UCLouvain

École polytechnique de Louvain

Classification of Biological Signals on the Symmetric Positive Definite Manifold

Author: **Clément Vaes**
Supervisors: **Pierre-Antoine Absil, Estelle Massart**
Readers: **Noémie Jaquier, John Lee**
Academic year 2019–2020
Master [120] in Mathematical Engineering

Abstract

Biological signal classification plays an important role in health-related real-time applications. It has been shown that using a covariance matrix representation of biological signals and exploiting the Riemannian geometry of the space of symmetric positive definite (SPD) matrices, to which covariance matrices belong, enables to ease signal discrimination. Unfortunately, classifying points on high-dimensional SPD manifolds is computationally expensive, so not well-suited for real-time applications. In this thesis, we propose to combine the covariance matrix representation with a structure-preserving dimensionality reduction algorithm. Namely, we define a mapping from a high-dimensional SPD manifold to a lower-dimensional SPD manifold, that preserves the intrinsic structure of the data. We combine this dimensionality reduction algorithm with two classification algorithms on Riemannian manifolds (i.e. minimum distance to Riemannian mean (MDRM) and k-nearest neighbors (KNN)) and test these approaches in two different setups: electroencephalography (EEG) and surface electromyography (sEMG). We show that we can reduce the computational time while gaining classification performance.

Contents

1. Introduction

Imagine a prosthetic arm offering the same movement possibilities as a human arm; imagine a motorized wheelchair controlled with your own thoughts. Reality, science-fiction, or did science-fiction become reality? As crazy as it may sound, science is moving forward at a high pace in this direction. How is this possible, one may think? Thought-controlled wheelchairs and prosthetic devices may appear to have nothing in common, yet their operation is based on the same invisible but measurable entity: biological signals. These are recordings of signals in living beings generated during biological events such as the information processing of a brain, the contraction of a muscle, or the beating of a heart. This work considers two methods for measuring biological signals: electroencephalography (EEG), used to measure the electrical activity from the brain, and surface electromyography (sEMG), used to measure the electrical activity produced by skeletal muscles. Both have recently received much attention within various research fields due to their numerous applications, mostly within the health sector. EEG signals are used to investigate brain activities, to study its neuronal functions and its to study neurophysiological properties. The knowledge acquired through EEG is notably used for the creation of brain-actuated wheelchairs [43] and more widely for medical, brain-computer-interaction, and neuromarketing purposes [24]. SEMG signals give valuable information about neuromuscular activity. These signals play an important role in biomechanical analysis [31] and allow the control of myoelectric prostheses [10]. Many of these applications rely on the ability to treat the measured EEG/sEMG signals by predicting a targeted class over a time window. This class typically corresponds to the movement of the motorized wheelchair or of the myoelectric prosthesis that is attempted by some person.

Therefore, the purpose of this thesis is to build classification algorithms able to deduce which brain (EEG) or hand (sEMG) activity was made by a subject from measured signals. The major difficulties in developing powerful algorithms are (1) that the measured biological signals are often really noisy, which makes the classification difficult, and (2) that most applications need real-time classification which imposes sharp time constraints. In this context, algorithms based on a covariance matrix representation of the measured biological signals have recently been shown to be particularly performant and robust in computer vision [12] and EEG-based brain-computer interfaces [13][14]. To fully exploit the structure of these covariance matrices, we use Riemannian geometry tools in order to classify the covariance matrices. In fact, covariance matrices belong to the space of symmetric positive definite (SPD) matrices also called the SPD manifold. As the classification algorithms considered in this thesis (Minimum Distance to Riemannian

Mean and K-Nearest Neighbors) require a notion of distance between two points, we propose to test various metrics that take the Riemannian structure of the SPD manifold into account.

Moreover, we propose to use a dimensionality reduction algorithm allowing us to map the covariance matrices onto a lower-dimensional SPD manifold by preserving the intrinsic structure of the data. By classifying the resulting points on this smaller SPD manifold, we show that the proposed approach results in lower computational time and improved classification performance.

The outline of this thesis is the following. First, we introduce the EEG and sEMG biological signals in Section 2 and the tools of Riemannian geometry necessary for an understanding of this work in Section 3. Then, we present the classification algorithms used on the SPD manifold in Section 4 and introduce the dimensionality reduction algorithm in Section 5. Finally, we experiment and discuss the proposed approach in two settings: the classification of EEG signals in Section 6 and the classification of sEMG signals in Section 7.

2. Biological signals

Biological signals or biosignals are the recordings of signals in living beings that are generated by biological events. These events generate electrical, chemical, and mechanical activities that often produce measurable and analyzable signals which are of great interest due to the understanding of the underlying physiological mechanisms they offer [16]. This work considers two types of biosignals measurements: *electroencephalography* (EEG) and *surface electromyography* (sEMG). Electroencephalography (resp. surface electromyography) is a way of recording electrical brain activity (resp. electrical muscle activity). These recordings are named electroencephalogram (resp. surface electromyogram) and measured with an electroencephalograph (resp. surface electromyograph).

Both EEG and sEMG rely on the use of sensors which are separated into two main categories: invasive sensors and non-invasive sensors. Although invasive sensors measure higher quality signals, they are particularly impractical as they are expensive and require surgeries which make it difficult to run tests in a laboratory. Therefore, demand for non-invasive, cheap, fast, and real-time sensors is receiving a great interest [32]. However, a significant obstacle in using such sensors is that the measured signal quality is reduced.

In this thesis, we apply classification algorithms on data measured with non-invasive sensors. This is a challenging task because the measured biosignals are of low quality and usual filters do not succeed in successfully getting rid of the noise.

In the following, we introduce both EEG and sEMG signals in detail.

2.1. Electroencephalography

EEG recordings are obtained from sensors placed on the surface of the scalp (see fig. 1). These sensors measure voltage fluctuations resulting from ionic currents within the neurons of the brain. Millions of neurons take part in controlling the behavior of the human body as they are information carriers between the body and the brain. Therefore, signals emitted by neurons are related to events a person lives or experiences. However, although the brain is the central organ of the nervous system, its functioning is only barely known. The EEG signals do not give a full understanding of the complexity of the brain, but may still be used to enlighten some phenomenons. Indeed, human behavior can be seen as a combination of motor and sensory states such as eye movement, lip movement, focus, ... [37]. These states come together with specific signal frequencies inside the brain that are measurable with EEG. This opens the doors to the exploitation

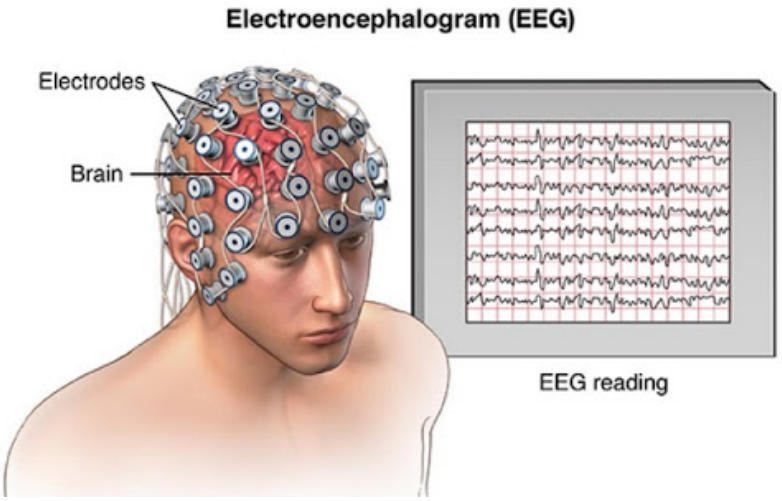


Figure 1: Representation of an electroencephalogram (extracted from *wetalkuav*).

of EEG signals.

2.1.1 SSVEP based brain computer interfaces

In this thesis, we focus on a specific EEG-based application, namely *Steady-State Visually Evoked Potentials* (SSVEP) based *brain-computer interfacing* [39]. A brain-computer interface (BCI) is a computer-based system that acquires brain signals (usually with EEG) and translates them into commands relayed to an output device to carry out the desired action. BCIs are at the center of many research studies and this work in particular due to their varied applications usually seeking to replace or restore useful functions to people disabled by neuromuscular disorders (e.g. stroke, lateral sclerosis, cerebral palsy or spinal cord injury) [38]. From the introductory example, we may find a wheelchair controlled by thoughts within the wide range of BCI applications.

BCIs usually rely on 3 major paradigms [1]:

- **Motor Imagery** is based on the subject kinetically imagining a movement. In fact, studies have shown that imagination activates areas of the brain responsible for generating the actual movement. The brain then emits signals which may be measured and used in applications.
- **Error Related Potential** is usually used to correct BCI errors. It is based on the fact that, when there is a mismatch between a subject's intention to perform a task and the actual response provided by the BCI, the brain generates an error-related potential.
- **Steady-State Visually Evoked Potentials** relies on the fact that when the retina is excited by visual stimulation between $3.5\ Hz$ and $75\ Hz$, the brain generates

electrical signals, mostly emitted in the *occipital* area of the brain, at the same (or a multiple of the) frequency. These signals are called steady-state visually evoked potentials [6].

The example on which we work uses the SSVEP paradigm. Let us consider a subject in a motorized wheelchair facing a screen containing 4 elements flashing at different frequencies and from which EEG data is recorded. Each of these elements is related to one specific movement of the chair (forward, backward, left, and right). By focusing on one specific element of the screen, the subject's brain produces measurable SSVEP. Therefore, the measured EEG signals can be used to deduce which specific element the subject was focusing on and move the chair accordingly. In practice, this is a difficult problem. Advanced classification algorithms are necessary in order to compute to which element the subject was looking at and the algorithms proposed until now have had varying degrees of success [21][30]. This work is therefore dedicated to proposing an improvement of the existing algorithms.

2.1.2 Other EEG applications

EEG is widely used in other domains, including:

- **Neuroscience** Various clinical applications of EEG exist. It is for example used to diagnose sleep disorders, schizophrenia, ... [3][46].
- **Social interaction** For understanding the effect of the brain on social behaviors leading, for example, to interpersonal behavior synchronization [28].
- **Neuromarketing** For understanding the human decision mechanism involved in the purchase of a product or a service [48].

2.2. Surface electromyography

Surface electromyography captures the electrical activity produced during a muscle contraction. Surface refers to the non-invasive type of signal measurement, as opposed to invasive electromyography. The sensors are placed on skin regions immediately above the muscle tissues and the measured electrical signals provide information about the intensity of muscle activation. It is hence possible, by placing sensors above multiple muscles, to understand how a group of muscles work together to execute a movement or a task [11]. The sEMG signals are usually recorded with a so-called *Myo armband* containing multiple electrodes (see fig. 2)

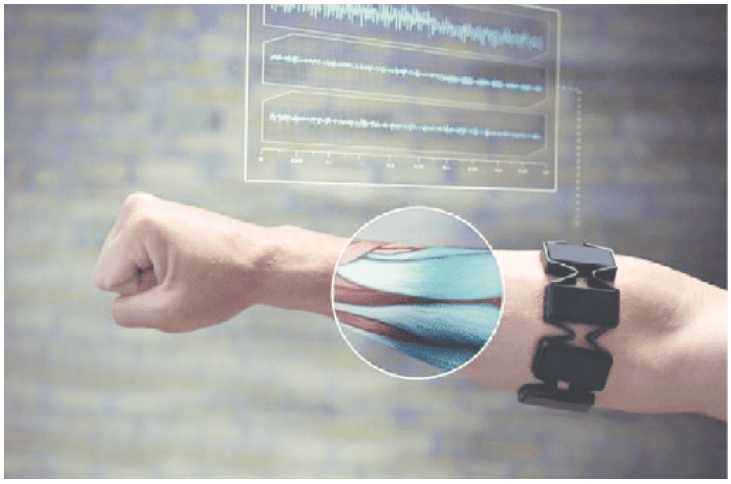


Figure 2: Myo armband [27].

2.2.1 Myoelectric prosthetic hands

The sEMG application on which this work focuses is the treatment of sEMG signals for myoelectric prosthetic hand control. This is a type of prosthesis used for upper limb replacement techniques based on sEMG measurements and dedicated to arm amputees [42]. These amputated subjects, having lost part of their arms due to an accident, trauma or any disease affecting the upper limbs, have still functioning muscles in the remaining part of the arm. The sEMG signals may thus be measured on these remaining muscles to represent hand movements, and inferring motion attempt from sEMG would allow efficient control of prosthetic hands. This way of proceeding is promising because each hand movement requires a specific contribution of arm muscles that may be found in the sEMG signals. Similarly to the EEG signals, classifying sEMG signals is complicated and this work aims proposing improved ways of doing what has been attempted in the literature before. In this thesis, we consider a database containing sEMG signals measured on forearm muscles with a Myo armband similar to the one shown in fig. 2.

2.3. Other sEMG applications

As a tool for understanding muscular activity at an affordable price, sEMG is widely used in fields requiring measurements of muscular activity. Here are some examples [31]:

Gait analysis and the evaluation of neuromuscular disorders Gait constitutes a really specific cyclic and complex movement which involves many muscles at the same time. People for which the gait is affected due to a prosthetic implant, an articular instability, or damaged ligaments may be helped with sEMG. In fact, sEMG allows studying the activation times and intermuscular coordination during this movement. Therefore, identifying disturbances affecting the gait by comparing with sEMG measures of a normal subject is possible.

Evaluating sports performance By the possibility sEMG offers to analyze dynamic situations, they are also widely used in the world of sports. By analyzing a group of muscles when achieving a specific movement, sEMG signals may be used in order to improve the efficiency of a specific movement and prevent an injury

3. Theory of manifolds

This section contains the background on differential geometry required for the understanding of the remaining of this thesis. The interested reader may find more information in the following books [2][25][44].
If familiar with these concepts, the reader may directly head to section 3.2.2.

The motivation for considering differential geometry in this thesis is to consider the intrinsic geometry of non-Euclidean data. We more specifically need to introduce the concept of a Riemannian manifold, which consists of a smooth manifold endowed with a Riemannian metric. Therefore, the outcome of the classification on Riemannian manifolds not only depends on the classifier but also on the Riemannian metric considered.

Considering EEG and sEMG signals, we use Riemannian manifolds for having a correct representation of the signals. In fact, as we consider a covariance matrix representation of these signals, differential geometry allows us to correctly treat their symmetric positive definiteness. Moreover, using Riemannian manifold gives us access to the various approaches that it offers for addressing the classification problem.

3.1. Smooth manifolds

We first introduce the theory of smooth manifolds, the type of manifolds considered along with this work. The notion of a topological manifold is necessary, as smooth manifolds are simply an extension of this definition.

DEFINITION 3.1: The topological manifold

A *topological manifold* is a Hausdorff, second countable, and locally Euclidean space. It is called an n-topological manifold if it is locally Euclidean of dimension n.

In order to extend the definition to smooth manifolds, the concept of a locally Euclidean space of dimension n is worth some clarifications. A space has such a property if every point $\boldsymbol{x}$ of the topological manifold $\mathcal{M}$ has a neighborhood U which is homeomorphic to an open subset of $\mathbb{R}^n$ through a continuous mapping ϕ. The pair (U, ϕ) is usually called a *chart*. Note that two neighborhoods U and V of two different points of a topological manifold may overlap. Some properties of the topological manifold, including its smoothness, depend on how chart mappings act on this intersection.

DEFINITION 3.2: C^∞-compatibility of charts

Two charts (U, ϕ) and (V, ψ) of a topological manifold $\mathcal{M}$ are C^∞-compatible if the following mappings are C^∞ :

$$\phi \circ \psi^{-1} : \psi(U \cap V) \to \phi(U \cap V), \qquad \psi \circ \phi^{-1} : \phi(U \cap V) \to \psi(U \cap V)$$

Moreover, if $U \cap V = \emptyset$, the two charts are then defined as being compatible.
Note that these two functions map open subsets of $\mathbb{R}^n$ to other open subsets of $\mathbb{R}^n$ and the notions of continuity and differentiability are hence well-defined.

As every point of an n-topological manifold has a neighborhood on which a homeomorphic mapping is defined in order to create a chart, we may extend the previous definition to all charts defined on the manifold.

DEFINITION 3.3: The C^∞-atlas

A C^∞-atlas is a collection of charts (ϕ_i, U_i) that cover the whole topological manifold $\mathcal{M}$ and such that all charts are pairwise C^∞-compatible. The atlas is then called a maximal atlas if it is not contained in any other atlas.

We have now covered all tools needed to define a smooth manifold.

DEFINITION 3.4: The smooth manifold

A smooth manifold is a topological manifold endowed with a maximal C^∞-atlas.

The differentiable structure of a smooth manifold allows extending Euclidean-space algorithms to smooth manifolds. Therefore, smooth manifolds are of great interest. We finally complete this part about smooth manifolds by giving an informal definition of what a tangent space defined at any point $\boldsymbol{x}$ of a smooth manifold is: A vector space containing all the possible directions in which one can tangentially pass through $\boldsymbol{x}$. Roughly speaking, it is a linear approximation of the manifold. The tangent space at a point $\boldsymbol{x}$ is be denoted $T_{\boldsymbol{x}}\mathcal{M}$ and its elements are be called *tangent vectors*.

3.1.1 Riemannian manifolds

In this thesis, all our algorithms rely on a definition of distance between two points. Some of these distances derive from a *Riemannian metric*.

DEFINITION 3.5: The Riemannian metric

A Riemannian metric g is defined as a collection of smoothly varying inner products $g_x(\cdot, \cdot)$ (i.e bilinear, symmetric positive definite forms) defined on the tangent space $T_{\boldsymbol{x}}\mathcal{M}$ at any point $\boldsymbol{x}$ of the manifold $\mathcal{M}$.

For a smooth manifold $\mathcal{M}$ endowed with a Riemannian metric g, the resulting couple $(\mathcal{M}, g)$ forms what is known as a *Riemannian manifold*.

While the Euclidean inner product is the natural way to go in a Euclidean space, we later show that this inner product might not always be convenient when working with smooth manifolds. It is hence important to define an adequate Riemannian metric, depending on the manifold we are working with and what we want to achieve in this space.

Distances on Riemannian Manifolds Let's consider a smooth curve

$$\gamma : I \to \mathcal{M}$$

that maps an interval $I \in \mathbb{R}$ onto a subset of the Riemannian manifold. Suppose that I is a closed interval defined as $I = [a, b]$ and that $\boldsymbol{x}$ and $\boldsymbol{y}$ are two points of this manifold. Then, if $\gamma(a) = \boldsymbol{x}$ and $\gamma(b) = \boldsymbol{y}$, γ is then a curve between these points. Its length may then be computed as

$$L(\gamma) = \int_a^b \sqrt{g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t))}\, dt$$

with $g_{\gamma(t)}(\cdot, \cdot)$ being the inner product derived from the Riemannian metric g .

If γ is the curve of infimum length between $\boldsymbol{x}$ and $\boldsymbol{y}$, it is called a *geodesic* and its length is the *Riemannian distance* between the two points for the Riemannian metric considered. It may be shown that this distance defines a *metric* on the manifold.
Geodesics are important tools when dealing with Riemannian manifolds. They may be understood as the generalization of lines to manifolds. Indeed, geodesics are shortest paths on a manifold as straight lines are shortest paths in a Euclidean space.

3.2. The SPD manifold

We have now defined all the necessary notions in order to present a widely used smooth manifold, the *Symmetric Positive Definite* (SPD) manifold.

DEFINITION 3.6: Symmetric Positive Definite Matrices

The set of Symmetric Positive Definite matrices (or SPD) matrices of dimension D, noted $\mathcal{S}_{++}^D$ is defined as:

$$\mathcal{S}_{++}^D = \{\boldsymbol{A} \in \mathbb{R}^{D \times D} \,|\boldsymbol{A} = \boldsymbol{A}^T \ \wedge \ \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} > 0, \ \forall \boldsymbol{x} \in \mathbb{R}^D \text{ and } \boldsymbol{x} \neq 0\}$$

SPD matrices are known to admit a Riemannian manifold structure. Therefore, it is of great interest to exploit the underlying structure of this space when developing algorithms considering SPD matrices. In fact, the space of SPD matrices of any dimension D, combined with an inner product, does not define a *Euclidean space* (vector space with inner product) as SPD matrices do not form a vector space (not closed under additions

or scalar multiplication) as shown in example 3.7.

EXAMPLE 3.7

Let $\boldsymbol{A} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $\boldsymbol{B} = -1 \,.\, \boldsymbol{A} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$. Then

$$\boldsymbol{A} \in \mathcal{S}_{++}^2 \quad \text{as for } \boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} \neq 0 \text{ we have : } \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} = \boldsymbol{x}_1^2 + \boldsymbol{x}_2^2 > 0$$

$$\text{However : } \quad \boldsymbol{B} \notin \mathcal{S}_{++}^2 \quad \text{as for } \boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} \neq 0 \text{ we have : } \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} = -\boldsymbol{x}_1^2 - \boldsymbol{x}_2^2 < 0.$$

This example shows us that the space of SPD matrices does not have the property of closure under scalar multiplication.

This observation explains why using tools of Euclidean geometry is neither natural nor adequate when dealing with SPD matrices. However, classification algorithms still require a notion of distance between 2 different SPD matrices. Although it might be tempting to consider the Euclidean distance for this purpose, it is no longer a suitable metric. Therefore, we need to use other distances that take the Riemannian structure of the space of SPD matrices into account.

3.2.1 Structure of the SPD manifold

The set of SPD matrices of dimension D, noted $\mathcal{S}_{++}^D$, forms a subset of the space of *symmetric matrices* of this same dimension noted Sym_D and defined as:

$$Sym_D = \{x \in \mathbb{R}^{D \times D} | x = x^T\}.$$

Sym_D defines a vector space of dimension $\frac{D(D+1)}{2}$ as it is the number of independent variables in a symmetric matrix of dimension D. Therefore, we may use $\mathbb{R}^{\frac{D(D+1)}{2}}$ to represent this vector space with each point defining a specific symmetric matrix. Having observed this, we may now show that $\mathcal{S}_{++}^D$ forms a convex cone in the space $\mathbb{R}^{\frac{D(D+1)}{2}}$.

