Problem set 6

Analysis for Neuroscientists

1. In this problem we'll reproduce a key result from Churchland et al. [1], who found that the *trial-to-trial variability* of the number of spikes produced by neurons in motor areas decreases during movement. Start by loading the file `PMDdata.mat`. This contains a length 4264 struct `PMDdata`, representing 533 neurons in 8 reach conditions. For our purposes we'll treat these as 4264 individual samples. `PMDdata(i).spikes` is a 23×900 array. This array corresponds to the spikes of neuron i over 23 trials, each of which is 900 ms long. The array is 0 when there is no spike and 1 when there is a spike in each of the 900 1 ms time bins. The reach target comes on at 400 ms.

   (a) The *Fano factor* is a measure of the trial-to-trial variability of a neuron's spiking. It's equal to Var (S) /Mean (S), where S is the number of spikes a neuron emits in a particular time window, and the variance and mean are computed over a set of trials. Write a function that takes in a #Trials $\times$ #Time bins matrix of 0s and 1s like described above and computes the Fano factor of the summed spike count in the time window `[tstart,tend]`. Use this as an outline (the body of this function can be written with only two lines, though you can use more if you want):

```
function[FF] = calcfanofac(M,tstart,tend)
    S = ...
    FF = ...
end
```

   (b) Compute the Fano factor for each sample, separately for the two time intervals 300-400 ms (right before the target appears) and 500-600 ms (after the target has appeared). Use this as an outline (you only need to fill in two lines):

```
load('PMDdata.mat');
N = length(PMDdata);

%time bins for the start and end of the pre-target and post-target intervals
pre_start = 300;
pre_end = 400;
post_start = 500;
post_end = 600;
```

```
%Fano factors for each sample
fano_pre = zeros(N,1);
fano_post = zeros(N,1);

%loop over samples, get spikes and compute Fano factors
for ii = 1:N
    M = PMDdata(ii).spikes;

    fano_pre(ii) = ...
    fano_post(ii) = ...
end

%ignore samples that have NaN values
valid_inds = ~(isnan(fano_pre) | isnan(fano_post));
fano_pre = fano_pre(valid_inds);
fano_post = fano_post(valid_inds);
```

(c) What is the average of `fano_pre` and of `fano_post`? Make a histogram showing the distributions of both of these quantities. Then do a statistical test to test whether their means or medians differ. Can you use a paired test? State which test you used and the p-value. The p-value should be very small—this is a large effect.

2. In the last homework, you analyzed the connectivity of Kenyon cells and compared it to a model with random connections. But we didn't have a way to say if this difference was significant. We'll do that now. The approach will be to compare the data to many simulated datasets with random connections and get a p-value. This is related to the general approach of comparisons to *shuffled data*.

   (a) Fill in the following code. `K_data` is a length-11 vector containing the number of Kenyon cells with exactly k connections. In the code, you are generating `K_shuffle`, a matrix where each column is length 11 and contains the number of Kenyon cells, for a randomly drawn connectivity matrix, with that many connections. Use this as an outline (you only need to fill in three lines):

```
load('kcs.mat');
conns = sum(J,2); %connections per Kenyon cell

N_KC = size(J,1); %number of Kenyon cells
N_PN = size(J,2); %number of presynaptic projection neurons

conn_prob = mean(mean(J)); %connection probability
```

```
edges = 0:11;
K_data = histcounts(conns,edges);

Ntrials = 1000;
K_shuffle = zeros(Ntrials,11);

for ti = 1:Ntrials
    J_shuffle = ... %random N_KC x N_PN matrix. equal to 1
                    %with probability conn_prob, 0 otherwise
    conns_shuffle = ...
    K_shuffle(ti,:) = ...
end
```

Hint: You can make a random $M \times N$ matrix of 0s and 1s, with a probability of a 1 being p, using `rand(M,N) < p`.

(b) For each of the 11 values of k, compare `K_data(k)` to `K_shuffle(:,k)`. Specifically, compute the fraction of elements of `K_shuffle(:,k)` that are "more extreme" than `K_data(k)`. You can do this by, for each k, asking what fraction of the elements of `K_shuffle(:,k)` are greater than or equal to `K_data(k)`, and what fraction are less than or equal to it, and taking the minimum of these two fractions. This is your p-value for the null hypothesis that the data is no different than the random shuffles you created. Store these p-values in a length-11 vector `pvals`, and print out the result.

(c) Given the 11 p-values you just generated above, for which values of k are there significantly different numbers of Kenyon cells in the data and in the shuffled models? Say which p-value threshold you used, given that we start with a criterion of $\alpha = 0.05$ for significance but need to take into account multiple comparisons.

References

[1] Churchland, M.M., et al. Stimulus onset quenches neural variability: a widespread cortical phenomenon. *Nature Neuroscience* **13**, 369–378 (2010).