Computational Neuroscience Project 3

Adarsh Mukesh 13BT30031
Siddharth S Jha 14EE30022

All questions were attempted by both of us. The brackets denote the primary person responsible for the submission.

Question 1: Gaussian/Non Gaussian (Siddharth)
It was seen that the resultant figure of the autocorrelation looked quite much like a dirac-delta function, and the graph was as in Figure 1. There was some non ideality which was seen later in the question no 4. The peak of the graph is at around 0.33 which is also the variance of the data, confirmed using var. Hence it is quite gaussian.

Question 2: PSTH (Siddharth)
The PSTH was evaluated with bin size 1ms. The corresponding plot is shown in figure 2. The PSTH for the test and train data was kept in separate variables.

Question 3: Poisson or Non Poisson (Adarsh)
Poisson or non-Poisson was shown using plots between mean and variance. Plots are Figures 3-8 for six different bin sizes.

Poisson nature:
<u>Figure 4:</u> All 4 poisson
<u>Figure 5:</u> Neuron 2,3 poisson
<u>Figure 6:</u> Only 3 is poisson
<u>Figure 7:</u> Only 3 is poisson
<u>Figure 8:</u> None poisson
<u>Figure 9:</u> None poisson

So as we keep increasing the bin size, the poisson nature of distributions keep reducing

Question 4: STA and h(t) (Adarsh)
The STA is shown in figure 10. h(t) is shown in figure 11.

In Figure 1 we see that the distribution is only nearly gaussian, and hence we need to add a correction term. h(t) is determined by multiplying a corrective factor Css^{-1} which is the inverse of the autocorrelation matrix.

Question 5: Nonlinearity (Siddharth)
The scatter plot between the convolution of h(t) with stimulus and the PSTH is given in Figure 12. The nonlinearity is determined by fitting a sigmoid function in the scatter plot. The fitting part was not working well.

Question 6: Pruning (Adarsh)

Question 7B: Victor Purpura (Adarsh & Siddharth)

The plot between the mutual information and the q values for four neurons is in figure 12. It is therefore seen that the information is maximum for middle values of q (0.1 to 10).
The Victor-Purpura distance algorithm was used from the Cornell website (originally written by Daniel Reich).

