Bachelor thesis

Artificial Intelligence

Radboud University

Improving the Event-related Potential Classification Performance of the Riemannian Pipeline using Shrinkage Regularization

Author:
Chiara Thöni
s1018813
chiara.thoni@student.ru.nl

First supervisor:
dr. M.W. Tangermann
Dept. of Artificial Intelligence
michael.tangermann@donders.ru.nl

Second supervisor:
J. Thielen MSc
Dept. of Artificial Intelligence
jordy.thielen@donders.ru.nl

June 18, 2021

Abstract

The aim in the field of brain-computer interfaces is to control an electronic device using brain responses that are recorded in an electroencephalogram. These brain signals can be decoded by means of the Riemannian metrics that are associated with the Riemannian manifold. This way of classifying allows for transfer learning possibilities and is more robust opposite to the Euclidean approach. However, to get to these desired characteristics, the data points that represent a single brain response need to be represented by their covariance matrices. This is a drawback of the framework, as the sample covariance matrix is a sub-optimal estimator of the true covariance matrix when the number of samples is low compared to the number of features. To improve the estimator and enhance the classification performance, I apply shrinkage regularization on the different submatrices of the epoch-based covariance matrix. To assess the effect of this method, I compare two decoding algorithms against a baseline using six datasets. The results show that there is a significant increase in classification performance for one out of the six datasets.

Contents

List of Figures

List of Tables

Conventions & Code

Mathematical Conventions

In the mathematical formulae that are used in this work I follow the conventions described in the table below.

Mathematical element	Meaning	Typeset
Scalars	Sample parameter	Lowercase Latin alphabet
	true or target parameter	Lowercase Greek alphabet
Vectors	Sample parameter	Bold lowercase Latin alphabet
	true or target parameter	Bold lowercase Greek alphabet
Matrices	Sample parameter	Bold uppercase Latin alphabet
	True or target parameter	Bold uppercase Greek alphabet

Table 1: The mathematical conventions used throughout this thesis.

Code

This work comes with python code that has been written to generate the figures and to carry the analyses for Chapter 4. The code has has been made publicly available under the MIT-license and can be found on:

`https://github.com/Racemuis/epoch-based-covariance-estimates`

Chapter 1

Introduction

The neurons that constitute our brain are a source of electric activity. The processes in the brain that underlie our functioning are characterized by measurable currents, or *brain waves*, that can be recorded, for example using electroencephalography (EEG) [1]. Analysing the brain signals is difficult due to a high trial-to-trial variability within subjects, a high variability between subjects, and a low signal-to-noise ratio that is caused by artifacts and neural background activity [2]. In the field of brain-computer interfaces (BCI), subjects are enabled to control a computer program, where their recorded brain waves are used as input to the interface. To record brain waves, a sample of the currents that are measured by the EEG channels is taken at a number of time points. As a result, the data is stored in a matrix with a shape *number of features* $\times$ *number of timepoints*. Both supervised and unsupervised machine learning techniques are used to decode the brain signals, prior to using it as input for the BCI [3]. The process of decoding is referred to as a decoding pipeline.
The conditions that make it difficult to use brain signals directly are worsened by small calibration data sets that are needed to train the supervised methods [3]. In addition, data could contain subclasses, which could require the use of different classifiers. Furthermore, a high prevalence of artifacts that make it harder to retrieve the signals of interest, can result in an effectively smaller data set [2], [4]. When a limited number of data points is available, it is challenging to estimate the true covariance that underlies the data [5], [6]. Still, the estimation is vital for supervised machine learning methods such as linear discriminant analysis (LDA), where the within and between class covariance matrices are used [2], [7].

1.1 Riemannian Geometry

To deal with the problem that has been sketched in the previous section, researchers propose to decode brain signals using Riemannian geometry [8]. The Riemannian algorithms are more robust and the framework allows for transfer learning [8]. However, the Riemannian metrics are only applicable on a manifold: a mathematical space that is locally Euclidean [9, p. 1]. To get from the Euclidean representation of the data to a manifold, the data needs to be represented differently.

When a covariance matrix is full rank, the matrix is symmetric positive-definite (SPD) [10]. The SPD matrices live on their own smooth manifold, that is curved as a convex cone [4], [8]. Thus, the Riemannian metric is applied to covariance matrices, rather than Euclidean feature vectors. Hence, the metric can be used if the covariance matrix of each data point, or more specifically, for each time window wherein a stimulus is presented to the participant, is calculated [4], [10]. This approach has a downside: compared to the estimator of the class-wise covariance matrix with which I started this chapter, the epoch-based covariance matrix is estimated over even fewer data points! Still, the dimensions of the epoch-based covariance matrix would be $p \times p$

where p is the number of channels or features.

For the classification of event-related potentials (ERPs), a brain responses that are time locked to a stimulus, this representation has got a drawback. Covariance matrices $\in \mathbb{R}^{p \times p}$ do not contain any temporal information. ERPs on the other hand, have got a specific signature in the temporal domain as they are time-locked to the onset of the stimulus [11]. Hence, it is proposed by Barachant et al. [12] to construct *super trials* of the data, where a so called *prototype* of the ERP response is concatenated to the recorded brain response. The prototype of the ERP response is obtained by averaging the single trial responses of a full class [8], [12]. As the prototype is concatenated to the response, the new shape of the data is $2p \times$ *number of samples*. As a result, the epoch-based covariance matrix would have the dimensions $2p \times 2p$. Moreover, this matrix consists out of submatrices that represent the covariance matrix of the response, the covariance matrix of the prototype and the cross-covariance matrix between the prototype and the response.

Due to the relatively small number of samples in a single epoch compared to the number of features, the SCM is a poor estimator of true covariance matrix [6]. As mentioned before, this problem can also arise when the covariance matrix of a full class is estimated. Blankertz et al. [2] propose to apply shrinkage regularization to the covariance matrices prior to classification as a countermeasure. They have shown that shrinking the within- and between-class covariance matrices improves the classification score when classifying the data points with LDA [2]. In this work, I aim to transfer this idea to the epoch-based covariance matrices by applying shrinkage regularization on each of the matrices that represent the single data points. Therefore, my research question is defined as follows:

1.2 Research Question

How can shrinkage regularization be applied to single data point covariance estimations to improve the classification score of the Riemannian pipeline with respect to the Riemannian pipeline without improved covariance estimations?

To assess the classification score, I use the area under the receiver operating characteristic (ROC), that is often referred to as ROC AUC.

I hypothesise that the application of shrinkage regularization to the submatrices of the epoch-based covariance matrices improves the classification score of the pipeline.

The rationale behind this hypothesis is that a better representation of the SCM leads to an improved classification score, as in [2]. To that end I constructed two different pipelines, *response shrinkage* and *super trial shrinkage*, to compare to a baseline. However, before I dive into my methods, I first introduce the reader with a background regarding the SCM, ERP classification and Riemannian geometry in chapter 2. In chapter 3 I present the work that has been done in the fields that are related to the subject of this thesis by highlighting different papers on Riemannian geometry and shrinkage regularization. Next, in the methods (chapter 4) I discuss the three pipelines that I have constructed and the evaluation techniques that I have used, so that I can go on to the results in chapter 5. Lastly, I discuss and summarize the findings in the chapters 6 and 7.

Chapter 2

Preliminaries

Purpose of the Chapter

Before I continue with Chapter 3, where I provide an overview of the relevant work for this thesis, I start in section 2.1 by covering the general idea of the sample covariance matrix. Next, I will touch upon the classification of brain signals in section 2.2. And lastly, in section 2.3, the notion of manifolds is discussed, to provide the reader with a background before diving into the Riemannian geometry that is tailored to the sample covariance matrices in the next chapter.

2.1 The Sample Covariance Matrix

2.1.1 Definition

In this thesis, the (improvement of the) sample covariance matrix is the central topic. To illustrate this statistical concept, consider a data matrix $\mathbf{X}$ with p features, and n samples:

$$\mathbf{X} \in \mathbb{R}^{p \times n} \tag{2.1}$$

$\mathbf{X}$ can be considered a concatination of the standing feature vectors $\mathbf{x}_t \in \mathbb{R}^{p \times 1}$ for each timepoint in n where a sample has been taken. The covariance matrix $\mathbf{\Sigma}$ captures the relations within and between the p features of the data. In reality, that true, or target covariance matrix is unknown. Therefore, $\mathbf{\Sigma}$ is estimated with the sample covariance matrix (SCM), that is calculated from the data [2]. The formula that describes the SCM, $\mathbf{S}$, of the data is shown in equation (2.2) [3]. To estimate the (co)variance, each feature vector, i.e., a single column of $\mathbf{X}$, needs to be mean-free. Therefore, the vector that contains the mean values per channel, $\boldsymbol{m} \in \mathbb{R}^{p \times 1}$, is subtracted from each feature vector $\mathbf{x}_t$.

$$\mathbf{S} = \frac{1}{n-1} \sum_{t=1}^{n} (\mathbf{x}_t - \boldsymbol{m}) \otimes (\mathbf{x}_t - \boldsymbol{m})^T \tag{2.2}$$

Because of the outer product in equation (2.2) the dimensions of $\mathbf{S}$ are $p \times p$. The variances within each feature are captured on the main diagonal of the matrix, whereas the covariances between the features are located on the off diagonal locations.
Due to the nature of the estimator, the covariance matrix is symmetric. In addition, if the eigenvalues of the SCM are strictly positive (greater than, and not including 0), the matrix is positive definite[1] [13, p. 81]. In that case, the covariance matrix is considered symmetric positive-definite (SPD). The SCM comes in different flavours and can also be used as an estimator when multiple different classes are concerned. This will be elaborated in section 2.2.2.

[1] If the eigenvalues of the covariance matrix are only nonnegative, the matrix is considered positive *semi*-definite [13, p. 81].

2.1.2 The Sample Covariance Matrix as Sub-Optimal Estimator

The SCM is the estimator that is commonly used to approximate the population covariance matrix [6], [14]. The quality of the estimation is dependent on the number of samples that are available [2], [3], [5], [6]. To explain this, the data, as introduced in the previous section, equation (2.1) are being considered. Here, p denotes the number of features, which is the dimension of the population covariance matrix and the estimator. n on the other hand, is used to indicate the number of samples that are collected. The distinct ratios of p to n can be grouped in to three groups given their impact on the quality of the SCM.

- $\frac{p}{n} > 1$: When the ratio of p to n is larger than one, which is the result when the number of samples is smaller than the dimension of the estimator, then the SCM is not full rank. In other words, some columns of the SCM are linear combinations of others. As a result of that, the matrix is *singular*: the determinant is equal to 0, meaning that the matrix is not invertible [3], [15]. This can be troublesome for machine learning techniques such as LDA (section 2.2.2), where the analytical solution requires the inversion of the within-class covariance matrix [16, p. 189].

- $\frac{p}{n} \leq 1$: the SCM will be full rank. However, if the ratio is not negligible, then the matrix is *ill-conditioned* [6]. Even though it is possible to compute the inverse of ill-conditioned matrices (because of their non-zero determinant), the estimation error will increase when doing so [6].

- $\frac{p}{n} << 1$: Only when the ratio $\frac{p}{n}$ is negligible, the estimator is well conditioned [6].

In Figure 2.1, the determinants are shown for different ratios of p to n. p is set to 100, so the population covariance matrix $\mathbf{\Sigma}$ that has been used in the figure is a symmetric matrix with the dimensions $\mathbb{R}^{100\times100}$. The matrix has been randomly sampled from a univariate Gaussian distribution with a mean of 0 and a standard deviation of 1. The n observations are simulated by a multivariate normal distribution that is parameterized with $\mathbf{\Sigma}$ while having a mean of 0. Apart from the determinants of the different sample covariance matrices, another property of the sub-optimal estimator becomes clear in Figure 2.1: the eigenvalues of the SCM are more dispersed than the eigenvalues of the true covariance matrix [2], [6]. The latter property of the sub-optimal estimator will return when discussing shrinkage regularization in section 3.2.1.

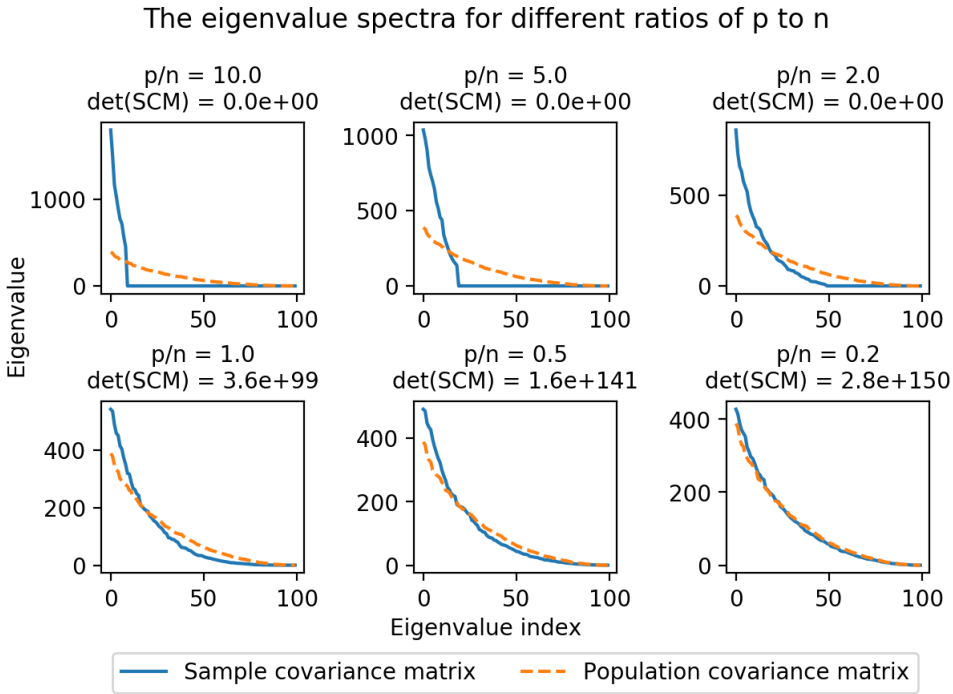


Figure 2.1: The eigenvalue spectra for different ratios of p to n. The eigenvalues are sorted in descending order. The population covariance matrix, $\mathbf{\Sigma} \in \mathbb{R}^{p \times p}$, has been sampled from a univariate Gaussian distribution, $\mathcal{N}(0, 1)$, where p has been set to 100. The n samples that serve as data have been simulated using a multivariate Gaussian distribution: $\mathbf{X} \sim \mathcal{N}(0, \mathbf{\Sigma})$. When the ratio is large, the large eigenvalues of the estimator are overestimated, whereas the small eigenvalues are underestimated. Moreover, the determinant of the SCM is equal to 0 if $\frac{p}{n} > 1$. All eigenvalues of the matrices that correspond to the different ratios of p to n are strictly positive.

2.2 The Classification of Brain Signals

In an EEG recording, the brain activity of the participant is measured using an array of p electrodes (also referred to as EEG channels) that are placed on different locations on the scalp [11, p. 223]. Consequently, at each time point t in the total number of time points n, there are p data points collected; one for each channel. The p data points per time point t can be stored in the feature vector of that time point: $\mathbf{x}_t \in \mathbb{R}^{p \times 1}$. If the standing feature vectors for all $t \in n$ are concatenated, they serve as the columns of the matrix $\mathbf{X} \in \mathbb{R}^{p \times n}$, that represents all data that is recorded by the p channels, at the n time points:

$$\mathbf{X} \in \mathbb{R}^{\text{channels} \times \text{time points}} \tag{2.3}$$

2.2.1 Event-Related Potentials

In my thesis, I have used among others the SPOT dataset that is discussed in section 4.2.1. The dataset has been collected during an auditory oddball paradigm. In this paradigm, event-related potentials (ERPs) are elicited and measured. As their name suggests, event-related potentials are brief changes in the brain signal that are a response to a sensory event or stimulus [11, p. 224]. A stimulus can be auditory or visual, and either the stimulus itself, a variant of the stimulus (the oddball) or even the absence of the stimulus can evoke an ERP [17, p. 14]. The responses are typically time locked to the event and their nomenclature is based on their sign and latency. For example, a positive peak around 300 ms after the presentation of the stimulus is called the P300, a negative with a latency of 100 ms can be denoted by N100 [11, p. 224]. Yet, this naming scheme is not strict, so that positive peaks with a latency between 300 and 1200 ms

still can be considered a P300 response [17, p. 31]. ERPs are time series data. Thus, a single trial in a paradigm, where for example multiple successive tones are presented to a participant, can be considered as a concatenation of the elicited ERPs. Each single ERP is called an epoch and is described by multiple feature vectors $\mathbf{x}_t$. Hence, the specification of the data that has been introduced in equation (2.3), can be updated as described in equation (2.4) to emphasize the different epochs in the trial.

$$\mathbf{X} \in \mathbb{R}^{\text{channels} \times (\text{epochs} \cdot \text{time points per epoch})} \tag{2.4}$$

An example of an ERP is shown in Figure 2.2. The figure shows an averaged ERP response over 260 target responses and 1300 non-target responses. The aggregation is needed as the evoked ERPs mixed with other electrical signals that originate from the brain [11, p. 224]. By averaging the epochs, the wave forms that are not of interest, tend to cancel out, and the ERP response remains. The channels of interest, Cz and Fz, are located on the mid-line of the scalp, where the first is centered, and the latter is located on the frontal lobe. In the figure, one can see a clear distinction between the target (orange) and the non-target (blue) responses. In the field of brain-computer interfaces, the actual automated classification of the ERP responses is a sequential process that is often referred to as a *pipeline*, which is the topic of the next section.

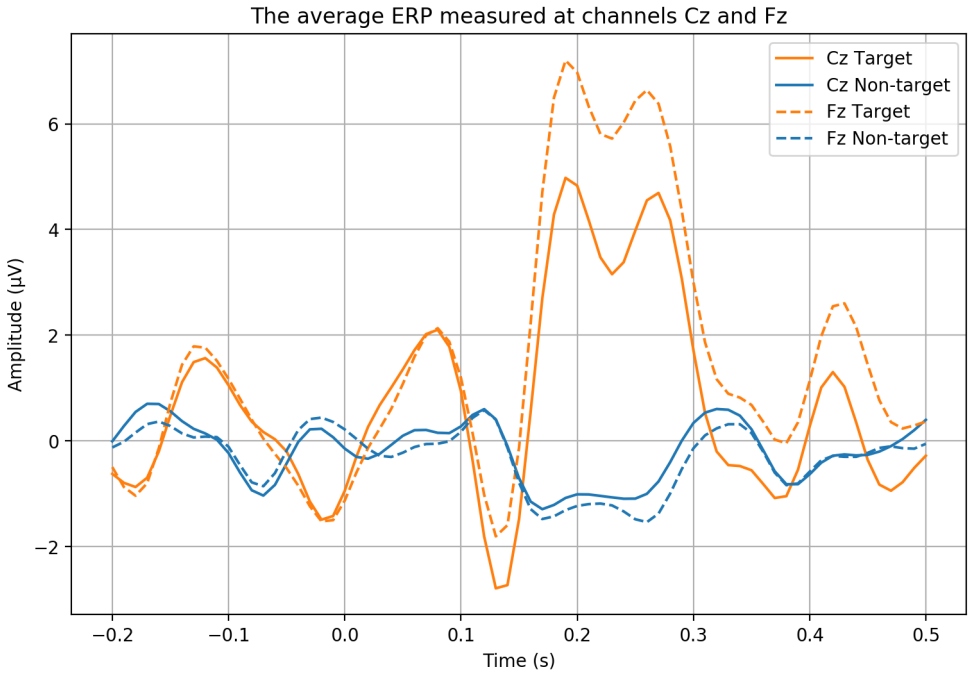


Figure 2.2: The electroencephalogram of an ERP response of a single subject that has been averaged over 260 target responses and 1300 non-target responses. The stimulus onset asynchrony (SOA), which is the time between two consecutive stimuli, is 226 ms [18]. This data is part of the SPOT dataset. In section 4.3.1, more information regarding this dataset and the pre-processing can be found.

2.2.2 The Classification Pipeline

ERPs can either be present (Figure 2.2, orange), or absent (Figure 2.2, blue). Hence, prior to using ERPs as an input for a brain-computer interface, a binary classification task needs to be solved: was an ERP elicited, or not? To solve this task, the raw brain data $\mathbf{X}$ (equation (2.3) and (2.4)) are decoded in a sequential pipeline. The pipeline can roughly be split up into 3 phases: pre-processing, feature extraction & classification [19], [20].

Pre-processing

Data pre-processing is the first step in the classification pipeline. An electroencephalogram contains multiple different wave patterns that live on a distinct frequency band. To distinguish the waves of interest from the noise, band pass, notch or Laplace filters are used [2]. Furthermore, baseline correction can be applied to reduce the effect of temporal drifts in the data [21]. The application of baseline correction is not always beneficial for the classification performance of the pipeline, as demonstrated in [3]. In addition, the data can be downsampled, to reduce the memory cost and speed up computations [22].

