DEVELOPMENT OF A HYBRID EEG-NIRS BRAIN-COMPUTER INTERFACE FOR MULTIPLE MOTOR TASKS

A Thesis

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Alessio Paolo Buccino

August 2015

DEVELOPMENT OF A HYBRID EEG-NIRS BRAIN-COMPUTER INTERFACE FOR MULTIPLE MOTOR TASKS

Alessio Paolo Buccino

APPROVED:

Marc Garbey, Chairman
Dept. of Computer Science

Ahmet Omurtag
Dept. of Biomedical Engineering

Shishir Shah
Dept. of Computer Science

Dean, College of Natural Sciences and Mathematics

Acknowledgements

I would like to thank the Atlantis CRISP international double degree program for providing me financial support during my MSc studies. I would also like to thank my thesis advisor Dr. Omurtag, who solely supervised my work in this thesis. I conducted all the experiments and analyses in this thesis under Dr. Omurtag's supervision with the resources provided in his laboratory in the Biomedical Engineering Department. I would like to thank also Dr. Anna Bianchi, my advisor from Politecnico di Milano, for her valuable suggestions. Special thanks go to Dr. Garbey, and Dr. Baselli, who coordinated and organized the double degree. I am very grateful to Dr. Shah for being available and interested in being part of my committee. I really thank Hasan Onur Keles, Haleh Aghajani, and Nesrine Aroua because they have been not only professional colleagues, but also nice and valued friends. Moreover, I would like to thank all my friends that supported me during this year, especially Gianluca Meloni and Valentina Simonetti for sharing this precious experience. I want to thank my family because despite the distance they have been really close and supportive. Finally, I thank my girlfriend, Silvia, simply because she is wonderful.

DEVELOPMENT OF A HYBRID EEG-NIRS BRAIN-COMPUTER INTERFACE FOR MULTIPLE MOTOR TASKS

An Abstract of a Thesis

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Alessio Paolo Buccino

August 2015

Abstract

Non-invasive Brain-Computer Interfaces (BCI) have demonstrated great promise for neuroprosthetics and assistive devices. Here we aim to investigate methods to combine Electroencephalography (EEG) and Near-infrared Spectroscopy (NIRS) in an asynchronous Sensory Motor rhythm (SMR)-based BCI. We attempted to classify 4 different executed and imagined movements: Right-arm, Left-arm, Right-hand, and Left-hand tasks.

Previous studies demonstrated the benefit of EEG-NIRS combination, without processing the NIRS signal with online implementable methods for an asynchronous paradigm. Since normally the NIRS hemodynamic response shows a long delay, we investigated new features, involving slope indicators, in order to immediately detect changes in the signals. Moreover, Common Spatial Patterns (CSPs) have been applied to both EEG and NIRS signals. Fifteen healthy subjects took part in the experiments, and, because 25 trials per class were available, CSPs have been regularized with information from the entire population of participants and optimized using genetic algorithms.

Different approaches have been investigated for feature extraction, classification, and signal association. The results showed that a hybrid EEG-NIRS approach enhances the performance of EEG or NIRS separately. Better performances are achieved for the motor execution paradigm, probably due to the subjects' inexperience in motor imagery, despite the small dataset available.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project Presentation

The thesis is divided in 6 different chapters, each one with a different purpose. The chapters can be read separately, but in order to have an overview of the work it is suggested to go through the chapters sequentially. The present chapter explains how the thesis is organized and gives a brief summary of the other chapters; moreover it explains what is the aim of the work. Chapter 2 introduces the field in which the present study is inserted. It explains what is a Brain-Computer Interface and what is the state of the art in the literature. Chapter 3 deals in detail with the methods used in the work, from the data acquisition to the data classification and evaluation passing through the signal processing. Chapter 4 presents the results of the methods and tries to give a meaningful explanation in the context. Chapter 5 draws the conclusions and it discusses issues and problems as well as positive aspects

and future development. Section 5.7 sums up the entire project and concludes the thesis.

1.2 Aim of the Work

The aim of the work is mainly to develop the Brain-Computer Interface (BCI). At the start of the project, in fact, there was no previous study on the topic, and the main goal was to develop and implement the tools and programs to be able to collect, analyze, and use the BCI. This included the design of the experimental setup and procedure, the development of the real-time feedback application for the data acquisition part, and the investigation of different methods to find the most suitable configuration in which Electroencephalography (EEG) and Near-infrared Spectroscopy (NIRS) can be combined to yield a robust BCI. The EEG-NIRS combination in the BCI field, in fact, is poorly understood. One of the purposes of the project is to understand whether a hybrid BCI including EEG and NIRS can be beneficial and feasible. In particular, the use the Common Spatial Patterns method, which has been widely studied for EEG-based BCI, is applied for NIRS and compared to more standard processing and feature extraction methods. Another key point of the project is the use of a minimal experimental setup distributed only on the motor cortex, in order to allow a faster setup in a possible product development and/or clinical translation. With the same philosophy, the experimental procedure is also shortened, yielding a small number of trials available for optimizing the classification. The performance and the applicability of the algorithms is therefore tested in a situation that could

be assimilable to a clinical environment. Lastly, in the study both motor execution and motor imagery are involved, and one of the aims is to understand whether no experience in the latter is an important factor in the system performance, since none of the participants have had ever before experienced motor imagery.

Chapter 2

Background

The background chapter is divided as follows: section 2.1 introduces the basic concepts and principles of Brain-Computer Interfacse, section 2.2 describes how the motor cortex and the motor control work, sections 2.3 and 2.4 present an overview of the recording systems used in the project, with particular focus on motor tasks, and section 2.5 reviews the state-of-the-art about EEG-based, NIRS-based, and EEG-NIRS-based Brain-Computer Interfaces.

2.1 Brain-Computer Interfaces

A Brain-Computer Interface (BCI) is a system that measures Central Nervous System (CNS) activity and translates it into artificial commands to replace, restore, enhance, supplement, or improve the natural CNS output [66].

The CNS is the part of the nervous system that includes only the brain and the spinal cord. Almost all BCI research has been focusing on measuring the signal from the brain itself. The great challenge of Brain-Computer Interfaces is to manage to decode brain activity without being invasive with the patient-user, i.e., acquiring signals from outside the brain. Different modalities of signals, representing and giving information about various aspects of the complex activity of the CNS, can be used for BCI purpose, such as signals measuring the electrical activity of populations of neurons, the magnetic fields induced by their activity, or the metabolic activity. The different modalities will be discussed in detail later on in this section.

Let us now make a step back in the past, to understand when, where, and how everything began. In 1924, Professor Hans Berger from the university of Jena, Germany, made an astonishing discover: the electrical activity of the brain could be measured by electrodes placed on the scalp, without the need to surgically open the scalp and place the electrodes right on the brain. Berger was inventing a new and non-invasive way to investigate the brain activity and functions: the Electroencephalography (EEG). Unfortunately, Berger's work, which culminated in several articles on the use of EEG for clinical diagnosis (Berger 1929, [3]), was prematurely shut down in 1938, when the German government forced him to an early retirement. His pioneering work was starting to be internationally recognized from the scientific community, and a young American neurophysiologist, Hebert Jasper, exported the EEG technique for the first time in the USA in 1935. Jasper was so thankful to Berger's work, that in 1938, just before WWII, he expressed his holiday greetings to the German neurologist with the drawing shown in Figure 2.1. The drawing can be

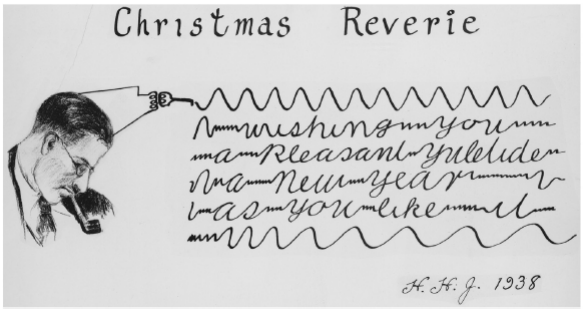


Figure 2.1: Herbert Jasper's drawing to Hans Berger, an early idea of BCI.

interpreted as a very early Brain-Computer Interface: brain signals, in fact, flow out from the head and are interpreted and translated in another artificial output, such as the English language.

Although the idea of interfacing the human brain with a computer or a machine was certainly earlier, the first appearance of the term *Brain-Computer Interface* was in 1977, coined by Jacques Vidal [62, 61]. In his works, Vidal used VEPs (Visual Evoked Potentials) to determine the direction of the gaze, with the purpose of controlling a cursor. Throughout the years, the term Brain-Computer Interface has been juxaposed with the term *Brain-Machine Interface* (BMI). This term was initially used to indicate direct cortical stimulation [21]; later it started to refer to systems that control external *machines* from cortical activity recorded from implanted microelectrodes [9]. At present the two terms (BCI and BMI) are considered synonyms and they are completely interchangeable. In this work, the word *Brain-Computer Interface* will be used, because it suggests a wider range of applications and more flexibility (the word *machine* seems too static with respect to the idea of *computer*).

BCI can be furtherly classified depending on their implementation:

Dependent/Independent: A *dependent* BCI uses signals that depends on a natural CNS output, for example muscular activity. An *independent* BCI, instead, uses signals that do not produce a CNS output, e.g., motor imagery or attention level [69].

Hybrid: A *hybrid* BCI uses different kind of brain signals. The term is not limited to those BCIs that use different modalities (e.g., EEG + NIRS or EEG + Magnetoencefalography), but it can also identify BCIs that use the same modality, in two diverse ways, for example an EEG-based BCI that produces its output from Sensory Motor Rhythms (SMRs) and Visual Evoked Potentials (VEPs).

Synchronous/Asynchronous: A synchronous BCI is not self-paced and the user can communicate with the BCI only in defined time frames, which are usually indicated by either visual or acoustic cues. On the other hand an asynchronous BCI does not require external cues and the user can communicate with it every time.

With this classifications, the aim of this work expressed in section 1.2 can be reformulated. For motor execution the BCI would be *dependent*, *hybrid*, and *synchronous*, while for motor imagery it can be defined as *independent*, *hybrid*, and *synchronous*. The only difference between the two paradigms is in the dependency of the output command on natural CNS output.

How do Brain Compute Interface work? How can they translate the user mental states into application commands? A BCI, first of all, must have four different components. It has to measure brain activity, it has to provide feedback, it has to

be real-time, and it must be intentional [17]. While the first three components are milestones of a BCI system, the intentionality can be argued. In fact, some consider a BCI also a so called *passive* system that does not require an intention by the user, but is based only on its state (e.g., an automatic brake system for lack of driver's attention). On the contrary, *active* BCI provide control and communication [17]. To understand how a BCI works, let us use Fig. 2.2 and 2.3 as support. Fig. 2.2 shows a general scheme of a Brain-Computer Interface: brain signals are acquired and processed in real-time. The processing consists of a pre-processing part (usually filtering and normalization) and a feature extraction step, that *extracts* particular characteristics from the signals, that are used by the translation algorithm to produce outputs to command and application, e.g., a computer program, a wheelchair or the grasping of a robotic hand. While Fig. 2.2 only show the final use of a BCI, i.e., the online or feedback application, in order to develop the algorithms to output the commands (translation algorithms) another phase is needed: the training or calibration phase. The translation algorithm, in fact, is usually a classification or regression algorithm that needs to *learn* the patterns and characteristics of the data in order to produce new output once it is presented new data. This kind of machine learning algorithms fall in the category of *supervised learning* methods, because they need a set of good or right examples to understand what is the best model to produce the desired output (the examples). For a better understanding of the process, Fig. 2.3 shows the two different phases of the *supervised learning* paradigm applied to an EEG-based BCI. The *calibration* phase precedes the *feedback* phase on a temporal line. First of all, for the training of the classifier/translation algorithm, the subject must perform a series

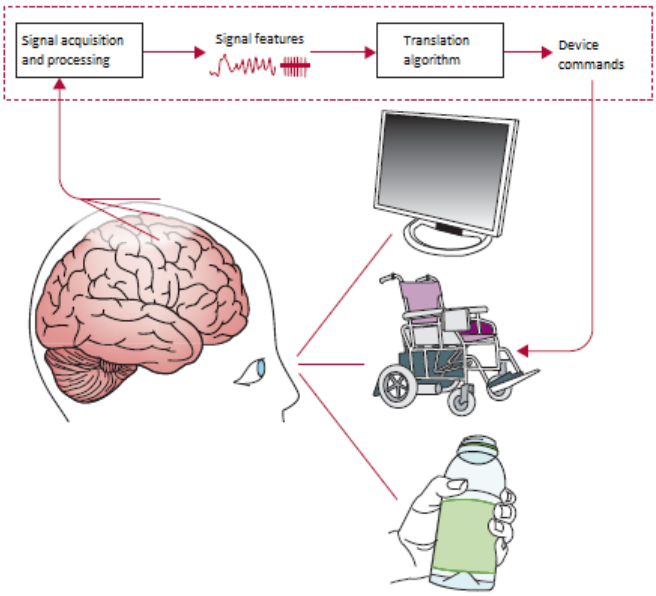


Figure 2.2: Basic design and operation of a Brain-Computer Interface: brain signals are acquired and processed. Then features are extracted from them and a translation algorithm is used to output commads to a device from the feature set. (Figure taken from [69]).

of tasks multiple times (in this case left- and right-hand motor imagery), to build the so called *training* set used to *learn* the patterns in the signals. From the classified trials (examples), signals *features* are extracted, e.g., powers in different frequency bands or variances of different channels. The machine learning algorithm creates a model, or more generically a set of rules, that maps the distributions of these *features* into outputs that minimizes the errors between the *right* examples provided (in the figure left-hand imagination is class -1, while right-hand is class +1) and the algorithm outputs. Once the algorithm is trained, it can be used for the real-time feedback application: from *new* brain signals features are extracted and used by the trained classifier to output commands to an application. The application itself will be the medium through which the loop between the user and the BCI is closed.

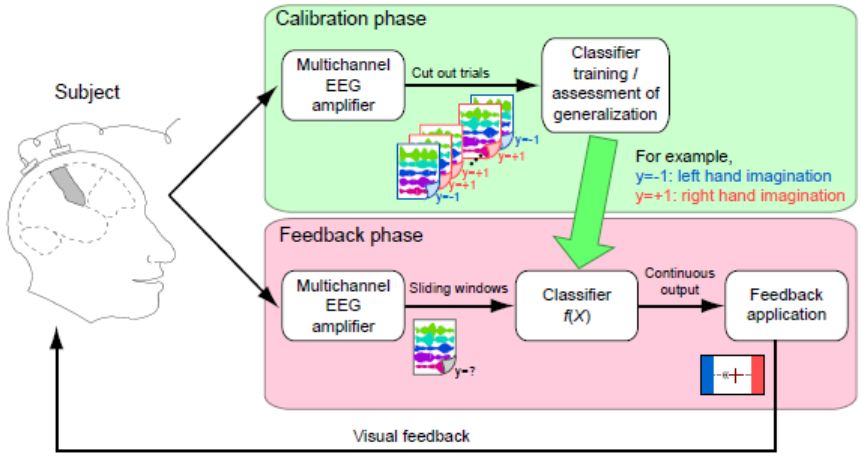


Figure 2.3: Detailed scheme of the *supervised learning* paradigm applied to an EEG-based BCI: in the calibration phase, the classifier is trained using the training data. The classifier is then used in the feedback phase to compute new outputs given new EEG signals. (Figure taken from [5]).

As far as the possible use of BCIs, the applications are diverse. As stated in the definition given at the very beginning of this section, the BCI output could serve as replacement, restoration, enhancement, supplement, or improvement. Let's give some concrete examples of BCI for all these possible purposes [66]:

Replacement: A BCI could serve as replacement for lost natural functionalities and output due to impairment or injuries. An example could be the use of a BCI to control the direction of a wheelchair in disabled people or one to select letters and compose sentences for a person who can no longer talk.

Restoration: Although restoration and replacement might seem quite alike, they are slightly different. In fact, as explained above, while a BCI could replace a lost function with a completely artificial device such as a wheelchair, it could

also restore a function no longer accessible, e.g., arm movements in a paralyzed person, by triggering an electrical stimulation of the muscles. In this case the functionality of the arm itself would be restored by intervening on the limb directly, without the use of a completely and external device.

Enhancement: A BCI could enhance capabilities already present in a person. Think about a very demanding task such as driving for a long distance: in this perspective a BCI could be used to detect a lack of attention in the driver, preventing an accident. The BCI would augment the performance of the person in the task.

Supplement: A supplement to the natural CNS output could be given by means of a BCI. In this sense, the BCI would add further capabilities to the natural muscular output: for example one could control an external robotic arm with the brain and use it in combination with his/her natural arms.

Improvement: It has been proven that rehabilitation success (e.g., in stroke patients) is higher when the patient actively participate in the physiotherapy sessions by thinking of moving his/her impaired arm, for example, while the physiotherapist actually moves it [69]. This tends to augment and facilitate the rehabilitation of the neural pathway due to the high plasticity of the neural system. In this framework, a BCI could be used to improve the outcome of a rehabilitative process by triggering the onset of an orthotic device when the patient thinks of the movement or by activating a Functional Electrical Stimulation (FES) on the impaired arm.

The BCI research, given all the possible and useful applications just described, is a fast-growing field that could and will help many people in improving their life conditions. Apart from the biomedical uses, which mainly fall in the categories of *replacement*, *restoration*, and *improvement*, in the last decades BCIs have been also targeting a wider range of potiential users, overall with the purposes of *enhancement* and *supplement*. They are starting to be accessible and low cost and they will probably be the new hi-tech trend of the years to come, for example for videogames and home automation. Probably, in a not-too-far future, we will be able to interact with virtual realities and to switch on and off the living room lights only with our thoughts.

After this *ex-cursus* with a future perspective, let us go back to the BCI system. The main complexity of interfacing a computer with a human brain is probably the fact that we are dealing with really highly plastic and adaptive systems, that can possibly learn one another. This is a key concept in the BCI field: the computer *learns* the user brain activity and how to map it in output commands; the user, during the real-time use of the system, *learns* to improve his/her performance through the BCI feedback [67]. This mutual adaptation makes it very difficult to evaluate the actual performance of a BCI. In fact, for the reasons expressed before, a *static* evaluation of the system, such as the accuracy of the classifier (right outputs / total outputs), is not enough, and it should be accompanied by an online evaluation not only in terms of accuracy, but also in terms of time, of subject satisfaction and improvement through the use. A global evaluation that takes into account all these possible factors and variables is usually very time consuming and difficult to perform. On the other

hand, a static evaluation is clearly the base of the BCI effectiveness, and it can be seen as a quantitative starting point to improve the user BCI performance. A metric that is usually suitable to evaluate many aspects of a BCI system is the Information Transfer Rate (ITR), that depends on the number of classes used, on the time needed to classify, and on the classification accuracy [5]. ITR is measured in bits/minute, and it actually indicate the amount of information that a user can communicate to the BCI system in a minute.

All Brain-Computer Interfaces start from the same point: acquiring or measuring brain activity. The measurement of brain activity can be done in many ways, differing from each other in terms of technology, invasiveness, and nature of the recorded signals. Brain activity can be measured from three different aspects: as electrical, magnetic, and metabolic activity. The first one is the result of billions of neurons that *talk* to each other through action potentials or spikes which have an electric nature: they are flows of ions, which makes the definition of electric current. The global electrical and highly dynamic activity, for the well-known relation between electrical and magnetic fields (Gauss - Faraday-Neumann laws) can be recorded also by looking at the magnetic fields produced by this activity. Whereas electrical and magnetic activity are extremely connected and correlated, the metabolic activity represents a completely different phenomenon: the consumption of energy from the brain. Usually the metabolism is measured through bio-markers, for example one can measure the concentration of hemoglobin, which is the carrier of oxygen in blood, in a region of the brain and relate it to the actual energy consumption of that region. The

following will give a quick and concise review of the different techniques to measure brain activity in its many natures.

As far as electrical activity, *electroencephalography* (EEG) has been introduced and will be treated in detail in subsection 2.3. More invasive techniques allow us to obtain higher quality signals in terms of Signal-to-Noise Ratio (SNR) and spatial resolution. The *electrocorticogram* (ECoG) uses a grid of electrodes placed just above the cortex on the arachnoid, and the scalp must be surgically opened. ECoG can acquire signals from a huge portion of the brain with a relatively high spatial resolution (2-10 mm). The most invasive technique to measure electrical activity of the brain is the *Local Field Potential* (LFP) recordings: LFPs are microlevel phenomena recorded within the cortex by microelectrodes inserted inside the brain. In particular, LFPs can record the synaptic activity of neurons within $\sim$1 mm and action potentials within $\sim$0.1 mm from the electrode tip [66]. Although their extremely high spatial resolution, it would be difficult to cover large areas of the brain using LFPs, and for this reason they are minimally used for BCI purposes.

The recording of the magnetic activity of the brain is called *Magnetoencefalography* (MEG). The main advantage of MEG with respect to EEG (here we compare MEG with EEG because of their non-invasive nature) is that the skull and other tissues separating the brain from the outside are transparent to magnetic fields; however, the orientation of magnetic sources play an important role making the MEG sensor selectively sensitive to sources of different orientations.

Metabolic or functional activity can be recorded through three different techniques: *Positron Emission Tomography* (PET), *functional Magnetic Risonance Imaging* (fMRI) and *Near-infrared Spectroscopy* (NIRS) (the latter will be discussed in detail in section 2.4). Due to its very low temporal resolution ($\sim$1 frame every 40 s), cumbersomeness, and cost, PET has not been targeted as a possible BCI signal acquisition system. fMRI use for BCI applications could be feasible in terms of temporal resolution, on the order of 1 s, but it is definitely prohibitive for the non-portability and cost of the system. The only suitable metabolic signal that satisfies the BCI requirements for temporal resolution and portability is the NIRS, which will be discussed in detail in section 2.4.

The next section introduces the basic mechanisms involved in motor control which modulate the non-invasive measurements that are used in the current work.

2.2 Motor Control and Motor Cortex

The development of the BCI described in this work relies on motor-related brain activity. This section has the purpose of introducing some general concepts about the complex and not entirely understood neural mechanisms underlying motor tasks. First of all, Fig. 2.4 shows the *actors* mainly involved in motor control. M1, colored in green, is the primary motor cortex and it has been identified and studied since 1870, when Eduard Hitzig and Gustav Fritsch discovered that the electrical stimulation of that part of the cortex resulted in the movement of controlateral parts of the body [66]. M1 is the main area (primary) involved in motor execution, but other areas,

such as the Premotor Cortex (PMC), the Supplementary Motor Area (SMA), the Posterior Parietal Cortex (PPC), and somehow the primary Somatosensory cortex (S1) actively participate in the motor control process.

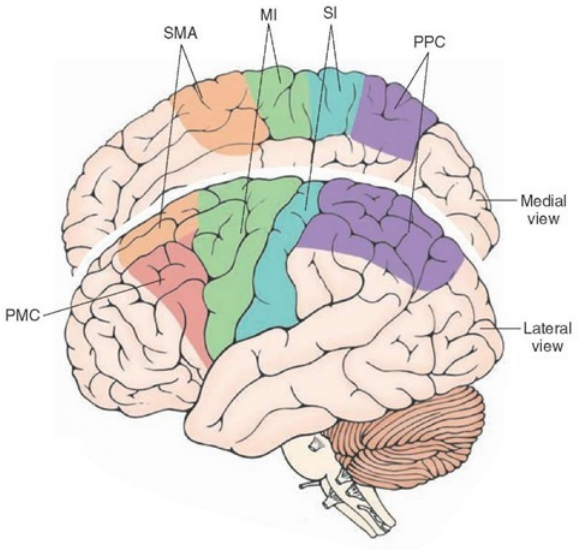


Figure 2.4: Motor cortex areas involved in motor control: M1 - Primary Motor Cortex, PMC - Pre Motor Cortex, SMA - Supplementary Motor Area, PPC - Posterior Parietal Cortex, and S1 - Primary Sensorymotor Cortex.

One of the most important discoveries about how the motor cortex functionality was made by Penfield, a pioneer neurosurgeon who mapped for the first time the somatotopic organization of the primary motor cortex [41]. His studies led to the familiar *Penfield's* (or *motor*) *homunculus*, displayed in Fig. 2.5. The areas devoted to control more complex parts of the body, such as hands or the face, are spatially more extended than others mapped for easier motor control functions, like the leg or the arm (made up of a less number of bigger muscles). The somatotopic organization of M1, though, is not as simple as depicted in *Penfield's homunculus*. More accurate cortical maps were obtained later, e.g., by Park et al. [39], using microstimulation

of the primary cortex in monkey. They demonstrated that different areas of M1 are involved in the control of distal and proximal parts of limbs and body parts. For the aim of this study, because non-invasive brain imaging cannot detect such small spatial features, as explained in sections 2.3 and 2.4, *Penfield's homunculus* can be considered an appropriate model to deal with the somatotopicity of the motor cortex.

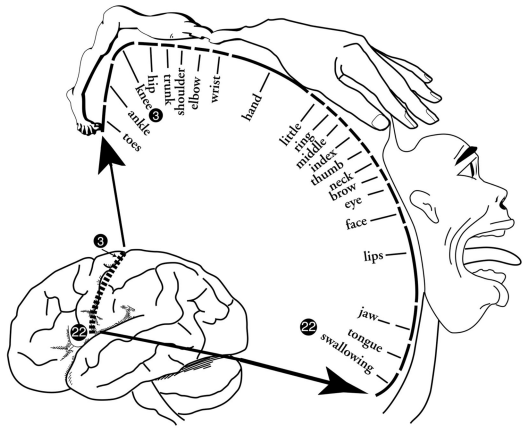


Figure 2.5: *Penfield's homunculus.* The drawing shows how the primary motor cortex (M1) is somatotopically organized, i.e., different motor cortex areas map the movements of different body parts.

Motor control has been widely studied, mainly through spike recordings and neuronal firing rates, which carry the information of when and to what extent a neuron fires action potentials, i.e., communicates with other neural circuits. By investigating the associations and response of neurons in different cortical locations, many aspects of motor control have been understood and the role of different motor areas have been identified. Of course, due to the complexity of brain connectivity and neural circuits, many aspects of motor control are still unknown, but an overview of the basic mechanisms involved in the control of movements can be outlined.

Motor control can be seen as a set of functions that are hierarchically organized and performed in series. The hierarchy concept can be treated having in mind four different dimensions: *time*, *encoding*, *complexity*, and *source* [66].

- The *time* dimension usually refers to the difference between *planning* and *execution* of a movement. Straightforwardly, planning precedes execution on the time scale, and different motor areas are involved: planning mainly takes place in the PMC, while execution is caused by activation in M1. Under the hierarchical point of view, therefore, PMC precedes M1 in the *time* dimension.

- The *encoding* dimension represents the level of abstraction in which a movement is described. For example, a movement can be thought as a target position, or, in a less abstract way, as a set of movement to reach that position, or, in the most concrete view, as a precise sequence of muscular contraction to achieve the different movements to reach that position. M1 neurons encode the kinematics of a movement [12], i.e., the position of the different joints and its derivatives (velocity and acceleration), as well as the direction of the movement (M1 are *directionally tuned*, i.e., their firing rate is higher when a movement is performed in a certain direction [16]). The PMC, instead, appears to be involved more in the target location rather than kinematic and dynamic aspects of the movements [2]. Also in the *encoding* dimension PMC hierarchically precedes M1.

- The *complexity* dimension refers to the differences between simple and complex movements. A complex movement can be seen as a sequence of simpler

movements performed in series, but it is not always the case. Sometimes a complex movement requires a higher level control because an elementary movement could affect the execution of another one. Complex movements, therefore, are thought to be controlled by specialized neural circuits that coordinate and connect many simple movements. One of the main area involved in complex movement execution and coordination is the SMA.

- The *source* dimension indicates the difference between an internal or an external initiation of a movement. For example, a pianist can play reading the music (external initiation) or by heart (internal initiation). Experimental evidence suggest that when the initiation is internal, that is completely volitional, the SMA is the main area involved, whereas when the movement is external, the PMC plays an important role [31]. It is interesting to observe how neurons belonging to M1, which are more involved in the execution of the movement, do not differentiate between internally and externally generated movements, providing support for the fact that M1 succeed both PMC and SMA in the hierarchical organization.

PPC and S1 areas have not been mentioned yet. The PPC contains areas devoted to the visual information processing, and it is involved in associating stimuli coming from different sensory inputs. This activity is strictly related to motor control, for example in the feed-forward mechanisms that adjusts the movements in order to catch a flying object perceived by the visual system. The S1 is important for motor control mainly because of the proprioception system, which consists of muscle spindles and Golgi tendon organs, and informs the CNS about the kinematics (position,

velocity, and acceleration) and dynamics (load and torques) of the different body parts. Proprioception is clearly extremely important in motor control mainly for its feedback.

Motor control, as described so far, appears as a set of articulated mechanisms that can be seen from different perspectives. Nevertheless, the cortical activities which have been treated are not accessible with non-invasive measurements techniques. The next two sections describe the Electroencephalography and Near-infrared Spectroscopy techniques, with particular interest in how the over described motor control mechanisms are projected and related to the measured activity.

2.3 EEG (Electroencephalography)

EEG, as introduced briefly in 2.1, is a medical imaging technique that reads scalp electrical activity generated by brain structures. Electroencephalography is defined as the electrical activity recorded from the scalp surface after being picked up by metal electrodes and conductive media [34]. The electrical activity measured on the scalp is hypothesized to be generated by changes in the membrane potentials of somas of large populations of pyramidal cortical neurons, which can be excitatory (EPSP - Excitatory Post Synaptic Potential) or inhibitory (IPSP - Inhibitory Post Synaptic Potential). The modulation of a post-synaptic neuron body would not be detectable from scalp recording. The oscillations of EEG signals must derive from the synchronous activity of tens of thousands of neurons [36].

EEG signals represent the difference in potential between two electrodes, an active

electrode and the so-called reference electrode. The electrode placement adhere to a standard that was first introduced by Jasper in 1958, the 10-20 system [20]. The name recalls the fact that electrode locations are determined by measuring the distance between two fiduciary points, *nasion* and *inion*, and dividing it in portions of 10% and 20%, as shown in Fig. 2.6. The original 10-20 locations allows the placement of 19 electrodes homogeneously distributed on the scalp. More recently, the international 10-20 configuration has been extended (American Electroencephalography Society, 1991 [52]) to the 10-10 system, which allows a larger number of electrodes, as shown in Fig. 2.6, panel C.

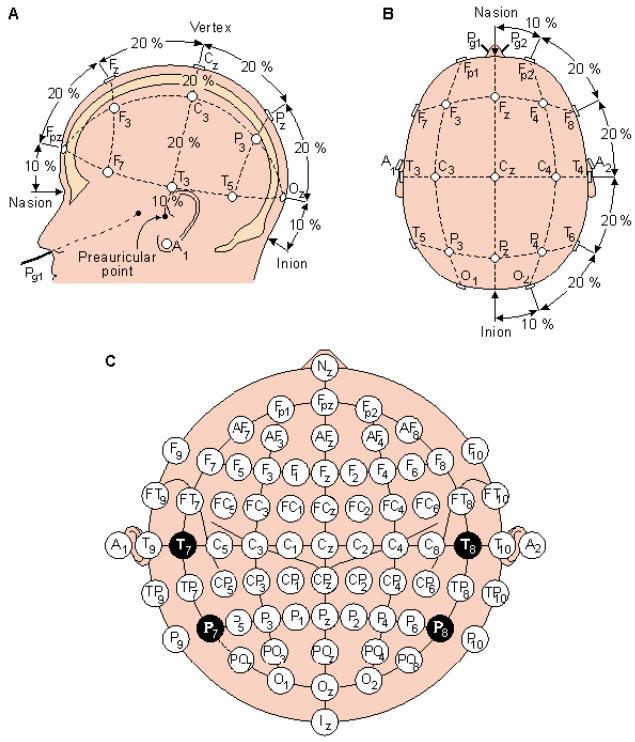


Figure 2.6: A: lateral view of 10-20 system electrode placement; B: top view of 10-20 system electrode placement; C: top view of electrode locations for the extended 10-10 standard configuration.

