

**A Look Into Human Brain Activity with EEG Data  
Surface Reconstruction**

by

Naveen Pothayath

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Data Science Program

by


---

April 2018

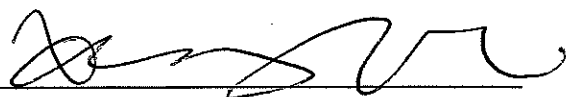
APPROVED:



Professor Fatemeh Emdad, Thesis Advisor



Professor Elke Rundenstiener, Thesis Advisor



Professor Xiangnan Kong, Thesis Committee Member

## Abstract

EEG has been used to explore the electrical activity of the brain for many decades. During that time, different components of the EEG signal have been isolated, characterized, and associated with a variety of brain activities. However, no widely accepted model characterizing the spatio-temporal structure of the full-brain EEG signal exists to date.

Modeling the spatio-temporal nature of the EEG signal is a daunting task. The spatial component of EEG is defined by the locations of recording electrodes (ranging between 2 to 256 in number) placed on the scalp, while its temporal component is defined by the electrical potentials the electrodes detect. The EEG signal is generated by the composite electrical activity of large neuron assemblies in the brain. These neuronal units often perform independent tasks, giving the EEG signal a highly dynamic and non-linear character. These characteristics make the raw EEG signal challenging to work with. Thus, most research focuses on extracting and isolating targeted spatial and temporal components of interest. While component isolation strategies like independent component analysis are useful, their effectiveness is limited by noise contamination and poor reproducibility. These drawbacks to feature extraction could be improved significantly if they were informed by a global spatio-temporal model of EEG data.

The aim of this thesis is to introduce a novel data-surface reconstruction (DSR) technique for EEG which can model the integrated spatio-temporal structure of EEG data. To produce physically intuitive results, we utilize a hyper-coordinate transformation which integrates both spatial and temporal information of the EEG signal into a unified coordinate system. We then apply a non-uniform rational B spline (NURBS) fitting technique which minimizes the point distance from the computed surface to each element of the transformed data. To validate the effectiveness of this proposed method, we conduct an evaluation using a 5-state classification problem; with 1 baseline and 4 meditation states comparing the classification accuracy using the raw EEG data versus the surface reconstructed data in the broadband range and the alpha, beta, delta, gamma and higher gamma frequencies. Results demonstrate that the fitted data consistently outperforms the raw data in the broadband spectrum and all frequency spectrums.

## Acknowledgements

I would like to thank Professor Fatemeh Emdad and Professor Elke Rundensteiner, my thesis advisors for their valuable contribution throughout the journey of this thesis. They have always been open for discussion, a source of motivation and provided me with guidance and tremendous support in the whole process. I would also like to thank Dr. Prasanta Pal, my thesis advisor from UMass Medical School, for being a part of this whole journey and providing me the resources and constant support, sharing some of his immense knowledge with me.

I would thank WPI for providing me this wonderful platform and motivating me to further my knowledge of Data Science.

In addition, I would like to express sincere gratitude to UMass Medical School for giving us all the resources and data at their disposal to complete this thesis. Dr. Judson Brewer has been an absolute inspiration in the field of neuroscience and his valuable contribution, analysis of results and validation of techniques in this thesis is greatly appreciated.

I would also like to thank my batch-mates Abhishek Easwaran, Praneeth Nooli and Karan Somaiah for their help in this thesis.

Lastly, I would like to thank my parents and my sister for their motivation and valuable support required throughout.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	What is EEG? . . . . .	1
1.2	Current Challenges . . . . .	2
1.3	Background . . . . .	3
1.4	Problem Statement . . . . .	4
1.5	Methodology . . . . .	4
1.6	Data . . . . .	5
1.7	Thesis Outline . . . . .	6
<b>2</b>	<b>Approach</b>	<b>7</b>
2.1	Median Absolute Deviation (MAD) Normalization . . . . .	7
2.1.1	Background . . . . .	7
2.1.2	MAD Normalization . . . . .	9
2.2	Outlier Removal using Nearest Neighborhood Interpolation (NNI) . .	12
2.2.1	Background . . . . .	12
2.2.2	Nearest Neighborhood Interpolation using Delaunay’s Triangulation . . . . .	12
2.2.3	Impact of NNI . . . . .	14
2.3	Spatio-Temporal Transformation (STT) . . . . .	14
2.3.1	Background . . . . .	14
2.3.2	Spatio-Temporal transformation . . . . .	14
2.3.3	Impact of STT . . . . .	17
2.4	Data Surface Reconstruction (DSR) . . . . .	18
2.4.1	Background . . . . .	18
2.4.2	DSR using Non-uniform Rational B-Spline (NURBS) . . . . .	18
2.4.3	Implementation . . . . .	22

2.4.4	Reconstructed surface using DSR . . . . .	26
2.4.5	Impact of DSR . . . . .	26
<b>3</b>	<b>Results</b>	<b>27</b>
3.1	Validation . . . . .	27
3.1.1	The Classification Problem . . . . .	31
3.1.2	Salt and Pepper Noise . . . . .	31
3.1.3	Data Transformation . . . . .	32
3.1.4	Fast Fourier Transform . . . . .	32
3.1.5	Nearest Centroid Classifier . . . . .	33
3.1.6	Salt and Pepper Noise . . . . .	35
3.2	Results of the GSTM . . . . .	35
3.2.1	Classification Scores With No Noise . . . . .	35
3.2.2	Classification Scores With 5 Percent Salt And Pepper Noise . . . . .	36
3.2.3	Image reconstruction with GSTM . . . . .	37
3.2.4	Impact . . . . .	38
<b>4</b>	<b>Discussion</b>	<b>40</b>
<b>5</b>	<b>Conclusion</b>	<b>41</b>
5.1	Summary . . . . .	41
5.2	Contribution . . . . .	41
5.3	Future Scope . . . . .	42

# List of Figures

1.1	EEG Data collection and recording procedure . . . . .	2
1.2	Biosemi cap . . . . .	5
2.1	GSTM Methodology . . . . .	8
2.2	Mean vs MAD Comparison . . . . .	11
2.3	Voronoi Triangulation . . . . .	13
2.4	Nearest Neighbor Interpolation . . . . .	13
2.5	Illustration of Spherical and Cartesian co-ordinate system . . . . .	15
2.6	Spatio-Temporal transformation . . . . .	16
2.7	Data before temporal transformation . . . . .	17
2.8	Data after temporal transformation . . . . .	17
2.9	B-spline basis function . . . . .	19
2.10	Curve Fitting using B-spline basis function . . . . .	21
2.11	Data Surface Reconstruction Example . . . . .	25
3.1	Raw data before Data Surface Reconstruction . . . . .	28
3.2	Fitted Surface after Data Surface Reconstruction . . . . .	29
3.3	Fitted Data Points on the surface . . . . .	30
3.4	State classification accuracy using nearest centroid classifier . . . . .	36
3.5	State Classification Accuracy with Noise using nearest centroid classifier	37
3.6	Reconstruction of Phantom Image . . . . .	38

# Chapter 1

## Introduction

### 1.1 What is EEG?

The electroencephalogram (EEG) is a recording of the electrical activity of the brain from the scalp. The waveforms recorded are thought to reflect the activity of the surface of the brain, the cortex. This activity is influenced by the electrical activity from the brain structures underneath the cortex.

EEG has been used to explore the electrical activity of the brain for over eight decades. During that time, many different components of the non-linear EEG signal have been isolated, characterized, and shown to associate with a variety of brain activities.

As seen in Figure 1.1, the process of collecting EEG data starts by placing small metal discs called electrodes on the scalp in special positions. These positions are identified by the recorder who measures the head using the International 10-20 System[1]. This relies on taking measurements between certain fixed points on the head. The electrodes are then placed at points that are 10 percent and 20 percent of these distances.

A digital EEG system converts the waveform into a series of numerical values. This process is known as Analogue-to-Digital conversion (ADC).

The values can be stored in the computer memory, manipulated and then re-displayed as waveforms on a computer screen. The rate at which the waveform data is sampled in order to convert it into a numerical format is known as the sampling rate.

The sampling rate is usually expressed in Hz, for example 240 Hz is 240 times

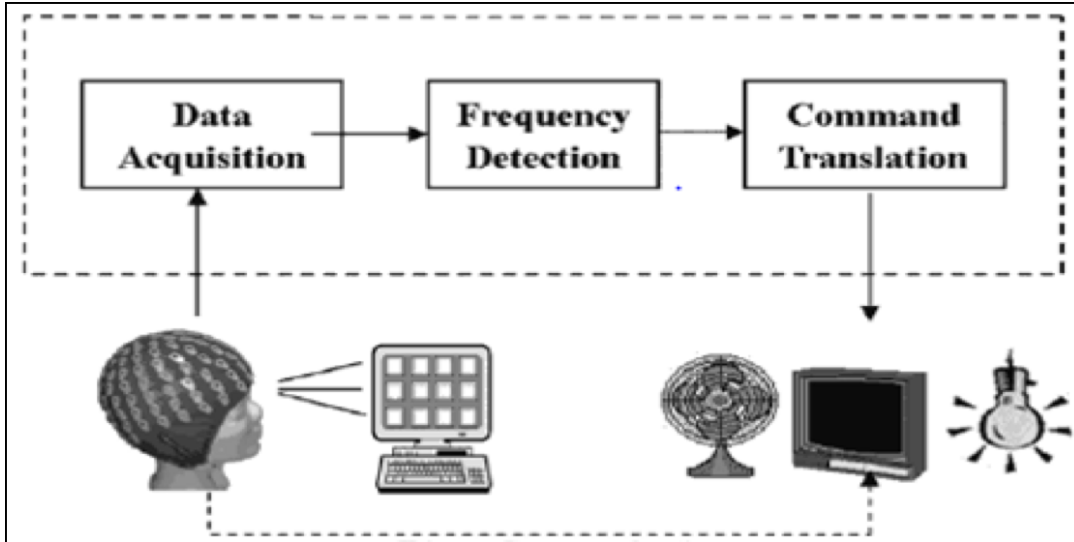


Figure 1.1: EEG Data collection and recording procedure

per second. The sampling rate used in our dataset is 2048 Hz, that is, for every second of data, 2048 rows of data are recorded.

The waveforms recorded by EEG data provide a blend of both spatial and temporal information. Spatially, EEG is quite simply defined by the locations of recording electrodes (ranging between 2 to 256 in number) placed on the scalp. Temporally, EEG is much more complicated and defined by the complex, non-linear EEG signal itself. The measured EEG signal is a composite of electrical signals generated by the activity of large assemblies of neurons. These neuronal units display diverse, independent, and rapidly changing activity patterns which make the EEG signal highly dynamic and non-linear in nature. Before reaching the scalp electrodes, these neuronal signals are spatially smeared, and highly attenuated while propagating through convoluted layers of biological tissue.

## 1.2 Current Challenges

As of yet, there is no widely accepted model characterizing the spatio-temporal structure of the full-brain EEG signal.

Instead of trying to model the complex spatio-temporal nature of the overall EEG signal, most EEG research avoids it by partially extracting targeted spatial and temporal components of interest [2]. While this strategy is useful for exploring region and frequency specific activity of the brain, it is still subject to significant noise



contamination, and often produces outcomes that are unreliable and challenging to reproduce [3]. These feature extraction drawbacks could be improved significantly if informed by a global spatio-temporal model of the EEG data

## 1.3 Background

The analysis of EEG has been taking place since decades. It is a widely studied field with specific concentration in certain areas. There are plenty of tools at disposal for the study of EEG. A common trait exists, however. Tools like MNE, EEGLAB, FASTER leverage either spatial or temporal aspects of EEG, however, none of them consider both. For example, a tool might provide statistical inference based analysis for temporal data and linear interpolation for spatial data. Also, these tools explicitly consider the sensors to be discrete points rather than something that arises out of a continuous surface, which the human brain is.

