Introduction to Cognitive Science

Project Designers: *Alireza Mahdavi, Kowsar Shams, Reza Salamat*
Instructor: Mohammadreza A. Dehaqani

Spring 2025

Assignment 2

Phase One

(1) Introduction Electrophysiology, a specialized subfield of biology, focuses on the investigation of the electrical characteristics of biological cells and tissues. Within the realm of neuroscience, this discipline is extensively employed to monitor the electrical activity of neurons in the brain. The data derived from such recordings play a vital role in elucidating the mechanisms by which the brain processes information and orchestrates behavior. The capacity to capture electrical signals at the level of individual neurons has profoundly transformed our comprehension of neural function, giving rise to a wide array of advanced methodologies for examining brain activity. Recent technological progress in electrophysiological methods has further enabled simultaneous recordings from extensive neuronal populations, offering novel and comprehensive perspectives on neuronal network dynamics. Consequently, electrophysiological data have become a cornerstone of contemporary neuroscience research, contributing significantly to numerous groundbreaking discoveries concerning brain function.
Keywords: electrophysiology, neuronal data, neural data, mutual information, d-prime, single unit activity, multi unit activity, population-level decoding, spike sorting, response dynamics, neural discriminability

(2) Spike Sorting from Scratch When recording extracellular signals from a specific cortical region, researchers are often interested in examining both the electrical activity of individual neurons and the collective electrical field generated by a surrounding population of neurons near the implanted electrode. However, the raw recorded data do not inherently distinguish the activity of individual neurons. To overcome this limitation, a series of mathematical and computational techniques can be applied to accurately identify the spike patterns of neurons located in close proximity to the electrode. This analytical process, which isolates and classifies the activity of single neurons from extracellular recordings, is referred to as "spike sorting."
As depicted in Figure 1, there are several steps involved in the process of spike sorting. In a nutshell, those steps are the as follows:

- **Filtering**:Applying a bandpass filter filter between 300Hz and 3000Hz. Recorded electrical activities below 300Hz is usually referred to as low-frequency potential or “LFP”.
- **Spike detection**: Following the previous step, spikes are detected by applying an amplitude threshold on the filtered signal. The threshold in this step should be chosen carefully, for picking higher values could lead to detecting no spikes, and choosing a relatively low value could lead to false-positive results due to noise crossing that threshold.
- **Feature extraction**: When recording extracellular activities, neurons that are in vicinity of the inserted electrode are often observed to exhibit distinguishable patterns of electrical activity when they perform an action-potential. This gives us a clue to separate each detected spike into clusters each with almost similar waveform features/patterns. In other words, we employ feature extraction methods in order to transform spike waveforms into more informative feature-set of smaller dimensionality.
- **Clustering**: The last step in spike sorting is to group the detected spikes into few clusters based on their extracted features. There are many clustering algorithms proposed for spike sorting, some of which you will use in this assignment.

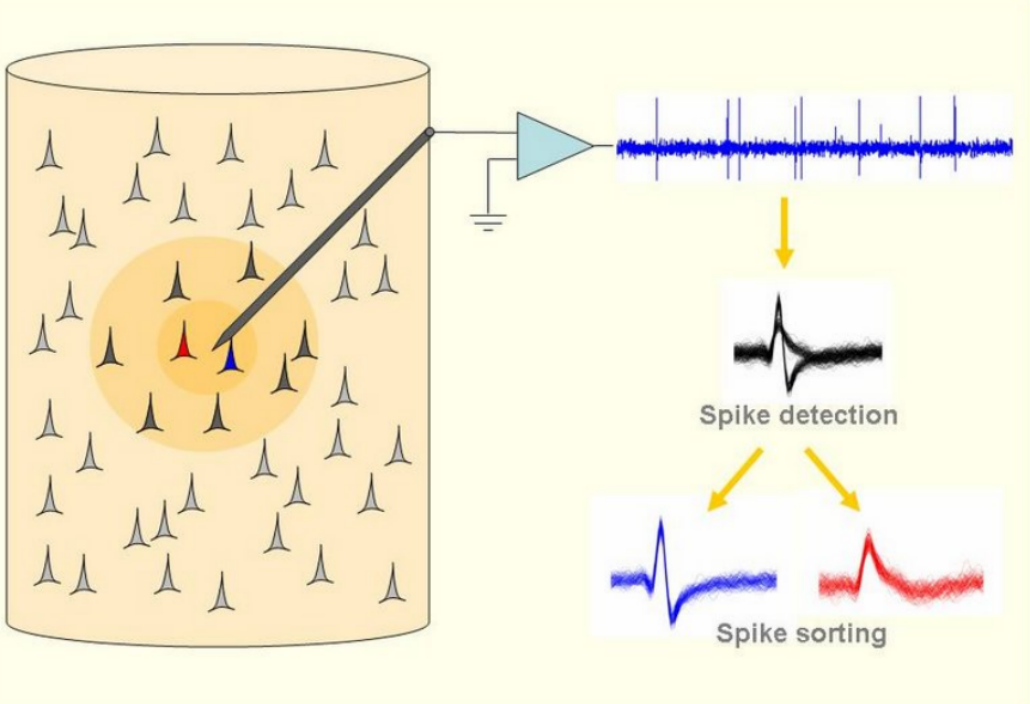

Figure 1: An illustration of the steps involved in spike sorting.

