## The problem description

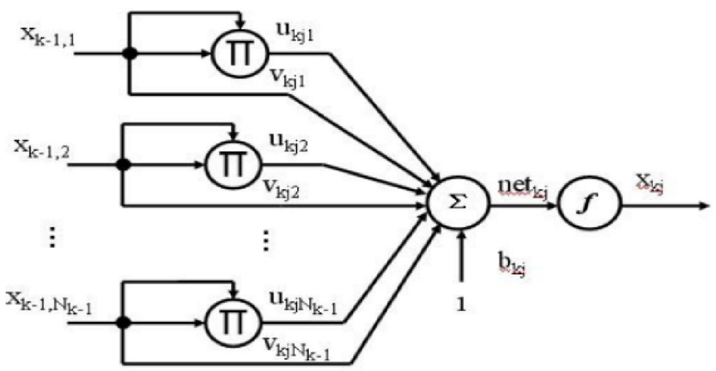
Suppose the output of each neuron in a multilayer perceptron network is

$$x_{kj} = f\left(\sum_{i=1}^{N_{k-1}} (u_{kji}x_{k-1,i}^2 + v_{kji}x_{k-1,i}) + b_{kj}\right)$$

$$\text{for } k = 2, 3, \cdots, M \text{ and } j = 1, 2, \cdots, N_k$$

where both $u_{kji}$ and $v_{kji}$ are the weights connecting the $i$th unit in the layer $k-1$ to the $j$th unit in the layer $k$, $b_{kj}$ is the bias of the $j$th unit in the layer $k$, $N_k$ is the number of units in the $k$ $(1 \leq k \leq M)$, and $f(.)$ is the sigmoidal activation function.

The structure of the unit is shown as the following figure.

