Neural Encoding

Naureen Ghani

February 10, 2018

Introduction

Neuroscientists are interested in knowing what neurons are doing. More specifically, researchers want to understand how neurons represent stimuli from the outside world with changes in their firing properties. This is an active area of study known as ***neural encoding.*** Scientists hope to identify neurons that activate specific behaviors, such as escape or pain. Using standard ***electrophysiological*** techniques, one can record the response of the neuron to each stimulus. ***Electrophysiology*** is the study of the electrical properties of biological cells and tissues. In neuroscience, it is used to record ***action potentials.*** There are fundamental methods to analyze spike trains of single neurons, which aim to characterize their encoding properties. These techniques are: ***raster plots, peri-stimulus time histograms,*** and ***tuning curves***.

Say you are studying a neuron from the visual system. You would first present the subject with a controlled visual stimulus. This stimulus has a defined orientation, luminance, and contrast. You perform a ***patch-clamp*** experiment to record the voltage response of the single cell when presented with a visual stimulus. You repeat this experiment over many trials to see how similar (or different) the neuronal responses are to the stimulus. A ***raster plot*** is a simple method to visually examine the trial-by-trial variability of the respones. You can examine what features these responses have in common by averaging over all responses to create a ***peri-stimulus time histogram.*** To capture how the average response of the neuron varies with some sensory or motor feature, you can generate a ***tuning curve*** that maps the feature value onto the average response of the neuron.

Spike Raster Plot

The most important information of ***spikes*** or ***action potentials*** is *timing.* Size and shape of spikes do not matter in this analysis. We are most interested in the frequency of firing. However, recent research on ***local field potentials*** and ***oscillations*** study the size and shape of spikes. For our analysis, we will graphically show the time a spike was present in the recorded voltage trace with a line.

We will use open-source data that was acquired from a single cell in the auditory cortex of a macaque monkey. The auditory stimulus was 1 of 13 tones of different frequencies (200 Hz to 16 kHz, logarithmically spaced) at 4 different sound levels (40, 50, 60, and 70 dB). There were ten trials for each frequency-sound level combination, for a total of 520 trials.

```
load('tonespike');
whos tonespike % see structure in command space
```

In your workspace, you will see that the data is stored in a structure (struct), and this structure has a size of 1x520. To obtain the spike timings of the first trial, type:

```
t = tonespike(1).spiketime; % Spike timings in the first trial
```

This is a vector containing 12 elements, or spike times. Every trial will have its own number of spikes, as the cell will not fire the same number of spikes for every stimulus. This is why the data is stored in a struct-array as this allows you to store vectors of different sizes. The second trial has only 5 spikes:

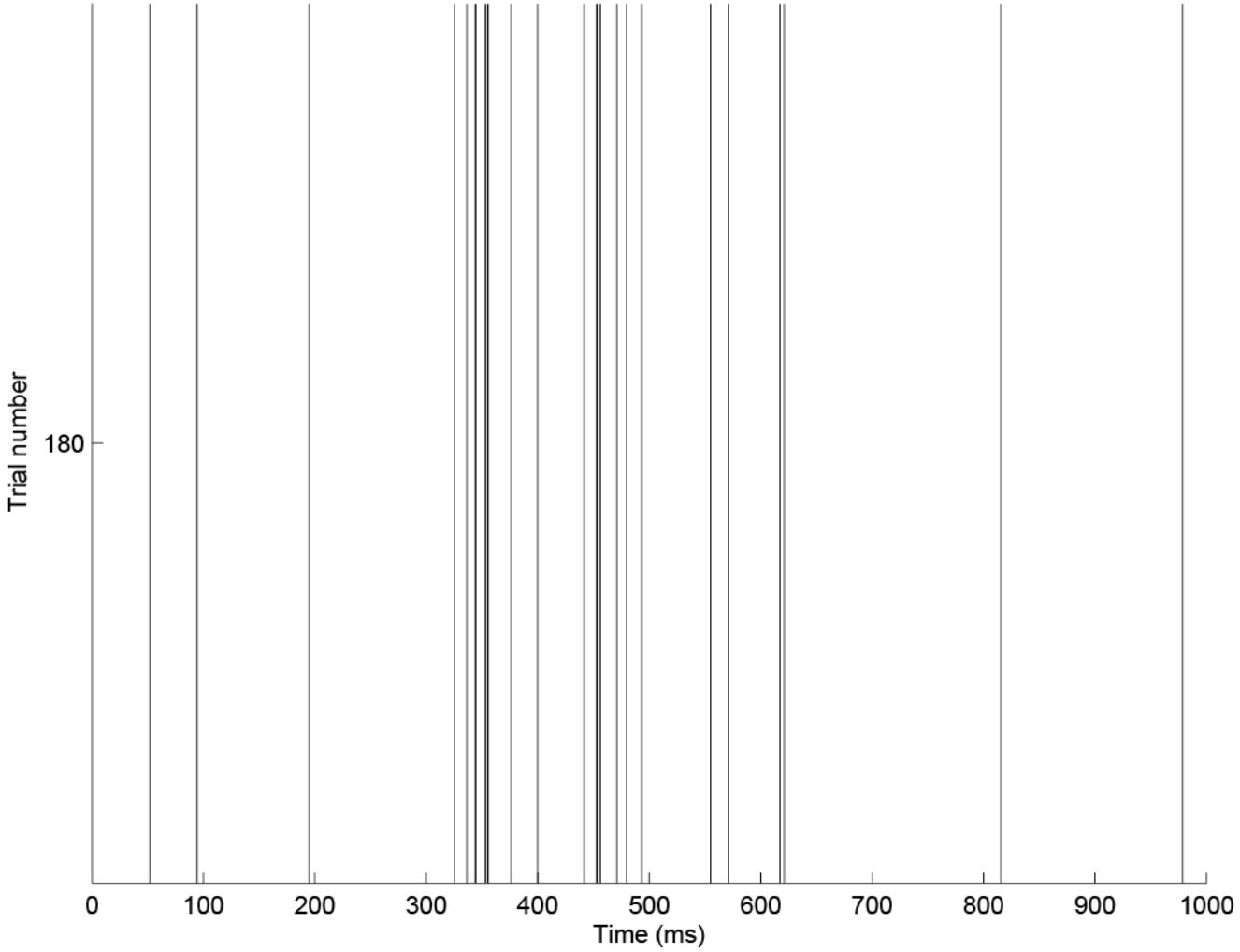
```
t = tonespike(2).spiketime; % Spike timings in the second trial
```

Let us plot a raster of the 180th trial:

```
t = tonespike(180).spiketime % Spike timings in the 180th trial
nspikes = numel(t); % number of spikes
for ii = 1:nspikes % for every spike
        line([t(ii) t(ii)],[179.5 180.5],'Color','k'); % draw a black vertical
           line at time t(x) and at trial 180 (y)
end

xlabel('Time (ms)'); % Time in milliseconds
ylabel('Trial number');
```

You may notice that after 300 ms, the spikes occur more often. This indicates that the tone (auditory stimulus) was presented at that time. In other words, the presentation of the sound in this single trial increases the firing rate of this neuron.



Let us now visualize all the trials over time to see if the increase in firing rate is consistent among cells. We will use a for loop to do this:

```
ntrials = numel(tonespike); % number of trials
for jj = 1:ntrials
        t = tonespike(jj).spiketime; % Spike timings in the jjth trial
        nspikes = numel(t); % number of spikes
        for ii = 1:nspikes % for every spike
                line([t(ii) t(ii)],[jj-0.5 jj+0.5],'Color','k');
                % draw a black vertical line of length 1 at time t(x) and at
                    trial jj (y)
        end
end

xlabel('Time (ms)'); % Time is in milliseconds
ylabel('Trial number');
```

This produces a figure similar to the previous raster plot. The number of spikes increase during tone presentation (between 300 and 450 ms) for about half of the trials (and after tone offset, there is another burst). You might notice that this neuron's activity is not very reproducible from trial to trial. This is to be expected, as neurons are usually tuned to specific stimulus features, and most trials contained different stimuli. However, even when repeating the same exact stimulus, the neural response will contain an element of randomness. In summary, there is inherent variability in neuronal data.

Spike Tuning Curve

While neuronal activity is inherently noisy, neurons still prefer particular values of certain features. For example, a neuron from the primary auditory cortex (A1) will response vigorously to the pure tones of a particular frequency, while the activity drops for tones of increasingly different frequencies. A "visual" neuron might respond preferentially for a particular specific orientation of a visual bar, or the velocity of a moving visual grating.

