenbrock's Valley Problem

Global Minimum at (1,1) Proof



Q1) a) $f_x(x,y) = -2(1-x) - 400x(y-x^2)$

$f_y(x,y) = 200(y-x^2)$

for $(x,y) = (1,1) \Rightarrow f_x(1,1) = -2(1-1) - 400(1-1^2) = 0$

$f_y(1,1) = 200(1-1^2) = 0$

$f_{xx} = 2 - 400y + 1200x^2$

$f_{xy} = -400x$

$f_{yx} = -400x$

$f_{yy} = 200$

$$D = \begin{vmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{vmatrix} = \begin{vmatrix} 2-400y+1200x^2 & -400x \\ -400x & 200 \end{vmatrix}$$

$$D\Big|_{(x,y)=(1,1)} = \begin{vmatrix} 802 & -400 \\ -400 & 200 \end{vmatrix} = 802 \times 200 - (400)^2 = 400$$

$D > 0$ and $f_{xx}(1,1) = 802 > 0$

$f(x,y)$ has global min at $(1,1)$. $f(1,1) = 0$

Steepest Gradient Descent Algorithm to Approximate $f(x,y) = (1-x)^2 + 100(y-x^2)^2$

With learning rate 0.001, gradient descent algorithm worked and function value converged to the global minimum value with the criteria function value < 0.0001 in 8592 iterations. Figure 1 shows the trajectory of (x,y) and the function value versus iterations.

Console Output

```
-----Gradient Descent-----
Randomly initialized (x,y) = (-0.095,-0.394)
Learning Rate = 0.00100
Minimum value for gradient descent is achieved in 8592 iterations with the error value 0.0000999
Final (x,y) = (0.99001,0.98008)
```

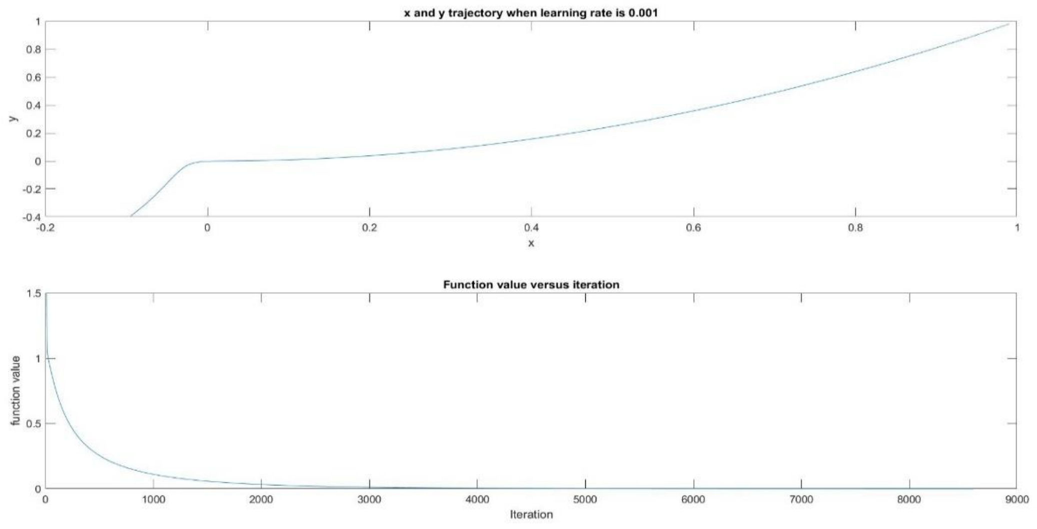
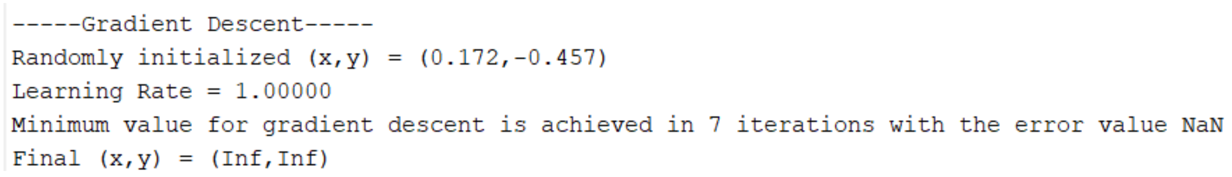


Figure 1 Part B1

With learning rate 1.0 , gradient descent algorithm is not working and function value diverged to the infinity. Figure Part A2 shows the trajectory of (x,y) and the function value versus iterations.

Console Output



```
-----Gradient Descent-----
Randomly initialized (x,y) = (0.172,-0.457)
Learning Rate = 1.00000
Minimum value for gradient descent is achieved in 7 iterations with the error value NaN
Final (x,y) = (Inf,Inf)
```

