Imperial College London

MEng Individual Project

Imperial College London

Department of Computing

Decoding Brain Signals to Assess the Perception of Music in the Psychedelic State

Supervisor:
Dr. Konstantinos Gkoutzis

Author:
Naman Wahi

External Supervisor:
Dr. Fernando Rosas

Second Marker:
Dr. Thomas Lancaster

June 15, 2020

Abstract

Psychedelics are a class of drugs which are known to induce an altered state of consciousness called the "psychedelic state". While often taken for recreational usage (despite their typically illegal status), much research has been conducted on their therapeutic uses. Research in psychedelic therapy has shown a lot of promise in the treatment of mental illnesses such as anxiety and depression, and music has shown to be an important contextual factor in psychedelic therapy. Therefore, a better understanding of the relationship between music and psychedelics could impact the way that psychedelic therapy is carried out.

In this project, we assess how the perception of music changes in the psychedelic state. More specifically, under the effect of LSD - the prototypical psychedelic. We will analyse a dataset with data from 13 test subjects. For each test subject, we have recordings of their brain signals both in and out of the psychedelic state. We also have the music the subjects were listening to while these brain signals were recorded.

Neural decoding is the process by which a representation of external stimuli (in this case the audio the subject was listening to) can be reconstructed from brain signals. How well the music stimulus can be decoded acts as a measure of similarity between the brain signals and the music. This similarity score acts as a proxy metric for the degree that the subject is perceiving the music.

In this project, linear methods are used to decode different representations of music. An artificial neural network developed by the Magenta project[1], originally intended to assist in the creative development of music, was re-purposed to serve as our music representation to decode from brain signals. This representation was decoded successfully using principal component regression. Although the LSD seemed to improve the decoding performance, this change was shown not to be statistically significant.

Although no significant effect of LSD was found, we have established an end-to-end procedure for decoder training, decoder evaluation, and assessing the effects of LSD. Furthermore, we have shown the limitations of linear decoding methods for various other failed decoding attempts.

Acknowledgements

First and foremost, I would like to thank my project supervisor Dr Konstantinos Gkoutzis for his continued support throughout the project. Despite this project being an external project, offered by the Department of Brain Sciences, he has shown great enthusiasm diving into the realm of neuroscience and psychedelic research alongside me.

Secondly, I would like to thank my external supervisor, Dr Fernando Rosas, from the Department of Brain Sciences for guiding me through the world of psychedelic research.

Special thanks to Dr Pedro Mediano for his excellent suggestions throughout the project.

Many thanks to Dr Thomas Lancaster for insights on how to make this project more accessible to those unfamiliar with psychedelic research.

Last but not least, I would like to thank my family and friends for supporting me throughout the year. Not just my friends and flatmates at university, but also my friends back home (𝖈𝖌).

Notes on the Title

Some may notice that this project was originally titled: “Assessing the Brain’s Decoding Capability of Musical Features in the Psychedelic State". The change in title was not a pivot in the direction of the project; rather, it was a way to more clearly convey the nature of this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

Psychedelics are a class of drugs which are known for their ability to alter consciousness in what is known as the “psychedelic state"[2]. Lysergic acid diethylamide (LSD), the prototypical psychedelic drug[2], will be the focus of this project; although, the effects of different psychedelics are very similar[3]. While psychedelics are often taken for recreational purposes[4], much research has been conducted on their therapeutic uses.

In 1966, the United States began the prohibition of psychedelics and the curtailment of their research; however, thanks to popular counterculture movements, their social impact has seen a revival[2]. What is known as the “Second Wave" of psychedelic research has shown that these drugs show promise for the treatment of mental illness; for example, the treatment of obsessive compulsive disorder (OCD), anxiety and depression in those with life-threatening conditions, major depressive disorder (MDD) and alcohol dependence (see [2] and its references). The investigation into the use of psychedelics in mental health treatment is one of the main aims of Imperial College's Centre for Psychedelic Research.

Research has indicated that the effects of psychedelics are very sensitive to contextual factors; as a result, they should be considered in a therapeutic context[5]. Contextual factors refer to the environment that the test subject experiences. Music has shown to be an important contextual factor in psychedelic therapy[3].

The Centre for Psychedelic Research's second aim is to study the way psychedelics affect conscious experiences as a way to understand consciousness as a whole. The effects of psychedelics on the brain include the increase in global connectivity[6] and diversity or randomness of brain signals[7]. The latter relates to the “Entropic brain hypothesis" which posits that the quality of a psychedelic experience relates to the randomness (or entropy) of brain activity[8]. These studies indicate that the brain in the psychedelic state is more interconnected, less constrained and less ordered.

While the aforementioned relaxed and disordered properties of the brain in the psychedelic state seem to be negative, there are many positive properties reported from those in a psychedelic state, including a heightened sense of perception and a sense of meaningfulness. Furthermore, these positive experiences in the psychedelic state appear to be correlated to the effectiveness of psychedelic therapy[9]. A better understanding of the experiences of music in a psychedelic state may play an important role in the design of psychedelic therapy.

1.2 Problem Setting and Objectives

The main aim of this project is to assess how the perception of music changes in the psychedelic state - more specifically under the influence of LSD. As mentioned above, music has been shown to be an important factor in psychedelic therapy; consequently, the results of this analysis could influence the way that psychedelic therapy incorporates music into the environment.

This project is in collaboration with Imperial's Centre for Psychedelic Research who will be providing the dataset that will be analysed. Those affiliated with the Centre will be referred to as the "client(s)" of the project. They will provide the necessary insights and suggestions in the domain of neuroscience (pertaining to the brain).

The dataset we will analyse consists of 13 test subjects. For each subject, we have two recordings of their brain signals. One of these recordings is of the subject under the effect of LSD i.e. in the psychedelic state. The other is of the subject not in the psychedelic state. The subjects were listening to music while these brain signals were being measured. These songs were provided alongside the brain signal data.

In this project's context, neural decoding refers to the reconstruction of the audio the subject was listening to from their brain signals. Research suggests that the ability to decode a specific sounds serves as a measure of attention the subject is paying to the sound[10]. More specifically, if a subject is paying more attention to a particular sound, then it is possible to decode the sound more accurately. The decoding accuracy serves as a measure of similarity between the subject's brain signals and the sounds they were listening to. In this project, this notion of similarity acts as a proxy metric of the degree to which the subject is experiencing/perceiving the music.

The research objectives of this project are:

- Adapting existing methods to establish a decoding model which can successfully reconstruct a representation of the music from the brain signals.
- Given that a decoding model is found, assess the degree to which LSD affects the decoding performance. The decoding performance being a proxy metric for the degree to which the subject is perceiving the music (or specific musical features).

The secondary objective is the software itself which can be used for this research. It should be a toolbox with reusable components which can be extended upon for further research.

1.3 Contributions

The contributions of this project are as follows:

- The methods from [10], showing that decoding can be used as a measure of similarity, are first reproduced and validated on a public dataset in Chapter 3. In Chapter 4, these methods were then shown not to generalise well to the dataset provided by the Centre for Psychedelic Research. These results indicate that while this method of audio decoding may have been effective for speech (as was done in [10]), it may not be suitable for music (as is done in this project).
- In Chapter 5, a novel decoding model is proposed. In this part of the project, we re-purpose a neural network-based representation of music, originally designed for the creative process of music development, in our decoding model. This representation of the music was successfully decoded compared to a baseline that was established.

  When analysing the effect that LSD had on the decoding performance, no statistically significant change was found. Nonetheless, an end-to-end procedure for creating a decoding model, evaluating its performance, and analysing the effect of LSD have been established. This procedure can be modified and re-used for future research.
- In Chapter 6, we attempt to decode short-term acoustic features from the music, most of which correspond to the timbral properties of a musical sound (a sound's characteristics). These specific features are said to affect the perceptual experience of music.

  No decoding model was found that could successfully decode these features. These results should be kept in consideration when attempting to decode these features in future research if that is the chosen direction.

Chapter 2

Background

2.1 Psychedelics and Music

In this section, we will introduce some details about psychedelic drugs and their relation to music. This section will serve as an introduction to the aspects of this project beyond the scope of Computing.

2.1.1 What are Psychedelics?

As mentioned in the introduction, psychedelics are a class of drugs which can induce an altered state of consciousness known as a psychedelic experience. The term psychedelics was coined by psychiatrist Humphrey Osmond in 1956 as a way to create a more inclusive term to describe the effects of psychedelic drugs: not only to include the induced psychosis-like effect and hallucinations, but also to include experiences with aesthetic, meaningful and spiritual significance[3].

2.1.2 Lysergic acid diethylamide (LSD)

Lysergic acid diethylamide (LSD) falls under the category of psychedelic drugs and can induce the psychedelic experience effects described above. The dataset from the Centre for Psychedelic Research that will be analysed in this project consists of subjects under the effects of LSD.

2.1.3 Relation to this Project

The aim of this project is to assess the perceptual experience of music in the psychedelic state (induced by LSD). This will be done by measuring the similarity between the brain signals and the music, this similarity measure being a proxy metric for how much the subject is perceiving the music. This degree of perception could be to the music in its entirety i.e. a general representation capturing the most informative musical features of the music; however, this perception can also be measured to specifically chosen musical features such as its roughness.

2.1.4 Terminology

Here we will cover some common terminology in neuroscience which will be used throughout the report

1. Placebo - A placebo is a substance with no therapeutic effect that is given to the subject. It acts as a control to see the effects that LSD has on the brain in comparison.

2. Condition - In this project, the condition is whether the subject has taken LSD or a placebo (referred to as the "LSD condition" and "placebo condition" respectively).

3. Stimulus - The stimulus is the external factor presented to the subject. In this project, the stimulus will typically mean the music from the dataset provided by the Centre for Psychedelic Research. However, we will look at audio speech stimuli when examining existing research.

4. p-value (denoted p) - Because of the research nature of this project, we will often want to test whether results are statistically significant via a statistical hypothesis test. Assume we have a null hypothesis, which is typically the default position stating there is no difference between groups of data or no association between two variables[11]. The p-value is the probability of our observations, assuming the null hypothesis is true[11]. If the p-value is lower than a certain threshold, the results are considered statistically significant. In this project, this threshold will be $p = 0.05$ (as requested by the client).

   An example null hypothesis could be: "The similarity of the brain signals and the music is the same under the LSD and placebo condition". If looking at our analysis results, we find that $p < 0.05$, then our results show that LSD has a statistically significant effect.

   Often times, the null hypothesis is implicit and the p-value is simply stated. For example, if it is stated that: "Method X outperformed method Y, this is shown to be statistically significant (Wilcoxon signed-rank test, $p = 0.03 < 0.05$)". The implicit null hypothesis is that method X does not outperform method Y. The choice of specific statistical hypothesis tests was the Wilcoxon signed-rank test, and this test yielded a significant p-value of 0.03.

2.2 Neural Decoding

In this project, we will be looking at neural decoding to assess the relationship between the brain signals and the music stimulus. Neural decoding can be described as a reconstruction of external stimuli from brain activity[12]. In this project, the external stimuli refer to audio, specifically music, that is being listened to. See Figure 2.1 for a visualisation of audio stimulus decoding. In this project, we construct various decoding models to describe this relationship between brain activity and the musical stimuli. This model can be described as a function f. The input to this function,

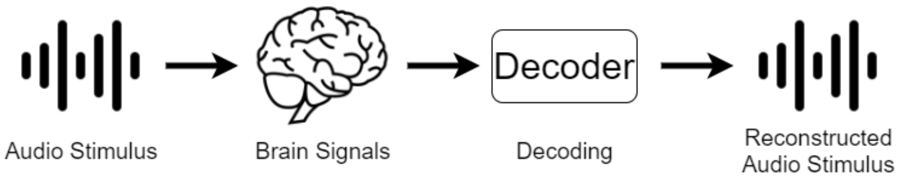


Figure 2.1: Decoding an audio stimulus.

$\boldsymbol{x}$, is a vector representation (or "features") of the brain activity. The output of the function, $f(\boldsymbol{x})$, is the reconstructed representation of the stimuli.

We will now look at some of the decoding models we will use in this project. The specific inputs and outputs of the model will not be specified as of now. What is important is the motivations behind choosing these models.

For the following methods, assume the brain signal features $\boldsymbol{x}$ is an m dimensional vector of real numbers ($\boldsymbol{x} \in \mathbb{R}^m$). For simplicity, assume the reconstructed stimulus output y is a single real number ($y \in \mathbb{R}$); the methods described, however, can generalise to the case where y is also a vector (denoted $\boldsymbol{y}$).

These models will be fit onto some data. Assume we have n input-output pairs of the form $(\boldsymbol{x}_i, y_i)$ where i denotes the ith pair. For the given model, f, let us denote its output (or "prediction") as $\hat{y}$.

2.2.1 Ordinary Least Squares Regression

The simplest model we will look at is ordinary least squares linear regression. In this model, the output is a weighted (linear) combination of the input features. More formally, given input features $\boldsymbol{x} = (x_1, x_2, ..., x_m) \in \mathbb{R}^m$, the output $\hat{y} \in \mathbb{R}$ for the model f is:

$$\hat{y} = f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + ... + w_m x_m \tag{2.1}$$

The vector $\boldsymbol{w} = (w_0, w_1, ..., w_m) \in \mathbb{R}^{m+1}$ are the weights/coefficients of this model. We want to find $\boldsymbol{w}$ to fit the observed data. More formally, we want to minimise the squared differences between the predictions and the observed values, hence the name ordinary least squares. Therefore we want to find $\boldsymbol{w}$ that minimises these squared differences. The loss function we wish to minimise is:

$$L(\boldsymbol{w}) = \sum_{i=0}^{n} (f(\boldsymbol{x}_i) - y_i)^2 \tag{2.2}$$

2.2.2 Ridge Regression

Ridge regression uses the same underlying model as ordinary least squares regression (Equation 2.1); however, the loss function (Equation 2.2) is modified by adding another term. The loss function for ridge regression is:

$$L(\boldsymbol{w}) = (\sum_{i=0}^{n} (f(\boldsymbol{x_i}) - y_i)^2) + \alpha \|\boldsymbol{w}\|_2^2 \tag{2.3}$$

This additional term, $\alpha \|\boldsymbol{w}\|_2^2$, is Euclidean norm of the coefficients in $\boldsymbol{w}$ (a measure of how large these coefficients are). This term penalises coefficients in $\boldsymbol{w}$ which are too large. The α parameter controls to what extent this penalty is applied. A higher α will force the coefficients to be smaller and vice-versa.

Ridge regression is designed to deal with multicollinearity of the inputs. This is where many of the features (dimensions) of $\boldsymbol{x}$ are highly correlated with each other[11]. This redundant information between the input features makes the model prone to overfitting (Section 2.3.2) where the model does not perform well on data it has not seen before. The stronger the regularisation (and thus smaller the coefficients in $\boldsymbol{w}$), the more robust the model is to the downsides of multicollinearity.

2.2.3 Principal Component Regression

Principal component analysis (PCA) is a dimensionality reduction technique. It transforms our m-dimensional data to an m'-dimensional space where $m' < m$ while attempting to preserve as much information about the data as possible. PCA works by transforming the data into uncorrelated variables called principal components[13]. The top m' principal components which explain the most variance in the data (informally, "contain the most information") are kept. The PCA transform itself is a linear combination of the m original features (implemented as a matrix multiplication).

Principal component regression (PCR) works by first applying PCA to the given data to reduce its dimensionality to m' from m before applying ordinary least squares linear regression.

Let PCA_d denote a function which performs the PCA transformation to reduce to dimensionality to size d. We have the following model:

$$\hat{y} = f(\boldsymbol{x'}) = w_0 + w_1 x'_1 + x_2 x'_2 + ... + w_{m'} x'_{m'} \tag{2.4}$$

Where $\boldsymbol{x'} = \text{PCA}_{m'}(\boldsymbol{x})$. The loss function we wish to minimise therefore becomes:

$$L(\boldsymbol{w}) = \sum_{i=0}^{n} (f(\boldsymbol{x'}_i) - y_i)^2 \tag{2.5}$$

Because the principal components (features of the transformed data) are uncorrelated, PCR is another way to address multicollinearity issue which ridge regression (2.2.2) aims to solve.

2.2.4 Partial Least Squares Regression

Partial least squares regression (PLS regression) is an extension to principal component regression. Partial least squares regression works by transforming both the inputs ($\boldsymbol{x}_i$'s) and outputs (y_i's) into a lower-dimensional space rather than just the inputs as was the case with PCR. Instead of finding a lower-dimensional space which maximises the input variance, this new space is chosen to maximise the correlation between the inputs and outputs[14]. PLS regression is another way to deal with the problem of multicollinearity.

2.3 Decoding Model Evaluation

In this section, we will outline some general statistical methods that will be used to evaluate the decoding models.

2.3.1 Metrics

Various metrics will be used throughout this project. For the following metrics, assume we have some model which is predicting some observed data points. Assume we have a dataset of n observation. Each observation is a pair in the form (x_i, y_i) for $i = 1...n$. The independent variable x is the input to the model, and the model predicts y, the dependent variable. For simplicity, assume that the data points are real numbers.

For a given input, x_i, the output of the model is $f(x_i) = \hat{y}_i$.

Mean Squared Error

The mean squared error is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2$$

It is simply the average of the squared distances between the predicted and observed outputs. The mean squared error has a minimum value of 0 when the predictions are perfect i.e. $\hat{y}_i = y_i$ for all $i = 1...n$. A low mean squared error is desired when evaluating a given model's performance.

The Coefficient of Determination, R^2

The coefficient of determination, commonly referred to as the R^2 score is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

where $\bar{y}$ is the mean of the outputs $y_1, y_2, ...y_n$.

The R^2 metric represents the proportion of the variation in y explained by the variation in x[11]. Informally, a high R^2 value indicates that changes in y are explained a significant amount by changes in x; thus, for any given model, a high R^2 value is desired.

The maximum value of R^2 is 1. A model which disregards the input x entirely and always predicts the mean value of the output y will have an R^2 score of 0. The R^2 can be negative if the model is arbitrarily worse than predicted the mean.

Consider the case where a model is created to predict the risk factor of a disease given the someone's age. If age was a high-risk factor, we would expect the model's R^2 scores to be high. However, if other factors were more important, such as diet, we would expect the model's R^2 score to be low.

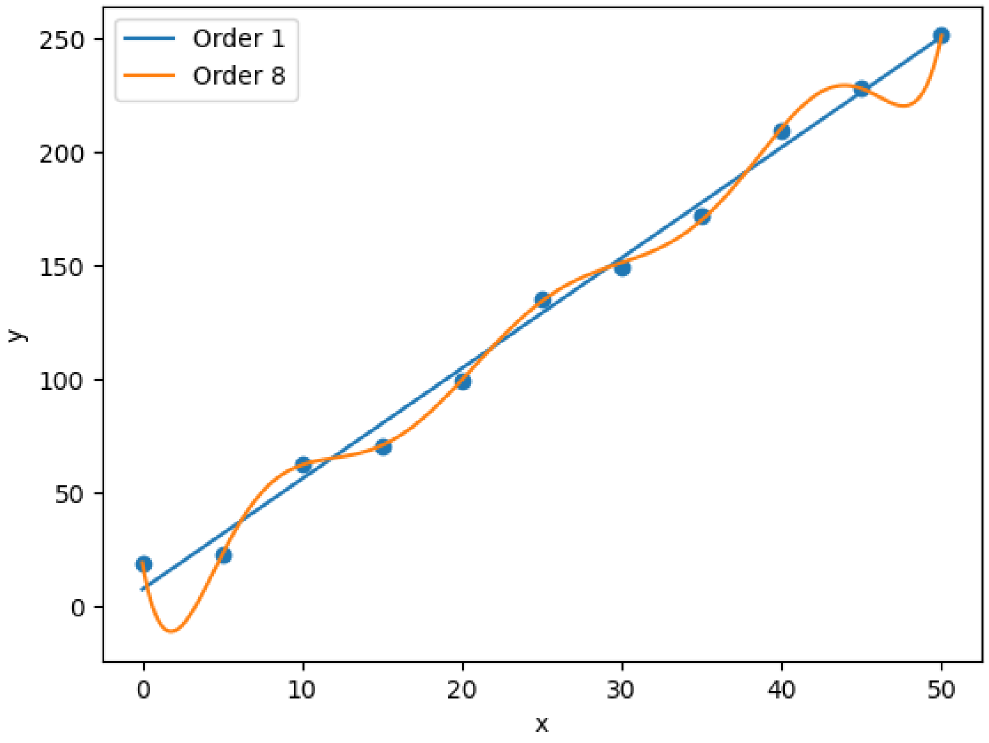


Figure 2.2: An example of overfitting. The data points appear to follow a linear relationship with some noise. The polynomial of order 1 model seems to be a reasonable fit for this data. The polynomial of order 8, a more complex model, passes nearly perfectly through every data point. There is not enough data to justify this complex model and it will likely perform poorly at predicting unseen data. What has likely happened is that the complex model has fit to some noise rather than the underlying linear relationship between x and y.

Pearson Correlation Coefficient

The Pearson correlation coefficient measures whether there is any linear correlation between the observed output y and predicted output $\hat{y}$. It is defined (for a sample) as[11]:

$$r = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{\hat{y}})^2}}$$

Where $\bar{y}$ and $\bar{\hat{y}}$ represent the means of the observed and predicted values of the dependent variable y respectively.

2.3.2 Model Evaluation

Overfitting

Overfitting occurs when a model performs very well on data it has seen, but doesn't generalise well to unseen data from the same distribution. A model is said to have overfitted if it has too many parameters (is too complex) than can be justified by the data[11]; therefore, having more data and using less complex models are ways to prevent overfitting. See Figure 2.2 for an example of overfitting.

Regularisation

Regularisation techniques are methods designed to prevent overfitting. They can penalise a model's parameters from being too large or penalise the number of parameters (i.e. the complexity) of a model[11]. Ridge regression, as seen in Section 2.2.2, uses regularisation to keep the model's weights small. The parameter α is the regularisation strength.

Cross-Validation

Cross-validation is used to evaluate a model's performance. The main idea behind its variants is that a model must not be evaluated on any data it has seen during the training process, which may lead to overfitting. Therefore, the data is partitioned into training data and test data in various ways. The model trained on the training partitions and is tested on the test partition - data it has not seen before.

The holdout method of cross-validation is the most straight forward. The data is simply partitioned into two splits: the training set and test set. The model is trained on the training set and evaluated on the test set.

K-fold cross-validation is another cross-validation variant. This typically consists of splitting the data into k partitions and the aforementioned holdout method is repeated k times[15]. Each of the k partitions is used as a test set with the other $k-1$ partitions forming the training set. The model is thus trained k times and an average is taken over the k test set performances. See Figure 2.3

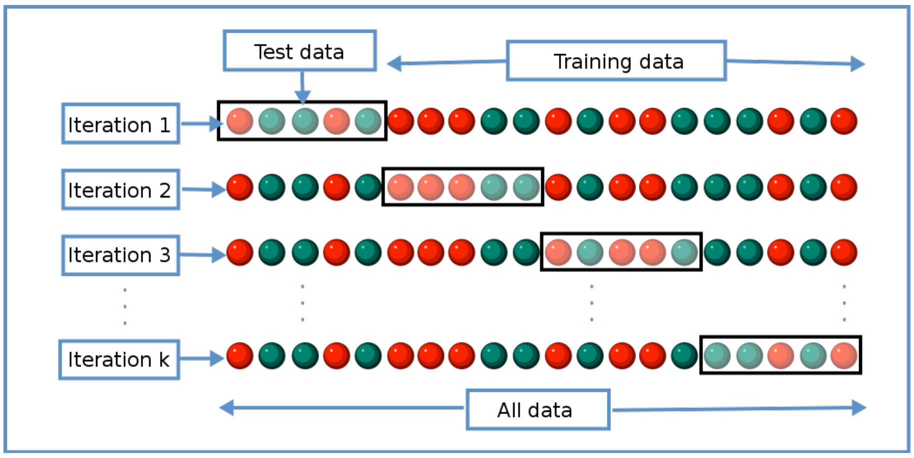


Figure 2.3: K-fold cross-validation. Attribution:
Gufosowa / CC BY-SA (https://creativecommons.org/licenses/by-sa/4.0)

Models often have parameters which are set and are not learned in a training process. These are called hyperparameters. Regularization strength, such as α in ridge regression (Section 2.2.2), is an example of a hyperparameter. When finding optimal hyperparameters to use, it is important not to evaluate which hyperparameters work best on the test set. This could lead to overfitting. Instead, nested forms of cross-validation are used. This is where the dataset is split into three sets: training, test and validation. The validation set is used to evaluate model performance for different hyperparameters. Once the best hyperparameters are found, the model is evaluated on the test set.

2.4 Signal Processing

In this project, we are dealing with signals: the brain signals and the audio signals (music or speech). In this section, we will look at some of the signal processing methods which will be used.

2.4.1 Signals

A signal is a function that represents the information of a given phenomenon or behaviour of a system[16]. We will usually consider signals as a function of time and denote them as $f(t)$.

2.4.2 Frequency

The frequency of a signal is the number of oscillations that occur per unit time. The SI unit of measurement is hertz (Hz) which denotes the number of repetitions per second. A frequency band refers to an interval of frequencies, e.g. 10Hz-20Hz.

2.4.3 Sampling Rate

Continuous-time signals are recorded as discrete-time signals by periodically measuring them. The sampling rate (or sampling frequency) of a discrete-time signal is the number of samples recorded per second (measured in Hz).

Downsampling

Downsampling refers to the reduction of the sampling rate of a discrete-time signal. This typically involves keeping every Mth sample of the original signal. Sometimes, a low-pass filter (Section 2.4.5) is applied beforehand to remove high frequencies from the signal.

2.4.4 Autocorrelation

Autocorrelation describes the correlation between a time-series signal and a lagged version of itself with respect to different time lags. Assume we have a random vector $\boldsymbol{x} = (f(1), f(2), ..., f(n)) \in \mathbb{R}^N$ representing the time series data for discrete-time signal $f(t)$, then the autocorrelation matrix, $\boldsymbol{R_{xx}}$ is defined as[17]:

$$\boldsymbol{R_{xx}} = E[\boldsymbol{xx}^\top] \in \mathbb{R}^{N\times N} \tag{2.6}$$

Where $\boldsymbol{R_{xx}}_{ij}$ denotes the correlation between $\boldsymbol{x}$ at time index i and $\boldsymbol{x}$ at time index j.