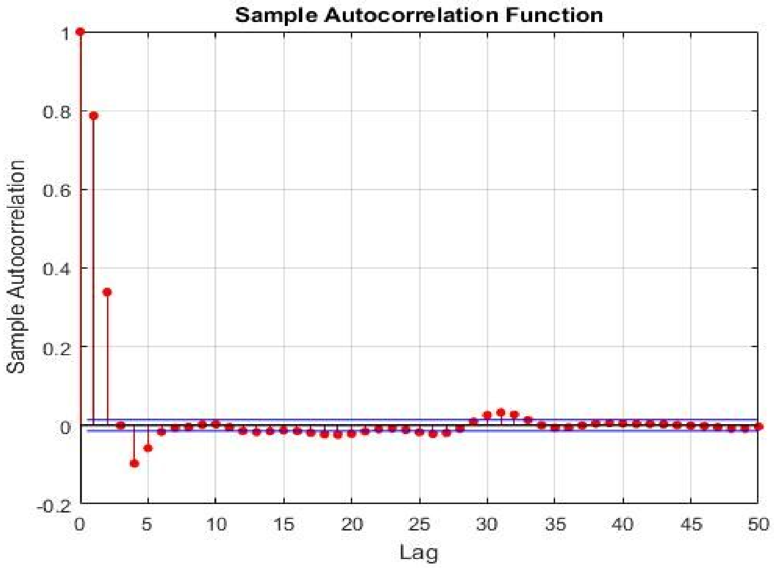


Figure 1: Autocorrelation of Stimulus

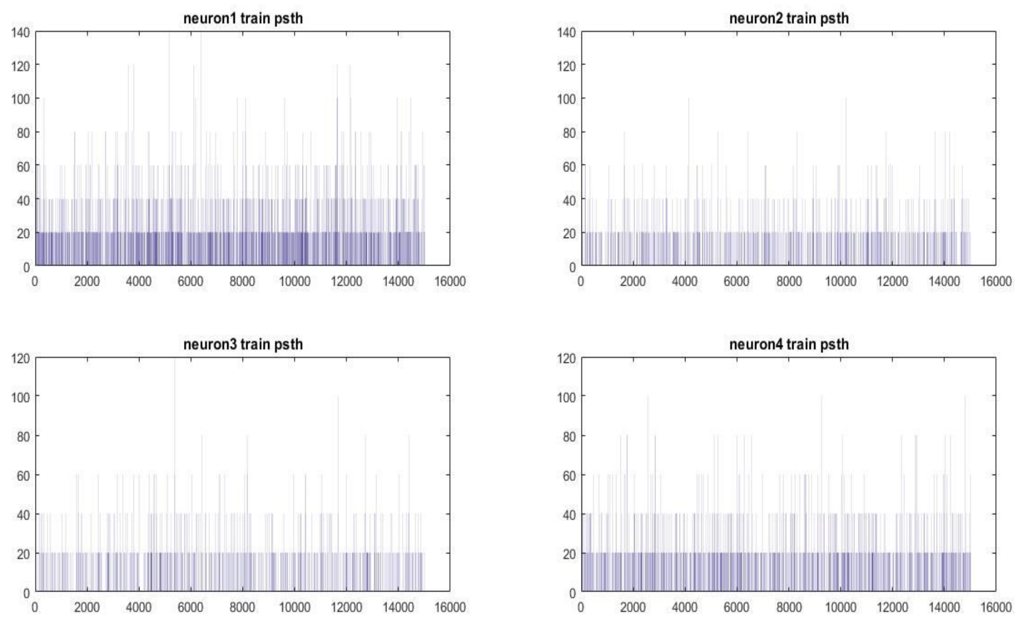


Figure 2: PSTH of Stimulus for first 15 seconds, 4 neurons

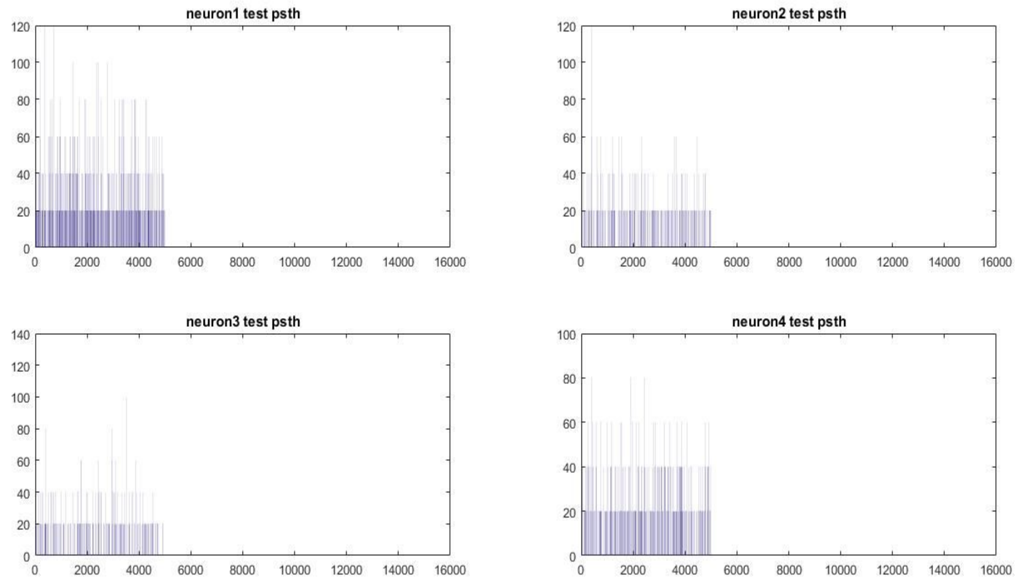


Figure 3: PSTH of Stimulus for last 5 seconds, 4 neurons

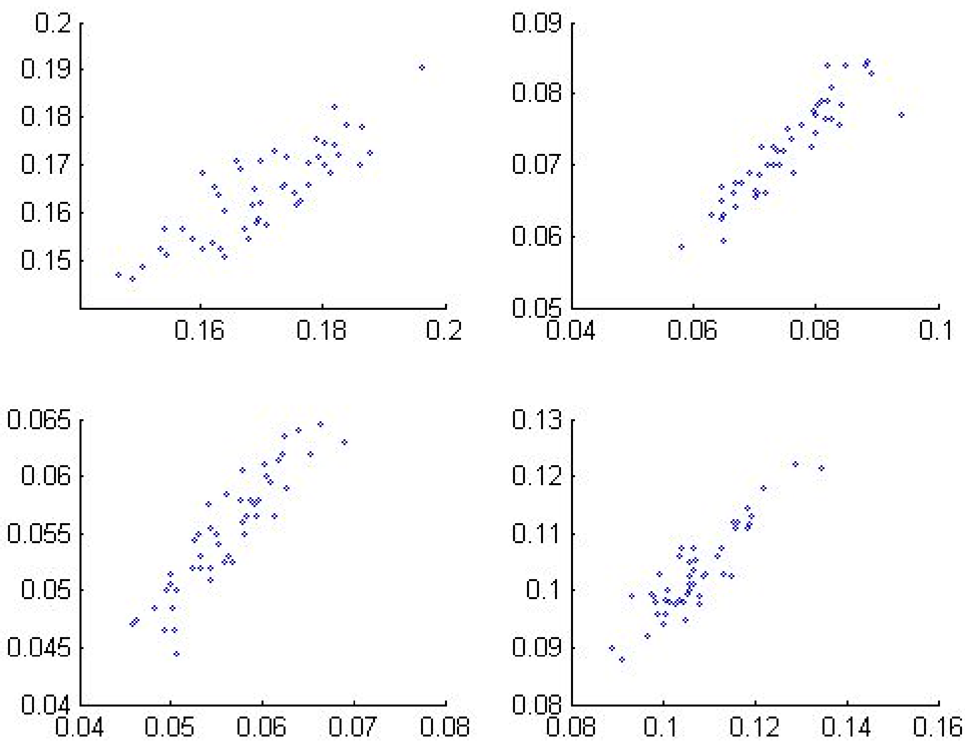