The "normal" EEG signal is usually composed of different brain waves that can be observed by means of spectral analysis. Some of those rhythms are so prevalent that they can also be observed from raw EEG signals, such as α activity, that can be, in most of cases, easily detected when subjects relax and close their eyes. Brain waves are classified depending on their frequency. Usually, for EEG, 5 different waves can be recognized: δ-band (1-4 Hz), θ-band (4-8 Hz), α-band (8-13 Hz), β-band (13-30 Hz), and γ-band (30-50 Hz).

Another identified rhythm, which is extremely important for sensorimotor modulations, is the μ rhythm. μ rhythm is between 8-12 Hz and it is located mainly over the motor cortex. It was first identified by Chatrian in 1959 [6], who called it *wicket* rhythm and observed that it is desynchronized during movement. The other important rhythm that is recognized as a Sensory Motor Rhythm (SMR), in EEG recordings, is the β rhythm. The blockage of a rhythm due to a motor task is usually addressed as Event Related Desynchronization (ERD), while the reappearance of the activity when the movement is ceased is known as Event Related Synchronization (ERS). SMR are modulated during motor execution, as well as during motor imagery [45]: in particular, prior to the movements, ERDs start in the controlateral areas, and then they spread and become bilaterally symmetrical during the execution of the movement [54]. This symmetry in the brain response to motor tasks makes it very challenging to detect the side of the movements in BCI applications, and it requires complex methods and algorithm to enhance the differences. Moreover, as shown in [17], motor imagery is a skill that takes a while to be learnt (in best cases 1 to 4 hours of practice are needed to achieve an acceptable performance).

Synchronization and desynchronization of EEG signals, measured on the scalp, reflect the activity of a large number of neurons, and they seem very different from the neural activity presented in section 2.2. Due to the volume conduction of the issues between the cortex, i.e., the main source of the activity captured by EEG, and the EEG electrodes, such as cerebro-spinal fluid, skull, and scalp (skin), the activity measured on the scalp is only a small and blurry portion of the underlying current sources generated in the cortex, and in this macro-scale the meaningful information that can be recorded is represented by large neuronal population firing synchronously and resulting in the previously described rhythms (or bands).

2.4 NIRS (Near-infrared Spectroscopy)

NIRS is an imaging technique involving the use of continuous light to non-invasively investigate brain tissue oxygenation. In other words, NIRS measures the changes in the concentration of oxygenated (HbO) and deoxygenated (HbR) hemoglobin. Hemoglobin is one of the most important protein in humans. It is contained in red blood cells and its role is to bind with oxygen molecules and deliver it to the different parts of the body. Oxygen is then used to produce energy, and therefore the measurement of the hemoglobin concentration can be correlated to the metabolic activity. NIRS is a spectroscopy technique, as the name suggests, and it estimates the hemoglobin concentration by lighting the investigated tissue with red and infra-red light and detecting the light intensity after it traversed the tissue.

The principle on which NIRS is based relies on the fact that the hemoglobin

interacts with light in a different way depending on its oxygenation. In a homogeneus medium, the attenuation of light is expressed by the Beer-Lambert Law [64], which states that the attenuation for the i-th chromophore A_i [unitless], defined as the logarithm to the base 10 of the input light, and the output light power (I_o and I), is proportional to the concentration of the absorbing molecule c_i [M], its molar extinction coefficient ϵ_i [$M^{-1}cm^{-1}$], and the optical path length l [cm] (considered linear):

$$A_i = log_{10}(\frac{I_0}{I}) = c_i \epsilon_i l \tag{2.1}$$

The assumptions for applying Beer-Lambert Law, i.e., homogeneous medium and linear optical path, are not satisfied in case of NIRS, because the tissues traversed by light from the source to the detector is highly heterogeneous and the optical path length is not linear but it has a banana shape, as shown in Fig. 2.7a. Therefore, a modification of the Beer-Lambert Law has been introduced, and it is addressed as Modified Beer-Lambert Law (MBLL) [64]. MBLL makes use of two corrective factors, DPF [unitless] that accounts for the increased length traveled by light, and G [unitless], which is a scattering dependent parameter (for further information on tissue optics refer to [64] and [11]). The MBLL, though, only corrects for the non-linear path length, but it still assumes a homogeneous medium. The MBLL, at time t and for wavelength λ, can be written as:

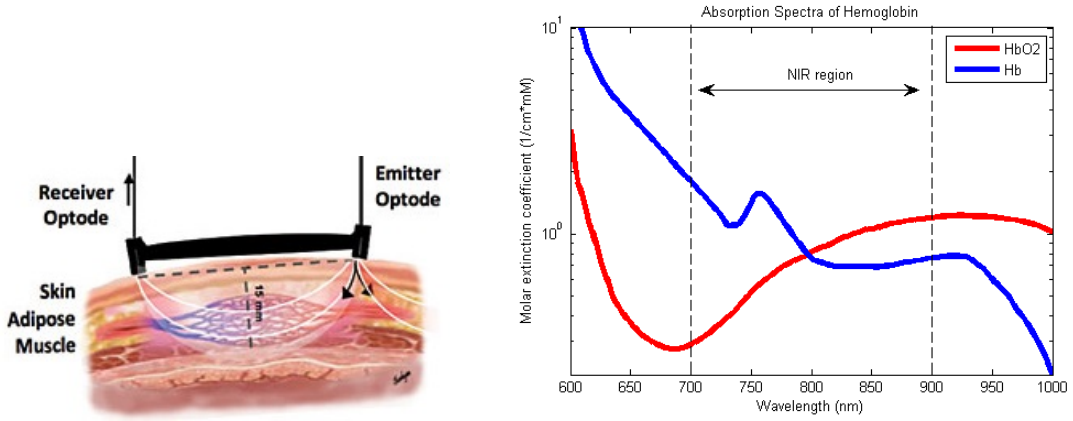
$$A(t,\lambda) = log_{10}(\frac{I_0(t,\lambda)}{I(t,\lambda)}) = \sum_i \epsilon_i(\lambda) c_i(t) DPF(\lambda) d + G(\lambda) \tag{2.2}$$

where ϵ_i and c_i are the wavelength-dependent molar extinction coefficient and concentration for chromophore i (i can be either oxy- or deoxy hemoglobin), and d is the source-detector distance. G can be assumed time-invariant, since the change in scattering can be neglected when compared with changes in absorption. Therefore, when considering the change of attenuation ΔA between the initial time point t_0 and a time point $t_1$the factor $G(\lambda)$ is canceled out [50]:

$$\Delta A(\Delta t, \lambda) = log_{10}(\frac{I(t_0, \lambda)}{I(t_1, \lambda)}) = \sum_i \epsilon_i(\lambda)\Delta c_i DPF(\lambda)d \tag{2.3}$$

where $\Delta c_i = c_i(t_1) - c_i(t_0)$. The non-homogeneity of the tissues traversed by light, though, causes an error in the quantification of the concentration of chromophores. For many applications of NIRS, including the one described in this work, an exact quantification of HbO and HbR concentration is not necessary, but only the trends of the signals are needed.

The last step involved in the estimation of HbO and HbR concentration relies on the fact that the molar extinction coefficient ϵ is wavelength dependent and it is different between HbO and HbR (ϵ_{HbO} and ϵ_{HbR}). As shown in Fig. 2.7b, by sampling the attenuation at two different wavelengths, usually one in the red spectrum (around 700 nm) and the other one in the infra-red spectrum (around 900 nm), a linear system of two equations and two unknowns can be obtained (one equation for each wavelength, $\lambda_1$and λ_2, unknowns are Δc_{HbO} and Δc_{HbR}):



(a) (b)

Figure 2.7: (a) Scheme of the banana shape optical path that light travels from source (Emitter Optode) and detector (Receiver Optode). (b) Molar extinction coefficient in function of light wavelength (ϵ) for oxy- and deoxy- hemoglobin, red and blue line respectively.

$$\begin{bmatrix} \Delta A(\lambda_1) \\ \Delta A(\lambda_2) \end{bmatrix} = d \begin{bmatrix} \epsilon_{HbO}(\lambda_1)DPF(\lambda_1) & \epsilon_{HbR}(\lambda_1)DPF(\lambda_1) \\ \epsilon_{HbO}(\lambda_2)DPF(\lambda_2) & \epsilon_{HbR}(\lambda_2)DPF(\lambda_2) \end{bmatrix} \begin{bmatrix} \Delta c_{HbO} \\ \Delta c_{HbR} \end{bmatrix}$$

and Δc_{HbO} and Δc_{HbR} are computed by solving the system. The solution with respect to Δc_{HbO} and Δc_{HbR} is Eq. 2.4, and it is used to compute the actual estimation of HbO and HbR changes in concentration.

$$\begin{bmatrix} \Delta c_{HbO} \\ \Delta c_{HbR} \end{bmatrix} = d^{-1} \begin{bmatrix} \epsilon_{HbO}(\lambda_1) & \epsilon_{HbR}(\lambda_1) \\ \epsilon_{HbO}(\lambda_2) & \epsilon_{HbR}(\lambda_2) \end{bmatrix}^{-1} \begin{bmatrix} \Delta A(\lambda_1)/DPF(\lambda_1) \\ \Delta A(\lambda_2)/DPF(\lambda_2) \end{bmatrix} \tag{2.4}$$

As far as motor-related changes in NIRS signals, the activation of a motor cortical area certainly results in a hemodynamic change, revealed by the estimation of HbO

and HbR concentration. Nevertheless, the actual contribution of cortical vessels in NIRS signal is a wider and more complex topic, because light, before reaching the cortical blood vessels, passes through other non-cortical vascularized tissues (such as skin and pial veins, located above the surface of the cerebral cortex), which contribute to the detected changes. However, NIRS-fMRI combined studies conducted on motor-related activity, such as [56, 14], showed a high correlation between NIRS signals and blood-oxygenation level-dependent (BOLD) fMRI images, confirming that NIRS can be used with the purpose of detecting changes in the hemodynamic response due to motor areas activation.

2.5 State-of-the-art

This section presents an overview of the past and current research involving SMR-based BCI. Other BCI paradigms exist and are used, but since the BCI development in this project can be considered SMR-based, only the state of the art about this approach is covered. For a wider overview on BCI methods and approaches refer to [68, 60].

This section separately treats EEG-based, NIRS-based, and EEG-NIRS-based BCI. EEG is the oldest measuring technique for BCI and the literature about it is huge. In the following subsection the review is particularly focused on the methods used to process and classify the signals.

2.5.1 EEG-based BCI

EEG is the most common and well established approach to BCI. Many different studies have been conducted throughout the last decades, and currently EEG seems to be the most suitable uni-modal approach to develop high performance BCI due to its high temporal resolution. The low spatial resolution problem is tackled by means of different techniques that allow us to increase the Signal-to-Noise Ratio (SNR) of the signals, managing to achieve high accuracy and performances. Nevertheless, several variables must be considered to describe a BCI. The first great difference among BCI studies is about the paradigm involved in the motor tasks, which can be either motor execution or motor imagery. In the latter case, as explained in section 2.3, the experience of the subjects should also be taken into account. Other factors are the number of classes involved in the classification (binary or multi-class), the number of trials per class that are used for training, the experimental design, the signal processing involved, the classification methods, and the performance of the system, which can be expressed as accuracy or Information Transfer Rate. Since a detailed and comprehensive review including all this aspects would be too wide and inadequate for the purpose of this work, here particular focus is given to the signal processing methods applied to separate the different classes. It should be emphasized that signal processing and classification methods are greatly correlated, in the sense that their choice is not independent, but often a certain analysis method calls for a specific classification approach. Due to the previously described ERD/ERS and the somatotopical distribution of motor related activity, most of the methods presented in various ways extract features that represent the frequency, time-frequency, and

spatial contents of EEG signals from different channels.

The first approach describes the frequency-content of each measurement channel with auto-regressive (AR) estimates of signal spectra. AR estimate is a parametric method to compute the power spectrum of a signal and to represent it with a small number of coefficients, which are then used to extract features to classify the signals as in [42]. A more sophisticated approach derived by AR, implemented in [46] and [43], involves the adaptive estimation of AR coefficients (AAR) by means of Kalman filter.

Another approach used for EEG processing is wavelets. The Wavelet Transform (WT) projects the signal on a family of frequency bands either continuous (CWT) or discrete (DWT). WT allows to represent the signal in a time-frequency representation, which is clearly beneficial in the detection of ERD/ERS patterns. Wavelet approaches can vary regarding feature extraction, selection, and classification methods, as shown in [18], [65], and [70].

The third most common method to classify EEG signals related to different motor tasks makes use of Common Spatial Patterns (CSPs). CSPs are used in this project with a regularization approach and are described in detail in the Methods chapter, section 3.2.2. Briefly, CSPs act overall in on the spatial content of the signals, optimizing linear spatial filters to maximize the separability between classes. CSPs have been widely used in SMR-based BCI, for example in [30], [44], and [5]. The latter work presents an exhaustive and complete description and review of the method.

Every methodology over described has advantages and disadvantages, and it

should usually be considered the general framework of the study, including all the variables named at the beginning of this section, in order to fully appreciate and evaluate each of the methods. Whereas for EEG the literature is rich and full of well established and interesting approaches, for NIRS-based BCI the state-of-the-art is much more circumscribed, due to the NIRS late appearance as communication system for BCIs. The following subsection reviews the studies performed in NIRS-based BCI involving motor-related tasks.

2.5.2 NIRS-based BCI

The first study investigating the use of NIRS for BCI was made by Coyle et al. in 2004 [7], followed by another one in 2007 by the same authors [8]. The two studies are very similar regarding the instrumentation and the experimental procedure. In particular, in [7] a single NIRS channel centered on C3 position of the EEG 10-20 system (see Fig. 2.6) was used to detect hand motor imagery, while in [8], two channels were placed in correspondence of C3 and C4 with 3 healthy subjects involved. Online feedback based on previous hand-gripping execution tasks was given to the subjects to enhance the performance. The *Mindswitch* paradigm allowed the user to select or not a particular goal (on-off). NIRS signals, in particular HbO, showed a peak in the response around 5-8 s, making the response quite slow for BCI purposes, despite the overall good accuracy (75% on average in [7] and around 80% in [8]). These two papers represented an important incentive in the consideration of NIRS for BCIs, but they certainly had the limitation of performing only a binary *on-off* classification between rest and an undefined motor imagery task.

In 2007, Sitaram et al. [53], studied the use of NIRS for the classification of right-left motor imagery on 5 healthy subjects. Using 20 measurement channels, they performed an offline analysis, in which a feature vector was extracted from every 10 s trial, and they managed to classify right and left motor imagery with an average accuracy of 73%, using Support Vector Machine (SVM), and 89% using Hidden Markov Model (HMM). Clearly, since the construction of the feature vector was done using an 8 s interval after the stimulus onset, the main drawback of this approach would be its long delay in a real-time performance, for which the time factor is extremely important. On the other hand, they managed to recognize with particularly high accuracy the laterality of the motor imagery.

Kanoh et al. in 2009 [22] investigated the changes in mental strategies involved right-hand motor imagery sessions performed in 5 consecutive days. They used a feedback based on the average concentration of HbO on 3 measurement channels (around C3, Cz, and C4), and they showed how the feedback becomes more robust as the three subjects used the online system. They also observed that the spatial distribution of HbO concentration (visualized with 52 channels) presents differences before and after training. Despite the interesting conclusions, the article cannot be considered as a comprehensive BCI study, but it focused on the importance of the feedback with a NIRS-based approach. Moreover, also in this case the paradigm only involved either rest or right-hand motor imagery, i.e., an *on-off* approach.

More recently (in 2013), Naseer et al. [32] performed a study about motor imagery of right and left wrist flexion. 10 healthy subjects had to imagine the execution of the movements after a preparatory training session. From the 17 channels, a single

feature set was extracted for every trial. Features consisted of the average of HbO and HbR values over the task period and the slopes of the two sets of signals, computed as the coefficients of the line fitting the data. The accuracies obtained were 78% using average features, and 87% using slope ones. The main issue of this approach, in spite of the high classification performance, is its inapplicability in real-time systems, since the feature vector is computed using the entire task signals.

As shown in this brief NIRS-based BCI review, the main concern about the use of NIRS signals is the delay of the hemodynamic response, which makes the online performance quite low in terms of Information Transfer Rate.

2.5.3 EEG-NIRS hybrid BCI

A multi-modal EEG-NIRS approach for BCI applications have been minimally investigated in the literature of the BCI field. Both the systems, though, have issues of diverse nature, such as spatial resolution for EEG and temporal resolution for NIRS: the combination of them, therefore, can no doubt improve and enhance the robustness and reliability of the application and have been studied in different ways. Surprisingly enough, only three articles of an hybrid EEG-NIRS approach for BCI have been found in the literature, probably due to the technical difficulty in the simultaneous and non-standardized setup of the two imaging systems together.

Khan et al. in 2014 [23] implemented a system capable of recognizing 4 different states. The EEG was used to detect motor related activity changes to control *right* and *left* commands, while NIRS signals from the prefrontal cortex (PFC) allowed

to choose between *forward* and *backward* by means of arithmetic mental tasks. In 2014 as well, Tomita et al. [58] studied the possibility of integrating NIRS with EEG in a steady-state visual evoked potentials (SSVEPs) paradigm, which is a complete different approach from the one used in this work and therefore will not be discussed.

The only work investigating whether NIRS signals can enhance the performance in SMR-based BCI is by Fazli et al. in 2012 [13]. This study is of particular interest for the current work, because it represents the first (and only) attempt to combine EEG and NIRS for motor classification. The purpose of Fazli's work was to investigate the performance of EEG- and NIRS-based classifiers on Right-hand - Left-hand motor execution and imagery tasks. In particular, the experimental setup consisted of 37 EEG electrodes, distributed along the whole head, and 24 NIRS optodes (12 sources and 12 detectors) arranged in 24 measurement channels. The experimental design included 48 trials per class for motor execution and 100 trials per class for motor imagery. It is important to specify that motor imagery tasks were divided in two blocks: during the first run a subject-independent classifier gave feedback to the subjects. During the second run a subject-specific classifier, estimated from the data of the first block, was used to improve the effectiveness of the feedback (this approach has been recently developed in [63] and it is called *Co-adaptive calibration*). The offline performance of the data involved the use of subject-dependent temporal filters and CSP for the EEG, while for the NIRS the baseline of each trial was subtracted (this kind of processing is unlikely applicable online for an asynchronous BCI) and the average of HbO and HbR served as features. Features were computed in time segments of 1 s with 50% overlap between each other. The results showed

that the improvement in accuracy when EEG and NIRS features are combined using a feature selection method is significant. The accuracy was computed dynamically, obtaining an accuracy signal over the trial (see section 3.4 for a detailed explanation of the dynamic accuracy computation). The accuracy peaks obtained were 93.2% for motor execution (when combining EEG and HbR derived features) and 83.2% for motor imagery (with EEG-HbO combination). The authors showed that EEG and NIRS provide information which is complementary in terms of mutual information. On the other hand, as shown in Fig. 2.8 in the second and third rows, the trend of the accuracy over the trials for NIRS signals is very slow, reaching a peak in accuracy after 6.5-7.5 s from the stimulus onset.

With the work just described in mind, it is important to emphasize the differences and developments proposed in this study. First of all, the current study has been performed with a remarkably shorter training time (in [13] the training lasted around 4 hours): this allows us to evaluate the capability of the system when only a small dataset is available (only 25 trials per class for both execution and imagery). The second main difference, which can be considered an improvement, regards the number of classes: whereas a binary classification was performed by Fazli et al., in this study 4 different motor tasks are involved, namely, right-arm, left-arm, right-hand, and left-hand. Moreover, here it has been tried to develop a compete asynchronous BCI, which needs the online detection of rest or task before performing any further classification. Under the setup point of view, the EEG electrodes density is lighter, with 21 active electrodes around the motor cortex only (see Fig. 3.5). For the NIRS the same number of optodes is used, i.e., 24, but they are arranged in 34 measurement

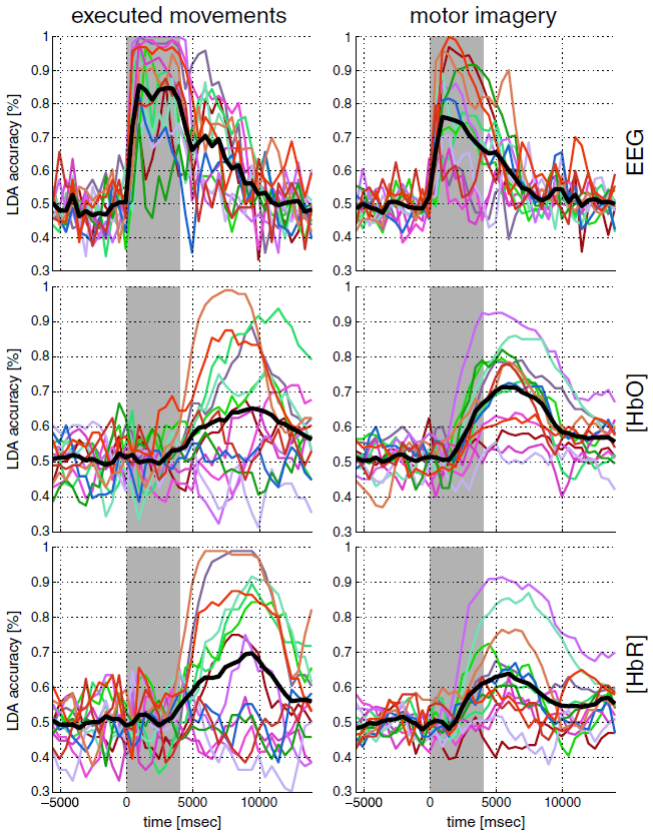


Figure 2.8: Figure from [13] that shows the trends of the accuracy of the different subjects (colored lines) and their average (black thick line). The first column represents executed movements, while the second one shows motor imagery performance. The first row reflects the accuracy of EEG-based classifier, the second one of HbO-based ones, and the third one of HbR-based ones.

channels. For the experimental procedure, such an advanced feedback system was not available, and an EEG-based online feedback was implemented based on SMR, telling the subject whether rest or task was detected. The signal processing of the EEG data is similar, apart from the use of regularization techniques for CSP, while for processing the NIRS data, in order to limit the delay observed in the accuracy trend, new methods were investigated, including CSP and slope-based features. Particular

attention should be given to the fact that both EEG and NIRS signals analysis, although performed offline, were treated with methods easily implementable in an online system.

Chapter 3

Methods

The following chapter will explain in details all the materials, instrumentation, and methods used in this work in order to achieve the aim of developing the multi-class Brain-Computer Interface. The first section (3.1) deals with the data acquisition, from the instrumentation to the protocol definition and experimental setup. Particular emphasis is given to the explanation of the real-time EEG-based feedback C++ software. Section 3.2 describes how the raw signals are processed to extract informative features, while section 3.4 treats the classification and validation aspect. Section 3.5 explains what kind of hypothesis tests are performed in order to find differences among tasks (e.g., motor execution and imagery), and processing techniques. Finally, section 3.6, deals witht the design of an online setup to evaluate the BCI in real-time.

3.1 Data Acquisition

3.1.1 Instrumentation

The following section will describe all the hardware and software used. All software were running on a Windows 7 workstation (WS) for recording and visualization, and on a Windows 7 laptop (LT) for presentation and feedback purposes.

EEG : The EEG system used for this project was microEEG from BioSignal Group [1]. The microEEG System is capable of recording 26 channels, it is easy to set up, it is portable, battery powered, and connects wireless via Bluetooth to the controller software, namely, microEEGCtrl, running on the WS. Despite its small size and ease of use, the system performs as standard clinical EEG systems: the only small difference, mainly due to the concentrated electronics, is in the lower amount of 60 Hz power line [37], which did not affect the current study because all EEG signals were filtered offline and online with a low-pass frequency cutoff of 25 Hz (within β band). As far as recording specifications, the Analog-to-Digital Converter (ADC) resolution is 16 bits, the input impedance is greater than 100 MΩ, and the Common-Mode-Rejection-Ratio (CMRR) of the operational amplifier is 85 dB. The compact microEEG system (weight: 88 g, size: $9.4 \times 4.4 \times 3.8$ cm) is shown in Fig. 3.1. The controller software, microEEGCtrl, allows us to search and connect to the detected device (the device needed to be previously paired to the WS). The two main uses of microEEGCtrl were the resistance check and the signal visualization, which allowed us to assess (from an engineer) the EEG quality. The resistance threshold for which

Figure 3.1: microEEG System.

an electrode was considered *good* was set to 20 kΩ, but sometimes even higher resistances have been accepted if a lower resistance could not be reached, after assessing the time course of the EEG signal. The visualization of the signals have been also extremely useful for performing basic EEG tests, such as α-wave test and eye blinking, and to assess whether there was a major interference with the NIRS lighting system and the passive EEG electrodes. The electrodes were standard Ag/AgCl ones, and in order to reduce the resistance between the electrode itself and the scalp, a conductive gel was injected inside the hole of the electrode holder with a syringe. The electrode holder is shown in Fig. 3.3a.

NIRS : The NIRS acquisition system was the NIRScout, by NIRx [35], in the extended configuration, that is equipped with 128 LED sources and 64 fiber-optic-wired detector (Si photodiode, sensitivity $<$ 1 pW) for a maximum of 4096 measurement channel (Fig. 3.3a shows the system optodes). The LED light wavelengths used for oxy- and deoxy- hemoglobin estimation were 760 nm (red) and 850 nm (infra-red). The sources and detectors can be arranged in different ways, and every source can

build up a measurement channel if the emitted light is acquired by one or more detectors. The sampling frequency can range between 2.5 and 62.5 Hz. The sampling frequency depends on the number of sources, detectors, and channels and can be increased by changing the lighting pattern on the recording software (NIRStar, running on the WS). In fact, the system has the capabilty of lighting up more than one source at the same time, resulting in a faster sampling rate. On the other hand, the *simultaneous* channels should be far enough apart to avoid cross-talk between different channels [35]. In the present work the latter capability has been used and *simultaneous* channels were always at least at 10 cm distance, assuring the absence of cross-talk and allowing a relative high sampling frequency of 10.42 Hz (12 sources, 12 detectors, 34 channels). Another important feature of the NIRScout system is the possibility of receiving and acquiring trigger/event signals through a 8 bit TTL parallel port (up to $2^8 - 1$ possible codes). This capability has been used to keep log of the experiment events and to synchronize NIRS and EEG signals. The entire NIRScout system, along with the running NIRStar software, is represented in Fig. 3.2.

CAP : The cap adopted for the equipment of EEG electrodes and NIRS probes was actiCAP 128, produced by BrainProducts. The cap is easy to wear, relatively comfortable, as confirmed by the experiment subjects, and it takes only some seconds to put it on the subject's head and fasten the chin belt. As the name suggests the cap is capable of holding up to 128 electrodes, but in this study we integrated the NIRScout holders inside the cap holes as well as EEG electrodes for multi-modal imaging. In order to maintain the proper distance between NIRS sources

Figure 3.2: NIRScout instrumentation and workstation rinning NIRStar software.

and detectors (3 cm, as specified in manual) plastic holders provided by NIRScout were used. To keep the relative position between NIRS probes and EEG electrodes, custom made laser-cut plastic holders were applied, consisting of a bigger circular hole of 7 mm radius for the electrode and a smaller 3.5 mm radius hole for the NIRS probe holder. Since, due to the disposition of the cap holes, electrodes and optodes happened to be at different distances, the center of the plastic holder holes ranged from 2.25 cm to 2.6 cm. Fig. 3.3b shows the internal part of the cap, with the NIRS and EEG-NIRS holders, while Fig. 3.5b shows the cap completely set up.

PRESENTATION : The software for presenting and guiding the subject through the experiment was Presentation, from NeuroBehaviouralSystems, running on the LT. The software is designed for precise stimulus delivery and accurate event logging, it can interface external hardware devices for input/output communication, it is

(a) (b)

Figure 3.3: (a) Sensors, from left to right: NIRS LED, NIRS Si photodiode, EEG Ag/AgCl passive electrode. (b) EEG-NIRS laser-cut holders (pointed by blue arrows) and NIRScout holders (ocra arrow) between NIRS sensor plastic spacers.

highly flexible and programmable. The programming language is a high level C-like object-oriented language, and it allows the programmer to create and precisely deliver stimuli of different nature (visual, and auditory), to include them in trials with controlled timing, and to flexibly decide the order the trials succeed each other and interact with external input/output. In particular the LT was equipped with a standard 25-pin TTL parallel port, which was used to send different event codes to the NIRS and EEG system, while a USB port was used as an input port to receive EEG-based real time feedback (explained in detail in section 3.1.3).

SYNCHRONIZATION : Since the acquisition systems were not the same for EEG and NIRS signals, they needed to be synchronized for offline analysis (conversely, for real-time applications, synchronization is not needed for the data flows are exactly simultaneous). The LT parallel port was directly interfaced with the NIRScout TTL input/output connector by means of a standard parallel cable. For

the EEG it has been observed that a direct interface with the microEEG event channel resulted in a highly noisy signal, full of spiky artifact due to the fast transition of the digital signals. In order to avoid this problem, an opto-isolator (Motorola 4N26) was used to uncouple the parallel port digital signal and the microEEG event channel. Only 1 pin out of the 8 data pins of the parallel port was opto-coupled to the microEEG, so the recognition of the event code was done using the NIRScout input trigger log rather than the microEEG event channel signal. For synchronization, at the beginning of the experiment, one single 255 code was sent by Presentation software and captured by both the NIRScout system and the microEEG event channel (as a high-amplitude spike, over 1-2 V).

3.1.2 Experimental Setup

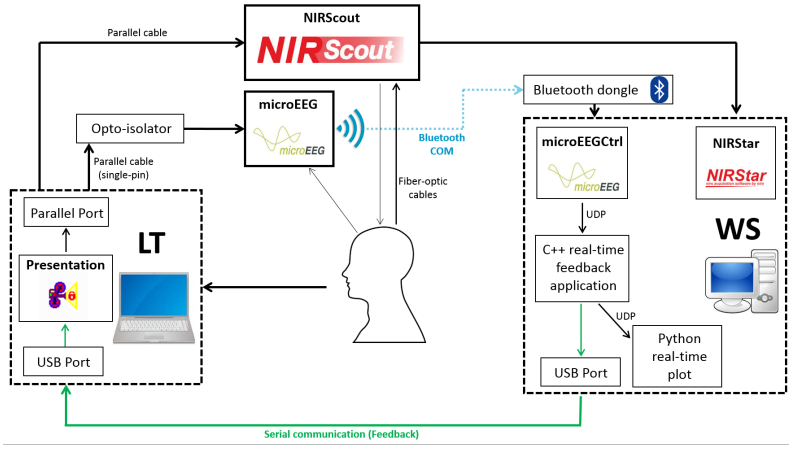


Figure 3.4: Experimental setup block diagram.

