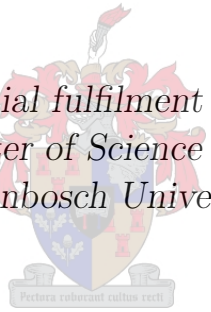
Single-Trial Classification of an EEG-Based Brain Computer Interface using the Wavelet Packet Decomposition and Cepstral Analysis

by

Shaun Lodder



Thesis presented in partial fulfilment of the requirements for the degree of Master of Science in Engineering at Stellenbosch University

Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Supervisor: Prof. J.A. du Preez

December 2009

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extend explicitly otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

December 2009

Abstract

Single-Trial Classification of an EEG-Based Brain Computer Interface using the Wavelet Packet Decomposition and Cepstral Analysis

S. Lodder

Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Thesis: MScEng (E&E)

December 2009

A Brain-Computer Interface (BCI) monitors brain activity by using signals such as EEG, EcOG, and MEG, and attempts to bridge the gap between thoughts and actions by providing control to physical devices that range from wheelchairs to computers. A crucial process for a BCI system is feature extraction, and many studies have been undertaken to find relevant information from a set of input signals.

This thesis investigated feature extraction from EEG signals using two different approaches. Wavelet packet decomposition was used to extract information from the signals in their frequency domain, and cepstral analysis was used to search for relevant information in the cepstral domain. A BCI was implemented to evaluate the two approaches, and three classification techniques contributed to finding the effectiveness of each feature type.

Data containing two-class motor imagery was used for testing, and the BCI was compared to some of the other systems currently available. Results indicate that both approaches investigated were effective in producing separable features, and, with further work, can be used for the classification of trials based on a paradigm exploiting motor imagery as a means of control.

Opsomming

Enkel-Lopie Klassifikasie van 'n EEG-Gebaseerde Brein-Rekenaar Koppelvlak met behulp van Golfies Pakkie Ontleding en Kepstrale Analise

("Single-Trial Classification of an EEG-Based Brain Computer Interface using the Wavelet Packet Decomposition and Cepstral Analysis")

S. Lodder

Departement Elektriese en Elektroniese Ingenieurswese,
Universiteit van Stellenbosch,
Privaatsak X1, Matieland 7602, Suid Afrika.

Tesis: MScIng (E&E)

December 2009

'n Brein-Rekenaar Koppelvlak (BRK) monitor brein aktiwiteit deur gebruik te maak van seine soos EEG, EcOG, en MEG. Dit poog om die gaping tussen gedagtes en fisiese aksies te oorbrug deur beheer aan toestelle soos rolstoele en rekenaars te verskaf. 'n Noodsaaklike proses vir 'n BRK is die ontginning van toepaslike inligting uit inset-seine, wat kan help om tussen verskillende gedagtes te onderskei. Vele studies is al onderneem oor hoe om sulke inligting te vind.

Hierdie tesis ondersoek die ontginning van kenmerk-vektore in EEG-seine deur twee verskillende benaderings. Die eerste hiervan is golfies pakkie ontleding, 'n metode wat gebruik word om die sein in die frekwensie gebied voor te stel. Die tweede benadering gebruik kepstrale analise en soek vir toepaslike inligting in die kepstrale domein. 'n BRK is geïmplementeer om beide metodes te evalueer.

Die toetsdata wat gebruik is, het bestaan uit twee-klas motoriese verbeelde bewegings, en drie klassifikasie-tegnieke was gebruik om die doeltreffendheid van die twee metodes te evalueer. Die BRK is vergelyk met ander stelsels wat tans beskikbaar is, en resultate dui daarop dat beide metodes doeltreffend was. Met verdere navorsing besit hulle dus die potensiaal om gebruik te word in stelsels wat gebruik maak van motoriese verbeelde bewegings om fisiese toestelle te beheer.

Acknowledgements

I would like to express my sincere gratitude to the following people and organisations:

- My parents, for their unconditional love and support throughout this study. This thesis would not have been possible without their encouragement and financial support.
- Prof. Johan du Preez, for being a pleasant study leader and giving me an invaluable amount of knowledge and wisdom that will carry me through my future endeavours.
- Ignatius Nothnagel, for proofreading this document.
- Danie Els, for his hard work in creating this thesis template used by me and so many other students.
- National Research Foundation, for their much needed financial aid over the past two years.
- The Graz BCI research group in Austria that provided the datasets used in this study.
- Everyone that contributed to the patrecII software. Their hard work saved me a lot of time.
- Fellow students Francois Olivier and John Gilmore, for lifting my spirits on those days when there was no light at the end of the tunnel.

