Onur Poyraz
2010401036
4 May 2015

Homework IV

In this project I deal with the continuous Hopfield Model.In the lecture and in this project we see that there is huge similarities between binary and continuous Hopfield network. Therefore in this project I look at the first project which I already done.

In this project it is given that there will be two neurons model with T=[0 1;1 0]. Also non-linear function is given us and we calculate the inverse of this non-linear function.

In the code I find the equilibrium points, trajectories and both 2D and 3D energy contour maps. I begin the my project with the initializing the values and components that determined before the algorithm. I put the code for this part is given below.

```
r1=1;
r2=1;
c1=1;
c2=1;
lambda=2; %scaling factor
T=[0 1;1 0]; % T-matrix
func=@(x) 2/pi*atan(lambda*pi*x/2); %non-linear function
inv_func=@(x) 2/(lambda*pi)*tan(pi*x/2); %inverse of non-linear function
energy= @(x,y) -1/2*(x*y+y*x)+1/lambda*integral(inv_func,0,x)/r1+1/lambda*integral(inv_func,0,y)/r2; % energy function
```

This part is obviously defines the parameters that chosen and the functions that will be used in the project.

After this part I find the first input and output values. First input is in 4th quadrant and it takes random values.

```
epoch=1; %number of iteration
x(epoch,:)=[rand -rand]; %initial input values
y(epoch,:)=func(x(epoch,:));
e(epoch)=energy(y(1),y(2));
```

From this point I begin the my main algorithm. In this part I calculated delta(x) values, x values from new deltas and e values. Also I control the convergence of energy in here.

```
while (1) %%algorithm part
  epoch=epoch+1;
  delta=[0 0];
  delta(1,1)=r1*y(epoch-1,2)*T(2,1)-r1*c1*x(epoch-1,1);
```

```
    delta(1,2)=r2*y(epoch-1,1)*T(1,2)-r2*c2*x(epoch-1,2);
    x(epoch,1)=x(epoch-1,1)+delta(1,1); %%update the x matrix
    x(epoch,2)=x(epoch-1,2)+delta(1,2);
    y(epoch,:)=func(x(epoch,:));  %%update the y matrix
    e(epoch)=energy(y(epoch,1),y(epoch,2));
    if abs(e(epoch)-e(epoch-1))<0.0001
        break
    end
end
```

From now on, I work on the plotting part. First of all I draw the convergence of trajectories and stable equilibrium points. I draw beginning point with blue "o" and I draw the stable equilibrium points with black "x".

```
figure(1);
plot(y(1,1),y(1,2),'ro',y(:,1),y(:,2)); %plot of the trajectories
hold on;
l=length(e);
plot(y(l,1),y(l,2),'kx',y(l-1,1),y(l-1,2),'kx'); %plot of the stable equilibrium points
```

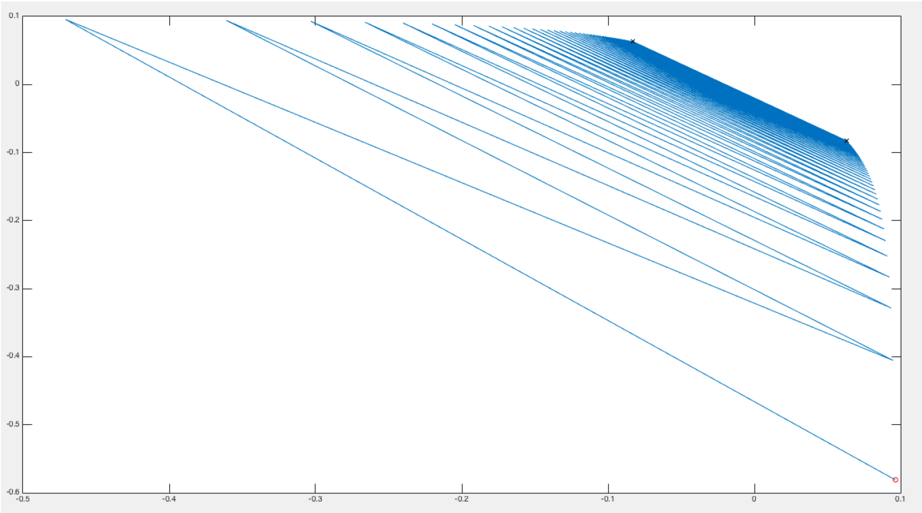
After plotting trajectories and equilibrium points I draw the energy contour map. For that first of all I create energyMatrix and I plot 2D and 3D contours.

```
energyMatrix = zeros(100,100);
v1 = linspace(-1,1,100);
v2 = linspace(-1,1,100);
for i=1:100
    for j=1:100
        if(i==j)
            continue;
        end
        energyMatrix(i,j)=energy(v1(i),-v2(j));
    end
end
v=0:0.25:1;
figure(2)
    contour(v1,v2,energyMatrix,v,'ShowText','on');
figure(3)
    contour3(v1,v2,energyMatrix,'ShowText','on');
```