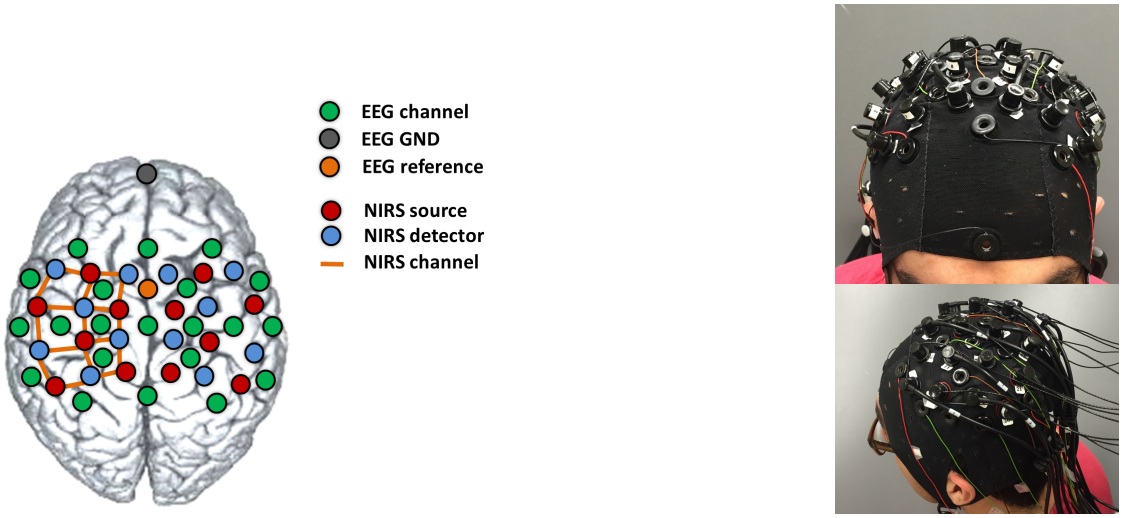
Fig. 3.4 shows how the different systems described in section 3.1.1 are connected and work with each other. The subject interacts with the system by looking at the

screen of the laptop on which Presentation is running. Presentation keeps log of the events of the experiment by sending triggers of different codes depending on the experiment phase through the parallel port. The latter is connected to the parallel event/trigger input port of NIRScout, and a single pin of the port is sent to the opto-isolator and recorded by the microEEG event channel. The NIRScout LEDs are wired through normal twisted cables, while the detectors are fiber-optic wired. Due to the nature of the LED wires, particular precaution must be taken during the disposition of EEG electrodes and NIRS LED probes. In fact, if the electrode and LED wires are too close to each other, a strong interference is found in the EEG signals and can be often detected just by inspecting the EEG signal. In order to avoid this issue, when all the probes are in place and the NIRS channels have been calibrated, the source wires are pulled up and hung on supports, so that the distance between the electrode wires and them is increased (as shown in Fig. 3.5b). The EEG is recorded by microEEG, which sends the data stream in real-time to the WS (which is paired to the device through a Bluetooth dongle). Through microEEGCtrl application the EEG signals can be visualized in real-time and streamed by third-party application. In the current setup, the EEG data were broadcast by a custom-developed C++ application (programmed in Microsoft Visual Studio Express 2013), that performed the online normalization, $\mu$and β band filtering, and average power computation (an extensive explanation of the application can be found in 3.1.3). The EEG band-powers averaged over custom chosen EEG channels were sent via UDP-IP local connection to a custom Python application designed for real-time plotting of the signal. The experiment consisted of a motor execution and a motor imagery part:

the green arrows in Fig. 3.4 were active only during the motor imagery block, with the purpose of giving real-time feedback to the subject performing the experiment. In particular, one single signal of μ or β EEG band-powers computed by the C++ application and averaged over significant channels chosen by the operators (usually from the ones in correspondence of the motor and/or parietal cortex) was sent back to the LT and read by Presentation via USB connection. For the USB host-to-host connection a normal USB cable could not be used, for it can interface only a host (a computer) with a device (such as a printer). To connect two computers a particular integrated circuit (IC), namely, a FTDI, has to be serially connected to both the USB ports. The FTDI circuit usually comes integrated in a cable and it provides connectivity between a USB and a serial UART (Universal Asynchronous Receive Transmit) port. The interface of an FTDI cable has a USB connector at one edge and Tx (Transmit), Rx (receive), Vcc, and GND pins at the other. The communication between the two computers is achieved by connecting the Tx pin of one FTDI cable to the Rx pin of the other, and vice versa.

The cap was equipped with 23 EEG electrodes (21 recording channel plus REF and GND), 12 NIRS sources, and 12 NIRS detectors combined in 34 measurement channels. As explained in the Introduction (section 1.2), one of the main aim of the entire project has been the investigation of a BCI system whose measurement probes are distributed only in a restricted area throughout the motor cortex. For this reason, EEG electrodes lay all between the F and the P parallel of the 10-20 system, apart from GND that was placed in Fpz. As shown in Fig. 3.5a, the list of EEG electrodes, from left to right and from top to bottom was: F3, Fz, F4, Fc5, Fc1, Fcz (REF),

Fc2, Fc6, C5, C3, C1, Cz, C2, C6, T4, P3, Pz, and P4. It is important to address the fact that the placement of the reference in Fpz does not affect the main part of the EEG analysis because of the use of spatial filter, which result in a change in the reference of the signals. Although the GND electrode was not placed within F and P parallels, as that would violate the design for which all the electrode and probes were within the F and P parallel, its location was chosen for its ease to setup and to get good conductivity, being placed on the forehead. Actually the GND, since all the measurement are bipolar, is used only to subtract differential voltage to measuring and reference electrodes, and it is considered as common mode voltage when it comes to differential amplifiers. Given that, the GND electrode could be placed, in theory, everywhere on the body. The NIRS probes were organized in a 4 by 3 grid for each of the two emispheres as shown in Fig. 3.5a. Sources and detectors were alternating to allow to pair the same source with multiple detectors. Each source was paired with a minimum of two detectors for corner sources, and a maximum of 4 detectors for internal ones. NIRS channels have been labeled according to their relative position (Anterior = A, Posterior = P, Medial = M, and Lateral = L) with respect to 10-20 system EEG location: for example the channel anterior to C3 was named C3A. With this in mind, the list of the channels from left to right and from top to bottom was: Fc3A, Fc1A, Fc2A, Fc3A, Fc3L, Fc3M, Fc1M, Fc2M, Fc4M, Fc4L, C3A, C1A, C2A, C4A, C3L, C3M, C1M, C2M, C4M, C4L, Cp3A, Cp1A, Cp2A, Cp4A, Cp3L, Cp3M, Cp1M, Cp2M, Cp4M, Cp4L, Cp3P, Cp1P, Cp2P, and Cp4P. The labeling of the NIRS channel has demonstrated to be extremely straightforward and useful in the signal visualization and processing.



(a) (b)

Figure 3.5: (a) EEG electrodes and NIRS optodes configuration on the cap. (b) Real picture of a subject wearing the cap completely mounted (with EEG electrodes, NIRS sources and detectors).

The following section explains the experimental procedure for the data collection.

3.1.3 Experimental Procedure

For the study, 15 healthy subject participated in the experiments. The subject were all male, aged between 22 and 54 (only one subject was above 30 years old). An important factor that needs to be taken into consideration is the fact that the subject never experienced motor imagery before: this topic will be discussed in more detail in section 5.3. The use of the NIRS technique forced the choice of having only male subjects. In fact, the hair represent the most important issue in the NIRS setup, preventing the sources and detectors to be in direct contact with the scalp and yielding bad quality signals. The subject were seated on a comfortable chair

during the entire duration of the experiment, facing the LT screen at a distance of approximately 1 m. The subject preparation of the subject include the following steps:

- measurement of the *nasion-inion* distance
- cap placement in the correct position with Cz at a half of the *nasion-inion* distance
- conductive gel application as shown in [26] and resistance check through microEEGCtrl
- optical gel application to all NIRS probes, moving aside the hair in order to allow the probes to couple directly with the scalp
- NIRS calibration with NIRStar and adjustments to probes
- EEG-NIRS interference check

Although the subject have been selected basing on the amount and thickness of hair, it must be said that for some of them (at least 4) the NIRS setup was far from perfect. When the setup time exceeded one entire hour (in some cases even some more) it was decided to settle and accept the lower quality of some of the channels. Often, though, noise was present only in relatively high frequencies, and it has been filtered out using the pre-processing described in section 3.2.1. For further processing techniques, in particular for the regularization of the Common Spatial Patterns (section 3.2.2), the entire set of channels was needed for all the subjects: no channels, thus, could be discarded from the analysis.

Protocol : A single experiment consisted of three separate parts: the motor execution part, the motor imagery "practice", and the actual motor imagery part. The entire experiment, not including the setup, lasted a maximum one and half hours.

The first part was the motor execution one; the subject was instructed to perform 4 different upper limb tasks, namely, right- and left- hand-gripping and right- and left- arm-raising. The subject was instructed to repeat continuously every movement for at approximately 1 Hz during the *task* phase of every trial. The experiment was guided by the screen of the LT, on which visual stimuli were presented. The stimuli, for this part, were images with a black background and with a big colored text in the middle. The texts presented for both the motor execution and motor imagery part were: *REST*, *READY*, *RIGHT ARM*, *LEFT ARM*, *RIGHT HAND*, and *LEFT HAND*. In order to ease the subject reaction when presented a task, the 4 different movement had different colors: red for right arm, blue for left arm, magenta for right hand, and cyan for left hand. The motor execution part, as well as the motor imagery one, consisted of 100 trials divided in 5 blocks of 25 blocks each. Trials in each block were presented in a random order, but the balance of the tasks was preserved for each block, so that in every block the subject was asked to perform each movement 5 times. Every trial started with the *REST* stimulus for approximately 5 s, followed by the *READY* one for 1 s and one of the 4 task stimuli for 6 s. After every block the subject could rest and relax for some seconds or start back with the next block at his pleasure. The motor execution part, thus, lasted a couple of minutes more than 20 minutes, usually between 22 and 25 minutes.

During the motor execution part, the C++ application running on the WS estimated online the EEG filtered signals and output (μ and β band-powers of every channel) parameters to perform a normalization of the feedback. This step was necessary to know *a-priori* the range of the output signal and to provide a significant feedback. The detailed explanation of the feedback application can be found in the next paragraph. The feedback consisted of a big circle at the center of the LT screen with a color code: red meant that a resting state was detected, while green, on the contrary, stood for motor imagery detection. The color ranged from the red to green passing to yellow and it could be every hue of the RGB model. The motor imagery "practice" was designed to allow the subject to familiarize with the feedback and to find the best strategy for motor imagery. This part of the experiment lasted from 5 to 20 minutes, depending on how comfortable the subject felt in using the feedback.

The third and last part of the experiment was the motor imagery part. The only differences between the motor imagery part and the motor execution part were the presence of the feedback and the timing. The organization and division of blocks and trials were exactly the same. Feedback was given to the subject only during the task because during the resting phase it could have made him focus more on the color of the circle than on the relaxation itself. The timings were longer than the motor execution part, because it has been observed that subject found more difficult to stop imagining the movement instantaneously: each trial started with the *REST* stimulus for approximately 8 s, followed by the *READY* one for 1 s and one of the 4 task stimuli with the feedback circle for 7 s. The motor imagery part, thus, lasted around 27 min plus the inter-block breaks.

Real-time Feedback Application : The C++ application used to compute the real-time feedback was developed with a highly object-oriented paradigm. This yielded an organized and thoughtful design, in which many different objects with different purposes strictly interact with each other in order to achieve the final purpose of the program. In order to give an overview of the program, every class is briefly described by its task and how it *talks* with other classes.

Main The main, which is not actually a C++ class, first of all opens the COM interface to interact with the microEEGCtrl application. Only if a microEEG is connected and the streaming status is *RECEIVING*, information about the number and the name of channels are their names are retrieved from microEEGCtrl and a UDP port is set for the data broadcast. The user is then prompted for: the COM port on which the USB transmission is going to be, whether online normalization has to be performed or parameters have to be loaded from file, whether the feedback should be μ or β power band, and which channels have to be averaged for the feedback. After that, a Controller object is created with all these parameters and the *Start()* member function of the Controller object created is called.

Controller The Controller class only contains a StreamingThread and a ProcessingThread member variables. When the *Start()* method is invoked the two StreamingThread and ProcessingThread objects "join" their threads. The threads start and keep running in parallel until the application is stopped. For the thread classes, *boost* libraries have been used (please refer to www.boost.org site for further information)

StreamingThread The StreamingThread class, as the name suggests, contains the actual thread. The thread is assigned to a the *Stream()* member function. This class contains a MicroEEGController class, which is designed for the low-level streaming of the data. The StreamingThread class has two important member variables: *tw_reached*, where tw stands for time window, and *stopped*, which are both boolean. Whereas the *stopped* variable task is straightforward (it is set to *true* when the user decides to stop the application by pressing any key), the role of the *tw_reached* flag needs more explanation. This flag is set to *false* from the class constructor, and the streaming of the data is done only if this flag is *false*. The actual timing is given by the *receiveEEG()* method of the MicroEEGConnector class, that, when a time window span is reached, returns a *true* boolean; the return value of the MicroEEGConnector *receiveEEG()* function is assigned to *tw_reached*. When *tw_reached* is *true*, the streaming stops for a moment to allow other classes (in particular the ProcessingThread one) to copy the raw data, and it is then reset to *false* once the data are copied. This mechanism assures that the raw data are always consistent and increases the robustness of the program.

ProcessingThread The ProcessingThread class contains a thread that is assigned to the *Process()* function. This class contains, as member variables, a StreamingThread, an EEGHandler, an USBController, and an UDPClient object, among others. The StreamingThread member variable is necessary, because it *registers* the ProcessingThread with the StreamingThread object that is streaming the data. The *Process()* function, which is the worker of the class,

keeps polling (checking) the status of the *tw_ reached* flag of the StreamingThread variable. When it is found to be *true*, it copies the new raw EEG data, resets *tw_ reached* to *false* (so that the streaming resumes), and calls the *HandleEEG()* method of the EEGHandler class, which does all the low-level processing and returns the normalized band-powers for $\mu$and β of all the channels. The feedback value is then computed with the μ or β powers by averaging over the user-chosen channels. The feedback value is then sent via USB by means of the USBController object, while both the μ and β *grand averages* (user-chosen) and the average over C3 and C4 are sent via UDP to the Python application through the UDPClient object, for the real-time plot. C3 and C4 were chosen to monitor the motor cortex only.

MicroEEGConnector This class is the one that interface directly with microEEGCtrl to stream the data via UDP. The streaming is done by means of a socket using *WinSock* library from Windows. It is the *receiveEEG()* that gives the timing to the rest of the program, and, as anticipated before, it returns a boolean which is *true* if a time window has been reached; otherwise it returns a *false*. The timing is managed using the Queue class. When a new datagram is received, it is decoded, i.e., *translated* from a packet of bytes into meaningful μV values, and the raw values of every channel are *enququed* in a Queue object, after *dequeuing* the oldest values. By setting the length of the Queue as the number of EEG samples in a time window (in this case, since the EEG is sampled at 250 Hz and the time window is 1 s, every Queue has a length of 250), it is assured that in every Queue there is a the last 1 s of each EEG channel. The

overlap between consecutive time windows is managed as follows: the number of samples that corresponds to the not overlapped values is computed, e.g., if the overlap is 90% (as in this case), the number of new values to be added to the Queues will be 0.1 x 250=25, and a counter is increased every time a new UDP datagram is received. When the counter is equal to the number of non-overlapped samples, it means that a time window is reached. The counter is then set to 0, and the Queues values are returned as a reference to a vector of vectors (the Queue is parsed to a vector). The returned matrix has size of the number of channels by the number of samples per time window and it represents the raw EEG data that will be processed.

EEGHandler This is the class that performs all the EEG processing. The main member function is *HandleEEG()* and it takes as input the raw EEG matrix and returns two different vectors, one containing the μ band-powers and the other with the β band-powers of every channel. Since the computation of the FIR (Finite Impulse Response) filtered signal is very time consuming due to the convolution operator, every channel is processed using a separate thread, making the processing massively parallel and less time consuming. Another hint used to diminish the time complexity is to filter only the new values, i.e., the non-overlapped ones, and, after having processed them, to shift all the resulting vectors backwards to make room for the values of the next time window. With this strategy, only 25 samples over 250 (the non-overlapped 10%) undergo the convolution operator with the filter coefficients. Another comment is needed for the normalization and saturation step; in order to perform online

normalization, mean and standard deviation of the signals and of the output values are computed as shown in Appendix A. For normalization, Eq. 3.1 in section 3.2.1 has been used. The $\log_2$ transform of the output signals has the purpose of making the output distribution more normal around the 0, which results in a more balanced feedback. The saturation stage has the role of limiting the effect of EEG artifacts of non-cerebral origin, such as EMG activity, talking, or yawning. This said, the processing consists of:

1. Band-pass FIR filtering (using MATLAB-designed filters)
2. Saturation of the signals at ± 15 μV
3. Normalization of the μ and β filtered signals
4. Low-pass filtering (moving average over 250 samples) of the absolute value of the signals (envelope)
5. Average over the time window
6. Normalization of the output values
7. $\log_2$ transform of the output values (to adjust for right skewness)
8. Saturation of the output signals at ± 3

USBConnector This class is used to send data via the USB port. It uses *boost* serial port classes, creating a serial_port object with the port defined by the

user. The writing member function, *USBWrite()*, takes as input a single unsigned char and writes it on the specified serial port using *boost write_some* method. When the user terminates the program the port is closed.

UDPClient This class is used to send out data to the Python application for the real-time plot. As the MicroEEGConnector class, the UDP connection is performed using *WinSock* libraries. In particular, the member function *sendData()* allows to send out a buffer of char of any length. The values to be sent via UDP, thus, must be parsed to a char array before they are sent.

The application has the feature of allowing the estimation the normalization parameters beforehand and to load them in following sessions. This capability has been added because, due to how the parameters are computed (see Appendix A), some time is needed in order to obtain significant output values (usually at least 5-7 minutes). In the current experimental procedure, thus, the parameters were estimated during the motor execution part and then loaded for the motor imagery part. This strategy made the feedback effective and well-normalized as soon as the subject started the "practice" session.

3.2 Signal Processing

3.2.1 Pre-processing

The pre-processing of the signals, described in this section, includes all the operations and transforms which are used in order to obtain signals on which the feature extraction step can be applied either directly or by means of further and more sophisticated techniques (such as Common Spatial Patterns, in section 3.2.2).

The starting point are the raw signals, i.e., the recorded EEG μV values and the light attenuation detected by the NIRS detectors for the red and infra-red wavelengths (*wl1* and *wl2*). Since the two modes are acquired with different hardware, the first step is the synchronization of the signals. This is achieved by clipping both the EEG and the NIRS signals from the first occurrence of the triggers on. In order to make the EEG and NIRS signals have the same time span, also the tails of the longer signal (either EEG or NIRS) are clipped. After synchronization, both the signals are from 0 s to the same terminal time.

The pre-processing of the EEG and NIRS consisted of band-pass filtering and normalization; however, for the NIRS signal, the oxy- and deoxy- hemoglobin (*HbO* and *HbR*) were computed before every other operation by means of the Modified Beer-Lambert law (MBLL, in Eq. 2.4).

The choice of the filters regarded both the type of filters (Finite Impulse Response - FIR, or Infinite Impulse Response - IIR) and the cutoff frequencies. It was

anticipated in section 3.1.3 that μ and β bands were used in the study. In particular, μ band was filtered between 8-12 Hz, while a restricted β band was considered, ranging 18-25 Hz (same band-pass used in [30]), because it is mainly in those bands that Event Related Synchronization and Desynchronization (ERS/ERD) take place [43]. In order to filter out both the DC, the extremely low frequencies, and the heart beat, NIRS signals were filtered between 0.01-0.2 Hz.

Regarding the type of filters, both FIR and IIR have their advantages and disadvantages. In short, the main advantages of FIR filters are the fact that they are *all-zeros* filters, and thus they always assure the stability of the system. In addition, their phase response is linear, meaning that the filtered signal has no distortion. On the other hand, in order to obtain a high order in the amplitude response (dB/decades), an elevate number of coefficients is needed; however, the group delay, i.e., the lag in number of samples by which every frequency is delayed, is linear with respect to the number of coefficients. Given n = the number of coefficients, if n is even, the group delay of the FIR filter is constant and is $\Phi = {}^{n}/_{2}$; if n is odd, $\Phi = {}^{n+1}/_{2}$. On the contrary, IIR filters are not assured to be stable for they also contain poles. Their group delay is not constant, which causes distortions in the output signals; however, their performance in terms of steepness of the amplitude response is higher with respect to FIR filters [38].

In the current study for the EEG signals, IIR filters (Butterworth 4th-order) have been preferred to FIR ones. In order to obtain an acceptable filter performance using a FIR filter, in fact, more than 200 coefficients would have been needed, meaning that the output signals would be delayed by around 0.5 s, given the EEG sampling

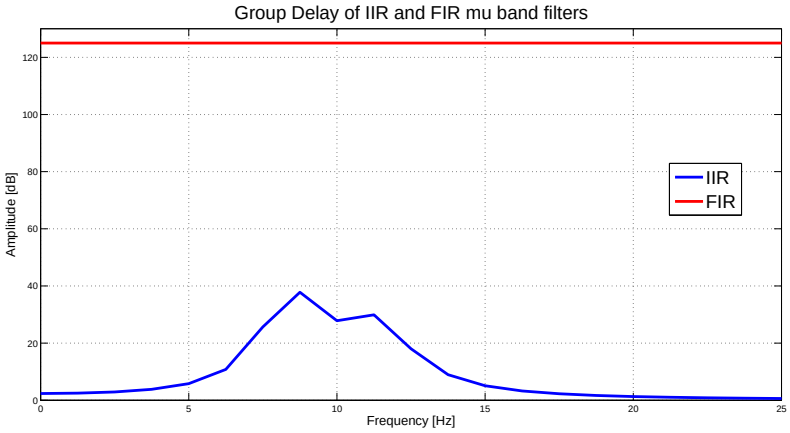


Figure 3.6: Group delay of a 250-coefficient FIR filter (red horizontal line) and a 4th-order IIR Butterworth filter with band-pass between 8 and 12 Hz.

frequency of 250 Hz. Fig. 3.6 shows the group delay of a FIR filter (250 coefficients) and an IIR (Butterworth 4th-order). It can be observed that the FIR filter has a constant group delay of 125 (0.5 s delay), while the IIR one has a maximum of 40 samples of delay (160 ms delay). The phase displacement of different frequencies does not exceed 20 samples in the filtered band, i.e., the distortion caused by the filter is not excessive. Moreover, band-power features are extracted from EEG signals and for this reason the distortions, which appear in the time domain, can be neglected.

For the NIRS signals, the choice of IIR filters over FIR ones was dictated by a simple fact: too many FIR coefficients would have been needed to filter out the DC, yielding an unacceptable delay of the filtered signals given the low sampling frequency of the NIRS system (10.42 Hz). Filtering out the DC is needed to avoid low-frequency trends of the signals. Fig. 3.7a shows the difference between an HbO signal filtered between 0.01-0.2 Hz (red line), and the same signal only low-pass filtered at 0.2 Hz (blue signal). It is clear that the low-pass filtered signal has a low descending trend

from the beginning to the end, which is not present in the band-pass filtered one. The upper cutoff frequency of the band-pass filter was chosen at 0.2 Hz (as in [13]) because higher frequency contents actually contains only noise with respect to the purpose of this study. Fig. 3.7b shows how the heart beat component is more present using a band-pass between 0.01-0.5 Hz (blue line) than using 0.01-0.2 Hz (both filters were Butterworth, 2nd-order). This said, the filter chosen for the NIRS signals was a 2nd-order Butterworth band-pass filter between 0.01-0.2 Hz. The order is only 2nd because higher orders yielded an unacceptably elevated group delay.

Normalization was performed by means of a Gaussian transform, where every signal was subtracted by its mean μ and divided by its standard deviation σ:

$$x_{norm}(t) = \frac{x(t) - \mu}{\sigma} \tag{3.1}$$

Before the normalization step, the EEG signals were clipped between $\pm 20\,\mu V$, in order to prevent any artifact to affect excessively the estimation of mean and standard deviation.

EEG-NIRS GUI : This short paragraph will describe the design and the main features of a custom-designed MATLAB GUI, used to pre-process and visualize the data. A GUI can speed up and ease the screening of the data and the choice of some pre-processing parameters (such as the choice of the filters) when dealing with multiple channel signals, even more for multi-modal imaging. Without the GUI, in fact, the visualization of time signals, spectra or even combination of signals of

different nature (e.g., EEG band-powers and HbO signals) would have been extremely tiring and time consuming. The GUI has been designed for combined EEG-NIRS only, and it cannot be used for visualization of the two kind of signals separately.

The GUI presents itself as shown in Fig. 3.8. It is composed by four main panels, namely, *LOAD*, *PRE PROCESS*, *VISUALIZE*, and *ANALYZE*. The *ANALYZE* feature will not be discussed here, but it was designed to add further GUI for more specific purposes. From the *LOAD* panel the user can easily load raw or pre-processed data (the latter can be saved from the GUI itself after the pre-processing). When a *Load* button is clicked, it opens a window and the user can select the files that s/he wants to open. After loading raw data (in the extension of *.edf* and *.wl1* for EEG and NIRS, respectively) from the pre-processing panel, the filter cutoff frequencies can be decided with the sliders or by manually inserting the text box at their sides. It must be emphasized that the pre-processing performed for visualization is different with respect to the one described in section 3.2.1. The main purpose of the GUI is in fact, the screening of the data, and in order to achieve a better visualization, *zero-phase* 4th-order Butterworth filters have been used both for the EEG and the NIRS signals. The user can also apply notch filters to eliminate the power line interference at 50 or 60 Hz. The user can also define what EEG channels s/he wants to consider for visualization and the configuration of the NIRS channels (source-detector pairs) using .txt files. Once the data are pre-processed they can be saved and visualized in the *VISUALIZATION* panel. The following are the time signals and spectra that can be plotted: raw NIRS signals (*wl1* and *wl2*), raw NIRS power spectra (HbO and HbR), pre-processed EEG spectra, and pre-processed time

signals (EEG, HbO, HbR, HbT - total hemoglobin concetration, EEG + HbO, EEG + HbR, HbO + HbR, EEG + EEG, EEG + HbO, EEG + HbR, and HbO + HbR). For all the time signals involving the EEG, the latter can be visualized as the pre-processed signal, or filtering it in the different EEG bands described in section 2.3. Another very important feature is the possibility of visualizing the EEG signals as band-power in the different frequency bands (as in Fig. 3.8). Last but not least, through the *EVENT* toggle button the user can visualize the different trigger events received during the experiment; different colors are assigned to different event codes, as shown in Fig. 3.8.

The GUI just described has been demonstrated to be extremely helpful in order to pre-process the signals and decide the best filter parameters.

3.2.2 Common Spatial Patterns (CSPs)

This section is about the Common Spatial Pattern (CSP) technique: a powerful transformation of spatially distributed signals in order to increase differences between mental states. It includes an overview of the CSP algorithm, including principles, implementation, and limitations. The next section (3.2.3) deals with the regularization techniques that are applied to overcome the limitations, in particular the over-fitting phenomenon. CSPs have been widely used for EEG-based BCI [30, 48, 5, 44, 13], and in this project a possible application on NIRS signals has also been investigated.

The Common Spatial Pattern performs a data-driven, *supervised* transform of the data that maximizes the difference in the variance of two different classes [5].

The output of the CSP algorithm is a matrix $W \in \mathbb{R}^{\mathrm{NxN}}$, where N is the number of channels (either EEG or NIRS channels), on which the original data are projected. Given $x(t) \in \mathbb{R}^{\mathrm{N}}$, the CSP filtered signals are:

$$x_{CSP}(t) = W^T x(t) \tag{3.2}$$

The resulting signal, $x_{CSP}(t)$, is still of dimension N. Every column of W, namely $w_j \in \mathbb{R}^{\mathrm{N}}$, is considered a *spatial filter*, because it project the vector $x(t)$ by weighing every channel with a different component (element of w_j). The result of the multiplication of a single time sample $x(t_0)$ of dimension Nx1 by w_j will be, in fact, a scalar value. Considering a generative model, in which a set of sources at time t is $s(t) \in \mathbb{R}^{\mathrm{N}}$is projected on the scalp, where the signals are measured, by means of a matrix $A = (W^{-1})^T$, the recorded signals can be written as:

$$x(t) = A^T s(t) = \sum_{j=1}^{N} a_j s_j(t) \tag{3.3}$$

where a_j, i.e., the j-th column of A, is called a *spatial pattern*. It is easy to demonstrate that by filtering $x(t)$ with the spatial filter w_j only the j-th source $s_j(t)$ is isolated and the sources $s(t)$ are identical the $x_{CSP}(t)$ defined in Eq. 3.2:

$$x_{CSP}(t) = W^T x(t) = W^T A^T s(t) = W^T \left[(W^{-1})^T\right]^T s(t) = I s(t) = s(t) \tag{3.4}$$

where I is the NxN identity matrix. In the filter-pattern view, the matrices A and W are also called *mixing* and *de-mixing* matrices, respectively. In fact the matrix A mixes the generative source activities among all the recording channels (Eq. 3.3 is a forward model), while the matrix W goes back, or in other word de-mixes the recorded signals back to the sources (Eq. 3.2 is a backward model).

The main principle of the CSP method, considering a linear mixing model that generates the signals [40], and assuming two different sets of sources for two separate classes, s_{C1} and s_{C2} (for the classes $\{1\ ,\ 2\}$), is to find the spatial filters that maximize the difference in variance in the filtered signals. The first components of the filtered signal ($x_{CSP}(t)$) will have maximum variance for class $C1$ and minimum for the class $C2$, while the last components, on the contrary, will have maximum variance for class $C2$ and minimum for $C1$. The extraction of these filters, under a discriminative view, can be seen as the optimization of an objective function [28, 5]. Since the purpose is to maximize the covariance for $C1$ and at the same time minimizing the one for $C2$, given the two covariance matrices Σ_{C1} and Σ_{C2}, the objective function can be written as:

$$J(\omega) = \frac{\omega^T \Sigma_{C1} \omega}{\omega^T \Sigma_{C2} \omega} \tag{3.5}$$

Since the filter ω can be re-scaled at pleasure, maintaining the vector directions, Eq. 3.5 can be subjected to the constraint that $\omega^T \Sigma_{C2} \omega = 1$. The optimization problem

can be now solved with Lagrange multipliers [4]:

$$L(\omega, \lambda) = \omega^T \Sigma_{C1} \omega - \lambda(\omega^T \Sigma_{C2} \omega - 1) \tag{3.6}$$

and the optimization occurs when the derivative with respect to ω is equal to 0:

$$\frac{\partial L(\omega, \lambda)}{\partial \omega} = 2\omega^T \Sigma_{C1} - 2\lambda \Sigma_{C2} = 0 \Longleftrightarrow \Sigma_{C1}\omega = \lambda \Sigma_{C2}\omega \tag{3.7}$$

Eq. 3.7 is a generalized eigenvalue problem and its solution yields a simultaneous diagonalization of the two covariance matrices, as shown in Eq. 3.8.

$$\begin{cases} W^T \Sigma_{C1} W = \Lambda_{C1} \\ W^T \Sigma_{C2} W = \Lambda_{C2} \end{cases} \tag{3.8}$$

It is important to notice that W can be re-scaled so that $\Lambda_{C1} + \Lambda_{C2} = I$ (I is the NxN identity matrix) [5]. This identity regarding the eigenvalues corresponding to different eigenvectors w_j is the key to increase the discrimination between $C1$ and $C2$: in fact the first components of W, i.e., the first eigenvectors, correspond to high eigenvalues in Λ_{C1}, meaning that the projection of the initial signals on those filters will have high variance for time samples belonging to$C1$ class and low variance for those belonging to $C2$. Conversely, the last components correspond to high eigenvalues for Λ_{C2} and low for Λ_{C1}: projections on them, thus, will yield high

variance for samples belonging to $C2$ class and low variance for those belonging to $C1$. Fig. 3.9 shows this behavior: the two top signals are EEG projected on the first and second component, while the two signals at the bottom are projected on the last two. CSP were computed to discriminate between *LEFT* and *RIGHT* tasks. Red and blue vertical lines represent the beginning of the right and left tasks, respectively, while the black vertical line is the starting of the rest period. It can be observed that the first two signals have higher variance during right tasks, and lower during left ones, and the last two signals have higher variance for left tasks than right ones.

