A Protocol for Isolating Neural Activity of Neurons and Analyzing Their Behavior in a Pattern Separation Task

Faraz Moradi Salavat

Thesis submitted in partial fulfillment of the requirements for the Master of Applied Science degree in Biomedical Engineering

School of Electrical Engineering and Computer Science

Faculty of Engineering

University of Ottawa

Abstract

Understanding how the human brain works can lead to new discoveries and improved treatments for brain related diseases and disabilities such as Alzheimer's and autism. One method for studying brain activity is through electrophysiological recordings, particularly through the use of in vivo recording techniques. While these techniques have advanced significantly over the years, data analysis tools have not kept pace, making it difficult to isolate the activity of individual neurons from the recordings. In this thesis, we propose a unified protocol for isolating the spike activity of a neuron from an electrophysiology recording. Additionally, we conducted customized spike train analysis on the recorded cells in a pattern separation task. Preliminary results suggest that changes in the neural activity of mossy cells was not significant. However, for granule cells and interneurons, responses to punishment and reward were observed.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisors, Dr. Leonard Maler and Prof. Dimitrios Makrakis for their unwavering support and encouragement throughout my academic journey, especially during the most challenging times. Their guidance and motivation have been instrumental in enabling me to complete my degree. I am also grateful to Dr. Jean-Claude Béïque for his valuable feedback and advice, which have enriched my knowledge and skills. I am extremely grateful for the financial aid provided by Dr. Leonard Maler and Dr. Jean-Claude Béïque, which enabled me to pursue my research interests with dedication and focus.

I would also like to acknowledge Nastaran Kosarmadar for her invaluable contribution to my research project and the amazing data she recorded, and to my fellow lab members, Érik Harvey-Girard and Kelly Xu, for their unwavering support and encouragement.

I would like to extend my gratitude to my family, who have supported me both emotionally and financially throughout my life in Canada. Without their support, I would not have been able to achieve my academic goals.

I am also thankful to my thesis committee for their time and effort in evaluating my work. Lastly, I am grateful to the University of Ottawa for the International Graduate Bursary I was awarded and the Teaching Assistantship positions assigned to me.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ACG	Auto-correlograms
CCG	Cross-correlograms
CA	Cornu Ammonis
CV	Coefficient of variations
DG	Dentate Gyrus
DS2	Type 2 Dentate Spike
EEG	Electroencephalogram
ECoG	Electrocorticography
EMG	Electromyogram
FFT	Fast Fourier Transform
fMRI	Functional Magnetic Resonance Imaging
FT	Fourier Transform
GCs	Granule Cells
GMM	Gaussian mixture models
ISI	Inter-spike Intervals
LFP	Local Field Potentials
MEG	Magnetoencephalography
MCs	Mossy Cells

NREM	Non-rapid Eye Movement Sleep
PCA	Principal Component Analysis
PSTH	Peristimulus Time Histogram
REM	Rapid Eye Movement Sleep
SC	Serial Correlation
SNR	Signal to Noise Ratio

Chapter 1- Introduction

1.1 Background

1.1.1 Hippocampus, Dentate Gyrus, and Pattern Separation

Hippocampus is a vital brain area that is involved in numerous cognitive functions, including spatial navigation, learning, and memory [1]. The hippocampus contains two brain regions: the dentate gyrus and the Cornu Ammonis (CA). The CA is anatomically and functionally differentiated into distinct subfields named CA1, CA2, CA3, and CA4 [2]. The dentate gyrus (DG) is part of the hippocampal formation in the temporal lobe of the brain, which also includes the hippocampus and the subiculum [3]. The DG is thought to contribute to the formation of new episodic memories, spontaneous exploration of novel environments, pattern separation, and other functions [4]. Pattern separation is defined as the ability to discriminate similar yet slightly different experiences. Theoretical and empirical studies have suggested that DG plays a role in this process [4]–[6]. The circuitry of the DG of the hippocampus is unique compared to other hippocampal subfields because there are two types glutamatergic principal cells instead of one: granule cells (GCs), and hilar mossy cells (MCs) [3], [7]. GCs make up the vast majority of the cells in the DG. MCs are the second excitatory cells after GCs in DG and receive excitatory inputs from GCs and back projections from different hippocampal regions [3].

1.1.2 Electrophysiology and Single-Cell Recording

Methods of studying the human brain have progressed rapidly in recent years. Currently, there are two modern methods of recording brain activity: in vitro (non-invasive) and in vivo (invasive). There are many recording techniques available for non-invasive recording, including electroencephalography (EEG), magnetoencephalography (MEG), and functional magnetic resonance imaging (fMRI). These methods are commonly used in hospitals for clinical diagnosis. The main problem with these methods is that they cannot record the activity of a single neuron. On the other hand, EEG devices record collective neuron activity from the surface of the skull, which produces a highly dampened signal. This signal is far from being able to discriminate single cell activity. Neurons are the basic functional units in the brain; they transmit information through the body using electrical signals called action potentials. Currently, invasive single-unit recordings provide the most precise recordings from a single neuron. A single unit is defined as a single, firing neuron whose spike potentials are distinctly isolated by a recording microelectrode.

Electrophysiology is the study of electrical activity within biological cells and tissues. In neuroscience, electrophysiological techniques are commonly used to measure and record the electrical activity of individual neurons or groups of neurons in the brain. Intracellular recording is a precise electrophysiological technique, used to measure electrical activities within individual neurons by inserting a finely-tipped electrode into the cell. This method provides insight into the membrane potential and intracellular currents, enabling a deeper understanding of action potential generation and synaptic transmission. Unlike extracellular recording, which captures electrical activity around a neuron, intracellular recording offers a detailed examination of the neuronal function at the cellular level [8].

Single-cell recording is a technique used in electrophysiology to record the activity of a single neuron [9]. By measuring the electrical impulses of a single neuron, researchers can gain insight into the functional properties of that neuron, as well as how it interacts with other neurons in the brain [10]. Electrophysiological studies have contributed greatly to our understanding of neural processing, including the mechanisms of learning and memory [1] and the neural basis of sensory perception [11]. In recent years, advances in electrophysiology have allowed for the recording of larger numbers of neurons simultaneously, leading to a better understanding of how neural networks function as a whole [12].

1.1.3 Spike Sorting

Spike sorting is an essential technique used in neuroscience to analyze the electrical signals produced by individual neurons, known as "spikes." These spikes are recorded using electrodes or other sensors placed near the neurons of interest, and the resulting signals are processed using a series of steps that are collectively known as spike sorting. Spike sorting is necessary because the electrical signals produced by neurons are often very small and can be difficult to distinguish from noise, and because multiple neurons can produce similar electrical signals, which can be difficult to separate [13].

The process of spike sorting typically involves several steps. First, the raw electrical signals are filtered to remove noise and amplify the signals produced by the neurons. Next, features are extracted from the signals, such as the amplitude, shape, or frequency of the spikes. These features are then used to group the spikes into clusters, with each cluster representing the activity of a single neuron. Finally, the quality of the clusters is assessed using various

metrics, such as the degree of separation between clusters or the consistency of spike waveforms, and any errors or ambiguities in the clustering process are corrected [13], [14].

Spike sorting is essential for accurately identifying the activity of individual neurons in complex neural circuits. It has become increasingly important in recent years as researchers have sought to record and analyze the activity of larger numbers of neurons simultaneously. Advances in microfabrication and signal processing technology have led to the development of new types of electrodes and recording systems that can simultaneously record the activity of hundreds or even thousands of neurons at once, making spike sorting an increasingly important and challenging task [12], [15].

Spike sorting is a critical step in the analysis of neural data that involves separating the electrical signals generated by individual neurons from the background noise [16]. Spike sorting toolboxes provide a set of algorithms and tools that enable researchers to perform this task efficiently and accurately. These toolboxes typically include features such as automatic or manual clustering, visualization of the results, and quality control metrics to assess the accuracy of the sorting. There are several spike sorting toolboxes available, including Kilosort, Tridesclous, and Spyking Circus [17], [18]. Each toolbox has its own strengths and weaknesses, and the choice of which one to use depends on the specific requirements of the experiment and the user's preferences[19]. Additionally, some toolboxes are optimized for specific types of data, such as extracellular recordings or calcium imaging.

1.1.4 Neural Signal Processing and Local Field Potentials

Neural signal processing is a rapidly evolving field that encompasses the study of how the nervous system generates, transmits, and processes electrical signals [20]. These signals, also known as action potentials or spikes, are the fundamental means by which neurons communicate with each other, allowing for the complex information processing that underlies brain function. However, in addition to action potentials (high-frequency signals), another type of electrical activity in the brain known as local field potentials (LFPs) or low frequency signals has gained significant attention in recent years.

LFPs are low-frequency electrical signals that arise from the collective activity of populations of neurons and are measured extracellularly using electrodes placed in or near brain tissue [21]. LFPs reflect the coordinated activity of multiple neurons in a particular brain region, providing a window into the functional dynamics of neural circuits. LFPs have been used to investigate a wide range of neural processes, including sensory processing, motor control, cognitive functions, and pathological conditions such as epilepsy.

The study of LFPs has provided valuable insights into the complex dynamics of neural activity and has advanced understandings of brain function. In addition to basic research, LFPs have also been widely used for treating movement disorders and epilepsy through deep brain stimulation [22]. Moreover, LFPs have been combined with other neuroimaging techniques, such as fMRI and EEG, to provide a more comprehensive understanding of neural activity and its relationship to behavior and cognition [23], [24].

LFPs have been widely used in the study of sleep to provide valuable insights into the dynamics of brain activity during different stages of sleep. For example, LFPs have been used to characterize sleep stages, with patterns of low-amplitude, fast-frequency oscillations observed during rapid eye movement (REM) sleep and slow-frequency oscillations with higher amplitudes observed during non-rapid eye movement (NREM) sleep [25]. LFPs have also been used to investigate sleep-related brain rhythms, such as delta waves, theta waves, and gamma waves, providing information about the timing, duration, and patterns of these rhythms during sleep [26]. Furthermore, LFPs have been used to explore sleep-related neural circuits, shedding light on the coordination and integration of neural activity across the brain during sleep [27]. LFPs have also been used to study sleep disorders, such as sleep apnea, insomnia, and narcolepsy, revealing abnormal patterns of brain activity that may contribute to these disorders [28].

1.1.5 Spike Train Analysis

Spike train analysis is a widely used method in neuroscience for studying the patterns and dynamics of neuronal activity. It involves the analysis of the timing and frequency of action potentials, or spikes, that are generated by individual neurons in response to various stimuli or during spontaneous activity [29], [30]. The analysis of spike trains provides valuable insights into the underlying mechanisms of neural processing, such as information coding, neural network dynamics, and plasticity, and has applications in diverse fields such as sensory processing, motor control, learning and memory, and clinical neuroscience [9], [31], [32].

One of the key challenges in spike train analysis is the inherent stochastic nature of spike generation, which can be influenced by a variety of factors, including intrinsic properties of the neuron, synaptic inputs, and network activity [33]–[35]. Therefore, advanced statistical and computational techniques are required to extract meaningful information from spike trains and reveal the underlying neural processes.

In recent years, there has been a significant advancement in spike train analysis methods, including approaches for spike detection, spike sorting, spike time interval analysis, and spike train correlation analysis, among others [31], [36]–[38]. These methods have been widely used to investigate various aspects of neural activity, such as neural coding, population dynamics, and functional connectivity, and have led to important discoveries and insights into the mechanisms of brain function.

1.2 Motivation and Problem Statement

Understanding the activity of a single neuron can help scientists gain insights into fundamental brain processes, develop neurotechnology, and diagnose and treat neurological disorders, benefiting both basic research and clinical applications. However, isolating the activity of a single neuron is a difficult process. The problem of spike sorting in neuroscience arises due to the fact that the electrical activity recorded from the brain can often contain overlapping signals from multiple neurons, making it difficult to accurately identify and separate individual spikes from different neurons. Spike sorting is crucial for understanding the functioning of neural circuits and deciphering the activity of individual neurons in the brain. There are several challenges associated with spike sorting, including: overlapping signals, variability in spike shapes, noise and artifacts, and large data volume.

To address these challenges, numerous spike sorting toolboxes have been developed by the neuroscience community. These toolboxes are software packages that provide a collection of algorithms and methods for performing spike sorting.

However, the abundance of various spike sorting toolboxes has made spike sorting a complex challenge for neuroscientists, including beginners. The issue is compounded by the fact that different scientists have developed spike sorters that accept different file formats and produce varying output data, with each sorter operating differently. Additionally, there is often a lack of clear instructions or documentation on how to use these sorters or integrate them with other toolboxes for further analysis. Furthermore, neuroscientists without a background in computer programming may find it challenging to customize the parameters of the sorters for different experimental designs. This complexity arises due to the varying requirements of different experimental designs, which demand specific adjustments to the sorter's parameters. As a result, the neuroscience community lacks a unified protocol to isolate spike activity of single neurons from an electrophysiology recording and ensure proper data formatting and parameters.

Following the design of a protocol for isolating neural activity of a neuron from electrophysiology recordings, further investigation was conducted to explore additional aspects of neuron behavior in a pattern separation task. Pattern separation refers to the ability to discriminate between similar yet slightly different experiences. The DG has been suggested to play a role in this process based on theoretical and empirical studies [39], [40]. However, the neural processing underlying pattern separation and the specific role of different DG cell types remain unclear. Of particular interest are the MCs, which are the

second excitatory cell type in the DG after GCs. Previous studies in electric fish conducted in our lab have shown that a specific telencephalic region, similar to the hippocampal hilus in rodents [41], exhibited increased activity in response to novel experiences [42]. Recent studies in mice have also suggested that MCs may be more sensitive to environmental changes compared to GCs [39]. To test this idea, after reaching a clean isolated unit, we tried to investigate how neural activity of MCs, GCs, and interneurons has changed in a designed pattern separation experiment performed by a Ph.D. student in our lab. Understanding the role of MCs in the learning and memory process, particularly as one of the most vulnerable cell types in the hippocampus to stress and aging, may contribute valuable insights to the research on neurodegenerative diseases and memory deficits such as Alzheimer's.

1.3 Research Objective and Contributions

In this dissertation, our primary objective was to design a comprehensive protocol for isolating neural activity of a neuron from electrophysiology recordings. A key focus of our work was to select a suitable spike sorter that could seamlessly integrate with other toolboxes such as Phy and Cell Explorer, both in terms of input and output data format. We also explored different approaches for converting and importing data from various recording boards and electrodes, and proposed customized solutions tailored to our experiment design. Furthermore, we extensively discussed various parameters of the toolboxes and elucidated how each toolbox can be interconnected with the output of the preceding one. In the final step, we refined and optimized the methods proposed in a recent paper to separate GCs and MCs from other units based on our experimental design. Overall,

the first part our work presents a comprehensive protocol that not only addresses the selection of appropriate spike sorters and data conversion techniques, but also provides enhancements to existing methods for isolating neural activity from electrophysiology recordings.

MCs and GCs are critical for memory and navigation as they are key components of the hippocampus, a brain region involved in learning and memory processes. MCs receive input from various brain areas and send their axons to the DG, where they synapse onto GCs. GCs, in turn, receive input from MCs and other sources, and they form the primary output of the DG. The neural coding of these cells, including their firing patterns and connectivity, has been linked to memory formation and spatial navigation. Understanding the intricate mechanisms of MCs and GCs' neural coding could provide insights into brain diseases such as Alzheimer's, epilepsy, and traumatic brain injury, which are associated with hippocampal dysfunction. In the concluding section of our study, we conducted a thorough spike train analysis of the behavioral tasks to elucidate the role of these cells in a pattern separation task. We employed a range of classic spike train analysis methods, and customized their coding and computations to align with our experimental design. These methods were employed using different data visualization libraries to give use better understanding of the neural activity of these cells in the designed experiment. In chapters 2 and 3, I have utilized and modified codes originally written by other contributors. In contrast, all codes presented in chapter 4 are my original work.

1.4 Thesis Organization

The rest of this document is organized as follows. Chapter 2 reviews behavioral task and different toolboxes to isolate a single neuron from an electrophysiology recording. Next, the method of separation of different neurons (MC and GC) was provided in Chapter 3. Chapter 4 shows various analyses that were performed on isolated neurons in a behavioral task. Finally, the last chapter concludes the work and suggests avenues for related work in the future.

Chapter 2 - Spike Sorting

The purpose of this chapter is to discuss the spike-sorting procedure chosen for data analysis. First, the behavioural task and recording setup is described. Then, different spike sorting toolboxes are compared to select the most suitable one for spike sorting purposes. A thorough explanation is provided for the manual curation process of the extracted units using Phy. In the final step, Cell Explorer was used to separate excitatory and inhibitory cells.

2.1 Recording and Behavior Setup

The extracellular recording is a method of electrophysiology recording that involves placing a microelectrode near a group of cells to measure the combined electrical activity of those cells. The microelectrode is typically inserted into the brain or peripheral nervous system to detect voltage changes that occur at the surface of cells when they are active. Extracellular recording is commonly used in research to study brain function and to diagnose and treat neurological disorders. It can provide important information about the activity of specific brain regions, the connections between brain regions, and the ways in which different brain regions interact to produce behavior.

2.1.1 Behavioral Task

For our experiment, we choose to use the Bussy Saksida touchscreen system [43], which is a computer-automated task made of a triangular chamber containing a touch-screen and a reward section (*Figure 2.1A*). Images appear on the screen and the mice are trained to touch the right or left side of the screen with their nose (poke the screen) based on the

presence of a vertical line. The role of the animal is to poke the left screen when it detects a vertical line, and to poke the right screen when there is no vertical line present. The visual stimuli consist of a reference image (36 horizontal bars, arranged 6x6), and different variations of test images (identical to the reference image, but with a certain percent of horizontal bars transformed into vertical bars among the 36) (*Figure 2.1B*).

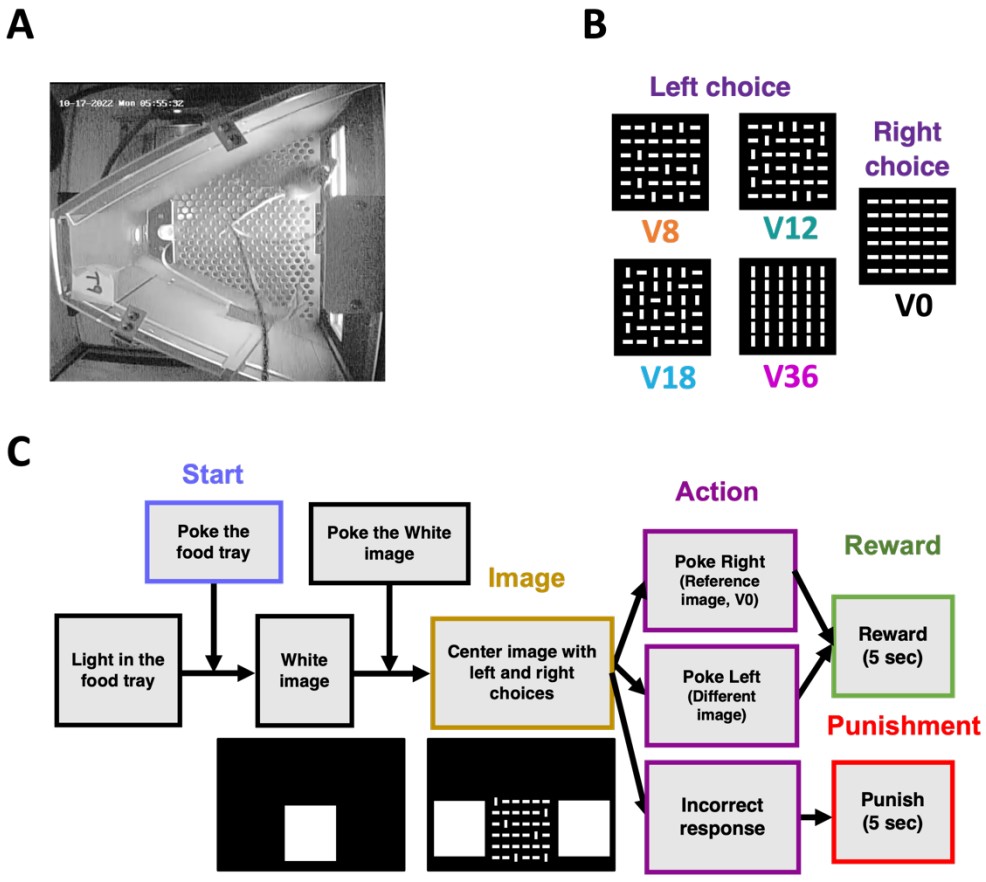

Figure 2.1 – Behavioral task overview. A) The behavioral task chamber. B) Five different images that the animals observe associated with right or left choice. C) The process of each trial performed by the animal from left to right.

For each trial, one image appears on the screen and the mice must choose right or left based on whether the image is the reference or test image (*Figure 2.1C*). Mice first go through food deprivation, habituation and pretraining where they learn the basic rules of the task, such as touching the screen, its association to reward, and initiating the trials. Then, they

start the training and test sessions where mice see either the reference image or a test image and they need to discriminate between them. The 'performance on the task' is calculated for each day by dividing the number of correct trials to the total number of trials. All animals were housed in the University of Ottawa Animal Care and Veterinary Services facility and all animal procedures were conducted with the approval of the University of Ottawa's Animal Care Committee and in accordance with guidelines set out by the Canadian Council of Animal Care. The recording used for analysis was obtained from a 3-month-old female DRD2-Cre mouse. Mice were individually housed in reverse 12h dark/12h light cycles. Animals had had water available ad libitum, but were food-restricted without weighting lower than 85% of their initial weight. The temperature of the room was kept around 22 Celsius and the humidity was 40%. Testing of each cohort took place over the course of approximately 3 months, for about one hour six days a week.

2.1.2 Recording Setup

After the animal is well-trained, single unit recording will be conducted using Cambridge Neurotech probes (P-series), which consist of two shanks with 32 channels. The geometrical properties of each electrode on the shank are represented in *Figure 2.2.A.* After the surgery and electrode implementation, it was observed that the recording quality of the left shank was not satisfactory, as no spikes were captured. As a result, the analysis was performed using the data recorded from the right shank. The in vivo recording was performed using Intan recording boards (*Figure 2.2.B*) [44] with sampling frequency

of 30kHz. Moreover, the recording board receives the TTL pulses generated while the animal is performing different tasks. Every recording cycle has three main phases:

1. Recording in the home cage, before the behavioral task.
2. Recording during the behavioral task.
3. Recording in the home cage, after the behavioral task.

The reason for having these three phases is that spike sorting (isolating the neural activity of a single neuron) is a machine-learning problem and at least a total 6 to7 hours of recordings are essential for isolating good-quality units. Mentioned total recording duration consists of multiple recordings, which are combined. The duration of the behavioral task for the animal would be a maximum of one hour, therefore recording before and after is essential for spike sorting. Moreover, for classifying MCs and GCs, which are discussed in later sections, the firing rates collected in the home cage before and after the behavioral task are important.

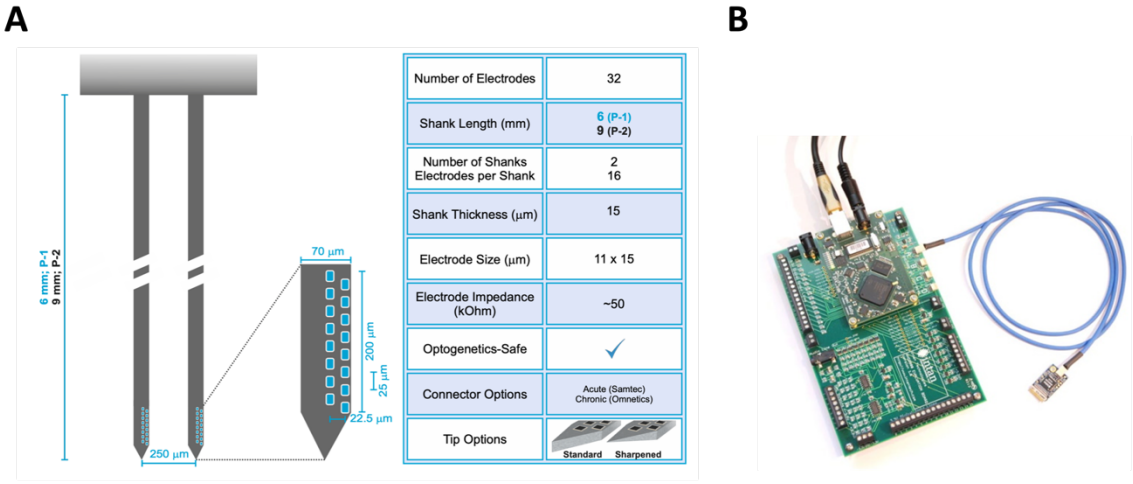

Figure 2.2 – Electrophysiology recording equipment. A) Cambridge Neurotech probe used for the experiment. B) Intan recording board used for recording electrophysiology activity.

2.2 SpikeInterface Framework

2.2.1 Background

SpikeInterface[1] [19] is an open-source software library that provides a set of tools for analyzing electrophysiology data. It is designed to be a platform-agnostic interface that can be used with a variety of different electrophysiology data formats, including data from microelectrode recordings, extracellular recordings, and high-density electrocorticography (ECoG) recordings. SpikeInterface includes a number of different modules for tasks such as spike sorting, feature extraction, and cluster quality assessment. It also includes tools for working with datasets that include multiple recording sessions, such as those collected in large-scale electrophysiology experiments. One of the main benefits of SpikeInterface is the availability of different sorters to compare the results of spike sorting. Moreover, various pre- and post-processing functions are included, which are necessary for electrophysiology analysis.

2.2.2 Available Sorters in the SpikeInterface

The available spike sorters in SpikeInterface include:

1. KiloSort: An open-source spike sorter that uses a combination of clustering algorithms and template matching to identify and classify individual spikes.
2. KiloSort2 (KS): An improved version of KiloSort that includes additional features such as automatic channel mapping and support for parallel processing.

[1] https://github.com/SpikeInterface/spikeinterface

3. Spyking-Circus (SC): An open-source spike sorter that uses a template-matching algorithm to identify individual spikes and a clustering algorithm to classify them into different neurons.
4. IronClust (IC): A spike sorter that uses a combination of clustering algorithms and machine learning techniques to identify and classify individual spikes.
5. HerdingSpikes (HS): A spike sorter that uses a template-matching algorithm to identify individual spikes and a clustering algorithm to classify them into different neurons.
6. Tridesclous (TDC): An open-source spike sorter that uses clustering algorithms and machine learning techniques to identify and classify individual spikes.
7. WaveClus: An open-source spike sorter that uses a combination of clustering algorithms and template matching to identify and classify individual spikes.
8. MountainSort: An open-source spike sorter that uses machine learning techniques to identify and classify individual spikes.

SpikeInterface is designed to be a platform-agnostic interface, which means that it can be used with a variety of different electrophysiology data formats and sorters. Users can choose the spike sorter that best fits their needs and data and can easily switch between different sorters if desired.

2.2.3 Comparison of Kilosort with other Spike Sorters

Within this section, a comparison will be made between the meta-data analysis of six specific spike sorters on both in-vivo and simulated datasets. The comparison will involve measuring the level of agreement between the sorters, assessing the performance of each

sorter based on the ground truth[2], and exploring the potential for combining the outputs of multiple spike sorters to enhance overall performance and alleviate the need for manual curation.

The dataset used for the first part of the analysis, which focused on in-vivo data, was obtained from Neuropixel recordings of a head-fixed mouse. The recordings were acquired from the Allen Institute for Brain Science, a renowned institution known for its high-quality neuroscience research[45], [46]. The recording has 246 active recording channels with a sampling frequency of 30 kHz covering part of the cortex (V1), the hippocampus (CA1), DG, and the thalamus (LP). The duration of the experiment was trimmed to 15 minutes. You can see a snippet of the recorded data in *Figure 2.3A*. The spike sorting was done using fixed default parameters for straightforward comparison. In *Figure 2.1B*. you can see the number of units found by each sorter. It is easy to see that there is no clear consensus among the sorters on the number of neurons in the recording. SC found the most units (628) and TDC the least (187). In *Figure 2.3C*., the total number of units for which *k* sorters agree is provided. Of all the extracted units, it was only 33 units that all sorters agreed on, which is surprisingly low. Different spike sorters, implement different algorithms and techniques for identifying and isolating spikes from raw data. These algorithms can vary in their assumptions, methodologies, and parameter settings, which can result to different algorithms producing different results despite processing the same set of data. We provide below some reasons that contribute to different sorters generating different spike sorting results while processing the same set of data (*Figure 2.3C-D*):

[2] A ground truth dataset is a reference or benchmark dataset that is considered to be the authoritative representation of the data, used for evaluating the accuracy and performance of machine learning models or algorithms.

1. Algorithmic Differences: Spike sorting algorithms can use different approaches to extract features from the recorded data, such as thresholding, template matching, or principal component analysis (PCA). They can also use different clustering methods, such as k-means, hierarchical clustering, or Gaussian mixture models (GMMs), to group spikes into clusters. These algorithmic differences can result in variations in the identification and separation of spikes, leading to different sorting outcomes.
2. Sensitivity to Parameters: Spike sorters often require setting various parameters, such as spike amplitude thresholds, feature extraction parameters, or clustering criteria. These parameters can significantly influence the performance of the sorter and impact the quality of the sorted spikes. Different sorters may have different default parameter settings or may require manual tuning, and variations in these parameter settings can affect the sorting results.