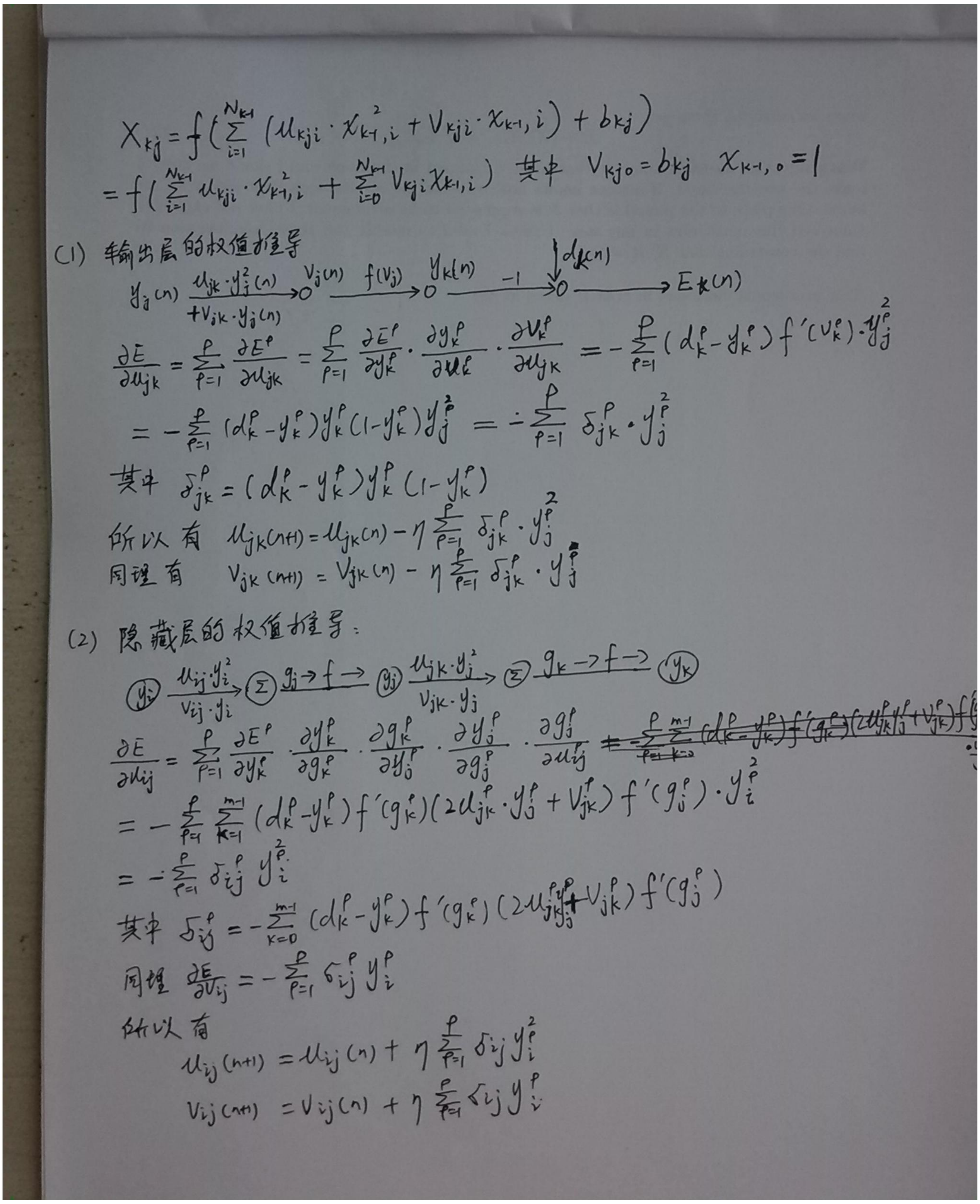


This network is called multi-layer quadratic perceptron (MLQP).

1. Please derive the back-propagation algorithms for MLQPs in both on-line learning and batch learning ways.
2. Write programs to implement the on-line learning and batch learning back-propagation algorithms for training MLQPs with one hidden layer
3. Run your programs to classify the two-spiral problem and compare the training time and generalization performance of the on-line learning and batch learning methods (the number of hidden units can be set to 10).

# Sulotion:

1.The batch learning way for BP takes all the samples in each iteration,  the details of the derivation are showed as bellow:



$$X_{kj} = f\left(\sum_{i=1}^{N_{k-1}} (u_{kji} \cdot x_{k-1,i}^2 + V_{kji} \cdot x_{k-1,i}) + b_{kj}\right)$$

$= f\left(\sum_{i=1}^{N_{k-1}} u_{kji} \cdot x_{k-1,i}^2 + \sum_{i=0}^{N_{k-1}} V_{kji} x_{k-1,i}\right)$ 其中 $V_{kj0} = b_{kj}$ $x_{k-1,0} = 1$

(1) 输出层的权值推导

$$\begin{tikzcd} y_j(n) \arrow[r, "u_{jk} \cdot y_j^2(n)", "+V_{jk} \cdot y_j(n)"'] & \overset{V_j(n)}{o} \arrow[r, "f(V_j)"] & \overset{y_k(n)}{o} \arrow[r, "-1"] & \overset{d_k(n)}{o} \arrow[r] & E_k(n) \end{tikzcd}$$

$$\frac{\partial E}{\partial u_{jk}} = \sum_{p=1}^{P} \frac{\partial E^p}{\partial u_{jk}} = \sum_{p=1}^{P} \frac{\partial E^p}{\partial y_k^p} \cdot \frac{\partial y_k^p}{\partial u_k^p} \cdot \frac{\partial u_k^p}{\partial u_{jk}} = -\sum_{p=1}^{P} (d_k^p - y_k^p) f'(v_k^p) \cdot {y_j^p}^2$$

$$= -\sum_{p=1}^{P} (d_k^p - y_k^p) y_k^p (1 - y_k^p) {y_j^p}^2 = -\sum_{p=1}^{P} \delta_{jk}^p \cdot {y_j^p}^2$$

其中 $\delta_{jk}^p = (d_k^p - y_k^p) y_k^p (1 - y_k^p)$

所以有 $u_{jk}(n+1) = u_{jk}(n) - \eta \sum_{p=1}^{P} \delta_{jk}^p \cdot {y_j^p}^2$

同理有 $V_{jk}(n+1) = V_{jk}(n) - \eta \sum_{p=1}^{P} \delta_{jk}^p \cdot y_j^p$

(2) 隐藏层的权值推导:

$$\begin{tikzcd} \boxed{y_i} \arrow[r, "u_{ij} \cdot y_i^2", "V_{ij} \cdot y_i"'] & \boxed{\Sigma} \arrow[r, "g_j \to f \to"] & \boxed{y_j} \arrow[r, "u_{jk} \cdot y_j^2", "V_{jk} \cdot y_j"'] & \boxed{\Sigma} \arrow[r, "g_k \to f \to"] & \boxed{y_k} \end{tikzcd}$$

