Table of Contents


```
% EE239.2 HW 3
```

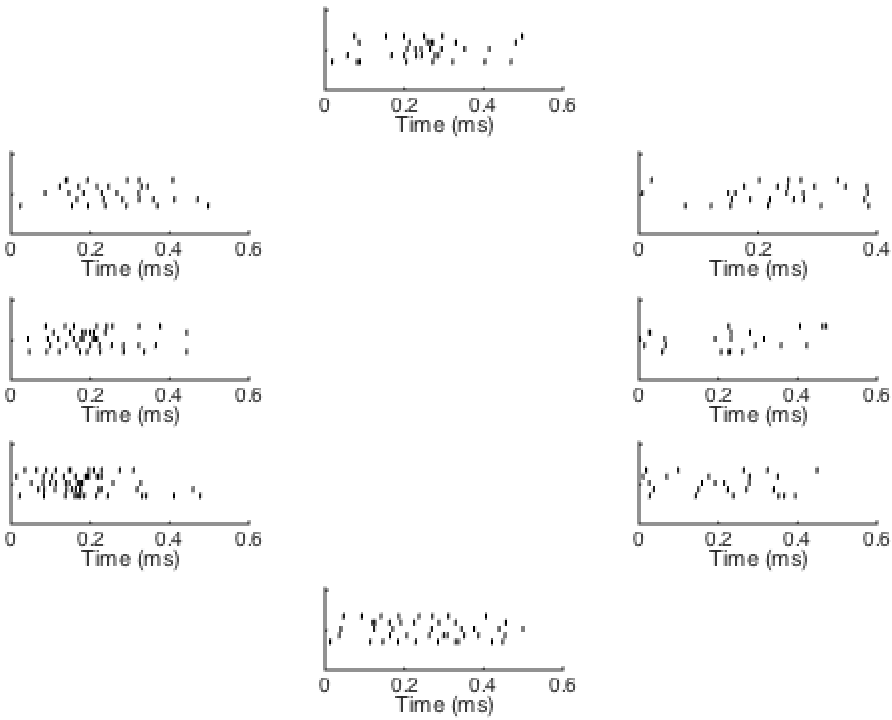
Problem 4: Real Neural Data

```
clc
clear
close all
```

Part A: Spike Trains

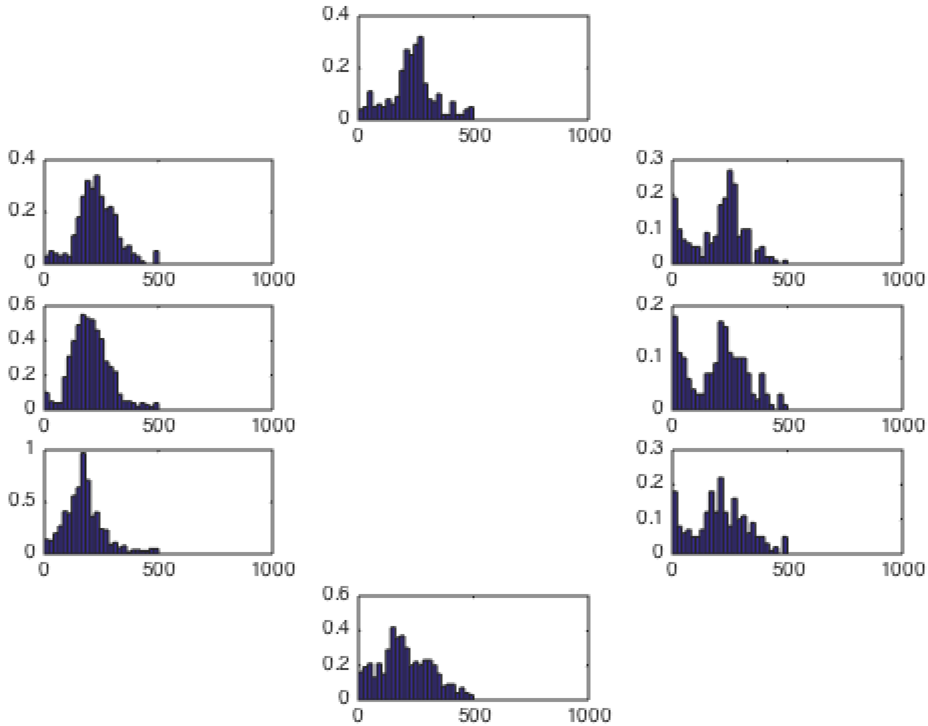
```
ps3_data = importdata('ps3_data.mat');
T_cell = {};
numTrials = 128;
spike_loc = {};
max_trains = zeros(8,5);
max_spikes = zeros(8,5);

for i = 1:size(ps3_data,2)
    for k = 1:size(ps3_data,1)
        spike_cell{i,k} = ps3_data(k,i).spikes;
        T_cell{i,k} = (find(spike_cell{i,k} > 0)-1)/1000;
        % subtract 1 because time indexing starts at t = 0 and MATLAB
        % indexing starts at 1
        % divide by 1000 to convert to seconds

        spike_sum(i,k) = sum(spike_cell{i,k});

    end
    [B,I] = sort(spike_sum(i,:),'descend');
    max_trains(i,:) = I(1:5);
    max_spikes(i,:) = B(1:5);
    % plotted five highest spike count trains for visualization

    for j = 1:size(max_trains,2)
        T_cell_plot{i,j} = T_cell{i,max_trains(i,j)};
    end
```

```
end

figure(1)
subplotRaster(T_cell_plot)
```