The application of CSP, as explained above, requires the estimation of the covariance matrices for different pair of classes. Before their computation, the signals must be scaled and centered, as suggested in [5]:

$$X = \frac{1}{\sqrt{T}} X_{original}(I_T - 1_T 1_T^T)$$

where $X_{original}$ is a short segment of signal corresponding to a certain class of size NxT (where N is the number of channels and T is the number of time points in the segment), I_T is the T-dimension identity matrix, and 1_T is a column vector of ones. X is the scaled and centered matrix of size NxT that is used for the estimation of the covariance. It should be emphasized that $X_{original}$, differently from what it could be inferred, does not refer to the raw data, but to the pre-processed ones (both for the NIRS and the EEG signals). After scaling and centering the tracts of the signals belonging to the two classes (e.g., Right - Left or Rest - Task), the covariance matrices for the two classes are computed by averaging the covariance matrices obtained in

every trial belonging to the class (Eq. 3.9). Let $X_i^{(k)} \in \mathbb{R}^{\mathrm{NxT}}$ be a signal segment of the k-th trial belonging to class i. The covariance matrix for class Ci ($\Sigma^{Ci} \in \mathbb{R}^{\mathrm{NxN}}$) is estimated as follows:

$$\Sigma_{Ci} = \frac{1}{T_{Ci}} \sum_k \frac{X_i^{(k)} X_i^{(k)T}}{trace(X_i^{(k)} X_i^{(k)T})} \tag{3.9}$$

Here, T_{Ci} is the number of trials for class Ci. In order to normalize the covariances computed in separate trials, which, because of non-stationarities in the signals, can have a different baseline energy, each estimate regarding every trial is divided by its trace (total amount of energy).

One of the main advantage of the CSP algorithm is interpretability. For the reasons explained so far, it is straightforward that the filters and the patterns can be easily plotted on a brain map in order to visualize the activity of every channel and relate it to a neurophysiological behavior of the brain activity. Fig. 3.10, for example, shows the CSP estimated from the EEG data shown in Fig. 3.9, in particular the first and last components, which should resemble the minimum variance for right and left movement tasks, respectively.

CSP also has well-known limitations. First of all, the main problem of this method is its tendency to over-fit the data used to compute the covariance matrices [49]. If the matrices are computed from a small sample size the CSP algorithm has too many degrees of freedom with respect to the quantity of trials provided to estimate the covariances. This phenomenon will be shown and quantified in detail in chapter 4, where a comparison between its performance using all the data or

within the cross-validation (CV, explained in 3.4) is made. The bias of the sample-based estimate of the covariance matrices can be partially mitigated by applying regularizing techniques, that will be discussed in more depth in the following section. In short, regularization aims at generalizing the covariance matrices to make them less dependent on the sample used for their estimation. Another issue regarding CSP is inter-subject translation. Being based on subject specific covariances, this algorithm tends to be extremely tuned on each subject separately. Also in this case, though, regularization strategies can provide help in generalizing the filter extraction among different subjects.

3.2.3 Regularized Common Spatial Patterns (RCSPs)

In this section there are references to concepts and terms that will be extensively treated in section 3.4; however, in order to allow an essential understanding of the arguments, each of them is briefly explained.

As introduced in the previous section, CSP particularly suffers from over-fitting, especially when few trials are available for training, i.e., to estimate the most suitable parameters of a model that allow to predict the output (class) from input data (feature vector). Over-fitting, in a broad sense, refers to the phenomenon for which the estimated model adheres excessively to the data used to estimate it, but it does not perform as well when it is applied to other data. In a machine learning or model identification view, over-fitting occurs when the model does not generalize over new observations, but it includes also the stochastic noise present in the training

data. This phenomenon can be quantitatively observed in the performance of the classification algorithms on the training set, testing set, or the combination of the two. In particular, over-fitting can be identified when the accuracy (correct predictions / total predictions) on the testing set is significantly lower than the one obtained on the training set. Clearly, this issue is much more severe for small training sets, and it is also affected by the number of degrees of freedom that can be tuned by the model. In this work CSP has been applied both to EEG and NIRS signals. Because of the very low sampling frequency of the NIRS (10.42 Hz) with respect to the one of the EEG (250 Hz), it is likely that the NIRS signals will be more affected by over-fitting than the EEG ones. Although the term is usually referred to classification or regression algorithm, it is extended to CSPs because they perform a *supervised* decomposition of the signals, in other words they need labeled data to be computed. Chapter 4 will show how CSP dramatically over-fit the data when they are computed using the entire dataset (*before* CV), instead of estimating covariances and spatial filter *within* each fold of the CV.

By looking at how CSPs are computed in Eqs. 3.5 and 3.7, it is clear that the only possible way to over-fit the data is in the estimate of the covariance matrix of the two classes. In order to overcome the sensitivity to over-fitting, one can add an *a-priori* knowledge to the model, or, in other words, regularize it. In this case regularization can be done at two different levels: at the covariance matrix estimation or at the objective function computation level [28, 29]. In the current study a regularization method regularizes only at the matrices level has been used, and the second kind of regularization is only mentioned and shortly introduced for completeness.

The covariance matrix can be regularized as follows:

$$\tilde{\Sigma}_C = (1-\gamma)\hat{\Sigma}_C + \gamma I \tag{3.10}$$

where:

$$\hat{\Sigma}_C = (1-\beta)s_C\Sigma_C + \beta\Gamma_C \tag{3.11}$$

In Eqs. 3.10 and 3.11, Σ_C denotes the sample-based covariance for class C, $\tilde{\Sigma}_C$ is the regularized estimate, s_C is a constant scalar, γ and β are the regularizing parameters ($\gamma, \beta \in [0,1]$), and Γ_C is a generic covariance matrix estimated from all the subject data. The regularized covariance matrix can be shrunk towards both a generic and global matrix and towards the identity matrix, by adjusting the parameters β and γ, respectively.

The objective function can be regularized by adding a regularizing term in it, in order to penalize solutions that do not satisfy a given condition:

$$J(\omega) = \frac{\omega^T \Sigma_{C1} \omega}{\omega^T \Sigma_{C2} \omega + \alpha P(\omega)} \tag{3.12}$$

The term α represents the degree to which the objective function is penalized if it does not satisfy the prior $P(\omega)$.

There exist in the literature many kinds of regularization methods that usually differ in the use of the regularizing parameters. In the current study it has been decided to use the Generic Learning Regularized CSP (GLRCSP) proposed by [29]. Generic Learning regularization makes use of both β and γ, but it does not use α: therefore, the objective function is the one in Eq. 3.5, while it is only the covariance matrix to be regularized. Before describing in detail how to compute $\tilde{\Sigma}_C$, which indeed is slightly more complex than Eqs. 3.10 and 3.11, one comment is due on what the shrinkage parameters α and β are actually doing. The parameter β, as anticipated before, make the sample-based covariance tend to a generic covariance matrix estimated from all the subject except the one for which the covariance is being regularized. This should improve the performance of the CSP, because they are not being estimated from the training data only; however, some information about the "average" behavior among all the population is added. The parameter γ, instead, by shrinking toward a scaled identity matrix, weighs less the correlations found in the training set.

Generic Learning Regularized CSP : The regularization of the covariance matrix for each class Σ_C is performed in two steps. First of all the shrinkage towards the generic matrix is applied and afterwards the one towards the identity matrix. In GLRCSP there is actually not a direct computation of Σ_C and Γ_C; $\hat{\Sigma}_C$ in fact is computed by normalizing the sums of the sample and the generic covariance, $S_C$and $\tilde{S}_C$. In particular:

$$S_C = \sum_{m=1}^{M} S_{C,m} \tag{3.13}$$

$$\tilde{S}_C = \sum_{t=1}^{T} S_{C,t} \tag{3.14}$$

and:

$$S_C = \frac{X_C X_C^T}{trace(X_C X_C^T)} \tag{3.15}$$

In other words, $S_{C,i}$ is a covariance matrix estimated from a single trial of class C, m and t are trials belonging to the specific subject or tho other subject respectively. M and T represent the total number of trials for the specific subject and the sum of all the trials belonging to the other subjects (for class C), respectively. The first step of regularization allows to compute $\hat{\Sigma}_C$ as:

$$\hat{\Sigma}_C(\beta) = \frac{(1-\beta)\, S_C + \beta \tilde{S}_C}{(1-\beta)\, M + \beta T} \tag{3.16}$$

Eq. (3.16) shows that the shrinkage towards the generic behavior of the population is not a simple average between the initial covariance and the generic one, but it also takes into account how many trials are available for the current subject and for the rest of them.

To shrink towards the identity matrix, the latter (I) is re-scaled basing on the total energy of the matrix computed at the first step ($\hat{\Sigma}_C$) and the number of channels N:

$$\tilde{\Sigma}_C(\beta,\gamma) = (1-\gamma)\hat{\Sigma}_C + \gamma\frac{trace[\hat{\Sigma}_C(\beta)]}{N}I \tag{3.17}$$

Eq. (3.17) yields the regularized matrix for class C. After both the sample-based covariance matrices have been regularized, $\tilde{\Sigma}_{C1}$ and $\tilde{\Sigma}_{C2}$, CSP can be estimated from Eq. (3.7) simply by substituting the sample-based covariances (Σ_{C1} and Σ_{C2}) with the regularized ones ($\tilde{\Sigma}_{C1}$ and $\tilde{\Sigma}_{C2}$).

It can be noticed, that in order to apply GLRCSP for a binary classification, 4 different parameters can be adjusted (γ_1, γ_2, β_1, and β_2). The performance of the regularization technique can be extremely affected by the choice of those parameters, but the selection of the optimal ones can be very time consuming. The *goodness* of a certain set of parameters, in fact, must be computed with CV. For k times (in this case $k = 10$), covariance matrices must be computed and regularized, CSPs have to be estimated, features extracted, and classifiers optimized from the training set, while the accuracy is obtained from the testing data of each fold. For example, considering the μ power EEG signals, in order to perform a single evaluation, around 30 s were needed. If one imagine to evaluate all the possible combinations for one binary classification (in this work at least 3 binary classification are performed) of γ_1, γ_2, β_1, and β_2, constraining their value to be [0, 0.1, 0.2, ..., 0.9], 10^4 possible combination could exist, which means 300000 seconds, i.e., around 83 hours, would be needed. And this is only for one of four signals (μ and β filtered signals for

EEG; HbO and HbR for NIRS) and a single binary classification (e.g., Right - Left). Another faster optimization technique, from what was just said, was required in order to regularize all the single signal covariances for the different classification (see section (3.4)). Apart from the time, the over described method has also the disadvantage of discretizing the parameters, that by definition are real numbers.

In order to overcome these issues, it was decided to opt for a stochastic optimization, in particular using genetic algorithms.

Genetic Algorithm : Genetic algorithms belong to the evolutionary programming class and are biologically inspired, resembling the way that natural selection makes species evolve based on their fitness.

In short, the fundamental unit of genetic algorithms is a *chromosome*, i.e., a representation of a possible solution. In the current case, a chromosome will be coded as 4 real numbers, representing γ_1, γ_2, β_1, and β_2. A chromosome is *good* if it has a high fitness value, which is computed by means of a fitness function. Related to this case, the fitness function will output the accuracy of the chromosome using a k-fold CV. The genetic search is achieved using the algorithm 3.1. The new

Algorithm 3.1 GENETIC ALGORITHM:
Given the population size L, and the stopping criteria Γ

1: Initialize population with L random chromosomes: $x_i, i = 1, 2, ..., L$
2: **repeat**
3: Compute fitness value for each chromosome: $f_i = fitness(x_i)$
4: Rank the chromosomes based on their fitness
5: Generate a new population using genetic operators
6: **until** Γ not satisfied
return x_{max}

population are generated with different genetic operators. The *selection* decides how the chromosomes to be mated are selected. The *crossover* operator is used to mix two different solutions and the *mutation* operator randomly changes a solution in order to add randomness in the population. In order to maintain the best fitness of a population in the offspring, *elitism* is used, i.e., copying the best solutions of a population in the newly generated one.

Different parameters had to be set to regulate the generation of new populations and the stopping criteria: L is the number of solutions per population, N the maximum number of generations, S is the number of stall generations, and e is the number of *elite* solutions. Other genetic algorithm parameters, such as the probability of *crossover* and *mutation* or the *selection* method, were left as default (MATLAB *ga()* function was used). For the EEG searches (μ and β filtered signals), the following parameters were set: $L = 10$, $N = 15$, $S = 5$, and $e = 1$. For the NIRS signals, since CSP and CV steps were faster due to the lower sampling frequency, a larger population size has been used, with increased iterations: $L = 20$, $N = 30$, $S = 10$, and $e = 2$.

The genetic algorithms are much faster than other search algorithms, but their main drawback is that they are not assured of finding an optimal solution. The genetic algorithm can sometimes get stuck in local minima, and the usually the way to avoid this issue is to run the search multiple times with different initial points. In the current work, multiple initialization is not used, since a wider exploration of the parameter space with respect to the method, introduced in the previous paragraph, is already performed by the application of genetic algorithm itself. For the purpose

of the study, a sub-optimal solution is acceptable when taking in consideration the high time complexity of the problem.

3.3 Feature Extraction

Starting from an initial set of data, in this case already pre-processed, feature extraction consists of deriving informative values from the data. Practically, extracting features transforms the initial set into a more compact and significant one, from which predictive models are built. For most of BCI applications, signals are usually segmented in blocks or time windows, possibly overlapped with each other, and for every time window a set of features is computed [66], the feature vector. To every feature vector there is a correspondent output value, which in he case of classification is a label of a class (e.g., +1 for right movement and -1 for left ones). After feature extraction, the initial dataset is shrunk into a new one with M observation (representing M time windows in the initial signals) and N features. Every observation has a corresponding label, as anticipated before; therefore the label vector has size Mx1. In the current work the time signals were segmented using a time window of 1 s with 50% overlap (as in [13]). These parameters were chosen because they provide enough information in a single time segment and they can yield a relative fast output command (1 command every 0.5).

The main role of the feature extraction step is to fine characteristics of the signals which are informative and significant for predicting the output. This task is usually achieved by further processing of the signal. The entire information inside every time

segment, in fact, must be collapsed into a relatively small dimensional vector, to avoid a high dimensionality of the feature space, which can result in a lower classification performance to to lack of generalization [24, 28].

In this work, two different kind of features were extracted from the EEG and NIRS signals: one from the CSP-filtered signals (named CSP features), and another one from the original non-CSP-filtered pre-processed signals (named NCSP features). For CSP-filtered signals, since the algorithms increase the differences between classes in terms of variances, feature extraction aims at capturing this difference. Using non-CSP-filtered signals, especially for the NIRS, features of different nature have been used. In chapter 4, the performance of the different feature extraction methods will be shown and evaluated. Since the feature extraction depends on the nature of the signals, it is described in separate paragraph for EEG and NIRS.

EEG : For the EEG, features were extracted separately from μ and β filtered signals. Actually, the processing for CSP- and NCSP-based features computation was exactly the same, but feature vectors differed in terms of number of channels used and number of time segment features concatenated. The variance indicator chosen was the Band-Power [45], which is equivalent of computing the envelope of the signals. The time signals, after the band filtering, were rectified and then a low-pass filter was used. The low-pass filter adopted in this work was a 250 coefficients rectangular FIR filter, i.e., a moving average filter.
For the NCSP-filtered signals, a feature vector consisted of the average of the band-power of each EEG channel. In this case, the size of the feature vector was 21.

A variance indicator was also chosen for NCSP features because of the stochastic nature of the EEG signal, which is usually translated into spectral features [66]. For the CSP-filtered signals, since most of the discriminative information lies in the first and last set of components, the first three and last three components were used and features were computed as the average band-power over the time window. Due to the small dimensionality of the CSP-derived feature vector, three consecutive feature vectors were concatenated to form a new feature vector that includes also information regarding the time evolution of the signals [27]. After this step the feature vector became an 18 dimension feature vector. The same concatenation was not performed for NCSP-derived features, for it would result in a very high dimensional feature space (63 features). Features were extracted separately from μ and β filtered signals.

NIRS : NIRS signals were processed in completely different ways to extract CSP and NCSP features and, such as EEG, HbO, and HbR signals were treated separately. For CSP ones, since only 10 NIRS samples are found in a time segment, the range of the signals seemed to represent a good indicator of the variance. It should be added that while for EEG an increase of variance is usually perceived as an average higher amplitude, for the NIRS signals it is easier to see it as an increase of low frequency oscillations (NIRS signals were filtered between 0.01-0.2 Hz). Fig. 3.11 shows how the amount of oscillations is modulated after CSP filters. Also for the NIRS, ranges were computed for the first and last three components of the CSP-filtered signals and three consecutive time segment features were concatenated to give an 18 dimension feature vector.

For NCSP features, two different kind of features were used: the average and a slope

indicator. Averages were computed only by extracting the mean of pre-processed NIRS signals, while the slope indicator, which was demonstrated to be very significant in motor execution/imagery classification in [32], was computed as the slope of the average features, i.e., by subtracting the previous time window average from the current one. Concatenation was avoided in order to not increase the feature space dimension, since the number of features was already 68 for both HbO and HbR. One last comment regards the use of the average over a time segment as feature; in NIRS BCI literature, e.g., [13] and [32], the average is also referred to a baseline preceding the task. Clearly, this approach is not translatable in an online application for an asynchronous BCI, since there is no cue when the task begins. The current study tried to overcome the real-time inapplicability of these feature by using band-pass filters instead of low-pass only [13, 32].

Both the EEG and NIRS features were $\log_2$ transformed and normalized with Eq. 3.1, in order to adjust for the right skewness of the features (the mode of the distribution is not centered and the right tail is longer) and to prepare to feature dataset for the classification step. The classification, in fact, is performed by means of Fischer's Linear Discriminant Analysis, which works better when dealing with normal distributions (achieved by log transform) and with equal variance (achieved with normalization) [10, 5].

3.4 Classification and Validation

Classification, in *machine learning*, means predicting an output class or label for a given input vector (feature vector); therefore, it basically consists in mapping the feature vectors with the labels of the classes.

Classification is usually performed in two steps: first, a model or hypothesis or classifier is estimated from the data, so that, second, new data can be classified by means of the classifier. In other words, a classifier has to *learn* from the data particular patterns, in order to recognize these peculiar patterns in new input data and predict their output. In this section only a binary classification is covered in detail. The multi-class classification used in the current study is, in fact, achieved by combining multiple binary ones.

When the model is tuned too much on the particularities of the specific dataset and does not aim at generalizing over new data, the estimation of a classifier parameters can result in over-fitting of the data used to estimate it. In practice, over-fitting yields a much worse performance when predicting the output of new unlabeled data. In order to overcome this issue, or at least to quantify it, usually the initial dataset is split into a training set and a testing set. The training set is used to estimate the parameters of the model, while the testing one is used to evaluate the capability of generalization of the model, since the testing data are hidden during the learning phase and they are labeled. The performance on the testing set, therefore, can be quantified in terms of accuracy:

$$Acc_{TEST} = \frac{correct\,predictions_{TEST}}{total\,observations_{TEST}} \tag{3.18}$$

In order to evaluate the real expected performance of a classifier, though, the division between training and testing set is not enough. The choice of the observations that will be part of the two sets, in fact, can bias the evaluation of the model. For a better evaluation, a k-fold cross-validation (CV) is performed. The k-fold cross-validation consists of repeating k times the split of the dataset, so that the training and testing sets are different for every fold. Every trial, thus, is part of the training set for $k-1$ times, and part of the training once. Before the trials were assigned to one of the two sets, they were randomized; in order to compare the performance of different sets of features, the random seed of the pseudo-random generator was always the same. In each fold, a model is estimated from the training set and evaluated on the testing one, and after the k iterations the accuracy is computed as the mean of the k accuracies. In this work two different approaches have been used to classify the data, and as explained later in this section, k was set to 10 or 5 (10-fold and 5-fold CV). Accuracies were computed on a subset of the features. In particular the first 2 seconds of every rest and task were discarded to consider only the steady states of rest and motor tasks.

Since in BCI application it is also important to have a temporal view, a dynamic evaluation of the performance was performed as follows. For each iteration:

- testing observations ($n = \frac{n_{TRIALS}}{k}$) were aligned and clipped in a time scale of

the trial, creating a 3D matrix $X \in \mathbb{R}^{\mathrm{nxTxM}}$ (where T is the number of time segment in the trial and M the dimension of the feature vector)

- predictions were performed using the trained classifier ($P \in \mathbb{R}^{\mathrm{nxT}}$) corresponding to the correct labels $L \in \mathbb{R}^{\mathrm{nxT}}$

- dynamic accuracies were computed time segment by time segment by applying Eq. 3.18 over the columns of P and L.

This dynamic accuracy resulted in a time signal over the trial length for each classifier, allowing to evaluate the role of time. Standard accuracy in Eq. 3.18, in fact, is only static and does not include the dynamic aspect, which is so crucial for BCI applications.

There are really many different classifiers that approach the problem in many ways. Some make use of Baesyan probabilities (e.g., Baesyan Decision theory), others use linear or quadradic models (Linear or Quadratic Discriminant Analysis), some of them try to cluster the data depending on their *entropy* (Decision Trees), and others use more complicated principle and optimization techniques that will not be discussed here (e.g., Artificial Neural Networks or Support Vector Machines). In the current project a Fischer's Linear Discriminant Analysis was used and it is described in the following paragraph. The choice of a linear classifier rather than more complex ones was recommended by different observations. First of all, when using CSP, the separability between classes is done by the spatial filters and a more complex classifier is not needed; moreover the use of a complex classifier would add a high number of

degree of freedom in the estimate of a model, worsening the phenomenon of overfitting. A simple linear classifier was used also for NCSP features in order to obtain a standard evaluation of the performances.

Fischer's Linear Discriminant Analysis (LDA) : As the name suggests, LDA is a linear classifier, i.e., it assigns predictions to a feature vector by applying a linear combination of the features. Let $\mathbf{x}$ be the feature vector, then:

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + \mathbf{w_0} \tag{3.19}$$

The classifier output depends on the sign of $g(\mathbf{x})$: e.g. if it is positive it predicts class +1 and if it is negative it predicts -1. Eq. 3.19 can be referred to as the decision rule of the classifier, and it can be graphically seen as a projection of the feature vector $\mathbf{x}$ on the weight vector $\mathbf{w}$. For a binary classification, for which LDA is designed, one can imagine an optimal hyperplane that divides the feature space in two regions (one for each class); let us call this optimal hyperplane *decision boundary* $\mathbf{d}$. Since the classes are separated based on the sign of $g(\mathbf{x})$, it is straightforward that the *decision boundary* equation would be $g(\mathbf{x}) = 0 \iff \mathbf{w}^T\mathbf{x} + \mathbf{w_0} = 0$. Assuming 2 points $\mathbf{x_1}$ and $\mathbf{x_2}$ that lie on $\mathbf{d}$, the following equality is satisfied:

$$\mathbf{w}^T\mathbf{x_1} + \mathbf{w_0} = 0 \ , \ \mathbf{w}\mathbf{x_2} + \mathbf{w_0} = 0 \implies \mathbf{w}^T(\mathbf{x_1} - \mathbf{x_2}) = 0 \implies \mathbf{w} \perp \mathbf{d}$$

The *decision boundary* is, therefore, perpendicular to the weight vector $\mathbf{w}$.

The classifier training can be reduced to an optimization problem that finds the direction $\mathbf{w}$ for which the 2 classes are maximally separable. Assuming normality and equal covariance for the distributions of the two classes, which have means μ_0, μ_1 and covariances Σ_0 and $\Sigma_1(=\Sigma)$, then the separation between the classes is defined by Fischer as:

$$S = \frac{\sigma^2_{between}}{\sigma^2_{within}} \tag{3.20}$$

where $\sigma^2_{between} = (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T$ and $\sigma^2_{within} = \Sigma_0 + \Sigma_1$. The denominator, which expresses to what extent the two distributions are "far" from each other, is to be maximized, while the denominator, which represents the "expansion" of the distributions around their means, should be minimized in order to achieve a better separability. By projecting S on $\mathbf{w}$ the estimation of the optimal weight vector is reduced to an optimization problem, i.e., the maximization of the objective function:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \sigma^2_{between} \mathbf{w}}{\mathbf{w}^T \sigma^2_{within} \mathbf{w}} \tag{3.21}$$

The projection that maximize Eq. 3.21 will yield the best performance of the classifier. The estimation of $\mathbf{w}$ can be visually appreciated in the 2D example in Fig. 3.12.

Under a probabilistic point of view, the posterior probability of a feature vector $\mathbf{x}$ belonging to a class i can be estimated applying the logistic finction to $g(\mathbf{x})$ in Eq.

3.19:

$$P(C_i|\mathbf{x}) = \frac{1}{1 - e^{(-g(\mathbf{x}))}} \tag{3.22}$$

This metric can be used as a score of the prediction, indicating the confidence (posterior probability) in the classification. It can be extremely useful when LDA is used for multi-class classification, because it suggests which of the multiple classes is more likely given the features.

Classification Approaches : Two different approaches were used to perform the complete 5 classes -Rest (Re), Right-Arm (RA), Right-Hand (RH), Left-Arm (LA), and Left-Hand (LH)- classification. Both the approaches require the estimation of multiple binary classifiers, that combined allow to predict each of the 5 classes. They have their advantages and disadvantages and will be evaluated and discussed in chapter 4.

The first approach requires the estimation of 3 different classifiers in order to discriminate between the following classes:

1. Rest vs Task (Re-T)
2. Right vs Left (R-L)
3. Arm vs Hand (A-H)

After the model estimation, in order to predict the one of the 5 classes, the 3 classifiers must be run in cascade. The Re-T classifier predicts first: if Re is predicted then

the class is 0, otherwise R-L and A-H classifiers will predict one of the remaining 4 classes. The order in which classifier 2 and 3 predict is not important. This approach has the advantages of giving as output an univocal class (the second approach does not), and it increases the number of trials per class by grouping together RA and RH into R, LA and LH into L, RA and LA into A, and RH and LH into H. This way there are 50 trials available for classification 2 and 3, and 100 for classification 1. The latter advantage, though, can have a downside: by grouping the classes, in fact, one can argue that some physiological information is lost, because different brain areas are involved. To evaluate this possibility, a second approach is adopted.

The second approach is built by 5 different classifiers:

1. Rest vs Task (Re-T)
2. Right-Arm vs Left-Arm (RA-LA)
3. Right-Hand vs Left-Hand (RH-LH)
4. Right-Arm vs Right-Hand (RA-RH)
5. Left-Arm vs Left-Hand (LA-LH)

Also in this case, the Re-T classification precedes all the others. If T is predicted, the others classifiers are run in parallel giving 4 outputs. The assignment to a class can be done using a majority voting principle and taking into account the score of every prediction. To provide a concrete example, imagine that for a feature vector, the classifier predicted task and for the other 4 classifier the output were the following: classifier 2 predicts RA with 85% confidence, classifier 3 predicts LH

with 60%, classifier 4 predicts RH with 55% of accuracy, and classifier 4 predicts LA with 70% confidence. Clearly, even if none of the class is predicted twice and, thus, the majority voting cannot be applied, the most likely class is RA because it has the higher confidence to be true. Of course it could be a misclassification, but the probability of error would be the minimum when predicting RA.

For each the above mentioned classifiers, different sets of CSP have been computed. The performance of every classification step has been evaluated with a 10-fold CV for the first approach and a 5-fold CV for the second one. Only 5 folds have been used for the second approach in order to assure that in every testing set there were 5 trials belonging to each class. The testing trials were non-randomized in this case, but at every fold, 4 experimental blocks were used as a training set and the other as a testing set (see section 3.4 for a detailed explanation on the experimental procedure). Moreover, every classifier was tested using features derived from different signals, with different CSP variants (including NCSP features explained before). SIGNALS:

- μ filtered EEG
- β filtered EEG
- HbO
- HbR
- μ+ β (EEG)
- HbO + HbR (NIRS)

- μ+ β + HbO + HbR (COM)

CSP:

- computed before CV (*batch*)
- computed within CV (*cv*)
- regularized and within CS (*reg*)
- no CSP (*noCSP*)
- CSP only for EEG (*CSPeeg*)

The abbreviations in parentheses refer to the ones that are going to be used in the statistical analysis to identify the different signals and CSP approaches. In order to identify the most suitable signals and CSP approach, as well as the best classification approach, the data need to be statistically analyzed, as described in the next section.

3.5 Statistical Analysis

The statistical analysis involves the use of tests in order to find significant differences among the data. To sum up, the datasets analyzed are made of the accuracies obtained by the classifiers with combination of features and CSP approaches, and the purpose is to find which are the best configurations that yield the highest accuracy. The statistical analysis has been performed in R environment.

Before performing any kind of test, though, the data are to be explored. Data exploration in practice consists of plotting the different distributions in order to get an idea of where the significant differences might be. Boxplots of the different signals, divided by the CSP approach for motor execution and motor imagery can give a first insight on the dataset, driving further hypotheses and analyses. Therefore, in the data exploration step, it is crucial in a statistical analysis to not waste any time looking for differences where they do not exist.

A hypothesis test can be performed using parametric or non-parametric statistics. Parametric statistical tests assume that the data are generated by a normal distribution (that can be described with a set of parameters) and that they have equal variance. When these assumptions are not satisfied, the only way to obtain reliable results is to opt for non-parametric statistical tests, which do not assume any particular distribution of the data, but instead are based on ordering the observations and considering the rank to extract significant differences. In order to assess the normality of the data, for every distribution the *Shapiro-Wilk* test was performed [51], while for equality of variance the *Levene* test was used [25]. If both the assumptions are satisfied, then the *t-test* and the *ANOVA* test can be used for 2 and multiple populations, respectively; otherwise the *Mann-Whitney U* test and the *Krukcal-Wallis* test should be used.

Statistical analysis is performed on Rest - Task classification separately, because it is common between both the classification approaches. For each of the approaches, then, the best configuration of signals and CSP methods is investigated in detail.

3.6 Online Evaluation

The online evaluation of the entire system is very important, as introduced in section 2.1, in order to understand and quantify the real potential and capability of the BCI. The real-time evaluation, in fact, goes beyond the concept of accuracy, taking into account a metric, the Information Bit Rate (ITR), that comprehend the accuracy, the time, and the possible choices. It is clear that two BCI with the same accuracy, but with the difference that one classifies between 2 classes and the other between 4 classes, may have completely different communication power (the 4-class BCI allows the user to perform more actions). However, the development of the real-time system that allows the computation of the ITR is very challenging and time consuming. In this section the design of such a system will be presented, but unfortunately it has not been implemented yet.

Fig. 3.13 shows the block diagram of the setup. Comparing it with the one in Fig. 3.4, it can be noticed that the laptop (LT) running Presentation software is no longer used, because the synchronization is not needed anymore since the data stream of the EEG and NIRS system occurs at the same time, and because there is no need to present stimuli for the subject. In fact the subject interacts only with a tablet with an Android application developed to test the performance in real-time. On the WS side, the C++ application is built on the basis of the one explained in section 3.1.3, with three main differences:

- first of all, NIRS signals must be streamed too. The application, then, behaves as a Client of NIRStar, the server that provides real-time data via TCP/IP.

The raw signals are extracted and filtered online with the same IIR filter used in the pre-processing (see section 3.2.1). In order to avoid the program to wait for NIRStar data packets while "losing" EEG ones, the NIRS streaming must be performed on another thread that run in parallel with StreamingThread and ProcessingThread ones.

