From Preprocessing to Exploratory Analyses: Enriching the Analysis of EEG Data with Interactive Data Visualization

Anders Schultz
Student ID: 201909248
Supervised by: Hans-Jörg Schulz
Co-supervised by: Kaare Mikkelsen

June 2024

Abstract

This thesis explores the enhancement of Electroencephalography (EEG) data analysis through the integration of data preprocessing methods and interactive data visualization. While methods for data preprocessing and analysis are well-established, artifact detection workflows lack seamless integration with interactive visualization tools.

A comprehensive preprocessing pipeline focused on filtering, data cleaning, data transformation, and creating relevant EEG metrics is proposed, transforming raw EEG signals into a clean format suitable for in-depth analysis.

At the core of this research is the development of an interactive visualization tool, designed to support EEG experts in their data preparation workflows. This tool was developed based on design requirements identified through informal client interviews and questionnaires from published litterature, and the implementation is inspired by the Reference Model Pattern. The tool provides an overview of EEG data with multiple coordinated views, including a spectrogram tick chart, a frequency line chart, and a correlation matrix, enabling real-time data exploration for artifact detection. It supports novel interaction techniques such as brushing and cross-filtering, and the user interface offers extensive customization options.

By combining robust preprocessing with advanced visualization techniques, this work aims to improve the accuracy and efficiency of EEG data analysis, ultimately contributing to better diagnostic and research outcomes.

TABLE OF CONTENTS

Chapter 1

Introduction

1.1 EEG signals

Electroencephalography (EEG) monitors spontaneous electrical activity in the brain, detecting voltage fluctuations from ionic currents within neurons. electroencephalography captures brain waves—neuronal activity patterns categorized by frequency, associated with states like sleep or active thinking—and event-related potentials, which are changes in response to stimuli or events.

electroencephalography is a scientific method for diagnosing neurological conditions such as epilepsy and seizures. The diagnostic process, shown in figure 1.2, typically begins with placing electrodes on the scalp of patients (1) according to an international system, as illustrated in figure 1.1. This is followed by a data recording (2), typicallylasting several hours. After data collection (figure 1.3), the analysis phases (3) & (4) begin, involving data preprocessing, artifact removal and identifying diagnostic patterns. Finally, the patient receives feedback and potential treatments are discussed (5).

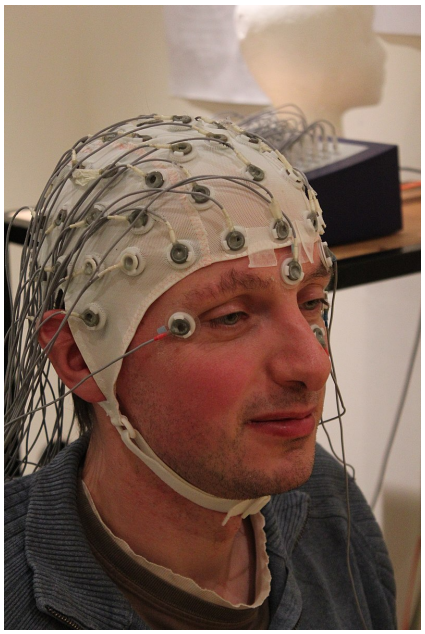


Figure 1.1: Electrodes placed in an EEG hood ensuring correct placements. Source: Wikimedia Commons

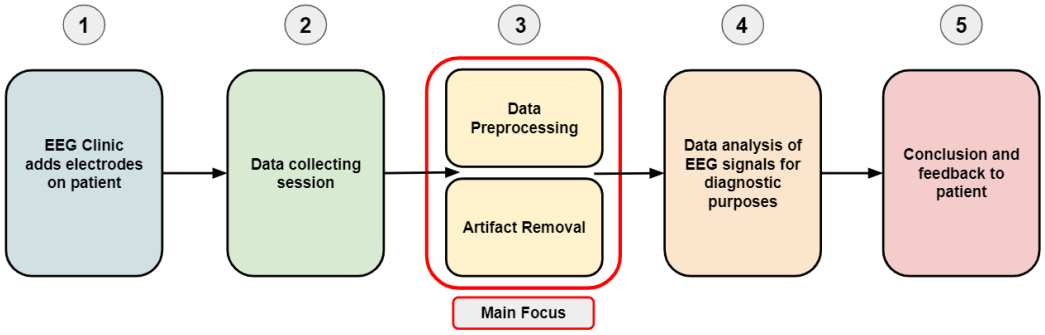


Figure 1.2: Typical Workflow for EEG Practitioners

This thesis focuses on phase (3) of the EEG workflow showcased in Figure 1.2, specifically data preprocessing and artifact identification.

The data captured by electrodes presents challenges such as electrode malfunctions, environmental noise, signal amplification, and artifacts from non-brain sources like eye movements or cardiac activity. These issues complicate data analysis. Developing advanced, interactive visualization tools to assist the work occurring in phase 3 can enhance analytical capabilities of EEG experts, leading to greater diagnostic accuracy and efficiency.

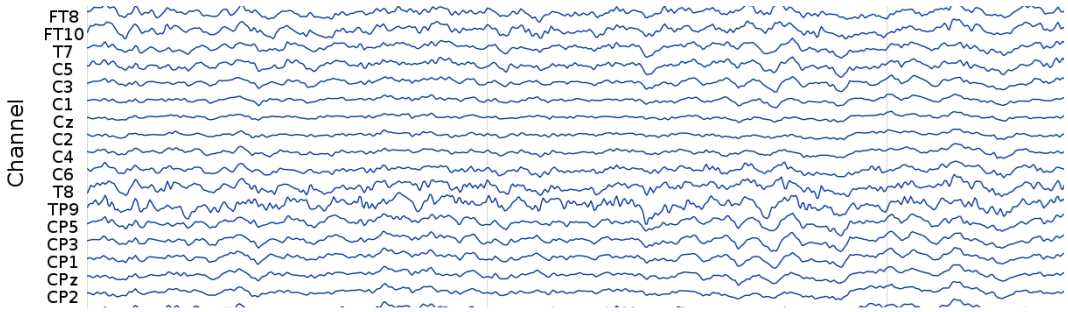


Figure 1.3: EEG signals from electrodes over time. Source: Wikimedia Commons

In recent decades, the explosion of big time series data has transformed sectors from finance to healthcare. This type of data, characterized by its high dimensional structure, extensive volume, and temporal dependencies, captures dynamic processes over time, offering detailed insights. For example, in environmental science, long-term climate data is crucial for understanding and predicting climate change, shaping global policies. Advanced data science and visualization tools can yield profound insights across fields. EEG signals are a prime example in healthcare, showcasing the promises and challenges of complex time series data.

1.2 Motivation and Challenges

In the traditional EEG workflow, interactive visualization tools are notably limited. Tools like EEGLAB and MNE Python have established data preprocessing, analysis, and visualization. However, despite these developments, interactive visualizations are not often available [16]. Current static EEG visualization methods, such as line charts, butterfly plots, and topoplots, provide great insights. Yet, according to EEG expert Kaare Mikkelsen (KM), the interactivity of these tools are mostly limited to panning, zooming, and marking artifact regions. Users cannot interactively explore and analyze i.e. electrode signal correlations through these visualizations; instead, they need to adjust the underlying code base, based on observations from (often static) visuals. Typically, these visualizations are used to inspect preprocessed data; when significant findings are detected, they are noted, the analysis is repeated, and new visuals are generated. This repetitive process, alternating between coding and visual inspection, can become cumbersome.

Introducing dynamic interaction and customizability can transform the role of visualization tools throughout different EEG workflow stages. Advanced interactive features, such as cross-filtering and real-time data querying, enhance the practical value of the visualizations utility in the data preprocessing and analysis phases Figure 1.2(3). This streamlines workflows and improves diagnostic accuracy, directly benefiting patient care by enabling faster and more precise diagnoses, leading to better treatment outcomes.

Thus, there is a high potential associated with integrating enhanced data visualization tools in EEG analysis. Such tools can help tackle data interpretation challenges and enhance the workflow for experts like KM, enabling them to visually engage with data. This project aims to embed an interactive data visualization tool within the structured EEG workflow, offering immediate visual feedback to various queries and selections, thereby optimizing efficiency in artifact detection and data insight.

1.3 Contribution of this Thesis

The objective of this research is to develop a visualization tool that enhances the phase 3 of the workflow of EEG professionals dealing with large time-series data (Figure 1.2). The contributions in this thesis include:

1. Adding additional steps to the traditional data preprocessing pipeline that calculates new relevant metrics for artifact detection and reduces the data volume.

2. Formulating design requirements for the visualization tool based on informal interviews with field experts and recent published questionnaires [13].

3. Showcasing a conceptual visualization design and software architecture based on these requirements to facilitate the workflow.

4. Developing a visualization tool prototype that offers comprehensive overviews and detailed views of EEG time series data, aiding practitioners in artifact analysis.

The visualization tool incorporates underutilized interaction techniques, fostering more exploration and supporting diverse tasks across the workflow. This tool is designed to augment the analytical capabilities of EEG experts by allowing simultaneous visual inspection and data manipulation.

Recent discussions with visualization expert Hans-Jörg Schulz (HJ) and seasoned EEG analyst Kaare Mikkelsen KM from Aarhus University revealed that cross-filtering, which filters data based on multiple views, is not utilized currently, highlighting an existing gap between visualization theory and tools used within the EEG community. This project aims to bridge that gap.

1.4 Structure of the Thesis

The thesis contains nine main chapters, each building on the previous one, developing a comprehensive understanding of the topic. Below is an overview of each chapter:

Chapter 2 reviews the existing literature on visualization techniques for big time-series data and current methods in the EEG sector, establishing the theoretical foundation.

Chapter 3 describes the data preprocessing pipeline for the EEG-data, developed for and used as part of this thesis. This includes an introduction to utilized software libraries and a detailed explanation of pipeline sections such as bypass filtering, handling missing data, aggregation, and spectrogram analysis.

Chapter 4 details the process of deriving design requirements for the visualization tool, beginning with a formative client interview and feedback from published questionnaires aimed at EEG field experts. It also analyzes data structures and derives relevant tasks.

Chapter 5 explores the design process for the application layout and the individual visualization elements, evaluating different layouts and visualization techniques, and concluding with considerations on color scales, mapping techniques, and outlier detection methods.

Chapter 6 describes the visualization software design structure, addressing aspects like dataset storage and utilization, creating visualizations, and enabling controlled inputs.

Chapter 7 illustrates how the tool supports a typical EEG signal analysis workflow and showcase the identification of interesting artifact patterns, demonstrating its real-world applications.

Chapter 8 assesses the functionality and usability of the tool, discussing its strengths and limitations, and explores potential broader applications in other fields.

Chapter 9 summarizes the key findings from the project, offers recommendations for future research, and concludes with a brief reflection on the project as a whole.

Chapter 2

Literature Review and Related Work

This chapter delves into the existing literature that underpins key areas of the project: Data visualization techniques for big time-series data and current methodologies used in EEG data preprocessing, artifact analysis, and workflow. This review equips us with essential background knowledge, offering insights into current advances and challenges within these fields. This is crucial for developing a visualization tool tailored to meet the specific needs of EEG practitioners. Understanding the current state-of-the-art in time-series visualization and EEG practices, sets the stage for proposing a solution incorporating interactive visualization tools into data analysis workflows.

2.1 Time Series Visualization Techniques

Before addressing specific visualization techniques, it is important to acknowledge the unique structure of EEG data. EEG recordings typically encompass signals from 20-50 electrodes recorded over linear time periods extending up to 8-10 hours, with data sampling rates exceeding 200 observations per second. This often leads to about 250 million data points and raw data .fdt files typically being 1GB. This high-density temporal data necessitates specific techniques to effectively represent such dense information.

Various multivariate visualizations for linear time in 2D have been proposed, explored, and documented in the literature. Each technique is designed for time series data, having distinct strengths and limitations. Researchers have focused on developing methods that optimize the display of large time-series data with multiple variables within constrained pixel spaces while still conveying high-quality information about the underlying data. This section reviews these techniques, providing a foundation for understanding how they can be applied or adapted to our tool.

The **Horizon Graph (HG)** (Figure 2.1), initially developed as "two-tone pseudo coloring" by Saito et al. [4], is a compact visualization technique ideal for multi-channel big time-series data. It divides the value range into multiple bands, each with a unique

color, and layers these bands within a single area chart. This approach significantly reduces space requirements while retaining detailed information.

Subsequent studies, such as Heer et al. [5], have explored the optimal number of bands and chart size for estimation accuracy, demonstrating that Horizon Graphs are superior to traditional line charts in a limited visual space. These findings led to recommendations on band count and chart pixel size.

Further refinements include the collapsed HG's in Dahnert et al. [7], enhancing data representation using 2D compression with bivariate color maps. Additionally, Perin et al. [6] introduced an Interactive Horizon Graph (IHG) incorporating baseline panning and y-axis zoom, outperforming standard HGs in tasks requiring complex matching and comparisons. This chart excels in space utilization and interactive features but has a steep learning curve for practitioners.

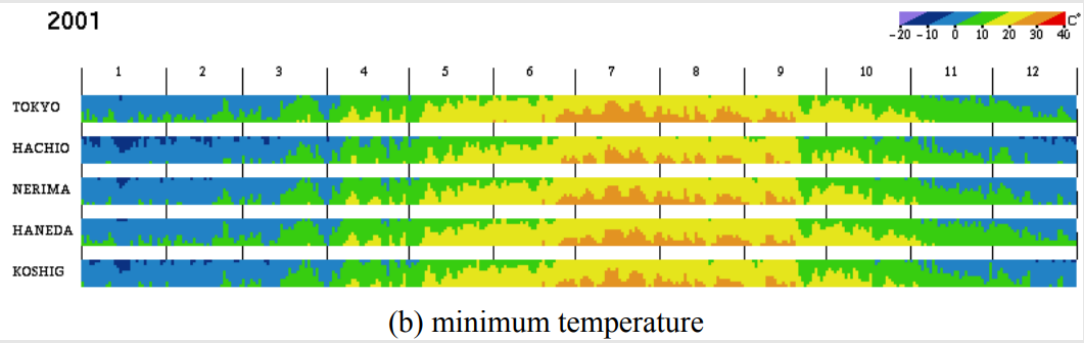


Figure 2.1: Horizon Graph showcasing the minimum temperature in 5 cities of Japan over 12 months [4]

The Heatmap, often considered a binned scatterplot, is a powerful method for visualizing time-series data. Typically, the x-axis represents time, the y-axis represents a variable, and color coding indicates the intensity of a third (often quantitative) variable.

Heatmaps are versatile and can display data from various fields. For instance, Pleil et al. [8] use heatmaps to visualize environmental and biomarker measurements, where the x-axis signifies time in days, the y-axis categorizes particle types, and color denotes concentration levels (see Figure 2.2). Similarly, Guo et al. [9] apply heatmaps to analyze Electrocardiogram (ECG) data, incorporating three variables, this time without a time metric.

Design variations can enhance readability. For example, Burris et al. [10] use a smoothing gradient (Gaussian Kernel) to represent flight paths, while others may use different bin shapes like hexagonal honeycombs, rectangles, ticks, and circles. Common interactive features like point selection, interval selection, and filtering can be integrated into heatmaps. It is also very space efficient.