Contents

List of Figures

List of Tables

Abbreviations

ACC	Classification Accuracy
ALS	Amyotrophic Lateral Sclerosis
AR	Autoregresive
ARBF	Adaptive Recursive Bandpass Filter
AAR	Adaptive Autoregresive
BCI	Brain Computer Interface
BP	Band Power
BRK	Brein-Rekenaar Koppelvlak
CEP	Cepstrum
$CEP_{complex}$	Complex Cepstrum
CEP_{phase}	Phase Cepstrum
CEP_{pwr}	Power Cepstrum
CEP_{real}	Real Cepstrum
CSP	Common Spatial Patterns
CSSP	Common Spectral Spatio Patterns
CSSSP	Common Sparse Spectral Spatial Patterns
CWT	Continuous Wavelet Transform
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DSLVQ	Distinction Sensitive Learning Vector Quantizer
DWT	Discrete Wavelet Transform
ECoG	Electrocorticography
EEG	Electroencephalography
EMG	Electromyography

ERD	Event-Related Desynchronization
ERR	Error Rate
ERS	Event-Related Synchronization
FE	Feature Extraction
FIR	Finite Impulse Response
FT	Fourier Transform
FFT	Fast Fourier Transform
fMRI	functional Magnetic Resonance Imaging
IIR	Infinite Impulse Response
LDA	Linear Discriminant Analysis
LMS	Least-Mean-Squares
LR	Logistic Regression
MD	Muscular Dystrophy
MEG	Magnetoencephalography
MFCC	Mel-Frequency Cepstral Coefficients
MI	Mutual Information
RBF	Radial Basis Function
RLS	Recursive Least-Squares
SBCSP	Sub-band Common Spatial Patterns
SCI	Spinal Cord Injury
SNR	Signal-to-Noise Ratio
STFT	Short Time Fourier Transform
STMI	Steepness of the Mutual Information
SVM	Support Vector Machine
TSD	Time-varying Signed Distance
UCLA	University of California, Los Angeles
WP	Wavelet Packets
WPD	Wavelet Packet Decomposition

Symbols

λ	scale factor
$\psi(\cdot)$	mother wavelet
u	translation parameter
$f(t)$	continuous time signal
$x[n]$	discrete time signal
$g[n]$	discrete low pass filter
$h[n]$	discrete high pass filter
a_{i}[n]	approximation coefficients at level i
d_{i}[n]	detail coefficients at level i
f_s	sample frequency
l_w	feature extraction window length
l_{avg}	band power average length
C3	EEG electrode reference position
C4	EEG electrode reference position
Cz	EEG electrode reference position
H_s	some hyperplane in the feature space
w_0	perpendicular distance from origin to H_s
μ	statistical mean
σ^2	statistical variance
$\boldsymbol{\alpha}$	Lagrange multipliers
$k(\mathbf{x}_j, \mathbf{x}_i)$	kernel function
$\phi(\cdot)$	kernel mapping
$\boldsymbol{\beta}$	logistic regression model weights
Err_R	error function

BCI_{cep}	cepstral based BCI
BCI_{wp}	wavelet based BCI
Err_l	likelihood error
Err_p	prior error
$\nabla\mathrm{Err}$	error gradient
p_e	chance agreement
p_0	classification accuracy
κ	kappa coefficient
I_t	MI at time t
$STMI_t$	steepness of MI at time t
γ	gamma (radial basis function)
t_{start}	trial start
t_{cue}	time of cue presentation
t_{ready}	cue notification
t_{end}	trial end

Chapter 1

Introduction

1.1 Motivation and topicality of this work

Many people suffer from neurological disorders that cause paralysis of the voluntary muscles in the body. Although conscious and able to think and reason, these individuals cannot speak or move, giving them no form of communication with the outside world. Examples of these disorders are motor neurone diseases such as amyotrophic lateral sclerosis (ALS) and muscular dystrophy (MD), and spinal cord injury (SCI).

Electroencephalography (EEG) is a non-invasive technique that measures the electrical activity on a user's scalp caused by neurons firing in the brain. It can be used to provide a non-muscular pathway for communication and control. Systems using this approach are commonly known as brain-computer interfaces (BCIs). To bridge the gap between thoughts and actions, a BCI distinguishes between different classes of brain activity, and provides direct control to physical devices.

There are many ways in which this technology can be applied to improve the quality of life for people suffering from muscular disorders. Examples are controlling a wheelchair, or providing input to a word processor on a computer.

Beyond medical applications, a practical BCI also offers an additional and independent communication channel to healthy users. This has a wide range of promising applications. Examples include computer games with intuitive control strategies and advanced virtual reality scenarios.

