Abstract

The report considers Peskin's model of pacemaker as network of N integrated and fire oscillators to study the synchronization observed in a network of neurons. Strogatz has proved that a fully connected network of neurons always synchronizes, We aim to study the robustness of networks which are not fully connected. We start with simulating probability of synchronization of a network of some particular number of neurons vs the fraction of synapses present. We then procure critical fractions from these graphs which form the basis of our next result. We observe that the critical fraction varies with number of neurons in the network in the beginning and then becomes saturated.

Contents

Chapter 1

Introduction:

The synchronization in general is termed as a process wherein different components of a system or of multiple systems reach to a common state. Like any kind of physicsl,chemical or bilogical oscillators, neurons can synchronize and exhibit a collective behavior that is not intrinsic to any individual neuron. Synchronization in neurons occurs when all the neurons start of with different initial conditions i.e different voltages, and as they evolve they reach at a common voltage, after they have synchronized they all have same voltage at a particular point. This can also be termed as a condition where all the neurons fire together.

We will be modelling neural network which closely resemble to pulse coupled oscillator network.In pulse-coupled oscillator (PCO) networks, an oscillator sends a pulse to its neighbors every time it fires. Each receiving oscillator experiences a discrete jump in its phase upon reception of the pulse. In case of neurons there is a jump in voltage value. The state of network varies continuosly with time until any oscillator receives a pulse.

The model used for neurons here is integrated and fire model. Subthreshold behavior of a neuron can be described by equation

$\mathring{V} = b - V$

where b is a constant, the value of V changes instantaneously to 0 when it crosses threshold value.

In this project the main focus was understanding the behaviour of networks of neurons when we start breaking the synapses in the network. I studied the tendency of a network to synchronize with fewer number of synapses. The report will discuss the effect on probability of synchronization of a network from the number of synapses present. It will furthur discuss how critical fraction of a network varies as we change the number of neurons. Critical fraction of a network is the fraction of synapses present vs total number of synapses in the beginning, at which we observe a sharp drop in probability of synchronization.

1.1 Model of neuron:

We will model the neurons according to following equation, which was proposed by Peskin.The neurons satisfy the following differential equation.

$$\frac{dv}{dt} = S - Lv$$

here S and L are positive constants and v is voltage of the neuron.

Solving this equation for v we get

$$v = \frac{S}{L} + (v_o - \frac{S}{L})exp(-Lt)$$

we see that the evolution is exponential with time constant $k = \frac{1}{L}$.

In this model break few synapses and then begin the evolution. We let all the neurons evolve till they reach a threshold value, once they do, then as the integrated and fire model prdicts the value of V becomes 0. Since all the neurons are coupled(i.e they aall are connected to each other with bidirectional synapses), pulse coupled to be precise, the firing neuron sends a pulse to all the other neurons, this causes a discrete and instantaneous change in their voltages. Over a period of time these neurons seem to synchronize, that is they have same voltages at all times.

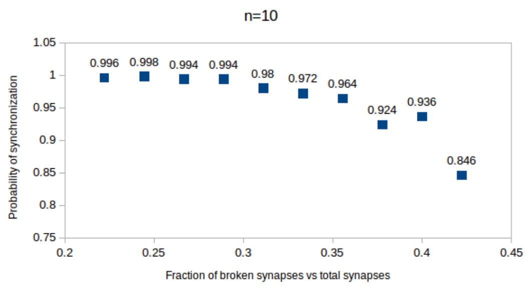
The neurons pass information to each other via synapses, these synapses are bidirectional, that means that if we have two neurons then they both can affect each other, if we break the synapse from first neuron to second then first neuron can not affect the second neuron however the second neuron will still affect the first neuron.We can break synapses between two neurons in 2 ways, either both of them together or just 1 at a time. We are more interested in breaking both the synapses and observing the behavior of the network.