Part B: Spike Histogram

```
bins = 0:0.020:1;

counts_sum = zeros(8,length(bins));
for i=1:100
    for j=1:8
        counts = histc(T_cell{j,i}, bins);
        counts_sum(j,:) = counts_sum(j,:) + counts;
        % count the number of spikes per bin
    end
end

figure(2)
counts_sum = counts_sum(:,1:50);
counts_avg = counts_sum/100;           % average across 100 trials
subplotCounts(counts_avg,bins)
```



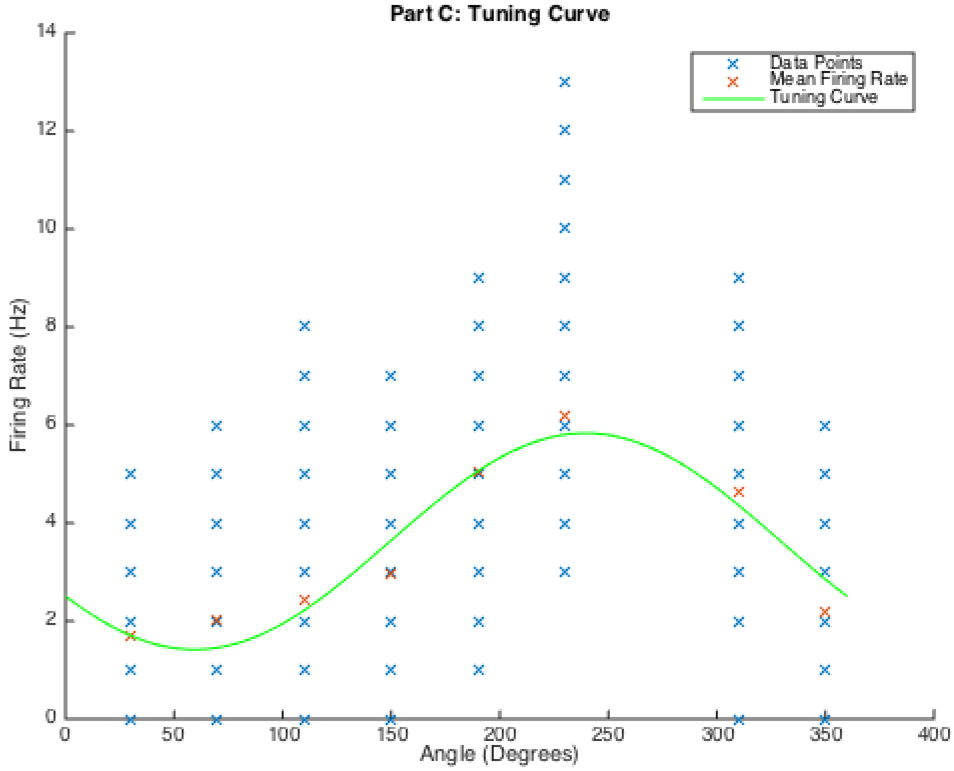
Part C: Tuning Curve

```
s = [30 70 110 150 190 230 310 350];
t_plot = 0:360;
f_rate = zeros(1,length(s)*numTrials);
rate = zeros(length(s),numTrials);

f_rate = zeros(1,length(s)*numTrials);
rate = zeros(length(s),numTrials);
for i=1:length(s)
    for j=1:numTrials
       rate(i,j) = length(T_cell{i,j});
    end
end

f_rate = reshape(rate',[1,length(s)*numTrials]);

s_rep = repmat(s,numTrials,1);
s_rep = reshape(s_rep, 1, numel(s_rep));
f_rate_mean = sum(rate,2)/numTrials;
% calculate the mean firing rate from the data and averave over the number
% of trials

F = @(x,xdata)x(1) + x(2)*cosd(xdata - x(3));
x0 = [1 1 100];
```

```
[x,resnorm,~,exitflag,output] = lsqcurvefit(F,x0,s,f_rate_mean');
% find the parameters of Equation 1 using least squares

r_0 = x(1);
r_max = x(2)+x(1);
s_max = x(3);
% parameters for tuning curve

figure(3)
scatter(s_rep,f_rate,'x');
f_rate_mean = sum(rate,2)/numTrials;
hold on;
scatter(s,f_rate_mean,'x')
plot(t_plot,F(x,t_plot),'g')
legend('Data Points','Mean Firing Rate','Tuning Curve')
xlabel('Angle (Degrees)')
ylabel('Firing Rate (Hz)')
title('Part C: Tuning Curve')
hold off
```