1.2 Background

During the execution of a unilateral movement (movement on one side of the body), an effect known as event-related synchronisation (ERS) occurs in the ipsilateral (same side) hemisphere of the brain, while another effect known as event-related desynchronisation (ERD) occurs in the contralateral (opposite side) hemisphere. ERS causes the neurons to fire in a synchronised manner,

whereas ERD causes them to fire in a de-synchronised manner [1, 2, 3, 4, 5, 6, 7]. For motor imagery, both effects take place in the mu (8-12Hz) and central beta (18-25Hz) frequency bands. This is of particular interest to this study because these effects also occur during the imagination of the movement, even if it is never physically executed.

Several types of brain function have been investigated for use in BCIs [8, 9], of which motor imagery is currently the most popular. One reason for this is that ERD and ERS occur over the motor cortex, an area of the brain which is close to the skull, where it can easily be measured by EEG [2, 3, 4, 5, 10, 11].

1.3 Advancements in BCI Research

BCI research started with a group of researchers at the University of California Los Angeles (UCLA) in the 1970s, and, from the papers published by them, the term *brain-computer interface* was first introduced into scientific literature [12, 13]. Since then, the field has grown exponentially in popularity. Although the fundamental ideas behind cognitive function originated from the early stages of BCI research, it was only later that the true potential of this technology was brought to light.

In the mid 1990s, several groups captured complex brain signals, and showed that it could be used to control physical devices [14]. The signals were mostly recorded over the motor cortex, and involved motor-related mental imagery, i.e., the imagination of physical movements.

In 2000, a research group implanted microwire arrays into the brains of two monkeys, and recorded cortical activity over a period of twelve and twenty-four months respectively [15]. The monkeys were trained to perform two tasks. For the first task, they were given a joystick and taught how to move an object on a digital screen. For the second task, food was placed on a tray and the monkeys were required to reach for it in order to pick it up. During both tasks, the motor activity that resulted in the arm movements was recorded, and a BCI was trained to distinguish between different types of movements. After this, real-time recordings were made and used as input signals to the system, which then transmitted a control output over the internet to a remote location where a robot arm was able to mimic the movements made by each monkey. The monkeys could not see the robot arm and therefore had no feedback from the BCI. This is known as an open loop system. Later, the group also managed to close the feedback loop by allowing the monkeys to see the robot arm and control its movements by adapting to the feedback they received [16].

Similar experiments were also performed by other groups, and in 2005 another group received the public's attention when they managed to successfully train a monkey to feed itself with a robotic arm controlled only by its thoughts [17].

Although these experiments show impressive results, much work is still

needed before similar trials can be conducted on humans. This is mainly due to the complex nature of the research and ethical boundaries.

The main focus in BCI research currently lies with the improvement of functional abilities in humans with severe disabilities. A condition known as *locked-in syndrome* leaves a patient without the ability to move any voluntary muscles in their body [18]. Being unable to move or speak, the patient has no means of communication with the outside world. Devices are currently being developed that will provide them with new ways of communication, and most of these are in the form of specialised digital word processors [19, 10].

To capture mental activity in human brains, some existing BCIs use invasive and partially invasive technologies such as electrode arrays and Electrocorticography (ECoG). The main focus, however, lies with non-invasive techniques such as Electroencephalography (EEG) [6, 7, 10, 20, 21, 22], because it can be used by any individual without the need for surgery. It is also more portable and affordable (and thereby more practical) than other non-invasive technologies such as functional Magnetic Resonance Imaging (fMRI) and Magnetoencephalography (MEG). The disadvantage of EEG signals and other non-invasive technologies is that the measurements only take place after the signals have passed through the skull. This results in added distortion to the signal, and the signal to noise ratio (SNR) is considerably reduced. Because of this, the robustness and accuracy of non-invasive BCIs are typically lower than their invasive and semi-invasive counterparts.

There are still many limitations to current BCI technology, and a few critical issues will have to be addressed before this field can reach its full potential. Current systems have low information transfer rates and can only transmit a few bits per minute. One of the main causes for this is the quality of the recorded signals. To improve the signal quality, more advanced electrodes and recording techniques are required. The second hurdle involves the response of the system to changes in mental activity. For a BCI to be used on devices such as wheelchairs in a real-time environment, the system will have to be agile and respond quickly to changes in mental activity.

Until these problems are addressed, the majority of BCIs will remain in the confinement of academic institutions. Few practical devices are currently available; however, with the current rate of advancement, many more should be expected in the near future.

1.4 Objectives of this study

EEG is the raw measurement of electrical activity measured from the scalp and is used as the sole input to many BCIs. The purpose of this study is to investigate techniques suitable for discriminating between different classes of motor imagery from EEG, and develop a BCI using these techniques. The primary objectives are summarised as follows:

- Find relevant information from EEG signals that can be used to separate two classes of motor imagery.
- Investigate methods that can be used to extract this information and construct features from it.
- Find classifiers that are suitable for the classification of the EEG-based features.
- Using the methods investigated, develop a BCI that is capable of discriminating between two classes of motor imagery.