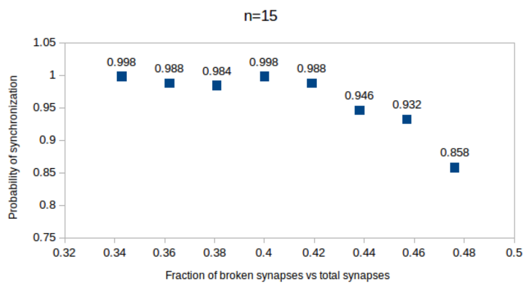
Chapter 2

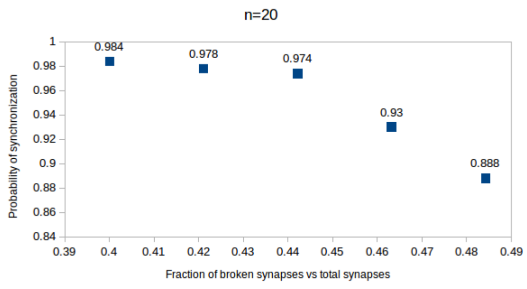
Results and analysis :

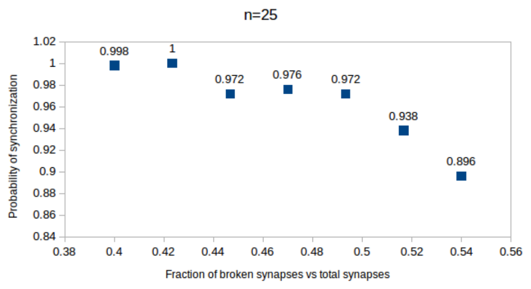
2.1 Relation between synchronization of a network of n neurons and synapses in the network

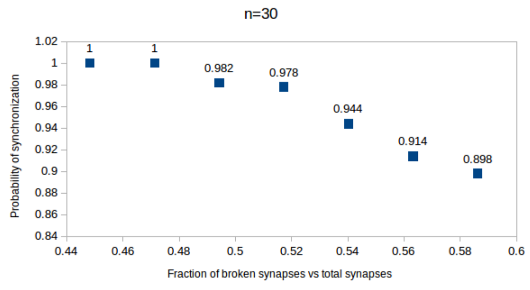
It can be shown that a network of n neurons has a total of $\frac{n(n+1)}{2}$ synapses. I chose different n and calculated the relationship between the synchronization of that network with number of synapses present. The basic idea is to keep breaking the synapses at random from a network till we achieve a significant fall in the probability of synchronization of a network. For practical purposes I have defined the significant fall to be 0.1 drop in probability from 1. The procedure for breaking synapses involves dividing the total number of synapses in 40 parts and then calculate the probability for each of the states. Due to constraints like processing power and time, we start from a particular fraction of synapses broken for each network, for most of the cases it is 0.5, please note that I have ommitted the calculations for fractions lower than this after making sure that the probability hovers very close to 1 for them, and since we are more interested in the critical fraction, we don't really need to focus on lower fractions. The simulations were done for 50 graphs and each of them had 10 different initial conditions. The figures below show the characterstics of each network of neurons.

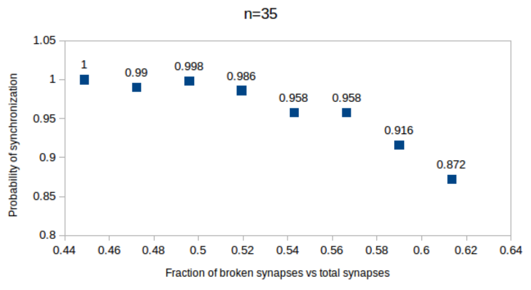

n=10
Probability of synchronization
0.996
0.998
0.994
0.994
0.98
0.972
0.964
0.924
0.936
0.846
Fraction of broken synapses vs total synapses

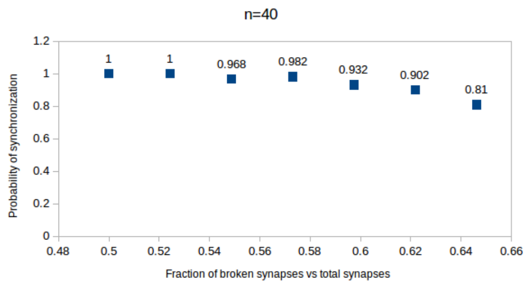

n=15
Probability of synchronization
0.998
0.988
0.984
0.998
0.988
0.946
0.932
0.858
Fraction of broken synapses vs total synapses