DEFINITION 3.8: The convex cone

A subset C of a vector space V is a convex cone if for each $\boldsymbol{X} \in C$, $\boldsymbol{Y} \in C$ and for any $\alpha > 0$, $\beta > 0$: $\alpha \boldsymbol{X} + \beta \boldsymbol{Y} \in C$

PROPOSITION 3.9: $\mathcal{S}_{++}^D$ forms a convex cone in $\mathbb{R}^{\frac{D(D+1)}{2}}$

As it is straightforward that $\mathcal{S}_{++}^D$ is a subset of $\mathbb{R}^{\frac{D(D+1)}{2}}$, we just need to show that for any $\boldsymbol{X} \in \mathcal{S}_{++}^D$, $\boldsymbol{Y} \in \mathcal{S}_{++}^D$ and any scalar $\alpha > 0$, $\beta > 0$: $\alpha \boldsymbol{X} + \beta \boldsymbol{Y} \in \mathcal{S}_{++}^D$

Proof. In order to prove that $\alpha\boldsymbol{X} + \beta\boldsymbol{Y} \in \mathcal{S}_{++}^D$, we show that

$$\forall \boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x}^T(\alpha\boldsymbol{X} + \beta\boldsymbol{Y})\boldsymbol{x} > 0.$$

By developing the product, we get:

$$\alpha(\boldsymbol{x}^T\boldsymbol{X}\boldsymbol{x}) + \beta(\boldsymbol{x}^T Y\boldsymbol{x}) > 0.$$

As $\boldsymbol{X}$ (resp. $\boldsymbol{Y}$) belongs to $\mathcal{S}_{++}^D$, we also have that $\boldsymbol{x}^T\boldsymbol{X}\boldsymbol{x} > 0$ (resp. $\boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{x} > 0$). The equation therefore becomes:

$$\alpha(> \mathbf{0}) + \beta(> \mathbf{0}) > 0,$$

which is always true as $\alpha > 0$ and $\beta > 0$. □

To gain a better intuitive understanding of what this cone represents, we may represent the cone of SPD matrices of dimension 2 graphically, as shown in example 3.10.

EXAMPLE 3.10

Let us consider the general symmetric matrix of dimension 2:

$$\boldsymbol{S} = \begin{pmatrix} \alpha & \beta \\ \beta & \gamma \end{pmatrix}$$

which has 3 independent variables (i.e α, β, and γ).
$\boldsymbol{S}$ is *Symmetric Positive Definite* if it has the additional property that

$$\forall \boldsymbol{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \in \mathbb{R}^2 \ : \ \boldsymbol{z}^t\boldsymbol{S}\boldsymbol{z} > 0.$$

By developing this equation for all $\boldsymbol{z}$, we get:

$$\begin{aligned} & \alpha z_1^2 + 2\beta z_1 z_2 + \gamma z_2^2 > 0 \\ \Leftrightarrow \quad & \alpha\left(z_1 + z_2\frac{\beta}{\alpha}\right)^2 - z_2^2\frac{\beta^2}{\alpha} + z_2^2\gamma > 0 \end{aligned}$$

This inequality is always satisfied if $\alpha > 0$ and $\gamma - \frac{\beta^2}{\alpha} > 0$. By writing the same inequality for γ, we get the last requirement that $\gamma > 0$.

This allows us to represent the SPD cone within the 3-dimensional Euclidean space having as axis α, β, and γ:

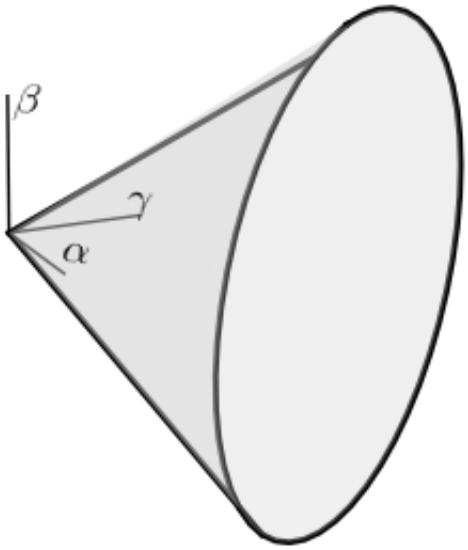


Figure 3: Representation of the SPD cone in $\mathbb{R}^3$.

3.2.2 Endowing the SPD manifold with a distance function

For classification purposes in the SPD manifold, it is essential to define a distance function between any two points of the manifold. Various distances and divergences are proposed in the literature to endow the SPD manifold with. This thesis considers four different definitions: the *Euclidean metric*, the *Log-Euclidean metric*, the *Affine-Invariant Riemannian metric* and the *Stein divergence*.

Euclidean metric This metric is the common way to measure distances between two square matrices. Being defined between any two square matrices, we might define it on the SPD manifold by simply restricting the metric to this space. It is then simply defined as

$$\delta_e(\boldsymbol{X}, \boldsymbol{Y}) = \|\boldsymbol{X} - \boldsymbol{Y}\|_{\mathcal{F}}, \tag{1}$$

where, $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ and $\|\cdot\|_{\mathcal{F}}$ refer to the Frobenius inner product and the Frobenius norm, respectively. These are defined as

$$\begin{aligned} \langle \boldsymbol{A}, \boldsymbol{B} \rangle_{\mathcal{F}} &= \mathrm{Tr}(\boldsymbol{A}^T \boldsymbol{B}), \\ \|\boldsymbol{A}\|_{\mathcal{F}} &= \sqrt{\mathrm{Tr}(\boldsymbol{A}^T \boldsymbol{A})}. \end{aligned}$$

The Affine-Invariant Riemannian Metric The most popular metric when dealing with manifolds is the Affine-Invariant Riemannian metric also called AIRM and proposed by *,Pennec et al.* [35]. Here, we do not enter into the details of how the metric was built. Instead, we simply define the Riemannian metric and the induced AIRM between two points of the SPD manifold [18] :

DEFINITION 3.11: The Affine-Invariant Riemannian Metric

By defining an inner product at any point $\boldsymbol{X}_1$ of the SPD manifold between two points $\boldsymbol{H}_1, \boldsymbol{H}_2 \in T_{X_1}\mathcal{M}$ as

$$\langle \boldsymbol{H}_1, \boldsymbol{H}_2 \rangle_{\boldsymbol{X}_1} = \langle \boldsymbol{X}_1^{-1/2} \boldsymbol{H}_1 \boldsymbol{X}_1^{-1/2}, \boldsymbol{X}_1^{-1/2} \boldsymbol{H}_2 \boldsymbol{X}_1^{-1/2} \rangle_{\mathcal{F}},$$

the induced Riemannian length (called the AIRM) between two points $\boldsymbol{X}_1$,$\boldsymbol{X}_2$ of the manifold is

$$\delta_a^2(\boldsymbol{X}_1, \boldsymbol{X}_2) = \left\| \log(\boldsymbol{X}_1^{-1/2} \boldsymbol{X}_2 \boldsymbol{X}_1^{-1/2}) \right\|_{\mathcal{F}}^2.$$

The affine invariance property The AIRM metric has some interesting properties among which the affine invariance from which the metric takes its name. Before defining affine invariance, we first need to introduce the general linear group.

DEFINITION 3.12: The general linear group

$GL(n)$ is defined as the general linear group of dimension n (i.e. set of $n \times n$ invertible matrices with the usual product operation).

Property. *For $\boldsymbol{M} \in GL(n)$, we say that a metric inducing a distance $d^2(\boldsymbol{X}; \boldsymbol{Y})$ between two points $\boldsymbol{X}$, $\boldsymbol{Y}$ of the SPD manifold is affine-invariant if [17]:*

$$d^2(\boldsymbol{X}, \boldsymbol{Y}) = d^2(\boldsymbol{M}\boldsymbol{X}\boldsymbol{M}^T, \boldsymbol{M}\boldsymbol{Y}\boldsymbol{M}^T).$$

We comment on this property in particular as it allows later in this thesis to add an essential constraint to the dimensionality reduction optimization problem (see section 5).

The Stein divergence Although the AIRM is widely used in the research field and fulfils the need of taking into account the Riemannian structure of the set of SPD matrices, it is unfortunately computationally expensive.
An alternative definition which is an approximation of AIRM with a smaller computational cost was later proposed by *Sra* [41]: the Stein divergence, a symmetric type of Bregman divergence. The Stein divergence is defined between any two points $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$ of the SPD manifold as:

$$\delta_s^2(\boldsymbol{X}_1, \boldsymbol{X}_2) = \log det \left(\frac{\boldsymbol{X}_1 + \boldsymbol{X}_2}{2} \right) - \frac{1}{2} \log det(\boldsymbol{X}_1 \boldsymbol{X}_2)$$

This divergence has the particularity of not being a distance as it does not satisfy the triangle inequality. However, it has some interesting properties and advantages compared to AIRM. In fact, it is easier to use and is computationally less expensive. The Stein

divergence also possesses the affine invariance property and respects non-Euclidean geometry.

Log-Euclidean metric *Arsigny et al.* proposed [4] a new family of Riemannian metrics called Log-Euclidean. Here, we do not enter into the details of how the metric was built. Instead, we define the Riemannian metric and its induced Log-Euclidean metric between any two points of the SPD manifold [18] :

DEFINITION 3.13: The Log-Euclidean metric

By defining an inner product at any point X_1 of the SPD manifold between two points $\boldsymbol{H}_1, \boldsymbol{H}_2 \in T_{\boldsymbol{X}_1}\mathcal{M}$ as

$$\langle \boldsymbol{H}_1, \boldsymbol{H}_2 \rangle_{\boldsymbol{X}_1} = \langle \mathbb{D}\log(\boldsymbol{X}_1)[\boldsymbol{H}_1], \mathbb{D}\log(\boldsymbol{X}_1)[\boldsymbol{H}_2] \rangle,$$

where $\mathbb{D}\log(\boldsymbol{X}_1)[\cdot]$ denotes the directional derivate at $\boldsymbol{X}_1$, the induced Riemannian length (called the Log-Euclidean metric) between two points $\boldsymbol{X}_1$,$\boldsymbol{X}_2$ of the manifold is

$$\delta_l^2(\boldsymbol{X}_1, \boldsymbol{X}_2) = \|\log(\boldsymbol{X}_1) - \log(\boldsymbol{X}_2)\|_{\mathcal{F}}^2 .$$

As for the Affine-Invariant Riemannian metric, this metric has excellent theoretical properties yielding similar results, but with less computational cost. This metric is not affine invariant but possesses other interesting affine properties such as invariance by inversion or by log multiplication.

Properties of the four defined distances For simplicity, even if the Stein divergence is not a metric but a divergence, we use the word metric or distance to refer to all four distances/divergence in the remainder of this thesis. This work alternatively uses each of the presented distances. The reason why we do not choose one specific distance to endow the SPD manifold with is because they all have specific properties that may be interesting in our applications. These properties are gathered in table 1 [47]. One metric is not a priori better than the other ones but just defines the distance between SPD matrices of the manifold in a different way. Hence, when using algorithms on the SPD manifold, using one metric or another affects the outcome of the algorithms and, therefore, may lead to different performances. Moreover, the different metrics also have different computational complexities, which is an important feature to consider when performing real-time classification. For these reasons, the four distances are used and compared throughout this thesis.

		Property		
		Symmetric	Triangle inequality	Inversion invariance
Metric	Euclidean	Yes	Yes	No
	AIRM	Yes	Yes	Yes
	Stein	Yes	No	Yes
	Log-Euclidean	Yes	Yes	Yes
		Affine invariance	Scale invariance	Rotation invariance
Metric	Euclidean	No	No	Yes
	AIRM	Yes	Yes	Yes
	Stein	Yes	Yes	Yes
	Log-Euclidean	No	Yes	Yes

Table 1: Distances defined on the SPD manifold and their properties

3.3. Stiefel and Grassmann manifolds

The Stiefel and the Grassmann manifold are used in section 5 in order to solve the dimensionality reduction optimization problem. In fact, in order to find an optimized mapping between a high-dimensional SPD manifold and a lower-dimensional one, we search for a matrix $\boldsymbol{W}$ belonging to one of these spaces.

The *Stiefel manifold* is an embedded submanifold of $\mathbb{R}^{D\times d}$ defined as [9] :

$$St(d,k) = \{\boldsymbol{U} \in \mathbb{R}^{D\times d} : \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_k\}, \tag{2}$$

where $\boldsymbol{I}$ is the identity matrix of dimension k. Then, by defining the orthogonal group

$$\mathcal{O}(d) = \{\boldsymbol{Q} \in \mathbb{R}^{d\times d} : \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}_k\}, \tag{3}$$

we may define an equivalence relation $\sim$ between two points of the Stiefel Manifold. This latter is written

$$\boldsymbol{U} \sim \boldsymbol{V} \iff \boldsymbol{U} = \boldsymbol{V}\boldsymbol{Q} \text{ for some } \boldsymbol{Q} \in \mathcal{O}(d). \tag{4}$$

Then, it allows us to defined equivalence classes as

$$[\boldsymbol{U}] = \{\boldsymbol{V} \in St(D,d) : \boldsymbol{U} \sim \boldsymbol{V}\}. \tag{5}$$

The set of all equivalence classes is called the *Grassmann manifold* and is written $Gr(D,d)$. More intuitively, the Stiefel manifold refers to all the orthogonal d-frames in $\mathbb{R}^D$ while the Grassmann manifold represents all the d-dimensional subspaces of $\mathbb{R}^D$.

4. Classification on the SPD manifold

The purpose of this section is to present the algorithms used in order to classify EEG and sEMG signals on the SPD manifold. Instead of developing algorithms to classify the raw measured signals, we use the spatial covariance matrix representation of the measured signals. As spatial covariance matrices are SPD, we aim for a classification on the SPD manifold. By classification, we refer to the task of assigning a data point to a set of categories. The categories refer to the frequency of the SSVEP stimulation concerning the EEG signals and the executed hand movement for the sEMG signals. This section first motivates the covariance matrix approach to classify the signals, followed by an explanation of the algorithms used on the SPD manifold and an introduction to the performance measures considered.

4.1. Working with covariance matrices

Whether data acquisition is made with EEG or EMG, the underlying idea is similar. D sensors are placed on different regions of the scalp (EEG) or of the forearm (EMG) of a certain subject. Sensors measure the electrical activity at a sampling frequency f during a time window T. During this time window, the visual stimulation oscillates at a specific frequency (for EEG) or a specific hand movement is being executed by the subject (for sEMG). We may then gather all data in a matrix $\boldsymbol{X} \in \mathbb{R}^{D \times m}$ representing one specific class, where $m = f \times T$ represents the number of measurements made by a sensor during the time window. Instead of considering this matrix $\boldsymbol{X}$ as such, we estimate the spatial covariance matrix, considering that the signals are centered, as

$$\boldsymbol{C} = \frac{1}{m-1} \boldsymbol{X} \boldsymbol{X}^T \in \mathbb{R}^{D \times D}. \tag{6}$$

We may notice that the estimator built in eq. (6) actually defines a SPD matrix of dimension D. By doing this process for all the measured signals, we finally get data points belonging to $\mathcal{S}_{++}^D$ with each of the points representing a specific class. Let us insist here on the importance of the time window T. Its value must be chosen depending on the intended application and by keeping in mind that (1) a time window should not be too long because it does not suit for real-time applications and that (2) a time window should not be too short because the estimated spatial covariance matrices do not correctly translate the correlation between the signals measured by the different sensors.

This way of exploiting biological signals has shown its robustness and performance in fields like computer vision [12] and BCI based on EEG data [13] [14]. Several

reasons lead to this choice [45]:

- The covariance of a set of signals is usually enough to discriminate it from another set of signals;
- Covariance matrices offer a way to fuse multiple features which might be correlated;
- The noise corrupting the signals, which is important and makes it difficult to classify raw signals (even after filtering), are largely filtered out during covariance computation;
- Computed covariance matrices ($\in \mathbb{R}^{D\times D}$) have only $\frac{D(D+1)}{2}$ independent variables by symmetry, compared to the raw signals ($\in \mathbb{R}^{D\times m}$) which have $m \cdot D$ independent variables.

For these reasons and the efficiency it has shown in the state of the art, measured EEG/EMG signals are, in this thesis, always transformed into covariance matrices before doing classification.

4.2. Introduction to classification methods

A lot of research has been made in order to develop classification algorithms in the Euclidean space. Here, we take advantage of the structure of the covariance matrices as we know that SPD matrices belong to a Riemannian manifold. Therefore, this section introduces classification algorithms commonly used in Euclidean spaces but adapted for a use in the SPD manifold. Two common classification algorithms are considered: *Minimum Distance to Riemannian Mean* (MDRM) and k*-Nearest-Neighbors* (KNN). These algorithms were chosen because of their simplicity and their low computational cost. Therefore, they are practical for real-time applications and allow us to test the dimensionality reduction approach used in this thesis.

As explained previously, measurements of all sensors on one single time window are represented in a covariance matrix belonging to one class. Classification methods rely on the assumption that we have at our disposal several data points for each of the different classes [1]. It is essential to note that the classification methods developed in this work are tested on a fixed dataset. Therefore, the data points are separated into a *training set* and a *test set*. The training set is used in order to train the classification algorithms and is considered of *fixed size*, while the test set is used in order to evaluate to performance of our classification algorithms. Naturally, we want to build algorithms able to assign a new incoming point to the correct class.

[1] The data points refer to points on the SPD manifold (i.e SPD matrices)

EXAMPLE 4.1

In order to illustrate our words, fig. 4 gives an example of a classification problem in $\mathbb{R}^2$ with two different classes. We have available training points for each of the two different classes and wish to classify a test point for which we do not know the class beforehand.
This is a simplified classification problem similar to what we wish to achieve with the EEG/EMG data.

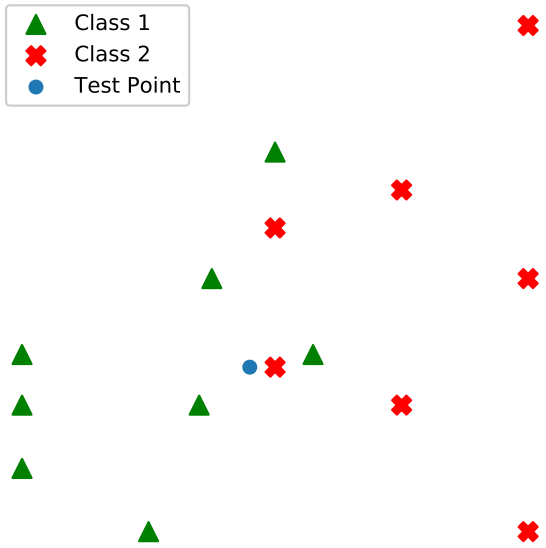


Figure 4: Classification problem in $\mathbb{R}^2$.

Formulation of a classification problem in a Euclidean space Let's consider a generic classification problem containing C classes. We have at our disposal some training points $\boldsymbol{x}_{c,i} \in \mathbb{R}^n$ with $c \in \{1, \dots, C\}$ being the index of the class and $i \in \{1, \dots, N_c\}$ being the index of the training point within this class. These points represent each of the C classes and our goal is to assign the correct class to some point $\boldsymbol{y} \in \mathbb{R}^n$. This example is used below to introduce both of the classification methods.

4.3. Minimum distance to Riemannian mean

This algorithm is an implementation of the Euclidean *Minimum Distance to Mean* (MDM) algorithm on a Riemannian manifold. We first present the algorithm in a Euclidean space before implementing it on the SPD manifold.

4.3.1 MDM in a Euclidean space

Let us consider the generic classification problem introduced above. If we want to classify a point $\boldsymbol{y} \in \mathbb{R}^n$, the MDM algorithm is executed as follows:

1. For each of the classes $c \in \{1, \dots, C\}$, we compute the mean of the data points belonging to c. This mean is computed as $\boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \boldsymbol{x}_{c,i}$.

2. Solve $\arg\min_{c \in \{1,\dots,C\}} \delta(\boldsymbol{y}, \boldsymbol{\mu}_c)$ where $\delta(\cdot, \cdot)$ is the usual Euclidean distance. It is defined between any two points $\boldsymbol{x}_{c_1,i_1}, \boldsymbol{x}_{c_2,i_2} \in \mathbb{R}^d$ as :

$$\delta^2(\boldsymbol{x}_{c_1,i_1}, \boldsymbol{x}_{c_2,i_2}) = \sum_{j=1}^{n} (x_{c_1,i_1,j} - x_{c_2,i_2,j})^2,$$

   where $x_{c,i,j}$ refers to the jth component of the point $\boldsymbol{x}_{c,i}$.

For a training set of fixed size, this algorithm is computationally efficient as the means of the classes just need to be computed once from the training set. Therefore, to classify a new point, we simply compute the distances between this point and the C computed means and select the class whose mean is closest to the point.

EXAMPLE 4.2

Figure 5 shows an example of MDM in $\mathbb{R}^2$ (same problem as in example 4.1). We may see that after computing the mean of both classes from the training points, the test point is closer to the mean of $class\ 1$ than to the mean of $class\ 2$. Therefore, MDM assigns this test point to $class\ 1$.

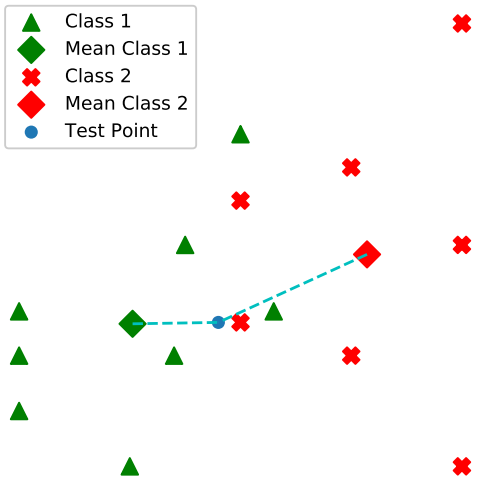


Figure 5: MDM classification in $\mathbb{R}^2$.