A practical BCI will be used in a real-time environment, which introduces additional requirements:

- All algorithms should be causal.
- The system has to be capable of accepting a continuous (EEG) input and producing a continuous control output.
- Apart from the classification accuracy, an important factor is the rate at which information can be transferred.

The focus of this study should be on the investigation of several techniques for each part of the BCI system. It is not possible or feasible to perform an in-depth implementation of each method, but adequate references should be provided to support further investigation for each of them.

1.5 Contributions

By following the objectives set out in the preceding section, the following contributions were made by this study:

- BCI systems were implemented using C++, MATLAB® and python scripts to evaluate the various methods investigated. These systems were capable of discriminating between left and right hand motor imagery, using both wavelet and cepstral based feature extraction algorithms.
- The wavelet packet decomposition was used to obtain a time-frequency spectrum of the individual EEG trials. This spectrum efficiently captured the changes in brain activity over time, and was used to extract information from the signal (relating to ERD and ERS) that could be used to discriminate between different classes of motor imagery. Mean error rates of 19.40% were achieved from the constructed feature sets.
- As an alternative feature type, the cepstral analysis of the EEG signals was also investigated. Features were constructed from the coefficients of the power cepstrum, and mean error rates of 23.42% were achieved.

- Linear discriminant analysis (LDA), support vector machines (SVM), and logistic regression (LR) were used to evaluate the feature sets. Results indicate that the LDA and LR classifiers were more suitable for the task than a non-linear SVM. This could be due to the underlying data model not being completely known, making the linear LDA and LR classifiers less prone to overfitting than the non-linear SVM.

- In 2008, our BCI was entered into a competition where it achieved a fourth place. After further development, results showed that an even higher rank could have been achieved.

1.6 Overview of this work

The structure of this thesis can be summarised as follows:

- Chapter 2 reviews the most commonly used methods in BCI research. Common spatial patterns, autoregressive parameters, Fourier transforms, and wavelets are discussed. The chapter is concluded by comparing these methods with the help of past BCI competitions.

- Chapter 3 introduces the wavelet transform as a frequency analysis technique. It starts by discussing the advantages and properties of a wavelet, and then proceeds to introduce the continuous wavelet transform, discrete wavelet transform, and wavelet packet decomposition respectively. An example is used to illustrate the strengths and weaknesses of each method.

- Chapter 4 presents the cepstrum, and describes its uses and advantages. Homomorphic deconvolution is used to show the potential of the cepstral domain, and an example is presented at the end of the chapter where a series of echoes are removed from a signal. Four types of cepstrum are described, namely: power-, complex-, real-, and phase-cepstrum.

- Chapter 5 explains how the features are constructed from the wavelet and cepstral domains using the techniques described in Chapter 3 and Chapter 4 respectively. The chapter starts by briefly describing motor-related brain activity, and then proceeds to the feature extraction algorithms. Illustrative diagrams are presented to show how both feature types are generated.

- Chapter 6 introduces the three classifier types used in this study to evaluate the constructed feature sets. The simplest classifier, linear discriminant analysis, is presented first, followed by the logistic regression model, and lastly support vector machines. Illustrative diagrams are included for clarity.

- Chapter 7 discusses the different performance measures used to evaluate the effectiveness of the two feature types. Apart from the accuracy and error rate, Cohen's kappa coefficient, mutual information, and the steepness of the mutual information are all presented.

- Chapter 8 describes and discusses the experiments conducted to evaluate the effectiveness of the two feature extraction methods. The first experiment is aimed at finding the classification accuracy and robustness of the BCIs implemented in this study. A second experiment is then used to measure the information transfer rate of the systems. In both experiments the results are compared to results obtained by other research groups using their systems.

- Chapter 9 concludes this study by providing a brief summary of the work done, recommendations for further work, and an overview of the entire system. Many techniques can be used to extend the existing system, and a brief description of a few is provided.

Chapter 2

A Review of BCI-Based Feature Extraction

One of the main components in BCI designs is feature extraction. Several algorithms have been proposed, and a summary of the most popular ones are provided below.

2.1 Common Spatial Patterns

The Common Spatial Pattern (CSP) algorithm uses spatial filters to separate classes in an EEG dataset. Given a two-class dataset, the algorithm will attempt to find directions in the input space where the variance of one class is maximised and the variance in the other is minimised. To find these directions, CSP calculates spatial filters based on the simultaneous diagonalisation of the covariance matrices of both classes [23]. In [24] it was shown how CSP can also be extended to work with multi-class problems where more than two classes are present.