3. Data Variability: Electrophysiology recordings can be noisy and complex, with variability in spike shapes, firing rates, and background noise levels. Different spike

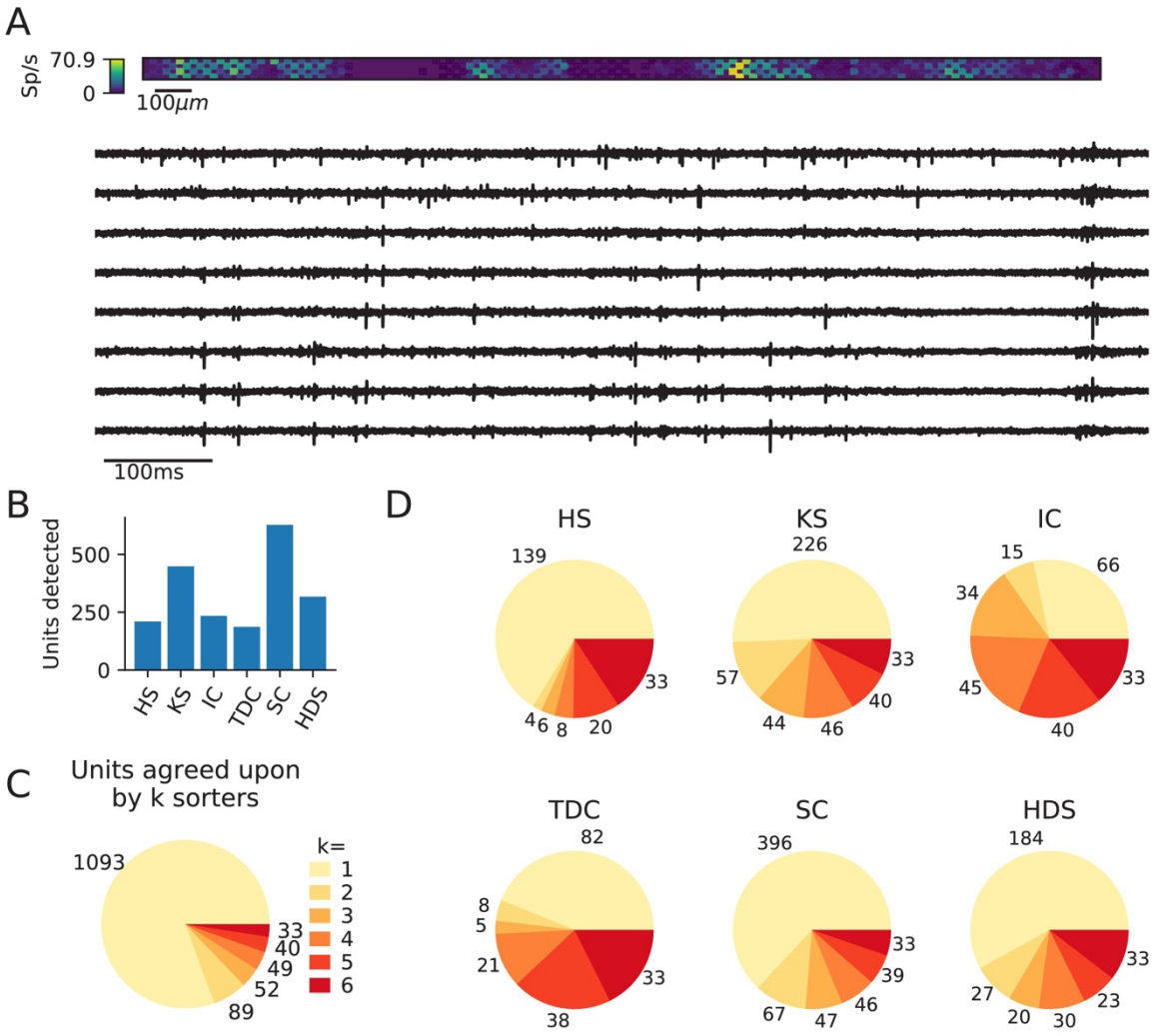

Figure 2.3 - Comparison of spike sorters on a real Neuropixels dataset. [19]

sorters may have different sensitivities to these variations in the data, which can affect their ability to accurately identify and separate spikes. Some sorters may perform better on certain types of data or recordings with specific characteristics, while others may struggle with different types of data, leading to variability in the sorting results.

4. Evaluation Metrics: Spike sorting results are often evaluated using metrics such as cluster quality, signal-to-noise ratio, or isolation distance. Different sorters may use different evaluation metrics or have different criteria for defining what constitutes a "good" or "bad" cluster. This can affect the interpretation of sorting results and lead to differences in the reported outcomes.

This low agreement raises the following question: which of the mentioned spike sorters are the best or more appropriate ones to use? In further analysis, Buccino et al. simulate a 10 min Neuropixels recording using the MEArec Python package[47]. The recording comprises the spiking activity of 250 biophysically-detailed neurons, of which 200 are excitatory and 50 are inhibitory cells selected from the Neocortical Micro Circuit Portal [48], [49] that exhibit independent Poisson firing patterns. A visualization of the simulated activity map and extracellular traces from the Neuropixels probe is shown in *Figure 2.4A*. Buccino et al. did the spike sorting using the same sorters and parameters on the ground truth dataset. Again, there is a large discrepancy in the number of detected units (*Figure 2.4C*). In *Figure 2.4D*, different metrics are represented for each spike sorter. Some spike sorters prioritize precision over recall, while others prioritize recall over precision. Additionally, the accuracy of most spike sorters is influenced by the signal-to-noise ratio (SNR) of the ground-truth units, although Kilosort2 achieves nearly perfect performance on low-SNR units. While the majority of spike sorters have a wide range of scores for each metric, Kilosort2 consistently achieves higher scores than the other spike sorters for most ground-truth units. *Figure 2.4E*. shows the breakdown of detected units for each spike sorter. Each unit is classified as well-detected, false positive, redundant, and/or over merged by SpikeInterface. This plot, interestingly, may shed some light on the remarkable

accuracy of Kilosort2. While Kilosort2 has the most well-detected units (245), this comes at the cost of a high percentage of false positive (147) and redundant (21) units.

Notably, Tridesclous detects very few false positive/redundant units while still finding many well-detected units. HDSort, on the flip side, finds more false positive units than any other spike sorter. For a comprehensive comparison of spike sorter performance on both real and simulated datasets, we refer the reader to the related SpikeForest project[50], [51].

The main reasons Kilosort achieves better results compared to other spike sorters might be the following:

1. Template matching: Kilosort uses template matching; a powerful technique that compares the shape of recorded spikes with pre-defined templates to identify and separate spikes generated by different neurons. Template matching can provide high accuracy in spike sorting, particularly for neurons with distinct spike shapes, making Kilosort effective in separating spikes from different units. Moreover, a novel post-clustering merging step was proposed that is based on the continuity of the templates substantially reduces the requirement for subsequent manual curation operations.
2. Parallel processing: Kilosort is optimized for parallel processing, allowing for fast and efficient sorting of large datasets. The software can take advantage of multi-core processors to speed up the sorting process, making it suitable for handling recordings containing large number of channels or produced by high-density electrode arrays.

3. Open-source and community-supported: Kilosort is an open-source software that is freely available to the research community. It has a large user base and is continuously updated and improved by the community, making it a well-supported and reliable option for spike sorting.

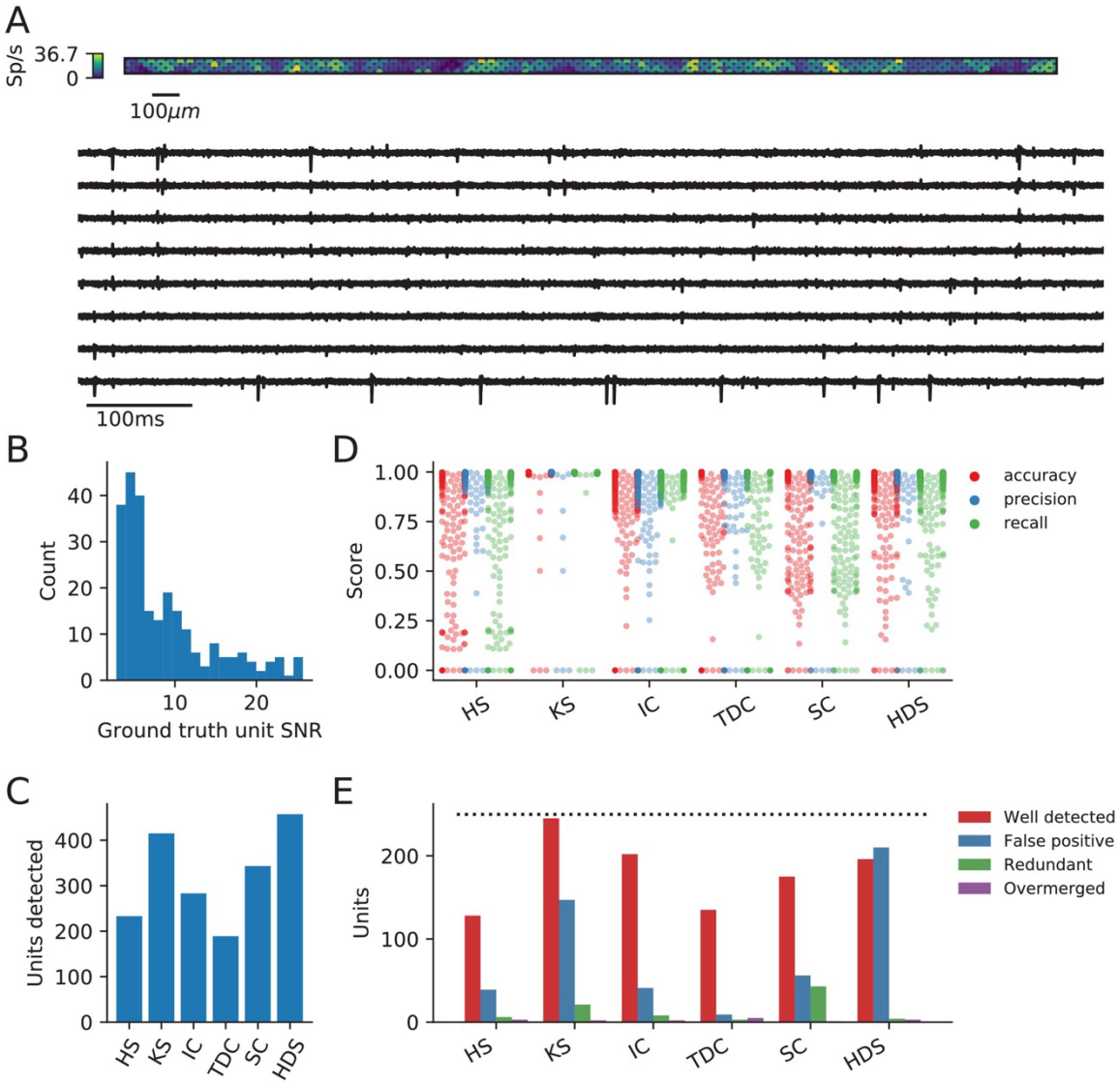

Figure 2.4 - Evaluation of spike sorters on a simulated Neuropixels dataset. [19]

2.2.4 Kilosort GUI and Data Formats

In the previous section, we discussed the SpikeInterface framework, which allows for the comparison of results from different spike sorters. Based on the SpikeInterface paper and the Spike Forest [52] project, we conluded that Kilosort [3][14] outperformed other sorters when working with simulated datasets. Additionally, we conducted a thorough investigation of top journal papers [53]–[55] in the field and reached out to famous neuroscience labs [56], [57] to gather insights on the best spike sorter to use at this time. Based on these efforts, we have decided to select Kilosort as our preferred spike sorter. Several factors contributed to our decision. Firstly, Kilosort has gained significant recognition and popularity, as evidenced by its high number of stars on GitHub. This indicates a strong user base and active developer community, which is crucial for troubleshooting and debugging support. Furthermore, the probe company we are using, Cambridge NeuroTech, highly recommends Kilosort for optimal performance with their probes. Their endorsement adds to the credibility and suitability of Kilosort for our specific experimental setup.

Installing Kilosort, a MATLAB-based toolbox, has many challenges. The first essential requirement for installing Kilosort is that the desktop computer should be capable of supporting Cuda[4]. This is available only for NVIDIA graphic cards and AMD Radeon graphic cards are not supported. Usually, it is recommended that neuroscience labs dedicate a specific PC for spike sorting purposes.

[3] https://github.com/MouseLand/Kilosort

[4] https://developer.nvidia.com/cuda-toolkit

After installing Kilosort using all the notes provided in the GitHub repository, the GUI can be opened (*Figure 2.5*) by typing *Kilosort* in the MATLAB command window. Different parameters can be imported to the GUI and they are discussed below:

- **Number of Channels:** The number of channels in the recording electrode.
- **Sampling Frequency (Hz):** Sampling rate of the recorded data.
- **Threshold [Th(1) Th(2)]:** During the optimization (process, spike detection thresholds Th(1)) or Th(2)) are set high enough to only capture sortable units, while during the final pass (, thresholds Th(2)). These) are set low enough to pick up all spikes of these units. It's worth noting that these thresholds are applied to the template projections; not to rather than the voltage. The system can handle the presence of noise, as an additional threshold is set per neuron and a splitting step is carried out to ensure clusters with multiple units are separated.
- **Lambda:** This parameter is used to control the trade-off between spatial and temporal regularization during spike sorting. It determines the strength of the spatial penalty applied to the template amplitudes, which influences the degree of spatial smoothness in the extracted spikes. A value of 50 corresponds to a strong bias, while a value of 0 indicates no bias. The optimal value for lambda depends on the specific dataset and the characteristics of the neural activity being recorded. It often requires some experimentation and tuning to find the appropriate balance between spatial and temporal regularization for accurate spike sorting results.

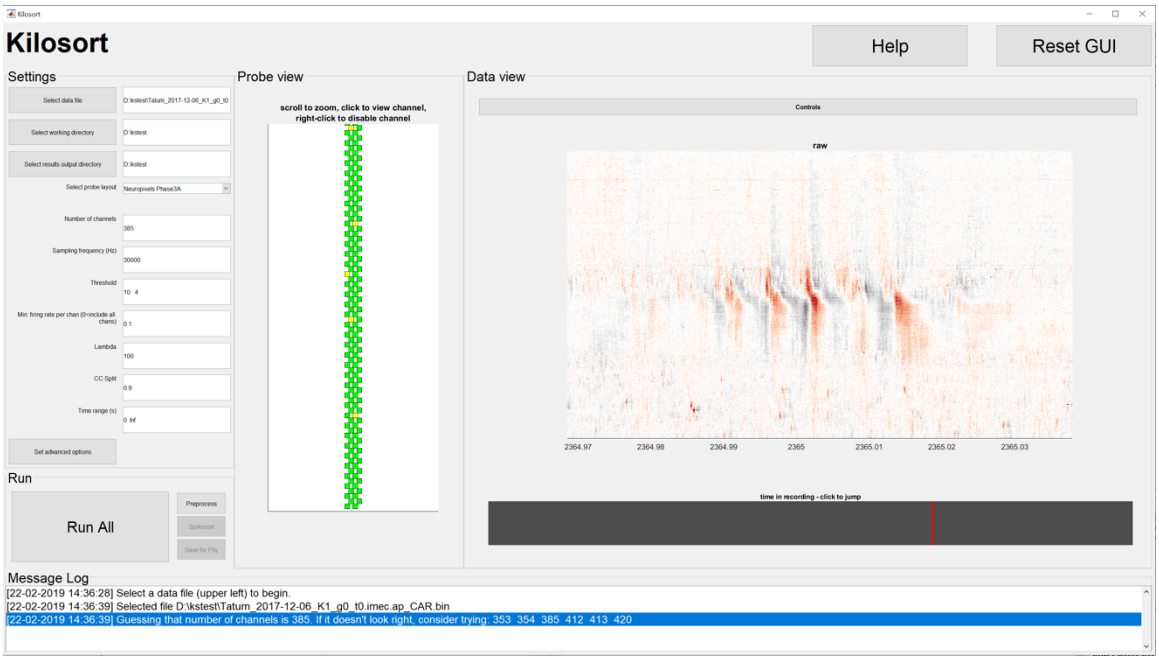

Figure 2.5 - Kilosort GUI.

2.2.4.1 Loading a Dataset

To get started, choose a dataset by clicking "select data file" located at the upper left of the GUI. The data file should be a raw binary file containing just the data matrix, size [nChannels x nTimepoints], with int16 datatype. The nChannels and nTimepoints represent the number of channels and number of timepoints (sampling rate * duration in second). If desired data can be loaded into a Matlab matrix size [nChannels x nTimepoints] and can be converted to int16 [5] datatype, then it can be written to an appropriate file using the code provided in the Appendix. After the file was selected, Kilosort will load it using the default channel map and options. In the next step, the Number of Channels and the Channel Map

[5] This term is used in computer programming and represents a signed integer that occupies 16 bits of memory

must be inputted. Channel Map refers to the configuration or assignment of individual channels or electrodes to specific recording sites or locations in a neural recording system. It defines the spatial arrangement and connectivity of the channels, indicating which channels correspond to the physical positions on an electrode array or probe (*Figure 2.2A*). If the channel map for desired probe is not in the drop-down list, two options are left. There are two ways to create a channel map. First, a channel by opening */configFiles/createChannelMapFile.m* required information code be inputted to the code and the channel map will be created. Second, entering the information directly into the GUI by choosing 'new' in the select probe layout (*Figure 2.5*) to a pop-up window which can be filled out. An example of a valid entry is depicted in *Figure 2.6*.

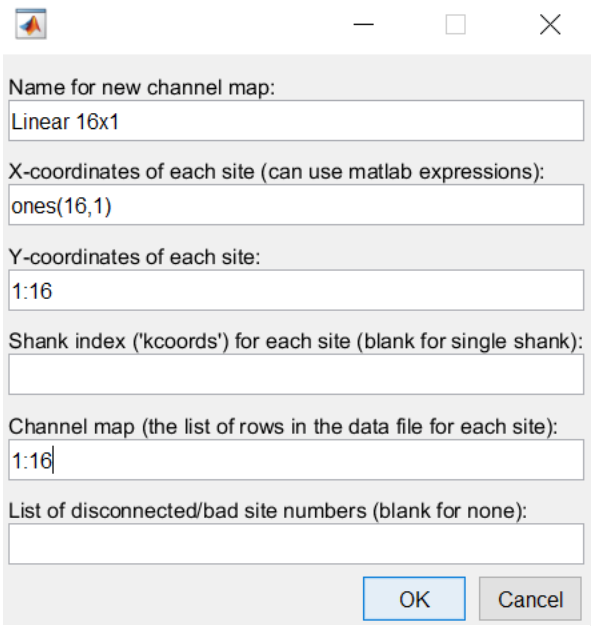

Figure 2.6 - Inputting probe's channel information to Kilosort GUI.

2.2.4.2 Viewing data

The Probe View (Figure 2.5 middle) shows the layout of the probe as specified in the channel map. Green/yellow channels are ones selected currently for viewing in the Data View. Blue/red channels are not currently visible in Data View. Yellow/Red channels are excluded from analysis and do not appear in Data View. In colormap-mode, the Data View has an image showing channels (y-axis) by time (x-axis) voltage values. In this mode, only one of the available datasets (raw, filtered, prediction, residual) can be seen at a time. Channels not selected for analysis (yellow on Probe View) are not displayed. In traces-mode (*Figure 2.7*), the Data View shows a set of traces for the selected channels, spaced out along the y-axis. In this mode, the different available datasets can be overlaid. Channels not selected for analysis (yellow on Probe View) are shown in light grey.

The datasets are:

- **Raw data:** the data straight from your file
- **Filtered:** after preprocessing (whitening, common average referencing, temporal filtering) as specified in the options.
- **Prediction:** based on Kilosort's processing of data, what does the tool think the 'true' data look like. Ideally, it should be a de-noised version of the filtered data.
- **Residual:** the difference between Filtered and Prediction.

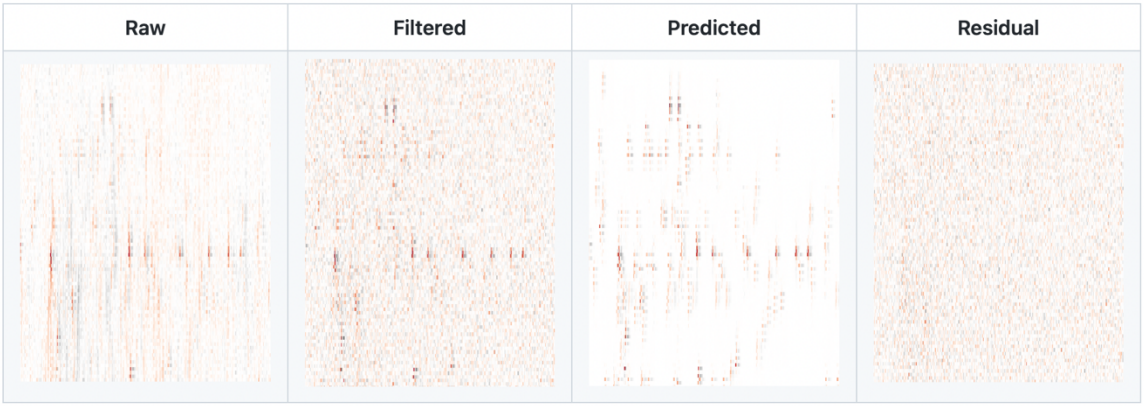

Figure 2.7 - Raw, Filtered, Predicted and Residual data depicted in the Kilosort GUI.

2.2.5 Spike Sorting using Kilosort in the SpikeInterface Framework

One of the main problems with the Kilosort GUI is the lack of pre- and post-processing options. Kilosort GUI is not able to concatenate recorded data from different states, which is usually the case for spike sorting. However, if quick spike sorting was essential on a sample recording, then it is a reasonable option. To have access to more powerful calculation options, we performed all the processing through the SpikeInterface framework.

SpikeInterface can load various file formats from recording boards. In our experiment, the data types were saved as a *.dat* file using an Intan recording board which is a standard format. The *BinaryRecordingExtractor* from the SpikeInterface Extractor module was used to load the data. This function needs 3 main inputs: recording path, sampling rate (30KHz), and number of channels (32). As mentioned before, one of the primary advantages of SpikeInterface is to perform spike sorting on concatenated datasets. This was performed on three main states mentioned in section 2.1.2. After loading the data, different

information about the recording object can be inspected to make sure everything was inputted correctly.

The geometric properties of the electrodes (channel map) used for recording are one of the most critical inputs for any spike sorter. Crucially, sorters can match detected spikes with extracted unit templates based on electrode location. This information can be passed to recordings in the SpikeInterface by using the ProbeInterface [6][58] framework. ProbeInterface includes most of the recording electrodes, such as Cambrdige Neurotech [59] and NeuroNexus [60], which are easily found by searching the electrode model. It is also possible to design custom channel maps based on the stacked electrodes, which is very likely to happen when recording from regions like the hilus.

In the next step, different preprocessing techniques, such as temporal or spatial filtering could be applied, which were not essential for this experiment. Spike waveforms typically have a fast-rising phase and a relatively brief duration, making them rich in high-frequency components. High-pass filtering is employed in spike detection to enhance the visibility and isolation of fast, transient events, such as action potentials or spikes, from neural recordings. By selectively allowing high-frequency components to pass through while attenuating low-frequency signals, high-pass filtering helps reveal the rapid voltage changes associated with spikes. Although it may seem logical to use a band-pass filter to isolate the frequency range of the spike signals, it can also remove some of the high-frequency components that are important for accurately characterizing the spikes.

[6] https://github.com/SpikeInterface/probeinterface

Therefore, it is safer to use a high-pass filter as all the information about the shape of a spike is important for us.

For performing spike sorting, different toolboxes can be selected by using the sorter module. Various parameters of Kilosort as discussed previously can be adjusted. In the final step, using the *export_to_phy* sub-module, all the features needed for Phy are saved in a folder for manual curation. Various errors can be received at this step which is recommended to look at the issues on the SpikeInterface GitHub repository.

2.3 Manual Curation of Extracted Units Using Phy

Phy[7][61] is an open-source software tool for the manual curation of extracted units from electrophysiology data. It is commonly used in combination with automated spike sorting algorithms, which are used to identify and classify individual spikes (or action potentials) in an electrophysiology recording. After automated spike sorting has been performed, Phy can be used to manually review and curate the results. This involves looking at the waveforms and other features of the extracted units (waveform, peak amplitude, number of spikes, auto correlogram) and determining whether they are accurately classified and belong to the same neuron. Phy provides a number of tools to assist with manual curation, including:

1. A graphical user interface (GUI) that allows users to view and interact with the extracted units.

[7] https://github.com/cortex-lab/phy

2. Tools for visualizing and comparing the waveforms and other features of the extracted units.
3. A suite of data analysis and visualization tools, including tools for clustering, dimensionality reduction, and spike sorting quality assessment.
4. A plugin system that allows users to extend the functionality of Phy with custom scripts or plugins.

Manual curation using Phy is an important step in the spike sorting process, as it helps to ensure the accuracy and reliability of the extracted units. It can be time-consuming, but it is a necessary step to ensure that the results of the spike sorting are of high quality and can be used to confidently answer research questions.

2.3.1 Phy GUI Interface

The Phy graphical user interface (GUI) is a tool for manually curating extracted units from electrophysiology data. It is designed to be user-friendly and intuitive, and it provides a

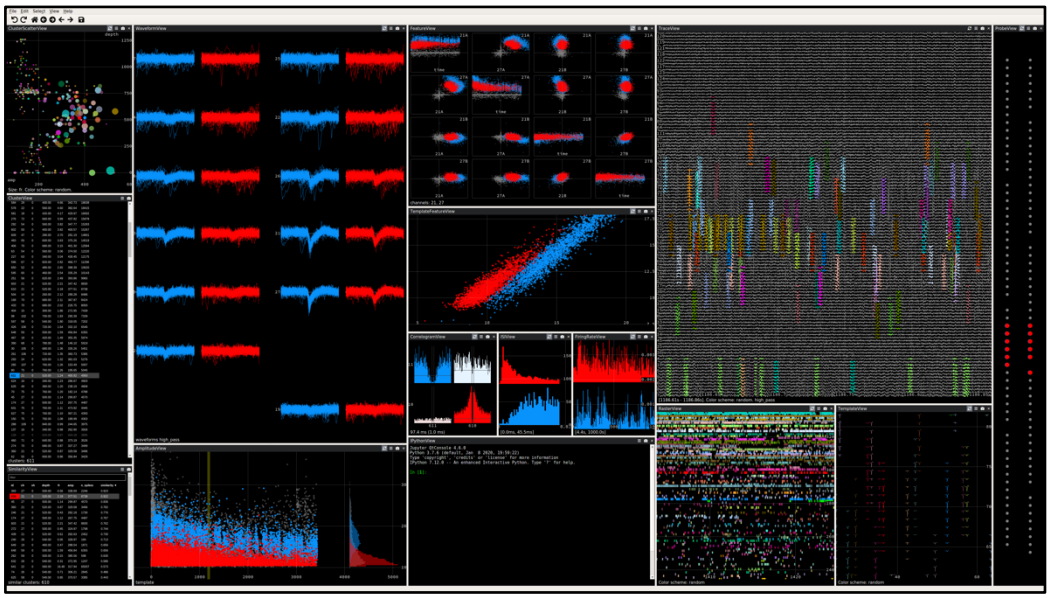

Figure 2.8 - Phy GUI interface.

number of features to assist with the manual curation process. Using the Phy GUI, users can view and interact with the extracted units, compare the waveforms and other features of different units, and use the various tools and features provided by Phy to manually curate the extracted units. The GUI is designed to be flexible and customizable, allowing users to tailor the interface to their specific needs and preferences. To open the GUI for your dataset using the Phy toolbox, navigate to the directory containing the dataset and the "params.py" file (SpikeInterface output), then execute the command "phy template-gui params.py" in the command-line or terminal. The GUI consists of several parts, and in each part, there is a different criterion and standard to consider for removing, merging, and separating clusters, which are discussed in the following sections.

Cluster view:

The Cluster view presents a comprehensive list of all the clusters contained within it. By clicking on one cluster, it can be selected, and multiple clusters can be selected by

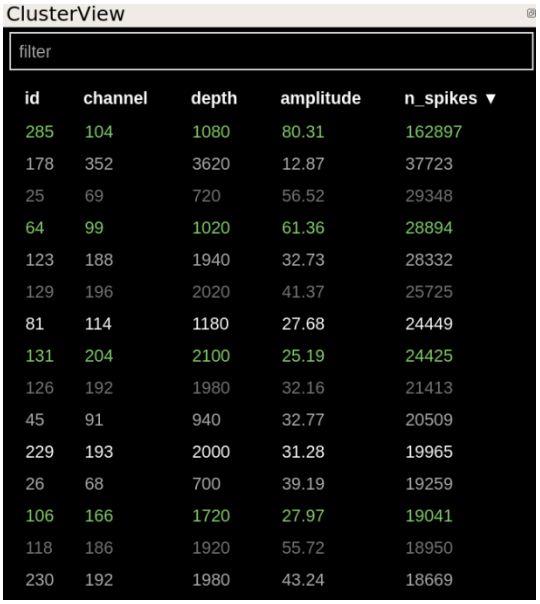

Figure 2.9 - Phy Cluster view. Most of the good quality units are labeled by green automatically after running the GUI.

holding down Control or Shift. The selected clusters will then be displayed across various graphical views (*Figure 2.9*), where clustering actions such as merge, split, move, label, and others can be performed on the selected clusters.

