1. **Homogeneous Poisson process** A homogeneous Poisson process of rate λ (measured in Hertz) is a point process where the probability of an event is independent of time t and independent of previous events. The probability P of an event within a bin of width Δt is

$$P = \lambda \cdot \Delta t$$

for sufficiently small Δt.

(a) Write a function that generates n homogeneous Poisson spike trains of a given duration T_{max} with rate λ.

Solution:

hompoissonspikes.m

```
function spikes = hompoissonspikes( trials, rate, tmax )
% Generate spike times of a homogeneous poisson process
% trials: number of trials that should be generated
% rate: the rate of the Poisson process in Hertz
% tmax: the duration of each trial in seconds
% returns a cell array of vectors of spike times

    dt = 3.33e-5;
    p = rate*dt;
    if p > 0.2
        p = 0.2
        dt = p/rate;
    end
    x = rand( trials, ceil(tmax/dt) );
    spikes = cell( trials, 1 );
    for k=1:trials
        spikes{k} = find( x(k,:) >= 1.0-p ) * dt;
    end
end
```

(b) Using this function, generate a few trials and display them in a raster plot.

Solution:

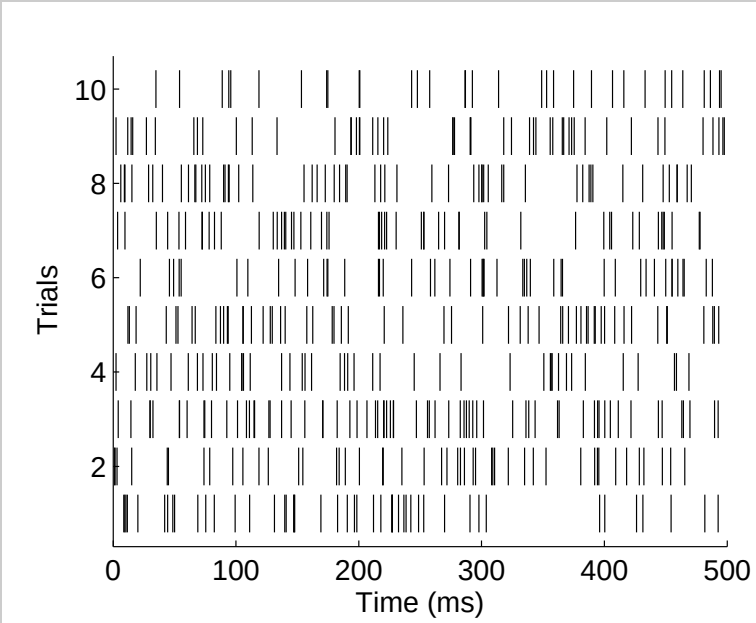
spikeraster.m

```
function spikeraster( spikes )
% Display a spike raster of the spike times given in spikes.
% spikes: a cell array of vectors of spike times

ntrials = length(spikes);
for k = 1:ntrials
    times = 1000.0*spikes{k}; % conversion to ms
    for i = 1:length( times )
      line([times(i) times(i)],[k-0.4 k+0.4], 'Color', 'k' );
    end
end
xlabel( 'Time [ms]' );
ylabel( 'Trials');
ylim( [ 0.3 ntrials+0.7 ] )
```

```
15
16 end
```

.

```
1 spikes = hompoissonspikes( 10, 100.0, 0.5 );
2 spikeraster( spikes )
```



(c) Write a function that extracts a single vector of interspike intervals from the spike times returned by the first function.

Solution:

isis.m

```
1  function isivec = isis( spikes )
2  % returns a single list of isis computed from spikes
3  % spikes: a cell array of vectors of spike times
4
5      isivec = [];
6      for k = 1:length(spikes)
7          difftimes = diff( spikes{k} );
8          isivec = [ isivec difftimes ];
9      end
10 end
```

(d) Write a function that plots the interspike-interval histogram from a vector of interspike intervals. The function should also compute the mean, the standard deviation, and the CV of the intervals and display the values in the plot.