Figure 4: Scatter plot between mean and variance for binsize=10ms

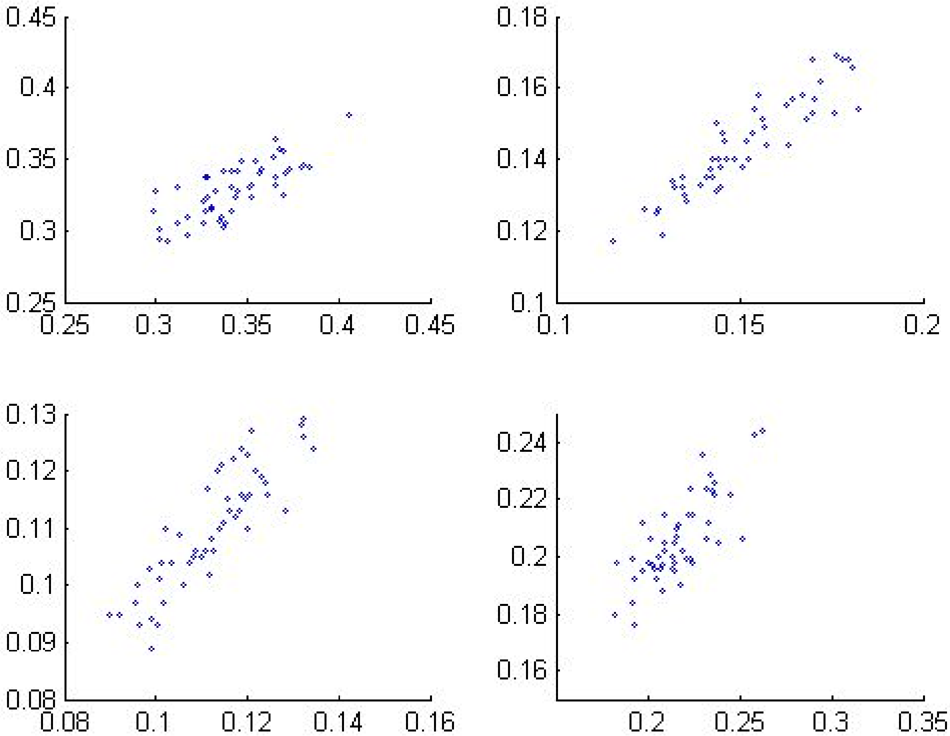


Figure 5: Scatter plot between mean and variance for binsize=20ms

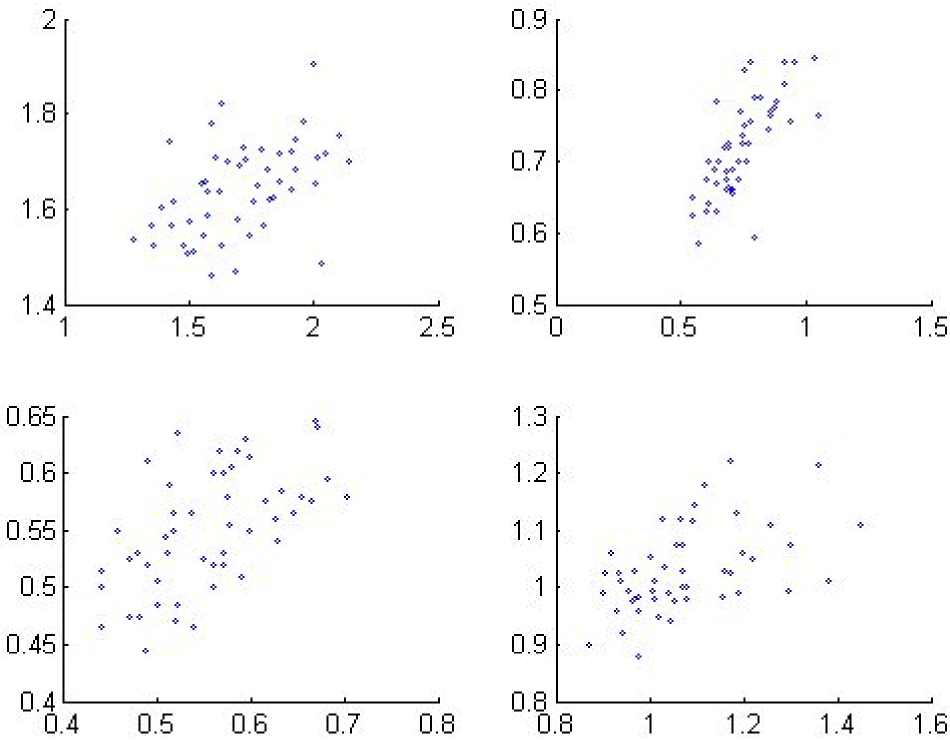


Figure 6: Scatter plot between mean and variance for binsize=50ms

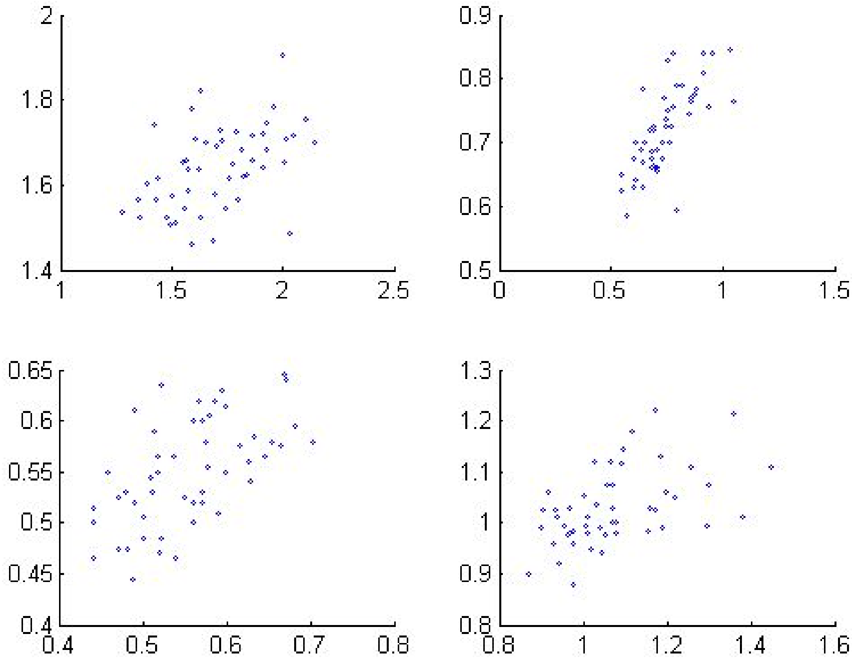


