SFWR ENG 4X03 Assignment 4

1.a)

```
function netbp(points,labels,neurons,learning_rate,niter,file)
%NETBP Uses backpropagation to train a network
%%%%%%% DATA %%%%%%%%%%
x1 = points(1,:);
x2 = points(2,:);
% Initialize weights and biases
rng(5000);
W2 = 0.5*randn(neurons(1),2); W3 = 0.5*randn(neurons(2),neurons(1)); W4 =
0.5*randn(neurons(3),neurons(2));
b2 = 0.5*randn(neurons(1),1); b3 = 0.5*randn(neurons(2),1); b4 = 0.5*randn(neurons(3),1);
% Forward and Back propagate
savecost = zeros(niter,1); % value of cost function at each iteration
for counter = 1:niter
    k = randi(14); % choose a training point at random
    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-labels(:,k));
    delta3 = a3.*(1-a3).*(W4'*delta4);
    delta2 = a2.*(1-a2).*(W3'*delta3);
    % Gradient step
    W2 = W2 - learning_rate*delta2*x';
    W3 = W3 - learning_rate*delta3*a2';
    W4 = W4 - learning_rate*delta4*a3';
    b2 = b2 - learning_rate*delta2;
    b3 = b3 - learning_rate*delta3;
    b4 = b4 - learning_rate*delta4;
    % Monitor progress
    newcost = cost(W2,W3,W4,b2,b3,b4);
    %fprintf("newcost = %f\n",newcost); % display cost to screen
    savecost(counter) = newcost;
end
% Show decay of cost function
save costvec

semilogy([1:1e4:niter],savecost(1:1e4:niter))

function costval = cost(W2,W3,W4,b2,b3,b4)
    costvec = zeros(10,1);
    for i = 1:10
        x =[x1(i);x2(i)];
        a2 = activate(x,W2,b2);
        a3 = activate(a2,W3,b3);
        a4 = activate(a3,W4,b4);
        costvec(i) = norm(labels(:,i) - a4,2);
    end
    costval = norm(costvec,2)^2;
end % of nested function

save(file,'W2','W3','W4','b2','b3','b4','savecost','learning_rate');
end
```

b)

```
function category = classifypoints(file,points)

load(file);

% Forward pass
a2 = activate(points,W2,b2);
a3 = activate(a2,W3,b3);
a4 = activate(a3,W4,b4);
% Categorizing output of point
category = a4(1,:) >= a4(2,:);

end
```

c) Larger values of learning rates seem to faster result in an accurate answer and more neurons result in more complex mapping. The best combination that I was able to run is a learning rate of 1 and a neuron configuration of [7 50 2]

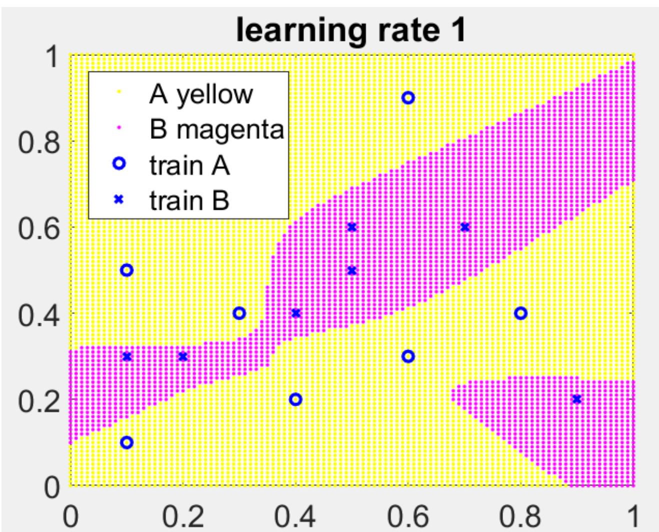

Figure 1: Result after training with learning rate 1 and neuron of [7 50 2]

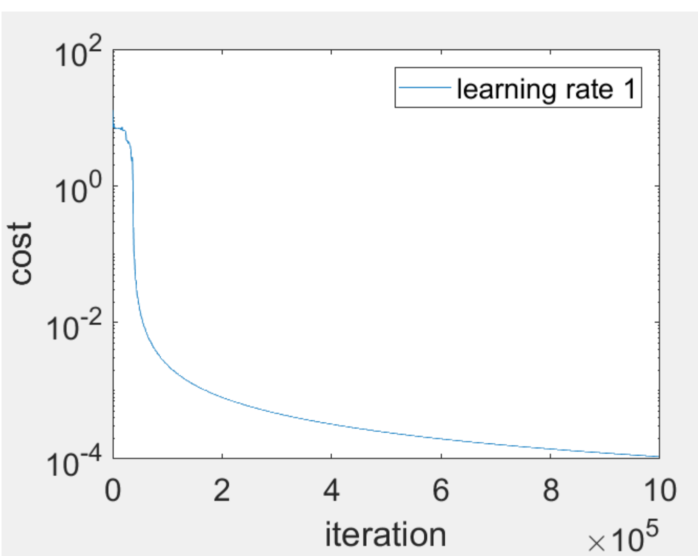

Figure 2: Cost vs iteration curve of learning rate of 1

2.