Feature Extraction

Next, the relevant features of the data can be extracted. The dimension of the feature vectors $\mathbf{x}_t \in \mathbb{R}^{p\times 1}$ is dependent on the number of channels on the EEG cap. The performance of the classifier that is used later on in the pipeline is negatively affected if there are relatively few samples compared to the number of features. More specifically, the number of samples that is needed to adequately classify the data $\mathbf{X}$ increases exponentially with the dimensionality of the feature vectors $\mathbf{x}_t$, a phenomenon that is also known as the *curse of dimensionality* [20], [23, p. 33].
Now, as the process of recording training data is time consuming and demanding for the subjects, most training datasets in BCI are small [19]. Hence, rather than increasing the number of samples, it is more common to decrease the number of features using dimensionality reduction techniques such as principal component analysis (PCA) and independent component analysis (ICA) [20].
PCA is a technique that identifies the principal directions in the data that correspond to the eigenvectors of the covariance matrix that correspond to the highest eigenvalues. The principal directions can be used as a new basis of the data, that describes a linear subspace [20]. ICA, on the other hand, is not just an algorithm that can be used to reduce the dimensionality by extracting the features of interest; it is also used as a method of blind-source separation (BSS), that allows for the unmixing of data by defining individual sources as linear combinations of the features [24]. PCA and ICA can also be considered as a pre-processing step because of their dimensionality reducing properties [19].

Another form of feature extraction is spatial filtering. Spatial filters are, just like ICA, a linear combination of the EEG channels. Yet, spatial filters produce a new set of "virtual channels" that aim to highlight a particular part of the brain [25]. In the methods (chapter 4), I use spatial filters that are proposed by Alexandre Barachant in [25]. These spatial filters are an adaption of the xDawn algorithm, that aims to reduce the signal-to-noise ratio (SNR) of the P300 responses by estimating spatial filters in an unsupervised way [26].

To continue with the mathematical definition of our data, I will introduce (spatial) filtering. Let f be the number of selected spatial filters, then the matrix containing the spatial filters can be denoted with $\mathbf{W} \in \mathbb{R}^{\text{channels}\times f}$. The filtered data is then defined as:

$$\mathbf{X}_{filter} = \mathbf{W}^T\mathbf{X} \tag{2.5}$$

Where $\mathbf{X}_{filter} \in \mathbb{R}^{f\times\text{time points}}$. More information regarding the estimation of the spatial filters using Barachant's adapted methods can be found in 3.1.2.

Classification

In the final step of the pipeline, a classification algorithm is used to classify the possibly pre-processed and filtered data. All the pipelines that will be introduced in the methods, use linear discriminant analysis (LDA). The binary LDA algorithm aims to assign class labels to data

points given their position with respect to a hyperplane, which dimension is one lower than the dimension of the data. The hyperplane can be described by all the points $\mathbf{x}$ where the outcome of equation (2.6) is 0 [17, p. 18].

$$\mathbf{w}^T\mathbf{x} + b = y \tag{2.6}$$

In this equation, $\mathbf{w}^T$ denotes the weights and b is the bias. Note that $\mathbf{w}$ and $\mathbf{x}$ can be multidimensional.

To illustrate the algorithm, let c_1 and $c_2 \in C$ be the two possible classes that a data point $\mathbf{x}$ could belong to. The hyperplane that is defined in equation (2.6) ought to be the decision boundary that separates the c_1 and c_2 *the best. The best*, can be defined with the Fisher criterion (2.9), that can be determined using with the between- (2.7) and within-class covariance matrices (2.8), and the weights $\mathbf{w}$ [16, p. 189].

The between-class covariance matrix is defined by:

$$\boldsymbol{S}_b = (\boldsymbol{m}_{c_1} - \boldsymbol{m}_{c_2})(\boldsymbol{m}_{c_1} - \boldsymbol{m}_{c_2})^T \tag{2.7}$$

Where the class means $\boldsymbol{m}_{c_i}$ are equal to $\frac{1}{N_{c_i}} \sum_{t \in c_i} \mathbf{x}_t$ and N_{c_i} denotes the number of elements in class c_i [16, p. 187].
Whereas the within-class covariance matrix can be calculated using the following method:

$$\boldsymbol{S}_w = \sum_{c_i \in C} \sum_{\mathbf{x}_t \in c} (\mathbf{x}_t - \boldsymbol{m}_{c_i})(\mathbf{x}_t - \boldsymbol{m}_{c_i})^T \tag{2.8}$$

Using the equations (2.7) and (2.8), the Fisher criterion is defined as in [16, p. 189]:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_b\mathbf{w}}{\mathbf{w}^T\mathbf{S}_w\mathbf{w}} \tag{2.9}$$

To find the weights $\mathbf{w}$ that correspond to the optimal decision boundary, one can maximize $J(\mathbf{w})$, which implies that the projection of the between-class covariance matrix on $\mathbf{w}$ is maximized, whereas the projection of the within-class covariance matrix is minimized. According to Bishop, the optimal value of weights $\mathbf{w}$ is proportional to the inverse of the within-class covariance matrix, multiplied by the difference of the two class means, which is formally denoted as equation (2.10) [16, p. 189].

$$\mathbf{w} \propto \mathbf{S}_w^{-1}(\mathbf{m}_{c_2} - \mathbf{m}_{c_1}) \tag{2.10}$$

Now that the value of the weights $\mathbf{w}$ is known, one can project the feature vector $\mathbf{x}$ on a single dimension, by applying equation (2.6) to the input vector[2]. To map the continuous output of equation (2.6) to a binary class label, the threshold constant or bias term b is used. For example, if $y < 0$ the vector $\mathbf{x}$ is assigned to one class and in case $y > 0$, $\mathbf{x}$ is assigned to the other class [16, p. 187]. There is no analytical solution for determining b [17, p. 21].

There is a small drawback when working with LDA: In order to perform optimally, there are three assumptions identified that need to be fulfilled [2].

1. **The features of each class are normally distributed.** According to Blankertz et al. [2], this assumption is met when ERP data are concerned.

2. **The two classes have got the same underlying covariance matrix.** This assumption is based on the fact that LDA separates the classes using a hyperplane. The assumption assures that the data is linearly separable (if assumption 1 holds as well).

[2]It is worth to mention that ERPs are time series data and should be classified in both the temporal and spatial domain. A single feature vector does not capture the temporal features of the response [2].

3. **The true class distributions are known.** This assumption can never hold as the population parameters merely can be estimated from the sample. Moreover, as discussed in 2.1.2, the SCM is terrible estimator of the population covariance matrix. Typically shrinkage regularization is applied to improve the estimate of the covariance matrix.

2.3 The Euclidean Space and Manifolds

It is not trivial to give a definition of our physical space, despite we are living in it. Thus, when discussing this topic, I need to fall back to the work by Euclid, dating from approximately 300 BC [27]. In his treatise, *The Elements*, Euclid introduces the axiomatic system that is nowadays known as the Euclidean geometry to us [27]. In mathematics, an axiomatic system can be endowed on a set, which makes it a space. The Euclidean space is fundamental in the field of geometry, and it is represented by the Cartesian coordinate system $\mathbb{R}^d$ [28]. The Euclidean space is also a metric space, as it is a set endued with a metric: the Euclidean distance. The Euclidean distance function is defined between points in the Euclidean space. To illustrate, the Euclidean distance between the two $a: (a_1, a_2)$ and $b: (b_1, b_2)$ in a two dimensional Cartesian coordinate system ($\mathbb{R}^2$), is defined as the length of the straight line between the two points:

$$\delta_E(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \tag{2.11}$$

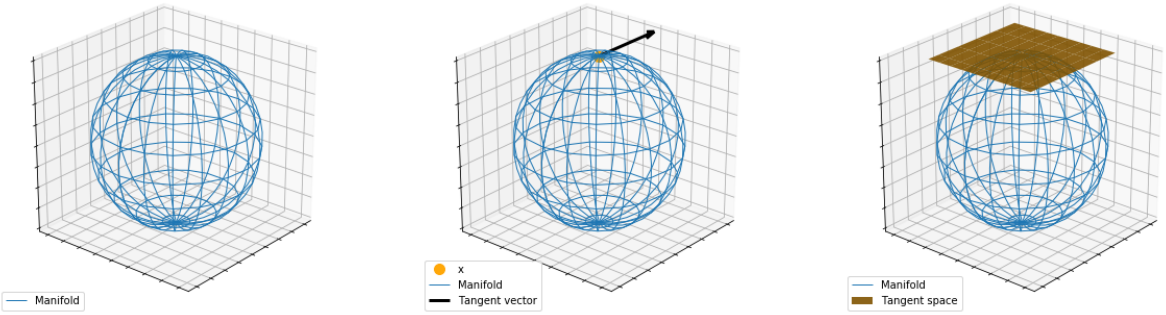
The definition in (2.11) is the result of the application of the Pythagorean theorem, and therefore this metric is also referred to as the Pythagorean distance.

In contrast to the Euclidean space, which is represented by a coordinate system with straight axes, and where distances are defined as a straight line, manifolds are curved. A manifold is a set of points that is infinitesimally Euclidean [29]. In other words, if one zooms in a lot on a manifold, then one will eventually end up with a Euclidean space. The example that is often used to illustrate the concepts of manifolds, is the earth. To continue with this illustration, consider the misconception that people in the Medieval times believed that the earth was flat [30]. If we would follow our own judgements of the earth, it does not sound like an unreasonable idea: our planet *seems* flat (and thus Euclidean), but that is because we observe it from our local (zoomed in) perspective which does not allow us to discover the curvature of the earth. Globally, the earth is a manifold that is more or less shaped as a sphere, as shown in Figure 2.3a.

2.3.1 Tangent Spaces

In the two dimensional Cartesian coordinate system, it is possible to calculate the tangent line to a point x on a curve y. This tangent can be found by taking the derivative of y in that point x. This notion can be generalized to the geometric tangent vector in Euclidean space $\mathbb{R}^n$ for any dimension n [31, p. 51]. If a manifold $\mathcal{M}$ is differentiable[3], then the derivatives of the curves going through point x on the manifold can be calculated (Figure 2.3b). The derivatives can be represented as vectors in $\mathbb{R}^d$ [29]. The combination of the all tangent vectors that go through x define the tangent space $\mathbb{R}^k$ to $\mathcal{M}$ at point x, that is commonly denoted with $\mathcal{T}_x\mathcal{M}$ (Figure 2.3c) [29], [31, p. 57], [32]. The tangent space is Euclidean, making it possible to deploy classifiers that are used for feature classification in the standard pipeline, such as LDA [4], [32].

[3]For more information regarding the differentiability of manifolds (as not every manifold is differentiable) I would like to refer the reader to [29] and [9, p. 4].



(a) A manifold $\mathcal{M}$ in $\mathbb{R}^3$ (b) A tangent vector at $x \in \mathcal{M}$ (c) $\mathcal{T}_x\mathcal{M}$

Figure 2.3: An impression of a spherical manifold embedded in $\mathbb{R}^3$. In (a), merely the manifold is shown. Next, in (b), a point $x \in \mathcal{M}$ is plotted, together with a tangent vector of an arbitrary length. In the last pane, (c), the tangent space $\mathcal{T}_x\mathcal{M}$ is plotted, that is defined as the combination of all tangent vectors at point x.

Chapter 3

Related Work

Purpose of the Chapter

This chapter will continue with the foundation that has been laid in the previous chapter, by discussing the work that has been done in the fields that are related to the subject of this thesis. In Section 3.1, the manifolds that are discussed in 2.3 will become more meaningful for the classification pipeline due to the addition of the *Riemannian metric*. We will see that the Riemannian pipeline has become increasingly popular in the field of BCI due to, among others, the beneficial invariance under congruence property of the Riemannian distance.
However, the pipeline requires estimates of the covariance matrix based on a single epoch. This can be a drawback, as we have seen in the previous chapter that the SCM is not a great estimator if the number of samples is small compared to the number of features. Fortunately, there exist methods that aim to improve the estimator, such as shrinkage regularization and time-decoupled LDA. These methods are discussed in Section 3.2. In particular, I highlight the paper by Sosulski, Kemmer & Tangermann on time-decoupled LDA in this section, as the pipelines that I will introduce in the methods are mainly adaptations of their work [3].

3.1 Riemannian Geometry in Brain-Computer Interfaces

The use of Riemannian geometry within BCI has become increasingly popular as among others the theoretical background has become more and more available [12] and it is more robust compared to the standard approaches based on feature vector classification [8], [33]. In addition, the performances of the pipelines that make use of Riemannian geometry are not inferior to the standard approaches: they have been used as basis for the winning methods BCI competitions such as the *DecMEG2014* and *BCI challenge 2015* [33]–[35].

3.1.1 Riemannian Manifolds

Even though the manifolds from Section 2.3 might allow for the construction of tangent spaces, they do not have a notion of distance. Certainly, this concept can be introduced to a smooth manifold[4], and for manifolds, the fitting structure to do this is the Riemannian metric [29], [36, p. 327]. The Riemannian metric is a chosen inner product on the tangent spaces $\mathcal{T}_x\mathcal{M}$ for each point x on the manifold $\mathcal{M}$ [8], [29], [32], [36, p. 328]. In equation (3.1) the inner product is defined for the two points $\mathbf{S_1}$ and $\mathbf{S_2}$ that live on the tangent space at point x [8], [32], [37].

$$\langle \mathbf{S_1}, \mathbf{S_2} \rangle_x = tr(\mathbf{S_1} x^{-1} \mathbf{S_2} x^{-1}) \tag{3.1}$$

Using the inner product from (3.1) the distance between two points $\mathbf{S_3}$ and $\mathbf{S_4}$ that are located on $\mathcal{M}$ can be determined. This distance is the Riemannian distance, and it is defined as the length of the geodesic: the shortest curve that passes through the $\mathbf{S_3}$ and $\mathbf{S_4}$. The definition

[4]The manifold $\mathcal{M}$ is considered smooth if $\mathcal{M}$ is locally Euclidean, and $\mathcal{M}$ allows for the application of calculus [9, p. 1]. More information regarding the premises can be found in 2.3.

of the Riemannian distance is shown in equation (3.2) [8], [32], [37]. In particular, in [37], the derivation from equation (3.1) to (3.2) is provided.

$$\delta_G(\mathbf{S}_2, \mathbf{S}_1) = \|\log(\mathbf{S}_1^{-\frac{1}{2}}\mathbf{S}_2\mathbf{S}_1^{-\frac{1}{2}})\|_F = \sqrt{\sum_{p=1}^{n} \log^2(\lambda_p)} \tag{3.2}$$

In (3.2), the F stands for the Frobenius norm (elaborated in equation (3.12)), and λ_p are the n eigenvalues of the matrix $\mathbf{S}_1^{-\frac{1}{2}}\mathbf{S}_2\mathbf{S}_1^{-\frac{1}{2}}$ [8].

The Invariance Under Congruence of the Riemannian Distance

The Riemannian distance has got a beneficial property for the area of brain-computer interfaces. As described by Barachant et al. [8], the Riemannian distance δ_G is invariant under congruence. This property makes the Riemannian pipeline appealing as it allows for generalization across subjects and sessions.

In their paper, Barachant et al. [8] define the matrix $\mathbf{ABA}^T$ as the congruence of matrix $\mathbf{B}$, where $\mathbf{A}$ is an invertible matrix. If EEG signal is considered as a linear combination of the sources that originate from the brain [38], then, the recorded feature vector can be written as $\mathbf{x}_t = \mathbf{A}\mathbf{u}_t$, where $\mathbf{A}$ can be considered the *mixing matrix*, and $\mathbf{u}_t$ are the underlying brain sources. Likewise, the SCM that can be computed from the recorded data $\mathbf{X}$, can be written as a combination of the covariance matrix of the unknown source ($\mathbf{U}$), and the mixing matrices:

$$\mathbf{S} = \mathbf{AUA}^T \tag{3.3}$$

In other words: $\mathbf{S}$ and $\mathbf{U}$ are congruent matrices. Let us now consider two subjects, $i \in 1, 2$. Both subjects attend two different recording sessions $j \in 1, 2$. Hence, there are 4 different mixing matrices $\mathbf{A}_{i,j}$, for the mixing matrix is highly specific for a subject and a session. For each of the sessions, a covariance matrix $\mathbf{S}_{i,j} = \mathbf{A}_{i,j}\mathbf{U}_i\mathbf{A}_{i,j}^T$ is calculated from the recorded data from each subject.
Because of the invariance under congruence, the Riemannian distance between the two pairs of covariance matrices for each subject is the same, as long as the covariance matrix of the underlying source for each subject is the same [8]:

$$\delta_G(\mathbf{S}_{1,1}, \mathbf{S}_{1,2}) = \delta_G(\mathbf{S}_{2,1}, \mathbf{S}_{2,2}) \tag{3.4}$$

If the underlying source covariance matrix is not the same, which is the case in a between subject generalization, then the performance will still be better in this generalization than the performance with the Euclidean metrics, as the highly specific mixing matrix has cancelled out [8].

3.1.2 The Classification of P300 Responses using Riemannian Geometry

Epoch Based Covariance Matrices

To exploit the advantageous properties of the Riemannian framework that are described above, the classification pipeline, as described in 2.2.2, needs to be based on Riemannian geometry. Consider the definition of the collected data of a session, $\mathbf{X}$, that is introduced equation (2.4). Here, the data is explicitly split up into epochs, where each epoch covers window in time. This definition can be reshaped to $\mathbf{X} \in \mathbb{R}^{e \times p \times n}$, where e is the number of epochs, p the number of (virtual) channels, and n the number of time points per epoch.

Effectively, $\mathbf{X}$ can be seen as the combination of e recordings of a single epoch $i \in e$:

$$\mathbf{X}_i \in \mathbb{R}^{p \times n} \tag{3.5}$$

In contrast to $\mathbf{X}_i$, that lies in the Euclidean space, the spatial covariance matrix $\mathbf{S}_i$ lives on the differentiable manifold of SPD matrices that is shaped like a convex cone [32]. Hence, in order to construct a Riemannian pipeline, the spatial covariance matrices are estimated for each time window of a single epoch $\mathbf{X}_i$ [4]. Thus, the single trial data that is used in a Riemannian pipeline is the concatenation of the e covariance matrices $\mathbf{S}_i$, calculated according to equation (3.6) (in this equation, the matrix $\mathbf{X}_i$ is assumed to be mean-free) [12].

$$\mathbf{S}_i = \frac{1}{n-1}\mathbf{X}_i\mathbf{X}_i^T \tag{3.6}$$

As such, a problem arises when classifying the data given the covariance matrices S_i: P300 responses (ERPs) are a time series signal. However, when merely the spatial covariance matrices of these signals are used for classification, the temporal structure is completely ignored!

To counter this issue, Barachant et al. [12] proposed to calculate the epoch based covariance matrices $\mathbf{S}_i$ from so called *super trials*, $\tilde{\mathbf{X}}_i$. To capture both the spatial and temporal information, the super trials are constructed by concatenating the epoch $\mathbf{X}_i$ over the first axis with a *prototyped response*, $\mathbf{P1}$. $\mathbf{P1}$ is defined as the Euclidean mean of all the epochs belonging to the target class [12]:

$$\mathbf{P1} = \frac{1}{|e|}\sum_{i \in e}\mathbf{X}_i \tag{3.7}$$

Next, $\tilde{\mathbf{X}}_i$ is constructed as follows:

$$\tilde{\mathbf{X}}_i = \begin{bmatrix}\mathbf{P1} \\ \mathbf{X}_i\end{bmatrix} \tag{3.8}$$

Note that the dimension of $\tilde{\mathbf{X}}_i$ is dependent on the number of classes for which the super trials are constructed. The single class case is shown in 3.8, yielding a resulting dimension of $\mathbb{R}^{(2\cdot p)\times n}$ [12]. If prototype responses were constructed for both classes, the resulting dimension would be $\mathbb{R}^{(3\cdot p)\times n}$, unless spatial filters are used [25].

Classification

Once the epoch based covariance matrices have been constructed from the super trials, the matrices, that can now be considered data points, can be classified. This can be done using both Riemannian geometry based classifiers and standard classification approaches that are created for classifying in the Eucledian space. Whereas the first mentioned can be applied on the manifold, the latter cannot, as they are defined using Euclidean distances [4]. Still, the standard classifiers can be used in the Riemannian pipeline, for they can be applied to the representations of the matrices that have been mapped to (Euclidean) tangent space.

As introduced in Section 2.3.1, the tangent space $\mathcal{T}_x\mathcal{M}$ to a manifold $\mathcal{M}$, is anchored at point x. In order to classify the data points, all points need to be projected to the same tangent space, that is, the tangent space that is tangent to the same x. The choice of this anchor is up to the researcher, and can for example be the (Riemannian) mean covariance matrix, an arbitrary data point $\mathbf{S}_i$, or the identity matrix [32], [3]. Once the anchor has been chosen, the logarithmic map (3.9), as defined by Barachant et al. [8], can be used to project the data points $\mathbf{S}_i$ to their representation on the tangent space. This mapping makes use of the matrix logarithm ($\mathbf{logm}(\mathbf{A})$) and the matrix square root ($\mathbf{A}^{\frac{1}{2}}$), that are not to be confused with the "normal" logarithm and square root, that can be applied to scalars. Both matrix flavours of the operations can be applied element-wise to the entries $\lambda_{i,i}$ of the diagonal[5] matrix $\mathbf{\Lambda}$, where $\mathbf{\Lambda}$ is

[5] A diagonal matrix is a sparse matrix where only the entries on the main diagonal are non-zero, examples of diagonal matrices are the identity matrix and the resulting matrix containing the eigenvalues after an eigenvalue decomposition [15, p. 60].

part of the decomposition of the diagonizable matrix $\mathbf{A} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^{-1}$ [4], [32]. After the operation on $\lambda_{i,i}$, yielding $\boldsymbol{\Lambda}'$, $\mathbf{logm}(\mathbf{A})$ or $\mathbf{A}^{\frac{1}{2}}$ is obtained by constructing the new value for $\mathbf{A}$ from the new diagnoal matrix $\boldsymbol{\Lambda}'$: $\mathbf{A}' = \mathbf{V}\boldsymbol{\Lambda}'\mathbf{V}^{-1}$ [32].
The mapping to $\mathcal{T}_{\mathbf{C}}\mathcal{M}$, as defined below, maps the data points $\mathbf{S}_i \in \mathbb{R}^{p\times p}$ to feature vectors $\mathbf{s}_i \in \mathbb{R}^{(p\cdot\frac{(p+1)}{2})\times 1}$ on the tangent space to $\mathcal{M}$ at $\mathbf{C}$ [4], [10], [32].

$$\mathbf{s}_i = \mathbf{C}^{\frac{1}{2}}\mathbf{logm}\left(\mathbf{C}^{-\frac{1}{2}}\mathbf{S}_i\mathbf{C}^{-\frac{1}{2}}\right)\mathbf{C}^{\frac{1}{2}} \tag{3.9}$$

Just like the logarithmic map is a mapping from the manifold to the tangent space, there also exists an exponential map, that maps the data from the tangent space to the manifold. This mapping is similar to the logarithmic map, only the matrix logarithm is exchanged for the matrix exponentiation [32]. In the methods I will only use the logarithmic mapping as defined in equation (3.9).