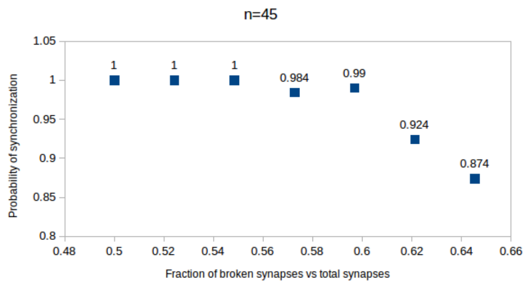

n=20
Probability of synchronization
0.984
0.978
0.974
0.93
0.888
Fraction of broken synapses vs total synapses

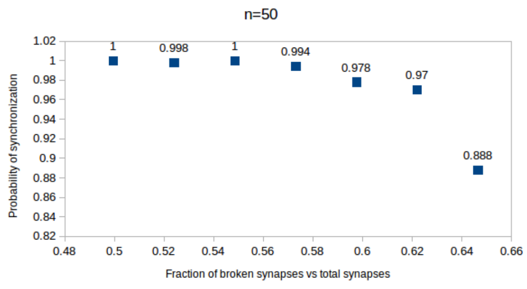

n=25
Probability of synchronization
0.998
1
0.972
0.976
0.972
0.938
0.896
Fraction of broken synapses vs total synapses

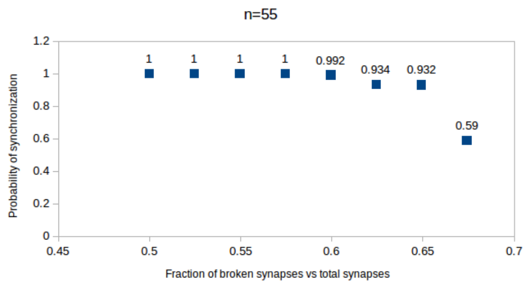

n=30
Probability of synchronization
1
1
0.982
0.978
0.944
0.914
0.898
Fraction of broken synapses vs total synapses

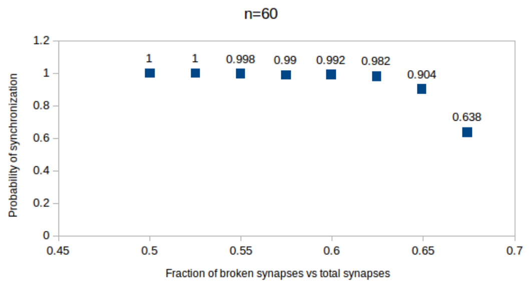

n=35
Probability of synchronization
1
0.99
0.998
0.986
0.958
0.958
0.916
0.872
Fraction of broken synapses vs total synapses

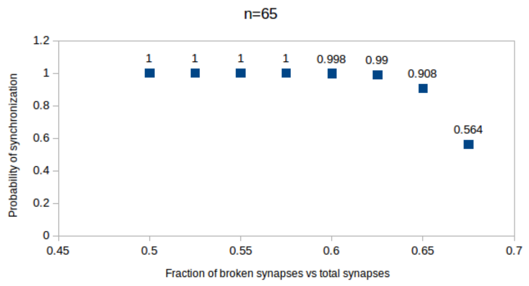

n=40
Probability of synchronization
1
1
0.968
0.982
0.932
0.902
0.81
Fraction of broken synapses vs total synapses

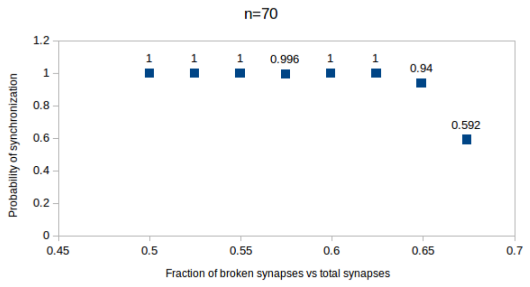

n=45
Probability of synchronization
1
1
1
0.984
0.99
0.924
0.874
Fraction of broken synapses vs total synapses

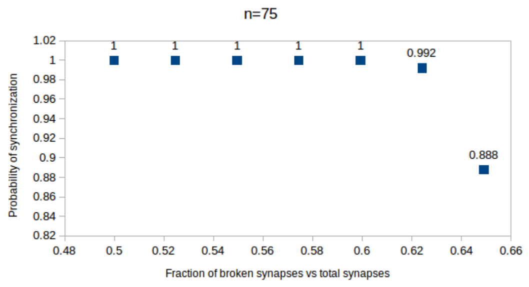

n=50
Probability of synchronization
1
0.998
1
0.994
0.978
0.97
0.888
Fraction of broken synapses vs total synapses