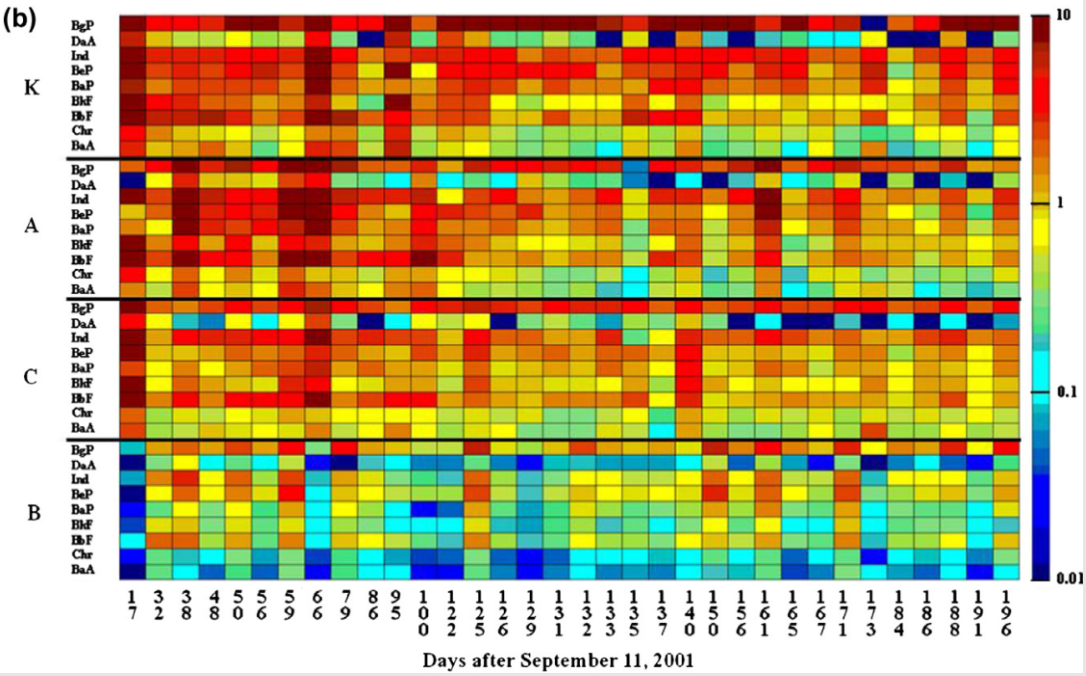


Figure 2.2: A Heatmap of air particle concentration levels after 9/11 [8]

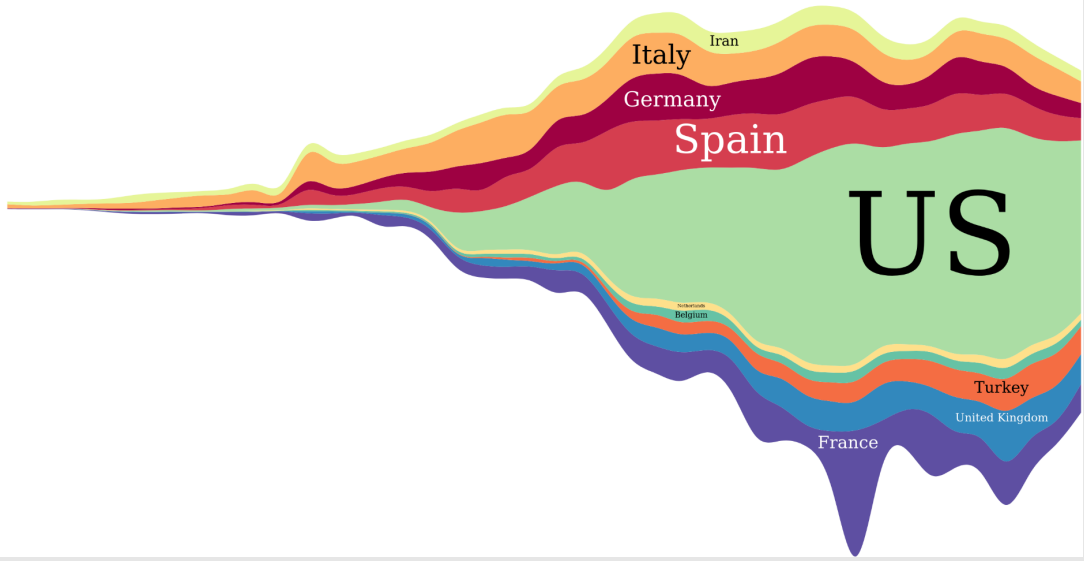


Figure 2.3: A stream graph showcasing country data over time [37]. Reduces sine illusion using SineStream

Stacked Area Charts, also known as stream graphs or theme rivers depending on their application, are designed to illustrate thematic changes and trends over time (see Figure 2.3). Introduced by Byron and Wattenberg [2] and Bu et al. [3], they address specific visualization challenges like cognitive distortions from the sine illusion and line width illusion. Stacked Area Charts are prominent for effectively representing thematic trends in various domains, including document collections and social studies. However, they struggle with direct value comparisons due to the mentioned cognitive illusions.

Continuous Triangular Model, an evolution of the discrete triangular model, is introduced by Qiang et al. [12]. This multi-scale visual representation displays all sub-intervals of a time-series in a two-dimensional field. Each point represents a sub-interval, with its value derived from a wide range of metrics (e.g., average, sum, maximum, or other complex metrics) applied to the time series within that sub-interval. This method is proposed for detecting time-related patterns in areas like traffic speed, football player movements (Figure 2.4), and optimal surfing locations throughout the year.

Sips et al. [11] explore additional applications such as climate and ocean modeling. The Continuous Triangular Model is particularly effective in identifying seasonal patterns across various scales, offering unique insights into time series data, though it may not be suitable for all types of time series.

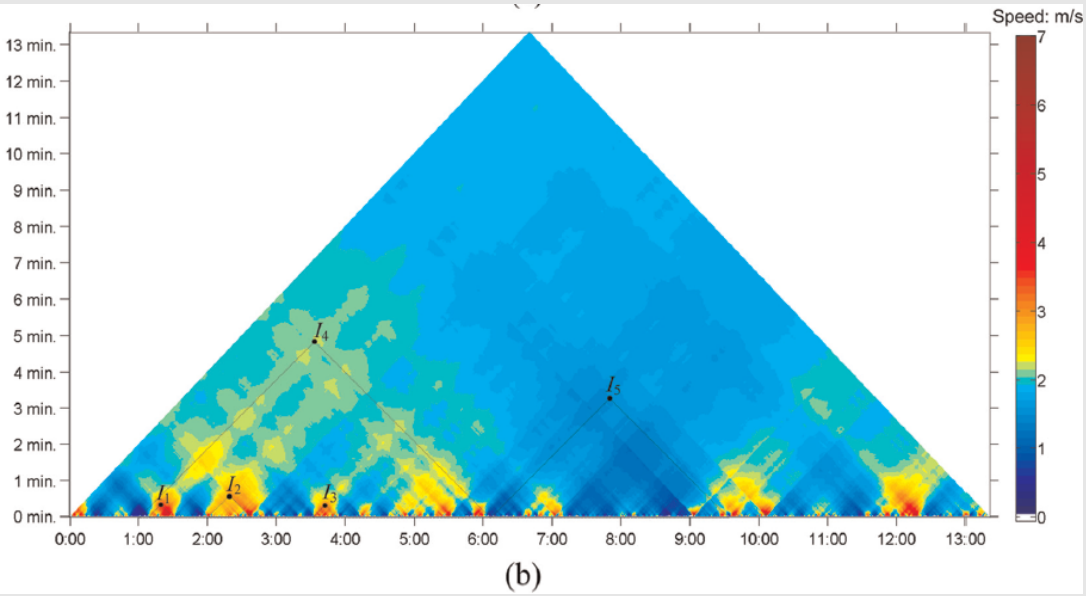


Figure 2.4: Continuous Triangular Model showing the speed of a soccer player in different time scales [12]

Section 5.1 selects a main overview chart based on the strengths and weaknesses of the four candidates described in this chapter.

2.2 EEG Data Cleaning, Analysis & Artifacts

This section explores the literature on EEG signal preprocessing and artifact removal methods. Mikheev et al. [13] conducted a questionnaire among EEG practitioners regarding analysis platforms used for EEG analysis and visualization. The results in Figure 2.5 show that popular software choices include EEGLAB [14], FieldTrip [15], and MNE python [16]. These open-source tools offer benefits such as data preprocessing, time-frequency analysis, statistical analysis, and visualizations. Performing preprocessing, data analysis, and visualizing results with basic interactions is standard in the EEG field. Additionally, figure 2.5 indicates that many practitioners use custom scripts, supporting the need for a customized interactive visualization tool for that type of workflow.

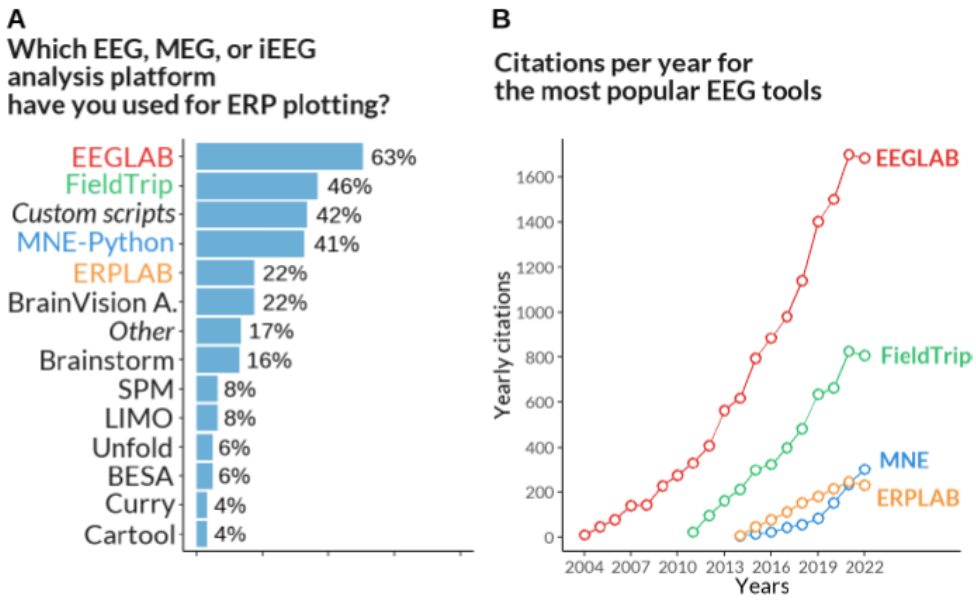


Figure 2.5: Mikheev et al. [13] questionnaire showing numbers on software usage (A) and citations per year (B)

Regarding artifact removal methods, Urigüen et al. [44] discuss state-of-the-art techniques for addressing various artifacts, such as muscle and cardiac artifacts, and the algorithms used to correct or remove them. They emphasize that no single algorithm can handle every scenario due to the problems complexity. S. D. Muthukumaraswamy [45] reviews high-frequency brain activity and muscle artifacts in Magnetoencephalography (MEG) and EEG, supporting the same claim. This highlights that manual inspection remains a reliable method for identifying artifacts and ensuring accurate analysis. Consequently, practitioners often combine artifact removal algorithms with visual inspection to achieve the best data quality. Effective visualization tools can enhance the artifact removal process during visual inspection.

Chapter 3

EEG Data Preprocessing

3.1 Motivation

Over the years, various standardized visualization pipelines have been proposed to describe the progression of data flow when creating visual representations of multivariate data, such as those by Haber and McNabb [21] and Card et al. [22]. A commonly accepted pipeline, showcased in Figure 3.1, originates from S. dos Santos and K. Brodlie [20].

This data pipeline has four main stages:

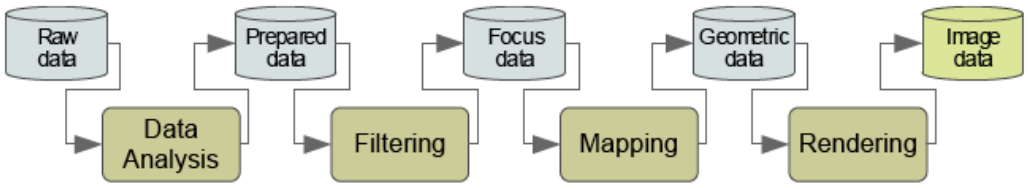


Figure 3.1: The visualization pipeline accommodating multivariate data visualization [20]

1. **Data Preprocessing & Data Analysis**: Preparing the dataset for visualization.
2. **Filtering**: Filtering the data based on desired inputs (within the visualization tool).
3. **Mapping**: Creating a representation of the visualization that includes all information related to the object, such as geometric properties and attributes.
4. **Rendering**: Displaying the data to the end user, who can then revisit any of the previous steps to change the output view.

The first stage, Data Preprocessing & Analysis, is an essential step for EEG data before visualization design and software implementation. Pedroni et al. [23] highlighted the lack of standardized preprocessing in EEG, which can cause problems in subsequent analysis.To address this, they introduced a schema for MATLAB called Automagic.

Given this context, the project incorporated a preprocessing protocol to ensure consistent data quality and the calculation of relevant metrics. This protocol includes data alignment based on established practices, such as band-pass filtering (removing noise by filtering a specific frequency range) and robust average referencing (normalizing channels based on the average value at all timestamps) [23]. This preprocessing is essential because raw EEG data sets are large, contains irrelevant frequencies and noise, and often includes faulty electrodes.

3.2 The preprocessing pipeline

This section describes the standardized preprocessing pipeline. The various steps in the preprocessing pipeline, which has been coded in Python as part of this thesis, are illustrated in Figure 3.2 3.2.The preprocessing code can be found in this open-source GitHub repository. Link: github.com/Andersschultz12/Master-Thesis-Anders-Schultz.

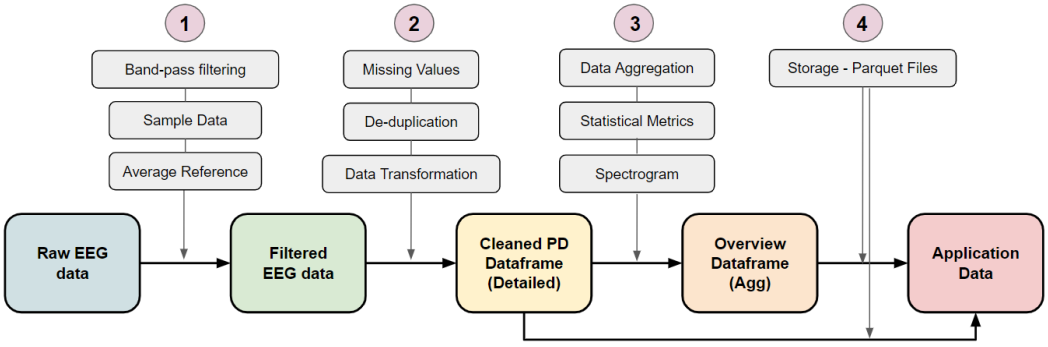


Figure 3.2: Overview of the different phases occurring in data preprocessing

The ordering of the preprocessing operations in Figure 3.2 might seem arbitrary, but it follows a logical progression from raw data acquisition to a ready-to-use format. The following points explain each phase of the data preprocessing protocol and the rationale behind their order:

Phase (1) involves band-pass filtering, performing robust average referencing, and sampling with a standardized rate. These steps ensure that meaningful data is retained, preventing clutter in the analysis.