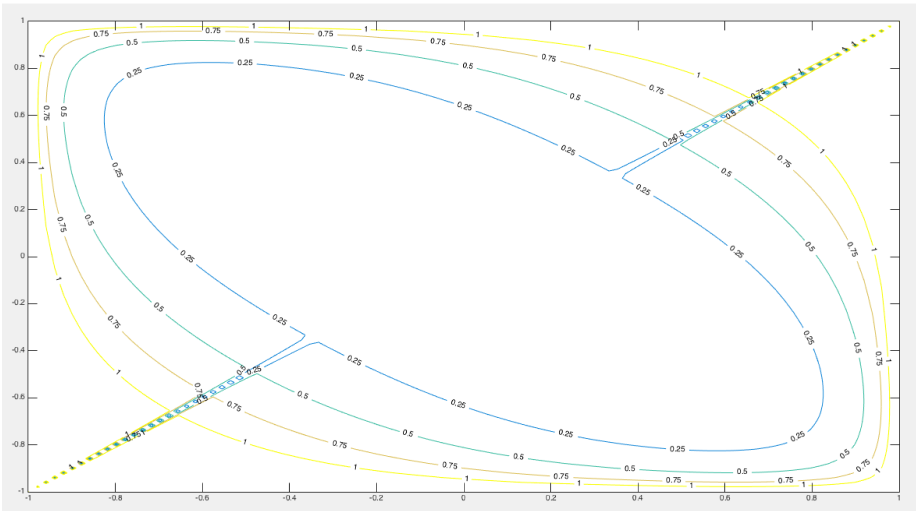
This is the whole of my code. Now I put the outputs of the system for different lambdas.

For lambda=1;

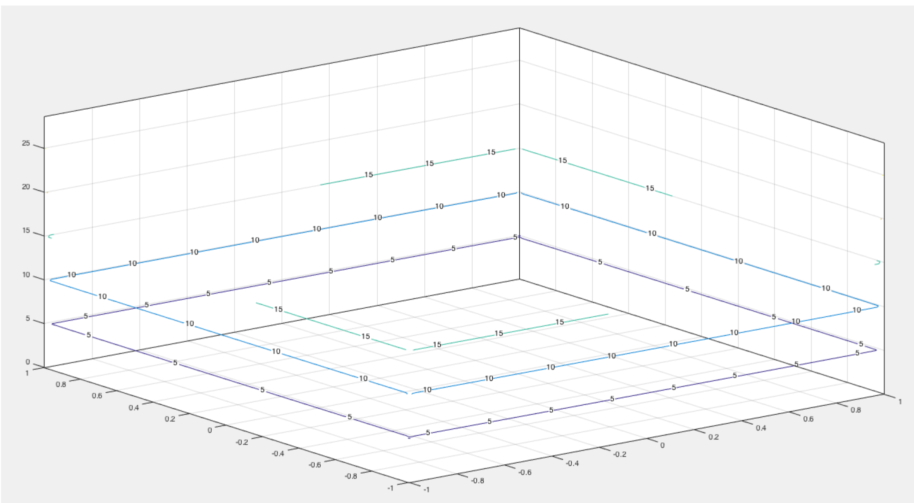
The red point is the initial random starting point. From here the trajectories goes to black equilibrium points.