2.4.5 Filters

We can think of a signal as being composed of many smaller signals with different frequencies. In signal processing, filters are used to remove or diminish certain frequencies of the signal. This diminishing effect is known as attenuation.

Filtering will be a common preprocessing step for the brain signals before decoding takes place.

Low-pass and High-pass Filters

For a given signal, a low-pass filter will pass frequencies lower than a certain threshold and attenuate frequencies higher than the threshold. Similarly, a high-pass filter will pass frequencies higher than a certain threshold and attenuate frequencies lower than the threshold. See Figures 2.4a and 2.4b for the attenuation of a low and high pass filter respectively.

Band-pass filter

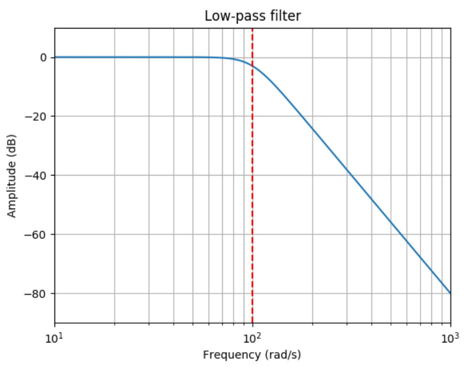
A band-pass filter passes frequencies within a given frequency band (range) and will attenuate frequencies outside of this range. This can be made by combining a low-pass and a high-pass filter[18]. An example of a band-pass filter being applied to a signal can be seen in Figure 2.5. See Figure 2.4c for the attenuation of a signal.

Filter banks

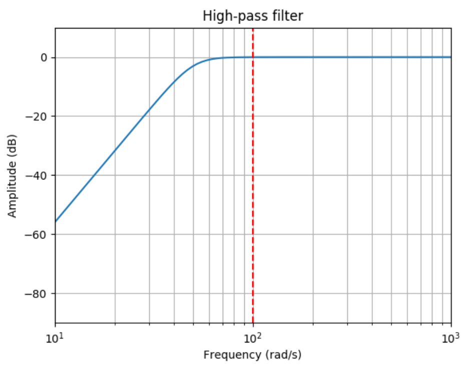
A filter bank is a series of band-pass filters which can be used to decompose a signal into its constituent frequency bands called sub-bands. See Figure 2.6 for the attenuation of the filters in a gammatone filter bank.

2.4.6 Envelopes

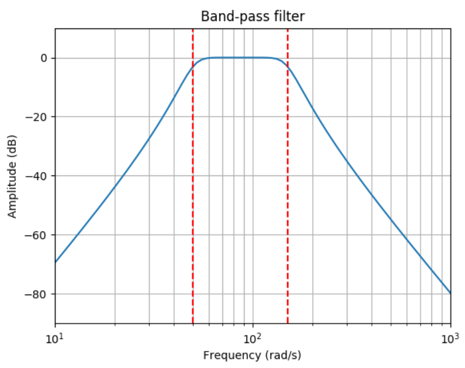
Signal envelopes are curves outlining the extreme points of an oscillating signal[19], this can be seen in Figure 2.7. Signal envelopes can be used as a representation of the audio stimulus as we shall see in Section 2.6.



(a) Low-pass filter with threshold 100Hz



(b) High-pass filter with threshold 100Hz



(c) Band-pass filter with frequency band 50-150Hz

Figure 2.4: Attenuation of low-pass, high-pass and band-pass filters

2.4.7 Time-Frequency Analysis

Frequency Domain

As stated in Section 2.4.1, we will usually consider a signal as a function of time, $f(t)$. This is the time-domain representation of a signal. However, we can also represent the signal with respect to the frequencies it is composed of. This new function, $\hat{f}(\xi)$, is f's frequency-domain representation. The frequency-domain representation of a signal is also known as its spectrum.

Consider an audio signal such as music, this signal is the combination of many constituent audio signals such as different instruments and notes playing at the same time. The frequency-domain representation of the music shows the breakdown of the constituent frequencies that make up the signal. A transform can be applied to time-domain function to get its frequency-domain representation. One example of this is the Fourier transform which decomposes a signal into a summation of different sine waves at different frequencies. An example of this can be seen in Figure 2.8.

Time-Frequency Representation

While a frequency-domain representation shows the frequency content, all information with respect to time is lost. For example, imagine there was a short high pitch burst in a piece of music. The frequency-domain representation of this would have no information about what time this burst occurred at. This is when a time-frequency representation (TFR) is useful. A time-frequency representation combines the representation in both the time and frequency domains.

Wavelet Transform

This section will give a very high-level overview of wavelets and wavelet transforms (as explained in [20]). A wavelet can be thought of, in simple terms, as a brief oscillation in time. The amplitude

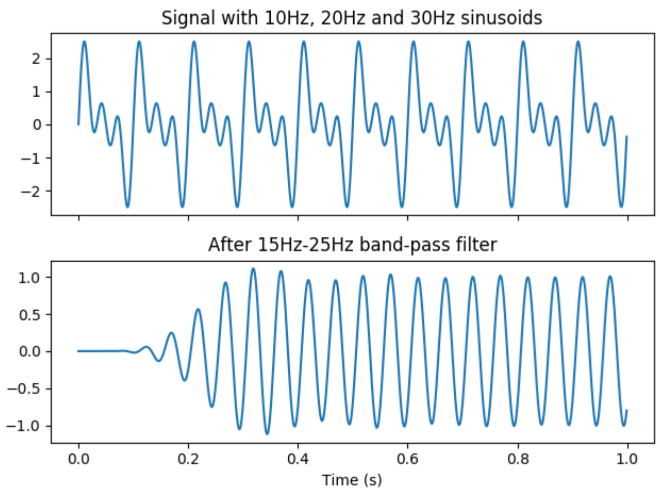


Figure 2.5: A 15Hz-25Hz band-pass filter is applied to a signal composed of thee sinusoids with frequencies of 10, 20 and 30Hz. The only signal that remains is the 20Hz component which falls in the range of the band-pass filter.

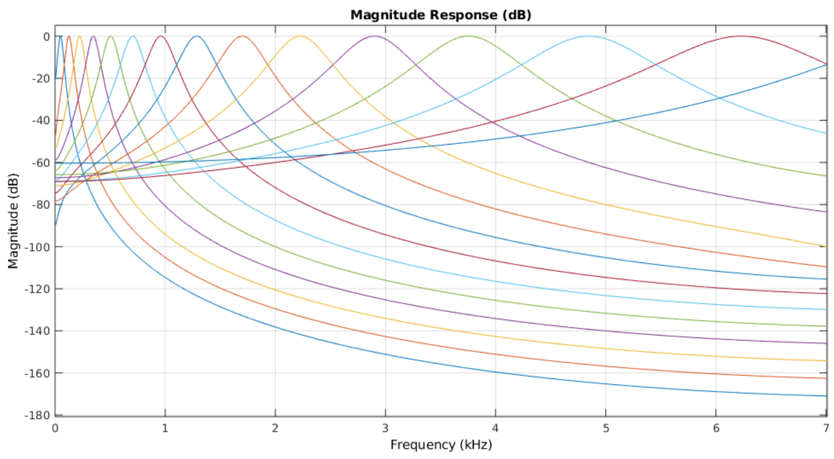


Figure 2.6: A Gammatone filter bank. Each line represents one of the individual filters which covers a particular range of frequencies (near its peak)

of a wavelet is 0 everywhere except for a certain region. See Figure 2.9 for an example of a Morlet wavelet. Wavelets are defined by a mother wavelet function $\psi(t)$ and are parameterised by two constants: a scale parameter a and a shift parameter b. These parameters transform the mother wavelet into child wavelets:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

a scales the width of the mother wavelet, and b shifts the mother wavelet in time. We can see examples of how a and b affect a Morlet wavelet in Figure 2.10.

For completeness, the wavelet transform equation for a time-domain signal $f(t)$ is:

$$X(a,b) = \int_{-\infty}^{\infty} \overline{\psi_{a,b}(t)} f(t)$$

For this project, the specifics of this transform are not important. An intuition into what a and b do is more important. Increasing the scale parameter a produces a more narrow (or "squashed") wavelet which can detect higher frequencies. Conversely, decreasing a produces a wider wavelet which can detect lower frequencies. The translation parameter b shifts the wavelet along the time dimension i.e. it shifts the section of the wavelet "pulse" where the amplitude is not 0. This is

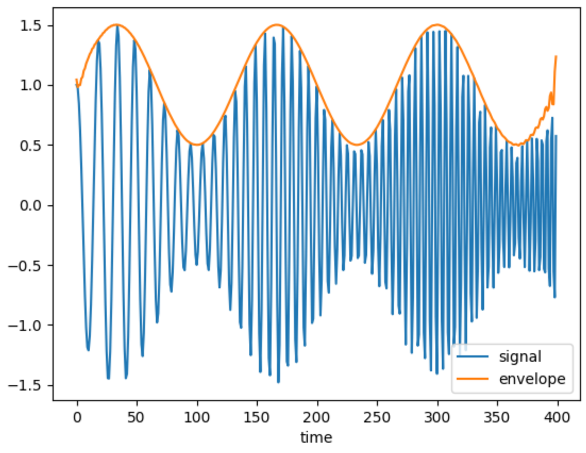
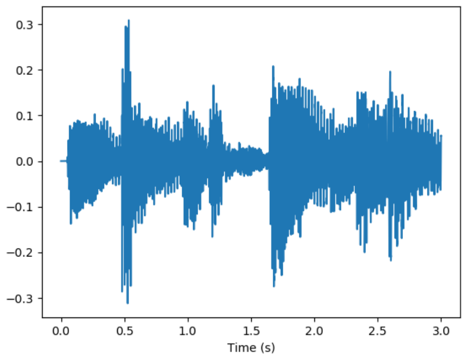
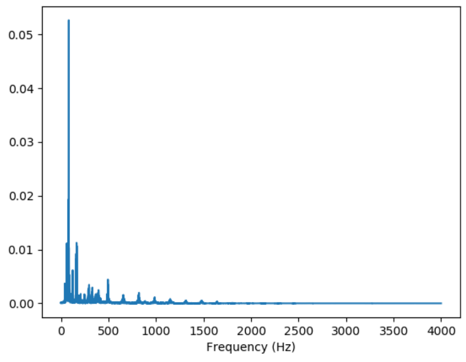


Figure 2.7: The envelope of a signal



(a) Time-domain Representation



(b) Frequency-domain Representation

Figure 2.8: The time and frequency-domain representation of an audio signal. Figure 2.8a shows the signal as a function with respect to time. Figure 2.8b shows the signal with respect to the various frequencies that comprise it via a Fourier transform. As we can see, most of the constituent frequencies lie in the range 0-500Hz.

how the frequency content at various times in the signal $f(t)$ can be analysed.

In summary, a corresponds to different frequencies and b corresponds to the time in the signal where they occur, thus we can form a time-frequency representation of the signal. A wavelet transform can be used to produce a scalogram, a matrix representing the "amount" of the signal at each time and each frequency. See Figure 2.11 for some examples of scalograms.

We will see how scalograms can be used as inputs for neural decoding in Section 2.9.

2.4.8 Energy

Given a discrete-time signal, $f(t)$, the energy E of the signal is defined as the sum of the square of the signal at each point in time[21]:

$$E = \sum_{t=-\infty}^{\infty} f(t)^2 \tag{2.7}$$

In this project, we will see the concept of energy as a measure of the loudness of a sound.

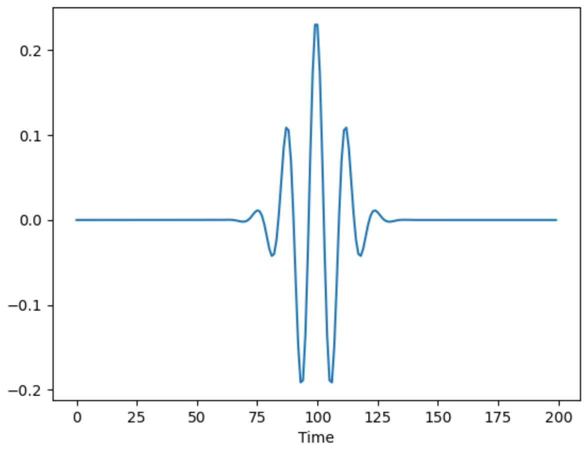


Figure 2.9: The real component of a Morlet wavelet.

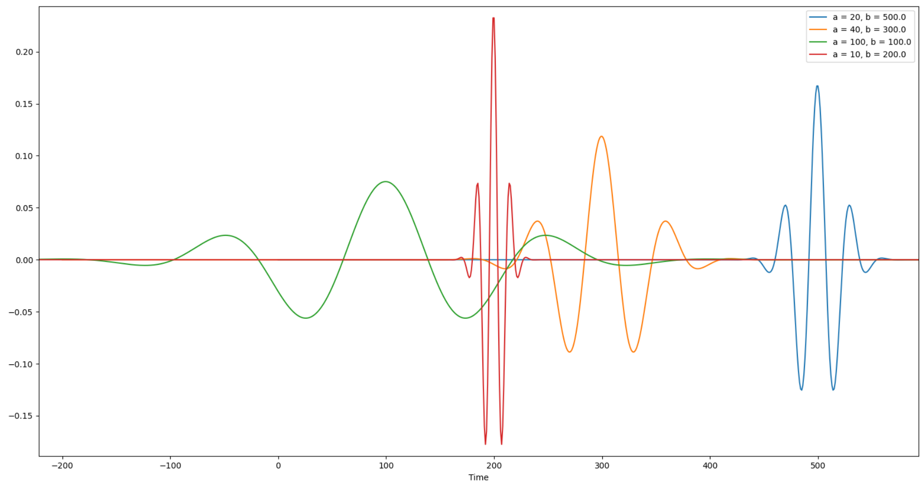


Figure 2.10: Morlet wavelets with different scale and shift parameters a and b respectively.

2.5 Neurophysiological Data

In this section, we will look at two ways of recording neurophysiological data (informally, "brain signals"): Electroencephalography (EEG) and magnetoencephalography (MEG). The dataset we are dealing with in this project will be with MEG data; however, we will also be looking at public datasets which use EEG data.

2.5.1 Electroencephalography (EEG)

EEG involves placing electrodes on the scalp and measuring the voltage from the brain activity[22] (Figure 2.12). The EEG recordings consist of multiple channels, each one being a signal corresponding to one of the electrodes.

While EEG recordings have a high temporal resolution in the order of milliseconds[22], they are limited by their spatial resolution making it hard to pinpoint specific regions of brain activity[25]. Furthermore, they are limited in identifying activity from deeper structures in the brain[26].

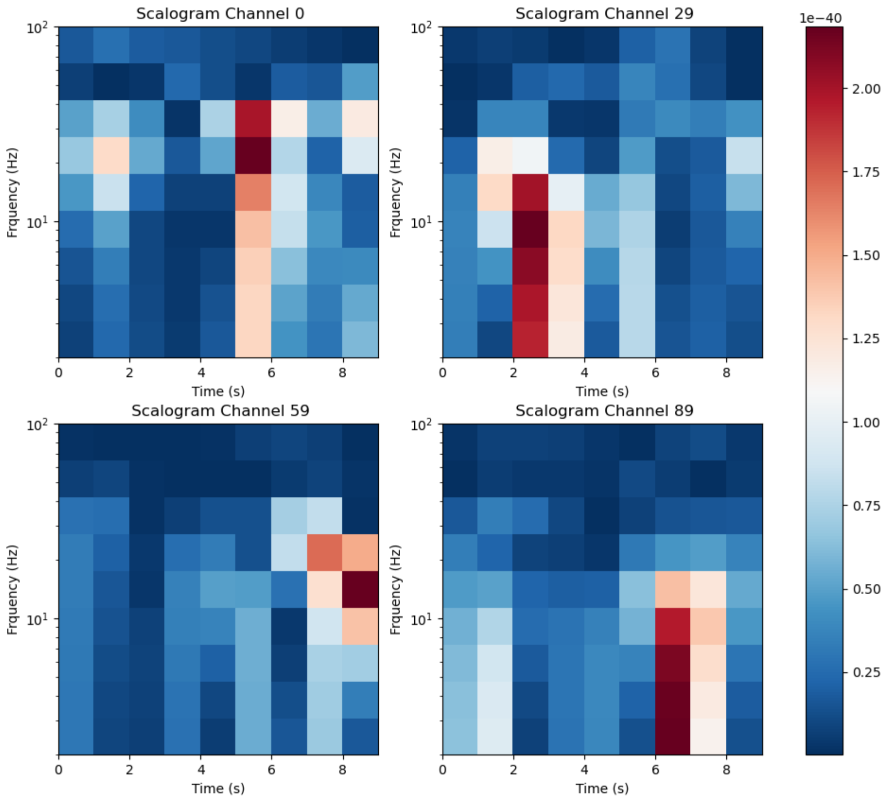


Figure 2.11: Scalograms of 4 different brain signals. The x-axis represents the signal at various times in 2 seconds. The y-axis represents the "amount" of the signal at various frequencies.

2.5.2 Magnetoencephalography (MEG)

Rather than measuring voltage, MEG records magnetic fields corresponding to the brain activity. Much in the same way EEG recordings have multiple channels corresponding to each electrode, MEG recordings have multiple channels corresponding to each sensor of the magnetic field. Both MEG and EEG recordings reflect the same underlying process in the brain[27]; therefore for the scope of this project, they can be thought of in the same way unless specified otherwise. MEG data can be thought of as a higher-resolution version of EEG data.

2.5.3 Source Reconstruction

For EEG/MEG data, source reconstruction is the process by which the brain activity is localised to the specific regions in the brains they emerged from[28]. In this project, we will work with MEG data after source reconstruction has been applied. Rather than having one channel per sensor, we will have one channel per region in the brain.

Representing EEG/MEG data

Representing EEG/MEG data is fairly straight forward. What we have is one signal for each channel (sensor or source reconstruction location). Each signal is a function in terms of time. Therefore we can represent EEG/MEG data as a two-dimensional matrix. One dimension representing the time of the brain activity and the other dimension representing the channel. See Figure 2.13 for some sample MEG data.

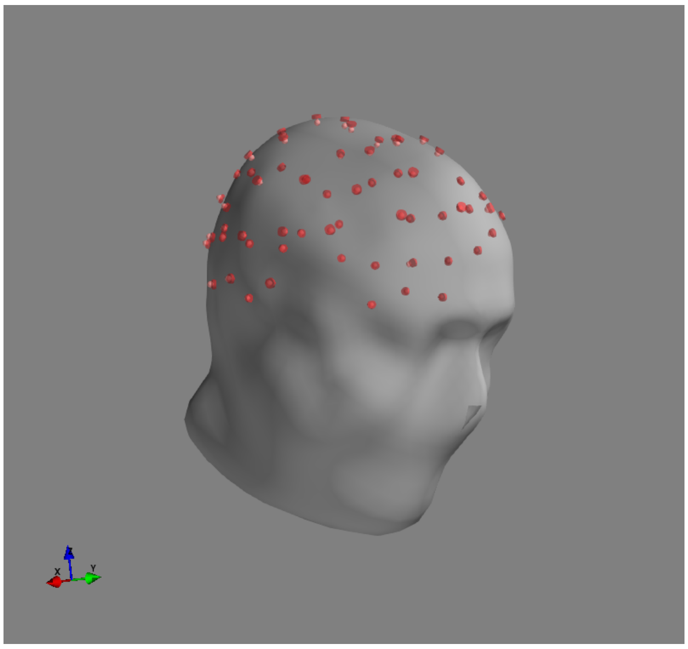


Figure 2.12: Visualisation of electrodes placed on the scalp. Images generated using the MNE framework[23][24]

2.6 Auditory Attention Detection (AAD)

In this section, we will look at the auditory attention detection (AAD) paradigm which aims to develop methods to detect which of two speakers is being attended to given the measurements of brain activity (namely EEG). This paradigm will form the basis for the decoding methods that will be used in this project.

In a setting where many stimuli are present, the brain's ability to focus attention onto one stimulus and filter out the other stimuli is known as the cocktail party effect[29]. The cocktail party scenario, as its name suggests, is the situation where the stimuli are multiple speakers and the listener is aiming to focus their auditory attention to just one of them. The methods outlined in this section will form a basis for the methods we will employ to assess the degree to which the brain is perceiving the music stimulus.

Under the AAD paradigm, we have a listener who is presented with two simultaneous audio streams, the attended and unattended speech. The attended and unattended speech are the audio stimulus the listener must pay attention to and ignore respectively. AAD methods typically involve decoding an envelope (Section 2.4.6) of the attended speech and comparing it to both attended and unattended speech envelopes to find the one that most closely resembles it[10]. The attended speaker can now be predicted as the speaker that corresponds to the speech envelope which most resembles the reconstructed envelope; if the predicted speaker was the attended speaker, we have a correct classification. This procedure is outlined in Figure 2.14.

2.6.1 AAD Procedure

In this section, we will outline the AAD procedure as outlined in [10] and [30].

Decoding

We are given an EEG recording with C channels. Let $M(t, c)$ denote the EEG value at time t at channel index $c \in \{1...C\}$. The reconstructed representation of the attended speech is a wave envelope (Section 2.4.6). The speech envelopes for the attended and unattended sources are

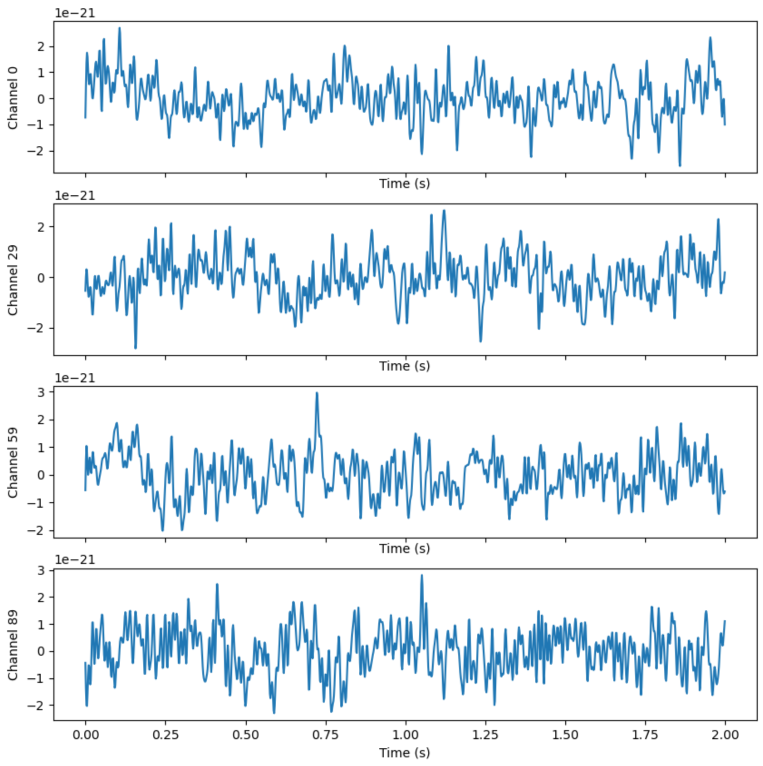


Figure 2.13: Sample of MEG data. Each graph corresponds to one signal at a particular channel with respect to time.

denoted by $s_a(t)$ and $s_u(t)$ respectively, where t is the time as before. The reconstructed stimulus of the attended speech, $\tilde{s}_a(t)$, is decoded as follows:

$$\tilde{s}_a(t) = \sum_{n=1}^{N} \sum_{c=1}^{C} \mathbf{D}_{nc} M(t + n - 1, c) \tag{2.8}$$

Where n is the time lag index to index samples starting from t and going onwards. Thus, to reconstruct the attended speech envelope, we use a weighted combination of all C channels of the EEG recording and times from t to $t + N - 1$ inclusive. The weights are given by the decoder $\mathbf{D} \in \mathbb{R}^{N \times C}$. Note that this is just a linear model as we have seen in ordinary least squares (Section 2.2.1) and ridge regression (Section 2.2.2).

N is typically chosen to account for time lags up to 250ms which has been shown to work well for reconstructing the attended speech envelope[10] (and has been verified by other AAD research[31][30][32]). These time lags account for the delay between the audio and the response in the brain. A visualisation of these lags is shown in Figure 2.15. We can rewrite 2.8 in a vectorised format by flattening the weights into a vector $\mathbf{d} \in \mathbb{R}^{NC}$ and the EEG recordings for time t giving us $\mathbf{m}(t) \in \mathbb{R}^{NC}$.

$$\mathbf{m}(t) = [M(t,1), M(t+1,1), ..., M(t+N-1,1), ..., M(t,C), M(t+1,C), ..., M(t+N-1,C)] \tag{2.9}$$

Thus we can express Equation 2.8 as

$$\tilde{s}_a(t) = \boldsymbol{d}^\top \boldsymbol{m}(t) \tag{2.10}$$

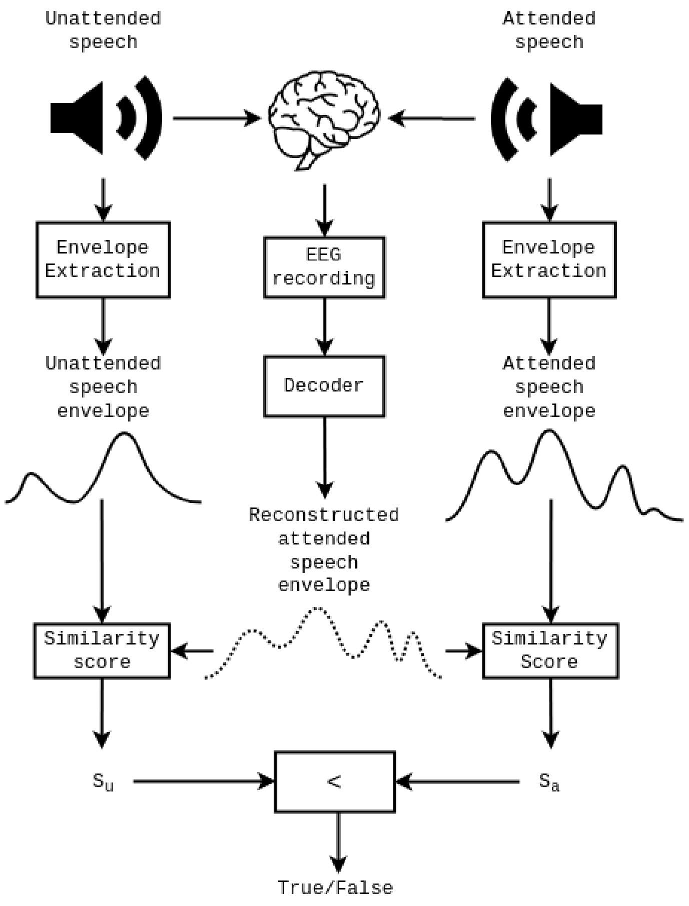


Figure 2.14: The AAD procedure outline. We apply envelope extraction methods to the attended and unattended speech and attempt to decode the attended speech envelope from the brain activity. Both unattended and attended envelopes are compared to obtain similarity scores S_u and S_a respectively. If the inequality in the last stage ($S_a > S_u$) evaluates to true, then we have a correct classification. Adapted from [10].

