UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU EN ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

How do our brainwaves perceive the passage of time? Quantifying neural correlates of time during a rhythm performance task

---

Autor: Luca Eric Di Croce

Director:	**Dr. Ignasi Cos**
Realitzat a:	**Departament d'Enginyeria Informàtica**

Barcelona, **June 13, 2023**

Contents

Abstract

The aim of this project is to establish a methodology to quantify the extent to which different brainwave signatures vary in their past data retention, and to determine how other factors interact with these variations, using previously obtained EEG recordings.

To achieve this, we quantified the amount of past values that strongly influence future values for each brainwave signature throughout different EEG time series. Specifically, we calculated the number of lags required for a univariate autoregressive computational model to predict a set of brainwave time-series with an error (RMSE) below a preset threshold. In this fashion, we could establish that a similar number of lags were required based on the brainwave signatures (alpha, beta, gamma, or unfiltered) throughout the different conditions of activities and the different EEG sets, with the number of lags required being around 3, 4, and 6 for alpha, beta, and gamma brainwaves, respectively, when trying to achieve a minimum RMSE value of 0.001. This covariation is displayed again when using a different sets of threshold RMSE values, with gamma consistently having a greater dependency to past data, and alpha a lesser one.

Our results indicate that brainwave signatures that are more related to active states can use past data for a longer period of time than brainwave signatures related to relaxed states. Furthermore, they suggest that active-state brainwaves show a more dilated time perception than their relaxed counterparts.

In future studies, this methodology may help to establish a technique to objectively analyze time perception variation through EEG readings.

Resum

L'objectiu d'aquest projecte és establir una metodologia per quantificar com les diferents freqüències d'oscil·lació d'ones cerebrals varien en la seva retenció de dades passades, i determinar com altres factors interactuen amb aquestes variacions, utilitzant enregistraments d'EEG obtinguts prèviament.

Per aconseguir-ho, vam quantificar la quantitat de valors passats que influeixen notablement en els valors futurs per cada signatura d'ones cerebrals al llarg de diferents sèries temporals d'EEG. Concretament, vam calcular el nombre de lags necessaris per a un model computacional autoregressiu univariat per predir un conjunt de sèries temporals d'ones cerebrals amb un error (RMSE) per sota d'un llindar preestablert. D'aquesta manera, vam poder establir que es necessitava un nombre similar de lags en funció de les freqüències d'oscil·lació de les ones cerebrals (alfa, beta, gamma o sense filtrar) al llarg de les diferents condicions d'activitats i els diferents conjunts d'EEG, amb el nombre de lags necessaris al voltant de 3, 4 i 6 per a les ones cerebrals alfa, beta i gamma, respectivament, quan s'intenta aconseguir un valor RMSE mínim de 0,001. Aquesta covariació es mostra de nou quan s'utilitzen conjunts diferents de valors RMSE llindar, amb gamma constantment tenint una dependència a un nombre més elevat de dades anteriors, i alfa a un nombre inferior.

Els nostres resultats indiquen que les freqüències d'oscil·lació de les ones cerebrals que estan més relacionades amb estats actius poden usar dades passades durant un període de temps més llarg que les signatures d'ones cerebrals relacionades amb estats relaxats. Aquest fet suggereix que les ones cerebrals en estat actiu mostren una percepció del temps més dilatada que les seves contraparts relaxades.

En estudis futurs, aquesta metodologia pot ajudar a establir una tècnica per analitzar objectivament la variació de la percepció del temps mitjançant lectures d'EEG.

Resumen

El objetivo de este proyecto es establecer una metodología para cuantificar la medida en que las diferentes frecuencias de oscilación de las ondas cerebrales varían en su retención de datos pasados y determinar cómo otros factores interactúan con estas variaciones, utilizando registros de EEG obtenidos previamente.

Para lograr esto, cuantificamos la cantidad de valores pasados que influyen fuertemente en los valores futuros para cada frecuencia de oscilación de las ondas cerebrales a lo largo de diferentes series temporales de EEG. Específicamente, calculamos la cantidad de lags necesarios para que un modelo computacional autorregresivo univariante prediga un conjunto de series temporales de ondas cerebrales con un error (RMSE) por debajo de un umbral preestablecido. De esta manera, pudimos establecer que se requería un número similar de lags en función de las frecuencias de oscilación de las ondas cerebrales (alfa, beta, gamma o sin filtrar) a lo largo de las diferentes condiciones de actividades y los diferentes conjuntos de EEG, siendo el número mínimo de lags requerido alrededor 3, 4 y 6 para ondas cerebrales alfa, beta y gamma, respectivamente, cuando se intenta alcanzar un valor RMSE mínimo de 0,001. Esta covariación se vuelve a mostrar cuando se usan conjuntos de diferentes valores RMSE umbral, con gamma mostrando una dependencia a un número superior de datos pasados y alfa mostrando dependencia a un número inferior.

Nuestros resultados indican que las frecuencias de oscilación de las ondas cerebrales que están más relacionadas con estados activos pueden usar datos pasados durante un período de tiempo más largo que las firmas de ondas cerebrales relacionadas con estados relajados. Además, sugieren que las ondas cerebrales en estado activo muestran una percepción del tiempo más dilatada que sus contrapartes relajadas.

En futuros estudios, esta metodología puede ayudar a establecer una técnica para analizar objetivamente la variación de la percepción del tiempo a través de lecturas de EEG.

Acknowledgments

First and foremost, I would like to thank Dr. Ignasi Cos for offering me the opportunity to research this topic, and for patiently guiding me through the field of time perception.

I would like to thank Dr. Ole Adrian Heggli (Aarhus University, Denmark) for giving us access to his EEG reading data.

I am grateful to my friends and my family, who gave me the confidence and trust I needed to do this work, and to Tumble, for the many years he spent keeping our days simple and happy.

Finally, I would like to express my deepest appreciation to Carlota, who gave me both support and understanding, and patiently sat through my monologues of my ideas about this project.

Chapter 1

Introduction

1.1 Introduction

Cognitive computational neuroscience is a relatively new field that has become possible due especially to major advances in artificial intelligence (AI) (see review, Kriegeskorte Douglas, 2018). Interestingly, the interface between artificial intelligence and neurobiology was already shown in studies as early as 1996, which noted the benefits of applying an autoregressive model, either univariate or multivariate, to electroencephalogram (EEG) readings (see review by Pardey *et al.*, 1996). Recently, the use of neural networks (NN), convoluted neural network (CNN), and artificial intelligence (AI) has led to focus on using biological neural networks simulation through CNN-AI to model higher-level cognition in humans (see reviews: Battleday *et al.*, 2021, Macpherson *et al.*, 2021).

Electroencephalography (EEG) is a non-invasive technique that can directly measure neural activities with high-temporal precision (see review: Cohen, 2017). Over the past decades, studies have shown a tight link between EEG patterns of neural oscillations and perceptual, cognitive, motor, and emotional processes (see review: Siegel *et al.*, 2012). EEG readings are presented as waveforms that contain a multitude of different base signals (Beste *et al.*, 2023). These signals can be separated based on their specific frequency band, of up to 1000 Hertz (Hz), which is referred to as their brainwave signature. Most signals occur at low frequencies (of 1–100 Hz); these are generally classified as delta (with frequencies of 1–4 Hz), theta (4–8 Hz), alpha (8–13 Hz), beta (13–30 Hz), and gamma (30 Hz–100 Hz) (Sadaghiani *et al.*, 2022). These brainwave signatures are usually observed in deep sleep, light sleep, a relaxed state, an awakened state, or a highly concentrated state, respectively. Critically, it is assumed that oscillatory neural activity in the human brain plays a key role in information processing (Esghaei *et al.*, 2022). Specifically, these low-frequency, high-amplitude oscillations are especially useful for carrying information over spatial distances (Buzsaki and Draguhn, 2004).

Within the multidisciplinary field of cognitive neurobiology, this study aims to analyze

the complex topic of objective time perception measurement. While many studies have delved on time perception variation using subjective measures (e.g., Benau and Atchley, 2020; Buetti and Lleras 2012; Lake *et al.*, 2016; Tamm *et al.* 2015), few, if any, have been able to study this using purely objective measures (e.g., Damsma *et al.*, 2021; Marinho *et al.*, 2019). Thus, we proposed to take an initial step towards that goal by quantifying the measure of past time that a brainwave can retain and consider relevant to current events, and how this can vary based on brainwave signatures and external factors.

More specifically, we used the well-established autoregressive model to quantify the variation on the amount of past data that influenced future EEG values for three different brainwave signatures, alpha (8 to 13 Hz), beta (13 to 30 Hz) and gamma (30 and above Hz), and for different experimental conditions. Specifically, these were the waves present in the Heggli *et al.* 2021 experiment, from which we were obtained our EEG readings.

To isolate the brainwave signatures we were interested in, we used independent component analysis (ICA) to establish more independent signals from the channels of the different EEG readings, which had been separated into multiple datasets by experimental condition beforehand. Then we separated the different brainwave signatures for each dataset with a series of bandpass filters.

Once we produced datasets of independent channels showing the different brainwave signatures, we determined the mean minimum number of lags necessary for an autoregressive model to successfully predict each different channel of a dataset. We then separated these values by brainwave signature and experimental condition, which allowed us to objectively quantify the amount of past data each set considered relevant when making future decisions.

The past data retention displays the amount of information a brainwave still considers relevant to the current time, with larger amounts of data retained indicating longer time-periods in which past data is still considered relevant to the current situation. We take this time-period to be the "present" time of said brainwave. When we refer to a brainwave time perception variation, we are talking about this time period expanding or contracting.

We were able to establish that alpha brainwaves consider less past data to be significant to current events than gamma, with beta ranking in between, and that the experimental conditions given had a small to null effect in our values. This shows the different time perception variation for each of these brainwave signatures, as well as a useful methodology when trying to objectively quantify time perception. We believe this methodology, if modified and applied to larger scopes of data, will have future advantageous uses when trying to study and quantify in an objective manner the time perception of participants.

1.2 Motivation

Cognitive neurobiology is a complex field of research that encompasses the study of cognition, emotion, conscience, time perception, and many other topics. In this study, we

focus on the topic of time perception, or, more accurately, on the time perception variation between brainwave signatures.

To do so, we have created a methodology that permits us to objectively quantify the amount of past data the brainwave considers relevant to the current situation. The more data that is included, the more of a dilated concept of the present time that brainwave has.

We hope that this methodology can help in future research, for instance, by i) being used on persons with temporary affected time perception, due either to altered mental or physical states, or with an otherwise (for instance, due to dementia), to establish whether there are variations in the amount of past data retained by the brainwaves, and ii) as the basis for expanding and using to quantify a more complete time perception.

1.3 Planning

As recommended by the Univeristat de Barcelona, we used the Gantt methodology to plan our schedule on a week-by-week basis.

The project was broken down into the following objectives (see Section 2 for details):

Obj. 1. Create a pipeline and preprocess data;

Obj. 2. Process the data by separating the alpha, beta and gamma brainwave signatures and apply PCA;

Obj. 3. Check data for correlation and fit data into an autoregressive model;

Obj. 4. Compare the different minimum lag times and draw conclusion.

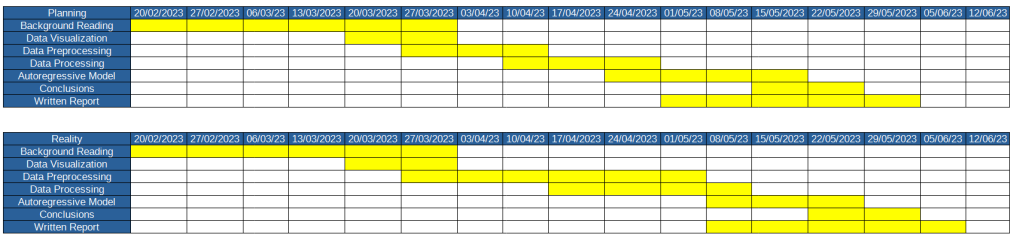


Figure 1.1: Gantt chart depicting the planned and actual project timing.

As indicated in the Gantt chart, I maintained the planned schedule until the data preprocessing phase. Instead of the planned two weeks, this phase took a total of 6 weeks due to complications in the application of the ICA methodology. However, following this, we were able to start working faster than expected and complete multiple phases simultaneously, which allowed me to finish only one week behind schedule.

Throughout this project, I held meetings with Dr. Cos every week or every two weeks (depending on his availability).

Chapter 2

Objectives

We aim to create a methodology that can quantify the perception of time of different brainwave signatures from electroencephalogram (EEG) data, and then to observe how external factors influence this perception. We therefore propose to determine: i) the objective amount of past events that significantly influence each brainwave signature, ii) how this amount varies based on brainwave signature, and iii) how experimental conditions modify this influence.

Obj. 1. Create a pipeline to analyze different raw EEG readings from previously published experiments (Heggli *et al.* 2021) (see Section 3.1.2). Of note, these data will need to be preprocessed until they are sufficiently cleaned for our purposes and in a more manageable data structure.

Obj. 2. Process data by separating the alpha, beta and gamma brainwave signatures and apply PCA to reduce the dimensionality of the data and to remove artifacts and less-significant signals. We will then establish whether there is: i) a correlation between channels of each dataset, and ii) an autocorrelation value for each channel of the datasets.