Bisection

```
f  = @(x) x - exp(2-sqrt(x));    f(x) = 1.877321666688658e+00
g  = @(x) x*sin(x^2)-1;          g(x) = 4.368127214401284e+00
```

Newton

```
h  = @(x) x^3-2*x-5;      h(x) = 2.094551481543604e+00
g  = @(x) x*sin(x^2)-1    g(x) = 3.930741781510152e+00
```

Fzero

```
fzero  f(x) = 1.877321666687555e+00
fzero  g(x) = 4.368127214401134e+00
fzero  h(x) = 2.094551481542327e+00
```

Newton's method for g(x) has a significantly larger error

3.a)

the computation never end as the iteartions cycle therefore it ininitely loops

b)

using and initial value of 1 ± 10^{-10} will converge because the slight offest will break the cycle

4.

Newton

```
f(x)   x=0 : Inf
f(x)   x=1 : 9.424769646797397e+00
```

fsolve

```
f(x) fsolve  x=1 : 3.129879311117518e+00
```

The values from newton and fsolve are not similar because fsolve found the nearest root which is at 3.13 whereas in newton's method, the initial point has a very small slope which causes the next iteration to pass the nearest zero

5.a)

$|d| > |f'(x_n)|$

b)

linear

c)

$d = f'(x_n)$

6.

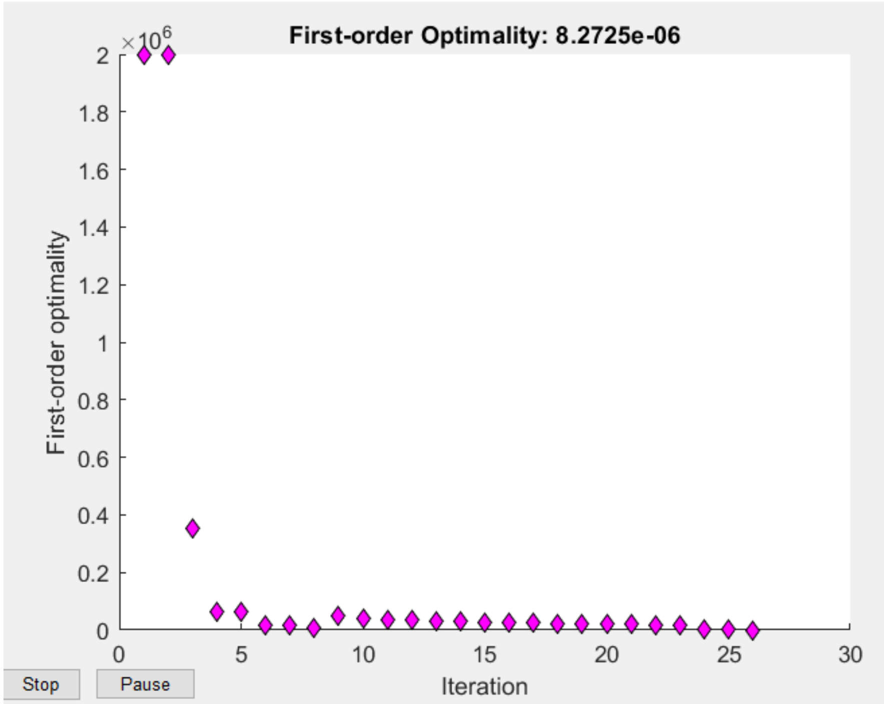

Figure 3: Plot of first order optimality

The iteration of zero converges at an exponetial manner and is able to find the root within 30 iterations.

7.