Cluster scatter view:

The cluster scatter view presents all clusters in a scatter plot (*Figure 2.10*). The cluster view in the GUI calculates the positions of each point on the x-axis, y-axis, marker size, and color based on four customizable fields chosen from the available columns. By default,

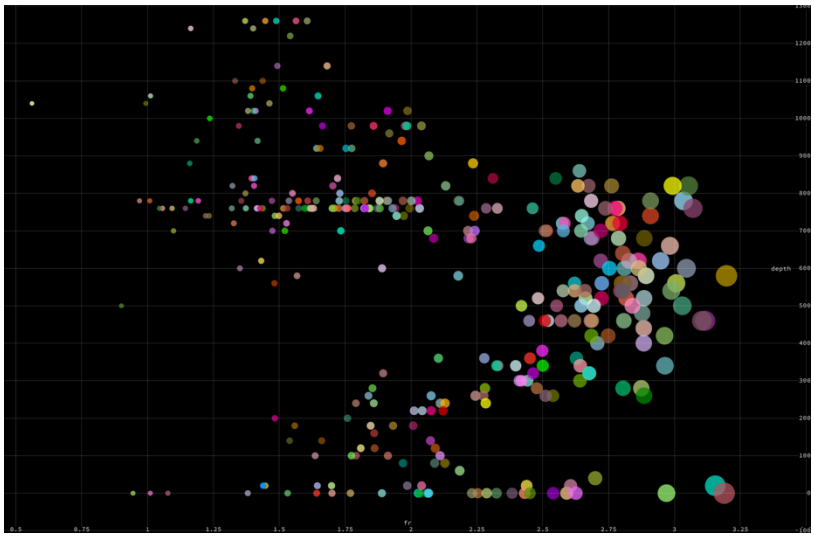

Figure 2.10 - Phy cluster scatter view. Each circle indicates a unit with waveform amplitude, depth and firing rate representing x axis, y axis and marker size.

these fields are assigned as follows: waveform amplitude for the x-axis, depth for the y-axis, and firing rate (log scale) for the marker size. To select a cluster, simply click on it, and to add a cluster to the selection, use shift+click. The color scheme mapping can be altered using shift+wheel. Alternatively, multiple clusters can be selected by drawing a lasso with ctrl+click.

Waveform view:

The waveform view displays the waveforms of a selected number of spikes on the relevant channels based on their proximity and amplitude to the peak waveform amplitude channel (*Figure 2.11*). By default, the parameter controller.n_spikes_waveforms is set to 100, which determines the maximum number of spikes per cluster to be visualized in the waveform view. The parameter controller.batch_size_waveforms is set to 10 by default, indicating the number of batches used to extract the waveforms. Each batch corresponds to a set of successive spikes, with different batch positions uniformly distributed across the entire recording. Control+click can be used to select a channel, which will affect the feature view.

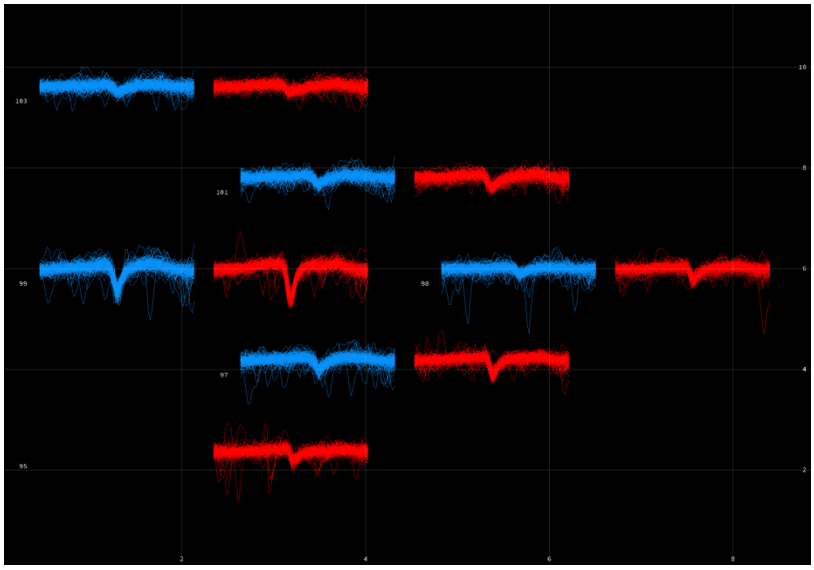

Figure 2.11 - Phy waveform view of the selected units.

Feature view:

The feature view displays the principal component features of a selected number of spikes in the chosen clusters on the relevant channels (*Figure 2.12*). Control+click can be used in

the waveform view to modify the exact channels. The principal components are labeled as A, B, C, etc. The background spikes from all clusters are depicted in grey.

The default value for the parameter controller.n_spikes_features in the feature view is 2500, which determines the maximum number of spikes per cluster that will be shown. Similarly, the parameter controller.n_spikes_features_background is set to 1000 by default, indicating the maximum number of background spikes to be displayed. These background spikes are evenly distributed in time throughout the entire recording and across all clusters.

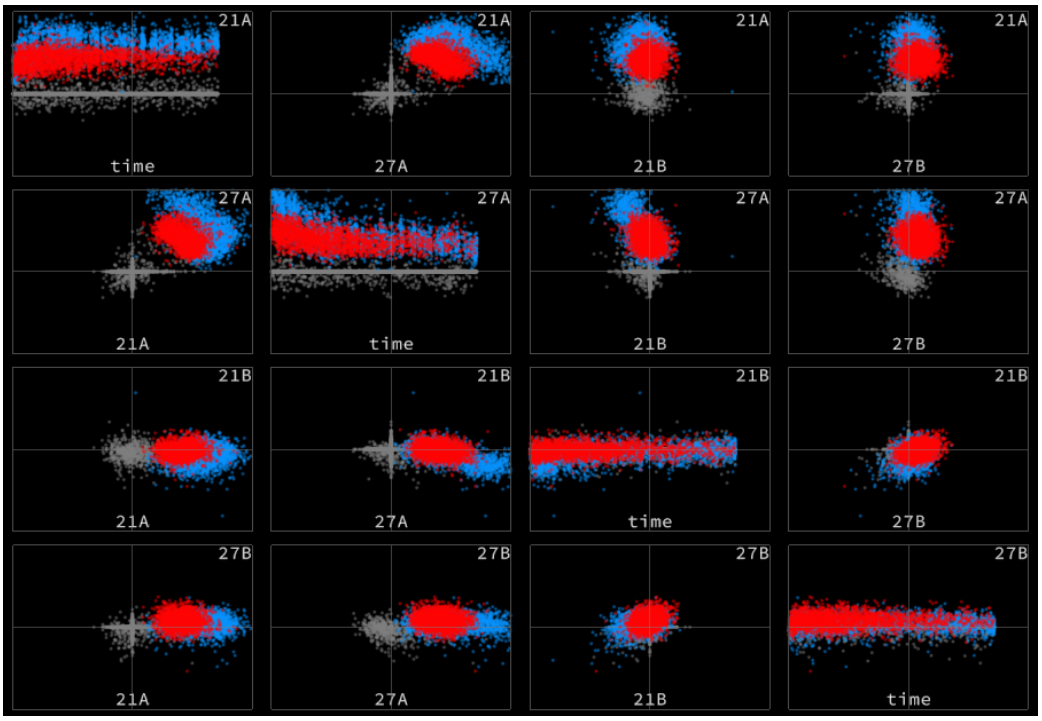

Figure 2.12 - The principal component view of spikes in a unit.

Correlogram view

The correlogram view presents the auto correlograms (ACG) and cross-correlograms (CCG) between all pairs of selected clusters (*Figure 2.13*). Each subplot in row i, column j, displays the cross-correlogram of selected cluster i versus cluster j. The horizontal line

indicates the baseline firing rate, and the vertical lines represent the refractory period, which is set to 2 ms by default. The parameter controller.n_spikes_correlograms is set to 100,000 by default, specifying the maximum number of spikes across all selected clusters to be utilized in computing the cross-correlograms. These spikes are selected randomly.

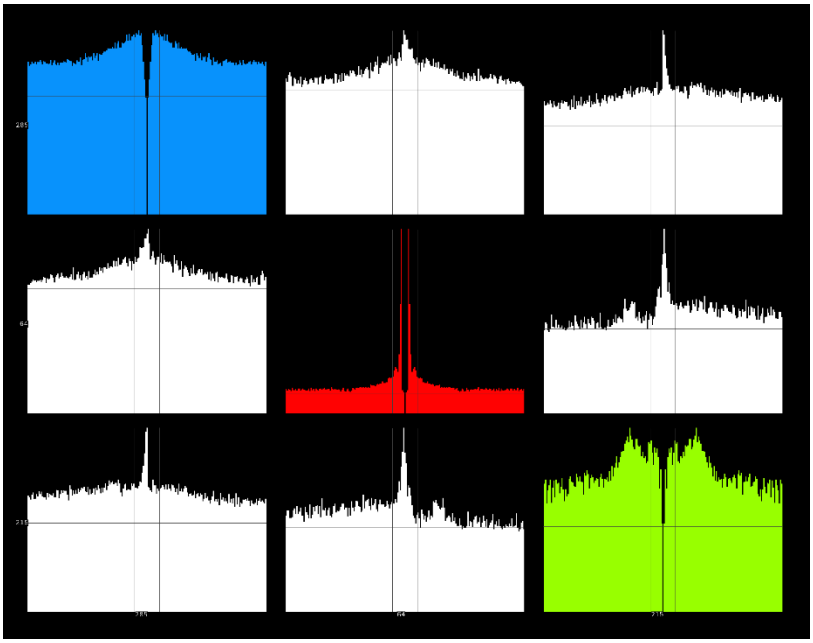

Figure 2.13 - Auto and Cross Correlogram view of selected units.

Amplitude view

The amplitude view displays the amplitude of a chosen set of spikes that belong to the selected clusters, along with vertical histograms on the right-hand side (*Figure 2.14*).

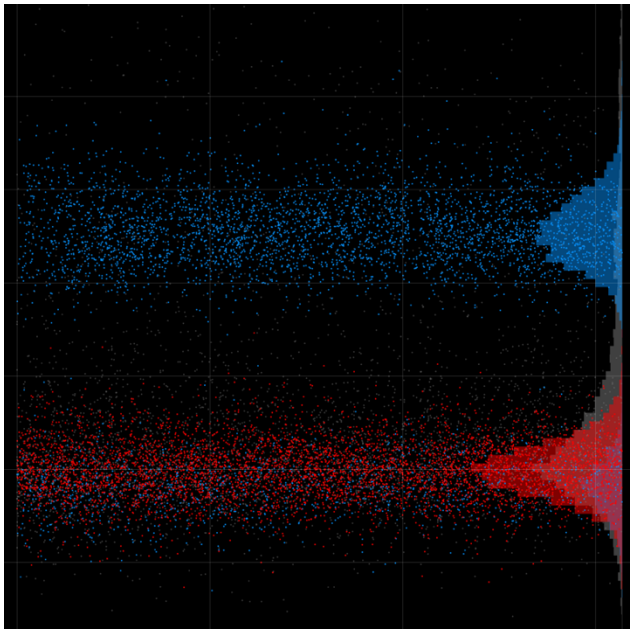

Figure 2.14 - The amplitude view of the spikes for two selected unit.

2.3.2 Criteria to Merge, Split, and Remove Units

As mentioned above, the GUI has many different features which can help evaluate the quality and reliability of an extracted unit. The main goal is to have a clean isolated neuron that has the following characteristics:

Waveform view: An important criterion for identifying units is that they should exhibit high amplitudes, which indicates that they are located close to the probe. Additionally, the highest amplitude of a unit should appear in no more than two or three channels that are adjacent to each other geometrically. For instance, if a neuron is located in close proximity to a channel on the probe that generates a spike with high amplitude, the amplitude of the unit should decrease as the distance between the neighboring channels and that channel increases. Conversely, a unit that produces high-amplitude spikes in all channels would be regarded as noise.

Correlogram view: This view (*Figure 2.13*) enables the visualization of the ACG and CCG. According to the refractory period hypothesis, a neuron cannot fire again within 2.3 milliseconds of its first firing. Therefore, in the ACG plot, the lags of 2.3 milliseconds are expected to be zero and empty. If there is noise in the refractory period or in the ACG plot overall, it is advisable to search for overlapping PCA features and divide the unit into different ones. Conversely, if two units with clean refractory periods are compared using CCG, there should be no similarity between the CCGs of the two units with different spike timing and features. A clean refractory period in CCG suggests that the two units have the same refractory period, which is incorrect, and they are likely two identical units that should be merged.

Amplitude view: The spike amplitude view of a neuron is expected to look Gaussian due to the central limit theorem[62]. This theorem states that when many independent and identically distributed (i.i.d.) random variables are added together, the resulting distribution tends toward a normal or Gaussian distribution, regardless of the original distribution of the individual variables. In the case of spike amplitudes, the i.i.d. random variables are the small electrical signals that are picked up by the recording electrodes, and the resulting distribution of these signals is expected to be Gaussian due to the central limit theorem. Therefore, a Gaussian distribution of spike amplitudes indicates that the underlying electrical signals are likely to be independent and identically distributed, which is a desirable property for accurate spike sorting. However, this assumption could be wrong depending on the type of cell or region of the recording. But it is always good to check units that do not have a gaussian amplitude distribution, to make sure other features look appropriate.

Feature view: Merging or splitting units is always done through this view. Various extracted features from spikes of units are represented and there is the possibility of checking the similarity of the selected units with others. Usually, it is expected that each unit has a homogenous feature view. However, it is possible to see two or three groups of features scattered which indicates the possibility of splitting units into different units. In total, it is expected that two different units have fewer shared features and any commonality in features raises the possibility of merging the units.

2.4 Separation of Excitatory and Inhibitory Cells Using Cell Explorer

Cell Explorer[8] is a tool for separating excitatory and inhibitory cells in electrophysiology data. It is designed to be used in combination with automated spike sorting algorithms, which are used to identify and classify individual spikes (or action potentials) in an electrophysiology recording.

After automated spike sorting has been performed, Cell Explorer can be used to identify excitatory and inhibitory cells based on their electrophysiological properties. This is typically done by examining the waveforms and other features of the extracted units and comparing them to known patterns for excitatory and inhibitory cells. Cell Explorer provides a number of tools to assist with the separation of excitatory and inhibitory cells, including:

1. A graphical user interface (GUI) that allows users to view and interact with the extracted units.

[8] https://github.com/petersenpeter/CellExplorer

2. Tools for visualizing and comparing the waveforms and other features of the extracted units.
3. A suite of data analysis and visualization tools, including tools for clustering, dimensionality reduction, and spike sorting quality assessment.
4. A plugin system that allows users to extend the functionality of Cell Explorer with custom scripts or plugins.

Separating excitatory and inhibitory cells is an important step in the spike sorting process, as it allows researchers to study the specific contributions of these different types of cells to brain function. Cell Explorer is designed to make this process more efficient and accurate, helping researchers to more confidently answer research questions related to excitatory and inhibitory cells.

2.4.1 Cell Explorer GUI Interface

In the standard configuration of Cell Explorer, there are two rows of plots (*Figures 2.15A*, *2.15B*, top panel). The top row displays population-level representations, while the bottom row shows single-cell features. By selecting a neuron in any of the plots, the other features of that neuron will automatically update. In the visualization interface, users have the ability to zoom and pan by scrolling and dragging any plot, as shown in *Figure 2.15C*. By clicking the middle mouse button, they can establish a link to the selected neuron, while a right mouse click allows them to select one or more neurons from any of the plots for further actions. The selected groups can be displayed individually or highlighted and superimposed on data within the same session, multiple sessions, or the entire database. This provides flexibility in examining and analyzing the selected neuron groups within

different contexts. Users can select clusters of neurons by drawing polygons using their mouse cursor, and the other features of the selected groups are shown separately through group actions. It is also possible to make multiple group selections for visualization and statistical comparison. The GUI interface offers side panels on both sides of the graphs, providing flexibility in customization. The left side panel offers various options, including custom group plots, color groups, display settings, selection of single-cell plots, and legends. On the other hand, the right-side panel is dedicated to single-cell actions, featuring navigation elements, cell-assignment actions, tags, and a table displaying metrics. Additionally, the left side panel incorporates a text field that allows users to apply custom filters, enabling numeric and string filtering based on specific criteria. These side panels enhance the user experience by providing a range of options and functionalities for data exploration and analysis. A message log located below the graphs keeps track of user actions. Cell Explorer provides a record of all user actions that can be undone step-by-step. There are various group plotting options available, including a 2D and 3D representation, a double histogram, a raincloud plot [63], and a customizable dimensionality-reduction plot (T-distributed stochastic neighbor embedding [t-SNE], principal component analysis [PCA], and uniform manifold approximation and projection [UMAP]) [64]. Axis scaling can be either linear or logarithmic (*Figure 2.15F*). Cell Explorer offers multiple visualization and measurement options for various cell features. One example is the visualization of spike waveforms, which can be done in several ways:

1. Single waveform: Displaying a single average waveform, filtered or raw, from the channel with the highest amplitude.

2. Population comparison: Comparing the waveform to Z-scored or absolute waveforms of the entire population.
3. Group averages: Comparing the waveform to group averages, such as specific cell types.
4. Probe-wide visualization: Representing waveforms across the probe using different methods, including probe-layout projection, image representation, and spike-amplitude distribution.

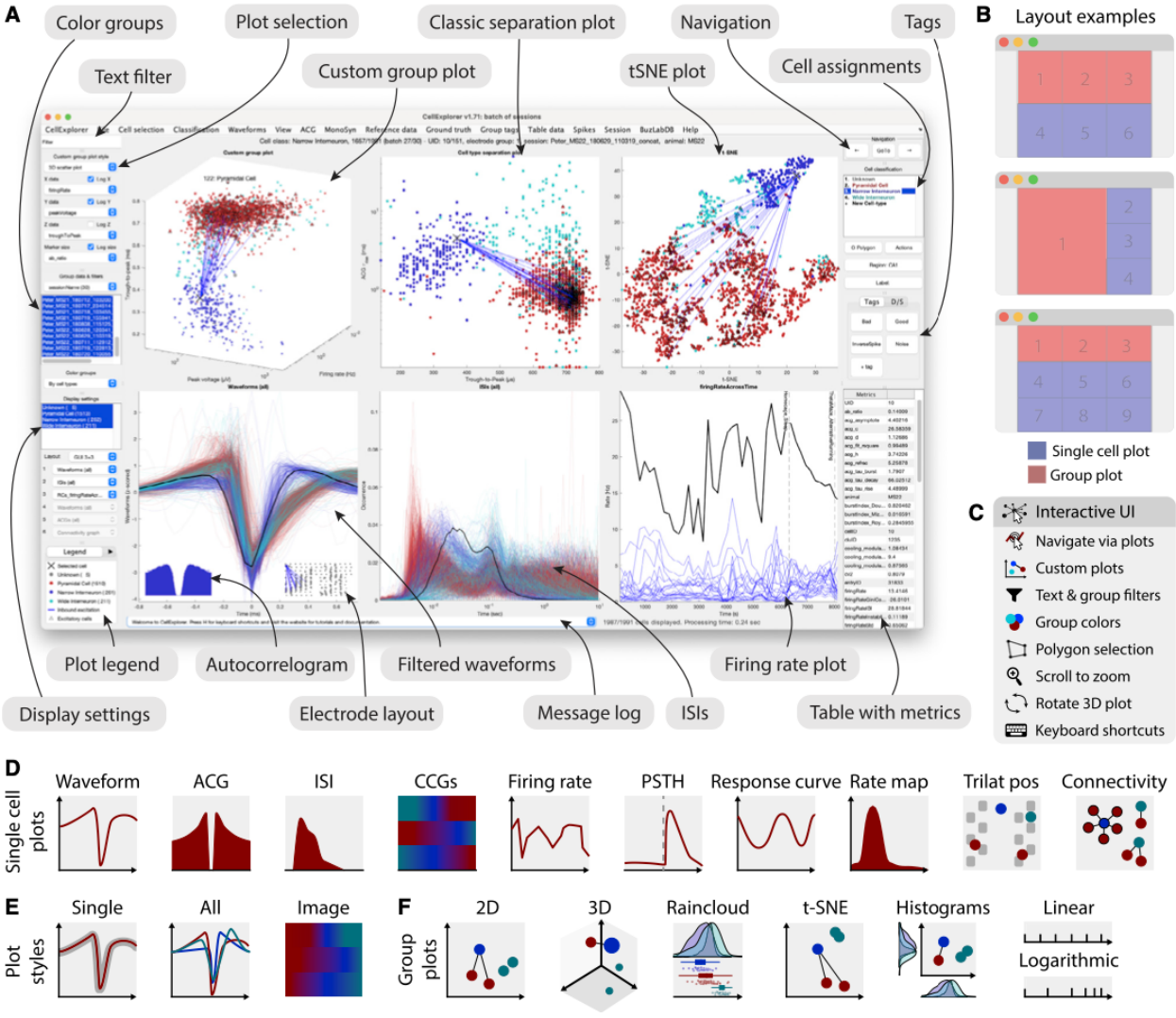

Figure 2.15 - Cell Explorer GUI [67].

5. Single-cell metrics: Calculating and displaying metrics for individual cells, such as trough-to-peak (the duration between the lowest point and the highest point of the waveform), amplitude-to-background ratio (measuring waveform asymmetry), peak voltage (amplitude of the waveform), and peak channel.
6. Temporal aspects: Analyzing the temporal aspects of spike amplitude throughout the recording as a stability measure, which can be visualized using a spike raster.

These various visualization and measurement options in Cell Explorer provide a comprehensive understanding of cell features, allowing users to explore and analyze their data from different perspectives.

Figure 2.16 demonstrates the flexible and dynamic functionality of the graphical interface module. The module showcases motifs of monosynaptically connected neuron clusters from the hippocampal CA1 area, identified by the processing module (*Figure 2.16A*). *Figure 2.16B* highlights a sub-network of interconnected neurons, with a specific neuron selected for further analysis (indicated by an arrow). Selected metrics for the single neuron are displayed in *Figure 2.16C-G*, ranging from first- to third-level metrics. Several panels display the metrics of the selected neuron alongside other neurons in the same dataset for comparative purposes. A middle mouse click on any neuron automatically updates all relevant panels, enabling quick and efficient screening and qualitative evaluation of multiple features. Once neurons of interest are marked, the graphical interface module in Cell Explorer facilitates quantitative comparisons with paradigm-specific features of the selected neuron(s) at different levels of metrics. For example, in the context of hippocampal neurons, comparisons can be made on features like place field characteristics, trial-by-trial

variability of firing patterns, firing specificity with respect to travel direction, spike-phase precession relative to theta-oscillation cycles, and several other relevant measures.

During the data-mining process, the graphical interface allows for the observation of unexpected features and outliers. It enables the identification of neuron instabilities, often referred to as "drifts," and aids in recognizing artifacts visually. These experimenter-supervised judgments are critical in evaluating the quality of quantified data processing and provide valuable insights into the validity and reliability of the obtained results.

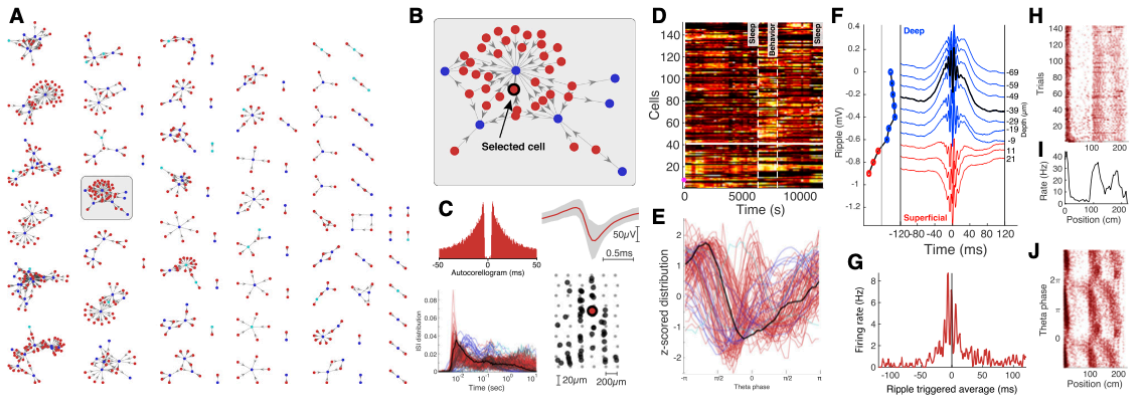

Figure 2.16 - Cell Explorer data exploration example.

2.4.2 Running Cell Explorer in MATLAB

Cell Explorer running on MATLAB can be easily installed based on instructions on their website [65]. As mentioned in the previous section, after spike sorting with Kilosort and manual curation using Phy, all the data and extracted features were stored in a specific folder. After inputting the directory of the desired folder, the following steps were performed to separate the cells into 3 main categories of Narrow Interneurons, Wide Interneurons, and Excitatory Cells:

1. **Session metadata:** In the first step, all the information about recordings is inputted into the Cell Explorer. By running the session template, different parameters can be imported using the menu in *Figure 2.17*. In the Epochs section, the time range of concatenated recordings (1. In the home cage before the behavioral task, 2. In the behavioral task, 3. In the home cage after the behavioral task, around 100GB) can be entered. Moreover, it is essential to always check the extracellular and spike sorting to ensure that the number of recording channels and the sampling rate are correct.

2. **Process Cell Metrics:** After inputting all the information about the recording by running *ProcessCellMetrics* all the metrics will start to compute. Depending on the number of units extracted by Kilosort, this can take from a few minutes to hours.

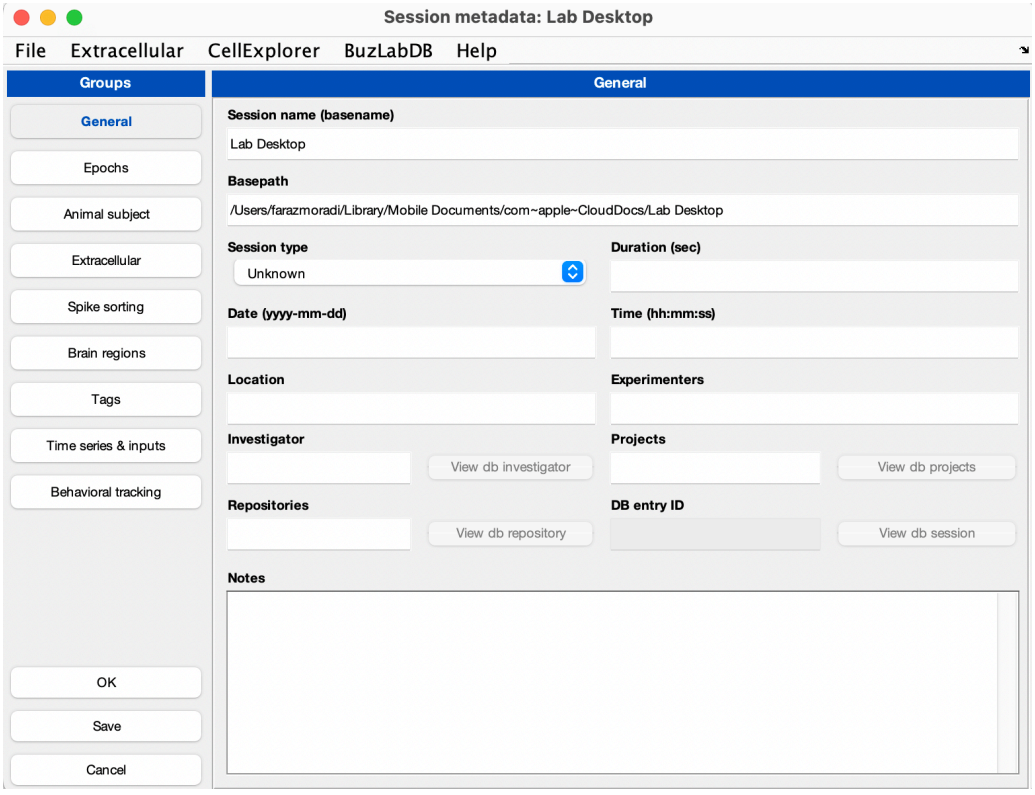

Figure 2.17 - Cell Explorer session metadata for inputting data.

3. **CellExplorer:** In the final step, by running *CellExplorer* the menu in *Figure 2.15* would be presented which contains all the desired metrics.

2.4.3 Cell Explorer Output Data Structure

Cell Explorer computes various metrics from each cell which are useful in other parts of the analysis. The information is stored in a .mat file, which can be loaded into Python. The name of the folder containing your data will be the first name in the file (*basename* was used here). A full description of each output file is provided below:

Basename.session.mat: The session struct contains all session-level metadata. All the experiment information such as sampling rate, duration of the recording, number of channels, different epoch timings, and lots of other metadata is stored in MATLAB structure which will be loaded as a dictionary in Python.

Basename.spikes.cellinfo.mat: This structure contains all the information about the time-based characteristics of the different units. Spike times in seconds and time points, maximum waveforms for the units, and raw and filtered waveforms.