Obj. 3. Fit data into an autoregressive model with i) all channels of a set, if correlation between channels for each set exists, or ii) individual channels if not. We will use different lag values, which will be incremented with each fit until our autoregressive model is able to predict the dataset or dataset channel with a root-mean-square error (RMSE) value under a certain predetermined threshold. We will repeat this last step for different RMSE threshold values to see how the different lag values react.

Obj. 4. Compare and draw conclusions for the different minimum lag times required for each brainwave signature, experimental condition and RMSE threshold value, based on the data we generate.

Overall, this methodology will provide a technique and pipeline to study and quantify

how time is objectively perceived by different brainwave signatures. We imagine that these could be easily applied to other EEG datasets and experimental conditions in the future.

Chapter 3

Design

To quantify the time perception variation for different brainwave signatures, and the effects that different behaviors have, we create four large tasks.

First, we will preprocess the data from Heggli *et al.* 2021 to suit our needs, by removing the parts that are not relevant to our study, applying an independent component analysis (ICA) and formatting the resulting data to make it more manageable.

Once this is achieved, we will start processing the data, where we will separate the brainwave signatures we are interested in (alpha, beta and gamma) for each experiment condition and apply PCA for dimensionality reduction, creating sets of time-series separated by both frequency bands and experimental conditions. Alpha, beta and gamma brainwave signatures were chosen as they were all expected to be present in the EEG readings, and would prove useful for comparison purposes, as they reportedly correlate respectively to relaxed, comfortable and heightened emotional states.

With those sets, we autocorrelate each of their channels to prove that an autoregressive prediction is possible and meaningful. We then iteratively train an autoregressive model for each channel, increasing the given lag, and calculate the RMSE (root-mean-square deviation) value until a threshold RMSE value is achieved.

Finally, we calculate the mean value of the minimum necessary lags we needed to successfully predict each channels of a dataset, and compare these by brainwave signatures and experimental conditions. With this resulting values, we will then be able to objectively quantify the variation on past data retention based on brainwave signatures or activities.

3.1 Previous work

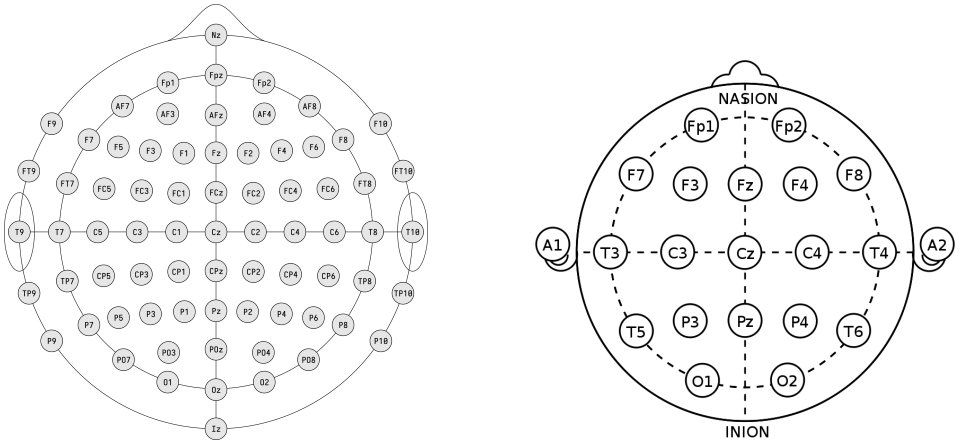
This study uses data from the experiments published previously (Heggli *et al.*, 2021) in the article "Transient brain networks underlying interpersonal strategies during synchronized

action". These data were provided to us for this purpose by Dr. Heggli.

3.1.1 Electroencephalography readings

Electroencephalography (EEG) is a generally non-invasive test that is used to analyze the brain biosignals and record the brain's activity. This is done by placing bioamplifiers and electrodes on the patient's scalp, which detects the spontaneous electrical activity of the brain.

Due to the electrical resistance of intermediary tissues and bone, biosignals from deeper brain layers are less intense. For this reason, to get a full picture of as much of the brain's activity as possible, electrodes are placed around the head, usually in a 10-10 or 10-20 pattern (Figs. 3.1a and 3.1b). Each electrode receives a signal, which is recorded as a distinct time-series. The time-series from each electrode are kept in separate channels.



(a) EEG 10-20 placement.

(b) EEG 10-10 placement.

Figure 3.1: The 10 and 20 determine the percentage of the front–back or right–left distance between electrode nodes. Source: Wikipedia.

The EEG readings are presented as waveforms. These readings tend to contain a multitude of different base signals. If these signals are separated, they can be classified based into different brainwave signatures based on their frequency band. Although there are still discrepancies, they are generally classified as either delta, with frequencies from 1 up to 4 Hz, theta, with frequencies from 4 up to 8 Hz, alpha, with frequencies from 8 up to 13 Hz, beta, with frequencies from 13 up to 30, and gamma, with frequencies higher than 30 Hz. These brainwave signatures are usually observed in deep sleep, light sleep, in a relaxed state, in an awakened state, or a highly concentrated state respectively (Fig. 3.2).

Currently, most EEG analysis, either to establish patterns or behaviors or to remove artifacts, are done by visual inspection from experts, although convoluted neural network AI models and autoregressive models have gained popularity and seen more widespread use in recent years (Suarkhi *et al.* 2021).

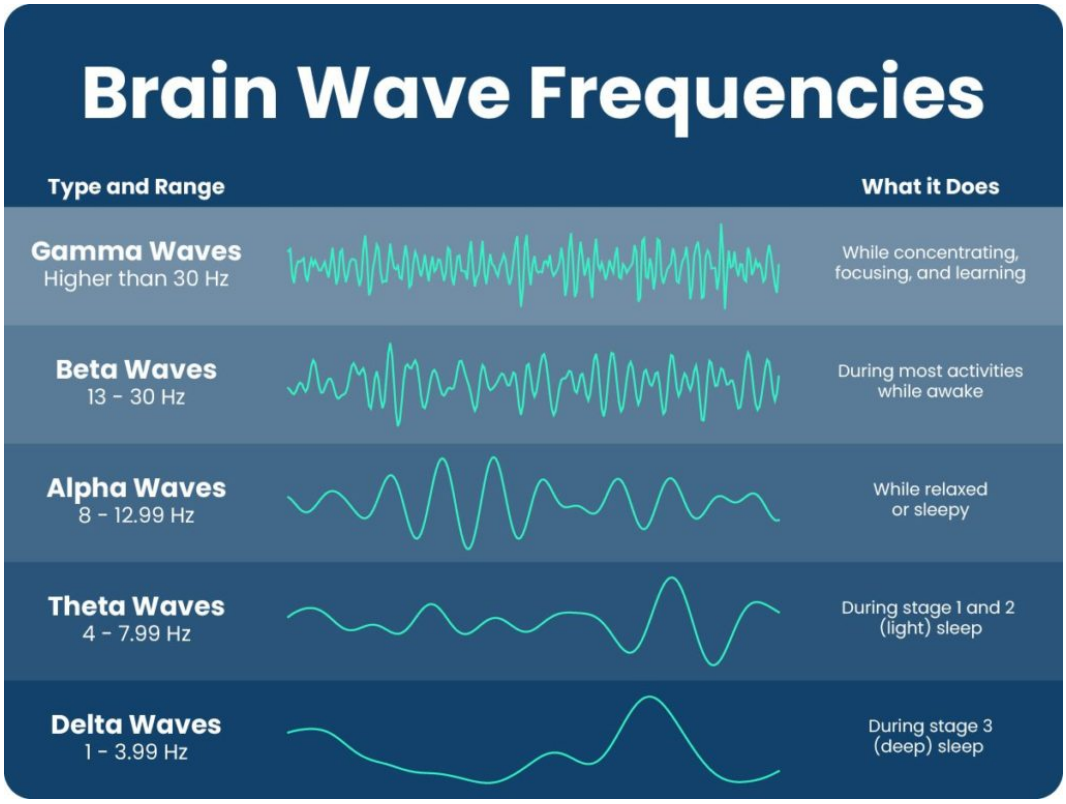


Figure 3.2: Brainwave signatures, their frequency band and the usual state they correlate to. Source: Sleep Foundation.

3.1.2 Heggli *et al.* 2021 study

The data used in this study was recollected by Heggli *et al.* 2021 for their study, "Transient brain networks underlying interpersonal strategies during synchronized action".

In the study, they selected a group of 20 musically-inclined individuals and made them perform a simple finger-tapping exercise, alone or in pairs. These participants were scanned by a continuous dual-EEG through 32 different scalp sites, further ahead referred to as regions or channels, that had been placed following the well-recognized 10-20 system. The EEG recordings were then digitized at a 1000 Hz sampling rate.

In the finger-tapping exercise, each participant was given a set of sound-proof headphones, a MIDI pad, and a computer screen, where a red cross-hair was displayed in the center. In the case of dyads (pairs), they were positioned to avoid visual contact with the other participant. They were then told to focus on the red cross-hair and were exposed to a rhythmic pattern while being shown the rhythm composition, and told to match it and continue it until the end of the test. This pattern could either be a 4/4 rhythm at 120 beats per minute (BPM) or a triplet of a 160 BPM 3-against-4 polyrhythm, henceforth referred to as metronome A and metronome B respectively. Throughout the tests, due to

the noise-canceling headphones, the participants were unable to hear their rhythm.

There was a total of 200 tests per participant. Of these 200 tests, in 100 of them, labeled as uni, the participant was alone and the computer-generated rhythmic pattern did not alter. These 100 uni tests consisted of 50 tests where the unaltering computer rhythm followed metronome A and 50 tests where metronome B was followed instead. In the other 100 tests, labeled "bidi", the participants were placed in pairs. In these, for the first two seconds, the participants were again exposed to a computer-generated rhythm, but in the last 10 seconds of the test, they heard only the rhythm marked by the other person's tapping. In 50 of these tests, both participants heard the same rhythm, which was labeled as consistent ("cons"), with 25 of these consistent bidi tests following metronome A, and 25 following metronome B. In the other 50 bidi tests, which were labeled as conflictive or conf, the participants were told that the rhythm they were listening to and had to emulate was the same one the other participant was hearing, as was the case in consistent tests, but each participant was secretly played distinct rhythms. This was done to study the interpersonal strategy each would use, consisting of either a mutual adaptation behavior or a leading-leading one.

The EEG recordings were started 2 seconds before the metronome start, and ended three seconds after the metronome finish, for a total of 17 seconds recorded. Due to the 1000 Hz sampling rate, this led to the collection of 17,000 data samples per test. Some of the tests had to be discarded due to excessive noise, and most participants had around 195 acceptable tests instead of the original 200, which gave roughly 3,315,000 data samples per participant.

3.1.3 Experimental data

We were granted access by Dr. OA Heggli to the EEG readings of two participants, each with 200 tests. These readings were sent in separate MatLab files (.mat), and indicated in the name which session they belonged to and if they were from the first or second participant of the pair. We received s02P2 and s06P1, which belonged to the second person of the second session and the first person of the sixth session. Each of these files contained one MatLab structure, named either P1 or P2 to indicate to which participant of the pair it belonged to. These structures contained two more substructures, labeled as uni, where the uni test data was kept, and bidi, where the bidi test data was kept.

The uni MatLab structure had four columns. Each of these columns generally has 100 rows, with each row containing all the pertinent data about a valid test. The first one, labeled EEG, contains a 32×17000 double matrix per uni row, with the 32 rows of said matrix correlating with the electrodes used and the 17000 rows containing the EEG readings of said electrodes.

The second column of the uni structure, labeled ELECTRODES, contains a 32×1 cell matrix per uni row. Each of these 32 cells has the name of the electrode used, which identifies its position in the 10-20 electrode placement consensus. It is organized in such a way that the cell of the first row of the 32×1 ELECTRODES matrix contains the name of the electrode

used to read the information displayed in the first row of the 32×17000 EEG matrix, with subsequent rows following the same organization. The third column of the uni structure, labeled TYPE, contains a string of char per row, which can be 'uni' or 'bidi', and indicate if the test in said row was either done without another participant, uni, or with bidi (see Section 3.1.3). Finally, the fourth column of the uni structure, labeled METRO, contains a double value per row, which may be either 0.5000 or 0.3750. The 0.5000 value indicates that the metronome followed in this test was metronome A, and the 0.3750 value indicates that the test followed instead the metronome B.

The bidi MatLab structure instead had 5 rows. This structure also had an EEG, ELECTRODES, TYPE, and METRO column, which served the same purposes as in the uni structure. But it also has an extra column, labeled CONDITION, which contains a string of chars, that can be either 'cons' or 'conf'. The 'cons' value indicates that the participants of the test represented in this row had the same metronome. The 'conf' value instead indicated that the test in this row was of conflictive rhythm so that while both participants were told that the other one had the same rhythm as them, each participant was given a different metronome.

3.2 Data preprocessing

The original test data received from Heggli was two MatLab files, each containing a data structure named sNPX, whereby N indicates the session numerical identifier and X indicates the participant numerical identifier, to identify either person 1 or 2. As the files that were sent to us in a MatLab encoding, and due to some complications found while applying the ICA (see Section 4.1.2), the data preprocessing phase of this study was done using MatLab R2023a.