Phase (2) addresses data cleaning by handling missing values, de-duplicating, and transforming the data to the desired format. This cleaning process ensures data integrity and accuracy, which is crucial for reducing errors in subsequent analysis. After-

wards, data is ready for detailed views in the tool.

Phase (3) involves performing data analysis and data enrichment. In this project, metrics such as spectrograms and statistical measures were calculated, gathering information relevant to EEG artifact analysis. Finally, data is aggregated into sections to achieve data reduction while retaining information.

Phase (4) stores the analysis results in a long data format and saves them in Parquet files. At this point, the data is ready for use in the visualization tool, which constitutes the next steps of the pipeline.

Each phase of the data preprocessing is explained further in the sections below:

Figure 3.3: **Facts about the raw EEG Data:**

This dataset was collected as part of a research project on ear-EEG sleep monitoring which took place in 2017. The data set contains nightly EEG recordings from 9 healthy participants ('subjects'). The dataset is formatted according to the Brain Imaging Data Structure. The EEG data format chosen is the '.set' format of EEGLAB.

The subjects were instructed to perform two recordings. In the first recording, they relax in a chair either reading or watching television, prior to going to bed. After this, the real recording took place during the night and began when the subject slept. The recording equipment was mounted in the afternoon, and the recordings took place at the subject's home. The data originates from openneuro.org. Link: openneuro.org/datasets/ds004348/versions/1.0.4

3.3 Phase 1: MNE Software

All four phases of data preprocessing and analysis are performed using Python. The first phase uses MNE-Python, an open-source package containing various standard methods for handling EEG measurements, including the preprocessing methods mentioned above [16].

MNE methods specialize in EEG, enabling users to apply band-pass filters with frequency thresholds, draw samples at specific rates, and perform average referencing.

The band-pass filter allows frequencies between **0.1 Hz and 100 Hz**, capturing the five brain rhythms [44]. This filtering uses the Fourier Transform (FT), discussed in the spectrogram analysis subsection. The **sample rate** is typically 200 or 250 measurements per second for all electrodes, adhering to standard EEG practices. The **average**

reference is defined as:

$$y_i = x_i - \frac{1}{N}\sum_{j=1}^{N} x_j$$

where N is the number of channels, x_i is the original signal for electrode i, and y_i is the re-referenced signal. This method highlights channels with unusual data patterns more clearly.

3.4 Phase 2: Data Cleaning

3.4.1 Missing Values & De-duplication

Due to the application of band-pass filtering, data cleaning is necessary before calculating additional metrics. Addressing missing values is a crucial step. Cheng et al. [29] recommend creating missingness maps to identify and understand the patterns of missing data, a practice followed here. There are two primary approaches to addressing missing values:

- **Imputation:** Replacing missing values with estimates. Algorithms include linear interpolation, placeholder values like the mean, median, or mode, and advanced techniques such as regression.
- **Amputation:** Removing data regions (rows or columns) with a high proportion of missing values, making meaningful analysis difficult.

General rules of thumb apply when deciding on amputation. If a **column** has **less than 5%** of its values missing, consider **row deletion**. If **more than 40%** of the values are missing, consider **column deletion**. For values between these thresholds, imputation is recommended, as its is too significant to ignore but too small to remove entirely. For example, a subject channel with 100% missing values after phase 1 was amputated. Linear interpolation replaced the remaining missing values after initial threshold checks.

3.4.2 Data Transformation

Data transformation ensures compatibility with visualization software. As shown in Figure 3.4, the data is reorganized from a wide data format to a long data format, the standard for visualization software, simplifying further analysis and manipulation. The data is now ready for detailed visual analysis, with a summary table created later for overview visualizations.

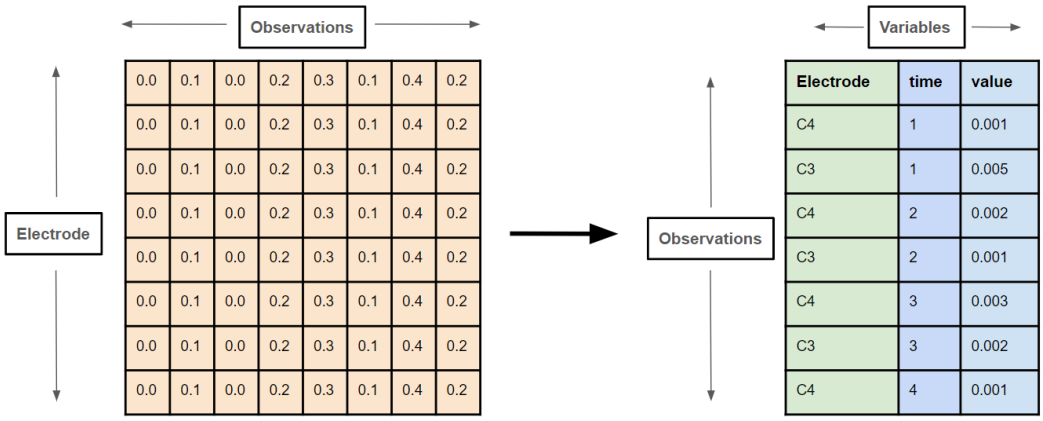


Figure 3.4: Transforming data from wide to long form enables visualization software and makes further manipulation straightforward

3.5 Phase 3: Data Analysis

3.5.1 Aggregation

EEG data consists of millions of observations per electrode, necessitating data reduction for effective chart rendering. aggregation methods divide the data into equally sized time intervals, allowing metric calculations based on these intervals.

3.5.2 Additional Metrics

Before aggregation, additional metrics are calculated to identify unique patterns that spectrogram analysis might miss. These metrics are then aggregated to the sum for each time interval.

(1) Sign Change (SC): This metric is calculated using the following logic:

$$SC(x_i) = \begin{cases} 0, & \text{if Sign}(x_i) == \text{Sign}(x_{i-1}) \\ 1, & \text{otherwise} \end{cases}$$

The aggregation sums the SC values of each time interval.

(2) Sequence Exceeding Threshold: This metric is calculated using the following logic:

$$SET(x_i, \text{Val}, \text{SeqN}) = \begin{cases} 1, & \text{if } |x_i - x_{i-1}| > \text{Val} \quad \text{for sequence length SeqN} \\ 0, & \text{otherwise} \end{cases}$$

This metric identifies regions where absolute deviations between consecutive points exceed a predefined threshold "Val". Different (threshold, sequence lengths) pairs highlight different aspects of the signal. The aggregation sums the occurrences of such

sequences in each time interval.

(3) Envelope: The envelope width is calculated using the following formula:

$$\text{envelope\_width} = 2 \times \text{moving\_std} \times \text{num\_std}$$

where moving_std is the rolling standard deviation of the measurements computed over a window of size 10 with a minimum period of 1. num_std is the number of standard deviations used to scale the envelope width. For each aggregated section, the envelope statistic is the mean value of all its data points.

(4) Statistical Measures: Various statistical measures can be calculated on each of the binned time intervals. For instance, the empirical mean, variance, and standard deviation:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{Var}(X) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \quad \text{Std}(X) = \sqrt{\text{Var}(X)}$$

or the minimum and maximum. Other metrics include the Mean Error (ME) and Mean Squared Error (MSE), to check how well a data chunk resembles a linear regression:

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

3.5.3 Spectrogram Analysis

A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. The spectrogram is often depicted as a Heatmap like Figure 3.5a where intensity is signified by the color for a given time and frequency. The method for generating a spectrogram is called the Fourier Transform (FT).

The FT is a mathematical transform used to analyze the frequencies present in a continuous signal by converting a time domain function of into a frequency domain function. Mathematically, the FT of continuous function f(t) is:

$$\hat{g}(f) = \int_{\infty}^{\infty} g(t)e^{-i\omega t}dt$$

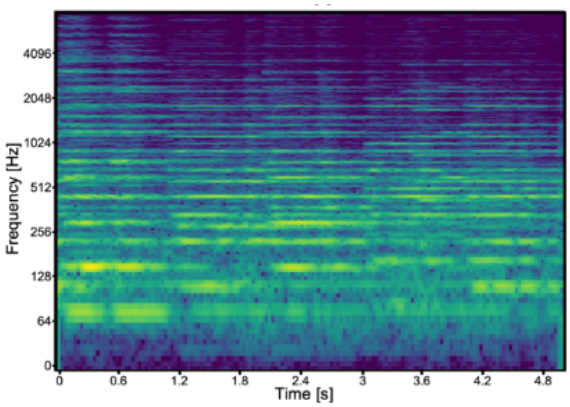
Where $\omega = 2\pi \cdot f$ refers to the angular frequency, i is complex number, t is time, and f is frequency. Perhaps explained more intuitively, it's like winding the graph around a circle in the complex plane with a frequency f, and then the centre of mass $\hat{g}(f)$ is calculated by integrating in the relevant time interval.

Since signal processing often involves discrete signals (in contrary to analysing continuous signals), the FT has a discrete version called the Discrete Fourier Transform (DFT), which. courtesy of Cochran et al. [30] is described as:

$$A_r = \sum_{k=0}^{N-1} X_k e^{-2\pi irk/N} \qquad r = 0, \ldots, N-1 \tag{3.1}$$

Where A_r is the r-th coefficient of the DFT and X_k denotes the k'th sample of the time series consisting of N samples. To draw a comparison between FT and DFT, $g(t)$ corresponds to the discrete X_k and $e^{i\omega t}$ corresponds to $e^{-2\pi irk/N}$.

The Fast Fourier Transform (FFT), introduced ion Cochran et al. [30], is an algorithm used to efficiently compute the DFT for time series data. Developed by Cooley and Tukey (1965), the Fast Fourier Transform is a revolutionary algorithm that uses the divide and conqueror technique where the initial step splits even and odd indice coefficients. Ultimately, the time complexity of calculating the DFT is reduced to $O(2Nlog_2N)$ compared to the naive complexity of $O(N^2)$. Calculating the DFT can be done using an FFT method in the NumPy library.



(a)

Variables

Obs.

Electrode	Time	Result	Frequency
C4	1	0.001	0
C3	1	0.005	0
C4	1	0.002	1
C3	1	0.001	1

(b)

Figure 3.5: (a): Example of spectrogram visualized (Source: Wikimedia commons), (b): Final data structure for spectrogram results after transformation

To easily interpret FFT results, taking either the absolute values $|x|$ (amplitude) or the squared absolute values $|x|^2$ (power) is often performed. The amplitude spectrum indicates how strong or weak each frequency is in the original signal, while the power spectrum represents the power or energy contribution of each frequency component. In signal processing, the power spectrum provides better information, and is thus utilized in the data proprecessing.

To match the previous aggregation, the time series is split into the same time segments, and the FFT algoithm is performed on each sample. This gives a spectrogram result with desired time regions. When transformed into the long data format, the structure resembles Figure 3.5b. When one frequency is selected, the format follows the format of the aggregated data, and the two datasets can be joined together for visualization purposes.

Chapter 4

Requirements for Visualization Tool Design

This chapter outlines the necessary information to design and develop an effective visualization tool. Key components include understanding the data structure and the existing methods and workflows used by EEG experts. To gather this information, an informal interview with EEG expert KM was conducted, and literature on state-of-the-art options by other practitioners was reviewed. This information is essential for determining the design requirements of the visualization tool needed to satisfy the needs of EEG data analysts and beyond.

4.1 Informal interview

This section reviews the interview conducted with EEG practitioner KM. The primary design goal of the visualization tool is to enhance his workflow, particularly in detecting, noting, and removing EEG artifacts. Artifacts are erroneous data measurements that should be excluded from further analysis. EEG artifacts can arise from various sources, such as electrode malfunctions, high noise levels, and non-brain activities, including eye movements, muscle movements, or cardiac activity. This interview provided crucial insights into data abstraction, task abstraction, and other important design considerations.

Data Abstraction: According to Schulz et al. [24], it is essential to define objective data descriptors that capture both regular and irregular attributes of time-series data. These attributes are crucial for decisions regarding visual elements like color maps. Below is an overview of the different data descriptors based on KM's insights, subdivided into Data Flow Descriptors (DFD) and Data Space Descriptors (DSD), as illustrated in Figure 4.1

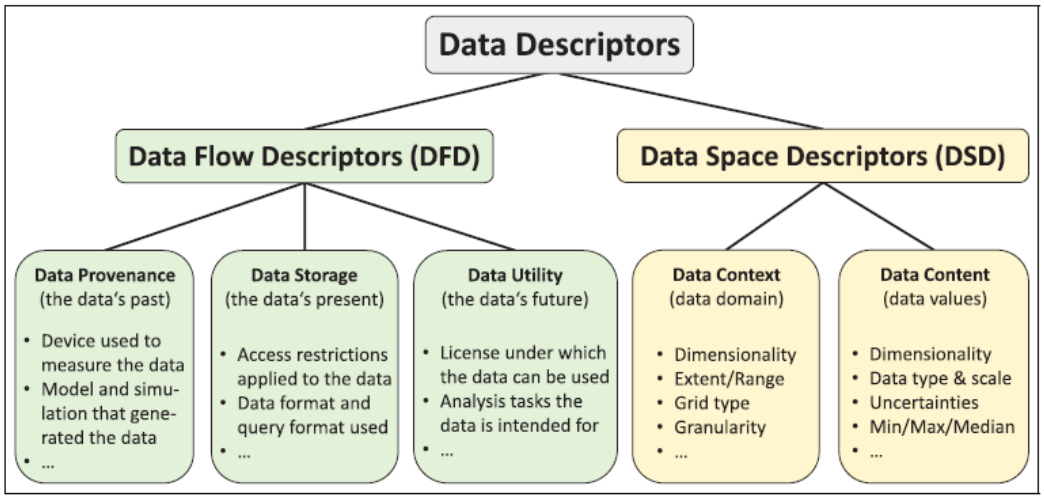


Figure 4.1: Schulz et al. [24] visual representation of the different objective data descriptors used in data abstraction

Data Flow Descriptors (DFD):

- **Data Provenance:** EEG data originates from electrodes attached to the patient's scalp, detecting voltage fluctuations during sleep sessions.
- **Data Storage:** The primary data are stored in extensive .fdt or .set files and organized according to Brain Imaging Data Structure (BIDS) standards, ensuring systematic data retrieval.
- **Data Utility:** This data helps EEG scientists diagnose patients effectively, leading to better scientific outcomes.

Data Space Descriptors (DSD):

- **Data Context:** Before preprocessing, the data is a tabular structure where each row corresponds to all measurements for a specific electrode over time. The frame of reference is electrode ID and time. The number of channels varies from 20-40 electrodes but can extend to hundreds for in-depth analyses. EEG sampling rates used in practical analysis are usually 200 samples per second or higher. This results in a dataset encapsulating millions of observations for each electrode, leading to a vast multivariate time series dataset. The data is valid for the given subject and can include both awake and asleep states during recording, with potential missing values if band-pass filtering is performed.
- **Data Content:** The measurements at the frame of reference (time & electrode) are continuous numerical voltage measurements. The typical value range varies from 10 to 100 microvolts, centering around a median with potential outliers. After preprocessing, the data column contents include a categorical electrode

variable, a time variable, and the metrics, including FFT results, being continuous numerical variables for all metrics discussed in Chapter 3.