In this graph stable equilibrium points are (0.063,-0.083) and (-0.083,0.063).



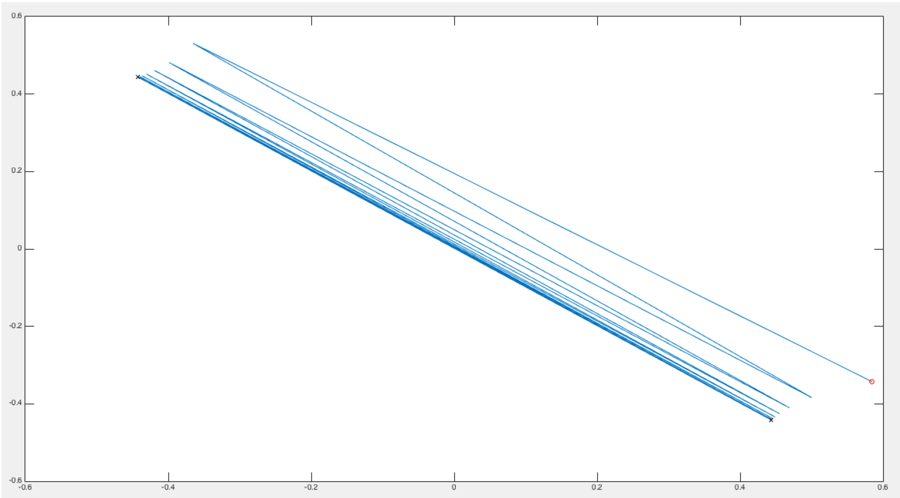
This is energy contour maps for this lambda.



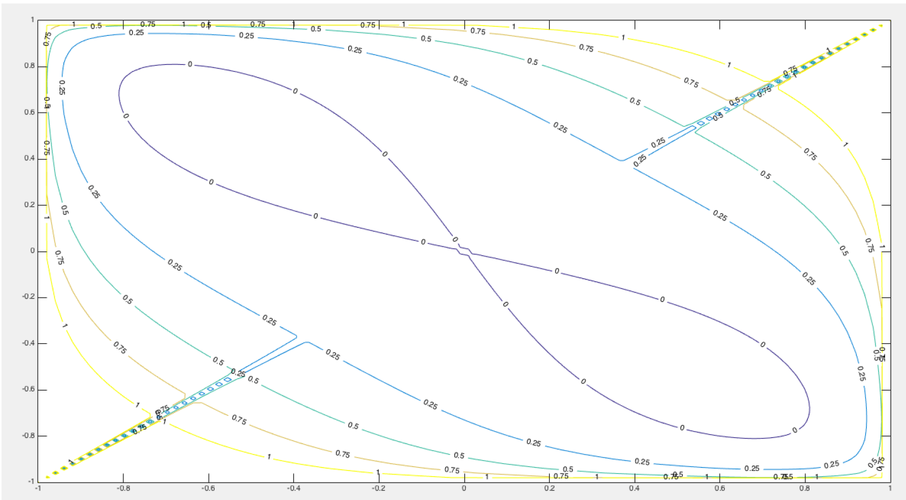
This is the 3D energy contour maps for this lambda.