We want to make the reconstructed envelope as close to the actual speech envelope as possible. This is done using ordinary least squares regression (Section 2.2.1) and produces the optimal decoder denoted $\boldsymbol{d}^*$:

$$\boldsymbol{d}^* = \underset{\boldsymbol{d}}{\text{argmin}}\, E[(\tilde{s}_a(t) - s_a(t))^2] = \underset{\boldsymbol{d}}{\text{argmin}} \sum_t (\boldsymbol{d}^\top \boldsymbol{m}(t) - s_a(t))^2 \tag{2.11}$$

Let $\tilde{\boldsymbol{s}}_a, \boldsymbol{s}_a \in \mathbb{R}^T$ denote the vectorised form of the reconstructed and real attended envelope respectively, and T be the total number of samples. This problem can be equivalently expressed as[33]

$$\boldsymbol{R} = [\boldsymbol{m}(1), \boldsymbol{m}(2), ..., \boldsymbol{m}(T)]^\top \in \mathbb{R}^{T\times NC} \tag{2.12}$$

$$\tilde{\boldsymbol{s}}_{\boldsymbol{a}} = \boldsymbol{R}\boldsymbol{d} \tag{2.13}$$

$$\boldsymbol{d}^* = \underset{\boldsymbol{d}}{\text{argmin}}\, \|\tilde{\boldsymbol{s}}_{\boldsymbol{a}} - \boldsymbol{s}_{\boldsymbol{a}}\| \tag{2.14}$$

$$= \underset{\boldsymbol{d}}{\text{argmin}}\, \|\boldsymbol{R}\boldsymbol{d} - \boldsymbol{s}_{\boldsymbol{a}}\| \tag{2.15}$$

Which has a know analytical solution of

$$\boldsymbol{d}^* = (\boldsymbol{R}^\top \boldsymbol{R})^{-1} \boldsymbol{R}^\top \boldsymbol{s}_a \tag{2.16}$$

where $\boldsymbol{R}^\top \boldsymbol{R}$ is the sample autocorrelation matrix (Section 2.4.4) when considering only N samples ahead of any given time t.

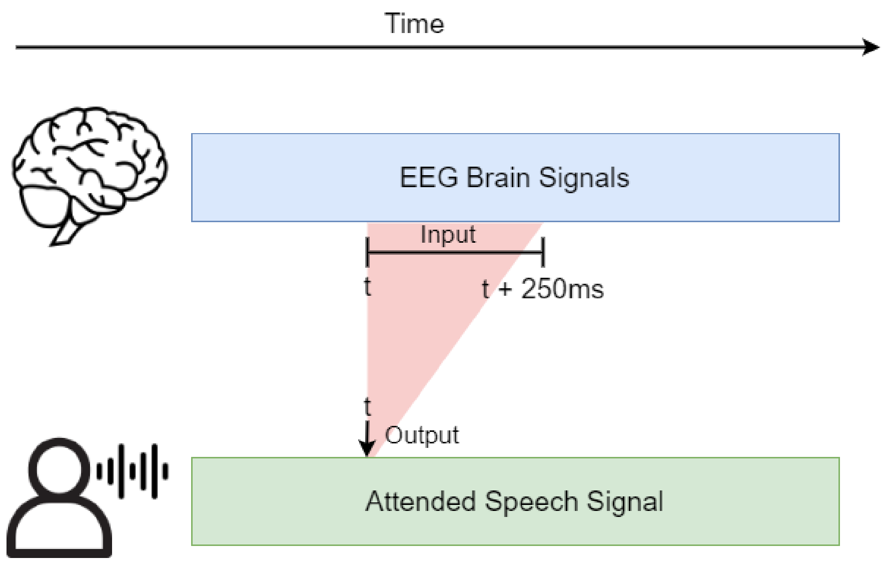


Figure 2.15: The time lags used in an AAD decoder. To decode an output of the speech at time t, a window of EEG brain signals with the range t to $t + 250$ms is used as an input.

Cross-Validation

K-fold cross-validation (Section 2.3.2) is used to evaluate decoding performance. The data is first partitioned into K sets. A decoder is trained on $K-1$ training partitions and evaluated on the remaining test partition. This is repeated by cycling through all K partitions and averaging the evaluation performance.

For each test subject, we have K measurements known as trials. Each trial is a partition of our data mentioned above. Let d_k represent the trained decoder for trial k. Let $\boldsymbol{d}_{-k}$ represent the combined decoder using all trials but trial k.

The decoders can be combined by averaging their weights as follows[10]:

$$\boldsymbol{d}_{-k} = \frac{1}{K-1}\sum_{i \in S} \boldsymbol{d}_i \tag{2.17}$$

$$S = \{1, 2..., K-1, K\} - \{k\} \tag{2.18}$$

A problem, however, was that the short period of the trials meant that each decoder was trained on a few data samples and would overfit (Section 2.3.2) the data. Using ridge regression (Section 2.2.2) is one way to account for this[32]. Another solution is to combine all the trials and train one decoder[30]:

$$\boldsymbol{d}^*_{-k} = \underset{\boldsymbol{d}_{-k}}{\operatorname{argmin}} \frac{1}{K-1}\sum_{i \in S} E[(\tilde{s}_{a,i}(t) - s_{a,i}(t))^2] \tag{2.19}$$

$$S = \{1, 2..., K-1, K\} - \{k\} \tag{2.20}$$

Where $\tilde{s}_{a,i}(t)$ is the reconstructed speech envelope for the trial i.

$$\tilde{s}_{a,i}(t) = \boldsymbol{d}^\top_{-k}\boldsymbol{m}_i(t) \tag{2.21}$$

Predicting the Attended Speaker

To predict the attended speaker, we calculate the Pearson correlation coefficients (Section 2.3.1) r_a and r_u for the correlation between the reconstructed envelope, $\tilde{s}_a(t)$, with the attended envelope,

$s_a(t)$, and the unattended envelope, $s_u(t)$, respectively. If $r_a > r_u$, this means the correct speaker has been identified. The accuracy of these predictions is how the AAD procedure is evaluated.

2.6.2 Motivation and Relevance

We will now look at the motivation for studying this area of research and its relevance to the project. The problem setting in the AAD scenario is different from the aims of this project. AAD aims to detect the audio stimulus that the subject was paying attention to when presented with two stimuli. The aim of this project, however, is to assess the degree to which the audio stimuli (music) is being perceived by the subjects in and out of the psychedelic state.

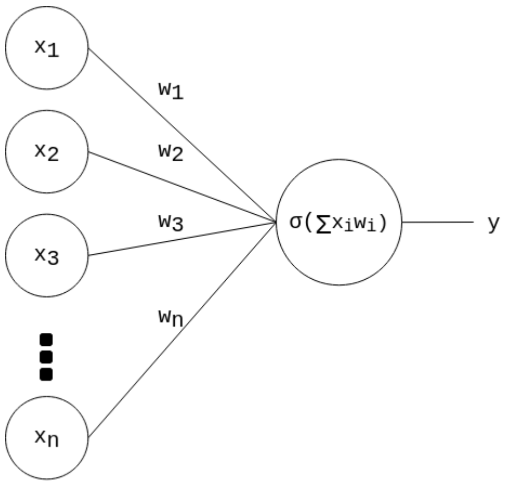
What AAD provides us with is a way to assess the similarity between the brain signals and the audio stimulus. The measure of similarity was how close the reconstructed stimulus envelope was to both the attended and unattended audio stimuli envelopes. What this research suggests is that when the subjects are paying attention to one stimulus, there appears to be a higher similarity between their brain signals and the stimulus they are focusing on. In this project, we will use this measure of similarity, i.e. the ability to decode a music representation from brain signals, as a proxy measure of the degree to which the brain is perceiving/experiencing the music.

2.7 Neural Networks to Extract Musical Features

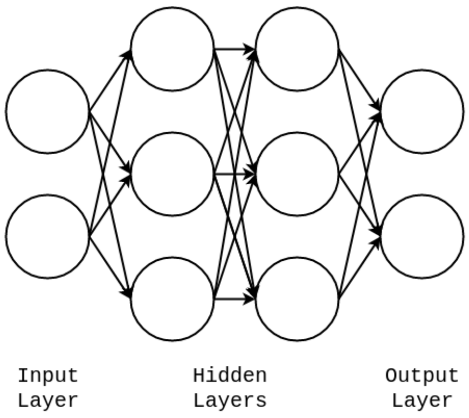
In this section, we will look at how neural networks can be used to generate a representation of music.

2.7.1 Neural Networks

Artificial neural networks (or simply neural networks) are models which can be used to represent arbitrary (non-linear) functions. Neural networks are composed of a series of interconnected nodes, typically arranged in layers as shown in Figure 2.16b. Each node will sum a weighted combination of inputs and apply a non-linear activation function to the result, as shown in Figure 2.16a. The inputs are either the input to the function itself or the output from a node's parents.



(a) Individual neuron applying the activation function σ to the weighted sum of the inputs



(b) Interconnected neurons forming a neural network

Figure 2.16: The structure of simple neural networks

Neural networks are a subset of machine learning. A machine learning model learns to perform a given task given some data, rather than explicitly being programmed what to do. It is "trained" on this data. This training procedure involves incrementally adjusting its weights to minimise some loss function. The loss function is a measure of the error of the neural network's performance.

Consider the task to classify handwritten digits in the MNIST dataset[34](see Figure 2.17). In this dataset, there are many handwritten digits images with their corresponding labels (e.g. a hand-drawn 6 with the label "6"). Now consider how a neural network could be used for this task. As an input, the neural network will accept the image of the handwritten digit. This would be a 784-dimensional vector corresponding to the value of each grayscale pixel in the 28×28 images. The output of the neural network would be a 10-dimensional vector:

$$y_{pred} = [p_0, p_1, p_2, p_3 ..., p_9]$$

where p_n represents the predicted probability that the input image is the digit n. Consider a training example of the handwritten digit 3 without loss of generality. When training the neural network, we would aim for $p_3 = 1$ and $p_n = 0$ for all $n \neq 3$. The further these probabilities are from this target, the higher the loss function will be.

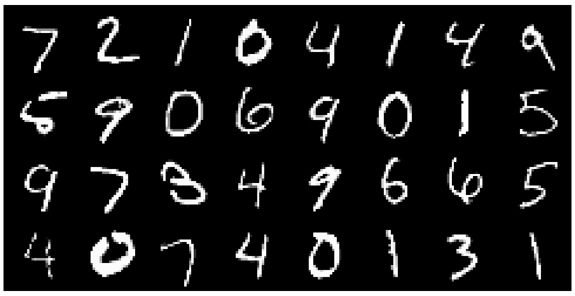


Figure 2.17: Sample digits from the MNIST dataset

2.7.2 Autoencoders

An autoencoder is a neural network that aims to find low-dimensional representations of a given dataset. These lower-dimensional representations (also called encodings) preserve as much information as possible by extracting key features from the data.

The autoencoder consists of two parts: an encoder and a decoder. The encoder transforms the high-dimensional input vector into a lower-dimensional encoding vector. The decoder then takes this encoding and attempts to reconstruct the original high-dimensional input. When the autoencoder is trained, the encoder learns what are the important features to preserve when producing these low-dimensional encodings, thereby allowing the decoder to reconstruct the original input. What an autoencoder is doing is a form of (lossy) data compression, where a smaller space is found which preserves as much information about the original data as possible. See Figure 2.18 for a diagram of an autoencoder applied to handwritten digits.

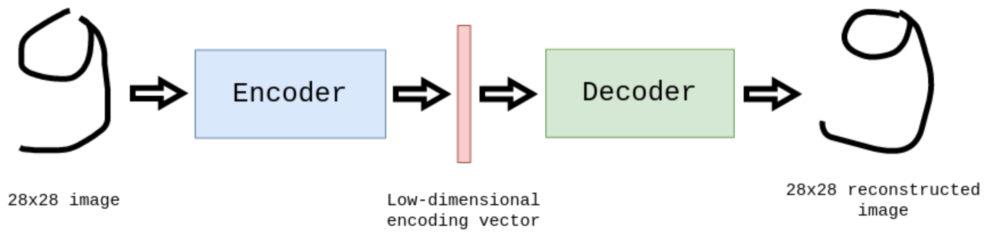


Figure 2.18: Autoencoder applied to handwritten digits.

2.7.3 WaveNet Encodings

WaveNet[35] is a neural network which can model and generate natural-sounding speech. It was also shown that this model could be applied to music. The breakthrough that WaveNet achieved was being able to generate audio one sample at a time. Given the high frequency and complexity of audio signals, this was considered a challenging task at the time.

We will now look at research building on the WaveNet model for music synthesis. The Magenta project[1] aims to use machine learning to facilitate the creative process of music. Magenta developed a WaveNet style autoencoder[36]. This autoencoder is trained on the NSynth dataset: a large dataset of musical notes with around 300k different notes from 1000 different instruments. This autoencoder extracts the most important features from musical sounds into a 16-dimensional vector.

For this project, we can treat this WaveNet autoencoder as a black box which takes a chunk of music as an input and outputs encodings of the musical features. We shall refer to these encodings as "WaveNet encodings" from here on out. See Figure 2.19 for examples of some WaveNet encodings applied to music. These encodings will be generated by a publicly available pre-trained model.

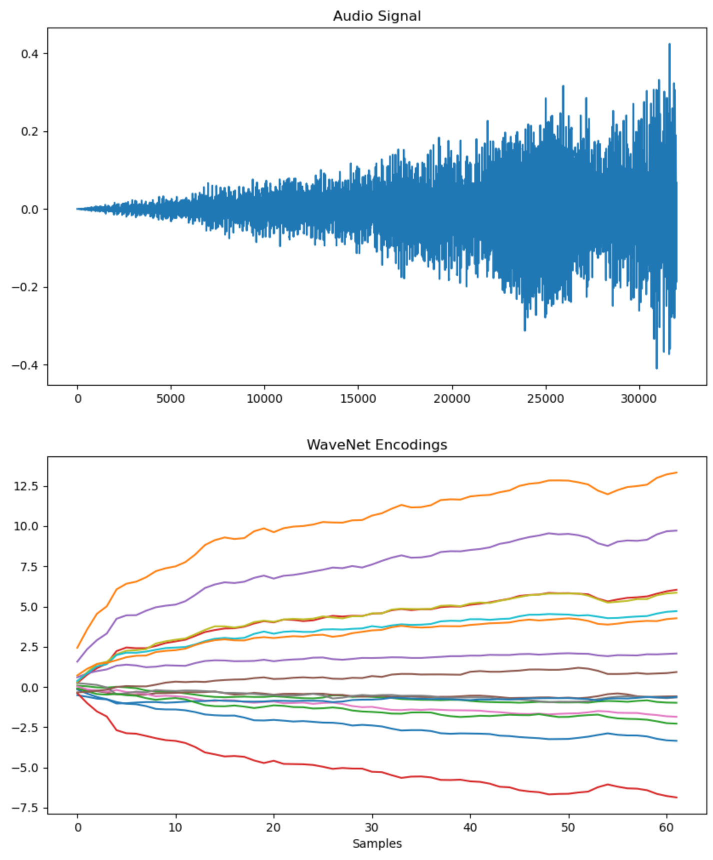


Figure 2.19: WaveNet encodings of one chunk of music. The top graphs show the original raw audio data of the music. The bottom graph shows the WaveNet encodings generated with each line corresponding to one of the 16 dimensions in the encodings. We can see some resemblance between the encodings and the audio signal.

2.7.4 Relevance

These WaveNet encodings were originally designed to aid in the creative process of music. For example, one can take the encodings of two different instruments and add them together, thereby combining them. This joint encoding can be used to generate new raw audio combining the characteristics of both instruments. This process can be seen in Figure 2.20. This, however, is not our use case. We have already seen envelopes as one possible method of representing the music

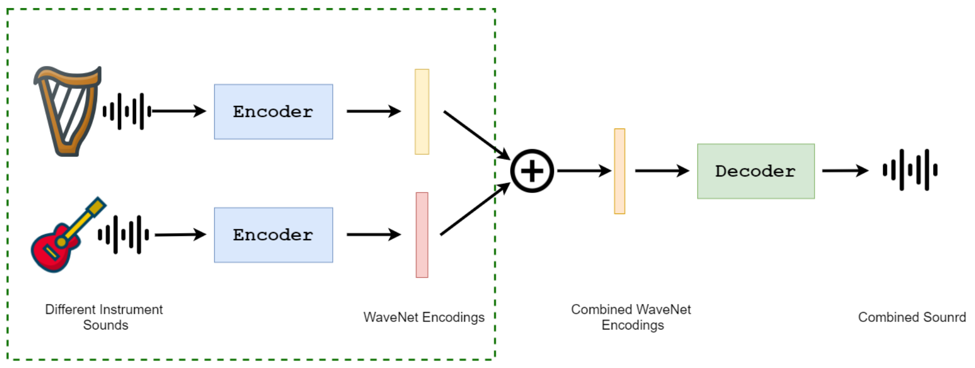


Figure 2.20: A visualisation of how WaveNet encodings can be combined to create new sounds. The dashed green box denotes the aspects of this diagram that are relevant to this project.

audio stimulus in Section 2.6 (note that we have seen it being used for speech). In this section, we have outlined another way to represent the music, namely WaveNet encodings to capture the musical features into a 16-dimensional vector.

2.8 Music Information Retrieval

So far, we have two possible ways to represent the musical stimulus:

- As an audio envelope as seen in Section 2.6
- As WaveNet encodings as seen in Section 2.7.3

In this section, we will look at specific acoustic features extracted through music information retrieval. Existing research on the effect of psychedelics looks at 23 acoustic features which contribute to the perception of music[3]. These features can all be extracted using the MIRToolbox (MIR - Music Information Retrieval) library[37].

We will now go over some of these acoustic features. Each one of these features is calculated on a small window of music. Let $f(t)$ denote the signal as a function of time corresponding to the window of music with T samples. The features (and their corresponding MIRToolbox function) are as follows:

1. RMS Energy (`mirrms`) - The root of the squared average of the amplitude. Corresponds to the signal's loudness.
$$\sqrt{\frac{1}{T}\sum_{t=0}^{T-1} f(t)}$$
2. Zero Crossing Rate (`mirzerocross`) - The rate of how frequently the audio waveform crosses the x-axis (See Figure 2.21). It is a simple measure of the noisiness of the signal.
3. Spectral Centroid (`mircentroid`) - The mean frequency in a signal. Calculated as a weighted average of its spectrum. See Figure 2.22.
4. High Energy-Low Energy Ratio (`mirbrightness`) - The proportion of the total energy that lies above the 1500Hz threshold. See Figure 2.22.
5. Spectral Spread (`mirspread`) - The standard deviation of the spectrum of the signal i.e. how spread out the frequencies are from the mean frequency.
6. Spectral Roll-off (`mirrolloff`) - The threshold frequency such that 85% of the signal's energy lies below that frequency. See Figure 2.22.
7. Spectral Entropy (`mirentropy`) - A measure of uncertainty of the signal's spectrum. The spectral entropy represents whether the signal's spectrum contains predominant peaks or not. A flat spectrum consisting of equal amounts of each frequency (known as white noise)

will maximise spectral entropy because there is high uncertainty of the frequencies[38]. Conversely, a single sharp peak at a given frequency will minimise the spectral entropy i.e. have a high certainty as the output is entirely governed by that single frequency[38].

8. Flatness (`mirflatness`) - The smoothness or spikiness of the data. Calculated as a ratio of the geometric mean and the arithmetic mean.

$$\frac{\sqrt[T]{\prod_{t=0}^{T-1} f(t)}}{\frac{1}{T}\sum_{t=0}^{T-1} f(t)}$$

9. Roughness (`mirroughness`) - An estimate of sensory dissonance in a sound. Sensor dissonance being the roughness/beating in a sound.
10. Spectral Flux (`mirflux`) - The distance between the spectra of adjacent windows of the music. In other words, how much the spectrum changes at any given point in time.
11. Sub-band Flux (`mirflux`) - The signal is decomposed into 10 sub-bands using a filter bank (Section 2.4.5) and the spectral flux is calculated for each of them. This yields 10 features.

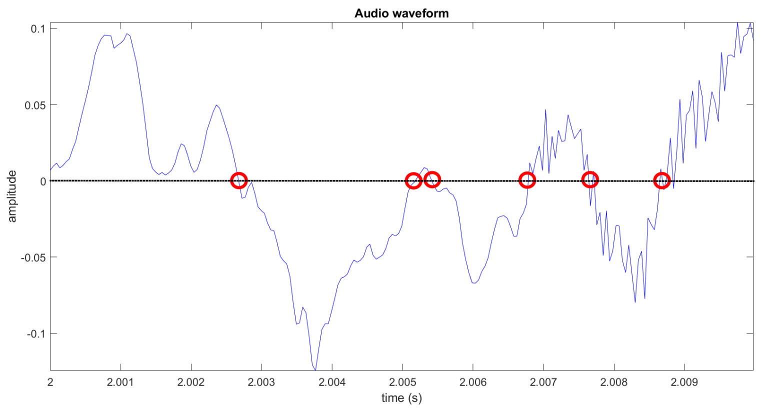


Figure 2.21: Zero crossing rate visualisation. Red circles show the zero crosses.

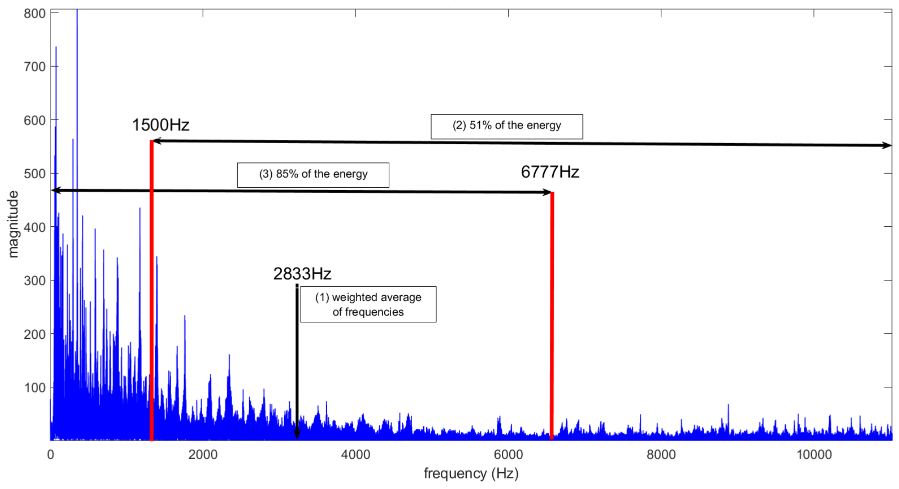


Figure 2.22: Visualisation of the spectrum (frequency-domain) of some music. The three annotated points correspond to three acoustic features. (1) - the spectral centroid. (2) - the high energy-low energy ratio. (3) - the spectral roll-off.

To calculate these features, the music is split into overlapping windows, and these features are calculated on a per-window basis, thus giving us the evolution of the features throughout the song.

The features listed above are considered short-term features and are calculated with window sizes of 25ms that overlap 50% with each other[38]. The RMS energy feature represents loudness and the others are timbral features[38]. The timbre of a musical sound is the characteristics which make voices or musical instruments sound different from one another[39].

2.9 Scalograms As Decoding Inputs

In Section 2.6, we saw a time-domain representation of the brain signals for the decoding input. In this section, we will look at another possible way to extract the features of the brain signals inspired by [40]. In [40], a decoding model was proposed to decode the arm motion of monkeys. The brain signal input was electrocorticographic (ECoG) measurements. For our purposes, the ECoG can be thought of an invasive alternative to EEG; while EEG electrodes are placed on the scalp, ECoG electrodes are surgically implanted on the surface of the brain[41]. The preprocessing steps were as follows:

1. For a given electrode (ECoG channel), a scalogram was calculated for a window of time $[t, t+\delta]$. Recall from Section 2.4.7 that a scalogram is a 2-dimensional matrix which shows the composition of the frequencies in a signal (along the frequency dimension) at different times (along the time dimension). 10 frequencies were chosen in the range 10-150Hz spaced out on a logarithmic scale. The time window of the signal was 1s long.
2. Each of these scalograms was downsampled to get 10 samples in the time domain spaced out across the 1s window. Therefore the scalogram size was 10×10 for each electrode.
3. Z-score normalisation took place along the frequency axis to get the values across different frequency bins to a comparable scale.

The input size to the decoder was 3200 for monkey A (32 channels, 10×10 scalogram) and 6400 for monkey B (64 electrodes, 10×10 scalogram). Partial least squares regression was used to handle the large dimensionality of the input and highly correlated (multicollinear) input features, which could lead to overfitting (Section 2.2.4).

2.10 Summary

In this chapter, we have outlined various components which we will use to assess the influence that LSD has on the perceptual experience of music.

We will use a reformulated variant of AAD (Section 2.6) to compute the similarity between the brain signals and the music. More specifically, this similarity measure will be how well we can reconstruct a representation of the music from the brain signals, i.e. decoding performance.