This abstraction captures the details inherent to EEG time-series data, leading to effective visualizations. With a better understanding of the data structure, the next step involves defining the main tasks associated with this dataset and the actions that facilitate insight.

Task Abstraction: Adrienko categorizes visual data analysis tasks into elementary and synoptic categories [26]. elementary tasks, focusing on individual references or characteristics of data, are subdivided into three sub-categories: lookup (direct and inverse), comparison (direct and inverse), and relation seeking (one and multiple). synoptic tasks involve analyzing aggregate behaviors or patterns across extensive data sets or volumes.

Insights from the interview with field expert KM provide practical context to these abstract categories, particularly within EEG data analysis workflows. KM describes the process of identifying EEG artifacts:

> *When we judge if a channel is dominated by artifacts in a give time, we both look at individual channels, and multiple channels in concert. (...) If a channel behaves different from the rest on a given timestamp, it's important. (...) electrodes need to always be judged with respect to others.* - Kaare Mikkelsen

Moreover, the identification of artifacts also involves analyzing the frequency spectrum of the EEG data:

> *(...) I look if the frequency spectrum is sensible (too smooth or dominated by high frequency). I look at the size of fluctuations or very sudden shifts (...) It can also be a problem if a pattern repeats itself too regularly.* - Kaare Mikkelsen

These practical insights relate directly to Adrienko's framework, where the EEG artifact workflow is facilitated by elementary tasks such as the direct comparison task and lookup tasks. These tasks guide the identification of regions that may contain artifacts. Synoptic tasks do not occur in this phase of EEG artifact detection. Therefore, it is important to design visualization tools using visualization techniques and interactive features that effectively support these elementary tasks.

In Figure 4.2, a common static EEG visualization technique is shown. Horizontally stacked line charts display data within a specified time interval, with artifacts distinctly marked in vibrant red. Consistent with KM's observations, identified artifacts show clear deviations compared to adjacent channels, either temporally or spatially. Entire channels are often highlighted when the signal suggests an unusable electrode.

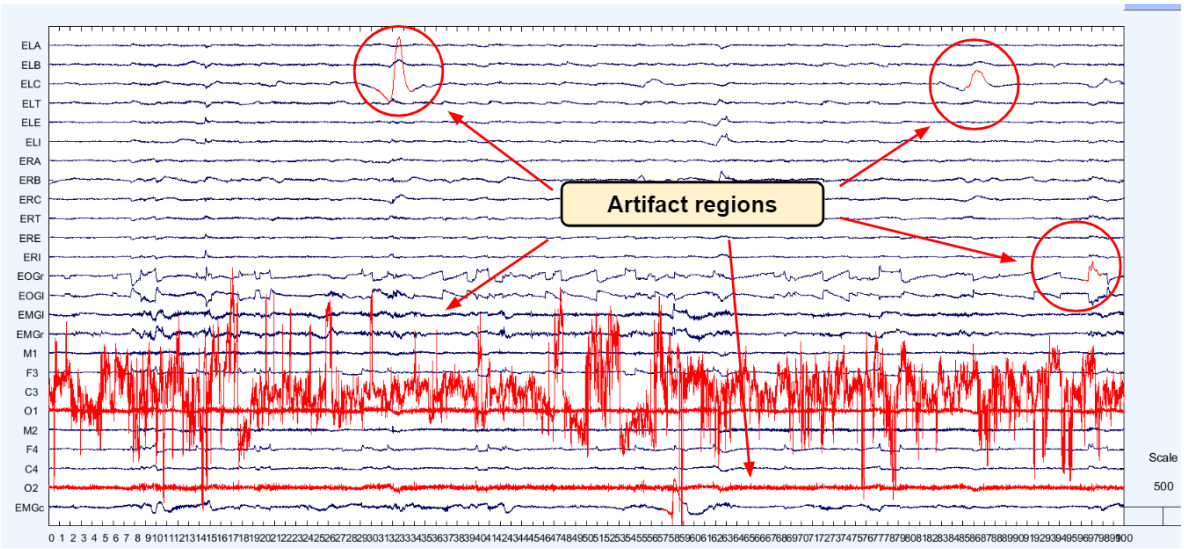


Figure 4.2: Artifacts marked in red in horizontal parallel line charts

4.2 Questionnaires

Understanding what field experts prioritize when using tailored visualization tools for EEG time series data analysis is critical. Mikheev et al. [13] conducted a survey of EEG data practitioners with varying backgrounds and levels of experience. The survey focused on what features practitioners deem important in the visualization tools they use. Although the study's findings do not imply causal relationships and suffers from a biased sample and small sizes, the insights gathered still serve as valuable guidelines reflecting the general opinions of experts analyzing EEG time-series data.

Figure 4.3 shows that **customization**—specifically the flexibility to tweak plot attributes such as colors and line widths —are the most valued features. This is closely followed by the ability for content to be **reproducible** for other practitioners using the same methodology and data, being **publishable** by saving results in different formats, and generating plots by **coding**. While interactivity, loading speeds, and the Graphical User Interface (GUI) are important, they are not as critical as the aforementioned features. These findings align with the initial discussions with KM, emphasizing his preference for customization, interactivity, and straight forward artifact notation.

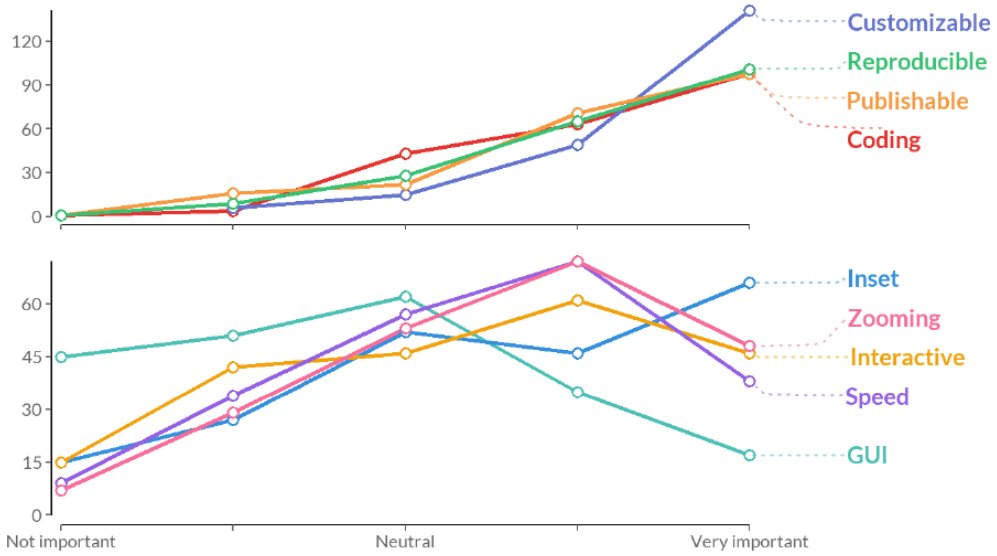


Figure 4.3: Rated importance of each feature. Y-axis showcases number of selections for a particular response. [13]

A significant majority of survey respondents use the full set of electrodes for their analyses, while others use varying numbers from 5% to 90%. This diversity highlights the need for the visualization tool to offer flexibility in adjusting parameters like the number of channels used in analyses. Furthermore, 39% of practitioners were unaware of the perceptual controversies surrounding color maps, emphasizing the need for software developers to encourage the use of more effective color maps by setting appropriate defaults and providing clear information about color mapping. Notably, the EEG field shows a prevalent use of rainbow color maps, known for not supporting color blindness and lacking a continuous increase in luminosity. Consequently, rainbow-like color scales will not be set as the default option.

4.3 The Requirements

The previous sections provided valuable insights into many aspects that need to be considered in designing the final visualization tool. Madsen et al. [27], who developed a tool for visual exploration of rheological test results from soft materials, produced a list of design requirements for their solution following close collaboration and informal interviews with domain experts. Using a similar method, based on informal interviews with Kaare and literature on questionnaires about popular priorities among practitioners, the following five design requirements will be addressed:

(D1): Customizability: The graphical interface should be configurable to align with the data and analysis tasks at various stages of the workflow.

(D2): Flexibility: The tool should adapt to various new data (new subjects) while still

providing the same functionality and inherent value.

(D3): Interactivity: The tool should enable direct visual data analysis within the software, eliminating the need to adjust and run new code.

(D4): Ease of Use: The software should be self-contained and assist in various aspects of the workflow, ultimately simplifying data analysis.

(D5): Saving Information: The software should have functionality to save relevant data for visualization and artifacts, and allow these parameters to be loaded later.

Chapter 5

Designing the Visualization

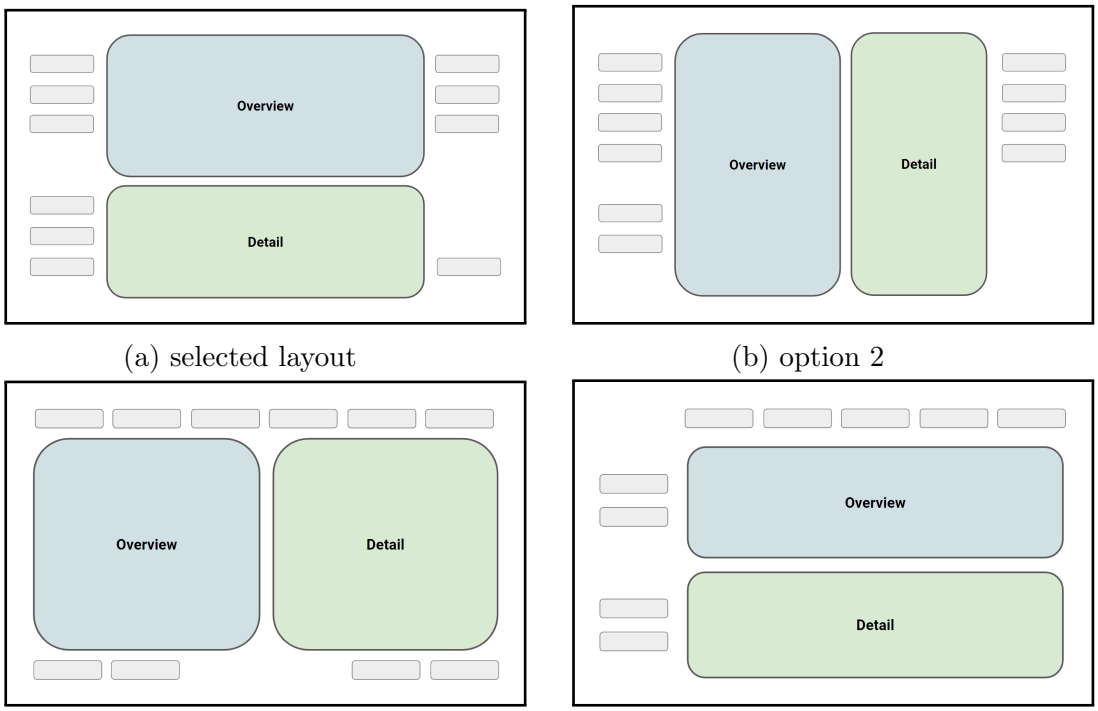
In this chapter, we'll use the acquired knowledge about the EEG field, the method of task and data abstraction, and the five identified design requirements to create design prototypes for the visualization tool

Generating design ideas for individual visualizations and the general layout was accomplished using the **5 Design-Sheet** methodology, described by Roberts et al. [25]. This method enabled effective brainstorming and gave inspiration for effectively structuring the tool and visuals.

During the brainstorming phase, an important consideration was the Visual Information Seeking Mantra by Ben Shneiderman [31]: *Overview first, zoom and filter, then details-on-demand*. Additionally, he mentions the seven tasks in information visualization: Overview, Zoom, Filter, Details-on-demand, Relate, History, and Extract. These tasks align with the desired workflow for the tool. The mantra has two layout strategies: separated views (Overview and Detail) and combined view (Focus and Context). Based on the desired workflow for practitioners and the design requirements, the overview and detail layout was found to be a great choice (see Figure 5.1). References to the five design requirements in Chapter 4 start now.

- The separated views structure allows for more customization (D1) of each section independently. This offers both control and direction for performing different levels of data analysis within the workflow. It also enables ease of use (D4) by having multiple self-contained views.
- It offers an intuitive workflow due to the clear distinction between views that support different tasks, thus reducing cognitive load through easier visual interpretation.
- It provides additional interactivity (D3) by allowing data manipulation at two different abstraction levels.
- It can reduce rendering time for big data by enabling the option to load overview and detail sections separately.

In term of the positioning of each section, several layout options are possible. Four possible layout structures are shown in Figure 5.1. These layout options meet all the design requirements discussed. Therefore, the choice depends on what best supports the general workflow. It is natural to inspect the overview first, then examine detailed views upon discovering something interesting. Thus, layouts 5.1a and 5.1d are preferred for placing the overview at the top. Layouts 5.1b and 5.1c are also viable, supporting the left-to-right reading culture in most countries. The final choice is layout 5.1a, as it places key views at the top-center, highlighting their importance and making them easy to notice quickly. It also allows for good placement of customization (D1) elements near relevant charts.



(a) selected layout (b) option 2

(c) option 3 (d) option 4

Figure 5.1: Layout options considered for overview & detail structure

5.0.1 Overview Chart Placement

A key question is whether to use a single comprehensive visualization or a Multiple Coordinated Views (MCV) approach, which displays different data aspects in a coordinated manner. Baldonado et al. [28] provide guidelines for Multiple Coordinated Views use in information visualization:

(R1) Rule of Diversity: Use MCV when there are many attributes, abstraction levels, or profiles.

(R2) Rule of Complementarity: Use MCV to highlight different aspects of data.

(R3) Rule of Decomposition: Use MCV to simplify information processing into understandable parts.

(R4) Rule of Parsimony: Each view should provide unique benefits.

(R5) Rule of Self-Evidence: Ensure clear relationships between the views.

(R6) Rule of Attention Management: Use design elements to direct user attention appropriately across different views.

Due to the data preprocessing, displaying at least two metrics simultaneously is possible and aids comparison and context (R1) (R2). Additionally, each view will provide new insights (R3). The relationships between views (R5) will be evident based on the selected metrics and interactive components. Interactive views also guide the user attention to appropriate locations (R6). Thus, an MCV setup enhances understanding of the preprocessed data. With the chosen layout, the overview includes:

- Two supplementary overviews showcasing unique data aspects to aid analysis.
- Two main overviews displaying key metrics over time across different electrodes.

As shown in 5.2, the supplementary views are placed on the left due to their higher abstraction level and the cultural bias of reading from left to right [40]. This placement ensures practitioners notice these views first, gather information, and then analyze the main Tick Chart overviews.