We will use the same data to plot the tuning curve of an A1 neuron in the auditory cortex. To obtain the stimulus values of the first trial, type:

```
load('tonespike'); % load data (struct called tonespike)
stim = tonespike(1).stimvalues % stimulus values
```

stim is a 4x1 vector. Which stimulus features do these 4 stimulus values represent? This is stored in the field *stimparams:*

```
params = tonespike(1).stimparams; % stimulus feature
```

This is a cell array of 4 strings: the first stimulus is frequency ('Freq [Hz]'), the second is sound level ('SPL [dB]'), the third sound duration ('Duration [ms]'), and the fourth speaker channel ('Channel'). For the current experiment, these features are the same for every trial. Here, we will create a frequency tuning curve, and so we are only interested in the first feature:

```
feature = stim(1); % frequency (Hz)
```

This first trial contains a tone of 250 Hz. How many spikes does the neuron produce during the presentation of this tone? For that, we need to select the spikes occurring during the stimulus presentation:

```
spiketime   = tonespike(1).spiketime; % stimulus was presented between 300 and
   450 ms
onset     = 300; % tone onset (ms)
offset   = 450; % tone offset (ms)
sel    = spiketime>=onset & spiketime<=offset; % selection vector!
Nspikes   = sum(sel); % number of spikes during tone presentation
```

The neuron spikes only twice (Nspikes) in this trial! As we can know from the raster plot of the data, the neuron's firing pattern varies idiosyncratically across trials. We are therefore usually not interested in single-trial spike numbers, but want to know how many spikes the neuron produces on average during all trials in which a 250-Hz tone is produced. To obtain this average, we first need to determine the tone frequencies that were presented in every trial. We need to do this using a for-loop:

```
ntrials = numel(tonespike); % number of trials
Feature = NaN(ntrials,1); % alaways initialize vector or matrix!
for ii = 1:ntrials
        values = tonespike(ii).stimvalues;
        Feature(ii) = values(1); % (Hz)
end
```

Note that we initialized a vector Feature before the for-loop by creating a vector containing only NaNs (Not-a-Numbers). In the loop this vector will be gradually (trial-by-trial) filled with the actual tone frequencies. Including this step is essential to reduce the time it takes for Matlab. It is also good practice: thinking about how large a vector or matrix should be increases your understanding of the data, and reduces programming errors.

Now we can select the trials in the tonespike structure with the 250 Hz tone presentation.

```
sel = Feature == 250; % select all trials with tone frequency uFeature(ii)
spiketime = [tonespike(sel).spiketime];
nrep = sum(sel); % number of stimulus repeats

onset = 300; % tone onset (ms)
offset = 450; % tone offset (ms)
sel = spiketime>=onset & spiketime<=offset; % stimulus was presented between 300
   and 450 ms
Nspikes = sum(sel); % number of spikes
```

Now we're getting somewhere: 170 spikes! Of course, this number is pretty useless, as it depends on the duration of the tone, the number of stimulus repeats and the sampling rate. A much more interesting number is the average firing rate of the neuron:

```
duration = offset-onset; % ms
Fs = 1000; % sampling rate (Hz)
firingRate = Nspikes/duration*Fs/nrep; % in spikes/s or Hz
```