Possible representations of the music we have seen are:

- Signal envelopes as seen in AAD (Section 2.6).
- WaveNet encodings (Section 2.7.3).
- Music information retrieval of short-term acoustic features that affect perceptual experience (Section 2.8).

Possible preprocessing of the brain signals, namely MEG, are:

- A time-domain representation of the signals. More specifically, a window of the signals represented as a function of time as seen in AAD (Section 2.6).
- A time-frequency representation of the signals. More specifically, scalograms at each of the brain signal channels (Section 2.9).

We will then need a decoder model which will take the preprocessed brain signals as an input and return a reconstructed musical stimulus representation as the output. In Section 2.2 we outline the decoding methods we will use in this project. In Section 2.6, we have seen how ordinary least squares regression can be used to reconstruct the audio envelope given a window of brain signals with respect to time. In Section 2.9, we have seen how partial least squares regression was employed when the high dimensional (and highly correlated) inputs of scalograms were used.

Once a given decoder has fit onto the training data, they will be evaluated with the metrics outlined in Section 2.3.1, mainly the R^2 score as advised by the client. Once sufficient decoding performance has been achieved (this will be defined in subsequent chapters) the difference that LSD has on the performance will be analysed.

Chapter 3

Auditory Attention Detection (AAD)

3.1 Introduction

Before looking at the dataset from the Centre for Psychedelic Research, we will start by trying to reproduce the results from the AAD research as seen in Section 2.6. The motivations behind this are as follows:

- It would serve as a way to gain familiarity with EEG (and MEG by extension) data
- Results obtained from this data could be compared to the results from research using the same dataset for validation. When these methods are validated, we can apply them to the project's main dataset knowing that they are implemented correctly.
- This would give some idea as to how elements of the code could be structured and composed.

3.2 Problem Setting

3.2.1 Dataset

The AAD dataset used in [30] is publicly available[42]. We will try to reproduce similar results in this chapter.

This data contains the EEG recordings of 16 normal hearing subjects. Each subject was exposed to two simultaneous audio stimuli - one in each ear. These stimuli were male narrations of Dutch stories. The subject was asked to focus on the stimuli in one ear and ignore the stimuli in the other ear i.e. exhibit the cocktail party effect[29]. As previously mentioned in Section 2.6, the audio stimulus which the subject is focusing on is called the attended stimulus and the audio stimulus which the subject is ignoring is called the unattended stimulus.

There are approximately 72 minutes of EEG recording for each of the 16 subjects. The EEG data consists of 64-channels (i.e. 64 brain signals) with an original sampling rate of 8196Hz.

This data was already preprocessed to some degree for us. The data was high-pass filtered with a 0.5Hz threshold and downsampled to a sampling rate of 128Hz. Artefacts, such as the ones mentioned in Section 2.5.1, were also removed. Therefore, the data did not need to be cleaned.

The audio stimuli were downsampled to 8000Hz and saved.

3.2.2 Objective

The EEG data and accompanying stimuli were split into 30s trials. For any given trial, the objective was to predict the attended stimuli.

3.3 Methodology

The steps taken were inspired by the paper which uses the same dataset[30]. Elements of other research in this area, such as in [10], were used when steps from [30] were ambiguous. Although some of the preprocessing steps may be slightly altered, the decoding and evaluation methodology used are identical to that in [30], thus allowing us to see if we could reproduce similar results.

3.3.1 Preprocessing

EEG Preprocessing

As we have discussed previously (Section 2.5.1), the EEG data can be thought of as 64 concurrent signals, with one signal corresponding to each channel (a sensor placed on the scalp of the test subject). The following preprocessing steps described, apply to each of the 64 signals.

Firstly, the EEG data was downsampled from 128Hz to 64Hz. Then, it was band-pass filtered within the range of 2-9Hz.

Audio Preprocessing

Firstly, the envelope was extracted from the audio stimuli. Then, this envelope was downsampled from a sampling rate of 8000Hz to a target sampling rate of 64Hz. Finally, a low-pass filter with threshold 9Hz was applied to this downsampled envelope.

Splitting Trials

The EEG data and stimuli were then split into trials 30s long. There were 147 trials per subject.

3.3.2 Comparing EEG and Audio

Firstly, a decoding model is trained to decode the attended stimulus envelope. This decoded envelope is compared to the envelopes of both the stimuli presented to the subject using the Pearson correlation coefficient. The stimulus with a higher correlation is classified as the attended stimulus. This process is evaluated by its ability to correctly predict the envelope of the attended stimulus. This is a binary classification problem.

Decoding Model

We will now be describing the model outline in Section 2.6. Recall the notation used:

- $s_a(t)$ denotes the attended audio envelope at time t
- $s_u(t)$ denotes the unattended audio envelope at time t
- $\hat{s}_a(t)$ denotes the decoded attended audio envelope at time t
- $M(t,c)$ denotes the EEG data at time t at channel c

To predict $\hat{s}_a(t)$, we consider a window of the (preprocessed) EEG data from t to $t + 250$ms. To visualise this, refer back to Figure 2.15 in the background section. Because the sampling rate after preprocessing is 64Hz, we can calculate the number of the samples, N, for a window of length $250ms$.

$$N = \text{sampling frequency} \times \text{window size} = 64 \times 0.25 = 16$$

Now, we can substitute $N = 16$ and $C = 64$ (number of channels) into Equation 4.1 to get our neural decoding model:

$$\tilde{s}_a(t) = \sum_{n=1}^{16} \sum_{c=1}^{64} \mathbf{D}_{nc} M(t + n - 1, c) \tag{3.1}$$

where $\boldsymbol{D} \in \mathbb{R}^{16 \times 64}$ represents the weights of the decoder. Notice that this is simply a linear model that is fit with ordinary least squares regression 2.2.1; as such, it can be equivalently expressed in the following notation:

- $\boldsymbol{x} \in \mathbb{R}^{1024}$ - the vector input to the decoder of size 1024. We have 16 time lags (n) and 64 channels (c); therefore, the vector size is $16 \times 64 = 1024$.
- $y \in \mathbb{R}$ - the output of the decoder as a single number. Represents $\tilde{s}_a$ at a given time t.
- $\boldsymbol{w} \in \mathbb{R}^{1024}$ - the flattened representation of the decoder weights $\boldsymbol{D}$.
- f - the decoder function. Given $\boldsymbol{x}$ as an input, returns a predicted output

  $$\hat{y} = f(\boldsymbol{x}) = w_0 + x_1 w_1 + x_2 w_2 + ... + x_{1024} w_{1024}$$

  . Where $\hat{y}$ is the predicted value of y.

K-fold Cross-Validation

K-fold cross-validation is used to evaluate performance where $K = 147$ is the number of trials. We cycle through each of the trials and use it as the test data. The decoder is then trained on the remaining 146 trials. Once the decoder is trained, it predicts the attended speech envelope $\hat{s}_a$ on the test trial. If $\hat{s}_a$ is more correlated to s_a than s_u, then we have a correct classification. This is is repeated for all trials for all subjects. Algorithm 1 outlines this procedure for one subject.

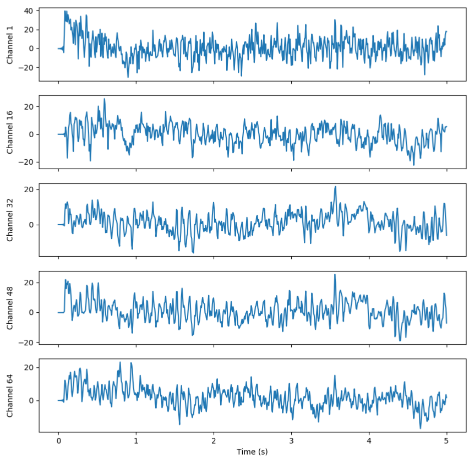
Algorithm 1: AAD procedure for one subject

```
input  : A list of trials, $trials$, for a given subject of length $T$
output: The estimated classification accuracy, $accuracy$, of detecting the attended speaker
$correct \leftarrow 0$;
for $i \leftarrow 0$ to $T$ do
    #split train and test trials;
    $testTrial \leftarrow trials[i]$;
    $trainTrials \leftarrow [trials[0] \dots trials[i-1], trials[i+1] \dots trials[T-1]]$;
    #train decoder on train trials;
    $decoder \leftarrow$ ordinaryLeastSquares($trainTrials$);
    #predict the attended stimulus envelope;
    $\hat{s}_a \leftarrow decoder$.predict($testTrials$);
    $s_a \leftarrow testTrial$.getAttendedEnvelope();
    $s_u \leftarrow testTrial$.getUnattendedEnvelope();
    #calculate the correlation between decoded and both actual envelopes;
    $r_a \leftarrow$ pearsonr($s_a$, $\hat{s}_a$);
    $r_u \leftarrow$ pearsonr($s_u$, $\hat{s}_a$);
    if $r_a > r_u$ then
        #The correct attended envelope has been predicted;
        $correct \leftarrow correct + 1$;
    end
end
#average the results across all trials;
$accuracy \leftarrow correct/T$
```

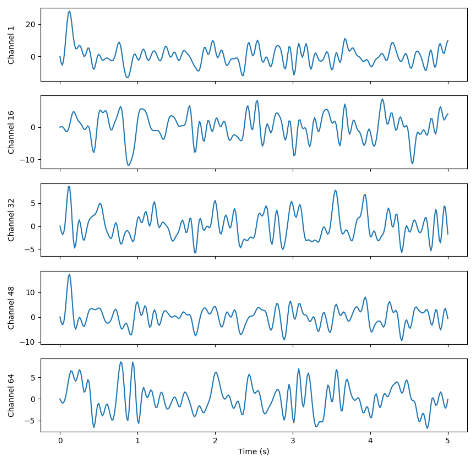
3.4 Visualisation of Data for Troubleshooting

After running the procedure outlined above, the accuracy for each subject around 50%. In other words, the procedure was not able to distinguish the attended envelope. To diagnose at what point the algorithm was going wrong, the code was extended to include simple functions to visualise the signals before and after preprocessing. These visualisations could be used to visualise the signal both in both the time and frequency domain.

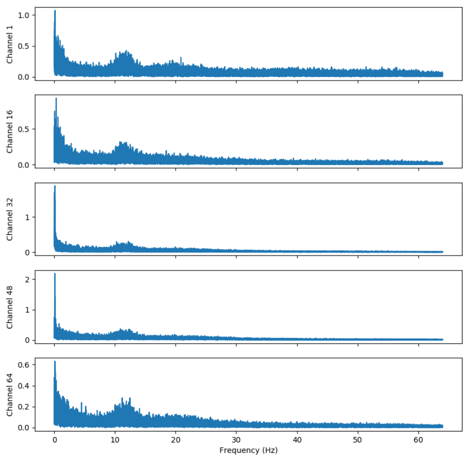
Figure 3.1 shows the effects of preprocessing on some select EEG channels. There seemed to be no problem in this respect. Looking at the time-domain transformation (Figure 3.1a to 3.1b), we see can see the EEG signals have been smoothed out as expected by filtering high frequencies and downsampling the signal. Looking at the frequency-domain transformation (Figure 3.1c to 3.1d), we can see that the band-pass filter with cutoffs of from 2-9Hz has successfully attenuated the frequencies outside this range. Naturally, the nest thing to look at is the audio preprocessing.



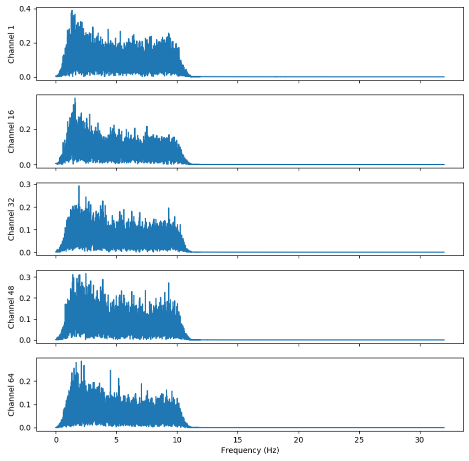
(a) Time-domain



(b) Preprocessed time-domain



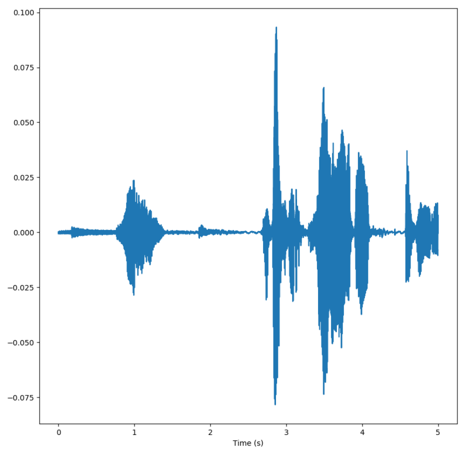
(c) Frequency-domain



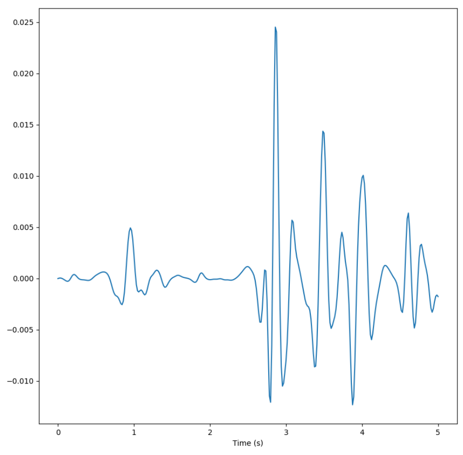
(d) Preprocessed frequency-domain

Figure 3.1: Time-domain and frequency-domain representations for subject 1. From the available 64 channels, the channels 1, 16, 32, 48, and 64 were shown.

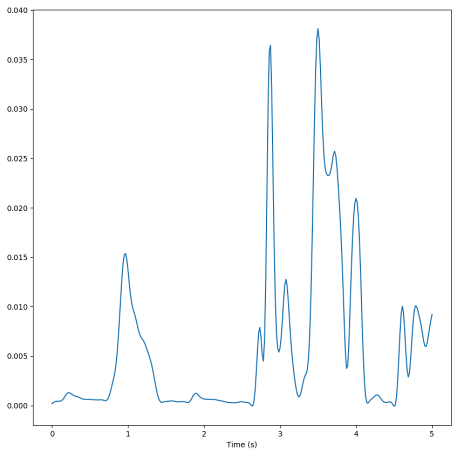
Figure 3.2a shows a sample for one of the audio signals the subject was presented. Figure 3.2b shows a sample of the preprocessed audio, in this case, the envelope after a filter has been applied. Recall from Section 2.4.6 that an envelope is meant to smooth over the amplitude of a signal; as a result, it should have values greater than 0. Notice, however, how in the initial preprocessing shown in Figure 3.2b, the envelope is oscillating about 0. Therefore, a mistake in audio preprocessing was suspected to be the reason for the poor classification performance. After some investigation, the problem seemed to be occurring from the filtering stage after the envelope extraction process. Upon further inspection, it was found that the specific type of filtering used did not fall under the type mentioned in [30]. After this was remedied, the speech envelopes looked more like Figure 3.2c. Notice how the envelope no longer oscillates about 0 and appears to match the approximate shape of Figure 3.2a.



(a) Raw audio track



(b) Initial erroneous audio preprocessing result



(c) Correct audio prepossessing result

Figure 3.2: Preprocessing of 5 seconds of the audio signals.

3.5 Results

After the filtering issue mentioned above was resolved, the results seemed to be in line with [30]. For each subject, the attended stimulus was predicted correctly with per-subject accuracy ranging from around 70-90%.

To make sure these results are statistically significant, we will investigate the probability of getting these results by chance. For each subject, we have 147 30s trials. Assume we randomly guess the attended stimulus each for each trial; therefore, we have a 50% probability of guessing the correct stimulus. Formally, this is a binomial distribution with $n = 147$ trials and $p = 0.5$ as the probability of success. The probability of getting 84 or more correct guesses (57.1% accuracy) is less than 0.05. Therefore, the performances for all subjects are statistically significant at the 5% significance level as seen in Figure 3.3

3.6 Implementation

In this section, we will discuss some relevant implementation details.

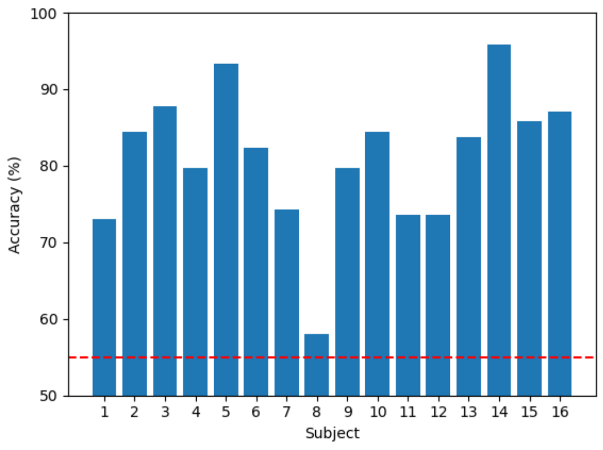


Figure 3.3: AAD results. The bars show the per subject accuracy. The dashed line shows the 57.1% threshold which would have the probability of 0.05 to beat by chance (random guessing). Figure based on Figure 2 from [30].

The choice of language for this toolbox was Python 3 with many libraries to support a wide variety of tools for scientific computing. Notable libraries include:

- NumPy[43]: provides an `ndarray` data structure to store multi-dimensional arrays and operations on them. The shape of an `ndarray` is typically denoted as a tuple. For example, an `ndarray` with 3 rows and 2 columns will have 2 dimensions and shape (3, 2).

- SciPy[44]: provides many modules for scientific computing built on top of the NumPy module. The signal processing module `scipy.signal` is used for most of the signal processing in this project.

- scikit-learn[45]: a machine-learning library used for the neural decoding model among other model evaluation metrics and utilities. It is designed to integrate with NumPy and SciPy.

- Matplotlib[46]: library for plotting graphs with Numpy support.

- MNE[23]: a library for the analysis of neurological data including EEG and MEG recordings. MNE was used to filter the EEG data during preprocessing. Where possible from here on out, MNE will be used for operations on the EEG/MEG data, thus taking advantage of the domain-specific decisions this library will make.

Notice how all the libraries build on NumPy's `ndarray` data structure. Thus, it followed that an `ndarray` was used to represent the EEG and audio data.

For our use case, a signal can be represented using with two fields, the data for the signal itself (the `ndarray`) and the sampling rate. Therefore, it made sense to wrap there two fields in a class to represent a signal - `Signal`. This class had immediate benefits, namely that one would not need to track the current sampling rate (which was subject to change during preprocessing) at all times for any given signal. Furthermore, a lot of the functionality common to signals could be abstracted away.

However, this object style approached turned out to be quite restrictive. A new instance method needed to be created for every additional third-party library function desired. For example, to use a new function, `foo`, from the `scipy.signal` module, a wrapper for this function must be made in the `Signal` class to call it. Furthermore, the concept of a `Signal` class is restrictive if we want to transform the signal into some features that are not a signal anymore (as we will see later). Thus, the object style approach was replaced with a functional style approach.

The class instance methods of `Signal` were extracted into modular functions where appropriate. If a function was simply a wrapper for a SciPy or MNE function, it was removed as we could now call these functions directly. As stated before, `ndarray`'s are nested arrays which have a fixed number of dimensions. Conventions were established to maintain a consistent API. For example, the last dimension for signals is the time dimension. Thus for a block of EEG data of 64 channels and 1000 samples, the convention to represent this data was an `ndarray` of shape $(64, 100)$.

3.7 Summary and Discussion

In this chapter, we have successfully reproduced the AAD results as seen in [30]. Reproducing these results have been useful when gaining familiarity with working with brain signals and audio signals. Having existing research provided a way to validate the components of the toolbox as they were developed. By failing to reproduce the given results by applying the wrong type of filtering, not only did that provide useful information about how to filter audio envelopes, but also demonstrates the benefit of visualising the data to troubleshoot.

Now that this decoding model has been validated. It will be re-applied to the dataset provided by the Centre for Psychedelic Research. This will be seen in the next chapter.

Chapter 4

Decoding Music Envelopes

4.1 Introduction

In this chapter, we will begin to look at the dataset provided by the Centre for Psychedelic Research. Unlike the AAD paradigm in the previous chapter, we are no longer in a setting where we are trying to predict the attended stimulus given two stimuli. Instead, we wish to see how LSD affects the similarity between brain signals and music. This similarity acts as a proxy metric for the degree to which the music is being perceived.

This will be attempted in the following way:

1. Neural decoding will be used as a measure of similarity between brain signals and music. The better that the decoder can reconstruct some representation of the music, the more similar we say the two are.

2. The decoding model will be the same as the one we have seen in Chapter 3. The input will be a window of filtered and downsampled MEG (instead of EEG). The output will be the envelope of the music stimulus.

3. If an adequate decoding model can be found, the effect LSD has on the decoding performance will be analysed.

4.2 Problem Setting

4.2.1 Dataset

We will now go through the details of the dataset provided by the Centre for Psychedelic Research.

In this dataset, we have 13 subjects. For each subject, we have two conditions: the LSD condition and the placebo condition. In the LSD condition, the subject has been given LSD. In the placebo condition, the subject has been given an inert substance with no therapeutic value as a way to see the effects of LSD.

For each of these conditions, the subject is exposed to some music as a stimulus. There were two music stimuli: song A and song B. The subject did not listen to the same song twice. If a subject listened to song A in the LSD condition, they would listen to song B in the placebo condition and vice-versa.

For each subject and under each condition, we have an MEG (Section 2.5.2) recording of their brain signals. The MEG was originally recorded with 271 sensors. Source reconstruction was then applied to localise the responses at 90 regions spread across the brain. Therefore, we have 90 channels for this MEG data. The sampling rate for the MEG data was 600Hz.

Each MEG recording was 7 minutes long, split into 210 2s trials. Some trials were too noisy and were removed accordingly. The trials deemed too noisy were decided by the client. Metadata was provided indicating the indices of the noisy trials. This was used to synchronise the MEG data to the corresponding audio stimulus the subject was listening to. The number of good (not noisy) trials remaining out of the original 210 can be seen in Table A.1 in the Appendix.

4.2.2 Objective

The objective of this chapter is twofold:

- Firstly, we aim to successfully decode the music given the MEG data.
- If the music envelope is decoded successfully, then further assessment will need to be done to assess how LSD affects the decoding performance.

4.3 Preprocessing

The preprocessing stages will be very similar to the AAD preprocessing stages described in Chapter 3. The input and output for our decoding model will be the same (with EEG replaced with MEG).

4.3.1 MEG preprocessing

The MEG data was downsampled from 600Hz to 64Hz. Then it was band-pass filtered in the range 2-9Hz.

4.3.2 Audio Preprocessing

The audio preprocessing steps are identical to the ones used in Chapter 3. The songs were downsampled to 8000Hz. Then, their envelopes were extracted. The envelopes were then downsampled to 64Hz. Finally, they were low-pass filtered with a threshold of 9Hz.

4.3.3 Synchronising the MEG and Music

The 2s windows in the (preprocessed) music which corresponded to the noisy trials were removed. This was done to synchronise the MEG and music.

4.4 Decoding Model

The decoding model will be the same as the previous chapter. The input to the model is the preprocessed MEG and the output is the decoded envelope of the music. The output is a linear combination of the inputs. This can be expressed as a modified version of Equation 3.1 in Chapter 3:

$$\tilde{s}_a(t) = \sum_{n=1}^{16} \sum_{c=1}^{90} \mathbf{D}_{nc} M(t+n-1, c) \tag{4.1}$$

Figure 4.1 shows the times for the input and output to the decoder.

Instead of 64 channels (Chapter 3), we have 90 channels now. The decoder will be trained with ridge regression (Section 2.2.2). Ridge regression is an extension of ordinary least squares regression seen in Chapter 3. Ridge regression was chosen because we have less data than the previous chapter which may lead to models overfitting.

4.5 Decoding Performance Evaluation

The preprocessed data are combined into a single dataset consisting of input-output pairs:

$$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), (\boldsymbol{x}_3, y_3), ...\}$$

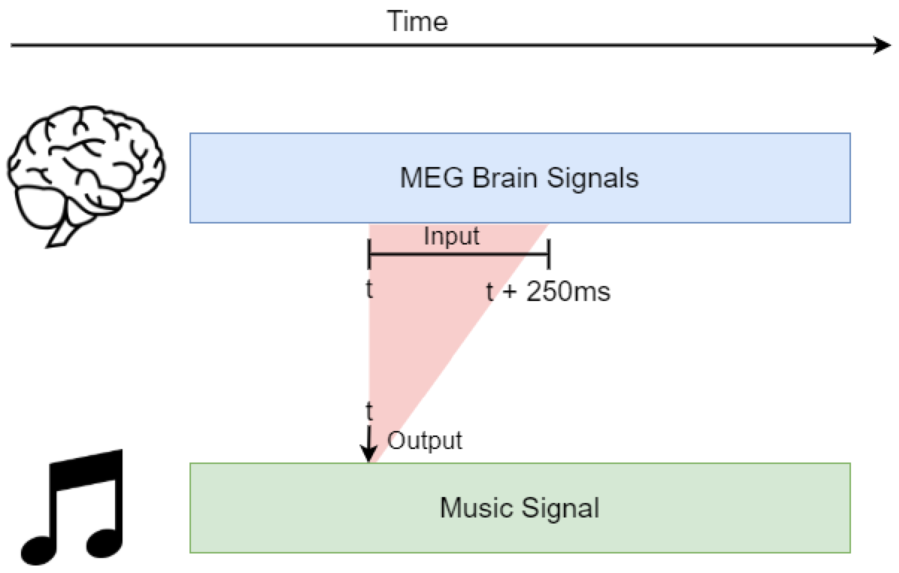


Figure 4.1: The time lags used the decoder. To decode an output of the music at time t, a window of MEG brain signals is in the range t to $t + 250$ms is used as an input.

In this section, we will describe how this data will be used to evaluate the decoding performance.

The metric we will be primarily considering is the R^2 score (Section 2.3.1).