The active frequency bands during the imagination of motor-related events are subject specific [25], and, to train a new subject, CSP requires manual selection of the optimal frequency bands. This is very time consuming, and may not always produce the optimal parameter set. In [26], Common Spectral Spatio Patterns (CSSP) was proposed to find a suitable set of frequency bands automatically. This is done by optimising the frequency filters together with the spatial filters for each input channel. The proposed method extended CSP to the state space, and used time delay embedding to individually tune the frequency filters for each EEG input channel. CSSP was later extended in [27] to Common Sparse Spectral Spatial Patterns (CSSSP) by optimising an arbitrary finite impulse response (FIR) filter in the algorithm.

In addition to CSSP and CSSSP, [28] proposed Sub-band CSP (SBCSP) to find the optimal frequency bands. SBCSP decomposes each input channel into a number of sub-bands with an infinite impulse response (IIR) filter bank. A

CSP-based feature set is constructed for each sub-band, and linear discriminant analysis (LDA) is used to calculate a score for each sub-band based on its classification capability. The scores are evaluated, and a decision is made on the optimal frequency bands for the given subject.

CSP, CSSP, CSSSP, and SBCSP were compared to each other in [28] on a well known, publically available dataset. Comparisons were based on the *t-test* statistical method [29], and it was found that there was no significant difference in classification accuracy between the four methods. It should be noted however that the results obtained from CSP were found after an exhaustive search and manual tweaking, whereas the other three methods could produce a robust and consistent solution over multiple subjects, using only automatic optimisation. The comparison thereby showed that CSSP, CSSSP, and SBCSP were effective in finding the optimal frequency bands for CSP based feature extraction, but they would not obtain better results than the standard CSP algorithm.

2.2 Autoregressive Parameters

Autoregressive (AR) and Adaptive Autoregressive (AAR) parameters are commonly used in EEG analysis, and this is a popular method for feature extraction in BCIs. The AR model predicts the value at a specific time point in a signal by using a combination of its past values. The order of the model is determined by the number of values used, and the robustness of the estimation depends on the chosen model order. Higher order models are easily affected by noise, whereas lower order models neglect the high frequency components in a signal. Suitable model orders have been discussed in many texts [30, 31].

A variety of methods exist to estimate the coefficients of an AR model. They include Yule-Walker equations, Burg's method, and the Levinson-Durbin algorithm [32]. Standard AR models are limited to stationary signals. This is a problem for EEG analysis, because the corresponding signals are rarely stationary, and the features contained within them change over time. AR models have therefore been adapted to work with non-stationary signals, and a sliding window is used with the concept of local stationary.

Adaptive algorithms such as least-mean-squares (LMS), recursive-least-squares (RLS), and Kalman filters have also been proposed [33, 34]. They compare each new observation with its predicted value, and the difference is used to update the model coefficients accordingly. The algorithms for AAR models only use information from the signal prior to the new observation. They are thereby suitable for use in on-line environments [31].

EEG signals are noisy and chaotic by nature [35]. This has an adverse effect on AAR models, and often leads to a substantial decrease in classification accuracy. A recent study proposed a new method that combines adaptive filters with AAR models in order to remove unwanted noise components from

the frequency spectrum [36]. The proposed method uses an Adaptive Recursive Bandpass Filter (ARBF) before feature extraction is performed. The filter places itself over the centre frequency of the dominant component in the signal, and dynamically adjusts itself as the centre frequency changes. By doing this, the subject specific frequency bands are found automatically. The results from [36] showed that the method was effective, and improved the classification accuracy and information transfer rate of the AAR model. The evaluations were performed on a well known, publically available dataset [37].

2.3 Fourier Analysis

Fourier transforms (FT) are well known in signal processing, and are applied to a wide variety of problems. For measured signals, the discrete Fourier transform (DFT) and Fast Fourier transform (FFT) was defined, and the FFT is known to be very efficient in spectral analysis. The standard Fourier transform is only suitable for stationary and periodic signals. In [38], it was shown that they are not effective in EEG analysis, because EEG signals have non-stationary characteristics.

To work with non-stationary signals, the Short Time Fourier transform (STFT) was introduced [39]. STFT uses a sliding window to capture local changes in frequency information, and makes the assumption that signals are stationary over small segments in time. The frequency resolution is determined by the length of the sliding window; a short window will capture detailed frequency information, whereas a long window will capture slow changes in the signal.

The disadvantage of STFT is that it has a limited frequency resolution, and other techniques such as AAR models [40] and wavelet analysis [41] have been defined to work with non-stationary signals instead. In [42], STFT was compared to the wavelet transform on electromyogram (EMG) signals, which have similar properties to EEG signals. A conclusion was made that the wavelet transform was more effective at extracting information from non-periodic signals than STFT. Similar conclusions were also made in studies involving STFT and ARR models [40].

