Name and department/organisation of proposing academic: MRC Brain Network Dynamics Unit, Nuffield Department of Clinical Neurosciences

Name of project mentor: Dupret David (david.dupret@bndu.ox.ac.uk) and Vitor Lopes-dos-Santos (vitor.lopesdossantos@bndu.ox.ac.uk)

Project title: Spiking dynamics of neuronal populations in the hippocampus

Brief description of the background to the project: the hippocampus is a neuronal circuit of the mammalian brain central to memory-guided behaviour. Information processing in this circuit relies on the computation of temporally structured spiking activity of glutamatergic neurons referred to as principal cells, which typically fire around 1Hz (1 spikes per second). Such coordinated activity is also shaped by: (*i*) a set of diverse local interneurons, which typically fire above 10Hz (10 spikes per second), as well as (*ii*) various network oscillations that report neural inputs converging to the hippocampus and are detected in the local field potentials (LFPs). This project leverages from a dataset that consists in the spike trains of 68 neurons recorded simultaneously using *in vivo* multichannel extracellular techniques to access hippocampus network activity from a behaving mouse. By nature, this dataset allows a whole range of analyses, from the most basic calculation of the average firing rate of each individual neuron all the way to advanced, multidimensional descriptions of cooperative spiking forming cell assemblies.

Brief description of data available and format of data: the data are contained in one directory that corresponds to a single recording session, which processing yielded multiple files. We refer to the name of the directory as the "*basename*" (i.e., mvl10-200109 for mouse mvl10 recorded the 9th of January 2020). Each recording day typically consists of many recording sessions (i.e., mvl10-200109_2 is the second session). Here we provide the following files for one session:

<basename>_<session>.res
description: a one-column list with number of integer samples (starting at 0) at which the peak of a spike is to be found (i.e. an ordered list of spike times [/sample])
format: <int>
notes: the unit of time is the sampling rate of the electrophysiology rig, which here is 20kHz (hence 20,000 samples per second).

<basename>_<session>.clu
description: a one-column list reporting the cluster (i.e. detected "cell" or "unit") identity assignment for each of the detected spikes in the .res file.
format: <int>
notes: the first line is the total number of clusters recorded. Note also that both cluster 0 and cluster 1 are pre-processing related noise clusters, and should not be analysed. The rest of the values in the .clu file (i.e., from cluster 2 onwards) is the cluster identity of each spike in the .res file. Thus, the .clu file has one more line (the first one) than the .res file (the checksum). Just to be clear:
line 1 = 1 + number of recorded neurons
line 2:end = ordered list of clusters; each line stating the cluster that fired the corresponding spike event in the .res file
cluster 1 = noise (do not consider the corresponding .res samples)
cluster >1 = consider the corresponding .res samples (and their assigned .clu identifier) for further analyses

Warning: if you are using python, then clu[1:] is aligned with .res. If you are using Python (which would be the preferred approach), then check how to use index operators to retrieve an element of a list (because lists are "zero indexed").

<basename>.des
description: an ordered list that assigns a cell (pheno)type to each cluster id.
format: <type><location>
notes: here, by convention, pyramidal cells are labelled *p* and interneurons with *b* or *i*. A *u* stands for unknown. This is followed by a code that reports the anatomical location of the recorded cell. For instance, number 1 indicates that the cell has been recorded in hippocampal CA1. For example, a *p1* and a *b1* are a pyramidal cell and a basket cell recorded in CA1, respectively.

<basename>_<session>.theta.cycles.<electrode>
description: a *nx6* array saved to a binary file in NumPy .npy format. Contains timestamps for 6 references of *n* theta cycles detected in a session for a given electrode. Columns consist of: (0) first ascending zero crossing, (1) peak, (2) descending zero crossing, (3) trough, (4) second ascending zero crossing, (5) peak of next cycle. Such cycles were detected in local field potentials sampled at 1,250Hz (one sample recorded every 0.8 ms). Thus, timestamps in this file are expressed in 0.8-ms (1/1,250 seconds) units. Therefore, divide timestamps by 1.25 if you want to to convert them to milliseconds.
notes: theta oscillations (~8Hz in mice) are a prominent network phenomenon of the hippocampus observed in local field potentials of rodents and other mammals. Each theta cycle is thought to chunk the activity of hippocampal cells in discrete packets of elementary information units.

Question(s) to be asked using the data: from increasing levels of complexity (with no real upper limit!)

1. Load the spike times (stored in the .res file) along with their associated cell identity (stored in the .clu file for each recorded spike in the .res; the phenotype of which is reported in the .des file). Do you now have the same number of lines extracted from the .res and the .clu files? That is, are these two files correctly aligned in your analytical framework? Do not forget, cluster 1 should not be used. So spare memory allocation and processing time on your computer!
2. Compute the mean firing rate (in spikes per second) for each p1 and b1 cells and plot (using matplotlib for instance) the distribution of the firing rates for each of these two cell types. Can you name what type of distributions these are? What measures could you compute to mathematically describe such distributions?
3. Compute the distribution of inter-spike intervals (ISI) for each p1 and b1 cell. Then plot the average ISI distribution for both cell types (include standard error of the mean). Horizontal axis must be in milliseconds. Hint: make sure you convert spike times loaded from res to milliseconds).
4. Write a function to compute a 'binned spike count matrix' from the res and clu files. In this matrix, each row represents a cell and each column represents the number of spikes that a given cell has discharged in a given time window. The length of such a time window must be provided by the user in milliseconds (as an input to the function). Hint: use a function to compute histograms (such as np.histogram) to make your code more efficient.