Figure 7: Scatter plot between mean and variance for binsize=100ms

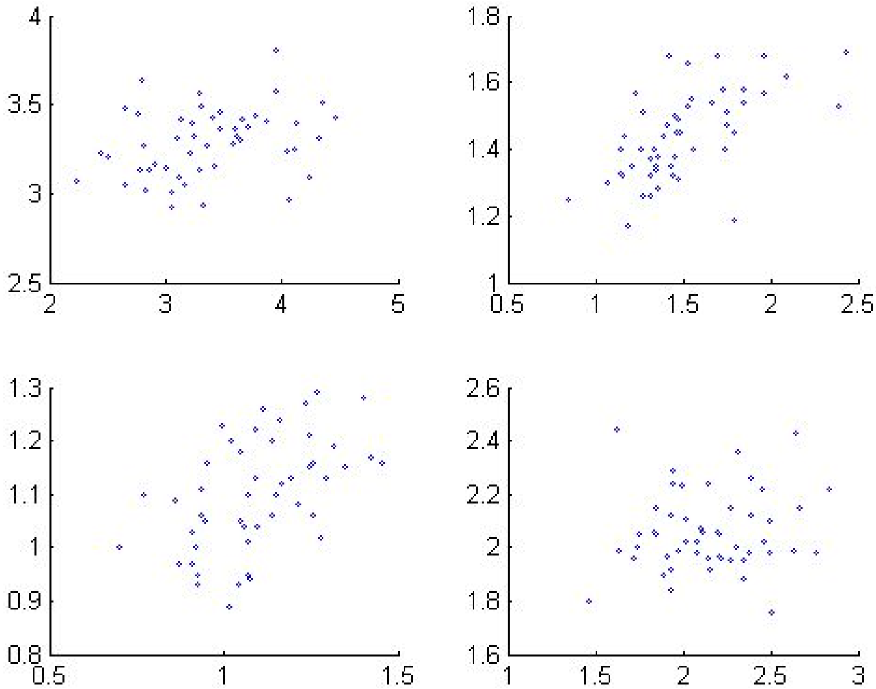


Figure 8: Scatter plot between mean and variance for binsize=200ms

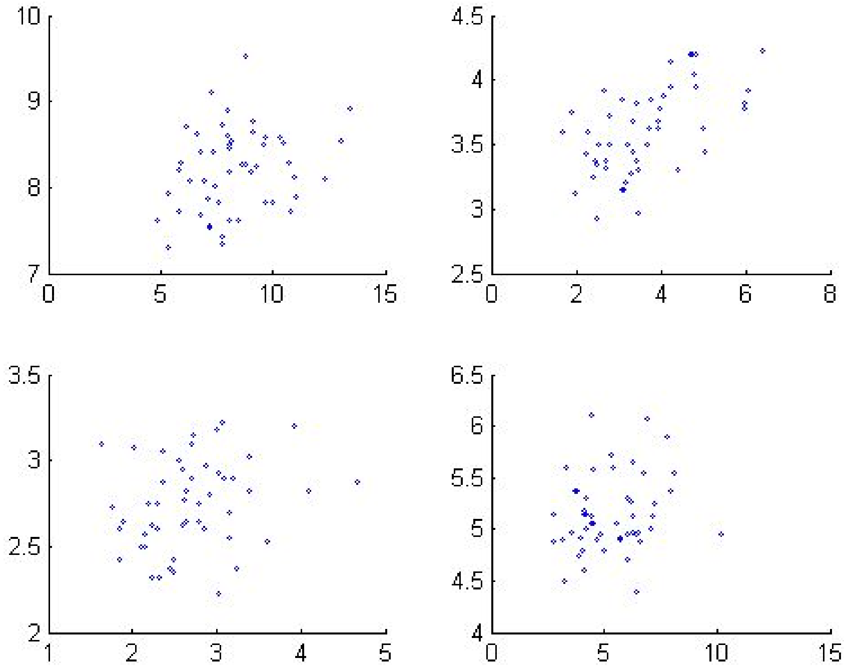


Figure 9: Scatter plot between mean and variance for binsize=500ms

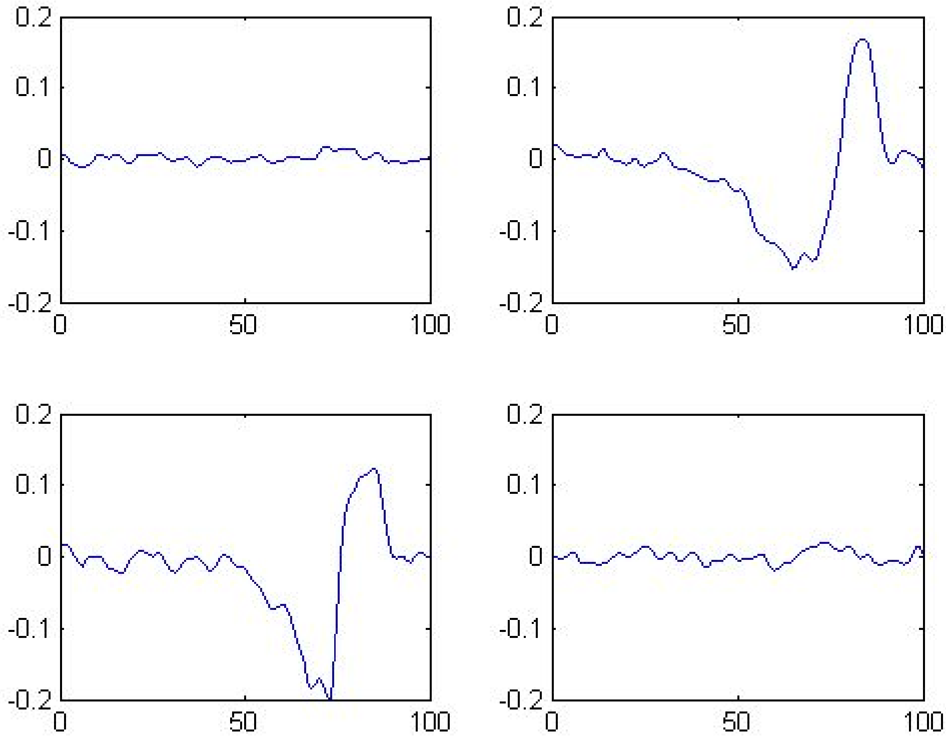
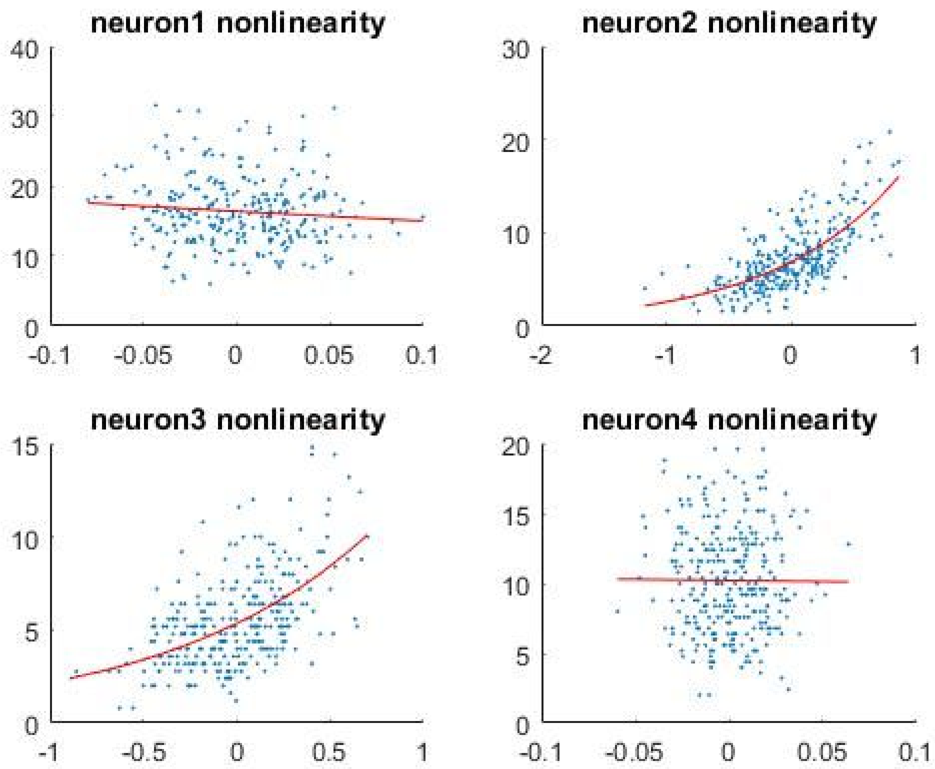