4.3.2 Implementation of MDM on the SPD manifold

As we have seen above, MDM completely relies on the definition of the distance between two points and the mean of a set of points. Even though these concepts are well-defined and have no ambiguity in a Euclidean space, it needs to be clarified in any Riemannian Manifold (the SPD manifold here). In section 3.2.2, we presented what distances we consider in the SPD manifold. The only remaining question is: how do we define the mean of a set of points on this manifold? This concept is obvious in Euclidean space but not straightforward in the SPD manifold. Here, we stick to a metric concept independent of the manifold structure known as the Karcher mean [7].

DEFINITION 4.3: The Karcher mean

Let us consider a Riemannian manifold $\mathcal{M}$ endowed with a metric $\delta_{\mathcal{M}}$ that defines a notion of distance between any 2 points of the manifold.
For a set of n points $\boldsymbol{x}_i \in \mathcal{M}$, the Karcher mean is defined as

$$\boldsymbol{\mu} = \underset{\boldsymbol{p}\in\mathcal{M}}{\arg\min} \sum_{i=0}^{N} \delta^2_{\mathcal{M}}(\boldsymbol{p}, \boldsymbol{x}_i).$$

Computing the Karcher mean requires solving an optimization problem. We will not enter into the details of how we computed these means as we used the Python implementations available at `https://github.com/alexandrebarachant/pyRiemann/blob/master/pyriemann/utils/mean.py`. With the definition of the mean associated with each metric on the SPD manifold, we may define the construction steps of the *Minimum Distance to Riemannian Mean* algorithm on the SPD manifold for a chosen metric $\delta_{\mathcal{M}}$

1. Compute the Karcher mean of the points that compose each class c. This latter is computed as $\boldsymbol{\mu}_c = \arg\min_{\boldsymbol{p}\in\mathcal{M}} \sum_{i=0}^{N_c} \delta^2_{\mathcal{M}}(\boldsymbol{p}, \boldsymbol{x}_{c,i})$.

2. Solve $\arg\min_{c\in\{1,\dots,C\}} \delta_{\mathcal{M}}(\boldsymbol{y}, \boldsymbol{\mu}_c)$. The distance between any two points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{M}$ being computed according to the chosen metric.

Let us now give an example for visualizing how metrics may influence the mean of a set of SPD matrices.

EXAMPLE 4.4: Comparison of metrics on the SPD manifold

Example 3.10 showed how the SPD cone representing $\mathcal{S}^2_{++}$ was built. Here, we add 2 points on this cone and show how distances and means change according to the chosen metric. Figure 6 shows both a Euclidean path and an AIRM geodesic between these points (The two other metrics are not represented here as they look very similar to the AIRM). Informally speaking, fig. 6 gives a visualization of the shortest paths between two points of the SPD manifold by considering two different metrics. This

gives an understanding of the importance of the metric choice and the impact it may have on classification algorithms. The conclusion is similar regarding the means, as they are on the path/geodesic at halfway between the two points. Therefore, the resulting means are also highly dependent on the metric choice.

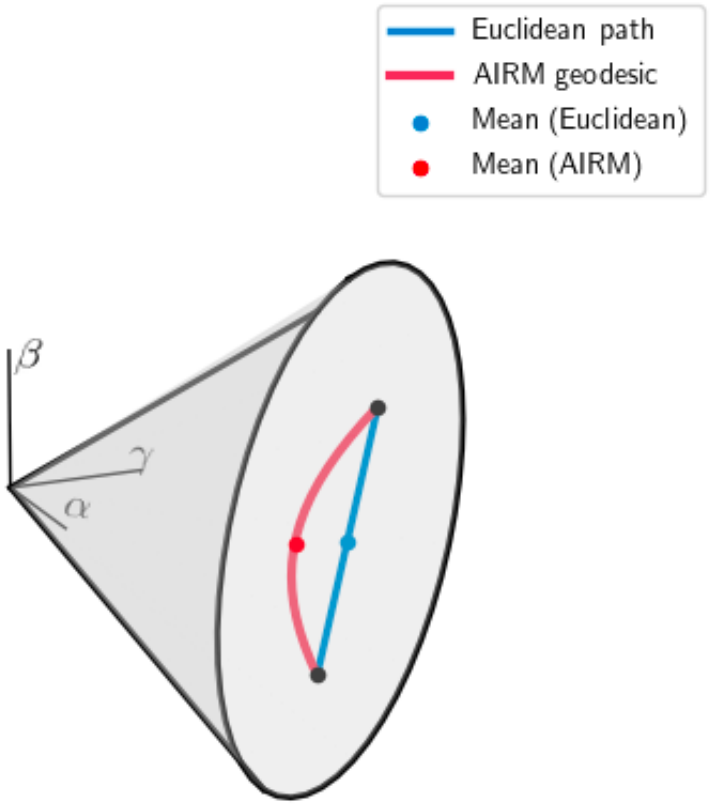


Figure 6: SPD cone with means.

4.4. K-nearest neighbors

Here, we introduce the second classification algorithm used in this thesis, namely *k-nearest neighbors* (KNN). As the name indicates, the algorithm computes the k nearest neighbors to some point we wish to classify, and assigns it the most represented class among these k points. Therefore, this intuitive algorithm relies on a single important parameter: k. As the underlying concept is similar in the Euclidean space and on the SPD manifold, we first recall its functioning in Euclidean space and then explain how the transition to the SPD manifold is made.

4.4.1 KNN in Euclidean space

We consider again the generic classification problem introduced above. Since we want to classify a point $\boldsymbol{y} \in \mathbb{R}^n$, the k-nearest neighbors algorithm is executed as follows:

1. Compute the Euclidean distance between $\boldsymbol{y}$ and each of the training points $\boldsymbol{x}_{c,i}$.

2. Select the k nearest training points to $\boldsymbol{y}$.

3. Assign $\boldsymbol{y}$ to the most represented class within the previously calculated set of k nearest points. If some classes are equally represented within the k nearest point, we randomly select a class among these classes.

Note that the distance used here is the usual Euclidean distance.

EXAMPLE 4.5: KNN in $\mathbb{R}^2$

This example helps us understanding how KNN works in $\mathbb{R}^2$. We consider the classification problem presented in example 4.1. Figure 7 shows how the algorithm classifies the *test point* using KNN for the parameter $k = 1$ or $k = 4$. This example shows the importance of this parameter. When $k = 1$, we are interested only in the closest point to our test point, which belongs to *class 2*. When $k = 4$, we may see that among the 4 closest points, 3 belong to *class 1* while only 1 belongs to *class 2*. For these reasons, when $k = 1$, KNN assigns our test point to *class 2*. However, when $k = 4$, KNN assigns our test point to *class 1*. These results show the importance of the parameter *k* and how it might completely alter classification results. In practice, we might either fix this parameter via cross-validation or empirically assign it a reasonable value.

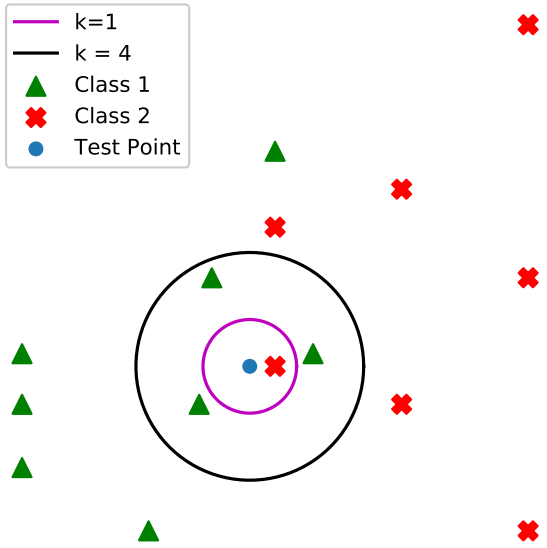


Figure 7: Classification with KNN in $\mathbb{R}^2$.

4.4.2 Implementation of KNN on the SPD manifold

This algorithm gets easily extended to the SPD manifold. In fact, it is simply obtained by using the metric chosen to endow the SPD manifold with, instead of the Euclidean distance. Notice that the KNN algorithm on the SPD manifold thus depends on the parameter k, but also on the metric choice.

4.5. Performance measures

As already explained, our data points are separated into a training set and a test set. Classification is performed on the test set, assuming that we do not know the class to which the test points belong. Therefore, we get after classification:

- A vector containing the actual classes of the test set;
- A vector containing the predicted classes of the test set by the classifier.

A question is legitimate : how de we deduce the classifier's performance from these two vectors? For this purpose, several metrics are possible [15] [33], but the choice of the metric depends on what we seek to achieve. In this work, we use two common metrics : the *F1 micro score* and the *F1 macro score*. Firstly, the F1 micro score, equivalent to the standard accuracy in a multi-class setting, translates the overall performance of the classifier. On the other hand, the F1 macro score averages a notion of performance over the individual classes and allows us to properly treat class imbalance. Therefore, these two metrics suit our requirements and give together a good evaluation of the performance of our classifiers. The F1 micro score and the F1 macro score propose different ways of interpreting confusion matrices in multi-class settings. Let us first give a toy example to understand confusion matrices in binary-class settings.

EXAMPLE 4.6: Confusion matrix for binary classifiers

Suppose we have a classifier for deciding whether a tumor is benign or malignant that was trained and tested on different sets. The results of the classification on the test set allow us to define a confusion matrix for this classifier:

Actual class / **Predicted class**	***malignant***	***benign***
malignant	*True Positives* (TP)	*False Positives* (FP)
benign	*False Negatives* (FN)	*True Negatives* (TN)

Let's clarify what the entries of this confusion matrix represent:

- TP is the amount of tumors which have been correctly classified as malignant;
- TN is the amount of tumors which have correctly been classified as benign;

- FP is the amount of tumors which have wrongly been classified as malignant;
- FN is the amount of tumors which have wrongly been classified as benign.

Confusion matrices are handy because they allow us to easily define several measures of performance [19], among which we find:

$$\text{Precision} = \frac{TP}{TP + FP};$$
$$\text{Recall} = \frac{TP}{TP + FN}.$$

Precision is the percentage of a result which is relevant while recall is the percentage of relevant results which were correctly classified. For the tumor classification problem, we may easily conclude that the recall is more important than the precision. In fact, a good precision means that when the algorithm classifies a good tumor as malignant, it is most probably true. However, if it the classification algorithm has a low recall, it means that most malignant tumors are classified as benign. Different measures translate different behaviors. Therefore, we should always choose performance measures wisely.

The confusion matrix for binary classification can be generalized to a multi-class setting.

DEFINITION 4.7: Confusion matrices in multi-class settings

Let us consider a multi-class classification problem with C classes. Data points have been separated between a training set on which we trained a classifier and a test set on which we tested this same classifier. For the test set, we know the vector of the actual test classes and a vector of predicted test classes by the classifier. We may then define a confusion matrix for each *class i* as:

Actual class / **Predicted class**	***class i***	**Other than *class i***
class i	TP_i	FP_i
Other than *class i*	FN_i	TN_i

This multi-class setting confusion matrix allows us to define the F1 micro score and the F1 macro score.

4.5.1 F1 micro score

Definition of the metric By defining the micro average precision and the micro average recall as

$$P_{micro} = \frac{\sum_{c=1}^{C} TP_i}{\sum_{c=1}^{C} TP_i + FP_i}, \text{ and } R_{micro} = \frac{\sum_{c=1}^{C} TP_i}{\sum_{c=1}^{C} TP_i + FN_i},$$

the F1 micro score is

$$F1_{micro} = 2\frac{P_{micro}R_{micro}}{P_{micro} + R_{micro}}.$$

Discussion of the metric We first need to mention that both the micro average precision and the micro average recall are computed by considering all samples together. Moreover, as a false negative of one class is considered as a false positive for the class to which the point was assigned by the classifier, the number of false negatives and of false positives considering all classes together are similar. Therefore, the micro average precision and the micro average recall are equal. As the F1 micro score is just the harmonic mean of these two values, we have

$$\text{Micro average precision} = \text{Micro average recall} = \text{F1 micro score}.$$

Also notice that it is simply equivalent to the standard accuracy (proportion of correct predictions) which is the most widely used measure of performance.

4.5.2 F1 macro score

By defining the macro average precision and the macro average recall as

$$P_{macro} = \frac{1}{n_C}\sum_{i=1}^{n_C} \frac{TP_i}{TP_i + FP_i} = \frac{\sum_{i=1}^{n_C} P_i}{n_C}, \text{ and } R_{macro} = \frac{1}{n_C}\sum_{i=1}^{n_C} \frac{TP_i}{TP_i + TN_i} = \frac{\sum_{i=1}^{n_C} R_i}{n_C},$$

the F1 macro score is

$$F1_{macro} = 2\frac{P_{macro}R_{macro}}{P_{macro} + R_{macro}}.$$

Discussion of the metric The F1 macro score is the harmonic mean of the macro average precision and the macro average recall, where the average is taken over the different classes. This means that this score is sensitive to the performance of the algorithm for individual classes, which is not the case for the F1 micro score.

Several comments need to be made on this metric. Firstly, when defined as such, all classes are considered to have the same importance. If we wish to give some more weight to some classes, we could also compute a weighted average of the precision and a weighted average of the recalls. This is not done in this work as we do not wish to give more importance to one class when compared to the others. Another important point to notice is that the F1 macro score gives equal weights to recall and precision, which might appear to be a big flaw. The problem is easy to understand if we consider the toy example given in example 4.6. The cost of classifying a malignant tumor as benign is not the same as classifying a benign tumor as malignant. This problem is similar to the wheelchair application of EEG signals. The cost of speeding up the wheelchair when we want it to slow down is not the same as the cost of slowing it down when we want to speed it up. As we are not interested in one precise application but rather in the improvement of existing classification algorithms based on EEG and sEMG signals, the F1 micro score and the F1 macro score satisfy our needs. However, for a future work which is oriented in a specific application, performance metrics giving a cost to each possible choice of the classifier could be explored.

4.5.3 Use of the performance measures

When dealing with a classification problem for which each class contains the same number of data points, it is called a *balanced* classification problem. In this case, the F1 micro score and the F1 macro score are similar. In fact, the F1 macro score averages precision and recall over the different classes, which is hardly different than averaging precision and recall over the points. Therefore, when classes are balanced, we use either one of them.

When dealing with an *unbalanced* classification problem, where each class does not contain the same number of data points, both measures may differ considerably. In fact, as the F1 micro score does not take the classes of the points into consideration, it might be misleading. If we take the example of a classification problem which has an overrepresented class, the F1 micro score gives more weight to this class as it contains more points. However, the F1 macro score gives equal weights to all the classes. Therefore, when dealing with an unbalanced classification problem, both metrics may be used as they enlighten different behaviors of the classifier.

5. Dimensionality reduction on the SPD manifold

One of the objectives of this thesis is to classify EEG/sEMG signals for real-time classification purposes. On a certain time window, we build an estimator for the spatial covariance matrix with the signals measured by the sensors. Therefore, classification is made from points belonging to $\mathcal{S}_{++}^{D}$, where D is the number of sensors. While several works exploited the aforementioned spatial covariance representation, we propose here to combine it with a dimensionality reduction algorithm. This approach is mainly motivated by the fact that achieving classification in a smaller space asks for smaller time complexity which is an essential and often limiting factor in real-time classification. Another motivation is to reduce the noise present in the original SPD manifold to have a better classification performance. For this purpose, we use a dimensionality reduction algorithm proposed by *Harandi et al.* in [17] that maps points from a high-dimensional SPD manifold onto a lower-dimensional and more discriminative SPD manifold. This dimensionality reduction algorithm keeps the SPD structure of the data points while grouping points from a same class and separating points from different classes. Therefore, it fits the requirements of our EEG and sEMG classification problems.

Dimensionality reduction is used in the resolution of our classification problem. In fact, we first use the training set at our disposition to learn the parameters allowing us to define an optimized mapping between the high-dimensional SPD manifold and the lower-dimensional one. This mapping is used to project all our training point on this lower-dimensional manifold. Then, we train the considered classifier on this SPD manifold. Classifying a point is done in two steps: (1) projecting the point onto the lower-dimensional SPD manifold with the mapping found previously and (2) classifying the projection of the point with the previously trained classifier.

5.1. Geometry aware dimensionality reduction

The approach consists in representing high-dimensional SPD matrices in a lower-dimensional and more discriminative SPD manifold. This is done by looking for a matrix $\boldsymbol{W} \in \mathbb{R}^{D\times d}$ that defines a generic mapping $f : \mathcal{S}_{++}^{D} \times \mathbb{R}^{D\times d} \to \mathcal{S}_{++}^{d}$ that would keep the interesting structure of the data as:

$$f(\boldsymbol{X}, \boldsymbol{W}) = \boldsymbol{W}^T \boldsymbol{X} \boldsymbol{W}. \tag{7}$$

We may easily check that if in eq. (7) $\boldsymbol{X} \in \mathcal{S}_{++}^{D}$ and $\boldsymbol{W} \in \mathbb{R}^{D\times d}$ is of full rank, then $f(\boldsymbol{X}, \boldsymbol{W}) \in \mathcal{S}_{++}^{d}$.

PROPOSITION 5.1

If $\boldsymbol{X} \in \mathcal{S}_{++}^{D}$ and $\boldsymbol{W} \in \mathbb{R}^{D \times d}$ a full rank matrix then $\boldsymbol{W}^T \boldsymbol{X} \boldsymbol{W} \in \mathcal{S}_{++}^{d}$.

Proof. If $\boldsymbol{W}$ is of full rank:

$$\begin{aligned} \forall \boldsymbol{x} \in \mathbb{R}^d \setminus \{0\} \quad &: \quad \boldsymbol{W}\boldsymbol{x} \neq 0 \\ \Rightarrow \forall \boldsymbol{x} \in \mathbb{R}^d \setminus \{0\} \quad &: \boldsymbol{x}^T(\boldsymbol{W}^T\boldsymbol{X}\boldsymbol{W})\boldsymbol{x} = (\boldsymbol{W}\boldsymbol{x})^T\boldsymbol{X}(\boldsymbol{W}\boldsymbol{x}) = \boldsymbol{y}^T\boldsymbol{X}\boldsymbol{y} > 0 \end{aligned}$$

as $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} \in \mathbb{R}^D \neq 0$ and $\boldsymbol{X}$ is symmetric positive definite. □

A first question that comes up is how to determine $\boldsymbol{W}$. As the goal is to solve a classification problem, we want to improve the distribution of the points on the SPD manifold in order to ease classification. A reasonable and simple idea considering the classifiers we use in this work is to push away points of different classes and bring together points of a same class as shown in fig. 8.
Let's consider a set of matrices $\boldsymbol{X}_i \in \mathcal{S}_{++}^{D}$ $\forall i \in \{1, \ldots, p\}$. In order to find the parameters of the matrix $\boldsymbol{W} \in \mathbb{R}^{D \times d}$ to build the mapping f of eq. (7), it is proposed in [17] to define a pairwise cost function between any $\boldsymbol{X}_i$, $\boldsymbol{X}_j$ as :

$$\mathcal{J}_{i,j}(\boldsymbol{W}; \boldsymbol{X}_i, \boldsymbol{X}_j) = \boldsymbol{A}_{i,j}\delta^2(\boldsymbol{W}^T\boldsymbol{X}_i\boldsymbol{W}, \boldsymbol{W}^T\boldsymbol{X}_j\boldsymbol{W}),$$

where $\delta(\cdot, \cdot)$ refers to the distance defined on the SPD manifold and $\boldsymbol{A}$ is an affinity matrix built later in this section. This induces the definition of a global cost function

$$\boldsymbol{L}(W) = \sum_{i,j} \mathcal{J}_{i,j}(\boldsymbol{W}; \boldsymbol{X}_i, \boldsymbol{X}_j), \tag{8}$$

which we seek to minimize with respect to $\mathbf{W}$.

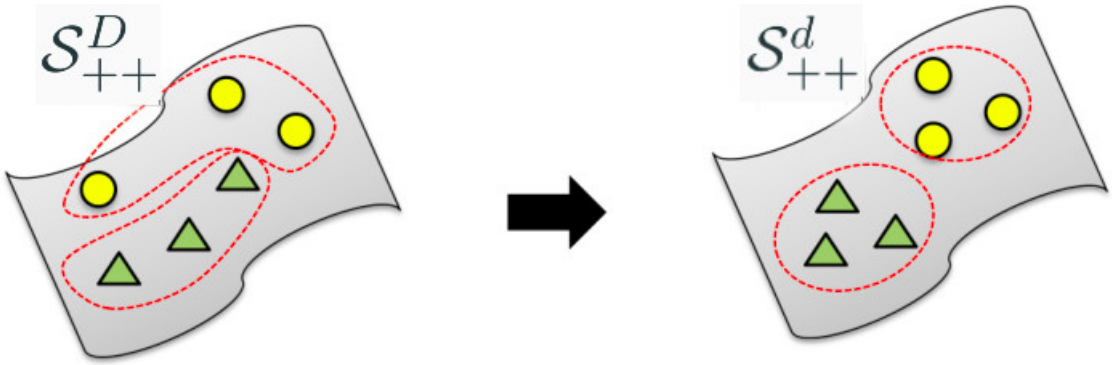


Figure 8: Mapping between a high-dimensional SPD manifold and a low-dimensional SPD manifold [17].

Let us insist on the fact that, for the remaining of this thesis:

- D refers to the dimension of the matrices belonging to the high-dimensional SPD manifold ($\mathcal{S}_{++}^{D}$), to which the covariance matrix representation of the EEG/sEMG signals belong.
- d refers to the dimension of the matrices belonging to the lower-dimensional SPD manifold ($\mathcal{S}_{++}^{d}$), to which the points belong after dimensionality reduction.