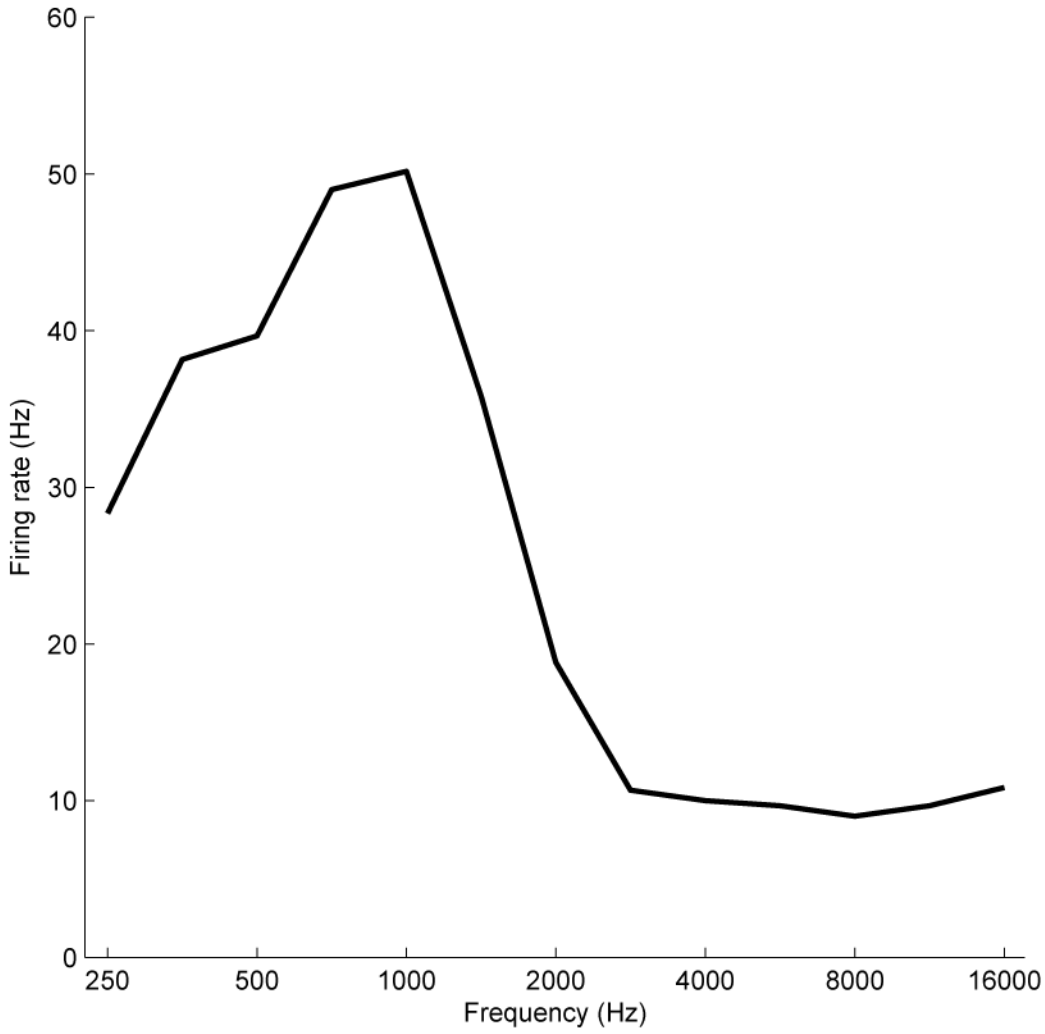
28 spikes/second! Finally, we need to do this for every tone to be able to determine the frequency tuning curve. Let's use a for-loop, and determine the firing rate for one frequency in every step:

```
uFeature    = unique(Feature); % the unique frequencies presented during the
   recordings
nFeature    = numel(uFeature); % the number of freqs
firingRate  = NaN(nFeature,1); % Initialization of firingRate vector
Fs      = 1000; % sampling rate (Hz)
onset    = 300; % tone onset (ms)
offset    = 450; % tone offset(ms)
duration  = offset-onset; % ms
for ii = 1:nFeature
        sel        = Feature == uFeature(ii); % select all trials with tone
           frequency uFeature(ii)
        spiketime    = [tonespike(sel).spiketime];
        nrep      = sum(sel); % number of stimulus repeats
        sel        = spiketime>=onset & spiketime<=offset; % stimulus was
           presented between 300 and 450 ms
        Nspikes      = sum(sel); % number of spikes
        firingRate(ii)  = Nspikes/duration*Fs/nrep; % in spikes/s or Hz
end
```

To plot this:

```
h = semilogx(uFeature,firingRate); % Plot
set(h,'Color','k','LineWidth',2,'LineStyle','-'); set(gca,'XTick',uFeature(1:2:
    end),'XTickLabel',round(uFeature(1:2:end))); box off
xlabel('Frequency (Hz)');
ylabel('Firing rate (Hz)');
axis square;
xlim([0.9*min(uFeature) 1.1*max(uFeature)]);
ylim([0 60]);
```

We have plotted the data with semilogx. This is similar to the plot function but the scale of the x-axis is logarithmic. The reason we use this is because auditory neurons respond to tones in a logarithmic fashion.



We can see in the tuning curve that a frequency of 1000 Hz produces on average the highest firing rate of 50 spikes/s. Or we can determine this in Matlab:

```
[mx,indx] = max(firingRate); % highest firing rate (spikes/s)
BF = uFeature(indx); % best frequency (Hz)
```