Magnetoencephalography, and electroencephalography (MNE)[4] is a very popular open-source software for exploring, visualizing, and analyzing human neurophysiological data. It is a collaborative effort of multiple institutes striving to implement, share best methods, and to facilitate distribution of analysis pipelines to advance reproducibility of research. MNE provides two complementary signal-processing approaches: signal space projection (SSP), and independent component analysis (ICA). The idea behind SSP is to estimate an interference subspace, and to use a linear projection operator which is applied to the sensor data to remove the interference from the data. The underlying assumption of SSP is that the noise subspace is orthogonal or at least sufficiently different from the signal subspace of interest to avoid signal loss in the projection. However, this assumption does not hold true in EEG data analysis as the signal subspace and noise subspace may overlap in real life scenarios. On the other hand, ICA is used to removing endogenous (biological) artifacts but it does not account for the exogenous artifacts.

The approach used in EEGLAB[5] for artifact rejection is to use 'statistical' thresholding that 'suggests' epochs to reject from analysis. An important aspect of EEGLAB is visual inspection. To reject artifacts in temporal data, EEGLAB suggests two methods. First, they propose rejecting data by visual inspection using their tool. Second, they use the data of a particular sensor and apply channel statistics to decide whether to reject a sensor or not.

FASTER[6] is another tool that performs linear interpolation of four nearby sensors in case they find an outlier in a particular sensor. This tool does not consider temporal information in its analysis.

Spline based interpolation[7] to replace fault values are used widely. They precisely locate sensor value extrema and converge faster towards the true potential of the sensor surface as compared to other classical methods of interpolation which are based on linear combination. However, these advantages come at an expense of lengthier computation time. The author in [7] attempted to find a single approach that would work for all cases but failed to do so.

In short, methods have been developed for removing various types of artifacts from EEG. However, as many are application dependent, or focused on a single type of artifact, it can be difficult to choose which approach(es) to take. Furthermore, most approaches involve at least some degree of supervision for classification of artifacts and leverage spatial or temporal EEG data to perform their respective analysis, but not an amalgamation of both.

## 1.4 Problem Statement

The aim of this thesis is to demonstrate a surface fitting technique for EEG which can be used to model and capture the overall spatio-temporal structure of EEG data as well as be resilient to outliers.

## 1.5 Methodology

In order to produce efficient and intuitive results, we use a four-step approach:

- We apply median absolute deviation (MAD) - a robust normalization technique that is more resilient to outliers than standard deviation - on the raw EEG data.
- We utilize a novel hypercoordinate transformation which integrates both - the spatial and temporal information of the EEG signal - into a unified coordinate system.
- We fit the transformed data with a non-uniform rational B-spline[8] [9] (NURBS) to perform spatio-temporal smoothing[9].

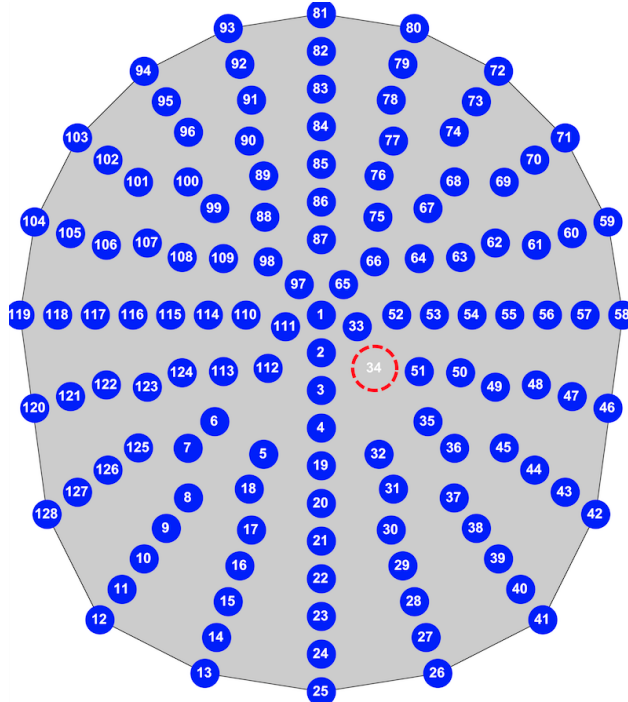


Figure 1.2: 2 dimensional representation of 128 sensors in a biosemi cap.

- We validate the fitting technique by comparing the classification results of 5 stage activity (baseline state and four meditative states) datasets of the raw and fitted data[10].

## 1.6 Data

The dataset used for testing was collected from 30 subjects using a 128 sensor Biosemi Active-two[1] device at a sampling rate of 2048 Hz as seen from Figure 1.2. Each EEG recording session consisted of a single baseline task followed by a 4-stage meditation task [11]. As reported by the subjects, each of these 4 meditation states were experientially distinct from each other.

The dataset contains 128 columns and based on the amount of time that the information is captured, the number of rows varies.

The data is labelled at the time of collection but not explicitly added to the dataset. Subjects are asked to perform various activities during the time of data collection, which is generally for an hour, and the activity performed during a time state is considered to be the label of those rows of data.

## 1.7 Thesis Outline

The outline of the thesis is illustrated in this section. Chapter 2 goes over the approach and implementation details of solving the problem. Section 2.1 details the median absolute deviation used to preprocess the data while the section 2.2 shows the interpolation done using nearest neighbors. Section 2.3 explains the spatio-temporal transformation done on EEG while section 2.4 shows the actual reconstruction of EEG after applying the methods above. Chapter 3 explains the validation methods used on the methodology and also explains the results obtained by applying Machine Learning techniques on the data. Lastly, Chapter 4 discusses the future work and we describe the conclusion in Chapter 5.

# Chapter 2

## Approach

This chapter explains the approach followed in the thesis to achieve the goal of creating a Global Spatio-Temporal Model (GSTM) using Data Surface Reconstruction (DSR) and the validation techniques used to test the effectiveness of the same.

As it can be seen from pipeline in Figure 2.1, main tasks in the process are:

- Median Absolute Deviation (MAD) Normalization
- Outlier Removal using NNI (Nearest Neighbor Interpolation)
- Spatio-temporal Transformation (STT)
- Data Surface Reconstruction (DSR)
- Global Spatio-temporal Model(GSTM)
- Non-uniform Rational B-Spline (NURBS) Fitting

The algorithms required to realize these tasks will be detailed in the following sections.

### 2.1 Median Absolute Deviation (MAD) Normalization

#### 2.1.1 Background

A lot of practitioners even today use an interval window of three standard deviations as a common practice to detect outliers[12]. This method is reliant on the fact that

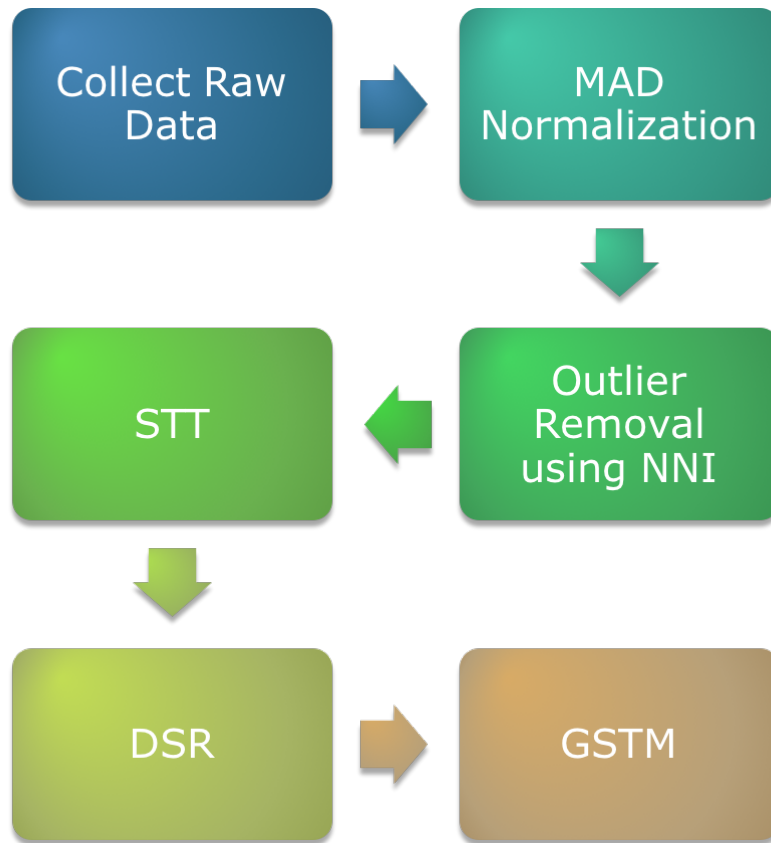


Figure 2.1: Different steps in the approach.

the sample in consideration is following a normal distribution, which enables the practitioner to define a range of two, two and a half or three standard deviations for detecting outliers. This method has several issues related to it. First and foremost, it assumes that the distribution is normal, that any data in consideration for outlier detection or any other task, is assumed to be following a normal distribution. Secondly, intuitively we can deduce that the mean and the standard deviation are both heavily influenced by outliers. This works even worse when the outliers have extreme range of values. Lastly, the method hardly works when it comes to small samples of data and it clearly has a fundamental problem. The whole purpose of detecting outliers in a large or a small sample of data is rendered redundant since the indicator and the measurement, both are heavily affected by the presence of outliers. Lets consider an example of sample data: 1,3,3,6,8,10,10 and 1000. The mean for this sample is 130.13 and the standard deviation is 328.80. This means the range for outliers – considering we assign a threshold of three standard deviations is – -856.27 to 1116.52. The mean is unrepresentative of the underlying distribution of the sample data and thus, it does not classify 1000 as an outlier.

### 2.1.2 MAD Normalization

The Median Absolute Deviation is a similar metric for measuring the central tendency for sample of data. The main advantage here is, that the metric is completely insensitive to the outliers. The metrics (estimator) breakdown point is the maximum set of numbers from the sample which will not force the estimator's value to give a false value. For mean, the breakdown point is minimal since the inclusion of even one outlier will change the estimator's value. Thus, the breakdown point for mean is ideally zero. Intuitively, for a median based approach, the breakdown value is 50 percent since, we will require more than 50 percent of the observations to be outliers in order to change the median to a different value. Another major advantage of the median is that it is completely immune to the sample size. Being the centre value, for a small or a large sample of data, the median isn't affected by the presence of outliers. It is also considered to be more robust than the classical interquartile range method for detecting anomalies.

$$MAD = \text{median}(|X_i - \text{median}(x)|) \quad (2.1)$$

For explaining the above observations and characteristics, let's consider the previous sample data i.e. 1,3,3,6,8,10,10 and 1000. Since the number of samples in the data is eight, the median value lies between the fourth and the fifth value. Equation 2.1 shows how to calculate the median absolute deviation. We first subtract each of the elements with the median value. The next step is to record the median of these subtracted values which is the MAD value. A constant  $b$  is often multiplied to the whole value under the assumption of normal distribution. In cases when the distribution is not normal, we take the 0.75 quantile of the respective distribution as the value of  $b$ . As you can clearly see, none of these values were operated on by any of the outlier values (in this case, 1000). Simulating the calculation, after subtracting from the median and taking absolute, we get 6, 4, 4, 1, 1, 3, 3 and 993. Sorting the list, we get 1, 1, 3, 3, 4, 4, 6 and 993. Thus, the median of the values is 3.5. Multiplying with 1.4826 gives 5.1891. Figure 2.2 displays the comparison between Mean and MAD. Depending on the users requirement, we can either use two or three MADs from median to specify the range for outliers. Thus, we get 22.5 (Median + 3\*MAD) and -8.57 (Median - 3\*MAD) as the range of values for detecting outliers. Compare it to the range of values obtained using the mean methodology and we can clearly see that it is more representative of the underlying data distribution. MAD also offers a consistent metric to eliminate values lying beyond the range. Similarly, using MAD (Median Absolute Deviation) we can clearly see that it eliminates 1000 as an outlier. The range of values produced for MAD is using three as the limit, which also provides an idea of how far off 1000 is from the standard data. Coming to the consistent constant, it is important to multiply this to the MAD value obtained from the sample, to consider the variable size of the sample data. It handles an extremely large number of values as well as with very small sample size. In simple words, it becomes an estimator for the population. Often times, the MAD also faces the issue of highlighting points which are on the border to be an outlier. This is another issue that the consistency constant takes care of, i.e. trying to even-out the outlier points.