The data structure received was formed by two substructures, termed uni and bidi, which held the single or dual finger-tapping exercise recordings, respectively. Each of these substructures had approximately 100 rows, one per each test that was performed and evaluated as valid, and 4 or 5 columns, which contained the EEG reading for that test, the electrodes identifier, the metronome identifier, 0.5000 for A or 0.3750 for B, and the type identifier (uni or bidi); additionally, the bidi substructure also included a condition column that indicated if the test was conflictive or consistent (see also Section 3.1.3).

Once we separated and assigned these substructures to new structures, also termed uni and bidi, we applied a series of notch filters at 50 Hz, 100 Hz, and 150 Hz to remove the power line noises mentioned in the EEG data preprocessing Section of the Heggli *et al.* 2021 paper.

From there, we removed the first 3000 and last 2000 datapoints of each test, keeping only the 12,000 data points that held the actual EEG test readings, and proceeded to concatenate all tests, by type and condition, forming the filtered_uni, filtered_bidi_cons, and filtered_bidi_conf datasets. We then separated each of these sets by the metronome, forming filtered_uni_A, filtered_uni_B, filtered_bidi_cons_A, filtered_bidi_cons_B, fil-

tered_bidi_conf_A, and filtered_bidi_conf_B.

We then applied an independent component analysis (ICA) tool to each of the 9 datasets. Through this, we were able to establish the separated signals over the 32 channels, isolating repetitive artifacts like pulses, breathing, and eye blinks.

Once the preprocessing was done, we saved the 9 resulting datasets to a MatLab file to be able to open and work on them in Python.

3.2.1 Data separation

The objective of this study is to objectively quantify the variation in the amount of past data that is relevant for different signatures, and be able to see how this amount changes based on the brainwave signature and the influence different experimental conditions have.

For this, we created 9 subsets of our tests, arranged based on the experimental conditions of the tests. The three explicit ones were: i) the uni subset, which had all tests of TYPE 'uni'; ii) the bidi_cons test, which had all tests of TYPE 'bidi' that also had CONDITION 'cons'; and iii) bidi_conf, which had all tests of TYPE 'bidi' that also had CONDITION 'conf'. Although not an explicit subset, we can also use the bidi_cons and bidi_conf in tandem to generate the bidi subset. From the 3 explicit subsets, we generated 6 more subsets, the uni_A, uni_B, bidi_cons_A, bidi_cons_B, bidi_conf_A, and bidi_conf_B. The final character, A or B, indicates the metronome these tests followed. Even though we did not expect the metronome to play an important role, it was nonetheless a potentially interesting condition to maintain for comparison purposes.

These separations were done by selecting the appropriate rows using the MatLab logical structures. For the bidi cons-conf separation, we made the logical structure using the strcmp function and compared the CONDITION column to the 'cons' value, with all 1 values being selected for the bidi_cons dataset and all 0 values being selected for the bidi_cons value. As for the metronome separation, we made the logical table by checking the METRO column values: values of 0.4 indicated that the row was metronome A, and of <0.4, metronome B.

3.2.2 Data segmentation

Even though each test had a 12-second duration, the Heggli *et al.* 2021 experiment recorded the previous 3 seconds prior to the test and the 2 seconds post-test. While this helped to eliminate the baseline artifacts and to establish a base pattern, these data for us introduced likely changes in the participant's mental state, as they moved from the more relaxed state from before the test, to a more focused state during the test, to once again a more relaxed state after the completion of the test. Therefore, we decided that it would be more beneficial to lose some data to ensure a higher degree of homogeneity in all segments in the same experimental condition subset, than to risk adding more unnecessary variations to the data that could potentially throw off the autoregressive model

predictions.

To that end, we discarded the first 3000 data points and the last 2000 data points of each test of each subset previously created, keeping only the remaining 12,000 data points. Once every test was trimmed, we proceeded to concatenate all tests of each subset together. This allowed us to use a subset as a longer time series instead of multiple shorter time series, which will help the ICA and the PCA analysis, and the autoregressive model fit, drastically reducing the time needed to compute them and increasing their analysis capabilities.

3.2.3 Notch filtering

In the original experiments, the reading equipment power lines interfered with the readings, which added a significant amount of 50 Hz, 100 Hz, and 150 Hz noise (Heggli *et al.* 2021). Considering the amount of noise, we decided once again worth losing any potential signals in those frequencies, as the interference could be more disruptive for our study than lost data.

For this, we applied three times the predefined second-order IIR notch filter from MatLab, one with a stopband at 50 Hz, one with a stopband at 100 Hz, and one with a stopband at 150 Hz. Notch filters, also referred to sometimes as band stop filters, serve to attenuate or remove signals in certain frequency ranges. They are generally formed by two filters, a high-pass filter, and a low-pass filter. These filters attenuate anything above, in the high-pass filter case, or below, in the low-pass filter case, of a certain cutoff range. By combining these two, we can attenuate all the signals in a certain frequency band, while keeping all other signals intact.

A notch filter can be defined by the stop-band frequency, which is the frequency range it is attenuating, the stop-band attenuation, which is the intensity at which it is attenuating the signals in that frequency range, and the transition from stopband to passband, which is how quickly the filter goes from letting a signal frequency pass to attenuating it.

3.2.4 Independent component analysis (ICA)

EEG readings record the brain's activity, including signals we are not interested in, which we refer to here as artifacts. To be able to properly analyze our readings, we have to first remove the artifacts present, such as those due to eye movement, sweat, pulse, or respiratory. Thankfully, the data handed to us already had the more severe cases of noisy readings removed, as stated in the data preprocessing Section of the Heggli *et al.* 2021 paper.

To remove the remaining artifacts, we used the fastICA function from the pca_ica MatLab toolbox, which implements a technique named Independent Component Analysis (ICA). This technique tries to establish all the source signal components of all channels and separate them to maximize the independence between each channel. To do so, the ICA technique presumes three things. It presumes that all source signals are non-Gaussian,

and therefore, will less closely resemble a Gaussian distribution than the sums of these signals, the original signal. It also presumes that the source signals are more independent of one another than the original signal, seeing how this one is formed from the sum of the source signals. Finally, it presumes for the same reason that the source signals are less complex than the original ones. With those presumptions, the ICA iterates through the given channels from the original signal until it can establish source signals that maximize the three previously stated presumptions.

This technique is applied to each subset, now a single longer time series with 32 channels. While separating the source components, the ICA will isolate the cyclic artifacts, such as breathing, heartbeats, and eye blinking into channels, as they are perceived as repeating source signals independently from the others.

3.3 Data processing

Once we had the 9 datasets with ICA applied, we transferred them to our Python IDE, PyCharm 2022.1, and we separated the signal datasets into alpha, beta, and gamma signal components using a bandwidth filter. As we were dealing with a large amount of data, with each set containing around 32×120000 double data points, and with a final total of 36 subsets, the computation cost is rendered extremely taxing on the computer's system. Considering that, we applied downsampling to each channel of each dataset, reducing the computational cost while minimizing data loss. We also applied the principal components analysis (PCA) technique to reduce data dimensionality, which not only further decreases the computational cost, but also removes the channels that contained more artifacts and therefore have a more minor impact on the data.

3.3.1 Bandpass filtering

Neural signals can be classified based on their frequency range, with the most widely recognized brainwave signatures being the delta signature, the theta signature, the alpha signature, the beta signature, and the gamma signature. The majority of studies agree that each frequency band can be correlated to a different type of mental state, with delta correlating to deep sleep, theta to light sleep, alpha to relaxed wakefulness, beta to concentration, and gamma to high concentration. In this study, we only consider the last three, alpha, beta, and gamma, to be pertinent, as the other two frequency band brainwaves should not be present in the EEG readings (see Section 3.1.1 and Section 3.1.2 for clarification).

A bandpass filter is a signal processing device that, given a certain frequency band, can attenuate all other frequencies not contained in that band. Like the notch filter mentioned earlier, it is also formed by two different filters, a high-pass filter, and a low-pass filter. These bandpass filters prove extremely useful when trying to isolate a signal of a known frequency band from a signal agglomeration.

Using a third-degree iirfilter from scipy.signal, setting the btype argument to "bandpass"

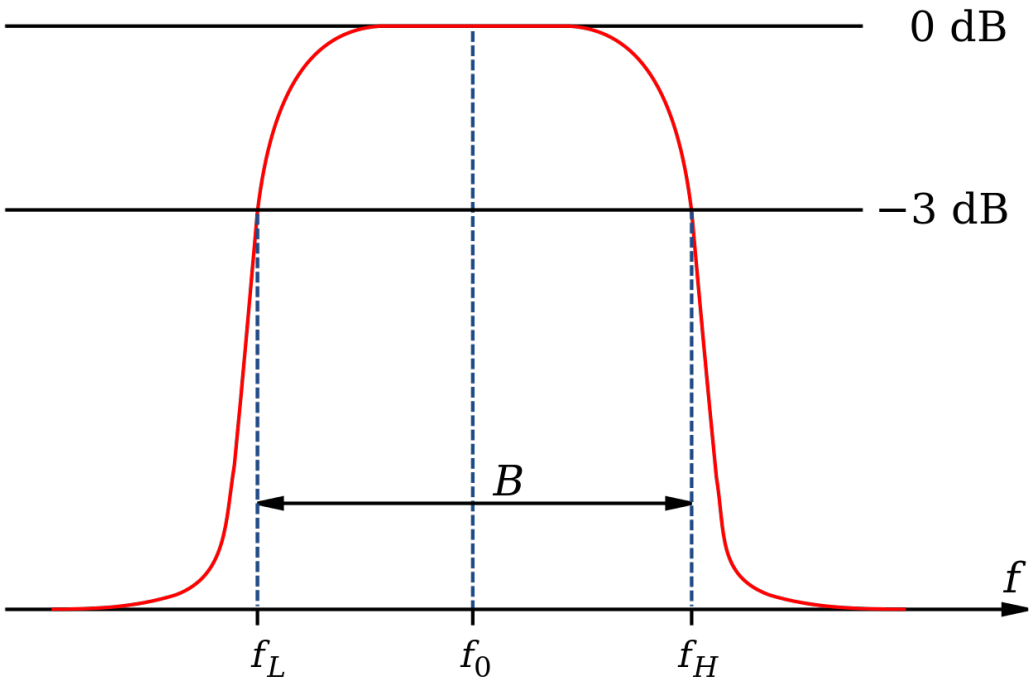


Figure 3.3: Example of a magnitude response to a bandpass filter. The desired frequency range(fL-fH), named bandpass, is left unaffected, while all other frequencies are attenuated. Source: Wikipedia.

and the ftype argument to “butter”, we applied 3 bandpass filters to each of our channels, with critical frequencies of 8/500 and 12/500, 12/500 and 30/500, and 30/500 and 80/500, to isolate alpha, beta, and gamma frequencies respectively, with the 500 representing our Nyquist rate, or half the sampling rate. This was done to each channel of the 9 original datasets, creating a further 27 more datasets, with 9 solely in alpha frequency bands, 9 in beta frequency bands, and 9 in gamma frequency bands.

3.3.2 Downsampling

Similar to subsampling, downsampling reduces a data series in size by a certain factor; however, it usually has a much smaller degree of information loss. For this, we created a copy of the original time series, but n-times smaller, whereby n is the downsampling factor. Each data point of this new time series is then given the values of the mean of those n datapoints which correlates to it in the original array, such as:

The higher the downsampling factor is, the less it costs to process that data but the more data information is lost. Even so, we considered it appropriate, and applied this downsampling to each channel of each dataset, always by the same factor, to reduce computational costs.

3.3.3 Principal component analysis (PCA)

The principal component analysis (PCA) is a statistical technique for dimensionality reduction in data. As implemented by the PCA function from the sklearn.decomposition library, this technique uses singular value decomposition (SVD) to establish the matrix eigenvector; from there, we calculate the rest of the components needed to define the matrix of arrays, which in our case is the dataset of channels, by finding the orthogonal vector to all previously established vectors that maximize the projected data variance.

Applying this technique to each dataset, we received a set of vectors, and the explained variance ratio of the projected data of each vector. With this variance, we can then select the minimum number of vectors we will need to reach a threshold explained variance ratio value. We will use these new vectors as our time-series channels.

3.4 The autoregressive model

An autoregressive model is a linear regressive model which predicts future values of a dependent variable using past values of said variable, instead of the values of independent variables. To be able to do so, the dependent variable must depend on its past values. This is the reason why autoregressive models are so commonly used on time series, and on EEG readings (see for instance Jaipriya and Sriharipriya, 2023). In addition to this, the simplicity of analyzing autoregressive models helped us to decide to use them. However, as mentioned above, an autoregressive model only works on a time series that is autocorrelated. To demonstrate that this was indeed the case, we first calculated the correlation between the channels of each same dataset, and then the autocorrelation of each channel.

3.4.1 Determining correlation between channels of the same dataset

Correlation is a statistical tool used to determine the degree to which two variables, or, in our case, time series, are related to each other. The most commonly used method of calculating this, and the one that was chosen for this project, is known as Pearson's product-moment coefficient, or correlation coefficient. This is calculated by dividing the covariance, which measures the joint variability of the two variables, by the product of the standard variation of the variables. This is mathematically represented as:

$$\rho X, Y = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

where ρ is the correlation coefficient, cov is the covariance, and σ represents the standard variation.

In this study, this is used to establish the similarity between channels of the same dataset by using the corr function of the pandas.DataFrame library. If we can demonstrate a high degree of similarity, this will let us apply the autoregressive model to a smaller number of channels, or even try using multiple channels in the autoregressive model fitting.