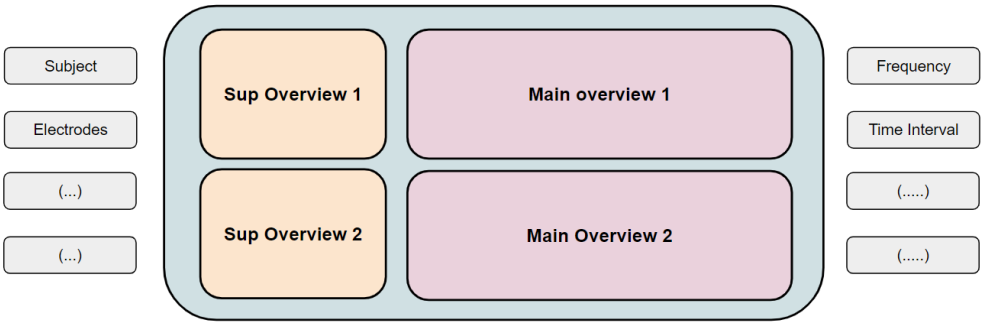


Figure 5.2: Sketch of overview chart placements in relation to each other

5.0.2 Detail Chart Placement

Similar considerations apply to the detail region layout. To support comparison tasks, users can select and compare two different channels, each with its own view. For each electrode, line charts of varying abstraction levels will display raw measurements, aiding in verifying assumptions from the overview.

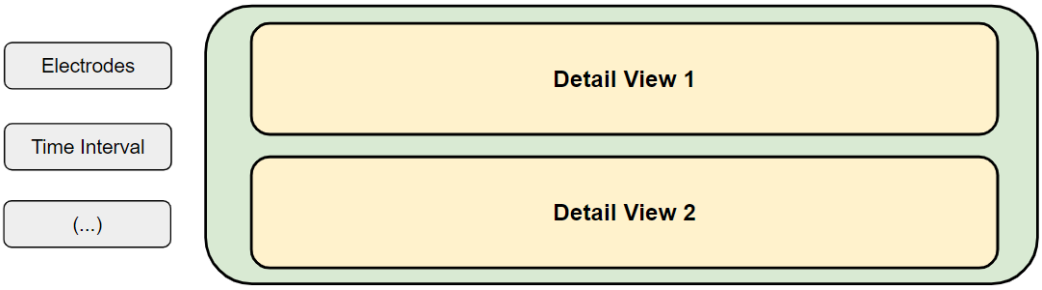


Figure 5.3: Detail layout design

Figure 5.3 shows a proposed layout, allowing for vertical comparison of different electrode combinations. The option to select specific time regions based on overview observations reduces rendering by displaying only regions of interest. These features enhance customization (D1), making the view more functional and intuitive.

5.1 Main Overview Charts

This section discusses the choice of the main overview visualization, aligning with MCV rules and design requirements, to showcase large EEG time series data efficiently. The primary tasks are direct electrode comparison and lookup. The four charts considered were: the Horizon Graph, Tick Chart Heatmap, Continuous Triangular Model, and Stacked Area Chart (see Chapter 2 for details). Through experimentation, the **Tick Chart**, a variation of the heat map using vertical lines as data points, was found to meet most requirements:

1. Suits data with a categorical variable (y-axis), a time period (x-axis), and any relevant metric (color mapping).
2. Supports customization (D1) for any set of electrodes, time, and metric.
3. Enables direct comparison between electrodes and over time.
4. Support great interactivity (D3) brushing and cross-filtering.

For a visual reference, see Figure 7.3. The other charts had merits but ultimately fell short:

The **Horizon Graph** is space-efficient and supports customization and interaction but requires a big learning curve and can lead to information overload.

The **Stacked Area Chart** is simple and offers similar interactions, but its design makes value comparison harder and it is not ideal for limited screen space.

The **Continuous Triangular Model** is unique for identifying patterns across interval sizes but struggles with EEG data comparison and has a steep learning curve. Highlighting dominant metrics in multi-scale pixels, seen in Qiang et al. [12], could partially

address this, but it may not effectively show relevant frequency patterns due to the signal's nature and lack of seasonality or trends.

5.1.1 Color scale & mapping

For the chosen Tick Chart Heatmap, a good color scale and mapping strategy are important. Given the primary tasks of data lookup and comparison, task-driven color coding can be adopted.

Lookup Tasks: These tasks involve searching for specific characteristics, facilitated by appropriate color scales and mapping methods. For data distributions where most values are similar, methods like Histogram Equalization and Box-Whisker Plot adaptation, described by Tominski et al. [32], can improve the color mapping. The Box-Whisker method subdivides data based on distribution quantiles, which are robust to outliers and improve distinction of neighboring values, as shown in Figure 5.4

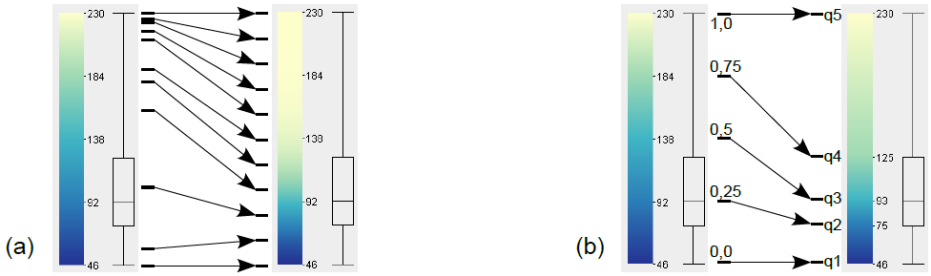


Figure 3: Color scale adaptation. (a) Histogram equalization; (b) Box-Whisker plot adaptation; Complementary Box-Whisker plots visualize data distribution.

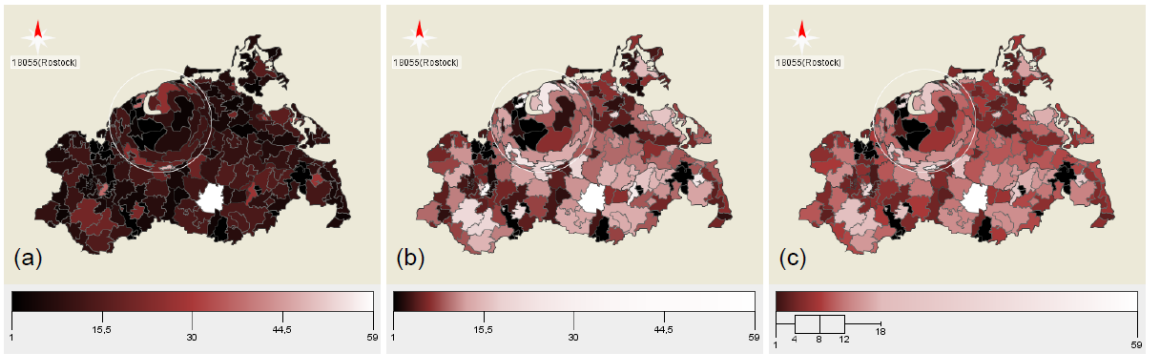


Figure 4: Visualization of quantitative data on a map. (a) Classic linear color coding; (b) histogram-equalized color coding; (c) color coding adapted based on Box-Whisker plot.

Figure 5.4: The two task-driven color coding methods in [32]

Comparison Tasks: For these tasks, color scales should be global for all channels and value ranges should be distinguishable with appropriate colors, each with unique combinations of hue, saturation, and brightness. Mikheev et al. [13] discussed the

prevalence of using the sub optimal jet (rainbow) color scale. Although not ideal, K. Moreland [34] explains the reasons for its prevalence, such as simplicity, aesthetics, or the fact that many visualization libraries use them as default. Crameri et al. [33] emphasize the importance of a uniform color scale that is colorblind-friendly.

Figure 5.5: Color scale selection: viridis and cividis being the p rimary defaults

Raja et al. [35] found that the viridis color scale excels in the correctness metric, and multi-hue scales generally perform better than single-hue scales. Nuñez et al. [36] propose a colormap called cividis (Figure 5.5), developed from the original viridis color scale, to support color deficiency. Cividis enables nearly identical visual-data interpretation, is perceptually uniform in hue and brightness, and increases in brightness linearly.

Given these considerations, viridis or cividis are excellent default choices due to their colorblind-friendly nature and ability to enhance analysis correctness. To facilitate EEG artifact lookup task, the scale will be adjusted to match the Interquartile Range (IQR) of the data distribution. For highlighting outliers, a bright red color will be used, leveraging the eye's sensitivity to red and high contrast with viridis colors.

5.1.2 Outlier Detection

To distinguish unusual data points, we implement an outlier detection method that updates dynamically based on parameters. The literature on outlier detection is extensive, and the best method depends on the data distribution. According to Leys et al. [38], a common practice is to define an outlier as the mean plus/minus three times the standard deviation:

$$\bar{x} - 3 \cdot SD < x_i < \bar{x} + 3 \cdot SD$$

However, this method can be problematic for non-normal distributions, due to the sensitivity of metrics like mean and std. Instead, Leys et al. propose using the Median Absolute Deviation (MAD) method:

$$\text{MAD} = b \cdot M_i(|x_i - M_j(x_j)|)$$

where x_j represents the original observations, M_i is the median of the time-series, and b is a constant typically set to 1.4826. For the decision criterion of identifying outliers, they recommend:

$$M - T \cdot MAD < x_i < M + T \cdot MAD \quad or \quad \frac{x_i - M}{MAD} > |\pm T|$$

where x_i is the specific data point value and T is the pre-defined threshold value. The threshold value is chosen based on what data deviations should classify as outliers. Given various metrics in the tick chart, selecting a universal threshold can be challenging. Thus, the paper's recommendation of $T = 2.5$ will be used as default

Songwon Seo [39] reviewed common outlier detection methods, including SD, Modified Z-score, Tukey Method, and MAD. They concluded that for non-symmetric data, methods related to quantile ranges or the median are preferred. Thus, the MAD method is chosen for outlier detection for all metrics.

Detected outliers are color-labeled with vibrant red. Testing revealed that plotting outliers in red effectively highlights areas worth exploring. Consequently, the tick visualization will use red for outliers, with the rest of the distribution using adjusted viridis mapping.

5.2 Supplementing Overviews

While the main overview tick charts provide valuable insights through spectrogram results and statistical metrics, additional overviews can enhance analysis by adding new data aspects.

5.2.1 Correlation Matrix

The **cross-correlation** is a widely used metric for time series in signal processing, and it can improve comparison tasks by showing relationships between different electrodes. It is defined by:

$$R_{xy}(k) = \sum_{i=1}^{n-k} x_{i+k} y_i, \quad k \geq 0 \qquad R_{xy}(k) = \sum_{i=1}^{n+k} x_i y_{i-k}, \quad k < 0$$

Given that the data is perfectly synchronized, cross-correlation with lags is unnecessary (since lags refer to the shift in time between two signals). Thus, the formula simplifies to:

$$R_{xy}(0) = \sum_{i=1}^{n} x_i y_i$$

where x_i and y_i represent points in each time series, essentially reducing to the dot product between them.

Köthur et al. [41] computed cross-correlation between subjects at different lags and intervals, creating a heat map for comparison. Since lag is not relevant here, a correlation matrix between pairs of electrodes for a given metric is implemented as an additional overview. This matrix is synchronized with the metric tick chart and showcases a different statistical measure for comparing electrodes. For a visual reference, see Figure 7.7

5.2.2 Frequency Overview

The spectrogram must relate to specific frequencies, as certain bands can indicate various states, including artifacts. Urigüen et al. [44] mention five brain frequency bands such as delta (0.5-4 Hz), theta (4-7 Hz), alpha (8-13 Hz), beta (14-30 Hz), and gamma (30-100 Hz). From the Interviews, KM also associated specific frequency ranges with artifacts.

S. D. Muthukumaraswamy [45] highlights frequency ranges linked to muscle activities, such as chewing (50-60 Hz) and brow wrinkling (30-40 Hz), and proposes methods for muscle artifact detection. However, with these methods, a clean dataset is not guaranteed. Therefore, an overview of spectrogram results across different frequency bands is crucial for understanding brain rhythms and artifacts.

Field expert KM suggests using a line chart showing the sum of spectrogram results (y-axis) against frequency (x-axis) for assessing different frequency regions. Each line represents an electrode, providing an overview to identify interesting spikes and abnormal behavior. This helps EEG experts focus on potentially abnormal frequencies for further study in the spectrogram tick chart.

Key considerations include chart width and height. Cleveland et al. [46] recommend banking to 45 degrees for optimal visual insight in line charts. The color of each electrode line is also distinct. For a visual reference, see Figure 7.5.

Chapter 6

Tool Implementation

Designing and implementing robust software frameworks for information visualization ensures functionality, consistency, and establishes a foundation for future expansion by grouping components into distinct roles.

This chapter explores the considerations for implementing a functional visualization tool.

6.1 Software Design

Choosing a suitable software framework is important for developing a flexible visualization tool that achieves the desired goals. Heer and Agrawala [42] present several design solutions to facilitate successful software design and implementation, supporting the use of classic design patterns in object-oriented software, as described by Gamma et al. (1994).

The design model employed in this project is the **Reference Model**, illustrated in Figure 6.1.

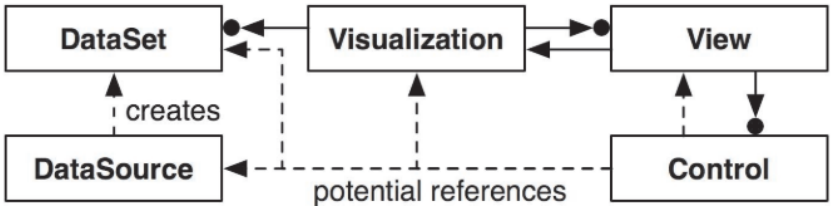


Figure 6.1: **The Reference Model Pattern.** A visualization manages visual models for one or more data sets, separating visual attributes (location, size, color, geometry, etc) from the abstract data. One or more views provide a graphical display of the visualization, while control modules process user input and may trigger updates at any level of the system.(courtesy of Heer and Agrawala [42])

The Reference Model Pattern resembles the standard Model–View-Controller (MVC)

pattern of user interface development. This design pattern supports the customization (D1) and interaction (D3) design requirements mentioned previously. Additionally, separating data and visual models and assigning clear roles to different system components is widely recognized as crucial in software architectural design. According to Heer and Agrawala [42], numerous software frameworks adopt and advocate for the reference model.

6.2 Implementation - Reference Model

This section delves into the software elements of the **Reference Model Pattern**, including the handling of the data source and dataset, the choice of software library for visualizing data, and the software library used for user manipulation and control of the view.

6.2.1 DataSource & Dataset

The DataSource component refers to the origin of raw EEG data. With the data preprocessing structure in place, the DataSource sends usable data to the DataSet component by storing data in compact Parquet files, a columnar storage file format, in the same directory.

The DataSet component reads one or more data files or connects to a database to load data for visualizations. This distinction allows multiple visualizations on the same dataset, enabling the Multiple Coordinated Views MCV overview discussed in Chapter 5. The DataSet object reads the relevant Parquet files based on user control inputs.

Other methods to implement the DataSet component include the DuckDB framework introduced by M. Raasveldt and H. Mühleisen [43]. DuckDB is a data management system designed to perform SQL queries, improving data loading performance. Although this thesis focuses on preprocessing, visualization design, and implementation, and does not incorporate DuckDB, the Reference Model pattern's decoupling of components allows for future replacement of the current DataSet loader with alternatives like DuckDB while maintaining the same visualization functionality.

6.2.2 Visualization & View

This section discusses the tools used for creating visualization objects and views. Currently, EEG practitioners use the MNE software for generating visuals (Chapter 3). While MNE provides preprocessing methods, data analysis functionality, and static visuals, it lacks advanced interactive features.

A widely recognized grammar for interactive visualizations is Vega-Lite [18], a declarative visualization grammar. Visualizations are specified using JSON data that follows the Vega-Lite JSON schema. The JSON format specifies what the visualization should

achieve but not how to compute it.