Solution:

isihist.m

```
function isihist( isis, binwidth )
% histogram of isis
% isis: vector of interspike intervals
% binwidth: optional width to be used for the isi bins

    if nargin < 2
        nperbin = 100; % average number of data points per bin
        bins = length( isis )/nperbin; % number of bins
        binwidth = max( isis )/bins;
        if  binwidth < 5e-4 % half a millisecond
            binwidth = 5e-4;
        end
    end
    bins = 0.5*binwidth:binwidth:max(isis);
    % histogram:
    [ nelements, centers ] = hist( isis, bins );
    % normalization (integral = 1):
    bindt = centers(2) - centers(1);
    nelements = nelements / sum( nelements ) / bindt;
    % plot:
    bar( 1000.0*centers, nelements );
    xlabel( 'ISI [ms]' )
    ylabel( 'p(ISI) [1/s]')
    % annotation:
    misi = mean( isis );
    sdisi = std( isis );
    disi = sdisi^2.0/2.0/misi^3;
    text( 0.5, 0.6, sprintf( 'mean=%.1f ms', 1000.0*misi ), 'Units', '
        normalized' )
    text( 0.5, 0.5, sprintf( 'std=%.1f ms', 1000.0*sdisi ), 'Units', '
        normalized' )
    text( 0.5, 0.4, sprintf( 'CV=%.2f', sdisi/misi ), 'Units', 'normalized'
         )
    %text( 0.5, 0.3, sprintf( 'D=%.1f Hz', disi ), 'Units', 'normalized' )
end
```

(e) Compute histograms for Poisson spike trains with rate $\lambda = 100\,\mathrm{Hz}$. Play around with T and n and the bin width (start with 1 ms) of the histogram. How many interspike intervals do you approximately need to get a "nice" histogram? How long do you need to record from the neuron?

Solution: About 5000 intervals for 25 bins. This corresponds to a $5000/100\,\mathrm{Hz} = 50\,\mathrm{s}$ recording of a neuron firing with 100 Hz.

(f) Compare the histogram with the true distribution of intervals T of the Poisson process

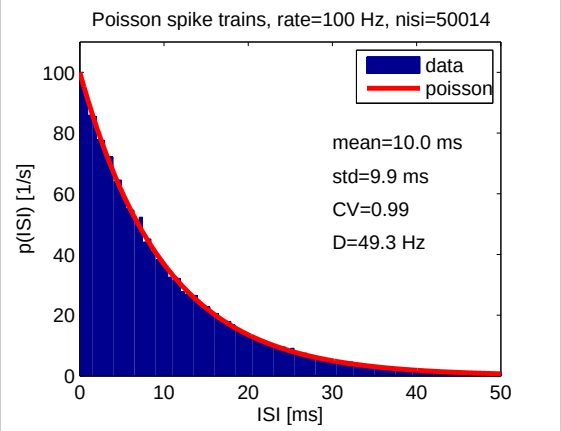
$$p(T) = \lambda e^{-\lambda T}$$

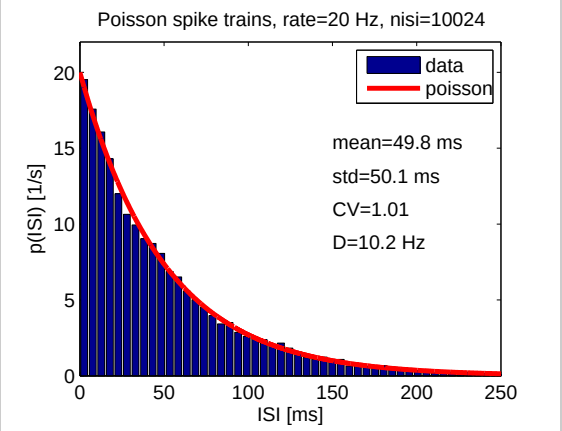
for various rates λ.