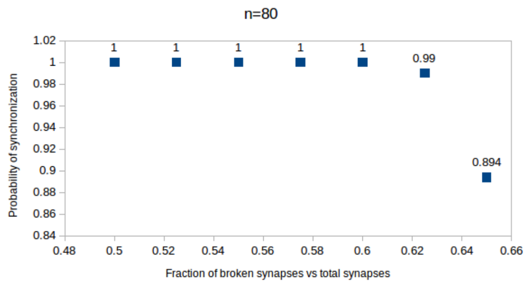

n=55
Probability of synchronization
1
1
1
1
0.992
0.934
0.932
0.59
Fraction of broken synapses vs total synapses

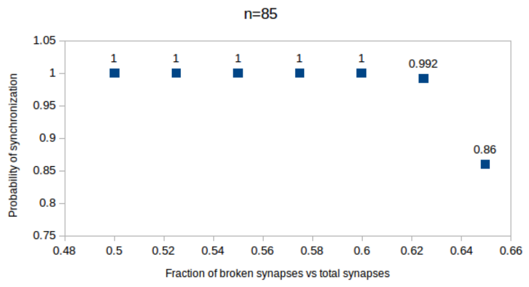

n=60
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
0.998
0.99
0.992
0.982
0.904
0.638

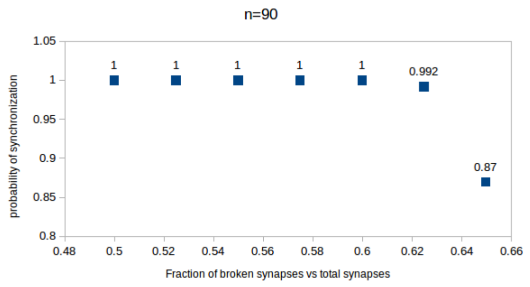

n=65
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.998
0.99
0.908
0.564

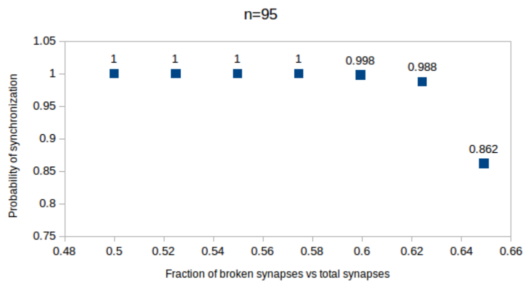

n=70
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
0.996
1
1
0.94
0.592

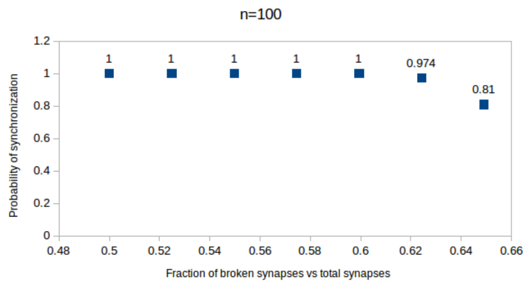

n=75
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
1
0.992
0.888

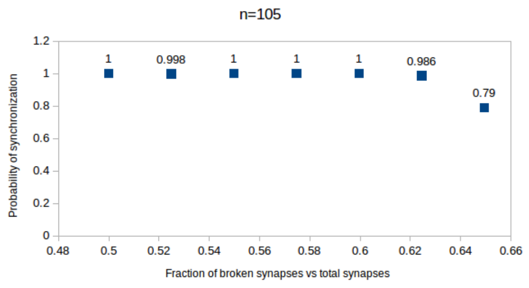

n=80
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
1
0.99
0.894

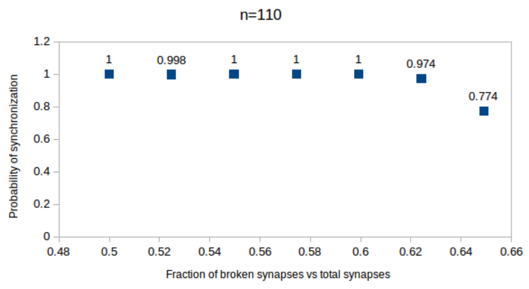

n=85
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
1
0.992
0.86

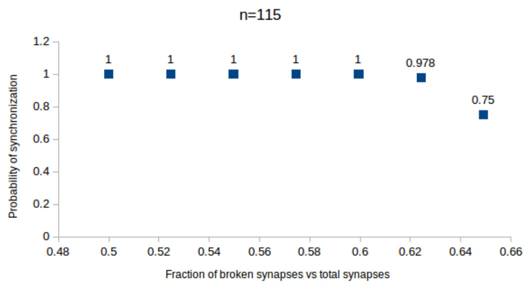

n=90
probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
1
0.992
0.87

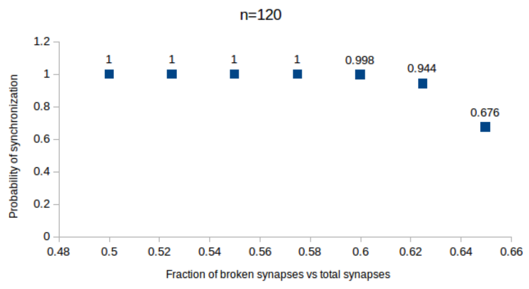