Basename.cellmetrics.cellinfo.mat: This structure contains the most critical information about different features extracted from each unit. ACGs with different bin sizes, inter-spike intervals (ISI) in various scales, firing rates, burst index, through to peaks, sharp wave ripples, labels of the cells, and monosynaptic connections. There is significantly large additonal information, which can be inputted and discussed through the cell explorer website. For our analysis, the pipeline was constructed on top of these data formats for further analysis.

Basename.session.mat is essential for us to separate the three states (in the home cage). *Basename.cellmetrics.cellinfo.mat* would be critical to find the labels of each extracted unit (narrow interneurons) and probably visualizing other features like ACG and ISI. All the raw spike times are extracted from *Basename.spikes.cellinfo* to perform further analysis.

Moreover, Cell Explorer provides many different plots displaying features extracted from spike sorting results. In *Figure 2.18* two main features have been extracted from sorted units: Burst Index [66], through to the peak of the waveform. Based on these two features and the shape of ACG excitatory and inhibitory cells are separated. More detailed information about how this separation was performed is available in the Cell Explorer paper [67]. Based on the classification, from the 7 extracted units, 4 are labeled as

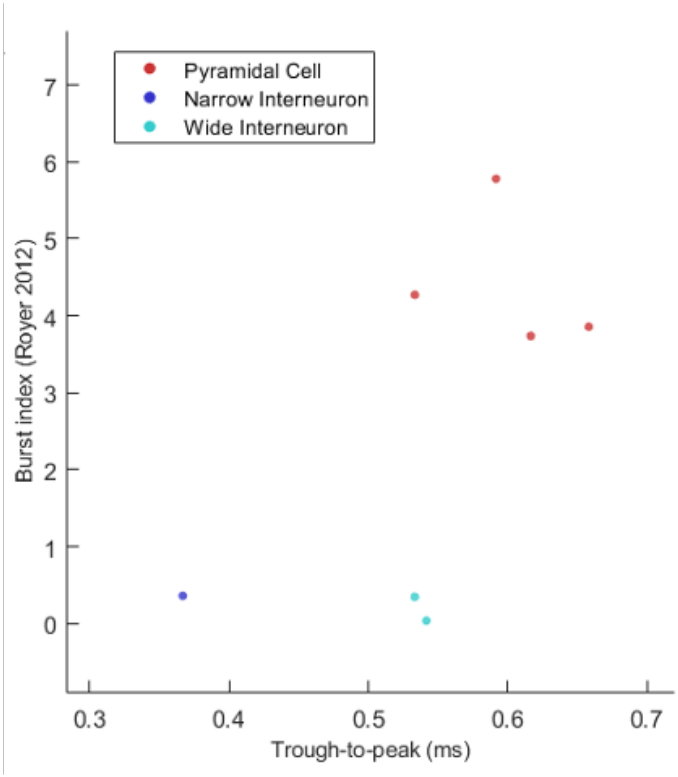

Figure 2.18 – Extracted units feature plot. 2D plot of burst index and through-to-peak for extracted units classified in three types of cells.

excitatory cells (combination of GCs and MCs), and 3 cells are labeled as interneurons (2 wide interneurons and 1 narrow interneuron).

Chapter 3 - Separation of Mossy Cells and Granule Cells

As previously mentioned, the hippocampus contains a variety of cells, particularly in the hilus region. The Cell Explorer toolbox was used to sort spikes and differentiate between excitatory and inhibitory neurons. However, while excitatory cells were successfully identified, discrimination between the two types of excitatory cells (MCs and GCs) had not yet been achieved. To address this issue, a method based on a recent study by Senzai et al.[68] was implemented, which identified three main features for distinguishing between MCs and GCs:

1. The depth of the recorded neuron is estimated by the amplitude and polarity of the local field potential (LFP) ‘‘type 2 dentate spike’’ (DS2)[69].
2. The firing rate ratio of the unit between the awake period in the home cage and non-rapid eye movement sleep (NREM).
3. A principal component classification of the second derivative of the unit waveforms.

The process of computing each feature will be discussed in detail in the upcoming sections.

3.1 Type 2 Dentate Spikes

Bragin et al. [69] showed that there are two novel population patterns in the dentate gyrus of the awake rat, termed type 1 and type 2 dentate spikes (DS1, DS2). Dentate spikes have high amplitude (2-4 mV) and short duration (<30 ms) field potentials that occur sparsely during behavioral immobility and slow-wave sleep. Current-source density analysis

revealed large currents in the dentate molecular layer's outer (DS1) and middle (DS2) thirds, respectively. DS1 and DS2 had similar longitudinal, lateral, and interhemispheric synchrony. Dentate spikes are patterns in the local field potential (LFP) that are characterized by synchronous firing of dentate units and have a large amplitude of 0.5-2.5 mV and a duration of 20-80 ms [69], [70]. DS2 is a dentate spike that shows a similar voltage-depth profile to entorhinal cortex-evoked responses in the dentate gyrus and exhibits a polarity reversal above the granule cell layer [69]. Thus, the characteristic amplitude versus depth profile of DS2 (*Figure 3.1A-B*) can identify the recording depth within and outside the granule cell layer with high spatial precision (~20 mm, *Figure 3.1A*). This criterion, therefore, localizes neurons within, above, or below the granule cell layer.

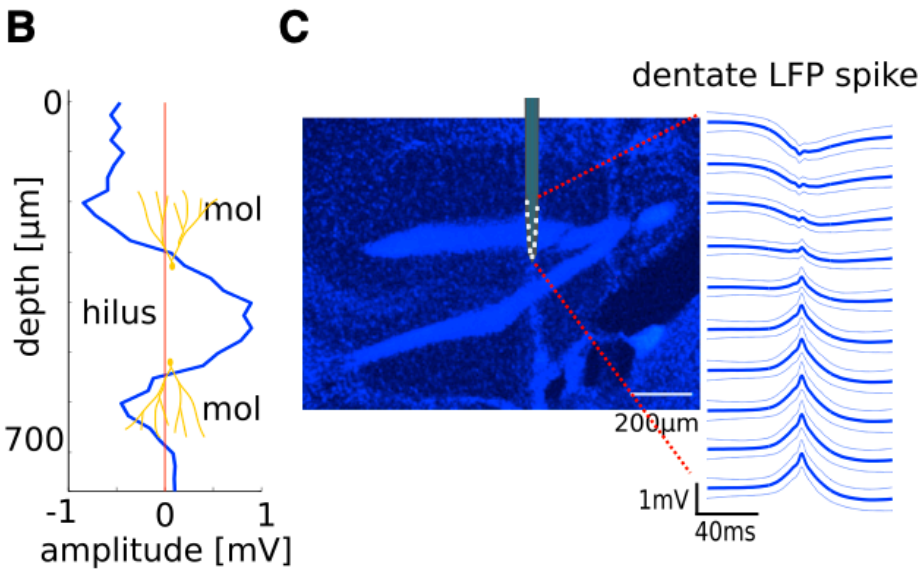

Figure 3.1 - DS2 results from Senzai et al. [68] B) Depth profile of the amplitude of DS2 recorded with a 32-site linear probe. C) LFP DS2 at different recording sites with histology results.

3.1.1 Computing DS2 Amplitude

Initially, the construction of the LFP involved applying a low-pass filter with a cut-off frequency of 450Hz. Considering that high-frequency signals have been removed from the data and the bandwidth of signal is lower than 450Hz, the high sampling rate of 30kHz becomes redundant and adds unnecessary computational overhead, making the data processing more challenging. Therefore, adhering to the Nyquist theory [71], we can reduce the sampling rate to 900Hz. To ensure that no data is missed, the dataset was down sampled to 1250Hz, making a balance between reducing computational costs and saving the necessary information for analysis. Afterwards the steps for computing the DS2 amplitude based on Senzai et al.[68] are as follows:

1. Extract a snippet of the recording for about 10 minutes.
2. Band-pass filter the data between 30-120 Hz to extract the gamma power for each channel and save the gamma power of each channel.
3. Back to step 1 and high-pass filter (10Hz) the data to remove the low frequencies (theta rhythm).
4. Find the events where the absolute microvolt of data is bigger than 5 times the standard deviation, and take the median of it across all channels. The maximum and minimum medians are the hilus and molecular layer channels.
5. After identifying the hilus and molecular layer channels, find the events in which the difference of the band pass filtered (2-50Hz) LFP recorded from a hilus location and molecular layer site exceeds 1.14mV.

6. If the mean LFP value at the molecular layer during DS2 is lower than the baseline value (–36 ms to –16 ms) by > 0.19 mV, the DS2 passes the criteria and is included to later analysis. Additionally, the peak time of DS2 is defined as the time when the dentate hilus wide-band LFP shows maximum value.

Now, the DS2 amplitude for each channel has been computed. From the Cell Explorer and also Phy it is possible to find the channel with the maximum waveform amplitude for each unit. After finding the peak channel amplitude for each unit, the DS2 amplitude of that channel is assigned to that unit. Therefore, for each unit, the DS2 amplitude was computed and this can produce a geometrical property for extracted units among the hilus and molecular layer.

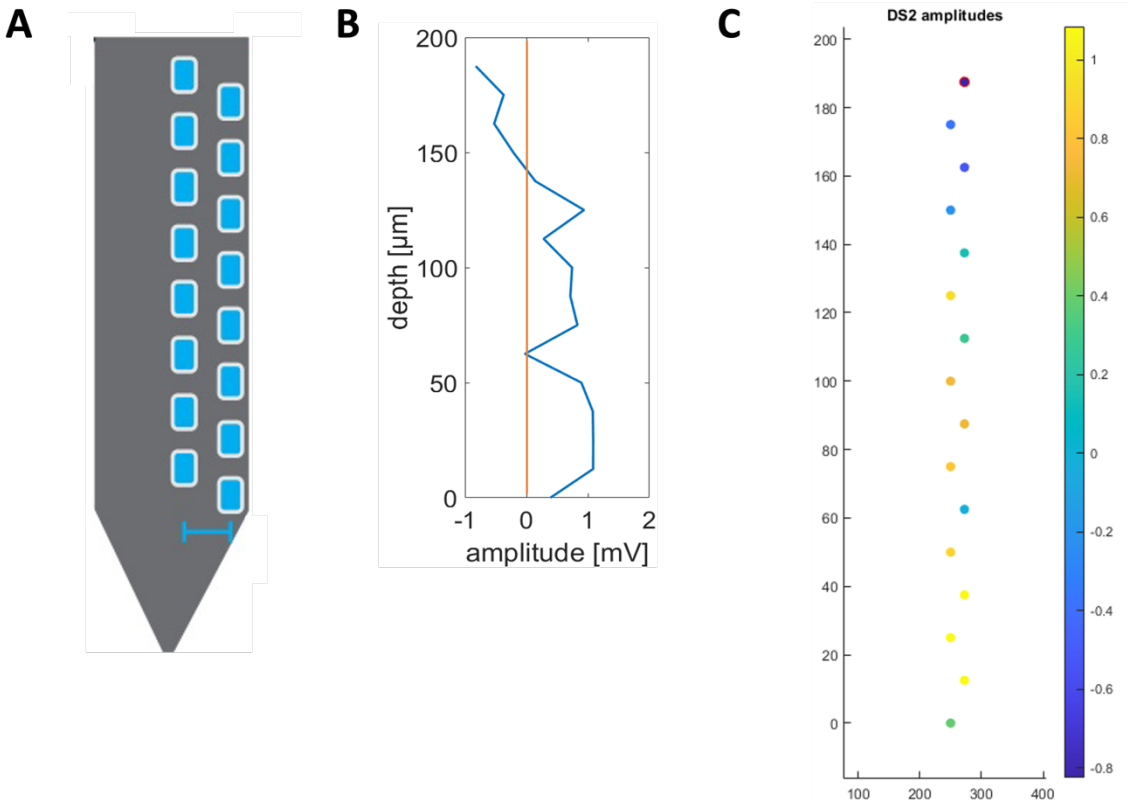

Figure 3.2 - DS2 results for the performed recording. A) The right shank of the used probe with channels. B) Depth profile with the DS2 amplitude of each channel on the probe. C) DS2 amplitude color bar of the channels on the probe.

Figure 3.2 provides a summary of the DS2 results, indicating that the recording was conducted within the hilus region. As mentioned in the section 2.1.2, from the two available shanks, shank right was used for spike sorting and analysis (*Figure 3.2A*). In *Figure 3.2B*, the DS2 amplitudes are plotted as related to the depth of the electrode. Starting from the top, after electrode 5 there is an increase of the DS2 amplitude, which will be considered as the beginning of the hilus layer. This can be seen in *Figure 3.2C,* which shows an increase in the DS2 amplitude at the bottom of the probe. These results agree with Senzai et al. [68] results in *Figure 3.1*.

3.2 Firing Rate Ratio Between Awake Period and NREM sleep

Sleep in mice consists of three stages: WAKE, non-REM, and REM (rapid eye movement sleep)[72]. These stages play different roles in the lives of mice, and can be identified by inspecting EEG and electromyogram (EMG) signals. Identifying the sleep stages of mice from their EEG and EMG signals (hereinafter referred to as “sleep stage scoring”) is one of the most important analyses in sleep research. For example, the symptoms of sleep disorders and the effects of sleeping pills can easily be verified by analyzing the rates/transitions of sleep stages. In our project, identifying sleep/wake cycles is important for differentiating GCs and MCs which show differing activity during sleep.

Sleep Stages of Mice

The sleep and wake states of mice can be classified into three stages: wakefulness, non-REM, and REM. Each stage is distinguished by its unique features in EEG and EMG

signals. Specifically, the peak frequency of EEG signals and the amplitude of EMG signals are critical features for differentiating between the stages.

Wake: During WAKE stages, mice are either awake or drowsy, and both their brains and bodies are active. As a result, the EEG signals exhibit mixed frequencies, while the amplitude of EMG signals tends to be high (*Figure 3.3A*).

Non-Rem: During non-REM stages, the brain exhibits cortical synchronization and the body is at rest. As a result, the peak EEG frequencies become lower and the amplitude of EEG signals becomes higher compared to the WAKE stages. In addition, the amplitude of EMG signals tends to be smaller (Figure 3.3B). Non-REM is the predominant sleep stage, accounting for over 90% of sleep.

Rem: In REM stages, the brain is as active as in WAKE stages, but the body exhibits a state of "REM atonia," which leads to lower EMG amplitudes (Figure 3.3C). REM atonia refers to the temporary paralysis or loss of muscle tone that occurs during REM sleep. It is important to note that healthy mice in REM stages remain completely still, except for breathing, eye movement, and occasional muscle twitches.

Additionally, the transitions between sleep stages follow certain rules. For example, REM stages typically occur at regular intervals during sleep and can last from several tens of seconds to several minutes. Moreover, healthy mice do not immediately transition from WAKE to REM stages. Understanding these rules of transition can improve the accuracy of sleep stage scoring.

The process of manually scoring sleep stages usually involves two steps: acquiring biological signals and identifying sleep stages. Firstly, experts measure EEG and EMG signals from the brain surface and neck muscles of mice for hours. The signals are divided

into epochs, which are constant time intervals ranging from 4 to 20 seconds. In the second step, experts visually examine the frequency components and amplitudes of the signals and assign a sleep stage label to each epoch based on the dominant features and the known rules governing sleep stage transitions.

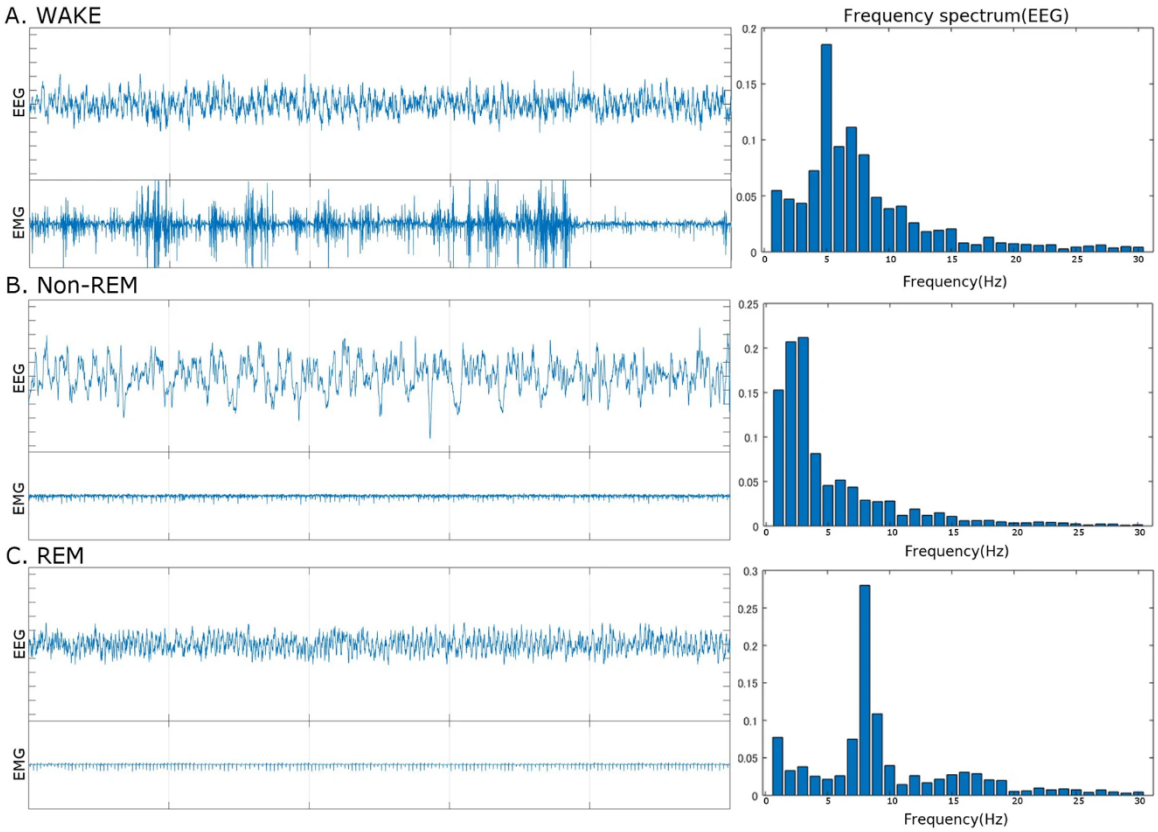

Figure 3.3 - Examples of EEG signal in different stages. [72]

3.2.1 Computing Awake / NREM Ratio for Each Cell

In order to assign the animal's state to the mentioned stages (Wake, NREM, REM), Brandon et al. [73] method was used. The steps based on their paper are outlined below:

1. LFPs were converted into spectrograms and principal component analysis was performed on each spectrogram (*Figure 3.4A-B*). Converting to spectrograms could be done using time-frequency transforms such as Wavelet or Short-time Fourier transforms.

2. The first principal component (PC1) of the spectrograms was computed to represent power in the low-frequency range, with weights of frequencies <25 Hz opposite in sign to those of frequencies in the gamma range.

3. Since the magnitude of PC1 showed a bimodal distribution (*Figure 3.4C*), a threshold was set at the trough between the two peaks of the distribution, which allowed us to classify each second as either NREM (high PC1 power) or ‘‘other’’ state (low PC1 power).

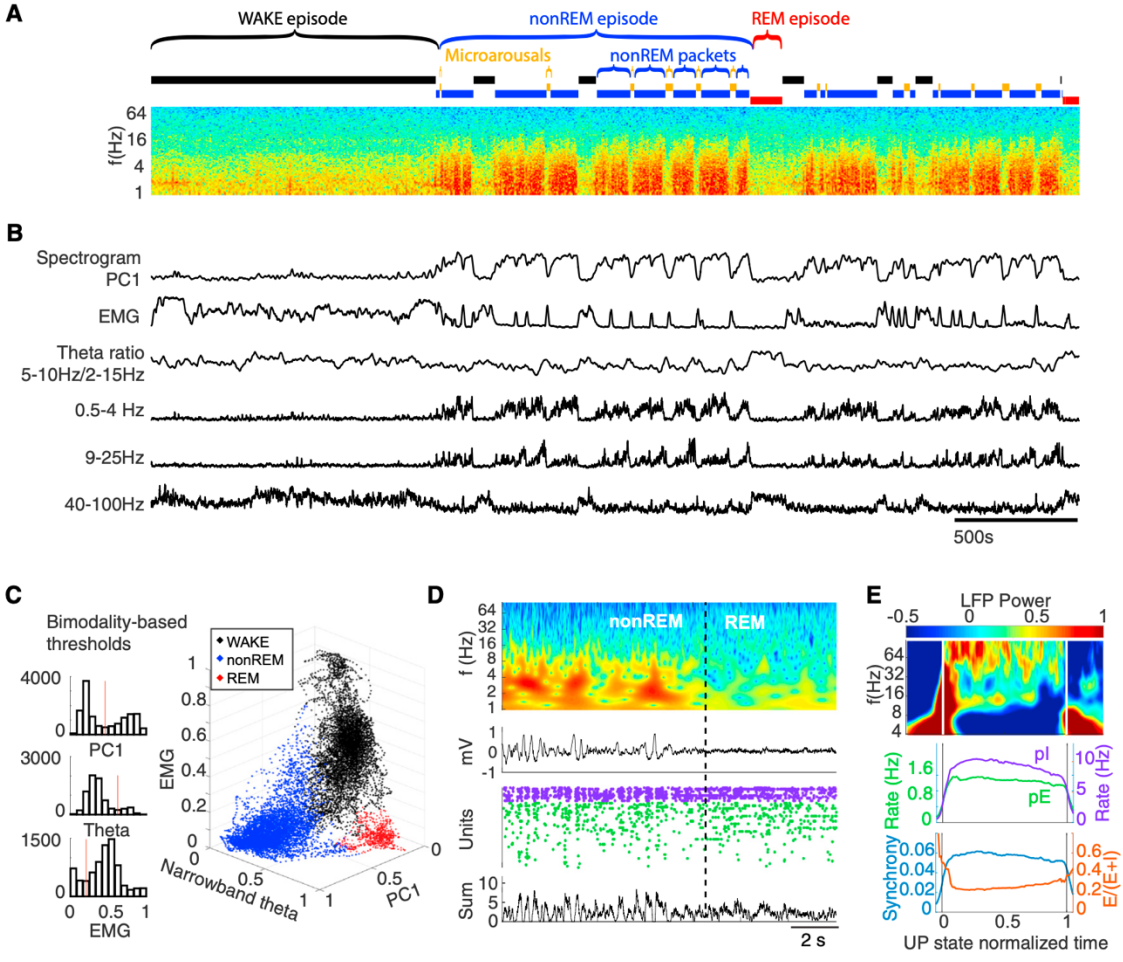

Figure 3.4 - Classification of brain states using Watson et al. [73]

4. The next step was to classify the ‘‘other’’ epochs using a narrow band theta power ratio (5–10 Hz/2–16 Hz) and electromyogram (EMG) measures (*Figure 3.4C*), also using cutoffs at the minima of bimodal distributions on a per recording session basis. Before proceeding, epochs with high theta power and low EMG activity were designated as REM. In code, the *SleepScoreMaster* subfunction from Buzcode[9] can be used which is based on the mentioned method. This function needs various inputs, which can be exported from Cell Explorer. In the output of this function, each second of the recording was assigned to one of the three states (Wake:1, NREM:3, REM:5). For computing the firing rate ratio of Wake and NREM, the spiking activity of the desired units for both states will be separated. Afterward, the firing rate of the unit for each state will be computed and divided by each other. Based on the two factors mentioned, Senzai et al. [68] depicted these features for a large number of cells in *Figure 3.5A*. MCs used to have higher DS2 amplitude and higher Wake/NREM ratio and for the GCs it was the reverse. In order to see how extracted

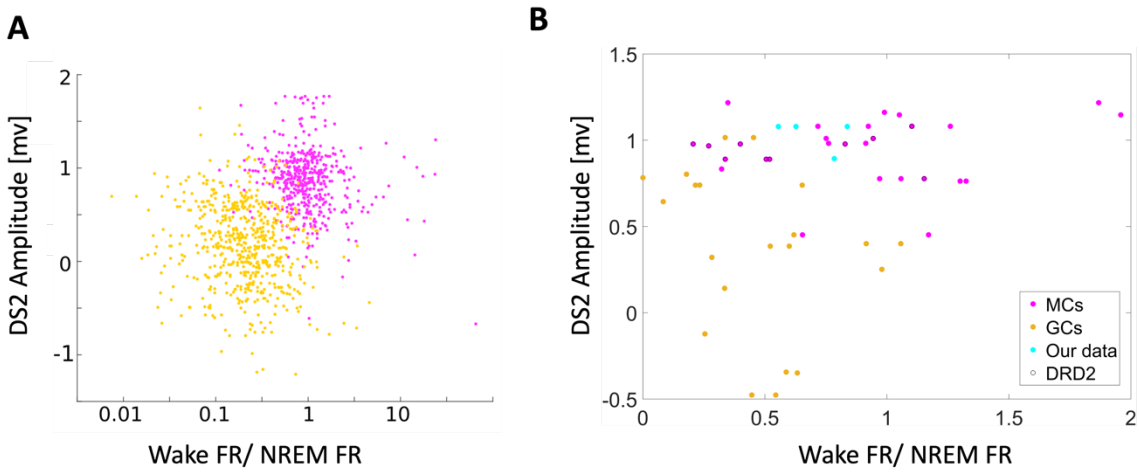

Figure 3.5 – MCs and GCs feature plot. A) 2D plot of DS2 amplitude and Wake / NREM firing rate ratio for GCs and MCs (1086 units) from Senzai et al. [68]. B) 2D plot of DS2 amplitude and Wake / NREM firing rate ratio for GCs and MCs from our recording (5 units) and one sample recording of Senzai et al.

[9] https://github.com/buzsakilab/buzcode

units, resemble real MCs and GCs, extracted features for the units were plotted against one sample recording from Senzai et al. (*Figure 3.5B*). From the 4 extracted units, 3 are closer to MCs, and 1 with a lower DS2 amplitude is closer to GC.

3.3 Second Derivative of the Unit Waveforms

In *Figure 3.6A*, Senzai et al. [68] have shown that the waveform shape of the MCs and GCs is a bit different. The waveforms of GCs used to be sharper than those of MCs. Therefore, they tried to compute the second derivative of the waveform which is an appropriate representative of waveform sharpness. This feature is easy to compute in MATLAB and was used in our classification task. In *Figure 3.6B*, the same features for our recording are shown. From extracted excitatory cells, the same unit mentioned in the

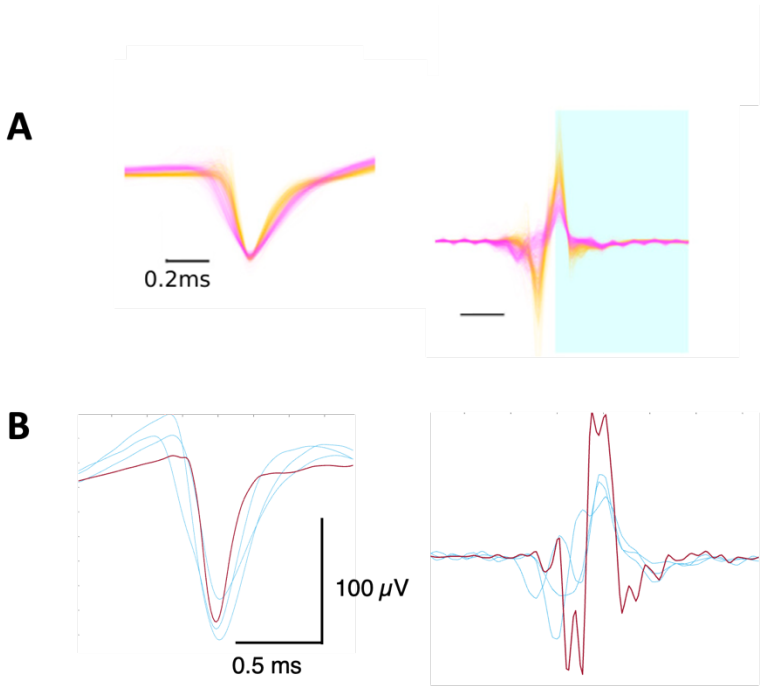

Figure 3.6 – MCs and GCs waveform plot. A) Normalized waveform and second derivative of GCs and MCs (1086 units) from Senzai et al. [68]. B) Normalized waveform and second derivative of GCs and MCs from our recording (5 units) and one sample recording of Senzai et al.

previous section as a GC has the sharpest waveform and highest second derivative, making it much more similar to a GC.

3.4 MCs and GCs Classifier (Clustering)

Clustering is the task of dividing the population or data points into a number of groups. In this way, data points in the same group are more similar to other data points in the same group and dissimilar to data points in other groups. It is basically a collection of objects based on their similarity and dissimilarity between them. Clustering is basically a method of unsupervised learning. An unsupervised learning method is a method in which references are drawn from datasets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, generative features, explanatory underlying processes, and groupings inherent in a set of examples.

3.4.1 K-means Clustering

K-means clustering is a well-known unsupervised machine learning algorithm that is relatively simple to implement. Unlike supervised learning, unsupervised learning algorithms such as K-means use input data without labeled outcomes to draw inferences. The goal of K-means clustering is to group similar data points together and uncover underlying patterns in the data. To accomplish this, the algorithm seeks a fixed number of clusters, or groups of data points that share certain similarities. The algorithm defines a target number k that refers to the number of centroids, or center points of each cluster, needed in the dataset. Each data point is assigned to the nearest centroid, and the algorithm works to minimize the in-cluster sum of squares. This process involves identifying k centroids, assigning data points to clusters, and recalculating the centroids until the

algorithm converges to a stable solution. In data mining, the K-means algorithm begins by randomly selecting a set of centroids, which are used as the starting points for each cluster, and then iteratively updates the positions of the centroids to optimize cluster assignments.