3.2 Improved Covariance Estimation Techniques

3.2.1 Shrinkage Regularization

In the preliminaries, the (disappointing) quality of the SCM has been discussed. Now, when calculating the SCM for only for a single time window, that has fewer samples than the full data set, the results may be even worse! Fortunately, a better estimator of the population covariance matrix is proposed by Ledoit & Wolf in 2004 [6]. Their estimator, $\mathbf{S}^*$, that is shown in equation (3.10), is the weighted average of the SCM ($\mathbf{S}$) and a so-called structured estimator ($v\boldsymbol{I}$). The weights are determined by the shrinkage parameter $\rho \in [0, 1]$. To make the structured estimator well-conditioned, Ledoit & Wolf constrained the covariance matrix in such way that all the variances are equal, and all the covariances are 0. From the definition of the SCM (2.2) it follows that an element of the matrix represents the variance if and only if it is located on the main diagonal. Thus, the structured estimator, as introduced by Ledoit & Wolf, is a diagonal matrix and can be represented by the identity matrix, $\boldsymbol{I}$, multiplied by a single value, v.

$$\mathbf{S}^* = \rho v\boldsymbol{I} + (1-\rho)\mathbf{S} \tag{3.10}$$

$\mathbf{S}^*$ is introduced to find an estimator that minimizes the mean squared error (MSE) given the estimator and the true covariance matrix. The MSE of estimator is defined in [6] as the expected value of the squared loss, shown in equation (3.11).

$$\mathrm{MSE}(\mathbf{S}^*) = E\left[\|\mathbf{S}^* - \boldsymbol{\Sigma}\|_F^2\right] \tag{3.11}$$

Ledoit & Wolf calculate in the sense of the Frobenius norm[6], a matrix norm that maps a matrix $\mathbf{A}$ to a measure of the size of its elements $a_{i,j}$ [15, p. 61]:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=0}^{m}\sum_{j=0}^{n} a_{i,j}^2} = \sqrt{tr\left(\mathbf{A}\overline{\mathbf{A}^T}\right)} \tag{3.12}$$

For a square matrix, such as covariance matrices, it holds that $m = n$.

[6] In [6], the Frobenius norm is divided by $\sqrt{p}$, this is done so that the norm of the identity matrix is 1.

In (3.12), the bar denotes the complex conjugate[7] Given the definition of the Frobenius norm, I now have the tools to decompose the MSE in (3.11) into the variance and the squared bias:

$$\begin{aligned} E\left[\|\mathbf{S}^* - \mathbf{\Sigma}\|_F^2\right] &= E\left[\|\mathbf{S}^*\|_F^2\right] - 2E\left[\langle \mathbf{\Sigma}, \mathbf{S}^* \rangle_F\right] + \|\mathbf{\Sigma}\|_F^2 \\ &= E\left[\|\mathbf{S}^*\|_F^2\right] - \underbrace{E\left[\|\mathbf{S}^*\|_F\right]^2 + E\left[\|\mathbf{S}^*\|_F\right]^2}_{0} - 2E\left[\langle \mathbf{\Sigma}, \mathbf{S}^* \rangle_F\right] + \|\mathbf{\Sigma}\|_F^2 \\ &= E\left[\|\mathbf{S}^*\|_F^2\right] - E\left[\|\mathbf{S}^*\|_F\right]^2 + \left(E\left[\|\mathbf{S}^*\|_F\right] - \|\mathbf{\Sigma}\|_F\right)^2 \\ &= \mathrm{Var}(\|\mathbf{S}^*\|_F) + \mathrm{Bias}^2(\|\mathbf{S}^*\|_F, \|\mathbf{\Sigma}\|_F) \end{aligned} \tag{3.13}$$

In (3.13), $\langle \mathbf{A}, \mathbf{B} \rangle_F$ is the Frobenius inner product between the matrices $\mathbf{A}$ & $\mathbf{B}$, yielding $tr(\mathbf{A}\overline{\mathbf{B}^T})$. Note that in the first step of the decomposition, the two inner products $\langle \mathbf{\Sigma}, \mathbf{S}^* \rangle_F$ and $\langle \mathbf{S}^*, \mathbf{\Sigma} \rangle_F$ are combined. Even though the matrix product is not commutative, it holds that $tr(\mathbf{AB}) = tr(\mathbf{BA})$ [15, p. 6].

Given this decomposition, Ledoit & Wolf present the weighted average in equation (3.10) as the best trade-off between the bias ($v\boldsymbol{I}$) and the variance ($\mathbf{S}$), because it is determined by minimizing the MSE [6]. By minimizing equation (3.11), the value for v was found to be equal to $\frac{tr(\mathbf{\Sigma})}{p}$: the inner product of the Frobenius norm[6]. Naturally, $\mathbf{\Sigma}$ is not known. But, according to Ledoit & Wolf, v can be consistently estimated using $\mathbf{S}$ [6].

The shrinkage parameter ρ determines the best weighted average in $\mathbf{S}^*$ of to the error in the (squared) bias and the error in the variance (equation (3.13)). In their paper, Ledoit & Wolf derived the optimal value for the shrinkage parameter, that is based on the sample covariance matrix, the data, $\mathbf{X} \in \mathbb{R}^{p \times n}$ and the structured estimator $v\boldsymbol{I}$ [6]:

$$\rho = \frac{\frac{1}{n^2} \sum_{k=1}^{n} \|\mathbf{x}_k \mathbf{x}_k^T - \mathbf{S}\|_F^2}{\|\mathbf{S} - v\boldsymbol{I}\|_F^2} \tag{3.14}$$

Here, the vector $\mathbf{x}_k \in \mathbb{R}^{p \times 1}$ indicates the kth column of the data matrix $\mathbf{X}$.
Hypothetically, one would like to base the value of the shrinkage parameter ρ on the deviation of the sample covariance matrix $\mathbf{S}$ from the population covariance matrix $\mathbf{\Sigma}$. If the difference is large, one would prefer to shrink more, whereas the value of the shrinkage parameter ought to be low if $\mathbf{S}$ and $\mathbf{\Sigma}$ are much alike. Yet, again we need to work with what we have: merely the data and the sample covariance matrix are to our disposal. Therefore, Ledoit & Wolf propose to estimate the difference between $\mathbf{S}$ and $\mathbf{\Sigma}$ with the individual matrices $\mathbf{x}_k \mathbf{x}_k^T$ that are assumed to be independent and identically distributed (iid) [6]. The basis of this proposal follows from (2.2): the $\mathbf{S}$ is the average of the individual matrices, thus, the difference $\mathbf{S}$ and $\mathbf{\Sigma}$ is approximated with how much each individual matrix differs from the average ($\mathbf{x}_k \mathbf{x}_k^T - \mathbf{S}$). The denominator does not function fully as a normalization term. To ensure that $\rho \in [0, 1]$, the result of the fraction is truncated to that interval. [6]
The denominator of equation 3.14 ensures that the value of the shrinkage parameter decreases when the difference between the SCM $\mathbf{S}$ and the structured estimator $v\boldsymbol{I}$ is large. In other words: if the shrinkage target, $v\boldsymbol{I}$, is not accurate, $\mathbf{S}^*$ will be mainly based on $\mathbf{S}$, as ρ is small [39].

In Figure 3.1, the intensity of the shrinkage parameter is shown for the estimated covariance matrices given different numbers of samples. The number of samples are, similarly to Figure 2.1 drawn from a multivariate normal distribution that is parameterized with the same population covariance matrix $\mathbf{\Sigma}$ as in Figure 3.1. To conclude this section with what I started with: the sample covariance matrix is an sub-optimal estimator if the number of samples is small. The

[7] The complex conjugate of $z = a + bi$, often denoted by $\overline{z}$ or z^*, is $\overline{z} = a - bi$. In the case of real numbers, where the imaginary part is absent, $z = \overline{z}$. The complex conjugate of a matrix $\mathbf{A}$, is the matrix $\overline{\mathbf{A}}$ where all elements $a_{i,j}$ are conjugated.

value of the shrinkage parameter is relatively high in these cases (top window), so that the structured estimator $v\boldsymbol{I}$ has a larger impact on $\mathbf{S}^*$ than the sample covariance matrix $\mathbf{S}$. When the number of samples increases, the SCM becomes a better and better estimator. This is shown in the lower window, where the MSE is depicted. As a consequence, weights in the improved estimator $\mathbf{S}^*$ shift more towards $\mathbf{S}$ as the shrinkage intensity decreases. The larger the number of samples becomes, the more similar $\mathbf{S}^*$ will become to $\mathbf{S}$.

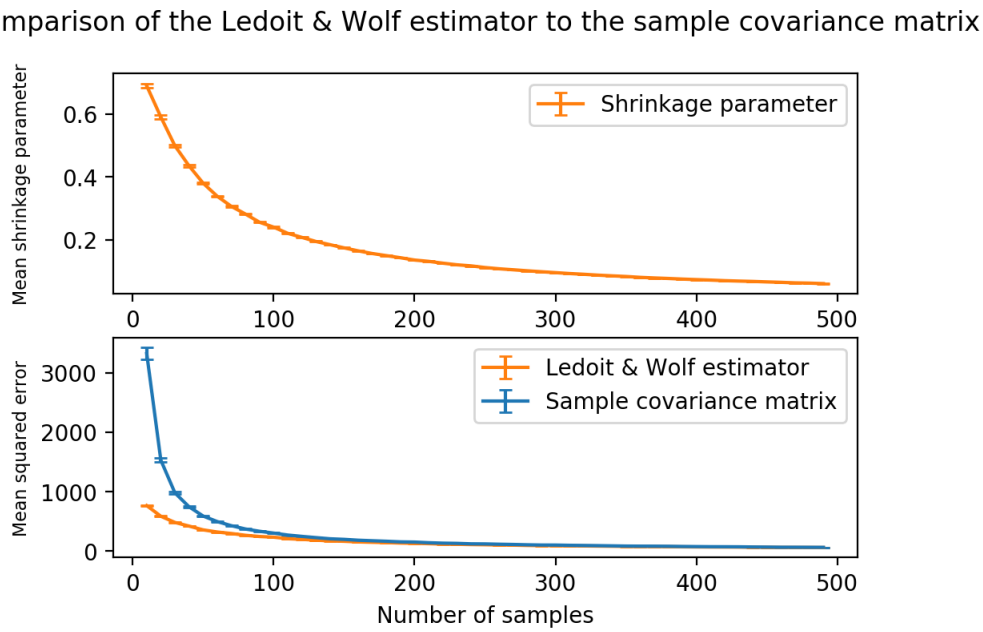


Figure 3.1: In the top window, the mean intensity of the shrinkage parameter according to equation (3.14) is shown for different number of samples ($\mathbf{X} \sim \mathcal{N}(0, \mathbf{\Sigma})$). The number of features of the data has been set to 31. The population covariance matrix, $\mathbf{\Sigma}$, has been sampled from a univariate Gaussian distribution with a mean of 0 and a standard deviation of 1, $\mathcal{N}(0, 1)$. The shrinkage intensity is averaged over 100 iterations and it is plotted including the standard error of the mean (SEM). In the bottom window, the MSE is shown for the SCM and the Ledoit & Wolf estimator, using the shrinkage parameter that is plotted above. Similar to the top window, the mean values are averaged over 100 iterations, and the SEM is shown using the error bars.

3.2.2 Time-decoupled LDA

Another way to cope with small training datasets, that typically not lead to a well-conditioned SCM [6], is introduced by Sosulski, Kemmer & Tangermann [3]. Their method, that is labelled time-decoupled LDA, aims to improve the covariance matrix by refining the within-channel covariance matrices at a specific time interval [3]. In contrast to the covariance matrix that is introduced in Section 3.1.2 that is based on a single epoch, the covariance matrix that is considered in this paper is the SCM of all epochs combined.

In mathematical sense, recall the structure of the data, $\mathbf{X}_i \in \mathbb{R}^{p \times n}$, that could be seen as e recordings of a single epoch i. If one considers all the epochs, then one can flatten the channel-timepoint structure, yielding a covariance matrix of the shape $p \cdot n \times e$. As a result, the structure of the resulting covariance matrix is equal to $p \cdot n \times p \cdot n$. Hence, this covariance matrix contains a temporal structure, for it describes the (co)variance between the p channels at each time point n [3].

At this point, Sosulski et al. [3] make the assumptions that the noise structure, that is incorporated in the data is normally distributed and unrelated to the paradigm. These assumptions are vital to their method, as it allows for the time-decoupling: If the premises are met, then one

can argue that the between channel covariance matrices (of size $p \times p$ for each of the timepoints $\in n$ is equal. Thus, rather than calculating these matrices one by one, one can estimate a single within channel covariance matrix, that is based on the data of the within channel matrices for all timepoints. Next, all the within channel covariance matrices, that are located on the main diagonal of the larger covariance matrix, can be replaced by the covariance matrix that has been estimated from all the pooled data [3].

Initially, I wanted to include time-decoupled LDA in my research as a improved covariance estimation technique. There are two moments in the pipeline that I considered for the application of the method: the construction of the epoch-based covariance matrices, and the application of LDA in the tangent space. However, the application of time-decoupled LDA turned out to be unfeasible for both options.

(1) The temporal structure of the covariance matrix that is estimated over all epochs, that materializes because of the different number of epochs e that are considered, is absent in the epoch-based covariance matrix. Hence, I could not identify submatrices on the main diagonal of the epoch-based covariance matrix for which the two assumptions, that are made by the authors of [3], hold.

(2) In the tangent space the temporal structure induced by the timepoints is lacking. Recall from Section 3.1.2 that the epoch-based covariance matrices are represented by their feature vectors in the tangent space (equation (3.9)). When LDA is applied, the between and within class covariance matrices are computed from these feature vectors. If the feature vector is in $\mathbb{R}^f$, then it follows from equations (2.7) and (2.8) it follows that between and within class covariance matrices are in $\mathbb{R}^{f \times f}$. Again, the temporal structure from the covariance matrix that is estimated from all epochs is missing here. It did not seem sensible to select arbitrary submatrices on the main diagonal of the matrices, as the two assumptions mentioned above are not likely to hold in that case.

Even though the application of time-decoupled LDA does not seem appropriate in the Riemannian pipeline, the paper by Sosulski et al. [3] is valuable for this work. In their evaluations, they compare their method against a Riemannian classification framework. The three pipelines that I will introduce in the next chapter are based on the pipeline that has been introduced in this paper. In addition, the paper relates to this work in terms of the evaluation of the pipelines. In both approaches, the same benchmarking algorithm is used. Moreover, the methods in this thesis are evaluated on a subset of the datasets that is used in [3].

Chapter 4

Methods

Purpose of the Chapter

In this chapter, I will reflect on the research methodology that I used to arrive to the conclusions presented in chapter 7. I devote the first section to the mother of all BCI benchmarks (MOABB). This is an open source project that aims among others to enhance the reproducibility within the field of BCI [40]. Moreover, the project contains handles to datasets that have been made publicly available. I have used all datasets from the MOABB that were recorded during a P300 paradigm, in addition to the SPOT dataset. The datasets are described in Section 4.2.
In the last section, 4.3, I introduce the three pipelines that I constructed to find an answer to my research question. This section relies heavily on the previous two chapters, where I acquainted the reader with the concepts that are used in the classification procedures.

4.1 Mother of all BCI Benchmarks

The mother of all BCI benchmarks is an open source project that is founded by Alexandre Barachant [40]. As the name suggests, the MOABB has been initiated to provide benchmark algorithms for EEG datasets. This ought to make the field of BCI more transparent and it should enhance the reproducibility of BCI experiments [40].

The framework of the MOABB is based on 4 concepts: the *datasets*, on which I will elaborate in Section 4.2; the *pipelines*, discussed in Section 4.3; the *paradigm* and the *evaluation*, that both are the subjects of Section 4.4.

4.2 Data

The datasets that I have used, have been recorded under different paradigms where the common divisor of the paradigms is the elicitation of a P300 response. Hence, all sets reflect EEG recordings of event-related potentials. A summary of datasets including their paradigms, number of subjects and hardware specifications is listed in Table 4.1. In the remainder of this section I provide a brief overview of the datasets, subjects and corresponding paradigms.

4.2.1 SPOT

The SPOT dataset has been recorded during an auditory oddball paradigm. In this paradigm, 90 tones have been presented to the participant. Of those 90 tones, the 75 non-target tones were low pitched (500 Hz), whereas the 15 target tones were high pitched (1000 Hz). Both types of tones lasted for 40 ms. Together, the 90 tones form a single trial.
In total, the data of 13 subjects have been recorded, where each subject performed 50-70 trials. Within these trials, the stimulus onset asynchrony differed, but each epoch was windowed to 1 second. The data has been recorded with a total of 37 EEG channels, of which 31 channels were located on the scalp. The sampling rate was equal to 1000 Hz [18]. While working with

Dataset	Subjects	Paradigm	Channels	Epochs per session (μ)	Window (s)	Reference
SPOT	13	Auditory oddball	31	90	$[0, 1]$	[18]
Braininvaders	24	Visual, flashes	16	480	$[0, 1]$	[41]
EPFL	9	Visual, 6 flashing images	32	833	$[0, 1]$	[42]
BNCI_1	10	P300 speller	16	96	$[0, 0.8]$	[43]
BNCI_2	10	P300 speller	8	2520	$[0, 0.8]$	[44]
BNCI_ALS	8	P300 speller	8	4200	$[0, 1]$	[45]

Table 4.1: An overview of the 6 datasets that have been evaluated. All datasets, with the SPOT dataset as exception, have been listed under the MOABB datasets. The channels that are listed indicate the scalp channels. Moreover, the original recording sampling rate has not been added to this table, as the data has been downsampled to 100 Hz during the analyses. The number of epochs per session can differ per participant. Hence, the number presented in the table is an average. Lastly, the window indicates the time window that is used to epoch the data. The elements of the window represent the starting and finishing time of the window with respect to the stimulus.

the SPOT dataset, I enabled the virtualization of the datasets. As a result, I classified the data per trial containing 90 epochs, rather than classifying all epochs that have the same SOA. This approach yields more samples, which can give more statistical power when a paired test is considered.

4.2.2 Braininvaders

The Braininvaders paradigm is visual. 12 flashes were presented to the 24 participants of the experiment during a single trial. Of these flashes, 2 flashes were considered targets, and 10 flashes were non-target stimuli. The first 7 subjects participated in 8 sessions, while the remaining subjects participated in a single session only. The data has been recorded with 16 electrodes, using a sampling rate of 512 Hz [41].

4.2.3 EPFL

While recording this dataset, 6 different images were presented in random order to the participants. The time between the stimuli is equal to 400 ms. 9 subjects participated in this study, of which the first 5 subjects were bound to a wheelchair, suffering from various communication and limb movement control disabilities. The data has been recorded with 32 electrodes, using a sampling rate of 2048 Hz [42].

4.2.4 BNCI_1

The BNCI_1 dataset has been recorded under the P300 speller paradigm. In a nutshell, this paradigm requires the subjects to spell words by gazing at characters that are presented on a computer screen. The data has been recorded by healthy 10 participants, using 16 electrodes with a sampling rate of 256 Hz [43].

4.2.5 BNCI_2

For this dataset, again the P300 speller paradigm was used to record the P300 responses of 10 participants. In contrast to the BNCI_1 datasets, only 8 electrodes were used. The sampling frequency was 256 Hz [44].

4.2.6 BNCI_ALS

The setup of the paradigm of the BNCI_als dataset is equal to the setups of BNCI_1 and BNCI_2. However, the data have been recorded with 8 subjects that are diagnosed with amyotrophic

lateral sclerosis (ALS). The brain activity has been recorded using 8 channels, with a sampling rate of 256 Hz [45].

4.3 Classification Pipelines

Regarding the classification pipelines, I am not so much interested in achieving the best performance. Rather, the aim is to improve the relative performance of the proposed methods compared to a baseline method. The baseline pipeline forms the raw basis of the two proposed improvements and is referred to as tangent space LDA. In this section, I first lay out the common basis of the three pipelines. This basis includes the data preprocessing, feature extraction and augmentation steps. While assessing the relative performance, it is vital to keep the basis consistent between the pipelines. In contrast, I did not focus on finding the optimal set of hyperparameters because of the same reason. Hence, I use the hyperparameters that Sosulski et al. [3] have identified as the optimal ones. The code can be found in my repository.

4.3.1 Preprocessing

The preprocessing steps are handled by the *paradigm* class of the MOABB. Here, the `MNE-Python` library, that is an open source Python package for working with EEG data, is used to convert the recordings to the raw data objects. [22].