5.2. The affinity matrix

The important structure of the original data is encoded within a matrix $\boldsymbol{A} \in \mathbb{R}^{p \times p}$, called the affinity matrix. This matrix defines a notion of affinity between any pair of data points within the training set. We wish to build this pairwise affinity by similarity measure on the low-dimensional SPD manifold. In [17], the affinity matrix relies completely on the distance defined on the manifold.
As the goal is to bring together points from the same class and push away points from different classes, an intuitive way of understanding how we build this affinity matrix is to say that for each of the training points, we want to:

- minimize the distance with the v_w closest points of the same class;
- maximize the distance with the v_b closest points of a different class.

Mathematically, if we consider a training set $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_p\}$ where each point in this set belongs to a specific class, the affinity matrix is defined as :

$$\boldsymbol{A} = \boldsymbol{G}_w - \boldsymbol{G}_b, \tag{9}$$

where

$$\begin{cases} \boldsymbol{G}_w(i,j) & = 1, \text{ if } \boldsymbol{X}_i \in N_w(\boldsymbol{X}_j) \text{ or } \boldsymbol{X}_j \in N_w(\boldsymbol{X}_i), \\ \boldsymbol{G}_b(i,j) & = 1, \text{ if } \boldsymbol{X}_i \in N_b(\boldsymbol{X}_j) \text{ or } \boldsymbol{X}_j \in N_b(\boldsymbol{X}_i), \end{cases} \tag{10}$$

and with

- $N_w(\boldsymbol{X}_i)$ representing the v_w closest training points to $\boldsymbol{X}_i$ from the same class,
- $N_b(\boldsymbol{X}_i)$ representing the v_b closest training points to $\boldsymbol{X}_i$ from a different class.

5.3. Optimization on Grassmann manifold

We now see how to solve the minimization problem presented in eq. (11). In order to ensure that the resulting mapping belongs to $\mathcal{S}_{++}^{d}$, we need $\boldsymbol{W}$ to be of full rank. A

way to impose this condition is to add the condition $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_d$ to the minimization problem. Let us note that any full rank matrix $\bar{\boldsymbol{W}} \in \mathbb{R}^{D\times d}$ may be written as $\bar{\boldsymbol{W}} = \boldsymbol{W}\boldsymbol{M}$ for some $\boldsymbol{W} \in \mathbb{R}^{D\times d}$ such that $\boldsymbol{W}^T\boldsymbol{W} = I_d$ and some $\boldsymbol{M} \in GL(d)$. Therefore, if the metric defined on the SPD manifold has the property of affine-invariance (i.e the AIRM and the Stein divergence in this thesis), imposing this orthonormality constraint entails no loss of generality as $\mathcal{J}_{i,j}(\boldsymbol{W}\boldsymbol{M}; \boldsymbol{X}_i; \boldsymbol{X}_j) = \mathcal{J}_{i,j}(\boldsymbol{W}; \boldsymbol{X}_i; \boldsymbol{X}_j)$ whenever $\boldsymbol{M} \in GL(d)$. However, if the metric has not the affine-invariance property (i.e the Log-Euclidean metric or the Euclidean metric in this thesis), we do lose some generality as the search space is smaller. However, for sake of simplicity when solving the optimization problem, we impose this orthonormality constraint for all the considered metrics.

Therefore, the optimization problem we wish to solve becomes:

$$\underset{\boldsymbol{W}\in\mathbb{R}^{D\times d}}{\arg\min}\, L(\boldsymbol{W}) = \sum_{i,j} \mathcal{J}_{i,j}(\boldsymbol{W}; \boldsymbol{X}_i, \boldsymbol{X}_j) \quad s.t.\ \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_d. \tag{11}$$

This constrained optimization problem is equivalent to :

$$\underset{\boldsymbol{W}\in St(D,d)}{\arg\min}\, L(\boldsymbol{W}) = \sum_{i,j} \mathcal{J}_{i,j}(\boldsymbol{W}; \boldsymbol{X}_i, \boldsymbol{X}_j),$$

where $St(\cdot,\cdot)$ is the Stiefel manifold introduced in section 3.3. Moreover, as all metrics endowing the SPD manifold considered in this thesis are invariant under rotation (see table 1), we have the additional property that for any rotation matrix $\boldsymbol{R}$, $L(\boldsymbol{W}) = L(\boldsymbol{W}\boldsymbol{R})$. Therefore, we might reduce the search space even more and write the optimization problem as

$$\underset{\boldsymbol{W}\in Gr(D,d)}{\arg\min}\, L(\boldsymbol{W}) = \sum_{i,j} \mathcal{J}_{i,j}(\boldsymbol{W}; \boldsymbol{X}_i, \boldsymbol{X}_j). \tag{12}$$

5.3.1 Resolution of the optimization problem

In order to solve the optimization problem on the Grassmann manifold, we use the Riemannian Conjugate Gradient method. This latter is an iterative method based on the definition of gradient on the manifold. Here, we do not enter into the details of this method as it requires more advance differential geometry tools. Further information about how to solve an optimization problem on matrix manifolds may be found in [2].

The outcome of this optimization problem depends on the chosen starting point. Here, we do not make an initial guess but rather solve the optimization problem with a random starting point. The problem is, in this thesis, always solved 4 times to minimize the dependence of the result on the starting point. Then, we choose among the solutions the matrix $\boldsymbol{W}$ leading to the smallest cost.

6. Experiments on EEG data

This section is dedicated to the experimentation of the aforementioned algorithms in order to achieve classification of SSVEP signals. As explained in section 2, SSVEP signals are natural responses of the brain after visual stimulation at specific frequencies. These signals are mostly emitted in the occipital lobe and are therefore measured with EEG electrodes placed on the scalp over this area. Here, we consider subjects exposed to different visual stimulations. Classification is achieved in order to deduce in real-time from the measured EEG signals to which visual stimulation a certain subject is looking at.

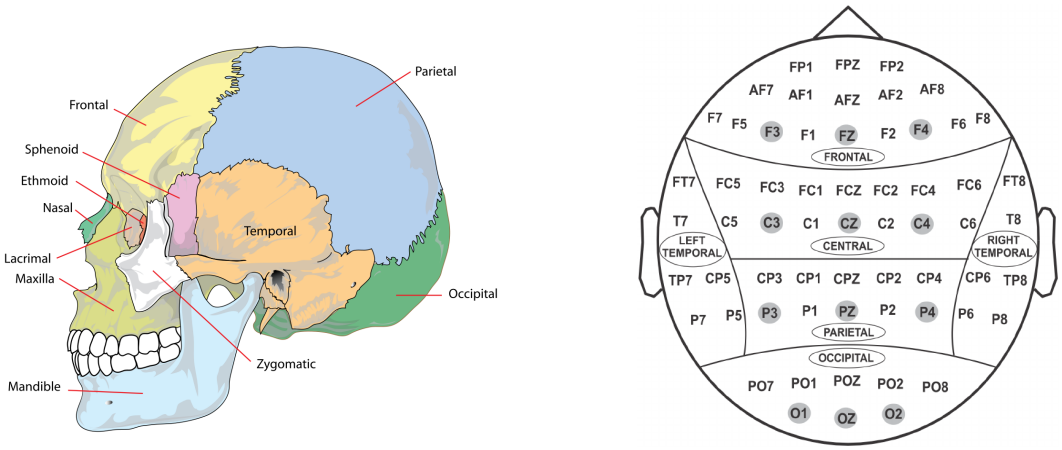
Hereunder, we first comment on the EEG dataset used. Then, we comment the covariance matrix representation of the EEG signals and discuss how we perform signal classification. This is followed by a presentation and a discussion of the classification results for the different metrics defined on the SPD manifold, and both classifiers (MDRM and KNN). Finally, we introduce the cross-subject learning problem.

6.1. Dataset and data measurement

The dataset used in this section is available in the following repository: `https://github.com/sylvchev/dataset-ssvep-exoskeleton` introduced in [22]. It contains EEG signals measured after visual stimulation. The stimulation consists of flashing stimuli (LEDs) at 4 different frequencies: $0\ Hz$ (*Rest*), $13\ Hz$, $17\ Hz$, and $21\ Hz$. These frequencies are the different classes of our classification problem. EEG signals measured on 12 subjects are available in the database. For the data acquisition, 8 EEG sensors were placed on the scalp of the subjects according to the 10-20 system on Oz, O1, O2, POz, PO3, PO4, PO7, and PO8 (see fig. 9b). These locations naturally correspond to different parts of the occipital area of the brain (see fig. 9a) where the frequency information of the flashing stimuli may mostly be found.

EEG sensors measured the signals at a sampling frequency of $256Hz$. As measured signals become relevant only over a certain time period, epochs of 3 seconds were recorded every 0.5 seconds. For most subjects, 16 epochs of 3 seconds were measured per visual stimulation. This applies to all subjects except for subject 10 (32 epochs of 3 seconds) and subject 12 (40 epochs of 3 seconds). Each of these epochs corresponds to one particular stimulus and is filtered between $12\ Hz$ and $45\ Hz$ to discard noise while allowing all stimulation frequencies and their first harmonics. It is then filtered around the 3 light stimuli frequencies ($13Hz$, $17Hz$ and $21Hz$) so that spatial covariance matrix estimator contains the initial frequency information.

Data of each epoch is placed in a matrix $\boldsymbol{Y} \in \mathbb{R}^{3n \times m}$, where n is the number of sensors (8 in our case) and m the sample length (3×256). The multiple 3 of n comes from the fact that the measured signals were filtered around the 3 initial frequencies. Therefore, we obtain for each epoch a matrix containing all the necessary measured signals in order to perform classification. Moreover, as an epoch contains measures resulting from one specific light stimulus, we have for all subjects (except for subject 10 and for subject 12) 16 matrices to represent each of the different classes.



(a) Regions of the brain (extracted from *wikimedia*)

(b) 10-20 EEG systems[23]

Figure 9: Visualization of the positioning of the EEG captors.

6.2. Covariance matrix representation of the data

Each matrix $\boldsymbol{Y} \in \mathbb{R}^{3n \times m}$ containing EEG data of one epoch and obtained as above is used to build a spatial covariance matrix estimator. A simple and intuitive way to achieve this is to compute the sample covariance matrix [5]:

$$\boldsymbol{\Sigma} = \frac{1}{m-1} \boldsymbol{Y}\boldsymbol{Y}^T.$$

The built estimator defines an SPD matrix of dimension $3n = 24$. Therefore, all the spatial covariance matrix estimators belong to the manifold $\mathcal{S}_{++}^{24}$ defined as

$$\mathcal{S}_{++}^{24} = \{\boldsymbol{\Sigma} \in \mathbb{R}^{24 \times 24} | \boldsymbol{\Sigma} \succ 0\}. \tag{13}$$

Each matrix $\boldsymbol{\Sigma} \in \mathcal{S}_{++}^{24}$ is associated to one of the following classes: Rest, $13\ Hz$, $17\ Hz$ or $21\ Hz$. Note that the spatial covariance matrix estimators are directly available in the database.

6.3. Classification of EEG signals

This section presents the classification procedure used to discriminate EEG signals. We first comment on how a dataset is separated into a training set and a test set. Then, we discuss the values of the parameters that play a role in the classification procedure. Finally, we explain how cross-validation is used to give values to some of these parameters.

6.3.1 Training and test sets

In order to generate reliable and meaningful results, we first separate the initial set of SPD matrices into a *training set* (for training the classification algorithms) and a *test set* (for testing the classification algorithms). Let us insist here that we choose training sets and test sets coming from one same subject. The case where these sets come from different subjects is another problem and is discussed later in this section. For now, we always consider *10 data points per class* as training set and the rest as test set. As all subjects in the database do not have the same amount of measured EEG data, the size of our test set may defer from one subject to another. We select *6 data points per class* for the training sets of all subjects excluding subject 10 and subject 12. These latter have respectively *22 data points per class* and *30 data points per class* in their training set. In order to show the most relevant results as possible, we randomly choose a training set of the previously discussed size at each execution of our classification algorithms. The performances of the classification algorithms are from $n_{iter} = 30$ different runs. We later present the performance not only as the median of all runs but also as boxplots that allow us to analyze the performance distribution. In fact, the line in the middle of the boxes represents the median of the performance distribution, the extremities of the boxes represent the first and third quartile, and the extremities of the whiskers represent the minimum and maximum values of this same distribution.

6.3.2 Parameters of the classification problem

Along the classification process of the EEG signals, many parameters play a crucial role :

- K, the number of neighbors considered during the KNN classification (see section 4.4);
- d, the dimension of the SPD matrices after projection on the lower-dimensional SPD manifold during the dimensionality reduction algorithm (see section 5);
- v_b and v_w, the parameters needed to build the affinity matrix for the dimensionality reduction process (see section 5.2).

Harandi et al. proposed in [17] to fix the value of the parameter v_w to the number of points in the least represented class. Using this approach and considering that the classes are balanced in this classification problem, we get $v_w =$

- 10 if we are not working with subject 10 or with subject 12
- 22 if we are working with subject 10
- 30 if we are working with subject 12

Then, the value of the parameter K is empirically fixed by the experimenter to $K = 5$. Concerning the two other parameters (i.e d and v_b), we use 5-fold cross-validation at each run to deduce their values.

6.3.3 Comments on cross-validation

Cross-validation is used to give a value to the parameters d and v_b. A classical way to choose their values is to find the pair $\{d, v_b\}$ which leads to the best classification performance. However, as we are here interested in real-time applications, we should also take the classification time into account. Moreover, we see later in this section that lowering the value of d does sometimes not impact the classification performance. When we use dimensionality reduction, classification is made on the manifold $\mathcal{S}_{++}^{d}$. Therefore, the parameter d has an important influence on the classification time, and we want it to be as small as possible without leading to a drop in classification performance. Therefore, we assign a score to each pair $\{d, v_b\}$ by considering both the resulting classification time and the classification performance (measured with the F1 micro score or the F1 macro score). This score is defined as

$$score = w_t \left(1 - \frac{t}{t_{ndr}}\right) + F1_{score}^2, \tag{14}$$

where:

- $\boldsymbol{w_t}$ is the weight given to the computational efficiency of the classification algorithm, in our case: $w_t = 0.1$;
- $\boldsymbol{t}$ is the time to classify one test point. This time is computed as the sum of the time needed to project this test point onto the lower-dimensional manifold and the time needed to classify this point on this same manifold;
- $\boldsymbol{t_{ndr}}$ is the time to classify a test point on the original higher-dimensional manifold;
- $\boldsymbol{F1_{score}}$ is the classification performance.

To choose between each pair $\{d, v_b\}$ during the cross-validation process, we then simply choose the pair resulting in the highest score. Here, we choose to give quadratic importance to the classification performance because we consider that an increase in performance is more important for higher performance. Naturally, the score presented in eq. (14) needs to be adapted according to the application considered. Many other ways could be used to define this score but we defined it as such because it seems reasonable for the EEG classification problem. In the remainder of this section, the classification results involving dimensionality reduction and cross-validation use the score presented in eq. (14) to assign values to d and v_b.

6.4. Classification Results

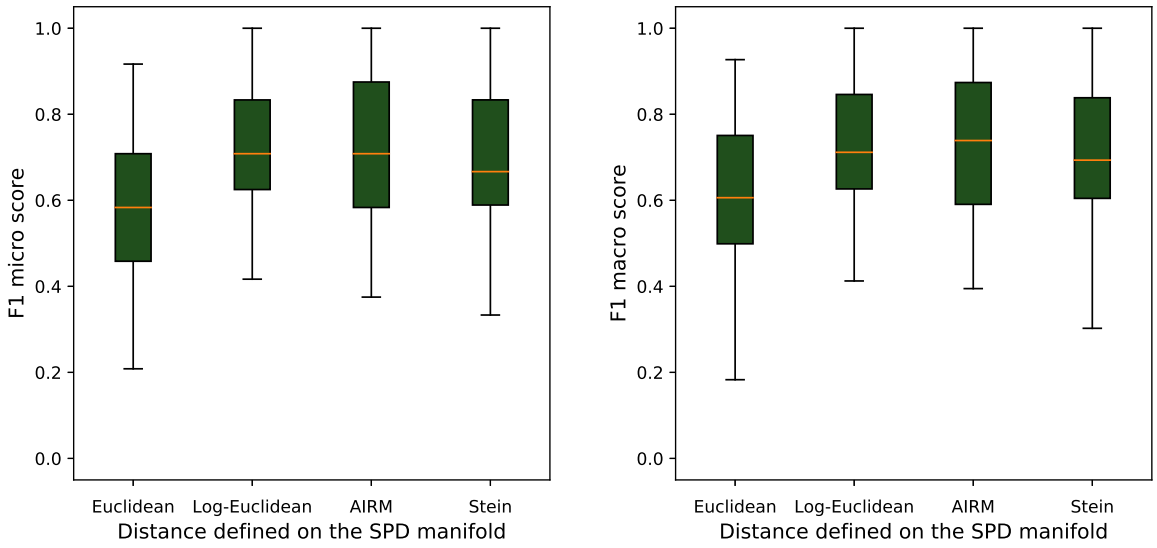
We now discuss the performance of the classification algorithms MDRM and KNN presented in section 4 by endowing the SPD manifold with one of the distances defined in section 3.2.2. Their performance is evaluated with the *F1 micro score* and the *F1 macro score*. The classification algorithms are tested on the manifold $\mathcal{S}_{++}^{24}$, but also on a lower-dimensional SPD manifold after using the dimensionality reduction algorithm presented in section 5. In the following, if the subject is not specified when showing results, it means that all the 12 subjects are taken into account. The classification is done on each of the subjects independently but the resulting performance is presented by combining the performance obtained for each subject. Therefore, as each classification algorithm is run $n_{iter} = 30$ times, the boxplots are built with a performance vector of size $n_{iter} \times n_{subjects} = 30 \times 12 = 360$.

6.4.1 Classification with MDRM

We first consider the MDRM classification algorithm.

Classification without dimensionality reduction The classification is performed on the manifold $\mathcal{S}_{++}^{24}$ and the performance obtained with the MDRM classifier may be observed in fig. 10. This figure shows that the Euclidean metric gives a poorer performance compared to the three other metrics. This is expected as the Euclidean metric does not take into account the Riemannian geometry of the SPD manifold. All the other metrics give a very similar performance, and we may not prefer one over the others by just looking at these performances. We also notice that both the F1 micro score and the F1 macro score exhibit comparable trends. This comes from the fact that all our classes are balanced, which leads these two performance measures to be very similar to each other.

Let us insist on the fact that even if fig. 10 takes all subjects into account, the classification performance is naturally not equivalent for all of these subjects. However, the



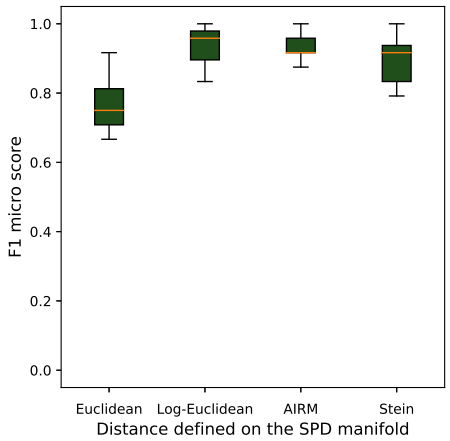
(a) F1 micro score in function of the distance defined on the SPD manifold.

(b) F1 macro score in function of the distance defined on the SPD manifold.

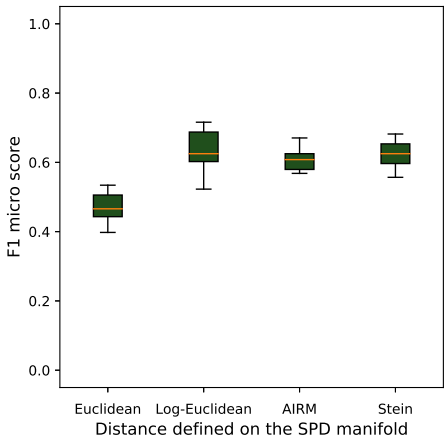
Figure 10: MDRM performance before dimensionality reduction.

conclusions that may be drawn from the relative performance of the classifier when the SPD manifold is endowed with one of the metrics is common to each of these subjects and translated in this figure. We should also keep in mind that the performance of a classifier highly depends on the quality of the data with which we are working. EEG data measured with non-invasive sensors is usually noisy and hardly treatable, which explains why the classification performance with only 4 classes is not close to 100%. Moreover, the measured EEG data of some subjects are particularly noisy implying a low average classification performance. Figure 11 shows an example of the difference in data quality that may exist. We indeed observe that we obtain a good classification performance for subject 3 for all the metrics, while the classification results are lower for subject 10. For these reasons, we never focus in this thesis on the exact classification performances but rather on the performance of one specific algorithm when compared to another algorithm.

Classification with dimensionality reduction Instead of working on the manifold $\mathcal{S}_{++}^{24}$, we now use the same classification algorithm (i.e MDRM) but on a lower-dimensional and more discriminative SPD manifold. To do so, we use the dimensionality reduction algorithm presented in section 5 before classification. This defines a mapping between the manifold $\mathcal{S}_{++}^{24}$ and a manifold $\mathcal{S}_{++}^{d}$. The parameter d plays an especially important role in the success of the classification algorithm which is discussed later on. As explained previously, this parameter d and the parameter v_b, needed to construct the affinity matrix (see section 5.2), is chosen through 5-fold cross-validation.



