Exercise: Analysis of spike-to-LFP phase locking

In this exercise we will look at one specific experimental condition during which the LFP activity was recorded in parallel to one spiking neuron. We will try to detect whether the recorded neuron shows a non-uniform distribution of spike-triggered LFP phases.

Load the data

`data/lfp1.mat` contains three interesting variables stored in the dictionary under the keys `time`, `sf`, and `lfp_matrix`. `lfp_matrix` contains the actual LFP signal in uV, where rows are the individual trials of an experiment (first index), and columns are the time bins (second index) corresponding to the trial time stored in `time` (in ms). `sf` contains the sampling frequency in Hz.

`data/spikes1.mat` contains the spike times of multiple neurons stored in the dictionary under the key `spike_cell`. `spike_cell` contains the data as a 3-dimensional array: the 1st dimension indexes trials, the 2nd dimension indexes neuron IDs, and the 3rd dimension indexes spike order. Let's use only the spike times of the first neuron in this exercise (in fact, the other neurons have no spikes).

Visualize the data

Let's first visually inspect the data. Set the trial index (i.e. `idx_trial`) to zero (`idx_trial = 0`).

Task 1:

- Change the value of `idx_trial` and plot data for different trials. Do you recognize LFP oscillations and phase-locking of the spikes?
- Let's quantify the oscillatory properties of the LFP using the power spectrum density (PSD). You can compute the PSD estimated by Welch's method (with a segment length of 256 data points by default).

Task 2:

- Plot the PSDs of other trials. How they differ across trials?
- Compute the average of the PSDs of all trials and plot it. What does the result tell about the oscillations in the LFP? (Hint: it is known that the oscillations at 50 Hz, 70 Hz and 100 Hz are experimental noise.)

Filter the LFP signal

LFPs are often complex signals composed of different frequency contributions. Therefore, to estimate the relationship of spikes to the LFP, you typically need to first filter the signal in a frequency range of interest.

We'll use a Butterworth filter to filter each LFP trial separately. To this end, we first use the `scipy.signal.butter()` function to obtain a set of filter coefficients. As `order` of the filter (number of coefficients) a value of 5 should be sufficient (alternatively, the function `scipy.signal.butterord()` yields an estimate of the optimal order). You can design a bandpass filter between `f1` Hz and `f2` Hz using:

```
(coeffb, coeffa) = scipy.signal.butter(order, np.array([f1, f2]) / (sf/2.0), 'pass')
```

where `sf` is the sampling frequency. The return value are the filter coefficients that realize this particular type of filter.

The vector `[f1, f2]` is divided by `sf/2` since the filter frequencies are given in units of the Nyquist frequency, which is half the sampling frequency.

Butterworth filters are part of a class of filters (IIR) that does not preserve phase information, i.e., the filtered signal is shifted in time, but each frequency by different amounts. Therefore, we must be careful to apply an appropriate filter technique that corrects for the phase distortions of the filter, i.e., a filter algorithm that is phase neutral.

A good way to achieve such neutrality is to use the `scipy.signal.filtfilt()` function (instead of a more basic routine `scipy.signal.lfilter()`). Technically, it implements a zero-phase filtering technique by applying the filter a second time in a time-reversed manner, thus canceling any phase shifts induced by filtering. Both filter routines take the coefficients returned by the `butter()` function as argument:

```
filtered_signal = scipy.signal.lfilter(coeffb, coeffa, signal)
filtered_signal = scipy.signal.filtfilt(coeffb, coeffa, signal)
```

Task 3:

- Plot a filtered LFP trace superimposed on its original LFP trace. How do they compare?
- Try various pass bands based on the PSD of the LFPs. What would be a good choice?

Quantify spike-to-LFP phase locking

Finally, let's analyze the spike-to-LFP phase locking using a direct decomposition of the LFP signal into phase and amplitude. For this, we will first calculate the analytic signal $z(t)$ from the Hilbert transform $H[.]$ of the LFP series $x(t)$ as

$$z(t)=x(t)+iH[x(t)],$$

and then, calculate the amplitude $A(t)$ and the phase $\phi(t)$ as

$$A(t)=\text{sqrt}(x(t)^2+H[x(t)]^2)\ ,\phi(t)=\text{atan2}(H[x(t)],x(t)).$$

You can use the `scipy.signal.hilbert()` function to calculate the analytical signal from an array of real numbers. Confusingly, despite its name, this function directly returns the analytic signal, not just the Hilbert transform. For calculation of the amplitude and the phase, you can use the `numpy.abs()` and `numpy.angle()` functions, respectively. Note that you need to use a properly band-pass filtered signal to calculate its amplitude and phase, to obtain these measures for oscillations at a specific frequency.

Task 4:

- Plot the band-pass filtered LFP and its Hilbert transform as functions of time. How do they compare?
- Further plot the obtained amplitude as a function of time in the same plot. How is it related to the LFP and the Hilbert transform?
- Further plot the obtained phase as a function of time in the same plot (you would need to scale the vartical magnitude of the phase plot properly). How is it related to the LFP and the Hilbert transform? In particular, what phase values correspond to the peaks and the troughs of the LFP signal?

Now we can calculate the phase locking value (PLV) between the LFP and spikes. To this end, you first identify the phase value at each spike time, then collect the phase values across spikes, and finally calculate the PLV as follows:

$$\text{PLV} = \frac{1}{N}\left|\sum_{i=1}^{N} e^{i\phi_i}\right|,$$

where N is the total number of spikes and ϕi is the LFP phase value at the time of the i-th spike.

Task 5:

- Collect the spike triggered phase values from all trials and plot their distribution. What does the distribution tell about the phase preference of the spikes?
- Calculate the PLV from the phase values collected from all trials. What does the number tell about?
- Calculate the PLVs for various frequency ranges and compare them. In which frequency range do you find a maximal and a minimal PLV?

Acknowledgements

Data are courtesy of Dr. Alexa Riehle, Institut de Neurosciences de la Timone (INT), UMR 7289, CNRS - Aix Marseille Univ., Marseille, France.