The passage above was an explanation of the process of Median Absolute Deviation Normalization. We tried to highlight a few points on why this a better method than a general mean based normalization method, how this method can detect accurate outliers. In our case, we use the same Median Absolute Deviation Normalization method for normalizing the data received through each sensor on the



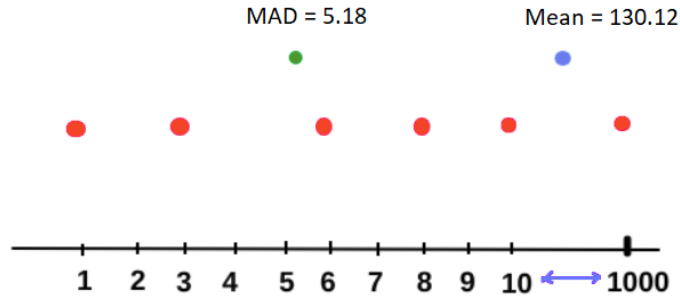


Figure 2.2: Mean vs MAD Comparison.

skull cap. In the case when there is an outlier for any instance of a particular sensor, we don't need to remove that entire sensor or entire instance. Instead, we replace that sensor value with the nearest non-outlier sensor value for that instance. In order to make amends to the detected outliers, the method that we used is Delaunay's Triangulation which is explained in the next section.

### Impact of MAD Normalization

We can safely conclude that MAD Normalization is an effective normalization technique. In terms of its impact on EEG, we can infer that it is highly resilient to outliers and can robustly remove them. [citation]

Also, we can see that extreme outliers - like the example given above - do not have the potential to impact the median value which is used to calculate the MAD.

In the study of EEG, outliers can be caused because of some indigenous artifacts like the blink of an eye or jaw clenching. For the following cases, there is a possibility that in all likelihood, the voltage values captured by the sensors in a region are high for a split of a second (which get captured for many rows of data). In such cases, if we use the standard normalization techniques, the mean could be displaced, resulting in an increase in the standard deviation. In order to have a central tendency across time, it is very important to identify extreme outliers which might be caused due to some exogenous factors like faulty sensors or signal interference.

This resilience provided by the MAD Normalization technique helps in neutralizing this effect and eventually has a very positive impact of outlier removal that

tars the study of EEG.

## **2.2 Outlier Removal using Nearest Neighborhood Interpolation (NNI)**

### **2.2.1 Background**

The human brain is a continuous surface. This means that activities which take place are not captured by just one sensor. Instead, the signature of a particular activity is spread across nearby sensors implying that there is an inherent correlation between nearby sensors. This means that if we see an abnormality in one sensor's captured voltage, it is more likely to be an outlier.

For such instances, we replace the abnormal values given by a sensor with the values of the neighboring sensors. In order to replace such outliers – which were detected by the MAD Normalization – we use the standard Nearest Neighbor Interpolation.

However, the sensors are not equally grid spaced and hence we use Delaunay's Triangulation in our scenario.

### **2.2.2 Nearest Neighborhood Interpolation using Delaunay's Triangulation**

We start by explaining what the Voronoi diagram[13] is, and then eventually move on to the explanation of Delaunay's Triangulation Method. Let's consider a set of sensors placed on brain scalp. Now, for any three sensors, the Delaunay's Triangulation method finds a triangle for the 3 closest sensors such that a circumcircle can be drawn through each edge of the sensor and no sensor will lie within these circles. Although, it is allowed for more than two sensors in the circumcircle. Figure 2.3 shows the process of creating triangles.

To get the Voronoi diagram out of these set of points, we need to start constructing perpendicular bisectors through each of the edges such that they intersect at a point. The Following image displays an idea of how the combination of perpendicular bisectors will look creating a voronoi region surrounding the points. These depictions will illustrate how the perpendicular bisectors are drawn and the intersec-



Figure 2.3: Voronoi Triangulation creation from a set of points

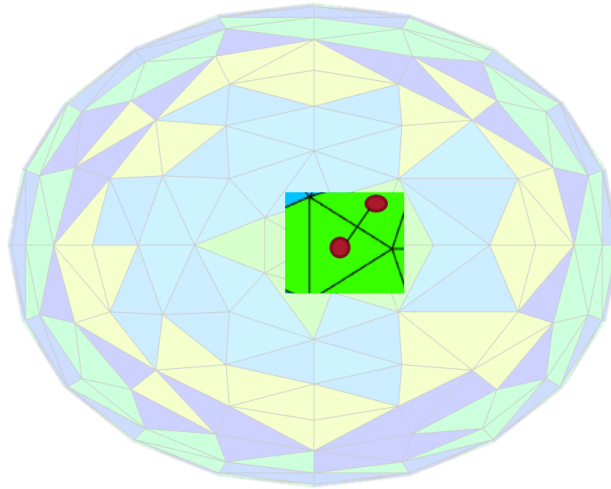


Figure 2.4: Nearest Neighbor Interpolation

tion of these edges help create a voronoi diagram from the Delaunays Triangulation method. So, the nearest neighbor for one point can be calculated by computing the distances to the centroid of the nearest voronoi regions, i.e. distance to the points in the adjacent regions. In this way, we can calculate the nearest neighbors for any set of points.

We use the above method in our problem. While using MAD normalization method, multiple extreme outliers in the sensors that record EEG data can be found. If the normalization method finds an outlier, using Delaunays Triangulation we identify the nearest neighboring sensor and replace the nearest non outlier value from the neighbor sensor for that instance as shown in the above Figure 2.4.

### 2.2.3 Impact of NNI

Based on our method used, we can infer that the values of our sensors will be consistent in any eventuality of a particular sensor failing. Also, we manage to retain the spatial information in the process since we only consider the nearby sensors in our interpolation technique.

From the EEG perspective, this technique can be used for impact analysis with respect to localized regions of the brain.

## 2.3 Spatio-Temporal Transformation (STT)

### 2.3.1 Background

In order to simplify the surface fitting process, it is useful to reduce the dimensionality of the EEG data. Raw EEG data contains information about both, the sensor location and the signal amplitude, which is four dimensional (4D). While a surface fitting algorithm could be used on four dimensional data, it would be much more computationally intensive, and its results would be less visually intuitive. Fortunately, a relatively simple coordinate transformation can be used to allow the temporal component of EEG to be embedded in the spatial domain. This spatio-temporal transformation (STT) can be represented as:

$$(x^i, y^i, z^i, S_t^i) \rightarrow (\chi^i, \gamma^i, \zeta^i) \quad (2.2)$$

Where  $x^i$ ,  $y^i$ , and  $z^i$  are the cartesian coordinates of electrode  $i$  and  $S_t^i$  is the normalized signal amplitude of electrode  $i$  measured at a given time point.  $\chi^i$ ,  $\gamma^i$  and  $\zeta^i$  are the transformed cartesian coordinates using spatio-temporal transformation method.

### 2.3.2 Spatio-Temporal transformation

Spatio-Temporal transformation is accomplished by encoding the signal amplitude of each electrode as a surface normal originating from the electrodes physical location. The length of the surface normal is proportional to the electrodes signal amplitude adjusted by an experimentally determined offset parameter. This parameter is added to the voltage value before smoothing. Higher values decrease accuracy but increase

smoothness (because high values are so much larger than the voltage making its importance less) and vice-versa. The desired value is based on the EEG application.

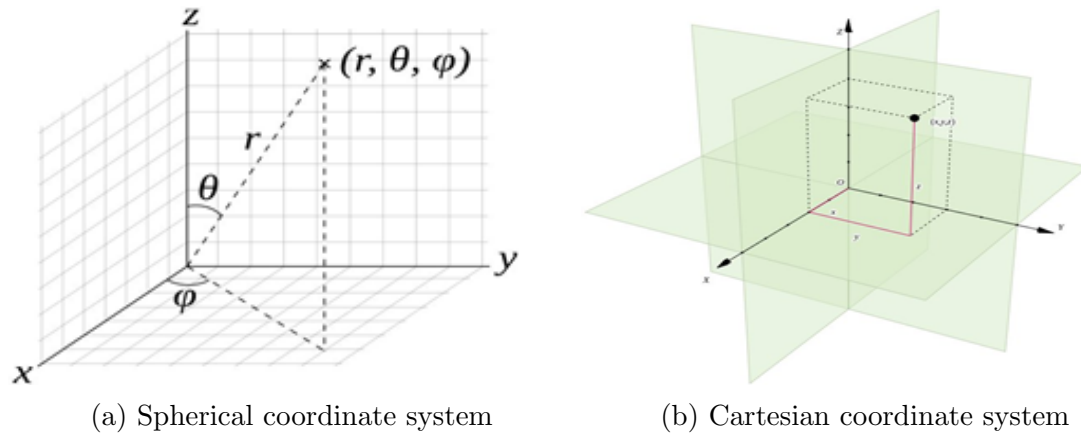


Figure 2.5: Illustration of Spherical and Cartesian co-ordinate system

Spherical coordinate system is used as an intermediate step to accomplish the spatio-temporal transformation. In the spherical coordinate system, the position of a point is specified by three numbers:

- The radial distance ( $r$ ) of that point from a fixed origin
- The polar angle ( $\theta$ ) measured from a fixed zenith direction
- The azimuth angle ( $\phi$ ) of its orthogonal projection on a reference plane that passes through the origin and is orthogonal to the zenith, measured from a fixed reference direction on that plane.

The spherical coordinate system generalizes the two-dimensional polar coordinate system as it can be seen as the three-dimensional version of the polar coordinate system. It can also be extended to higher-dimensional spaces and is then referred to as a hyperspherical coordinate system. The Spherical coordinate system and cartesian coordinate system are visualized as shown in Figure 2.5.

The spherical coordinates  $\theta$  and  $\phi$  are measured in degrees. If we approximate the human head as a sphere, then intuitively, spherical angles can be considered as a measurement from an imaginary center in the middle of the head. The detailed explanation of spatio-temporal transformation (STT) is as explained from the below steps:

Step-1):  $x$ ,  $y$  and  $z$  are converted to theta and phi

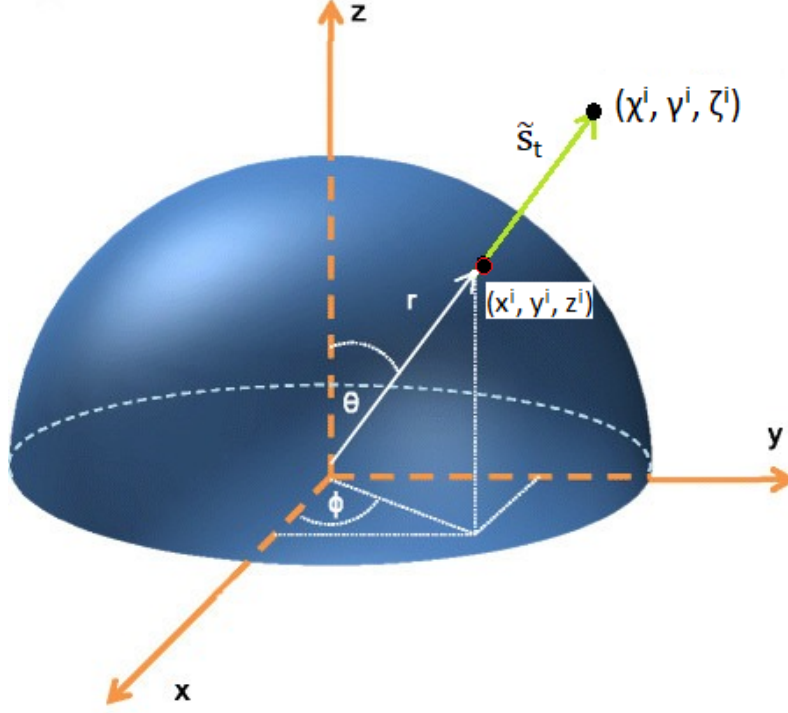


Figure 2.6: spatio-temporal transformation

Step-2): Offset parameter together with signal amplitude of that electrode at a given time point ( $S_t$ ) constitutes our radial distance  $r$  in spherical coordinate system as  $r = (\text{offset} + S_t)$