In this question, you are to perform spike-sorting analysis on a single channel recording of a population of simulated neurons. Don't worry! You will be instructed through every steps of this analysis like a cookbook recipe! But, before proceeding any further, please read this web page: Click here to visit the website

Getting Started

1. Load the dataset stored in `singleIT.mat`. Each data point is associated with the voltage amplitude recorded at a certain time (sampling rate of the recording is 30 kHz). Then, plot a diagram illustrating the amplitude against time.
2. Plot the histogram of the recorded voltage amplitudes for the entire dataset. What can be inferred about the background noise by looking at this diagram (distribution, etc.)?

Filtering the Data

1. Design a highpass Butterworth filter at 300Hz. Set the filter order to 7. Then, apply the filter on the data using MATLAB's `filtfilt` command. If you prefer Python, you can execute this command: `scipy.signal.filtfilt`.
2. Plot two diagrams depicting: 1) the unfiltered data and 2) the data after applying the filter.

Detecting the Spike

1. Using the following equations, calculate the voltage threshold (θ).

$$\theta = 5\sigma_n, \quad \sigma_n = \text{median}\left(\frac{|\mathbf{x}|}{0.6745}\right) \tag{0.1}$$

2. Extract the peaks in the data. A certain point in the data is considered a peak if the first derivative of former and later data points with respect to that certain point differ in sign.

3. By comparing the amplitude of each peak point to θ, select each point whose amplitude is greater than or equal to the threshold.Next, extract the waveforms for each detected spike from the filtered signal. The waveform associated with each spike is a short timeseries that includes the data-points from 2ms before the peak to 2ms after the peak.

4. Plot a diagram depicting the waveform for every detected spike in one single diagram. Are there any noticeable difference among the overall shapes of these waveforms? Store these waveforms in a 2D matrix for the next step.

Extracting Features

1. Apply PCA on the waveforms matrix. In MATLAB, you can simply call pca command with proper arguments. Equally, you can invoke `sklearn.decomposition.PCA` function if you are programming in Python.

2. By comparing the score for each principal component, choose the three most informative components (PC_1, PC_2, PC_3) for the next part.

Clustering the Spikes

1. Using K-Means algorithm, cluster the waveforms based on PC_1, PC_2, and PC_3 features. You are not required to implement this algorithm yourself.

2. Visualize the results by depicting three scatter plots for every possible pair of PC_1, PC_2, and PC_3 features. Each dot in these plots represents one waveform whose color indicates the cluster its associated waveform belongs to.

3. Repeat part (1) and (2) with different values of K for the clustering algorithm. Which value of K gives the best results? Explain why.

Now, answer the following questions. Your must justify your answers by thoroughly analyzing the results and presenting proper diagrams in your report.

1. Load `spikes.mat file.` This file contains a vector of time points in which true action potentials have occurred. Using this data and the spikes you detected previously, evaluate the performance of this spike-sorting pipeline. It is up to you to come up with a proper evaluation metric. Justify your answer by plotting diagrams and analyzing your observations

2. Use the following equation to determine a new threshold for detecting spikes (θ_new).X_t is the amplitude of data at time t.

$$\theta_{\text{new}} = 0.9 \times \max_{0 \le t \le T}(X_t) \tag{0.2}$$

   Next, repeat the subsequent steps to sort the detected spikes. Do you think choosing the new threshold (θ_new) improved the results of spike-sorting? Justify your answer.

3. Instead of using PCA algorithm to extract features, use tSNE algorithm (For MATLAB use tsne command and for Python, you may use `sklearn.manifold.TSNE`) and repeat the subsequent steps of spike-sorting pipeline. Do you observe any improvement in the results? Justify you answer.

(3) Spike Sorting from Scratch As we learned in the previous question, a crucial step in analyzing extracellular data is to differentiate between different neuron activities. However, manually sorting spikes can be extremely time-consuming and laborious, especially for studies that involve multiple recording sessions. To address this issue, various toolboxes and software have been developed to facilitate the spike sorting process. One such software is ROSS! ROSS1 is a MATLAB- 1:Please feel free to star its repository on GitHub it would make the developers very happy! based offline spike sorting software that enables researchers to perform automatic and manual spike sorting tasks efficiently. It provides various functions to modify the automatic sorting results, such as merging and denoising, and offers useful visualizations to help achieve better results.

Figure 2: Please feel free to star its repository on GitHub; it would make the developers very happy!.