First, the data has been filtered using a forward and backward infinite impulse response (IIR) filter, in the range from 0.5 to 16 Hz. Next, the data was downsampled to 100 Hz and split in single epochs. The length of the epochs is described by the window length, that is shown in Table 4.1. The window length differs per dataset. Thus, I will take the SPOT dataset as an example for the sake of the explanation of the preprocessing steps. In this dataset the epoch starts directly when the stimulus is presented, and ends one second thereafter. In combination with the sampling rate, this yields epochs of 100 samples. Thus, a single trial can be seen as a matrix containing 90 epochs (one for each tone), where each epoch has got 31 features (the EEG channels) and lasts for 100 time points:

$$\mathbf{X} \in \mathbb{R}^{90\times31\times100} \tag{4.1}$$

There was no baseline correction applied to the data, following the example of [3].

4.3.2 Spatial Filtering & Covariance estimation

The next step in the pipeline is Spatial filtering. Recall from Section 3.1.2, that the data should be enhanced with the prototype response of a single class, or both classes, to preserve the temporal structure. In this case, only the prototype of the target class is used, as proposed by Sosulski et al.. In their work, they state that the classification performance improves when only prototypes are created from the target class, rather than from both classes or merely the non-target class [3].

Both the data, $\mathbf{X}$, and the prototype are spatially filtered prior to the concatenation over the first axis to create the super trials. The aim of the spatial filters, $\mathbf{W}$, is to increase the signal to noise ratio of a single class. The algorithm to find filters has been proposed by Barachant, and is a variant on the xDAWN algorithm. The quest to find the optimal spatial filters boils down to an eigenvalue composition [25]. In this work, the spatial filters with the 5 best eigenvalues have been chosen to compose $\mathbf{W}$, yielding the final filter matrix $\mathbf{W} \in \mathbb{R}^{31\times5}$. The super trials are constructed per epoch. To formalize their construction, I again decompose the data of one session $\mathbf{X}$ into the smaller matrices representing a single epoch $\mathbf{X}_i \in \mathbb{R}^{31\times100}$, for each

$i \in \{1, ..., 90\}$: the range of the number of epochs [25].

$$\tilde{\mathbf{X}}_i = \begin{bmatrix} \mathbf{W}^T\mathbf{P1} \\ \mathbf{W}^T\mathbf{X}_i \end{bmatrix} \in \mathbb{R}^{2\cdot 5\times 100} \tag{4.2}$$

In the definition above, **P1** is the prototyped response of the target class, introduced in Section 3.1.2. The filters **W** are specific for each class [25].

Once the super trials have been constructed, the epoch based covariance matrix, $\tilde{\mathbf{S}}_i$, can be estimated using the definition of the SCM, in equation (3.6). The dimension of the resulting matrix is 10×10. Moreover, the covariance matrix can be split into 3 distinct submatrices by selecting a subset of the rows and columns in $\tilde{\mathbf{S}}_i$. To illustrate this, let me first define $\tilde{\mathbf{S}}_i$ itself, as it is presented in [12]:

$$\tilde{\mathbf{S}}_i = \begin{bmatrix} \mathbf{S1} & \mathbf{C}^T_{\mathbf{P1},\mathbf{X_i}} \\ \mathbf{C}_{\mathbf{P1},\mathbf{X_i}} & \mathbf{S}_i \end{bmatrix} \in \mathbb{R}^{10\times 10} \tag{4.3}$$

In this formalization, the submatrix $\mathbf{S1} \in \mathbb{R}^{5\times 5}$ represents the spatial covariance matrix of **P1**. This submatrix is the same for all $\tilde{\mathbf{S}}_i$, as all signals are augmented with the same prototype response **P1**, and filtered with the class specific filters **W**. Next, the submatrix $\mathbf{S}_i \in \mathbb{R}^{5\times 5}$ is the spatial covariance matrix of the response. Thus, if the data were not augmented with the prototype response, the resulting covariance matrix would be merely $\mathbf{S}_i$. Next, Barachant et al. [12] identified the last submatrix as the cross covariance matrix $\mathbf{C}_{\mathbf{P1},\mathbf{X_i}} \in \mathbb{R}^{5\times 5}$ [12]. Note that this matrix, that is the result of the multiplication $\mathbf{X}_i\mathbf{P1}^T$, is not symmetric[8]. This is in contradiction with the matrices **S1** and $\mathbf{S}_i$. According to Barachant et al. [12], the cross covariance matrix is of the highest importance when classifying the responses [12]. In this particular case, where the prototyped response corresponds to the target class, the cross covariance between the epoch $\mathbf{X}_i$ and the prototype will be high, as both matrices describe the same type of response. In contrast, the cross covariance is likely to be low when the two matrices describe an opposite response, which is the case when $\mathbf{X}_i$ has been collected during an epoch with a non-target label [12].
Barachant et al. have developed an open source python library, `PyRiemann`, that accompanies their work. This library provides funtions to construct the super trials with spatial filtering, that I have used in my implementation.

4.3.3 Tangent Space LDA

Having introduced the first steps (1 to 4 in Figure 4.1) of the Riemannian pipeline, I continue with the implementations of the proposed methods because the three pipelines now diverge. As discussed in Section 3.1, the epoch based covariance matrices act like single data points in the pipeline after having calculated the covariance matrices of the super trial, $\tilde{\mathbf{S}}_i$. Now, recall that these data points can be classified with either a Riemannian or an Euclidean classifier. As depicted in Figure 4.1, I decided to deploy an Euclidean classifier on the tangent space. The Euclidean classifiers are more advanced than their Riemannian counterparts [8]. It was reasoned that, if the improvement of the epoch based covariance matrices bring about any improvement in the classification score, then this is possibly easier to detect when a more advanced classifier is used.

[8]The multiplication $\mathbf{P1}\mathbf{X}_i^T$ yields the submatrix in the top right: $\mathbf{C}^T_{\mathbf{P1},\mathbf{X_i}}$.

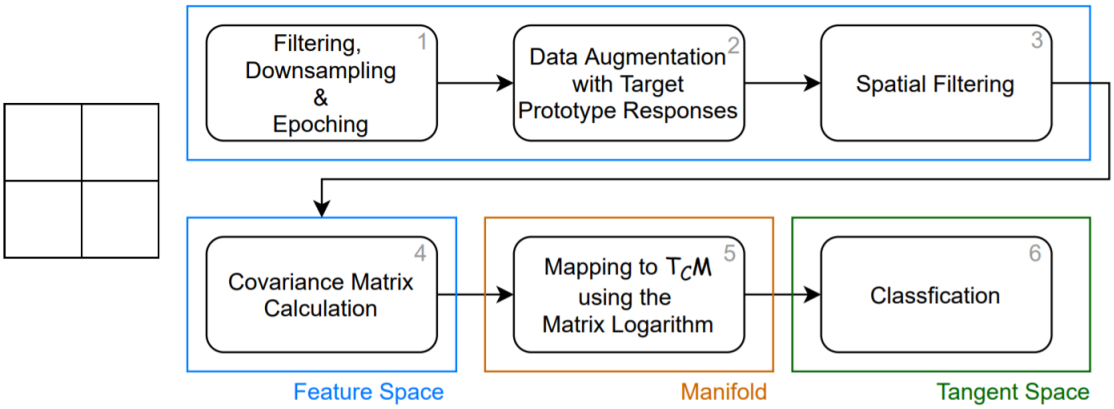


Figure 4.1: The schematic representation of the standard pipeline. The proposed pipelines are instances of this general formulation. In the figure, the different locations are marked as well. In this case, *feature space* is used to describe the Euclidean space wherein the standard classification pipeline acts. The reference of the tangent space $\mathcal{T}_C\mathcal{M}$ is the Riemannian mean of all data points. LDA has been used as classification algorithm.

As a consequence, the epoch based covariance matrices $\tilde{\mathbf{S}}_i$ are mapped to their representation on the tangent space. The Riemannian mean of all matrices $\tilde{\mathbf{S}}_i$ has been chosen as the anchor of the tangent space (step 5 in Figure 4.1). The mapping has been performed using the logarithmic map, described in equation 3.9 in Section 3.1.2. After projection, each of the 90 matrices is represented as a vector with $\frac{10(10+1)}{2} = 55$ entries, that is ready to be classified.

The algorithm that has been selected for the last step is LDA; as introduced in Section 2.2.2. In particular, LDA has been chosen as this allowed for the possibility to apply Time-Decoupled LDA in the tangent space, as described in Section 3.2.2. Unfortunately, the application of the algorithm was not possible, as described in the same section as linked before. The particular LDA algorithm that has been used, comes from the `scikit-learn` library [46]. This specific algorithm finds the optimal decision boundary by minimizing the sum of squares of the residuals, which, in other words, aims to get the predictions of the model as close as possible to the target values [16, p. 189]. As residuals typically belong to a regression problem, the target values are defined as the prior probability of the class, which is the number of data points belonging to that class, divided by the total number of data points [16, p. 190]. According to Bishop, the Fisher criterion, as discussed in 2.2.2, is an instance of the least squares solver when binary class labels are considered [16, p. 189]. The baseline pipeline includes shrinkage of the within-class covariance matrix of the LDA. Even though this is already an "improvement", this is considered common practice [2], therefore, I decided to regard this as baseline, and not as proposed improvement.

4.3.4 Response Shrinkage

The response shrinkage pipeline aims to improve the classification score by adding a shrinkage step to the vanilla pipeline described in the previous section. Recall the covariance matrices $\tilde{\mathbf{S}}_i$ that have been formalized in equation 4.3. Out of the submatrices $\mathbf{S1}$, $\mathbf{C_{P1,X_i}}$ & $\mathbf{S}_i$, the covariance matrix of the response, $\mathbf{S}_i$, has been selected as a subject for the application of shrinkage regularization. As a result, the pipeline that is shown in Figure 4.1, can be expanded to the pipeline shown in the figure below.

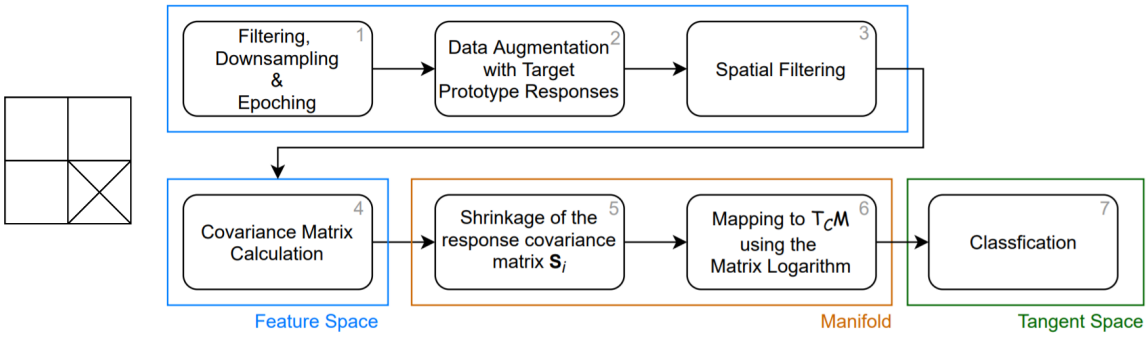


Figure 4.2: The schematic representation of the response shrinkage pipeline. This is an extension of the standard pipeline, where a shrinkage step (5) has been a added.

This method can be described as follows: Ledoit-Wolf shrinkage (section 3.2.1) is merely applied to the matrices $\mathbf{S}_i \subset \tilde{\mathbf{S}}_i$ from equation (4.3), that are located on the bottom right. As the original data are used in the calculation of the shrinkage parameter, only the data that correspond to $\mathbf{W}^T\mathbf{X}_i$ have been extracted from the super trial $\tilde{\mathbf{X}}_i$ (equation (4.2)). Prior to the application of shrinkage, the data is z-scored. The effect of the transformation is shown in the figures 4.4 & 4.5. In the figures, the left panel depicts the original covariance matrix, $\tilde{\mathbf{S}}_i$, where no shrinkage is applied. This definition is used in the tangent space LDA pipeline. In the middle the matrix that is used in the response shrinkage pipeline is shown. On the right, the matrix from the super trial shrinkage pipeline, that will be discusses in the next section, is described.

The rationale behind this pipeline is based on the fact that the response covariance matrix is a sub-optimal estimator of the true covariance matrix. This is because the number of samples, on which this matrix is based, is rather small as the matrix is based on a single epoch only. The length of the epochs differs from 80 to 100 samples for the datasets that were considered[9]. Thus, the ratio of p to n, as used in the analysis in Section 2.1.2, ranges from $\frac{5}{80}$ to $\frac{5}{100}$. The application of Ledoit-Wolf shrinkage on the epoch based covariance matrices could be beneficial for the classfication performance as the matrices become a better estimator[10]. When the epoch-based covariance matrices are a better representation of the response, the classification of the responses might improve compared to the situation where the epoch-based matrices are worse representations.
Both the methods for the Ledoit-Wolf shrinkage and the z-scoring come from the `scikit-learn` python library [46].

4.3.5 Super Trial Shrinkage

The second proposed improvement is embodied by the super trial shrinkage pipeline. This pipeline is in fact very similar to the pipeline that has been discussed in the previous section. The difference between the two pipelines is that out of the submatrices from $\tilde{\mathbf{S}}_i$, Ledoit-Wolf shrinkage has not only been applied to $\mathbf{S}_i$, but also to $\mathbf{S1}$. The new pipeline is depicted in Figure 4.3.

[9]The length of the epoch is dependent on the time window of the epoch, and the sampling rate. The time windows for the datasets are shown in Table 4.1, and the sampling rate that has been used for all sets is 100 Hz.
[10]Ledoit-Wolf shrinkage is defined to minimize the mean squared error, thus, this statement holds in terms of the MSE.

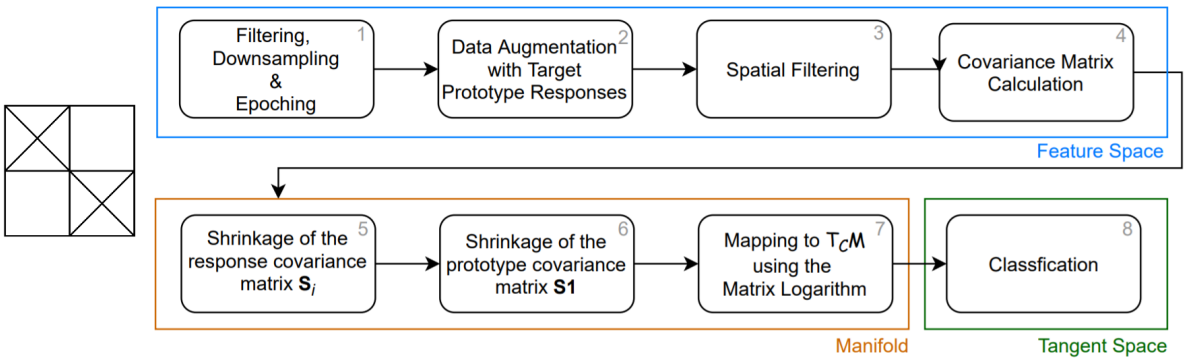


Figure 4.3: The schematic representation of the super trial shrinkage pipeline. This is an extension of the standard pipeline, where a shrinkage step of the response covariance matrix (5) and a shrinkage step of the prototype covariance matrix have been added.

Again, shrinkage regularization has been applied to the submatrices only. Note that the regularization and z-scoring has been applied to the matrices separately.

The idea behind this improvement is comparable to the one described in the previous section. Even though the prototype is based on all data points of a class, these individual data points remain sub-optimal estimators. Hence, there could be a benefit of shrinking the prototype covariance matrix, next to the response covariance matrix.

Below, two examples of the epoch based covariance matrices from the three different pipelines are shown. Both epochs are taken from the first trial in the first session of the first subject of the SPOT dataset. In total, 12 target responses have been averaged to from the prototype **P1**. The corresponding SOA is 226 ms. The figures are placed here to give an impression. Larger annotated versions of the figures are listed in Appendix A.

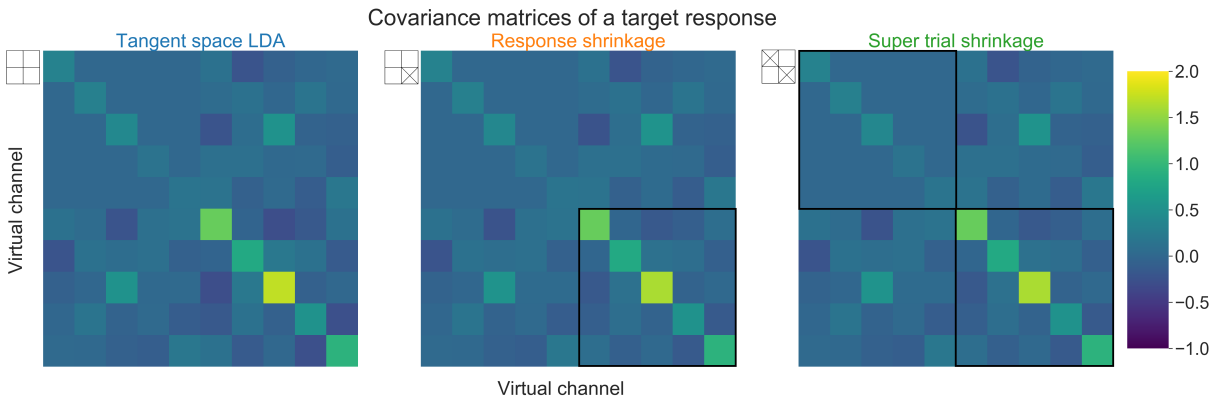


Figure 4.4: Examples of the epoch based covariance matrices that correspond to the 3 pipelines. The matrices describe a target response. On the left, the matrix from tangent space LDA is shown, where no shrinkage is applied. Shrinkage has been applied to the lower right submatrix of the matrix in the middle panel, the matrix from the response shrinkage pipeline. Lastly, on the right, the matrix from the super trial shrinkage pipeline is shown. It is remarkable the upper left submatrix of the super trial shrinkage matrix does not differ from the other two matrices. This will be more detailed in the discussion chapter. The superimposed squares indicate where the matrix has been improved.

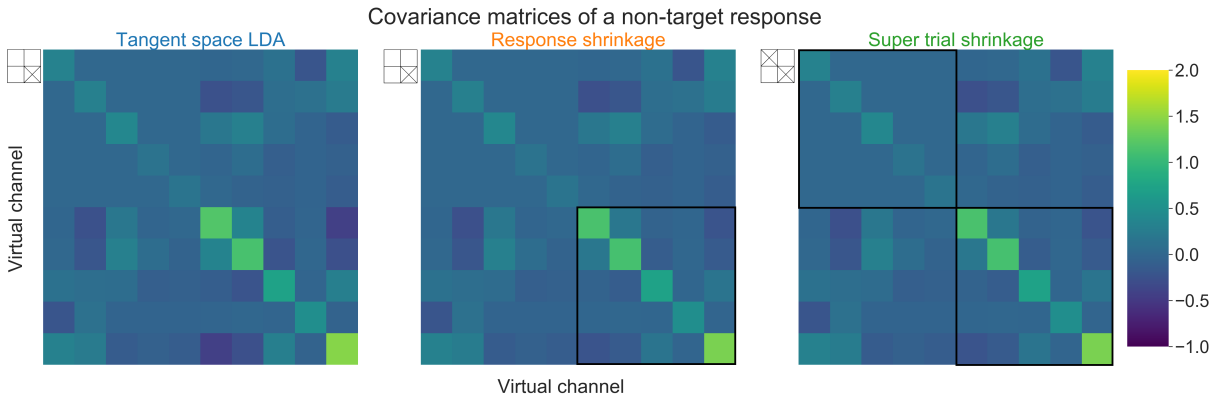


Figure 4.5: Examples of the epoch based covariance matrices that correspond to the 3 pipelines. The matrices describe a non-target response. Again, it seems that the shrinkage parameter for the upper left matrix of the super trial shrinkage pipeline was rather low. The superimposed squares indicate where the matrix has been improved.

Properties of the Shrunken Covariance Matrices

As described in the sections above: in both the response shrinkage and the super trial shrinkage piplines, shrinkage is not applied to the entire matrix, but to a submatrix. Because of this, the operation does not guarantee the SPDness of the resulting matrix.
Recall from Section 3.1.2 that only the matrices that are symmetric positive-definite (SPD) live on the manifold on which the Riemannian metrics are defined. The covariance matrix $\tilde{\mathbf{S}}_i$ is symmetric. This property is preserved when shrinkage is applied to the submatrices, as the matrices are located symmetrically around the diagonal of their main matrix. However, to be positive definite, all leading *principal minors* of $\tilde{\mathbf{S}}_i$ should have a strictly positive determinant. The *principle minors* are the submatrices that are anchored to the upper left of $\tilde{\mathbf{S}}_i$ [47]. This condition is known as Sylvester's criterion. Instead of going into the determinant of the principal minors, one could also say that all pivots of $\tilde{\mathbf{S}}_i$ should be strictly positive.
If shrinkage is only applied to the half of the columns of the epoch-based covariance matrices, then it can occur that some columns in $\tilde{\mathbf{S}}_i$ or in one of its principal minors can be described as linear combinations of others. As a consequence, the determinant of the affected (sub)matrix (or one of the pivots of $\tilde{\mathbf{S}}_i$) is 0, which implies that Sylvester's criterion is not met, meaning that $\tilde{\mathbf{S}}_i$ is not positive definite [15, p. 18].