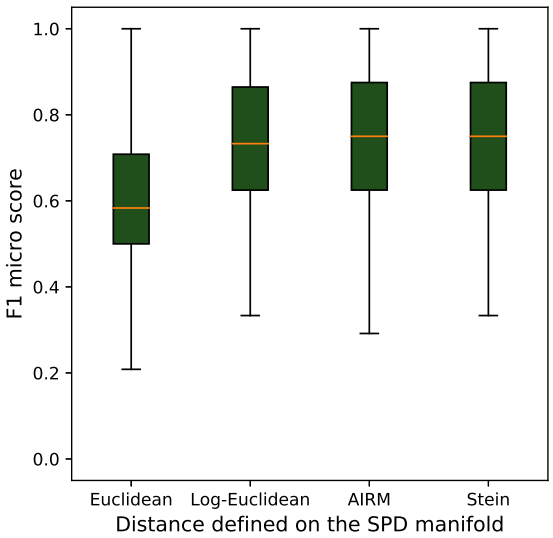
(a) F1 micro score of subject 3 in function of the distance defined on the SPD manifold.



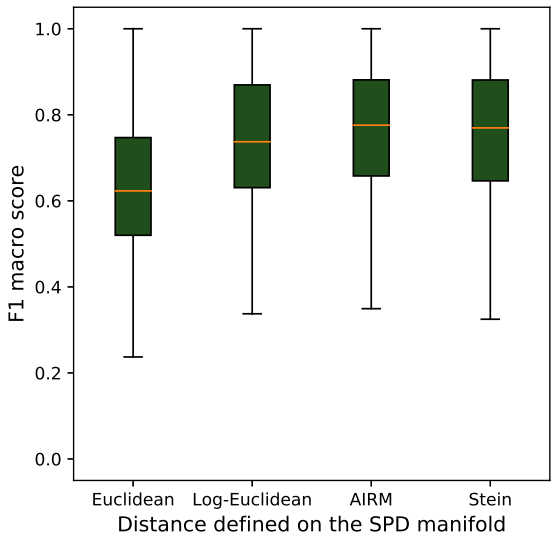
(b) F1 micro score of subject 10 in function of the distance defined on the SPD manifold.

Figure 11: MDRM performance before dimensionality reduction for subjects 3 and 10.

Figure 12 shows the performance of the MDRM classifier on this lower-dimensional manifold. We observe a slight improvement for both the F1 micro score and the F1 macro score compared to the classification performance obtained before dimensionality reduction and presented in fig. 10. This result is especially interesting considering the fact that classification is performed on a lower-dimensional SPD manifold.



(a) F1 micro score in function of the distance defined on the SPD manifold.

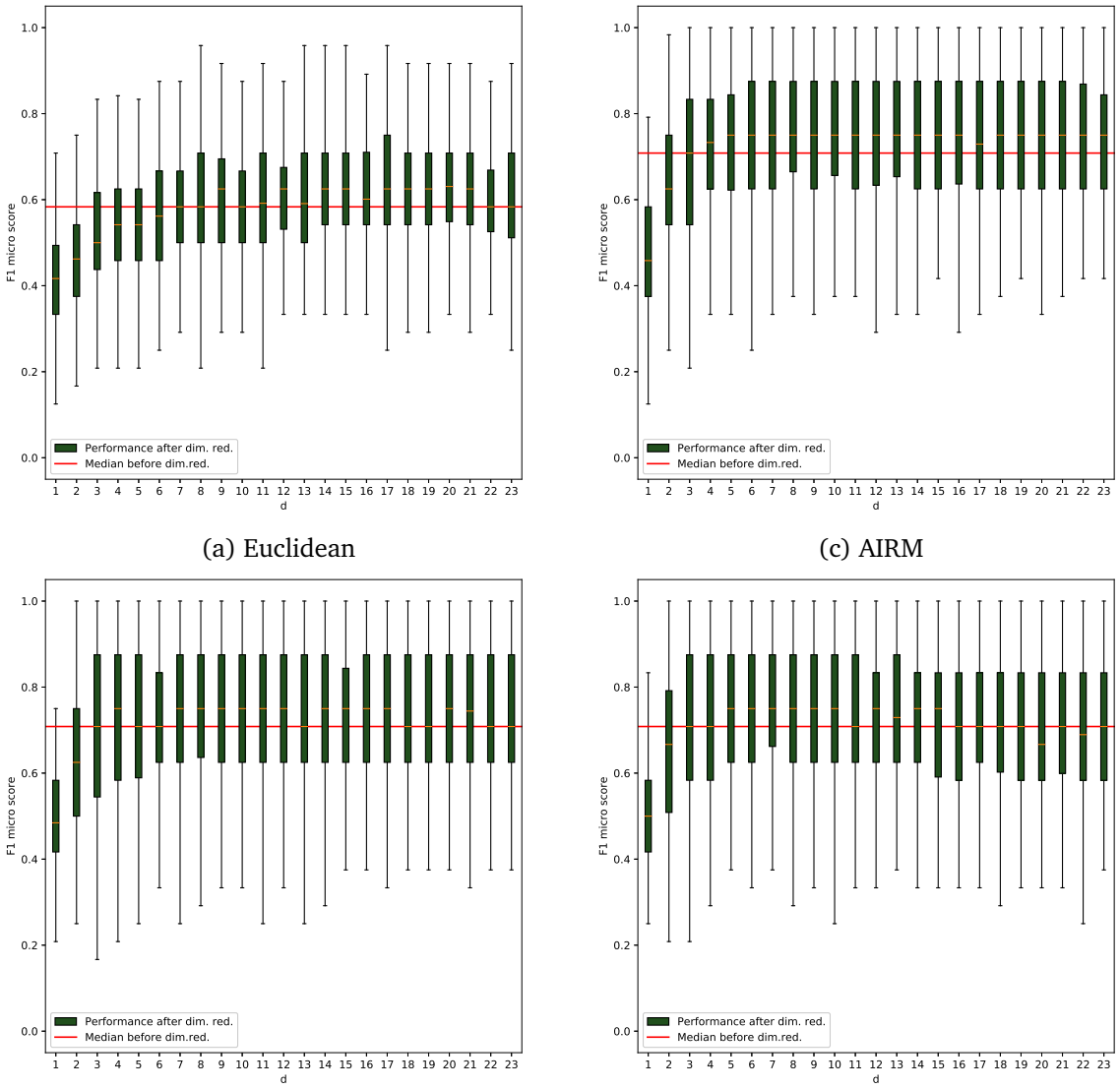


(b) F1 macro score in function of the distance defined on the SPD manifold.

Figure 12: MDRM performance after dimensionality reduction considering all subjects.

To better analyze the influence of the dimensionality reduction algorithm, we now give a specific value to the parameter d. As the estimated covariance matrices belong to $\mathcal{S}_{++}^{24}$, d is given a value within the set $[1, \ldots, 23]$. Figure 13 shows the F1 micro score obtained

for each of the different distances by fixing the parameter d to each of its possible values. We only represent the performance with the *F1 micro score* as the results with the *F1 macro score* are very similar. We first observe that the classification performance does not grow linearly with the dimension of the lower-dimensional manifold. Instead, we see that for each of the metrics, the F1 micro score grows quickly with the parameter d for the first few dimensions, and then stagnates. This may be explained by the fact that all the entries of the 24×24 initial covariance matrices are not essential for classification purposes. Dimensionality reduction discards the non-essential features and allows us to work in a smaller space without losing precision. Moreover, the fact that dimensionality reduction, in addition to working in a smaller space, enhances classification performance may be explained by the fact that the noise in the initial covariance matrices is partially filtered by the dimensionality reduction algorithm. We also sometimes notice a slight drop in performance for bigger dimensions. This comes from the fact that the greater the dimension, the greater the search space during the optimization on Grassmann manifold (see eq. (11)). The solver (i.e. Conjugate Gradient) converges then less easily to an optimal solution leading to a possible drop in performance.



(a) Euclidean

(c) AIRM

(b) Log-Euclidean

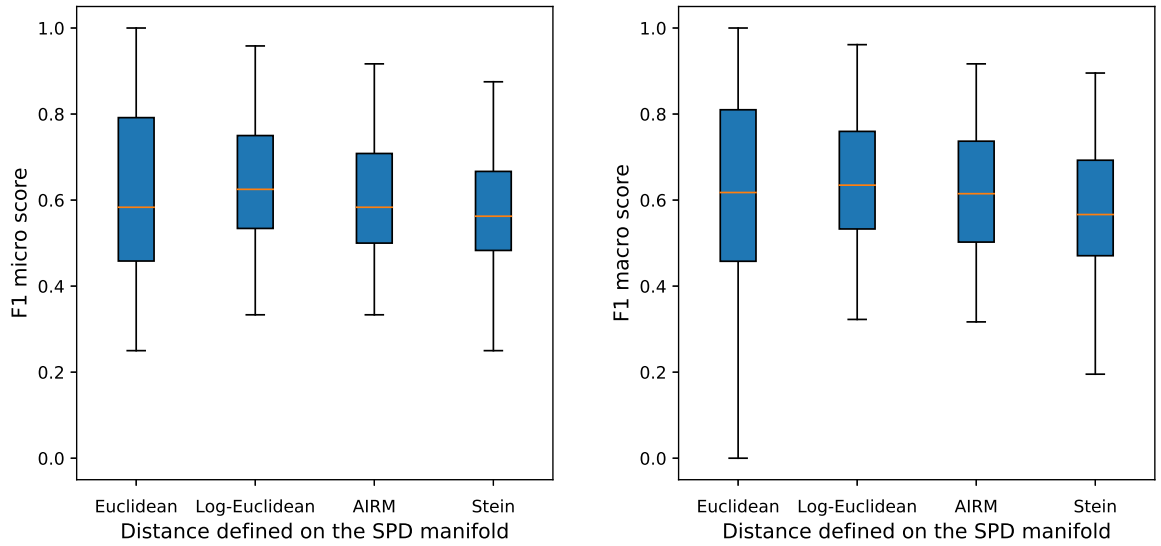
(d) Stein

Figure 13: Relation between MDRM performance (measured with the F1 micro score) and d for the different distances.

6.4.2 Classification with KNN

In a similar way to how we proceeded for the MDRM classifier, we now test the *KNN* classifier on EEG data. Remember that the parameter k was empirically set by the experimenter to $k = 5$.

Classification without dimensionality reduction Figure 14 shows the *F1 micro score* and the *F1 macro score* obtained after classification with the KNN classifier. All four metrics give comparable results, which are lower than the results obtained with the MDRM classifier in fig. 10.



(a) F1 micro score in function of the distance defined on the SPD manifold.

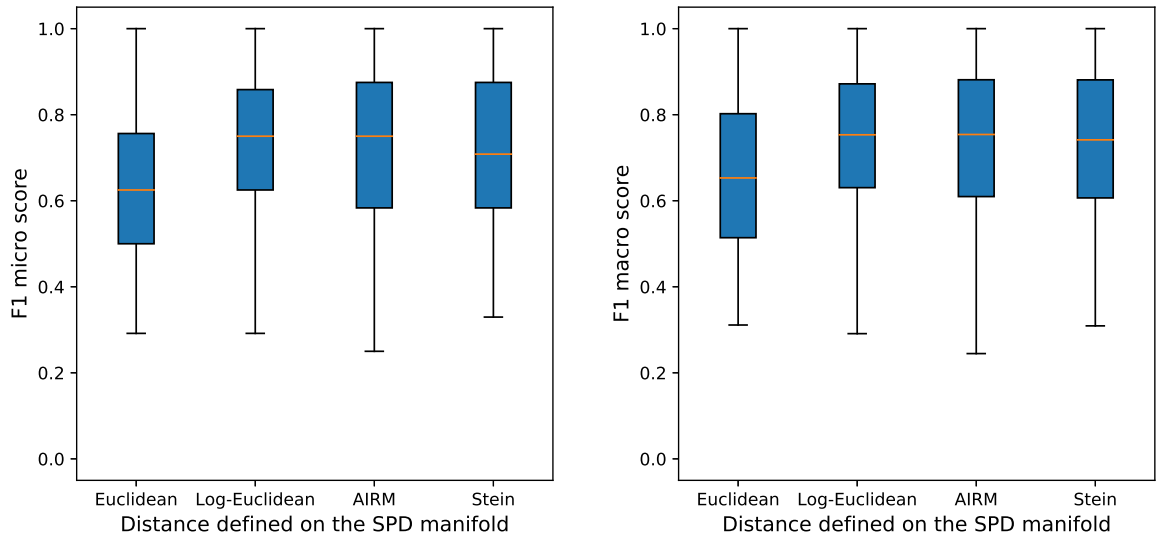
(b) F1 macro score in function of the distance defined on the SPD manifold.

Figure 14: KNN performance before dimensionality reduction considering all subjects.

Classification with dimensionality reduction When using the dimensionality reduction algorithm with cross-validation before classifying with KNN, we observe from fig. 15 an important performance rise of the classifier's performance. This confirms that the dimensionality reduction algorithm results in a better repartition of the SPD matrices allowing a better classification performance. In fig. 27, we find the performance of this classifier when fixing the parameter d. The dimensionality reduction algorithm and the dimension of the lower-dimensional manifold have a similar influence for KNN and MDRM.

6.4.3 Discussion and comparison of the classifiers

As shown previously, both classifiers show a rise of their performance for all the metrics on the SPD manifold when using the dimensionality reduction algorithm before the



(a) F1 micro score in function of the distance defined on the SPD manifold.

(b) F1 macro score in function of the distance defined on the SPD manifold.

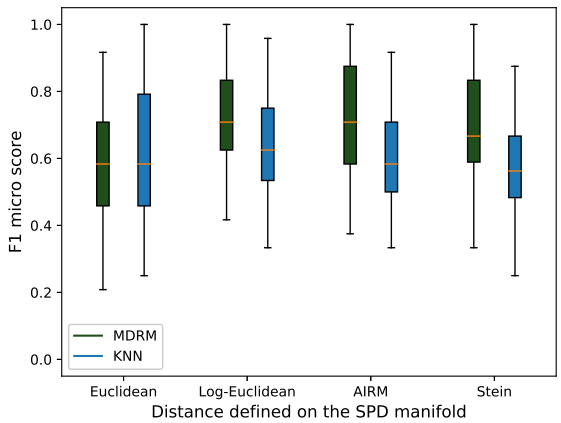
Figure 15: KNN performance after dimensionality reduction considering all subjects.

classification process. Interestingly, the difference in classification performance between MDRM and KNN is significantly reduced after dimensionality reduction. This suggests that essential features for the classification may be extracted by projecting the data points onto a lower-dimensional SPD manifold but that some data points stay hard to classify for both classifiers.

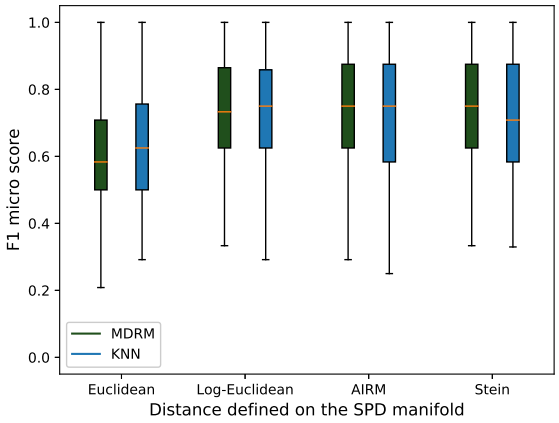
Table 2 shows a summary of the median of the classification performances for both the MDRM and the KNN classifier in function of the metric defined on the SPD manifold. This table shows the improvement offered by the dimensionality reduction algorithm. It is observed for all the metrics and is especially important for the KNN classifier. The resulting performance after dimensionality reduction is similar for both classifiers.

Distance / Classifier	**Euclidean**	**Log-Eucl.**	**AIRM**	**Stein**
MDRM (no dim. red.)	58.33	70.83	70.83	66.66
MDRM (with dim. red.)	58.33	73.33	75.00	75.00
Improvement with dim. red.	0.00	2.50	4.17	8.34
KNN (no dim. red.)	58.33	62.50	58.30	56.25
KNN (with dim. red.)	62.50	75.00	75.00	70.85
Improvement with dim. red.	4.17	12.5	16.7	14.6

Table 2: Median F1 micro score (%) depending on the distance defined on the SPD manifold and the classifier used.



(a) Comparison before dimensionality reduction



(b) Comparison after dimensionality reduction

Figure 16: Comparison of the performance of MDRM and KNN before and after dimensionality reduction

The improvement of the dimensionality reduction algorithm is indisputable. One question remains, how does it really affect the data points? We showed before that the manifold of SPD matrices of dimension more than 2 may not be drawn in a 3D-Euclidean space. Therefore, we propose to use *multidimensional scaling* (MDS) [8] to visualize the data points belonging to the SPD manifold. We use the Python implementation proposed by *scikit-learn* [34].

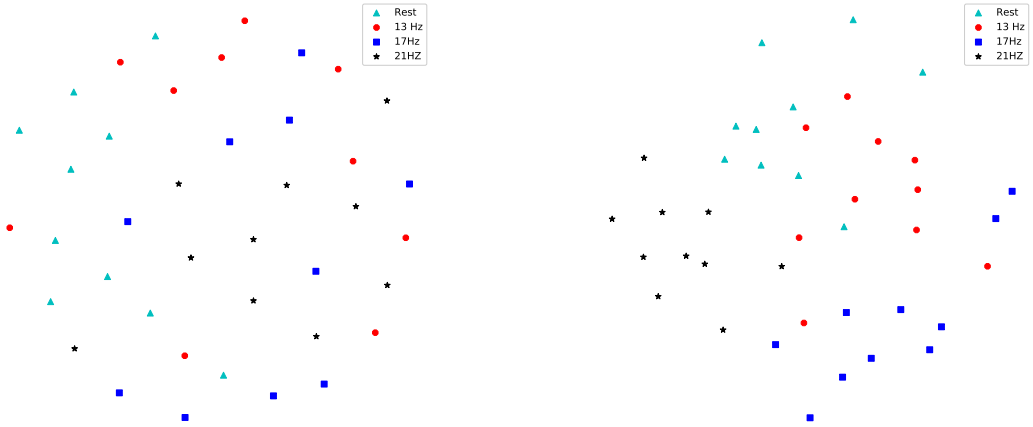
multidimensional scaling This method provides a visual representation of the pairwise distances among a set of points. Here, we use it in order to represent the points lying on the SPD manifold in a 2D-Euclidean space. Those spaces are not homeomorphic and, therefore, the representation is not exact. However, it gives an indication of the similarity between the points. Multidimensional scaling uses the matrix representing all the pairwise distances between the points to solve an optimization problem that minimizes the sum of the squared pairwise distances errors in the 2D-Euclidean space. Mathematically, if we have n SPD matrices $\boldsymbol{x}_i \in \mathcal{S}_{++}^D$ for $i \in \{1, \dots, n\}$, we search for some points $\boldsymbol{y}_i \in \mathbb{R}^2$ for $i \in \{1, \dots, n\}$ such that

$$\{\boldsymbol{y}\}_{i=1}^n = \underset{\{\boldsymbol{y}\}_{i=1}^n}{\arg\min} \sum_{i=1}^n \sum_{j=i}^n (\delta_{\mathcal{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \delta(\boldsymbol{y}_i, \boldsymbol{y}_j))^2. \tag{15}$$

In this equation, $\delta_{\mathcal{M}}(\cdot, \cdot)$ is the distance defined on the SPD manifold and $\delta(\cdot, \cdot)$ is the usual Euclidean metric.

Figure 17 shows a visual representation of a training set of subject 4 before and after dimensionality reduction using the Stein metric. Even though it does not translate the

exact reality, we observe a better separation of the different classes after dimensionality reduction.



(a) Before dimensionality reduction.

(b) After dimensionality reduction.

Figure 17: Visualization with MDS of a training set of subject 4 using the Stein metric.

6.4.4 The influence of dimensionality reduction on the computational time of the classifiers

Beyond the fact that dimensionality reduction improves the performance of the classification of EEG signals, an important motivation for using this algorithm is the gain in time complexity. In fact, achieving classification on lower-dimensional SPD manifolds requires less mathematical operations, leading to a smaller time complexity. As for many classification problems, a trade-off between a low time cost and a high classification performance should be achieved when classifying EEG data. Both usually do not come together and, therefore, we need to weight their importance as previously explained in section 6.3.3.

Let's note that when we wish to classify a point $\boldsymbol{Y} \in \mathcal{S}_{++}^{D}$ with dimensionality reduction, the classification time is computed as the sum of

- The time needed to project $\boldsymbol{Y}$ on the lower-dimensional manifold $\mathcal{S}_{++}^{d}$ (i.e computational time of defining $\boldsymbol{Y_2} = \boldsymbol{W^T Y W}$, where $\boldsymbol{W}$ is the solution found after optimization on Grassmann manifold);
- The time needed to classify $\boldsymbol{Y_2}$ on the lower-dimensional manifold $\mathcal{S}_{++}^{d}$ (with MDRM or KNN).

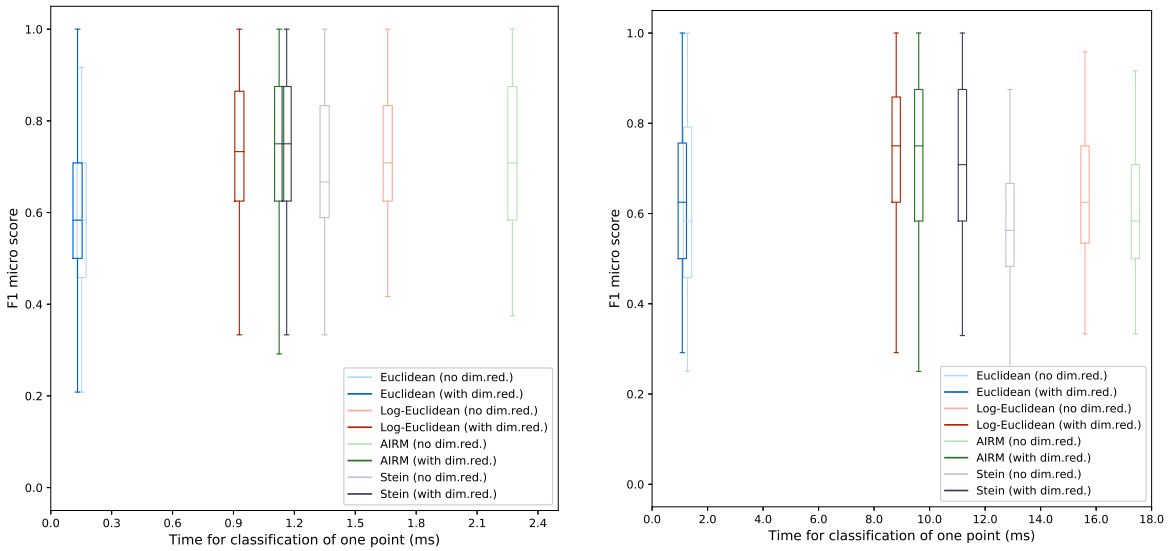
However, when we wish to classify this same point on the original SPD manifold, the computational time simply consists of the time needed to classify $\boldsymbol{Y}$ on the higher dimensional manifold $\mathcal{S}_{++}^{D}$ (with MDRM or KNN).