Before starting, it is recommended to read the toolbox documentation. The data for this question are provided in file `ross-data.mat`. In this question, you will work with ROSS as follows:

Detection

The first step is activity detection. In this step, spikes are extracted from the bandpass filtered signal. A common way to detect spikes is to determine activities that cross a threshold. The threshold is calculated based on the estimated noise power. You can adjust the default options, such as filter type and thresholding method, to ensure an accurate detection phase. The detected spikes and their corresponding times of occurrence can be saved as a MAT file.

1. Load the raw extracellular data and adjust the provided settings for filtering and thresholding. Then, by clicking the "Start Detection" button, the detection results will appear in a PCA plot.

Auto Sorting

Auto sorting is a clustering method used to assign each spike to the corresponding neuron automatically. Here, a mixture of t-distributions, GMM, k-means, and template matching for clustering purposes is used. Additionally, a statistical filtering method for noise (outlier) removal and a rich alignment method are provided. The output of this section is cluster indices for each spike waveform, which can be saved as a MAT file.

1. Use the auto-sorting feature and observe the results in PCA and waveform plots. How many clusters exist in the data?

Manual Sorting

Manual sorting provides several tools to modify the auto sorting output to improve it. These tools include Merge, Delete, Resort, Denoise, and Manual grouping or deleting in PCA domain of clusters. Additionally, you can tag the neurons with arbitrary comments for further analysis. The cluster indices, alongside the neuron tags, can be saved as a MAT file.

1. To make manual changes to the automatic results, select the 'Manual Sorting' tab. Then, select the 'Denoise' option and set the data plot percentage and denoising threshold to 85.

2. Select one of the clusters and apply the automatic sorting process to it. Does the selected cluster divide into new clusters? If so, how many?

Visualization

The software provides a rich set of visualization tools to help you better understand the analyzed extracellular data. These tools include, but are not limited to, Inter-spike interval, Neuron lifetime, Waveforms, 3D plot, and PCA domain plots. You can also track detected spikes on the raw data. In the 3D plot, you can choose arbitrary features for each axis, such as principal components and time of occurrence. Each neuron is assigned a unique color across all visualizations in the software, which is shown in the legend section of the main window.

Here are some suggested visualization tasks to perform:

1. Plot the waveforms of the different clusters in separate plots and compare them to each other.
2. Use a 3D plot to display the first two PCA components over time.
3. Plot the entire raw data with the detected spikes and compare the waveform shapes of the different clusters.

These visualization tools will allow you to gain a deeper understanding of the data and provide valuable insights that can help improve your analysis.

(4) Analysis of Single Neuron Activity Understanding how individual neurons encode and process information is a fundamental question in neuroscience. One way to study neural responses is by analyzing patterns of action potential firing times, which serve as a code for conveying information. Single neuron analysis is a powerful tool for investigating the information processing capabilities of individual neurons, and two commonly used metrics are d-prime and mutual information.

This project uses neural recordings from the **inferior temporal (IT) cortex** of macaque monkeys, a brain region critical for visual object recognition. The data were collected while the monkeys viewed various visual stimuli, categorized into four semantic groups: *face*, *body*, *natural*, and *artificial* objects.

The main dataset, `vasatiData.mat`, is a 3D matrix of size **92** $\times$ **5000** $\times$ **550**, where:

- **92** is the number of recorded neurons,
- **5000** is the number of trials,
- **550** represents time points per trial, recorded at a **sampling rate of 1 kHz** (i.e., 550 ms duration per trial).

Each neuron's spiking activity can be represented as a *raster plot* of size 5000 $\times$ 550, showing activity over time across all trials.

The file `Trials.mat` contains the stimulus index shown in each of the 5000 trials, providing trial-level stimulus labeling shared across all neurons.

The file `StmLabels.mat` provides the semantic category of each stimulus, using the following mapping:

- **0**: Face
- **1**: Body
- **2**: Natural
- **3**: Artificial

This structure allows for categorization of trials and enables the analysis of how neurons in the IT cortex represent different visual stimulus categories. The dataset is well-suited for studying cognitive processing, neural selectivity, and the temporal dynamics of visual categorization.

PSTH Analysis

As part of the analysis, you are required to generate **Peri-Stimulus Time Histograms (PSTHs)** to visualize the temporal firing patterns of neurons in response to different stimulus categories.

- Select **at least one neuron** from the dataset.
- Use the `Trials.mat` file to group the trials based on the presented stimulus.
- Use the `StmLabels.mat` file to assign each stimulus to one of the four semantic categories:
  - **0**: Face
  - **1**: Body
  - **2**: Natural
  - **3**: Artificial

- For each category, compute and plot the PSTH using the corresponding subset of trials.

The PSTHs should show the average firing rate of the selected neuron over time (aligned to stimulus onset), separately for each of the four categories. This visualization helps in understanding how the neuron's response varies with different types of visual input.