If a matrix is not SPD, then it is not located on the manifold and it cannot be mapped to the tangent space [8]. This event does not happen often for most datasets, but if it occurs, the program catches the exception by replacing the shrunken non-SPD matrix by the original matrix, as used in tangent space LDA. To identify the matrices that are not SPD, I try to compute the Cholesky decomposition, as proposed in [48]. If the matrix is not positive definite, the function that comes from the `numpy` library and calculates the composition, raises an `LinAlgError` [49]. This method is faster than computing the eigenvalues of the matrix to see whether one of the eigenvalues is negative [50]. In Table 4.2 the percentage of the covariance matrices is shown that was not SPD after the application of shrinkage regularization to a submatrix. The percentage tends to differ between the different datasets. It becomes clear from the table that the percentages are generally lower for the response shrinkage pipeline. The latter observation is possibly due to the fact that in the corresponding pipeline shrinkage regularization is only applied once. This would result in a lower probability of obtaining a non-SPD matrix compared to the situation where both submatrices are shrunken.

Percentage of non-SPD covariance matrices of the pipelines

Dataset	Response shrinkage	Super trial shrinkage
SPOT	0.25%	0.25%
Braininvaders	0.13%	0.13%
EPFL	0.86%	0.86%
BNCI_1	0.25%	0.25%
BNCI_2	0.57%	0.57%
BNCI_als	0.06%	0.06%

Table 4.2: The percentage of covariance matrices per pipeline and dataset that was not SPD after the application of shrinkage regularization on the submatrices. The percentage of non-SPD matrices is higher for the super trial shrinkage pipeline than for the response shrinkage pipeline. The percentages for both pipelines are relatively high for the EPFL dataset. As a consequence, a substantial part of the matrices has been replaced by the non-shrunken variant that is used in tangent space LDA.

4.4 Evaluation

The evaluation happens entirely under the hood of the MOABB: it is handled by the two remaining pillars of the benchmark, the *paradigm* and the *evaluation* classes.
The *paradigm* loads and preprocesses the data. In addition, it specifies the scoring method. I used the `P300` paradigm as I am working with ERP data. The scoring method that goes with this paradigm is the area under the receiver operating characteristic (ROC) curve (ROC AUC) [40]. Each score is validated using 5-fold stratified cross-validation. In contrast to a single classification score, k-fold cross-validation creates k train and test splits to generate multiple scores. Eventually, the scores are averaged, so that the final score is more robust. The process is shown in Figure 4.6. As the class distribution is not balanced, stratified cross-validation is used to preserve the balance as much as possible in the distinct splits [46].

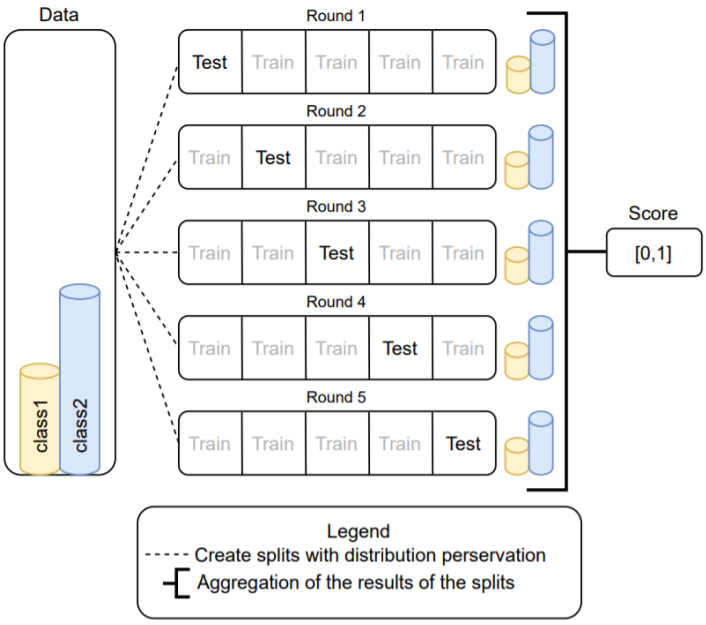


Figure 4.6: A schematic representation of stratified 5-fold cross-validation. The data is copied to 5 folds, where each of the folds has different train and test splits. Because stratified k-fold cross-validation is used, the aim of the algorithm is to keep the class distributions more or less equal.

The *evaluations* class contains all the details that are needed for the scoring. I chose the `WithinSessionEvaluation` to obtain a score for each session of each subject, which gave me more handles to inspect and reason about the results compared to the `CrossSessionEvaluation`, that only yields a single score per participant. The sessions are defined by the SOAs that correspond to the data: all responses with the same SOA are considered a single session. As mentioned in Section 4.2.1, the SPOT dataset is an exception to this notion. For this dataset the sessions have been split up into *virtual datasets*, that represent a single trial.

4.5 Statistical analysis

To find out whether a difference in performance between the three pipelines is not a result of chance, I have run statistical tests on the data. In particular, I have used the Wilcoxon signed rank test from the `scipy` python library [51], [52]. This test has been chosen as the scores are paired and the distribution of the scores is non-Gaussian for some datasets because of ceiling effects, as shown in figure 5.4. I used the two-sided test, as we have seen that the difference can swing in both ways: the proposed pipelines might perform better, or worse than the baseline.

For each dataset, the scores per subject served as samples for the test. Rather than using a fixed significance level of $\alpha = 0.05$, the Holm-Bonferroni correction was applied because a statistical test has been conducted for each dataset. This way, the probability of a false positive result is limited to approximately 0.05, regardless of the number statistical tests that has been carried out.

Chapter 5

Results

Purpose of the Chapter

I first present the classification scores of the pipelines given the different datasets. Next, I show the subject-wise results for a selection of the datasets. The subject-wise results for the remainder of the datasets can be found in appendix B.
I finalize by summarizing the results from the statistical tests that have been applied to the classification scores.

5.1 Classification Performance

As mentioned in the previous chapter, each (virtual) session of each dataset has been assigned an average ROC AUC score, that is based on a 5-fold stratified cross validation. To assess the performance of the three pipelines over the different datasets, the mean is taken over all subjects and sessions. The comparison is shown in figure 5.1. From the figure it seems that there is only a slight difference in the accuracy scores of the pipelines when the SPOT dataset is classified. The performances of the resonse shrinkage and super trial pipelines on the other datasets are very close to the performances of the tangent space LDA pipeline.

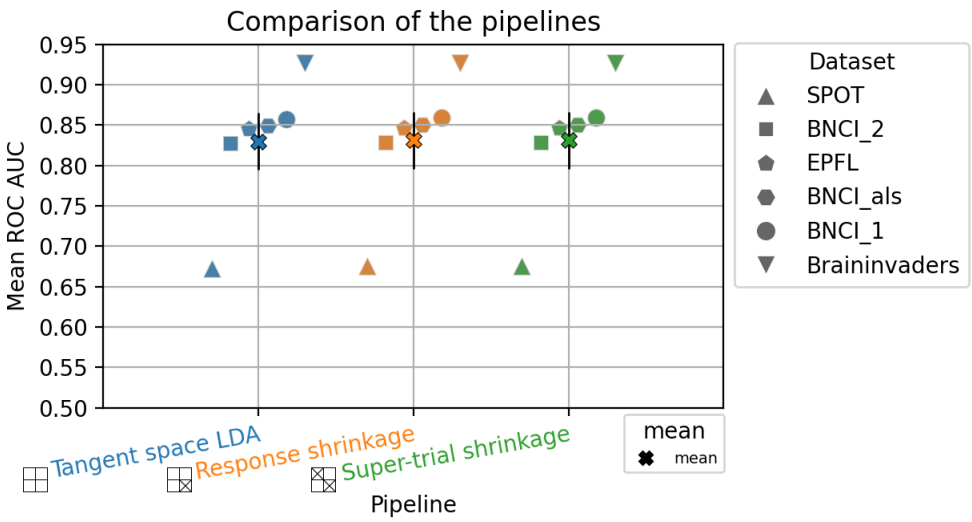


Figure 5.1: The comparison of the three pipelines over all datasets. To acquire the mean `ROC AUC` scores for each dataset, all sessions of all participants have been averaged. The mean of the datasets has been indicated with the cross marker. The errorbars indicate two standard errors around that mean (SEM). The datasets and the legend have been sorted in an ascending order in accordance with the score of the tangent space LDA pipeline. The mean scores over all datasets for the pipelines from left to right are 0.8230, 0.831 & 0.831.

These findings seem still valid when the scores are examined in more detail. In table 5.1, the mean scores `ROC AUC` of the three pipelines are presented, together with the corresponding SCM. To obtain the mean scores, the scores of the subjects per dataset have been averaged. The SCM is calculated over all participants.

The mean `ROC AUC` scores of the three pipelines

Dataset	*tangent space LDA*		*Response shrinkage*		*Super trial shrinkage*	
	μ	SEM	μ	SEM	μ	SEM
SPOT	0.6720	0.0231	0.6754	0.0230	0.6754	0.0230
Braininvaders	0.9333	0.0106	0.9329	0.0105	0.9329	0.0105
EPFL	0.8456	0.0264	0.8465	0.0267	0.8465	0.0267
BNCI_1	0.8570	0.0176	0.8589	0.0172	0.8589	0.0172
BNCI_2	0.8273	0.0224	0.8283	0.0227	0.8283	0.0227
BNCI_als	0.8491	0.0158	0.8501	0.0158	0.8501	0.0158

Table 5.1: The mean (μ) and SEM of the performances of each pipeline for each dataset. The mean has been computed by averaging all sessions for all participants of a single dataset. The standard error has been calculated over the number of subjects per dataset.

From the table it becomes clear that the proposed pipelines enhance the classification score for all datasets but the Braininvaders dataset. There is an improvement of 0.51% for both pipelines compared to the tangent space LDA pipeline when the SPOT data is regarded. For EPFL, BNCI_1, BNCI_2 & BNCI_als datasets, this increase is respectively equal to 0.11%, 0.22%, 0.12% & 0.12%. On the other hand, as mentioned before, the performance of both proposed pipelines worsens when the Braininvaders dataset is considered: there is a decrease in performance of 0.04% compared to the baseline. In addition, given the SEM, it is shown in the table that the dispersion of the scores is more or less the same for the three pipelines. The score of both proposed pipelines is very similar. I will devote more attention to this observation in the discussion.

To provide a more detailed view, the scores for SPOT and the Braininvaders datasets are presented individually in the figures 5.2 & 5.3. In Figure 5.2, it is shown that the pairwise performance for a single participant is generally higher for the two proposed pipelines. Subject 6 is an exception to this observation. The two pipelines did not provide any improvements for the Braininvaders dataset. It is very hard to see a difference between the subjects, as the mean peformance decrease is equal to 0.4%. Still, subjects 13 & 17 can be used as an example where the performance has slightly declined. On the other hand, for some subjects the classification score was increased. The latter observation holds for example for subjects 3 and 4.

For the remaining datasets, similar figures can be found in appendix B. For these datasets it holds that both proposed pipelines yield better performances compared to the baseline, yet, the improvements are small. To find out what scores are not only small, but also marginal, I used statistical tests to identify what differences could be considered significant. This is the topic of the next section.

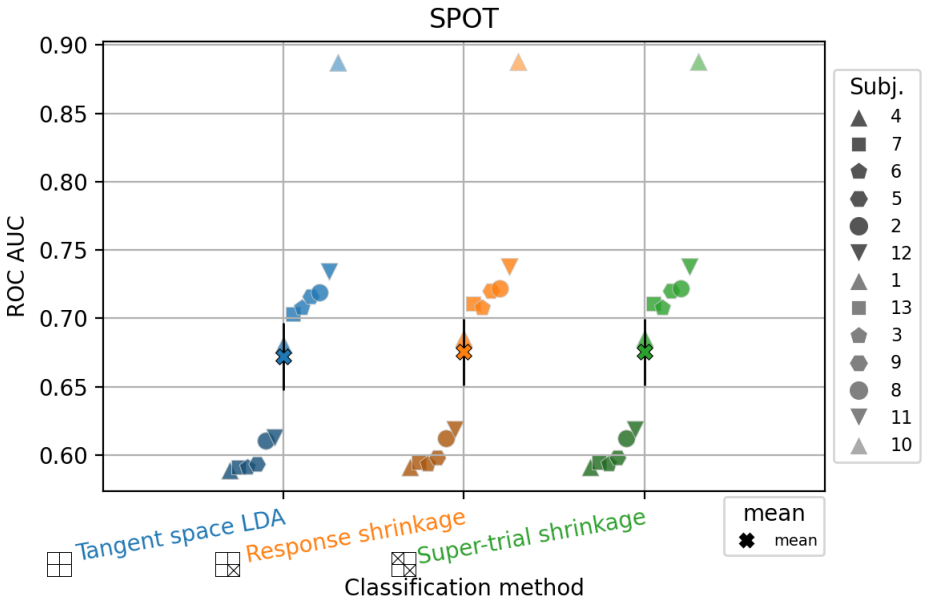


Figure 5.2: The comparison of the three pipelines over the SPOT dataset. To acquire the mean `ROC AUC` scores for each participant, all sessions of that participant have been averaged. The mean of the datasets has been indicated with the cross marker. The errorbars indicate two standard errors around that mean (SEM). The datasets have been sorted in an ascending order in accordance with the score of the tangent space LDA pipeline.

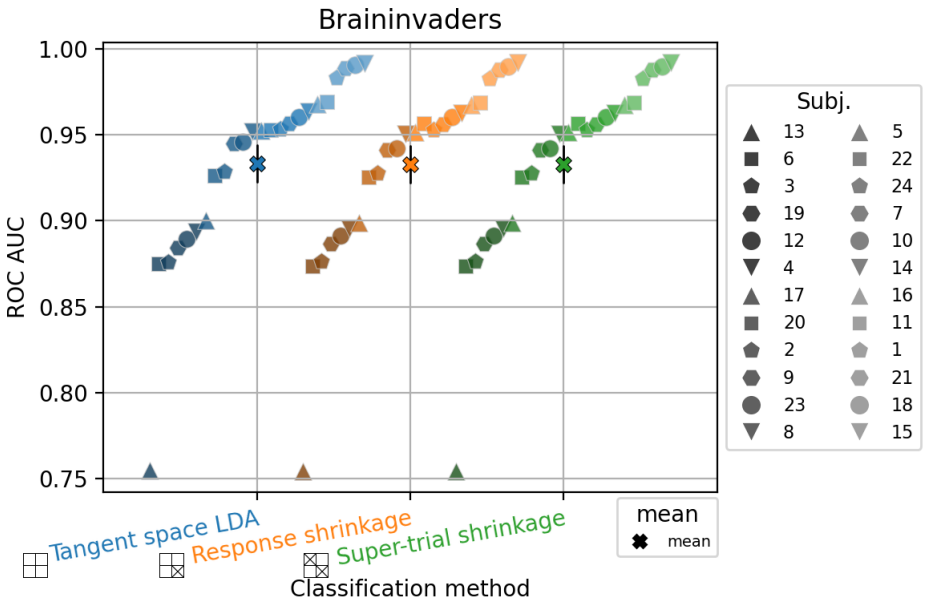


Figure 5.3: The comparison of the three pipelines over the Braininvaders dataset. To acquire the mean `ROC AUC` scores for each participant, all sessions of that participant have been averaged. The mean of the datasets has been indicated with the cross marker. The errorbars indicate two standard errors around that mean (SEM). The datasets in the legend have been sorted in an ascending order in accordance with the score of the tangent space LDA pipeline.

5.2 Statistical Significance

As mentioned in Section 4.5, I used the Wilcoxon signed rank test because of the ceiling effect in the score. The BNCI_1 dataset can be used as an example of this effect, shown in Figure 5.4.

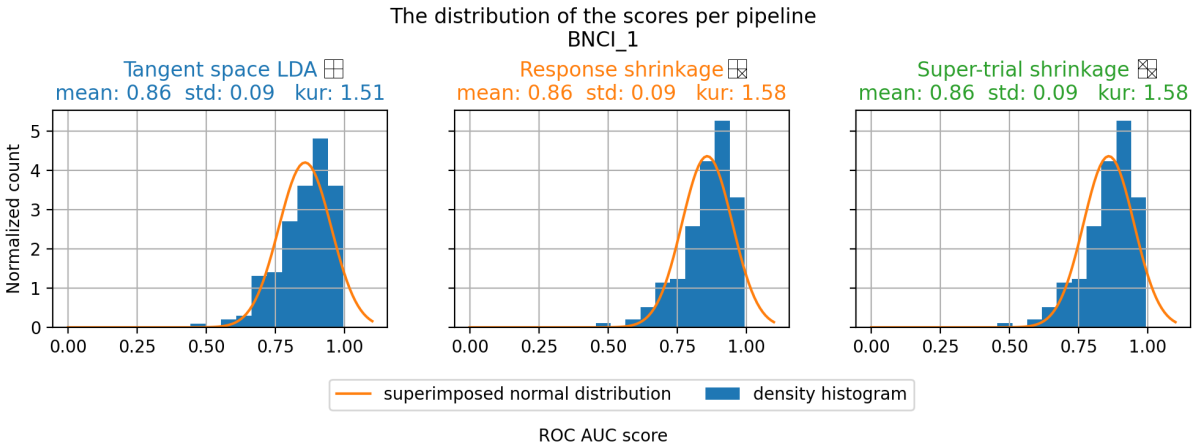


Figure 5.4: The normalized histogram of the `ROC AUC` scores of the BNCI_1 dataset. The histogram is superimposed with a gaussian distribution that is parameterized with the mean and standard deviation of the scores. Because of the ceiling effect (the `ROC AUC` is merely defined on the interval [0,1]), the distribution is left skewed. This is also reflected in the kurtosis (abbreviated with *kur*). Fishers definition of the kurtosis is used, implying that a Gaussian distribution has a kurtosis of 0.0 [52].

The p-values that have been found are presented in Table 5.2.

The p-value per dataset

Dataset	*Response shrinkage vs. tangent space LDA* *p-value*
SPOT	0.000488
Braininvaders	0.0951
EPFL	0.156
BNCI_1	0.105
BNCI_2	0.193
BNCI_als	0.109

Table 5.2: The results of the two-sided Wilcoxon signed rank test. The scores the resonse shrinkage pipeline have been compared to the baseline. The mean scores per participant served as samples for each test. The number of participants differs per dataset, as shown in table 4.1.

To determine whether the found changes in performance are significant under the Holm-Bonferroni correction, the p-values are first sorted in ascending order. Next, the p-values are compared with the adjusted significance level. The adjusted significance level according to the Holm-Bonferroni method is equal to $\frac{\alpha}{m-index}$. Here, *index* is the index of the p-value in the sorted list, starting at 0, and m is the number of conducted tests. I did not compare both methods to the baseline for their performances are more or less equal, as shown in table 5.1. Rather, I only compared the response shrinkage pipeline to the baseline. The algorithm considers the significance of the p-values one by one, until a the null hypothesis cannot be rejected. In that case, the remainder of the null hypotheses is accepted, indicating that there is no significant difference in these cases [53].

After having applied the Holm-Bonferroni correction, it becomes clear that the null hypothesis can be rejected for the improvement of the classification performance of the SPOT dataset $\left(0.000488 < \frac{0.05}{6}\right)$. The next p-value in the list, that corresponds to the score of super trial on the Braininvaders dataset, compared to the score of tangent space LDA on the same dataset, is larger than the significance level $(0.0951 > \frac{0.05}{5})$. Thus, all null hypotheses that correspond to the remaining p-values are accepted, meaning that there is no significant difference between the baseline and the proposed methods for the concerned datasets.

Chapter 6

Discussion

Purpose of the Chapter

In the previous chapter, the classification scores of the three pipelines have been presented: The classification score of one dataset has been improved significantly. For another dataset there is a slight and insignificant decrease in performance, whereas there is an insignificant increase in performance for the remaining datasets. The goal of this chapter is to discuss the cause of the differences in performances that have been shown. To guide this discussion, I first review the Ledoit-Wolf shrinkage regularization algorithm by investigating the mean values of the shrinkage parameter. Next, I analyze the effect of shrinkage regularization on the MSE and Riemannian distance. Furthermore, I discuss the locations that I have identified for the application of shrinkage regularization in section 6.4. Lastly, I have a look at the differences between the datasets, that are possibly reflected in the differences between the results.

6.1 Assessing the Degree of Shrinkage Regularization

One of the possible causes for the insignificant change in performance could be the degree of shrinkage that has been applied. If the value of the shrinkage parameter is low, then one would expect that there is only a marginal difference between the 3 pipelines. Therefore, the first thing that I will discuss is the value of the shrinkage parameter.

In section 3.2.1, that provided a brief overview of shrinkage regularization as proposed by Ledoit & Wolf, it became clear that the tradeoff between the SCM, $\mathbf{S}$, and the structured estimator, $v\boldsymbol{I}$, is regularized by the shrinkage parameter ρ. The values for both v and ρ can be calculated analytically. This solution is based on the minimization of the mean squared error between the true covariance matrix, $\boldsymbol{\Sigma}$ and the regularized estimator $\mathbf{S}^*$ [6].