The strengths of Vega-Lite are its composability (great for MCV layout) and interactivity (D3). It supports the composition of multiple views and layers, enabling complex interactive graphics to be built from simpler components. Users can overlay points on bars, combine multiple plots into a dashboard, or create multi-view displays. These views can incorporate a rich set of interaction techniques, including basic ones like tooltips, zooming, panning, and more advanced techniques like brushing and linking, with simple declarative commands. It renders visualizations using Scalable Vector Graphics (SVG) or canvas, enabling web-based interaction.

To bridge the gap between the Vega-Lite framework and Python, the visualization library Vega-Altair [17] is utilized. Vega-Altair is a declarative visualization tool based on the Vega-Lite visualization grammar. Its Python Application Programming Interface (API) generates Vega-Lite JSON specifications, which can then be rendered in a user interface. It also allows saving the visual specification as HyperText Markup Language (HTML), providing further flexibility in loading the visualization object.

Figure 6.2: The software libraries utilized from data preprocessing to implementation of the visualization tool. From left to right: Python, MNE, Vega-lite, Vega-altair, Streamlit

6.2.3 Control

This section explores the software library used to showcase visualization objects and allow users to trigger updates at different levels of the system. Streamlit is an open-source, pure Python web framework that supports several important features for showcasing and interacting with data visualizations. It offers its own cloud server, making it easy to showcase the tool and is compatible with various cloud platforms, including Streamlit Sharing, Heroku, Google Cloud, and other popular services.

Streamlit supports various interactive widgets like sliders, buttons, radio buttons, and checkboxes, enabling users to control inputs dynamically. This aligns well with the design requirements of customizability (D1), flexibility (D2), and ease-of-use (D4), allowing users to control and filter multiple data frames and perform a wide range of tasks within the same tool. Streamlit is compatible with Vega-Altair objects and MCV files, supporting the rendering of visualizations.

Although many Python-supported web framework tools exist, such as Dash, Django, Web2py, and Flask, Streamlit is particularly suitable for several reasons:

- It is tailored for data science visualization tools and has extensive built-in support for displaying data visualizations with Altair.
- Adding a widget is as simple as declaring a variable, eliminating the need to write a backend, define routes, handle HTTP requests, connect a frontend, or write HTML, Cascading Style Sheets (CSS), and JavaScript. This simplifies the learning curve for data scientists experienced in data preprocessing, analysis, and visualization but less so in web development.
- It supports rapid prototyping since any changes to the script are immediately reflected in the tool, enhancing an experimental development process.

Chapter 7

Visualization Tool Review and Workflow

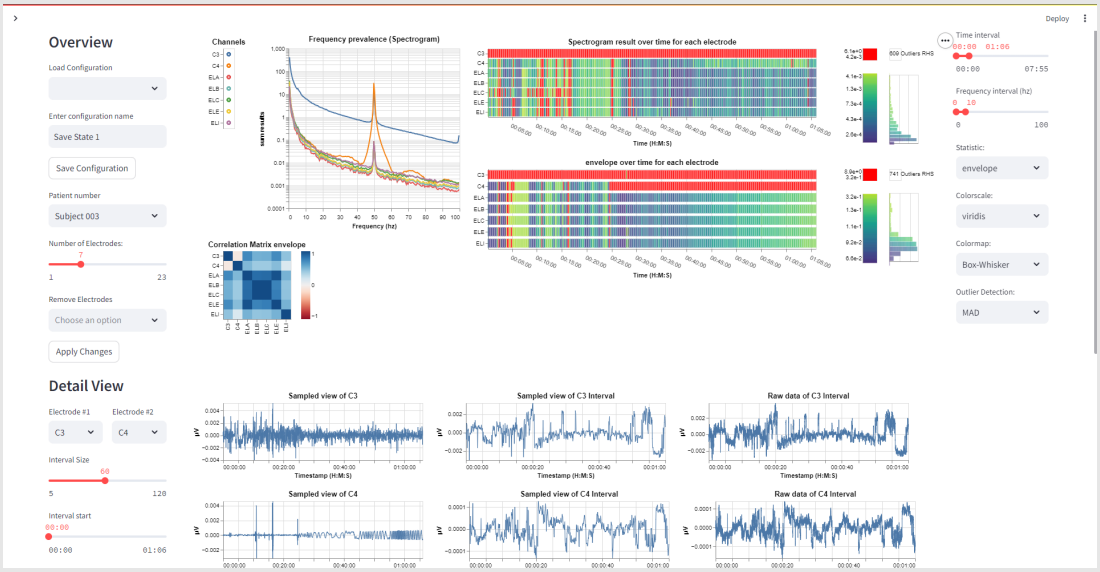


Figure 7.1: **The main user interface:** contains four sections. The far left and right contains customization options for all views through different widgets. The top middle contains the overview with 4 charts. The bottom middle contains the detail view with line charts for selected electrode pairs.

This chapter introduces the user interface of the finalized visualization tool by describing a realistic workflow during a session with the tool. It highlights the patterns and discoveries that can be uncovered with the tool, especially in cases where pure statistical data analysis might fail. Refer to the screenshots of the interface in Figures 7.1, 7.2, and 7.6. For more information on individual visualization, refer to different boxes,

like Figure 7.3, which provides details on the Tick Chart overview. Information for all components is detailed throughout this chapter. Revisiting the graph content in Chapter 4 also provides additional knowledge.

Check the YouTube Video explaining all relevant components of the visualization tool, including charts, selections, and interactions (Link: youtube.com/watch?v=q0UK1dZ_DFI). The video also describes a workflow for identifying notable patterns. The code base for the visualization tool can be found in the open-source GitHub repository (Link: github.com/Andersschultz12/Master-Thesis-Anders-Schultz).

7.1 General Workflow

This section highlights the properties of the visualization tool for different stages of the workflow, emphasizing its potential in EEG analysis. It proposes a workflow aimed at locating specific artifact regions through the lookup and comparison of electrodes. Refer to Figure 7.4 for a graphical representation of the workflow process.

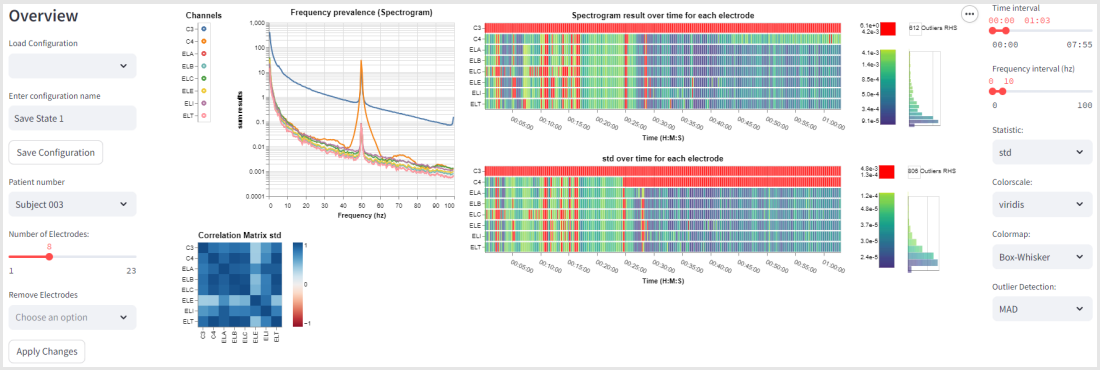


Figure 7.2: **Overview**.
Showcasing the full layout with four components: Top-left: Frequency Chart, Top-right: Spectrogram Tick Chart, Bottom-left: Correlation Matrix, and Bottom-right: Metric Tick Chart. A wide range of selection options, including save states, subject, electrodes, and chart customization like color coding, time interval, and frequency, are showcased on the far left and right.

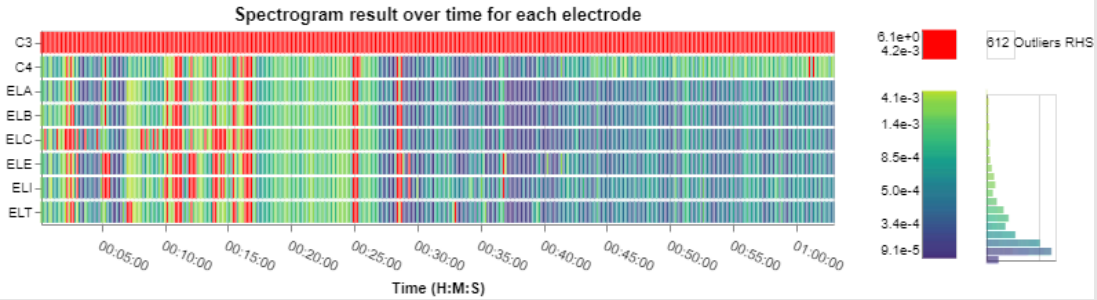


Figure 7.3: **Tick Chart (Spectrogram & Metrics)**.
This Tick Chart shows the FFT result (color) over time (x-axis) for each of the selected electrodes (y-axis). On the right is a color bar and a histogram showing the distribution of data values. Interactions include an x-axis brush selection on the tick chart and y-axis brush selection on the color bar, enabling effective cross-filtering. Interaction input shows changes in both overview tick charts simultaneously.

Overview Inspection (step 1)

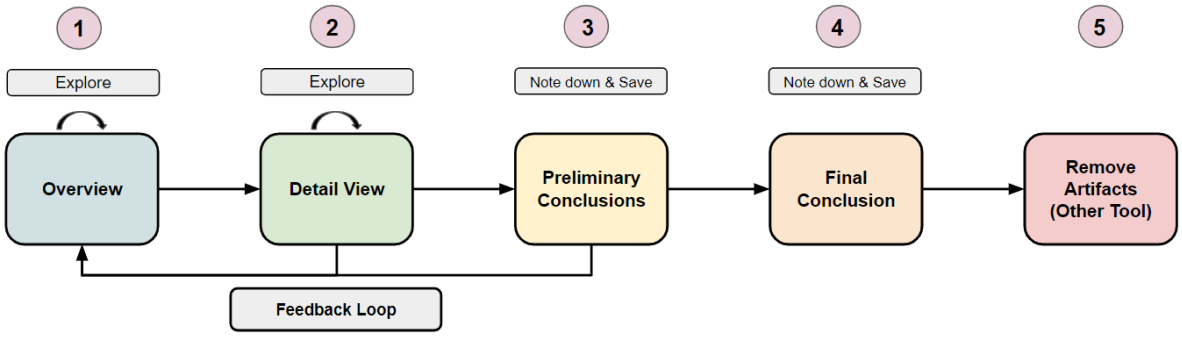


Figure 7.4: Graphical description of the workflow performed using this visualization tool

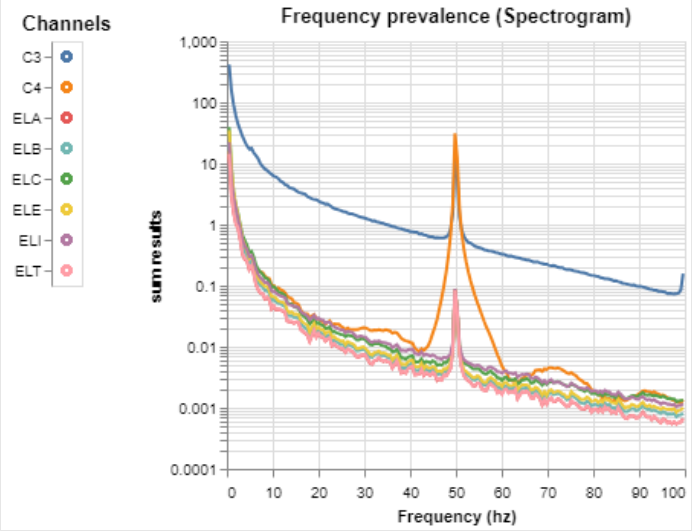
To follow along, refer to the full overview in Figure 7.2 and the individual charts in Figures 7.3, 7.5, 7.7. Initially, inspect the data for a given patient at a higher level of abstraction using the overview section with your preferred default settings as the starting point. The first goal is to locate candidate regions and electrodes for artifact removal later. A proposed task flow is:

1. Check the frequency overview 7.5 to quickly grasp the prominence of different frequencies for different electrodes. Additionally, scan the correlation matrix for electrodes that behave differently from others.

2. Adjust the frequency interval to a desired range, and begin inspecting the spectrogram and metric tick chart in Figure 7.3 for potentially interesting regions, with the primary goal of comparing electrodes in time and space.

3. The process of locating interesting regions can be done in multiple ways. Outlier detection highlights values that stand out from the original data distribution. Cross-selection with brush intervals on the color bar highlights common data values, and the x-axis brush can further highlight specific time regions.

Figure 7.5: **Frequency Line Chart**. Showcases relevant frequencies (x-axis) and the cum-sum of the FFT results given frequency over all timestamps (y-axis). Each electrode has a unique color (legend on the left). The legend allows point selection, highlighting electrodes in **all** overview charts.



Detail Inspection (step 2)

Follow along with Figure 7.6 and 7.9. Once you've located an interesting region warranting further exploration, continue to the detail view:

1. Select two electrodes worth comparing based on the discoveries in the overview section.

2. Dial in the important time range and starting position for the views.

3. Look further into the raw data by using the interactive brush between the second and third detail views. This is also showcased in the YouTube Video.

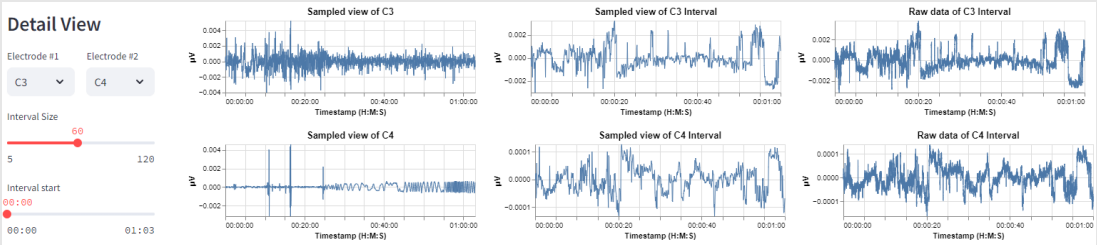


Figure 7.6: **Detail view**.
Showcasing the full layout with three line chart components for each selected electrode for comparison. The far left panel contains important selection options, including electrodes and the time region of interest.

Preliminary Conclusions (step 3)

Based on your observations, assess whether the identified region is an artifact by checking for patterns that violate known EEG signal characteristics. Depending on the importance of the observation, you can note it down in the notepad or save the state to showcase the finding later, allowing the practitioner to continue the workflow from this particular instance.

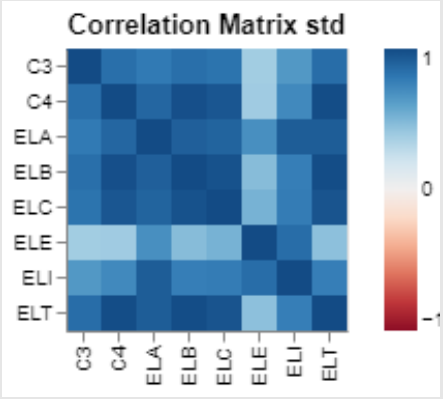


Figure 7.7: **Correlation Matrix**.
This shows the correlation between different pairs of electrodes with zero lag. The correlation is calculated at the selected time interval and the correlation matrix represents the same metric utilized in the Metric Tick Chart.