Questions:

- Based on the PSTHs, which category (or categories) does the selected neuron respond to most strongly? Describe the neuron's firing behavior and discuss whether it appears to be *category-selective* (i.e., more sensitive to faces, bodies, natural, or artificial objects).
- **Bonus** [1]**:** Prove that the Peri-Stimulus Time Histogram (PSTH) is a sufficient statistic for estimating the rate parameter $\lambda(t)$ of a Poisson process. Make sure you actually understand your proof if you choose to do it! You will be thoroughly questioned on it.

Fano Factor Analysis

In addition to the PSTH analysis, you are required to examine the variability of the neural response across trials using the **Mean-Matched Fano Factor (MMFF)**.

The **Fano Factor** is a common measure in neuroscience used to quantify the trial-to-trial variability of neural spike counts. It is defined as the ratio of the variance to the mean of spike counts within a specified time window:

$$\text{Fano Factor} = \frac{\text{Var}(N)}{\text{E}(N)}$$

where N is the spike count in a given window. A Fano Factor of 1 indicates Poisson-like variability, values greater than 1 indicate higher variability, and values less than 1 suggest more regular firing.

The **Mean-Matched Fano Factor** method controls for differences in mean firing rate by comparing Fano Factors across conditions after matching them based on their firing rate distributions [2].

- Compute the spike count for each trial within a fixed time window (e.g., 100–300 ms post-stimulus onset).
- Use the provided code to calculate the **mean-matched Fano Factor** for each category: Face, Body, Natural, and Artificial.
- Plot the mean-matched Fano Factor values for all four categories.

This analysis allows you to assess whether the variability in the neuron's response is modulated by stimulus category, independent of differences in average firing rate.

Question: Which category (or categories) shows the highest or lowest variability in neural response after mean-matching? What might this suggest about the neuron's reliability and selectivity in processing those stimuli?

For more details on the method and interpretation, refer to the work described in [2].

Category Classification Using SVM

In this section, you will evaluate how well the firing rate patterns of a neuron (or neurons) can be used to classify stimulus categories using a **Support Vector Machine (SVM)** classifier.

You are provided with example code in both **Python and MATLAB** to help you implement the SVM classification and evaluate the results. You may modify and extend this code to suit your analysis.

- Extract **firing rate features** by computing spike counts within a fixed time window (e.g., 100–300 ms post-stimulus onset) for each trial.
- Use the category labels from `StmLabels.mat` to assign each trial to one of the four stimulus categories:
  - 0: Face
  - 1: Body
  - 2: Natural

[1] Applied only this assignment.

  - 3: Artificial

- Train an SVM classifier using these firing rate features. Multiclass classification can be performed using strategies such as one-vs-rest or one-vs-one.
- Use cross-validation (e.g., 5-fold) to assess generalization performance.
- Plot a bar graph showing the **overall classification accuracy**.
- Also, compute and plot the **recall** for each category to evaluate how well the model identifies each type of stimulus.

This classification task allows you to test whether the information in the neural responses is sufficient to decode stimulus category, and whether certain categories are more easily distinguished than others.

Questions:

- How accurate is the SVM classifier in predicting stimulus categories from neural activity?
- Which categories are best and worst recognized (based on recall)?
- What does this imply about how different visual categories are represented in the neural data?

Time-Time Decoding Analysis

In this section, you will perform a **time-time decoding analysis** to investigate how neural representations evolve over time and how information about stimulus category is maintained or transformed.

This method involves training an SVM classifier at one time point and testing it at every other time point, producing a two-dimensional decoding matrix.

- Use the codes from the previous SVM section as a basis for implementing time-time decoding.
- For each trial, extract neural activity in short time bins across the full duration (e.g., 0–550 ms post-stimulus).
- Train an SVM classifier at a specific time bin and test it across all other time bins, repeating this process across the entire time range to generate a 2D decoding matrix.
- Perform statistical testing (e.g., permutation test, cluster-based correction, or FDR) to identify **significantly above-chance regions** in the decoding matrix.
- Visualize the result as a 2D heatmap, with significant areas clearly marked.

Questions:

- Which time periods contain reliable information for decoding stimulus category?
- Do the significant regions in the decoding matrix lie mainly along the diagonal (suggesting *feedforward dynamics*) or are there off-diagonal regions as well (indicating *recurrent or sustained processing*)?
- Interpret the observed decoding dynamics in light of cortical processing theories, referring to [3].

For theoretical background and methodology, refer to [3].

Mutual Information Analysis Across Time

In this section, you will quantify how much information about the stimulus category is contained in the neural activity over time using **Mutual Information (MI)**.

Mutual Information measures the dependency between neural responses and stimulus labels, offering an information-theoretic approach that does not rely on a classifier. It captures both linear and non-linear relationships between variables.

- For each time point (or short time window), compute the MI between the spike count of the neuron(s) and the stimulus category labels.

- Use appropriate binning or kernel density estimation techniques to estimate probability distributions for MI calculation.
- Plot the mutual information as a function of time (e.g., from 0 to 550 ms post-stimulus).
- Use permutation testing (e.g., shuffling labels) to create a null distribution and determine statistically significant time points.
- Highlight the time periods with significant mutual information on the plot.