2.4 Wavelet Analysis

The wavelet transform (WT) was designed specifically to work with signals that have non-stationary properties, and, due to the characteristics of EEG, it is a very suitable tool for feature extraction in BCI designs [43]. The basis function of a WT is generic and localised in time. This enables it to capture frequency changes at specific time points, and also to use a basis function that is suitable for the characteristics of the signal. The wavelet transform

has a continuous (CWT) and discrete (DWT) form, and the DWT uses a dyadic grid to filter the signal into high and low pass bands. Because only the low passed signal is filtered at each decomposition level, the resulting spectrum has a high time resolution at high frequencies, and a high frequency resolution at low frequencies. In [44], the DWT was successfully used to extract information relating to ERS and ERD from EEG signals using symmetric electrode pairs. The weighted energy difference between the electrode pairs were used as features for a BCI.

The wavelet packet decomposition (WPD) is a generalization of the DWT, where the time-frequency resolution can be adjusted as desired [43]. Signals are recursively decomposed into high and low passed sub-bands (or bases), and the resolution of the spectrum is determined by the chosen decomposition level. In [45, 46], the sub-band energy from the last decomposition level was used to construct features from EEG signals. However, in [47, 48, 49] it was shown that the performance of a BCI depends greatly on the chosen bases.

Therefore, [47, 48, 49] proposed wavelet packet best basis decomposition (WPBBD) for EEG-based feature extraction. WPBBD automatically finds the best combination of bases to represent a signal, and it has been shown that the corresponding features achieve significantly higher classification rates than conventional WPD-based features. Because motor-related brain function is subject specific [25], the algorithm also finds the best features for each individual.

2.5 Cepstra

The cepstrum is calculated by finding the spectrum of the log spectrum of a signal. It captures information about the rate of change in the frequency spectrum and was originally used to measure seismic activity that resulted from earthquakes [50]. Since then it has become very popular in the analysis of speech and music [51]. The cepstrum is also a useful tool for homomorphic signal processing, because signals convolved in time are linearly separable in the cepstral domain.

Even though the cepstrum is widely used for many problems, literature describing its use in EEG analysis could not be found. Thus, to the best of our knowledge, cepstral based features have not been investigated for the use in Brain-Computer Interfaces.

2.6 BCI Competitions and Methods used

Many studies relating to EEG-based feature extraction use common datasets to evaluate and compare algorithms [28, 35, 36, 52, 53]. The most popular datasets are found at the BCI competition repositories [37, 54, 55], and con-

tain EEG, ECoG, and MEG data that were made publicly available. The competitions have all been used to track the progress of BCI research. Each competition consists of multiple datasets, and for each dataset a different challenge was presented. Due to the large number of research groups participating in these competitions, a comparison between their methods can provide a good indication as to the popularity and effectiveness of each technique.

To compare the techniques discussed in this chapter, a summary of the three most recent competitions is provided. Table 2.1 presents the number of participants that used CSP, AAR parameters, and wavelet analysis in each challenge. An extra column is added for the entries that used other methods or who did not provide information about their techniques. Fourier analysis was only reported once, and is therefore not showed in its own column. Take note that CSP, AAR parameters, and wavelets may have been used by some participants that did not specify their methods. The second column indicates the number of electrode channels provided by each dataset, and the last column shows the winning method for each challenge.

The challenges chosen for this comparison were all related to tasks based on motor imagery, and required the classification of multi-class problems. Three competitions were used: BCI Competition II [37], BCI Competition III [54], and BCI Competition IV [55]. A method count for each is presented, and a summary is provided at the end of the table.

From Table 2.1 we see that CSP was the most frequently reported method, and was used by 24% of the participants. AAR parameters followed in second place with 17%, and wavelet transforms in third with 9%. Although the remaining methods were grouped into one column, none of them were as popular as these three. Many of the other techniques also reported the use of signal band power to construct their features, which indicates that the most effective features are found from frequency based methods.

From the last column in Table 2.1, we see that CSP-based methods achieved first place in seven out of nine challenges, and AAR parameters and wavelet analysis each achieved first place in one. During BCI Competition II, AAR parameters and wavelet based methods were the most frequently used. However, in BCI Competition III and BCI Competition IV, CSP became more popular. This may have been as a result of the improvements made in spectral and spatial optimisation algorithms corresponding to CSP from 2005 onwards [26, 27, 28]. It should be noted that the objectives for each challenge was different, and there is not enough information provided in Table 2.1 to compare the performance of the various techniques.

Although CSP was the most popular method in the competitions, it should be noted that they were mainly used in the datasets where many electrode channels were provided. This is because CSP requires a large number of channels to calculate effective spatial filters. Even though it has been shown that the number of channels can be reduced, an average of ten to twenty channels are typically still required to produce effective features [56].