To explore to which degree shrinkage regularization has been applied to the different submatrices, I calculated the mean shrinkage parameter over the different datasets. These values are shown in table 6.1. This process allowed me to discover why the scores of the two proposed pipelines are so similar: the only difference between the two pipelines is the application of shrinkage regularization to the prototype covariance matrix, located in the upper left. The value of ρ, that corresponds to this operation, is either 0 or 1 for each of the trials. If the $\rho = 0$, then there is no shrinkage applied, and $\mathbf{S}^* = \mathbf{S}$. As a result, the covariance matrix of the prototype is equal for all 3 pipelines. If the value of the shrinkage parameter is equal to 1, then $\mathbf{S}^* = v\boldsymbol{I}$, where $v = \frac{\mathrm{tr}(\mathbf{S})}{p}$. Yet, the eventual covariance matrix does not differ in these two cases. This is due to two reasons:

(1) The covariance matrix of the prototype is almost diagonal (see appendix A). The entries that are not on the main diagonal are non-zero, yet, they are in the order of 10^{-15}, so they are rounded down. As a consequence, the covariance matrix of the prototype only stores the variances of each feature.

(2) The data is z-scored prior to the application of shrinkage and the creation of the covariance matrices. As a consequence, the variance of each feature is 1.

If the value of the shrinkage parameter is 1, then the matrix is replaced by $v\boldsymbol{I}$. Because the SCM of the prototype is (almost a) diagonal matrix (1), where (2) each value on the main diagonal is equal to 1, then $v = \frac{\sum_p 1}{p} = \frac{p}{p} = 1$ and thus $v\boldsymbol{I} = \boldsymbol{I}$. Hence, the SCM of the prototype response is replaced by the identity matrix. Next, the z-scoring is made undone by scaling up $\mathbf{S}^*$, which yields approximately $\mathbf{S}$. As a result, the two proposed pipelines in practice are very similar.

The mean values of ρ for the SCM of the response covariance matrix are a good reflection of the individual values. Thus, the shrinkage regularization that has been applied to this submatrix was more effective.

The mean value of the shrinkage parameter ρ

Dataset	Response covariance matrix	Prototype covariance matrix
SPOT	0.32	0.72
Braininvaders	0.28	0.75
EPFL	0.21	0.75
BNCI_1	0.36	0.72
BNCI_2	0.26	0.76
BNCI_als	0.36	0.80

Table 6.1: The mean value of the shrinkage parameter ρ for the response and prototype submatrices. The value of the shrinkage parameter is defined on the interval $[0, 1]$, this is maintained by truncating the value [6]. The value of ρ for the prototype matrix is a lot higher than the shrinkage parameter that corresponds to the response matrix. This table only shows the mean. In fact, when inspecting the data, it turns out that the value of ρ for the prototype matrix is either 0 or 1.

From the values of ρ from Table 6.1 for the response covariance matrix, I cannot deduce that the value for ρ is very different for the SPOT dataset compared to the other datasets. Hence, with this analysis I cannot conclude what is the cause of the differences in classification performance between the datasets is.

6.2 The Efficiency of Ledoit-Wolf Shrinkage Regularization

To determine the efficiency of the Ledoit-Wolf lemma, I compared the values of ρ that have been analytically computed according to equation (3.14) to the value of ρ that minimizes the MSE according to a grid search strategy. In essence, true covariance matrix, $\boldsymbol{\Sigma}$, should be known before one can compute the MSE with this matrix in regard. Obviously, for the datasets introduced in section 4.2, $\boldsymbol{\Sigma}$ is unknown. To still carry out the analyses I made the assumption that EEG signals follow a multivariate normal distribution with a mean of 0. This assumption has also been made in [12], [38]. The covariance matrix that parameterizes the distribution has been sampled from a univariate gaussian distribution, with a mean of 0 and a variance of 1[11]. This covariance matrix is used as the true (or target) covariance matrix, and will thus be denoted with $\boldsymbol{\Sigma}$. Given $\boldsymbol{\Sigma}$ and the assumption introduced above, the data, $\mathbf{X}$, is sampled from a multivariate normal distribution $\mathcal{N}(0, \boldsymbol{\Sigma})$. This process is repeated for 20 iterations. Where each iteration is parameterized with a fresh $\boldsymbol{\Sigma}$. In each iteration the MSE is averaged over 100 different samples with the same $\boldsymbol{\Sigma}$ to obtain a more informed statistic, in combination with the SEM. I chose two different combinations of samples and features. In the first combination,

[11] All matrices and data have been sampled with the same seed.

with 31 features and 100 samples, the SCM is likely not a good estimator of the true covariance matrix. The number of features is set to 31 as this lines up with the 31 unfiltered channels of the SPOT dataset. The 100 samples represent the 100 timepoints. In the second combination, the 5 features ought to represent the 5 spatially filtered features after the application of xDAWN.

In the figures 6.1 & 6.2 the value of the shrinkage parameter and the corresponding MSE are shown for two different ratios of the number of features to the number of samples. From the figures it becomes clear that the value of the shrinkage parameter according to the Ledoit-Wolf lemma is considerably larger than the shrinkage parameter that results from the grid search. This effect enlarges when the ratio is small, as shown in 6.2 in particular. Because of the small ratio, the value of the shrinkage parameter decreases slightly. However, the value of the parameter that minimizes the MSE is a lot smaller.

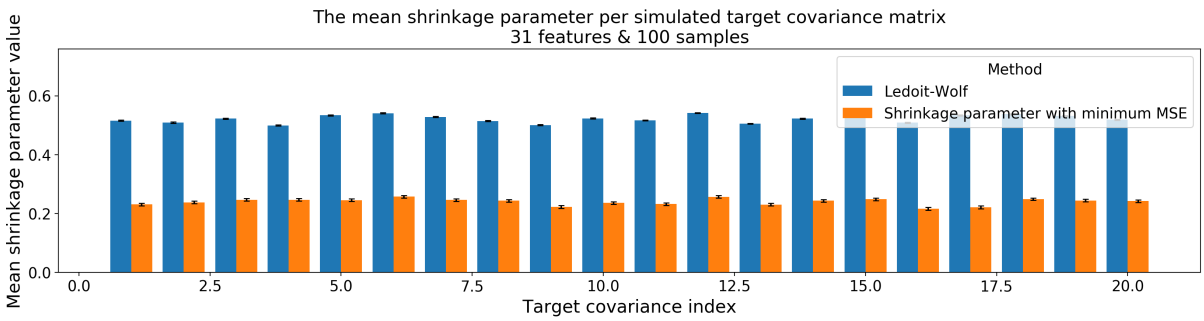


Figure 6.1: The mean value of the shrinkage parameter ρ per simulated target covariance matrix $\in \mathbb{R}^{31\times 31}$. For the estimation of the sample covariance matrices, 100 samples have been generated using a normal distribution.

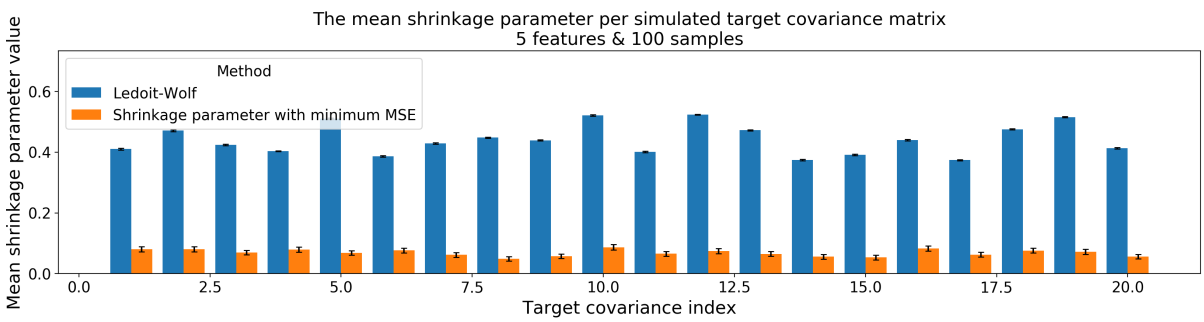


Figure 6.2: The mean value of the shrinkage parameter ρ per simulated target covariance matrix $\in \mathbb{R}^{5\times 5}$. For the estimation of the sample covariance matrices, 100 samples have been generated using a normal distribution. The value of the grid search shrinkage parameter (orange) is a lot smaller than the value of the SCM that has been estimated using to Ledoit-Wolf shrinkage. This is because the SCM is already a good estimator of the true covariance matrix, as the number of samples compared to the number of features is larger than before.

Regarding the MSE it becomes clear from figures 6.3 & 6.4 that the mean squared error of the Ledoit-Wolf estimator algorithm is lower than the SCM depicted in green. However, this difference becomes smaller when the ratio of the features to the samples becomes smaller. This is to be expected, as a small ratio implies that the SCM is a sufficient estimator [6]. Moreover, the MSE from the Ledoit-Wolf estimator is larger than the MSE that corresponds to the grid search shrinkage.

I hypothesize that the efficiency of the Ledoit-Wolf shrinkage regularization is more comparable to the results in figure 6.2 than the results from 6.1. This is because the two instances are more

similar with respect to the ratio of the number of features to the number of samples. This would imply that the effect of the regularization is small as the SCM is already an appropriate estimate of the target covariance matrix.

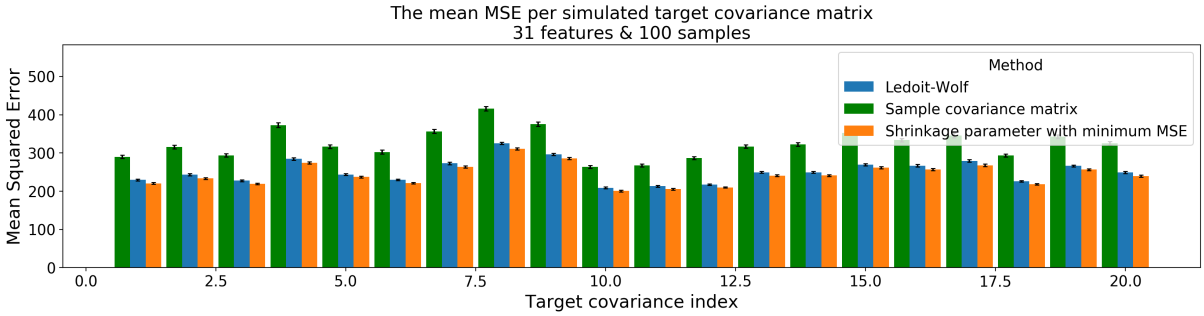


Figure 6.3: The mean value of the MSE per simulated target covariance matrix $\in \mathbb{R}^{31\times31}$. For the estimation of the sample covariance matrices, 100 samples have been generated using a normal distribution.

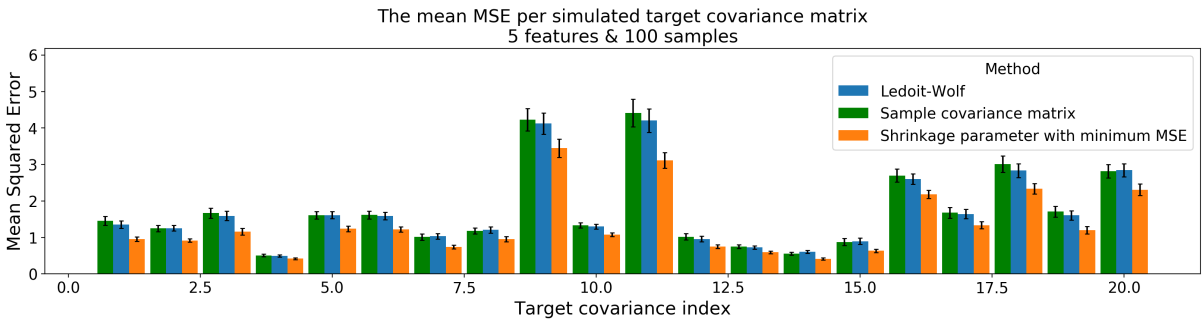


Figure 6.4: The mean value of the MSE per simulated target covariance matrix $\in \mathbb{R}^{5\times5}$. For the estimation of the sample covariance matrices, 100 samples have been generated using a normal distribution. The values of the MSE are a lot lower compared to Figure 6.3, as the SCM is a better estimator given the higher number of samples.

Shrinkage regularization as proposed by Ledoit & Wolf is not the only existing shrinkage technique. In [39], an overview is provided of the other techniques and different results can be obtained from them. I have used the technique by Ledoit & Wolf as this is the technique that I was the most familiar with.

6.3 Shrinkage Regularization & the Riemannian Distance

Whereas I just discussed the MSE between the simulated target covariance matrices and their corresponding SCM, it is also worthwhile to consider the effect of shrinkage regularization on the Riemannian distance. The technique proposed by Ledoit & Wolf minimizes the MSE and does not give any guarantees about the Riemannian distance [6]. I assessed the effect of shrinkage on the Riemannian distance using the same strategy as in the previous section, naturally, I now calculated the Riemannian distance instead of the MSE. The distances, that have been calculated using the `PyRiemann` library, are shown in Figures 6.5 & 6.6. Both sample covariance matrices where shrinkage regularization has been applied suffer from an increase in the Riemannian distance compared to the original SCM. As a result, the shrunken SCM is a worse estimator of the true covariance matrix when the Riemannian distance is concerned! This effect possibly has drawbacks when Riemannian classifiers that are based on the Riemannian distance, such as minimum distance to mean [8], are deployed on the manifold.

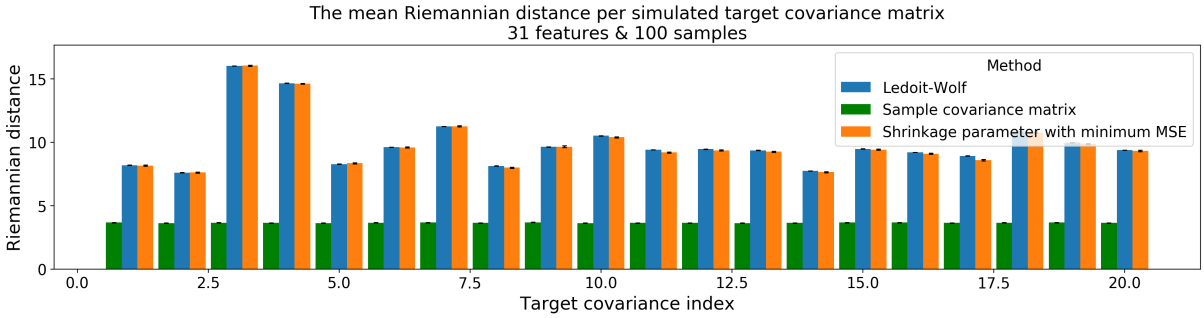


Figure 6.5: The mean value of the Riemannian distance per simulated target covariance matrix $\in \mathbb{R}^{31\times31}$. It can be seen that the shrunken matrices consistently increase the Riemannian distance compared to the SCM without shrinkage

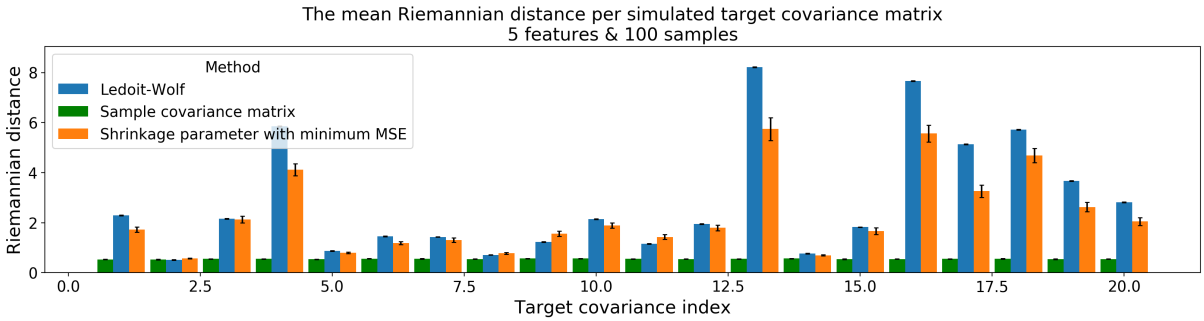


Figure 6.6: The mean value of the Riemannian distance per simulated target covariance matrix $\in \mathbb{R}^{5\times5}$. Again, shrinkage regularization increases the Riemannian distance, but the differences are smaller in this case.

6.4 The Importance of the Cross-covariance Matrix

Having discussed shrinkage regularization itself, I continue with the location in the composite covariance matrices where the technique is applied. In the two proposed pipelines, shrinkage regularization has merely been applied to the submatrices on the main diagonal. The submatrices on the off diagonal, $\mathbf{C}_{\mathbf{P1},\mathbf{X}_i}$, that store the cross-covariance matrix between the response and the prototype, are ignored. This contrasts with the importance of the submatrices as defined by Barachant & Congedo [12]. In fact, the covariance matrix of the prototype is the same for all epoch-based covariance matrices [12]. Hence, it is not of great value when trying to classify the responses. In retrospect, while it seemed interesting to see the effect of the application of shrinkage regularization on the SCM of the response, it was not a great idea.

I also considered about shrinking the entire matrix to involve the cross-covariance matrix in the shrinkage process. I chose not to do so as shrinkage regularization effectively makes a tradeoff between a diagonal matrix and the SCM, where both operands are multiplied by a scalar. The cross-covariance matrix, not being on the main diagonal, would only have been multiplied by the scalar in that situation. Moreover, this scalar would be merely based on the trace of the sample covariance matrices of the prototype and the response. I do not know, nor has it been reported in the literature, whether it is appropriate to apply the Ledoit-Wolf Shrinkage technique on an artificially created covariance matrix, that has been obtained from data where the response has been concatenated with the prototype. Shrinkage regularization, as proposed by Ledoit & Wolf, is in principle proposed to find a better estimator for covariance matrices [6]. I have not found an example in the literature that reports the application on a cross-covariance matrix. As a consequence, I could not involve the cross-covariance matrix in the process of shrinkage

regularization, whereas involving it might have a larger effect on the classification score due to its importance [12].

6.5 Differences Between Datasets

Despite the caveats of the application of shrinkage regularization to the epoch-based covariance matrices that have been discussed in this chapter, the classification scores of the shrinkage pipelines were higher for 5 out of 6 datasets. One of the reasons for this effect could be that the difference between the mean score on the SPOT dataset and the mean score over all datasets is 0.159 for the baseline. On the other hand, the mean value of the Braininvaders dataset lies 0.102 above the average. Hence, it seems that some improvement can be achieved if the score of the baseline is on the low end of the spectrum.

The difference in scores between the datasets can have numerous causes, such as the degree of noise or the number of data points (epochs) that are available per session (Table 4.1). The latter feature is of interest as it determines the number of training data points for the classifier, as well as the number of samples[12] on which the prototype is based on. The quality of the prototype influences the cross-covariance matrix, that has been discussed in the previous section. The number of epochs per session for the SPOT dataset is relatively low (90). However, this also holds for the BNCI_1 dataset (96). Yet, the scores of the baseline regarding last-mentioned dataset are a lot higher ($\mu = 0.8273$) than for the SPOT dataset ($\mu = 0.6720$). If the difference was caused by the small number of target responses, I would expect the score from the BNCI_2 dataset to be lower as well. Similarly, the number of target responses per epoch is equal to 15 and 16 for the SPOT and BNCI_1 datasets respectively.

[12] In particular, the prototype is based on all the target responses within a single session.

Chapter 7

Conclusion

To improve the estimation of epoch-based covariance matrix for the Riemannian pipeline, I propose to apply Ledoit-Wolf shrinkage regularization on its constituents. Shrinking the submatrix that represents the brain response leads to a significant improvement of the classification scores for one out of the six datasets that have been considered. From the results, it appears that the method can be used to enhance the classification score of the datasets that are more difficult to classify. It is yet unclear which characteristics of the data determine whether or not the proposed method is a true enhancement. Possibly, this could be investigated in future work.
There are also limitations to this method. The application of shrinkage regularization increases the Riemannian distance between the target and the sample covariance matrix. In addition, there is no guaranteed preservation of the symmetric positive definiteness of the covariance matrices. Furthermore, the cross-covariance matrix remains unaltered, while this matrix is of great importance for the classification of the data points. In further research, more attention can be devoted to this part of the matrix. Lastly, I have constrained myself to the use of Ledoit-Wolf shrinkage. This is however not the only variant of shrinkage regularization, and possibly different results can be obtained by using other methods.

Returning to the research question: How can shrinkage regularization be applied to single data point covariance estimations to improve the classification score of the Riemannian pipeline with respect to the Riemannian pipeline without improved covariance estimations?

I conclude that the application of Ledoit-Wolf shrinkage regularization on the submatrix of the response can lead to an improvement of the ROC AUC score for datasets that are accompanied by relatively low classification scores.

Bibliography

[1] B. Kolb, I. Q. Whishaw, and G. C. Teskey, *An Introduction to Brain and Behaviour*, 5th ed. New York, NY: Worth Publishers, 2016, ch. 4, pp. 107–137.

[2] B. Blankertz, S. Lemm, M. Treder, S. Haufe, and K.-R. Müller, "Single-trial analysis and classification of ERP components —A tutorial," *NeuroImage*, vol. 56, no. 2, pp. 814–825, Jun. 2011. DOI: 10.1016/j.neuroimage.2010.06.048.

[3] J. Sosulski, J. Kemmer, and M. Tangermann, "Improving Covariance Matrices Derived from Tiny Training Datasets for the Classification of Event-Related Potentials with Linear Discriminant Analysis," *Neuroinformatics*, pp. 1–16, Dec. 2020. DOI: 10.1007/s12021-020-09501-8.

[4] H. Kolkhorst, J. Veit, W. Burgard, and M. Tangermann, "A Robust Screen-Free Brain-Computer Interface for Robotic Object Selection," *Front. Robot. AI*, vol. 7, no. 38, Mar. 2020. DOI: 10.3389/frobt.2020.00038.