This analysis allows you to track how informative the neural activity is about stimulus identity across time.
Questions:

- At what time(s) is mutual information about category the highest?
- How does this result compare to your decoding and Fano Factor analyses?
- What does this tell you about the timing of category-selective information in inferior temporal cortex?

Bonus[2]: Category Discriminability Using d-prime

In this section, you will quantify how well a neuron can distinguish between different stimulus categories using the **d-prime** (d') metric from signal detection theory.

The d-prime value measures the separation between the response distributions to two different categories and is defined as:

$$d' = \frac{\mu_1 - \mu_2}{\sqrt{\frac{1}{2}(\sigma_1^2 + \sigma_2^2)}}$$

where μ_1 and μ_2 are the means, and σ_1 and σ_2 are the standard deviations of the spike counts for two different stimulus categories.

- Select pairs of categories (e.g., Face vs. Body, Face vs. Artificial, etc.).
- Compute spike counts in a fixed post-stimulus window (e.g., 100–300 ms).
- For each pair of categories, calculate the d-prime value based on the spike count distributions.
- Plot the resulting d-prime values as a bar plot for each category pair.

This analysis provides a measure of how selectively a neuron responds to different visual categories and complements your classification and mutual information analyses.
Questions:

- Which category pairs are best discriminated by the neuron?
- Are the results consistent with the SVM and mutual information findings?
- What does the d-prime pattern tell you about category selectivity in the neural population?

[2] Applied only this assignment.

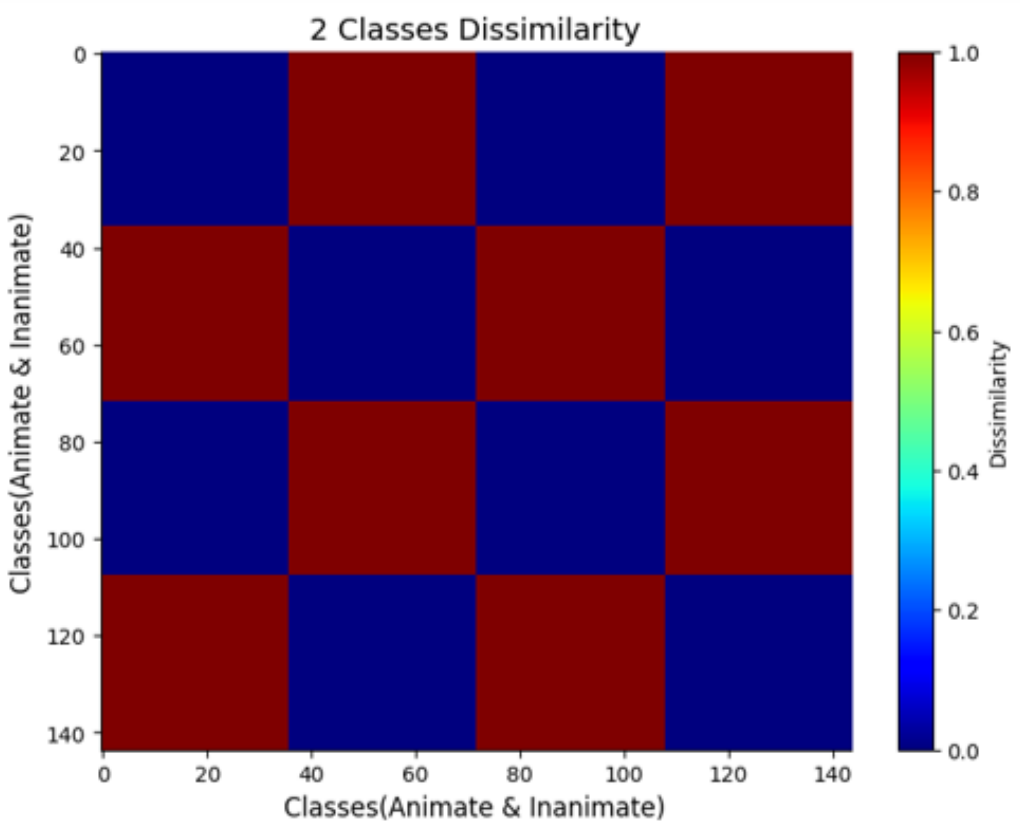

Figure 3: An example of a Ground truth model.

(5) Analysis of Population Activity Population decoding is a technique commonly used in neuroscience to infer the stimulus or action being performed by an animal based on the activity of a population of neurons. The basic idea behind population decoding is that different stimuli or actions evoke different patterns of neural activity, and these patterns can be decoded to infer the underlying stimulus or action.

Representational Dissimilarity Matrix (RDM) and Kendall's Tau Correlation

In this section, you will explore the representational geometry of the neural responses using a **Representational Dissimilarity Matrix (RDM)** and evaluate how well the neural structure aligns with the ground truth stimulus category structure over time.