Solution:

hompoissonisih.m

```
% generate spike times:
rate = 20.0;
spikes = hompoissonspikes( 10, rate, 50.0 );
% isi histogram:
isivec = isis( spikes );
isihist( isivec );
hold on
% theoretical density:
xmax = 5.0/rate;
x = 0:0.0001:xmax;
y = rate*exp(-rate*x);
plot( 1000.0*x, y, 'r', 'LineWidth', 3 );
% plot details:
title( sprintf( 'Poisson spike trains, rate=%g Hz, nisi=%d', rate, length(
    isivec ) ) )
xlim( [ 0.0 1000.0*xmax ] )
ylim( [ 0.0 1.1*rate ] )
legend( 'data', 'poisson' )
hold off
```





(g) What happens if you make the bin width of the histogram smaller than Δt used for generating the Poisson spikes?

Solution: The bins between the discretization have zero entries. Therefore the other ones become higher than they should be.

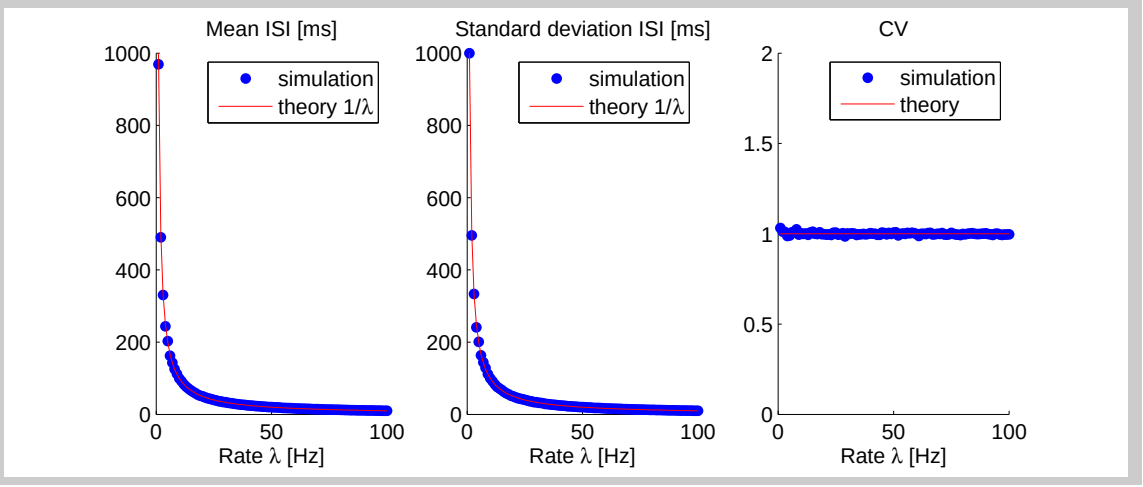
(h) Plot the mean interspike interval, the corresponding standard deviation, and the CV as a function of the rate λ of the Poisson process. Compare the simulations with the theoretical expectations for the dependence on λ.

Solution:

hompoissonisistats.m

```
rates = 1:1:100;
avisi = [];
sdisi = [];
cvisi = [];

for rate = rates
```

```
    spikes = hompoissonspikes( 10, rate, 100.0 );
    isivec = isis( spikes );
    av = mean( isivec );
    sd = std( isivec );
    cv = sd/av;
    avisi = [ avisi av ];
    sdisi = [ sdisi sd ];
    cvisi = [ cvisi cv ];
end

f = figure;
subplot( 1, 3, 1 );
scatter( rates, 1000.0*avisi, 'b', 'filled' );
hold on;
plot( rates, 1000.0./rates, 'r' );
hold off;
xlabel( 'Rate \lambda [Hz]' );
ylim( [ 0 1000 ] );
title( 'Mean ISI [ms]' );
legend( 'simulation', 'theory 1/\lambda' );

subplot( 1, 3, 2 );
scatter( rates, 1000.0*sdisi, 'b', 'filled' );
hold on;
plot( rates, 1000.0./rates, 'r' );
hold off;
xlabel( 'Rate \lambda [Hz]' );
ylim( [ 0 1000 ] )
title( 'Standard deviation ISI [ms]' );
legend( 'simulation', 'theory 1/\lambda' );

subplot( 1, 3, 3 );
scatter( rates, cvisi, 'b', 'filled' );
hold on;
plot( rates, ones( size( rates ) ), 'r' );
hold off;
xlabel( 'Rate \lambda [Hz]' );
ylim( [ 0 2 ] )
title( 'CV' );
legend( 'simulation', 'theory' );
```