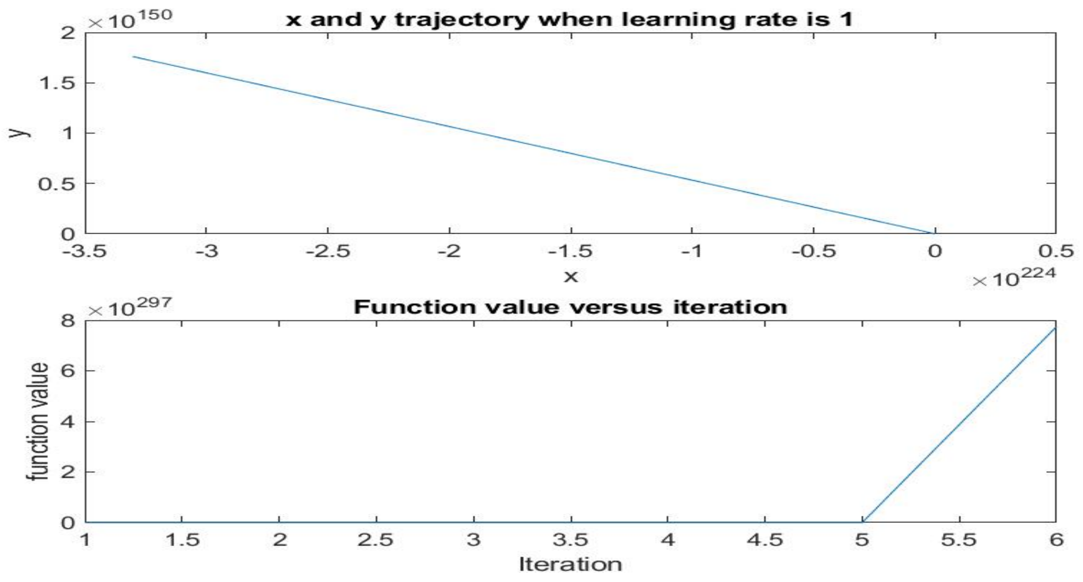


Figure 2 Part B2

Newtons Method to Approximate $f(x,y) = (1-x)^2 + 100(y-x^2)^2$

Refer to Figure 3, function value converged the global minimum value in 6 iterations using Netwon's method. This is much faster than the gradient descent algorithm which takes 8592 iterations to achieve the same error rate.

```
----Newton's Method-----
Minimum value for Newton's method is achieved in 6 iterations with the error value 0.0000000
Final (x,y) = (1.00000,1.00000)
```

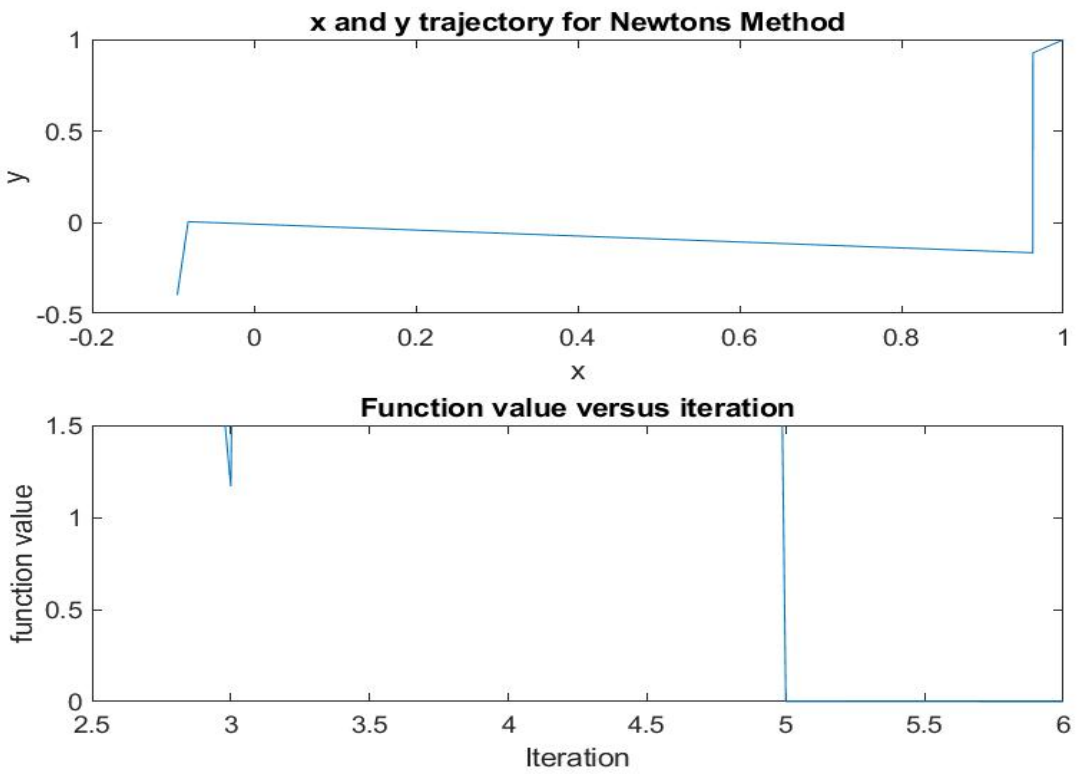


Figure 3 Part C

MLP to approximate $y = 1.2sin(\pi x) - cos(2.4\pi x)$ $for\ x \in [-1.6, 1.6]$

a) MLP in sequential mode with different 1-n-1 (n= 1,2,…,10,20,50, 100).

As a training function, gradient descent ('traingd') is used to implement the function approximation. As a default, 'tansig' activation function is used in the first layer and 'purelin' activation function is used in the output layer. In the Figure 4, subplot at the top shows the plot of network and desired function in the range (-1.6,1.6). The one at the buttom shows the same in the range (-3,3). As deduced from Figure 4, outside of the trained range, network cannot mimic the behaviour of desired function. Since the weights are fixed, the output of the first layer becomes either close to 1 or -1 because of 'tansig' activation function. The output of the first layer is feeded in to second layer which has 'purelin' activation function. The weighted sum of inputs of second layer alternates between two almost fixed number outside of the range because the input of second layer is either close to -1 or 1. This explains the saturation around 0 and -2 in the figures at the outside of the trained region.

Refer to Figure 4, there are eight linear parts of the function from x= -1.6 to x=1.6. As learned in the course, an MLP with eight neurons in 1 hidden layer is required to approximate the function without under-fitting or over-fitting. Figure 4 shows that eight neurons is enough to approximate the function in the given interval. Less than eight number of neurons result in under-fitting. As the number of neurons increases from eight to ten, the occurrence of overfitting can be observed from Figure 4. Twenty and fifty neurons results in more increase in the overfitting.