- For each time slice (or window), compute the neural response patterns (e.g., average firing rate or spike count) for each stimulus or category.
- Construct an RDM by computing pairwise dissimilarities (e.g., Euclidean distance or 1 - correlation) between response patterns of all stimulus conditions.
- Repeat this process for each time slice to obtain time-resolved RDMs.
- Create a **ground truth RDM** based on category labels: entries in this matrix should reflect whether two stimuli belong to the same or different categories.
- For each time slice, compute the **Kendall's tau correlation** between the neural RDM and the ground truth RDM.
- Plot Kendall's tau correlation as a function of time.

This analysis allows you to assess how well the representational structure of neural activity matches the underlying stimulus categories over time.

Questions:

- At which time points does the neural RDM best match the categorical structure?
- How does this result relate to your decoding and mutual information analyses?
- What does the temporal evolution of the RDM alignment tell you about the emergence of category representations in the inferior temporal cortex?

Bonus [3]: Generalized Linear Model (GLM) to Predict Ground Truth RDM

In this optional advanced section, you will apply a **Generalized Linear Model (GLM)** to model how well the **neural RDM** and additional **visual features** explain the ground truth category structure.

What is a GLM?

A Generalized Linear Model (GLM) is a flexible extension of linear regression that allows for modeling of data that may not follow a normal distribution. It can be used to predict a dependent variable (e.g., entries in a ground truth RDM) from one or more independent variables (e.g., neural RDM values, image feature distances).

Task:

- Flatten the upper triangle (or full vectorized form) of the **ground truth RDM**, the **neural RDM**, and the **visual feature RDM**.
- Construct a GLM where the target variable is the vectorized ground truth RDM, and the predictors are the neural RDM and the visual feature RDM (e.g., extracted from CNN features or low-level statistics of the stimuli).
- Fit the model for each time slice to examine how well neural and visual features explain the ground truth dissimilarities.
- Plot the regression weights or explained variance (e.g., R^2) over time.

This analysis will help you determine whether the brain's representational structure aligns more with low-level visual features or with higher-level categorical distinctions, and how this alignment changes over time.

Questions:

- Does the neural RDM explain unique variance in the ground truth beyond what visual features can?
- At what time points does the neural representation reflect higher-level categorical information versus lower-level visual features?
- What does this tell you about the transformation from visual input to categorical encoding in the inferior temporal cortex?

[3] Applied beyond this assignment to overall assignments grade.

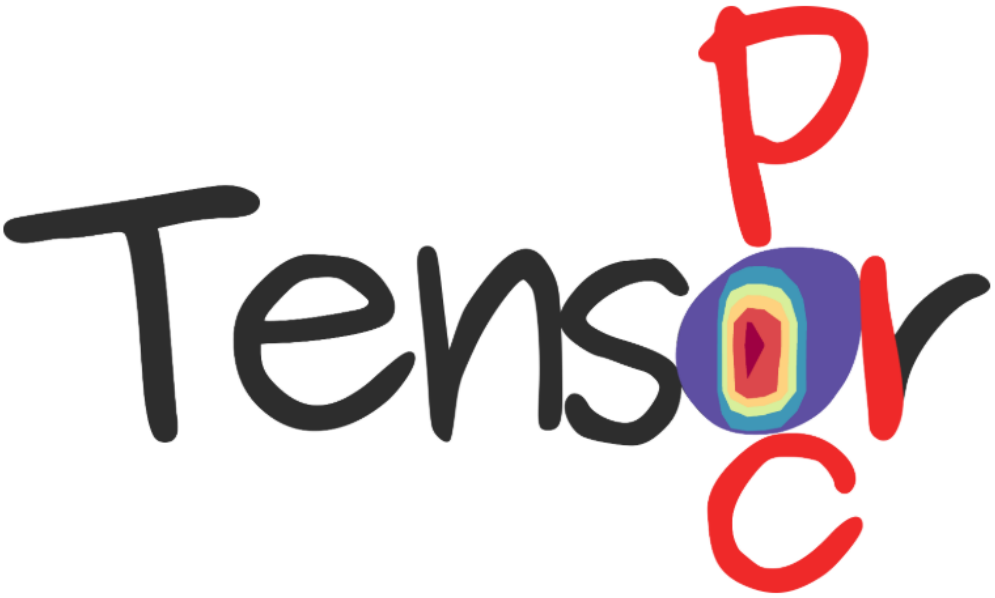

Figure 4: Tensorpac is an open-source Python toolbox for computing Phase-Amplitude Coupling (PAC) using tensors and parallel computing. This software provides a modular implementation which allows one to combine existing methods for measuring PAC and chance distribution.

(6) Phase-Amplitude Coupling (PAC) and Spectrum Analysis In this section, you will investigate Phase-Amplitude Coupling (PAC) in the neural data across different stimulus categories. PAC quantifies how the phase of a low-frequency oscillation modulates the amplitude of a high-frequency oscillation, and it is a key feature of neural communication and coordination.