$\frac{\partial E}{\partial u_{ij}} = \sum_{p=1}^{P} \frac{\partial E^p}{\partial y_k^p} \cdot \frac{\partial y_k^p}{\partial g_k^p} \cdot \frac{\partial g_k^p}{\partial y_j^p} \cdot \frac{\partial y_j^p}{\partial g_j^p} \cdot \frac{\partial g_j^p}{\partial u_{ij}}$ ~~$= -\sum_{p=1}^{P} \sum_{k=0}^{m-1} (d_k^p - y_k^p) f'(g_k^p)(2u_{jk}^p y_j^p + V_{jk}^p) f$~~

$$= -\sum_{p=1}^{P} \sum_{k=1}^{m-1} (d_k^p - y_k^p) f'(g_k^p)(2u_{jk}^p \cdot y_j^p + V_{jk}^p) f'(g_j^p) \cdot {y_i^p}^2$$

$$= -\sum_{p=1}^{P} \delta_{ij}^p {y_i^p}^2$$

其中 $\delta_{ij}^p = -\sum_{k=0}^{m-1} (d_k^p - y_k^p) f'(g_k^p)(2u_{jk}^p y_j^p + V_{jk}^p) f'(g_j^p)$

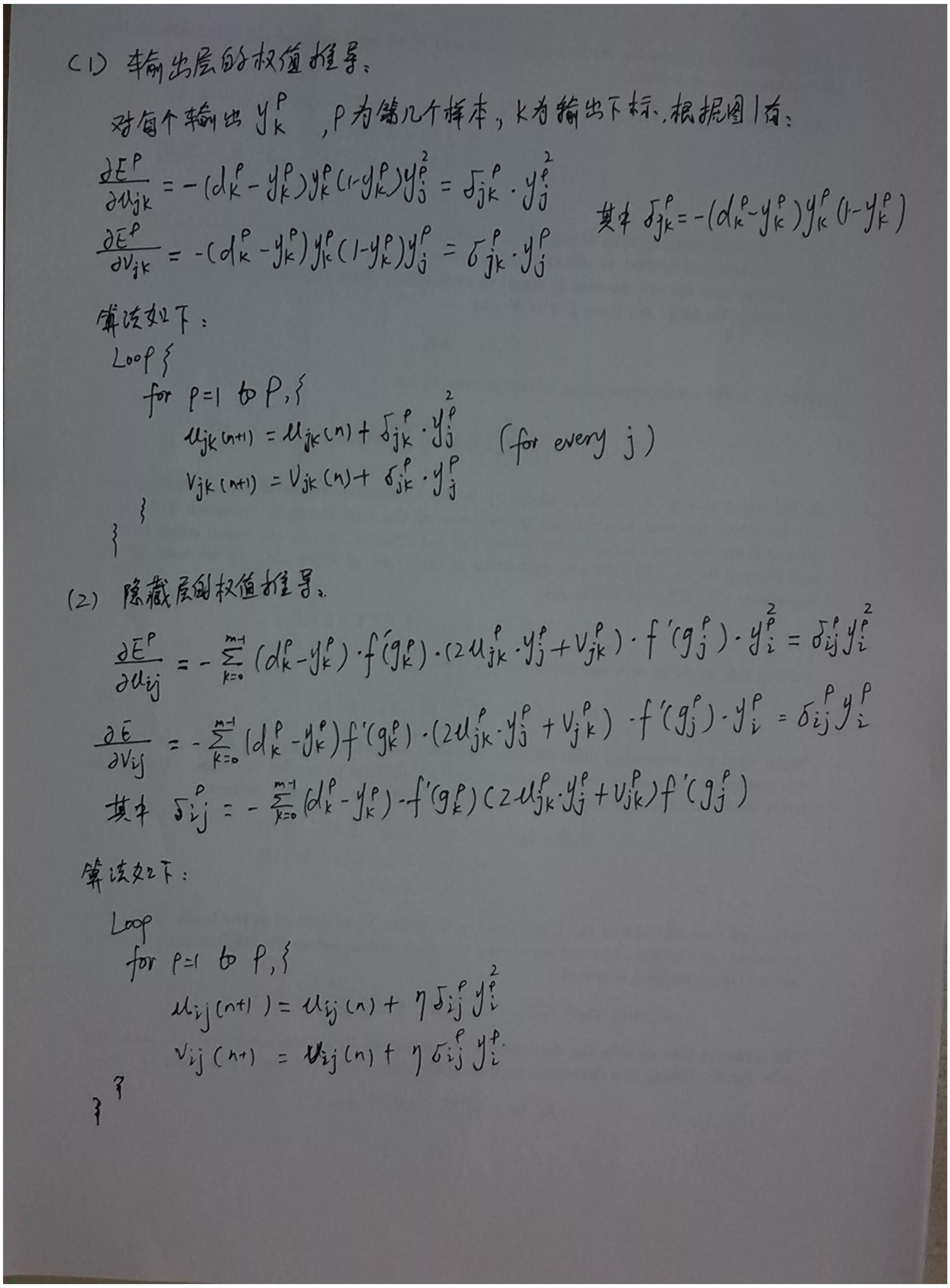
同理 $\frac{\partial E}{\partial V_{ij}} = -\sum_{p=1}^{P} \delta_{ij}^p y_i^p$