Console Output showing the time spent to train the network

```
LayerNum: 50 Epochs: 1000
Elapsed time is 1008.620618 seconds.
```

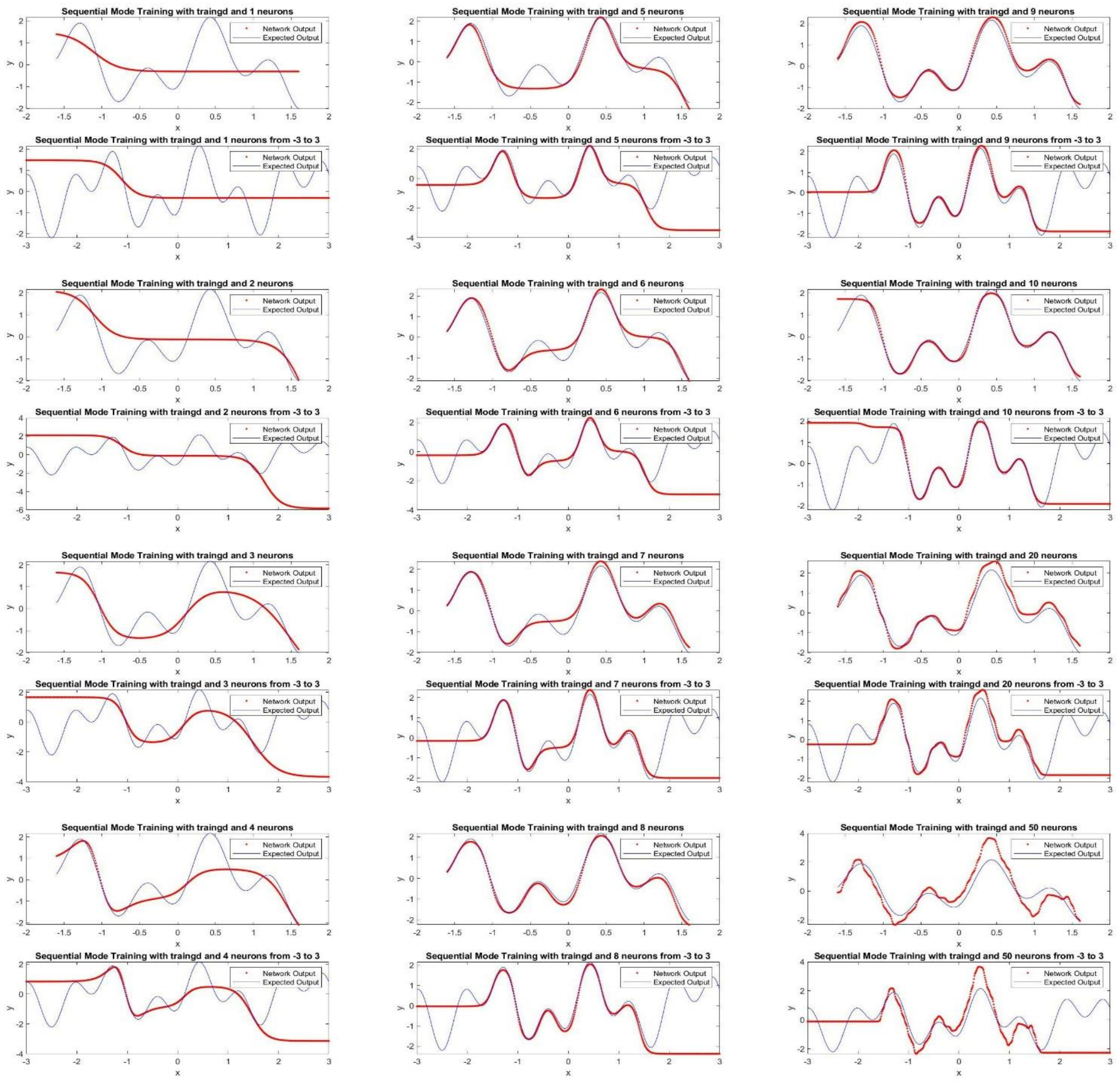
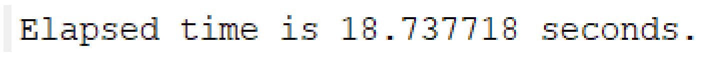


Figure 4 Q2 Part A

b) Batch mode with trainlm algorithm

Console Output showing total training time:



```
Elapsed time is 18.737718 seconds.
```