3.4.2 K-means Clustering on Extracted Features

Earlier chapters have discussed the extraction of three main features from the units, including DS2 amplitude and Wake/NREM ratio, which are single numbers computed for each unit. However, the second derivative of each unit's waveform is a single dimension array that requires conversion into a proper format for the K-means classifier. To accomplish this, PCA was conducted on the average waveforms, which were first normalized by peak amplitude and up-sampled to 100kHz using spline interpolation. Waveform PCA was performed on the time between 0 and 0.8ms [10] period of the second derivative of the up-sampled average waveform, resulting in w-PC1 and w-PC2. To estimate the unit's anatomical location, the amplitude of the LFP DS2 at the recording site, where the units' maximum waveform amplitude was observed, was used. Firing rate ratios for each unit were also calculated between nonREM sleep (NREM) and waking. Excitatory cells were classified into two clusters using the K-means method based on w-PC1, w-PC2, DS2 amplitude, and the ratio of firing rate during NREM and waking. It is worth noting that due to the requirement for large datasets in machine learning problems, sample recordings from Senzai et al. [68] were utilized to increase the number of cells and improve the classification performance.

[10] The length of a waveform is approximately 1-2 ms. Also, the belief is that the waveform of GC is sharper than MC, and the sharpness will be utilized for classification purposes. By extracting the segment from 0 to 0.8 ms of the waveform, the slope of the spike during depolarization to repolarization, which is contains the sharpness will be extracted.

Chapter 4- Spike Train Analysis

In the previous sections, all the methods and processes needed to achieve an isolated unit were mentioned. Different toolboxes were discussed for spike sorting and separation of excitatory and interneurons from each other. Moreover, a K-means classifier was discussed for the classification of MCs and GCs. From the extracted units, 1 GC, 3 MC, 2 Wide Interneurons, and 1 Narrow Interneuron were discovered. The recordings from these units were for one day (one mouse) and were divided into three phases (section 2.1.2). A summary of the extracted units with their firing rate in different phases is provided in *Table 4.1*.

In this chapter, different spike train metrics were computed on the units to compare how their activity alters in the mentioned phases. Also, the activity of these cells during each trial in the behavioral section will be discussed. A list of the used libraries in Python and their subfunctions is provided in the Appendix

Table 4.1– Summary of the firing rate and cell properties of extracted units.

		Firing Rate (FR) in Different States (Hz)		
Unit Number	**Cell Type**	**Before**	**Behavior**	**After**
1	Granule Cell	2.16	1.55	2.29
2	Mossy Cell	1.4	2.84	2.02
3	Unknown (left Shank)	1.1	1.41	0.66
4	Interneuron (Narrow)	27.58	40.05	24.06
5	Mossy Cell	2.62	3.81	2.4
6	Interneuron (Wide)	6.01	8.22	5.24
7	Mossy Cell	3.43	3.34	3.1
8	Interneuron (Wide)	5.23	13.5	4.1
Summary of Statistics	Mean FR for MC	2.48	3.33	2.51
	STD FR for MC	1.02	0.49	0.55
	Mean FR for Interneurons	12.94	20.59	11.13
	STD FR for Interneurons	12.68	17.06	11.21

4.1 Comparing Different States

To explore how the behavior of units alters across three distinct states, six key spike train metrics were computed (*Figure 4.1*). This sub-module was developed in Python with full compatibility with Cell Explorer outputs. These metrics can be computed for any desired recording snippet from the original data (different sates). An example of the computed metrics for a cell at a sample phase is provided in *Figure 4.1*. These metrics will be carefully examined and explained in the following sections.

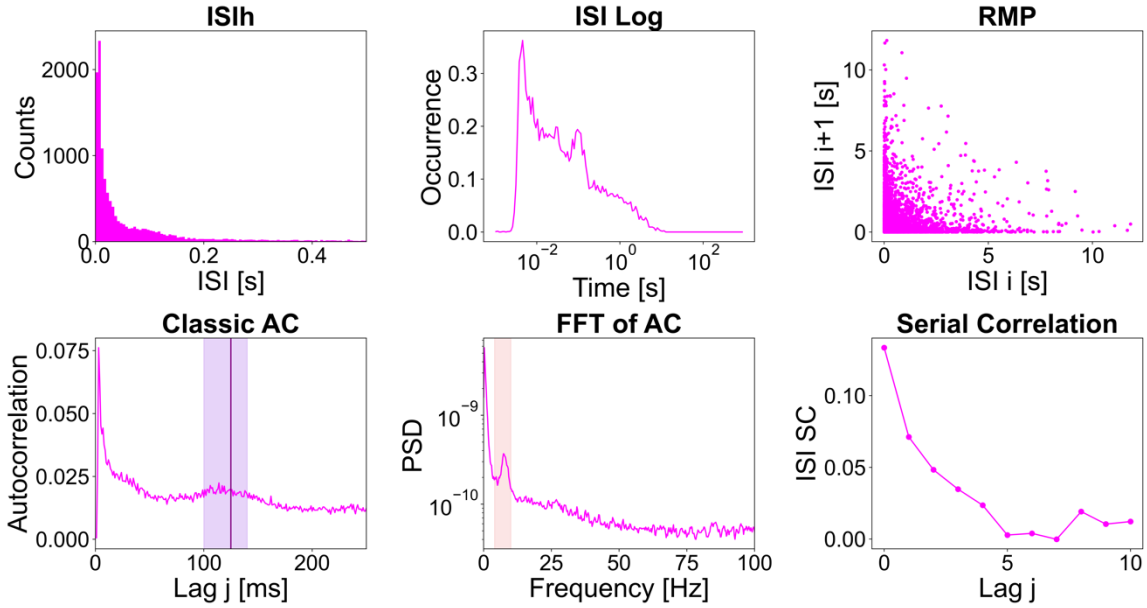

Figure 4.1 - Spike train metrics calculated for a MC in a behavioral state.

4.1.1 Inter Spike Interval

Inter-spike intervals (ISIs) refer to the time intervals between consecutive action potentials or spikes in the firing pattern of a neuron. In other words, ISI is the time elapsing between two consecutive spikes been generated by a neuron. ISIs are an important aspect of the temporal structure of neural activity, as they provide information about the patterns and

dynamics of neuronal firing. The analysis of ISIs is commonly used in the field of neuroscience to understand the behavior of individual neurons and neural networks. It can be used to study the mechanisms underlying the generation of action potentials, the coding of information in neural networks, and the modulation of neuronal firing by various stimuli. Computing the ISI can be done easily, by using *Numpy.diff* , which computes the difference of elements in an array. After computing, plotting the distribution of ISIs can provide the neuroscience researcher with valuable information. In the literature, two ways of plotting the distribution of ISIs are common:

4.1.1.1 ISI Histogram

The ISI histogram can provide insights into the temporal patterns and dynamics of neuronal firing. The shape of the histogram can reveal whether a neuron has a regular or irregular firing pattern. A neuron with a regular firing pattern will have a narrow ISI histogram, with peak at a specific time interval, while a neuron with an irregular firing pattern will have a wider ISI histogram, with no clear peak.

The ISI histogram can also provide information about the refractory period of the neuron. Refractory period is the time-period during which a neuron is unable to fire another action potential, and is typically measured as the minimum ISI that can be observed in the histogram. A shorter refractory period indicates that the neuron can fire action potentials more frequently, while a longer refractory period indicates that the neuron has a slower firing rate.

For computing the histogram of ISIs, *matplotlib.plt.hist* was applied. This function needs another input besides ISIs, which is bins. For creating the bins variable, *np.arrange* was used with a 5ms resolution (bin size).

Comparing the ISI histogram (*Figure 4.2*) of three different neurons indicates MC and Interneuron spike activity does not change significantly during different states. However, for the GC, the number of short ISIs has decreased compared to the home cage. A decrease in short ISIs may imply a modification in the temporal encoding and integration of signals by the GC. This can affect the processing of sensory information, learning, or other cognitive functions associated with the task.

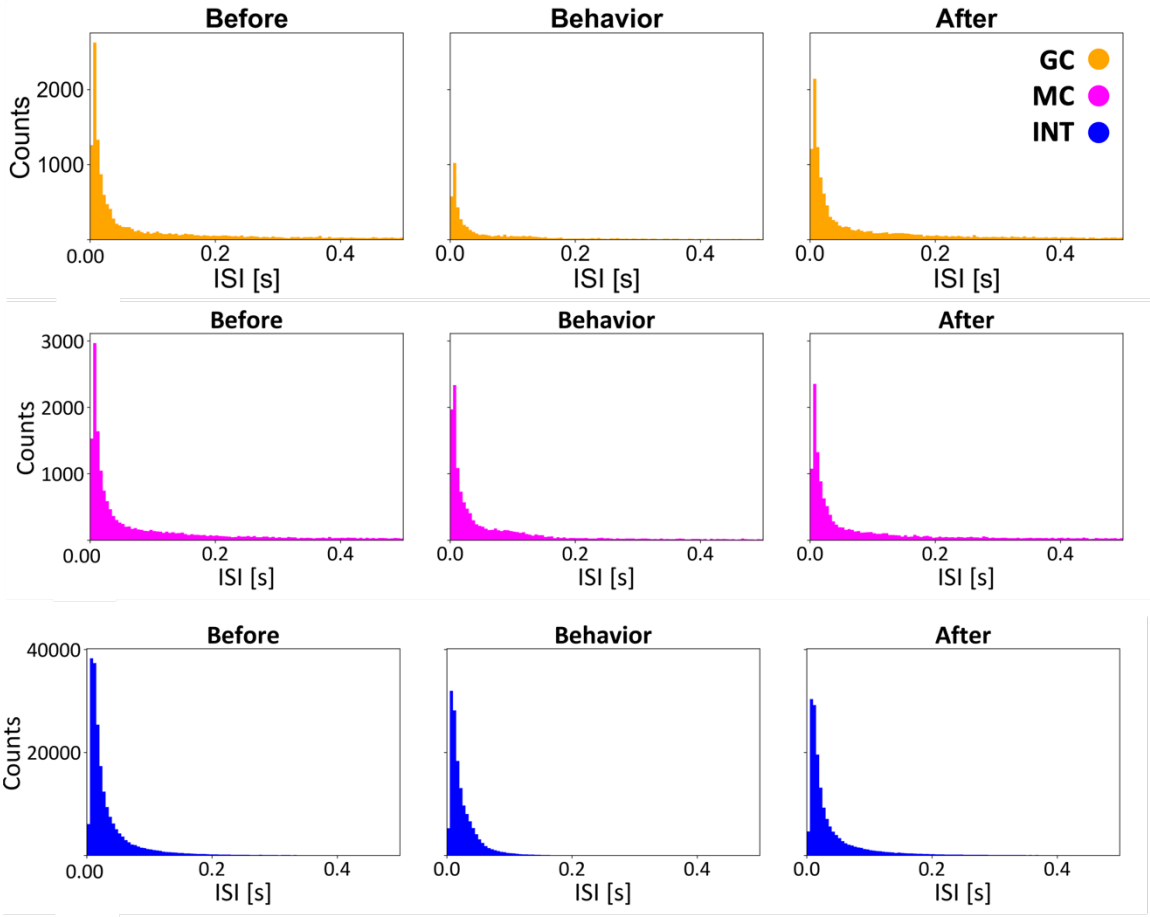

Figure 4.2 - ISIh for GC, MC and an interneuron in three different states.

4.1.1.2 ISI Log

The ISI log histogram is a variation of ISI histogram, in which the time intervals between successive action potentials or spikes in the firing pattern of a neuron are binned on a logarithmic scale. Moreover, after computing the ISIs, log10 of the intervals is computed.

In a regular ISI histogram, the time intervals are binned evenly, which in result will produce improper visualization of the distribution of short scale ISIs. By binning the ISIs on a logarithmic scale, the ISI log histogram can provide a more accurate representation of the distribution of short scale ISIs, especially for neurons with a wide range of firing rates.

In an ISI log histogram, the bins are spaced logarithmically, meaning that each bin covers a range of ISIs that is proportional to its position on the logarithmic scale. For example, if the bins are spaced at intervals of 0.1 logarithmic units, the first bin would cover ISIs from 1 ms to 10 ms, the second bin would cover ISIs from 10 ms to 100 ms, and so on. The ISI log histogram is particularly useful for analyzing the firing patterns of neurons with a wide range of firing rates, such as those that exhibit burst firing or irregular firing patterns.

For computing the ISI log (*Figure 4.3*), all the steps are the same as the ISI histogram except that np.log10 was used on ISIs before binning. Moreover, normalization was done by dividing the histogram values by the length of each bin and the total number of spikes. In the ISI log, it appears that GC activity hasn't changed much. Interneurons show a slight shift toward shorter ISIs and for the MC it seems there is an increase in the probability of

the short ISIs in the behavior. This could indicate more burstiness of MCs in a pattern separation task.

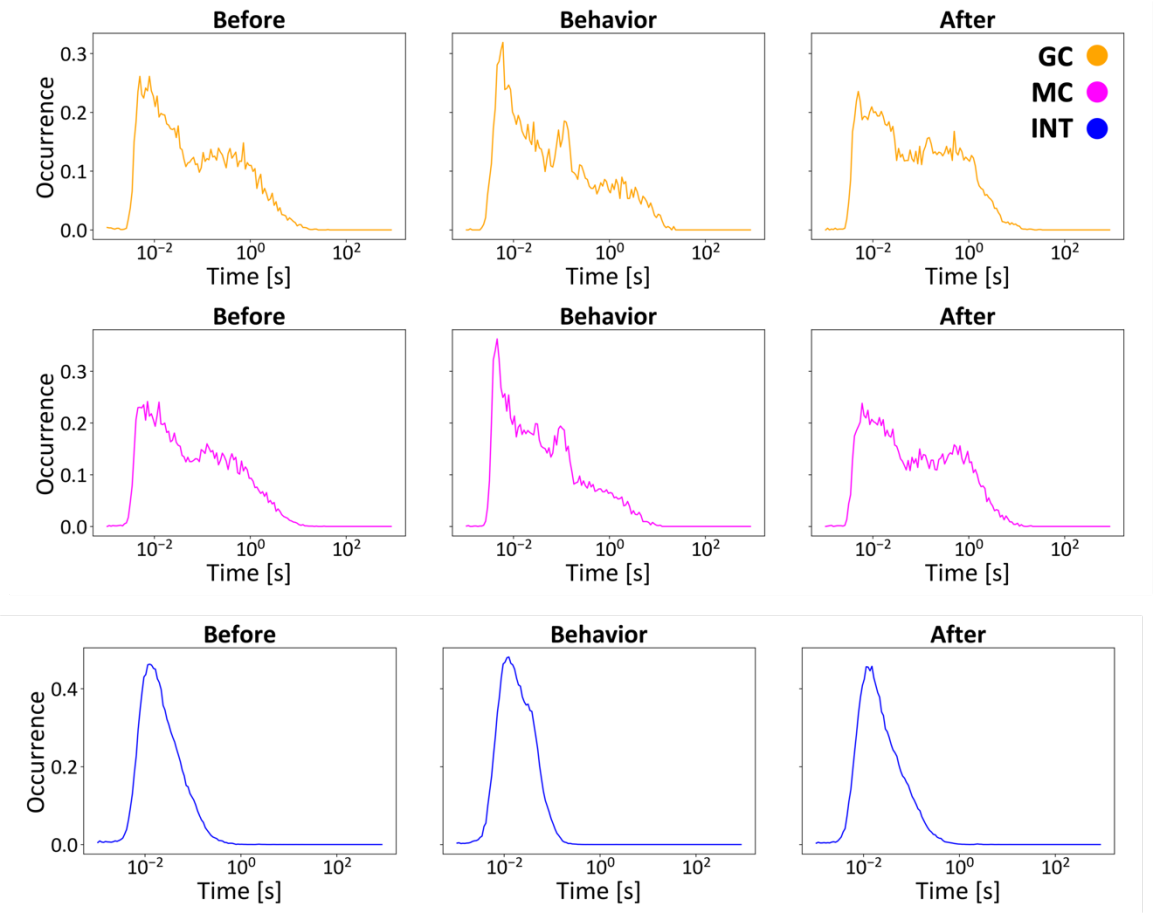

Figure 4.3 - ISI log scale for GC, MC and an interneuron in three different states.

4.1.2 Return Map

Return maps, also known as Poincaré maps, are graphical representations of dynamical systems in which the evolution of a system over time is depicted by plotting the values of a state variable (such as position or velocity) at discrete intervals against the values of the same variable at a previous time step. In other words, return maps show how a system's state at one point in time maps onto its state at a later time. These maps are often used to

study the behavior of complex systems, such as chaotic systems, and can help identify patterns or periodicities in the system's behavior.

The return map of a neuron is a graphical representation of the neuron's behavior over time. Specifically, it shows the neuron's output at a given time as a function of its input at the previous time. More specifically, the return map is a plot of the ISI of a neuron at time t+1 as a function of its input at time t. The shape and characteristics of the return map can reveal important information about the neuron's behavior, such as its sensitivity to input, the presence of oscillations or other patterns, burstiness, and the existence of multiple stable states. For GC and MC, there is no significant change in the return maps (*Figure 4.4*). They both show similar firing of short and long ISIs in all environments. . For interneurons, it

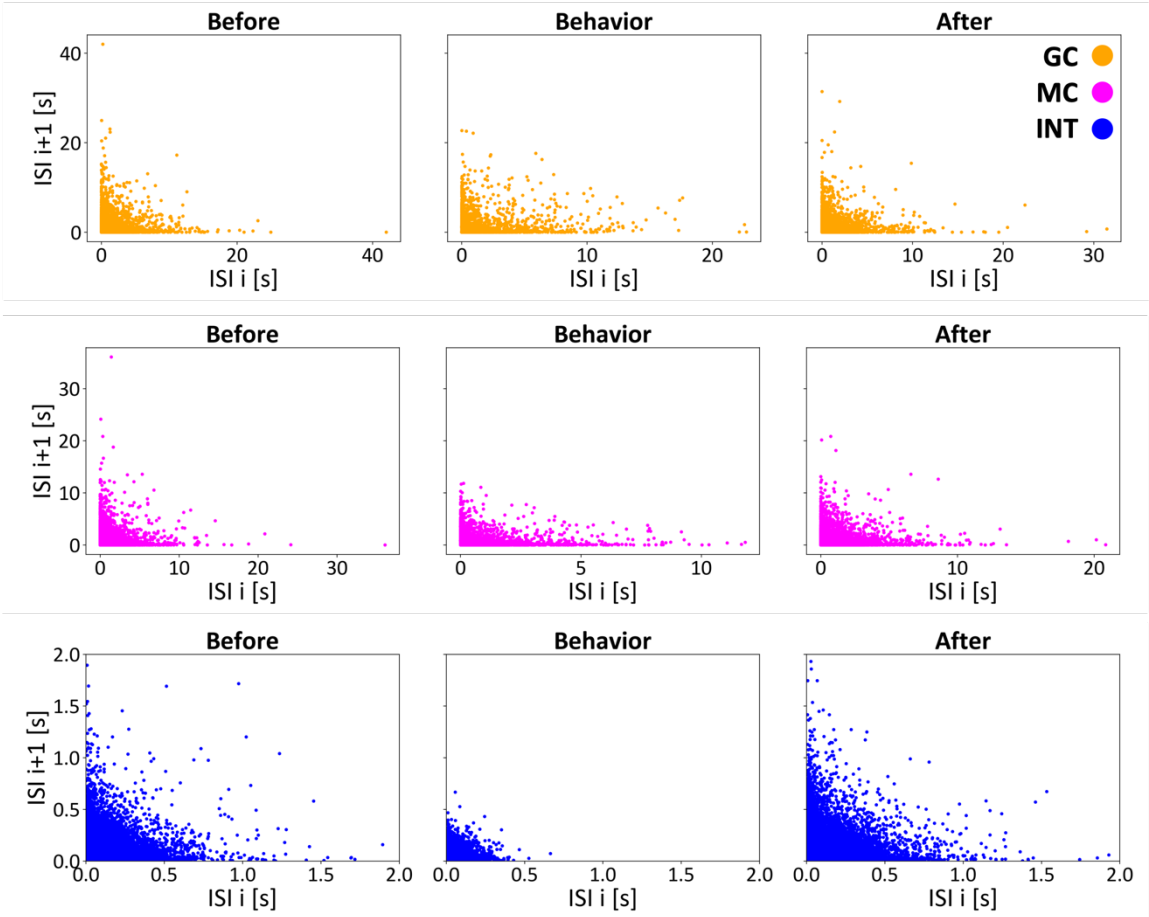

Figure 4.4 - ISI return maps for GC, MC and an interneuron in three different states.

seems the short ISIs have increased significantly in behavior phase comparing to home cage.

4.1.3 Auto Correlation

Autocorrelation is a statistical concept that measures the degree to which a time series is correlated with its own past values at different time lags. In other words, it measures the similarity between observations of a variable separated by a given time lag. Autocorrelation is often used in time series analysis to identify patterns or relationships between consecutive observations. It can be calculated using a mathematical formula that compares the correlation of the variable with itself at different lags. Positive autocorrelation indicates that the variable tends to follow similar patterns over time, while negative autocorrelation indicates that the variable tends to oscillate between positive and negative values.

The autocorrelation of a neuron refers to the degree of similarity between the neuron's firing rate at different time lags. In other words, it measures the correlation between the neuron's activity at different times.

The autocorrelation of a neuron can provide information about the temporal structure of its firing pattern. For example, a neuron with a high autocorrelation at short time lags may indicate that it tends to fire in bursts or in a rhythmic pattern, while a neuron with low autocorrelation at short time lags may indicate a more random or sporadic firing pattern.

The autocorrelation of a neuron is often analyzed in the context of a larger neural network, where it can provide insights into the dynamics and organization of the network as a whole. It can also be used to model and predict the future activity of the neuron and the network, which is important in fields such as computational neuroscience and artificial intelligence.

Computing the autocorrelation for long recordings would be challenging, as the number of lags to compute the correlation is very large. Hence, for decreasing computational cost, *scipy.signal.fftconvolve* was used. This function is used to perform convolution between two signals using the fast Fourier transform (FFT) algorithm. The main benefit of using this function is that it is considerably faster than performing convolution in the time domain.

Excitatory units (GC and MC) often exhibit burst firing patterns characterized by a rapid succession of spikes within a short time window. These bursts can result in a more rapid decay of the autocorrelation function because the high-frequency spiking tends to introduce

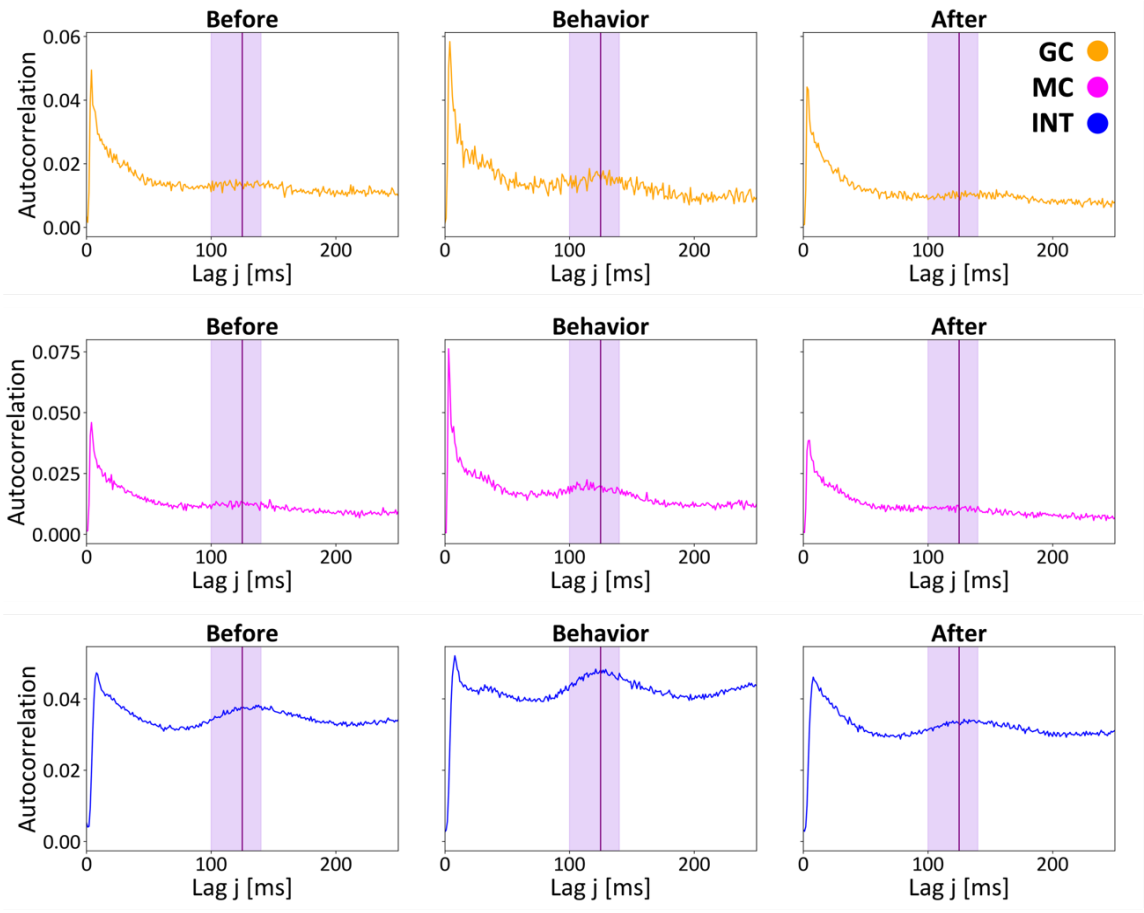

Figure 4.5 - Autocorrelation of spike trains for GC, MC and an interneuron in three different states. The lag range of 100 to 140 ms is highlighted by light pink and the exact lag of 125 ms by a pink line.

temporal correlations that diminish quickly as time progresses. In contrast, interneurons may have more regular firing patterns, which can result in a slower decay of the autocorrelation function (*Figure 4.5*). For all units, low correlation is observed in the first lags (1-5ms) which indicates the refectory period of each unit. The refractory period of a neuron refers to a brief period of time following the generation of an action potential (a rapid change in electrical potential that allows a neuron to transmit information). During this refractory period, the neuron is temporarily unresponsive or less responsive to subsequent stimuli, and it is more difficult to generate additional action potentials.

For all units a local peak in the range of 100 to 140 ms. (~125 ms) was observed. The presence of a peak at a particular lag in the autocorrelation of a cell could indicate that the activity of the cell at that particular lag is highly correlated with its own activity at that same lag. Specifically, a peak at lag 125ms in the autocorrelation suggests that the cell's activity is highly synchronized with itself at a time delay of 125ms.

4.1.4 Power Spectrum Density of Autocorrelation

The power spectrum of the autocorrelation function is a mathematical representation of the frequency content of a time series. It is obtained by taking the square amplitudes of the Fourier transform (FT) of the autocorrelation function.

In the context of neuron activity, the power spectrum of the autocorrelation function provides information about the rhythmic or periodic patterns of the neuron's firing rate. Specifically, it indicates the frequency components that are present in the neuron's activity.

A high power at a particular frequency in the power spectrum indicates that the neuron's firing rate is modulated at that frequency. For example, a peak in the power spectrum at 10

Hz indicates that the neuron's firing rate tends to oscillate at a frequency of 10 Hz. This could be indicative of a neural network that is involved in processing information related to a particular behavior or cognitive task that occurs at a 10 Hz frequency.

The power spectrum of the autocorrelation function is a useful tool for analyzing and understanding the rhythmic and periodic patterns of neural activity in the brain. It can provide insights into the functional organization of neural networks and the mechanisms underlying neural coding and information processing.

4.1.4.1 Welch's Method for Estimating PSD

For computing the PSD of the auto correlation, Welch's method [74] was used to reach a smooth estimation of PSD. The Welch method is a spectral analysis technique used to estimate the power spectral density (PSD) of a signal. This method involves dividing the signal into overlapping segments, applying a window function to each segment, and computing the Fourier transform (FT) of each segment. The resulting spectral estimates from each segment are then averaged together to obtain the final PSD estimate. The method is useful when dealing with non-stationary signals, where the PSD may vary over time. By using overlapping segments, the method can provide a more accurate estimate of the PSD over time than traditional methods that assume stationarity. For computing PSD using the Welch method (*Figure 4.6*), *scipy.signal.welch* was used. The input sampling rate was set as 1000Hz as the length of each bin to convert the spike train to 0,1 format was set at 1ms. Interestingly, as seen in Fig. 4-6, both GC and MC exhibit a peak in the theta (4-10 Hz) range when they are in the behavior state. This could suggest that there is a rhythmic pattern in the spike train that is occurring at a frequency within the theta range. Theta oscillations

are a common rhythmic pattern observed in the brain, particularly in the hippocampus, and are often associated with memory processes [9], [12], [75]. Therefore, if you observe a peak in the theta range of the PSD of the auto-correlation of a spike train, it may suggest that the neuron or network is involved in some sort of memory-related processing. Mentioned peak was happening for interneurons in all states.