(i) Write a function that computes serial correlations for the interspike intervals for a range of lags. The serial correlations ρ_k at lag k are defined as

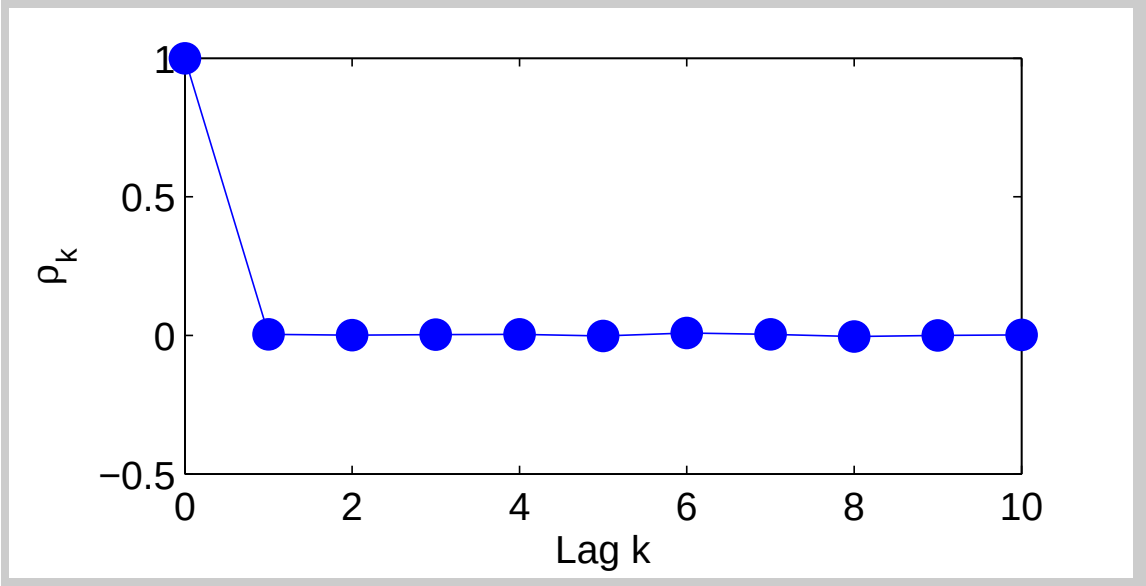
$$\rho_k = \frac{\langle (T_{i+k} - \langle T \rangle)(T_i - \langle T \rangle) \rangle}{\langle (T_i - \langle T \rangle)^2 \rangle} = \frac{\mathrm{cov}(T_{i+k}, T_i)}{\mathrm{var}(T_i)}$$

Use this function to show that interspike intervals of Poisson spikes are independent.

Solution:

isiserialcorr.m

```
function isicorr = isiserialcorr( isis )
% serial correlation of isis
% isis: vector of interspike intervals

    lags = 0:10;
    isicorr = lags;
    for k = 1:length(lags)
        lag = lags(k);
        cc = corrcoef( [ isis(1:end-lag)', isis(1+lag:end)' ] );
        isicorr(k) = cc( 1, 2 );
    end
    % plot:
    plot( lags, isicorr, '-b' );
    hold on;
    scatter( lags, isicorr, 100.0, 'b', 'filled' );
    hold off;
    xlabel( 'Lag k' )
    ylabel( '\rho_k')
end
```