For lambda=1.2;

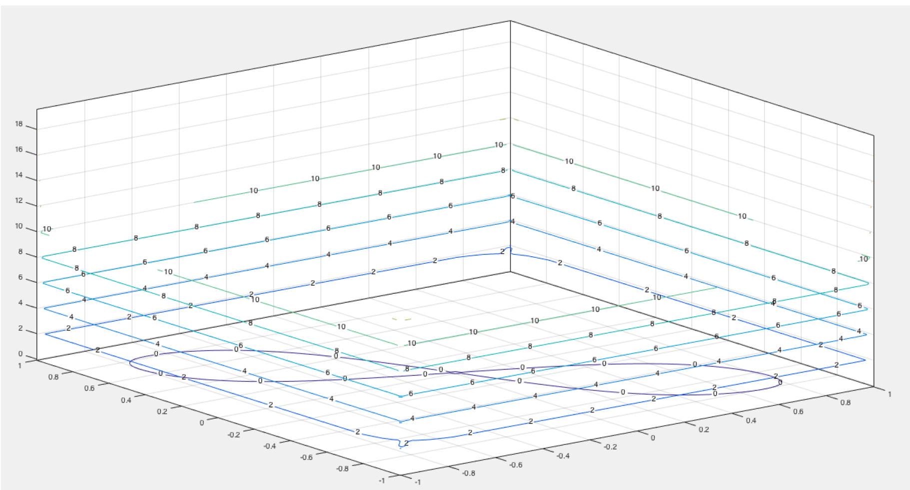
The red point is the initial random starting point. From here the trajectories goes to black equilibrium points.



In this graph stable equilibrium points are (-0.442,0.443) and (0.443,-0.442).



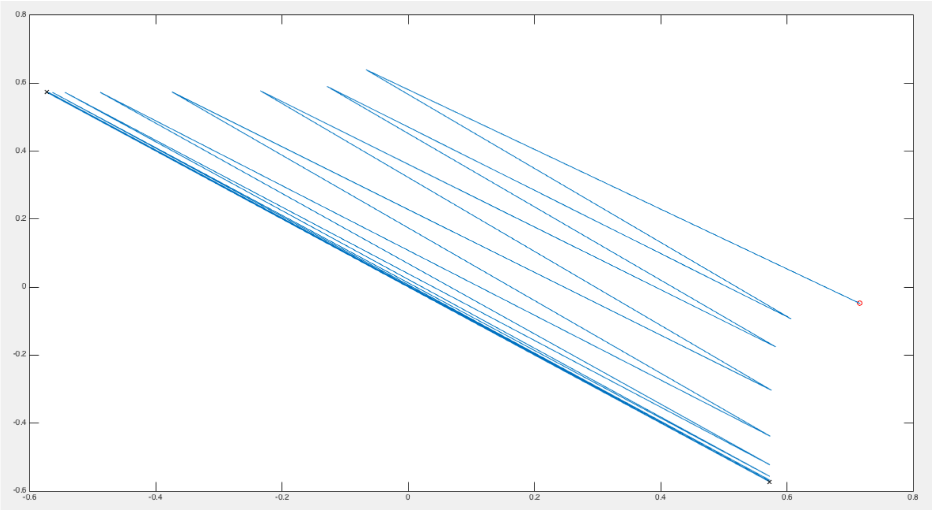
This energy contour for lambda=1.2.



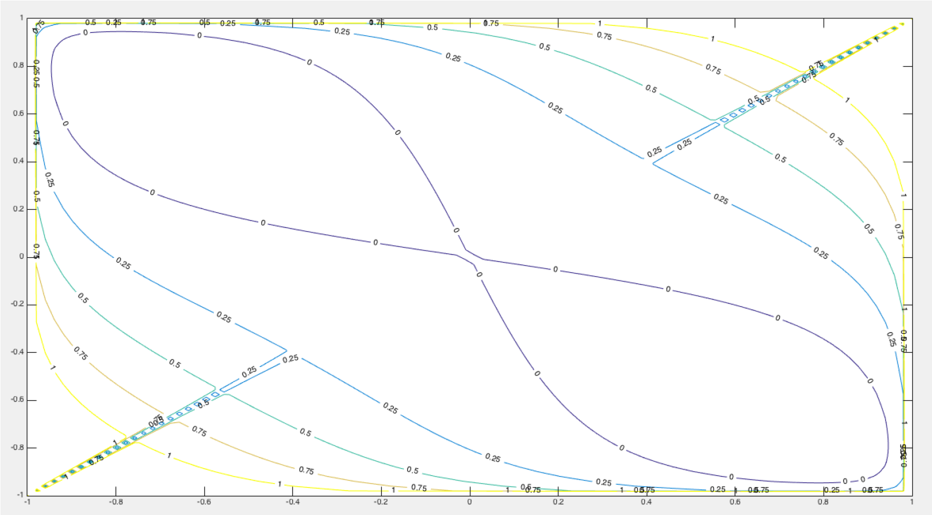
3D energy contour for the system with lambda=1.2