所以有

$$u_{ij}(n+1) = u_{ij}(n) + \eta \sum_{p=1}^{P} \delta_{ij} {y_i^p}^2$$

$$V_{ij}(n+1) = V_{ij}(n) + \eta \sum_{p=1}^{P} \delta_{ij} y_i^p$$

The online learning way for BP only takes one sample in each iteration,  and the details of the derivation are showed as bellow:



(1) 输出层的权值推导:

对每个输出 $y_k^p$ ,p为第几个样本，k为输出下标，根据图1有:

$$\frac{\partial E^p}{\partial u_{jk}} = -(d_k^p - y_k^p) y_k^p (1-y_k^p) {y_j^p}^2 = \delta_{jk}^p \cdot {y_j^p}^2$$

$$\frac{\partial E^p}{\partial v_{jk}} = -(d_k^p - y_k^p) y_k^p (1-y_k^p) y_j^p = \delta_{jk}^p \cdot y_j^p$$

其中 $\delta_{jk}^p = -(d_k^p - y_k^p) y_k^p (1-y_k^p)$

算法如下:

```
Loop{
    for p=1 to P,{
        $u_{jk}(n+1) = u_{jk}(n) + \delta_{jk}^p \cdot {y_j^p}^2$   (for every j)
        $v_{jk}(n+1) = v_{jk}(n) + \delta_{jk}^p \cdot y_j^p$
    }
}
```

(2) 隐藏层的权值推导。

$$\frac{\partial E^p}{\partial u_{ij}} = -\sum_{k=0}^{m-1} (d_k^p - y_k^p) \cdot f'(g_k^p) \cdot (2u_{jk}^p \cdot y_j^p + v_{jk}^p) \cdot f'(g_j^p) \cdot {y_i^p}^2 = \delta_{ij}^p {y_i^p}^2$$

$$\frac{\partial E}{\partial v_{ij}} = -\sum_{k=0}^{m-1} (d_k^p - y_k^p) f'(g_k^p) \cdot (2u_{jk}^p \cdot y_j^p + v_{jk}^p) \cdot f'(g_j^p) \cdot y_i^p = \delta_{ij}^p y_i^p$$

其中 $\delta_{ij}^p = -\sum_{k=0}^{m-1} (d_k^p - y_k^p) \cdot f'(g_k^p)(2u_{jk}^p \cdot y_j^p + v_{jk}^p) f'(g_j^p)$

算法如下:

```
Loop
    for p=1 to P,{
        $u_{ij}(n+1) = u_{ij}(n) + \eta \delta_{ij}^p {y_i^p}^2$
        $v_{ij}(n+1) = v_{ij}(n) + \eta \delta_{ij}^p y_i^p$.
    }
}
```

2 I use python to implement this two algorithm, the source code is already

attached,you can just use the python IDE to run the souce code.

3.When we analyse the algorithm ,we will find that the online learning BP updates the parameters with respect to a single training example while batch learning has to scan through the entire training set before taking a single step which is a costly operation if the training set is large. Online learing BP can start making progess right away,and continue to make progress with each example it looks at. When we run the program we will reach the conclusion that online learning gets the parameters close to the minimum much faster than the batch learning BP. When we run the souce code ,it’s abvious that in order to reach a error rate little than 30 percent , the batch learning BP need to iterate more than 57000 times while the online learing BP only need 1000 iterations to reach perfect performance.So we often prefer to choose online learning BP rather than batch learning BP when the training set is large.