Recall that ridge regression has a regularisation parameter α, this parameter needs to be tuned. To tune this parameter we will be doing nested cross-validation (Section 2.3.2). The outer cross-validation will be used to evaluate the decoding performance. The inner cross-validation will be used to find the best value(s) of α. The outer cross-validation will be k-fold cross-validation with $k = 10$ and the inner cross-validation will be a holdout with a 90:10 split. The details of these steps are as follows:

1. Firstly, split the data (input-output pairs) into 10 partitions. Select 1 partition as a test set. Combine the remaining 9 partitions to form a training set.
2. Take 10% of this new training set to form a validation set. This validation set will be used to evaluate various values of the regularisation parameter α. The remaining 90% of the training set is the new training set.
3. For a given selection of α values - say 0, 0.01, 0.1 and 1 - train the decoder on the training set. Then, evaluate the decoder on the validations set. Choose the value of α which corresponds to the best performance on the validation set.
4. Train a decoder on the whole training set now (training + validation) with this best value of α. Evaluate its performance on the test set (the selected fold).
5. Repeat step 1, with each of the 10 folds acting as the test fold once. Take an average of the decoder's performance evaluated in step 4.

This procedure is shown in Algorithm 2. The procedure is repeated for each of the 13 subjects and each of the two conditions (LSD and placebo) - giving us 26 R^2 scores.

The different values of α tried can be seen in Section C.1 in the Appendix.

4.6 Results

These results can be seen in Table B.1 in the Appendix. As we can see, all the R^2 scores are negative. This indicates that the performance is worse than predicting the mean of the output constantly (ignoring the input entirely). Due to this poor decoding performance, the subsequent analysis of the comparison between the LSD and placebo condition was omitted.

4.7 Implementation

The code was structured by separating runnable scripts and library functions. This meant that generic functions which had the potential to be reused were put in the library function where possible.

The runnable script was the procedure outlined in this section. The library functions included general utilities with anticipated future usage. For example, a simple utility to extract a window of a signal.

4.8 Summary and Discussion

While the linear decoding model worked well in Chapter 3, it did not generalise to our problem at hand - as shown by the poor decoding performance. Therefore, the effects of LSD could not be assessed.

There are some differences between the AAD dataset and our dataset. One particular difference highlighted by the client was the type of audio stimulus. Because music is more complex than speech, it is possible that using envelope extraction as a representation did not generalise well. Looking at Figure 4.2, we can see the more complex music envelope compared to the speech envelope.

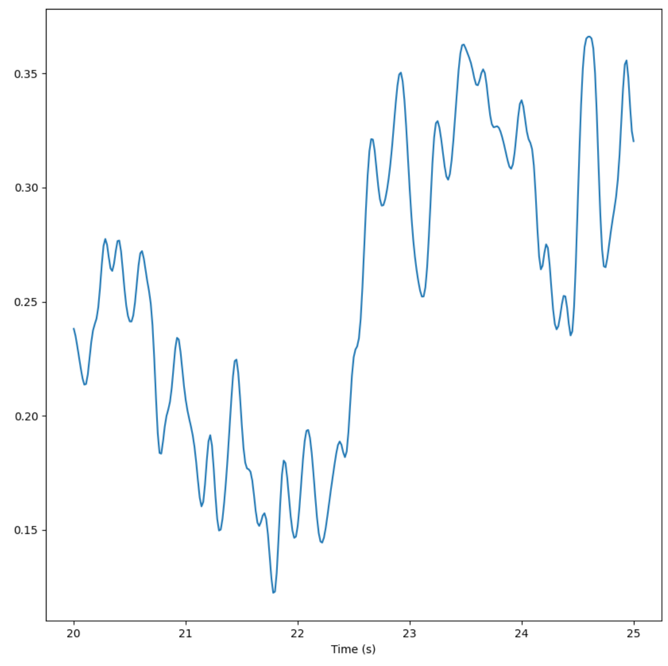
Other notable differences between this chapter and Chapter 3's decoding model are:

- The use of MEG and not EEG.
- The fewer data points available.
- The greater number of channels - 90 vs. 64 - making the decoder more complex (have a larger number of parameters).

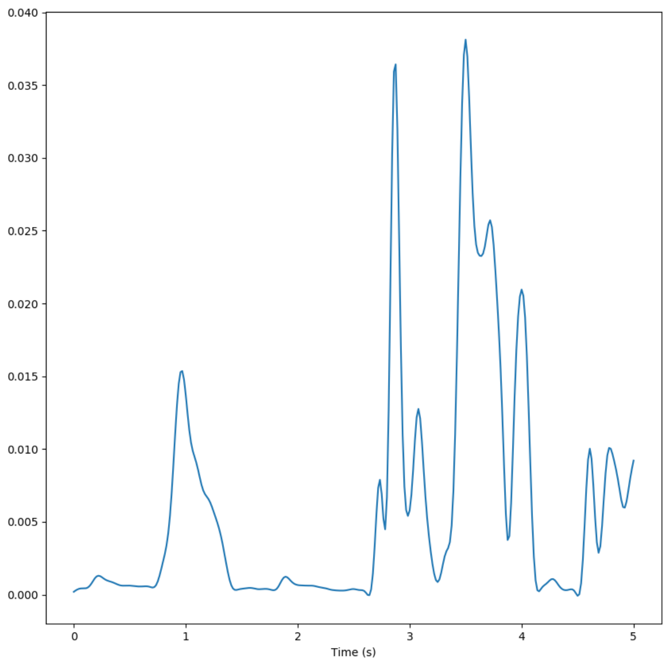
These differences also may have contributed to the decoding model's failure to generalise.

Algorithm 2: Procedure for decoding music envelopes.

```
input  : data: a list of input-ouput pairs $[(x_1, y_1), (x_2, y_2), ...], (x_N, y_N)$ for the model to
         train on.
output: The estimated decoder $R^2$ performance
# split the data into 10 folds;
$folds$ = kFoldSplit($data, k = 10$);
#Keep track of the total $R^2$ score;
$testR2 = 0$;
for $i = 0$ to 10 do
    #Create training and test folds;
    $testSet = folds[i]$;
    $trainSet$ = concatenate($folds[0] \ldots folds[i-1], folds[i+1] \ldots folds[9]$);
    #Further split the training set to get a validation set;
    $trainSet, valSet$ = holdoutSplit($trainSet, valProportion = 0.1$);
    #Keep track of best found $R^2$ score and $\alpha$ that achieved it;
    $bestR^2 = -\infty$;
    $bestAlpha = -1$;
    foreach $\alpha$ in $[0, 0.01, , 0.1, 1, ...]$ do
        #for each proposed value of $\alpha$
        #Get training set inputs and outputs;
        $\boldsymbol{X}_{train} = trainSet.$getRegressionInputs();
        $\boldsymbol{y}_{train} = trainSet.$getRegressionOutputs();
        #Get validation set inputs and outputs;
        $\boldsymbol{X}_{val} = valSet.$getRegressionInputs();
        $\boldsymbol{y}_{val} = valSet.$getRegressionOutputs();
        #Fit a model using ridge regression on training data;
        $decoder$ = ridgeRegression($\boldsymbol{X}_{train}, \boldsymbol{y}_{train}, \alpha$);
        #Evaluate $R^2$ on validation set and adjust best score/$\alpha$ accordingly;
        $\hat{\boldsymbol{y}} = decoder.$predict($\boldsymbol{X}_{val}$);
        $valR^2$ = r2Score($\hat{\boldsymbol{y}}, \boldsymbol{y}_{val}$);
        if $valR^2 > bestR^2$ then
            $bestR^2 = valR^2$;
            $bestAlpha = \alpha$;
        end
    end
    # Combine validation set back into training set;
    $trainSet$ = concatenate($trainSet, valSet]$);
    #Get (entire) training set inputs and outputs;
    $\boldsymbol{X}_{train} = trainSet.$getRegressionInputs();
    $\boldsymbol{y}_{train} = trainSet.$getRegressionOutputs();
    #Get test set inputs and outputs;
    $\boldsymbol{X}_{test} = testSet.$getRegressionInputs();
    $\boldsymbol{y}_{test} = testet.$getRegressionOutputs();
    #Fit a model using ridge regression using the best value of $\alpha$;
    $decoder$ = ridgeRegression($\boldsymbol{X}_{train}, \boldsymbol{y}_{train}, bestAlpha$);
    #Evaluate decoder on the test set and add to running sum;
    $\hat{\boldsymbol{y}} = decoder.$predict($\boldsymbol{X}_{test}$);
    $testR2 = testR2+$ r2Score($\hat{\boldsymbol{y}}, \boldsymbol{y}_{test}$);
end
#Divide the running performance sum by the number of folds to get the mean;
$testR2 = testR2/10$;
```



(a) Envelope extraction of song A sample.



(b) Envelope extraction of the speech sample.

Figure 4.2: A sample speech and music envelope.

Chapter 5

Decoding WaveNet Encodings

5.1 Introduction

In this chapter, we will modify some elements of the previous chapter with the same goals in mind. More specifically, the decoding model will be changed from the one seen in Chapters 3 & 4.

The decoding model will now use WaveNet encodings of the music as the output (Section 2.7.3). The choice of a new representation for the music was due to the suspicion that envelopes were a large factor for the poor decoding performance seen in Chapter 4.

5.2 Problem Setting

The problem setting is the same as the one described in Chapter 4 in Section 4.2. Again, we are using the dataset provided by the Centre for Psychedelic Research. The objective is to first establish a successful decoding model. If this decoding model is established, then the comparison between the LSD and placebo condition will be carried out.

5.3 Evaluation Metrics

We will be using the R^2 metric as seen in Chapter 4 as the primary metric. As a secondary metric, we will also be looking at the mean squared error. A good decoding model will have a high R^2 score and a low mean squared error score.

Recall from Section 2.3.1 the difference between these two metrics. The R^2 score measure how much of the decoding model's outputs are explained by the inputs. The mean squared error can be thought of as how far the outputs are to their targets. Looking at both metrics will give us a more rounded idea of our decoding performance.

When selecting one model over the other during a hyperparameter search, we will use the R^2 score (the primary metric) to choose between them.

5.4 Establishing a Baseline

We will see later on in this chapter that we will need a baseline to compare against our decoding results. The reason a baseline was not established in the previous chapter was that the decoding performance was so poor, indicated by large negative R^2 scores, that this comparison was not necessary.

5.4.1 Constant Estimator

A constant estimator ignores all inputs and always predicts the mean of the outputs of the training data. A simple example of a constant estimator can be seen in Figure 5.1.

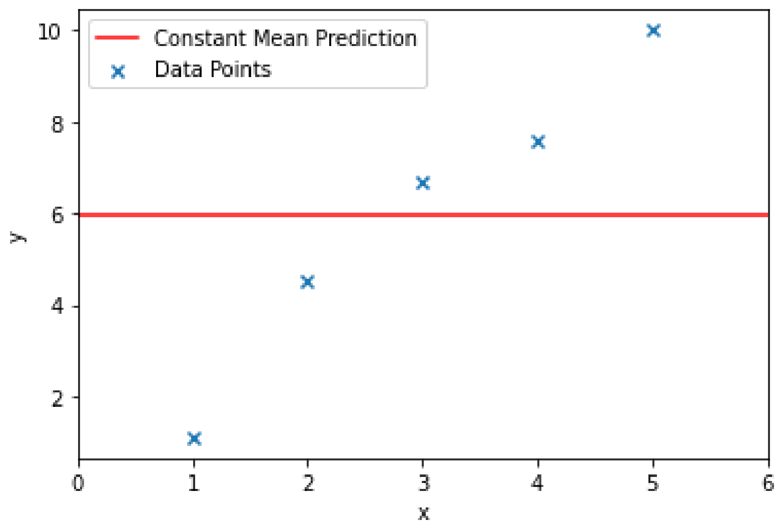


Figure 5.1: An example of a constant estimator on a simple dataset of 1-dimensional inputs and outputs.

5.4.2 Surrogate Dataset

A surrogate dataset is another way to get a baseline metric. Assume we have the dataset, $\mathcal{D}$, of input-output pairs:

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2)(x_3, y_3), ..., (x_N, y_N)\}$$

A surrogate dataset $\mathcal{D}'$ can be made by shuffling the input and outputs breaking the input-output relationship such that the model is effectively making predictions on random noise. An example of what $\mathcal{D}'$ could be is:

$$\mathcal{D}' = \{(x_1, y_3), (x_2, y_7)(x_3, y_1), ..., (x_N, y_4)\}$$

See Figure 5.2 for a visualisation of this. The advantage of a surrogate dataset is that it gives

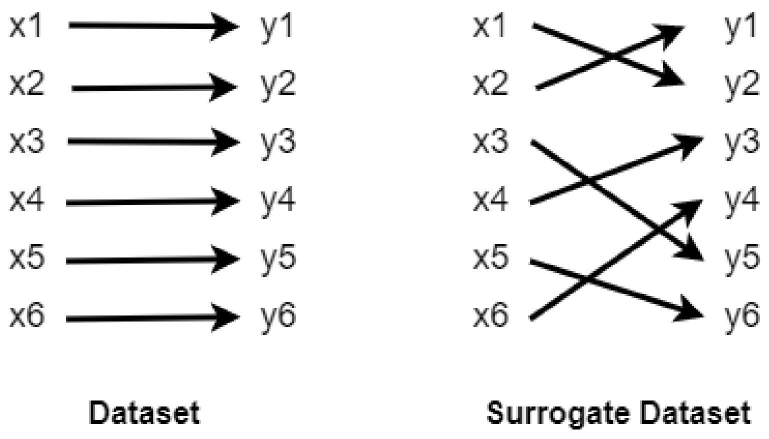


Figure 5.2: A dataset with 6 input-output pairs being shuffled to form a surrogate dataset.

us some idea about the spread of the decoding performance when training on random noise. The surrogate dataset performance will serve as the main baseline when comparing to the actual performance.

5.5 Preprocessing

We will now look at the input and output features of the decoding models after preprocessing has been applied to the MEG and the music.

5.5.1 MEG Preprocessing

The MEG preprocessing steps will now be outlined. Here is each stage of the MEG preprocessing:

1. Firstly, for any given time t in the music, we take a slice of the MEG from t to $t+250$ms. 250ms was used to account for the delay between the stimulus and the response in the brain. This window is the same one seen Figure 4.1 in Chapter 4.

   At this point, the shape of each input array is $(90, 150)$. Where 90 is the number of MEG channels and 150 is the number of samples in 250ms for a 600Hz sampling rate.

2. A wavelet transformation is applied to each of the 90 MEG windows to obtain a scalogram (a time-frequency representation, Section 2.4.7). A scalogram is a matrix that shows the frequency content for each of the 150 samples in a 250ms window. One axis of the scalogram represents different times in the signal and the other represents different frequencies. The frequency axis corresponds to 10 different frequencies in the range from 2-100Hz on a log scale. This range was chosen based on an analysis of the frequency content in the MEG signals and was approved by the client. See Figure 5.3a for an example of a scalogram.

   At this point, the shape of each input array is $(90, 10, 150)$ because a scaleogram matrix of size $(10, 150)$ is computed for each of the 90 MEG channels. The second dimension corresponds to each of the 10 frequencies we are interested in. The third dimension of size 150 corresponds to each sample in the 250ms window.

3. The scaleogram is then downsampled along the time axis. Instead of using all 150 samples in the range t to $t + 250ms$, 10 samples spanning this range are kept. See Figure 5.3b for an example of a downsampled scalogram.

   After this stage, the shape of each input array is $(90, 10, 10)$ which is the same shape as the previous stage with 150 reduced to 10.

4. Then the 3-dimensional array of size $(90, 10, 10)$ is flattened to a single vector of size 9000. Note that this step does not modify the data in any way, it is just a simpler way to represent the same information. We can now thing of the MEG representation as a vector of 9000 features.

5. Finally, feature scaling is applied. Each feature of the 9000-dimensional vector is scaled to have mean 0 and standard deviation 1. This is done by calculating the mean, μ, and standard deviation, σ, for each of the 9000 features across the training data (not the entire dataset to prevent overfitting). For each feature x, the standardised feature, x' is:

$$x' = \frac{x - \mu}{\sigma}$$

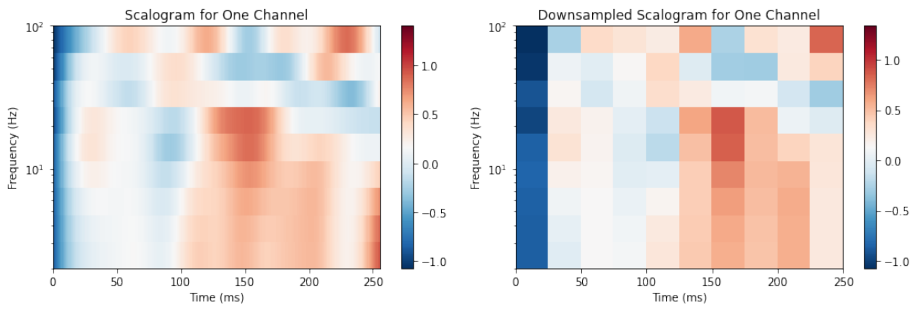
   Feature scaling is done to make each of the features comparable in scale to one another, especially features which correspond to different frequency bins of the scalogram.

To summarise, for each 250ms window of MEG, a time-frequency representation is calculated for each of the 90 source reconstructed regions of the brain (the 90 MEG channels). The time-frequency representation is a scalogram matrix calculated using a Morlet wavelet transform. After preprocessing, a 9000-dimensional vector of time-frequency information is created for each 250ms MEG window. An overview of this procedure can be seen in Figure 5.4.

The choice of a scaleogram as the MEG features the model was requested by the project's client. It was inspired by the work showing how scalograms can be used as a decoding input as seen in Section 2.9. The preprocessing steps outlined above are similar to the ones described in Section 2.9.

5.5.2 Music Preprocessing

Instead of using signal envelopes (Chapters 3 and 4), WaveNet encodings will be used as the representation for music. The details of these encodings can be found in Section 2.7.3. The songs



(a) A scalogram before downsampling in time.

(b) A scalogram after downsampling in time.

Figure 5.3: An example of a scalogram before and after downsampling.

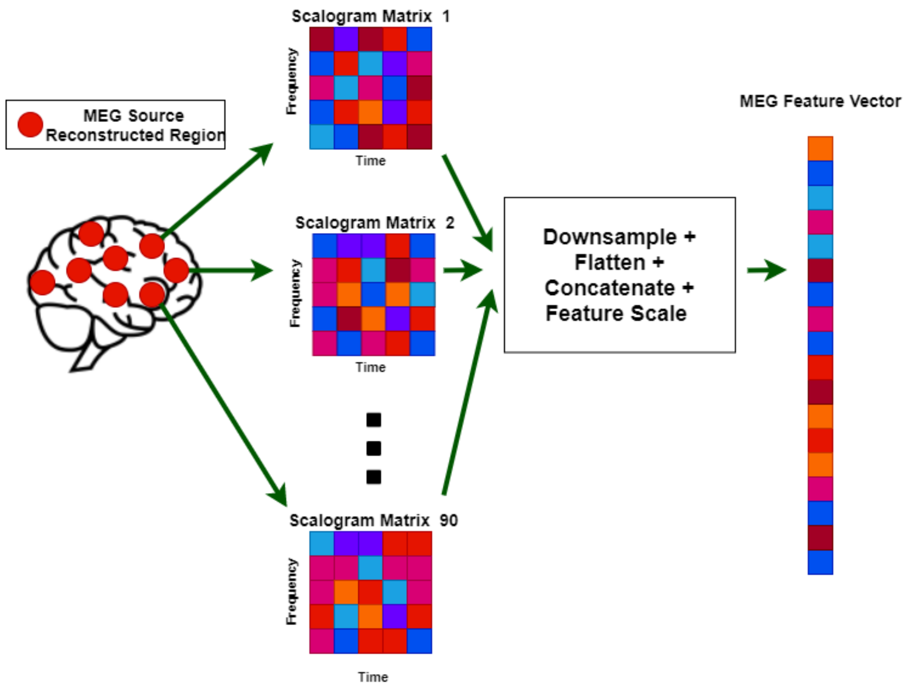


Figure 5.4: An overview of the MEG preprocessing for the decoding of WaveNet encodings.

A and B were passed to a pre-trained WaveNet autoencoder producing encodings of these songs. The encodings were 16-dimensional vectors. Encodings were generated for each 32ms.

The choice of using WaveNet was suggested by the client; however, no specific variant of WaveNet was requested. There were many choices of different WaveNet models to choose from. The WaveNet model created by the Magenta project (outlined in Section 2.7.3) was specifically chosen as it was designed specifically for music. As well as the vast number of instruments it can encode, it was also trained to encode vocals. Vocals were something present in both songs A and B.

5.6 Decoding Model

We will now discuss the 3 decoding models. Some notation:

- $\boldsymbol{x} \in \mathbb{R}^{9000}$ - the MEG features. This is the input to the model.
- $\boldsymbol{y} \in \mathbb{R}^{16}$ - the WaveNet encodings of the music. This is the output of the model.
- f_θ - some decoder parameterised by the hyperparameters θ.

Notice how decoding inputs and outputs are now much larger compared to Chapter 4. Before, the input to the model was a vector of size 1440. Now, the input to the model is a vector of size 9000. Before, the output of the model was just a single number, and now it is a vector of size 16. Because of these higher dimensional inputs and outputs, new regression methods will be proposed.

In Section 2.9, the problems associated with the high-dimensional scalogram input were addressed using partial least squares regression (Section 2.2.4). Recall that partial least squares regression is an extension to principal component regression (Section 2.2.3). For the following experiments, both partial least squares regression and principal component regression were tried alongside ridge regression.

The hyperparameters, θ, differ for the regression model being used. In the case of ridge regression, as we have already seen in Chapter 4, the parameters we will tune is the regularisation strength α. Both principal component regression and partial least squares regression involve a form of dimensionality reduction. The size of this lower-dimensional space is also a hyperparameter. This hyperparameter is referred to as the number of components.

5.6.1 Decoding Performance Evaluation

The decoding performance evaluation for this chapter will be an extension of the one described in Chapter 4 (Section 4.5). The steps are now as follows:

1. Firstly, split the data into 20, partitions. Select 1 partition as a test set. Combine the remaining partitions as the training data.

2. Take 10% of this training data to form a validation set. This validation set will be used to evaluate the different hyperparameters.

3. We have a selection of hyperparameters $\{\theta_1, \theta_2, ...\}$. For each $\theta \in \{\theta_1, \theta_2, ...\}$, train a decoder f_θ on the training data. Then, evaluate the decoder on the validation data. Choose the best hyperparameters $\hat{\theta}$ as the ones that correspond to the best performance on the validation data. The best performance is given the one with the higher R^2 score.

4. Train a decoder on the whole training set (training + validation) with $\hat{\theta}$. Evaluate its performance on the test set (the selected partition on stage 1). In this evaluation, we will be computing both the R^2 score the mean squared error.

5. Train a constant estimator on the whole training set and evaluate it on the test set to get the constant baseline performance.

6. Shuffle the train and test inputs thereby breaking the relationship between input and output. This dataset is the surrogate dataset. Then train the decoder with the best hyperparameters, $f_{\hat{\theta}}$, on the surrogate training data and evaluate it on the surrogate test data.

7. Repeat step 1, with each of the 20 folds acting as a test fold once. Take the mean R^2 and mean squared error across all of the folds.

This procedure is the same as the one described in Algorithm 2 in Chapter 4 with some extensions. Firstly, we generalise the hyperparameters depending on the regression method used. Secondly, we are recording the mean squared error as a secondary metric as well as the R^2 score. Thirdly, at the end of each hyperparameter search (one for each test fold) we evaluate the model with two baselines: constant estimator and performance on the surrogate dataset.

This procedure is repeated for each subject under each condition. Therefore, for each subject and condition, we have 3 numbers. The actual performance, constant baseline performance, and surrogate baseline performance.

The values of the hyperparameter θ used for each regression method can be seen in Section C.2 in the Appendix.

5.7 Results

5.7.1 Ridge Regression

The ridge regression performance was very poor as can be seen in Table B.2 in the Appendix. All of the R^2 scores are negative indicating the model can't fit onto the data and is performing worse than naively predicting the mean of the outputs. One possible reason is the complexity of this model in terms of the number of parameters. With 9000 input features and 16 output features, a ridge regression model (a linear model) has $9000 \times 16 = 144000$ parameters. Assuming we have no bad trials and have the full 7 minutes of data (one subject in one condition), we have around 13000 data points. In the best-case scenario, we have nearly 10 times the number of parameters to fit as the number of data points which makes training a model difficult. This model is very prone to overfitting (see Section 2.3.2).

5.7.2 Partial Least Squares Regression

The full results of partial least squares regression can be seen in Table B.4 in the Appendix. In both the LSD and placebo condition, the decoding R^2 performance (as well as the mean squared error) outperformed the surrogate baseline performance across all subjects. This was shown to be statistically significant ($p < 0.05$, Wilcoxon signed-rank test).

5.7.3 Principal Component Regression

The full results for principal component regression can be seen in Table B.3 in the Appendix. Similar to the partial least squares results, the decoding R^2 (and mean squared error) performance outperformed the surrogate baseline across all subject. Again, this was statistically significant ($p < 0.05$, Wilcoxon signed-rank test). Principal component regression performs the best out of all the three decoders tried; consequently, we will examine these results from here on out. A visualisation of these results can be seen in Figure 5.5 comparing the R^2 and mean squared error score of each subject and condition to both the constant and surrogate baseline performance. As we can see, every decoder achieves a lower mean squared error and higher R^2 score compared to the constant and surrogate baseline.

Recall that the advantage of using a surrogate baseline was to get an idea about the spread of the baseline performance. This can be seen in Figure B.1 in the Appendix for one subject. Looking at the spread of the data for this subject is a good visual aid to see the baseline performance(s) being beaten.

Figure 5.6 shows the best number of components found when doing a hyperparameter search for principal component regression. We can see that most of the data lie in the 200-440 range. Compare this to the original input dimension of 9000. This shows that a small proportion of the high-dimensional input is required to beat the baseline performance(s).

5.8 Alternative Decoding Approaches

In this section, we will go over some alternative attempts at finding a decoding model. These results were not successful at beating the baseline performance.

