

```

tr = csvread('training_set.csv');
val = csvread('validation_set.csv');

x = tr(:,1:2);
t = tr(:,3);

x_val = val(:,1:2);
t_val = val(:,3);

eta = 0.02;
M0 = 2;
M1 = 26;
M2 = M1/2;
MOutput = 1;

weight0 = (1/M0).*randn(M1,M0);
weight1 = (1/M1).*randn(M2,M1);
weight2 = (1/M2).*randn(MOutput,M2);
Weight = {weight0, weight1, weight2};
theta0 = 0.2*(1 - 2*rand(1,M1));
theta1 = 0.2*(1 - 2*rand(1,M2));
theta2 = 0.2*(1 - 2*rand(1,MOutput));
Theta = {theta0, theta1, theta2};
Error = {zeros(1,M1), zeros(1,M2), zeros(1,MOutput)};
ThetaError = Theta;

% bias = 1 - 2*rand;
B = 1/2;
nbrOfTargets = length(t);
%CVec = zeros(nbrOfTargetPatterns,1);

p = length(t);
p_val = length(t_val);

%Outputs = {zeros(p,M1), zeros(p,M2), zeros(p,1)};
counter = 2;
plotEvery = 10000;
lenK = 3*10^6;
lenVec = lenK / plotEvery + 1;
H_tr = zeros(lenVec,1);
H_val = zeros(lenVec,1);
C_val = zeros(lenVec,1);

for k = 1:lenK
    mu = randi([1,p]);
    V1 = LayerValue(Theta{1}, Weight{1}, x(mu,:), 1);
    V2 = LayerValue(Theta{2}, Weight{2}, V1, 1);
    output = Output(Weight{3}, V2, Theta{3}, 1);
    V = {x(mu,:), V1, V2, output};
    % H = EnergyFunction(t, output);
    % C = ErrorFunction(t, output, p);
    if k == 1
        H = EnergyFunction(t, output);
        H_tr(1) = H;
        V1_val = LayerValue(Theta{1}, Weight{1}, x_val, p_val);
        V2_val = LayerValue(Theta{2}, Weight{2}, V1_val, p_val);
        output_val = Output(Weight{3}, V2_val, Theta{3}, p_val);
        H_val(1) = EnergyFunction(t_val, output_val);
        C_val(1) = ErrorFunction(t_val, output_val, p_val);
    end
    if mod(k, plotEvery) == 0
        H_tr(counter) = H;
        V1_val = LayerValue(Theta{1}, Weight{1}, x_val, p_val);

```

```

V2_val = LayerValue(Theta{2}, Weight{2}, V1_val, p_val);
output_val = Output(Weight{3}, V2_val, Theta{3}, p_val);
H_val(counter) = EnergyFunction(t_val, output_val);
C_val(counter) = ErrorFunction(t_val, output_val, p_val);
disp(k)
disp(C_val(counter))
if C_val(counter) < 0.12
    break;
end
counter = counter + 1;
end
Error{3} = calcOutputError(Weight{3}, V2, t(mu), output, mu, Theta{3}, B);
for i = 2:-1:1
    Error{i} = calcLayerError(Error{i+1}, Weight{i+1}, Weight{i}, V{i},...
        mu, Theta{i}, B);
end
for j = 1:3
    Theta{j} = Theta{j} - eta * Error{j};
    Weight{j} = Weight{j} + eta * Error{j}' * V{j};
end
end
figure(1)
plot([1, plotEvery:plotEvery:lenK], H_tr)
hold on
plot([1, plotEvery:plotEvery:lenK], H_val)
hold off
figure(2)
plot([1, plotEvery:plotEvery:lenK], C_val)

csvwrite('w1.csv', Weight{1});
csvwrite('w2.csv', Weight{2});
csvwrite('w3.csv', Weight{3});
csvwrite('t1.csv', Theta{1});
csvwrite('t2.csv', Theta{2});
csvwrite('t3.csv', Theta{3});

function V = LayerValue(bias, weight, inputVal, p)
    sumTot = weight * inputVal'; % matrix with (j, mu)
    V = tanh(sumTot - repmat(bias', 1, p))';
end

function H = EnergyFunction(t, output)
    sumVec = (t - output).^2;
    H = (1/2) * sum(sumVec);
end

function C = ErrorFunction(t, output, p)
    signOutput = sign(output);
    signOutput(signOutput == 0) = -1;
    sumVec = abs(signOutput - t);
    C = (1/(2*p)) * sum(sumVec);
end

function Error = calcOutputError(weight, inputVal, t, output, mu, bias, B)
    sumTot = weight * inputVal';
    b = sumTot - bias';
    gprim = B * (1 - (tanh(B * b)).^2);
    Error = (t - output) * gprim;
end

function Error = calcLayerError(prevError, prevWeight, weight, inputVal, mu, bias, B)
    sumTot = weight * inputVal';
    b = sumTot - bias';