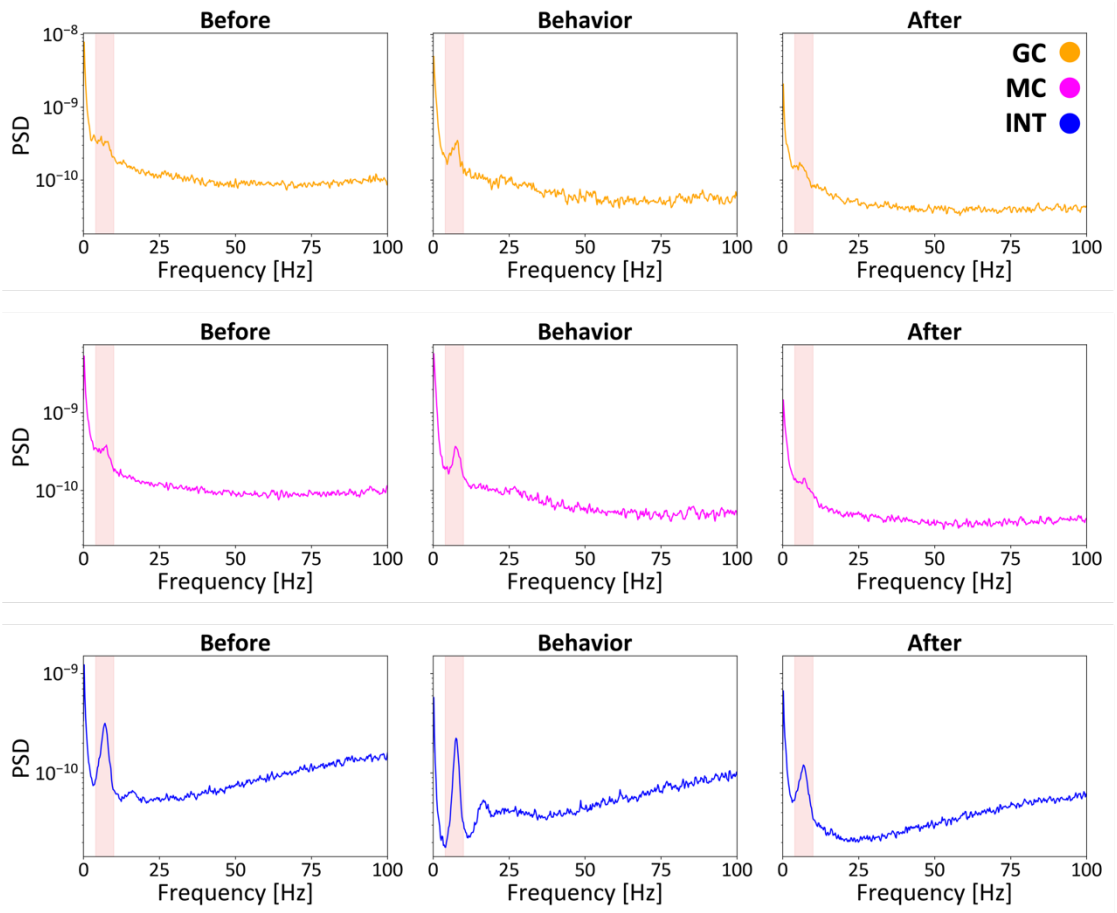

Figure 4.6 – PSD of autocorrelation function for GC, MC and an interneuron in three different states. The theta range (4-10 Hz) is highlighted by red.

4.1.5 ISI Serial Correlation

ISI serial correlation (SC) is a measure of the correlation between successive ISIs in the firing pattern of a neuron. It refers to the extent to which the duration of one ISI is related to the duration of the previous ISI.

ISI SC can be calculated using various methods, such as the autocorrelation function, the cross-correlation function, or the time-rescaling theorem. These methods allow researchers to quantify the degree of correlation between successive ISIs and to identify patterns or structures in the temporal dynamics of neural activity.

The presence of noticeable ISI SC indicates that the firing pattern of the neuron is not completely random, but is influenced by previous firing events. If the ISIs are positively correlated, meaning that a long ISI is likely to be followed by another long ISI and a short ISI is likely to be followed by another short ISI, this suggests that the neuron is exhibiting a burst firing pattern. If the ISIs are negatively correlated, meaning that a long ISI is likely to be followed by a short ISI and vice versa, this suggests that the neuron is exhibiting a regular firing pattern[76], [77].

ISI SC is an important aspect of the temporal structure of neural activity, as it provides information about the mechanisms underlying the generation and modulation of action potentials in individual neurons and neural networks. It is commonly used in the field of neuroscience to study the functional organization of neural networks, the coding of information in neural activity, and the effects of various stimuli or drugs on neuronal firing patterns.

For computing the ISI SC (*Figure 4.7*) the Chacron et al. [78]method was used. Let us denote by $\{Ij\}$ the ISI sequence with mean $< I >$ and variance $VAR(I)$. The serial correlation coefficients $(SCCs)$ ρ_j defined for lag j by:

$$\rho_j = \frac{< I_{j+k} I_k > - < I_k >^2}{VAR(I)} [78]$$

For a GC and MC, the ISI SC starts decreasing after the first lag (*Figure 4.7*). This can indicate that there is some degree of SC in the ISIs, but that the dependence on previous ISIs decreases quickly as the lag between ISIs increases. Specifically, a positive correlation at a lag of 1 means that the value of the current ISI is moderately correlated with the value of the previous ISI (lag 0), but the decrease of correlation at a lag of 10, suggesting that the influence of past ISIs on the current ISI rapidly diminishes as the lag increases. The only difference between GC and MC could be that MC ISI SC decreases at a higher rate in the behavior state compared to GC, which is difficult to interpret. The main difference between GC and MC may be that SC decreases more rapidly with increasing time lag for MC in the behavior state compared to GC. A faster decay of the serial correlations for one cell compared to another could suggest that the first cell has a shorter memory or refractory period than the second cell. This means that the first cell is less likely to fire again immediately after a spike than the second cell, and its spiking activity is less influenced by its own recent history.

For interneuron (Figure 4.7), SC increases in the first 5 lags and decreases in the 5 last lags. The initial increase in the serial correlation at short time lags indicates that the interneuron is more likely to fire again soon after a spike, which is consistent with a refractory period or other short-term dependence mechanism. The subsequent decrease in the serial

correlation at longer time lags suggests that the influence of past spikes on the interneuron's spiking behavior diminishes over time.

4.1.5.1 ISI Coefficient of Variations

The coefficient of variation (CV) is a commonly used measure in spike train analysis that quantifies the irregularity or variability of the ISIs of a neuron. It is defined as the ratio of the standard deviation of the ISIs to the mean ISI, expressed as a percentage. In mathematical terms, the coefficient of variation was calculated as:

$$CV = (standard\ deviation\ of\ ISIs\ /\ mean\ ISI)\ x\ 100\%$$

The coefficient of variation provides a way to compare the variability of spike trains across different experimental conditions or different neurons. A higher CV indicates that the spike train is more irregular, with more variable ISIs, while a lower CV indicates a more regular pattern of spiking[79], [80]. The coefficient of variation is a useful measure because it is independent of the mean firing rate of the neuron, which can vary widely across neurons or experimental conditions. It is also relatively easy to calculate and interpret, making it a popular choice in spike train analysis.

Although the CV for GC and MC (*Figure 4.7*) does not show much variation during different states, interneurons exhibit a significant decrease in CV (<1) during behavior, indicating a more regular pattern of spiking. The reason behind this behaviour could be changes happening in the input it receives from other neurons in the network during a pattern separation.

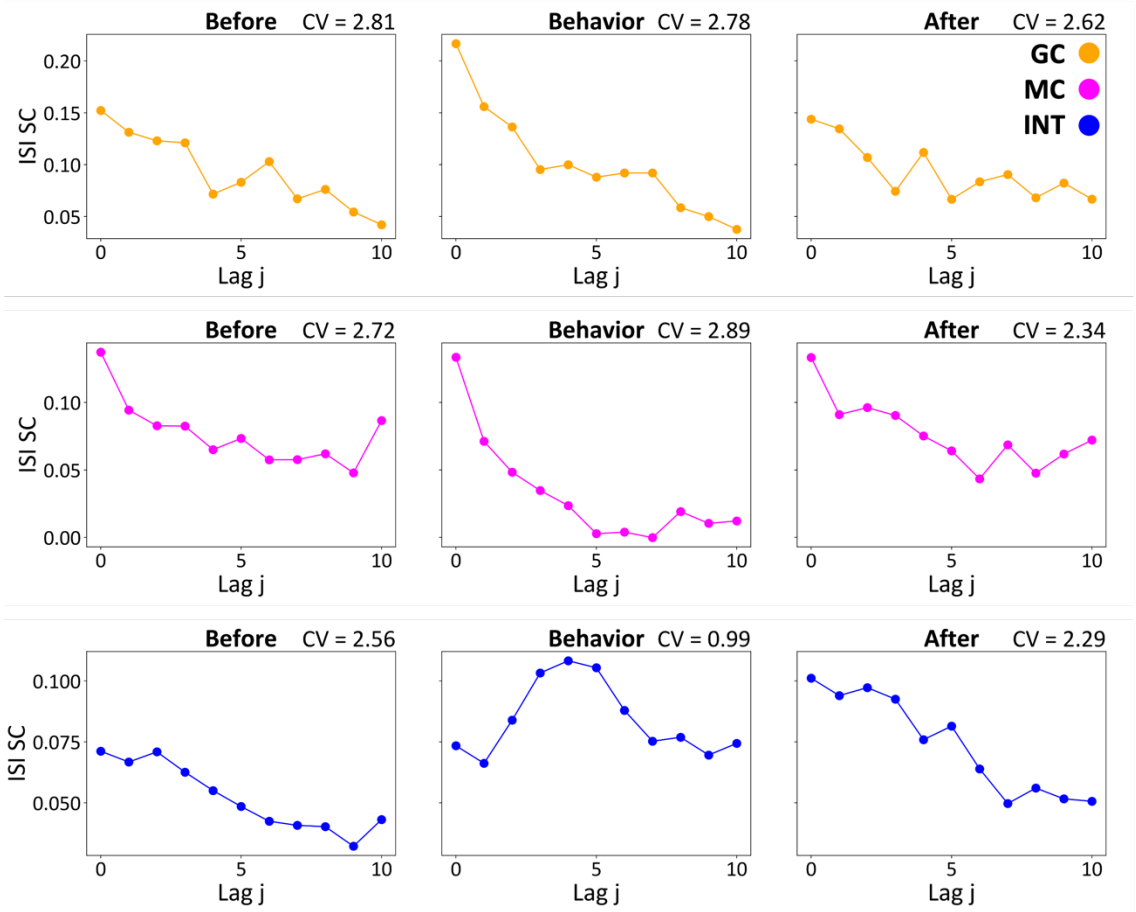

Figure 4.7 - Serial correlation of the ISIs for GC, MC and an interneuron in three different states. The Coefficient of variations is indicated on top right of each plot.

4.2 Behavioral Analysis

Before beginning the behavioral analysis, some guidelines and notes are essential to mention. The behavioral phase was explained more in detail in the section 2.1.1. However, more information about the number of trials and images was not provided, which will be discussed here. In total, 127 trials on 5 different images (V0, V8, V12, V18, V36) were performed on one day of recording. The accuracy of the animal on this day was 62.9%. For synchronization of the animal behavior and recording, TTL pulses are sent from the touch screen to the recording board on specific events. It is important to mention a few guidelines and notes for behavioral analysis. In *Figure 2.1B*, different states during each trial are

depicted with different colors, which are unified throughout the whole analysis. As these colors are critical, a summary (*Figure 4.8*) is provided below:

Start: When the animal pokes the food tray (*Figure 2.1A* left corner)

Image: When the animal touches the white screen in the middle, an image appears immediately afterward.

Action: After the animal has made a decision, it indicates its choice by poking the left or right screen. However, it's likely that the decision was made a few seconds prior to the actual screen poke, as the animal may struggle to physically touch the screen

Reward: When the animal was correct, a reward (in the form of a milkshake) was placed in the food tray, prompting the animal to approach and drink it.

Red: An incorrect response from the animal resulted in a 5-second light punishment being administered.

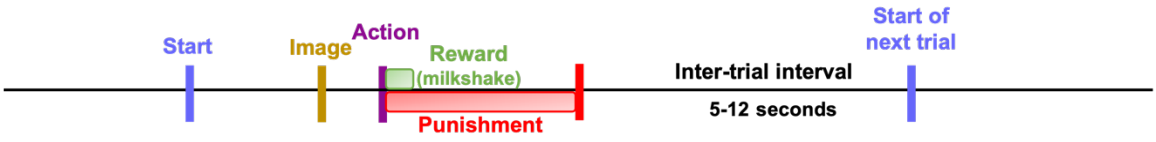

Figure 4.8 - A summary of the tasks that the animal does in one trial.

4.2.1 Raster Plots

A raster plot is a simple method to visually examine the trial-by-trial variability of the response. You can examine what features these responses have in common by averaging overall responses to create a peri-stimulus time histogram.

Raster plots are easy to plot and compare across multiple trials if the length of trials is the same. However, in this experiment, the biggest problem is the variability of the length of each trial and the timing of different events. In order to see responses to different events, trials were aligned through specific events. An example of a raster plot can be seen in *Figure 4.9*. Each line represents one trial. The lines with gray background color are incorrect trials and the lines with a white background are correct ones. When the animal is wrong on an image, the image will appear again till the animal make the right decision which is defined by correction trials (light blue on the right corner). The spikes in each line have different colors which are based on the represented image on that trial.

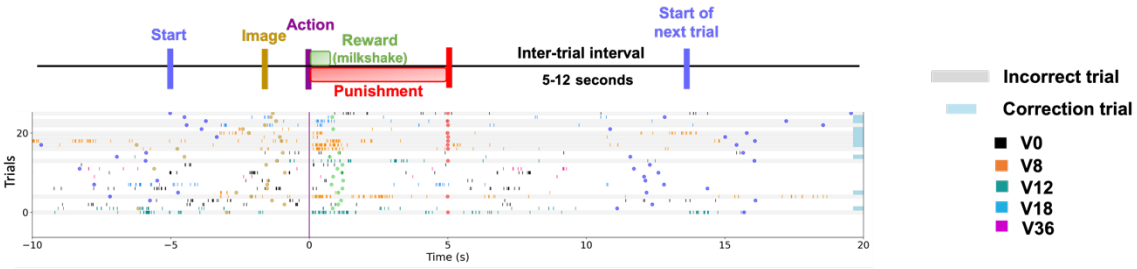

Figure 4.9 - A raster plot example of the animal with behavioral tasks performed.

4.2.1.1 Raster Plot Analysis

After inspecting the raster plots of different units, interesting results were discovered. In the first extracted unit (classified as GC), a burst of activity was initiated when the animal made an incorrect decision (*Figure 4.10*). Moreover, this result was consistent with just the GC, not the MCs or interneurons. This raises the possibility that an error signal is sent to the hilus or hippocampus when the animal makes an incorrect decision in a pattern separation task. MCs (*Figure 4.11*) remained quiet and observed no increase or decrease in their activity. For interneurons (*Figure 4.12*), there is a significant reduction in spike

activity after the animal has made the correct decision and is consuming the reward. This reduction was consistent among all interneurons. In order to check how these alternations are consistent and significant, more spike train analysis was performed in the sections below.

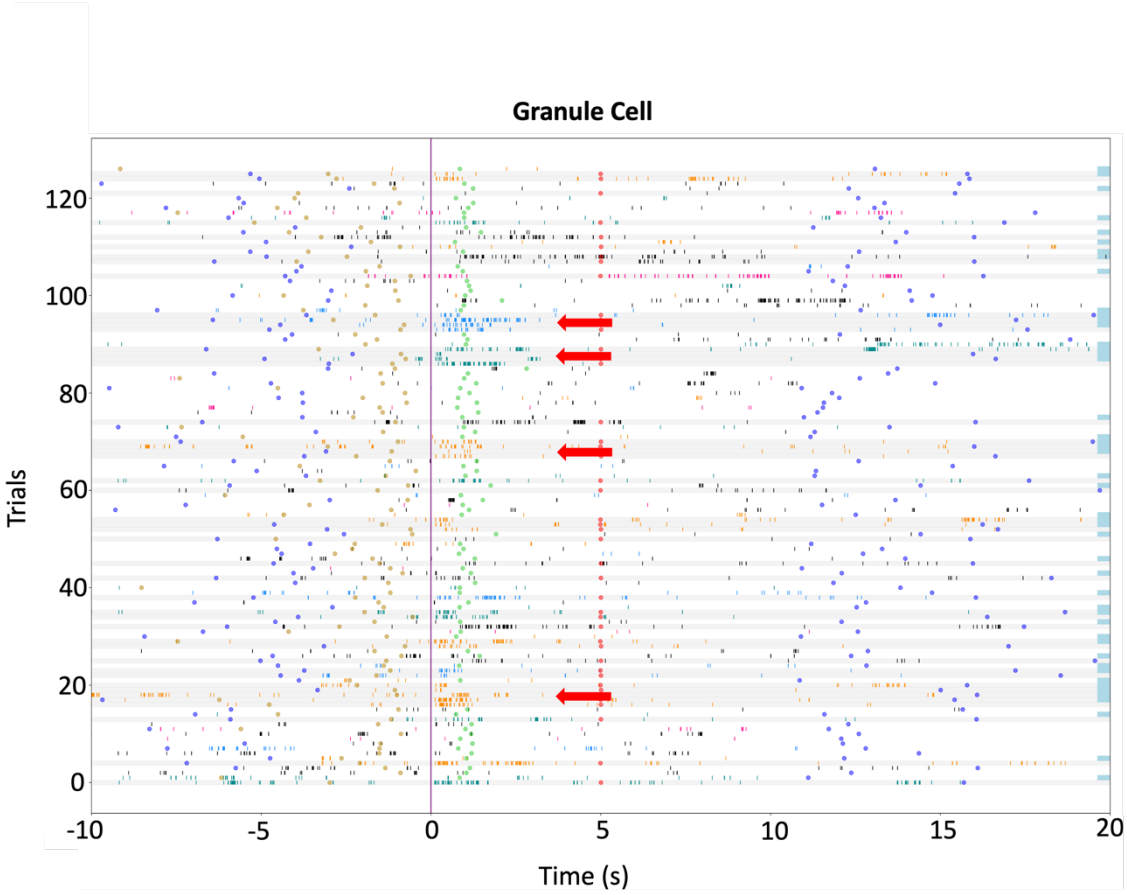

Figure 4.10 - Raster plot of a GC in the behavior. The bursts after an incorrect decision indicated by a red arrow.

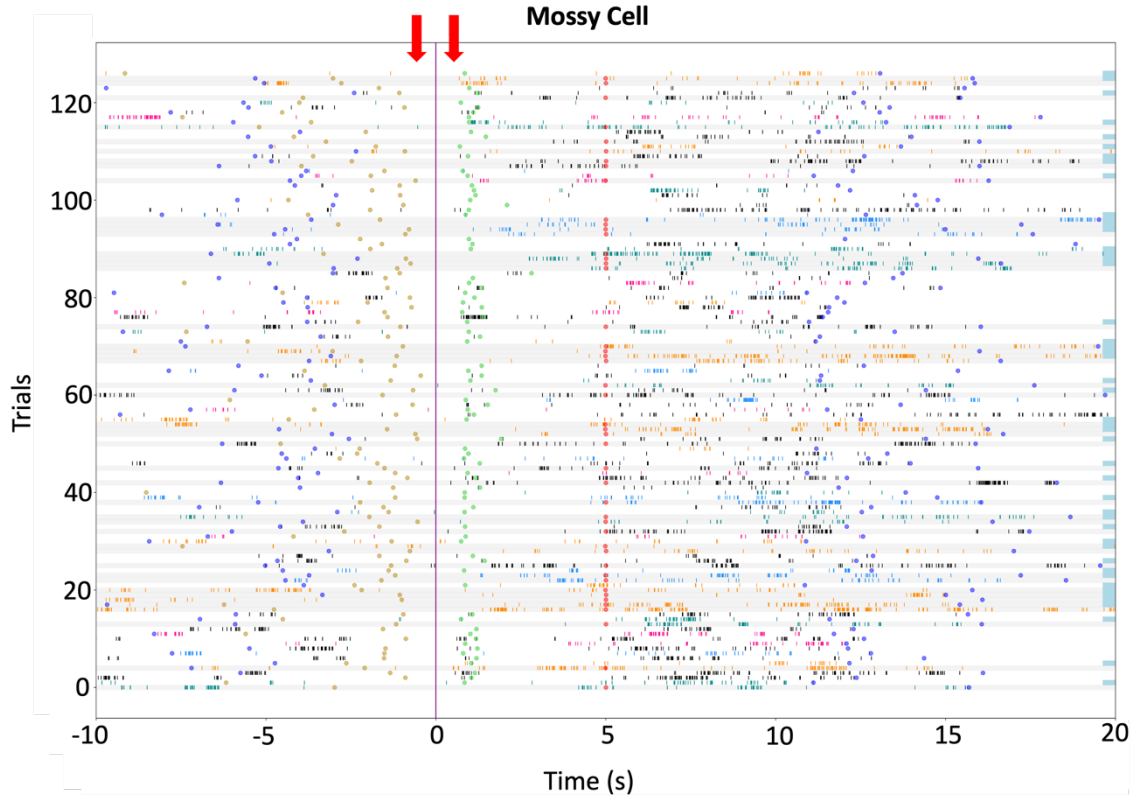

Figure 4.11 - Raster plot of a MC in the behavior. No specific neural activity observed before or after the action indicated by a red arrow.

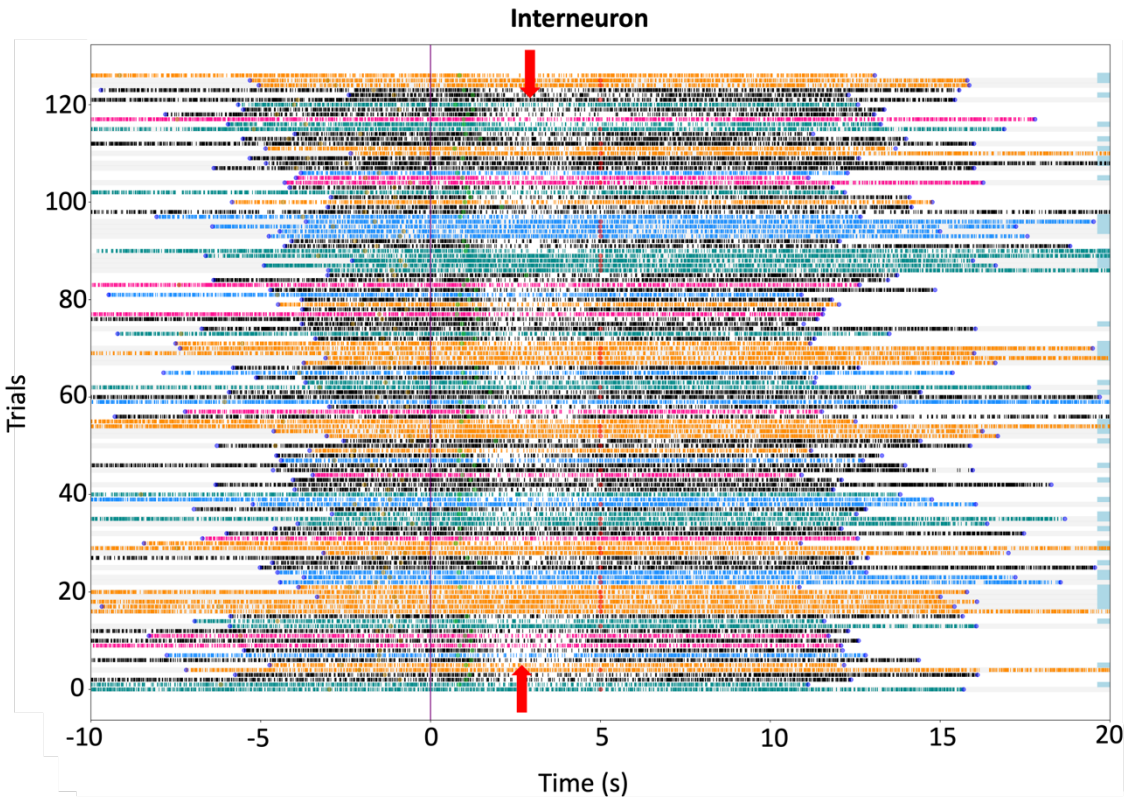

Figure 4.12 - Raster plot of an interneuron. The decrease in neural activity while consuming the reward indicated by a red arrow.

4.2.2 Peristimulus Time Histogram (PSTH)

In neurophysiology, the peri-stimulus time histogram and post-stimulus time histogram, abbreviated PSTH or PST histogram respectively, are histograms of the times at which neurons fire. It is also sometimes called a pre-event time histogram or PETH. These histograms are used to visualize the rate and timing of neuronal spike discharges in relation to an external stimulus or event. The prefix peri, is typically used in the case of periodic stimuli, in which case the PSTH shows neuron firing times wrapped around one cycle of the stimulus. To make a PSTH, a spike train of a single neuron is aligned with an identical

stimulus's onset, or a fixed phase point repeatedly presented to an animal. The aligned sequences are superimposed over time and then combined to construct a histogram.

4.2.2.1 Computing PSTH

In a typical PSTH, the length of all trials is the same. So, for computing PSTH, the bin size should first be defined. Afterward, using *matplotlib.hist* the number of spikes in each bin is calculated and plotted. Moreover, if the firing rate is desired, instead of reporting the number of spikes, it will be divided by the bin size. However, when the length of each trial is different, this would not be a proper way to compute PSTH. Imagine starting at time -10 on the raster plot (*Figure 4.9*) and counting the number of spikes in each 200 ms bin. In the first bin, there will be only one trial, and in the next bins, this will increase till the time reaches close to zero when lots of trials are available. Hence, when the histogram gets close to time zero, there is an increase in the number of spikes in common across all trials, which is fake (*Figure 4.13B*, Standard PSTH).

Normalized PSTH

After counting the number of spikes in each bin, to prevent bias caused by the number of trials, the number of spikes is divided by the number of trials that spikes were counted from (*Figure 4.13B*, Normalized PSTH).

Example 1: First bin, 5 spikes across 2 trials: 5/2 is reported as the normalized spike count (*Figure 4.13A*).

Example 2: 10th bin, 150 spikes across 124 trials: 150/124 is reported as a normalized spike count (*Figure 4.13A*).

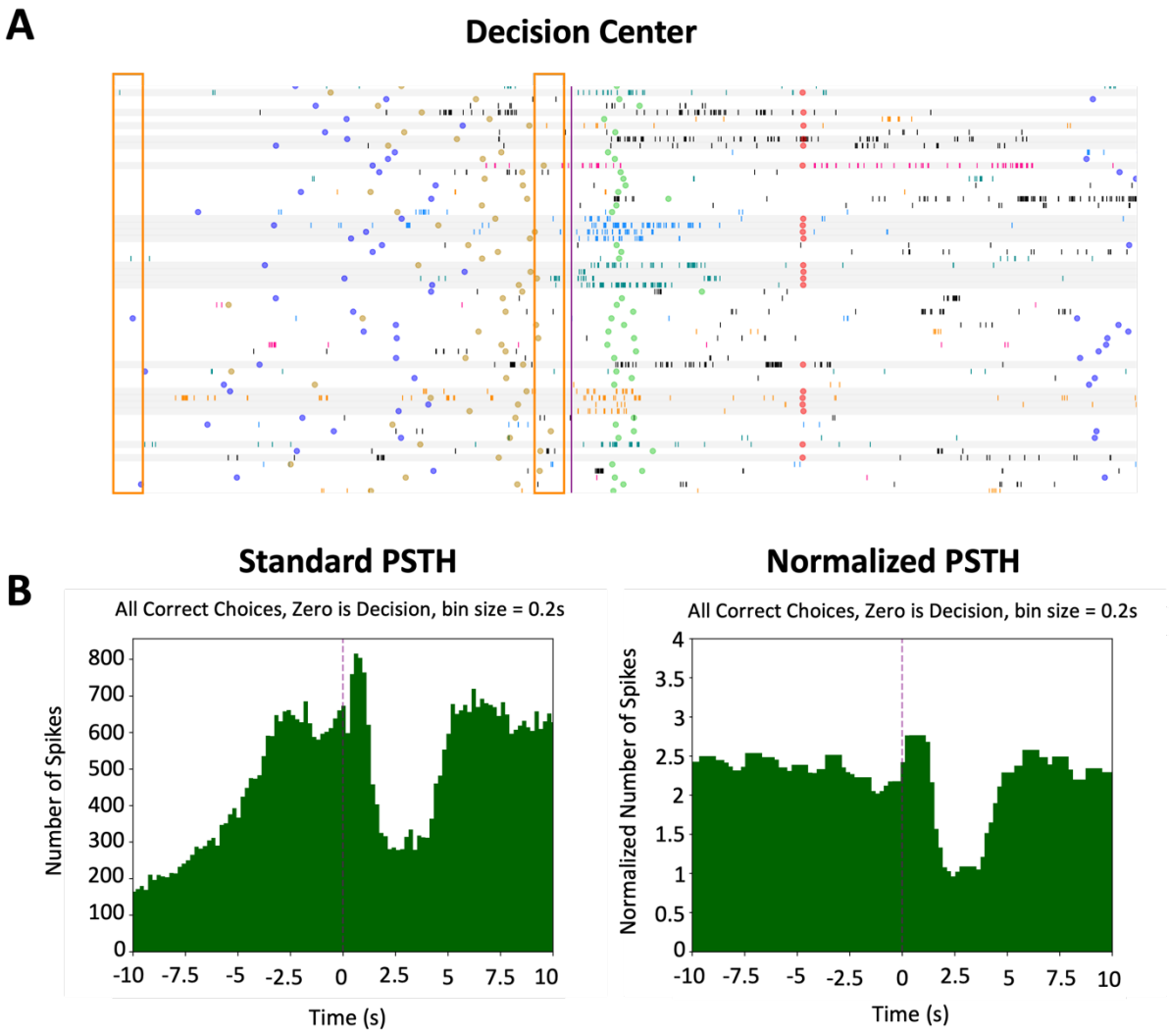

Figure 4.13 -PSTH analysis overview. A) Example of a computing the number of spikes for each bin in PSTH. B) Comparison of Standard and Normalized PSTH.