(j) Write a function that generates from spike times a histogram of spike counts in a count window of given duration W. The function should also plot the Poisson distribution

$$P(k) = \frac{(\lambda W)^k e^{\lambda W}}{k!}$$

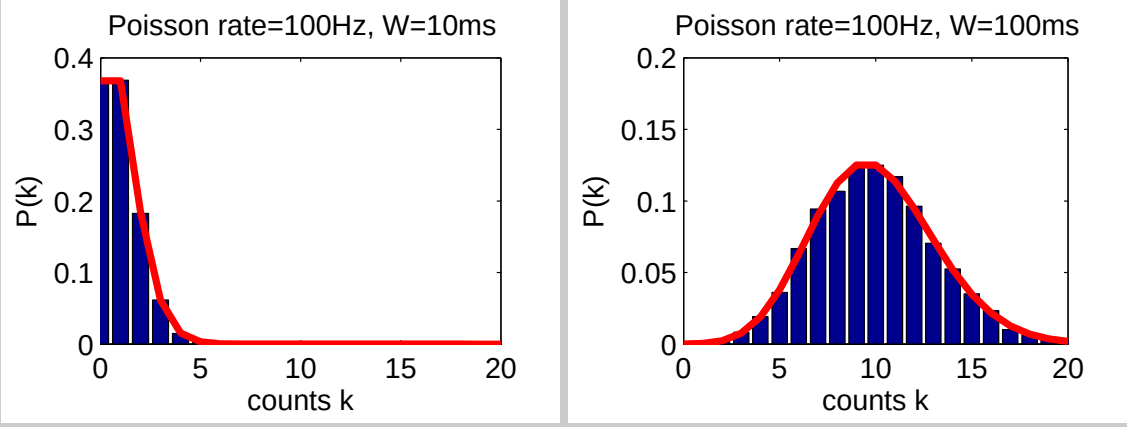
for the rate λ determined from the spike trains.

Solution:

counthist.m

```
function counthist( spikes, w )
% computes count histogram
% spikes: a cell array of vectors of spike times
% w: observation window duration for computing the counts

    tmax = spikes{1}(end);
    n = [];
    r = [];
    for k = 1:length(spikes)
        for tk = 0:w:tmax-w
            nn = length( find( ( spikes{k} >= tk ) & ( spikes{k} < tk+w ) )
                );
            n = [ n nn ];
        end
        rate = (length(spikes{k})-1)/(spikes{k}(end) - spikes{k}(1));
        r = [ r rate ];
    end
    [counts, bins ] = hist( n, 0:1:20 );
    counts = counts / sum( counts );
    bar( bins, counts );
    hold on;
    rate = mean( r );
    x = 0:1:20;
```

```
    l = rate*w;
    y = l.^x.*exp(-l)./factorial(x);
    plot( x, y, 'r', 'LineWidth', 3 );
    xlim( [ 0 20 ] );
    hold off;
    xlabel( 'counts k' );
    ylabel( 'P(k)' );
end
```



(k) Write a function that computes mean count, variance of count and the corresponding Fano factor for a range of count window durations. The function should generate tow plots: one plotting the count variance against the mean, the other one the Fano factor as a function of the window duration.

Solution:

fano.m

```
function fano( spikes )
% computes fano factor as a function of window size
% spikes: a cell array of vectors of spike times

    tmax = spikes{1}(end);
    windows = 0.01:0.01:0.01*tmax;
    mc = windows;
    vc = windows;
    ff = windows;
    for j = 1:length(windows)
        w = windows( j );
        n = [];
        for k = 1:length(spikes)
            for tk = 0:w:tmax-w
                nn = length( find( ( spikes{k} >= tk ) & ( spikes{k} < tk+w
                    ) ) );
                n = [ n nn ];
            end
        end
        mc(j) = mean( n );
        vc(j) = var( n );
        ff(j) = vc( j )/mc( j );
```

```
    end

    subplot( 1, 2, 1 );
    scatter( mc, vc, 'filled' );
    xlabel( 'Mean count' );
    ylabel( 'Count variance' );

    subplot( 1, 2, 2 );
    scatter( 1000.0*windows, ff, 'filled' );
    xlabel( 'Window W [ms]' );
    ylabel( 'Fano factor' );
    ylim( [ 0.0 0.5 ] );
end
```