```
function netbp_full
close all; %%%%%%%%%%%%%%%%%%%%%%%%
tic
%NETBP_FULL
%   Extended version of netbp, with more graphics
%
%   Set up data for neural net test
%   Use backpropagation to train
%   Visualize results
%
% C F Higham and D J Higham, Aug 2017
%
%%%%%%% DATA %%%%%%%%%%
% xcoords, ycoords, targets
%x1 = [0.1,0.3,0.1,0.6,0.4,0.6,0.5,0.9,0.4,0.7];
%x2 = [0.1,0.4,0.5,0.9,0.2,0.3,0.6,0.2,0.4,0.6];
%y = [ones(1,5) zeros(1,5); zeros(1,5) ones(1,5)];

data = load('dataset.mat');

x1 = data.X(:,1);
x2 = data.X(:, 2);
y = data.Y';
y([1, 2],:)=y([2,1],:);

figure(1)
clf
a1 = subplot(1,1,1);
plot(x1(1:42),x2(1:42),'ro','MarkerSize',12,'LineWidth',4)
hold on
plot(x1(43:84),x2(43:84),'bx','MarkerSize',12,'LineWidth',4)
a1.XTick = [0 1];
a1.YTick = [0 1];
a1.FontWeight = 'Bold';
a1.FontSize = 16;
xlim([0,1])
ylim([0,1])

%print -dpng pic_xy.png

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initialize weights and biases
rng(5000);
W2 = 0.5*randn(5,2); %
W3 = 0.5*randn(3,5); %%
W4 = 0.5*randn(2,3);
b2 = 0.5*randn(5,1); %
b3 = 0.5*randn(3,1);
b4 = 0.5*randn(2,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Forward and Back propagate
% Pick a training point at random
eta = 0.05;
```

```
Niter = 1e6;
batches = 4; %%%%%%%%
savecost = zeros(Niter,1); %%%%%%%%
accuracies = zeros(1, Niter); %%%%%%%%
for counter = 1:Niter %%%%%%%%%%
    for batch=1:batches
        k = randi(84);
        x = [x1(k); x2(k)];
        % Forward pass
        a2 = activate(x,W2,b2);
        a3 = activate(a2,W3,b3);
        a4 = activate(a3,W4,b4);
        % Backward pass
        delta4 = a4.*(1-a4).*(a4-y(:,k));
        delta3 = a3.*(1-a3).*(W4'*delta4);
        delta2 = a2.*(1-a2).*(W3'*delta3);
        % Gradient step
        W2 = W2 - eta*delta2*x';
        W3 = W3 - eta*delta3*a2';
        W4 = W4 - eta*delta4*a3';
        b2 = b2 - eta*delta2;
        b3 = b3 - eta*delta3;
        b4 = b4 - eta*delta4;
        % Monitor progress

        [newcost, accuracy] = cost(W2,W3,W4,b2,b3,b4);   % display cost to screen
        accuracies(counter) = accuracy;%%%%%%%%%%%%%%
        savecost(counter) = newcost;
        if (accuracy > 0.97)
            break
        end
    end
    if (accuracy > 0.97) %&& (accuracy < 1)           %%%%%%%%%%%
        fprintf("*** break ***\n"); %%%%%%%%%%
        break                       %%%%%%%%%%%%
    end                             %%%%%%%%%%%%
end
%newcost = newcost   %%%%%%
%accuracy = accuracy %%%%%%
fprintf("Iterations: %i\n", counter); %%%%%%%%
%accuracies = accuracies(1:counter); %%%%%%%%%

figure(2)
clf
semilogy([1:1e4:Niter],savecost(1:1e4:Niter),'b-','LineWidth',2)
xlabel('Iteration Number')
ylabel('Value of cost function')
set(gca,'FontWeight','Bold','FontSize',18)
print -dpng pic_cost.png

%%%%%%%%%% Display shaded and unshaded regions
N = 500;
Dx = 1/N;
Dy = 1/N;
xvals = [0:Dx:1];
```

```
yvals = [0:Dy:1];
for k1 = 1:N+1
    xk = xvals(k1);
    for k2 = 1:N+1
        yk = yvals(k2);
        xy = [xk;yk];
        a2 = activate(xy,W2,b2);
        a3 = activate(a2,W3,b3);
        a4 = activate(a3,W4,b4);
        Aval(k2,k1) = a4(1);
        Bval(k2,k1) = a4(2);
     end
end
[X,Y] = meshgrid(xvals,yvals);

figure(3)
clf
a2 = subplot(1,1,1);
Mval = Aval>Bval;
contourf(X,Y,Mval,[0.5 0.5])
hold on
colormap([1 1 1; 0.8 0.8 0.8])
plot(x1(1:42),x2(1:42),'ro','MarkerSize',12,'LineWidth',4) %%%%%%%
plot(x1(43:84),x2(43:84),'bx','MarkerSize',12,'LineWidth',4) %%%%%%%%
a2.XTick = [0 1];
a2.YTick = [0 1];
a2.FontWeight = 'Bold';
a2.FontSize = 16;
xlim([0,1])
ylim([0,1])

print -dpng pic_bdy_bp.png

figure(4) %%%%%%%%%%%%%%%%%
clf
semilogy([1:1e3:Niter],accuracies(1:1e3:Niter),'b-','LineWidth',2)
title('Iterations vs Accuracy (step size of 100)')
xlabel('Iteration Number')
ylabel('Accuracy')
set(gca,'FontWeight','Bold','FontSize',12)



toc

  function [costval, accuracy] = cost(W2,W3,W4,b2,b3,b4)

     costvec = zeros(84,1);
     total_pts = 84;                                    %%%%%%%
     class_pts = 0;                                     %%%%%%%%
     accuracy = 0;
     for i = 1:84
         x =[x1(i);x2(i)];
         a2 = activate(x,W2,b2);
```

```
        a3 = activate(a2,W3,b3);
        a4 = activate(a3,W4,b4);
        y1 = a4(1); y2 = a4(2);
        costvec(i) = norm(y(:,i) - a4,2);

        if (y1 > y2) && isequal(y(:,i),[1;0])        %%%%%%%%
            class_pts = class_pts + 1;                          %%%%%%%%
        elseif (y1 < y2) && isequal(y(:,i),[0;1])    %%%%%%%%
            class_pts = class_pts + 1;                          %%%%%%%%
        end                                          %%%%%%%%
        %total_pts = total_pts + 1;                  %%%%%%%%
    end
    costval = norm(costvec,2)^2;
    accuracy = class_pts/total_pts;                  %%%%%%%%

  end % of nested function

end
```