3.4.2 Determining the autocorrelation of channels

Autocorrelation is the correlation of a variable or time series to a lagged version of itself. A lagged version of a time series, is, simply put, a series that, instead of being at a time t, is at a previous time, t-n, where n is the lag. Therefore, when we apply autocorrelation to a time series, we are correlating the time-series variable to its past value. If this autocorrelation is significant, it generally proves that the time series is non-random, and future values can be predicted by past ones.

Autocorrelation(S) = Correlation(St, St-n), where S represents a time-series, t represents a given time time series, n is the lag value, which must be smaller or equal to t, n=< t.

As stands to reason, an autocorrelation of a time-series s with lag n signifies that any value vt from the time-series s is correlated to the past value vt-n, but does not necessarily need to be correlated to all the past values between vt-n and vt. To establish that, we must autocorrelate the time series for each of those lag values individually (see also Surakhi *et al.* 2021).

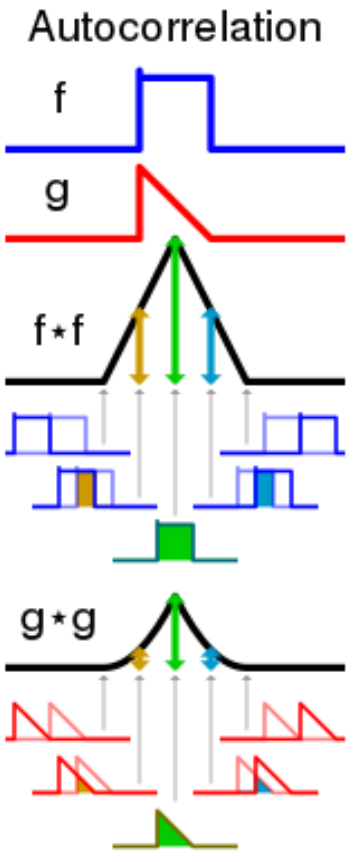


Figure 3.4: Visual representation of an autocorrelation. The colored zone indicates the autocorrelation coefficient. Source: Bitstream.

There are a couple of ways to autocorrelate a time series. The most intuitive one is following the same process as with the correlation method, of dividing the covariance by the product of the standard deviations of the variable, with the only change being that the variables used to calculate the covariance and the standard deviation are the value and its lagged value. The autocorrelation can also be calculated by taking the real part of the Fourier transform, as understood by the Wiener–Khinchin theorem.

In this study, we originally used statsmodels.graphics.tsaplots's method plot_acf to visualize the autocorrelation coefficient based on the lag, which in term implements matplotlib.pyplot's xcorr method, to calculate the cross-correlation on a certain lag, which is performed by numpy's correlate method with mode full. But due to the large processing cost, we decided to manually implement the two methods described above, and we applied them throughout every channel of every dataset, and, through visual inspection, we established if our processed EEG time series values are dependent on past values or not. If they are, then we will be able to fit the autoregressive model with the channels, and successfully, to a higher or lower degree based on the autocorrelation coefficient, predict these future values.

For further autocorrelation visualization, and to check for outlier values, we also use the lag_plot function from the pandas.plotting library on each channel, plotting the correlation of said channel to its own lagged version, for lag values from 1 to 20.

3.4.3 Autoregressive model

An autoregressive model is a linear time-series prediction model that calculates future values based on past values by assuming that past values are indicative of future results. Like the regressive linear models, it takes several parameters and assigns a weight to each of them, but unlike a regressive model, the parameters are not the values of independent variables, but the past values of the dependent variable. To compute the new predicted values, the autoregressive model adds the product of each parameter by its weight and adds one weight extra, also described as w0, or white noise. When we fit an autoregressive model, the model assigns aleatory values to each weight, including w0, and changes each weight value to minimize the loss function. Mathematically speaking, it is defined as:

$X_t = \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t$

where φ is the parameters of the model, ε is white noise, and $X_{t\text{-}i}$ is the past values of the dependent variable, the time series. The number of parameters is referred to, in this study, as the number of lags.

When we train, or fit, an autoregressive model, we try to find the weights that will minimize the loss function, which in our case is the root-mean-square error (RMSE) measure. This measure is a common and reliable way of seeing how far the predicted data is from the actual data. It does so by subtracting each estimated value from each actual value, squaring it, dividing it by the total number of values, and finally, calculating the square root. The formula is defined as:

$\sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{X}_i)^2}{N}}$

where N represents the number of datapoints, x represents the datapoint values amd represents the predicted value.

Note again that the goal of training an autoregressive model is to predict values as close to the actual values as possible, by calculating the parameter weights that would minimize the RMSE. In this implementation of the autoregressive model, we use the stochastic gradient descent algorithm. This algorithm, given a certain learning rate, or array of learning rates, assigns a random value to each weight, and then randomly selects either one or a small batch. It then calculates the derivative of that weight function for the current weight value and subtracts the product of that times the learning rate to the original weight. This process is repeated until a suitably small RMSE value is accomplished.

$\omega := \omega - \eta \nabla Q_i(\omega)$

with ω being the parameter, η being the learning step, and Q_i(ω) being the previous formula for the parameter.

The specific AR model that was implemented was AutoReg from statsmodels.tsa.ar_model. As we expected each channel to be mostly independent of one another but highly autocorrelated, we decided to train the AR model based on each channel individually. To do so, we iterated through each channel, doing the train-test split, with most of the data going to the train, except for the last 2000 values, which went to the test. Once we had the train-test split of a channel, we trained the AR model by using the fit() method, adding as parameters the train segment of that channel and an incremental lag value, which started at 1. Once the AR model was fitted with the training data, we extracted the fitted parameters, or weights, from the params attribute. We then manually predicted the test data points, passing as parameters the previous actual test values. We calculated the RMSE and plotted the actual test values and the predicted test values over one another. This process was repeated until we reached a minimum threshold RMSE value.

3.5 Quantifying time perception

The overarching objective of this study is to quantify the perception of time in different brainwaves and to determine the impact that external or internal stimuli can have on it. Note that, in this study, we refer to the time perception of a brainwave to indicate the data retention that a brainwave has, as evident by the amount of past data it seems to consider for future actions. In other words, we are trying to establish the amount of previous data that affects future data. To be able to quantify that, we decided to use autoregressive (AR) models.

AR models use a certain number of past data, determined by the number of lags, to predict future data. To establish the minimum number of lags we would need, we set the

initial number of lags as 1, which is the smallest number of parameters possible for an autocorrelated time series, we fitted an AR model with one of our channels, and we tried predicting the channel. We repeated this process, increasing the number of lags by 1 each iteration until the AR model prediction had an RMSE value lower than a predetermined threshold value.

Once we were under that minimum RMSE threshold value, we stored the minimum necessary number of lags that had been required for that channel and repeated the process for each channel of each dataset. With all the channels done, we can establish the minimum necessary lag of each brainwave signature or experimental condition by simply taking the mean of the minimum necessary number of lags of all channels of the datasets that met that criterion. By being able to reduce that minimum necessary number of lags to a single value, we were able to plot them with their respective standard deviation error, and easily compare how the amount of data considered relevant to current events varied across brainwave signatures and experimental conditions.

Chapter 4

Results

4.1 Data preprocessing

4.1.1 Notch filtering

A notch filter attenuates all signals with frequencies inside a certain frequency band (see Section 3.2.3 for clarification). We placed this frequency band at 50 Hz, 100 Hz, and 150 Hz to suppress the noise produced by the equipment powerlines in the experiment. We also decided to give it a relatively small bandwidth, of only 5 Hz. Both of these measures were then normalized to radians per second (rad/s); this resulted in 0.1 rad/s, 0.2 rad/s, and 0.3 rad/s for the notch frequency band, and in 0.01 rad/s for the bandwidth.

We then plotted our magnitude response using the MatLab ftool, to visualize how our filter acted (Fig. 4.1). As we expected, the magnitude response for the 50 Hz, 100 Hz and 150 Hz shows significant attenuation, while maintaining most of the surroundings frequencies with little to no attenuation.

4.1.2 Independent component analysis (ICA)

The ICA technique is an extremely useful tool for automatically isolating artifacts and source signals from EEG readings or from any other waveform time-series. In this study, we applied it on each of our datasets to isolate artifacts into separate channels, so they could be more easily removed by PCA. Of note, this was the most problematic step to implement and took longer than expected (see Section 1.3 for clarification).

Implementation of independent component analysis (ICA) in Python

We originally started using JupyterLab version 3.0.14, with a Python kernel, for this project. However, due to the ICA implementation, as detailed below, we decided to switch integrated development environment (IDE) in favor of a more computationally oriented

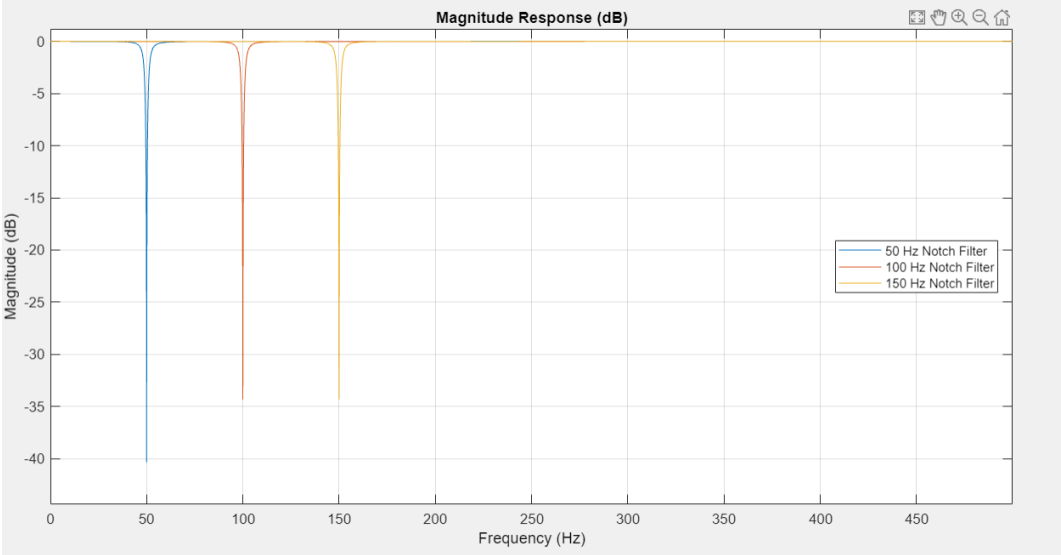


Figure 4.1: Magnitude response to different frequencies for the 50 Hz, 100 Hz and 150 Hz notch filters used represented in blue, red and orange respectively. As we can see, while the desired frequencies are attenuated, the frequencies surrounding them are left mostly at the original magnitude.

IDE. After opening the files and displaying the data in JupyterLab, we tried using Python's fastICA from the sklearn.decomposition library to reduce signal overlap and remove artifacts in future steps (by visual inspection or using the PCA technique). However, when we applied the fastICA to any dataset, the algorithm was unable to converge into significantly independent signals.

We next raised the iteration limit of the fastICA from 200 to 1,000,000; at this point, it required around 15 hours to compute per dataset. We also raised the tol parameter, which determines the tolerance the function has in considering the signals independent, and we tried concatenating similar datasets to increase the amount of information points it had available. When that proved ineffective, we wondered whether this inability to find independent signals was due to data heterogeneity. To address this, we segmented the datasets back into tests, and then divided the tests into smaller, 0.5-second time-series; however, it was still unable to converge.

After a month of iterating through different parameters and data preprocessing pipelines, we switched to MatLab, as their specific ICA implementation showed more promise.

Implementation of independent component analysis (ICA) in MatLab

MatLab is one of the most widely used IDE in the field of neurology and is the IDE that was used for the Heggli *et al.* 2021 paper. Using the fastICA implementation from the pca_ica MatLab toolbox, we were able to apply it to our datasets and have it converge in under 100 iterations.

Indeed, using this ICA algorithm implementation we were able to converge our datasets in under 100 iterations into separated source signals, thereby reducing the noisiness of the channels (Figs. 4.2 and 4.3).

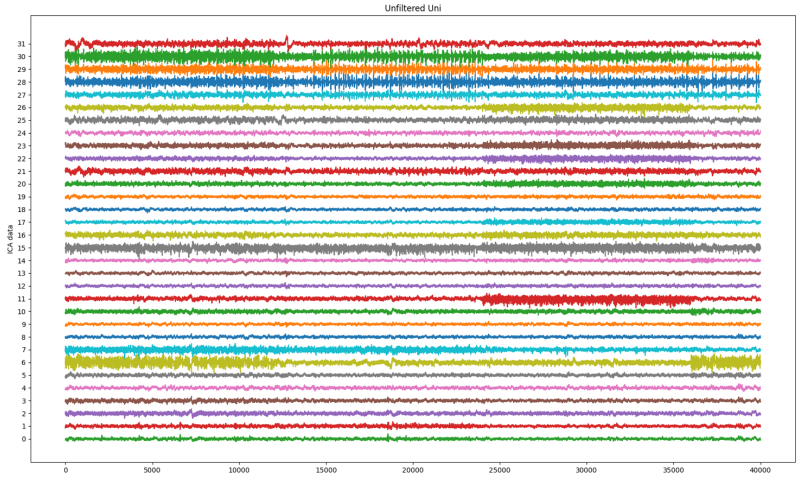


Figure 4.2: Plotted uni_A EEG dataset before ICA application. Channels are shown as erratic.