Figure 18 shows, for both MDRM and KNN, the impact of the metric and the dimensionality reduction algorithm on both the classification performance measured with the F1 micro score (similar for F1 macro score) and the time needed to classify a data point. These results were obtained with a non-optimized Python code on a computer with an i7-5930K cpu (6 cores) and 31 GB of RAM. We observe that for each of the metrics, dimensionality reduction improves both the computational time and the performance of the classifiers. Another interesting point is that the time to classify a point is greater for the KNN classifier. This is expected as KNN requires much more mathematical operations compared to MDRM. As both classifiers give very similar performances after dimensionality reduction, MDRM therefore appears more suitable for the EEG classification problem considered here.

Concerning the different metrics defined on the SPD manifold, we observe that the Euclidean metric leads to a small classification time compared to the other metrics. However, it results in a poorer classification performance than the other metrics, as noted previously. This metric may be useful in some applications with sharp time constraints when using another metric on the SPD manifold would not fit the time requirements. The AIRM, Log-Euclidean metric, and Stein divergence generate similar classification performances after dimensionality reduction, where the lowest classification time is obtained by using the Log-Euclidean metric. We also observe that the Stein metric gives to smallest computational time before dimensionality reduction. However, the trend is reversed after dimensionality reduction. This is explained by the fact that reducing the dimension of the SPD manifold results in a lower gain in computational time with the Stein metric than with the AIRM and the Log-Euclidean metric. Finally, we emphasize that the choice of metric highly depends on the time constraints imposed by the intended application and on the score we defined in eq. (14).

6.5. Cross-subject classification

Until this point, the data points of each subject was separated into both a training set and a test set. The classification algorithms were then always trained and tested on data from the subject. In fact, a binding step is the fact that BCI usually requires some calibration stage which might be inconvenient for the patients needing to undergo multiple calibration sessions. Therefore, there is a great interest in building classification algorithms trained on some subject's data that could efficiently classify data from other



(a) MDRM classifier

(b) KNN classifier

Figure 18: Performance-Computational efficiency relation for both classifiers by considering all subjects

subjects. This question has recently attracted a lot of interest within the BCI research field. This is a typical kind of *transfer learning problem*. Within the machine learning field, transfer learning refers to the ability to use previous knowledge gained on some problem to solve a different but related problem. In this thesis, we focus on the *cross-subject learning* problem, which considers data from one subject as a training set and data from another subject a test set.

Some examples of transfer learning applied to BCI can be found in [13], [20], [26], [29], and [49]. In this work, we propose an approach for cross-subject learning for a BCI related problem by using the dimensionality reduction algorithm of section 5. We use a simple cross-subject learning approach which consists in using the data from one subject as a training set of the classification algorithm and testing the obtained algorithm on data from other subjects.

The difficulty of cross-subject learning Here, we give an explanation of the reasons that prevent us from having a similar classification performance with cross-subject learning compared to the case where the training and test data come from the same subject. fig. 19 gives a visualization of the repartition of the data points representing each of the 4 classes by considering the data from all the subjects[2]. We immediately see the difference compared to fig. 17 where a single subject was considered. Figure 19 shows that covariance matrices representing one same class, but coming from different subjects are not necessarily close to each other on the SPD manifold. This behavior shows

[2]Note that not all points are represented on the plot for a better visualization

that when one subject is visually stimulated by a flashing LED at a certain frequency, the outcoming signals measured from the occipital lobe do not necessarily have the same correlation as if the exact same visual stimulation was presented to some other subject. Indeed, this would not be a problem if the points from different classes were separated enough. However, it does not seem to be the case as we may see from fig. 19.

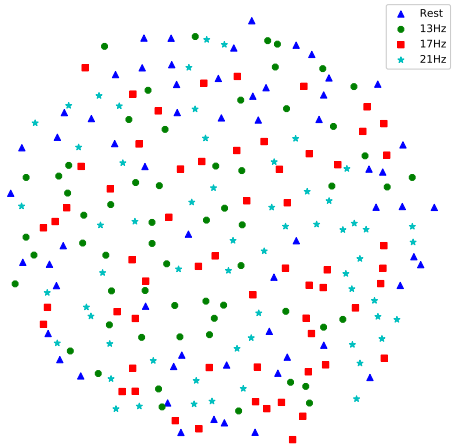
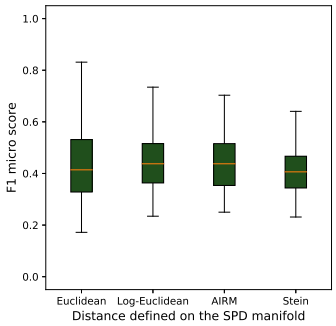


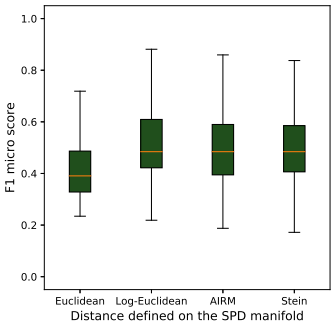
Figure 19: Visualization of the data points on the SPD manifold by considering all subjects. The visualization is obtained with MDS using the Stein metric.

Cross-subject classification results Even though fig. 19 show deeply intertwined data points announcing bad classification performance, we should not forget that we use a visualization tool which does not perfectly translate the repartition of the points on the SPD manifold. Therefore, we evaluate the cross-subject classification performance. To do so, we alternatively consider the data points from each of the 12 subjects as a training set and the data points from the other 11 subjects as a test set. We use fixed values for the parameters that intervene in the classification, namely $K = 5$, $v_w = 16$, $v_b = 9$, $d = 6$. Figure 20 and table 3 show the classification performance of our classifiers evaluated with the F1 micro score. First, we observe that the classification results are always greater than the chance rate (25% in our case), which shows that the covariance matrices from one subject and belonging to a certain class are not completely unrelated to the covariance matrices from another subject and belonging to this same class. However, they are naturally not as good as the performances obtained when considering a training and a test set coming from a same subject (see table 2). We also observe that the dimensionality reduction algorithm which projects the data points onto the manifold of SPD matrices of dimension 6 improves considerably the performance for the Log-Euclidean metric, the AIRM, and the Stein divergence. Note that, as we

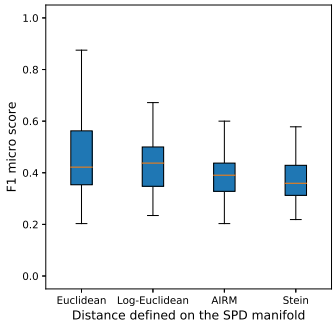
projected the data onto a much smaller and more discriminative space, the classification time is also reduced.



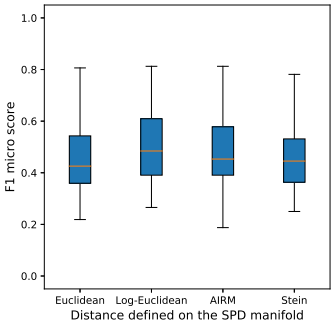
(a) MDRM without dimensionality reduction.



(c) MDRM with dimensionality reduction.



(b) KNN without dimensionality reduction.



(d) KNN with dimensionality reduction.

Figure 20: Cross-subject learning performance in function of the metric.

Discussion about cross-subject learning Multiple methods have been proposed to solve this cross-subject learning problem on the SPD manifold. Here, we only explored a simple naive approach, and concluded that the dimensionality reduction approach leads to significant improvements for the MDRM and the KNN classifier. Therefore, we believe that combining this approach with state of the art EEG cross-subject learning algorithms would result in a gain of computational time and classification performance.

Distance / Classifier	**Euclidean**	**Log-Eucl.**	**AIRM**	**Stein**
MDRM (no dim. red.)	41.41	43.75	43.75	40.63
MDRM (with dim. red.)	39.06	48.43	48.43	48.44
Improvement with dim. red.	−2.35	+4.68	+4.68	+7.81
KNN (no dim. red.)	42.18	43.75	39.06	35.94
KNN (with dim. red.)	42.58	48.43	45.31	44.53
Improvement with dim. red.	+0.40	+4.68	+6.25	+8.59

Table 3: Median F1 micro score (%) depending on the distance defined on the SPD manifold and the classifier used. The training set was alternatively set to the data points of each of the 12 subjects and the test sets were set to the data points of the remaining 11 subjects.

7. Experiments on sEMG data

This section is dedicated to the experimentation of the abovementioned algorithms for classifying surface electromyography (sEMG) signals. The sEMG signals used in this work measured the muscular activity of forearm muscles during different hand movements. The classification of these signals is essential in the conception of myoelectric prostheses. For the classification, we use the spatial covariance matrix signal representation of the sEMG signals and work on the SPD manifold. The use of covariance matrices has been motivated earlier in section 4. For this application, we can add to another intuitive argument which comes from the fact that each hand movement stimulates a group of arm muscles in a very specific way. As the sensors measuring sEMG signals are placed on different muscles, it is not the intensity of the signals that matters, but rather the correlation between them.

Hereunder, we first comment on the sEMG dataset used. Then, we comment the covariance matrix representation of the sEMG signals and discuss how we perform signal classification. Finally, we present and discuss the classification results for the different metrics defined on the SPD manifold, and both classifiers (MDRM and KNN).

7.1. Dataset and data measurement

The Ninapro publicly available database [3], proposes sEMG data from different electromyography acquisition setups. Detailed information about the data acquisition, the dataset validation, and the comparison of different sEMG acquisition setups may be found in [36]. Here, we use their 5th Ninapro database, where two *Thalmic Myo* armbands, produced by the company *North*[4], were used to collect the data on 10 different non-amputated subjects. These two armbands were placed as shown in fig. 21 and both consist of 8 medical grade stainless steel sEMG single differential electrodes evenly placed around the armband. The upper armband is placed close to the elbow with the first electrode on the radio humeral joint while the lower one is placed closer to the hand, tilted by $22.5°$ to fill the gaps left by the electrodes of the other Myo. Both of the armbands measure sEMG signals at a sampling frequency of $200Hz$ with a resolution of 8-bit signed. A cyber-glove is used to define a ground truth when measuring sEMG data.

This double Myo setup is used to collect data from 50 different hand movements. We only consider 41 movements in this thesis, corresponding to the movements of *exercise B* and *exercise C* of fig. 28 to which we added the movement *Rest*. Therefore, we

[3] Available at `http://ninaweb.hevs.ch/`
[4] `https://support.getmyo.com/hc/en-us`

consider the same 41 hand movements as in [36] when the different sEMG acquisition setups were compared. For the acquisition protocol, each movement was repeated 6 times. Between each of these repetitions, the subject goes back to the rest position.

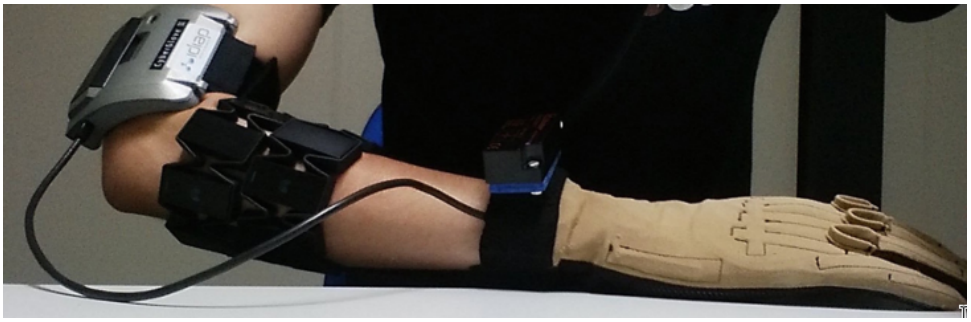


Figure 21: Double myo setup[36]

7.2. Covariance matrix representation of the data

The ninapro database proposes raw measured signals. However, as the classification algorithms used in this work rely on a covariance matrix representation of the data, we first need to compute these matrices.

7.2.1 Signal normalization

As we perform classification from estimated spatial covariance matrices, a signal that has a standard deviation way greater than the other signals leads to improper covariance matrices and thus bad classification results. Therefore, we first normalize the sEMG signals. Moreover, what we are here especially interested in when performing classification from covariance matrices is the correlation between the signals measured by different sensors. Absolute values do not matter and hence, we do not lose any essential data for classification when normalizing. Although normalization of surface electromyography is a research subject in its own and many normalizing methods have been proposed [40], we use the classic way of normalizing time signals in this thesis.
Let $\boldsymbol{x}$ be a time series measured by one sEMG sensor and containing $\boldsymbol{N}$ measures. We may compute an unbiased estimator for its mean

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

and another unbiased estimator for its standard deviation

$$\sigma_x = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})}{N-1}}.$$

This allows us to define a normalized signal as

$$\boldsymbol{x}_{norm} = \frac{\boldsymbol{x} - \bar{x}}{\sigma_x}. \tag{16}$$

As the test signals are in principle unknown during the training part, the mean and standard deviation estimators are computed from the training signals and the test signals are then normalized with these estimators, unrelated to the test signals, to simulate real-time conditions.

7.2.2 Windowing and construction of the covariance matrices

As proposed in [36], we choose a fixed size time window $T = 200\ ms$. Consecutive time windows are chosen to have a $100\ ms$ overlap. Depending on the fact if we consider one single armband (8 sensors) or both of the armbands (16 sensors), we build estimators for the spatial covariance matrices belonging to the manifold $\mathcal{S}_{++}^{8} = \{\boldsymbol{\Sigma} \in \mathbb{R}^{8\times 8} | \boldsymbol{\Sigma} \succ 0\}$ or to the manifold $\mathcal{S}_{++}^{16} = \{\boldsymbol{\Sigma} \in \mathbb{R}^{16\times 16} | \boldsymbol{\Sigma} \succ 0\}$. In either way, we simply gather all the measured data during the time window into a matrix $\boldsymbol{X} \in \mathbb{R}^{8\times m}$ (resp. $\boldsymbol{X} \in \mathbb{R}^{16\times m}$), where m is the number of measurements made by one sensors during the time window, i.e. $m = 0.2 \times 200 = 40$ (the time window times the sampling frequency). The spatial covariance matrix estimator is then defined as

$$\boldsymbol{\Sigma} = \frac{1}{m-1} \boldsymbol{X}\boldsymbol{X}^T. \tag{17}$$

7.3. Classification of sEMG signals

This section presents the classification procedure used to discriminate sEMG signals. We first comment on how the dataset is separated into a training set and a test set. Then, we discuss the values of the parameters that play a role in the classification procedure.

7.3.1 Training and test sets

As mentioned above, each movement was repeated 6 times by every subject. We assign the repetitions 1,3,4 and 6 to the training set and 6 and the repetitions 2 and 5 to the test set. Considering the huge amount of data proposed in the Ninapro database, we fix this training set and consider that the training/test sets are big enough such that the resulting performance of the classification algorithms are representative.

7.3.2 Parameters of the classification problem

Many parameters intervene in the classification process. The huge amount of data makes it difficult to perform cross-validation on the training sets due to limited computational power. Therefore, the following parameter values were therefore assigned empirically

by the experimenter: $K = 5$ (for KNN); v_w is set to the number of data points belonging to the least represented class for a certain training set; $v_b = \frac{v_w}{4}$. The parameter d which defines the dimension of the SPD matrices after dimensionality reduction is not given a precise value nor defined by cross-validation. This section studies its influence on the classification performance.

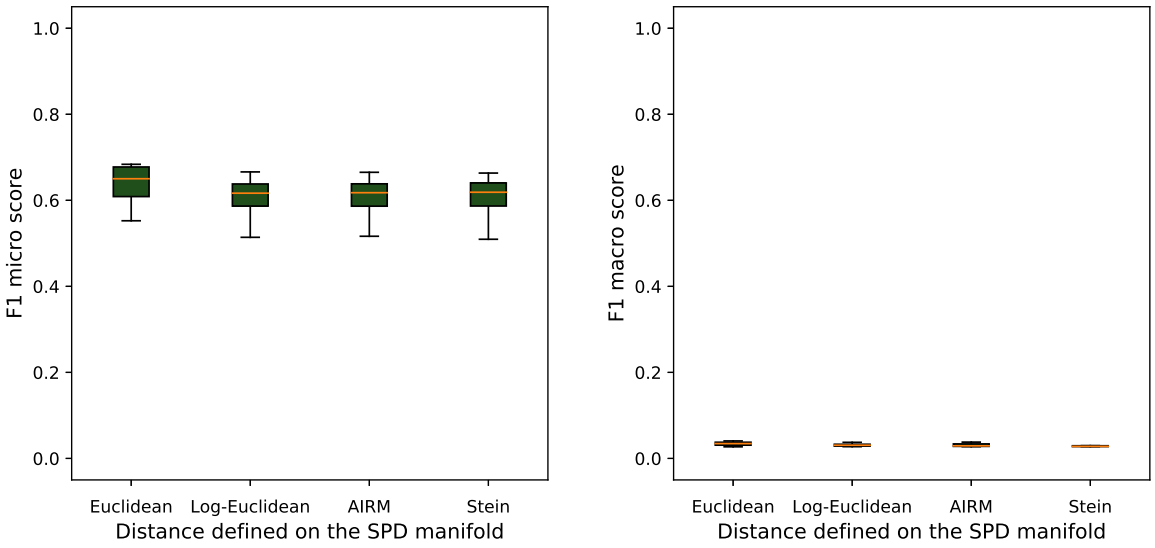
7.4. Classification results

As the acquisition setup used to measure the sEMG signals contains two Myo's, this compares the performance of the classifiers by considering the data measured by the electrodes of one Myo or of both Myos.

7.4.1 Classification with MDRM

Classification without dimensionality reduction Figure 22 shows the performance of the MDRM classifier by considering only the signals measured by the 8 sensors of the upper Myo (see fig. 21). These results may seem inconsistent as the F1 micro score is relatively high while the F1 macro score is low. This is actually a great example to see how the F1 micro score might be misleading when dealing with unbalanced classification problems. In fact, as each subject goes back into the rest position after each hand movement, the rest class is overrepresented. Even though it slightly differs from one subject to another, each subject has around 65% of their training points belonging to the class *rest*. As the F1 micro score computes a harmonic mean of the total precision and the total recall without considering the classes, a perfect precision and recall for the points of the class rest and a zero precision and recall for the other 40 classes leads to a F1 micro score of 65% and a F1 macro score of 2.43%. Therefore, major differences may appear between the F1 micro score and the F1 macro score for this unbalanced sEMG classification problem. Such differences are seen in fig. 22, showing that the classification is performant for the points of the class rest, but not for the other classes.

As we want here to classify hand movements controlling arm prosthesis, we want a reasonable performance for the classification of each of the hand movements and not only for some specific movements. Therefore, we mainly focus on the performance of the classifiers as measured with the F1 macro score when dealing with this sEMG classification problem. Considering the poor F1 macro score resulting from classification with MDRM, this classifier is not suitable. The conclusions are similar for the lower Myo or even for both Myo's together. Dimensionality reduction with MDRM does not improve results either. Therefore, we focus on KNN for the sEMG classification problem.



(a) F1 micro score in function of the distance defined on the SPD manifold.

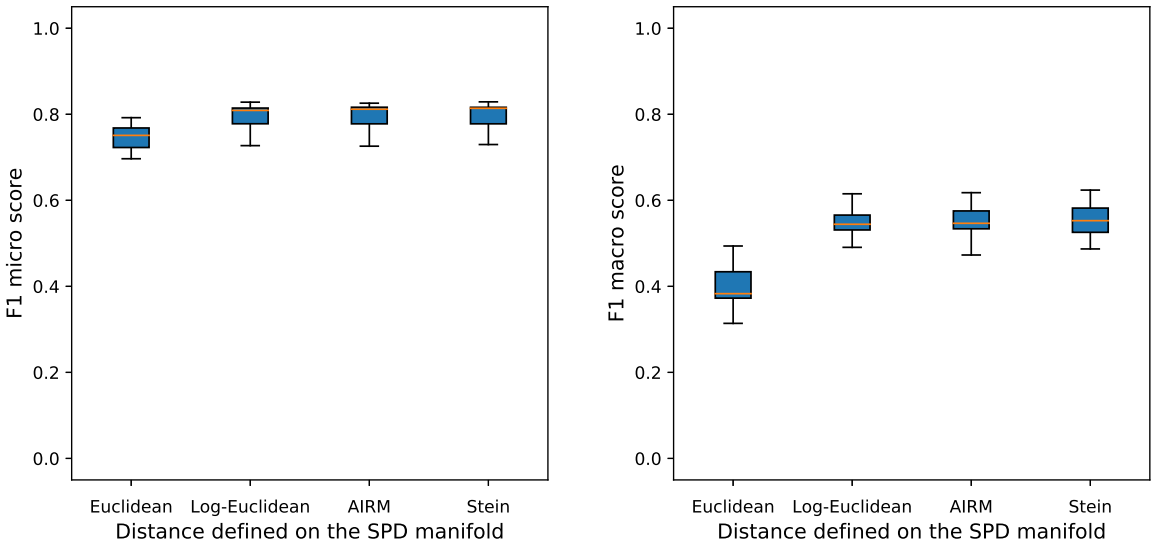
(b) F1 macro score in function of the distance defined on the SPD manifold.