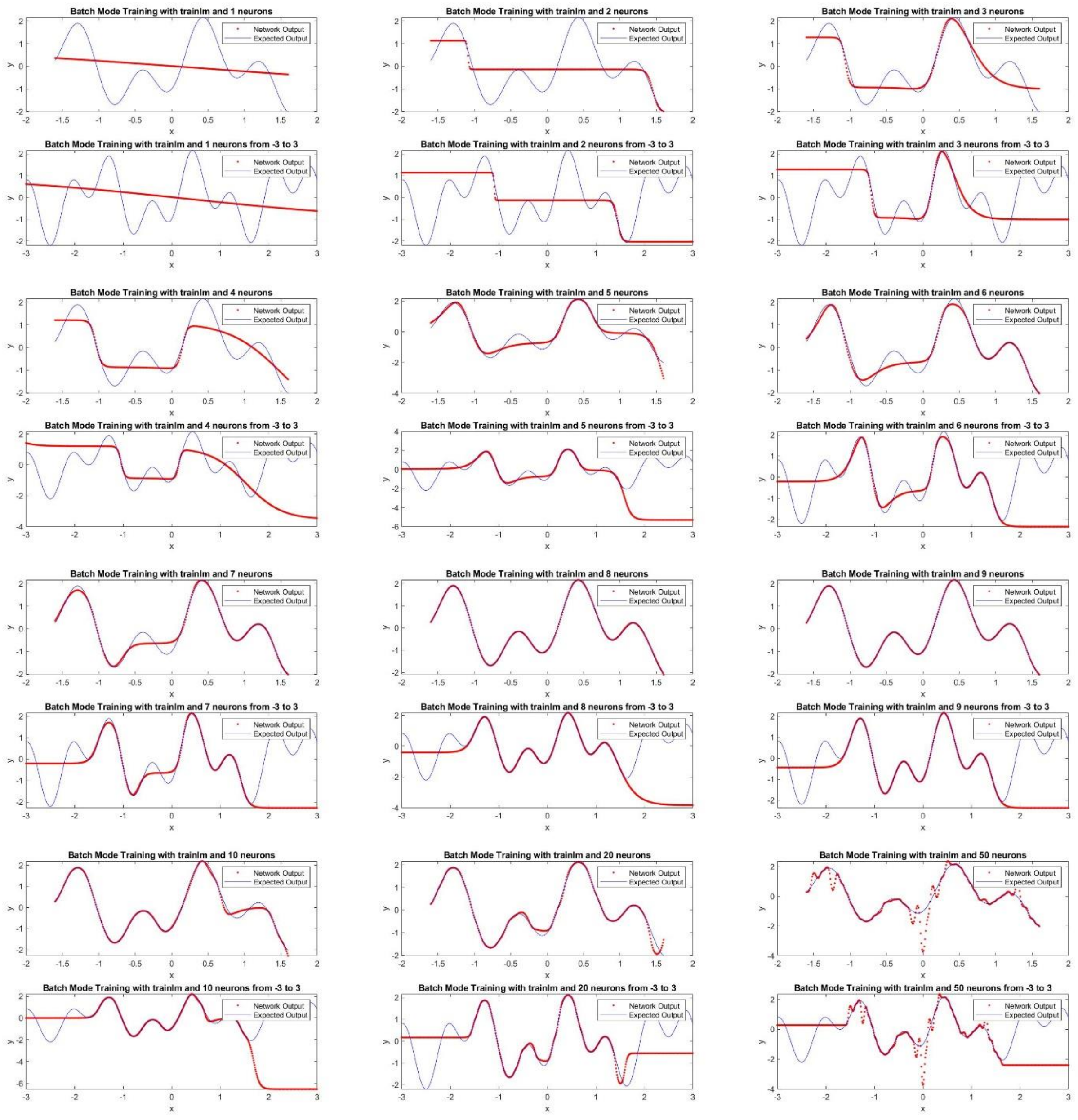


Figure 5 Part B

As in part A, the overfitting and underfitting occurs obviously when the number of neurons are not eight or nine from Figure 5. The performance of batch mode seems better for eight neurons compared to sequential learning. It updates the weights at the end of each epoch; however, the weights are updated in each sample. Therefore, the weight update of batch learning is more reasonable since it updates considering average error of total samples while sequential learning updates its weights based on individual sample error. However, there is possibility that batch learning can be trapped in local minimas of loss function then it may not be able to work. For this example, it worked really well. It took 1008 seconds to complete Part B. It took 18.73 seconds for batch learning which is significantly faster than sequential learning. It cannot approximate the function in the outside of limited range of samples.

c) Batch mode with trainbr algorithm

Figure 6 shows that at least six neurons is enough to approximate given function in the interval x= -1.6 to x= 1.6 using regularization with the 'trainbr' training function. When the number of neurons increase toward fifty, the regularization prevents overfitting and the results are proper fitting.

Outside of the range of samples, this model cannot predict the function as well.