```

```
gprim = B * (1 - (tanh(B * b)).^2);  
Error = prevError * prevWeight .* gprim';  
end  
  
function output = Output(weight, inputVal, bias, p)  
    sumTot = weight * inputVal';  
    output = tanh((1/2) * (sumTot - repmat(bias', 1, p)))';  
end
```

```

tr = csvread('training_set.csv');
val = csvread('validation_set.csv');

x = tr(:,1:2);
t = tr(:,3);

x_val = val(:,1:2);
t_val = val(:,3);

eta = 0.03;
M0 = 2;
MOutput = 1;

convergedAtIteration = zeros(100,1);
MSetTrial = zeros(100,2);

for trial = 1:100
    disp(trial);
    M1 = 41 - randi(11); % 37 best so far, k = 1 020 000 for C_val<0.12
    M2 = 21 - randi(11); % 11 best so far, k = 1 020 000 for C_val<0.12

    MSetTrial(trial, :) = [M1, M2];

    weight0 = (1/M0).*randn(M1,M0);
    weight1 = (1/M1).*randn(M2,M1);
    weight2 = (1/M2).*randn(MOutput,M2);
    Weight = {weight0, weight1, weight2};
    theta0 = 0.2*(1 - 2*rand(1,M1));
    theta1 = 0.2*(1 - 2*rand(1,M2));
    theta2 = 0.2*(1 - 2*rand(1,MOutput));
    Theta = {theta0, theta1, theta2};
    Error = {zeros(1,M1), zeros(1,M2), zeros(1,MOutput)};
    ThetaError = Theta;

    B = 1/2;
    nbrOfTargets = length(t);

    p = length(t);
    p_val = length(t_val);

    plotEvery = 10000;
    lenK = 1.4*10^6;

    for k = 1:lenK
        mu = randi([1,p]);
        V1 = LayerValue(Theta{1}, Weight{1}, x(mu,:), 1);
        V2 = LayerValue(Theta{2}, Weight{2}, V1, 1);
        output = Output(Weight{3}, V2, Theta{3}, 1);
        V = {x(mu,:), V1, V2, output};
        if k == 1
            V1_val = LayerValue(Theta{1}, Weight{1}, x_val, p_val);
            V2_val = LayerValue(Theta{2}, Weight{2}, V1_val, p_val);
            output_val = Output(Weight{3}, V2_val, Theta{3}, p_val);
            C_val(1) = ErrorFunction(t_val, output_val, p_val);
        end
        if mod(k, plotEvery) == 0
            V1_val = LayerValue(Theta{1}, Weight{1}, x_val, p_val);
            V2_val = LayerValue(Theta{2}, Weight{2}, V1_val, p_val);
            output_val = Output(Weight{3}, V2_val, Theta{3}, p_val);
            C_val = ErrorFunction(t_val, output_val, p_val);
            if C_val < 0.12
                break;
            end
        end
    end
end

```

```

end
Error{3} = calcOutputError(Weight{3}, V2, t(mu), output, mu, Theta{3}, B);
for i = 2:-1:1
    Error{i} = calcLayerError(Error{i+1}, Weight{i+1}, Weight{i}, V{i},...
        mu, Theta{i}, B);
end
for j = 1:3
    Theta{j} = Theta{j} - eta * Error{j};
    Weight{j} = Weight{j} + eta * Error{j}' * V{j};
end
end

convergedAtIteration(trial) = k;
disp(k);
end

function V = LayerValue(bias, weight, inputVal, p)
    sumTot = weight * inputVal'; % matrix with (j, mu)
    V = tanh(sumTot - repmat(bias', 1, p))';
end

function H = EnergyFunction(t, output)
    sumVec = (t - output).^2;
    H = (1/2) * sum(sumVec);
end

function C = ErrorFunction(t, output, p)
    signOutput = sign(output);
    signOutput(signOutput == 0) = -1;
    sumVec = abs(signOutput - t);
    C = (1/(2*p)) * sum(sumVec);
end

function Error = calcOutputError(weight, inputVal, t, output, mu, bias, B)
    sumTot = weight * inputVal';
    b = sumTot - bias';
    gprim = B * (1 - (tanh(B * b)).^2);
    Error = (t - output) * gprim;
end

function Error = calcLayerError(prevError, prevWeight, weight, inputVal, mu, bias, B)
    sumTot = weight * inputVal';
    b = sumTot - bias';
    gprim = B * (1 - (tanh(B * b)).^2);
    Error = prevError * prevWeight .* gprim';
end

function output = Output(weight, inputVal, bias, p)
    sumTot = weight * inputVal';
    output = tanh((1/2) * (sumTot - repmat(bias', 1, p)))';
end

```