5. Run a sanity check by creating a plot of your binned activity matrix. For example, you could plot a raster-gram for your cells on the top of the binned activity matrix in such a way you can check if the number of spikes of a given neuron corresponds to the counts displayed in the matrix for the same time periods.
6. Load the theta.cycles file (mvl10-200109_2.theta.cycles.7) for the corresponding session. You will use this data for subsequent analyses.
7. Compute the perievent time histogram for the activity of all p1 and b1 cells using the trough of theta cycles as the reference events. For this, use a binned activity matrix calculated with 5-ms time bins. Plot the average perievent time histograms for p1 cells and b1 cells.
8. Write a function to compute a 'theta binned activity matrix' containing p1 and b1 cells. Your bins will not have a fixed length as in item 4, but they will be defined by the natural time windows of theta cycles. Each time bin will be from the ascending zero crossing of a theta cycle to its descending zero crossing. Note that values in the theta.cycle file are stored as data points sampled at 1,250Hz (see data description above). Create a sanity check plot as in 5, but for this matrix. Use this matrix for the next items. Hint: note that np.histogram can be used even if you don't have fixed length time bins.
9. Run Principal Component Analysis (sklearn module is recommended) for the theta binned activity matrix containing only p1 cells and find how many principal components are needed to extract 95% of the variance of the data. Plot the eigenvalues of the obtained components.
10. Compute a GLM (linear regression from sklearn module recommended) to predict the spiking activity of each p1 cell by the activity of all p1 remaining cells. Before doing so, z-score the activity of each cell in your activity matrix (why does one need to do this?). Make sure you cross validate your predictions to avoid biases (that is, do not test your model in the same data sample as the one you used to train your model). More specifically, train your regression on 80% of your theta time bins, and apply this model to predict the activity on the remaining 20%. Do this iteratively until all theta cycles have been included in the testing set. Choose a metric to quantify the performance/accuracy in predicting the activity of each p1 cell. Use this proposed metric for the following questions.
11. Prediction accuracy in the previous item can be partially explained by global fluctuations (internal dynamics) of the firing rate in the hippocampus. Thus, the accuracy of the model might not be entirely coming from the model itself (i.e., the beta weights of your regression). A typical control for that is to re-run the prediction procedure but using a *surrogate* version of your predictor matrix (regression *independent variables*). In this specific example, the *surrogate* version preserves global rate fluctuations but destroys single unit information in your predictor matrix. Thus, to create a surrogate activity matrix, shuffle the activity of your predictor cells within each theta cycle (remember to use the cell-z-scored activity matrix before applying the shuffling). The shuffling must be random and independent for each cycle. Then re-run the prediction as in question 10 but using this surrogate predictor matrix. To test if your original predictions are statistically significant, run 100 surrogate predictions of each p1 cell and use these performances to estimate a null hypothesis distribution: i.e., the p value for the prediction of each p1 cell is calculated as the probability that the original prediction is equal or lower than a surrogate prediction. Report how many p1 cells could be statistically predicted for $p<0.05$. Is this number of significant predictions higher than what you expect by chance?

12. Design a surrogate procedure if you are to control auto-correlation biases in your predictions (instead of global fluctuations as in the previous item).

Best suited to: Beginner, Intermediate and Advanced

Potential statistical approaches needed: one can deploy many types of analyses on such dataset, including multivariate methods and machine learning. Here, for the sake of training, we suggest a few basic calculations followed by Principal Component Analysis on correlation/covariance matrices and General Linear Models.

Software needed: it is important to note that this type of text files is not suited for Excel-based analyses due to their substantial length. We recommend performing the analyses (suggested above) using Python, R, matlab, or equivalent frameworks. To help, we provide the beginning of an example Jupyter notebook that you may wish to use as a backbone for subsequent analyses.

Recommended reading:

1. V. Lopes-dos-Santos, S. Ribeiro, A. B. L. Tort, Detecting cell assemblies in large neuronal populations. *J. Neurosci. Methods*. 220, 149–166 (2013).
2. K.D. Harris, J. Csicsvari, H. Hirase, G. Dragoi, G. Buzsáki, Organization of cell assemblies in the hippocampus. *Nature*. Jul 31;424(6948):552-6. (2003).
3. S. Trouche S, V. Koren, N.M. Doig, T.J. Ellender, M. El-Gaby, V. Lopes-Dos-Santos, H.M. Reeve, P.V. Perestenko, F.N. Gara, P.J. Magill, A. Sharott, D. Dupret. A Hippocampus-Accumbens Tripartite Neuronal Motif Guides Appetitive Memory in Space. Cell. Mar 7;176(6):1393-1406.e16. (2019).

Note: the answers to these questions will be provided on request, but at the end of the training session. Should you need any help with the coding, please email Vitor.