Step-3): Once we have  $r$ ,  $\theta$  and  $\phi$ , we get back to Cartesian coordinate system as shown below with the help of Spherical coordinate system to Cartesian coordinate system transformation:

$$\chi_i = (\text{offset} + S_t) * \sin\theta * \cos\phi \quad (2.3)$$

$$\gamma_i = (\text{offset} + S_t) * \sin\theta * \sin\phi \quad (2.4)$$

$$\zeta_i = (\text{offset} + S_t) * \cos\theta \quad (2.5)$$

This entire transformation is shown in Figure 2.6

Step-4): In order to capture the temporal component of EEG data, composite hypercoordinate point clouds consisting of multiple EEG time points were used for surface fitting. These composite point clouds consisted of 8 consecutive time points as shown in Figure 2.7, and so each surface generated by the algorithm was based

Temporal Data	Spatial Data(Sensors)											
	1	2	3	4	5	.....	128					
<b>1</b>	0.4	0.45	0.66	0.9	1.2	.....	0.99					
<b>2</b>	0.56	2.33	1.4	0.7	0.55	.....	0.78					
<b>3</b>	0.99	4.3	2.3	2.4	0.88	.....	0.76					
	1.2	3.5	3.2	3.7	3.2	.....	1.24					
	1.3	3.4	2.7	2.1	1.4	.....	2.3					
	0.34	0.9	2.4	2.1	1.8	.....	3.1					
<b>8</b>	0.23	0.57	1.2	1.5	1.4	.....	0.5					

Figure 2.7: Points before the transformation

Temporal Data	Spatial Data(Sensors)											
	1	2	3	4	5	.....	1024					
<b>1</b>	[ 0.4	0.45	0.66	0.9	1.2	.....	0.99]					

Figure 2.8: Points after the transformation

on 1024 hypercoordinate points as shown in Figure 2.8 (i.e. 128 sensors x 8 time samples). The size of point cloud composites can be adjusted to increase or decrease the amount of temporal fitting for each surface.

The offset parameter given in step 2 also helps to prevent the normal lines from accidentally overlapping each other and thus causing the smoothing algorithm to become confused over which surface the point belongs to.

This process yields a hypercoordinate point cloud represented by the tips of each surface normal. The point cloud obtained from the spatio-temporal transformation are used as input to the surface fitting algorithm which in later section demonstrates the improved robustness and general reproducibility of surface fitted EEG data.

### 2.3.3 Impact of STT

Thus, in order to produce efficient and intuitive results, a novel hypercoordinate transformation is utilized which integrates both the spatial and temporal information of the EEG signal into a unified coordinate system without losing any information.

## 2.4 Data Surface Reconstruction (DSR)

### 2.4.1 Background

An important thing to do to visualize EEG is try to estimate the best possible fit for common geometry types like sphere or hemisphere which is generally the shape of an EEG biosemi cap. To achieve this, we make use of sensor characteristics like noise, bias and range-dependent errors as well as data collection geometry like the location of a sensor and its orientation relative to the objects in the point cloud.

A common way to represent geometry in computer graphics, CAD and computer vision are b-spline curves and surface as well as their abstraction to non-uniform rational b-splines (NURBS)[14][15][16]. These techniques are heavily used in a wide array of fields including the entertainment industry, mechanical, electrical and medical engineering and in architecture and computer vision. Simple methods like least-squares minimization are employed for approximating point clouds using b-splines. More advanced techniques redefine the distance measure between the data points and the surface leading to a more robust, faster fitting procedure. B-spline surfaces are defined as the tensor product of their basis functions. This means that their boundaries are four-sided with the edges having the same order and degrees of freedom as the surface itself. Consider a disk where the surface is planar (1st order) and the edge of a circle (2nd order). It becomes clear that the edges in both cases have to be modeled separately.

### 2.4.2 DSR using Non-uniform Rational B-Spline (NURBS)

NURBS is one of the most employed surface fitting models, provided that it is a standard representation of curves and surfaces and is widely supported by modern standards like OpenGL and IGES, which are used for graphics and geometric data exchange. In addition, the NURBS surface model has stability, flexibility, local modification properties and is robust to noise.

In order to fit a NURBS surface to an unorganized and scattered point cloud, several approaches have been presented [17] [18]. Such approaches fit to the cloud a network of NURBS patches with some degree of continuity between them.

Let us start with the so called knot vector  $E = \omega_1, \omega_2, \dots, \omega_{n+p+1}$ , a non-decreasing set of coordinates in the parameter space  $\omega_c = [1, n+p+1]R$ , where  $n$  is the number of basis functions used for the B-spline curve and  $p$  is the polynomial order. The



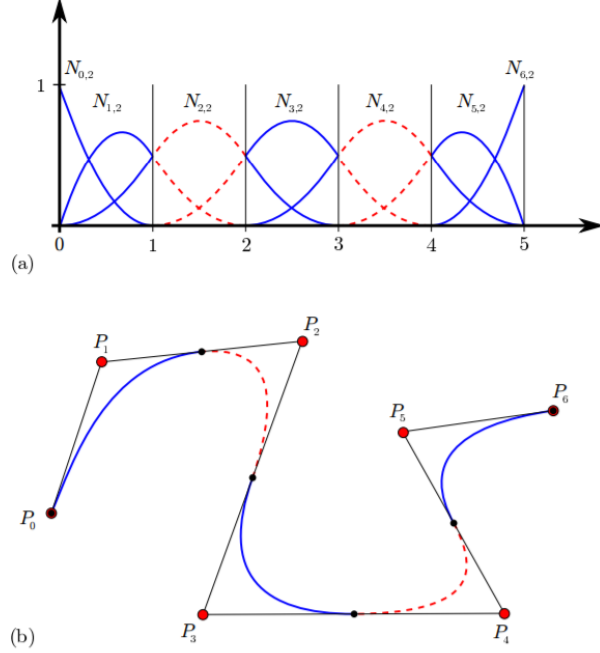


Figure 2.9: B-spline basis function and curve using the basis function

knots partition the parameter space into elements. With the knot vector in hand, the B-spline basis functions are defined with the Cox-de Boor recursion formula

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \quad (2.6)$$

for  $p = 0$  and for  $p = 1, 2, 3, \dots$ , they are defined by

$$\sum_{i=1}^n N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \quad (2.7)$$

There are several important features that come with this definition. The basis constitutes a partition of unity, that is,

$$\sum_{i=1}^n N_{i,p}(\xi) = 1 \quad (2.8)$$

Also the basis function is point-wise non-negative over the entire domain, that is,  $N_{i,p}(\xi) \geq 0, \forall \xi$ . Another important feature is, that each  $p$ -th order function has  $p - 1$  continuous derivatives across the element. Figure 2.9a shows basis functions of order  $p = 2$  with interpolating knots at  $\xi = 0$  and  $\xi = 5$

### B-spline curves

A B-spline curve  $c(\xi) : \Omega_c \subset \mathbb{R}^3$  with the parametric domain  $cR$  is constructed by linear combinations of B-spline basis functions. Given  $n$  basis functions  $N_{i,p}$  with  $i = 1, 2, \dots, n$ , the  $n$  vector-valued coefficients of the basis functions, called control vector  $b_i \in \mathbb{R}^3$ , defines the curve as:

$$c(\xi) = \sum_{i=1}^n N_{i,p}(\xi) b_i \quad (2.9)$$

The idea is to manipulate the B-spline curve  $c(\xi)$ , by changing the values of the control points  $b_i$ . The  $i$ -th control point defines the B-spline curve at its region of influence determined by the basis function  $N_{i,p}(\xi)$ . A quadratic ( $p = 2$ ) B-spline curve is shown in Figure 2.9b. An important characteristic of B-spline curves is that affine transformations of the curve are obtained by applying it directly to the control points. Furthermore B-spline curves obey a strong convex hull property resulting from the non-negativity and partition of unity properties of the basis, combined with the compact support of the functions. This leads to the fact that the B-spline curve is completely contained within the convex hull defined by its control points. This feature will be exploited for defining B-spline surfaces described in Section 2.2.2.

The measure for the distance is not necessarily the Euclidean distance between  $p$  and its closest point on the surface, which is also known as point distance-minimization (PDM). Another possibility is to use tangent-distance minimization (TDM) or squared-distance-minimization (SDM). For the EEG data we used the PDM distance-minimization.

The curves are typically initialized using principal-component-analysis (PCA), where the curve is initialized in the plane formed by the two biggest eigen vectors of the data. The eigenvalues are used to define the radius of the initialized control points from the mean. Although smoothness is inherent in B-splines an additional parameter can be defined by a non-discrete regularization term for problems of high degree of freedom (i.e. high number of knots / control-points). The closest points on the curve are calculated using Newton's method. Equivalent to 2D curve fitting the algorithms for 3D are implemented in the class where 3D data and curves are processed.

### B-spline surfaces

A B-spline surface  $S(\xi) : \Omega_s \subset \mathbb{R}^3$  with the parametric domain  $(u, v) \in \Omega_s \subset \mathbb{R}^2$  is constructed by linear combinations of the tensor product of B-spline basis functions.

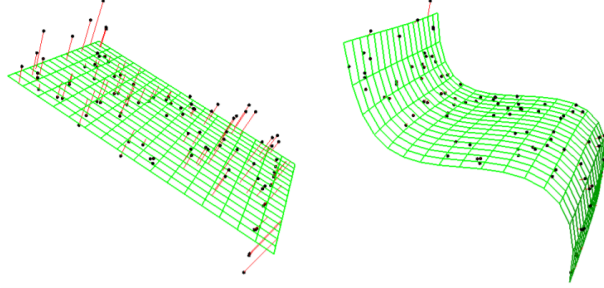


Figure 2.10: Curve Fitting using B-spline basis function

Given  $n$ ,  $m$  basis functions  $N_{i,p}$  and  $M_{j,q}$  with  $i = 1, 2, \dots, n$  and  $j = 1, 2, \dots, m$ , the vector-valued coefficients, called control grid  $B_{i,j} \in R^3$ , defines the surface as

$$S(u, v) = \sum_{i=1}^n \sum_{j=1}^m N_{i,p}(u) M_{j,p}(v) B_{i,j} \quad (2.10)$$

The same characteristics as for B-spline curves apply for B-spline surfaces. The derivatives in a given parametric direction may be determined from the respective one-dimensional basis function.

The algorithms for surface fitting are implemented similar to curves. In SDM a mixture of tangential-distance and point-distance is used depending on the local curvature of the curve at the closest point of a data point. In this method, we allow the user to define the influence of tangent- and point-distance manually, leading to a lower computational load while convergence is faster than PDM but still robust enough. Unfortunately the required storage for fitting is as high as needed for SDM. Again the surface is initialised using PCA.

### Triangulation

Triangulation of NURBS curves and surfaces is quite straight forward. The parameter space  $\Omega_c$  and  $\Omega_s$  is uniformly sampled respectively. As for non trimmed surfaces a triangulated plane of a certain resolution in  $u$  and  $v$  direction is created, covering the whole domain  $\Omega_s$ . Then each vertex has to be tested if it lies inside or outside the curve. Therefore the cross product of the vector of a vertex to its closest point on the curve with the tangent vector at the closest point is checked whether it is positive or negative with respect to  $z$  direction. The  $z$  direction is defined by the coordinate frame of the surface given by the PCA (or initialized by the user).

### Surface Solving

The mathematical problem of fitting a B-spline curve or surface to a point-cloud can be described as linear system. 2.10 can be reformulated as:

$$f_c = A_c b_c \quad (2.11)$$

where  $f_c$  is the vector of points to be fitted.  $A_c$  is a matrix containing the basis function, which multiplied by the vector of control-points  $b_c$  results in the curve at the respective parameter . Minimizing equation 2.11 in the PDM sense leads to minimizing

$$\|A_c b_c - f_c\|^2 \quad (2.12)$$

Equivalent to curve fitting Equation 2.11 leads to:

$$f_s = A_s b_s \quad (2.13)$$

and

$$\|A_s b_s - f_s\|^2 \quad (2.14)$$

Such a system can be solved using standard solvers such as:

- Eigen::JacobiSVD
- SuiteSparse::umfpack\_di\_solve<sup>2</sup>

where the first is a dense and the second a sparse solver which enormously speeds up the computation time. The choice of the solver can be made in the build configuration by enabling or disabling the flag `USE_UMFPACK`.