For lambda=1.4;

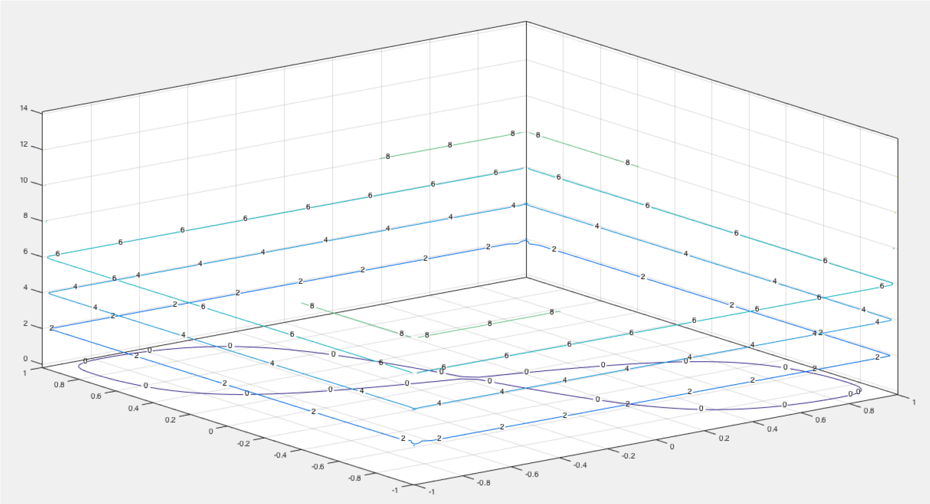
The red point is the initial random starting point. From here the trajectories goes to black equilibrium points.



In here stable equilibrium points are (0.573,-0.573) and (-0.573,0.573).



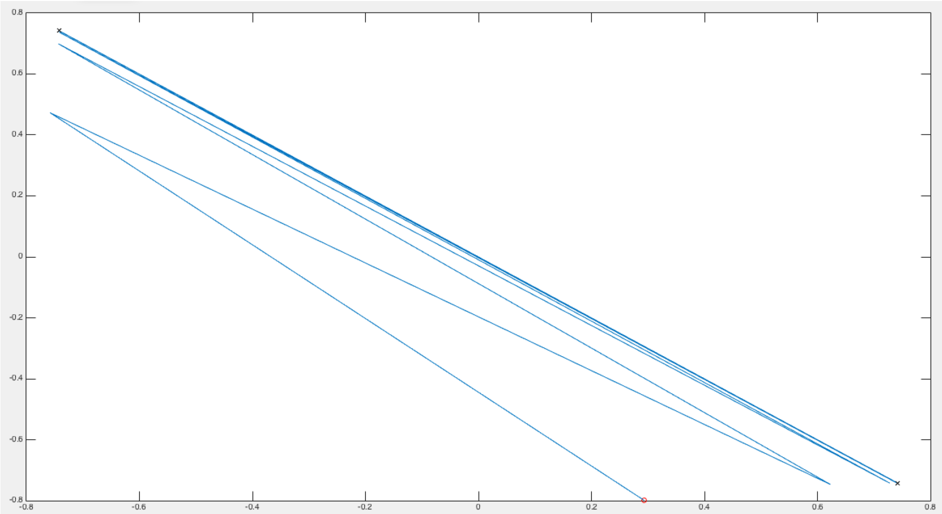
Energy contour maps for lambda=1.4.