Figure 22: MDRM performance for the Upper Myo before dimensionality reduction considering all subjects.

7.4.2 Classification with KNN

Classification without dimensionality reduction Figure 23 shows the performance of the KNN classifier on sEMG data when only considering the upper Myo. We observe again that the F1 micro score is higher than the F1 macro score, which shows that the performance of the classifier is not perfectly balanced between all the classes. However, in contrary to the MDRM classifier, the F1 macro score is rather satisfying as it reaches a score of around 60% for the metrics Log-Euclidean, AIRM, and Stein. Similarly to what we concluded with the EEG signals, the Euclidean metric has a poorer performance than the three other metrics. Similar results are observed in fig. 29 when considering both Myo. We also observe that, when both of the Myo's are considered, both the F1 micro score and the F1 macro score are slightly improved. This shows that measuring sEMG signals with two different Myo's do not give redundant signals. On the contrary, the signals measured by one Myo completes the signals measured by the another one, leading to better classification accuracy.

Classification with dimensionality reduction We now evaluate the performance of KNN augmented with the dimensionality reduction algorithm presented in section 5. Figure 24 shows the effect of the dimensionality reduction algorithm on the performance of the KNN classifier measured with the F1 macro score. Results are obtained by fixing the value of d and by using the KNN classifier on the manifold $\mathcal{S}_{++}^{d}$ after mapping the initial data points on this lower-dimensional manifold. The performance of the classifier increases almost linearly with the parameter d for all metrics on the SPD



(a) F1 micro score in function of the distance defined on the SPD manifold.

(b) F1 macro score in function of the distance defined on the SPD manifold.

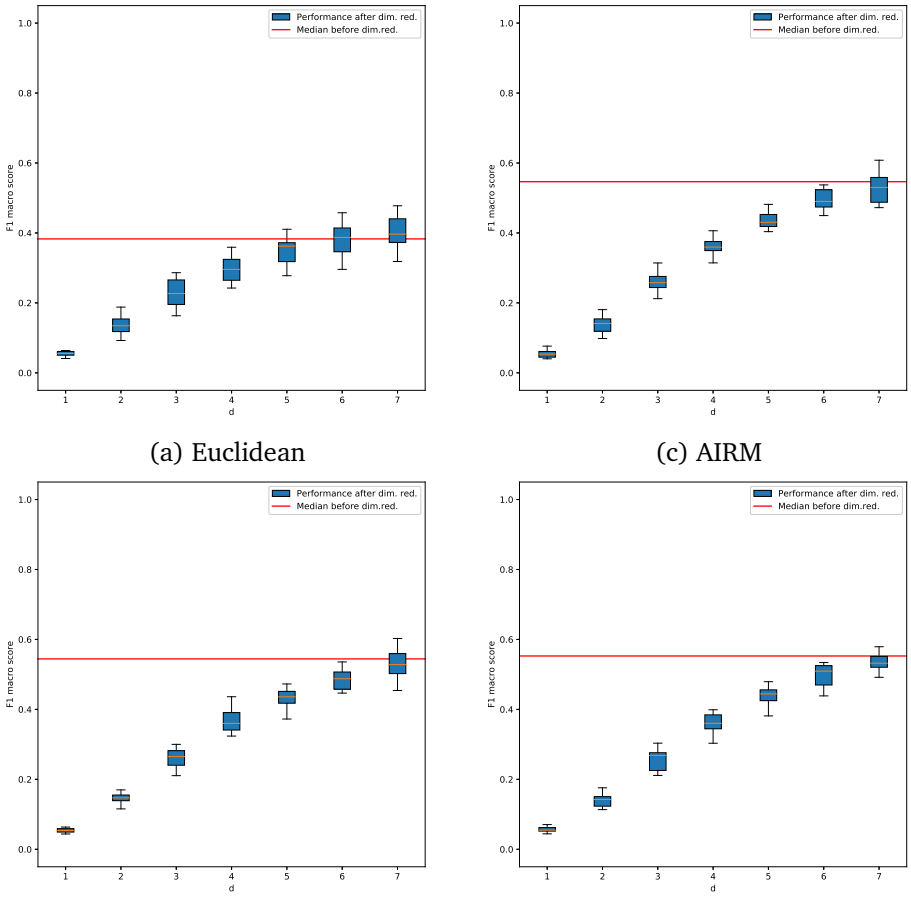
Figure 23: KNN performance for the Upper Myo before dimensionality reduction considering all subjects.

manifold. We have thus a very different behavior compared to the EEG signals where the performance was firstly rapidly increasing with d and then stable. Results are similar when considering the two Myos, as shown in fig. 30. As the initial estimated covariance matrices have dimension 16×16, we consider $d \in [1, \dots, 15]$. We see again that the classifier's performance grows linearly with the value of d at the beginning but tends to stabilize after. However, it does not improve the classification performance compared to the classification before dimensionality reduction.

7.4.3 Discussion and comparison of the classifiers

We observed that the KNN classifier leads to better results than the MDRM classifier when considering the F1 macro score, which is the score which matters the most considering our problem. However, none of the classifiers showed an improvement when using the dimensionality reduction algorithm. This is even more restrictive as the performance of the classifier grows linearly with the parameter d of our dimensionality reduction problem. Therefore, unlike what we saw for EEG data, reducing the dimensionality of the problem necessarily reduces the performance. Therefore, the classification process cannot be accelerated by keeping the same performance.

As an explanation, we may first hypothesize that the original raw sEMG signals contain less noise than the EEG signals. In fact, the dimensionality reduction process projects our data points on a lower space by keeping the original structure of the data and by trying to regroup points from similar classes and drive away points from different classes. If



(a) Euclidean (c) AIRM

(b) Log-Euclidean (d) Stein

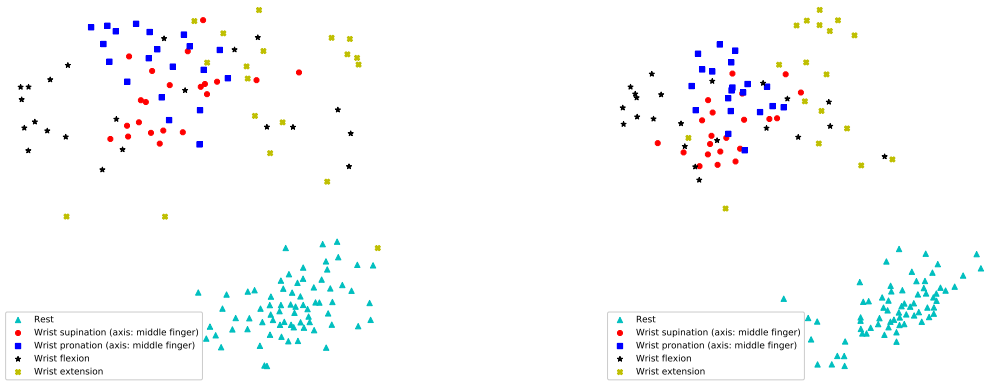
Figure 24: Relation between KNN performance and d for the different metrics defined on the SPD manifold and considering the Upper Myo with 41 hand movements.

the original points are really noisy, this procedure also removes part of the original noise. SEMG signals may contain less noise than the EEG signals as the EEG sensors measure the activity of the brain at the surface of the scalp. Therefore, many layers separate the source of the signals with the sensors. In the sEMG case, the sensors measure the muscular activity by being placed on the skin above these muscles. There are thus fewer layers separating the source to the measurement. Naturally, the more layers we have, the more noise appears. This is thus a possible explanation for the behavior of the dimensionality reduction algorithm.

Secondly, we hypothesize that the inevitable drop in performance when reducing dimension is caused by the high amount of classes considered. In fact, having more classes requires a bigger space to discriminate data points from these different classes. With EEG signals, we could highly reduce the dimension of the SPD manifold without affecting performance because the classification problem was only considering 4 classes. However, as we consider 41 classes for the sEMG problem, reducing dimension creates an inevitable drop in performance.

Classification with 5 hand movements To take comfort in these explanations, we evaluate the KNN classifier to classify 5 different hand movements. The movements chosen are: rest, wrist supination (axis: middle finger), wrist pronation (axis: middle finger), wrist flexion, and wrist extension.

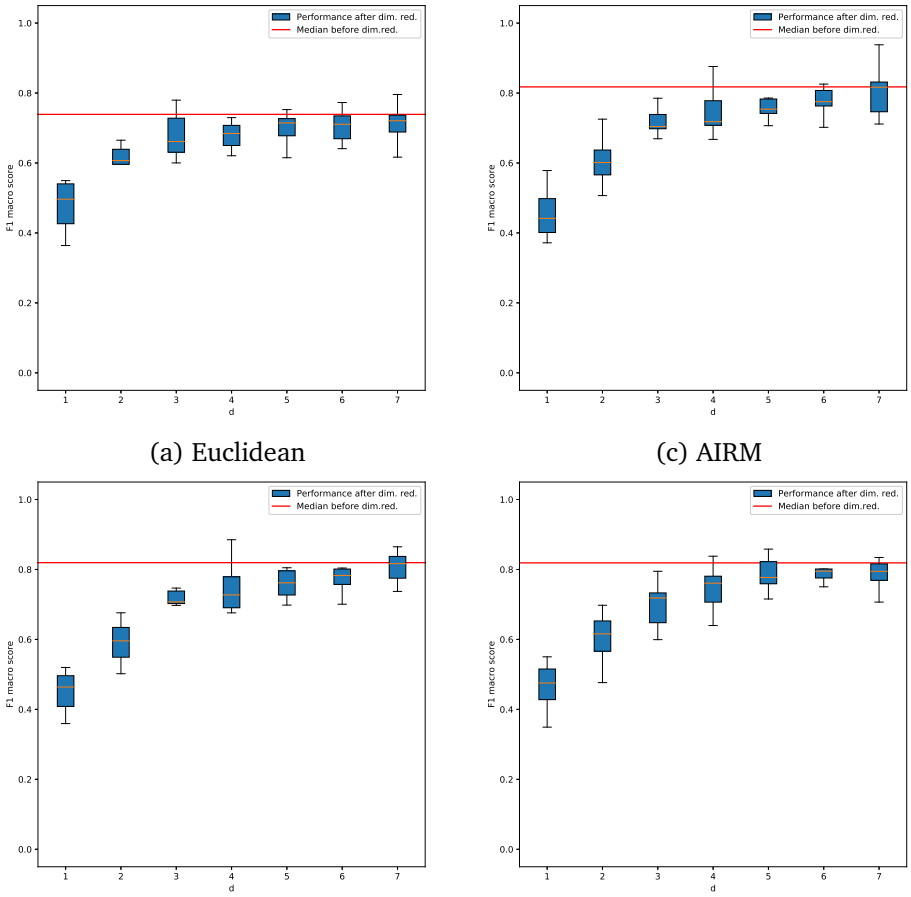
Figure 25 shows a visualization of the data points before and after dimensionality reduction. We first see that the class *rest* is isolated from the other classes, which explains that the classification of points belonging to this class are well classified. This leads to a F1 micro score way greater than the F1 macro score, as the class rest is overrepresented. Secondly, we observe that the original data points form classes which have specific structures. Unfortunately, the points of these different classes are somewhat intertwined, which explains the difficulty of the classification process with MDRM and KNN. The dimensionality reduction, even though the number of classes is not high, still does not separate the classes efficiently. Moreover, we see in fig. 26 and in fig. 31 that when we consider only 5 hand movements, reducing the dimension of the SPD manifold does not imply a drop in performance as important as what we saw in fig. 24 when we had 41 movements. Therefore, we could use dimensionality reduction in this case in order to decrease the computational time needed for classification, without a high loss in performance.



(a) Before dimensionality reduction.

(b) After dimensionality reduction.

Figure 25: Visualization of the training data of subject 1 considering the Upper Myo with 5 hand movements. Visualization is obtained with multi-dimensional scaling by considering the Stein metric.



(a) Euclidean

(c) AIRM

(b) Log-Euclidean

(d) Stein

Figure 26: Relation between KNN performance and d for the different metrics defined on the SPD manifold and considering the Upper Myo with 5 hand movements.

8. Conclusion & future work

In this thesis, we proposed to classify EEG and sEMG signals using a covariance matrix representation. We took advantage of the Riemannian structure of the SPD manifold to which the covariance matrices belong by using appropriate distances. Moreover, we used two classification algorithms, namely Minimum Distance to Mean (MDM) and K-Nearest-Neighbors (KNN), adapted to the SPD manifold. We chose to combine the covariance matrix representation with a dimensionality reduction algorithm to define a mapping between the original SPD manifold and a lower-dimensional SPD manifold, by keeping the intrinsic structure of the data.

We first tested our approach on EEG signals measured after visual stimulation at 4 different frequencies. The related classification problem consisted in assigning to the covariance matrices representing the EEG signals the correct frequency of the initial visual stimulation. Solving this problem requires to train and test a classification algorithm on separate sets of data. When these sets of data come from a same subject, we showed that the dimensionality reduction steps leads to both a gain in performance and a lower classification time for both classifiers and all the metrics defined on the SPD manifold. Depending on the classifier/metric combination considered, we reached a gain in classification performance up to 15% and a reduction of classification time up to 50%. We also proposed to solve this classification problem by considering training and test sets coming from different subjects. We have shown that this setting leads to an overall lower classification performance but that dimensionality reduction still leads to significant improvements.

We then considered sEMG signals measured on forearm muscles after the execution of various hand movements by some subject. The problem consisted in classifying covariance matrices representing the sEMG signals between 41 different classes representing separate hand movements. To solve this problem, we considered training and test sets coming from a same subject. Using the proposed classification algorithms on the covariance matrices led to a classification performance of around 60%. However, we showed that mapping the covariance matrices on a lower-dimensional SPD manifold leads inevitably in a decrease of this performance. This decrease is less important when considering fewer classes and a gain in computational time is then possible with dimensionality reduction without strongly impacting the performance.

Overall, the dimensionality reduction is particularly performant for EEG signals and less performant for sEMG signals. We hypothesize that this behavior is due to two main

reasons. First, EEG signals are more noisy than sEMG signals. In fact, many layers separate the source of the measured signals and the electrodes placed at the surface of the scalp in the EEG setting. These layers cause a lot of noise which is partly eliminated during dimensionality reduction, leading to an improvement of the performance of the classifiers. In the sEMG setting, the electrodes are placed on the skin just above the muscles resulting in lower noise. Therefore, dimensionality reduction has in this setting no denoising effect and does not improve classification performance. The second reason is that the sEMG classification problem considers much more classes than the EEG classification problem. Therefore, discriminating data points from different classes requires a high-dimensional SPD manifold, and we can not map these points onto a lower-dimensional SPD manifold without losing their intrinsic structure. Here, we insist on the fact that we do not advocate a specific classifier or metric as both the need in classification performance and computational efficiency depends on the intended application.

The outcome of this thesis motivates several future works. First of all, the covariance matrix representation used is not yet common within sEMG based research studies. Therefore, we wish to compare this approach with state of the art sEMG classification algorithms. Then, we may study more sophisticated classification algorithms combined with the dimensionality reduction technique in order to reach better classification performances. Also, we want to further investigate cross-subject classification algorithms by using transfer learning. Finally, we plan to test our classification algorithms in an online setting with a brain-actuated wheelchair and a myoelectric prosthesis.

References

[1] ABIRI, R., BORHANI, S., SELLERS, E., JIANG, Y., AND ZHAO, X. A comprehensive review of EEG-based brain-computer interface paradigms. *Journal of Neural Engineering* (2018).

[2] ABSIL, P.-A., MAHONY, R., AND SEPULCHRE, R. *Optimization algorithms on matrix manifolds*. Princeton Universty Press, 2007.

[3] ALMUHAMMADI, W. S., ABOALAYON, K. A. I., AND FAEZIPOUR, M. Efficient obstructive sleep apnea classification based on EEG signals. In *2015 Long Island Systems, Applications and Technology* (2015), pp. 1–6.

[4] ARSIGNY, V., FILLARD, P., PENNEC, X., AND AYACHE, N. Log-Euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine* (2006), pp. 411–421.

[5] BARACHANT, A., BONNET, S., CONGEDO, M., AND JUTTEN, C. Classification of covariance matrices using a Riemannian-based kernel for BCI applications. *Neurocomputing* (2013), pp. 172–178.

[6] BEVERINA, F., PALMAS, G., SILVONI, S., PICCIONE, F., AND GIOVE, S. User adaptive BCIs: SSVEP and P300 based interfaces. *PsychNology Journal* (2003), pp. 331–354.

[7] BINI, D., AND IANNAZZO, B. Computing the Karcher mean of symmetric positive definite matrices. *Linear Algebra and its Applications* (2013), pp. 1700 – 1710.

[8] BORG, I., AND GROENEN, P. Modern multidimensional scaling: Theory and applications. *Journal of Educational Measurement* (2006), pp. 277 – 280.

[9] BOUMAL, N. An introduction to optimization on smooth manifolds. unpublished, N.D.

[10] CASTELLINI, C., AND VAN DER SMAGT, P. Surface EMG in advanced hand prosthetics. *Biological cybernetics* (2008), pp. 35–47.

[11] CAVALCANTI GARCIA, M. A., AND VIEIRA, T. Surface electromyography: Why, when and how to use it. *Revista Andaluza de Medicina del Deporte* (2011), pp. 17–28.

[12] CHERIAN, A., AND SRA, S. Positive definite matrices: Data representation and applications to computer vision. In *Algorithmic Advances in Riemannian Geometry and Applications*. Springer, 2016, pp. 93–114.

[13] CONGEDO, M., BARACHANT, A., AND BHATIA, R. Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review. *Brain-Computer Interfaces* (2017), pp. 1–20.

[14] CONGEDO, M., RODRIGUES, P. L. C., BOUCHARD, F., BARACHANT, A., AND JUTTEN, C. A closed-form unsupervised geometry-aware dimensionality reduction method in the Riemannian manifold of SPD matrices. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (2017), pp. 3198–3201.

[15] DÖRING, M. Performance measures for multi-class problems, 2018. Accessed from `https://www.datascienceblog.net/post/machine-learning/performance-measures-multi-class-problems/` [Online; visited 23-April-2020].

[16] ESCABÍ, M. A. Biosignal processing. In *Introduction to Biomedical Engineering (Second Edition)*, Biomedical Engineering. Academic Press, 2005, pp. 549 – 625.

[17] HARANDI, M., SALZMANN, M., AND HARTLEY, R. Dimensionality reduction on SPD manifolds: The emergence of geometry-aware methods. *IEEE Trans. on Pattern Analysis and Machine Intelligence* (2018), pp. 48–62.

[18] HUANG, Z., WANG, R., LI, X., LIU, W., SHAN, S., VAN GOOL, L., AND CHEN, X. Geometry-aware similarity learning on SPD manifolds for visual recognition. *IEEE Transactions on Circuits and Systems for Video Technology* (2016), 2513–2523.

[19] JAKUB, C. The ultimate guide to binary classification metrics, 2019. Accessed from `https://towardsdatascience.com/the-ultimate-guide-to-binary-classification-metrics-c25c3627dd0a#c5ca` [Online; visited 20-May-2020].

[20] JAYARAM, V., ALAMGIR, M., ALTUN, Y., SCHOLKOPF, B., AND GROSSE-WENTRUP, M. Transfer learning in brain-computer interfaces. *IEEE Computational Intelligence Magazine* (2016), pp. 20–31.

[21] KALUNGA, E., CHEVALLIER, S., BARTHÉLEMY, Q., DJOUANI, K., HAMAM, Y., AND MONACELLI, E. From Euclidean to Riemannian Means: Information Geometry for SSVEP Classification. In *Geometric Science of Information* (2016), Lecture Notes in Computer Science, pp. 595–604.

[22] KALUNGA, E. K., CHEVALLIER, S., RABREAU, O., AND MONACELLI, E. Hybrid interface: Integrating BCI in multimodal human-machine interfaces. In *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics* (2014), pp. 530–535.

[23] KAMARAJAN, C., PANDEY, A. K., CHORLIAN, D., MANZ, N., STIMUS, A. T., ANOKHIN, A. P., BAUER, L. O., KUPERMAN, S., KRAMER, J., BUCHOLZ, K. K., SCHUCKIT, M. A., HESSELBROCK, V. M., AND PORJESZ, B. Deficient event-related theta oscillations in individuals at risk for alcoholism: A study of reward processing and impulsivity features. *PLoS ONE* (2015).

[24] LAI, C. Q., IBRAHIM, H., ABDULLAH, M. Z., ABDULLAH, J. M., SUANDI, S. A., AND AZMAN, A. Literature survey: Recording set up for electroencephalography (EEG) acquisition. In *IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)* (2018), pp. 333–338.

[25] LEE, J. M. *Introduction to Riemannian manifolds*. Springer International Publishing, 2018.

[26] LI, X., SONG, D., ZHANG, P., ZHANG, Y., HOU, Y., AND HU, B. Exploring EEG features in cross-subject emotion recognition. *Frontiers in Neuroscience* (2018).

[27] LIANG, P., YANG, C., WANG, N., AND LI, R. A discrete-time algorithm for stiffness extraction from sEMG and its application in antidisturbance teleoperation. *Discrete Dynamics in Nature and Society* (2016), pp. 1–11.

[28] LIU, D., LIU, S., LIU, X., ZHANG, C., LI, A., JIN, C., CHEN, Y., WANG, H., AND ZHANG, X. Interactive brain activity: Review and progress on EEG-based hyperscanning in social interactions. *Frontiers in psychology* (2018), p. 1872.

[29] LIU, W., KE, Y., LIU, P., DU, J., KONG, L., LIU, S., AN, X., AND MING, D. A cross-subject SSVEP-BCI based on task related component analysis. *IEEE Engineering in Medicine and Biology Society* (2019), pp. 3022–3025.