### 2.4.3 Implementation

We implemented NURBS on a point-cloud to obtain a smooth, parametric surface representation.

To summarize, the algorithm consists of the following steps:

- Initialization of the B-spline surface by using the Principal Component Analysis (PCA). This assumes that the point-cloud has two main orientations, i.e. that it is roughly planar.

- Refinement and fitting of the B-spline surface.
- Circular initialization of the B-spline curve. Here we assume that the point-cloud is compact, i.e. no separated clusters.
- Fitting of the B-spline curve.
- Triangulation of the trimmed B-spline surface.

We have used the open-source point cloud library for the implementation. Some of the parameters are discussed below:

- *order* is the polynomial order of the B-spline surface. In our case we use the order 3. With order 2 there was lost of information in some sections of the surface and with more it was taking a lot of time to compute and the result was almost same.
- *refinement* is the number of refinement iterations, where for each iteration control-points are inserted, approximately doubling the control points in each parametric direction of the B-spline surface. We used refinement as 5, at every increase of refinement the number of control points would double. So it will increase the computation exponentially.
- *iterations* is the number of iterations that are performed after refinement is completed. The iterations were tested from 5 to 20 and chose to keep 15, as it was making the surface smooth.
- *mesh resolution* the number of vertices in each parametric direction, used for triangulation of the B-spline surface. The more the resolution the better number of points can be recovered from the surface. We chose 500 as the resolution.
- *interior smoothness* is the smoothness of the surface interior. How strict the surface has to be in incorporating raw points on the surface. We chose 0.2 as we wanted a smooth surface.
- *interior weight* is the weight for optimization for the surface interior. Similarly to interior smoothness you can weight the points on the surface. We used 0.5.

- *boundary smoothness* is the smoothness of the surface boundary. Boundaries can be treated differently, but the idea is the same. We kept the same value 0.2.
- *boundaryweight* is the weight for optimization for the surface boundary. We kept the parameter 0.5. The below parameters were kept same as recommended by the PCL Nurbs fitting library.
- *addCPsAccuracy* the distance of the supporting region of the curve to the closest data points has to be below this value, otherwise a control point is inserted.
- *addCPsIteration* inner iterations without inserting control points.
- *maxCPs* the maximum total number of control-points.
- *accuracy* the average fitting accuracy of the curve, w.r.t. the supporting regions.
- *iterations* maximum number of iterations performed.
- *closest point resolution* number of control points that must lie within each supporting region. (0 turns this constraint off)
- *closest point weight* weight for fitting the curve to its closest points.
- *closest point sigma2* threshold for closest points (disregard points that are further away from the curve).
- *interior sigma2* threshold for interior points (disregard points that are further away from and lie within the curve).
- *smooth concavity* value that leads to inward bending of the curve (0 = no bending; less than 0 inward bending; greater than 0 outward bending).
- *smoothness* weight of smoothness term.

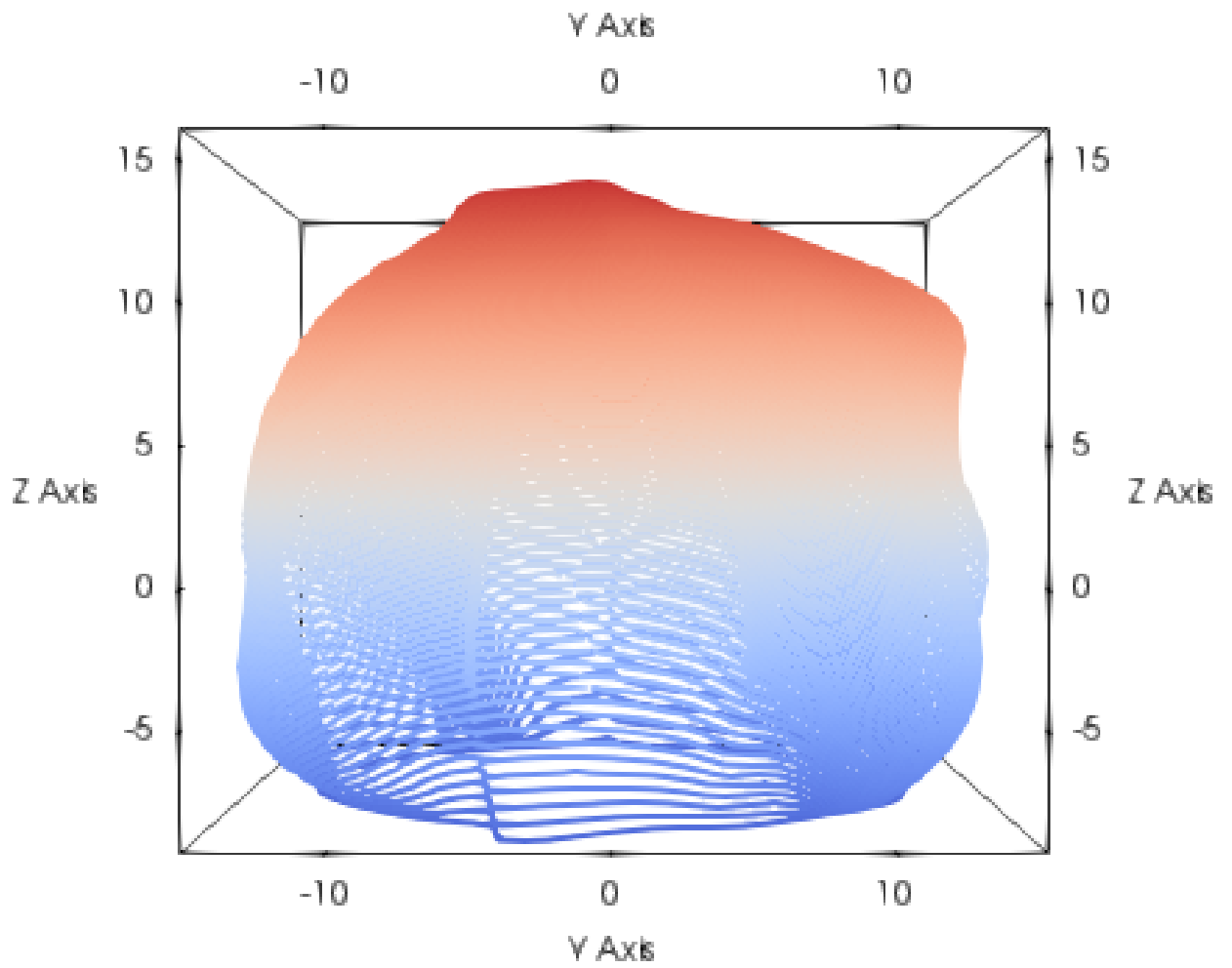


Figure 2.11: Example of a fitted surface using DSR

#### 2.4.4 Reconstructed surface using DSR

Figure 2.11 shows an example of a reconstructed surface using the methods described above after tweaking some parameters. As seen from the figure, once the minimized B-spline surface was fitted, it was then used to calculate a new value for each of the 128 sensor locations. First, the original hypercoordinate points were projected onto the B-spline surface and their values were replaced by that of their nearest neighbor of  $(\chi^i, \xi^i, \zeta^i)$  on the fitted surface. These nearest neighbor points represent the new spatio-temporally fitted point cloud with one to one correspondance to the original  $(\chi^i, \xi^i, \zeta^i)$  point cloud. These points are reverted back to original EEG signal coordinate by an inverse transformation process.

#### 2.4.5 Impact of DSR

DSR using NURBS helps in improving the robustness of the data by being resilient to outliers. In addition, it provides an underlying backbone structure of the data as well as gives an elegant visualization capability of the data. This helps in providing an intuitive interpretation of data.

In the following section, we demonstrate the general reproducibility that our DSR provides.



# Chapter 3

## Results

This chapter shows the results of the algorithms and approaches mentioned in the previous sections. It also explains the testing setup, evaluation approach for results and details the data used to generate the results. The following sections talk about the results of applying two classification algorithms on the raw and the fitted data and validate the results by analyzing the classification accuracy on unseen test data.

### 3.1 Validation

A key feature of our new data modeling strategy is its highly consistent spatio-temporal structure for a given subject in a given session.

Firstly, we have to ensure that our approach does not cause loss of information. While there are numerous transformations available, they cause loss of information and the underlying structure of the data. In our case, we want to test whether our approach causes any loss of information.

Secondly, we have incorporated spatial and temporal aspects of the data into consideration and fitted the surface which enables us to use and leverage both spatial and temporal information without losing out on either. To ensure that this spatio-temporal data does not perform poorly in classification is an important validation.

Thirdly, the approaches used by us enable us to get rid of outliers and help in having a dataset with lesser variance. Outliers adversely affect any analysis and the fact that we have eliminated outliers helps us understand if we have lost out on any information in the process.