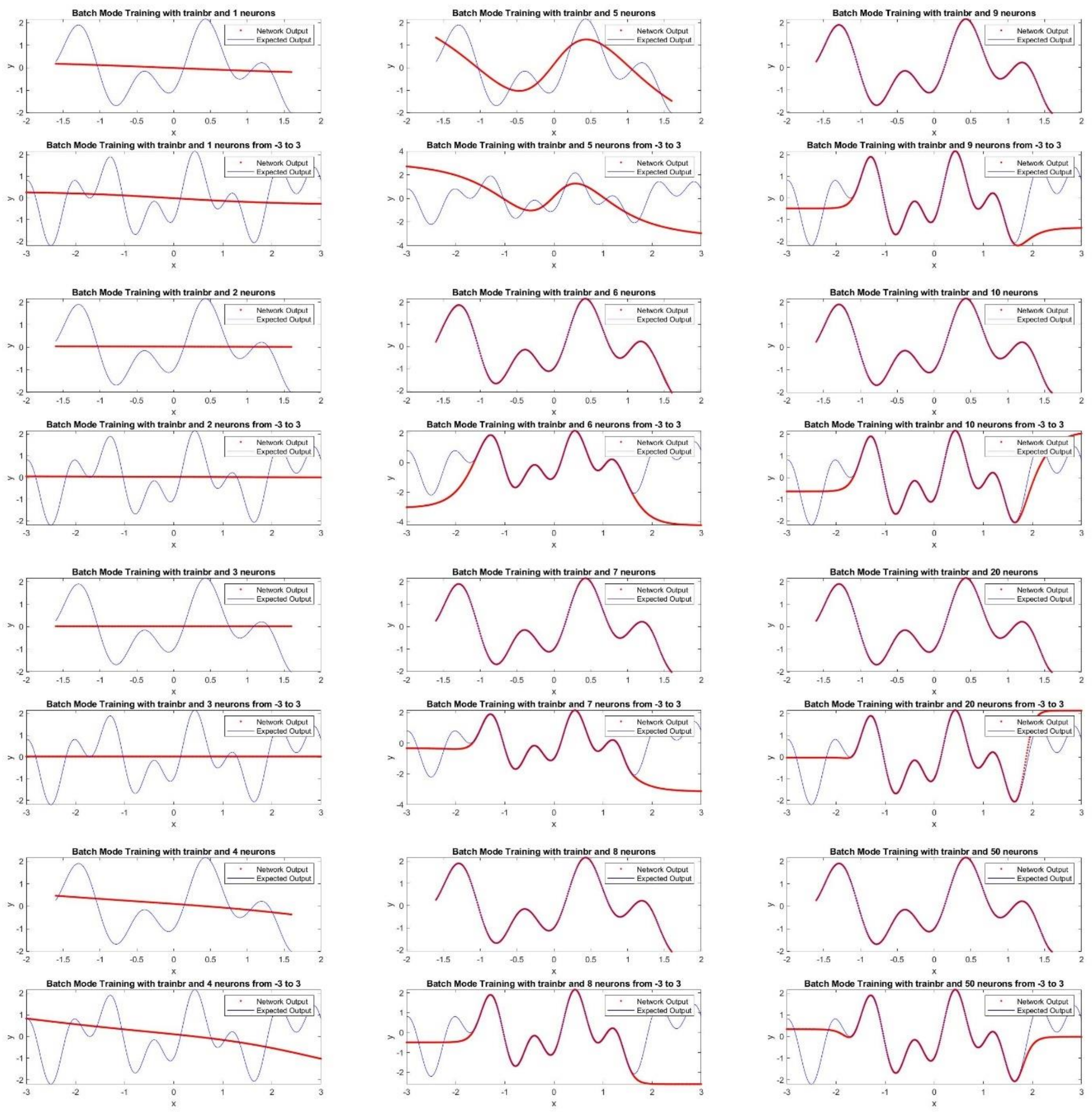


Figure 6 Part

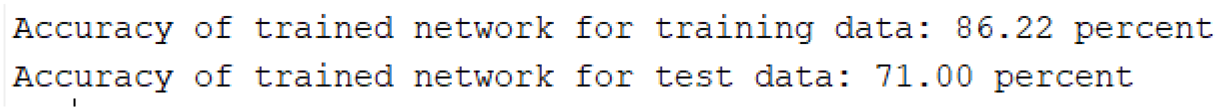
Image Classification

Matriculation number: A0248629N

Mod(29,3) = 2 -> Group 2 = Deer vs Ship

Part A

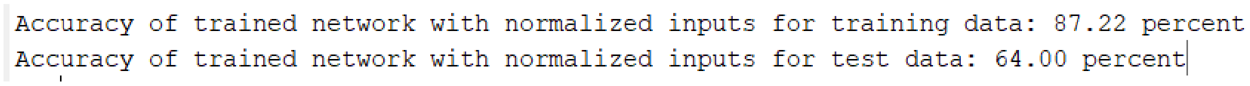
From the output of the training, single layer perceptron can perform reasonably good in the training data. However, the performance in the test data is %71 accuracy which is not very good.



```
Accuracy of trained network for training data: 86.22 percent
Accuracy of trained network for test data: 71.00 percent
```

Part B

As the training output shows below, the accuracy of trained network doesn’t increase as the input data normalized.

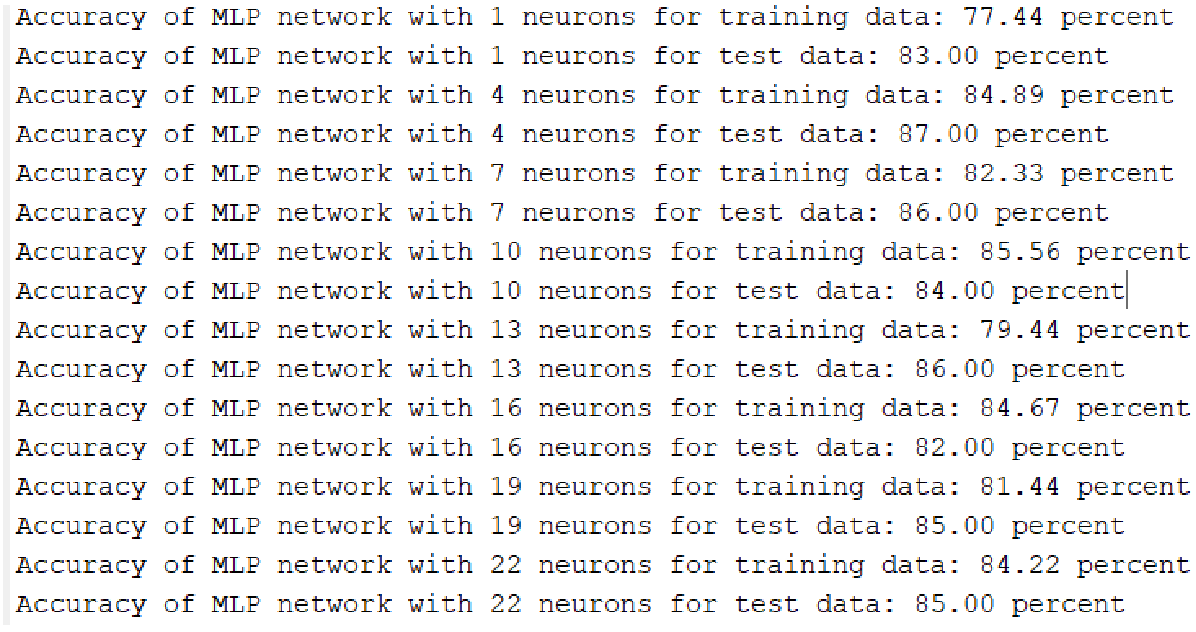


```
Accuracy of trained network with normalized inputs for training data: 87.22 percent
Accuracy of trained network with normalized inputs for test data: 64.00 percent
```

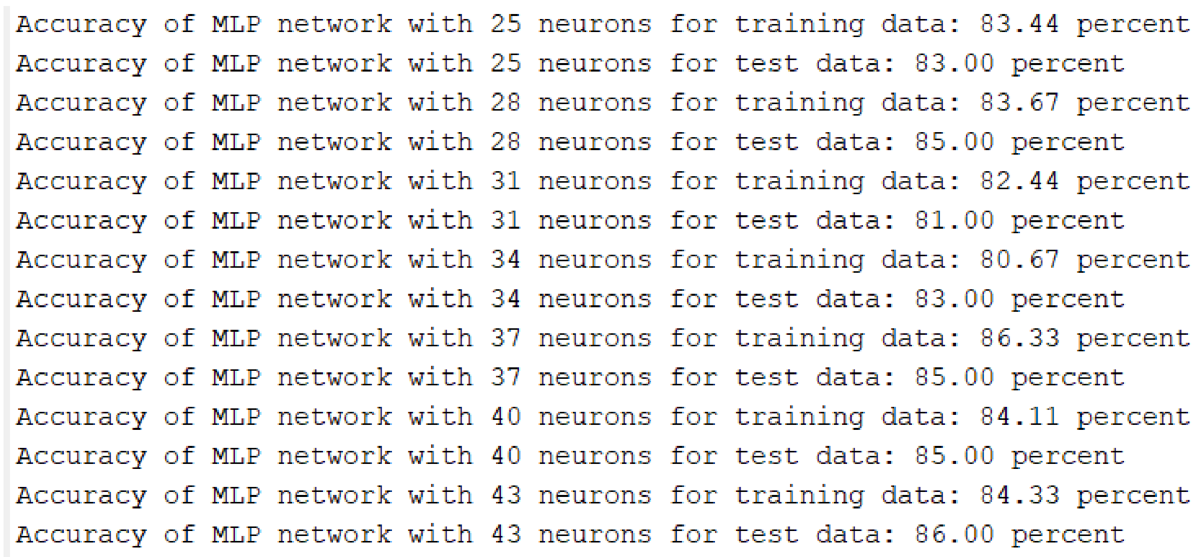
Part C

The MLP used for this part is 1-n-1 type network. In the training part, network is trained for the different numbers of neurons. For 1,4,7,…,49 neurons, the network is trained. As a training function ‘traingda’ is used. The reason of using it is to observing its good performance by trial and error. The activation functions of first layer and output layer are ‘tansig’ and ‘logsig’ respectively. From the training ouput, it can be easily seen that almost for all different numbers of neurons, network achieves approximately the same accuracy.