Table 2.1: A summary of the techniques used in BCI Competitions II, III, and IV. The first and second columns show the different datasets used (all relating to motor imagery tasks), and the number of electrode channels provided by each. The third, fourth and fifth columns show the number of times CSP, AAR parameters and wavelet analysis was used in each challenge, and the sixth column groups the remaining techniques (known and unknown) together. The last column presents the winning method for each challenge.

BCI Competition II (2003)						
	Channels	CSP	AAR	Wavelets	Other	Winner
Data set III	3	0	4	1	4	Wavelets
BCI Competition III (2005)						
	Channels	CSP	AAR	Wavelets	Other	Winner
Dataset I	64	3	7	2	17	CSP
Dataset IIIa	60	2	0	0	1	CSP
Dataset IIIb	2	0	2	1	4	Other
Dataset Iva	118	7	2	2	6	CSP, AAR
Dataset IVc	118	2	0	1	4	CSP
BCI Competition IV (2008)						
	Channels	CSP	AAR	Wavelets	Other	Winner
Dataset 1	64	9	1	1	14	CSP
Dataset 2a	22	5	0	0	0	CSP
Dataset 2b	3	3	0	1	2	CSP
Summary						
		CSP	AAR	Wavelets	Other	
	All datasets	48	31	17	90	

The long-term goal of BCI research is to produce user applications that can one day be used in day-to-day activities. Therefore, a system requiring fewer electrodes will be more practical than one with many. Wavelets and AAR paramaters are capable of finding band energy over selected areas of the brain with single electrodes, and this makes them more attractice for practical devices than CSP. For this reason, it was decided to investigate one of these methods instead. AAR parameters are very refined already, but many improvements can still be made on wavelet based techniques. To search for improvements in frequency based techniques, wavelets were therefore chosen as a frequency based, feature extraction method.

Cepstral analysis has not been used for BCI feature extraction before. Therefore, it was chosen as an alternate technique with the aim of searching for new characteristics in EEG data that may help with single trial classifications.

Chapter 3

The Wavelet Transform

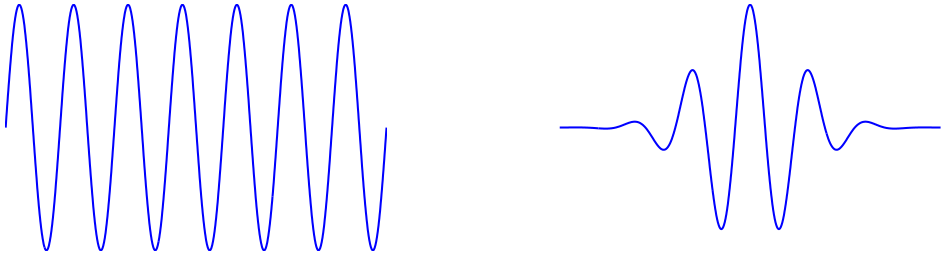
The spectrum of a signal plays an integral part in many signal processing applications and a lot of research has been done to find the best way to calculate it. The most commonly used method is the Fourier transform (FT), which uses a combination of phase-shifted, amplitude-modulated sine waves to approximate the signal. However, because a sine wave is periodic, the FT can only approximate periodic signals.

In most applications the signals will be non-stationary. In order to overcome this limitation, the FT is combined with windowing to form the short time Fourier transform (STFT) [39]. In the STFT, a window is slid over the signal and the FT is calculated over the window, thereby localizing the spectrum in time. The length of the window determines the resolution of the spectrum. A short window will capture high frequency changes, while a longer window will capture slower frequency changes. The window is, however, of a fixed length, and this limits the spectrum to a fixed resolution at all frequencies. Apart from this, the FT also causes undesired boundary effects such as leakage and the Gibbs effect. These effects are inherited by the STFT, which results in a deteriorated spectrum.

The wavelet transform (WT) was designed to overcome the limitations of the FT and STFT. It finds a multi-resolution, time-varying spectrum for any given signal, stationary or non-stationary. This is achieved by using wavelets as the basis function instead of waves [41]. Waves oscillate in time or space and are periodic. Wavelets are similar to waves, but the important difference is that they are localized in time and have finite energy. Fig. 3.1 shows the difference between a wave and a wavelet.

The WT uses a combination of scaled, time-shifted wavelets to approximate a signal. The scale factor λ, also known as the dilation coefficient, allows for multi-resolution analysis, with the time shift of each individual wavelet providing a time-varying spectrum that can capture the changes in non-stationary signals.