n=95
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.998
0.988
0.862

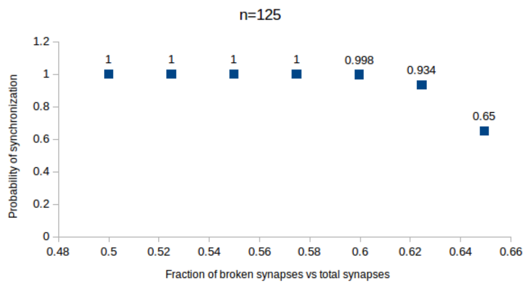

n=100
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
1
0.974
0.81

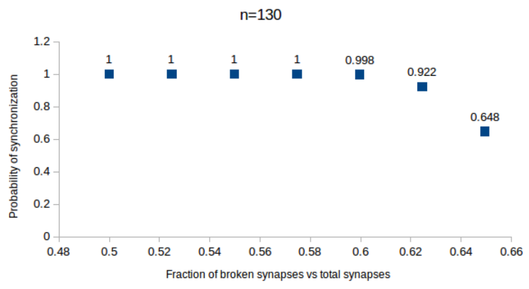

n=105
Probability of synchronization
Fraction of broken synapses vs total synapses
1
0.998
1
1
1
0.986
0.79

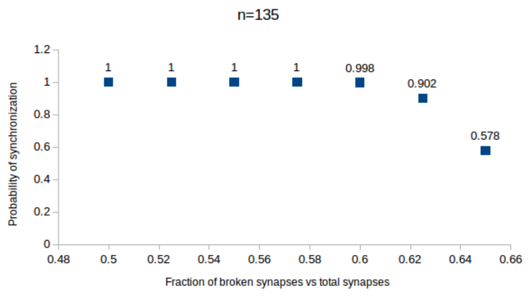

n=110
Probability of synchronization
Fraction of broken synapses vs total synapses
1
0.998
1
1
1
0.974
0.774

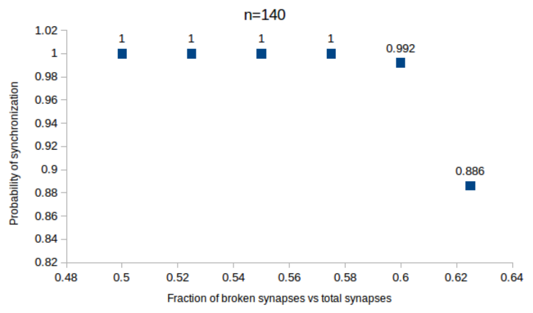

n=115
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
1
0.978
0.75


n=120
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.998
0.944
0.676


n=125
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.998
0.934
0.65


n=130
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.998
0.922
0.648


n=135
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.998
0.902
0.578


n=140
Probability of synchronization
Fraction of broken synapses vs total synapses
1
1
1
1
0.992
0.886