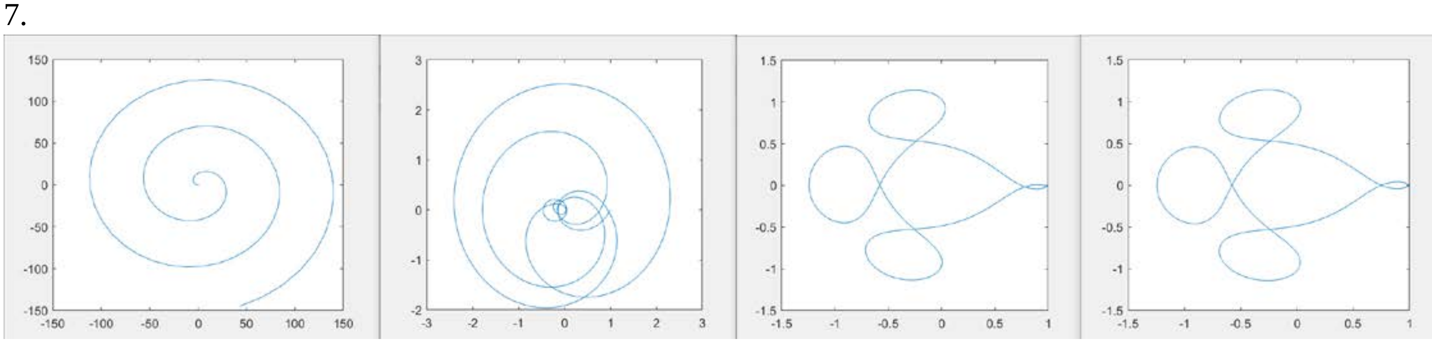

Figure 4: Plots with steps size 100, 1000, 10 000, 20 000 respectively

The least amount of step sizes needed for the correct plot is 10 000 steps

8.