- the second main difference lies on the fact that now the application not only has to compute EEG power bands, but it has to compute the features described in section 3.3 and to classify using the subject-dependent trained classifiers. In particular it should compute the feature set accordingly to the configuration and approach that yields the best performance for the subject using the BCI.

- and last, instead of sending the output via USB to Presentation, the program sends it to the Android application via Bluetooth.

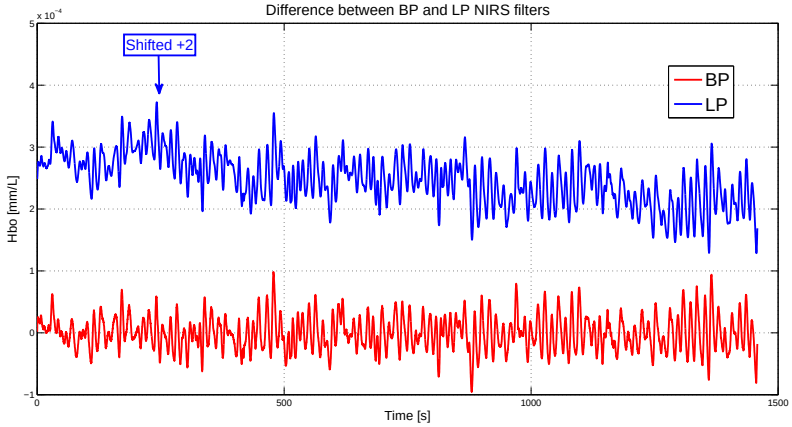
In order calculate the ITR, the Android and the C++ applications needs to communicate; the implementation can be uni-directional if the Android application acts as a *Master*, deciding the timing and the nature of the evaluation. In that case it would be the Android app itself that computes the ITR. For example, a possible design could be a cue-based one, in which 4 arrows pointing left, right, up, or down tell the subject which movement to perform: right arm, left arm, right hand, or left hand. One of the four arrows appears randomly on the screen, let us say the left one which corresponds to the left arm; then the Android app waits until a classification output is delivered from the C++ application: if the classification matches the expected one, it counts one hit, otherwise one error. Then the arrow disappears and after some

seconds (e.g., 2-3 s) another arrow appears and so on for a fixed amount of time (e.g., 2 minutes). Given C the number of classes (C=4), the accuracy measured p, the number of decisions (number of arrows presented) d , and the duration in minutes T , the ITR can be computed as [5]:

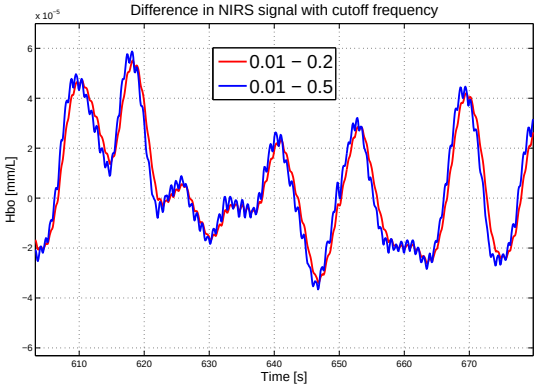
$$ITR = \frac{d}{T}(p\,log_2(p) + (1-p)log_2(\frac{1-p}{C-1}) + log_2 C) \tag{3.23}$$

Of course a crucial point in the real-time evaluation and use of the BCI is the classification step. In fact, although speed is a key factor of a BCI performance, it is probably more important to have a higher accuracy with a slower speed than the opposite. With this perspective, the classification should not be forced to output a decision at every time segment (0.5 s). The output could be filtered out to increase the robustness of the system using to different approaches. The first way takes into account the score of the LDA classifier output in Eq. 3.22, i.e., the degree of certainty of the classification. In order to minimize mistakes, the C++ application would send classification output only when the confidence of the classification is above a certain threshold. Another way to increase robustness is to filter the outputs with a majority vote approach, which means classifying not only in base of the current output, but also taking into account the previous ones (e.g., the last 3 or 4). Without an online test, though, it is difficult, if not impossible, to simulate these approaches and to decide what is the best one (which can also be a hybrid approach) and what are the most suitable parameters.

The reader can notice that this cue-based design is not exploiting the asynchronous capability, i.e., the possibility of detecting rest and task in a continuous way. A self-paced design could consist of a maze that the subject could traverse in 4 different directions (left, right, up, or down) trying to get to the end in the minimum amount of time. Nevertheless, even though this design could resemble more closely a real application (e.g., a wheelchair control), in this case the evaluation in terms of ITR would not be as straightforward as the cue-based design.



(a)



(b)

Figure 3.7: (a) Comparison between low-pass (0-0.2 Hz, blue) and band-pass (0.01-0.2 Hz, red) filters on NIRS HbO signal (b) Comparison between 2 band-pass filter: 0.01-0.5 Hz (blue) and 0.01-0.2 Hz (red).

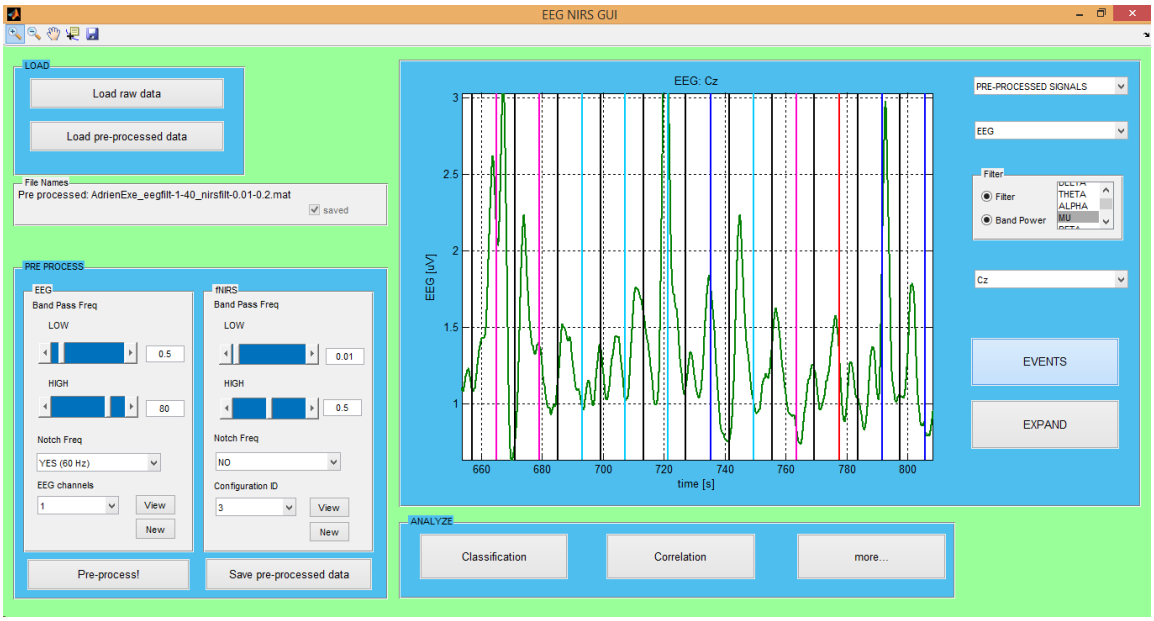


Figure 3.8: EEG-NIRS GUI: on the top left there is the *LOAD* panel; on the bottom left, the *PRE PROCESS* panel; on the right the *VISUALIZATION* panel enables to plot raw signals, spectra and pre-process data in different combinations of signals and channels; with the *FILTER* panel inside the *VISUALIZATION* one, EEG signals can be filtered in different bands and band-power visualized; the *ANALYZE* panel is thought to allow the user to add custom processing to the GUI.

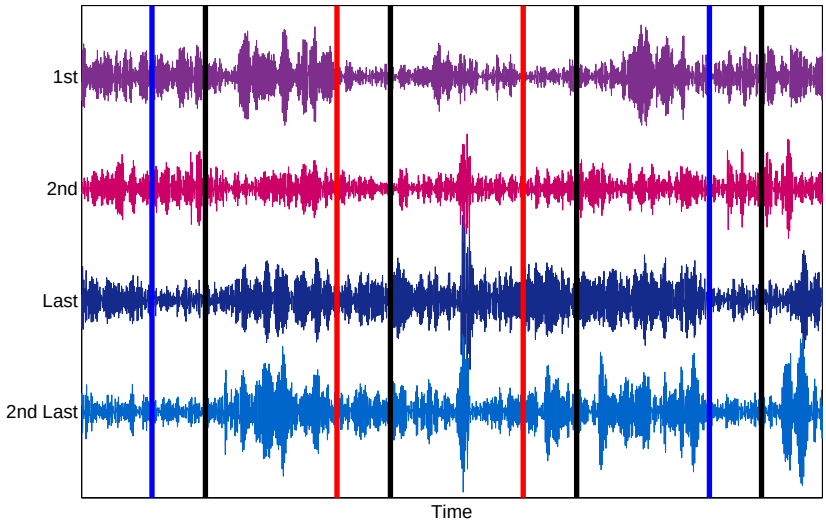


Figure 3.9: Time signals of 1st, 2nd (top 2 rows), Last and 2nd Last (bottom 2 rows) CSP components for Right - Left classification: red and blue vertical lines represent the beginning of right and left tasks, respectively, and black vertical lines show the beginning of rest phase. The time between the beginning of the task and the beginning of the rest is 4 s. The first 2 signals have higher variance for left tasks and lower for right ones, while the last 2 rows have the opposite behavior, i.e., higher variance for right tasks and low for left ones.

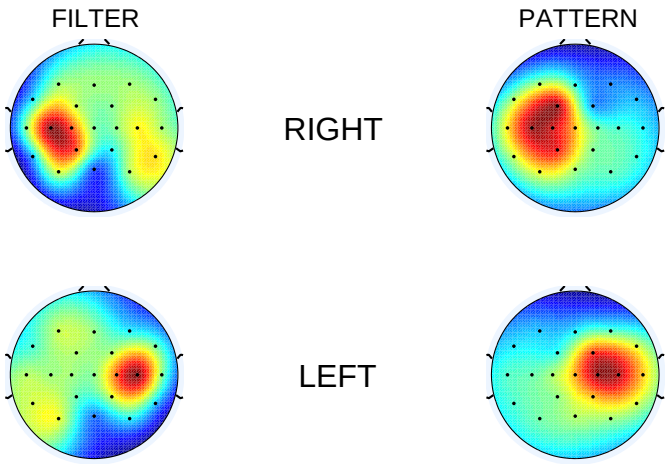


Figure 3.10: Scalp plots of CSP filters (on the left) and corresponding pattern (on the right) for Right - Left motor execution classification: the top row, representing the first component of CSP, enhances most of all the channels on the left emisphere, i.e., the controlateral side of right movements, while the bottom row, displaying the last CSP components, enhances the right emisphere, which is the controlateral side of left movements.

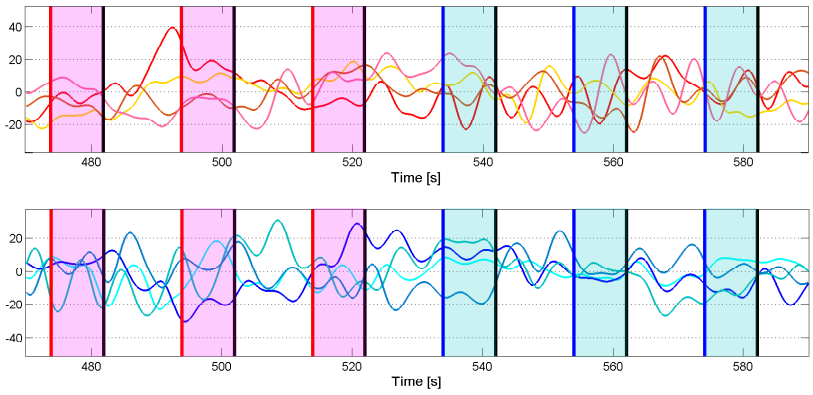


Figure 3.11: NIRS HbO time series of first 4 (top panel) and last 4 (bottom panel) CSP components for Right - Left classification. The x-axis represents the time [s], while the y-axis is uniteless because the signals are normalized. Red and Blue shaded areas correspond to right and left trials, respectively. Black lines are the beginning of rest. It can be observed that for right tasks (red areas) the first components have lower oscillations, whereas they start oscillating remarkably for left tasks (blue areas). The last components show an opposite behavior: they have minimum oscillations for left tasks (blue areas) and maximum for right ones (red areas).

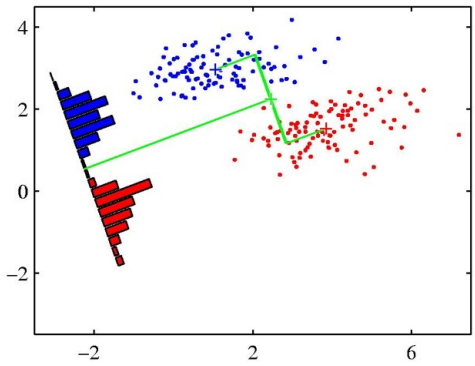


Figure 3.12: Image representing the maximized separation given by the projection on the optimized weight vector: whereas the right and left distributions are not separable in the standard reference system, the become completely separable if projected on $\mathbf{w}$.

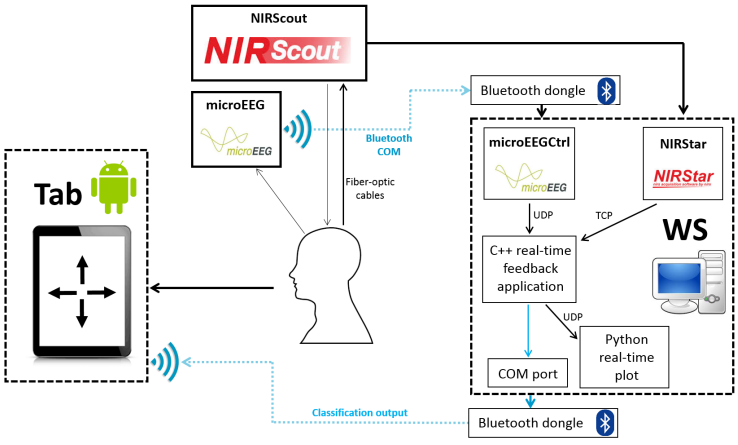


Figure 3.13: Setup block design for an online evaluation of the system. Note that the 2 "Bluetooth dongle" blocks are actually the same and they are divided for sake of presentation.

Chapter 4

Results

The results are divided into three sections. The first one presents the results about the Rest - Task classification (section 4.1), which is common between the 2 approaches as described in section 3.4. The second part focuses on the first approach, i.e., Right - Left and Arm - Hand classification (section 4.2), while the third section is about the second approach, which consists of pairwise classification among the 4 classes (section 4.3). In every section there is a data exploration part and an analysis of the CSP approaches for Execution and Imagery tasks separately, as well as a comparison between Execution and Imagery tasks. Moreover, the dynamic accuracy evaluation is presented for each classifier. All the figures involving box plots and significance levels are depicted in Appendix B for sake of conciseness. As explained in section 3.5, parametric statistics is applied when possible (when assumptions are satisfied), but it will not be specified in the text - only the significant differences are reported.

The CSP approaches that appear in the box plots and are used in this chapter

have the following labels:

batch CSP are computed *before* CV

cv CSP are computed *within* CV

reg CSP are computed *within* CV with regularization and optimized parameters

noCSP CSP are *not* used

CSPeeg *reg* is used for EEG signals and *noCSP* for NIRS ones

4.1 Rest - Task

4.1.1 Execution

First of all, let us see whether there are significant differences among accuracies depending on the CSP approaches. The EEG and NIRS signals are considered separately, and afterwards the combination of the signals is taken into account. Table 4.1 contains the accuracies of all signal-derived features and their combination divided by CSP approach.

Fig. B.1a (Appendix B) shows the μ and β performance with box plots; however, the difference is significant only between the μ-*batch* and μ-*noCSP* approaches. For both the signals it can be observed that CSP over-fit, but not excessively, and that the regularization process improves the performance (see Table 4.1). Accuracy is definitely worse without using CSP (*noCSP* around 5% less than *reg*). For the NIRS

signals, box plots and significance levels are shown in Fig. B.1b (Appendix B). Even if NIRS regularization increases the performance (*batch* accuracy is higher than *cv* accuracy), the *noCSP* approach yields extremely high accuracy that outperforms the use of CSP (*reg*). There is no doubt that the *noCSP* is the best strategy for Rest - Task classification using NIRS-based classifiers. For the combination of features, only the regularized approach is reported, because it improves the performance with respect to *cv* and is not biased as is the *batch* approach. Therefore, as shown in Fig. B.2, only *reg* and *noCSP* are shown for EEG (μ + β) and NIRS (HbO + HbR), while *reg*, *noCSP*, and *CSPeeg* are shown for COM (all 4 signals together). For the EEG signals, regularization is significantly better than *noCSP* at the 5% significance level. The NIRS-based classifier yields a significantly higher accuracy when CSP are not used ($p<0.001$), and the combination of EEG and NIRS features improves the overall accuracy (but not significantly with respect to *noCSP*).

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	80.3±4.4	83.5±4.6	74.3±3.9	71.1±3.6			
cv	77.7±6	82.3±5.7	65.5±4.2	62.2±3.6			
reg	78.6±5.7	82.8±5	69.4±4.1	65.9±4	**85.2±4.6**	69.8±4.5	86.2±4
noCSP	73.3±6.4	77.5±7.5	90.5±6	89±7.1	80.4±6.7	**92.4±5.3**	93.7±4.1
CSPeeg							**94.2±3.4**

Table 4.1: Rest - Task Motor Execution: accuracies obtained with a 10-fold CV for single signals features and combination of them. The combination of the features improves the overall accuracy.

Observing the performance of the classifiers in terms of accuracy, it can be observed that both for EEG and NIRS, the combination of the signals improves the accuracy, and that the use of features derived from the all four signals yields the highest accuracy. Such a high accuracy (94.2%) in the Rest - Task classification for

motor execution could allow the use of the BCI in a synchronous or self-paced mode, in which the user can control the BCI output any time, without the need of visual or auditory cues.

4.1.2 Imagery

Table 4.2 shows the accuracies in motor imagery tasks for the 4 signals separately and for their combination. Box plots of performances for EEG and NIRS signals are displayed in Figs. B.3a and B.3b (Appendix B). For EEG signals there is no significant difference, but regularization slightly improves the performance. As in motor execution, the best approach for NIRS is without the use of CSP, which is significantly better than all the other possibilities (*noCSP* accuracy is ∼10-15% higher with respect to *reg*). Combining features improves the accuracy both for EEG and NIRS (Fig. B.4). The difference between *noCSP* and *reg* is not significant for EEG; however, it is highly significant for the NIRS (also in this case *noCSP* is the best approach). By using features derived from all the different signals the highest accuracy is achieved (85.8%); such a high accuracy could be translated in an asynchronous BCI for motor imagery paradigm.

4.1.3 Execution vs Imagery

Tables 4.1 and 4.2 showed that for both execution and imagery tasks, the highest accuracy for Rest - Task classification is obtained when features derived from all channels are combined, applying regularized CSP on EEG signals and not using

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	75.4±7.1	72.2±7	66.7±5.1	65.6±3.7			
cv	72.3±9.2	66.9±8.9	59.1±5.8	58.1±4.8			
reg	73.8±9.1	70.4±7.8	63.6±6.1	61.2±4.8	**74.8±9.2**	63.1±6.4	76.3±8.5
noCSP	69.1±8	66.5±6.7	79.1±8.6	77.9±8.7	71.3±8	**82.8±7.9**	84.9±7.4
CSPeeg							**85.8±7.2**

Table 4.2: Rest-task Motor Imagery: accuracies obtained with a 10-fold CV for single signals features and combination of them. The combination of the features improves the overall accuracy.

CSP for NIRS ones. In order to look for differences between execution and imagery performance, only the configuration COM-*CSPeeg* is taken in consideration, because it resulted in the highest performance. The distributions of accuracies for execution (EXE) and imagery (IM) is shown in Fig. 4.1. There is a remarkably significant difference between the two tasks ($p<0.001$), and motor execution yields a higher accuracy with respect to motor imagery.

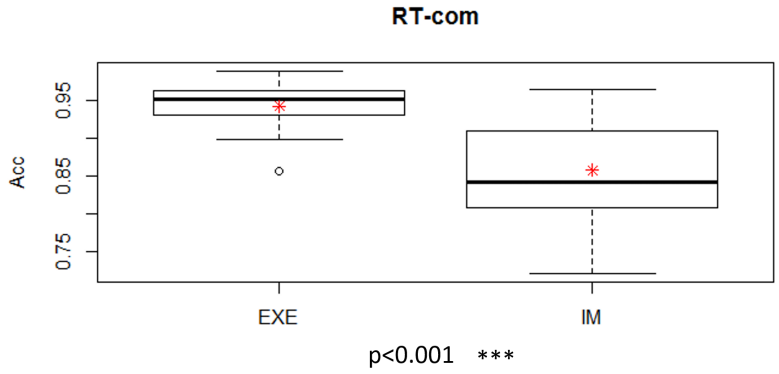


Figure 4.1: Rest-task classifier performance using *CSPeeg* approach for motor execution (EXE) and motor imagery (IM).

Whereas, it has been shown that the mental strategies for motor execution and motor imagery are similar (in section 2.3), it can be concluded that execution is a

more reliable strategy in this case. One of the reason that could explain the worse performance of motor imagery is that the subjects had no experience at all in this kind of mental exercise and they found it very difficult. With more practice and a more robust feedback, the second distribution would probably approach the first one, filling the gap between performances.

4.1.4 Dynamic Evaluation

The dynamic evaluation, as described in section 3.4, consists of plotting the mean accuracy of the different testing trials averaging on the time windows. Fig. 4.2 displays the trend of the accuracy for Rest - Task classification. The black vertical lines mark the beginning and end of the task. The feature configuration used is *CSPeeg* for both execution and imagery tasks, because it yielded the highest accuracy. The colored lines are the accuracies of the different subjects, and the black thicker line is the average of them. It can be noticed that right after the cues the accuracy drops. This is not due to a deficit of the classifier, but it can be explained due to the fact that it takes some reaction time for the subject to actually start the motor task and also to end it when told to do so. Moreover, the classifiers have been trained by discarding the first two seconds of task and rest periods and this for sure contributes to the worse performance when the label changes. As observed in Tables 4.1 and 4.2, the best performance for both motor execution and imagery is obtained when EEG and NIRS features are combined. The NIRS-based classifiers do not suffer from the hemodynamic delay that is usually observable in NIRS signals. This can be explained by the fact that not only the average of the signal is used as a feature,

but so is the slope, and since the signals start increasing as soon as the movement begins, the accuracy is not delayed. As shown in Fig. 4.1, then, executed movements are better classified than imagery ones, and this is reflected in the dynamic accuracy trends.

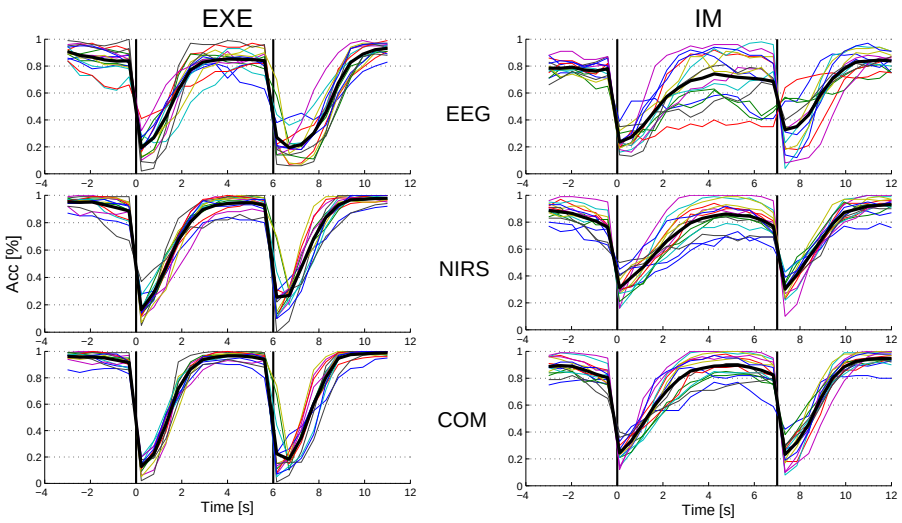


Figure 4.2: Dynamic accuracy plots for Rest - Task classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

4.2 First Approach

The first approach to classification, after having classified task with the Rest - Task classifier, consists of recognizing if the movement is right or left and then arm or hand, or vice versa.

4.2.1 Execution

Table 4.3 contains the results of the Right - Left classifiers (for the combination of features only *reg*, *noCSP*, and *CSPeeg* configurations are shown). It can be observed that neither the separate signals, nor the combination of EEG and NIRS, result in a high accuracy. The combination of all signals allows to reach an accuracy of 72.2% with *CSPeeg* configuration. For EEG signals, though, CSP significantly outperforms *noCSP* approach by around 5% of accuracy ($p<0.1$ for μ and $p<0.05$ for β), while for NIRS the best performance is achieved without the use of CSP (significantly only for HbO, $p<0.01$). The reader can notice how prevailing the phenomenon of over-fitting is; for EEG the drop between *batch* and *cv* is around 15-18% for μ and β, and for NIRS it is 20% in both cases. Regularization improves the performance (around 4% for EEG and 7-8% for NIRS signals), but definitively not enough to consider the use of single-signal-based classifiers. In Figs. B.5a, B.5b, and B.6 (Appendix B) all the box plots and the significant differences are displayed. Regarding the

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	73.8±6.5	72.1±5.5	74.8±5.7	72.8±4			
cv	57.5±11.1	54.4±8.6	54.1±5.3	52.1±4.2			
reg	61±9.8	58.7±7.3	62.2±4.3	60.9±4.6	62.2±8.9	63.1±5.8	67.1±7.4
noCSP	55.5±8.6	53.7±6.3	70.6±9.4	65±8.5	56.9±6.5	70±7.8	71.2±7.4
CSPeeg							**72.2±6.9**

Table 4.3: Right - Left Motor Execution: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

classification between arm and hand tasks, Table 4.4 shows the performance of the different classifiers. Also for the Arm - Hand classifiers, a great amount of over-fitting

is observed, which is partially corrected with regularization (drop of around 8-10% for both EEG and NIRS signals after regularization, while before regularization it is between 12-14%). The best performance is achieved by combining EEG and NIRS features, but unlike Rest - Task and Right - Left classifiers, though, CSP perform slightlt better than *noCSP*. The highest accuracy, in fact, occurs when regularized CSPs are applied to both EEG and NIRS signals (83.6%). Box plots with significant differences can be found in Figs. B.7a, B.7b, and B.8 (Appendix B).

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	78.5±6.7	74.4±5.5	85.9±6.2	83.9±6			
cv	66.3±13.5	60.1±9.5	73.9±9.7	71.2±10.7			
reg	69.3±12.3	65.8±8.1	79.4±8.7	76.7±10.7	71±11.9	80.4±9.1	**83.6±9.6**
noCSP	63±10.9	60.2±6	75.5±8.1	73.4±7.4	66.1±10	76.9±6.4	78.3±6.1
CSPeeg							79.9±7.1

Table 4.4: Arm - Hand Motor Execution: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

4.2.2 Imagery

Table 4.5 shows the accuracies obtained for Right - Left classification in motor imagery tasks. It can be observed that imagining a motor task does not yield results as high as executing a task: for Right - Left classification the maximum accuracy is obtained combining EEG and NIRS features without the use of CSP (63.2%), even though it is only slightly higher than *CSPeeg* approach, which results in 62.5% of accuracy. The average accuracy, though, is low; it would probably be not enough to

be translated in a working BCI. An important observation is that the *noCSP* approach allows to reach the maximum performance, even if the single signal classifiers have equal or lower performance with respect to the *reg* approach. This means that the information carried by the separate signals is somehow mixed when CSPs are applied. Nevertheless, this is only a suggestion, and further experiments should be conducted to prove that. Figs. B.9a, B.9b, and B.9b (Appendix B) show the box plots of all the configurations. Arm - Hand classification performance is in line with

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	75.4±7.4	72.1±5.1	68.7±5.3	68.1±2.9			
cv	58.4±11.8	52.9±7.7	48.5±3	48.8±3.2			
reg	61.9±10.1	57.6±6	55.5±1.4	55.5±2.3	62.1±8.9	56.6±2	58.7±7.8
noCSP	57.2±9.2	50.5±7.1	54.9±6.6	55.5±6	56.8±9.4	56.4±7.7	**63.4±7.5**
CSPeeg							62±7.9

Table 4.5: Right - Left Motor Imagery: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

the Right - Left one for motor imagery tasks; the maximum accuracy is 63.4% when combining all features and applying regularized CSP (same configuration as motor execution tasks). In this case, however, the accuracy would probably be too low for a BCI application. The distributions of the performances for Arm - Hand motor imagery classification are shown in Figs. B.11a, B.11b, and B.12 (Appendix B).

4.2.3 Execution vs Imagery

The comparison between execution and imagery motor tasks is presented using the configuration of features that yielded the highest performance for Right - Left and

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	73.6±3.7	70.5±4.9	67.8±3.3	67.1±2.2			
cv	55.6±6.7	50.8±6.1	48.3±2.9	49.3±2.8			
reg	58.7±5.9	55.4±4.6	55.6±1.7	54.1±2.5	59.5±6	55.5±2	**60.8±4.4**
noCSP	52.8±6.8	50.5±6.1	53.3±5.2	52.3±5.3	53.5±7.5	51.7±4.6	53.6±5.8
CSPeeg							56.5±6

Table 4.6: Arm - Hand Motor Imagery: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

Arm - Hand classifiers, i.e., the combination of all signal features (COM) and *CSPeeg* for R-L execution, *noCSP* for R-L imagery, and *reg* for both A-H execution and imagery. Fig. 4.3 displays the distributions of the accuracies for Right - Left classification divided by execution (EXE) and imagery (IM). A paired statistical test showed that they are highly significantly different ($p<0.001$), and the performance for executed movements overcomes the one for imagined ones.

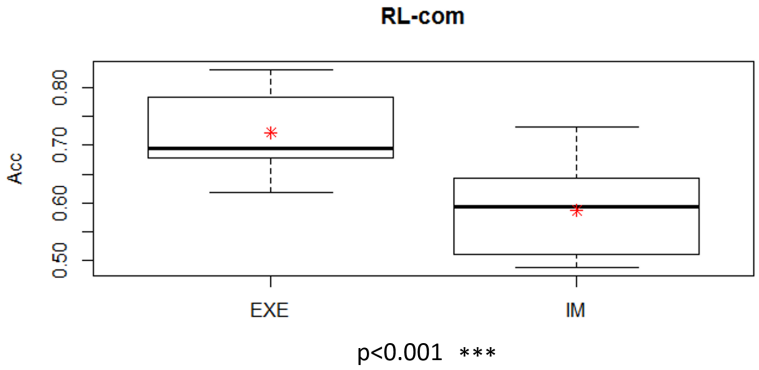


Figure 4.3: Right - Left classifier performance using *CSPeeg* approach for motor execution (EXE) and *noCSP* for motor imagery (IM).

Also, the difference for Arm - Hand classification is highly significant in favor of execution tasks. Both Right - Left and Arm - Hand classifications are much better

for execution than imagery tasks. The main reason why this occurs is most likely the lack of experience of the subject in motor imagery strategy and the inadequacy of the feedback. A more in depth discussion on motor imagery classification performance can be found in section 5.3.