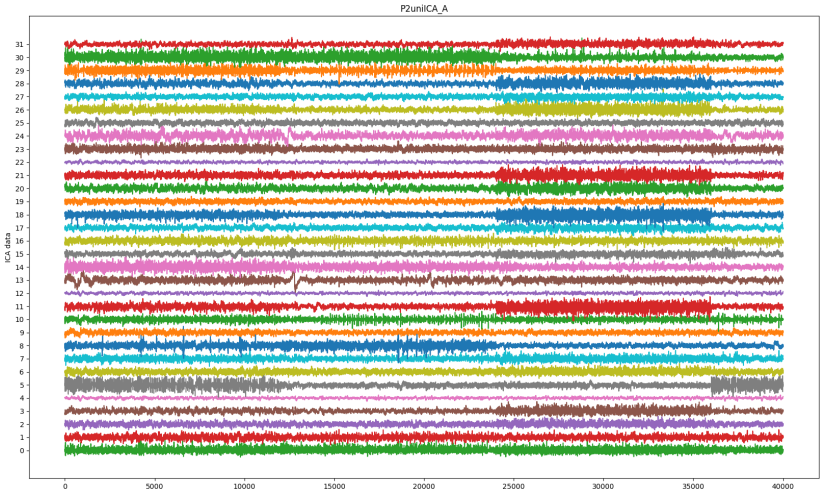


Figure 4.3: Plotted uni_A EEG dataset after ICA application. Some channels are left as null lines, and channels are generally less erratic.

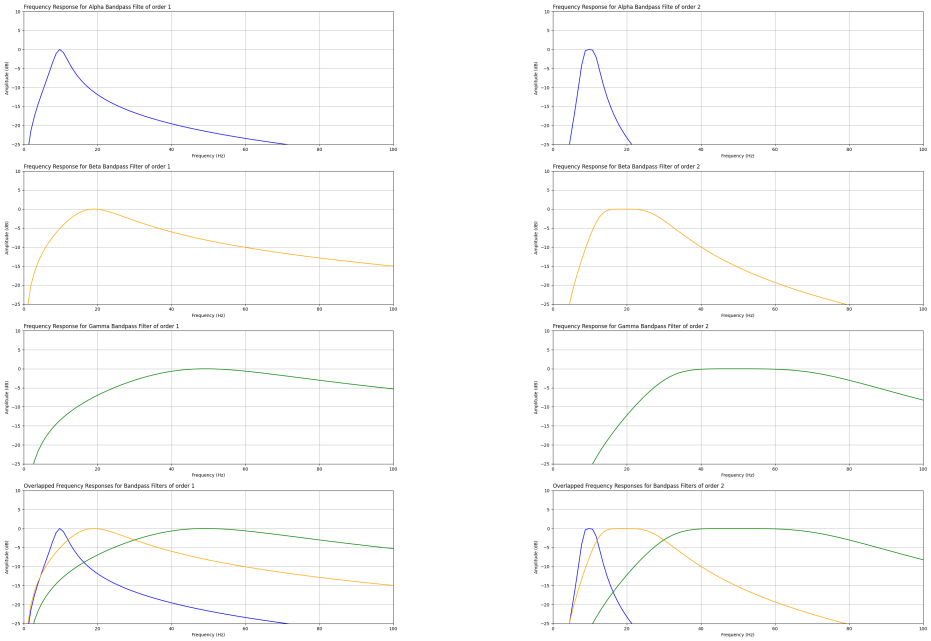
4.2 Data processing

4.2.1 Bandpass filtering

We used the Butterworth bandpass filters to isolate the alpha, beta, and gamma brainwave frequencies in our signals (see also Section 3.3.1). To choose the filter order that better

suits our needs, we plotted the magnitude response of our filters, using freqz from the scipy.signal library.

Neither order 1 nor 2 had steep-enough cutoffs for our needs (Figs. 4.4a, 4.4b, 4.5a, 4.5b, 4.6a and 4.6b). In turn, order 5 spiked in the alpha range, and order 6 is not able to do such a small bandpass as the one given in alpha. That leaves us with orders 3 and 4, from which we chose to use order 3.



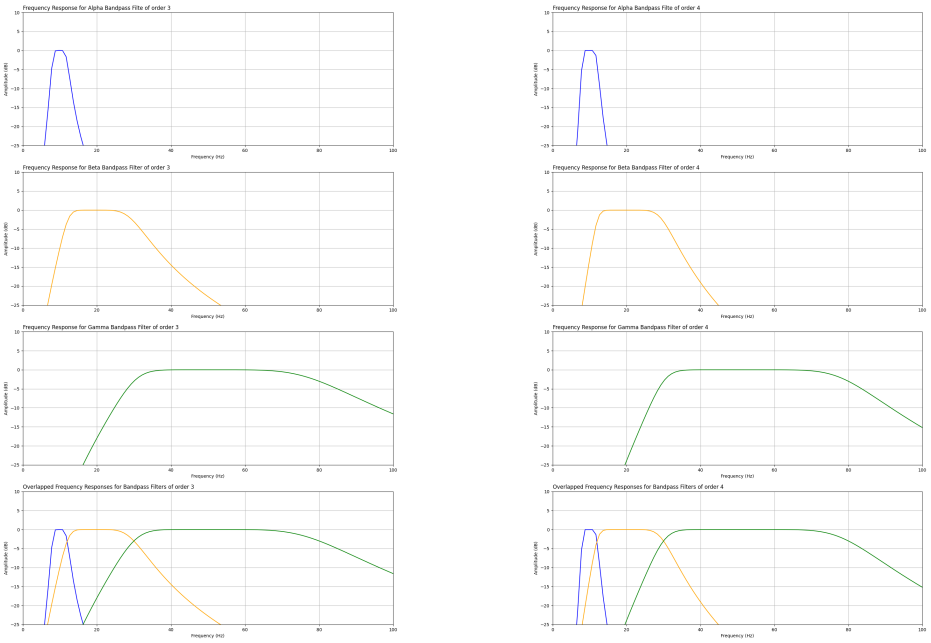
(a) Magnitude response for bandpass order 1.

(b) Magnitude response for bandpass order 2.

Figure 4.4: Magnitude response for bandpass filter based on filter order. Order 1 and 2 do not adapt well enough for our needs, not attenuating the bordering frequencies fast enough.

4.2.2 Downsampling

We originally downsampled our datasets by a factor of 5, which was the highest downsampling factor we could achieve without dropping our sampling rate (of originally 1000 Hz) under the Nyquist rate of our gamma frequency, 160 Hz. This proved to be very problematic for the autoregressive (AR) model training, so we lowered the downsampling rate to 2. However, even with the minimum downsampling factor possible, the AR model was unable to predict a RMSE of less than 0.4. Considering this, and given the minimum benefits it provided, we decided not to use the downsampling function in the final project.



(a) Magnitude response for bandpass order 3.

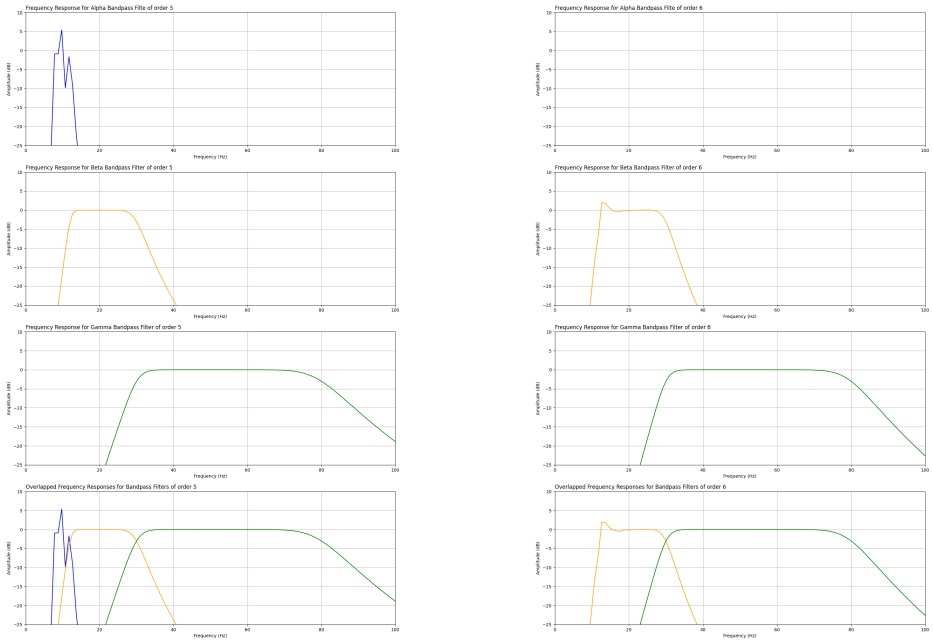
(b) Magnitude response for bandpass order 4.

Figure 4.5: Magnitude response for bandpass filter based on filter order. Order 3 and 4 adapt the best to the bandpasses.

4.2.3 Principal component analysis (PCA)

We applied the PCA technique to reduce our data dimensionality and to eliminate the channels that contribute less pertinent information (i.e., artifacts channels) to our time-series. We decided to keep only enough channels to explain 75% of all data variance, as we preferred losing data over adding confusion and noise to our datasets, which would complicate the AR training.

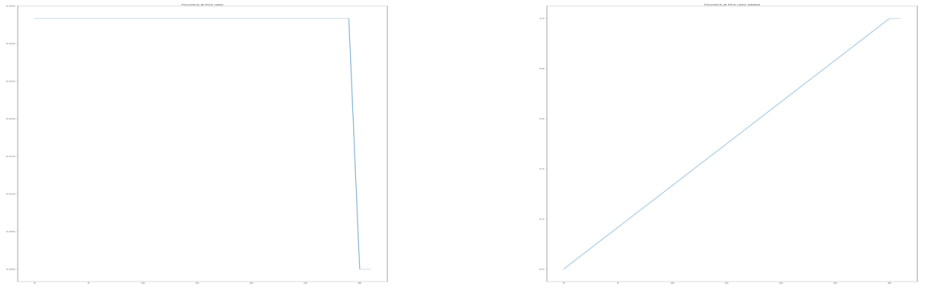
Due to the ICA application, the unfiltered data had an explanation ratio similar for each component, as expected (Figs. 4.7a and 4.7b); however, the data that passed by the bandpass filter had much different explanation ratios (Figs. 4.8a, 4.8b, 4.9a, 4.9b, 4.10a and 4.10b). This is due to the channel independence in the unfiltered data, where every channel contributes the same. This channel independence is lost in the datasets in which we applied our bandwidth filters.



(a) Magnitude response for bandpass order 5.

(b) Magnitude response for bandpass order 6.

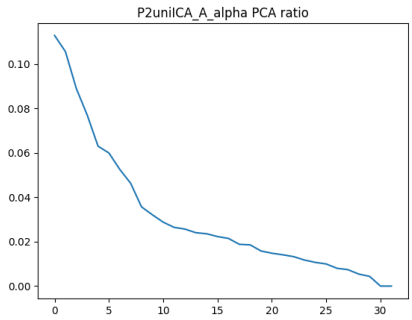
Figure 4.6: Magnitude response for bandpass filter based on filter order. Order 5 and 6 cannot properly adapt to the small size of the small alpha bandpass.



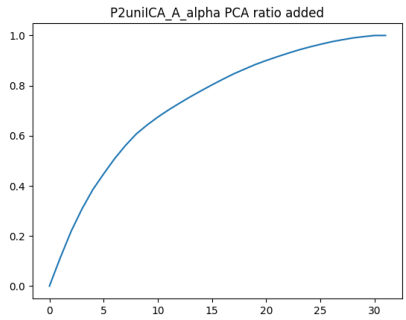
(a) Explained variance in unfiltered.

(b) Added explained variance in unfiltered.

Figure 4.7: Explained variance ratio of each channel of the unfiltered signature dataset uni_A after the ICA application, from larger to smaller, and compounded explained data variance. Each channel shows widely different ratios of explained variance.

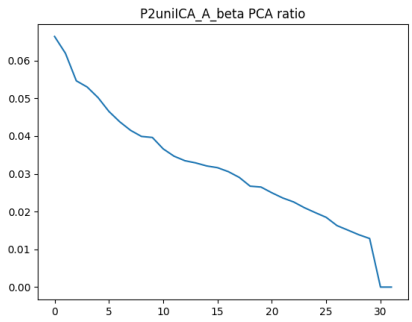


(a) Explained variance in alpha.

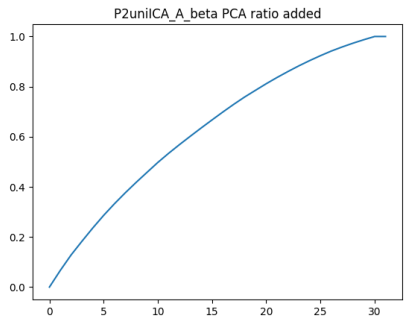


(b) Added explained variance in alpha.

Figure 4.8: Explained variance ratio of each channel of the alpha signature dataset uni_A after the ICA application, from larger to smaller, and compounded explained data variance. Each channel shows widely different ratios of explained variance.



(a) Explained variance in beta.