[5] J. H. Friedman, "Regularized Discriminant Analysis," *Journal of the American Statistical Association*, vol. 84, no. 405, pp. 165–175, Aug. 1989, ISSN: 01621459. [Online]. Available: http://www.jstor.org/stable/2289860.

[6] O. Ledoit and M. Wolf, "A well-conditioned estimator for large-dimensional covariance matrices," *Journal of Multivariate Analysis*, vol. 88, no. 2, pp. 365–411, Feb. 2004. DOI: 10.1016/S0047-259X(03)00096-4.

[7] H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller, "Optimal Spatial Filtering of Single Trial EEG During Imagined Hand Movement," *IEEE transactions on rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 8, pp. 441–6, Jan. 2001. DOI: 10.1109/86.895946.

[8] M. Congedo, A. Barachant, and R. Bhatia, "Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review," *Brain-Computer Interfaces*, vol. 4, no. 3, pp. 155–174, Mar. 2017. DOI: 10.1080/2326263X.2017.1297192.

[9] J. Lee, *Introduction to Smooth Manifolds*, 2nd ed. New York, NY: Springer-Verlag, 2012, ch. 1, pp. 1–29.

[10] F. P. Kalaganis, N. A. Laskaris, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "A Riemannian Geometry Approach to Reduced and Discriminative Covariance Estimation in Brain Computer Interfaces," *Ieee Transactions on Bio-Medical Engineering*, vol. 67, no. 1, pp. 245–255, Apr. 2020. DOI: 10.1109/TBME.2019.2912066.

[11] B. Kolb, I. Q. Whishaw, and G. C. Teskey, *An Introduction to Brain and Behaviour*, 5th ed. New York, NY: Worth Publishers, 2016, ch. 7, pp. 209–244.

[12] A. Barachant and M. Congedo, "A Plug & Play P300 BCI Using Information Geometry," *CoRR*, vol. abs/1409.0107, 2014. arXiv: 1409.0107. [Online]. Available: http://arxiv.org/abs/1409.0107.

[13] C. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. New York, NY: Springer-Verlag, 2006, ch. 2, pp. 67–136.

[14] S. Deshmukh and A. Dubey, "Improved Covariance Matrix Estimation With an Application in Portfolio Optimization," *IEEE Signal Processing Letters*, vol. 27, pp. 985–989, May 2020. DOI: 10.1109/LSP.2020.2996060.

[15] K. B. Petersen and M. S. Pedersen, *The Matrix Cookbook*, 2012.

[16] C. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. New York, NY: Springer-Verlag, 2006, ch. 1, pp. 179–224.

[17] D. Hübner, "From Supervised to Unsupervised Machine Learning Methods for Brain-Computer Interfaces and Their Application in Language Rehabilitation," Ph.D. dissertation, Jan. 2020, ch. 2, pp. 9–32.

[18] J. Sosulski and M. Tangermann, "Electroencephalogram signals recorded from 13 healthy subjects during an auditory oddball paradigm under different stimulus onset asynchrony conditions. dataset," *Journal of Neural Engineering*, 2019. DOI: 10.6094/UNIFR/154576. (visited on 03/05/2021).

[19] F. Lotte, M. Congedo, A. Lécuyer, L. Fabrice, and B. Arnaldi, "A review of classification algorithms for EEG-based brain-computer interfaces," *Journal of Neural Engineering*, vol. 4, Jul. 2007. DOI: 10.1088/1741-2560/4/2/R01.

[20] A. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, Jan. 2000. DOI: 10.1109/34.824819.

[21] P. Goektepe and A. Tzovara. (2019). "EEG Data Processing - Baseline Correction," [Online]. Available: https://neuro.inf.unibe.ch/AlgorithmsNeuroscience/Tutorial_files/BaselineCorrection.html (visited on 05/03/2021).

[22] A. Gramfort, M. Luessi, E. Larson, D. A. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, and M. S. Hämäläinen, "MEG and EEG data analysis with MNE-Python," *Frontiers in Neuroscience*, vol. 7, no. 267, pp. 1–13, Dec. 2013. DOI: 10.3389/fnins.2013.00267.

[23] C. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. New York, NY: Springer-Verlag, 2006, ch. 1, pp. 1–66.

[24] A. Hyvärinen and E. Oja, "A Fast Fixed-Point Algorithm for Independent Component Analysis," *Neural Computation*, vol. 9, no. 7, pp. 1483–1492, Jul. 1997. DOI: 10.1162/neco.1997.9.7.1483.

[25] A. Barachant, *MEG decoding using Riemannian Geometry and Unsupervised classification.* [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.713.5131&rep=rep1&type=pdf (visited on 05/03/2021).

[26] B. Rivet, A. Souloumiac, V. Attina, and G. Gibert, "Xdawn algorithm to enhance evoked potentials: Application to brain–computer interface," *Biomedical Engineering, IEEE Transactions on*, vol. 56, pp. 2035–2043, Sep. 2009. DOI: 10.1109/TBME.2009.2012869.

[27] Taisbak, Christian Marinus and Waerden, Bartel Leendert van der, *Euclid*, Jan. 2021. [Online]. Available: https://www.britannica.com/biography/Euclid-Greek-mathematician (visited on 04/01/2021).

[28] Weisstein, E. W., *Space.* From *Mathworld* —A Wolfram Web Resource. [Online]. Available: https://mathworld.wolfram.com/Space.html (visited on 04/01/2021).

[29] S. Sommer, T. Fletcher, and X. Pennec, "Introduction to differential and Riemannian geometry," in *Riemannian Geometric Statistics in Medical Image Analysis*, Academic Press, May 2020, pp. 3–37, ISBN: 978-0-12-814725-2. DOI: 10.1016/B978-0-12-814725-2.00008-X.

[30] L. B. Cormack, "Flat Earth or Round Sphere: Misconceptions of the Shape of the Earth and the Fifteenth-Century Transformation of the World," *Ecumene*, vol. 1, no. 4, pp. 363–385, Aug. 1994. DOI: 10.1177/147447409400100404.

[31] J. Lee, *Introduction to Smooth Manifolds*, 2nd ed. New York, NY: Springer-Verlag, 2012, ch. 3, pp. 50–75.

[32] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Classification of covariance matrices using a Riemannian-based kernel for BCI applications," *Neurocomputing*, vol. 112, pp. 172–178, Jul. 2013. DOI: 10.1016/j.neucom.2012.12.039.

[33] F. Yger, M. Berar, and F. Lotte, "Riemannian Approaches in Brain-Computer Interfaces: A Review," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, pp. 1753–1762, 10 Oct. 2017. DOI: 10.1109/TNSRE.2016.2627016.

[34] *"DecMeg2014 - Decoding the Human Brain"*, Aug. 2014. [Online]. Available: https://www.kaggle.com/c/decoding-the-human-brain (visited on 04/01/2021).

[35] *BCI Challenge*, Apr. 2015. [Online]. Available: https://neuro.embs.org/2015/bci-challenge/ (visited on 04/01/2021).

[36] J. Lee, *Introduction to Smooth Manifolds*, 2nd ed. New York, NY: Springer-Verlag, 2012, ch. 13, pp. 327–344.

[37] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian Framework for Tensor Computing," *International Journal of Computer Vision*, vol. 66, pp. 41–66, Jan. 2006. DOI: 10.1007/s11263-005-3222-z.

[38] C. J. Marco Congedo Cédric Gouy-Pailler, "On the blind source separation of human electroencephalogram by approximate joint diagonalization of second order statistics," *Clinical Neurophysiology*, vol. 119, pp. 2677–2686, 12 Sep. 2008. DOI: 10.1016/j.clinph.2008.09.007..

[39] J. Schäfer and K. Strimmer, "Statistical Applications in Genetics and Molecular Biology," *Statistical Applications in Genetics and Molecular Biology*, vol. 4, no. 1, Nov. 2005.

[40] V. Jayaram and A. Barachant, "MOABB: trustworthy algorithm benchmarking for BCIs," *Journal of Neural Engineering*, vol. 15, no. 6, pp. 1741–2552, Dec. 2018. DOI: 066011.

[41] G. F. P. Van Veen, A. Barachant, A. Andreev, G. Cattan, P. L. Coelho Rodrigues, and M. Congedo, "Building Brain Invaders: EEG data of an experimental validation," no. 1, May 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02126068.

[42] U. Hoffmann, J.-M. Vesin, T. Ebrahimi, and K. Diserens, "An efficient P300-based brain–computer interface for disabled subjects," *Journal of Neuroscience Methods*, vol. 167, no. 1, pp. 115–125, Jan. 2008, ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2007.03.005.

[43] P. Aricò, F. Aloise, F. Schettini, S. Salinari, D. Mattia, and F. Cincotti, "Influence of P300 latency jitter on event related potential-based brain-computer interface performance," *Journal of neural engineering*, vol. 11, no. 3, May 2014. DOI: 10.1088/1741-2560/11/3/035008.

[44] C. Guger, S. Daban, E. Sellers, C. Holzner, G. Krausz, R. Carabalona, F. Gramatica, and G. Edlinger, "How many people are able to control a P300-based brain-computer interface (BCI)?" *Neuroscience Letters*, vol. 462, pp. 94–8, Sep. 2009. DOI: 10.1016/j.neulet.2009.06.045.

[45] A. Riccio, L. Simione, F. Schettini, A. Pizzimenti, M. Inghilleri, M. Belardinelli, D. Mattia, and F. Cincotti, "Attention and P300-based BCI performance in people with amyotrophic lateral sclerosis," *Frontiers in human neuroscience*, vol. 7, p. 732, Nov. 2013. DOI: 10.3389/fnhum.2013.00732.

[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, Oct. 2011.

[47] Weisstein, Eric W., *Positive Definite Matrix.* From *Mathworld* —A Wolfram Web Resource. [Online]. Available: `https://mathworld.wolfram.com/PositiveDefiniteMatrix.html`, (visited on 26/05/2021).

[48] N. J. Higham, "Computing a nearest symmetric positive semidefinite matrix," *Linear Algebra and its Applications*, vol. 103, pp. 103–118, May 1988, ISSN: 0024-3795. DOI: `10.1016/0024-3795(88)90223-6`.

[49] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: `10.1038/s41586-020-2649-2`.

[50] *Determine whether matrix is symmetric positive definite.* [Online]. Available: `https://www.mathworks.com/help/matlab/math/determine-whether-matrix-is-positive-definite.html`, (visited on 26/05/2021).

[51] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[52] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, Feb. 2020.

[53] S. Holm, "A Simple Sequentially Rejective Multiple Test Procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, Sep. 1979.

List of Abbreviations

ALS amyotrophic lateral sclerosis. 23, 24

BCI brain-computer interfaces. 1, 5, 11, 15, 16, 22

BSS blind-source separation. 11

EEG electroencephalography. 5, 9, 11, 16, 22, 24, 38

ERP event-related potential. 6, 9, 10, 12, 17, 22, 30

ICA independent component analysis. 11

iid independent and identically distributed. 19

IIR infinite impulse response. 24

LDA linear discriminant analysis. 5, 6, 8, 11, 12, 13, 15, 20, 21, 26

MDM minimum distance to mean. 40

MOABB mother of all BCI benchmarks. 22, 23, 24, 30

MSE mean squared error. 18, 19, 20, 27, 37, 38, 39, 40

PCA principal component analysis. 11

ROC receiver operating characteristic. 6, 30

ROC AUC receiver operating characteristic area under the curve. 6, 32, 43

SCM sample covariance matrix. 1, 6, 7, 8, 9, 13, 15, 16, 18, 19, 20, 25, 33, 37, 38, 39, 40, 41, 43

SEM standard error of the mean. 20, 32, 33, 34, 38

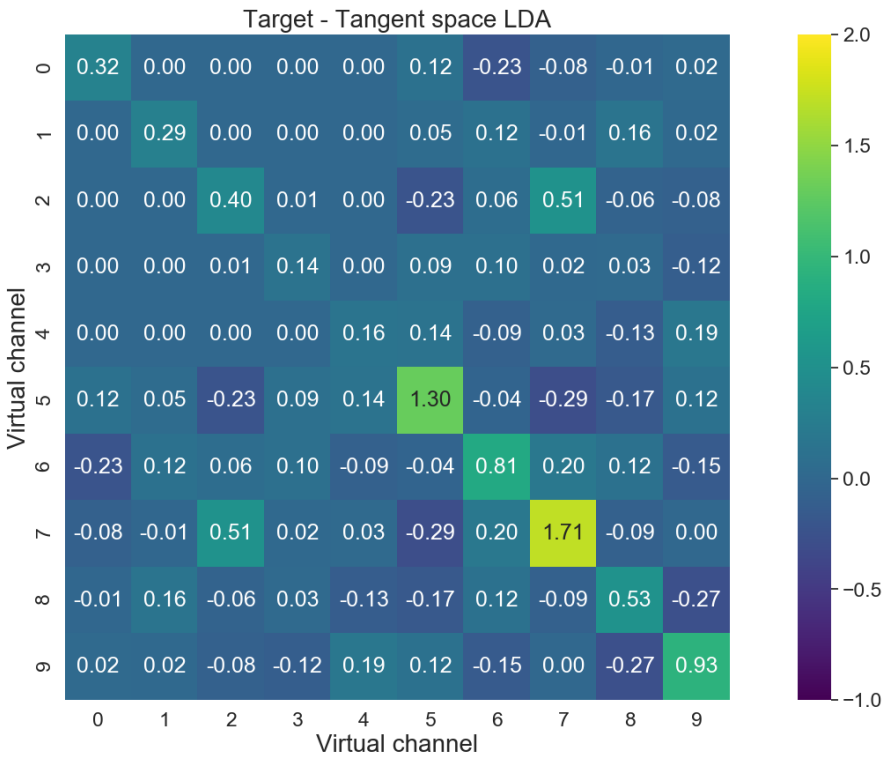
SNR signal-to-noise ratio. 11

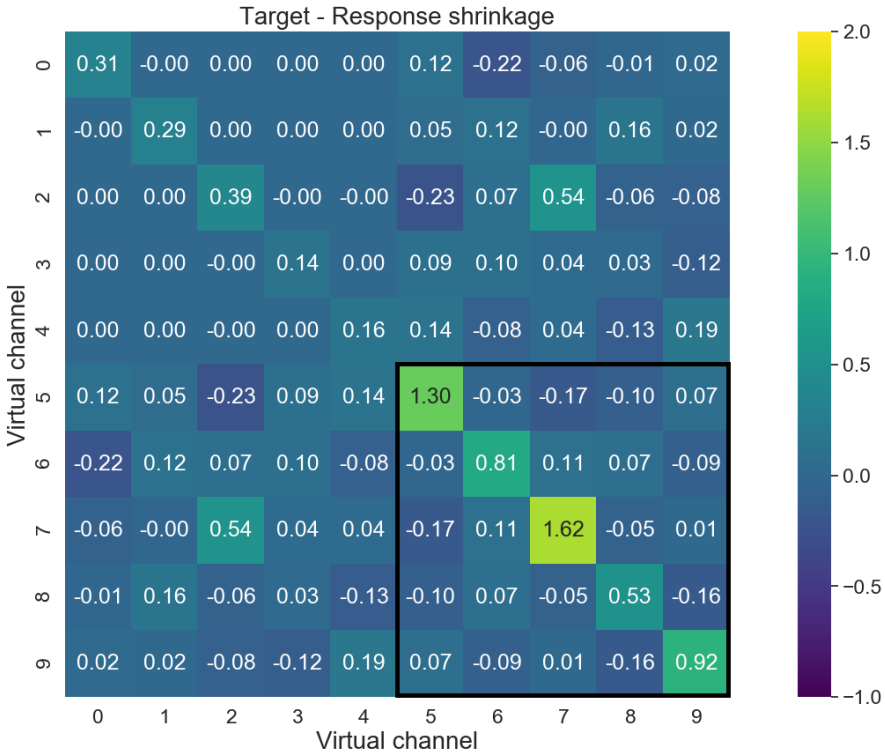
SOA stimulus onset asynchrony. 10, 22, 23, 28, 31

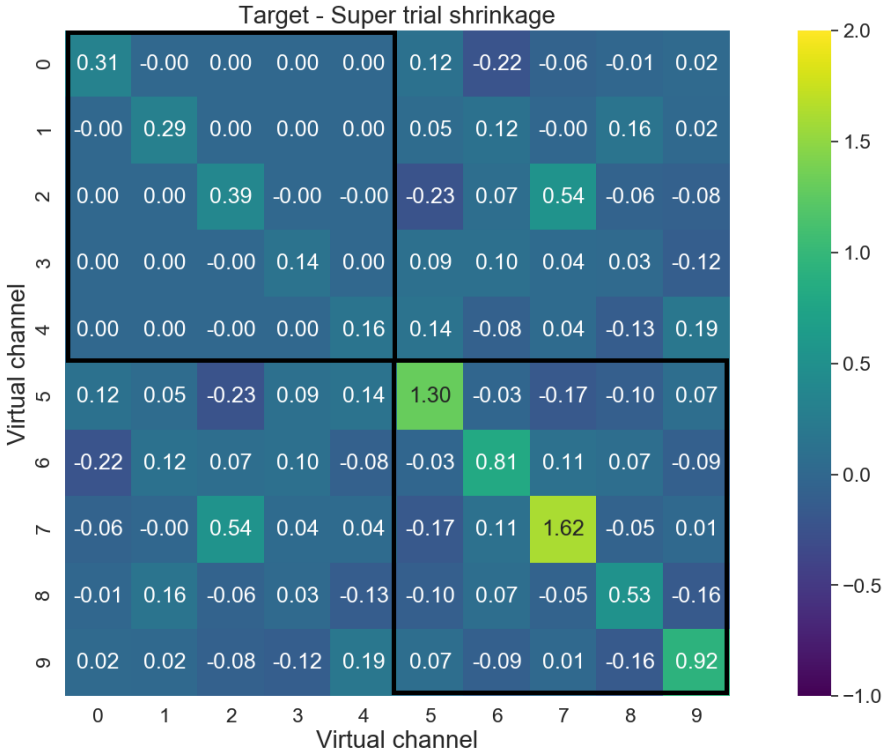
SPD symmetric positive-definite. 5, 7, 17, 29, 30