Figure 10: Spike Triggered Average for four neurons

Figure 11: Whitened STA



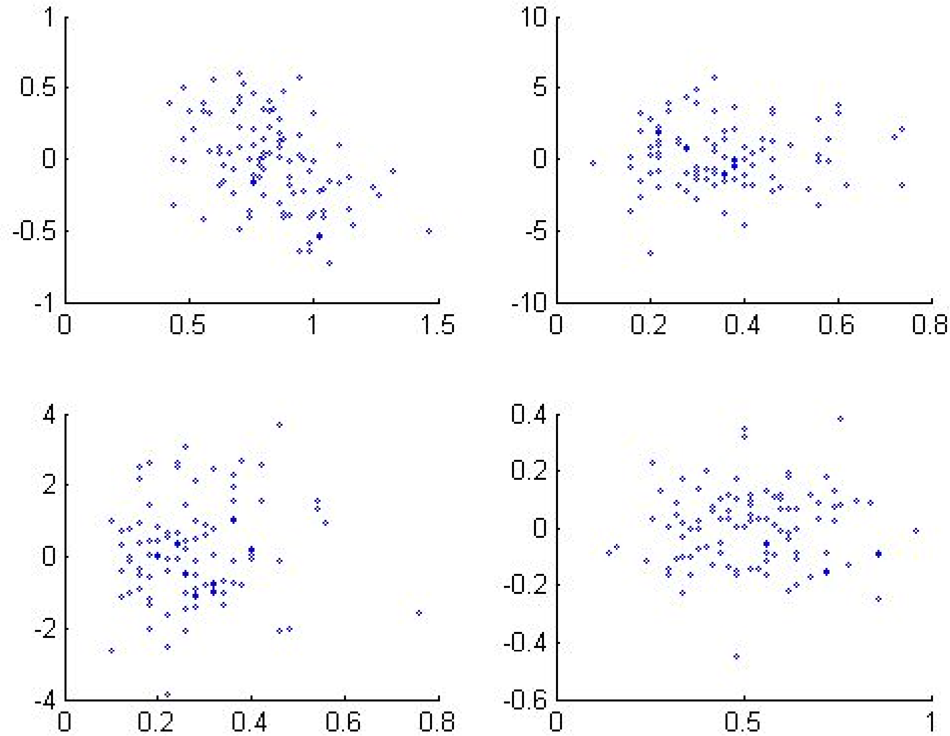


Figure 12: Scatter plot between PSTH and Predicted rate

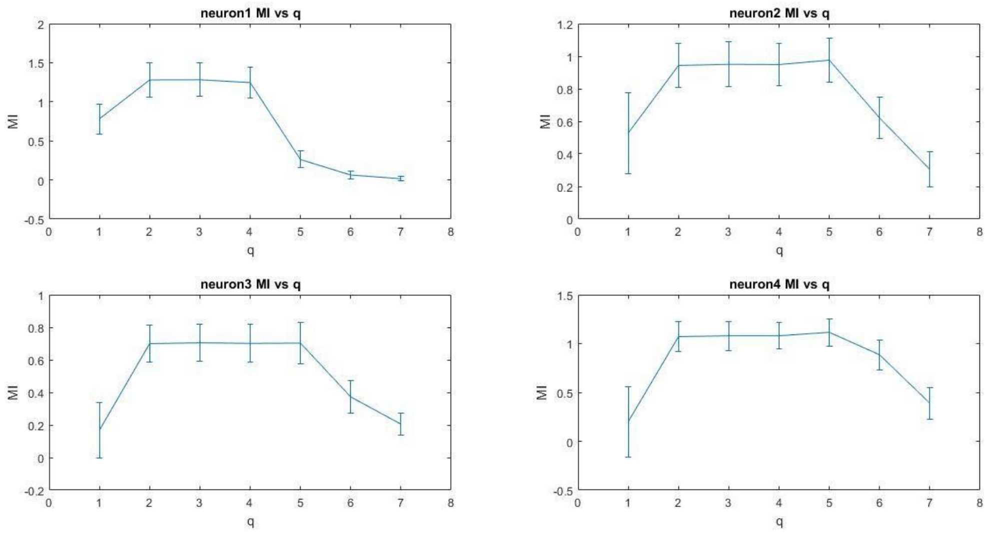


Figure 13: Mutual Information using Victor-Purpura SDM

Code:

https://github.com/thesidjway/Computational-Neuroscience

```
%% autocorr
load('CN_Project3_2016.mat');
```

```
trn_stm=Stimulus(1,1:15000);
test_stm=Stimulus(1,15001:end);
for i=1:4
   for j=1:50
      trn_spk{i,j}=[];
      test_spk{i,j}=[];
      a=All_Spike_Times{i,j};
      for k=1:length(a)
         if a(1,k)<15
            trn_spk{i,j}=[trn_spk{i,j},a(1,k)];
         else
            test_spk{i,j}=[test_spk{i,j},a(1,k)-15];
         end
      end
   end
end
figure
autocorr(Stimulus,50);
%% PSTH
figure
binsz=0.05;
trn_psth=zeros(4,15/binsz);
for n_n=1:4
   for i=1:50
      a=trn_spk{n_n,i};
      for j=1:length(a)
         trn_psth(n_n,fix(a(j)/binsz)+1)=trn_psth(n_n,fix(a(j)/binsz)+1)+1;
      end
   end
   trn_psth(n_n,:)=trn_psth(n_n,:)./(50*binsz);
   subplot(2,2,n_n);
   bar(trn_psth(n_n,:));
   str=strcat('neuron ',num2str(n_n),' train psth');
   title(str);
end

binsz=0.05;
test_psth=zeros(4,5/binsz);
figure
for n_n=1:4
   for i=1:50
      a=test_spk{n_n,i};
      for j=1:length(a)
         test_psth(n_n,fix(a(j)/binsz)+1)=test_psth(n_n,fix(a(j)/binsz)+1)+1;
      end
   end
```

```
    test_psth(n_n,:)=test_psth(n_n,:)./(50*binsz);
    subplot(2,2,n_n);
    bar(test_psth(n_n,:));
    str=strcat('neuron ',num2str(n_n),' test psth');
    title(str);
end
%% Poisson or non poisson
binsz=[0.01,0.02,0.05,0.1,0.2,0.5];
for i=1:length(binsz)
    figure
    for n_n=1:4
        poss{n_n,i}=zeros(50,20/binsz(i));
        for j=1:50
            a=All_Spike_Times{n_n,j};
            for ii=1:length(a)
                poss{n_n,i}(j,fix(a(ii)/binsz(i))+1)=poss{n_n,i}(j,fix(a(ii)/binsz(i))+1)+1;
            end
        end
        poss{n_n,i}=poss{n_n,i}./binsz(i);
        subplot(2,2,n_n)
        scatter(var(poss{n_n,i},0,2),mean(poss{n_n,i},2),5);
        str=strcat('neuron ',num2str(n_n));
        title(str);
    end
end
%% STA
stim=[zeros(1,99), trn_stm];
sta=zeros(4,100);
figure
for n_n=1:4
    spk=zeros(50,100);
    filt=zeros(1,100);
    k=0;
    for i=1:50
        a=trn_spk{n_n,i};
        for j=1:length(a)
            if fix(a(j)/0.001)>0
                filt=filt+stim(fix(a(j)/0.001):fix(a(j)/0.001)+99);
                k=k+1;
            end
        end

    end

    sta(n_n,:)=filt./k;
    subplot(2,2,n_n);
```

```
    plot(sta(n_n,:));
    ylim([-0.2 0.2]);
    str=strcat('STA neuron',num2str(n_n));
    title(str);
end
x=[];
for i=1:15000
    x=[x;stim(i:i+99)];
end
css=x'*x;
css=css./15000;
figure
for n_n=1:4
    sta_w(n_n,:)=(css^-1)*sta(n_n,:)';
    subplot(2,2,n_n)
    plot(sta_w(n_n,:));
    str=strcat('whitened STA neuron',num2str(n_n));
    title(str);
end
%% nonlinearity
param(1,:)=[16.38,-0.8765];
param(2,:)=[6.886 0.9832];
param(3,:)=[5.372 0.9023];
param(4,:)=[10.22 -0.154];
figure
for n_n=1:4
    y_t(n_n,:)=conv(sta_w(n_n,:),trn_stm);
    subplot(2,2,n_n);
    xdata=mean(reshape(y_t(n_n,1:15000),50,[]),1);
    ydata=trn_psth(n_n,:);
    f_fit=param(n_n,1)*exp(param(n_n,2)*sort(xdata));
    scatter(xdata,ydata,1);
    hold on
    plot(sort(xdata),f_fit,'r');
    str=strcat('neuron',num2str(n_n),' nonlinearity');
    title(str);
end
figure
lam_t=[];
for n_n=1:4
    yt=conv(sta_w(n_n,:),test_stm);
    y=mean(reshape(yt(1:5000),50,[]),1);
    lambda=param(n_n,1)*exp(param(n_n,2)*y);
    for j=1:length(lambda)
        lam_t(n_n,j)=lambda(j);
    end
```

```
    subplot(2,2,n_n);
    scatter(test_psth(n_n,:),lam_t(n_n,:),5);
    grid on
    xlabel('observed values')
    ylabel('predicted values')
    str=strcat('neuron',num2str(n_n),' prediction');
    title(str);
end
```

```
function CNProject()
load('CN_Project3_2016.mat');
%%================================================================%
%
%%  Computational Neuroscience (EC60007) Project 3
%%
%%  Siddharth S Jha,  14EE30022
%%  Adarsh Mukesh,    13BT30031
%%
%%================================================================%
%
%%  Question no 1: Gaussian Estimation
k=zeros(100);
j=zeros(100);

for i = -49:50
    if(i>0)
    for l=1:20000-i
        j(i+50)=j(i+50)+(Stimulus(l)*Stimulus(l+i));
    end
    j(i+50)=j(i+50)/(20000-i);
    end
    if (i<=0)
    for l=-1*i+1:20000
        j(i+50)=j(i+50)+(Stimulus(l)*Stimulus(l+i));
    end
    j(i+50)=j(i+50)/(20000+1*i-1);
    end
end

t=linspace(-50,50,100);
```

```
figure(1)
plot(t,j);

%%===============================================================%
%
%%  Question no 2: PSTH Evaluation

psth=zeros(4,20000);
for i = 1:4
   for j = 1:50
        [m,n]=size(All_Spike_Times{i,j});