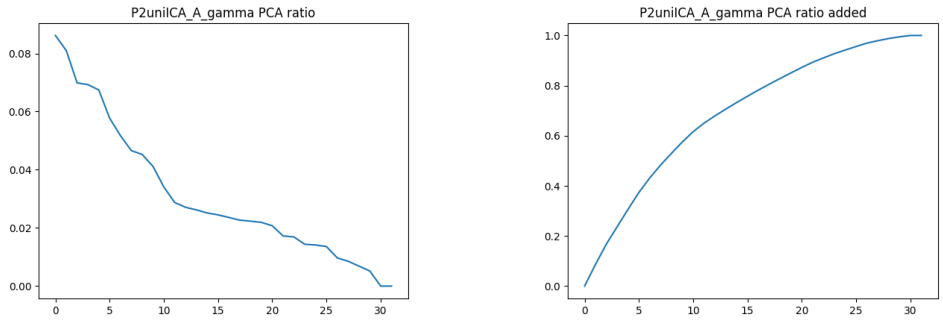
(b) Added explained variance in beta.

Figure 4.9: Explained variance ratio of each channel of the beta signature dataset uni_A after the ICA application, from larger to smaller, and compounded explained data variance. Each channel shows widely different ratios of explained variance.

4.3 The autoregressive model

4.3.1 Determining correlation between channels of the same dataset

To determine whether the channels of a dataset were correlated to one another, we applied pandas.DataFrame's corr() method. To do so, we converted each dataset to a Panda's DataFrame, which included all channels of said set. As expected based on the ICA application, we verified that all channels showed little to no correlation between themselves (Figs. 4.11a — d).



(a) Explained variance in gamma.

(b) Added explained variance in gamma.

Figure 4.10: Explained variance ratio of each channel of the gamma signature dataset uni_A after the ICA application, from larger to smaller, and compounded explained data variance. Each channel shows widely different ratios of explained variance.

4.3.2 Determinig the autocorrelation of channels

For our AR model to be functional on a time-series, the time-series values must be determined by its own past values. In order to establish that, we use autocorrelation, a statistical tool that determines how much a time-series is correlated to a lagged version of itself.

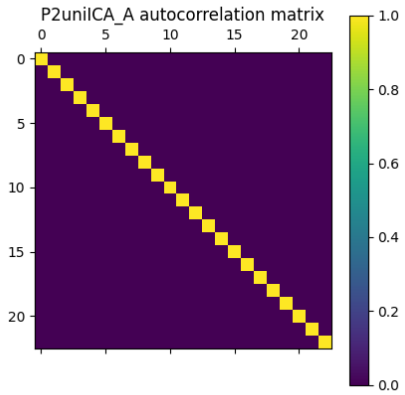
To check if the channels of our datasets were autocorrelated, we iterated through each of them, for lags between 1 and 20, using the lag_plot function from the pandas.plotting library. This tool also helps to detect outliers in our data.

As seen in Figs. 4.12 and 4.13, we found no outliers in our data. Furthermore, we could clearly observe that the correlation decreases as the lag increases.

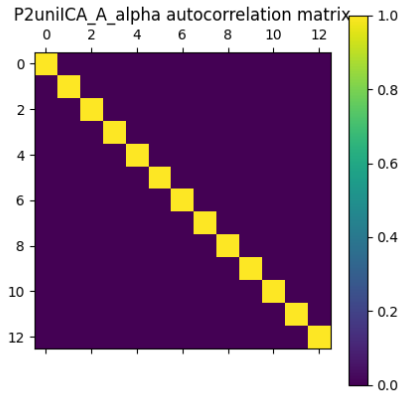
In order to get a more global view of the autocorrelation of the channels based on the lag, we originally used the more intuitive approach, as defined in Section 3.4.2; however, this method proved inefficient for such large datasets. Therefore, we decided instead to establish the autocorrelation coefficient by calculating the real part of the Fourier's transformation of normalized data to get the autocorrelation matrix, using the numpy.fft's fft method. By dividing this matrix by the data variance and length, we received the autocorrelation coefficient, which we plot with matplotlib.pyplot's stem (Figs. 4.14 and 4.15).

4.3.3 Autoregressive Model

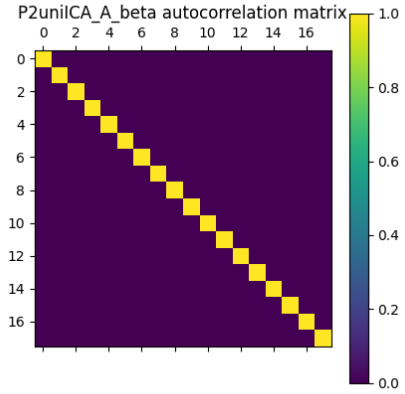
Once we knew the channels were autocorrelated, we made and fitted an AR model for each individual channel and kept increasing the lag for the model from 1 to 25. Any value over 25 took too much time to compute and was left simply as 25+; we considered that no reasonable lag could be used to predict with the indicated accuracy.



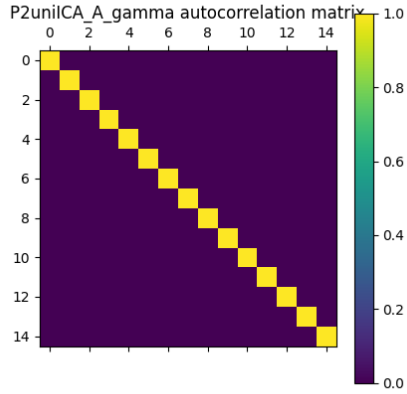
(a) Graphical matrix correlation for unfiltered.



(b) Graphical matrix correlation for alpha.



(c) Graphical matrix correlation for beta.



(d) Graphical matrix correlation for gamma.

Figure 4.11: Graphical matrix representation of the correlation between channels of the uni_A dataset, where 1 indicates perfect correlation, and 0 indicates no correlation. As expected, the channels remain independent from one another, thanks to the ICA application.

For our model prediction, we used past observed values as variables, not past predicted values. While this implies that our model can only predict data that we already know, this is not an issue for our study, as our purpose was not to predict future EEG readings per se; rather, we aimed to observe the number of past data needed to do so. Each channel held between 25 and 50 concatenated tests, with each having 12,000 data points, for a total of between 300,000 to 600,000 data points per channel. Due to this large amount of data, and our single value prediction method, we used only 2000 data points out of the 300,000 to 600,000 datapoints for the testing, as no more was considered to be needed.

As we only used between 0.003 to 0.006 of our data to test, we were able to use the rest

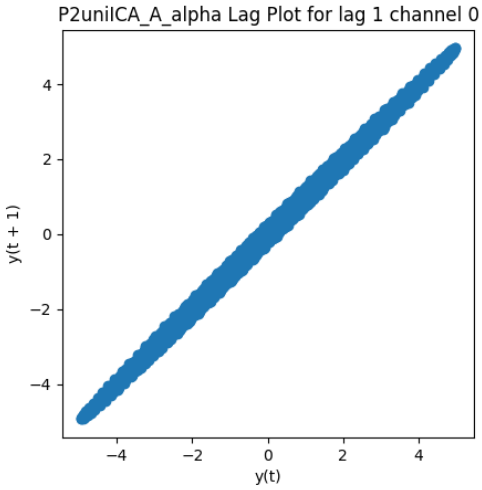


Figure 4.12: Plotted autocorrelation values for each datapoint in lag 1 for channel 0 of alpha uni_A dataset. The cohesive shape and pattern indicates a strong correlation. No stranded points (outliers) are observed.

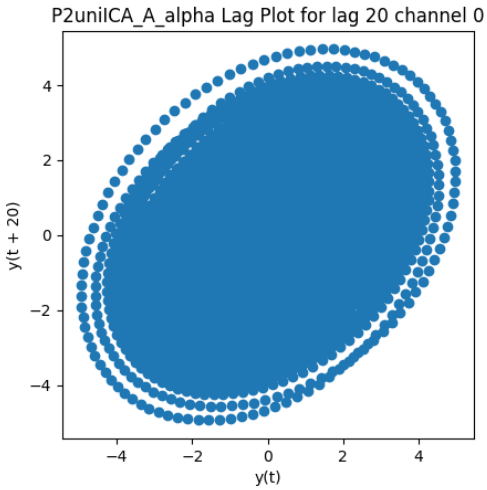


Figure 4.13: Plotted autocorrelation values for each datapoint in lag 20 for channel 0 of alpha uni_A dataset. The circular shape and pattern indicates a weak correlation. No stranded points (outliers) are observed.

of the data for the fitting process. Once we had trained our model, we extracted the fitted weights using the params attribute and manually predicted the values the model forecasted for the 2000 testing points for that specific channel (Figs. 4.16 and 4.17).

We repeated the fitting process while increasing the lag until either the RMSE value was below a threshold, which defaulted to 0.001 if none was explicitly indicated, or until we reached 25 lag. This was done for each channel separately, and the minimum number of

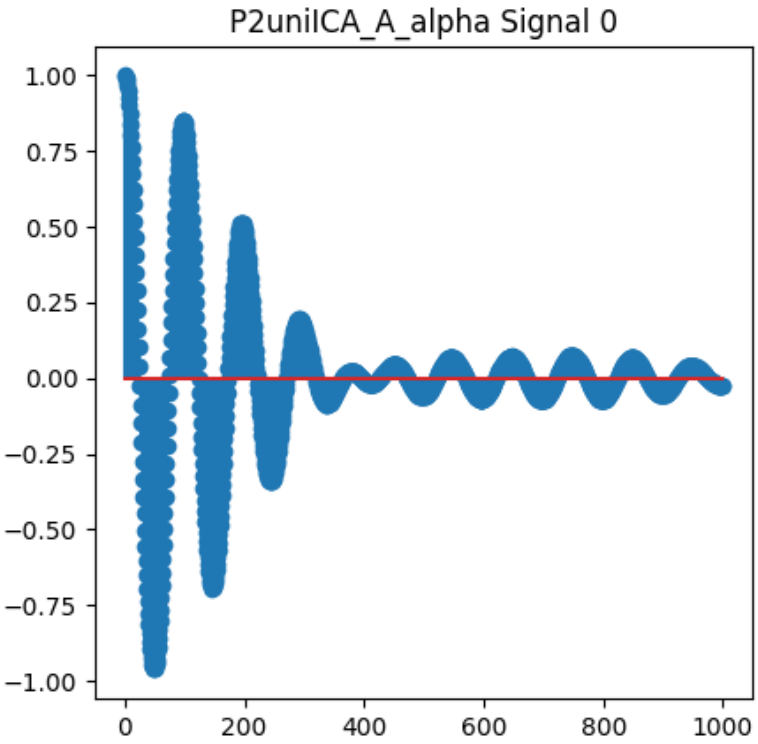


Figure 4.14: Plotted autocorrelation coefficients for lag values from 1 to 1000. We can observe the cyclical nature of the data by the waveform our autocorrelation coefficients adopts.

lags needed for that channel were saved (Fig. 4.18).

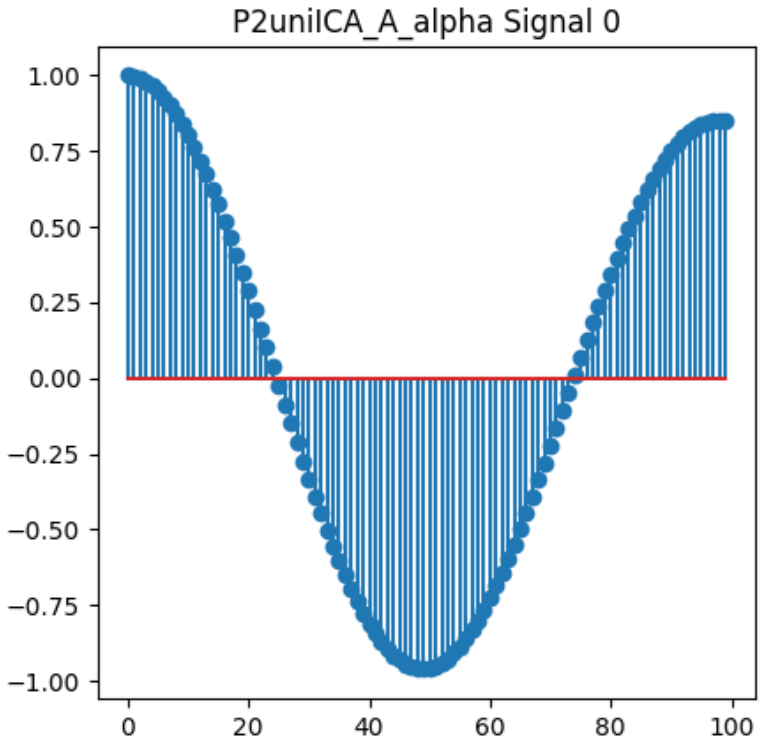


Figure 4.15: Plotted autocorrelation coefficients for lag values from 1 to 40. As observed, lag past 25 would barely be beneficial, if at all.

4.4 Quantifying time perception

Once we had trained and stored the minimum number of lags needed to predict each channel successfully, we analyzed that minimum necessary number of lags and established the mean value for each dataset. Specifically, we considered that the mean value was the amount of past information retained for the brainwaves in each dataset. The larger the amount of past data retained, the more data the brainwaves consider to be directly related to the current time, which implies a more dilated time perception.

4.4.1 Brainwave signature comparison

The most promising parameter to compare in our study was the neural brainwave signatures. Comparing the deviation in the minimal necessary lag for valid prediction by brainwave signature allowed us to establish the difference in time perception for alpha, beta and gamma brainwaves. This proved to be the most determining factor in the time retention variation, as alpha brainwaves consistently scored lower time retention, and gamma higher (Fig. 4.19).