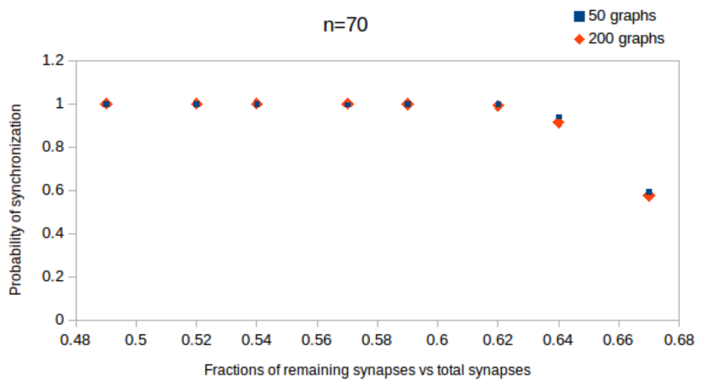


Figure 2.1:
Plot shows probability of synchronization on y axis and fraction of remaining synapses vs total synapses on x axis. The network has 70 neurons. Note that both the sets of data points are very close to each other, signifying low sensitivity on number of graphs.

2.2 The sensitivity of results on the number of graphs

As mentioned earlier the simulations for each neuron network was carried for 50 graphs, however we can not be sure if the mentioned number of graphs will be enough to achieve accurate results. Hence, in this section we will compare the sensitivity of results on the number of graphs, the comparision will be done between 50 and 200 graphs. Figure 2.1 shows the comparision for n=70 and figure 2.2 for n=50. We can conclude that there is not a significant difference in the results.

2.3 Critical fraction vs number of neurons

We earlier observed that critical fraction, i.e. the fraction of broken synapses vs total synapses at which there is sudden drop in probability , is different for different number of neurons. Figure 2.2 shows the plot of critical fraction vs number of neurons. We can see that the critical fraction becomes saturated at 0.6 for bigger values of n. The equation that fits the graph was found to be $f(x) = -1.165E - 09x^4 + 7.601E - 07x^3 - 0.15E - 03x^2 + 0.014x + 0.191$.

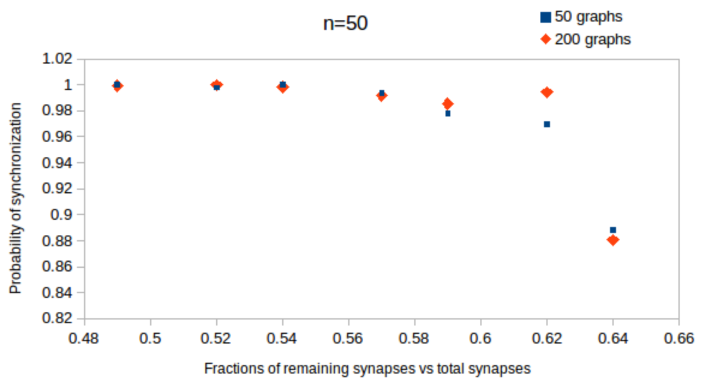


Figure 2.2:
Plot shows probability of synchronization on y axis and fraction of remaining synapses vs total synapses on x axis. The network has 50 neurons. Note that both the sets of data points are very close to each other,except at one point, signifying low sensitivity on number of graphs.

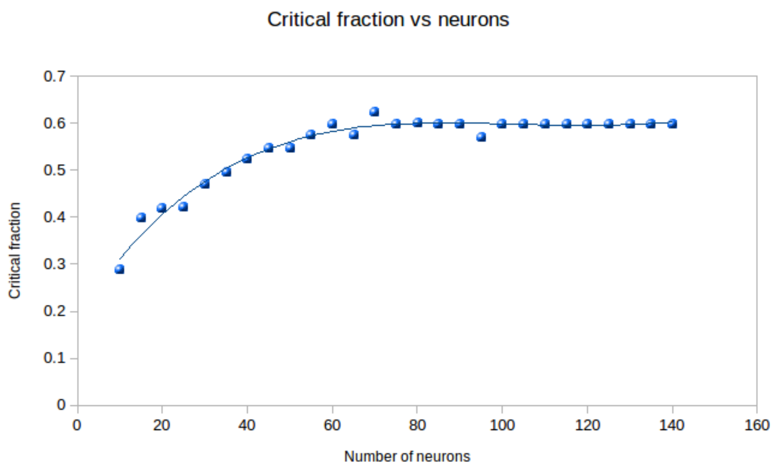


Figure 2.3:
Plot of critical fraction vs Number of neurons in a network. The saturation can be clearly observed as we increase the number of neurons.