5.8.1 Non-Shuffled Output Pairs

Recall from Section 2.3.1 that a predictor which always predicts the mean of the outputs should have an expected R^2 score of 0. The interpretation of this being that the variance in the outputs is not explained by the inputs (which are completely ignored). Therefore, we would expect the mean R^2 score across all the test folds for the constant estimator baseline to be close to 0. However, this value was very negative with values approaching -1. The client requested that this be investigated before looking at the decoding performance.

The explanation for this result was the way the data was being partitioned for k-fold cross validation. Our dataset of input-output pairs is ordered by time. Let us ignore the inputs because

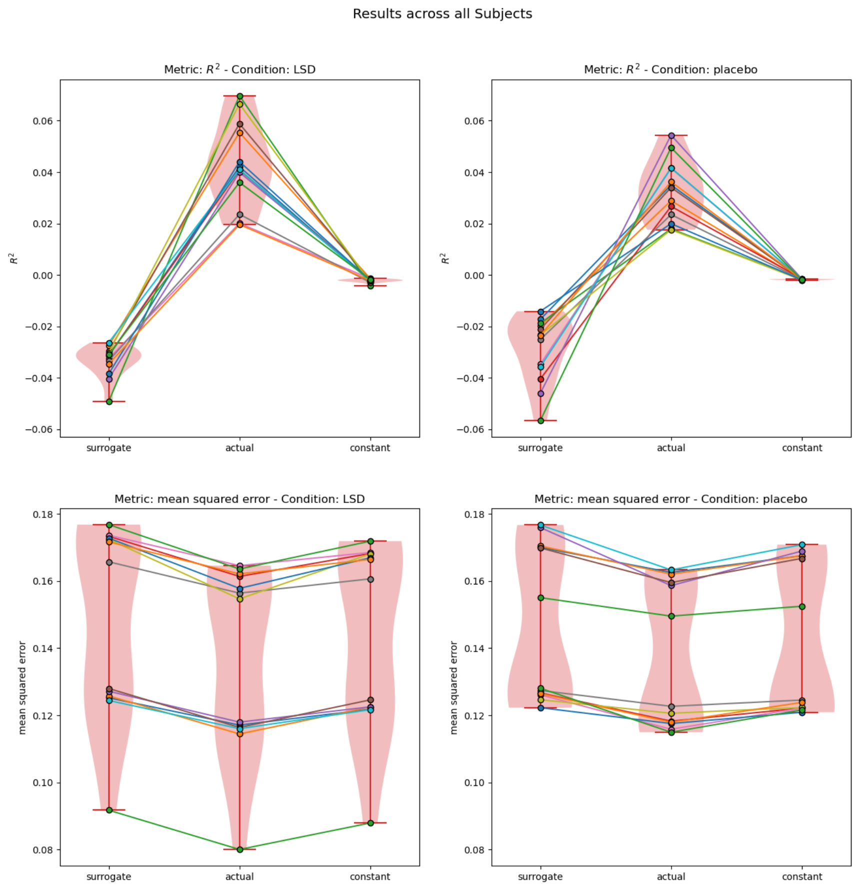


Figure 5.5: Results across all subjects. The x-axis denotes the type of performance being measured. the surrogate dataset baseline performance ("surrogate"), the actual performance ("actual")) and the constant estimator baseline performance ("constant"). The y-axis shows the scores of the metric being evaluated. The top two graphs show the R^2 scores and the bottom two show the mean squared errors. The left graphs show the LSD condition and right graphs show the placebo condition. Each line represents the performance of one subject. The shaded regions are violin plots showing the distributions of the performances for each subject.

they are ignored by the constant estimator. The outputs of the dataset represent 7 minutes (420 seconds) worth of encodings for the given song. When splitting the dataset into 10 partitions, we would have one continuous 42-second chunk used as the test data. The mean of the training outputs was not representative of the mean of the test outputs, therefore the constant estimator model achieved an R^2 score of much less than 0.

The solution to this was to shuffle the pairs of data samples before partitioning the dataset. Therefore samples across the whole song were spread out across the different partitions. Note that this is not the same as the shuffling described with the surrogate dataset in Section 5.4.2. In that form of shuffling, the inputs and/or outputs were shuffled (independently) to break the relations between the input-output pairs. In this form of shuffling, the pairs themselves were shuffled. Take the ordered dataset of pairs, $\mathcal{D} = \{p_1, p_2, p_3, ...p_N\}$ where p_i is on input-output pair $p_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$. The shuffling of this dataset before partitioning will yield a new dataset $\hat{\mathcal{D}}$ which might have the form:

$$\hat{\mathcal{D}} = \{p_3, p_5, p_N, ..., p_6\}$$

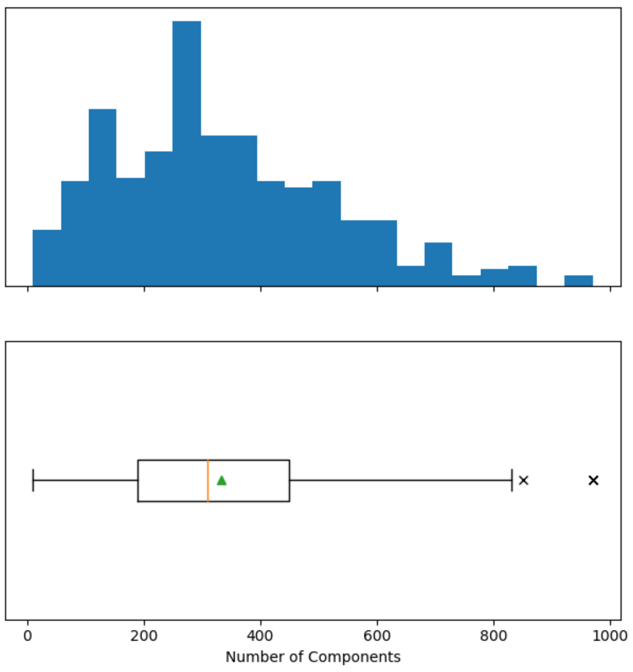


Figure 5.6: Distribution of the best number of components found during a hyperparameter search for principal component regression. The x-axis shared between both plots show the number of components (dimensions) kept after performing principal component analysis for dimensionality reduction. The top graph shows a histogram of the distribution of these numbers of components. The bottom graph shows a box and whisker plot. The box shows the interquartile range of the data. The orange line cutting the box shows and green triangle show the median and mean of the data respectively. The whiskers show the furthest data points 1.5 times the interquartile range above and below and third and first quartile respectively. The crosses determine all other points outside the whispers.

After this change, the constant estimator's R^2 performance was closer to 0 - in line with what was expected.

What this investigation shows is the non-stationary nature of the music. Informally, its properties change throughout the song. Therefore, under the approval of the client, the decoder was trained on these shuffled pairs which are representative of the whole song. The idea behind this was that the decoder should be trained on samples representative of the whole distribution of data.

5.8.2 Using Whole 2 Second Trials

Initially, the inputs and the outputs of the decoding model corresponded to the entire 2s trial. Each trial had 62 WaveNet encodings, each of size 16. These encodings were concatenated into a single vector of size $62 \times 16 = 992$. The scalograms were still downsampled to 10 time samples across the two seconds so the decoder input dimensionality of 9000 was the same.

Assuming the best-case scenario where no trials have been removed, our dataset has only 210 entries. Furthermore, the decoders were more complex than the ones described above due to their output dimensionality being 62 times larger. This overly complex model was failed to train with so few data points. Recall the problems faced when training the ridge regression model in Section 5.7.1 with so many model parameters compared to the number of data points. This problem was even worse when using the whole 2 second trial for each input-output pair. There was 62 times fewer data and 62 times the output size (and therefore 62 times more parameters in ridge regression).

A second reason to not use the whole 2s trials for each input-output pair was based on neuroscience domain knowledge. The client recommended to not do this because a lot of brain activity goes on in 2 seconds. Therefore, it made more sense to split these trials up into smaller chunks and perform decoding at a more fine timescale.

For these reasons, the trial was split into (just under) 62 outputs - one corresponding to each 16-dimensional vector as seen above.

5.9 Comparing LSD and Placebo Condition

Recall the objectives stated in 5.2. Now that the music representation has been successfully decoded, we have some similarities scores between the brain signals and the music - the R^2 score showing the decoding performance. In this section, we will analyse how LSD affects this notion of similarity.

Looking at Figure 5.7, we can see that the mean R^2 score was slightly higher for the LSD condition than the placebo condition. The mean placebo R^2 score is close to 0.03 and the mean LSD R^2 score is close to 0.04.

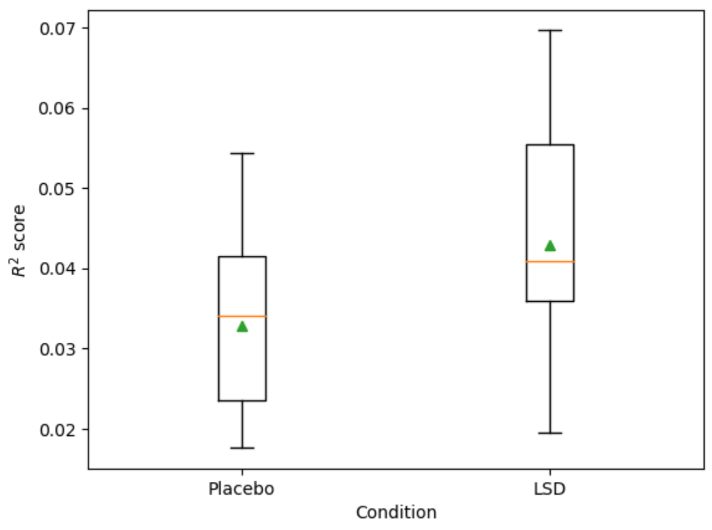


Figure 5.7: A box and whisker plot comparing the LSD and placebo performance. The box shows the interquartile range of the data. The orange line cutting the box and green triangle show the median and mean of the data respectively. The whiskers show the furthest data points 1.5 times the interquartile range above and below and third and first quartile respectively.

We can not simply conclude that the LSD condition outperforms the placebo condition. Two things must be accounted for:

1. Firstly, we need to account for other factors which may have affected the performance. The condition of LSD or placebo is the factor we wish to assess. However, the subject and song in each condition are two factors that also may affect performance. Figure 5.8 shows us the per-subject difference in performance between the LSD and placebo condition while also highlighting the song being listened to in the LSD condition.

2. Secondly, it is necessary to test that this empirical improvement in performance is statistically

significant between the conditions. This will account for both the size of this performance increase (which, as we have seen, is quite small) and the number of subjects - 13.

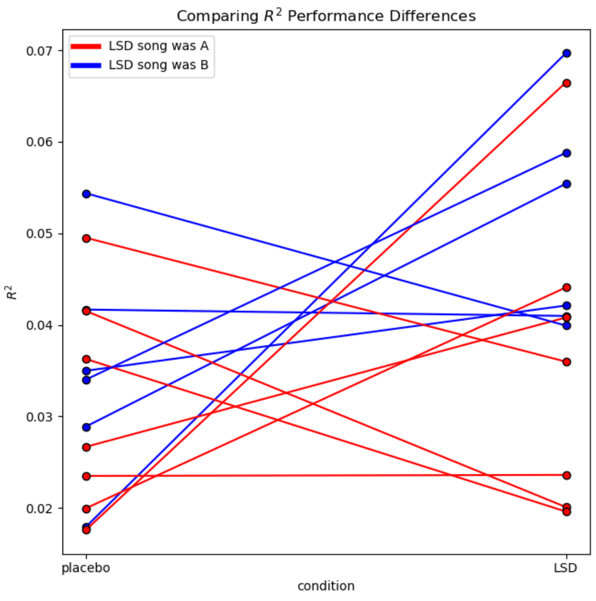


Figure 5.8: Graph showing the change in the R^2 from the LSD to placebo condition. The lines are color-coded by which of the two songs each subject was listeing to in the LSD condition.

5.9.1 Linear Mixed Effects Model

Advised by the client, a linear mixed effect model was suggested to account for the other factors which may affect the decoding performance/similarity: the subject and the song listened to for each condition. Some notation:

1. s - the subject. This is a categorical value corresponding to each subject.
2. m - the music the subject was listening to. This is a binary variable with value 1 for song A and value 0 for song B.
3. c - the condition (LSD or placebo) variable. This is a binary variable with value 1 for the LSD condition and value 0 for the placebo condition,
4. r - the R^2 score observations. This is a continuous variable.

A linear mixed effect model is a model which considers both fixed and random effects. The fixed effects are the variables that we are interested in seeing the effect of, in this case, the condition c. The random effects are the variables that we are not directly testing but may also contribute to our observations r. In this case, the random effects are the subject and music (s and m). The specific linear mixed effect model that will be used is:

$$r = a + b \times c + a_s + a_m + e \tag{5.1}$$

The first part of the equation, $a + b \times c$ is a simple linear transformation of the fixed effect c with gradient b and (global) intercept a. The e term represents some noise/error term with mean 0. The terms a_s and a_m are random intercept terms corresponding to the random effects s and m respectively. So, alongside this model's global intercept a, there is a subject-specific intercept a_s and a song specific intercept a_m.

The linear mixed effects model was fit with the 26 data points available (13 subjects with performance measures in both conditions). Each data point is the subject, music, and condition as the mixed effects and the mean cross-validation R^2 performance as the observation.

The client suggested to use a linear mixed effect model; however, this can take many forms. A specific model was not recommended and this was a decision that needed to be made. The choice of model used in Equation 5.1 is a model where the random effects change the intercept of the model. It is also possible to have random effects change the gradients (or both). These models, however, are more complex and would be harder to justify with only 26 data points. Therefore, a simpler model was favoured.

5.9.2 Likelihood Ratio Test

To test the significance of the condition on the R^2 observations, a new model was constructed by adapting Equation 5.1:

$$r = a + a_s + a_m + e \tag{5.2}$$

In this model, the condition, c, is ignored.

A likelihood ratio test is used to compare how both of these models fit the data. It tests whether the linear mixed effects model described in Equation 5.1 fits the data better than the same model with the condition term removed as seen in Equation 5.2. If this improvement in fit is seen, it suggests that the condition affects the R^2 decoding performance.

5.9.3 Results

The results of the likelihood ratio test were not statistically significant ($p = 0.06952 > 0.05$). Therefore, we have not found any significant difference between the LSD and placebo condition.

5.10 Implementation

As was done in Chapter 4, a new runnable script was created for the decoding stage described in this chapter. Common functions between this chapter's script and the script corresponding to the previous chapter were refactored into library functions where possible.

Computing the WaveNet encodings for the songs was a lengthy process and would around 30 minutes on a local development machine. Another problem using this pre-trained WaveNet model is that it required dependencies outside of the Python ecosystem, thus making it hard to port to other machines such as a more powerful machine to run the training procedure outside of development. The solution to this was to save these WaveNet encodings once and load them, rather than compute them every time.

One script was used to train the decoders and another was used report decoding performance (and produce the graphs seen in this chapter). Decoding results were saved into a specifiable output folder. The choice to save the results to load them for analysis later was made due to how many models were being trained training. For each subject, we have two conditions. For each condition, we run 20-fold cross-validation. For each of these 20 folds, we experiment with a variety of hyperparameters as well as the surrogate dataset. Therefore, 100s of decoders were trained, a lengthy process. Serialising these results meant that once the results were generated, they could be analysed and visualised without needing to run the entire training procedure again. However, this did introduce a coupling between these scripts. When modifying the decoder training script, the analysis script, which expects the data in a particular format, needed to be kept in mind

Although it would be ideal to keep the code all under the Python ecosystem. When analysing the difference between the LSD and placebo condition, an R programming language script was used instead. R has an advantage that statistical models can be written in a few lines of code. The script to load the data, create the two linear mixed effect models, and calculate a likelihood ratio test (Section 5.9) was 4 lines of code.

5.11 Summary and Discussion

In this chapter, we successfully decoded a representation of the music from the brain signals. The decoding model has the following components:

- The input to the decoder are scalograms of the brain signals. One scalogram is created for each of the 90 channels of the MEG data.
- The output of the decoder is a 16-dimensional WaveNet encoding of the music.
- Principal component regression is used to map this relation from inputs to outputs.

Although this model has shown to beat the baseline performance, the difference between the LSD and placebo condition was found to be insignificant.

There are some limitations to the approaches and results in this chapter:

- Although the decoding performance was better than baseline, the R^2 scores were still small (close to 0).
- The musical features are uninterpretable. WaveNet encodings represent some lower-dimensional representation of the music which can't be related back to features that humans can understand such as tempo or rhythm.

A more detailed assessment of the limitations will be outlined in the evaluation (Chapter 7).

In the next chapter, we will look at interpretable musical features to address the second limitation mentioned above.

Chapter 6

Decoding Short Term Musical Features

6.1 Introduction

In the previous chapter, the output of our model were WaveNet encodings of the music. In this chapter, we will be looking at short-term musical features which affect the perceptual experience of music. The motivations behind this are as follows:

- Unlike WaveNet encodings seen in Chapter 5, the features will be interpretable. This means we know how they are calculated (unlike a black-box neural network). Additionally, we also have some idea as to what musical properties they correspond to.
- We will be looking at different musical features. This would mean that we can investigate how the perceptual experiences differ for these features separately.
- This may also give us a better decoding model than the one seen in the previous Chapter which had low R^2 scores.

6.2 Problem Setting

The problem setting is the same as the one described in Chapters 4 (Section 4.2) and 5 (Section 5.2). In this chapter, the music will be represented as one of the short-term acoustic features outlined in Section 2.8 in the background.

The first objective is to successfully decode the chosen acoustic features (separately). For the features that are successfully decoded, the effect LSD has on the decoding performance will be analysed.

6.3 Evaluation Metrics

Again, we will be using the R^2 score as the primary metric for evaluation. The mean squared error was also calculated but will be omitted as it provides no further meaningful analysis in this chapter

6.4 Preprocessing

6.4.1 Musical Features

The features will be the short-term acoustic features outlined in Section 2.8, which affect the perceptual experience of music. Refer to this section for more details about their definitions. The features are as follows:

- RMS energy

- zero crossing rate
- spectral centroid
- high energy-low energy ratio
- spectral spread
- spectral roll-off
- spectral entropy
- flatness
- roughness
- spectral flux
- sub-band flux for 10 different sub-bands

Each one of these features corresponds to a single (real) number at a given time t in the song. The RMS energy feature corresponds to the loudness of the song. The remaining features are all timbral features, which relate to the characteristics of a musical sound which are unique to different instruments/voices. These features are considered short-term musical features and were calculated on short windows of music. These windows were 25ms in length and consecutive windows overlapped by 50%. Figure 6.1 depicts this for one feature.

6.4.2 MEG features

As suggested by the client, we will consider a window of MEG at the same resolution as the musical feature we are examining. For the window of each musical feature corresponds to the time range $[t, t + 25\text{ms}]$, we will consider MEG in the range $[t + \delta, t + \delta + 25\text{ms}]$. The MEG window is of the same length delayed by a time lag δ. This can be seen in Figure 6.2. The δ parameter is meant to account for the delay between hearing the music and the response in the brain. This delta parameter was varied in the 170ms-210ms range as this was shown to be effective in the AAD paradigm[10]. We can represent this window of MEG as a two-dimensional matrix of size 90×15, $\boldsymbol{M} \in \mathbb{R}^{90\times 15}$ for a 600Hz signal in 25ms $(150 = 600 \times 0.025)$.

Three representations of this MEG window, $\boldsymbol{M}$, were considered:

- Firstly, we square every element in $\boldsymbol{M}$. Then the signal is downsampled in time to keep 5 evenly spaced samples of the original 15. Then, the result is flattened into a single vector of size 1350, $x \in \mathbb{R}^{450}$.

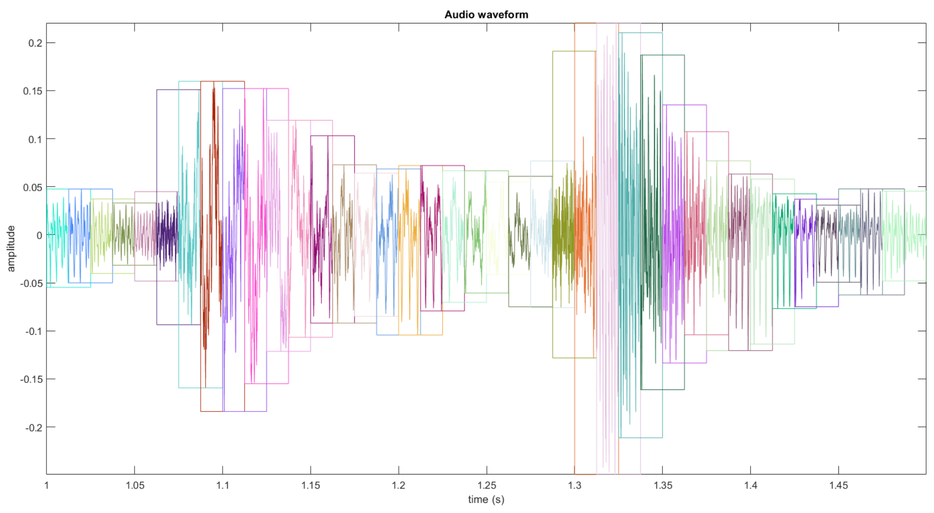
  The squaring step was advice from the client. A squared signal is its instantaneous power. The advantage of squaring the signal is that we get an all positive input and don't have to consider the signals fast oscillations about 0. The downsampling was done to space the samples of the MEG signals out. More spaced-out samples are less correlated with each other, reducing the problem of multicollinearity of features discussed in Section 2.2. Another advantage of downsampling is that it reduces the dimensionality of the input by a factor of 3, thus making the model's less complex and prone to overfitting.

- Taking the square of each element in $\boldsymbol{M}$ and then taking a mean across the 15 samples in time for each of the 90 channels. This gives us a vector $\boldsymbol{x} \in \mathbb{R}^{90}$ where each value in $\boldsymbol{x}$, x_i, is given by:

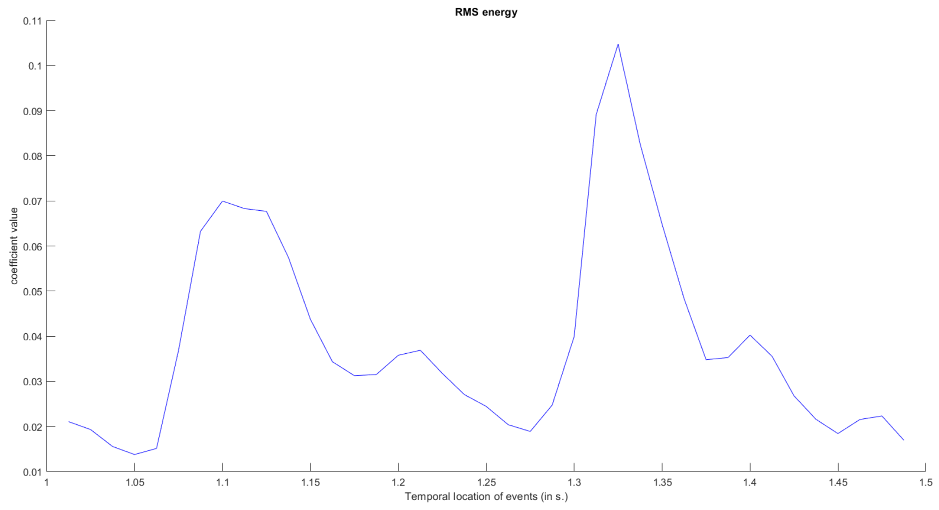
$$x_i = \frac{\boldsymbol{M}_{i,1}^2 + M_{i,2}^2 + \cdots + M_{i,14}^2 + \boldsymbol{M}_{i,15}^2}{15}$$

  where $\boldsymbol{M}_{i,t}$ represents the MEG window at channel i and time sample t.

  This is similar to the instantaneous power above except it is averaged over time. This can be interpreted informally as the "amount" of activity at 90 different regions of the brain.



(a) Overlapping windows of music.



(b) RMS energy for each window.

Figure 6.1: Overlapping windows of a sample of music and the RMS energy computed for each window.

While some temporal information is lost, the advantage is the smaller dimensionality input. As mentioned above, this leads to simpler models which could make decoding less prone to overfitting.

- A per-channel scalogram matrix was calculated as was done in Chapter 5, Section 5.5.1. The only difference was the number of samples kept along the time-axis of the matrix. Instead of 10 time samples, 5 samples were used, thus giving us 90 scalograms of size 5×10. When flattened, we have a single vector of size 4500, $\boldsymbol{x} \in \mathbb{R}^{4500}$.

  Many of the musical features are calculated with information about the frequency (notably those prefixed with "spectral"). Therefore, scalograms were a way to incorporate the frequency information of the MEG signals in the model.

After these steps, feature scaling is applied to each feature of the preprocessed-input so the data has mean 0 and standard deviation 1 (the last stage of scalogram preprocessing seen in Section 5.5.1).

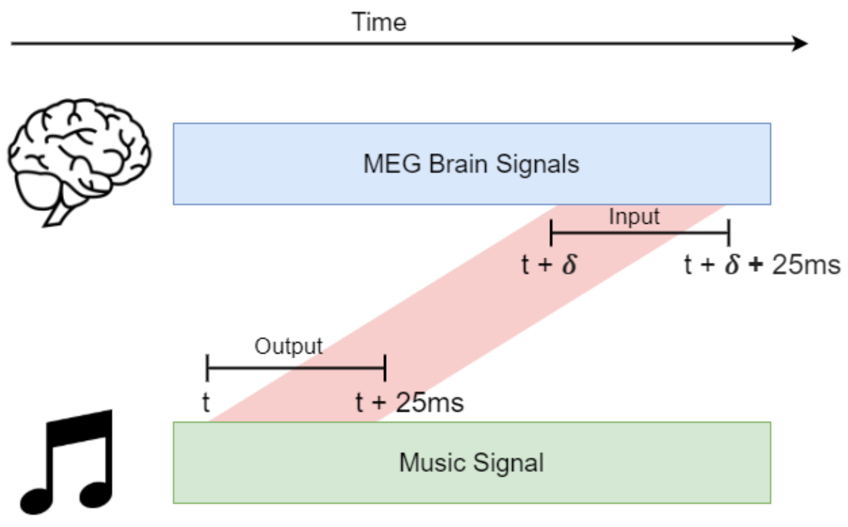


Figure 6.2: The times in the MEG and music signals of the input and output of the decoder.