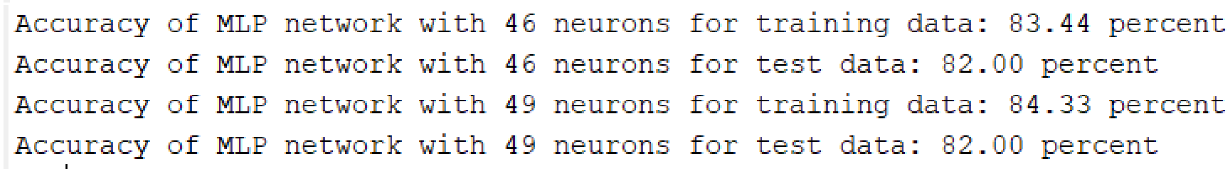
Training Output



```
Accuracy of MLP network with 1 neurons for training data: 77.44 percent
Accuracy of MLP network with 1 neurons for test data: 83.00 percent
Accuracy of MLP network with 4 neurons for training data: 84.89 percent
Accuracy of MLP network with 4 neurons for test data: 87.00 percent
Accuracy of MLP network with 7 neurons for training data: 82.33 percent
Accuracy of MLP network with 7 neurons for test data: 86.00 percent
Accuracy of MLP network with 10 neurons for training data: 85.56 percent
Accuracy of MLP network with 10 neurons for test data: 84.00 percent
Accuracy of MLP network with 13 neurons for training data: 79.44 percent
Accuracy of MLP network with 13 neurons for test data: 86.00 percent
Accuracy of MLP network with 16 neurons for training data: 84.67 percent
Accuracy of MLP network with 16 neurons for test data: 82.00 percent
Accuracy of MLP network with 19 neurons for training data: 81.44 percent
Accuracy of MLP network with 19 neurons for test data: 85.00 percent
Accuracy of MLP network with 22 neurons for training data: 84.22 percent
Accuracy of MLP network with 22 neurons for test data: 85.00 percent
```



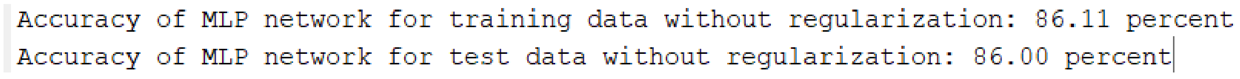
```
Accuracy of MLP network with 25 neurons for training data: 83.44 percent
Accuracy of MLP network with 25 neurons for test data: 83.00 percent
Accuracy of MLP network with 28 neurons for training data: 83.67 percent
Accuracy of MLP network with 28 neurons for test data: 85.00 percent
Accuracy of MLP network with 31 neurons for training data: 82.44 percent
Accuracy of MLP network with 31 neurons for test data: 81.00 percent
Accuracy of MLP network with 34 neurons for training data: 80.67 percent
Accuracy of MLP network with 34 neurons for test data: 83.00 percent
Accuracy of MLP network with 37 neurons for training data: 86.33 percent
Accuracy of MLP network with 37 neurons for test data: 85.00 percent
Accuracy of MLP network with 40 neurons for training data: 84.11 percent
Accuracy of MLP network with 40 neurons for test data: 85.00 percent
Accuracy of MLP network with 43 neurons for training data: 84.33 percent
Accuracy of MLP network with 43 neurons for test data: 86.00 percent
```



```
Accuracy of MLP network with 46 neurons for training data: 83.44 percent
Accuracy of MLP network with 46 neurons for test data: 82.00 percent
Accuracy of MLP network with 49 neurons for training data: 84.33 percent
Accuracy of MLP network with 49 neurons for test data: 82.00 percent
```

Part D

I choose the MLP with n = 19 neurons. It has %81.44 accuracy for training data and %85 accuracy for test data. It has slightly higher accuracy value from the average accuracy value limited in the range of chosen neurons.



```
Accuracy of MLP network for training data without regularization: 86.11 percent
Accuracy of MLP network for test data without regularization: 86.00 percent
```

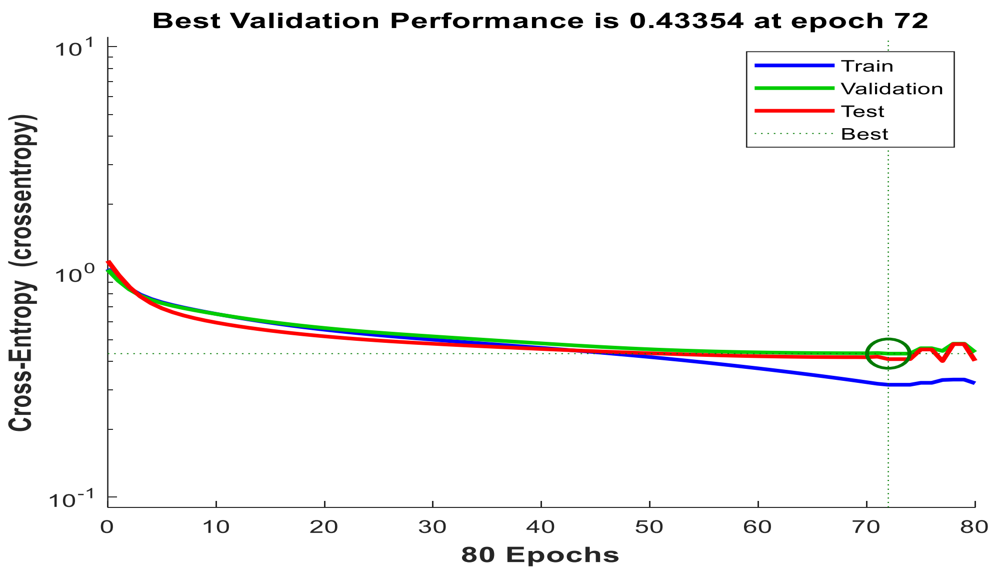


Figure 7 Performance Graph of Training

Figure 7 and the gradient plot of Figure 8 shows that the performance of the network is maximized in the 72th epoch and gradient decreased to its minimum. After 72th epoch, training started overfitting.