4.2.2.2 Analysis of PSTH

As mentioned in the section 4.2.1, the graphs can be centered on different events (image, decision, reward, punishment). However, the results that are represented are trustable for a few seconds before and after the center line. Hence, a color bar at the top indicates the number of trials in common.

The PSTH was computed for each unit through different centers and for both correct and incorrect trials (green and red). As mentioned in the previous section, for GC a burst of activity was observed in incorrect trials. This increase in activity is more apparent in the PSTH when it is centered on the action (*Figure 4.14C*). However, in the correct trials, the spike activity increased slightly before and after the action which is not significant. For interneurons, in the correct trials, there is a reduction in spike activity (*Figure 4.15A, D*).

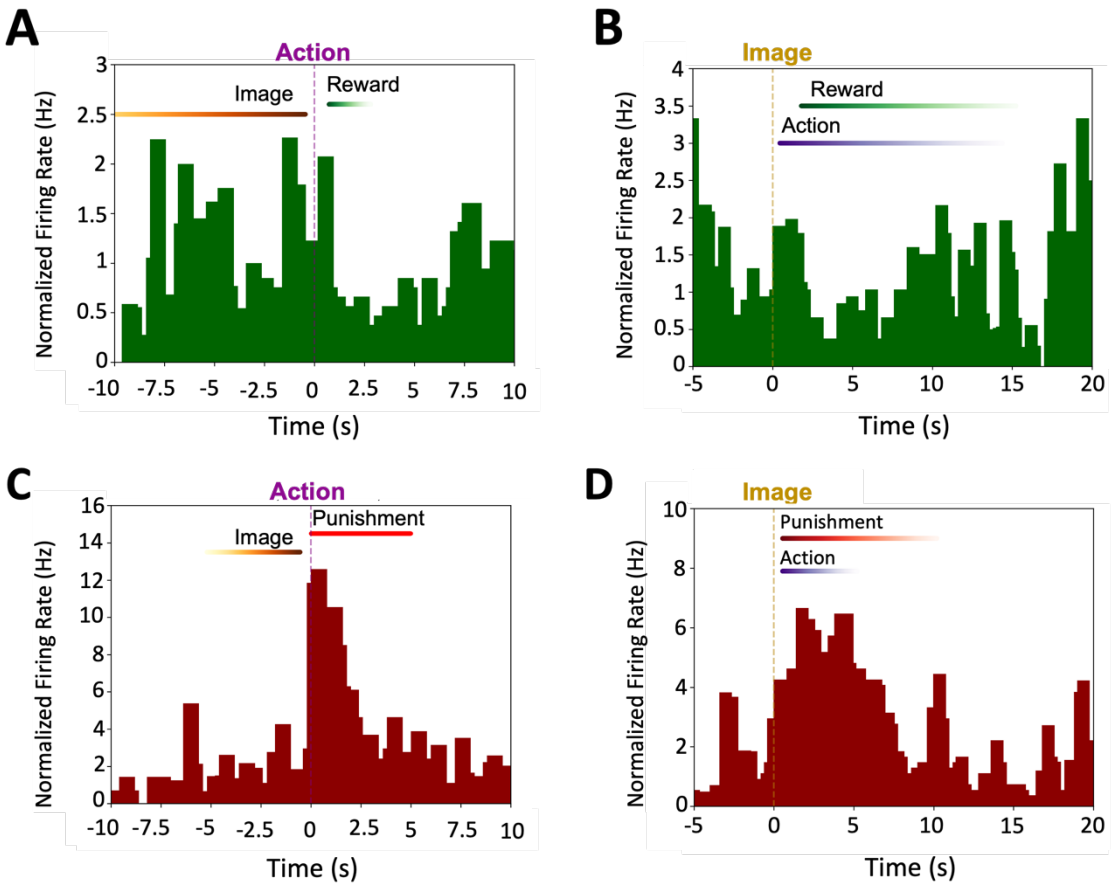

Figure 4.14 - Normalized PSTH for a GC. A) Correct trials, action centered. B) Correct trials, image centered. C) Incorrect trials, action centered. D) Incorrect trials, image centered.

This reduction is more significant when it is centered on reward consumption (*Figure 4.15C*).

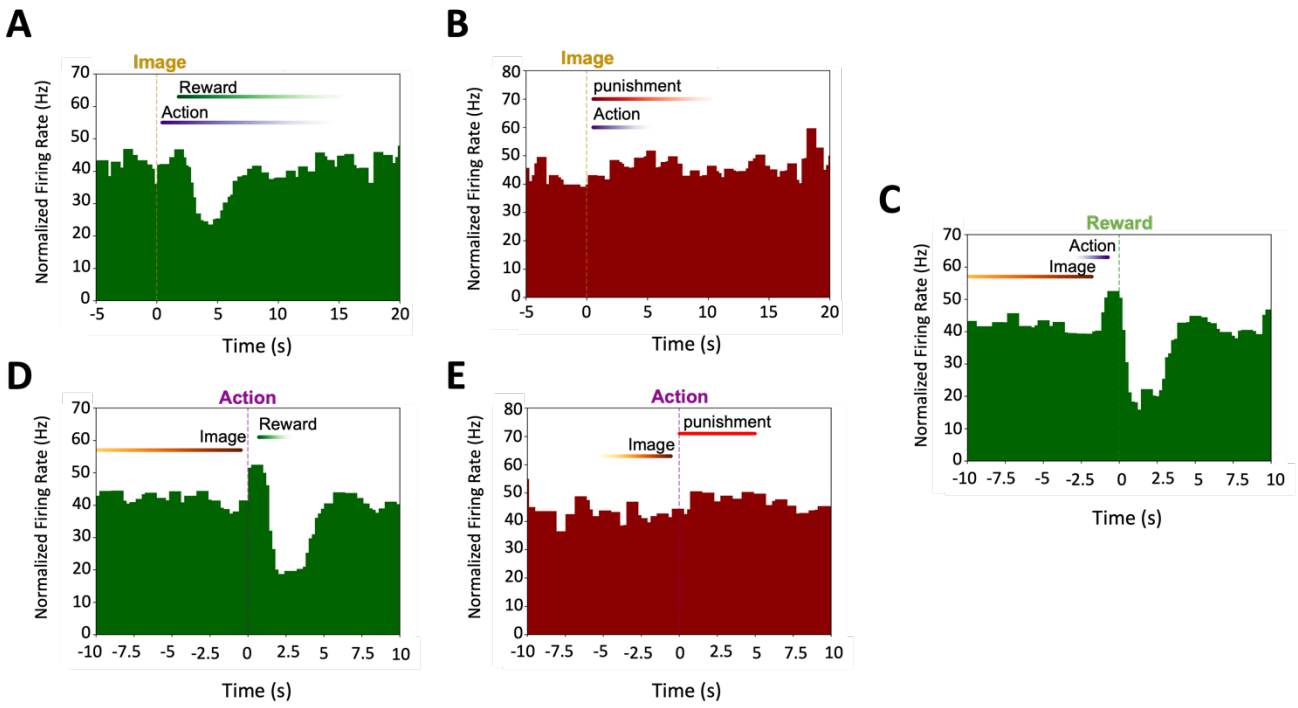

Figure 4.15 - Normalized PSTH for an interneuron. A) Correct trials, image centered. B) Incorrect trials, image centered. C) Correct trials, reward center. D) Correct trials, action centered. E) Incorrect trials, action centred.

4.2.2.3 Image Selectivity Responses

In order to check the image selectivity response of each unit, the PSTH was computed for V0 in comparison with different images (V#). For all units, there was not any significant increase or decrease in spike activity while comparing different images. However, for GC increase in spike activity, while the animal makes an incorrect decision, the increase in the spike activity has a delay for V0 compared to V# (Figure 4.16B, D). This was noted

as a delay to the increase in spike activity which was computed using causal filters in further sections.

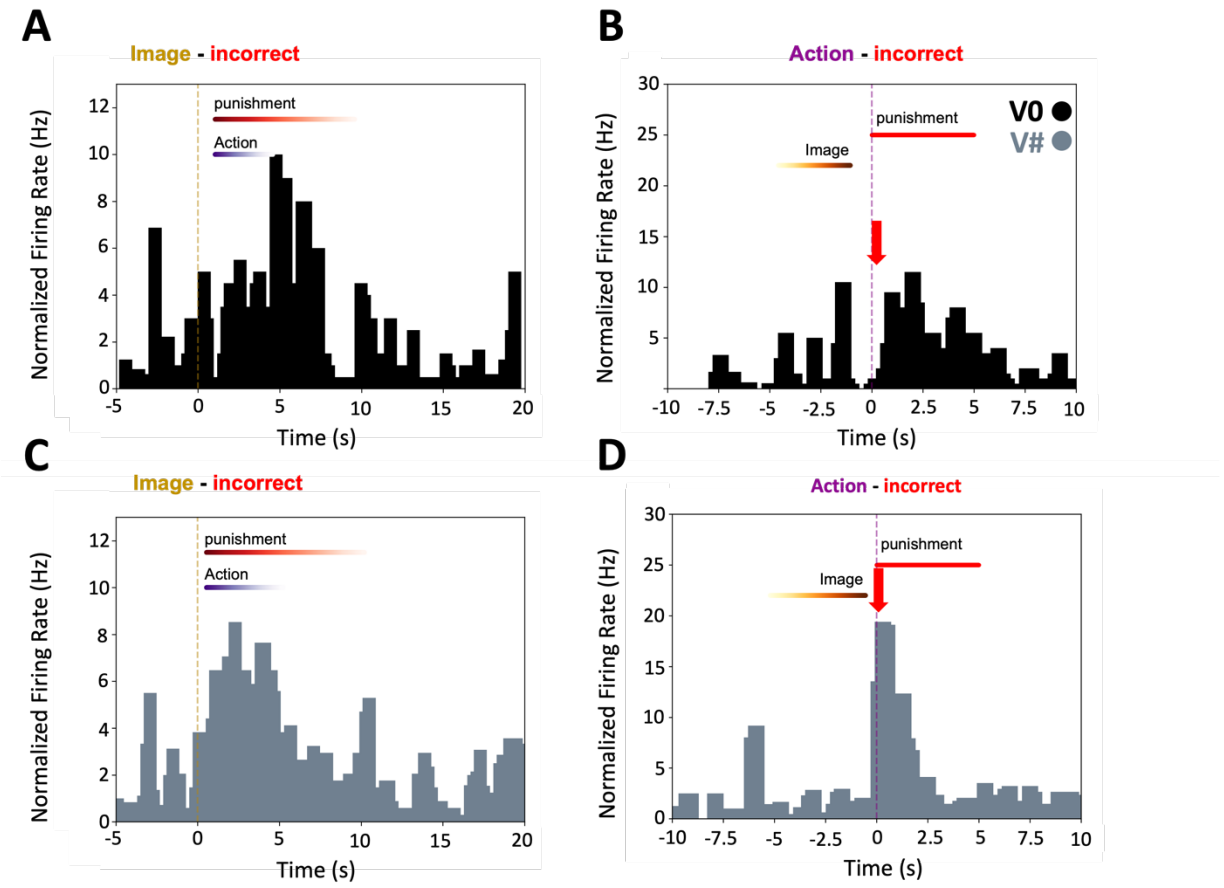

Figure 4.16 - Image selectivity responses of a GC. A) Incorrect trials, image centered for V0 image. B) incorrect trials, action centered for V0 image. C) Incorrect trials, image centered for all images except V0. D) Incorrect trials, action centered for all images except V0.

4.2.3 Firing Rate Across Time, Firing Rate, and Peak Firing Rate

The firing rate of a neuron refers to the number of times the neuron fires action potentials (or “spikes”) per second. This firing rate is an important indicator of the activity level of the neuron and can provide information about its role in the brain.

A high firing rate typically indicates that the neuron is active and is receiving a strong input from other neurons or sensory stimuli. In contrast, a low firing rate may suggest that the neuron is relatively inactive or inhibited. The firing rate for each unit was computed using two methods: Firing Rate Across Time and Firing Rate in Behavior.

4.2.3.1 Firing Rate Across Time

Comparing different states:

In this method, the firing rate was computed for each state mentioned in section 2.1.2. The firing rate was plotted as a 1D signal which shows the changes in three states through time and Hz (*Figure 4.17*). Comparing firing rates among different unit types, for MC and interneuron, we found that there was an increase in firing rate when the animal was in the behavior phase of the experiment. However, the activity of GC remains stable and without significant alternations among different phases.

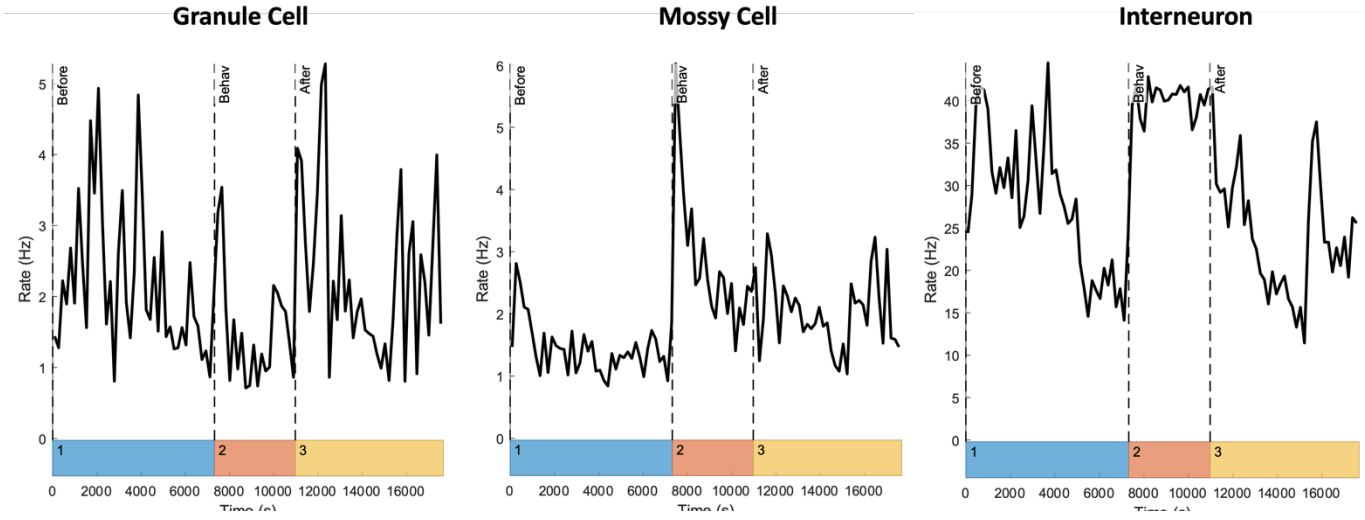

Figure 4.17 - Firing rate activity of GC, MC and Interneuron for 3 different states (1. In the home cage before experiment, 2. In the behavior, 3. In the home cage after the experiment).

4.2.3.2 Firing Rate in Behavior

Comparing trials:

For each trial, 6 mentioned segments were separated as below:

Baseline: 5 seconds before the blue dot (beginning of the trial).

Start to Image: The time it takes for the animal to reach and touch the screen from the food tray.

Decision: 1 second after the animal has seen the image, which is believed to be the decision-making time.

Decision to action: The time it takes for the animal to touch the screen for a decision (struggle time).

Feedback: 5 seconds after the decision on which animal will receive the reward or punishment.

After reward: 4 seconds after the animal has received the reward.

The firing rate was calculated for each segment using the standard method (the number of spikes/time in Hz). Hence, for each segment, a distribution of firing rates is achieved.

For visualizing the data, a box plot with a scatter plot on top was used for each segment (*Figure 4.18*). A box plot is a graphical rendition of statistical data based on the minimum, first quartile, median, third quartile, and maximum. The term “box plot” comes from the graph looking like a rectangle with lines extending from the top and bottom. The first quartile, third quartile, and median were indicated using black lines. Moreover, the variability of the data can be seen by the scatter plot.

4.2.3.2.1 Image Selectivity Responses

As mentioned in the section 4.2.1.1, an increase in spike activity of GC was observed when the animal was incorrect. In *Figure 4.18*, the firing rate for different segments was plotted for each image to observe how this increase in firing rate is consistent among different images for a GC. From the segment Feedback (5s after action), it can be seen that the firing rate of the GC increases during the incorrect response when the number of vertical lines increases, or in other words, the image becomes more complicated for the animal. However, the decrease in firing rate while receiving the rewards does not show any image selectivity responses for interneurons (*Figure 4.19*), and it is consistent over all images.

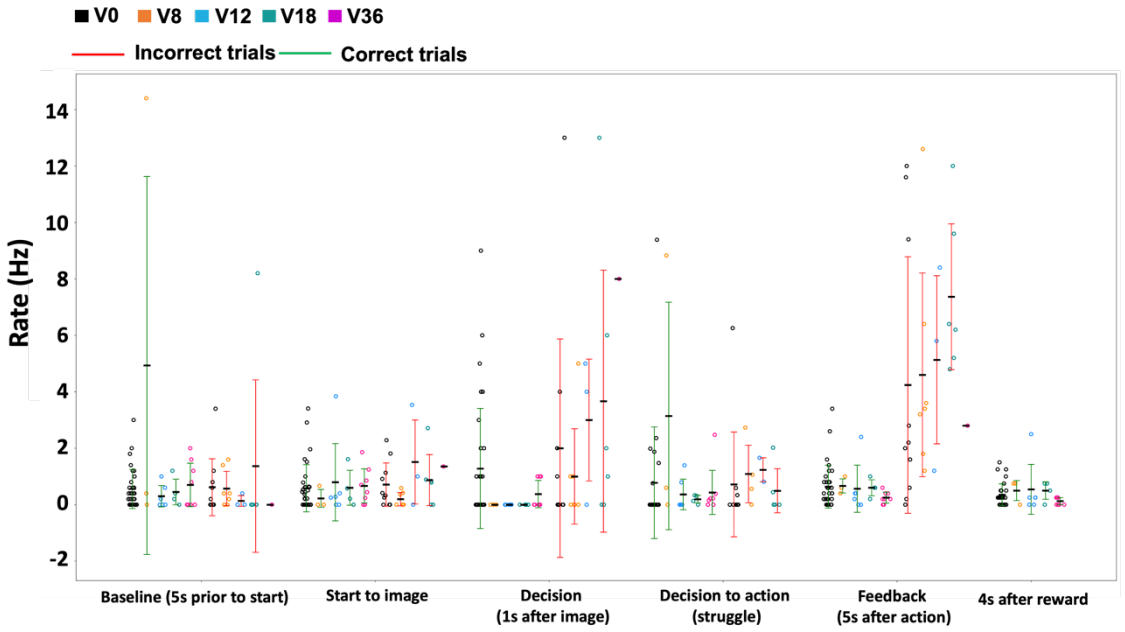

Figure 4.18 - Firing rate box plots of a GC for 6 segments.

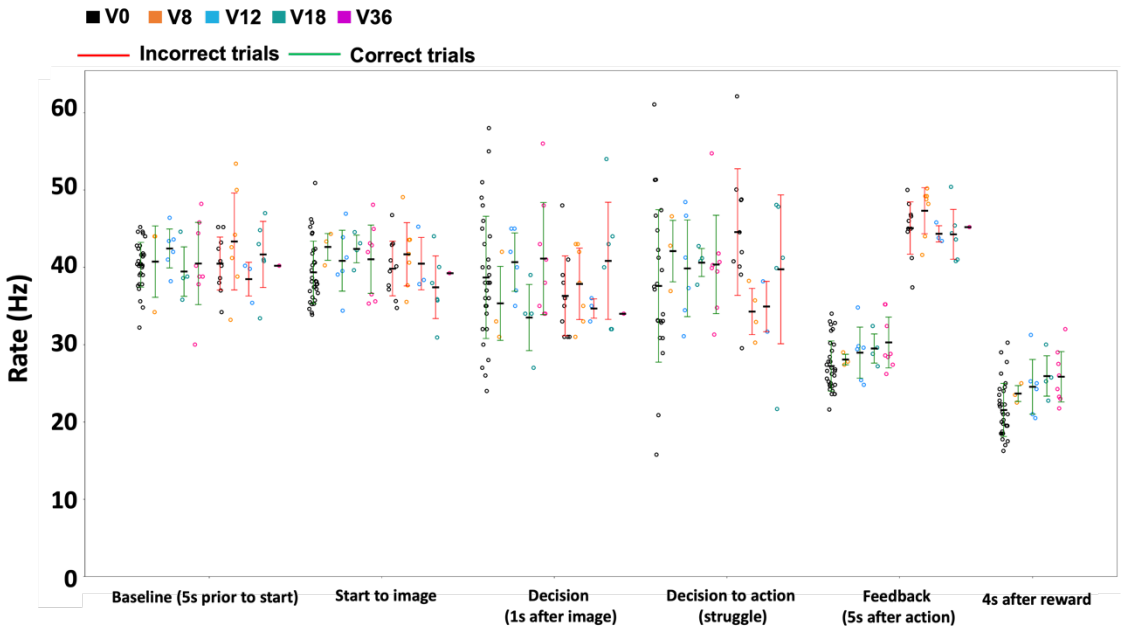

Figure 4.19 - Firing rate box plots of an interneuron for 6 segments.

4.2.3.3 Peak Firing Rate

Another important metric that gives a better understanding of the firing behavior of a neuron is the peak firing rate. The shortest ISI is the time between two consecutive action potentials, and it can provide an estimate of the fastest rate at which the neuron can fire. In this method instead of using the classic formula (number of spikes/time), the shortest ISI (section 4.1.1) was computed and then reversed (1/minimum ISI) to get estimate of the peak firing rate that neuron. The procedure of computing was the same as mentioned for the firing rate (6 segments, *Figures 4.20,4.21*). The image selectivity is roughly visible in the GC error signal, and for the interneuron no difference was observed comparing to firing rate plot.

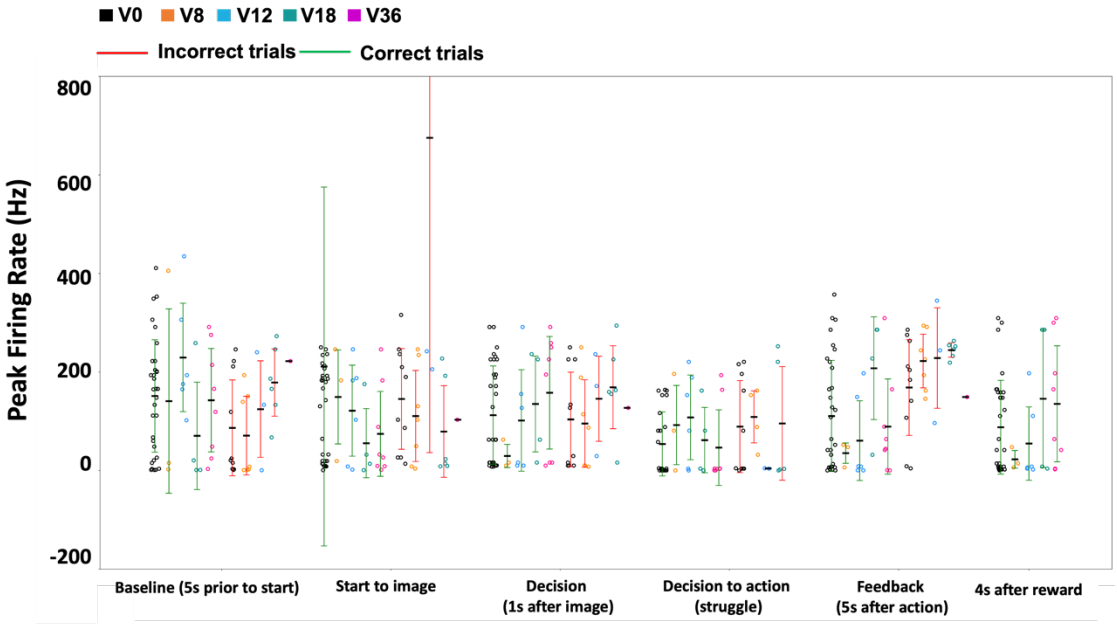

Figure 4.20 - Peak firing rate box plots of a GC for 6 segments.

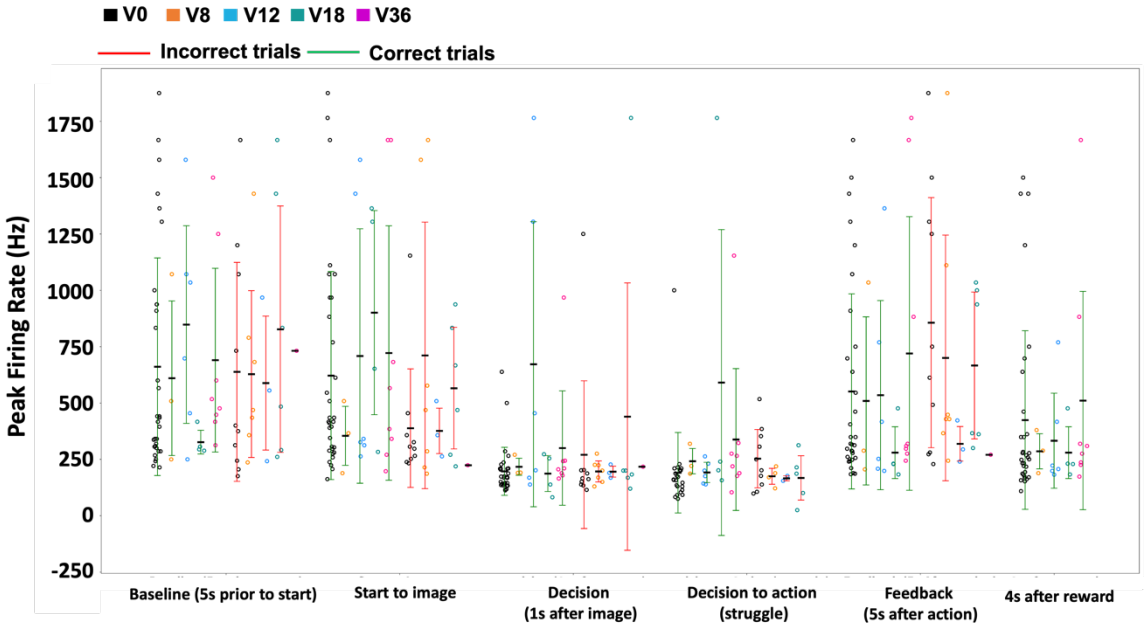

Figure 4.21 - Peak firing rate box plots of an interneuron for 6 segments.

4.2.4 Causal Filters and Spike Train Kernels

Estimating the firing rate as a continuous-time function from a single trial is a challenging task since the underlying process provides only a sparse representation of the spike data. The PSTHs were computed in the section 4.2.2, which shows a coarse estimate of the firing rate. To produce a smooth estimate of the firing rate, different Kernel Smooth (causal filter) methods have been proposed. The main idea behind estimating firing rates using different kernels (Gaussian, Exponential) is as follows:

Let $t_1, t_2, \ldots, t_n$ be a sequence of spike times (i.e. a spike train) which can be expressed mathematically as:

$$p(t) = \sum_{i=1}^{n} \delta(t - t_i) \sum_{i=1}^{n} \delta(t - t_i) [81]$$

where $\delta(t)$ is Dirac function and n is the total number of spikes. The underlying rate function also known as firing rate, $\lambda(t)$, can be estimated by using kernel smoothing, a method which convolves the spike train with a kernel function $K(t)$ as follows,

$$\hat{\lambda}(t) = \int_{-\infty}^{\infty} K(\tau)\, \rho(t-\tau) d\tau \int_{-\infty}^{\infty} K(\tau)\, \rho(t-\tau) d\tau \; [81]$$

Eq (2) can also be represented as the sum over kernel functions centered at spike times t_i,

$$\hat{\lambda}(t) = \sum_{i=1}^{n} K(t - t_i) \sum_{i=1}^{n} K(t - t_i) \text{ [81]}$$

The effectiveness of kernel smoothing technique depends on the choice of a kernel function and the selection of a smoothing parameter (i.e., bandwidth). A kernel function is required to be a non-negative, normalized to a unit area, having a zero first moment and a finite variance[82], [83].