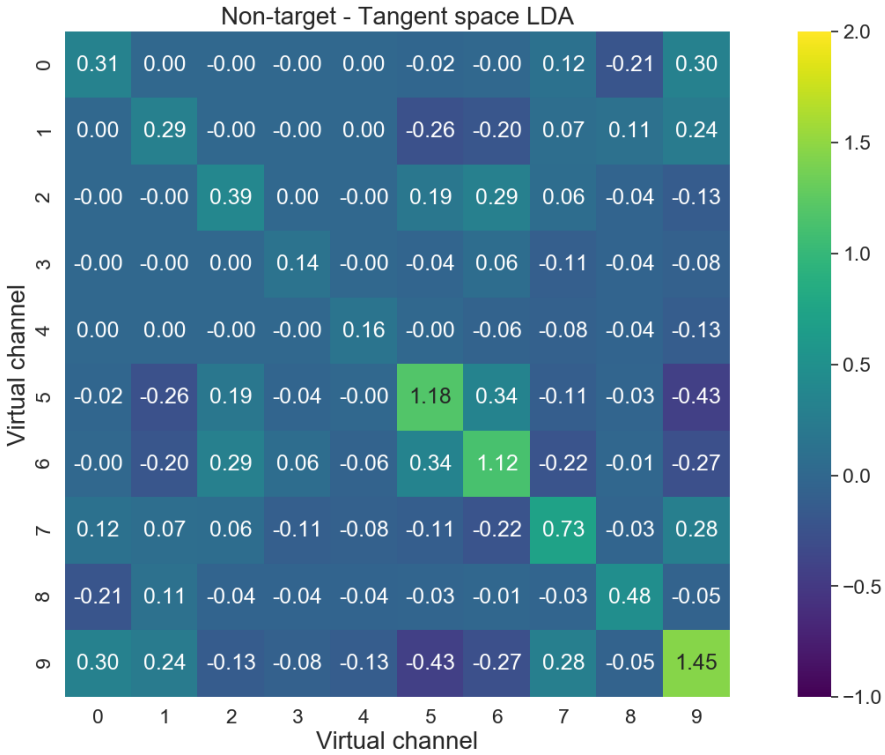
Appendix A

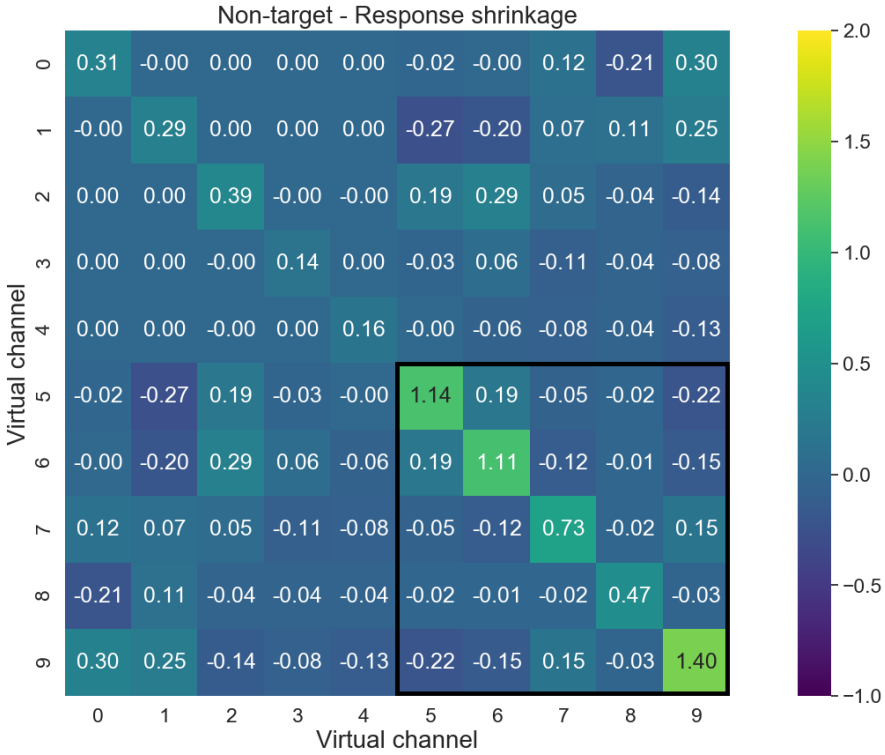
Visualization of Covariance Matrices

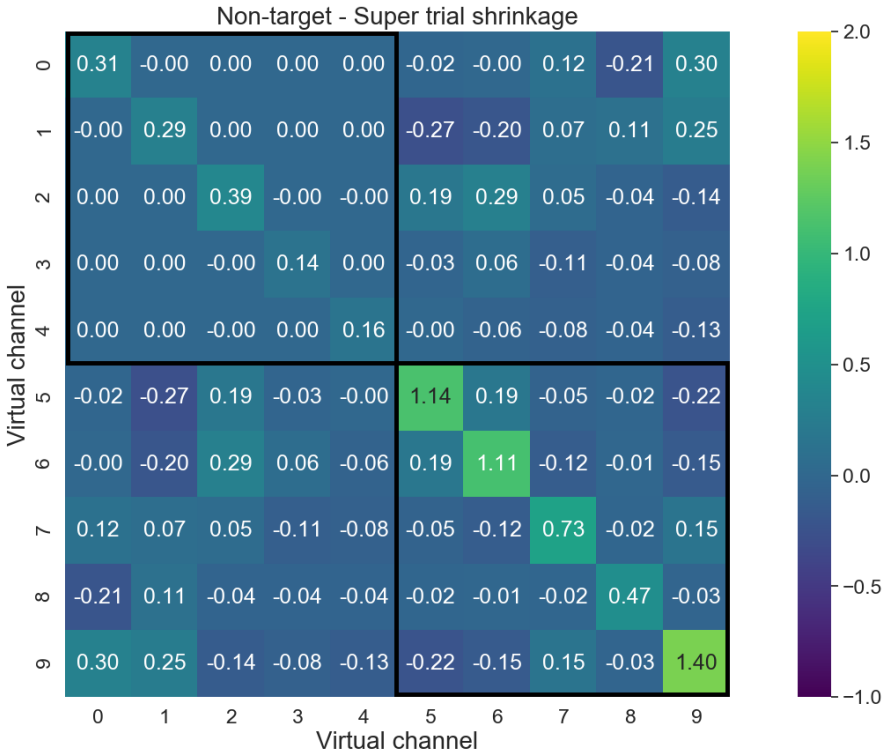



Target - Response shrinkage
Virtual channel
Virtual channel
0 1 2 3 4 5 6 7 8 9
0.31 -0.00 0.00 0.00 0.00 0.12 -0.22 -0.06 -0.01 0.02
-0.00 0.29 0.00 0.00 0.00 0.05 0.12 -0.00 0.16 0.02
0.00 0.00 0.39 -0.00 -0.00 -0.23 0.07 0.54 -0.06 -0.08
0.00 0.00 -0.00 0.14 0.00 0.09 0.10 0.04 0.03 -0.12
0.00 0.00 -0.00 0.00 0.16 0.14 -0.08 0.04 -0.13 0.19
0.12 0.05 -0.23 0.09 0.14 1.30 -0.03 -0.17 -0.10 0.07
-0.22 0.12 0.07 0.10 -0.08 -0.03 0.81 0.11 0.07 -0.09
-0.06 -0.00 0.54 0.04 0.04 -0.17 0.11 1.62 -0.05 0.01
-0.01 0.16 -0.06 0.03 -0.13 -0.10 0.07 -0.05 0.53 -0.16
0.02 0.02 -0.08 -0.12 0.19 0.07 -0.09 0.01 -0.16 0.92
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0


Target - Super trial shrinkage
Virtual channel
Virtual channel
0 1 2 3 4 5 6 7 8 9
0.31 -0.00 0.00 0.00 0.00 0.12 -0.22 -0.06 -0.01 0.02
-0.00 0.29 0.00 0.00 0.00 0.05 0.12 -0.00 0.16 0.02
0.00 0.00 0.39 -0.00 -0.00 -0.23 0.07 0.54 -0.06 -0.08
0.00 0.00 -0.00 0.14 0.00 0.09 0.10 0.04 0.03 -0.12
0.00 0.00 -0.00 0.00 0.16 0.14 -0.08 0.04 -0.13 0.19
0.12 0.05 -0.23 0.09 0.14 1.30 -0.03 -0.17 -0.10 0.07
-0.22 0.12 0.07 0.10 -0.08 -0.03 0.81 0.11 0.07 -0.09
-0.06 -0.00 0.54 0.04 0.04 -0.17 0.11 1.62 -0.05 0.01
-0.01 0.16 -0.06 0.03 -0.13 -0.10 0.07 -0.05 0.53 -0.16
0.02 0.02 -0.08 -0.12 0.19 0.07 -0.09 0.01 -0.16 0.92
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0

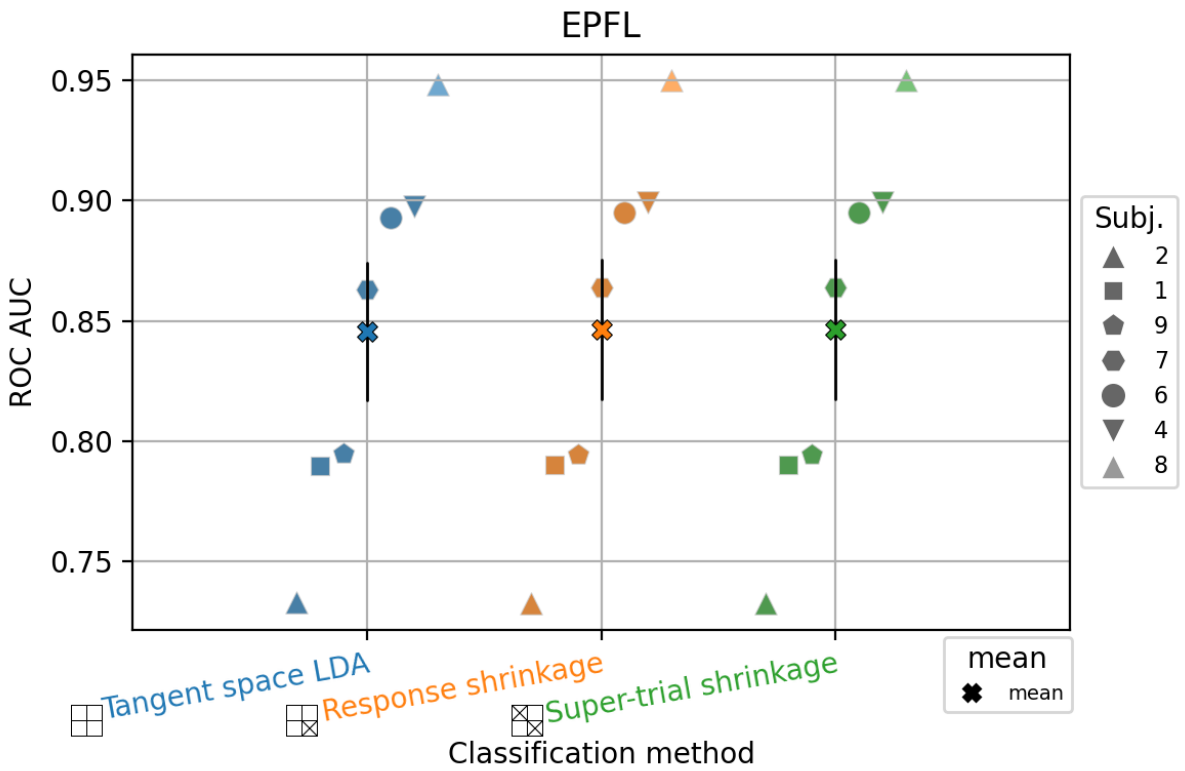

Non-target - Tangent space LDA
Virtual channel
Virtual channel
0 1 2 3 4 5 6 7 8 9
0.31 0.00 -0.00 -0.00 0.00 -0.02 -0.00 0.12 -0.21 0.30
0.00 0.29 -0.00 -0.00 0.00 -0.26 -0.20 0.07 0.11 0.24
-0.00 -0.00 0.39 0.00 -0.00 0.19 0.29 0.06 -0.04 -0.13
-0.00 -0.00 0.00 0.14 -0.00 -0.04 0.06 -0.11 -0.04 -0.08
0.00 0.00 -0.00 -0.00 0.16 -0.00 -0.06 -0.08 -0.04 -0.13
-0.02 -0.26 0.19 -0.04 -0.00 1.18 0.34 -0.11 -0.03 -0.43
-0.00 -0.20 0.29 0.06 -0.06 0.34 1.12 -0.22 -0.01 -0.27
0.12 0.07 0.06 -0.11 -0.08 -0.11 -0.22 0.73 -0.03 0.28
-0.21 0.11 -0.04 -0.04 -0.04 -0.03 -0.01 -0.03 0.48 -0.05
0.30 0.24 -0.13 -0.08 -0.13 -0.43 -0.27 0.28 -0.05 1.45
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0

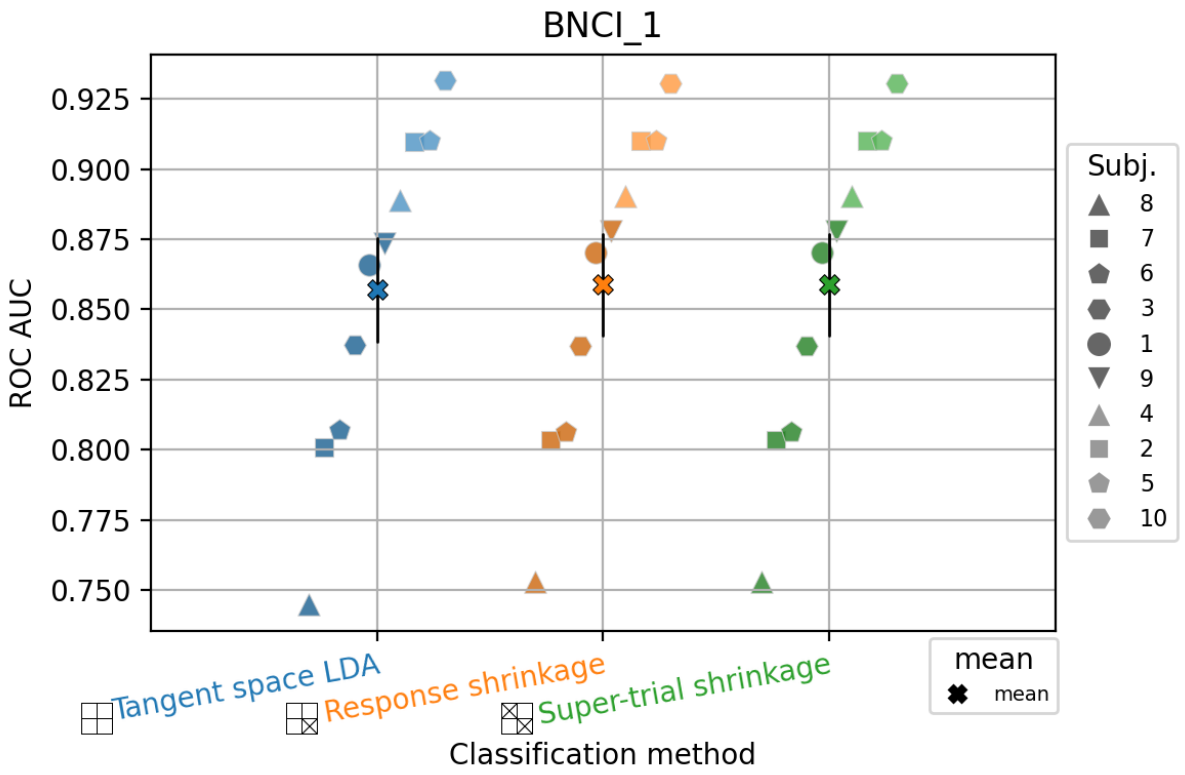

Non-target - Response shrinkage
Virtual channel
Virtual channel
0 1 2 3 4 5 6 7 8 9
0.31 -0.00 0.00 0.00 0.00 -0.02 -0.00 0.12 -0.21 0.30
-0.00 0.29 0.00 0.00 0.00 -0.27 -0.20 0.07 0.11 0.25
0.00 0.00 0.39 -0.00 -0.00 0.19 0.29 0.05 -0.04 -0.14
0.00 0.00 -0.00 0.14 0.00 -0.03 0.06 -0.11 -0.04 -0.08
0.00 0.00 -0.00 0.00 0.16 -0.00 -0.06 -0.08 -0.04 -0.13
-0.02 -0.27 0.19 -0.03 -0.00 1.14 0.19 -0.05 -0.02 -0.22
-0.00 -0.20 0.29 0.06 -0.06 0.19 1.11 -0.12 -0.01 -0.15
0.12 0.07 0.05 -0.11 -0.08 -0.05 -0.12 0.73 -0.02 0.15
-0.21 0.11 -0.04 -0.04 -0.04 -0.02 -0.01 -0.02 0.47 -0.03
0.30 0.25 -0.14 -0.08 -0.13 -0.22 -0.15 0.15 -0.03 1.40
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0

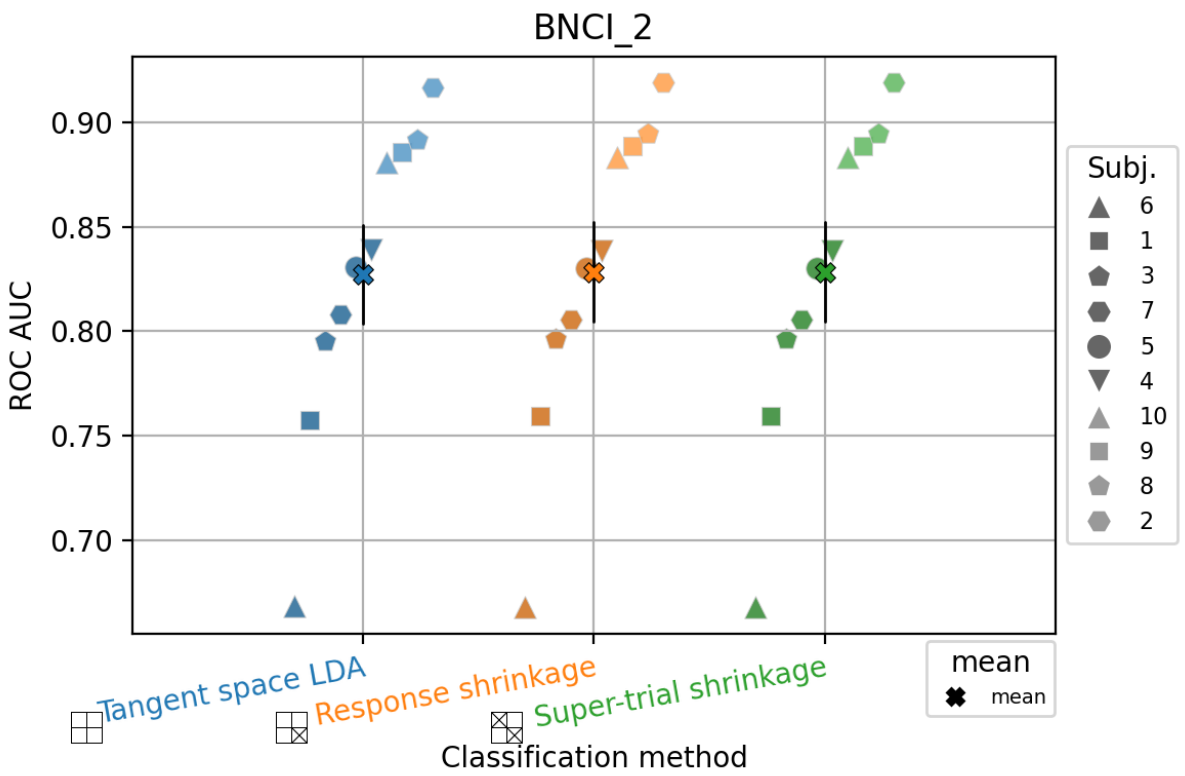

Non-target - Super trial shrinkage
Virtual channel
Virtual channel
0 1 2 3 4 5 6 7 8 9
0.31 -0.00 0.00 0.00 0.00 -0.02 -0.00 0.12 -0.21 0.30
-0.00 0.29 0.00 0.00 0.00 -0.27 -0.20 0.07 0.11 0.25
0.00 0.00 0.39 -0.00 -0.00 0.19 0.29 0.05 -0.04 -0.14
0.00 0.00 -0.00 0.14 0.00 -0.03 0.06 -0.11 -0.04 -0.08
0.00 0.00 -0.00 0.00 0.16 -0.00 -0.06 -0.08 -0.04 -0.13
-0.02 -0.27 0.19 -0.03 -0.00 1.14 0.19 -0.05 -0.02 -0.22
-0.00 -0.20 0.29 0.06 -0.06 0.19 1.11 -0.12 -0.01 -0.15
0.12 0.07 0.05 -0.11 -0.08 -0.05 -0.12 0.73 -0.02 0.15
-0.21 0.11 -0.04 -0.04 -0.04 -0.02 -0.01 -0.02 0.47 -0.03
0.30 0.25 -0.14 -0.08 -0.13 -0.22 -0.15 0.15 -0.03 1.40
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0

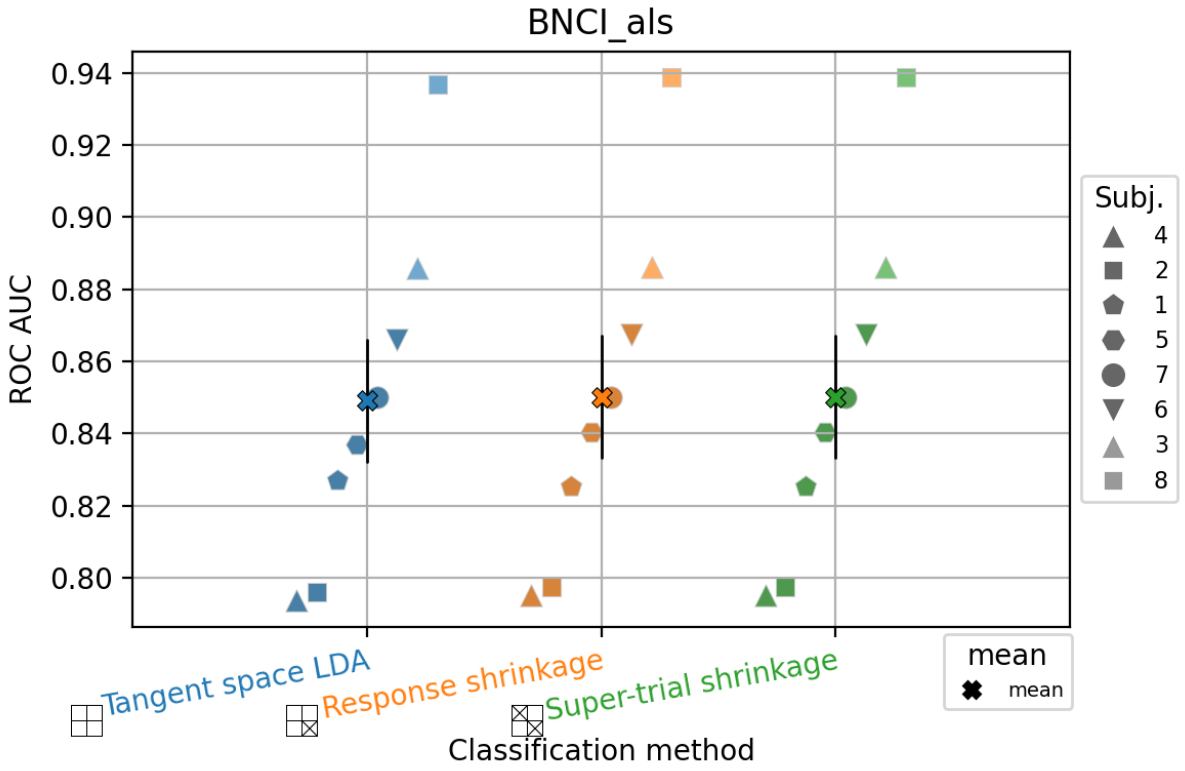
Appendix B

Classification Scores per Dataset




BNCI_1
ROC AUC
0.925
0.900
0.875
0.850
0.825
0.800
0.775
0.750
Tangent space LDA
Response shrinkage
Super-trial shrinkage
Classification method
Subj.
8
7
6
3
1
9
4
2
5
10
mean
mean


BNCI_2
ROC AUC
0.90
0.85
0.80
0.75
0.70
Tangent space LDA
Response shrinkage
Super-trial shrinkage
Classification method
Subj.
6
1
3
7
5
4
10
9
8
2
mean
mean


BNCI_als
ROC AUC
0.94
0.92
0.90
0.88
0.86
0.84
0.82
0.80
Tangent space LDA
Response shrinkage
Super-trial shrinkage
Classification method
Subj.
4
2
1
5
7
6
3
8
mean
mean