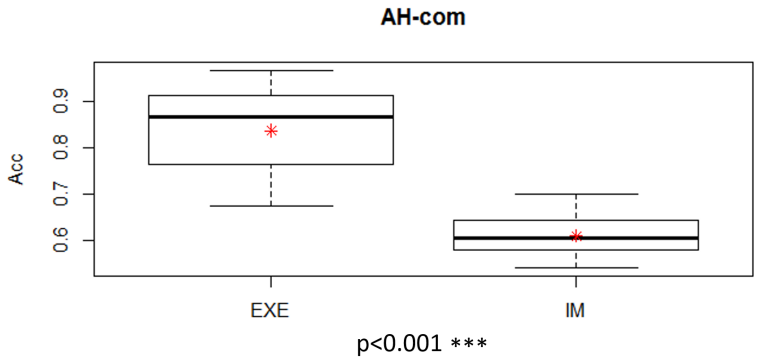


Figure 4.4: Arm - Hand classifier performance using *reg* approach for both motor execution (EXE) and motor imagery (IM).

4.2.4 Dynamic Evaluation

Fig. 4.5 displays the trend of the accuracy along the trials for Right - Left classifiers, divided between motor execution and motor imagery. The colored lines are the performances of single subjects and the black line is the average accuracy. The configuration of features is *CSPeeg* for both execution and imagery. The trend of the accuracy confirms that NIRS classifiers outperforms EEG ones for executed movements, while for imagined ones, despite the overall low performance, EEG-based classifiers seems slightly better than NIRS ones. The combination of EEG- and NIRS-derived features improves the performance in all cases. The signals outside

the vertical lines, which represent the beginning and the end of the task, are not significant, because being in the resting phase they would not be classified. The performance of Right - Left classifier is not very high for motor execution, and it seems to increase slowly, reaching a maximum around 3 s after the task. Nevertheless, it must be noticed that the readiness of NIRS classifiers is higher than the EEG ones, i.e., the slope is steeper and the accuracy raises faster. However, for a working BCI, probably the Right - Left classification would cause many errors and it should be improved, mainly by means of considering a larger dataset for training the CSP and the classifiers. A further discussion on this topic can be found in section 5.2. Regarding motor imagery, the performance is on average unacceptable (as shown in Table 4.5), being slightly over 0.5, which indicates the performance of a completely random classifier.

Fig. 4.6 shows the dynamic accuracy for the Arm - Hand classifiers. For this classification, the use of regularized CSP has shown the best results both for EEG and NIRS. For the Right - Left classification the performance for motor imagery is excessively low and not enough for a BCI; however, for motor execution, displayed in the left column of the figure, the average accuracy is high (around 80%) and steady during the entire task. The use of CSP makes the response of the NIRS-based classifier faster (it gets to a steady point around 2 s after the task visual cue), and the overall accuracy over time could certainly be translated in a working BCI. By enlarging the training set, though, the performance probably would increase.

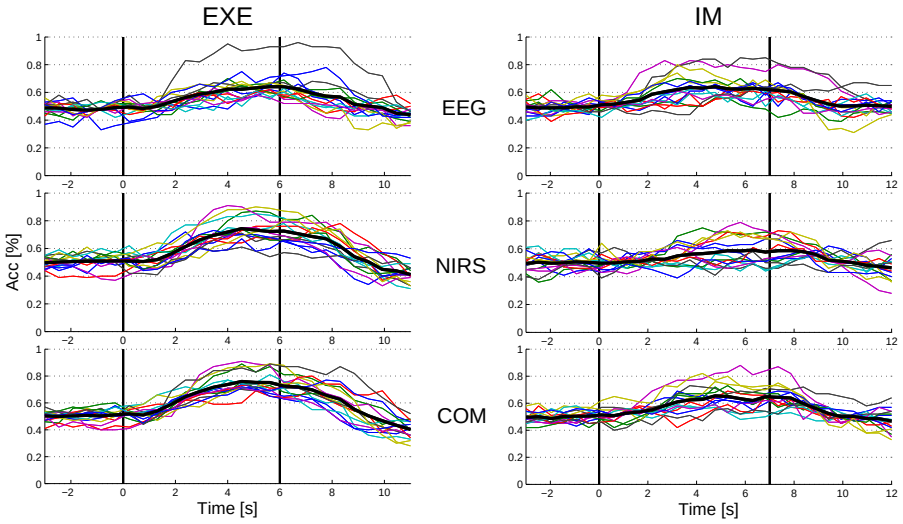


Figure 4.5: Dynamic accuracy plots for Right - Left classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

4.3 Second Approach

For the second approach the 4 different pairwise classifiers are analyzed separately, both for motor execution and motor imagery.

4.3.1 Execution

Table 4.7 and Figs. B.13a, B.21b, and B.14 (Appendix B) show the performance in Right-arm - Left-arm classification. The best performances are obtained when regularized CSP are applied for both EEG and NIRS and the features are combined

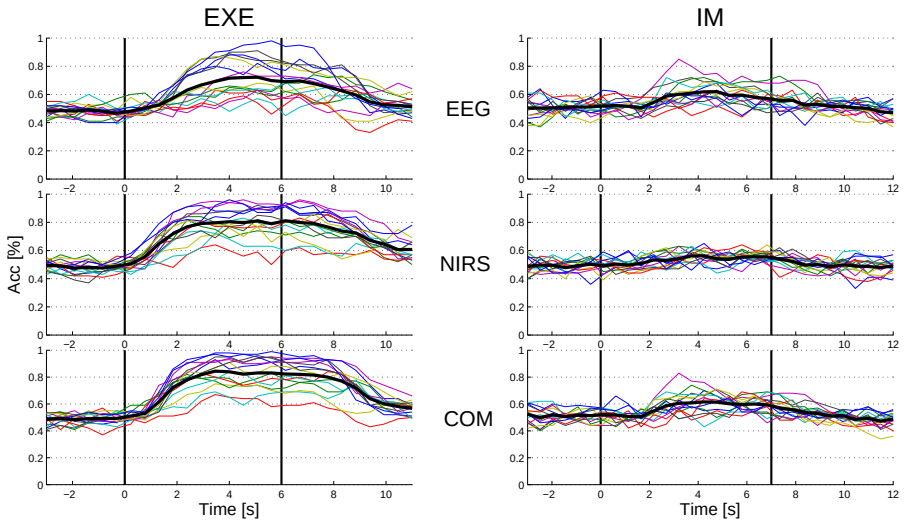


Figure 4.6: Dynamic accuracy plots for Arm - Hand classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

(72.3%). In comparison with Right - Left motor execution classifier, though, it can be observed that here the NIRS performs better when CSP are used, while by clustering together right and left tasks, the best performance is obtained in *noCSP* configuration. The highest accuracy is in line with Right - Left classification one (72.2%, in Table 4.3). Over-fitting is severe (16-22% of accuracy drop with respect to *batch*), but the performance is improved significantly (except for the μ-based classifier) by regularization (12-14% of accuracy drop with respect to *batch*). While regularization is significantly better than *noCSP* for EEG, it is not the case for NIRS, as can be inferred from the box plots.

The performance of the Right-hand - Left-hand classifier is contained in Table

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	82.8±6.1	80±5.1	82.8±5.6	79.9±4			
cv	63.6±14.5	58.9±9.1	56.4±5	53.6±4.6			
reg	69.5±11.2	66.7±7.3	67.8±6.3	66.3±5.6	69.3±10	67.7±6.7	**72.3±9.5**
noCSP	59.3±9.2	57.6±8.8	72.3±7.2	64.2±8.6	60.1±7.6	68.4±7.7	66.4±8
CSPeeg							69.9±5.6

Table 4.7: Right-arm - Left-arm Motor Execution: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

4.8, and the distributions are displayed in Figs. B.15a, B.15b, and B.16 (Appendix B). In this case the maximum accuracy (65.3%, using EEG and NIRS features with regularized CSP) is around 7% lower than the Right - Left classifier one. It can be observed that the over-fitting phenomenon is higher than in the Right-arm - Left-arm case; the accuracy drop from *batch* to *cv* is around 23-27%. Even with regularization, the over-fitting is partially corrected and the accuracy drop from *batch* to *reg* is 15-19%. Regularization significantly improves the performance for both the EEG-based and NIRS-based classifiers (from *cv* to *reg*). Also in this case, the *reg* approach is significantly better than the *noCSP* only for EEG signals, while for NIRS not only there is no significance difference. Also, the *noCSP* approach has better performance for HbO-, HbR-, and NIRS-based classifiers. When combining EEG and NIRS features, though, the best performance is obtained with the use of CSP for both the signals. Given the performances just presented, the recognition of right and left hand can be identified as the less accurate classification among the others (for both approaches); chapter 5 will develop further explanations on the topic.

Table 4.9 displays the performances for the Right-arm - Right-hand classifiers.

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	78.1±7.1	75.8±5.3	82.7±5.4	80±4			
cv	55.7±12.7	52±9.4	55.5±6.4	52.9±59.7			
reg	62.3±9.6	61±6.8	63.4±6.2	61.8±4.2	61.5±7.9	63.5±4.9	**65.3±7**
noCSP	54.9±8.5	52.7±6.9	67.1±10.9	64.5±8.3	54±8.8	63.9±10.9	61.9±9.9
CSPeeg							64.2±9.7

Table 4.8: Right-hand - Left-hand Motor Execution: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

Even for this classification the highest accuracy is reached when EEG and NIRS are combined with the *reg* approach (80%). The accuracy is only slightly lower than the Arm - Hand motor execution classifier one (83.6% in Table 4.4). Over-fitting is staunched by regularization, which limits the accuracy drop from *batch* to *reg* to 10-13%. The application of regularizing techniques improves significantly the performance for all classifiers and yields the highest accuracy both for EEG-based and NIRS-based classifiers. Box plots are shown in Figs. B.17a, B.17b, and B.18.

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	83.3±5.4	80.4±2.7	88.9±4.2	85.8±5.9			
cv	64.9±12.3	60.1±8.3	69.2±8.6	66.9±12			
reg	71.3±10.8	67.3±7.2	77.7±8.9	75.7±9.3	71.5±11	78.4±8.9	**80±9.7**
noCSP	61.6±10.5	60.8±6.6	71±10	69.8±9	65.5±8.9	68.2±10.6	67.9±8.2
CSPeeg							70±9.3

Table 4.9: Right-arm - Right-hand Motor Execution: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

The results for the Left-arm - Left-hand classifiers are shown in Table 4.10 and the box plots are displayed in Figs. B.19a, B.19b, and B.20 (Appendix B). Except for the μ-based classifier, regularization is significantly better and it stems over-fitting.

Before regularization, in fact, the drop in accuracy from *batch* is around 19-20% for all signals, while with regularization it is around 9-11%. Regularization, like for the Right-arm - Right-hand classifier case, improves the accuracy of both EEG- and NIRS-based classifiers. In fact, the best performance is reached when EEG and NIRS features are joined with regularized CSP (80.1%), being only slightly lower than Arm - Hand classifier performance.

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	82.3±5.6	78.8±5.5	88.2±5.4	87.5±4.9			
cv	63.9±16.2	58.3±12.7	69.3±9.2	68.6±11.4			
reg	71.9±12.2	66.3±9.1	79.3±8	76.5±9.4	70.5±13	79.9±9.1	**80.1±9.7**
noCSP	61.3±12.3	59.7±7.7	74.7±8.5	71.3±7.1	63±11.6	71.3±8.5	71.7±7.3
CSPeeg							72±8.8

Table 4.10: Left-arm - Left-hand Motor Execution: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

The next section will present the results of the same 4 classifiers for motor imagery tasks.

4.3.2 Imagery

For Right-arm - Left-arm motor imagery classifiers, the results are displayed in Table 4.11 and Figs. B.21a, B.21b, and B.22(Appendix B). The best performance is obtained when EEG and NIRS features are combined using regularized CSP (67.9%). The accuracy is not extremely lower than in case of motor execution; an analysis of the difference between motor execution and motor imagery performances can be found in the next subsection (4.3.3). Over-fitting (22-27% drop from *batch* to *cv*) is

severe and it is diminished with regularization down to 14-16%, from *batch* to *reg*. It can be noticed that regularization is clearly the best strategy both for EEG- and NIRS-based classifiers. In fact, *reg* performance is significantly better than *noCSP* for all classifiers.

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	81.8±6.3	77.8±6.3	78.7±5.8	75.9±4			
cv	59.1±13.4	53.4±7	51.1±4.1	50±5.2			
reg	67.1±11	63±6.8	60.8±3.7	59±2.8	67.8±9.8	59.5±2.5	**67.9±8.7**
noCSP	58.8±11.6	55.4±7.2	53.9±8	53.8±5.5	59.2±10	53.8±7.1	55.1±6.2
CSPeeg							58±7.9

Table 4.11: Right-arm - Left-arm Motor Imagery: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

Table 4.12 shows the performances of Right-arm - Right-hand motor imagery classifiers. Differently from the same classification in motor execution tasks, for motor imagery regularization performs better than *noCSP* also for NIRS (for motor execution the best performance was obtained with the *noCSP* approach). For the COM configuration, the performance of *reg* with respect to *noCSP* and *CSPeeg* is significantly higher, and it yields an accuracy of 64.3%. In terms of over-fitting, regularization significantly increases the performance with respect to *cv*; the drop from *batch* to *cv* in fact is around 23-28% and from *batch* to *reg* it is diminished to 16-18%. The box plots are shown in Figs. B.23a, B.23b, and B.24.

In Table 4.13 displays the accuracies of the classifiers for Right-arm - Right-hand motor imagery classification. Box plots are displayed in Figs B.25a, B.25b, and B.26. Regularization in this case is significantly better than the *noCSP* approach both for

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	80.1±4.4	76.6±5.1	76.9±5.2	74.6±3.6			
cv	56.8±9.9	50.9±7.6	48.5±3.5	47.6±3.8			
reg	64.1±7.6	58.6±4.1	59.2±2.5	58.9±2.6	63.2±5.9	59.3±3.4	**64.3±5.1**
noCSP	56.1±8.3	51.2±5	56.3±8.1	56.6±7.3	55.7±6.7	57±6.7	56.7±6.8
CSPeeg							58.9±4.9

Table 4.12: Right-hand - Left-hand Motor Imagery: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

EEG and NIRS. Over-fitting, before regularization, causes a drop in accuracy of around 24-27% from *batch* to *cv*, and the use of regularization makes it decrease to 16-18%. The highest performance is 63.2% and it is obtained when using regularized CSP on the combination of EEG and NIRS derived features.

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	78±4.8	76.9±4.6	78.2±4.9	75.6±4.6			
cv	53.6±4.5	52.3±4.8	52.7±4.9	48.2±4.5			
reg	62±7.4	59.6±3.5	60.2±2.6	59.2±2.6	61.8±6.3	59±3.2	**63.2±6.1**
noCSP	54.1±8.9	53±5.3	54.7±4.7	53.6±6.7	55±6.5	52.8±5.6	53.9±5.6
CSPeeg							55.4±5.1

Table 4.13: Right-arm - Right-hand Motor Imagery: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

The results for Left-arm - Left-hand motor imagery clasifiers can be found in Table 4.14 and Figs. B.27a, B.27b, and B.28. The maximum accuracy is 63.2% (exactly like Right-arm - Right-hand classification), and it occurs when EEG and NIRS features are joined in *reg* configuration. The over-fitting phenomenon can be observed in the accuracy drop from *batch* to *cv* (26-27%), and for all the classifiers the use of regularization improves significantly the performance, as it is shown in the

box plots. The use of CSP makes the performance highly significantly better than not using it for both EEG and NIRS derived classifiers and for the combination of all features.

	μ	β	HbO	HbR	EEG	NIRS	COM
batch	79.6±4.4	76±3.9	78.4±4.8	75.3±3.9			
cv	54.9±7.6	50±5	51.1±4	49.3±4.8			
reg	62.8±5.8	59.5±4.3	59.1±3.1	58±2	61.4±6.3	59.4±3.1	**63.2±5.1**
noCSP	53.4±6.9	52.5±5.3	52.5±6.9	52.1±5.9	54.1±6.5	52.2±5.6	51.8±6.3
CSPeeg							54.2±5.9

Table 4.14: Left-arm - Left-hand Motor Imagery: accuracies obtained with a 10-fold CV for classifiers trained on features derived from the 4 signals separately and with combinations of them with different approaches for CSP.

4.3.3 Execution vs Imagery

Exactly like the first classification approach analysis, the comparison between execution and imagery performance has been done on the configuration that allowed the highest accuracy. For all the 4 classifiers (RA-LA, RH-LH, RA-RH, and LA-LH) the best performance is achieved with regularized CSP using EEG- and NIRS-derived features all together (COM-*reg*).

Fig. 4.7 displays the distributions of the accuracy of the Right-arm - Left-arm classifier, on the left, and of the Right-hand - Left-hand classifier, on the right. For both classifiers, there is not a significant difference between executed and imagined tasks. This result is different than the one presented in section 4.2.3, where the performance of the Right - Left classifier was significantly higher for execution than for imagery. Although there is no significant difference for the Right-arm - Left-arm

classifiers, by looking at the distributions it is clear that the executed movements allow, on average, higher performance. Moreover, the accuracy obtained with the motor execution is almost the same as the one reached for the Right - Left classifiers from the first classification approach. Instead, the motor imagery classification for Right-arm - Left-arm is higher than the Right - Left one by around 4%. In the Right-hand - Left-hand classification accuracy, it can be noticed that the two distributions are almost the same, and there is no clear difference performance between executed and imagined movements. While the accuracy of the motor imagery classifier is more or less the same as the one obtained by the Right - Left classifier, for motor execution tasks the classification is much worse than the Right - Left classifier (Right - Left: 72.2%; Right-hand - Left-hand: 65.3%). The worse performance could be due to the small amount of data used for training and the subsequent severe over-fitting. This result also confirms that the worst performance among all the classifier is the one obtained in the Right-hand - Left-hand classification.

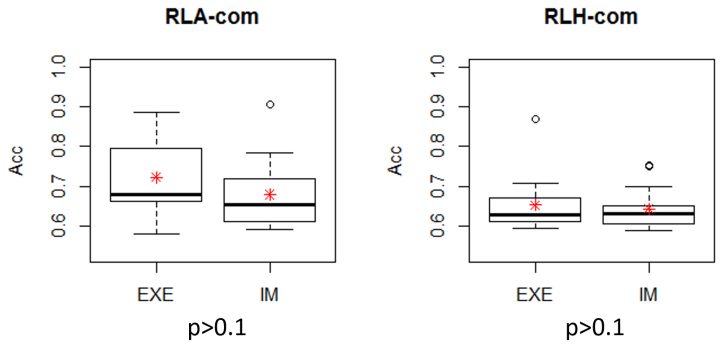


Figure 4.7: Left panel: Right-arm - Left-arm classifier performance using *reg* approach for motor execution (EXE) and motor imagery (IM). Right panel: Right-hand - Left-hand classifier performance using *reg* approach for motor execution (EXE) and motor imagery (IM).

Fig. 4.8 displays the distributions of the Right-arm - Right-hand classifier (on the left) and of the Left-arm - Left-hand one (on the right), divided by executed and imagined tasks. For both the classifiers the motor execution performance is significantly higher than the motor imagery one. Although the accuracy of imagined tasks, which is 63.2% for both the classifiers, is slightly higher than the accuracy obtained for the Arm - Hand classifier of the first approach (60.8%), the gap between motor execution and motor imagery is still wide. The low performance of the Arm - Hand recognition for imagery can be due to the fact that the mental strategies to imagine arm and hand movements is quite difficult to learn without a feedback telling the subject whether it is recognizing one or the other mental imagery. This topic is discussed widely in section 5.3.

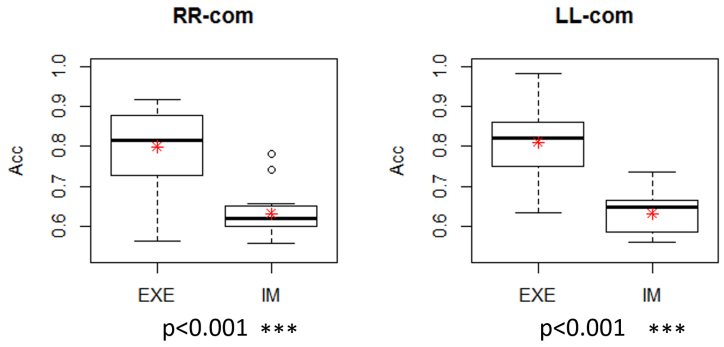


Figure 4.8: Left panel: Right-arm - Right-hand classifier performance using *reg* approach for motor execution (EXE) and motor imagery (IM). Right panel: Left-arm - Left-hand classifier performance using *reg* approach for motor execution (EXE) and motor imagery (IM).

4.3.4 Dynamic Evaluation

For all the 4 classifiers the accuracy along time over the task has been computed. In all figures the top row shows the EEG-based classifier performance, the middle row the NIRS-based one, and the bottom row represents the accuracy for a classifier based on both EEG and NIRS features. All the classifiers used are the ones trained using the regularized CSP approach, because it resulted in the best results in all cases. The left column of each figure contains the performances for motor execution, while the right column the ones for motor imagery tasks. For how the dynamic accuracy is computed, the only tract of interest is between the vertical lines, i.e., during the task. In fact the other classifiers are used only when the Rest - Task classifier outputs task.

Fig. 4.9 displays the dynamic accuracy for the Right-arm - Left-arm classifier. As observed in Tables 4.7 and 4.11, the combination of features from EEG and NIRS yields the best performance (on the bottom), and the main contributor is for sure the EEG signals, which definitely performs better than NIRS both for execution and imagery. Regarding the time response of classification, the response for EEG seems to have a highest slope than NIRS, but the combination of them outperforms both of them also in terms of speed, i.e., slope of the accuracy trend.

Fig. 4.10 shows the accuracy trend over the trial for the Right-hand - Left-hand classifier. Apart from a very good performing subject in motor execution tasks mainly due to the EEG signals (top row, left column), the EEG accuracy is mediocre, and it seems exactly like the imagery one. The NIRS-based classifier

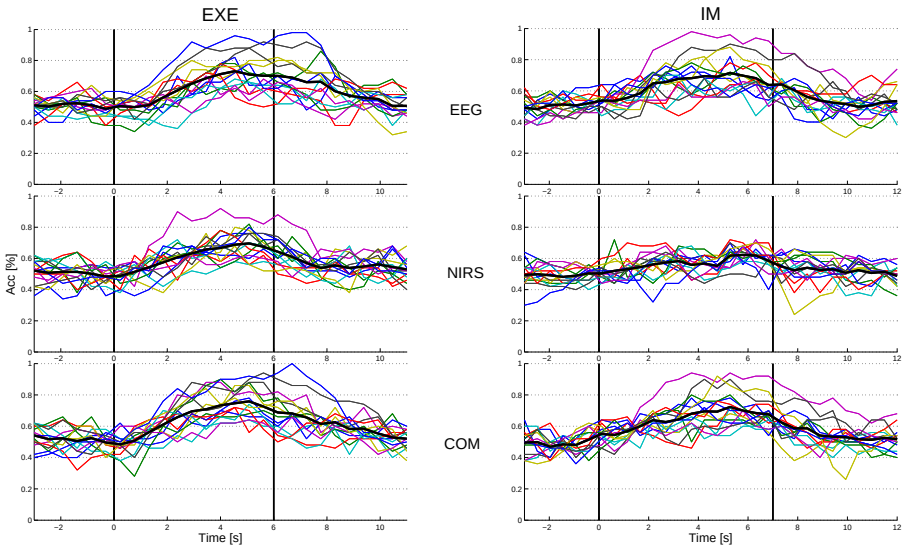


Figure 4.9: Dynamic accuracy plots for Right-arm - Left-arm classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

appears to be slightly better than the EEG-based one, and a difference between execution and imagery is more perceptible. Despite the combination of features, the trend of the accuracy for this classification seems extremely inadequate in order to recognize Right-hand and Left-hand. It should not be ignored, though, that for this second approach all the 4 classifiers work together in order to select the output class. A more comprehensive dynamic evaluation should investigate the synergy in the classification, but that can only be done with an online setup that allows the evaluation of the parameters for the classification, as introduced in section 3.6. A further discussion on the appropriateness of the current dynamic evaluation with respect to an online one can be found in section 5.4.

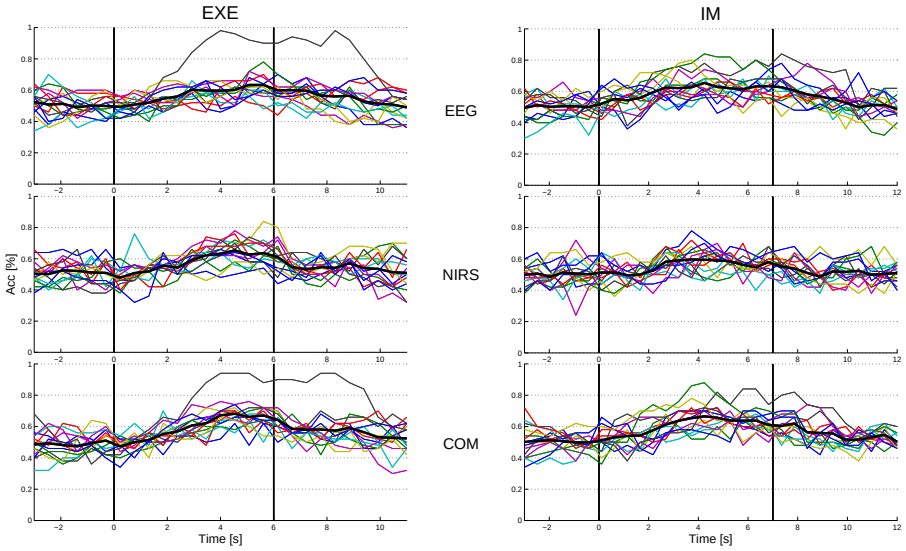


Figure 4.10: Dynamic accuracy plots for Right-hand - Left-hand classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

The performance over time of Right-arm - Right-hand classifiers for execution and imagery can be found in Fig. 4.11. The trends clearly reflects the results observed in Tables 4.9 and 4.13. There is a large difference in the results between motor execution and motor imagery, and for motor execution both EEG and NIRS contribute to the improvement in classification of the COM configuration. Some subjects clearly benefit from the combined use of EEG and NIRS: for example, for the subject represented by the red line in the EEG performance of motor execution (top row on the left) the use of EEG only would be definitely inadequate to reach a sufficient accuracy; however, when NIRS features are added, the overall accuracy gets between 70% and 80% for all the task duration. As far as readiness of response,

one can notice that the slope of the NIRS classifier is higher than the EEG slope. In the COM configuration the accuracy gets steady after around 2 seconds from when the subjects are presented the task cue. If 1 s of reaction time is accounted, then only 1 s is necessary for the classifier to perform well.

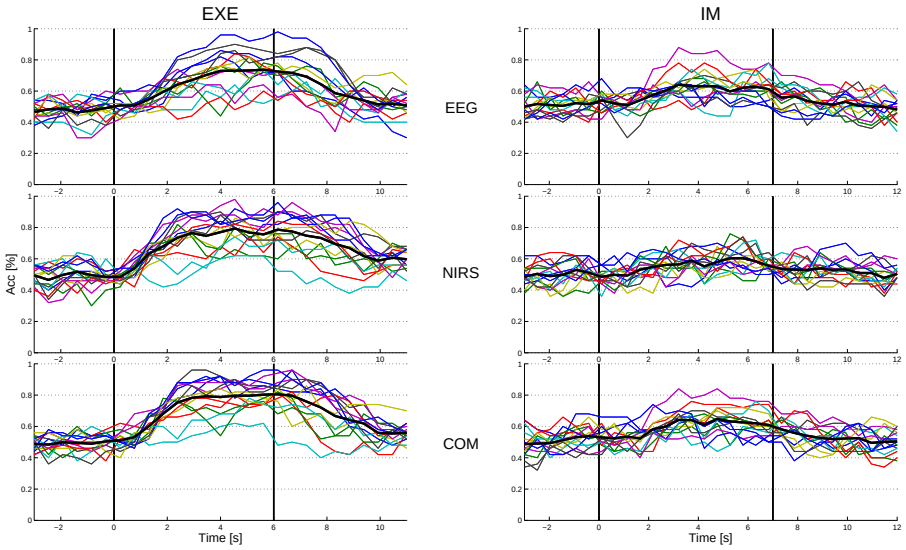


Figure 4.11: Dynamic accuracy plots for Right-arm - Right-hand classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

Fig. 4.12 contains the accuracy trend over the trial for the Left-arm - Left-hand classifiers. The performance of these classifiers is very similar to the one just described, namely, Right-arm - Right-hand. Also for this classification, in fact, a huge difference between execution and imagery performance is observed. Moreover, the EEG-NIRS combination is very beneficial for some subjects, for example the two

subjects represented by the purple lines, for which the EEG performance is quite low and fluctuating, while the NIRS performance is higher and stable. The advantages and disadvantages of an EEG-NIRS combination will be dealt in detail in section 5.1. Lastly, the response of the classifier is quite fast, with the COM configuration that reaches a steady state after 2-2.5 s from the task cue.

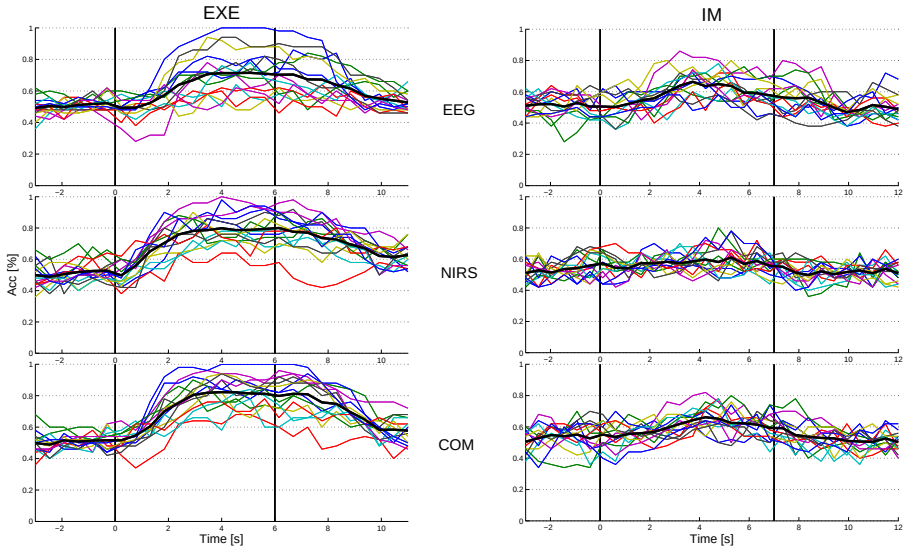


Figure 4.12: Dynamic accuracy plots for Left-arm - Left-hand classifiers: the first column shows executed tasks, the second one imagined ones. The first row represents the accuracy obtained with EEG features, the second row with NIRS ones, and the third one the combination of EEG and NIRS features. The x axis is time in seconds and the y axis the accuracy (from 0 to 1). The vertical lines delimit the task phase of the trial. Colored lines represent single subjects, the black thick line is the average of all subjects.

Chapter 5

Discussion

The discussion chapter is divided in multiple sections, each of them treating a different topic that emerged during the study.

5.1 EEG-NIRS combination

The results show that the combination of EEG and NIRS for BCI purposes can enhance the classification accuracy, especially when a low-resolution EEG configuration is used. In the current study, for example, the EEG spatial resolution chosen in the experimental setup (21 measuring channels distributed from Frontal to Parietal 10-20 system parallels, see Fig. 3.5) was lower than many other studies of SMR-based BCIs involving EEG only (55 in [5], 60 in [43], 118 in [29]), or in combination with NIRS (37 electrodes in [13]). Due to the volume conduction of the tissues between

the cortex, i.e., the main source of activity captured by EEG, and the EEG electrodes (such as cerebro-spinal fluid, skull, and scalp), the activity measured by a single electrode is much smaller than the one that could be measured on the brain membrane (up to 1000 time smaller, from mV to μV). This drop of potential makes it very difficult to localize the different sources from the EEG signals [59] and it calls for the use of high-resolution EEG, in which the elevate number of sampling points on the scalp makes the localization much more accurate. On the other hand the use of high-resolution configurations makes the translation of BCIs into the clinical environment more complex.