What is PAC?

Phase-Amplitude Coupling (PAC) measures the interaction between low-frequency and high-frequency components of the neural signal. It is often studied in brain regions where information processing is thought to be organized in frequency bands (e.g., theta-gamma coupling).

Mathematically, PAC is defined as:

$$\text{PAC}(f_L, f_H) = \langle A(f_H) \cdot \cos(\theta(f_L)) \rangle$$

Where: - f_L and f_H are the low and high-frequency bands, respectively. - $A(f_H)$ is the amplitude of the high-frequency oscillation. - $\theta(f_L)$ is the phase of the low-frequency oscillation.

What is LFP?

Local Field Potentials (LFPs) represent the extracellular potential generated by the summed activity of a large population of neurons. These signals are often used to study large-scale neural activity and to infer functional connectivity within brain regions.

Plotting PAC and Spectrum

You will use the `Tensorpac` (Click here to visit the website) library to analyze PAC and plot the spectral characteristics of the neural data for the four stimulus categories (Face, Body, Natural, Artificial).

- First, compute the PAC for neural data from one or more neurons for each of the four categories. You should compare at least two methods for computing PAC using the Tensorpac library:
  - Method 1: *Modulation Index (MI).*
  - Method 2: *Canolty Method.*
- Plot the PAC results for each category (Face, Body, Natural, and Artificial). This should include PAC values across different frequency pairs (low and high frequencies).
- Additionally, compute and plot the power spectrum for any one of the categories to explore how frequency-specific activity is distributed in the neural signal.

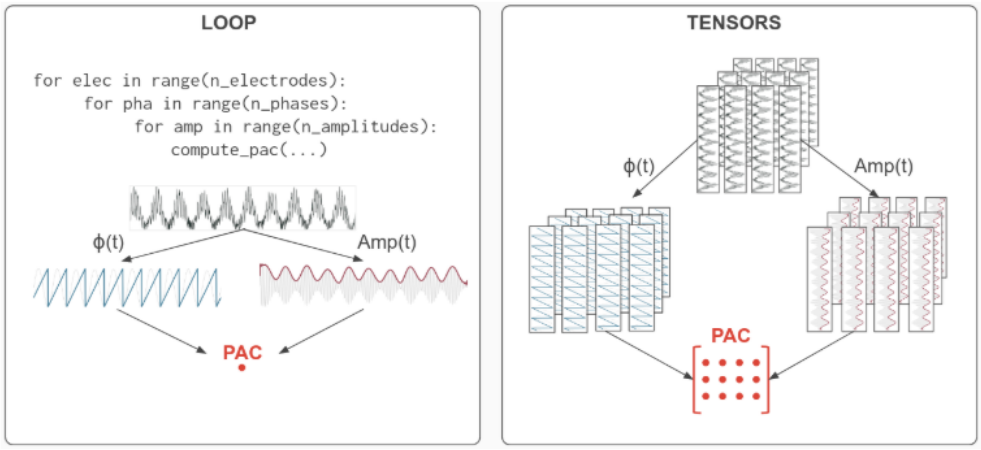

Figure 5: On the left, a traditional loop implementation to compute PAC between vectors. On the right, an illustration of the tensor-based implementation.

Bonus: If you implement both methods (without using the library) for PAC and compare them in your analysis, you will get bonus points on the total assignment scores.

Questions:

- What are the PAC patterns across the different stimulus categories (Face, Body, Natural, Artificial)? Do certain categories exhibit stronger PAC in specific frequency bands?
- How do the two PAC methods compare in terms of the detected coupling strength and its interpretation?
- How does the spectrum of neural activity vary across categories? Which frequencies are most active for each category?
- Do you observe any significant differences in PAC between categories, and what might this suggest about neural representation in different sensory contexts?

For theoretical background and methodology, refer to [5].

Phase Two (40 bonus [4])

(1) Overview of the Dataset In this part, you will be working with electrophysiological data recorded from a non-human primate (Macaca mulatta) during a visual task. The goal of the experiment was to investigate how neurons in the inferotemporal cortex (ITC) respond to synthetic and natural images, using a closed-loop paradigm [6]. Please note that this is not the full recording dataset, but rather a trimmed version prepared for the purposes of this assignment.

The dataset consists of three main components:

1. **spike_times**: Contains timestamps of neural spikes recorded from a specific brain region. The activity may correspond to one or more neurons.
2. **tags**: Indicates discrete events that occurred during the recording session. Details are explained below.
3. **time_for_each_tag**: For every tag, this array contains the timestamp at which the tag was recorded.

Each recording session (referred to as a *block*) consists of two parts: a **main section** and an **auxiliary section**.

- In the **main section**, 64 images generated by a generative model are presented to the monkey.
- In the **auxiliary section**, a smaller number of natural images (30–50), such as faces, bodies, and objects, are shown.