        for p = 1:n
            temp=ceil(All_Spike_Times{i,j}(p)*1000);
            psth(i,temp)=psth(i,temp)+1;
        end
    end
end
figure(2)
%PSTH for 4 neurons
    ax1=subplot(4,1,1);
    plot(1000*psth(1,1:20000),'r');
    xlabel(ax1,'time (ms)');
    ylabel(ax1,'r(t)');
    ax2=subplot(4,1,2);
    plot(1000*psth(2,1:20000));
    xlabel(ax2,'time (ms)');
    ylabel(ax2,'r(t)');
    ax3=subplot(4,1,3);
    plot(1000*psth(3,1:20000),'r');
    xlabel(ax3,'time (ms)');
    ylabel(ax3,'r(t)');
    ax4=subplot(4,1,4);
    plot(1000*psth(4,1:20000));
    xlabel(ax4,'time (ms)');
    ylabel(ax4,'r(t)');
smallpsth=zeros(4,100);
for i = 1:100
    for neur_no=1:4

smallpsth(neur_no,i)=mean(psth(neur_no,15000+i*50-49:15000+50*i));
    end
end
%%===============================================================%
%
%%  Question no 3: Poisson or Non-Poisson

bintimes=[10, 20, 50, 100, 200, 500];

for i = 1:6
    noofbin(i)=20000/bintimes(i);
```

```
    binfreq(i)=1000/bintimes(i);
end

spikes=zeros(4,6,2000);
for neur_no=1:4
    for freq_no=1:6
        for iter = 1:20000
            poissonmat{neur_no,freq_no,iter}=0;
        end
    end
end

varmat=[];
meanmat=[];

for neur_no=1:4
    for freq_no=1:6
        for j = 1:50
        [m,n]=size(All_Spike_Times{neur_no,j});
            for p = 1:n

temp=ceil(All_Spike_Times{neur_no,j}(p)*binfreq(freq_no));

spikes(neur_no,freq_no,temp)=spikes(neur_no,freq_no,temp)+1;
            end
%
poissonmat{neur_no,freq_no}=spikes(neur_no,freq_no,1:noofbin(freq_n
o));
            for o=1:noofbin(freq_no)
                for
b=bintimes(freq_no)*o-bintimes(freq_no)+1:bintimes(freq_no)*o

poissonmat{neur_no,freq_no,b}=spikes(neur_no,freq_no,o);
                end
%
poissonmat{neur_no,freq_no,10*o}=spikes(neur_no,freq_no,o);
            end
            spikes=zeros(4,6,20000);
            varmat(neur_no,freq_no,j)=
var([poissonmat{neur_no,freq_no,:}]);
            meanmat(neur_no,freq_no,j)=
mean([poissonmat{neur_no,freq_no,:}]);
        end
    end
end


figure(3) %for 10ms
subplot(2,2,1)
scatter(varmat(1,1,:),meanmat(1,1,:),5)
```

```
subplot(2,2,2)
scatter(varmat(2,1,:),meanmat(2,1,:),5)
subplot(2,2,3)
scatter(varmat(3,1,:),meanmat(3,1,:),5)
subplot(2,2,4)
scatter(varmat(4,1,:),meanmat(4,1,:),5)

figure(4) %for 20ms
subplot(2,2,1)
scatter(varmat(1,2,:),meanmat(1,2,:),5)
subplot(2,2,2)
scatter(varmat(2,2,:),meanmat(2,2,:),5)
subplot(2,2,3)
scatter(varmat(3,2,:),meanmat(3,2,:),5)
subplot(2,2,4)
scatter(varmat(4,2,:),meanmat(4,2,:),5)

figure(5) %for 50ms
subplot(2,2,1)
scatter(varmat(1,3,:),meanmat(1,3,:),5)
subplot(2,2,2)
scatter(varmat(2,3,:),meanmat(2,3,:),5)
subplot(2,2,3)
scatter(varmat(3,3,:),meanmat(3,3,:),5)
subplot(2,2,4)
scatter(varmat(4,3,:),meanmat(4,3,:),5)

figure(6) %for 100ms
subplot(2,2,1)
scatter(varmat(1,4,:),meanmat(1,4,:),5)
subplot(2,2,2)
scatter(varmat(2,4,:),meanmat(2,4,:),5)
subplot(2,2,3)
scatter(varmat(3,4,:),meanmat(3,4,:),5)
subplot(2,2,4)
scatter(varmat(4,4,:),meanmat(4,4,:),5)

figure(7) %for 200ms
subplot(2,2,1)
scatter(varmat(1,5,:),meanmat(1,5,:),5)
subplot(2,2,2)
scatter(varmat(2,5,:),meanmat(2,5,:),5)
subplot(2,2,3)
scatter(varmat(3,5,:),meanmat(3,5,:),5)
subplot(2,2,4)
scatter(varmat(4,5,:),meanmat(4,5,:),5)


figure(8) %for 500ms
subplot(2,2,1)
```

```
scatter(varmat(1,6,:),meanmat(1,6,:),5)
subplot(2,2,2)
scatter(varmat(2,6,:),meanmat(2,6,:),5)
subplot(2,2,3)
scatter(varmat(3,6,:),meanmat(3,6,:),5)
subplot(2,2,4)
scatter(varmat(4,6,:),meanmat(4,6,:))


%%================================================================%
%
%%  Question no 4: Spike Triggered Average
averagemat=zeros(4,100);
staspikes=zeros(4,1,200);
Stim15=Stimulus(1:15100);

mean_rate=zeros(4);
for neur_no=1:4
    n_spikes=0;
    for j=1:50
        v=[];
        v=All_Spike_Times{neur_no,j}<15;
        newspike15=All_Spike_Times{neur_no,j}(v);
        [m,n]=size(newspike15);
        n_spikes=n_spikes+n;
        for spk_no=1:n
            for tim=1:100
                averagemat(neur_no,tim)=
averagemat(neur_no,tim)+Stim15(max(1,ceil(1000*(newspike15(spk_no))
)-tim));
            end
        end
    end
    mean_rate(neur_no)=n_spikes/750;
    for tim=1:100
        averagemat(neur_no,tim)= averagemat(neur_no,tim)/n_spikes;
    end
end

figure(9)
subplot(2,2,1)
plot(averagemat(1,:));
ylim([-0.2 0.2]);
subplot(2,2,2)
plot(averagemat(2,:));
ylim([-0.2 0.2]);
subplot(2,2,3)
plot(averagemat(3,:));
ylim([-0.2 0.2]);
subplot(2,2,4)
```

```
