Back propagate algorithm Q3 soluation

two layer neural network
three inputs - two neuron in hidden layer - two ouput
hidden function = sigmoid - output function = linear

intial wieght hidden layer θij(1) = [0.3 0.3 0.7;0.9 0.6 0.1]
intial wieght output layer θij(2) = [0.9 0.4;0.2 0.1]
learning rate eta α = 1

iteration (1)
==============

input x(1) = [0.8;1;0.9]
Target y(1) = [1;1]

step(1): Forword (cal the output of NN)

```
Z(2) = θ(1) * xi          =   [1.17;1.41]
a(2) = sigmoid(Z(2))      =   [0.76;0.8]
Z(3) = θ(2) * a(2)        =   [1;0.23]
a(3) = Z(3)               =   [1;0.23]
```

step(2): find the error

```
er =  y -a(3)             =   [0;0.77]
```

step(3): propagate the error to the output layer

```
δ(3) = er*dg(Z(2))            =   [0;0.77]
Dθ(2) = α*δ(3)*a(2)           =   [0 0.59;0 0.62]
```

step(4): propagate the error to the hidden layer

```
     er2 = θ(2)T * δ(3)              =   [0.15;0.08]
     dg(Z(2)) = a(2).(1-a(2))        =   [0.18;0.16]
     δ(2) = er2*dg(Z(2))             =   [0.03;0.01]
     Dθ(1) = α*δ(2)*x                =   [0.02 0.01;0.03 0.01;0.03 0.01]

     step(5) cost =     0.30
    ----------------------

    step(6): Update Wieghts :
    ----------------

    θ(1) = θ(1) + Dθ(1)        =   [0.32 0.91;0.33 0.61;0.73 0.11]
    θ(2) = θ(2) + Dθ(1)        =   [0.9 0.79;0.4 0.72]

iteration (2)
==============
      input x(1)          =    [0.8;1;0.9]
    Target y(1)           =    [1;1]

    step(1): Forword (cal the output of NN)
    ----------------------
     Z(2) = θ(1) * xi          =   [1.24;1.44]
     a(2) = sigmoid(Z(2))      =   [0.78;0.81]
     Z(3) = θ(2) * a(2)        =   [1.03;1.2]
     a(3) = Z(3)               =   [1.03;1.2]

     step(2): find the error
    --------------
     er =  y -a(3)             =   [-0.03;-0.2]

     step(3): propagate the error to the output layer
    -----------------------------
     δ(3) = er*dg(Z(2))              =   [-0.03;-0.2]
```

```
     Dθ(2) = α*δ(3)*a(2)          =   [-0.02 -0.16;-0.02 -0.16]

     step(4): propagate the error to the hidden layer
    -----------------------------
     er2 = θ(2)T * δ(3)           =   [-0.19;-0.16]
     dg(Z(2)) = a(2).(1-a(2))     =   [0.17;0.15]
     δ(2) = er2*dg(Z(2))          =   [-0.03;-0.02]
     Dθ(1) = α*δ(2)*x             =   [-0.02 -0.02;-0.03 -0.02;-0.03 -0.02]

     step(5) cost =     0.02
    ----------------------

    step(6): Update Wieghts :
    ----------------

    θ(1) = θ(1) + Dθ(1)       =   [0.3 0.89;0.3 0.59;0.7 0.09]
    θ(2) = θ(2) + Dθ(1)       =   [0.88 0.63;0.38 0.56]

iteration (3)
==============
      input x(1)          =    [0.8;1;0.9]
    Target y(1)           =    [1;1]

    step(1): Forword (cal the output of NN)
    ----------------------
     Z(2) = θ(1) * xi         =   [1.17;1.38]
     a(2) = sigmoid(Z(2))     =   [0.76;0.8]
     Z(3) = θ(2) * a(2)       =   [0.97;0.93]
     a(3) = Z(3)              =   [0.97;0.93]

     step(2): find the error
    --------------
     er =  y -a(3)            =   [0.03;0.07]
```

```
 step(3): propagate the error to the output layer
-----------------------------
 δ(3) = er*dg(Z(2))          =   [0.03;0.07]
 Dθ(2) = α*δ(3)*a(2)         =   [0.02 0.05;0.02 0.06]

 step(4): propagate the error to the hidden layer
-----------------------------
 er2 = θ(2)T * δ(3)          =   [0.07;0.05]
 dg(Z(2)) = a(2).(1-a(2))    =   [0.18;0.16]
 δ(2) = er2*dg(Z(2))         =   [0.01;0.01]
 Dθ(1) = α*δ(2)*x            =   [0.01 0.01;0.01 0.01;0.01 0.01]

 step(5) cost =     0.00
----------------------

step(6): Update Wieghts :
----------------

θ(1) = θ(1) + Dθ(1)      =   [0.31 0.9;0.31 0.6;0.71 0.1]
θ(2) = θ(2) + Dθ(1)      =   [0.9 0.68;0.4 0.62]
>>
```