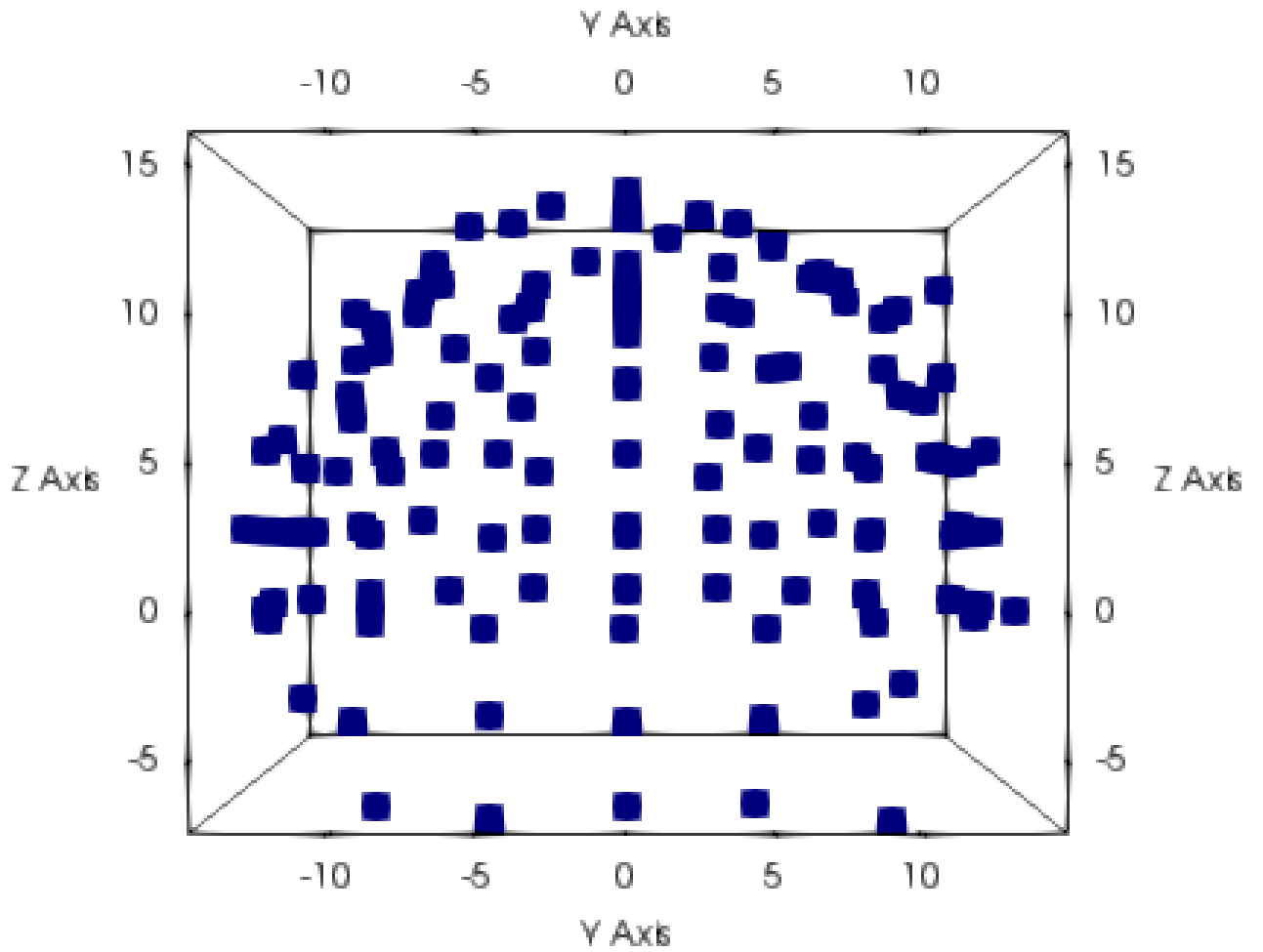


Figure 3.1: Raw data points before DSR

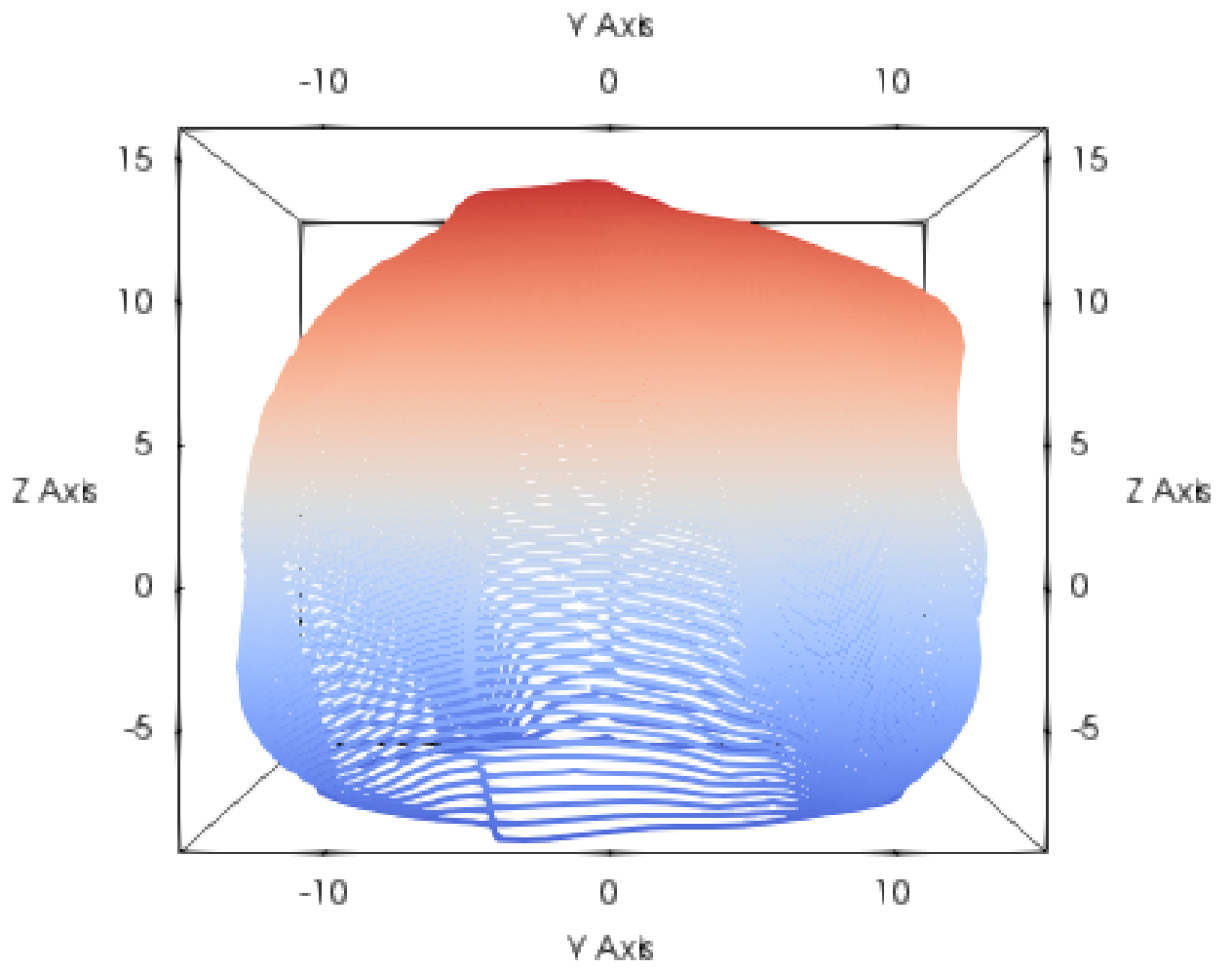


Figure 3.2: The surface fitted after DSR

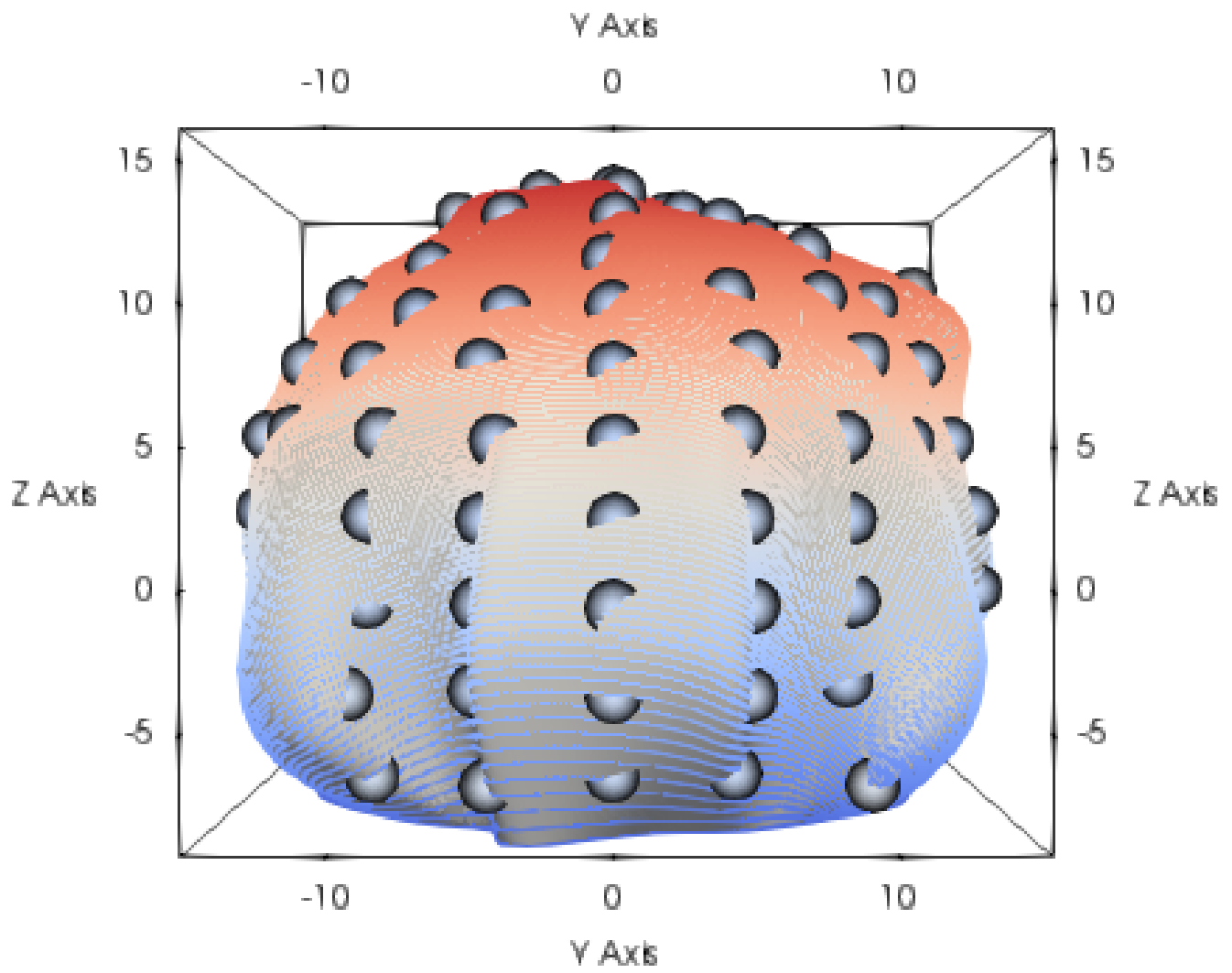


Figure 3.3: The fitted data points on the surface

To demonstrate this, we trained a nearest centroid classifier using both raw EEG data and surface fitted EEG data for three users and then compared its performance with the raw data.

One important thing to note is that the data is in essence, a time-series data. Apart from validating our approach in the time-series, we have also validated the results in the frequency spectrum. To achieve this, we applied Fast Fourier Transform on the data and converted the time-series data to various frequency domains. We obtained Alpha, Beta, Lower Gamma and Higher Gamma frequency ranges from the time-series data and applied the classification models on the raw and fitted datasets to obtain information about the classification accuracy in the frequency domain.

The sub-sections below explain each method used and the section that follows shows the results of the methods applied.

### **3.1.1 The Classification Problem**

Our goal was to leverage the spatio-temporally fitted data to test the classification accuracy on different activities. As mentioned before, we have four meditation states and one baseline state, each labelled as such. We used the dataset of three subjects and tried to classify the activities. Here, we combined the same baseline activity performed by all three users into a single class called 'baseline'. Similarly, for the other four meditation states, we combined all the three users' data for the corresponding activity and labelled them as such.

Thus, each class of our data contains the baseline/meditation activities of three users making this a five-class classification problem.

### **3.1.2 Salt and Pepper Noise**

Salt and Pepper Noise is also known as impulse noise. This noise can be caused by sharp and sudden disturbances in the image signal. It presents itself as sparsely occurring white and black pixels.

By adding this noise to our data, we validate if it has an effect on the classification accuracy.

### 3.1.3 Data Transformation

The dataset that was originally used for fitting the surface was a broadband signal, that is, it contains all the frequency spectrum and is a time-series data. However, we have also performed Fast Fourier Transform on the data to capture the underlying information in the frequency spectrum. This is demonstrated in the section below.

### 3.1.4 Fast Fourier Transform

A fast Fourier transform (FFT) is an algorithm that samples a signal over a period of time (or space) and divides it into its frequency components[19]. These components are single sinusoidal oscillations at distinct frequencies each with their own amplitude and phase. Over the time period measured, the signal contains many distinct dominant frequencies.

An FFT algorithm computes the discrete Fourier transform (DFT) of a sequence, or its inverse (IFFT). Fourier analysis converts a signal from its original domain to a representation in the frequency domain and vice versa. An FFT rapidly computes such transformations by factorizing the DFT matrix into a product of sparse (mostly zero) factors. As a result, it manages to reduce the complexity of computing the DFT from  $O(n^2)$ , which arises if one simply applies the definition of DFT, to  $O(n \log n)$ , where  $n$  is the data size.

The frequencies of the brain can be noted in order of high to low by the order gamma, beta, alpha, theta and delta. Each one has a specific purpose but it is important to note that each one is emoting throughout the duration of waking state and will show up concurrently on an EEG. Here is a description of the 5 different frequencies that we have used and what they imply.

#### **Delta (0-3 Hz)**

The Delta frequency is between 0-3 Hz. As explained in [reference], it is generally the frequency with the highest amplitude whilst having the slowest waves. It is normal as the dominant rhythm in infants up to one year and in stages 3 and 4 of sleep. It may occur focally with subcortical lesions and in general distribution with diffuse lesions, metabolic encephalopathy hydrocephalus or deep midline lesions. It is usually most prominent frontally in adults (e.g. FIRDA - Frontal Intermittent Rhythmic Delta) and posteriorly in children e.g. OIRDA - Occipital Intermittent Rhythmic Delta).

#### **Alpha (8-12 Hz)**

has a frequency between 8 and 12 Hz. Is usually best seen in the posterior regions of the head on each side, being higher in amplitude on the dominant side. It appears when closing the eyes and relaxing, and disappears when opening the eyes or alerting by any mechanism (thinking, calculating). It is the major rhythm seen in normal relaxed adults. It is present during most of life especially after the thirteenth year.