6.5 Decoding Model

In [3] (psychedelic research), these musical features were all used together as a representation of the music (after applying dimensionality reduction). The result was a vector that represented the music, much like the WaveNet encodings seen in Chapter 5. In [3], these musical features were used as an input to a regression model, with brain activity being the output (the opposite of decoding).

In this project, these musical features will be the output of the regression methods because we are concerned with neural decoding. The features will be looked at individually. This way, we can investigate how LSD affects the perception of specific, interpretable musical features. For example, we might find that under LSD, the brain signals are similar to the music's roughness but not its RMS energy (assuming that both of these acoustic features have been successfully decoded). Another advantage of decoding the features separately is that the difficulties training complex models with high-dimensional outputs (as seen with WaveNet encodings in Chapter 5) are avoided.

In notation, the decoding can be described as:

- $\boldsymbol{x} \in \mathbb{R}^m$ - the m dimensional input representation after the preprocessing steps in Section 6.4.2.
- $y \in \mathbb{R}$ - the given musical feature (a real number) output seen in Section 6.4.1.
- f_θ - the decoder mapping the inputs to outputs parameterised by θ.

The choice of the decoder will be dependent on the choice of the MEG preprocessing steps. If scalograms are used, we will have a high dimensionality input of 4500 and principal component regression will be used as this has shown to work in Chapter 5. Otherwise, ridge regression was used.

6.5.1 Decoding Performance Evaluation

In this section, the decoding performance evaluation will be outlined. This procedure will be applied per-feature, per-subject, and per-condition.

Again, nested cross-validation will be used as seen in Chapters 5. The outer cross-validation will be k-fold cross-validation to evaluate the decoding performance. The inner cross-validation will be a holdout method to find the best parameters. Unlike Chapter 5, however, 5-fold cross-validation will be done instead of 10. This was done to account for the fact that this procedure is repeated for each one of the 20 features mentioned in Section 6.4.1; therefore, 5-folds was used to reduce the total time training decoders.

The time lag δ(Section 6.4.2), was originally set to 200ms. The entire procedure outlined above was also repeated for δ values of 175ms and 225ms. These values were inspired by AAD research that found that values in this range 170 - 250ms corresponded to the important time lags when reconstructing audio speech stimuli[10].

The values of the hyperparameters, θ, used for each regression method can be seen in Section C.3 in the Appendix.

6.6 Results

The results for $\delta = 200$ms can be seen in the following tables in the Appendix:

- Table B.5 for the instantaneous power MEG preprocessing.
- Table B.6 for the mean power MEG preprocessing.
- Table B.7 for the scalogram MEG preprocessing.

In each of these tables, notice the very negative R^2 score for each feature, indicating that the models have not fit the data well. Similar results were found for the other values of δ tried. Despite the various attempts outlined in this chapter, these musical features were not successfully decoded. Therefore, the subsequent analysis of the effect LSD had on decoding performance could not take place.

6.7 Implementation

A new runnable script was created corresponding to this chapter. Again, common utilities between all three runnable scripts were refactored into library functions where possible.

To implement the music preprocessing seen in this chapter, a Matlab script was written to prepossess the musical features. This was done because the toolbox used to extract these feature, MIRToolbox[37], is a Matlab library. A script was written to accept a song as the input (either song A and B), calculate these features, and save them locally. These saved musical features were then loaded into Python.

6.8 Summary and Discussion

In this chapter, we have seen some attempts to decode specific musical features using three variations of MEG preprocessing as an input. These attempts were unsuccessful. This is not to say that the brain signals don't reflect these musical features. Assuming there is some relation between the brain signals and these musical features, here are possible reasons that this relationship may have failed to be captured through neural decoding:

- We have considered both a time-domain (a function of time) and time-frequency representation (function of both time and frequency) representation of the MEG data. It is possible that these MEG features might not be the correct features to use to show any potential underlying similarity between brain activity and the musical features. Perhaps another representation, such as a frequency-domain (function of frequency) representation would be more suitable.
- The choice of regression models may not be suited to express this relationship. We have looked a linear relationship between the MEG features and the musical features. It is possible that the true underlying relationship between the brain signals and the musical features is more complex than a linear relationship and thus we may need a non-linear model (such as neural networks) to represent it.
- The choice of time lag parameter, δ was inspired by research where the reconstructed stimulus was an envelope of speech[10] and not music. It is a possibility that these musical features could correspond to the MEG for different δ values than were tried in this chapter.

- It is possible that the 25ms window which a musical feature represents may correspond to a window larger than 25ms of MEG, rather than a 25ms window as we have tried.

It is worth noting that only the short term musical feature - all but one of which correspond to timbral features - were being assessed. There were three musical features in [3] that correspond to long term musical features that we have not looked at. These long term features would correspond to tonal and rhythmic features of the music[38]. The reason that short-term features were chosen, was because:

- Most of the features which affect perceptual experiences were short-term features so short term features were a natural starting point. They were all calculated at the same time resolution so scripts using them could simply switch out the feature being decoded.
- Long term features were calculated over much larger window sizes of 3s with 33% overlap rather than 25ms with 50% overlap. This means that there would be much fewer data points for our regression analysis, thus making it harder to train a model to capture this relationship.
- The fact that the MEG data was split into 2s trials (with noisy trials removed) would mean that additional data points of these long term features would need to be dropped because they do not correspond to 3 seconds of contiguous MEG trials - further reducing the number of data points to assess.

Chapter 7

Evaluation

In this chapter, we will evaluate the project with respect to its objectives.

The primary objective of this project was to assess the degree to which LSD affects the perceptual experience of music and/or specific musical features. We will first look at the degree to which this was achieved.

The secondary aim of this project was the implementation of the toolbox itself which was used to generate these results. This implementation will also be briefly evaluated.

7.1 Research

The research objective of this project can be split into two parts:

- The first aim is to successfully decode some representation of the music.
- Given that the first aim is met, develop a method to assess how LSD affects the decoding performance.

In this section, we will assess the research aspect of the project with respect to these aims.

7.1.1 Summary of Results

In Chapter 3, the AAD results were reproduced on a publicly available dataset.

In Chapter 4, the decoding method implemented in Chapter 3 were applied to decode music envelopes. Despite using ridge regression to account for the higher dimensional inputs and fewer data points, the music envelopes were not successfully decoded. This was shown by the negative R^2 scores. This was suspected by the client to be due to the complexity of music envelopes compared to speech.

In Chapter 5, the music representation was switched from envelopes to WaveNet encodings. These were successfully decoded using a time-frequency representation of the MEG. Although the LSD condition performance was found to be slightly higher, it was found not to be statistically significant at the 5% significance level (although close at $p = 0.06952$) when using a likelihood ratio test on a random intercept linear mixed effects model.

In Chapter 6, various attempts to decode interpretable short-term musical features were attempted. However, none of these attempts were able to decode the features, as shown by negative R^2 scores.

In the case of music envelopes and short-term musical features, because no successful decoder was found, we had no metric to assess perceptual experience (decoder performance/similarity), therefore the effects of LSD could not be assessed.

In the case where WaveNet encodings were used as a decoding output, a decoder was found which beat an establish baseline performance. Thus, the assessment of LSD could be carried out, which resulted in not finding any statistically significant difference. Any further implications drawn from these results would lie in the realm of neuroscience (i.e. the client). Some limitations, as we shall see, should be kept in mind when looking at this project's results.

7.1.2 Limitations

In this section, we will outline the limitations of the approaches attempted with respect to the research objectives of the project.

Using Only Linear Models

A possible limitation which may be affecting the decoding performance would be the choice of models in our regression analysis. All three regression methods considered - ridge regression, principal component regression, partial least squares - were based on an underlying linear model. It is possible that more expressive regression models, such as neural networks, would be more successful at decoding the music representation. In this project, rather than changing the regression methods, the input and output features were changed. The reasoning was that these linear methods had been successful in some decoding tasks; however, it could be argued that once the linear methods achieved a baseline improvement when decoding WaveNet encodings, rather than proceeding to different output feature, more complex regression models should be experimented with to further increase this decoding performance before doing assessing effects of LSD.

WaveNet Encodings

WaveNet encodings were the only musical representation that was successfully decoded. One disadvantage of WaveNet encodings is that they are uninterpretable. The numbers in a WaveNet encoding carry no human-interpretable meaning; consequently, not much can be said about what specifically in the music affected the perceptual experience. These encodings represent a general “essence" of the musical sound. This lack of interpretability also made it hard to tell what was affecting the model performance; as a result, many different approaches needed to be tried and tested empirically rather than narrowing down some methods using domain-specific knowledge in neuroscience. Another problem faced with WaveNet encodings is the fact they are a vector and not a single number (like envelopes). This makes the decoders more complex (16 times the number of parameters in the case of ridge regression) which makes them harder to train and prone to overfitting. To summarise, poor decoding performance can be attributed to one or more of the following factors:

- The use of linear models opposed to more sophisticated ones.
- Ineffective decoding design (e.g. MEG window size) decisions due to the lack of understanding of what the encodings represent.
- Difficulty training complex models (defined by the number of parameters) associated with the high dimensional outputs.
- A genuine lack of similarity between the brain signals and the music.

Not Considering the Amount of Data per Subject

One implicit assumption made was that number of MEG trials per client did not affect the decoding performance. Recall that out of the total 210 2s trials, trials that were deemed excessively noisy (by the client) were removed. Table A.1 in the Appendix shows the remaining trials for each subject. We can see that for most subjects, the number of trials was around 200 i.e. only around 10 bad trials were removed. However, for two of the subjects, a large number of trials were removed in the LSD condition. This could negatively affect the decoding performance of the LSD condition.

It is worth noting that even if not many trials are removed, each trial may correspond to many data points because they are split up into smaller timescales. Therefore, the difference between five and ten bad trials may correspond to a much larger difference in the number of data points.

Using Decoding to Measure Similarity

Another possible limitation is using decoding as a method to compare the similarity between the LSD and placebo condition. We make an implicit assumption that if one condition outperforms the other, this is because the brain signals of that condition are more similar to the music stimulus. However, we are not considering other possible differences between the signals in both of these conditions not pertaining to the music stimuli. For example, consider the case where the LSD brain signals did contain more information about the music stimulus, but also contained more information not related to the stimulus - noise as far as we are concerned. This noise may make it hard to decode the stimulus which may lead us to wrongly believe that the brain signals in the LSD condition do not resemble the stimulus as much as they do. One could argue that the signals should have been analysed beforehand for their properties irrespective of the stimuli (e.g. signal complexity).

Inherent Limitations

There are some limitations of this project out of our control which should also be considered. After a discussion with the client, the following points were brought up:

- Despite cleaning attempts of the MEG data, there still will be some noise present which could be decreasing decoding performance.

- MEG has poor spatial resolution compared to invasive procedures, such as ECoG, where electrodes are implanted into the skull (mentioned in Section 2.9). This decreases the accuracy of how MEG represents the brain activity.

- A limitation of music as the stimulus is that there are many simultaneous instruments playing at once. Although we can extract musical features which represent the musical sound as a whole, it may be more effective if we were able to assess the music on an instrument-by-instrument basis.

- The nature of the music was ambient. This could potentially be why decoding the music representation was hard. Music with more distinct "events" or maybe of a more rhythmic nature may be easier to decode. This may be the reason why the speech stimuli was decoded successfully - the presence of distinct bursts of speech (events) followed by periods of temporary silence.

- There were only 13 subjects to analyse. With more subjects, smaller differences in decoding performance may be shown to be statistically significant.

7.1.3 Achievements

The nature of this project is primarily data exploration. In data exploration, sometimes approaches fail as we have seen numerous times this project. In this section, the project's achievements will be assessed.

Showing Limitations of Existing Methods

In Chapter 4, a possible limitation of the existing decoding audio stimulus methods was shown. While audio envelopes were useful as a way to decode speech envelopes in the AAD paradigm, they did not generalise well to music envelopes in this project. By first implementing these methods to reproduce the AAD results, there is some validity that these methods were implemented correctly. This further suggests that the limitations could be due to the transition from speech to music and not a failure in implementation.

It is important to note, however, that there were some other differences other than the change from speech to music which could have been factors in the inability to generalise the AAD methods. These factors include using MEG instead of EEG, the number of brain signal channels, and having fewer data per subject.

Ruling Out Methods that do not Work

All the failed decoding attempts, which resulted in models that would not fit the data at all with negative R^2 scores, still serve a purpose. These negative results are useful because they serve as a way to show the unsuccessful attempts. Future research should consider these results when deciding the next stages.

It should be noted that the choices of the decoding models proposed throughout this project were not arbitrary. In this report, justifications were given for the choices at each stage. As such, these are methods which are likely to be considered in future research which can now be ruled out (or reproduced expecting similar negative results).

Successfully Decoding WaveNet Encodings

Although the decoding performance was not high, WaveNet encodings were still decoded to some degree. Recall from Section 2.7.3 that the specific WaveNet autoencoder that generated these encodings was originally intended to aid the musical creation process. In this project, we have proposed a novel way to re-purpose these encodings, namely as a representation for music stimulus when studying its similarity with brain activity.

Because decoding WaveNet encodings was achieved to some degree, it would be reasonable for future research to try and improve the decoding performance. This could be done using either more powerful models, such as neural networks, or adjusting the input and output preprocessing.

Developing an End-to-End Procedure

In this project, an end-to-end method for assessing the way LSD affects the perceptual experience of music using neural decoding has been developed. To summarise the key aspects of this method developed throughout this project:

- Nested cross-validation to assess decoding performance and search over a given decoder's hyperparameters.
- Using a surrogate dataset to establish a baseline performance to beat for a given decoder model.
- Using a (linear) mixed effects model to model the relationship between the condition and the decoding performance, while accounting for the song listened to and subject.
- Using a likelihood ratio test to evaluate whether the mixed effects model fits the data better than the same mixed effect model without the condition variable, thus showing whether the condition was important to fit the data.

This procedure can be re-applied for future research.

7.2 Implementation

In this section, we will briefly evaluate the implementation of the toolbox that was developed for this project.

Instead of a single generic script for all experiments, separate scripts were created to run the procedure outlined for envelope decoding (Chapter 4), WaveNet decoding (Chapter 5), and short-term musical feature decoding (Chapter 6). Furthermore, the decoding and analysis of these results were split into separate scripts.

The main advantage of splitting the decoding procedure itself from the analysis of its results was the ability to modify the analysis without needing to re-run the entire decoding training procedure, which would take a lot of time. The limitation of this approach is the coupling created between these separate scripts. For example, when modifying the way the results are saved in the decoding script, this may break the analysis script expecting the results to be in a specific format. The advantage of this split, however, was considered more important than this limitation.

Furthermore, by splitting the decoding and analysis, an R script could be used for the analysis of the effect of LSD (Chapter 4). Recall, that the advantage of this R script was it could easily perform the statistical analysis desired in 4 lines of code.

The main disadvantage of having separate scripts for the different musical features is that there is some code duplication between them - something that is typically avoided when developing software. However, there are some advantages of separating the scripts for each experiment. Firstly, the scripts themselves were more simple and modular, requiring only a few arguments. On the other hand, a single script which could run all of these experiments would require many arguments and would be very complex (a lot of "if ... else" blocks for different experiments). The subtle differences between each experiment also meant that analysis scripts could be tailored for each experiment. For example, if we know one experiment uses partial least squares regression, the analysis could be written such that it could analyse the optimal number of components.

By splitting the code into executable scripts and library functions, common behaviour could be extracted between different experiments as was needed. This lead to a compromise of maintaining modularity between experiments, and reducing code duplication.

Chapter 8

Conclusion and Future Work

In this project, we have looked at neural decoding as a means to compare the similarity between brain signals (MEG recordings) and different representations of a music stimulus. This similarity acts as a measure of perception and was compared across the LSD and placebo condition of a subject when the music representation was successfully decoded.

While audio envelopes may be an appropriate representation for decoding speech signals (assuming a linear decoding model), we have seen that this does not necessarily generalise to music envelopes. This is seen when the audio decoding model developed in Chapter 3 failed to decode music in Chapter 4. One possible reason for this could be the increase in complexity that the shape of a music envelope has compared to a speech envelope with distinct "bursts" of sound surrounded by silence.

Suspecting that an audio envelope representation a large factor for poor decoding performance, in Chapter 5, a new representation of music was used. This representation was WaveNet encodings of music created by the Magenta project[1] which capture musical features in a 16-dimensional vector. These encodings contained the "essence" of the musical sound as opposed to the loudness at each point as was the case for music envelopes. To account for the high-dimensional inputs, principal component regression has been shown to successfully decode these representations (something that ridge regression failed to do). Although the average LSD performance was higher, which may suggest that the subject is perceiving the music more under the effects of LSD, this was shown not to be statistically significant using a linear mixed effects model for analysis. These results, however, should be considered under the limitations outlined in the evaluation (Chapter 7, Section 7.1.2), most notably the low R^2 scores.

In Chapter 6, we explored the decoding of short-term musical features which are known to affect the perceptual experience of music. One of these features corresponded to the loudness of a sound and the remaining features corresponded to a sound's timbral properties. Despite the various decoding models attempted, these features could not be successfully decoded.

There are many avenues for future work on the analysis of dataset provided by the Centre for Psychedelic Research, many of which would address the limitations of the research done in this project (Chapter 7, Section 7.1.2). These include:

- Using more sophisticated models, such as neural networks, which may result in better decoding performance. A good place to apply this would be to the WaveNet encoding experiment, which already has some decoding success.
- Looking at methods other than neural decoding as a measure for similarity. For example, comparing the complexity of the brain signals to the complexity of the music[47].
- Investigating the similarity between brain signals and long-term features which affect the perceptual experience of music[3]. These features would correspond to the tonal and rhythmic properties of music[38].

Appendix A

Subject Metadata

Subject ID	LSD Trials	Placebo Trials
041213_1	206	210
040914_1	100	207
290514_2	194	206
010514_1	198	198
260614_1	194	203
040914_2	199	204
140514_2	209	201
010813_4	208	210
070814_2	200	203
240107_6	206	206
310714_1	199	207
200814_1	135	202
230911_1	208	209

Table A.1: Number of 2s MEG trials per subject after removing noisy trials.

Appendix B

Results

Subject ID	R^2 LSD	R^2 Placebo
041213_1	-0.4315	-0.5308
290514_2	-0.4050	-0.6050
040914_2	-0.5374	-0.3701
240107_6	-0.3384	-0.5499
310714_1	-0.5760	-0.3893
040914_1	-0.4750	-0.5605
140514_2	-0.5361	-0.3892
260614_1	-0.5270	-0.3902
010514_1	-0.3667	-0.5152
200814_1	-0.7050	-0.3633
230911_1	-0.5493	-0.4045
010813_4	-0.3471	-0.5016
070814_2	-0.4586	-0.3704

Table B.1: Music envelope decoding results.

Subject ID	Song	Condition	R^2	Surrogate R^2	MSE	Surrogate MSE
010514_1	B	LSD	-141.4220	-141.4819	28.5194	28.5264
010514_1	A	Placebo	-103.9803	-104.0717	28.2627	28.2815
010813_4	B	LSD	-142.7011	-142.6837	28.5382	28.5296
010813_4	A	Placebo	-105.2147	-105.2398	28.2636	28.2638
040914_1	B	LSD	-191.5627	-191.4686	28.6191	28.6036
040914_1	A	Placebo	-115.2326	-115.1526	28.3385	28.3185
040914_2	A	LSD	-105.7895	-105.8342	28.3170	28.3274
040914_2	B	Placebo	-140.6960	-140.8117	28.5164	28.5365
041213_1	B	LSD	-140.7629	-140.8247	28.5310	28.5493
041213_1	A	Placebo	-105.8245	-105.7746	28.3219	28.3077
070814_2	A	LSD	-104.1677	-104.1789	28.1525	28.1502
070814_2	B	Placebo	-143.5653	-143.7069	28.5543	28.5791
140514_2	A	LSD	-106.3745	-106.4010	28.2314	28.2360
140514_2	B	Placebo	-138.6973	-138.7793	28.4923	28.5066
200814_1	A	LSD	-109.0116	-108.7443	27.9008	27.8376
200814_1	B	Placebo	-137.9469	-137.9189	28.4735	28.4646
230911_1	A	LSD	-105.1572	-105.1745	28.2565	28.2533
230911_1	B	Placebo	-141.6131	-141.5276	28.5255	28.5067
240107_6	B	LSD	-139.9398	-139.9579	28.4660	28.4639
240107_6	A	Placebo	-104.9135	-104.9704	28.2667	28.2730
260614_1	A	LSD	-103.7118	-103.6740	28.3223	28.3098
260614_1	B	Placebo	-140.0498	-140.1384	28.4458	28.4560
290514_2	B	LSD	-137.4221	-137.4656	28.3796	28.3844
290514_2	A	Placebo	-105.6980	-105.7942	28.1830	28.2008
310714_1	A	LSD	-105.6186	-105.6917	28.2305	28.2424
310714_1	B	Placebo	-143.0533	-143.0248	28.5685	28.5662

Table B.2: WaveNet decoding ridge regression results

Subject ID	Song	Condition	R^2	Surrogate R^2	MSE	Surrogate MSE
010514_1	B	LSD	0.04097	-0.02629	0.11605	0.12444
010514_1	A	Placebo	0.04167	-0.03571	0.16331	0.17669
010813_4	B	LSD	0.04211	-0.03137	0.11711	0.12524
010813_4	A	Placebo	0.03499	-0.01727	0.16256	0.17003
040914_1	B	LSD	0.06969	-0.04915	0.08007	0.09177
040914_1	A	Placebo	0.01796	-0.0188	0.14956	0.15506
040914_2	A	LSD	0.0408	-0.03124	0.16138	0.17334
040914_2	B	Placebo	0.02668	-0.04044	0.11828	0.1267
041213_1	B	LSD	0.05542	-0.02753	0.11448	0.12579
041213_1	A	Placebo	0.0289	-0.01974	0.16206	0.17046
070814_2	A	LSD	0.02009	-0.03257	0.1646	0.17364
070814_2	B	Placebo	0.04151	-0.03471	0.11587	0.12612
140514_2	A	LSD	0.0196	-0.03456	0.16215	0.17167
140514_2	B	Placebo	0.03627	-0.02346	0.11798	0.1265
200814_1	A	LSD	0.02361	-0.03337	0.15641	0.16574
200814_1	B	Placebo	0.0235	-0.02511	0.1227	0.12742
230911_1	A	LSD	0.06649	-0.03026	0.15471	0.17243
230911_1	B	Placebo	0.01762	-0.02332	0.12064	0.12467
240107_6	B	LSD	0.03992	-0.04035	0.11798	0.12719
240107_6	A	Placebo	0.05437	-0.04602	0.15865	0.17594
260614_1	A	LSD	0.03598	-0.03094	0.16359	0.17682
260614_1	B	Placebo	0.04951	-0.05662	0.115	0.12807
290514_2	B	LSD	0.0588	-0.02962	0.11641	0.128
290514_2	A	Placebo	0.03403	-0.02111	0.15955	0.16988
310714_1	A	LSD	0.04409	-0.03843	0.15782	0.1728
310714_1	B	Placebo	0.01997	-0.01411	0.11757	0.12229

Table B.3: WaveNet decoding principal component regression results.

Subject ID	Song	Condition	R^2	Surrogate R^2	MSE	Surrogate MSE
010514_1	B	LSD	0.0273	-0.08063	0.1175	0.13479
010514_1	A	Placebo	0.02409	-0.05651	0.1655	0.18325
010813_4	B	LSD	0.02267	-0.05502	0.11828	0.13093
010813_4	A	Placebo	0.03071	-0.05528	0.16287	0.17956
040914_1	B	LSD	0.04454	-0.11041	0.08177	0.10126
040914_1	A	Placebo	0.01175	-0.03554	0.14991	0.15962
040914_2	A	LSD	0.02988	-0.05188	0.16188	0.17971
040914_2	B	Placebo	0.01382	-0.06465	0.11949	0.13293
041213_1	B	LSD	0.04173	-0.06499	0.11619	0.13321
041213_1	A	Placebo	0.01901	-0.05297	0.16345	0.17892
070814_2	A	LSD	0.00732	-0.0516	0.16618	0.17961
070814_2	B	Placebo	0.02894	-0.06411	0.1169	0.13275
140514_2	A	LSD	0.01914	-0.05237	0.16201	0.17774
140514_2	B	Placebo	0.02429	-0.0513	0.11946	0.13221
200814_1	A	LSD	0.01567	-0.06849	0.15727	0.17465
200814_1	B	Placebo	0.01766	-0.06312	0.12347	0.13527
230911_1	A	LSD	0.0554	-0.08284	0.15583	0.18646
230911_1	B	Placebo	0.0053	-0.05682	0.12147	0.13186
240107_6	B	LSD	0.01555	-0.06654	0.12003	0.13341
240107_6	A	Placebo	0.04185	-0.06832	0.15971	0.18389
260614_1	A	LSD	0.03239	-0.06888	0.16372	0.18726
260614_1	B	Placebo	0.03663	-0.06355	0.11637	0.13213
290514_2	B	LSD	0.04597	-0.06813	0.11751	0.13579
290514_2	A	Placebo	0.02755	-0.06011	0.16061	0.18004
310714_1	A	LSD	0.03072	-0.06883	0.15978	0.18185
310714_1	B	Placebo	0.01253	-0.05031	0.11824	0.1289

Table B.4: WaveNet decoding partial least squares regression results.