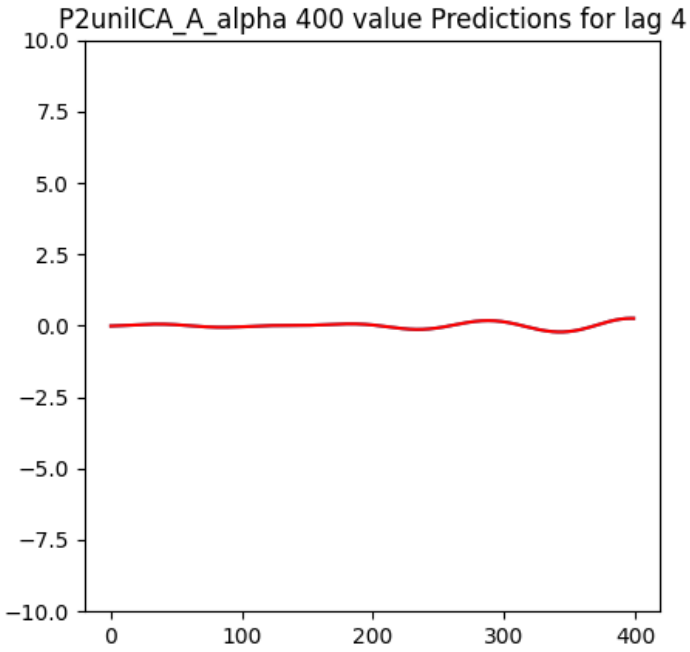


Figure 4.16: First 400 predicted values for AR model with lag 4 trained on channel 1 of alpha uni_a dataset. The red line indicates the predicted values while the blue line indicates the true values. Note that the blue line may be hard to differentiate from the red line covering it.

4.4.2 Experimental condition comparisons

The experimental conditions, which includes the metronome distinction of being either A or B and the type, uni or bidi, and condition, conflictive, or consistent, had little to no impact on time retention (Fig. 4.20). We believe this is due to the similarity of the mental, emotional and physical state the participants were in throughout all tests.

Of note, the deviation error is either null or almost null in all cases (Fig. 4.20). This signifies that the minimum necessary lag for each brainwave frequency barely varied throughout all the experimental conditions.

4.4.3 Root mean square error (RMSE) threshold value comparison

To further visualize how the past data retention varies based on brainwave frequencies, we repeated the autoregressive fitting of all datasets four times with variations of the RMSE threshold value, ranging from 1x10–3 (which is the default value) to 1x10–6 (Fig. 4.21).

Of note, the variation seems to follow an almost exponential growth, with the minimum number of lags needed for gamma increasing to 25+, as the prediction RMSE value plateaus between $1x10^{-4}$ and $1x10^{-5}$.

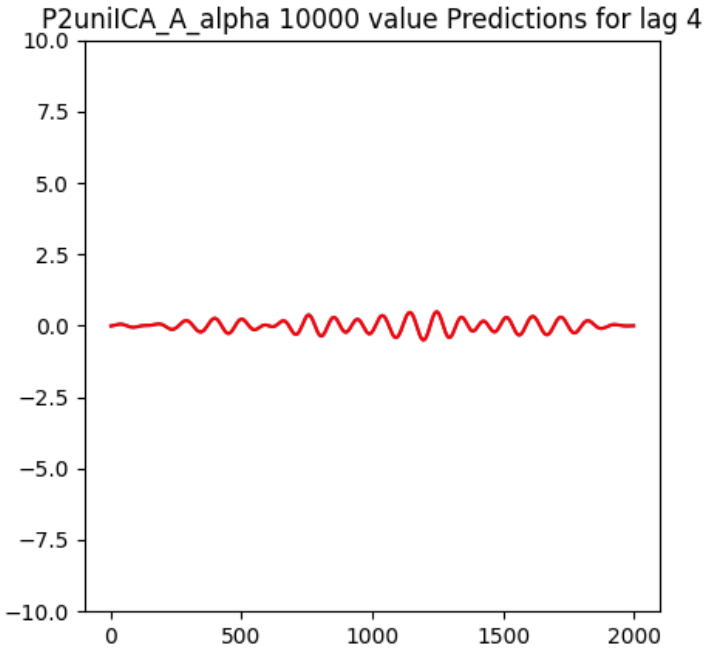


Figure 4.17: First 2000 predicted values for AR model with lag 1 trained on channel 1 of alpha uni_a dataset. The red line indicates the predicted values while the blue line indicates the true values. Note that the blue line may be hard to differentiate from the red line covering it.

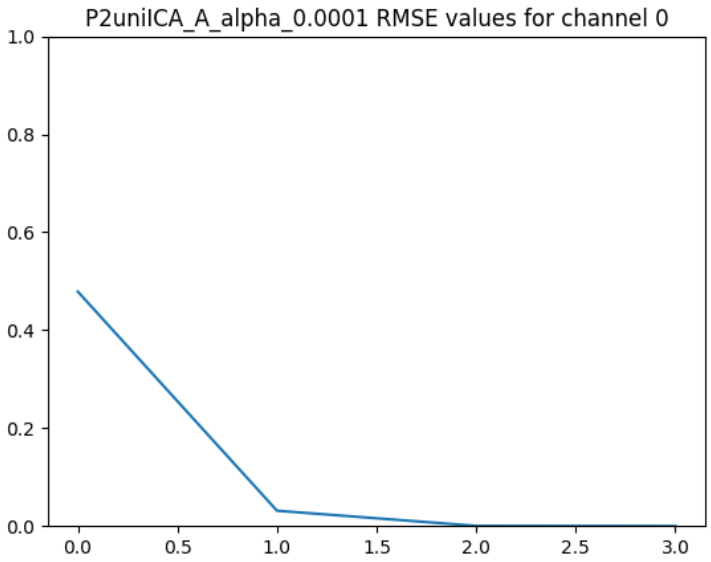


Figure 4.18: Plot of the lag values and corresponding RMSE of the AR model trained on an alpha uni_A channel. As the lag values increase, we can see the RMSE decrease. Initial RMSE values are 0.4.

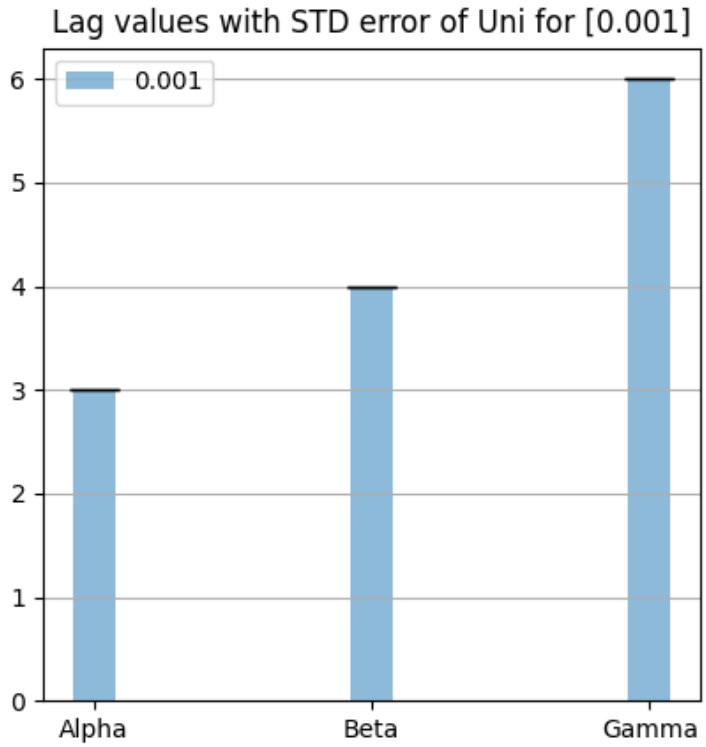


Figure 4.19: Plotted minimum necessary lag based on brainwave signature for RMSE threshold value of 0.001. As seen, alpha needs 4 minimum necessary lags, beta needs 5, and gamma needs 7 minimum necessary lags.

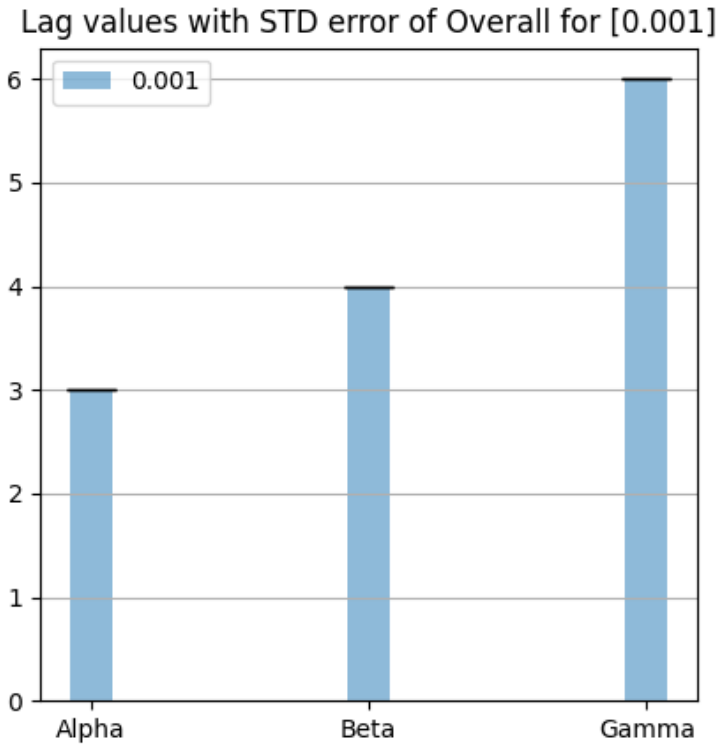


Figure 4.20: Plotted minimum necessary lag based on experimental conditions for RMSE threshold value of 0.001. As seen, minimum necessary lags do not vary much throughout uni, bidi_cons and bidi_conf.

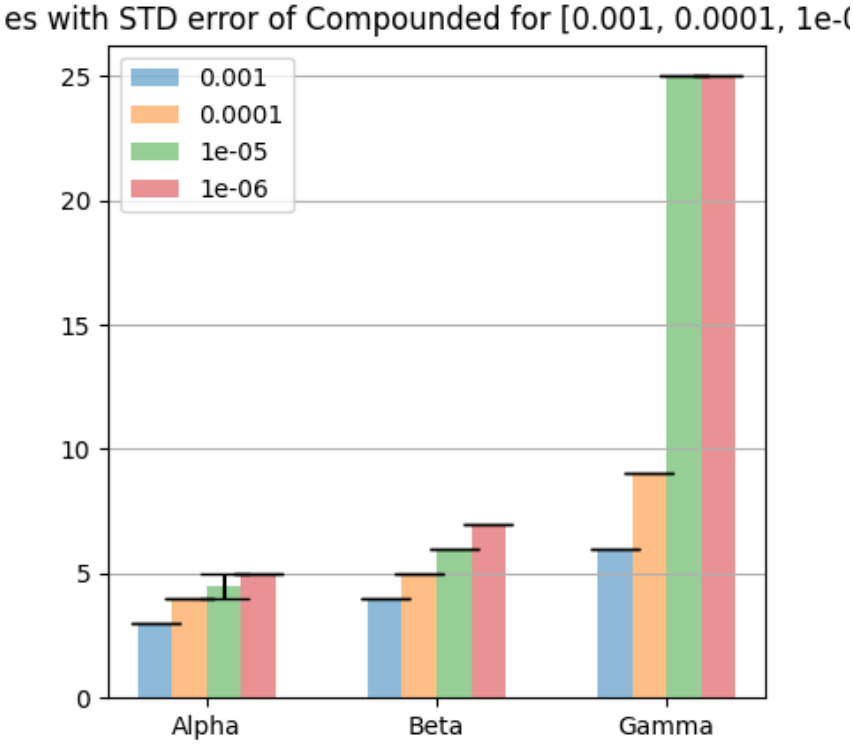


Figure 4.21: Plotted minimum necessary lag based on brainwave signatures and RMSE threshold values. The number of minimum necessary lags tend to grow exponentially, with gamma surpassing our quantifiable limit (25 lags).

Chapter 5

Conclusions

In this study, we have established a method to quantify the amount of past data that is considered by different neural brainwave signatures in a rhythm exercise. To an extent, this has allowed us to start to develop a methodology to objectively quantify variation in time perception. Specifically, we have successfully made a pipeline that, from unprocessed EEG readings, can establish the dependency of alpha, beta, and gamma signatures bands on past data, and can determine how different activities affect this time perception. This is accomplished, in our case, by filtering the data to remove powerline interference, selecting the EEG segments that correlate to the test and separating the different potential source signals using the ICA method; this was all done in MatLab.

We then switched to Python and proceeded to separate each frequency band, applied PCA to keep those signal components that are more relevant to reduce data dimensionality and data artifacts, and create a method that would determine the minimum amount of lags needed to predict each channel (or signal component) through an AR model under a threshold RMSE value. This minimum lag value will then be compared by neural brainwave signatures and external conditions.

Of note, the experimental conditions of the activity conducted (e.g., uni, bidi, bidi cons, or bidi conf) had no noticeable impact on our results (see Section 3.1.3). In stark contrast, the brainwave signature had a strong influence on the minimum lag needed to successfully predict each time series (see Figs. 5.1, 5.2, 5.3 and 5.4).