#### **Beta (12-30 Hz)**

beta activity is "fast" activity. It has a frequency between 12 and 30 Hz. It is usually seen on both sides in symmetrical distribution and is most evident frontally. It is accentuated by sedative-hypnotic drugs especially the benzodiazepines and the barbiturates. It may be absent or reduced in areas of cortical damage. It is generally regarded as a normal rhythm. It is the dominant rhythm in patients who are alert or anxious, have their eyes open.

#### **Gamma (31-59 Hz)**

The gamma rhythm, a relatively high frequency (30 to 80 Hz) component of these fluctuations, has received a great deal of attention. Gamma is modulated by sensory input and internal processes such as working memory and attention. Numerous theories have proposed that gamma contributes directly to brain function, but others argue that gamma is better viewed as a simple byproduct of network activity.

#### **Higher Gamma (61-90 Hz)**

The higher gamma is one of the least studied frequency bands because of its high susceptibility to noise. As a result, there aren't many applications and they do not correspond to any particular activity.

### **3.1.5 Nearest Centroid Classifier**

In machine learning, a nearest centroid classifier or nearest prototype classifier is a classification model that assigns to observations the label of the class of training samples whose mean (centroid) is closest to the observation. In effect, this makes it similar to the label updating phase of the k-means algorithm. It also has no parameters to choose, making it a good baseline classifier. It does, however, suffer on non-convex classes, as well as when classes have drastically different variances, as equal variance in all dimensions is assumed.

Thus, Nearest Centroid is one of the most simple classifiers which tries to learn

the underlying structure of the individual classes and classify the data based on those classes.

### **Distance Metric**

The distance metric is the metric to use when calculating distance between instances in a feature array, in our case, the input data. The centroids for the samples corresponding to each class is the point from which the sum of the distances (according to the metric) of all samples that belong to that particular class are minimized. Our choice of metric to use was the euclidean distance.

Euclidean distance refers to the distance between two points. These points can be in different dimensional space and are represented by different forms of coordinates. In one-dimensional space, the points are just on a straight number line. In two-dimensional space, the coordinates are given as points on the x- and y-axes, and in three-dimensional space, x-, y- and z-axes are used. Finding the Euclidean distance between points depends on the particular dimensional space in which they are found.

In our case, we calculate the euclidean distance between the centroid of a class and all the points within that class and then classify the point to the class to which the euclidean distance is the smallest.

### **Shrunk Threshold**

Nearest shrunk centroid classification makes one important modification to standard nearest centroid classification. [20]It shrinks each of the class centroids toward the overall centroid for all classes by an amount we call the threshold . This shrinkage consists of moving the centroid towards zero by threshold, setting it equal to zero if it hits zero. For example if threshold was 2.0, a centroid of 3.2 would be shrunk to 1.2, a centroid of -3.4 would be shrunk to -1.4, and a centroid of 1.2 would be shrunk to zero.

After shrinking the centroids, the new sample is classified by the usual nearest centroid rule, but using the shrunk class centroids.

The user decides on the value to use for threshold. Typically one examines a number of different choices. To guide in this choice, PAM does K-fold cross-validation for a range of threshold values. The samples are divided up at random into K roughly equally sized parts. For each part in turn, the classifier is built on the other K-1 parts then tested on the remaining part. This is done for a range of threshold values, and the cross-validated misclassification error rate is reported for each threshold value. Typically, the user would choose the threshold value giving



the minimum cross-validated misclassification error rate.

### 3.1.6 Salt and Pepper Noise

## 3.2 Results of the GSTM

In our case we used the Euclidean Distance as our metric and did not input a value for the shrunken threshold to prevent the centroid from shrinking and affecting any analysis of our data surface fitting method.

This section shows the results obtained for the various frequency bands as well as the broadband data for the various activities which formulated our 5-stage classification problem. We applied a dual-stage approach. We test the classifier predictions on the fitted dataset and compare it with the raw dataset. In addition to this, we add 5 percent salt-and-pepper noise to the data and apply the same fitting technique and check the classification accuracy on that dataset as well.

We applied the Nearest Centroid Classifier on the datasets and calculated the test accuracy on each of the dataset. For comparison, we did a test accuracy calculation of the raw data with the fitted data over broadband and the individual frequency bands.

The figure 3.4 demonstrates our results.

### 3.2.1 Classification Scores With No Noise

As seen from the Figure 3.4, the test accuracy upon fitting a nearest centroid classifier with raw and fitted data is shown. The Broadband data shows the largest increase in test accuracy with the raw data having an accuracy of 26.8 percent and the fitted data having an accuracy of 56 percent.

Similarly, for Delta, Beta, Gamma and Higher Gamma ranges, the fitted data performs better than the raw data.

One point to be noted is the Alpha range. This is the only range which shows a slight decrease in accuracy for the fitted data when compared with the raw data.

Figure 3.4 also demonstrates the percentage change between the raw vs the fitted data. Considering the raw data to be the baseline, we compare the corresponding changes obtained with the fitted data for the whole range of datasets. We can note that the percentage improvement obtained with the broadband data is the highest,

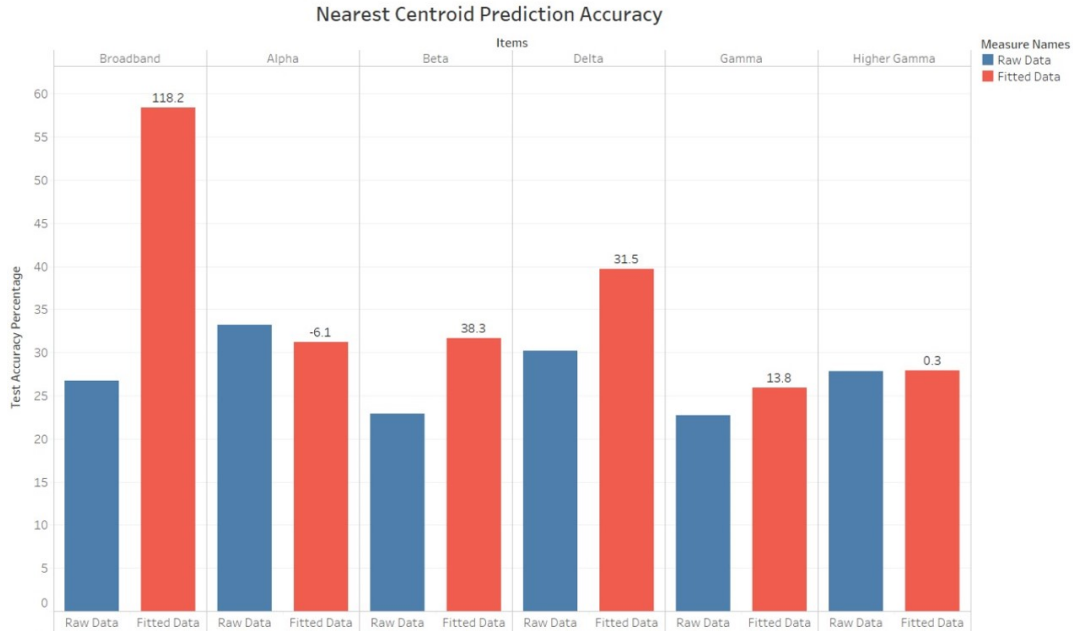


Figure 3.4: This figure shows the state classification accuracy obtained using the nearest centroid classifier on broadband, alpha, beta, gamma and higher gamma data for the raw and the fitted data.

a whopping 118 percent increase. However, as noted earlier, the Alpha band showed a slight decrease of 6 percentage. On the other hand, the Delta, Beta, Gamma and Higher Gamma ranges showed significant improvements.

### 3.2.2 Classification Scores With 5 Percent Salt And Pepper Noise

The dataset with 5 percent Salt and Pepper Noise cuts a similar figure in terms of the prediction accuracy for the fitted data. In fact, the classifier performs worse on the raw data, as is expected. This is owing to the fact that noise affects the classification and the underlying structure of the data. However, with the noise, we get a notion that the fitted surface ensures that the noise is handled effectively in this case, as the classification accuracy suggests.

The Figure 3.5 show the classification accuracy upon using the salt and pepper noise. Adding salt and pepper noise to the data has in fact improved the test accuracy in case of raw data and has maintained its accuracy with respect to the fitted data

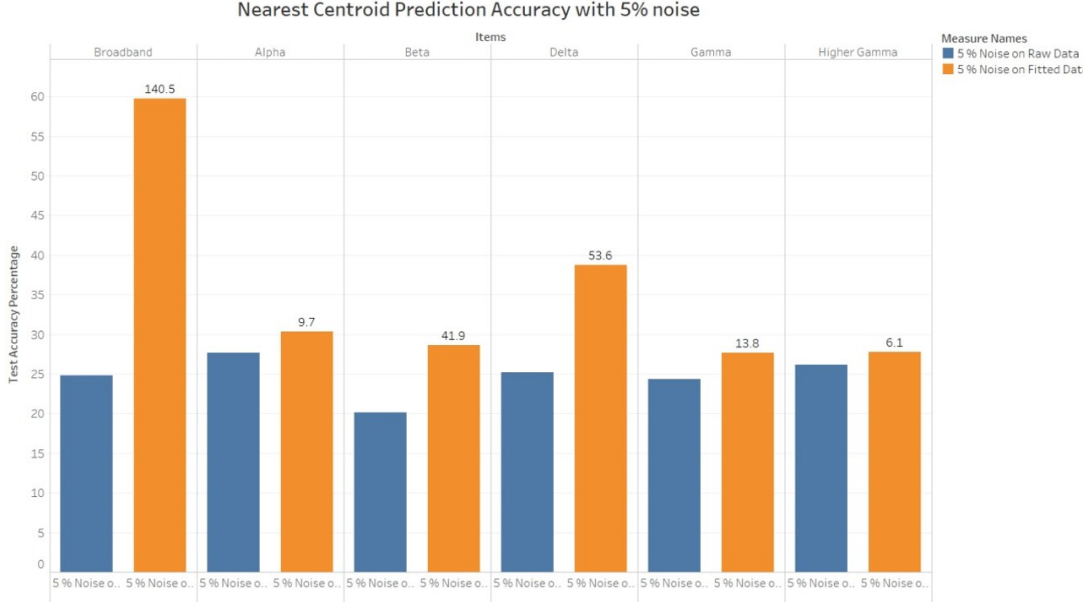


Figure 3.5: This figure shows the state classification accuracy obtained using the nearest centroid classifier on broadband, alpha, beta, gamma and higher gamma data for the raw and the fitted data with 5 percent noise

### 3.2.3 Image reconstruction with GSTM

Data reconstruction techniques often enhance certain data features while degrading the quality of others. In applications where the ideal outcome of the reconstruction is not well characterized, phantom data sources are used as a secondary validation measure [21]. To the best of our knowledge, there is no standardized phantom for spatio-temporal EEG data. As an alternative, we used the Shepp-Logan phantom [22] to indirectly validate our proposed method. The Shepp-Logan phantom is widely used for the validation of 2D image reconstruction algorithms. [23, 24] [23] [24] To make the phantom similar to STFs intended input, we randomly sampled a small fraction of standardized grayscale Shepp-Logan phantom and added random noise to each pixel value. This sparsely sampled, noisy phantom then underwent a modified version of the STT transformation used on our EEG data. The  $x_i$  and  $y_i$  pixel coordinates were treated as being equivalent to  $(\chi^i, \xi^i)$  and the noisy grayscale value of each pixel was equivalent to  $\zeta^i$ . The phantom's hypercoordinate point cloud was then processed using the STF with appropriately adjusted fitting parameters. This produced a reconstructed image which was analyzed to produce an estimate of the algorithm's performance on EEG data. This test is used to demonstrate that the

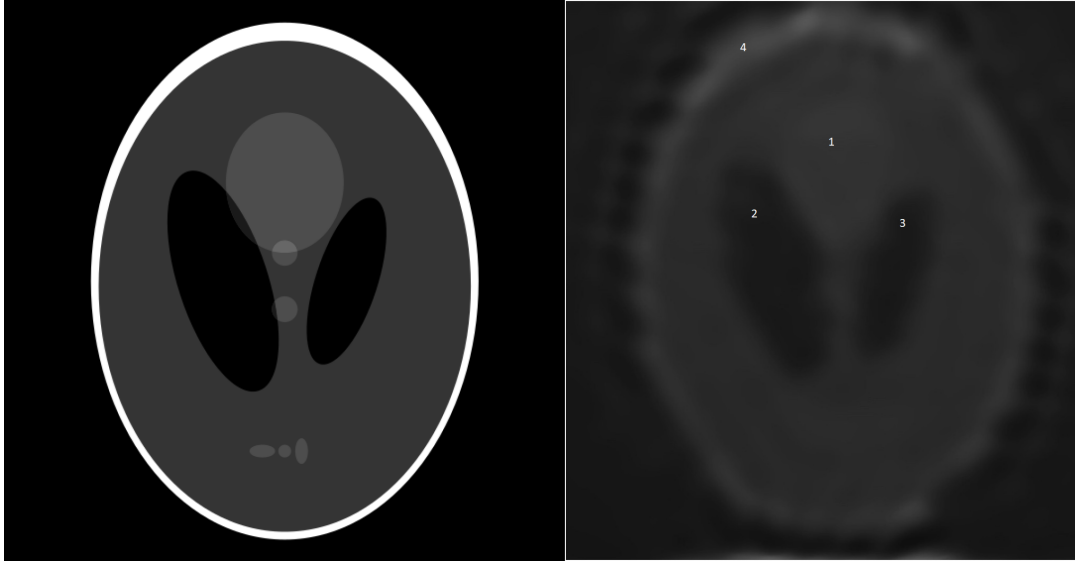


Figure 3.6: Reconstruction of Phantom Image using DSR

STF preserves major defining features of a 3D data set while reducing more minor and outlying features.