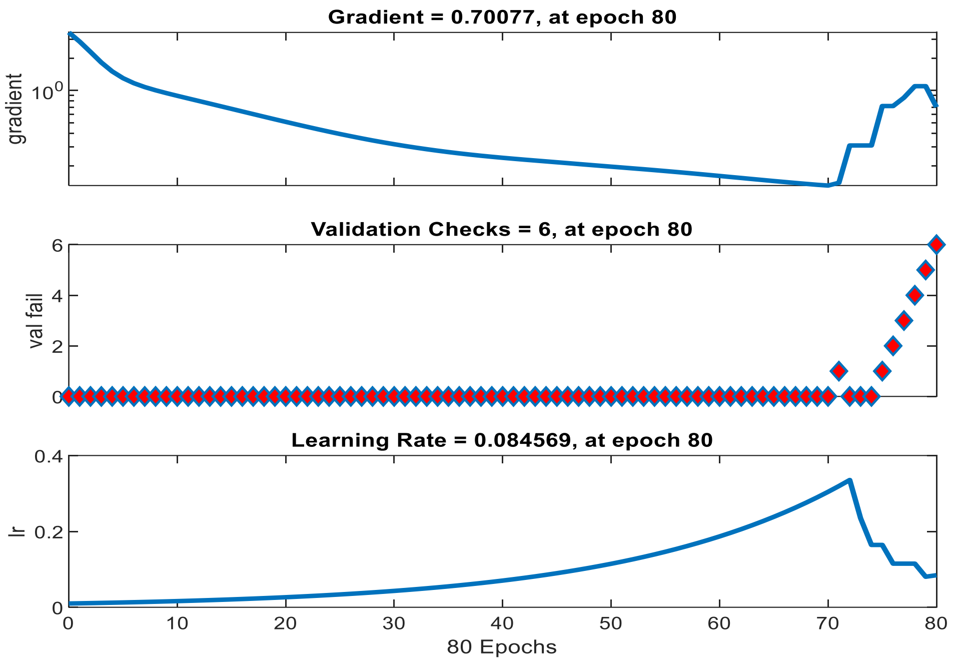
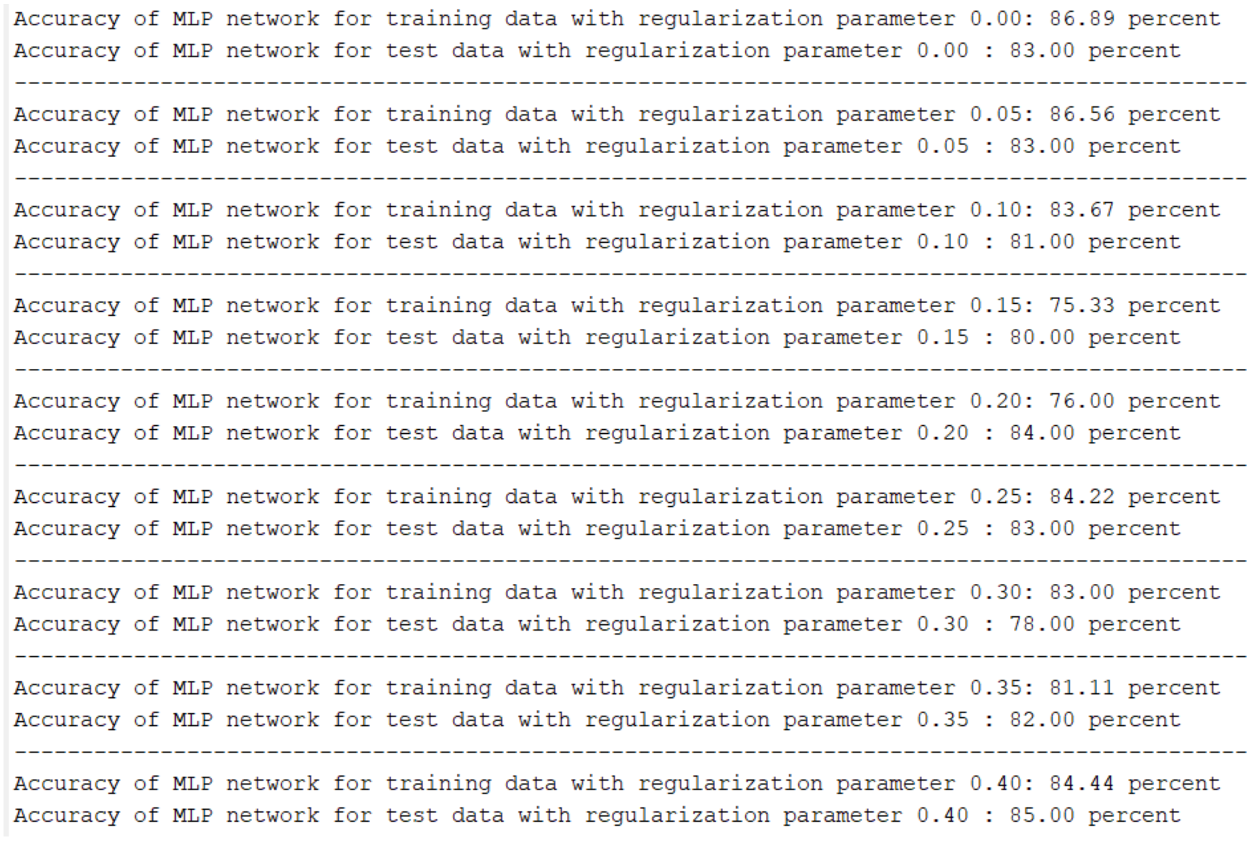