The NIRS experimental setup can be considered as complex as the EEG one, consisting of 24 optodes (12 sources and 12 detectors) with respect to the 23 electrodes (21 measuring electrodes, a GND electrode, and a Reference electrode). In most cases, with this configuration, the NIRS-based classifiers allowed higher performances to be reached than the EEG alone, especially for motor execution (motor imagery is discussed aside in section 5.3). This was the case for Rest - Task (without CSP), Right - Left (without CSP), Arm - Hand (using CSP), Right-arm - Right-hand and Left-arm - Left-hand (using CSP) classifiers. Only for Right-arm - Left-arm and Right-hand - Left-hand classification the EEG and NIRS performance was about the same, as shown in Tables 4.7 and 4.8. The main concern about the use of NIRS is that the hemodynamic signals are slow and it takes some seconds for them to reach their peak. This affects on classification accuracy, which in [13] reaches its peak around 6-7 seconds after the movement onset. The results obtained in this study, though, do not show such a long delay of the NIRS-based classifiers, and this is

probably because of the features used (this issue is explored in section 5.4).

The better outcome of the NIRS with respect to the EEG can be tackled in terms of spatial resolution and source localization. The classification of different motor tasks, or whatsoever other kind of activities, can be seen as the recognition of patterns in the signals generated by different sources localized along the cortex. For example, having in mind the Penfield's homunculus in Fig. 2.5, the activity that generates an arm movements will be more central than the one causing a hand movement. As explained above, it is very difficult to localize those sources from a low-resolution EEG, and, in other words, the activity measured for the arm and the hand movements will be extremely similar (of course this is very subject-dependent and it accounts for different electric conduction conditions, brain patterns, etc.). The NIRS signals, on the contrary, do not suffer because of volume conduction, as the measurement is performed around the middle point between the source and the detector location. What each NIRS measurement channel picks up, then, is the oxy- an deoxy- hemoglobin concentration estimation over the light pathway between the source and the detector, and nothing else. It is clear then that for localizing the source of a brain activity, a higher resolution EEG signal is needed in order to yield a good BCI performance; however, the combination with the NIRS definitely improves the accuracy of the system and can be beneficial in the development of robust BCIs.

One of the key factors when dealing with BCI research and development is the clinical translation, e.g., how easy would be for medical doctors to include a BCI in a rehabilitation therapy. One of the main issue about current BCI is their complexity from the setup point of view. If every time it takes one hour or more to setup a system

for a patient, then it would be extremely unlikely to be used in the clinical world. The challenge is to try to reduce the setup in order to diminish the time to prepare the BCI. Another layer of complexity is added by how the BCI algorithm needs to be trained with the subject's data. A training dataset, in fact, is needed to learn each subject's characteristics, and a smaller setup may result in the need of a bigger training dataset. The combination of EEG and NIRS for sure is worse with respect to a single system under this perspective, because not one system, but two need to be setup. The time required can vary, not only because of the "density" of the setup, but also and especially because of the technology used for the two systems. For the EEG, dry electrodes are being used more and more, and they provide measurements comparable with standard clinical wet electrodes [57, 15]. They can be equipped in measurement systems that can be setup in around 2 minutes [47]. With such a technology the EEG setup could be redesigned in order to have a higher spatial resolution, which would certainly improve the performance. Regarding the NIRS, in the current study the setup has been very time consuming and sometimes troubled, depending on the subjects. The main problem, as explained in section 3.1.1, is represented by the hair, that needs to be moved aside for every optode in order to guarantee the optical coupling between the optode ans the scalp. On possible solution, that is also being developed by NIRx, is to place a holder on every optode with a spring inside that pushes it down, making the optical coupling easier. With this technology the setup time for NIRS would diminish dramatically; time would also decrease because in some cases the use of gel to move the hair is not needed at all.

The combined use of EEG and NIRS has been seen to be beneficial, increasing the performance of each of the two systems used separately. When combining features derived from the two measurement systems, though, a further step should be added to maximize the performance of the classifiers: feature selection. Feature selection is a technique to reduce the dimensions of the problem and selects the most significant features. The selection of features indeed simplifies the model, and it can enhance the generalization by reducing over-fitting [19].

Another important topic that contributes massively on the final performance, because of the methods that have been used, is the experimental design, and in particular the amount of data available for training.

5.2 Experimental Design, CSP, and Over-fitting

As suggested in the previous sections, the performance of the classification cannot be detached from the amount of available data. Clearly, a bigger dataset for training the models would result in a better performance, because intuitively more examples are provided to learn from. On the other hand, collecting data is extremely time consuming, and it can be boring for the subject if it takes too long. For the current experiment, since it has been decided to acquire data of different kinds (different tasks and both motor execution and motor imagery) instead of focusing on a restricted problem (e.g.,the recognition of only executed right-left hand movements), the training datasets are small, i.e., only 25 trials per class for executed and imagined

movements in a total of 200 trials. Of course, in case of the restricted problem described before, there could be 100 trials for each class, and that would have certainly resulted in a better performance. Moreover, since the aim of the study was more exploratory than exploiting, it has been opted for the solution that allowed us to have more classes for both motor execution and imagery, maintaining a bearable experimental time for the subjects. In the trade-off between number of classes/conditions and numbers of trials, the first one has been preferred. In the literature the number of trials per class is usually higher than the one used in the current work, for example [43] used 60 trials per class, [13] had 48 executed movements and 100 imagined trials per class, and 140 trials per class were used in [29].

After having provided the motivation behind the experimental design and procedure, let us analyze the drawbacks of such choices. The main disadvantage of having a small training dataset is the use of CSP. CSPs are known to over-fit the data in case of small datasets, as widely treated in section 3.2.2. The over-fitting is mainly due to the fact that the CSP method is powerful, in the sense that it can make use of many degrees of freedom in order to find the optimal filters that increase the separability of the training data. When the optimized filters excessively adhere on the data used for training, they perform much worse on testing data not used for training and never "seen" by the algorithm.

The amount of over-fitting is severe in the current study, as seen in the drop accuracy that is observed between the *batch* CSP approach (CSP computed with all the data before CV) and the *cv* one (CSP computed on training data within CV). In the Results chapter it can be noticed that the accuracy drop in every classification

is always between 12%, in the best cases, and up to 30% in the worst ones. The phenomenon is, in most cases, more prevalent when CSP is applied on NIRS signals. This is because in case of NIRS the estimation of covariance matrices, on which CSPs are built, is done on fewer time points with respect to the EEG signal, resulting in a higher bias in the estimation. In order to limit the over-fitting phenomenon, regularized CSPs have been used. As described in section 3.2.3, the idea is to add *a-priori* information to the covariance estimation based on the rest of the population. Clearly, regularization also would definitely benefit from having a bigger training dataset. In fact, the covariance matrices of each subject would be better estimated, resulting in a generic covariance (see Eq. 3.17) which better represents the behavior of the all population. Moreover, one cannot expect that regularization would magically fix everything from such a small dataset. No doubt it improves the performance drastically in some cases, but the solution for a real improvement of the performance can only be found along with a bigger initial dataset for each class for each subject.

An increased *a-priori* knowledge of the brain activity, which could be acquired by collecting more data, not only would definitely enhance the regularization performance, but it could also be used for another purpose of extreme importance. The current study tried to also collect data on motor imagery tasks. Nevertheless, the imagination of movements is a very difficult task at the beginning, and it certainly needs practice. An initial dataset on motor executed and/or motor imagery tasks can indeed be used to train general classifiers that can be used to give feedback to the subject and make the get used to the BCI system, as explained in the next section.

5.3 Motor Imagery and Execution

It has been shown that when a movement is imagined rather than executed, the brain activity patterns are similar to the ones observed when the movement is actually executed [45]. Motor imagery, though, is a complex mental strategy and it needs to be learnt. In particular, the brain activity that most resembles the motor execution patterns occur when the motor imagination is kinesthetic [33, 55], i.e., when the subject is able to picture his/her muscles contracting, even if they are not. Graimann and Pfurtscheller in [17] state that training is necessary for motor imagery, and it takes at least from 1 to 4 hours on average for a subject to be able to control with good performance a 2-choice BCI. For many subjects a longer training session is needed to achieve a good control. In the current study there has not been the possibility to train the subjects sufficiently before motor imagery tasks. The only training, as described in setion 3.1.3, lasted about 10-20 minutes and allowed the subject to familiarize with the feedback, which could respond based on ERD/ERS in the EEG signals. The feedback, therefore, told the subject whether an imagined movement was perceived or the subject was resting. Such a feedback is probably too naive to really make the subject practice and learn the best motor imagery strategy. The feedback, in fact, is essential for understanding how to perform a certain mental activity, especially when multiple motor tasks need to be imagined. This was the case in this study, where the subjects were asked to imagine the movement of right and left arms and hands. A proper feedback would communicate to the subject which of the classes is being recognized, e.g., right-hand, left-hand, right-arm, or left-arm, so that the user could adjust his/her mental strategy in order to meet

what the classifier "expects". As introduced in section 2.1, in fact, the feedback is a founding part of a BCI and it allows the subject to learn and adapt to the system. For this work, though, the development of a complex feedback system was not feasible. In fact, a subject-independent feedback is usually built on a great amount of data (usually EEG) previously available. In particular the entire set of labeled data is used to train a generic classifier that will tell the subject what is the behavior of the average population. Sometimes, as in [63], the experimental procedure is even split into multiple sessions, and the feedback passes from being completely subject-independent, to include subject-dependent features that result in a customized feedback for every subject. Due to the unavailability of previous data on which building up a more complex feedback, a SMR feedback has been developed. The lack of information provided by the feedback and the complete inexperience of the subjects on motor imagery tasks can be probably addressed as the main reasons for the motor imagery failure. The performance of the classifiers for motor imagery tasks was in fact below an acceptable level for a BCI application (around 70%) for every classifier trained, except for one; the only case in which motor imagery resulted in a relatively high accuracy was for the Rest - Task classification. Clearly, this classification can be considered easier with respect to the others, but such a high accuracy (85.8%) could be in part due to the feedback used. As stated before, the feedback gave information about Rest - Task to the subjects, who may have learned how the system responded and how to adjust their mental strategy to meet the visual positive feedback (the feedback circle turned green), rather than the negative one (the feedback circle turned red). Because of the overall low accuracy of motor

imagery mental strategy due to the reasons just explained, in the following sections its performance is neglected and the discussion will deal only with motor execution tasks.

Lastly, feedback is not only extremely important during data collection, but it is the *end effector* of every BCI system. In order to actually measure the effectiveness of a BCI, the user should be able to observe the output of the classification and get used to the control command provided by the system. The only way to evaluate a BCI as a whole and complex is to develop a real-time feedback session.

5.4 Dynamic and Real-time Evaluation

In the Results chapter a dynamic evaluation of each classifier has been proposed. The dynamic evaluation, though, is performed offline, and even if it adds a new point of view on the classifiers' performance, it does not take into account the whole complexity of a real-time evaluation. The dynamic accuracy evaluation that has been implemented has certainly been helpful in drawing conclusions about some aspects of the classification. An important point that emerged from the dynamic evaluation is about the response of NIRS-based classifiers. Differently from what has been shown in [13], i.e. NIRS classification has a lag in the accuracy peak of 6-7 s, the proposed methods used to classify using NIRS did not suffer from the same delay. The reason why the delay is not as severe as previous results can be found in the set of features computed from the NIRS signals. Both the regularized CSP approach and the use of HbO and HbR slopes in combination with the averages seems promising in advancing

the NIRS as a valuable alternative for BCI purposes, with or without the use of EEG. In particular, for Arm - Hand, Right-arm - Right-hand, and Left-arm - Left-hand classifiers, for which regularized CSP have been adopted, the readiness of the NIRS response allows to reach a steady classification accuracy after around 2 s from the presentation of the task cue, as shown in Figs. 4.6, 4.11, and 4.12. Even when CSP are not used to process the NIRS signals before feature extraction, the inclusion of HbO and HbR slopes as features results in classifiers that do not present a long delay in the reaching of the accuracy peak: this is the case of the Rest - Task and Right - Left classifiers. For the Rest - Task classifiers the accuracy peak is reached after about 2 s from the cue appearance (Fig. 4.2), while for the Right - Left classifier (Fig. 4.5), the accuracy takes about 3-3.5 s to get to a steady point. The performance of the Right-arm - Left-arm and the Right-hand - Left-hand is not considered here because it is extremely biased by the over-fitting of the CSP.

Whereas the dynamic evaluation proposed allowed to observe helpful and interesting hints about the classification dynamics, it cannot be regarded as sufficient to evaluate the entire BCI system. First of all, the dynamic accuracy has been done on the single classifiers separately, without combining to output only one of the 4 classes. The translation of the classifiers output into a BCI command, in fact, is a delicate matter. For the first approach of classification, described in section 3.4, the BCI command can be assigned trivially by outputting the classified class at every time segment, i.e., having one new and independent output every 0.5 s. For example, if at a certain time point the Rest - Task classifiers predicts task, the Right - Left one predicts right, and the Arm - Hand predicts arm, then the right-arm class is used as

BCI command for that time segment. The problem of such an approach is mainly due to the high number of mistakes that the combination of classifiers will commit, which can be due to an error in one of each of the 3 classifiers. One way to avoid misclassification and develop a more robust BCI could be filtering the last 4 classification outputs, for example, with a majority vote approach. Practically, if the Rest - Task classifier output task/rest/task/task, the Right - Left one right/-/left/right, and the Arm - Hand predicted arm/-/arm/hand, then the final prediction would be right-arm, because 3 of the 4 last predictions were task, 2 of the 3 were right, and 2 of the 3 were hand. The drawback of this approach is that the responsiveness of the system to a new class would be slower (it acts as a low-pass filter). Another possibility to clean out the BCI output could be to consider the confidence of each classification. This approach is applicable to both of the classification paradigms in section 3.4. An output command can be delivered only when the classifiers are "sure" about their prediction, i.e., when the likelihood of of the prediction being correct (expressed in Eq. 3.22) is above a certain threshold for all the classifications. In case of uncertainty, the BCI could stop and reject the classification, waiting for more confident predictions.This technique would, though, worsen the performance in terms of time.

What has just been stated should make clear that a real-time classification with feedback is necessary for a BCI system to be entirely evaluated. An online setup would give the possibility to decide which of the classification approaches and BCI output translations described above have the best online performance. The Information Transfer Rate (ITR in Eq. 3.23) could be used as term of reference to evaluate in

a complete sense the entire system, including information about accuracy, complexity of the classification, and time, which has been ignored so far. Moreover, being composed of a highly adaptive system as the human brain, the online use of the BCI would definitely become better performing as the user experiences it, and the effect of "practice" could be investigated as well.

5.5 Classification Approaches: Which One is the Best?

As described in section 3.4, two different approaches were investigated for classification. The first approach made use of 3 classifiers (Rest - Task, Right - Left, and Arm - Hand); the second approach consisted of 5 different classifiers (Rest - Task, Right-arm - Left-arm, Right-hand - Left-hand, Right-arm - Right-hand, and Left-arm - Left-hand). What is the best one?

From the performance tables displayed in the Results chapter, the best method seems to be the first one, composed by Rest - Task, Right - Left and Arm - Hand classifiers. Although the small amount of trials used to estimate the CSP and the classifiers, the accuracies obtained for all 3 classifiers, on average, is above 70%: i.e. 94.2% for Rest - Task, 72.2% for Right - Left, and 83.6% for Arm - Hand. The enhanced performance of the first approach with respect to the second one, which consists of 4 pairwise classification between the different classes, can be explained in terms of number of training samples and over-fitting. The first approach, in fact,

merges two classes into a single one for each classification, doubling the training set size. This is the reason why over-fitting seems to be prevalent in the second approach, in which a drop from *batch* to *cv* of around 16-27% is observed, rather than the first one, where the drop is around 12-18%. On the other hand it would be interesting to evaluate the second approach having the double amount of training data. If the performance of the single classifiers would increase and reach the one of the first approach ones, then probably the second approach would be more reliable, because the information provided by the different classifiers could be synergistically combined to obtain a more robust classification. The synergy of the classifiers used in the second approach could be evaluated also offline, deciding a rule to arrange the confidence of the predictions of the single classifications into one single prediction. Nevertheless, an online evaluation would be much more reliable and truthful and it should be used to really establish which one of the approaches works best.

5.6 Future Development

In the previous sections of this chapter, several new ideas and possible developments have been proposed to tackle different issues that have emerged during the work. Let us list all the future development and possible improvements in a clear and concise way:

- Investigate a smarter combination of features for hybrid EEG-NIRS classifiers involving a feature selection methods

- Tackle the over-fitting problem by designing an experimental procedure that allows to have an increased number of trials for each class and evaluate and quantify the benefit in the classification accuracy
- Design and build a generic classifier based on the data collected for this study to develop a more complex feedback system and investigate the performance of motor imagery tasks
- Develop a real-time setup for the online evaluation of the BCI, as outlined in section 3.6
- Evaluate the performance of classification online

5.7 Conclusions

In conclusion, the work presented in this thesis has been very challenging and fascinating. Let us sum up the main points in terms of methods used and results obtained.

The project has been broad and it included different activities. First of all, the experimental setup needed to be chosen and the procedure defined. The setup consisted of 21 EEG and 34 NIRS channels (the latter composed by 12 sources and 12 detectors) distributed along the motor cortex, between the F and the P parallel of the international 10-20 system. The data were collected from 15 healthy subjects (all males). Each subject was asked to perform 4 different executed and imagined motor tasks (right arm, right hand, left arm, and left hand) and 25 trials per class

per condition were recorded. During imagery tasks, subjects were given a real-time feedback based on EEG ERD/ERS.

The recorded data were pre-processed and synchronized. Features were extracted using regularized CSP for EEG, while for NIRS both regularized CSP and averages and slopes of each channel were separately investigated. As a classifier, LDA was chosen for its simplicity and its robustness against over-fitting. Performances were evaluated both in terms of static accuracy (correct predictions / total predictions) and by computing a dynamic accuracy during the trial.

Results showed that a hybrid approach combining EEG and NIRS measurements enhances the performance of the classifiers in all cases. NIRS accuracy outperformed EEG accuracy most of the times, mainly because CSP tends to over-fitting the training data when the dataset is small. Over-fitting was, in fact, one of the main issues encountered in the study, and it was so severe because of the experimental setup and procedure previously chosen. An asynchronous BCI appeared to be feasible because of the motor execution performance (94.2%) in the Rest - Task classification. Motor execution on average allowed to reach acceptable accuracies, possibly translatable in a real-time BCI. Motor imagery paradigm, instead, did not yield sufficient results, probably because of the total inexperience of the subjects in such activity and for the simplicity of the provided feedback. Two approaches have been proposed for the 4-class classification: the results obtained with the current setup suggested that the best one is the first one (consisting of classifying Rest - Task, Right - Left, and Arm - Hand), which yielded an accuracy of 94.2% for Rest - Task, 72.2% for Right - Left, and 83.6% for Arm - Hand.

The wish of the author is that the work done will be helpful in the fast-growing field of the BCI research and that hopefully within a few years BCI systems will be translatable systematically in the clinical environment or in the houses of impaired patients and improve life conditions of many people.

Appendix A

Online Normalization

Mean Estimation

Starting from the optimal mean estimation, having N samples x_i, with $i = 1, 2, ..., N$:

$$\hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

the sum over the N samples can be split in the sum from 1 to $N-1$ plus the last sample available, x_N:

$$\hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} x_i = \frac{1}{N} \left(\sum_{i=1}^{N-1} x_i + x_N \right) = \frac{1}{N} \sum_{i=1}^{N-1} x_i + \frac{1}{N} x_N$$

The only variables needed to apply this algorithm are: N, which counts the number of samples received online starting from 1, and $\sum_{i=1}^{N-1} x_i$, which is an accumulator

that sums up all the values received from the first ($i = 1$) to the one before he current one ($i = N - 1$). The variance estimation uses the estimate of the $\hat{\mu}$ parameter. It can be noticed that the sum is split into one part containing only values from the *past* ($\sum_{i=1}^{N-1} x_i$) and a part with the *current* or *update* value (x_N).

Variance Estimation

The variance estimation is based on the expansion of the square in the correct variance formula and, as for the mean, on the splitting of the sum of all values in a *past* and a *current* part.

$$\hat{\sigma^2} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\mu})^2 = \frac{1}{N-1}\left[\sum_{i=1}^{N-1}(x_i - \hat{\mu})^2 + (x_N - \hat{\mu})^2\right]$$

By expanding the square in the sum:

$$\hat{\sigma^2} = \frac{1}{N-1}\sum_{i=1}^{N-1} x_i^2 + \frac{N-1}{N-1}\hat{\mu}^2 - \frac{2\hat{\mu}}{N-1}\sum_{i=1}^{N-1} x_i + \frac{(x_N - \hat{\mu})^2}{N-1}$$

which simplified bring to:

$$\hat{\sigma^2} = \frac{1}{N-1}\left(\sum_{i=1}^{N-1} x_i^2 - 2\hat{\mu}\sum_{i=1}^{N-1} x_i\right) + \hat{\mu}^2 + \frac{(x_N - \hat{\mu})^2}{N-1}$$

The variance can be correctly estimated online by means of 3 variables that are updated every time a new sample is received: N and $\sum_{i=1}^{N-1} x_i$ are the same as the mean estimation, and $\sum_{i=1}^{N-1} x_i^2$, that accumulates the squared values from $i = 1$ to

$i = N-1$. Also for the variance estimation it is clear the division between a *past* part ($\frac{1}{N-1}\left(\sum_{i=1}^{N-1} x_i^2 - 2\hat{\mu}\sum_{i=1}^{N-1} x_i\right) + \hat{\mu}$), and a *current* part ($\frac{(x_N-\hat{\mu})^2}{N-1}$), which represents the update.

The estimation of variance must follow the estimation of mean, because it makes use of the current mean parameter $\hat{\mu}$. After mean and variance are updated, the count of the samples is increased ($N = N + 1$), and the current value has to be accumulated in $\sum_{i=1}^{N-1} x_i$ and $\sum_{i=1}^{N-1} x_i^2$.

Online normalization is used in the real-time C++ feedback application to normalize:

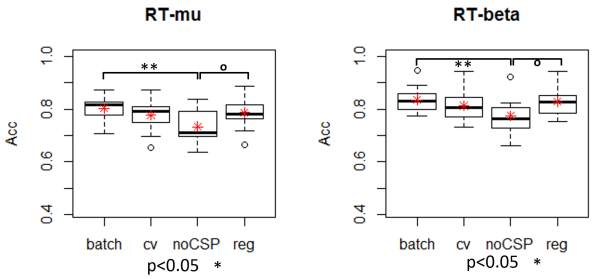
- the EEG signals from every channel, after they have been filtered
- the output values (band-powers in μ and β), to provide a standardized feedback to all subjects

Appendix B

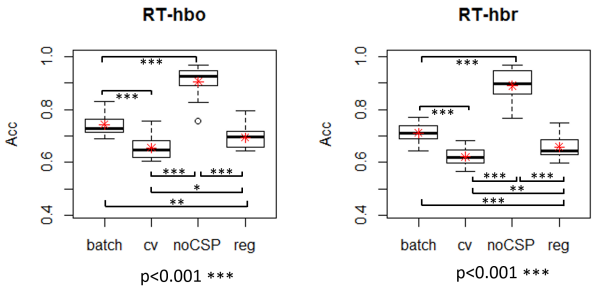
Box Plots and Significance

Box plots and significance of statistical tests performed. In the following figures the symbols refers to:

o	$p < 0.1$
*	$p < 0.05$
**	$p < 0.01$
***	$p < 0.001$



(a)



(b)

Figure B.1: Rest - Task Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

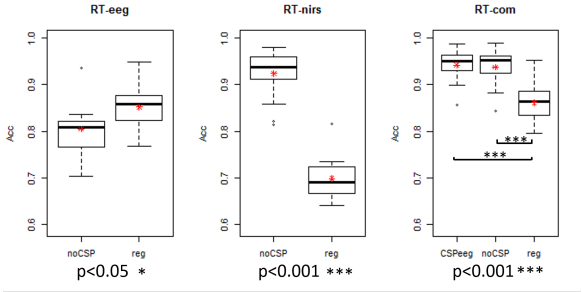
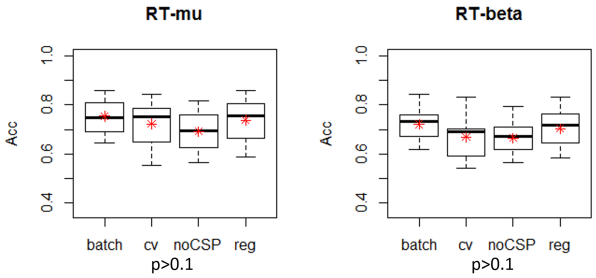
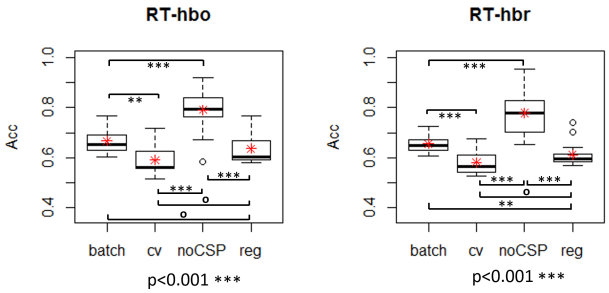


Figure B.2: Rest - Task Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.3: Rest - Task Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

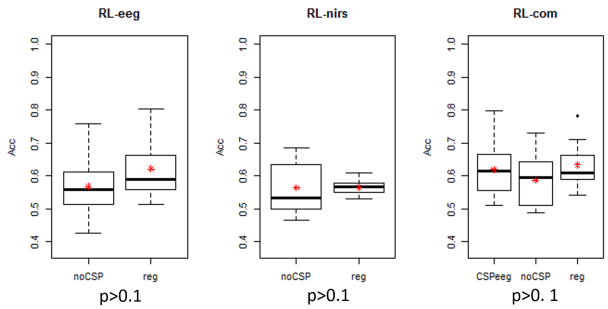
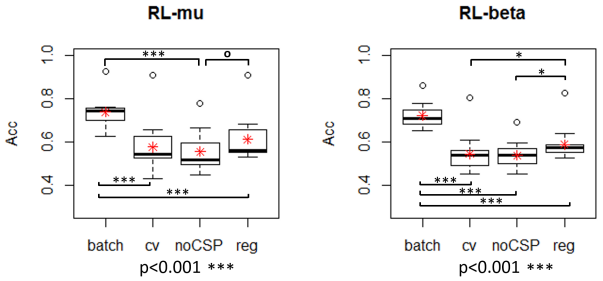
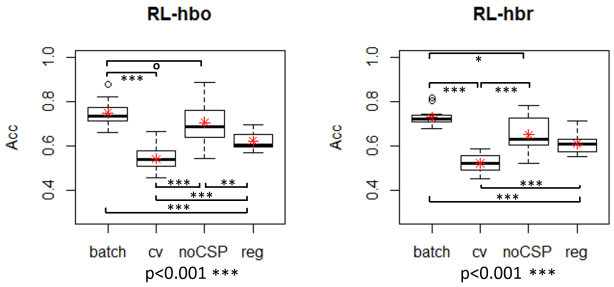


Figure B.4: Rest - Task Motor Imagery: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.5: Right - Left Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

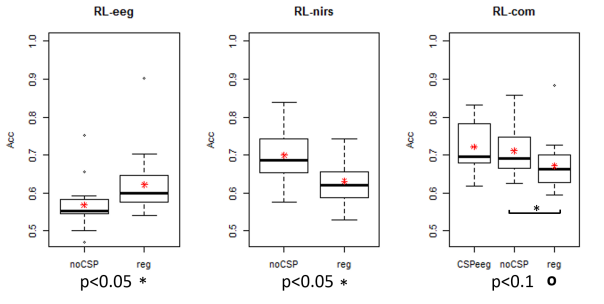
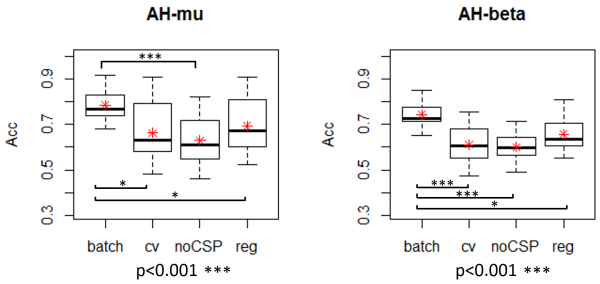
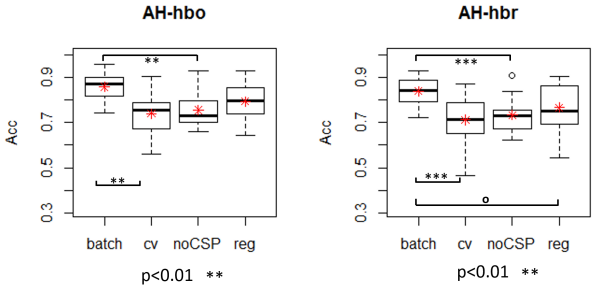


Figure B.6: Right - Left Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.7: Arm - Hand Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

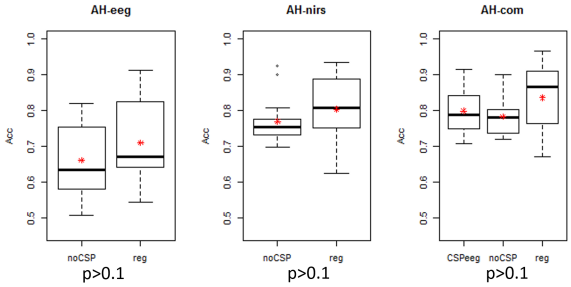
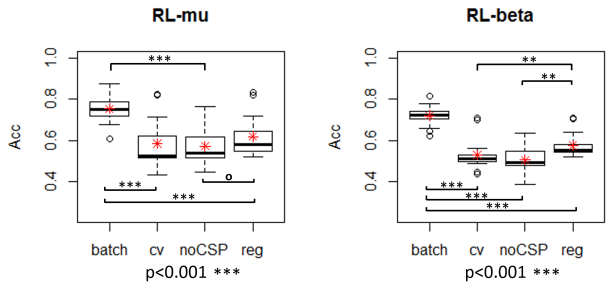
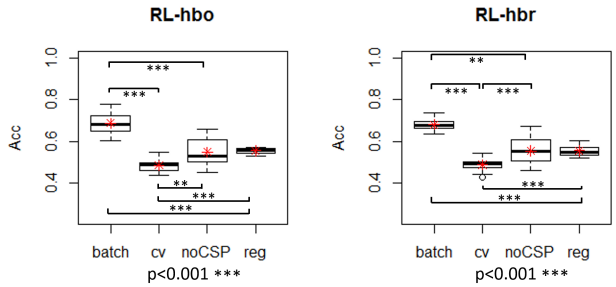


Figure B.8: Arm - Hand Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.9: Right - Left Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