Chapter 3

Discussions

We have seen that number of neurons in a network have an effect on the probability of synchronization of that network, we have also discussed the behavior of a network near critical fraction. Note that the simulations had some constraints due to limited computation power. However, it has been shown that sensitivity of results on number of graphs for each simulation is low.

We have also seen that the critical fraction seems to saturate in the data set. This can have important implications in brain study. Local synchronization of oscillatory neuronal behavior in cortical networks plays a fundamental role in many aspects of perception and cognition,We know that synaptic pruning happens in our brain, and it does not necessarily effect our mental health, however there are some mental conditions which lead to extreme loss in number of synapses. We might be able to determine the safe amount of loss of synapses, and predict the onset of some conditions.

Furthur works may include studying a different model of a neuron and observing the difference between the results, a good starting point will be studying leaky integrated and fire model, wherein the input is not constant. We can also study the characteristics of a graph which makes it more robust than the graph with same number of connections but different structure, we can calculate individual probability of synchronization for each such graph(with some particular number of initial cases), for some particular number of synapses in the network,and observe which of these graphs shows most robustness. The implication of such results can be very interesting and important.

It must be noted that our results are heavily dependent on the computation power we could use. These results may be furthur explored and scrutinized by increasing the time for simulation.

Chapter 4

Appendix

The simulations were done in c++, the program for simulation is attached below.

```
#include<bits/stdc++.h>
#include<ctime>
#include<random>
using namespace std;
const int n=35;    //number of neurons
float S=1.000;
float L=1.00;
float h=0.002;    //time steps float
T=1000.00;
float thresh=0.80 // value above which the particle
jumps assuming the particle will jump
float excite=0.01;     //excitation in each particle due
to one particle crossing threshold value float error =0.001;
int arr[n+1][n+1];          //mapping array
void iterate(float pos[][2],float pos_ini[][2],
vector<vector<float > > &spike_time);
int is_sync(float pos[][2],float pos_ini[][2],
int num,ofstream &myfile,vector<vector<float > > &spike_time);
void map_neuron(float pos[][2],float pos_ini[][2],
int num,ofstream& myfile1);
void run(float pos[][2],float pos_ini[][2],ofstream &myfile,
vector<vector<float > > &spike_time);
float time(float v,float v_o);
void rungekutta(int i,float p[][2]);
void rungekutta(int i,float pos[][2])
{
  float k1,k2,k3,k4,k5;
  float v=pos[i][0];
  k1=h*(S-L*v);
  k2=h*(S-L*(v+0.5*k1));
  k3=h*(S-L*(v+0.5*k2));
  k4=h*(S-L*(v+k3));
  pos[i][0]+=(1.0/6.0)*(k1+2*k2+2*k3+k4);
  pos[i][1]+=h;
}
void iterate(float pos[][2],float pos_ini[][2],
vector<vector<float > > &spike_time)
{
        int L[n+1]={0};
        for(int i=0;i<n;i++)
        {
      rungekutta(i,pos);
        }
for(int i=0;i<n;i++)
{
if(pos[i][0]>=thresh)
    L[i]=1;    }
for(int i=0;i<n;i++)
{      int cnt=0;
  for(int j=0;j<n;j++)
   {
 if(i==j)
    continue;
  else
{
if(L[j]==1 && arr[i][j]==0)
        {
        cnt++;
    }
}
}
if(pos[i][0]+cnt*excite>=thresh )
      {
     pos[i][0]=0.0;
     spike_time[i].push_back(pos[i][1]);
      }
else
    {
          pos[i][0]+=cnt*excite ;
    }
  }
}
int is_sync(float pos[][2],float pos_ini[][2],
int num,ofstream &myfile,vector<vector<float > > &spike_time)
```

```
{
        run(pos,pos_ini,myfile,spike_time);
        int count=0;
 for(int i=1;i<n;i++)
          {          if(fabs(pos[i][0]-pos[0][0])>error)
            {
                         count++;
                 }