Feature	Condition	mean_r2	median_r2
Spectral Centroid	LSD	-13.0487	-5.4668
Spectral Centroid	Placebo	-13.7365	-8.0541
Spectral Entropy	LSD	-13.0487	-5.4668
Spectral Entropy	Placebo	-13.7365	-8.0541
Flatness	LSD	-13.0488	-5.4667
Flatness	Placebo	-13.7365	-8.0541
Spectral Flux	LSD	-13.0488	-5.4668
Spectral Flux	Placebo	-13.7365	-8.0541
High Energy-Low Energy Ratio	LSD	-13.0488	-5.4667
High Energy-Low Energy Ratio	Placebo	-13.7365	-8.0541
RMS Energy	LSD	-13.0487	-5.4668
RMS Energy	Placebo	-13.7365	-8.0541
Spectral Roll-off	LSD	-13.0487	-5.4668
Spectral Roll-off	Placebo	-13.7365	-8.0541
Roughness	LSD	-13.0488	-5.4668
Roughness	Placebo	-13.7365	-8.0541
Spectral Spread	LSD	-13.0488	-5.4668
Spectral Spread	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 1	LSD	-13.0487	-5.4668
Spectral Flux Sub-band 1	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 2	LSD	-13.0488	-5.4668
Spectral Flux Sub-band 2	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 3	LSD	-13.0487	-5.4668
Spectral Flux Sub-band 3	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 4	LSD	-13.0487	-5.4668
Spectral Flux Sub-band 4	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 5	LSD	-13.0487	-5.4668
Spectral Flux Sub-band 5	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 6	LSD	-13.0488	-5.4668
Spectral Flux Sub-band 6	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 7	LSD	-13.0488	-5.4668
Spectral Flux Sub-band 7	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 8	LSD	-13.0488	-5.4668
Spectral Flux Sub-band 8	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 9	LSD	-13.0488	-5.4668
Spectral Flux Sub-band 9	Placebo	-13.7365	-8.0541
Spectral Flux Sub-band 10	LSD	-13.0487	-5.4668
Spectral Flux Sub-band 10	Placebo	-13.7365	-8.0541
Zero Crossing Rate	LSD	-13.0488	-5.4668
Zero Crossing Rate	Placebo	-13.7365	-8.0541

Table B.5: Short-term acoustic feature results. Decoded using ridge regression and a window of the MEG's instantaneous power.

Feature	Condition	Mean R^2	Median R^2
Spectral Centroid	LSD	-12.9272	-5.4348
Spectral Centroid	Placebo	-13.6638	-8.0028
Spectral Entropy	LSD	-12.9272	-5.4348
Spectral Entropy	Placebo	-13.6638	-8.0028
Flatness	LSD	-12.9272	-5.4348
Flatness	Placebo	-13.6638	-8.0028
Spectral Flux	LSD	-12.9272	-5.4348
Spectral Flux	Placebo	-13.6638	-8.0028
High Energy-Low Energy Ratio	LSD	-12.9272	-5.4348
High Energy-Low Energy Ratio	Placebo	-13.6638	-8.0028
RMS Energy	LSD	-12.9272	-5.4348
RMS Energy	Placebo	-13.6638	-8.0028
Spectral Roll-off	LSD	-12.9272	-5.4348
Spectral Roll-off	Placebo	-13.6638	-8.0028
Roughness	LSD	-12.9272	-5.4348
Roughness	Placebo	-13.6638	-8.0028
Spectral Spread	LSD	-12.9272	-5.4348
Spectral Spread	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 1	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 1	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 2	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 2	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 3	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 3	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 4	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 4	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 5	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 5	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 6	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 6	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 7	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 7	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 8	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 8	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 9	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 9	Placebo	-13.6638	-8.0028
Spectral Flux Sub-band 10	LSD	-12.9272	-5.4348
Spectral Flux Sub-band 10	Placebo	-13.6638	-8.0028
Zero Crossing Rate	LSD	-12.9272	-5.4348
Zero Crossing Rate	Placebo	-13.6638	-8.0028

Table B.6: Short-term acoustic feature results. Decoded using ridge regression and a window of the MEG's mean instantaneous power.

Feature	Condition	Mean R^2	Median R^2
Spectral Centroid	LSD	-12.6387	-5.4626
Spectral Centroid	Placebo	-12.0829	-8.2981
Spectral Entropy	LSD	-12.6463	-5.4601
Spectral Entropy	Placebo	-12.0670	-8.2832
Flatness	LSD	-12.6118	-5.4635
Flatness	Placebo	-12.0846	-8.2138
Spectral Flux	LSD	-12.6365	-5.4641
Spectral Flux	Placebo	-12.0862	-8.2189
High Energy-Low Energy Ratio	LSD	-12.6230	-5.4624
High Energy-Low Energy Ratio	Placebo	-12.0754	-8.2222
RMS Energy	LSD	-12.6193	-5.4616
RMS Energy	Placebo	-12.0901	-8.2303
Spectral Roll-off	LSD	-12.6358	-5.5143
Spectral Roll-off	Placebo	-12.0886	-8.2253
Roughness	LSD	-12.6384	-5.5048
Roughness	Placebo	-12.0869	-8.2901
Spectral Spread	LSD	-12.6181	-5.4624
Spectral Spread	Placebo	-12.1056	-8.2691
Spectral Flux Sub-band 1	LSD	-12.6244	-5.5053
Spectral Flux Sub-band 1	Placebo	-12.0736	-8.2837
Spectral Flux Sub-band 2	LSD	-12.6182	-5.4629
Spectral Flux Sub-band 2	Placebo	-12.0871	-8.2887
Spectral Flux Sub-band 3	LSD	-12.6387	-5.4607
Spectral Flux Sub-band 3	Placebo	-12.0429	-8.29943
Spectral Flux Sub-band 4	LSD	-12.6307	-5.4633
Spectral Flux Sub-band 4	Placebo	-12.0520	-8.2168
Spectral Flux Sub-band 5	LSD	-12.6276	-5.5014
Spectral Flux Sub-band 5	Placebo	-12.0475	-8.2231
Spectral Flux Sub-band 6	LSD	-12.6235	-5.4627
Spectral Flux Sub-band 6	Placebo	-12.0714	-8.2231
Spectral Flux Sub-band 7	LSD	-12.6105	-5.4640
Spectral Flux Sub-band 7	Placebo	-12.0898	-8.2900
Spectral Flux Sub-band 8	LSD	-12.6218	-5.4639
Spectral Flux Sub-band 8	Placebo	-12.0846	-8.2348
Spectral Flux Sub-band 9	LSD	-12.6582	-5.5116
Spectral Flux Sub-band 9	Placebo	-12.0917	-8.2743
Spectral Flux Sub-band 10	LSD	-12.6477	-5.4620
Spectral Flux Sub-band 10	Placebo	-12.0904	-8.2047
Zero Crossing Rate	LSD	-12.6294	-5.4621
Zero Crossing Rate	Placebo	-12.0687	-8.2846

Table B.7: Short-term acoustic feature results. Decoded using principal component regression and MEG scalograms.

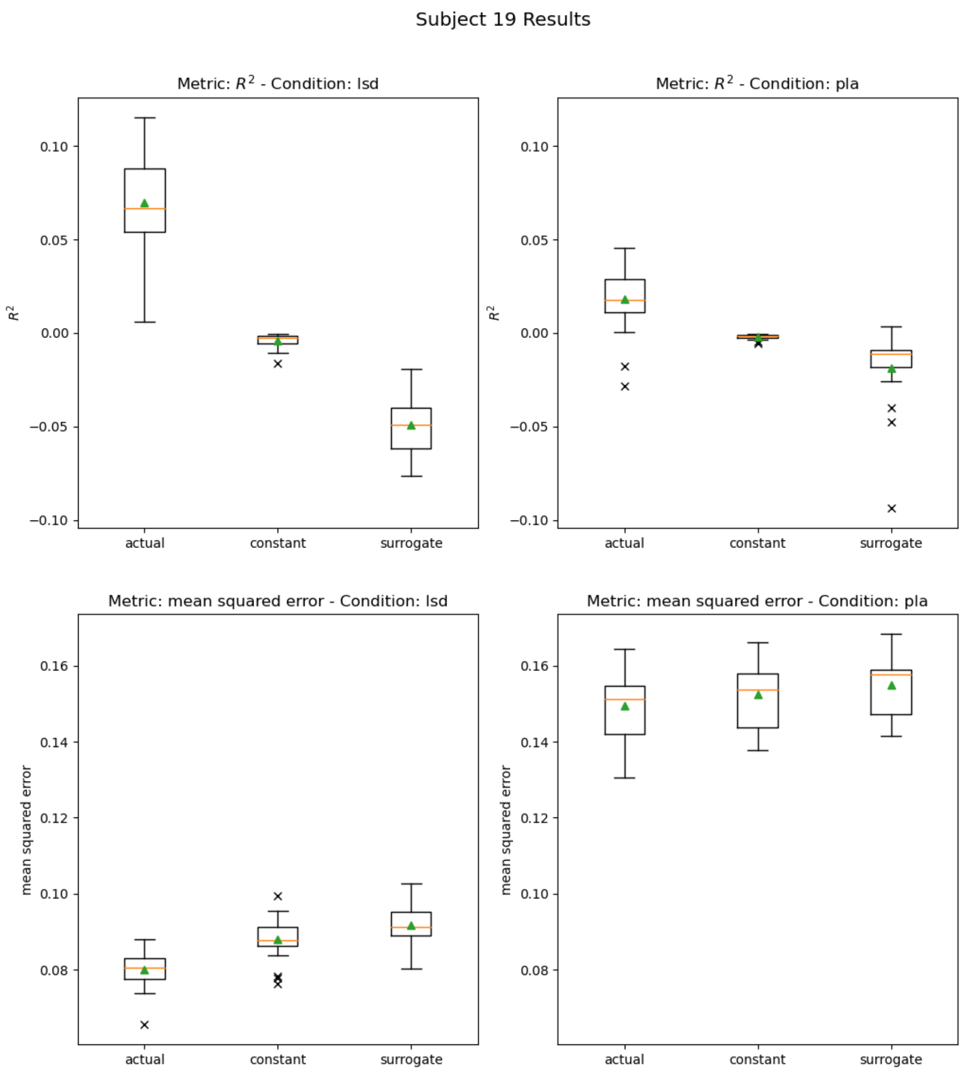


Figure B.1: WaveNet decoding performance for one subject chosen at random with principal component regression. The top graphs show the R^2 performance. The bottom graphs show the mean squared error performance. The x-axis shows the different performance measures: actual performance, constant baseline and surrogate baseline. The box shows the interquartile range of the data. The orange line cutting the box shows and green triangle show the median and mean of the data respectively. The whiskers show the furthest data points 1.5 times the interquartile range above and below and third and first quartile respectively. The crosses determine all other points outside the whiskers.

Appendix C

Hyperparameter Search Ranges

In this section, the hyperparameters for each regression method in each chapter will stated.

C.1 Music Envelope Decoding

Hyperparameters used in Chapter 4.

- Ridge regression - The values of the regularisation strength α were $\{0, 0.1, 0.5, 1, 10, 50, 100, 500, 1000\}$.

C.2 WaveNet Decoding

Hyperparameters used in Chapter 5.

- Ridge regression - The values of the regularisation strength α were $\{0.01, 0.1, 1, 10, 1000, 10000, 100000, 1000000\}$.
- Partial least squares regression - The values of the number of components used were the integers in the range $[2, 14]$.
- Principal component regression - The values of the number of components used were integers in the range $[10, 1000]$ spaced by 20.

C.3 Short-term Musical Feature Decoding

Hyperparameters used in Chapter 6.

- Ridge regression - The values of the regularisation strength α were $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$.
- Principal component regression - The values of the number of components used were integers in the range $[5, 1000]$ spaced by 50.

Bibliography

[1] *Magenta*. Magenta. Library Catalog: magenta.tensorflow.org. URL: https://magenta.tensorflow.org/ (visited on 05/09/2020).

[2] Robin L Carhart-Harris and Guy M Goodwin. "The Therapeutic Potential of Psychedelic Drugs: Past, Present, and Future". In: *Neuropsychopharmacology* 42.11 (Oct. 2017), pp. 2105–2113. ISSN: 0893-133X, 1740-634X. DOI: 10.1038/npp.2017.84. URL: http://www.nature.com/articles/npp201784 (visited on 01/14/2020).

[3] Mendel Kaelen et al. "The hidden therapist: evidence for a central role of music in psychedelic therapy". In: *Psychopharmacology* 235.2 (Feb. 2018), pp. 505–519. ISSN: 0033-3158, 1432-2072. DOI: 10.1007/s00213-017-4820-5. URL: http://link.springer.com/10.1007/s00213-017-4820-5 (visited on 01/14/2020).

[4] Teri S Krebs and Pål-Ørjan Johansen. "Over 30 million psychedelic users in the United States". In: *F1000Research* 2 (Mar. 28, 2013). ISSN: 2046-1402. DOI: 10.12688/f1000research.2-98.v1. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3917651/ (visited on 06/07/2020).

[5] Robin L Carhart-Harris et al. "Psychedelics and the essential importance of context". In: *Journal of Psychopharmacology* 32.7 (July 2018), pp. 725–731. ISSN: 0269-8811, 1461-7285. DOI: 10.1177/0269881118754710. URL: http://journals.sagepub.com/doi/10.1177/0269881118754710 (visited on 01/14/2020).

[6] Enzo Tagliazucchi et al. "Increased Global Functional Connectivity Correlates with LSD-Induced Ego Dissolution". In: *Current Biology* 26.8 (Apr. 25, 2016), pp. 1043–1050. ISSN: 0960-9822. DOI: 10.1016/j.cub.2016.02.010. URL: http://www.sciencedirect.com/science/article/pii/S0960982216300628 (visited on 01/14/2020).

[7] Michael M. Schartner et al. "Increased spontaneous MEG signal diversity for psychoactive doses of ketamine, LSD and psilocybin". In: *Scientific Reports* 7.1 (Apr. 19, 2017), pp. 1–12. ISSN: 2045-2322. DOI: 10.1038/srep46421. URL: https://www.nature.com/articles/srep46421 (visited on 01/14/2020).

[8] Robin L. Carhart-Harris et al. "The entropic brain: a theory of conscious states informed by neuroimaging research with psychedelic drugs". In: *Frontiers in Human Neuroscience* 8 (2014). ISSN: 1662-5161. DOI: 10.3389/fnhum.2014.00020. URL: http://journal.frontiersin.org/article/10.3389/fnhum.2014.00020/abstract (visited on 01/15/2020).

[9] Leor Roseman, David J. Nutt, and Robin L. Carhart-Harris. "Quality of Acute Psychedelic Experience Predicts Therapeutic Efficacy of Psilocybin for Treatment-Resistant Depression". In: *Frontiers in Pharmacology* 8 (Jan. 17, 2018), p. 974. ISSN: 1663-9812. DOI: 10.3389/fphar.2017.00974. URL: http://journal.frontiersin.org/article/10.3389/fphar.2017.00974/full (visited on 01/15/2020).

[10] James A. O'Sullivan et al. "Attentional Selection in a Cocktail Party Environment Can Be Decoded from Single-Trial EEG". In: *Cerebral Cortex* 25.7 (July 2015), pp. 1697–1706. ISSN: 1460-2199, 1047-3211. DOI: 10.1093/cercor/bht355. URL: https://academic.oup.com/cercor/article-lookup/doi/10.1093/cercor/bht355 (visited on 01/06/2020).

[11] Brian Everitt and Anders Skrondal. *The Cambridge dictionary of statistics*. OCLC: 1104393679. 2010. ISBN: 978-0-511-78974-8 978-0-511-78713-3 978-0-511-78827-7. URL: http://proxy.cegepat.qc.ca/login?url=http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&AN=329356 (visited on 05/08/2020).

[12] Pascal Wallisch. “Neural Decoding I”. In: *MATLAB for Neuroscientists*. Elsevier, 2014, pp. 329–336. ISBN: 978-0-12-383836-0. DOI: 10.1016/B978-0-12-383836-0.00021-7. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780123838360000217 (visited on 05/09/2020).

[13] Yunze He et al. “Coil-Based Rectangular PEC Sensors for Defect Classification”. In: *Transient Electromagnetic-Thermal Nondestructive Testing*. Elsevier, 2017, pp. 55–90. ISBN: 978-0-12-812787-2. DOI: 10.1016/B978-0-12-812787-2.00004-6. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780128127872000046 (visited on 05/10/2020).

[14] Frederick H. Long. “Multivariate Analysis for Metabolomics and Proteomics Data”. In: *Proteomic and Metabolomic Approaches to Biomarker Discovery*. Elsevier, 2013, pp. 299–311. ISBN: 978-0-12-394446-7. DOI: 10.1016/B978-0-12-394446-7.00019-4. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780123944467000194 (visited on 05/10/2020).

[15] Jeff Schneider. *Cross Validation*. URL: https://www.cs.cmu.edu/~schneide/tut5/node42.html (visited on 05/08/2020).

[16] Roland Priemer. “Signals and Signal Processing”. In: *Introductory Signal Processing*. Google-Books-ID: QBT7nP7zTLgC. World Scientific, 1991, p. 1. ISBN: 978-9971-5-0919-4.

[17] John A. Gubner. “Random vectors and random matrices”. In: *Probability and random processes for electrical and computer engineers*. OCLC: ocm70147423. New York ; Cambridge: Cambridge University Press, 2006, p. 334. ISBN: 978-0-521-86470-1.

[18] E. R. Kanasewich. “Band pass filters”. In: *Time Sequence Analysis in Geophysics*. University of Alberta, 1981, p. 260. ISBN: 978-0-88864-074-1.

[19] C. Richard Johnson Jr, William A. Sethares, and Andrew G. Klein. “Envelope of a Bandpass Signal”. In: *Software Receiver Design: Build your Own Digital Communication System in Five Easy Steps*. Google-Books-ID: LNea1qui1KcC. Cambridge University Press, Aug. 18, 2011, p. 417. ISBN: 978-1-139-50145-3.

[20] Laurent Navarro, Guy Courbebaisse, and Michel Jourlin. “Logarithmic Wavelets”. In: *Advances in Imaging and Electron Physics*. Vol. 183. Elsevier, 2014, pp. 41–98. ISBN: 978-0-12-800265-0. DOI: 10.1016/B978-0-12-800265-0.00002-3. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780128002650000023 (visited on 06/13/2020).

[21] “Chapter 4 - Advanced Time-Frequency Signal and System Analysis”. In: *Time-Frequency Signal Analysis and Processing (Second Edition)*. Ed. by Boualem Boashash. Oxford: Academic Press, Jan. 1, 2016, pp. 141–236. ISBN: 978-0-12-398499-9. DOI: 10.1016/B978-0-12-398499-9.00004-2. URL: http://www.sciencedirect.com/science/article/pii/B9780123984999000042 (visited on 06/13/2020).

[22] Andrea Biasiucci, Benedetta Franceschiello, and Micah M. Murray. “Electroencephalography”. In: *Current Biology* 29.3 (Feb. 2019), R80–R85. ISSN: 09609822. DOI: 10.1016/j.cub.2018.11.052. URL: https://linkinghub.elsevier.com/retrieve/pii/S0960982218315513 (visited on 01/15/2020).

[23] Alexandre Gramfort et al. “MNE software for processing MEG and EEG data”. In: *NeuroImage* 86 (Feb. 2014), pp. 446–460. ISSN: 10538119. DOI: 10.1016/j.neuroimage.2013.10.027. URL: https://linkinghub.elsevier.com/retrieve/pii/S1053811913010501 (visited on 01/19/2020).

[24] Alexandre Gramfort. “MEG and EEG data analysis with MNE-Python”. In: *Frontiers in Neuroscience* 7 (2013). ISSN: 1662453X. DOI: 10.3389/fnins.2013.00267. URL: http://journal.frontiersin.org/article/10.3389/fnins.2013.00267/abstract (visited on 01/19/2020).

[25] Borís Burle et al. “Spatial and temporal resolutions of EEG: Is it really black and white? A scalp current density view”. In: *International Journal of Psychophysiology* 97.3 (Sept. 2015), pp. 210–220. ISSN: 01678760. DOI: 10.1016/j.ijpsycho.2015.05.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167876015001865 (visited on 01/17/2020).

[26] Ramesh Srinivasan. “Methods to Improve the Spatial Resolution of EEG”. In: *International Journal of Bioelectromagnetism* 1.1 (1999), p. 10.

[27] David Cohen and B.Neil Cuffin. "Demonstration of useful differences between magnetoencephalogram and electroencephalogram". In: *Electroencephalography and Clinical Neurophysiology* 56.1 (July 1983), pp. 38–51. ISSN: 00134694. DOI: 10.1016/0013-4694(83)90005-6. URL: https://linkinghub.elsevier.com/retrieve/pii/0013469483900056 (visited on 05/09/2020).

[28] P.S. Lewis, J.C. Mosher, and R.M. Leahy. "Neuromagnetic source reconstruction". In: *1995 International Conference on Acoustics, Speech, and Signal Processing*. 1995 International Conference on Acoustics, Speech, and Signal Processing. Vol. 5. Detroit, MI, USA: IEEE, 1995, pp. 2911–2914. ISBN: 978-0-7803-2431-2. DOI: 10.1109/ICASSP.1995.479454. URL: http://ieeexplore.ieee.org/document/479454/ (visited on 05/09/2020).

[29] Lee M. Miller. "Neural Mechanisms of Attention to Speech". In: *Neurobiology of Language*. Elsevier, 2016, pp. 503–514. ISBN: 978-0-12-407794-2. DOI: 10.1016/B978-0-12-407794-2.00041-9. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780124077942000419 (visited on 01/17/2020).

[30] Wouter Biesmans et al. "Auditory-Inspired Speech Envelope Extraction Methods for Improved EEG-Based Auditory Attention Detection in a Cocktail Party Scenario". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25.5 (May 2017), pp. 402–412. ISSN: 1534-4320, 1558-0210. DOI: 10.1109/TNSRE.2016.2571900. URL: http://ieeexplore.ieee.org/document/7478117/ (visited on 01/04/2020).

[31] Gregory Ciccarelli et al. "Comparison of Two-Talker Attention Decoding from EEG with Nonlinear Neural Networks and Linear Methods". In: *Scientific Reports* 9.1 (Aug. 8, 2019), pp. 1–10. ISSN: 2045-2322. DOI: 10.1038/s41598-019-47795-0. URL: https://www.nature.com/articles/s41598-019-47795-0 (visited on 01/10/2020).

[32] Bojana Mirkovic et al. "Decoding the attended speech stream with multi-channel EEG: implications for online, daily-life applications". In: *Journal of Neural Engineering* 12.4 (Aug. 2015), p. 046007. ISSN: 1741-2552. DOI: 10.1088/1741-2560/12/4/046007.

[33] Simon Van Eyndhoven, Tom Francart, and Alexander Bertrand. "EEG-Informed Attended Speaker Extraction From Recorded Speech Mixtures With Application in Neuro-Steered Hearing Prostheses". In: *IEEE Transactions on Biomedical Engineering* 64.5 (May 2017), pp. 1045–1056. ISSN: 0018-9294, 1558-2531. DOI: 10.1109/TBME.2016.2587382. URL: http://ieeexplore.ieee.org/document/7505982/ (visited on 01/09/2020).

[34] Yann LeCun, Corinna Cortes, and Chris Burges. *MNIST handwritten digit database*. URL: http://yann.lecun.com/exdb/mnist/ (visited on 05/06/2020).

[35] Aaron van den Oord et al. "WaveNet: A Generative Model for Raw Audio". In: *arXiv:1609.03499 [cs]* (Sept. 19, 2016). arXiv: 1609.03499. URL: http://arxiv.org/abs/1609.03499 (visited on 05/07/2020).

[36] Jesse Engel et al. "Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders". In: *arXiv:1704.01279 [cs]* (Apr. 5, 2017). arXiv: 1704.01279. URL: http://arxiv.org/abs/1704.01279 (visited on 05/09/2020).

[37] Olivier Lartillot and Petri Toiviainen. "A Matlab Toolbox for Musical Feature Extraction from Audio". In: *Proc. of the 10th Int. Conference on Digital Audio Effects (DAFx-07), Bordeaux, France, September 10-15, 200* (2007), p. 8.

[38] Vinoo Alluri et al. "Large-scale brain networks emerge from dynamic processing of musical timbre, key and rhythm". In: *NeuroImage* 59.4 (Feb. 15, 2012), pp. 3677–3689. ISSN: 1095-9572. DOI: 10.1016/j.neuroimage.2011.11.019.

[39] *TIMBRE | meaning in the Cambridge English Dictionary*. Cambridge Dictionary. Library Catalog: dictionary.cambridge.org. 2020. URL: https://dictionary.cambridge.org/dictionary/english/timbre (visited on 05/30/2020).

[40] Zenas C. Chao, Yasuo Nagasaka, and Naotaka Fujii. "Long-term asynchronous decoding of arm motion using electrocorticographic signals in monkey". In: *Frontiers in Neuroengineering* 3 (2010). Publisher: Frontiers. ISSN: 1662-6443. DOI: 10.3389/fneng.2010.00003. URL: https://www.frontiersin.org/articles/10.3389/fneng.2010.00003/full#F1 (visited on 05/30/2020).

[41] Bernhard Graimann et al. “A Comparison between using ECoG and EEG for direct brain communication”. In: *IFMBS proceedings, EMBEC05 European medical and biological engineering* 11 (July 8, 2005). URL: https://www.researchgate.net/publication/307135204_A_Comparison_between_using_ECoG_and_EEG_for_direct_brain_communication (visited on 06/14/2020).

[42] Neetha Das, Tom Francart, and Alexander Bertrand. *Auditory Attention Detection Dataset KULeuven*. Version Number: 1.0.0 type: dataset. Aug. 30, 2019. DOI: 10.5281/ZENODO.3377911. URL: https://zenodo.org/record/3377911 (visited on 05/31/2020).

[43] Numpy Developers. *NumPy*. URL: https://numpy.org/ (visited on 06/13/2020).

[44] SciPy Developers. *SciPy*. URL: https://www.scipy.org/ (visited on 06/13/2020).

[45] scikit-learn Developers. *scikit-learn: machine learning in Python*. URL: https://scikit-learn.org/stable/ (visited on 06/13/2020).

[46] Matplotlib Developers. *Matplotlib: Python plotting*. URL: https://matplotlib.org/ (visited on 06/13/2020).

[47] Sarah M. Carpentier et al. “Complexity matching: brain signals mirror environment information patterns during music listening and reward”. In: *bioRxiv* (July 5, 2019), p. 693531. DOI: 10.1101/693531. URL: https://www.biorxiv.org/content/10.1101/693531v1 (visited on 01/11/2020).