Repeating the Process (step 1-3)

The workflow can continue in various directions after the initial task flow:

1. Explore different regions showcasing similar behavior in the overview as the previous one.

2. Explore different statistical measures and frequencies, attempting to spot new patterns not noticeable by the original metrics.

3. Change the time interval to analyze new regions of the same session.

With each new discovery, save states can be created, images can be saved, and the text editor can gather more artifact information for the given subject.

Final Conclusions (step 4)

Once you have noted down everything important in the text editor, finalize the full conclusion about artifacts in the data. The notepad can then save this information as a .txt file, summarizing the analysis conducted on the data.

Artifact removal (step 5)

When all important artifacts are noted down, the workflow using this visualization tool is complete. EEG practitioners can proceed to remove the artifacts from the data and then gather important insights. The later stages after artifact removal involve data analysis of the patient in 30+ second intervals using machine learning models trained on EEG signals.

7.2 Useful patterns found

A small test session was conducted with EEG expert KM, testing both the workflow and the ability to identify interesting patterns in the data. This section dives into some noteworthy examples and additional exploration afterward.

Known test subject data was provided by KM. This EEG session had already been reviewed by KM and other EEG practitioners, allowing us to test if he could spot some of the known important patterns and artifacts during his workflow.

Figure 7.8: **Facts about the test data:**
In the evaluation of the visualization tool, a specific patient with well known data artifacts were used. That patient was analysed before by EEG expert KM. EEG measured the activity of this patient during sleep. Besides that, the data has the same BIDS format properties as the original data.

One of the known artifacts is related to two specific electrode pairs called EOGl and EOGr. Using the tick charts with the spectrogram and standard deviation metric, and brushing on spectrogram outlier values, a region of **combined outliers** at 7-8 minutes for that electrode pair is noticeable. Further exploration in the detail charts, as seen in Figure 7.9, reveals a peculiar signal pattern, marking it as a clear case of an artifact.

This confirms that the tool is capable of highlighting relevant regions through the main overview.

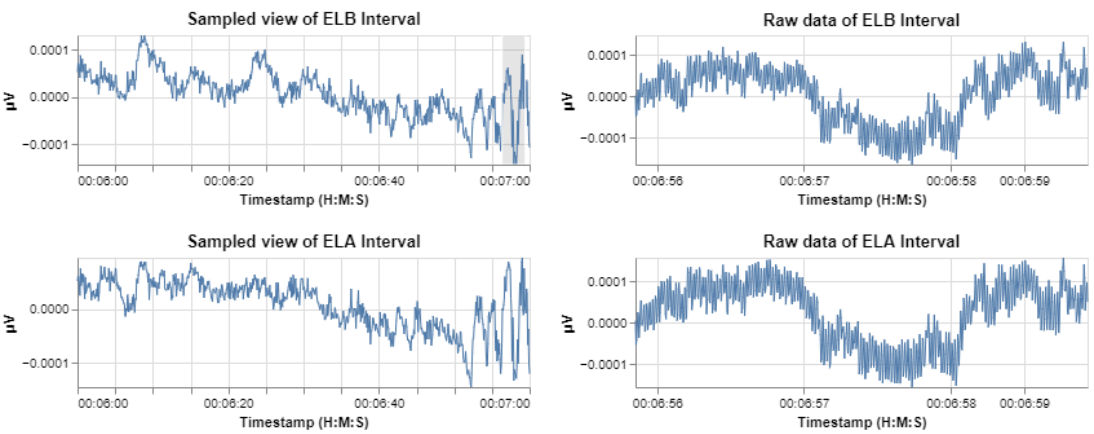


Figure 7.9: **Detail line charts**.
This figure showcases a section of the detail view. There are two line charts for each selected electrode: one with a sampled set of points (left) and the other with the entire raw data (right). It is possible to use an x-axis brush on the left chart to inspect the raw values further in **both** charts on the right, facilitating synchronized comparison.

Some broader patterns that can be immediately noticed using the tool include:

- **Global Artifact Region:** At around 10 minutes and 50 seconds, there is a region with outlier values in every channel for standard deviation over about 20 seconds, suggesting a global artifact. Detailed view inspection shows a significant spike in microvolts, indicating deviations from the norm. These time region artifacts are easily spotted from the colors in the main overviews.
- **ELI Electrode Outlier:** At 5 minutes and 5 seconds, the ELI electrode shows outlier behavior across multiple metrics. Further inspection reveals a more prominent spike compared to other electrodes. This allows checking if this is a unique case for ELI or if other electrodes have similar, less intense spikes.
- **Frequency Outliers in C3 and C4:** Electrodes C3 and C4 show significant outliers in the frequency overview at around 50 Hz. Further inspection in the tick charts, with the frequency interval set to include 50 Hz, reveals that C3 consistently shows outliers. From 25 minutes onwards, electrode C4 also shows consistent outliers in every envelope and standard deviation time period, indicating more intense deviations than normal.

Chapter 8

Discussion

In this chapter, we explore the scope of the contributions presented in the previous sections and their broader implications. Additionally, we analyze the results within the context of existing literature and address the research questions posed at the beginning. Finally, we consider the advantages and limitations of the visualization tool and propose potential avenues for future research.

8.1 Limitations of the tool

Previously, we introduced the general workflow involved in handling EEG data, from the moment a patient walks into the clinic to the final feedback based on data analysis. The scope of this visualization tool is to assist EEG practitioners with initial data preprocessing and the task of discovering EEG artifacts. This tool aims to simplify these aspects of the workflow by integrating methods, views, and customization options tailored to various task flows.

However, it is important to note that the tool does not assist with all aspects of the EEG data analysis pipeline:

1. **Data Collection and Raw Data Storage**: The tool does not support practitioners in collecting and storing the raw EEG data itself. This data must be gathered using other methods.

2. **Removing Artifacts from Data**: The tool can be utilized to locate and note down artifacts, but it cannot yet remove them by updating the EEG data accordingly.

3. **Post-Preprocessing Data Analysis**: The tool does not provide any diagnostic aids or assist in subsequent analyses, such as those involving machine learning models or other post-artifact processing goals.

8.2 Advantages of Utilizing The Tool

This tool offers several advantages that should make it more effective in assisting with preprocessing than conventional methods for EEG practitioners. This however, still needs to be tested in a real life scenario. To better understand these benefits, let's revisit the traditional process of analyzing EEG data for artifact detection, described in Figure 8.1:

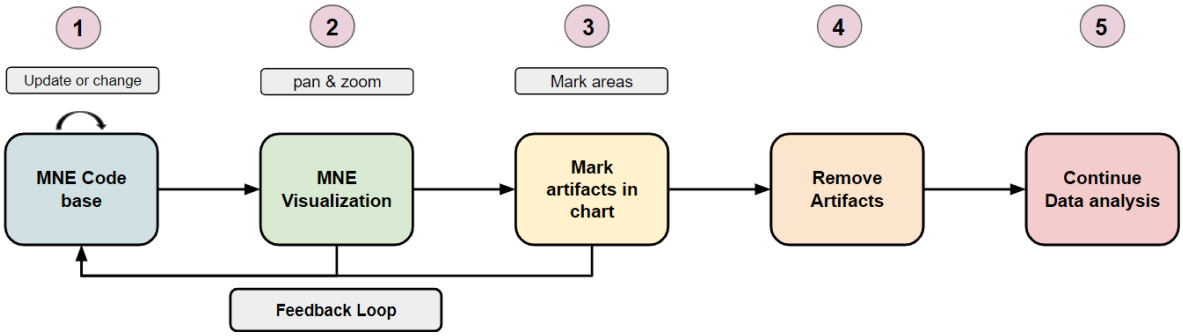


Figure 8.1: Simplified graphical description of a typical workflow performed by EEG experts currently

- **(1)** Practitioners perform necessary band-pass filtering, sampling, and additional preprocessing.
- **(2)** Visualizing EEG data using conventional vertical line charts provided by the MNE system.
- **(3)** Exploring the visualization with pan and zoom, which can lead to marking artifacts.
- **(1-3)** Re-coding and generating new static visualizations based on observations. Repeating these steps until all reasonable checks have been completed and artifacts are marked.
- **(4)** Remove the marked artifacts from the data.
- **(5)** Continue with data analysis on corrected EEG data.

The purpose of this visualization tool is to eliminate some of the obstacles present in the traditional preprocessing method. Some of these advantages are (see design requirements in section 4.3):

1. The tool offers a high degree of **Customizability** (D1) and **Flexibility** (D2), facilitating easier exploration of the full dataset.

2. It is a more self-contained tool set (D4) than the current set of tools, consolidating the workflow into a single visualization tool without compromising data and visual quality. Tasks such as lookups, comparisons, overview explorations, detailed inspections, and saving parameters and notes are all embedded within the tool.

3. It features a wide range of interactive elements (D3), enabling functionalities not commonly used in the traditional workflow. For example, it employs methods like brushes for cross-filtering.

4. The tool allows users to save parameter configurations (D5) within the directory and reload visuals later for further analysis or secondary inspections of the same data.

8.3 Future Improvements

While the proposed solution offers many advantages, there are areas where enhancements can further improve the tool. Below are some potential improvements, not ranked in any specific order:

(1) Frequency interval selection: Currently frequencies are selected using fixed intervals. Vega-Altair could allow a frequency brush on the frequency chart to interact with the spectrogram Tick Chart, reducing time spent switching between charts and providing better data context.

(2) Multiple Brushes: Limiting data exploration to a single brush on a single chart restricts analysis. Although Vega documentation does not explicitly support multiple brushes, this functionality could be achieved through advanced MCV editing or using D3 JavaScript. [19].

(3) Storing parameters of a brush selection: A text editor for notes is useful but not very effective for storing information. Currently, users can't save state parameters from a selected brush region with a single click. Implementing this feature would significantly improve the fulfillment of the design requirement for saving information (D5), simplifying the process of marking artifact regions. Since brush interactions do not alter the underlying data in the Python environment using Vega-Altair, implementing this requires different libraries and is not straight forward given the scope of this project.

(4) Expanding with new visualization techniques: Although the current tick chart is effective, exploring various time series visualization techniques could enhance customization (D1) and analysis. Adding a toggle to switch between visualization techniques would improve exploration and analysis, particularly in ambiguous situations.

(5) Faster Automatic visualization updates: The visualization does not update immediately when new parameters are selected, partly due to processing and rendering times of at least five seconds. This impacts the ease-of-use requirement (D4). Utilizing

a database storage system and faster SQL-based data retrieval, such as DuckDB [43], could enable real-time updates and reduce waiting times.

(6) Incorporation of uncertainty parameters: Statistical analysis and metric calculations do not always convey the full story of the data. Incorporating methods to display uncertainty, such as confidence intervals, can help practitioners understand the reliability of specific values. Possible implementations include a two-dimensional color scale (representing value and confidence interval size) or a toggle to display uncertainty.

(7) Detail View Improvements: Currently, the detail view is limited to comparing two electrodes and specific interval sizes. Ideally, it should be possible to link brushes from the overview chart to the detail chart, provided the rendering can be optimized for different data sizes.

(8) Bugs and glitches As the visualization tool is still in early development, unknown bugs and glitches may exist. With additional time and effort, these issues can be resolved to ensure consistency and flexibility (D2) for various data types.

Most of the recommendations mentioned are fairly straight forward to implement, but due to the time restraints on the project, there was not enough time to implement every feature. Point number 2, 3 and 5 also require extra time to learn new libraries that enable such implementations.

8.4 Concluding End User Interview

At the conclusion of the project, EEG expert KM was introduced to the prototype visualization tool for testing, and a brief reflection on the realisation of the project design goals was conducted:

- Creating a tool with relevant customizability (D1) was successful, allowing data selections on important parameters. The custom feature to change subject data achieves the desired flexibility (D2) by enabling work on multiple subjects within a single session.

- The tool's ease of use (D4) was also appreciated, as it requires fewer tools to achieve the same levels of analysis.

- The interactive brush features (D3) enabling cross-filtering are unique and innovative for this application domain. The tool successfully introduced new cross-filtering methods, though there is room for expanding the associated features and functions. Interaction brushes for selecting desired frequencies are currently missing.

- A significant shortcoming is the inability to clearly mark areas of artifacts using a brush. This feature is crucial for a standard EEG workflow, and its absence means the requirement to save information (D5) was not fully satisfied.

8.5 Applications within other subject areas

The visualization tool developed for analyzing EEG time series data holds significant potential for broader applications beyond the realm of EEG.

This section explores the versatility of our tool by demonstrating its potential in other fields that extensively utilize large time series datasets. By showcasing these potential extensions, the robustness and adaptability of the tool is emphasized.

Healthcare Monitoring: Beyond EEG data, the healthcare sector continuously generates vast amounts of multivariate time series data through patient monitoring systems such as Multichannel ECG and MEG data. Since these methods resemble similar data structures and measurement procedures, it potentially leads to the same challenges in data preprocessing and analysis. Thus, this tool can also potentially identify artifacts in these datasets, but this needs to be investigated further to confirm the needs in these fields.

Finance and Stock Market Analysis: Multivariate Time series data is crucial for tracking stock prices, market indices, and economic indicators. Some features of the visualization tool, such as correlation analysis are already widely used. However, the Tick Charts could potentially allow multiple stocks to be compared at the same time with relevant metrics.

Environmental and Climate Data Analysis: Time series data is used to monitor changes in climate, weather patterns, and ecological systems. The tool can potentially be tailored to analyze large datasets from meteorological stations, satellite observations, and sensor networks. Detecting anomalies in this data might help predict weather events and understand climate change.

It's important to emphasize that more research is needed to understand if the methods discussed in this project are already utilized in the above mentioned fields.

Chapter 9

Conclusions and Future Work

The aim of this thesis two to develop and evaluate an interactive visualization tool aimed at enhancing the current workflow utilized by EEG practitioners. By establishing a robust data preprocessing pipeline, identifying design requirements, and implementing these through software development, the tool provides a solution for visually identifying and analyzing artifacts within EEG data.

The tool introduces features such as customizable selections, dynamic cross-filtering, detailed view comparisons, and the flexibility to perform various workflow tasks like comparison and lookup. These features address some of the limitations found in traditional EEG analysis methods. Although the tool has achieved many of its goals, there is still room for improvement, and future research can build upon this foundation.

9.1 Recommendations

Based on the conclusions drawn in the previous sections, here are some actionable recommendations to improve preprocessing, the current visualization design, and the software implementation:

1. **Implement a SQL-Based Database (e.g., DuckDB):** Improve data loading performance and overall system responsiveness. Adapt the current data handling to use DuckDB, leveraging its efficient data retrieval and management capabilities.

2. **Develop Standardized Preprocessing Protocols:** Ensure consistency and comparability of results by establishing and documenting a standardized preprocessing protocol that can be widely applied in the field.

3. **Real Life Usability Testing:** Engage with practitioners and experts in the field to integrate their insights and enhance the visualization tool's applicability across different disciplines.

4. **Address Limitations and Extend Research:** Overcome the limitations identified in this study by conducting further research to explore advanced techniques for data preprocessing and storing information associated with brush interactions.