It should be emphasized that the past data retention displays the amount of information that a brainwave still considers relevant to the current time, with larger amounts of retained data indicating longer time-periods in which past data is still considered relevant for the current situation. We take this time-period to be the "present" time of each brainwave. Therefore, when we refer to a brainwave time perception variation, we are referring to this time period expanding or contracting.

From this, we concluded that gamma brainwaves showed a greater past data retention, followed by beta and then alpha. This seems to indicate that frequency bands that are more related to active states are more influenced by older time data then frequency bands related to relaxation, and that these gamma brainwaves therefore have a more dilated time perception.

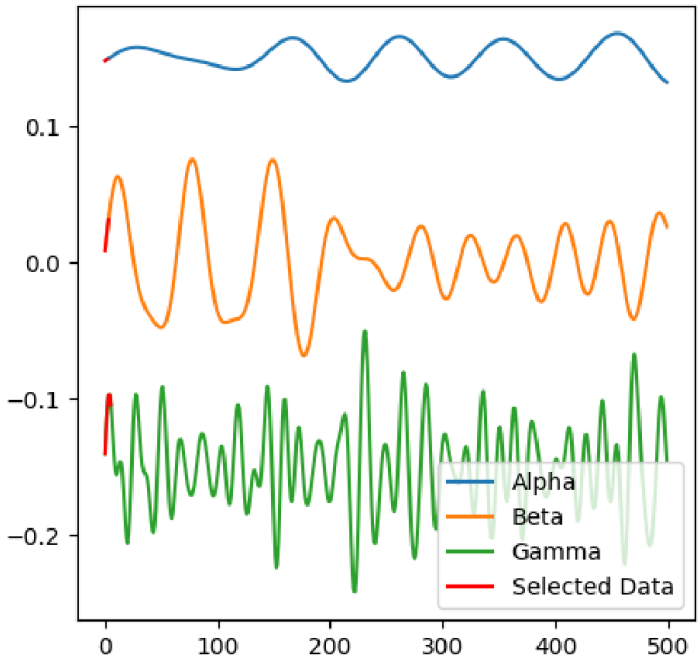


Figure 5.1: Visual representation of the amount of data retained by each brainwave signature in RMSE 0.001.

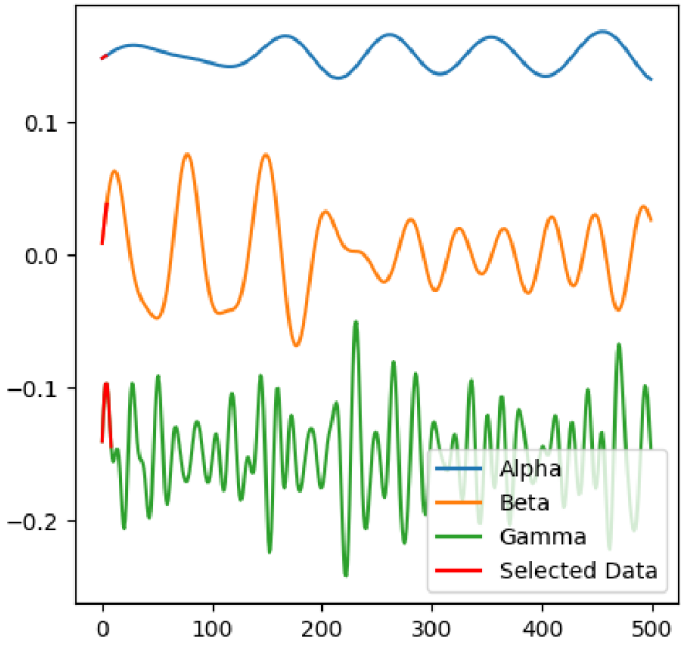


Figure 5.2: Visual representation of the amount of data retained by each brainwave signature in RMSE 0.0001.

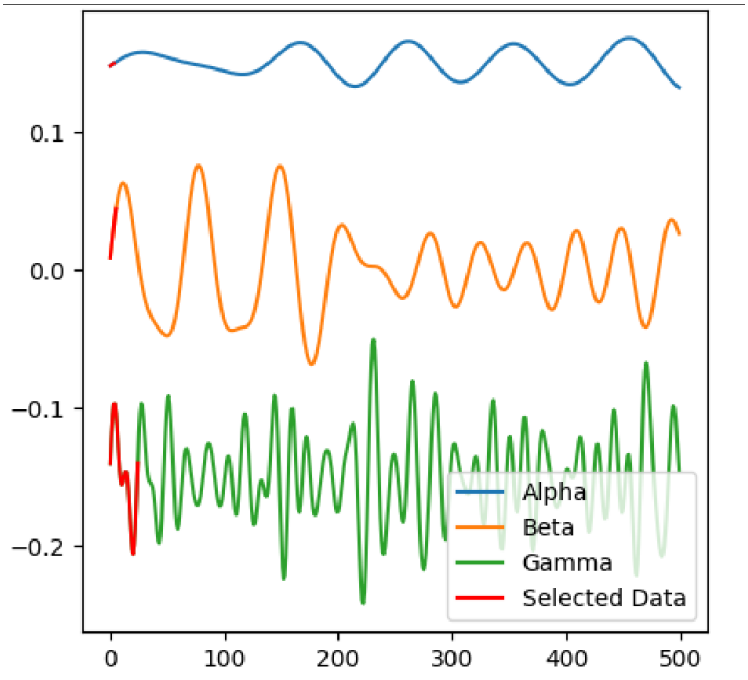


Figure 5.3: Visual representation of the amount of data retained by each brainwave signature in RMSE 10-5.

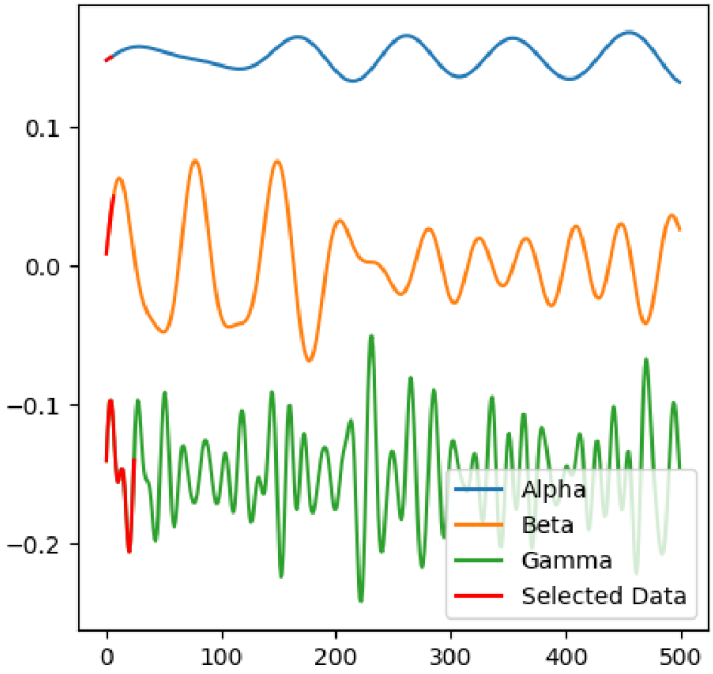


Figure 5.4: Visual representation of the amount of data retained by each brainwave signature in RMSE 10-6.

Chapter 6

Future work

6.1 Frequency normalization

A main concern throughout this study was the possibility that higher lag values were not do to higher time retention of the frequency bands, but due solely to the more complex nature of higher frequency data, and the necessity of more data points for successful prediction that it causes.

In future work, it may be interesting to try to equalize all frequencies by artificially decreasing the beta and gamma frequency to match the alpha waves number of datapoints per wave cycle, and see if the minimum lag number is affected.

6.2 Larger activity comparison

One of the stated goals of this study was to produce a methodology that may help establish a more objective approach to quantifying time perception variation. In this study, the different activities that the participants were taking part of, be it finger-tapping alone, in conflicting rhythm pairs or consistent rhythm pairs, had no appreciable effect to the necessary minimum lags.

We believe this might have been due to the similarity of each activity, and the low difference in emotional or physical state. It may prove that with a different variety of activities, this may help in studying the perception of time, as altered by emotional states (Buetti and Lleras, 2012), physical state (Tamm *et al.* 2015), or substances (Yanakieva *et al.*, 2019).

It may also be interesting to quantify in such a manner the effects of time perception altering substances such as stimulants, depressants or hallucinogens, as it would give a more objective measure.

Programs used

The MathWorks, Inc. (2022). MATLAB version: 9.13.0 (R2022b). Available at: https://www.mathworks.com

JetBrains, (2017). PyCharm version 2022.1. Available at: https://www.jetbrains.com/pycharm/whatsnew/2022-1/

Bibliography

[1] Battleday RM, Peterson JC, Griffiths TL (2021). From convolutional neural networks to models of higher-level cognition (and back again). *Annals of the New York Academy of Sciences*, 1505(1), 55-78.

[2] Benau EM Atchley RA (2020). Time flies faster when you're feeling blue: sad mood induction accelerates the perception of time in a temporal judgment task. *Cognitive Processing, 21(3)*, 479-491.

[3] Beste C, Münchau A, Frings C (2023). Towards a systematization of brain oscillatory activity in actions. *Commun*. Biol. 6(1), 137.

[4] Buetti S Lleras A (2012). Perceiving control over aversive and fearful events can alter how we experience those events: an investigation of time perception in spider-fearful individuals. *Frontiers Psychol.*, 3, 337. https://www.frontiersin.org/articles/10.3389/fpsyg.2012.00337/full

[5] Buzsaki G, Draguhn A (2004). Neuronal oscillations in cortical networks. *Science*, 304(5679), 1926-1929.

[6] Cohen MX (2017). Where does EEG come from and what does it mean? *Trends Neurosciences*, 40(4), 208-218.

[7] Damsma A, Schlichting N van Rijn H (2021). Temporal context actively shapes EEG signatures of time perception. *J Neuroscience*, 41(20), 4514-4523.

[8] Esghaei M, Treue S, Vidyasagar TR (2022). Dynamic coupling of oscillatory neural activity and its roles in visual attention. *Trends Neurosci.*

[9] Frings C, Hommel B, Koch I, Rothermund K, Dignath D, Giesen C, ... Philipp A (2020). Binding and retrieval in action control (BRAC). *Trends Cogn. Sci.*, 24(5), 375-387.

[10] Heggli OA, Konvalinka I, Cabral J, Brattico E, Kringelbach ML, Vuust P (2021). Transient brain networks underlying interpersonal strategies during synchronized action. *Soc. Cogn. Affect. Neurosci.* 16(1-2), 19-30.

[11] Jaipriya D Sriharipriya KC (2023). Brain Computer Interface-Based Signal Processing Techniques for Feature Extraction and Classification of Motor Imagery Using EEG: A Literature Review. *Biomedical Materials Devices*, 1-13.

[12] Kriegeskorte N Douglas PK (2018). Cognitive computational neuroscience. *Nature Neuroscience*, 21(9), 1148-1160.

[13] Lake JI, LaBar KS Meck WH (2016). Emotional modulation of interval timing and time perception. *Neuroscience Biobehavioral Reviews*, 64, 403-420.

[14] Marinho V, Pinto GR, Bandeira J, Oliveira T, Carvalho V, Rocha K, ... Teixeira S (2019). Impaired decision-making and time perception in individuals with stroke: Behavioral and neural correlates. *Revue Neurologique*, 175(6), 367-376.

[15] Pardey J, Roberts S Tarassenko L (1996). A review of parametric modelling techniques for EEG analysis. *Medical Engineering Physics*, 18(1), 2-11. https://doi.org/10.1016/1350-4533(95)00024-0.

[16] Sadaghiani S, Brookes MJ, Baillet S (2022). Connectomics of human electrophysiology. *NeuroImage*, 247, 118788.

[17] Siegel M, Donner TH Engel AK (2012). Spectral fingerprints of large-scale neuronal interactions. *Nat Rev Neuroscience*, 13(2), 121-134.

[18] Surakhi O, Zaidan MA, Fung PL, Hossein Motlagh N, Serhan S, AlKhanafseh M ... Hussein T (2021). Time-lag selection for time-series forecasting using neural network and heuristic algorithm. *Electronics*, 10(20), 2518. https://www.mdpi.com/2079-9292/10/20/2518

[19] Tamm M, Jakobson A, Havik M, Timpmann S, Burk A, Ööpik V, ... Kreegipuu K (2015). Effects of heat acclimation on time perception. *Int J Psychophysiol*. 95(3), 261-269. https://doi.org/10.1016/j.ijpsycho.2014.11.004.

[20] Yanakieva S, Polychroni N, Family N, Williams LT, Luke DP Terhune DB. (2019) The effects of microdose LSD on time perception: a randomised, double-blind, placebo-controlled trial. *Psychopharmacology*. 236, 1159–1170. https://doi.org/10.1007/s00213-018-5119-x

Annex

Due to the large number of files created from the study, we have chosen to publish them in a zip folder, named data.zip, available through OneDrive in the following link:

Link to Data.zip

This folder contains all the files generated by both the the MatLab software described in the data preprocessing phase (Section 3.2 and Section 4.1), and the files generated with the PyChram software in the data processing phase and the autoregressive model phase (Section 3.3, Section 3.4, Section 4.2 and Section 4.3).