Each image presentation is called a *trial*, which includes 100 ms of image display, followed by 100 ms of a blank screen.

Section boundaries are indicated using tags:

- Start and end of the main section: tags 226 and 227
- Start and end of the auxiliary section: tags 228 and 229

Each trial has associated tags:

- Tag 100: stimulus onset
- Tag 200: stimulus offset
- Tag 223: trial was successful
- Tag 224: trial was unsuccessful

The goal is to obtain **64 successful** trials in each main section.

Important: If the main section of a block does not contain exactly 64 successful trials, you should consider the entire block invalid and assign a value of 0 to the neurons' firing rates in your analysis.

Timestamp Information: Timestamps in `time_for_each_tag` and `spike_times` are numerical values indicating the number of recorded samples from the start (i.e., not in seconds or milliseconds). The sampling rate is **30 kHz**. For example:

- A tag recorded 1 second after the start has timestamp = 30,000
- A spike, recorded 2 seconds after the start has timestamp = 60,000

Here is a simplified example:

```
tags = [226, 100, 200, 223, 100, 200, 224, 227]
time_for_each_tag = [0, 3000, 6000, 7000, 10000, 13000, 14000, 15000]
spike_times = [3200, 3250, 6200, 10200, 13300, 13500]
```

[4] Applied beyond this assignment to overall assignments grade.

Interpretation:

- `tags[0] = 226` at time 0 ⇒ Start of the *main section*
- First trial:
  - Tag 100 at 3000 → stimulus shown
  - Tag 200 at 6000 → stimulus ends
  - Tag 223 at 7000 → successful trial
- Second trial:
  - Tag 100 at 10000 → stimulus shown
  - Tag 200 at 13000 → stimulus ends
  - Tag 224 at 14000 → unsuccessful trial
- Tag 227 at 15000 → End of the *main section*

Spiking activity:

- During first trial (3000–7000): spikes at 3200, 3250, 6200
- During second trial (10000–14000): spikes at 10200, 13300, 13500

(2) Tasks

1. As mentioned, the goal was to replicate the design described in the closed-loop[6] paper. Read the paper thoroughly, and using what you've learned in phase 1—as well as your own creativity—try to clearly demonstrate that why the closed-loop experiment was successful.
2. In your opinion, what alternative criteria could we use as a fitness function in the genetic algorithm? What would using such a criterion imply? In other words, what neuronal property is targeted by the this process, and how can the pictures in last generation be interpreted?

   Note: There is no single correct answer here. What matters most is your **creativity** and your understanding of the topic. Give ChatGPT and other bots a break, and think for yourself!

References

[1] Ramin Toosi, Mohammad Ali Akhaee, and Mohammad-Reza A. Dehaqani. "An automatic spike sorting algorithm based on adaptive spike detection and a mixture of skew-t distributions." *Scientific Reports* 11.1 (2021), pp. 1–18.

[2] Churchland, M. M., et al. *Stimulus onset quenches neural variability: a widespread cortical phenomenon.* Nature Neuroscience, 13(3), 369–378, 2010.

[3] King, J.-R., & Dehaene, S. *Characterizing the dynamics of mental representations: the temporal generalization method.* Trends in Cognitive Sciences, 18(4), 203–210, 2014.

[4] Etienne Combrisson, Timothy Nest, Andrea Brovelli, Robin A. A. Ince, Juan L. P. Soto, Aymeric Guillot, and Karim Jerbi. "Tensorpac: An open-source Python toolbox for tensor-based phase-amplitude coupling measurement in electrophysiological brain signals." *Journal of Neuroscience Methods* 2021.

[5] Canolty, R. T., Knight, R. T. (2010). The functional role of cross-frequency coupling. *Trends in Cognitive Sciences, 14*(12), 506–515. https://doi.org/10.1016/j.tics.2010.09.00

[6] Ponce, C. R., Xiao, W., Schade, P. F., Hartmann, T. S., Kreiman, G., & Livingstone, M. S. (2019). *Evolving images for visual neurons using a deep generative network reveals coding principles and neuronal preferences.* Cell, 177(4), 999–1009.

Final Notes

- All reports and analysis scripts must be developed individually; collaboration is not permitted.
- Submit your assignment as a ZIP file containing a PDF named `report.pdf` and a folder named `scripts` that includes all analysis codes.
- Embrace this project as a chance to explore single-neuron dynamics deeply, leveraging mathematical rigor to decode neural activity. Remember, these analytical techniques are powerful tools—not just ends in themselves.
- At the same time, you're encouraged to look beyond formulas and numbers: interpret your results in the broader context of brain function and cognitive neuroscience.
- Whether you are captivated by mathematical intricacies, fascinated by computational challenges, or inspired by neuroscience discoveries, let curiosity guide your exploration and we'll be here to guide you through it!

There are very few people who are going to look into the mirror and say, 'That person I see is a savage monster;' instead, they make up some construction that justifies what they do.

Noam Chomsky (the father of modern linguistics)