5. **Expand the Visualization Tool:** Incorporate additional visualization techniques and uncertainty parameters to deepen the understanding and reliability of the data.

9.2 Concluding Reflections

This project presented many new and difficult challenges related to preprocessing, visualization design, and software implementation. Being introduced to the field of EEG and the assocaited data, I faced a steep learning curve in understanding the data structure, the field of EEG signal processing, the state-of-the-art practices for marking artifacts in the data and doing litterature research. This ultimately shaped the preprocessing structure, design requirements, and execution of the visualization software.

Additionally, I had no prior experience with the Vega-Lite grammar and the Vega-Altair library, which required me to read extensive documentation and test various visualization techniques.

Despite these challenges, this project has been an incredibly instructive and exciting journey. I have gained invaluable experience in working with a client, conducting questionnaires, creating design requirements, and implementing software capable of handling large time series data.

In hindsight, there are certainly things I could have done differently. However, this knowledge will stay with me for future projects as a data scientist. Despite the steep learning curve and the complexities involved, the thesis journey has been a pleasant experience, thanks to excellent guidance from my supervisor, Hans-Jörg Schulz, and co-supervisor, Kaare Mikkelsen.

List of Figures

Acronyms

API Application Programming Interface. 33

BIDS Brain Imaging Data Structure. 18

CSS Cascading Style Sheets. 34

DFD Data Flow Descriptors. 17
DFT Discrete Fourier Transform. 15
DSD Data Space Descriptors. 17

ECG Electrocardiogram. 6, 46
EEG Electroencephalography. 1, 3–5, 9, 11, 12, 14, 17–21, 23, 26, 28, 30, 32, 36, 39, 40, 42, 43, 45–47

FFT Fast Fourier Transform. 16, 19, 37, 38, 50
FT Fourier Transform. 12, 15

GUI Graphical User Interface. 20

HG Horizon Graph. 5
HJ Hans-Jörg Schulz. 4
HTML HyperText Markup Language. 33

IHG Interactive Horizon Graph. 6
IQR Interquartile Range. 28

KM Kaare Mikkelsen. 3, 4, 17, 19, 20, 30, 40, 51

MAD Median Absolute Deviation. 28
MCV Multiple Coordinated Views. 24–26, 32, 33, 44
ME Mean Error. 15
MEG Magnetoencephalography. 9, 46
MSE Mean Squared Error. 15

SVG Scalable Vector Graphics. 33

Glossary

Aggregation The process of producing a summary for different data intervals. 14
Amplitude The magnitude of change in a waveform or signal. 16
Artifact Unwanted interference in the EEG data from non-brain sources. 1, 5, 9, 12, 17, 19, 28, 30, 36, 40, 42
Average reference A common referencing technique in EEG where the average of all electrodes is used as a reference. 12

Band-Pass Filtering A technique used to filter out frequencies within a certain range. 11, 13, 18
Banking to 45 A graphical technique that adjusts the aspect ratio of line charts so that the average absolute slope is 45 degrees. 30
Brush Allows users to select and highlight specific areas of data on a graph.. 44

Canvas An HTML element used to draw graphics via scripting (usually JavaScript). 33
Color mapping The assignment of data values to colors in a visualization. 21, 27
Comparison Task A task involving the comparison of different data sets or variables. 25
Continuous Triangular Model A model showing continuous data variations over time in a triangular form. 8, 26
Correlation Matrix A table showing correlation coefficients between variables. ii, 29, 36, 39, 50
Cross-correlation A measure of similarity between two signals as a function of the time-lag applied to one of them. 29
Cross-filtering A technique that allows for the filtering of data based on selections in multiple visualizations. 3, 26, 37, 45, 50

D3 JavaScript A JavaScript library for producing dynamic, interactive data visualizations. 44
Data Preprocessing The process of cleaning and transforming raw data before analysis. 1, 12, 25, 32, 42
Declarative visualization A method of creating visualizations by specifying what the visualization should show rather than how to compute it. 32, 33
Dynamic Interaction Real-time manipulation and exploration of data visualizations. 3

EEGLAB An interactive Matlab toolbox for processing EEG data. 3
Electrode A conductor through which electricity enters or leaves. 1, 5, 11, 14, 17, 21, 36, 45
Electroencephalography A method to record electrical activity of the brain. 1
Elementary Task A basic task that is a fundamental building block for more complex tasks. 19
Envelope The boundary that defines the outer limits of a signal's amplitude. 15, 41

Fast Fourier Transform An efficient algorithm to compute the Discrete Fourier Transform. 16

Focus and Context A visualization technique that highlights important details while showing context around it. 23

Fourier Transform A mathematical transform that converts a time-domain signal into its frequency components. 15

Gaussian Kernel A function used to smooth data, emphasizing central values. 6

Heatmap A multivariate data visualization technique showing data values represented as colors. text. 6, 15, 26

Horizon Graph A compact visualization technique for time-series data. 6, 26, 49

Informal Interview Unstructured interviews conducted to gather qualitative information. i, 3, 17

Long data format A data format where each row is a single observation, useful for time-series data. 12, 13, 16

Lookup Task A task involving finding specific data points or values within a data set. 19

Missingness Map A visualization showing the locations and patterns of missing data. 13

MNE Python A software package for processing and analyzing EEG/MEG data. 3

Multiple Coordinated Views A visualization technique that uses multiple views to explore data from different perspectives. 24, 32

Noise Unwanted variations in the data that obscure the true signal. 2, 11, 17

Overview and Detail A visualization technique that provides overview and detailed views separated. 23

Power The rate at which energy is transferred or converted. 16

Preprocessing Pipeline A sequence of steps used to clean and prepare data before analysis. i, 3, 11, 47

Reference Model Pattern A software architecture that resembles the Model-View-Controller design pattern. 31, 50

Sign Change A point where the value of a signal changes from positive to negative or vice versa. 14

Spectrogram A visual representation of the spectrum of frequencies in a signal over time. 14, 15, 30, 38

Stacked Area Chart A chart displaying the total and individual values of multiple time series over time. 26

Synoptic Task A task that provides an overview or summary of data. 19

Task Abstraction The process of deriving the important tasks present the workflow of the client. 19

Tick Chart A Heatmap variation with data points as a vertical line instead of squares. 26, 27, 36, 37, 44, 50

Time Series A sequence of data points typically measured at successive times. 2, 16, 18, 44

Visualization Pipeline The process flow of data through various stages of visualization. 10

Wide data format A data format where each row represents a single entity with multiple observations across columns. 13

Bibliography

[1] S. Havre, E. Hetzler, P. Whitney and L. Nowell: *"ThemeRiver: visualizing thematic changes in large document collections"* Vol. 8 Issue 1. (2002) [URL] DOI:10.1109/2945.981848

[2] L. Byron and M. Wattenberg: *"Stacked Graphs – Geometry & Aesthetics"* Vol. 14 Issue 6 (2008) [URL] DOI: 10.1109/TVCG.2008.166

[3] C. Bu, Q. Zhang, Q. Wang, J. Zhang, M. Sedlmair, O. Deussen and Y. Wang *"SineStream: Improving the Readability of Streamgraphs by Minimizing Sine Illusion Effects"* Vol 27. Issue 2 (2020) [URL] DOI: 10.1109/TVCG.2020.3030404

[4] T. Saito, H.N. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya and T. Kaseda: *"Two-tone pseudo coloring: compact visualization for one-dimensional data"* (2005) [URL]

[5] J. Heer, N. Kong, and M Agrawala: *"Sizing the Horizon: The Effects of Chart Size and Layering on the Graphical Perception of Time Series Visualizations"* (2009) [URL]

[6] C. Perin, F. Vernier and J.-D. Fekete: *Interactive horizon graphs: improving the compact visualization of multiple time series* [URL] DOI:10.1145/2470654.2466441

[7] M. Dahnert, A. Rind, W. Aigner and J. Kehrer *Looking beyond the horizon: Evaluation of four compact visualization techniques for time series in a spatial context* [URL]

[8] J. D. Pleil, M. A. Stiegel, M. C. Madden and J. R. Sobus: *"Heat map visualization of complex environmental and biomarker measurements"* [URL]

[9] H. Guo, W. Zhang, C. Ni, Z. Cai, S. Chen and X. Huang: *Heat map visualization for electrocardiogram data analysis* Vol. 20 Art. 277 (2020) [URL] DOI:10.1186/s12872-020-01560-8

[10] J. Burris, K. Ballard, S. R. Wilson and D. J. Edwards: *"Visualization and comparison of aircraft trajectories using Gaussian-Smoothed heatmaps"* (2021) [URL] DOI:10.1080/08982112.2021.1976795

[11] M. Sips, P. Köthur, A. Unger, H-C. Hege and D. Dransch: *"A Visual Analytics Approach to Multiscale Exploration of Environmental Time Series"* Vol. 18, Issue 12 (2012) [URL]

[12] Y. Qiang, S. H. Chavoshi, S. Logghe, P. De Maeyer and N. Van de Weghe: *"Multi-scale analysis of linear data in a two-dimensional space"* Vol. 13 Issue 3 (2013) [URL]

[13] V. Mikheev, R. Skukies and B. V. Ehinger: *"The Art of Brainwaves: A Survey on Event-Related Potential Visualization Practices"* (2024) [URL]

[14] J. R. Iversen and S. Makeig: *MEG/EEG Data Analysis Using EEGLAB* (2019) [URL] DOI:10.1007/978-3-319-62657-4_8-1

[15] R. Oostenveld, P. Fries, E. Maris and J-M. Schoffelen: *FieldTrip: open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data* (2011) [URL] DOI:10.1155/2011/156869

[16] A. Gramfort, M. Luessi, E. Larson, D. A. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, M. Hämäläinen: *"MEG and EEG data analysis with MNE-Python"* (2013) Vol. 7 [URL] DOI: 10.3389/fnins.2013.00267

[17] J. VanderPlas, B. E. Granger, J. Heer, D. Moritz, K. Wongsuphasawat, A. Satyanarayan, E. Lees, I. Timofeev, B. Welsh and S. Sievert: *"Altair: Interactive Statistical Visualizations for Python"* (2018) [URL]

[18] A. Satyanarayan, D. Moritz, K. Wongsuphasawat and J. Heer: *Vega-Lite: A Grammar of Interactive Graphics* (2017) [URL] DOI: 10.1109/TVCG.2016.2599030

[19] M. Bostock, V. Ogievetsky and J. Heer: *D^3 Data-Driven Documents* (2011) Vol. 17 Issue 12. [URL] DOI:10.1109/TVCG.2011.185

[20] S. dos Santos and K. Brodlie: *"Gaining understanding of multivariate and multidimensional data through visualization"* Vol. 28 Issue 3. (2004) [URL] DOI:10.1016/j.cag.2004.03.013

[21] R. B. Haber and D. A. McNabb: *Visualization ldioms: A Conceptual Model for Scientilic Visualization System* (1990) [URL]

[22] S. K. Card, J. D. Mackinlay and B. Shneiderman: *"Readings in Information Visualization: Using Vision To Think"* (1999) [URL]

[23] A. Pedroni, A. Bahreini and N. Langer: *"Automagic: Standardized preprocessing of big EEG data"* Vol. 200 Issue 15 (2019) [URL]

[24] H-J. Schulz, T. Nocke, M. Heitzler, and H. Schumann: *A systematic view on data descriptors for the visual analysis of tabular data* Vol. 16 (2017) [URL] DOI:10.1177/1473871616667767

[25] J. C. Roberts, C. J. Headleand, P. D. Ritsos *"Five Design-Sheets: Creative Design and Sketching for Computing and Visualisation"* (2017) [URL] DOI:10.1007/978-3-319-55627-7

[26] G. Adrienko and N. Adrienko: *Exploratory Analysis of Spatial and Temporal Data* (2006) [URL]

[27] J. Madsen, L. Sode, J. Frost Dahl, M. Corredig and H-J. Schulz: *"Visual Exploration of Rheological Test Results from Soft Materials"* [URL] DOI:10.1109/TestVis57757.2022.00006

[28] M. Q. W. Baldonado, A. Woodruff and A. Kuchinsky: *"Guidelines for using multiple views in information visualization"* (2000) [URL] DOI:10.1145/345513.345271

[29] X. Cheng, D. Cook and H. Hofmann: *"Visually Exploring Missing Values in Multivariable Data Using a Graphical User Interface"* Vol. 68 (2015) [URL]

[30] W.T. Cochran, J.W. Cooley, D.L. Favin, H.D. Helms, R.A. Kaenel, W.W. Lang, G.C. Maling, D.E. Nelson, C.M. Rader and P.D. Welch: *"What is the fast Fourier transform?"* Vol. 55 Issue 10. (1967) [URL] DOI: 10.1109/PROC.1967.5957

[31] Ben Shneiderman: *"The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations"* 1996 [URL]

[32] C. Tominski, G. Fuchs, and H. Schumann: *"Task-Driven Color Coding"* (2008) [URL]

[33] F. Crameri, G. E. Shephard and P. J. Heron: *"The misuse of colour in science communication"* (2020) [URL] DOI:10.1038/s41467-020-19160-7

[34] K. Moreland: *"Why We Use Bad Color Maps and What You Can Do About It"* (2016) [URL] DOI:10.2352/ISSN.2470-1173.2016.16.HVEI-133

[35] Raja Mubashar Karim, Oh-Hyun Kwon, Chanhee Park and Kyungwon Lee: *"A Study of Colormaps in Network Visualization"* (2019) [URL]

[36] J. R. Nuñez, C. R. Anderton and Ryan. S. Renslow: *"Optimizing colormaps with consideration for color vision deficiency to enable accurate interpretation of scientific data"* (2018) [URL]

[37] W. Aigner, S. Miksch, H. Schumann and C. Tominski: *"Visualization of Time-Oriented Data"* Second Edition (2023) [URL]

[38] C. Leys, C. Ley, O. Klein, P. Bernard and L. Licata *"Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median"* (2013) [URL]

[39] Songwon Seo: *"A Review and Comparison of Methods for Detecting Outliers in Uni variate Data Sets (Master thesis)"* (2002) [URL]

[40] T. M. Spalek and S. Hammad: *"The Left-to-Right Bias in Inhibition of Return Is Due to the Direction of Reading"* [URL]

[41] P. Köthur, C. Witt, M. Sips, N. Marwan, S. Schinkel and D. Dransch: *"Visual Analytics for Correlation-Based Comparison of Time Series Ensembles"* Vol. 34 Issue 3 (2015) [URL] DOI:10.1111/cgf.12653

[42] J. Heer and M. Agrawala: *"Software Design Patterns for Information Visualization"* Vol. 12 Issue 5. (2006) [URL] DOI: 10.1109/TVCG.2006.178

[43] M. Raasveldt, H. Mühleisen: *"DuckDB: an Embeddable Analytical Database"* (2019) [URL]

[44] J. A. Urigüen and B. Garcia-Zapirain: *"EEG artifact removal—state-of-the-art and guidelines"* Vol. 12 number 3 (2015) [URL] DOI 10.1088/1741-2560/12/3/031001

[45] S. D. Muthukumaraswamy: *"High-frequency brain activity and muscle artifacts in MEG/EEG: a review and recommendations"* Vol. 7 (2013) [URL] DOI:10.3389/fnhum.2013.00138

[46] W. S. Cleveland, M. E. McGill and R. McGill: *The Shape Parameter of a Two-Variable Graph* [URL]