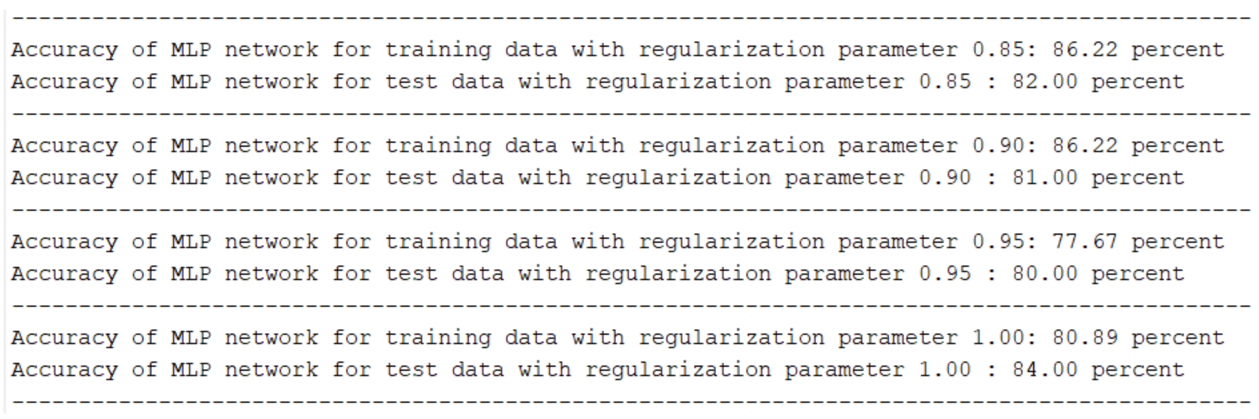
Figure 8 Training State of Network for 80 epochs

Accuracy level of network for different regularization parameters.

```
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.45: 81.56 percent
Accuracy of MLP network for test data with regularization parameter 0.45 : 83.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.50: 77.67 percent
Accuracy of MLP network for test data with regularization parameter 0.50 : 84.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.55: 77.44 percent
Accuracy of MLP network for test data with regularization parameter 0.55 : 80.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.60: 82.11 percent
Accuracy of MLP network for test data with regularization parameter 0.60 : 80.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.65: 79.44 percent
Accuracy of MLP network for test data with regularization parameter 0.65 : 83.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.70: 83.22 percent
Accuracy of MLP network for test data with regularization parameter 0.70 : 84.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.75: 82.67 percent
Accuracy of MLP network for test data with regularization parameter 0.75 : 82.00 percent
-------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.80: 84.56 percent
Accuracy of MLP network for test data with regularization parameter 0.80 : 81.00 percent
```



```
Accuracy of MLP network for training data with regularization parameter 0.00: 86.89 percent
Accuracy of MLP network for test data with regularization parameter 0.00 : 83.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.05: 86.56 percent
Accuracy of MLP network for test data with regularization parameter 0.05 : 83.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.10: 83.67 percent
Accuracy of MLP network for test data with regularization parameter 0.10 : 81.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.15: 75.33 percent
Accuracy of MLP network for test data with regularization parameter 0.15 : 80.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.20: 76.00 percent
Accuracy of MLP network for test data with regularization parameter 0.20 : 84.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.25: 84.22 percent
Accuracy of MLP network for test data with regularization parameter 0.25 : 83.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.30: 83.00 percent
Accuracy of MLP network for test data with regularization parameter 0.30 : 78.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.35: 81.11 percent
Accuracy of MLP network for test data with regularization parameter 0.35 : 82.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.40: 84.44 percent
Accuracy of MLP network for test data with regularization parameter 0.40 : 85.00 percent
```

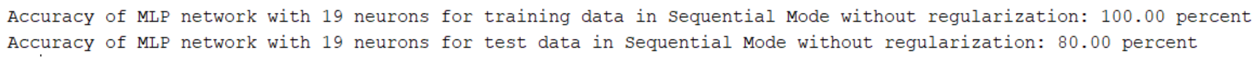


```
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.85: 86.22 percent
Accuracy of MLP network for test data with regularization parameter 0.85 : 82.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.90: 86.22 percent
Accuracy of MLP network for test data with regularization parameter 0.90 : 81.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 0.95: 77.67 percent
Accuracy of MLP network for test data with regularization parameter 0.95 : 80.00 percent
-------------------------------------------------------------------------------------------
Accuracy of MLP network for training data with regularization parameter 1.00: 80.89 percent
Accuracy of MLP network for test data with regularization parameter 1.00 : 84.00 percent
-------------------------------------------------------------------------------------------
```

As it can be sen from training output of network with different regularization parameters, regularization parameter doesn't affect the overall accuracy of entire network. The accuracy levels turned out to be around %80.

Part E

The setup of network is the same as in the batch learning in Part C. There is no regularization either. The only difference is the training mode which is sequential mode in this case. It took almost 25 minutes to train the model whereas it takes 30 seconds in batch mode. The training output is below. The accuracy level for trained data is %100 which couldn't be achieved in any of the steps used the batch mode. The test data accuracy of sequential learning is almost the same as in the batch mode. Since both of them produce a close amount of accuracy, it is reasonable to choose batch learning which is much faster than sequential learning.



```
Accuracy of MLP network with 19 neurons for training data in Sequential Mode without regularization: 100.00 percent
Accuracy of MLP network with 19 neurons for test data in Sequential Mode without regularization: 80.00 percent
```

Part F

Batch mode and sequential mode provide almost the same amount of accuracy. Batch mode is super fast compared to sequential learning; however , it is prone to get stuck in local minimum points of loss function. I tried several training functions. Even though they give a reasonable accuracy level, sometimes they get stuck at local minima and training stops . As a result, it cannot provide sufficient accuracy level in most of such cases. Some of them provide accuracy levels around %70 but most of them was around %80 . Mini batch mode should be more convenient for large data sets as in this example. It decreases the likelihood of local minima problem and fasten the training significantly compared to sequential learning.