          }
            if(count==0)
        {
                         return 1;
          }
            else
            return 0;
}
void map_neuron(float pos[][2],int num,ofstream &myfile1)
{    std::random_device rd;
   std::mt19937 gen(rd());
  std::uniform_int_distribution<> dis(0,(n-1));
 for(int i=0;i<n;i++)
 {
        for(int j=0;j<n;j++)
                arr[i][j]=0;
 }
  for(int i=0;i<num;i++)
  {
   while(1)
    {
      int v1= dis(gen);
      int v2= dis(gen);
      if(arr[v1][v2]==1)
        continue;
      if(v1!=v2)
      {
       arr[v1][v2]=1;
      arr[v2][v1]=1; //condition for second case
        break;
      }
    }
  }
  for(int i=0;i<n;i++)
  {
    for(int j=0;j<n;j++)
    {
      char s=arr[i][j]+'0';
      myfile1<<s<<" ";
     }
     myfile1<<"\n";
   }
   myfile1<<"\n";
}
void run(float pos[][2],float pos_ini[][2],ofstream  &myfile,
vector<vector<float > > &spike_time)
{    int count=0;
   int times=T/h;
   int i=0;
   while(times--)
   {
     iterate(pos,pos_ini,spike_time);
   }
}
void generate_rand(float pos[][2],float pos_ini[][2])
{
        for(int i=0;i<n;i++)
   {
     pos[i][0]= static_cast <float> (rand()) /
     (static_cast <float> (RAND_MAX/thresh));
     pos[i][1]= 0.0;
     pos_ini[i][0]=pos[i][0];
     pos_ini[i][1]=0.0;
   }
}
int main()
{
        float pos[201][2];
     float pos_ini[201][2];
     ofstream myfile,myfile1,spike;
     myfile.open("endpositions.txt");
     myfile1.open("map.txt");
     spike.open("spike_time.txt");
     int seed;
     cin>>seed;
     srand(seed);

     int k=0;
     int incre= (n*(n-1))/80;

     int N1=50;            // no of graphs generated
     int N2=10;
     myfile1<<"neurons: "<<to_string(n);
     myfile1<<"\n";
```

```
    spike<<"neurons: "<<to_string(n);
    spike<<"\n";


   int start=(n*(n-1)*50)/200;
for(int i1=start;2*i1<=(n*(n-1));i1+=incre)
  {
      float p=0.0;
      int sum=0;
      myfile1<<"Synapses broken: "<<to_string(i1);
      myfile1<<"\n";
      spike<<"Synapses broken: "<<to_string(i1);
      spike<<"\n";
    for(int i=0;i<N1;i++)
         {
            map_neuron(pos,i1,myfile1);
            for(int j=0;j<N2;j++)
            {
              vector<vector<float > > spike_time;
              for(int counter=0;counter<n;counter++)
              {
               vector<float > v;
                spike_time.push_back(v);
              }
              generate_rand(pos,pos_ini);
              sum+=is_sync(pos,pos_ini,i1,myfile,spike_time);
              int spikeI=spike_time[0].size();
              if(spikeI <10)
              {
                for(int ii1=0;ii1<n;ii1++)
                {
                  for(int jj1=0;jj1<spikeI;jj1++)
                  {
                    spike<<spike_time[ii1][jj1]<<" ";
                  }
                  spike<<"\n";
                }
              }
              else
             {
                for(int ii1=0;ii1<n;ii1++)
                {
                 for(int jj1=spikeI-10;jj1<spikeI;jj1++)
                  {
                    spike<<spike_time[ii1][jj1]<<" ";
                  }
```

```
                    spike<<"\n";
                  }
                }
              spike<<"\n";
            }
          }
        float fraction = (float)(2*i1)/(n*(n-1));
           p= (float)sum/(N1*N2);
          cout<<fraction<<" "<<p<<endl;
         if(fabs(1-p)>0.1)
          {
            //cout<<n<<" "<<fraction;
            break;
          }
}
myfile.close();
myfile1.close();
spike.close();

}
```

Bibliography

[1] Renato E. Mirollo; Steven H. Strogatz,*Synchronization of Pulse-Coupled Biological Oscillators. SIAM Journal on Applied Mathematics, Vol. 50, No. 6. (Dec., 1990), pp. 1645-1662.*

[2] Izhikevich, Eugene M. Dynamical Systems in Neuroscience: the Geometry of Excitability and Bursting. MIT Press, 2014.

[3] *Strogatz Steven S.,Nonlinear Dynamics and Chaos. Perseus Books, 1994.*

[4] www.scholarpedia.org/article/Coherent_activity_in_excitatory_pulse-coupled_networks.