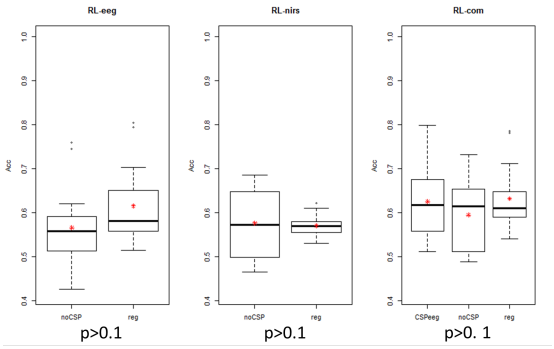
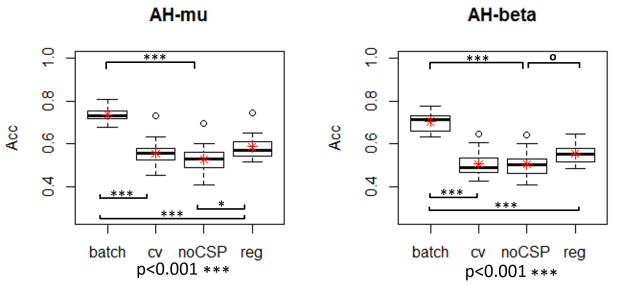
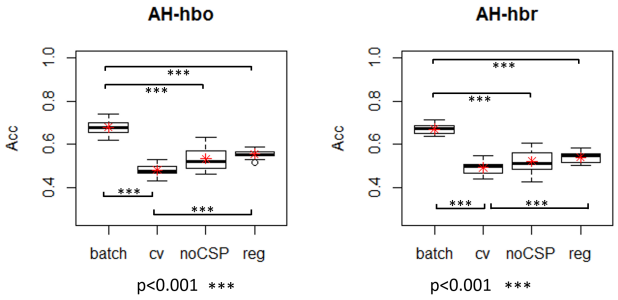


Figure B.10: Right - Left Motor Imagery: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.11: Arm - Hand Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

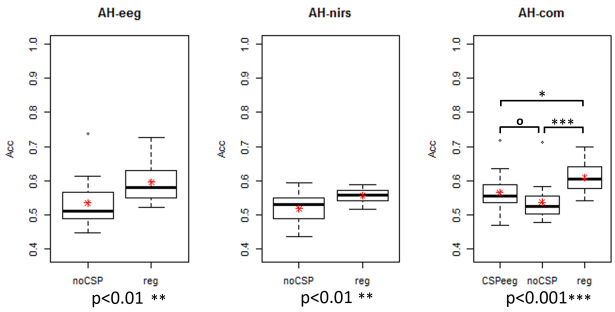
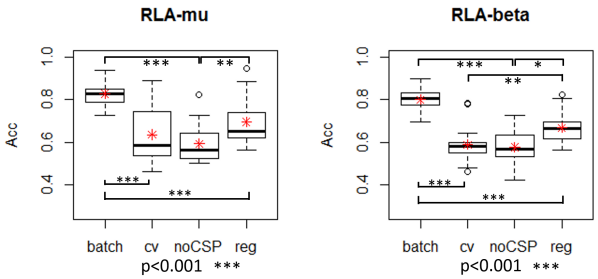
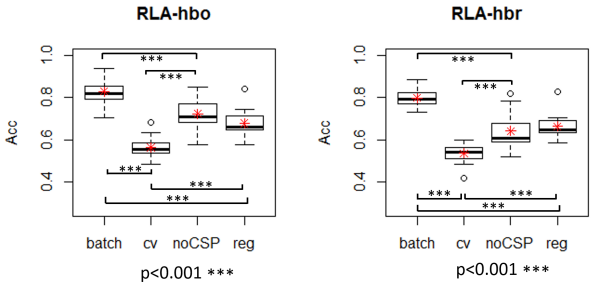


Figure B.12: Arm - Hand Motor Imagery: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.13: Right-arm - Left-arm Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

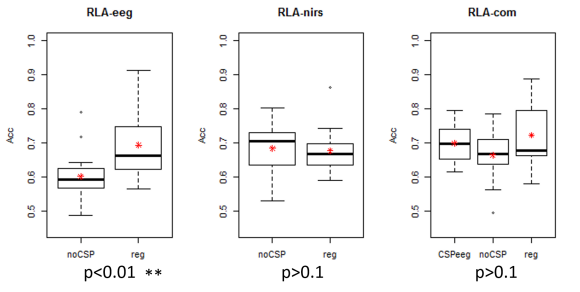
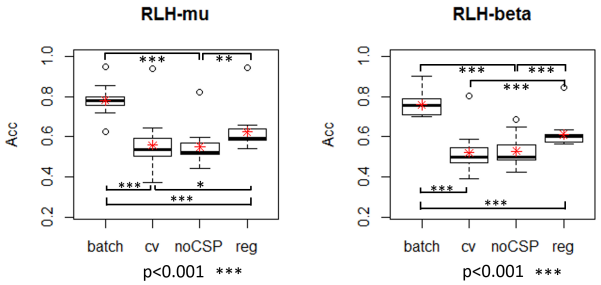
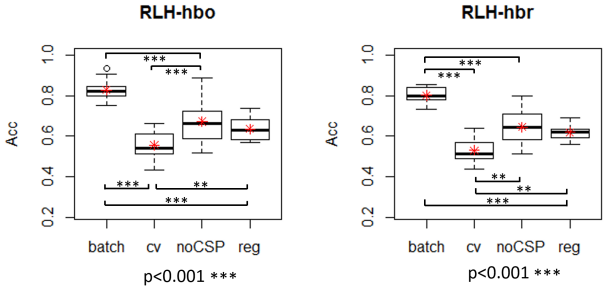


Figure B.14: Right-arm - Left-arm Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.15: Right-hand - Left-hand Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

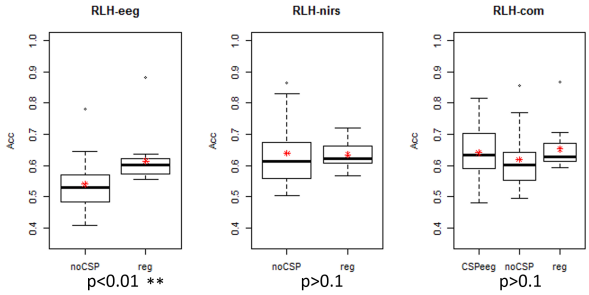
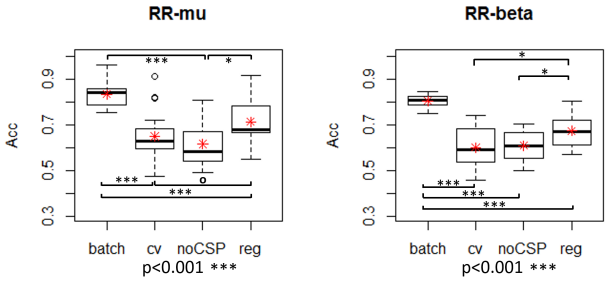
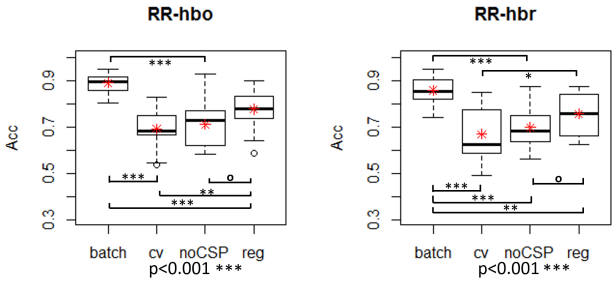


Figure B.16: Right-hand - Left-hand Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.17: Right-arm - Right-hand Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

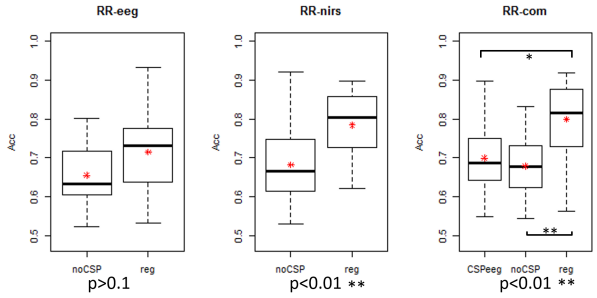
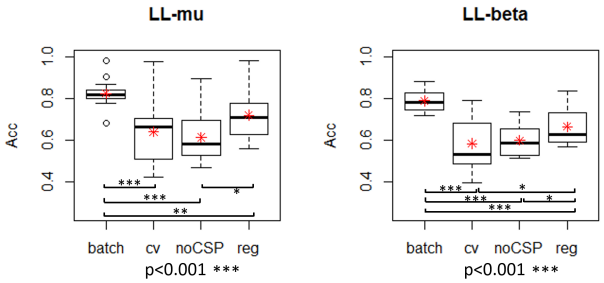
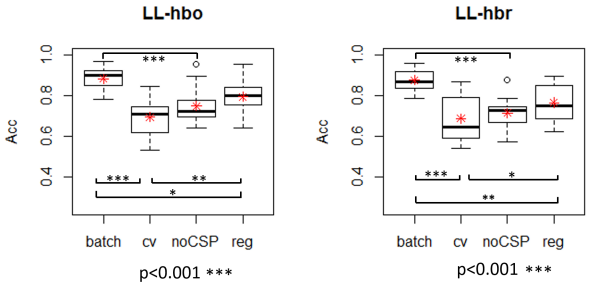


Figure B.18: Right-arm - Right-hand Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.19: Left-arm - Left-hand Motor Execution: a) EEG-based classifiers. b) NIRS-based classifiers.

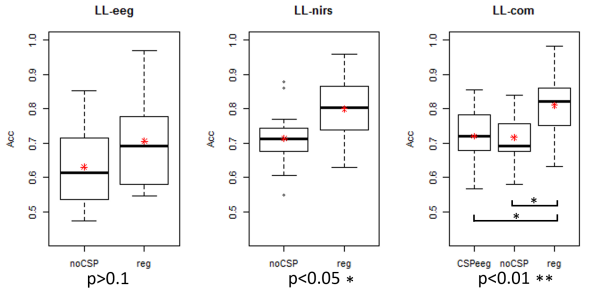
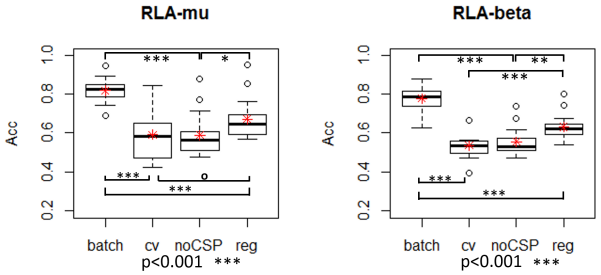
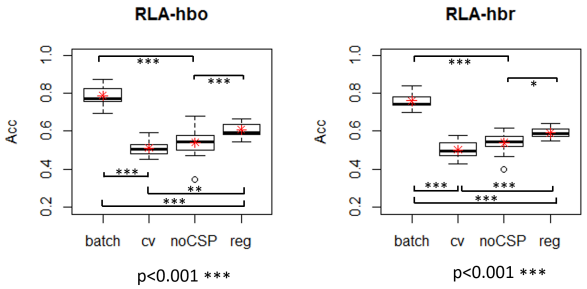


Figure B.20: Left-arm - Left-hand Motor Execution: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.21: Right-arm - Left-arm Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

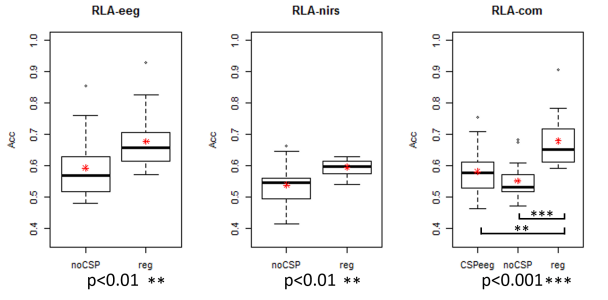
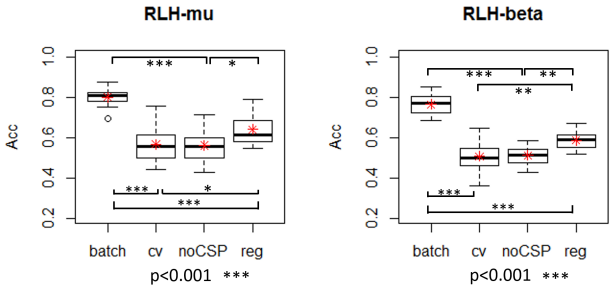
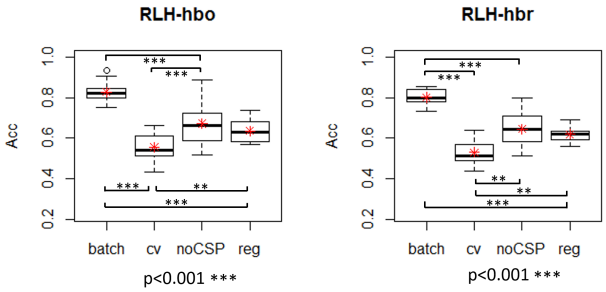


Figure B.22: Right-arm - Left-arm Motor Imagery: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.23: Right-hand - Left-hand Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

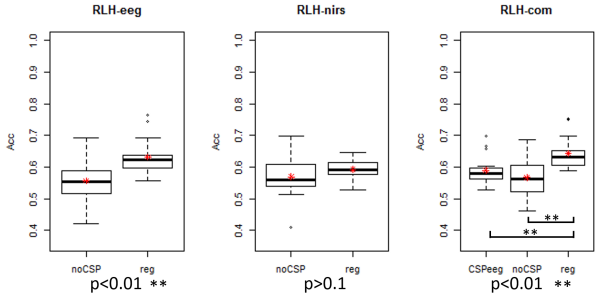
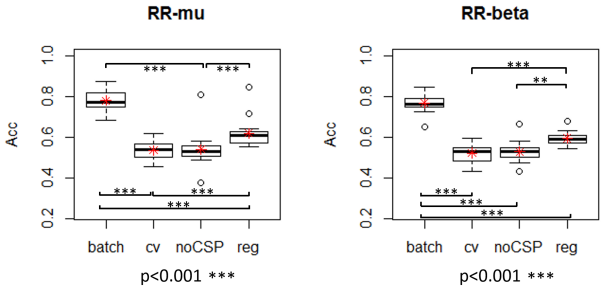
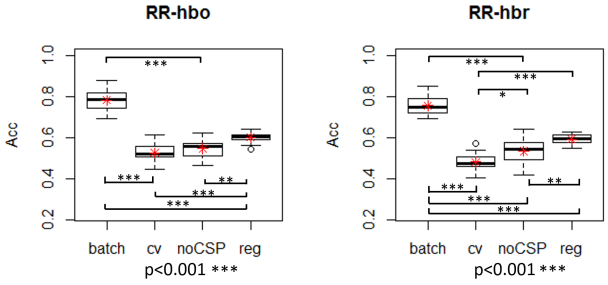


Figure B.24: Right-hand - Left-hand Motor Imagery: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.25: Right-arm - Right-hand Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

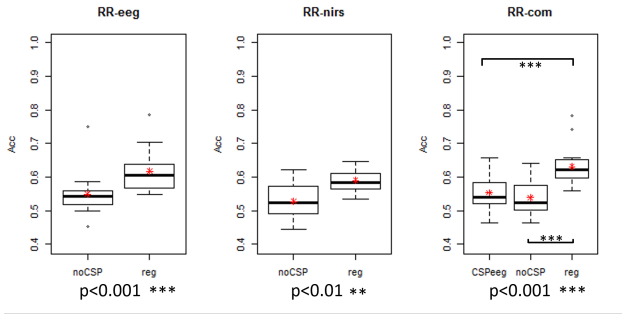
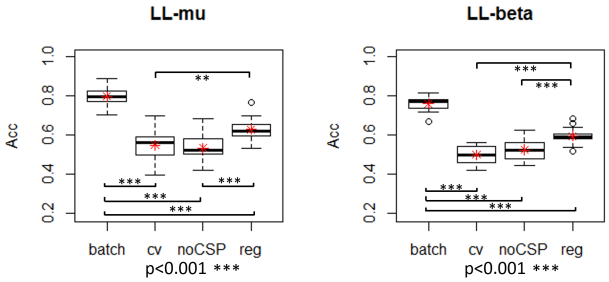
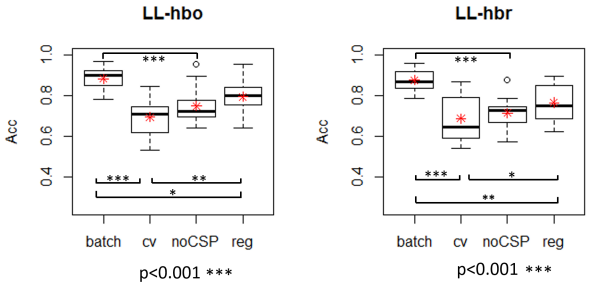


Figure B.26: Right-arm - Right-hand Motor Imagery: EEG-NIRS-based classifiers.



(a)



(b)

Figure B.27: Left-arm - Left-hand Motor Imagery: a) EEG-based classifiers. b) NIRS-based classifiers.

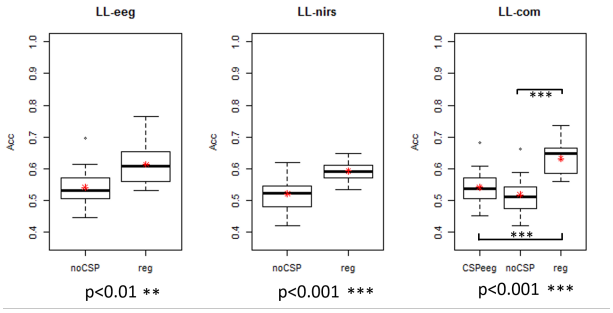


Figure B.28: Left-arm - Left-hand Motor Imagery: EEG-NIRS-based classifiers.

Bibliography

[1] www.microEEG.com.

[2] G. E. Alexander and M. D. Crutcher. Neural representations of the target (goal) of visually guided arm movements in three motor areas of the monkey. *Journal of Neurophysiology*, 64(1):164–178, July 1990.

[3] P. D. H. Berger. Uber das Elektrenkephalogramm des Menschen. *Archiv far Psychiatrie und Nervenkrankheiten*, 87(1):527–570, Dec. 1929.

[4] D. Bertsekas and D. Bertsekas. *Nonlinear Programming*. Athena Scientific, Nashua, NH, 1999.

[5] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K.-R. Muller. Optimizing Spatial filters for Robust EEG Single-Trial Analysis. *IEEE Signal Processing Magazine*, 25(1):41–56, 2008.

[6] G. E. Chatrian, M. C. Petersen, and J. A. Lazarte. The blocking of the rolandic wicket rhythm and some central changes related to movement. *Electroencephalography and Clinical Neurophysiology*, 11(3):497–510, Aug. 1959.

[7] S. Coyle, T. Ward, C. Markham, and G. McDarby. On the suitability of near-infrared (NIR) systems for next-generation brain-computer interfaces. *Physiological Measurement*, 25(4):815, Aug. 2004.

[8] S. M. Coyle, T. E. Ward, and C. M. Markham. Brain-computer interface using a simplified functional near-infrared spectroscopy system. *Journal of Neural Engineering*, 4(3):219, Sept. 2007.

[9] J. P. Donoghue. Connecting cortex to machines: recent advances in brain interfaces. *Nature Neuroscience*, 5:1085–1088, Oct. 2002.

[10] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, NY, 2012.

[11] T. Durduran, R. Choe, W. B. Baker, and A. G. Yodh. Diffuse optics for tissue monitoring and tomography. *Reports on Progress in Physics*, 73(7):076701, July 2010.

[12] E. V. Evarts. Relation of pyramidal tract activity to force exerted during voluntary movement. *Journal of Neurophysiology*, 31(1):14–27, Jan. 1968.

[13] S. Fazli, J. Mehnert, J. Steinbrink, G. Curio, A. Villringer, K.-R. Muller, and B. Blankertz. Enhanced performance by a hybrid NIRS-EEG brain computer interface. *NeuroImage*, 59(1):519–529, Jan. 2012.

[14] L. Gagnon, M. A. Yacel, M. Dehaes, R. J. Cooper, K. L. Perdue, J. Selb, T. J. Huppert, R. D. Hoge, and D. A. Boas. Quantification of the cortical contribution to the NIRS signal over the motor cortex using concurrent NIRS-fMRI measurements. *NeuroImage*, 59(4):3933–3940, Feb. 2012.

[15] G. Gargiulo, R. A. Calvo, P. Bifulco, M. Cesarelli, C. Jin, A. Mohamed, and A. van Schaik. A new EEG recording system for passive dry electrodes. *Clinical Neurophysiology*, 121(5):686–693, May 2010.

[16] A. P. Georgopoulos, J. F. Kalaska, R. Caminiti, and J. T. Massey. On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *The Journal of Neuroscience*, 2(11):1527–1537, Nov. 1982.

[17] B. Graimann, B. Allison, and G. Pfurtscheller. Brain-Computer Interfaces: A Gentle Introduction. In B. Graimann, G. Pfurtscheller, and B. Allison, editors, *Brain-Computer Interfaces*, The Frontiers Collection, pages 1–27. Springer Berlin Heidelberg, 2009.

[18] W.-Y. Hsu and Y.-N. Sun. EEG-based motor imagery analysis using weighted wavelet transform features. *Journal of Neuroscience Methods*, 176(2):310–318, Jan. 2009.

[19] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*, volume 103 of *Springer Texts in Statistics*. Springer New York, New York, NY, 2013.

[20] H. H. Jasper. The ten twenty electrode system of the international federation. *Electroencephalography and Clinical Neurophysiology*, 10:371–375, 1958.

[21] A. B. Joseph. Design considerations for the brain-machine interface. *Medical Hypotheses*, 17(3):191–195, July 1985.

[22] S. Kanoh, Y.-m. Murayama, K.-i. Miyamoto, T. Yoshinobu, and R. Kawashima. A NIRS-based brain-computer interface system during motor imagery: System development and online feedback training. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2009. EMBC 2009*, pages 594–597, Sept. 2009.

[23] M. J. Khan, M. J. Hong, and K.-S. Hong. Decoding of four movement directions using hybrid NIRS-EEG brain-computer interface. *Frontiers in Human Neuroscience*, 8:244, 2014.

[24] S. Lemm, B. Blankertz, T. Dickhaus, and K.-R. Muller. Introduction to machine learning for brain imaging. *NeuroImage*, 56(2):387–399, May 2011.

[25] H. Levene. Robust tests for equality of variances. *Contributions to Probability and Statistics. Essays in Honor of Harold Hotelling*, pages 279–292, 1961.

[26] G. A. Light, L. E. Williams, F. Minow, J. Sprock, A. Rissling, R. Sharp, N. R. Swerdlow, and D. L. Braff. Electroencephalography (EEG) and event-related potentials (ERP) with human participants. *Current protocols in neuroscience / editorial board, Jacqueline N. Crawley ... [et al.]*, CHAPTER:Unit–6.2524, July 2010.

[27] F. Lotte, M. Congedo, A. Lacuyer, and F. Lamarche. A review of classification algorithms for EEG-based brain-computer interfaces. *Journal of Neural Engineering*, 4, 2007.

[28] F. Lotte and C. Guan. Regularizing common spatial patterns to improve BCI designs: unified theory and new algorithms. *IEEE Transactions on Biomedical Engineering*, 58(2):355–362, Feb. 2011.

[29] H. Lu, K. Plataniotis, and A. Venetsanopoulos. Regularized common spatial patterns with generic learning for EEG signal classification. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2009. EMBC 2009*, pages 6599–6602, Sept. 2009.

[30] J. Muller-Gerking, G. Pfurtscheller, and H. Flyvbjerg. Designing optimal spatial filters for single-trial EEG classification in a movement task. *Clinical Neurophysiology*, 110(5):787–798, May 1999.

[31] H. Mushiake, M. Inase, and J. Tanji. Neuronal activity in the primate premotor, supplementary, and precentral motor cortex during visually guided and internally determined sequential movements. *Journal of Neurophysiology*, 66(3):705–718, Sept. 1991.

[32] N. Naseer and K.-S. Hong. Classification of functional near-infrared spectroscopy signals corresponding to the right- and left-wrist motor imagery for development of a brain-computer interface. *Neuroscience Letters*, 553:84–89, Oct. 2013.

[33] C. Neuper, R. Scherer, M. Reiner, and G. Pfurtscheller. Imagery of motor actions: Differential effects of kinesthetic and visual-motor mode of imagery in single-trial EEG. *Cognitive Brain Research*, 25(3):668–677, Dec. 2005.

[34] E. Niedermeyer and F. H. L. d. Silva. *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott Williams & Wilkins, Philadelphia, PA, 2005.

[35] NIRx Medical Technologies LLC 15 Cherry Lane, Glen Head, NY 11545, USA, and NIRx Medical Technologies LLC 15 Cherry Lane. NIRScout User Guide, Sept. 2012.

[36] P. L. Nunez and R. B. Silberstein. On the relationship of synaptic activity to macroscopic measurements: Does co-registration of EEG with fMRI make sense? *Brain Topography*, 13(2):79–96, Dec. 2000.

[37] A. Omurtag, S. G. A. Baki, G. Chari, R. Q. Cracco, S. Zehtabchi, A. A. Fenton, and A. C. Grant. Technical and clinical analysis of microEEG: a miniature wireless EEG device designed to record high-quality EEG in the emergency department. *International Journal of Emergency Medicine*, 5:35, Sept. 2012.

[38] A. V. Oppenheim. *Digital Signal Processing*. Prentice-Hall, Upper Saddle River, NJ, 1975.

[39] M. C. Park, A. Belhaj-Saaf, M. Gordon, and P. D. Cheney. Consistent features in the forelimb representation of primary motor cortex in rhesus macaques. *The Journal of Neuroscience*, 21(8):2784–2792, Apr. 2001.

[40] L. Parra and P. Sajda. Blind source separation via generalized eigenvalue decomposition. *J. Mach. Learn. Res.*, 4:1261–1269, Dec. 2003.

[41] W. Penfield and E. Boldrey. Somatic Motor and Sensory Representation in the Cerebral Cortex of Man as Studied by Electrical Stimulation. *Brain*, 60(4):389–443, Dec. 1937.

[42] W. Penny, S. Roberts, E. Curran, and M. Stokes. EEG-based communication: a pattern recognition approach. *IEEE Transactions on Rehabilitation Engineering*, 8(2):214–215, June 2000.

[43] G. Pfurtscheller, C. Brunner, A. Schlogl, and F. H. Lopes da Silva. Mu rhythm (de)synchronization and EEG single-trial classification of different motor imagery tasks. *NeuroImage*, 31(1):153–159, May 2006.

[44] G. Pfurtscheller, P. Linortner, R. Winkler, G. Korisek, and G. Muller-Putz. Discrimination of motor imagery-induced EEG patterns in patients with complete spinal cord injury. *Computational Intelligence and Neuroscience*, 2009:104–110, Apr. 2009.

[45] G. Pfurtscheller, C. Neuper, D. Flotzinger, and M. Pregenzer. EEG-based discrimination between imagination of right and left hand movement. *Electroencephalography and Clinical Neurophysiology*, 103(6):642–651, Dec. 1997.

[46] G. Pfurtscheller, C. Neuper, A. Schlogl, and K. Lugger. Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. *IEEE Transactions on Rehabilitation Engineering*, 6(3):316–325, Sept. 1998.

[47] F. Popescu, S. Fazli, Y. Badower, B. Blankertz, and K.-R. Muller. Single Trial Classification of Motor Imagination Using 6 Dry EEG Electrodes. *PLoS ONE*, 2(7):e637, July 2007.

[48] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Transactions on Rehabilitation Engineering*, 8(4):441–446, Dec. 2000.

[49] B. Reuderink and M. Poel. Robustness of the common spatial patterns algorithm in the BCI-pipeline, http://doc.utwente.nl/64884/, July 2008.

[50] F. Scholkmann, S. Kleiser, A. J. Metz, R. Zimmermann, J. Mata Pavia, U. Wolf, and M. Wolf. A review on continuous wave functional near-infrared spectroscopy and imaging instrumentation and methodology. *NeuroImage*, 85, Part 1:6–27, Jan. 2014.

[51] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, Dec. 1965.

[52] F. Sharbrough, G. Chatrian, R. Lesser, H. Lüders, M. Nuwer, and T. Picton. American electroencephalographic society guidelines for standard electrode position nomenclature. *J. clin. Neurophysiol*, 8(2):200–202, 1991.

[53] R. Sitaram, H. Zhang, C. Guan, M. Thulasidas, Y. Hoshi, A. Ishikawa, K. Shimizu, and N. Birbaumer. Temporal classification of multichannel near-infrared spectroscopy signals of motor imagery for developing a brain-computer interface. *NeuroImage*, 34(4):1416–1427, Feb. 2007.

[54] A. Stancak Jr. and G. Pfurtscheller. The effects of handedness and type of movement on the contralateral preponderance of mu-rhythm desynchronisation. *Electroencephalography and Clinical Neurophysiology*, 99(2):174–182, Aug. 1996.

[55] C. M. Stinear, W. D. Byblow, M. Steyvers, O. Levin, and S. P. Swinnen. Kinesthetic, but not visual, motor imagery modulates corticomotor excitability. *Experimental Brain Research*, 168(1-2):157–164, Aug. 2005.

[56] G. Strangman, J. P. Culver, J. H. Thompson, and D. A. Boas. A quantitative comparison of simultaneous BOLD fMRI and NIRS recordings during functional brain activation. *NeuroImage*, 17(2):719–731, Oct. 2002.

[57] B. A. Taheri, R. T. Knight, and R. L. Smith. A dry electrode for EEG recording. *Electroencephalography and Clinical Neurophysiology*, 90(5):376–383, May 1994.

[58] Y. Tomita, F.-B. Vialatte, G. Dreyfus, Y. Mitsukura, H. Bakardjian, and A. Cichocki. Bimodal BCI using simultaneously NIRS and EEG. *IEEE Transactions on Biomedical Engineering*, 61(4):1274–1284, Apr. 2014.

[59] S. P. van den Broek, F. Reinders, M. Donderwinkel, and M. J. Peters. Volume conduction effects in EEG and MEG. *Electroencephalography and Clinical Neurophysiology*, 106(6):522–534, June 1998.

[60] T. Vaughan. Guest editorial brain-computer interface technology: a review of the second international meeting. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11(2):94–109, June 2003.

[61] J. Vidal. Real-time detection of brain events in EEG. *Proceedings of the IEEE*, 65(5):633–641, May 1977.

[62] J. J. Vidal. Toward direct brain-computer communication. *Annual Review of Biophysics and Bioengineering*, 2(1):157–180, 1973.

[63] C. Vidaurre, C. Sannelli, K.-R. Muller, and B. Blankertz. Co-adaptive calibration to improve BCI efficiency. *Journal of Neural Engineering*, 8(2):025009, Apr. 2011.

[64] A. Villringer and B. Chance. Non-invasive optical spectroscopy and imaging of human brain function. *Trends in Neurosciences*, 20(10):435–442, Oct. 1997.

[65] T. Wang and B. He. An efficient rhythmic component expression and weighting synthesis strategy for classifying motor imagery EEG in a brain-computer interface. *Journal of Neural Engineering*, 1(1):1, Mar. 2004.

[66] J. Wolpaw and E. W. Wolpaw. *Brain-Computer Interfaces: Principles and Practice*. Oxford University Press, New York, NY, Dec. 2011.

[67] J. R. Wolpaw. Brain-computer interfaces as new brain output pathways. *The Journal of Physiology*, 579(3):613–619, Mar. 2007.

[68] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan. Brain-computer interface technology: a review of the first international meeting. *IEEE transactions on rehabilitation engineering*, 8(2):164–173, June 2000.

[69] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, 113(6):767–791, June 2002.

[70] N. Yamawaki, C. Wilke, Z. Liu, and B. He. An enhanced time-frequency-spatial approach for motor imagery classification. *IEEE transactions on neural systems and rehabilitation engineering*, 14(2):250–254, June 2006.