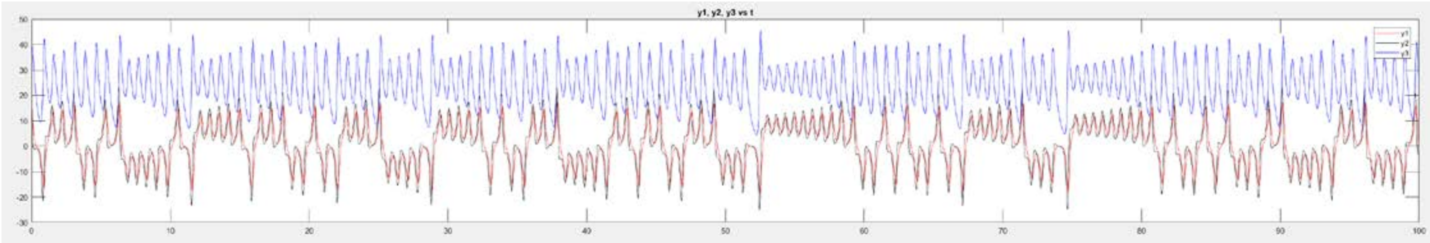

Figure 5: plot of y_1, y_2, y_3, vs t

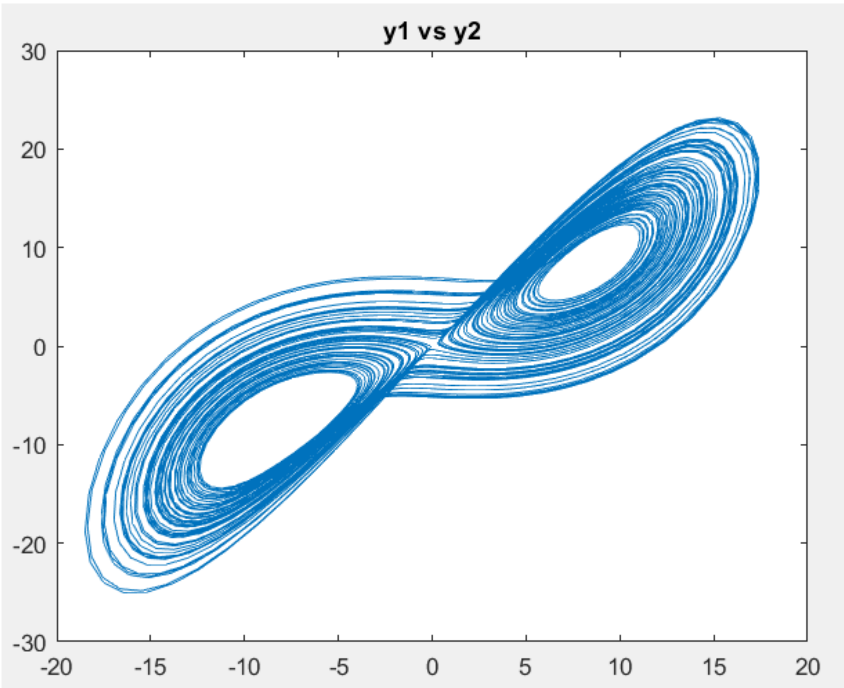

Figure 6: plot of $y_1$vs y_2

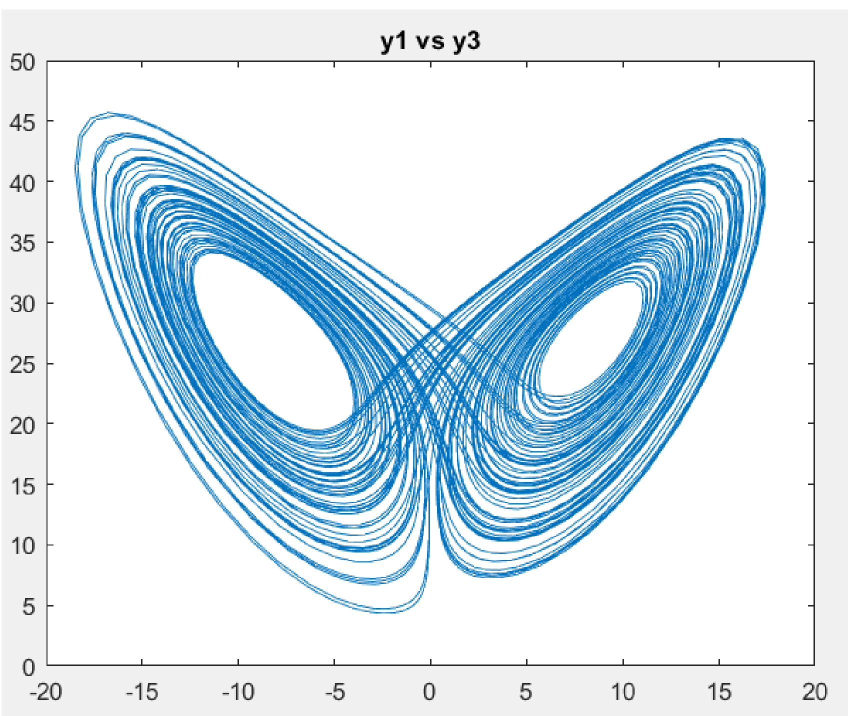

Figure 7: plot of $y_1$vs y_3

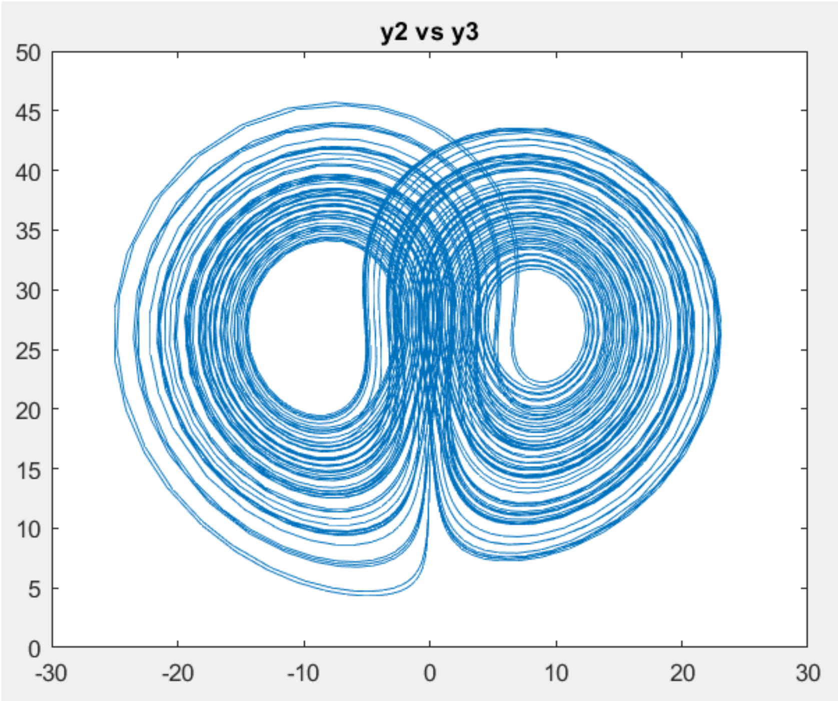

Figure 8: plot of $y_2$vs y_3

When the initial conditions are slightly changed, the qualitative figure of the graph remains the same but at the extreme points of the path slightly divert to different places either curving wider or tighter.