4.2.4.1 Computing Delay

In the section 4.2.2.3, it was mentioned that a delay was observed on the V0 image on GC error signal (increase in firing rate). The purpose of this section is to compute the delay of V0 compared to other images using causal filters and instantaneous firing rates. To compute the delay three main steps are performed and are described below:

1. In order to see that the increase or decrease in a firing rate is significant or not, statistical inferences from the firing rate of a neuron when it was in rest mode are

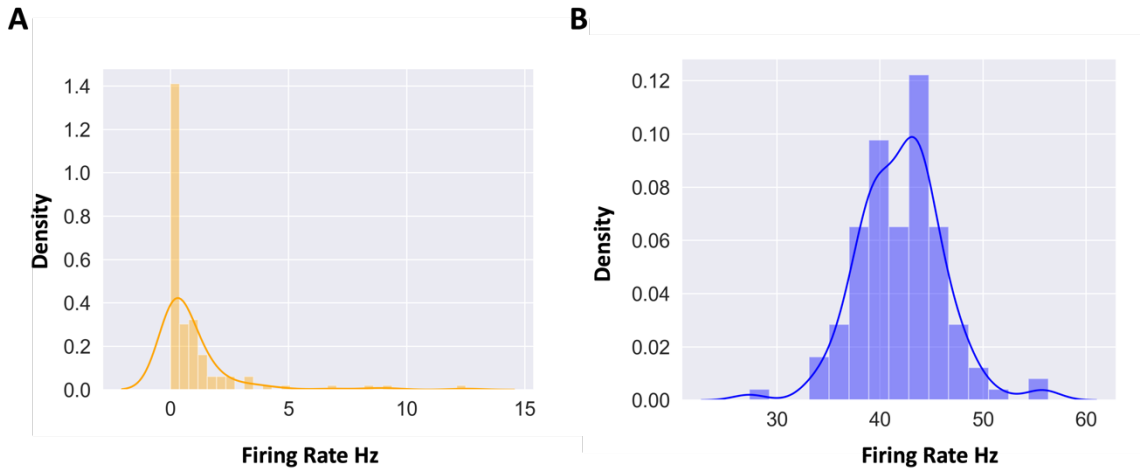

Figure 4.22 – GC and Interneuron firing rate distribution. A) Firing rate distribution of a GC for all trials 6 second before poking the food tray. B) Firing rate distribution of an interneuron for all trials 6 second before poking the food tray.

essential. Hence, for all units 6 seconds before the animal poked the food tray (start of the trial) was separated and then the firing rate was calculated for all trials (126). A distribution of firing rate of the units (excitatory or inhibitory) was achieved in *Figure 4.22*. Interestingly, the excitatory cells (GC, MCs) had a gaussian distribution while interneurons had an exponential distribution. From the GC firing rate distribution mean and standard deviation (std) was computed for future steps.

2. For computing causal filters, the Elephant[11] library in Python was used. Elephant is a popular open-source library for analyzing electrophysiological data in Python. The library provides a comprehensive set of tools and methods for processing and analyzing various types of neural signals, such as spike trains, local field potentials (LFPs), and extracellular field potentials. The *instantaneous_rate* function in the Elephant library is a useful tool for estimating the instantaneous firing rate of a neuron from its spike train. The function takes a spike train as input and returns a time series of the instantaneous firing rate estimated at each time point. The instantaneous firing rate is a measure of the firing activity of a neuron at a particular moment in time and is often used to analyze the temporal dynamics of neural activity. The *instantaneous_rate* function in Elephant library provides several options for computing the instantaneous firing rate, including the kernel density estimator (KDE), the Wiener filter, and the adaptive kernel estimator. These methods use different algorithms to estimate the firing rate, each with its own advantages and limitations. One important input to this function for selecting desired kernels is *sigma* which is a parameter that controls the bandwidth of the

[11] https://elephant.readthedocs.io/en/latest/modules.html

kernel used in the kernel density estimator (KDE) method for estimating the firing rate from the spike train. The optimal value of *sigma* depends on the characteristics of the spike train and the experimental context. Based on trial-and-error, the *sigma=500* and exponential kernels were used for estimating the firing rate of single trial (*Figure 4.23*).

3. From Action (0 second), if the firing rate was increased more than the mean + 2 * std of the firing distribution calculated in step 2, the duration would be saved as delay time. This was performed for all incorrect trials for a GC across all images. As expected, the delay for V0 is interestingly higher than for other images (*Figure 4.24*).

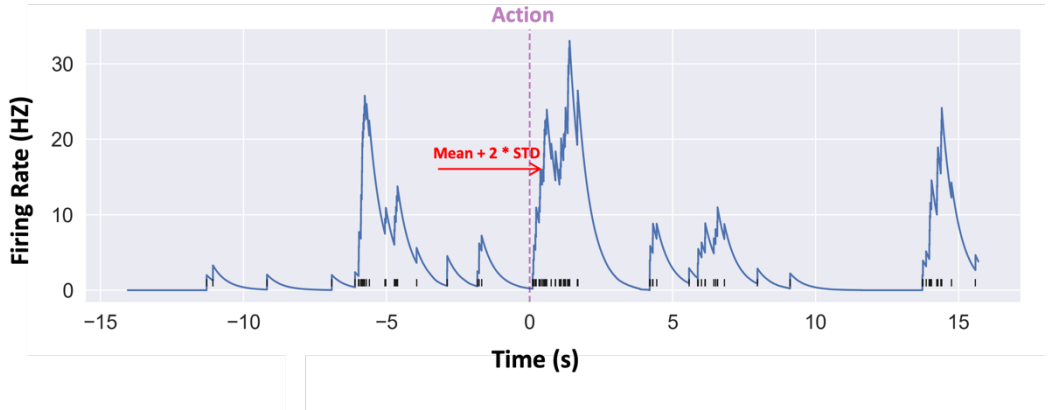

Figure 4.23 Estimated firing rate for an action centered trial of a GC using exponential causal filters. The mean + 2 * SD was used to find the time of the increase of firing rate reported as delay time.

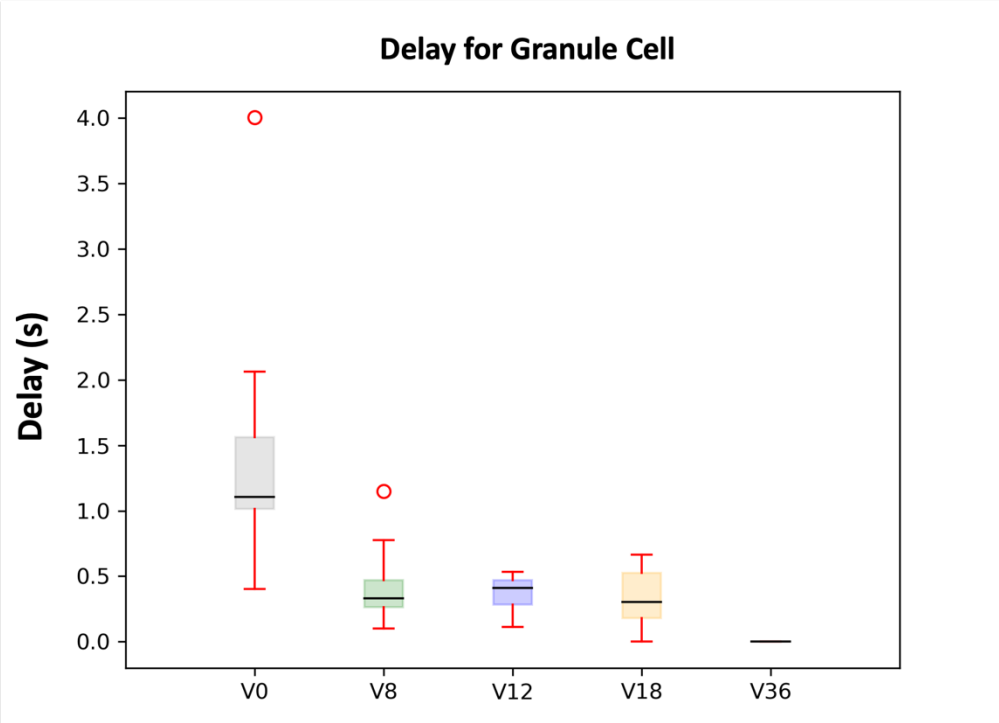

Figure 4.24 - Computed delay for different images using causal filters.

Chapter 5– Conclusion and Future Work

In this thesis, we propose a unified protocol for isolating the spike activity of neurons from electrophysiology recordings using different toolboxes. While various spike sorters and pre- and post-processing toolboxes have been published in the neuroscience community, the proper method for their use and how to connect the output of one toolbox to the input of another toolbox have not been thoroughly discussed. Furthermore, we separate different types of neurons, specifically MCs and GCs, using a method proposed by Senzai et al., and ensure that the classifier input is compatible with previous outputs. Subsequently, we apply various spike train analysis methods to investigate the role of these cells in a pattern separation task. All the analysis is customized to the experimental design, and the spike train analysis methods are adapted accordingly.

The proposed protocol consists of four main phases: Spike Sorting, Manual Curation, Separation of Excitatory and Inhibitory Cells, and Separation of GCs and MCs from Excitatory Cells. For spike sorting, the performance of different sorters was compared [19], and based on the results Kilosort, being the most trustworthy and commonly used sorter, was employed. For manual curation, Phy was used, and the input files required for this toolbox were discussed from Kilosort output. The separation of excitatory and inhibitory cells was performed using Cell Explorer, which also outputs various features. The entire post-spike train analysis was performed on the output format of this toolbox. The separation of GCs and MCs from excitatory cells was performed using a method proposed

by Senzai et al. [68], with the inputs of the model customized to match the experimental design and Cell Explorer output format.

The spike train analysis consists of two main phases: Comparing Different States and Behavioral Analysis. In the Comparing Different States phase, the neural activity of each neuron was analyzed in three main states (section 2.1.2). Various spike metrics, such as ISI, Return Maps, Auto Correlation, and ISI Serial Correlation, were computed and compared among the mentioned states. In the Behavioral Analysis phase, the activity of each neuron was analyzed while performing a pattern separation task across multiple trials. Spike train analysis methods, such as raster plots, PSTH, and firing rates, were computed and customized to match the experimental design.

In our future work, we plan to expand our study by increasing the number of animals and making adjustments to the behavioral task. Additionally, we will be using new electrodes of higher quality, which are double shanked to allow for recording from a larger area in the hilus. The increased number of samples (animals) will pose challenges, but it will enable us to thoroughly test the effectiveness of the proposed protocol on diverse datasets.

During the spike train analysis, we observed specific responses among different units, such as error signals in GCs and reduction of activity during reward consumption in interneurons. In future experiments, we aim to investigate the consistency of these results across different animals and explore if more prominent image selectivity responses can be discovered. This will provide us with a deeper understanding of the characteristics of GCs and MCs in the hippocampus and their role in pattern separation.

References

[1] H. Eichenbaum and N. J. Cohen, 'Can We Reconcile the Declarative Memory and Spatial Navigation Views on Hippocampal Function?', *Neuron*, vol. 83, no. 4. 2014. doi: 10.1016/j.neuron.2014.07.032.

[2] N. Burgess, E. A. Maguire, and J. O'Keefe, 'The human hippocampus and spatial and episodic memory', *Neuron*, vol. 35, no. 4. 2002. doi: 10.1016/S0896-6273(02)00830-9.

[3] D. Amaral and P. Lavenex, *Hippocampal Neuroanatomy*. 2006.

[4] J. K. Leutgeb, S. Leutgeb, M. B. Moser, and E. I. Moser, 'Pattern separation in the dentate gyrus and CA3 of the hippocampus', *Science (1979)*, vol. 315, no. 5814, 2007, doi: 10.1126/science.1135801.

[5] R. P. Kesner, 'An analysis of the dentate gyrus function', *Behavioural Brain Research*, vol. 254. 2013. doi: 10.1016/j.bbr.2013.01.012.

[6] A. Sahay *et al.*, 'Increasing adult hippocampal neurogenesis is sufficient to improve pattern separation', *Nature*, vol. 472, no. 7344, 2011, doi: 10.1038/nature09817.

[7] J. K. Leutgeb and S. Leutgeb, 'Book review: The dentate gyrus: A comprehensive guide to structure, function, and clinical implications', *Hippocampus*, 2009, doi: 10.1002/hipo.20662.

[8] R. D. Keynes, 'Ionic Channels of Excitable Membranes', *Trends Neurosci*, vol. 8, 1985, doi: 10.1016/0166-2236(85)90088-8.

[9] G. Buzsáki, 'Large-scale recording of neuronal ensembles', *Nature Neuroscience*, vol. 7, no. 5. 2004. doi: 10.1038/nn1233.

[10] C. M. Gray, P. König, A. K. Engel, and W. Singer, 'Oscillatory responses in cat visual cortex exhibit inter-columnar synchronization which reflects global stimulus properties', *Nature*, vol. 338, no. 6213, 1989, doi: 10.1038/338334a0.

[11] D. H. Hubel and T. N. Wiesel, 'Receptive fields, binocular interaction and functional architecture in the cat's visual cortex', *J Physiol*, vol. 160, no. 1, 1962, doi: 10.1113/jphysiol.1962.sp006837.

[12] G. Buzsáki and K. Mizuseki, 'The log-dynamic brain: How skewed distributions affect network operations', *Nature Reviews Neuroscience*, vol. 15, no. 4. 2014. doi: 10.1038/nrn3687.

[13] M. S. Lewicki, 'A review of methods for spike sorting: The detection and classification of neural action potentials', *Network: Computation in Neural Systems*, vol. 9, no. 4, 1998, doi: 10.1088/0954-898X_9_4_001.

[14] M. Pachitariu, N. Steinmetz, S. Kadir, M. Carandini, and K. D. Harris, 'Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels', 2016.

[15] N. A. Steinmetz *et al.*, 'Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings', *Science (1979)*, vol. 372, no. 6539, 2021, doi: 10.1126/science.abf4588.

[16] C. Rossant *et al.*, 'Spike sorting for large, dense electrode arrays', *Nat Neurosci*, vol. 19, no. 4, 2016, doi: 10.1038/nn.4268.

[17] P. Yger, G. L. B. Spampinato, and E. Esposito, 'Fast and accurate spike sorting in vitro and in vivo for up to thousands of electrodes', *bioRxiv*, 2016.

[18] C. Vargas-Irwin and J. P. Donoghue, 'Automated spike sorting using density grid contour clustering and subtractive waveform decomposition', *J Neurosci Methods*, vol. 164, no. 1, 2007, doi: 10.1016/j.jneumeth.2007.03.025.

[19] A. P. Buccino *et al.*, 'SpikeInterface, a unified framework for spike sorting', *Elife*, vol. 9, Nov. 2020, doi: 10.7554/eLife.61834.

[20] G. Buzsáki, C. A. Anastassiou, and C. Koch, 'The origin of extracellular fields and currents-EEG, ECoG, LFP and spikes', *Nature Reviews Neuroscience*, vol. 13, no. 6. 2012. doi: 10.1038/nrn3241.

[21] G. T. Einevoll, C. Kayser, N. K. Logothetis, and S. Panzeri, 'Modelling and analysis of local field potentials for studying the function of cortical circuits', *Nature Reviews Neuroscience*, vol. 14, no. 11. 2013. doi: 10.1038/nrn3599.

[22] A. J. Kundishora *et al.*, 'Restoring conscious arousal during focal limbic seizures with deep brain stimulation', *Cerebral Cortex*, vol. 27, no. 3, 2017, doi: 10.1093/cercor/bhw035.

[23] A. K. Engel and P. Fries, 'Beta-band oscillations-signalling the status quo?', *Current Opinion in Neurobiology*, vol. 20, no. 2. 2010. doi: 10.1016/j.conb.2010.02.015.

[24] S. L. Bressler and C. G. Richter, 'Interareal oscillatory synchronization in top-down neocortical processing', *Current Opinion in Neurobiology*, vol. 31. 2015. doi: 10.1016/j.conb.2014.08.010.

[25] M. Steriade, D. A. McCormick, and T. J. Sejnowski, 'Thalamocortical oscillations in the sleeping and aroused brain', *Science (1979)*, vol. 262, no. 5134, 1993, doi: 10.1126/science.8235588.

[26] V. V. Vyazovskiy, C. Cirelli, M. Pfister-Genskow, U. Faraguna, and G. Tononi, 'Molecular and electrophysiological evidence for net synaptic potentiation in wake and depression in sleep', *Nat Neurosci*, vol. 11, no. 2, 2008, doi: 10.1038/nn2035.

[27] M. M. Halassa *et al.*, 'State-dependent architecture of thalamic reticular subnetworks', *Cell*, vol. 158, no. 4, 2014, doi: 10.1016/j.cell.2014.06.025.

[28] J. Sarnthein, A. Morel, A. von Stein, and D. Jeanmonod, 'Thalamic theta field potentials and EEG: High thalamocortical coherence in patients with neurogenic pain, epilepsy and movement disorders', *Thalamus Relat Syst*, vol. 2, no. 3, 2003, doi: 10.1016/S1472-9288(03)00021-9.

[29] D. H. Perkel, G. L. Gerstein, and G. P. Moore, ‘Neuronal Spike Trains and Stochastic Point Processes: I. The Single Spike Train’, *Biophys J*, vol. 7, no. 4, 1967, doi: 10.1016/S0006-3495(67)86596-2.

[30] G. L. Gerstein and N. Y. S. Kiang, ‘An Approach to the Quantitative Analysis of Electrophysiological Data from Single Neurons’, *Biophys J*, vol. 1, no. 1, 1960, doi: 10.1016/S0006-3495(60)86872-5.

[31] Rieke F and Warland D, ‘Spikes: exploring the neural code’, *MIT press*, 1999.

[32] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. 2014. doi: 10.1017/CBO9781107447615.

[33] W. R. Softky and C. Koch, ‘The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs’, *Journal of Neuroscience*, vol. 13, no. 1, 1993, doi: 10.1523/jneurosci.13-01-00334.1993.

[34] A. Destexhe, M. Rudolph, and D. Paré, ‘The high-conductance state of neocortical neurons in vivo’, *Nat Rev Neurosci*, vol. 4, no. 9, 2003, doi: 10.1038/nrn1198.

[35] T. P. Vogels and L. F. Abbott, ‘Signal propagation and logic gating in networks of integrate-and-fire neurons’, *Journal of Neuroscience*, vol. 25, no. 46, 2005, doi: 10.1523/JNEUROSCI.3508-05.2005.

[36] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, ‘Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering’, *Neural Comput*, vol. 16, no. 8, 2004, doi: 10.1162/089976604774201631.

[37] E. N. Brown, R. E. Kass, and P. P. Mitra, ‘Multiple neural spike train data analysis: State-of-the-art and future challenges’, *Nature Neuroscience*, vol. 7, no. 5. 2004. doi: 10.1038/nn1228.

[38] S. Shinomoto, K. Shima, and J. Tanji, ‘Differences in Spiking Patterns among Cortical Neurons’, *Neural Comput*, vol. 15, no. 12, 2003, doi: 10.1162/089976603322518759.

[39] M. A. Yassa and C. E. L. Stark, ‘Pattern separation in the hippocampus’, *Trends Neurosci*, vol. 34, no. 10, pp. 515–525, Oct. 2011, doi: 10.1016/j.tins.2011.06.006.

[40] E. T. Rolls, ‘Pattern separation, completion, and categorisation in the hippocampus and neocortex’, *Neurobiol Learn Mem*, vol. 129, 2016, doi: 10.1016/j.nlm.2015.07.008.

[41] S. B. Elliott, E. Harvey-Girard, A. C. C. Giassi, and L. Maler, ‘Hippocampal-like circuitry in the pallium of an electric fish: Possible substrates for recursive pattern separation and completion’, *Journal of Comparative Neurology*, vol. 525, no. 1, 2017, doi: 10.1002/cne.24060.

[42] E. Harvey-Girard, J. Tweedle, J. Ironstone, M. Cuddy, W. Ellis, and L. Maler, ‘Long-term recognition memory of individual conspecifics is associated with telencephalic expression of Egr-1 in the electric fish apteronotus leptorhynchus’, *Journal of Comparative Neurology*, vol. 518, no. 14, 2010, doi: 10.1002/cne.22358.

[43] T. J. Bussey, T. L. Padain, E. A. Skillings, B. D. Winters, A. J. Morton, and L. M. Saksida, 'The touchscreen cognitive testing method for rodents: How to get the best out of your rat', *Learning and Memory*, vol. 15, no. 7, 2008, doi: 10.1101/lm.987808.

[44] 'https://intantech.com/RHD_USB_interface_board.html'.

[45] 'Allen Institute for Brain Science (authors) (2019) Allen Brain Observatory Neuropixels Allen Brain Map. ID 766640955. https://portal.brain-map.org/explore/circuits/visual-coding-neuropixels'.

[46] J. H. Siegle *et al.*, 'A survey of spiking activity reveals a functional hierarchy of mouse corticothalamic visual areas', *bioRxiv*. 2019.

[47] A. P. Buccino and G. T. Einevoll, 'MEArec: A Fast and Customizable Testbench Simulator for Ground-truth Extracellular Spiking Activity', *Neuroinformatics*, vol. 19, no. 1, 2021, doi: 10.1007/s12021-020-09467-7.

[48] S. Ramaswamy *et al.*, 'The neocortical microcircuit collaboration portal: A resource for rat somatosensory cortex', *Front Neural Circuits*, vol. 9, no. OCT, 2015, doi: 10.3389/fncir.2015.00044.

[49] H. Markram *et al.*, 'Reconstruction and Simulation of Neocortical Microcircuitry', *Cell*, vol. 163, no. 2, 2015, doi: 10.1016/j.cell.2015.09.029.

[50] J. Magland *et al.*, 'Spikeforest, reproducible web-facing ground-truth validation of automated neural spike sorters', *Elife*, vol. 9, 2020, doi: 10.7554/eLife.55167.

[51] 'https://spikeforest.flatironinstitute.org'.

[52] J. Magland *et al.*, 'SpikeForest, reproducible web-facing ground-truth validation of automated neural spike sorters', *Elife*, vol. 9, May 2020, doi: 10.7554/eLife.55167.

[53] A. Fernández-Ruiz *et al.*, 'Gamma rhythm communication between entorhinal cortex and dentate gyrus neuronal assemblies', *Science (1979)*, vol. 372, no. 6537, Apr. 2021, doi: 10.1126/science.abf3119.

[54] I. Zutshi, M. Valero, A. Fernández-Ruiz, and G. Buzsáki, 'Extrinsic control and intrinsic computation in the hippocampal CA1 circuit', *Neuron*, vol. 110, no. 4, pp. 658-673.e5, Feb. 2022, doi: 10.1016/j.neuron.2021.11.015.

[55] A. Oliva, A. Fernández-Ruiz, F. Leroy, and S. A. Siegelbaum, 'Hippocampal CA2 sharp-wave ripples reactivate and promote social memory', *Nature*, vol. 587, no. 7833, pp. 264–269, Nov. 2020, doi: 10.1038/s41586-020-2758-y.

[56] 'https://www.ntnu.edu/kavli'.

[57] 'https://buzsakilab.com/wp/'.

[58] 'https://probeinterface.readthedocs.io/en/main/'.

[59] 'https://www.cambridgeneurotech.com'.

[60] 'https://www.neuronexus.com'.

[61] 'https://phy.readthedocs.io/en/latest/'.

[62] 'https://en.wikipedia.org/wiki/Central_limit_theorem'.

[63] M. Allen, D. Poggiali, K. Whitaker, T. R. Marshall, and R. A. Kievit, 'Raincloud plots: A multi-platform tool for robust data visualization [version 1; peer review: 2 approved]', *Wellcome Open Res*, vol. 4, 2019, doi: 10.12688/wellcomeopenres.15191.1.

[64] L. Van Der Maaten and G. Hinton, 'Visualizing data using t-SNE', *Journal of Machine Learning Research*, vol. 9, 2008.

[65] 'https://cellexplorer.org'.

[66] S. Royer *et al.*, 'Control of timing, rate and bursts of hippocampal place cells by dendritic and somatic inhibition', *Nat Neurosci*, vol. 15, no. 5, 2012, doi: 10.1038/nn.3077.

[67] P. C. Petersen, J. H. Siegle, N. A. Steinmetz, S. Mahallati, and G. Buzsáki, 'CellExplorer: A framework for visualizing and characterizing single neurons', *Neuron*, vol. 109, no. 22, 2021, doi: 10.1016/j.neuron.2021.09.002.

[68] Y. Senzai and G. Buzsáki, 'Physiological Properties and Behavioral Correlates of Hippocampal Granule Cells and Mossy Cells', *Neuron*, vol. 93, no. 3, 2017, doi: 10.1016/j.neuron.2016.12.011.

[69] A. Bragin, G. Jando, Z. Nadasdy, M. Van Landeghem, and G. Buzsaki, 'Dentate EEG spikes and associated interneuronal population bursts in the hippocampal hilar region of the rat', *J Neurophysiol*, vol. 73, no. 4, 1995, doi: 10.1152/jn.1995.73.4.1691.

[70] M. Penttonen, A. Kamondi, A. Sik, L. Acsády, and G. Buzsáki, 'Feed-forward and feed-back activation of the dentate gyrus in vivo during dentate spikes and sharp wave bursts', *Hippocampus*, vol. 7, no. 4, 1997, doi: 10.1002/(SICI)1098-1063(1997)7:4<437::AID-HIPO9>3.0.CO;2-F.

[71] C. E. Shannon, 'Communication in the Presence of Noise', *Proceedings of the IRE*, vol. 37, no. 1, 1949, doi: 10.1109/JRPROC.1949.232969.

[72] M. Yamabe, K. Horie, H. Shiokawa, H. Funato, M. Yanagisawa, and H. Kitagawa, 'MC-SleepNet: Large-scale Sleep Stage Scoring in Mice by Deep Neural Networks', *Sci Rep*, vol. 9, no. 1, 2019, doi: 10.1038/s41598-019-51269-8.

[73] B. O. Watson, D. Levenstein, J. P. Greene, J. N. Gelinas, and G. Buzsáki, 'Network Homeostasis and State Dynamics of Neocortical Sleep', *Neuron*, vol. 90, no. 4, 2016, doi: 10.1016/j.neuron.2016.03.036.

[74] 'https://en.wikipedia.org/wiki/Welch%27s_method'.

[75] G. Buzsáki, *Rhythms of the Brain*. 2009. doi: 10.1093/acprof:oso/9780195301069.001.0001.

[76] O. Avila-Akerberg and M. J. Chacron, 'Nonrenewal spike train statistics: Causes and functional consequences on neural coding', in *Experimental Brain Research*, 2011. doi: 10.1007/s00221-011-2553-y.

[77] L. Ramlow and B. Lindner, 'Interspike interval correlations in neuron models with adaptation and correlated noise', *PLoS Comput Biol*, vol. 17, no. 8, 2021, doi: 10.1371/journal.pcbi.1009261.

[78] M. J. Chacron, L. Maler, and J. Bastian, ‘Electroreceptor neuron dynamics shape information transmission’, *Nat Neurosci*, vol. 8, no. 5, 2005, doi: 10.1038/nn1433.

[79] W. J. Wilbur and J. Rinzel, ‘An analysis of Stein’s model for stochastic neuronal excitation’, *Biol Cybern*, vol. 45, no. 2, pp. 107–114, Sep. 1982, doi: 10.1007/BF00335237.

[80] W. J. Wilbur and J. Rinzel, ‘A theoretical basis for large coefficient of variation and bimodality in neuronal interspike interval distributions’, *J Theor Biol*, vol. 105, no. 2, pp. 345–368, Jan. 1983, doi: 10.1016/S0022-5193(83)80013-7.

[81] N. Ahmadi, T. G. Constandinou, and C. S. Bouganis, ‘Estimation of neuronal firing rate using Bayesian Adaptive Kernel Smoother (BAKS)’, *PLoS One*, vol. 13, no. 11, 2018, doi: 10.1371/journal.pone.0206794.

[82] M. Nawrot, A. Aertsen, and S. Rotter, ‘Single-trial estimation of neuronal firing rates: From single-neuron spike trains to population activity’, in *Journal of Neuroscience Methods*, 1999. doi: 10.1016/S0165-0270(99)00127-2.

[83] H. Shimazaki and S. Shinomoto, ‘Kernel bandwidth optimization in spike rate estimation’, *J Comput Neurosci*, vol. 29, no. 1–2, 2010, doi: 10.1007/s10827-009-0180-4.

Appendix

1. All the codes are provided in a private GitHub repository[12]. Codes require in steps of Section 2.2.4.1 are provided below:

```
>> datI = int16(dat);
>> whos datI
  Name        Size                 Bytes  Class    Attributes

  datI      385x30000            23100000  int16

% This dataset is 385 channels by 30000 time samples.
>> fid = fopen('D:\my\path\myNewFile.bin', 'w');
>> fwrite(fid, datI, 'int16');
>> fclose(fid);
```

2. A table of the python libraries, sub-functions used for spike train analysis in chapter 4.

Name of the Library	Name of the function	Purpose of use
Numpy	Np.diff [13], Np.arange [14]	Computing ISIs, and evenly spaced values within a given interval.
Matplotlib	Matplotlib.pyplot [15], Matplotlib.plt.hist [16]	For data visualization and computing the PSTH.
Scipy	Scipy.signal.fftconvolve[17], Scipy.signal.welch [18]	For computing fast convolution in autocorrelation and power spectrum density suing Welch's method
Elephant	instantaneous _rate [19]	For computing the instantaneous firing rate using causal filters.

[12] https://github.com/farazmoradi/Pattern-Seperation

[13] https://numpy.org/doc/stable/reference/generated/numpy.diff.html

[14] https://numpy.org/doc/stable/reference/generated/numpy.arange.html

[15] https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html

[16] https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html

[17] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.fftconvolve.html

[18] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.welch.html

[19] https://elephant.readthedocs.io/en/v0.7.0/reference/statistics/elephant.statistics.instantaneous_rate.html