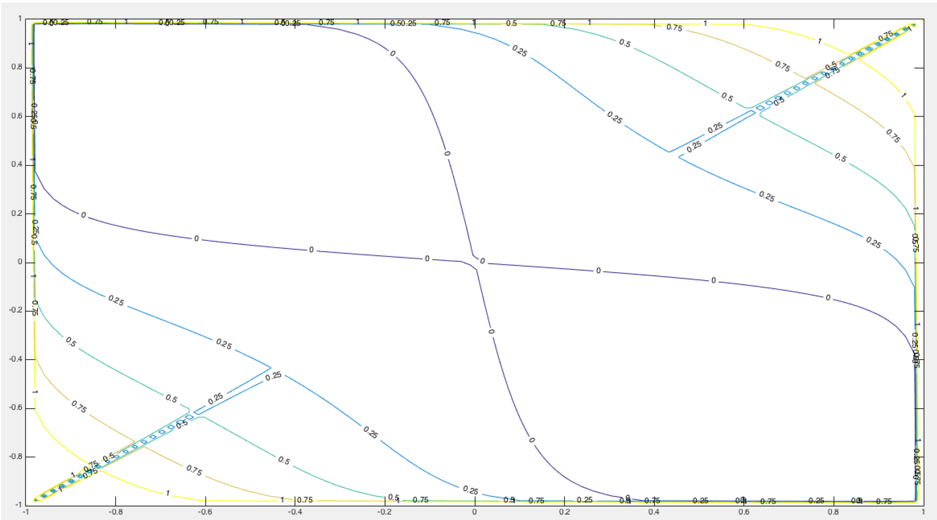
3D energy contour maps of system with lambda=1.4.

For lambda=2;

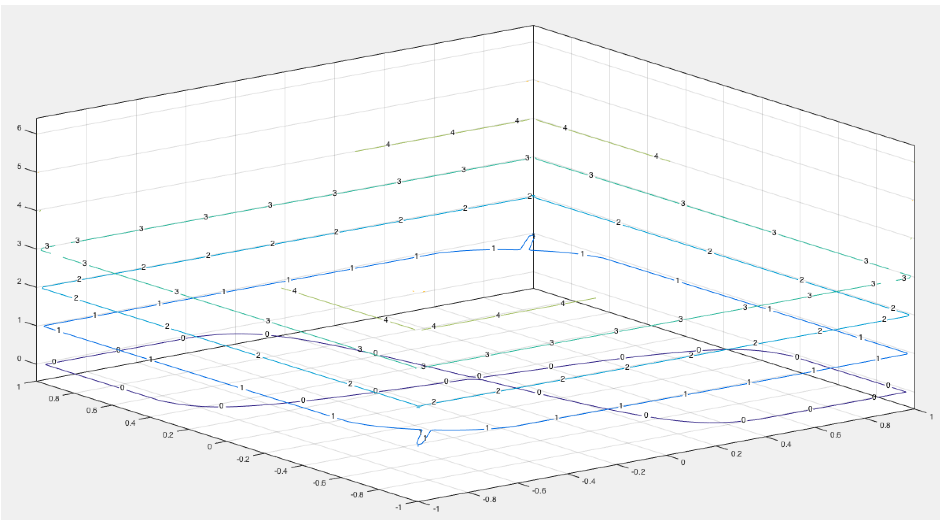
The red point is the initial random starting point. From here the trajectories goes to black equilibrium points.



The stable equilibrium points of this graph is that (0.742,-0.742) and (-0.742,0.742).



Energy contour of system for lambda=2.



3D energy contour of system.

We can see that when lambda is small our equilibrium points converges to zeros however when lambda is high our equilibrium points goes through the corners of the system.