plot(averagemat(4,:));
ylim([-0.2 0.2]);

%% Whitened STA
corr_new=zeros(101);
for i = 0:100
    for l=1:20000-i
        corr_new(i+1)=corr_new(i+1)+(Stimulus(l)*Stimulus(l+i));
    end
    corr_new(i+1)=corr_new(i+1)/(20000-i);
end
figure(10)
plot(corr_new);
corr_mat=zeros(100,100);

for row=1:100
    for col=1:100
        corr_mat(row,col)=corr_new(abs(row-col)+1);
    end
end
cssinv=inv(corr_mat);
ratenew=mean_rate(:,1);
corrected_averagemat=cssinv*transpose(averagemat);

for i =1:4
    corrected_averagemat(:,i)=corrected_averagemat(:,i)*ratenew(i);
end

%% Question 5: Nonlinearity
conv=zeros(4,5000);
for neur_no=1:4
    for tau=15000:19900

conv(neur_no,tau-14999)=(Stimulus(tau:tau+99))*corrected_averagemat
(:,neur_no);
    end
end
smallconv=zeros(4,100);
for i = 1:100
    for neur_no=1:4
        smallconv(neur_no,i)=mean(conv(neur_no,i*50-49:50*i));
    end
end



figure(11)
subplot(2,2,1)
plot(corrected_averagemat(:,1));
subplot(2,2,2)
```

```
plot(corrected_averagemat(:,2));
subplot(2,2,3)
plot(corrected_averagemat(:,3));
subplot(2,2,4)
plot(corrected_averagemat(:,4));

figure(12)
subplot(2,2,1)
scatter(smallpsth(1,:),smallconv(1,:),5);
subplot(2,2,2)
scatter(smallpsth(2,:),smallconv(2,:),5);
subplot(2,2,3)
scatter(smallpsth(3,:),smallconv(3,:),5);
subplot(2,2,4)
scatter(smallpsth(4,:),smallconv(4,:),5);
end

%
% %%==============================================================%%
% %%  Question: Victor Purpura Distance
% num_trial=10;
% MI=zeros(num_trial,7);
% for vp_trial=1:num_trial
%     q=[0, 0.001, 0.01, 0.1, 1, 10, 100];
%     rng(vp_trial);
%     start_pt=rand(1,8);
%     start_pt=start_pt*15;
%     end_pt=start_pt+0.1;
%
%
%     for trial_no = 1:50
%         for stpt=1:8
%             temp1=[];
%             temp1=All_Spike_Times{1,trial_no}>start_pt(stpt);
%             spktemp1=All_Spike_Times{1,trial_no}(temp1);
%             temp2=[];
%
temp2=All_Spike_Times{1,trial_no}<=start_pt(stpt)+0.1;
%             spktemp2=All_Spike_Times{1,trial_no}(temp2);
%             vpmat{stpt,trial_no}=intersect(spktemp1,spktemp2);
%         end
%     end
%
%     confusion=zeros(7,8,8);
%
%     for qno=1:7
%         vpmindist=zeros(8,50);
%         vpmindist=inf+vpmindist;
%         for m=1:50
%             for n=1:8
```

```
%                  minj=[];
%                  for i = 1:50
%                      for j = 1:8
%                          if ~(m==i && n==j)
%
if(spkd(vpmat{j,i},vpmat{n,m},q(qno))<=vpmindist(n,m))
%
vpmindist(n,m)=spkd(vpmat{j,i},vpmat{n,m},q(qno));
%
if(spkd(vpmat{j,i},vpmat{n,m},q(qno))==vpmindist(n,m))
%                                      minj=[minj,j];
%                                  else
%                                      minj=[j];
%                                  end
%
%                              end
%          %                     vpdist(n,m)=min(a,vpdist(m,n));
%                          end
%                      end
%                  end
%                  for it=1:length(minj)
%
confusion(qno,n,minj(it))=confusion(qno,n,minj(it))+1/length(minj);
%                  end
%              end
%          end
%
%
%
%      end
%      confusion=confusion/400;
%      for qno=1:7
%          for x = 1:8
%                  for y=  1:8
%
MI(vp_trial,qno)=MI(vp_trial,qno)+confusion(qno,x,y)*log2(confusion
(qno,x,y)/(sum(confusion(qno,x,:))*sum(confusion(qno,:,y))));
%                  end
%          end
%      end
% end
% MIavg=sum(MI(:,:));
% MIavg=MIavg/num_trial;
% figure(13)
% plot(linspace(-3,2,6),MIavg(2:7));
% end
%
%
%
%
```

```
% function d=spkd(tli,tlj,cost)
% %
% % d=spkd(tli,tlj,cost) calculates the "spike time" distance
% % (Victor & Purpura 1996) for a single cost
% %
% % tli: vector of spike times for first spike train
% % tlj: vector of spike times for second spike train
% % cost: cost per unit time to move a spike
% %
% %  Copyright (c) 1999 by Daniel Reich and Jonathan Victor.
% %  Translated to Matlab by Daniel Reich from FORTRAN code by
Jonathan Victor.
% %
% nspi=length(tli);
% nspj=length(tlj);
%
% if cost==0
%    d=abs(nspi-nspj);
%    return
% elseif cost==Inf
%    d=nspi+nspj;
%    return
% end
%
% scr=zeros(nspi+1,nspj+1);
% %
% %     INITIALIZE MARGINS WITH COST OF ADDING A SPIKE
% %
% scr(:,1)=(0:nspi)';
% scr(1,:)=(0:nspj);
% if nspi & nspj
%    for i=2:nspi+1
%       for j=2:nspj+1
%          scr(i,j)=min([scr(i-1,j)+1 scr(i,j-1)+1
scr(i-1,j-1)+cost*abs(tli(i-1)-tlj(j-1))]);
%       end
%    end
% end
% d=scr(nspi+1,nspj+1);
% end
```