### 3.2.4 Impact

It can be inferred from the results obtained by using the nearest centroid classifier that the fitted data has almost consistently outperformed the raw data in classification. There is a significant improvement in the fitted data for the broadband signal and there are good improvements in the fitted data for the frequency bands, except Alpha.

Also, using the spatio-temporal data has not only not lost any information, but has in fact improved the accuracy in most cases and almost maintained the same accuracy in some others. This is a good breakthrough which suggests that the spatio-temporal transformation of the data does not lead to a loss of information.

We can also note that since the fitted data consistently outperforms the raw data, our method for data surface reconstruction is a very efficient alternative to the existing methods for interpolation.

Lastly, adding salt and pepper noise to the data has in fact improved the test accuracy in case of raw data and has maintained its accuracy with respect to the fitted data. This is a very encouraging sign that shows robustness and resilience of DSR towards external noise being added to the dataset.

The above results show strong support to our hypothesis that our DSR model more realistically captures the spatio-temporal structure of EEG.

# Chapter 4

## Discussion

Although the core methodology has been developed on EEG data, it can easily be adapted to situations where physical measurements are performed on a physically defined surface. Also, our model does not assume any particular kind of underlying geometry of brain or other part of the body with EGM signal. In situations where actual coordinates of the measurement points are available, the accuracy of our overall measurements and inferences about the underlying biological system would be more accurate. The granularity of the fitted data can be controlled by the various fitting parameters. In case of large  $r$  the fitted surface would conform to underlying measurement surface (Biosemi cap in case of EEG). In EEG applications where seamless user experience (e.g. neurofeedback) is highly desirable, our method works very well as random fluctuations are automatically eliminated. Our modeling technique would open up a whole new paradigm for EEG research due to increased clarity in the surface fitted data. Some of the future research would involve discovery of new task based signal signatures on the data from fitted surfaces. Although, the raw data normalization based on modified z-score followed by outlier removal is a reasonable logical step to eliminate unwanted data points, this may also artificially influence sensors completely unrelated to the group of sensors that may preferentially influence the median and median absolute deviation values. A brain surface region based section-wise normalization would be more accurate however this is issue is not the scope of this paper. Since the  $r$  value in STT influences the detailed information preserved on the data surface,  $r$  should be carefully selected based on the relative spread of the data on which the surface is fitted.

# Chapter 5

## Conclusion

### 5.1 Summary

Having an integrated spatio-temporal model of the EEG signal should provide a more comprehensive and robust picture of the underlying signal. These improvements should enhance our ability to differentiate cognitively distinct states in EEG data. Surface fitting not only enhances the quality of the measured signal, it also acts as a filter to random noise. In Fig. 3 we observe that pixel values contaminated with random noise is reasonably reconstructed even with a small randomly chosen fraction of the original image. In Fig. 4 symbol X shows that the state prediction probability (SPP) is reduced significantly when we add random noise to the raw data in different frequency bands however, the same remains very stable even in the presence of random noise.

This establishes the stability of our method. Our modeling technique would open up a whole new paradigm for EEG research due to increased clarity in the surface fitted data. Some of the future research would involve discovery of new task based signal signatures on the data from fitted surfaces.

### 5.2 Contribution

The dataset used for testing was collected from N human subjects using a 128 sensor Biosemi Active-two EEG device [1] at a sampling rate of 2048 Hz. Each EEG recording session consisted of a single baseline task followed by a 4-stage meditation task 8. These 5 cognitively distinct state (CDS) were designed by the experiment

and independently reported by the participants after each recording session. After recording, the data was filtered by first calculating the modified Z-score (MZS) for each sampled time point. Then, to eliminate extreme outliers, electrode values with an absolute MZS greater than a selected threshold (in this case 3.0) were replaced with that of their closest non-outlying neighbor. This way, even with the presence of extreme outliers within a small group of individual electrodes the filtered data was reasonably clean.

In order to simplify the DSR process, it is useful to reduce the dimensionality from 4 dimensions to 3 dimensions. EEG data is intrinsically 4-dimensional and contains information about both 3D sensor location and signal amplitude. Although surface fitting is feasible with 4D EEG data, 3D data is preferable as it significantly reduces processing time and produces more physically intuitive results. 3D EEG data can be produced by using a simple coordinate transformation which encodes EEGs temporal information in the spatial domain.

In order to capture the temporal component of EEG data, composite hypercoordinate point clouds were created using groupings of 8 consecutive EEG time points. The algorithm then used these composite point clouds to generate a fitted surface which minimized the point distance between the set of 1024 points (128 sensors x 8 time samples). Note that the size of the point cloud composites used with the algorithm can be adjusted to increase or decrease the amount of temporal fitting for each surface. Since, addition of more statistical instances of the measured time points increases the number of samples used for fitting the surface, this algorithm naturally gets adapted to perform temporal fitting without any modification to the algorithm.

## 5.3 Future Scope

Based on our research, we can see three different directions where this research could head.

Firstly, the concept of keypoint sampling can be applied. In this concept, we sample from one or more information-rich areas. Here, the areas with more discontinuity has more number of samples being extracted while the area from lesser discontinuity has lesser sampling. This ensures that we extract more information from feature-rich areas while the areas that do not exhibit more information are not



sampled thus ensuring that redundancy does not occur.

Secondly, studies can be done to identify the optimal number of sensors required to make the sensor cap for GSTM. More sensors increase the cost of a sensor cap and having lesser sensors makes an EEG cap to be more commercially viable.

Lastly, we can perform DSR for various frequency bands to extract maximum information for each frequency band. Right now, we are using a generic DSR framework but having a specific framework would help in further analysis of EEG.

# Bibliography

- [1] BV BioSemi. Biosemi activetwo.[EEG System]. *Amsterdam: BioSemi*, 2011.
- [2] Dennis J McFarland, Lynn M McCane, Stephen V David, and Jonathan R Wolpaw. Spatial filter selection for eeg-based communication. *Electroencephalography and clinical Neurophysiology*, 103(3):386–394, 1997.
- [3] Vojkan Mihajlović, Bernard Grundlehner, Ruud Vullers, and Julien Penders. Wearable, wireless eeg solutions in daily life applications: what are we missing? *IEEE journal of biomedical and health informatics*, 19(1):6–21, 2015.
- [4] Daniel M Goldenholz, Seppo P Ahlfors, Matti S Hämäläinen, Dahlia Sharon, Mamiko Ishitobi, Lucia M Vaina, and Steven M Stufflebeam. Mapping the signal-to-noise-ratios of cortical sources in magnetoencephalography and electroencephalography. *Human brain mapping*, 30(4):1077–1086, 2009.
- [5] Arnaud Delorme and Scott Makeig. Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *Journal of neuroscience methods*, 134(1):9–21, 2004.
- [6] Hugh Nolan, Robert Whelan, and RB Reilly. Faster: fully automated statistical thresholding for eeg artifact rejection. *Journal of neuroscience methods*, 192(1):152–162, 2010.
- [7] F Perrin, J Pernier, O Bertnard, MH Giard, and JF Echallier. Mapping of scalp potentials by surface spline interpolation. *Electroencephalography and clinical neurophysiology*, 66(1):75–81, 1987.
- [8] Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise smooth surface reconstruction. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 295–302. ACM, 1994.
- [9] Weiyin Ma and Jean-Pierre Kruth. Parameterization of randomly measured points for least squares fitting of b-spline curves and surfaces. *Computer-Aided Design*, 27(9):663–675, 1995.

- [10] Ennio A Vivaldi and Alejandro Bassi. Frequency domain analysis of sleep eeg for visualization and automated state detection. In *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, pages 3740–3743. IEEE, 2006.
- [11] Poppy LA Schoenberg, Andrea Ruf, John Churchill, Daniel P Brown, and Judson A Brewer. Mapping complex mind states: Eeg neural substrates of meditative unified compassionate awareness. *Consciousness and cognition*, 57:41–53, 2018.
- [12] Christophe Leys, Christophe Ley, Olivier Klein, Philippe Bernard, and Laurent Licata. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49(4):764–766, 2013.
- [13] Franz Aurenhammer. Voronoi diagrams a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.
- [14] Thomas Mörwald, Jonathan Balzer, and Markus Vincze. Modeling connected regions in arbitrary planar point clouds by robust b-spline approximation. *Robotics and Autonomous Systems*, 76:141–151, 2016.
- [15] Volker Blanz, Albert Mehl, Thomas Vetter, and H-P Seidel. A statistical method for robust 3d surface reconstruction from sparse data. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 293–300. IEEE, 2004.
- [16] Jonathan C Carr, Richard K Beatson, Jon B Cherrie, Tim J Mitchell, W Richard Fright, Bruce C McCallum, and Tim R Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76. ACM, 2001.
- [17] Fabio Remondino. From point cloud to surface: the modeling and visualization problem. *International Archives of photogrammetry, Remote Sensing and spatial information sciences*, 34, 2003.
- [18] In Kyu Park, Il Dong Yun, and Sang Uk Lee. Constructing nurbs surface model from scattered and unorganized range data. In *3-D Digital Imaging and Modeling, 1999. Proceedings. Second International Conference on*, pages 312–320. IEEE, 1999.
- [19] Benjamin Blankertz, Ryota Tomioka, Steven Lemm, Motoaki Kawanabe, and K-R Muller. Optimizing spatial filters for robust eeg single-trial analysis. *IEEE Signal processing magazine*, 25(1):41–56, 2008.

- [20] Matteo Pardo and Giorgio Sberveglieri. Random forests and nearest shrunken centroids for the classification of sensor array data. *Sensors and Actuators B: Chemical*, 131(1):93–99, 2008.
- [21] Pierre Jannin, Elizabeth Krupinski, and Simon K Warfield. Validation in medical image processing. *IEEE Transactions on Medical Imaging*, 25(11):1405–9, 2006.
- [22] Lawrence A Shepp and Benjamin F Logan. The fourier reconstruction of a head section. *IEEE Transactions on nuclear science*, 21(3):21–43, 1974.
- [23] H Michael Gach, Costin Tanase, and Fernando Boada. 2d & 3d shepp-logan phantom standards for mri. In *Systems Engineering, 2008. ICSENG'08. 19th International Conference on*, pages 521–526. IEEE, 2008.
- [24] Hengyong Yu, Shiyong Zhao, and Ge Wang. A differentiable shepp–logan phantom and its applications in exact cone-beam ct. *Physics in Medicine & Biology*, 50(23):5583, 2005.