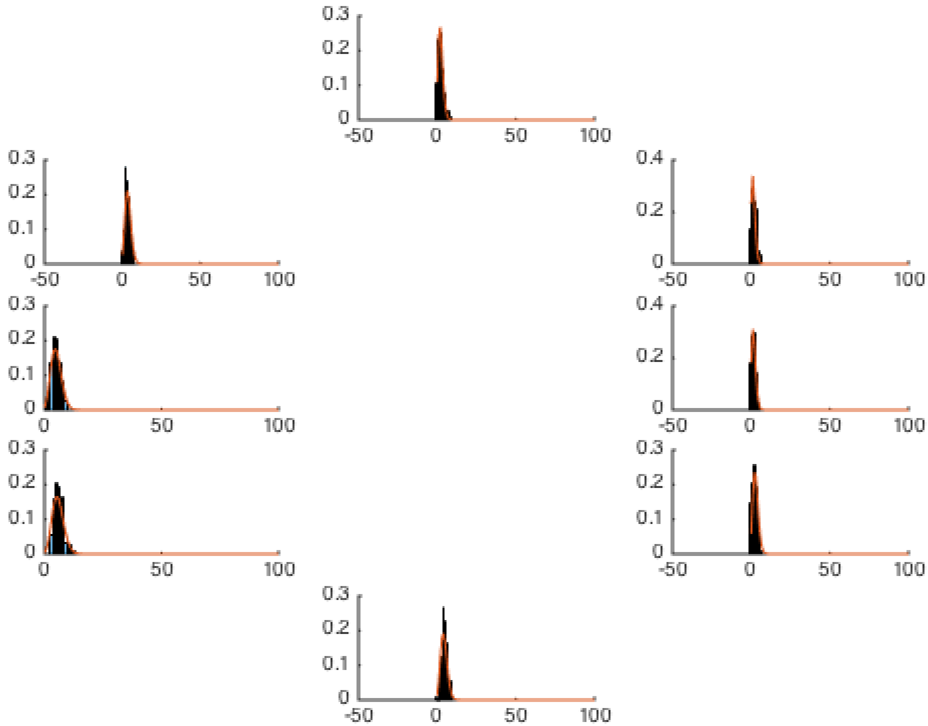
Local minimum found.

Optimization completed because the size of the gradient is less than the default value of the function tolerance.



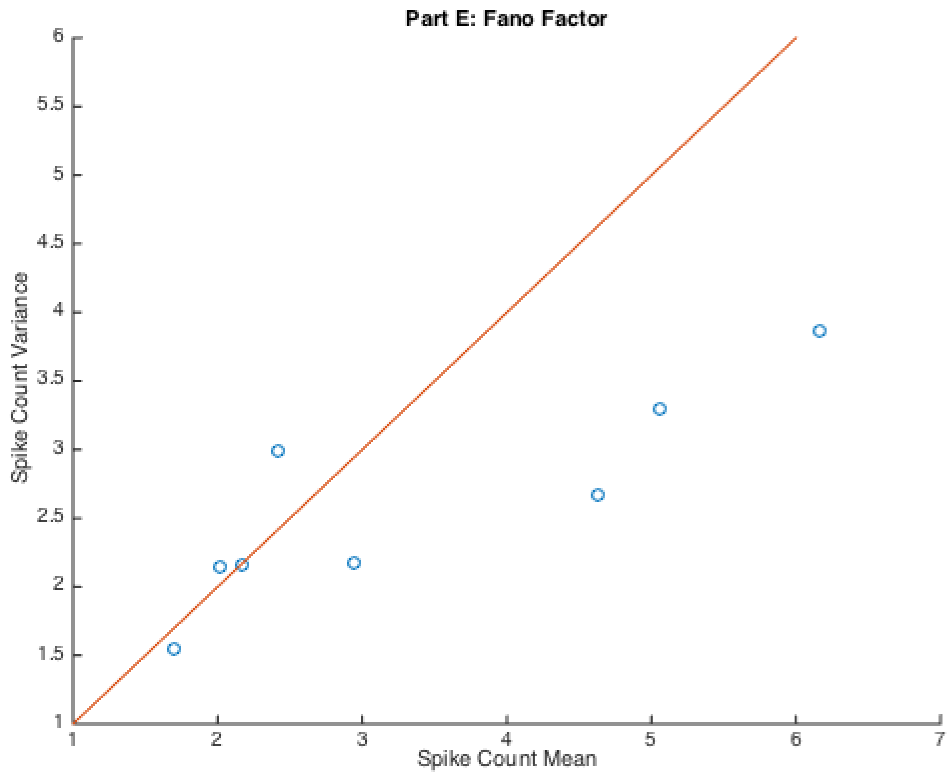
Part D: Count Distribution

```
lambda = r_0 + (r_max-r_0) * cosd(s - s_max);
figure(4)
subplotHist(rate, lambda)

% The empirical distributions differ from the idealized Poisson distributions
% because of the refractory periods and inhomogeity of the Poisson process.
% The exponential distribution used to generate the ISIs does not account
% for the refractory period.
```