[30] MASSART, E. M., AND CHEVALLIER, S. Inductive means and sequences applied to online classification of EEG. In *Geometric Science of Information (GSI)* (2017), pp. 763–770.

[31] MASSÓ, N., REY, F., ROMERO, D., GUAL, G., TUTUSAUS, L., AND GERMÁN ROMERO, A. Surface electromyography applications. *Medicina de l'esport* (2010).

[32] MICHELI, L., MOSCONE, D., AND PALLESCHI, G. Biosensors for non-invasive measurements. *Biosensors for Medical Applications* (2012), pp. 263 – 300.

[33] MISHRA, A. Metrics to evaluate your machine learning algorithm, 2018. Accessed from https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234 [Online; visited 20-May-2020].

[34] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., AND DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* (2011), pp. 2825–2830.

[35] PENNEC, X., FILLARD, P., AND AYACHE, N. A Riemannian Framework for Tensor Computing. *International Journal of Computer Vision* (2006), pp. 41–66.

[36] PIZZOLATO, S., TAGLIAPIETRA, L., COGNOLATO, M., REGGIANI, M., MÜLLER, H., AND ATZORI, M. Comparison of six electromyography acquisition setups on hand movement classification tasks. *PLOS ONE* (2017).

[37] SATHEESH KUMAR, J., AND BHUVANESWARI, P. Analysis of electroencephalography (EEG) signals and its categorization. *Procedia Engineering* (2012), pp. 2525 – 2536.

[38] SHIH, J. J., KRUSIENSKI, D. J., AND WOLPAW, J. R. Brain-computer interfaces in medicine. *Mayo Clinic Proceedings* (2012), pp. 268 – 279.

[39] SINGLA, R. SSVEP-based BCIs. In *Evolving BCI therapy - engaging brain state dynamics* (2018), IntechOpen. Available from https://www.intechopen.com/books/evolving-bci-therapy-engaging-brain-state-dynamics/ssvep-based-bcis.

[40] SOUSA, A., AND TAVARES, J. Surface electromyographic amplitude normalization methods: A review. *Electromyography: New Developments, Procedures and Applications* (2012), pp. 85–102.

[41] SRA, S. A new metric on the manifold of kernel matrices with application to matrix geometric means. In *Advances in Neural Information Processing Systems 25* (2012), Curran Associates Inc., pp. 144–152.

[42] SUDARSAN, S., AND SEKARAN, E. Design and development of EMG controlled prosthetics limb. *Procedia Engineering* (2012), pp. 3547–3551.

[43] TANG, J., LIU, Y., HU, D., AND ZHOU, Z. Towards BCI-actuated smart wheelchair system. *BioMedical Engineering OnLine* (2018).

[44] TU, L. W. *An introduction to manifolds*. Springer International Publishing, 2011.

[45] TUZEL, O., PORIKLI, F., AND MEER, P. Region covariance: A fast descriptor for detection and classification. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2006), 589–600.

[46] VAN DER STELT, O., AND BELGER, A. Application of electroencephalography to the study of cognitive and brain functions in schizophrenia. *Schizophrenia bulletin* (2007), pp. 955–970.

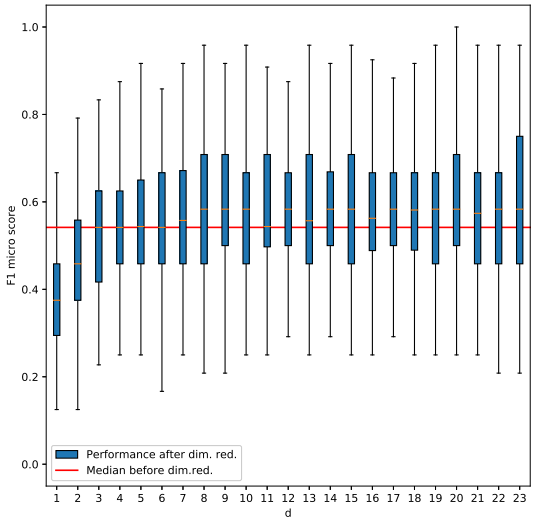
[47] VEMULAPALLI, R., AND JACOBS, D. W. Riemannian metric learning for symmetric positive definite matrices, 2015.

[48] YADAVA, M., KUMAR, P., SAINI, R., ROY, P., AND DOGRA, D. Analysis of EEG signals and its application to neuromarketing. *Multimedia Tools and Applications* (2017), pp. 19087–19111.

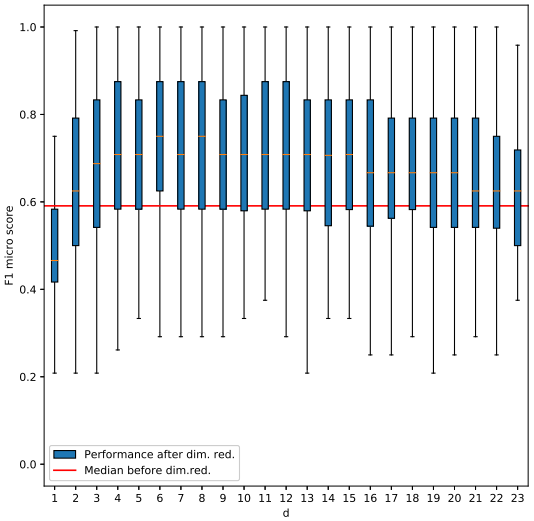
[49] ZANINI, P., CONGEDO, M., JUTTEN, C., SAID, S., AND BERTHOUMIEU, Y. Transfer learning: A Riemannian geometry framework with applications to brain–computer interfaces. *IEEE Transactions on Biomedical Engineering* (2018), pp. 1107–1116.

A. Appendix

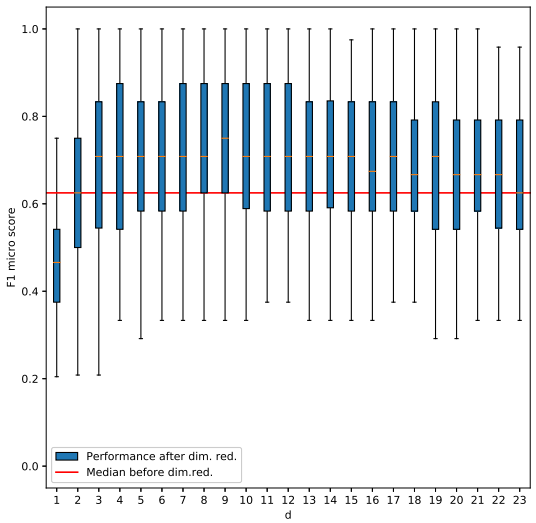
This appendix contains figures which were not placed in the main text because they were either slightly redundant with other figures, or too large.



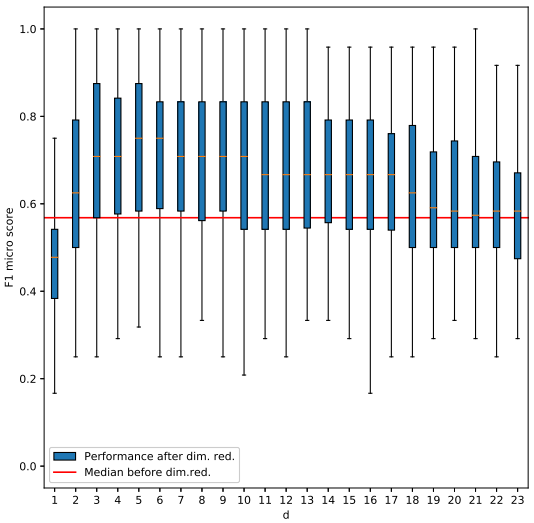
(a) Euclidean



(c) AIRM



(b) Log-Euclidean



(d) Stein

Figure 27: Relation between KNN performance and d for the different distances.

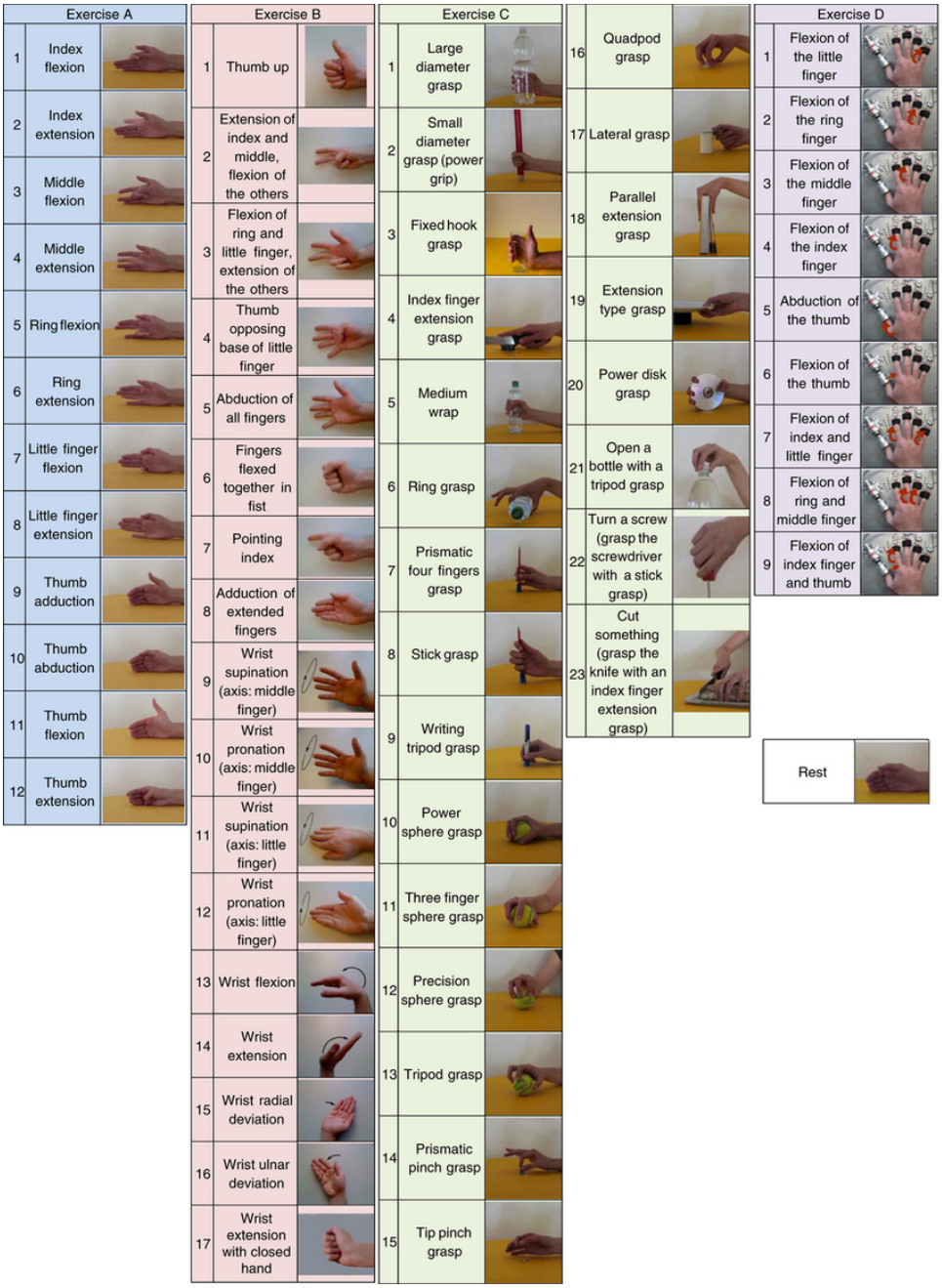
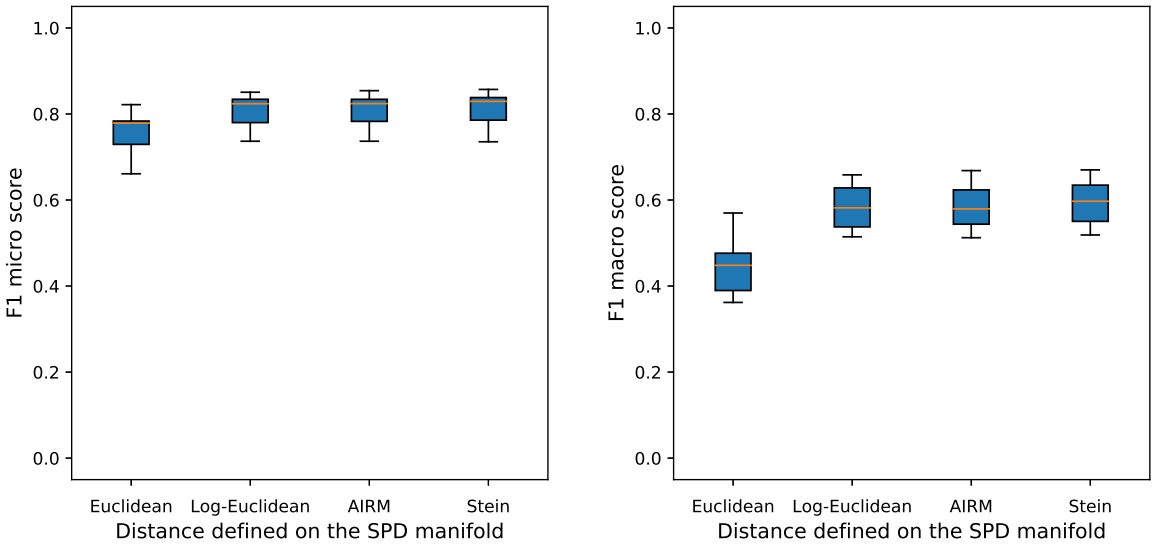


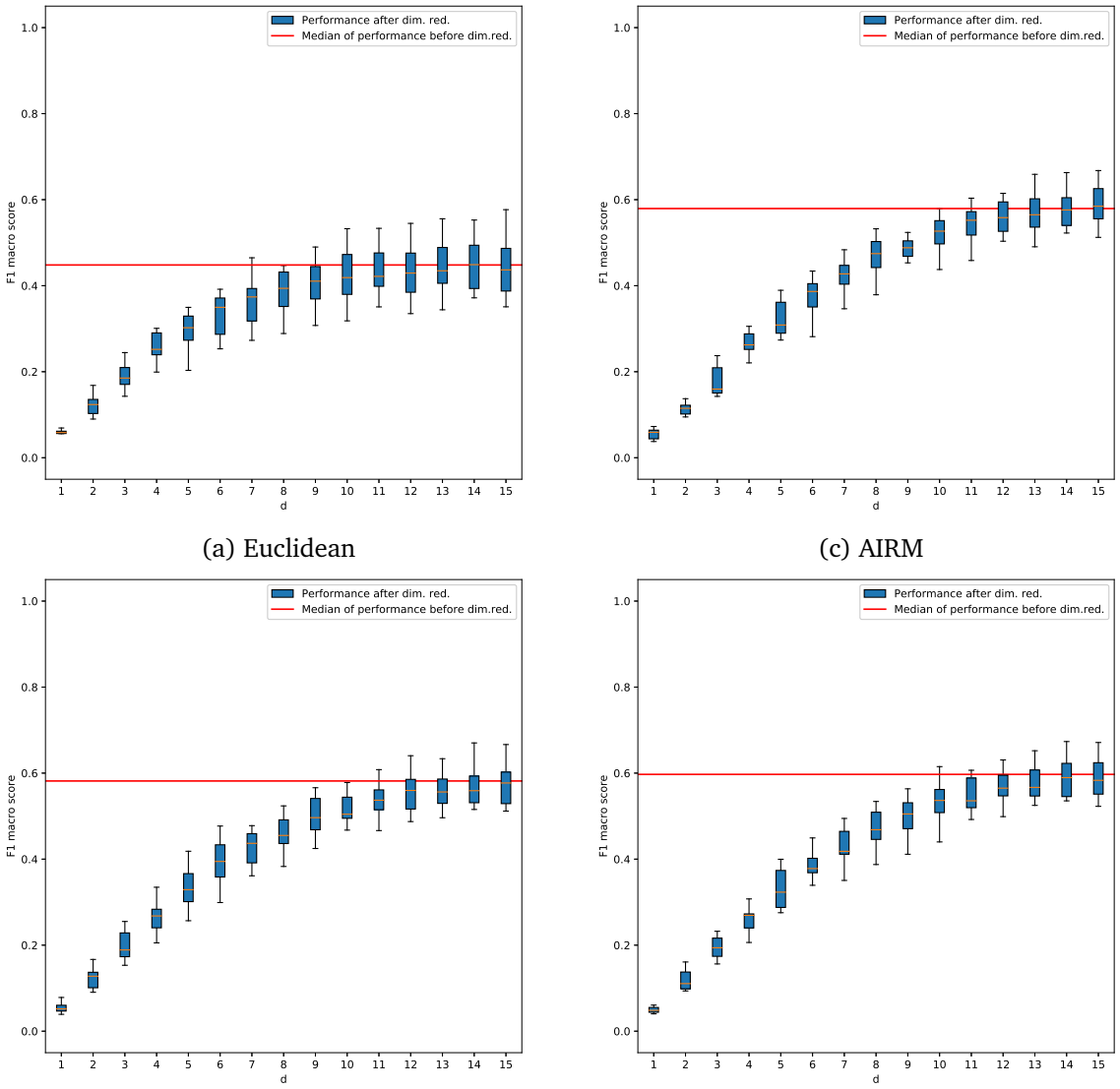
Figure 28: Hand movements of the Ninapro database[36].



(a) F1 micro score in function of the distance defined on the SPD manifold.

(b) F1 macro score in function of the distance defined on the SPD manifold.

Figure 29: KNN performance for the double Myo setup before dimensionality reduction considering all subjects.



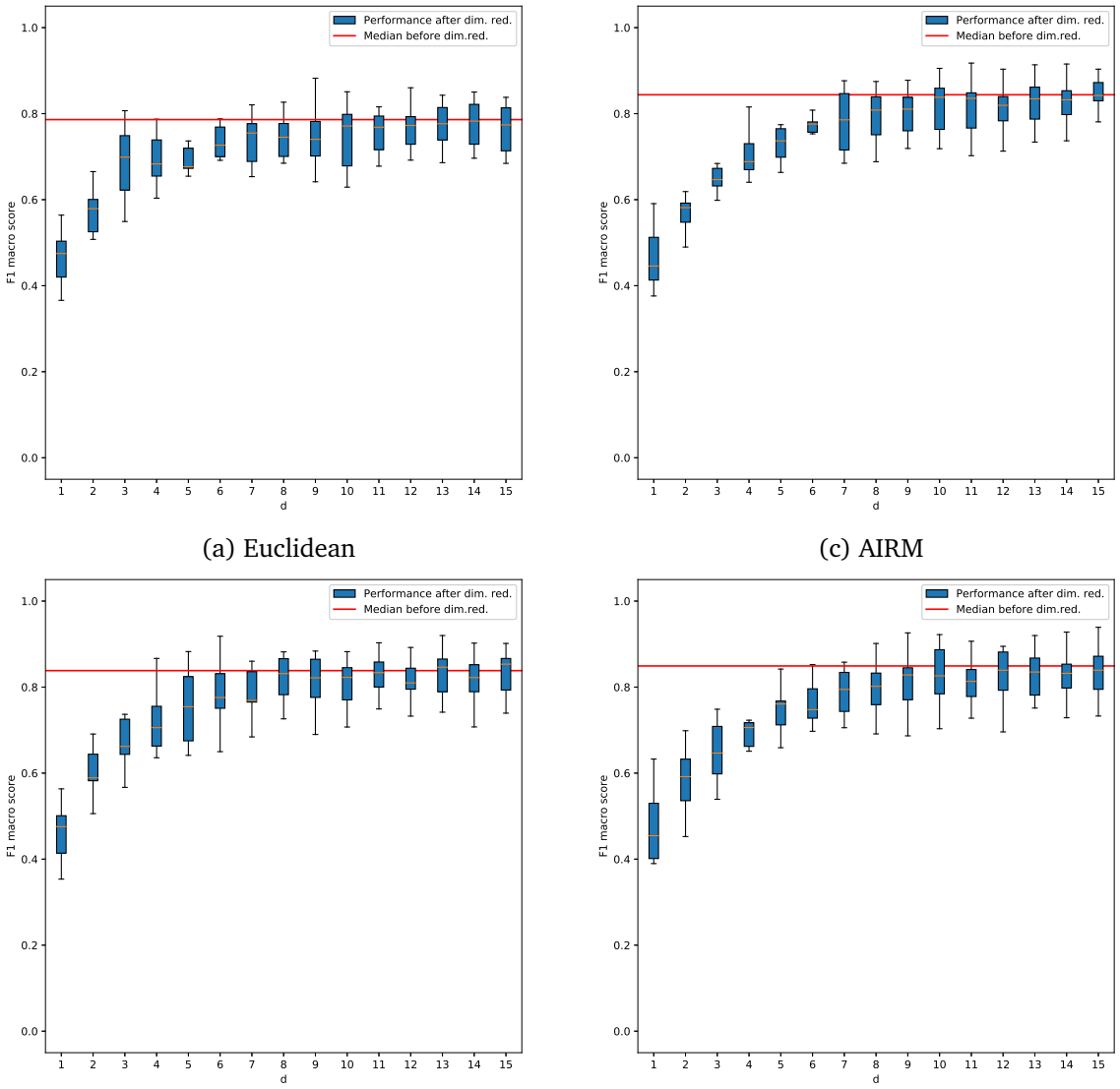
(a) Euclidean

(c) AIRM

(b) Log-Euclidean

(d) Stein

Figure 30: Relation between KNN performance and d for the different metrics defined on the SPD manifold, and considering the double Myo setup.



(a) Euclidean

(c) AIRM

(b) Log-Euclidean

(d) Stein

Figure 31: Relation between KNN performance and d for the different metrics defined on the SPD manifold, and considering the double Myo setup with 5 hand movements.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**