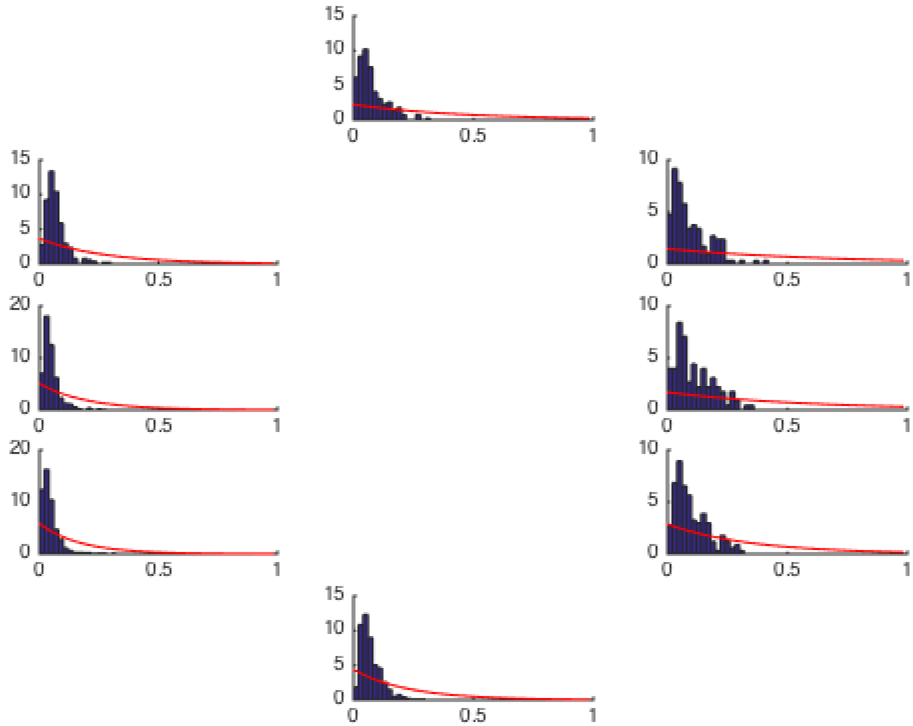
Part E: Fano Factor

```
counts_mean = mean(rate,2);
counts_var = var(rate,1,2);
figure(5)
hold on
scatter(counts_mean, counts_var)
plot(1:max(counts_mean),1:max(counts_mean))
title('Part E: Fano Factor')
xlabel('Spike Count Mean')
ylabel('Spike Count Variance')
hold off

% The points do not lie on the 45 degree line because as firing rate
% increases, the ISI distribution tends towards a Gamma distribution, and
% so the process overall becomes sub-Poisson. We can see this in the plot,
% as the data points drop below the 45 degree line.
```



Part F: ISI Distribution

```
ISI = cell(8,1);
ISI_dist = cell(8,1);
mu = 1./lambda;

for i = 1:length(mu)
    for k = 1:numTrials
        ISI{i} = [ISI{i}, diff(T_cell{i,k})];
    end
    ISI_hist = histcounts(ISI{i},bins,'Normalization','pdf');
    ISI_dist{i} = ISI_hist;
end

figure(6)
subplotISI(ISI_dist,bins(1:end-1),mu)

% The empirical distributions of ISIs differ from the idealized
% exponential distributions because of refractory periods, as well as the
% fact that the firing rate varies with time and is therefore not a
% homogeneous Poisson process, which is what ideal exponential
% distributions generate.
```



Published with MATLAB® R2014b