In the wavelet domain, frequency change is inversely related to λ. Long scales capture low frequency changes (global information) in the time series,



(a) (b)

Figure 3.1: Examples of (a) a wave and (b) a wavelet

whilst short scales capture high frequency changes (detailed information). Consequently, the WT has a high time resolution at high frequencies and a high frequency resolution at low frequencies.

As with the FT, the WT has a continuous and discrete form. These are discussed in Sections 3.2 and 3.3 respectively.

3.1 Wavelet Properties and the Mother Wavelet

The use of wavelet transforms has another advantage over Fourier transforms. As mentioned in the preceding section, the FT has a sine wave for a basis function, whilst the WT has a wavelet. Unlike a sine wave which is fixed, the wavelet defined in the transform is a generic function referred to as the mother wavelet, which can be implemented with any function relating to the characteristics of the data, as long as it satisfies three constraints [43]:

i) $\int_{-\infty}^{\infty} \psi(u)du = 0$

ii) $\int_{-\infty}^{\infty} \psi(u)^2 du = 1$

iii) $\Psi(f) = \int_{-\infty}^{\infty} \psi(u)e^{-i2\pi fu}du$, such that $\int_{-\infty}^{\infty} \frac{|\Psi(f)|^2}{|f|}df < \infty$.

Constraints (i) and (ii) require the wavelet function to be uniform, because (i) requires any oscillating movement above zero to be cancelled by the oscillating movement below zero, and (ii) requires $\psi(\cdot)$ to have some oscillating movement away from zero. Constraint (iii) is an admissibility condition that ensures that the inverse transform can be calculated. Some commonly used wavelets are presented in Fig. 3.2.

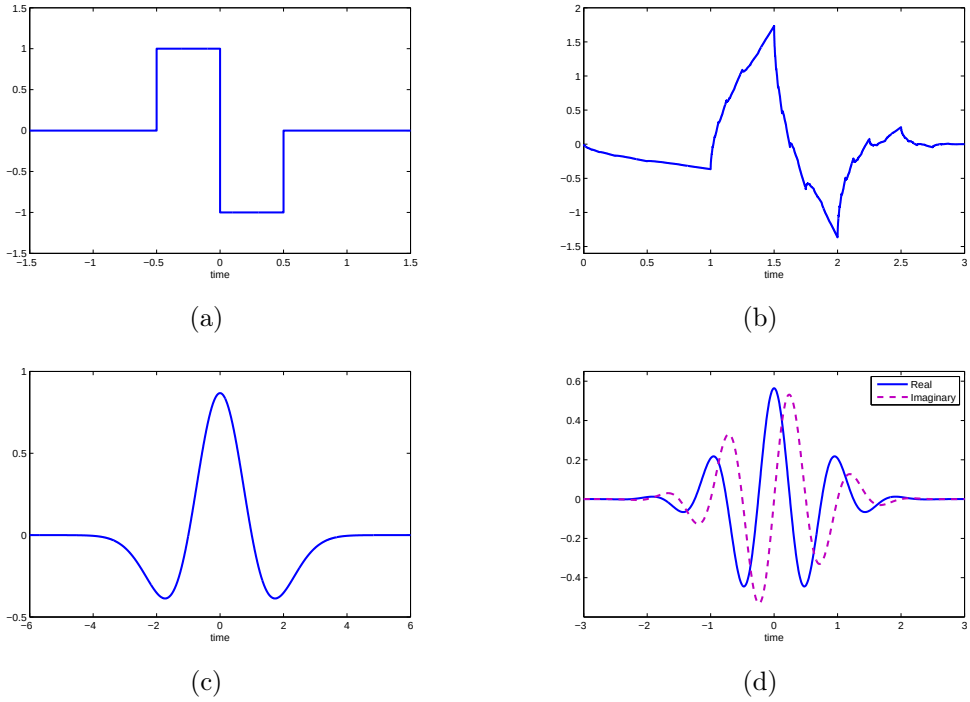


Figure 3.2: Four commonly used wavelets: (a) Haar, (b) Second Order Daubechies, (c) Mexican Hat, and (d) Complex Morlet.

3.2 Continuous Wavelet Transform

The continuous wavelet transform (CWT) is defined as [43]:

$$X_{CWT}(u,\lambda) = \frac{1}{\sqrt{|\lambda|}} \int_{-\infty}^{\infty} x(t)\psi^* \left(\frac{t-u}{\lambda}\right) dt, \tag{3.2.1}$$

where $x(t)$ is a signal continuous in time and $\psi^* \left(\frac{t-u}{\lambda}\right)$ is the mother wavelet of scale λ, centered at time u. The asterisk denotes the complex conjugate. The transformation is multiplied by the scale factor $|\lambda|^{-1/2}$ in order to normalize the energy at all scales.

The translation parameter u relates to the location of the wavelet being shifted over the time signal, and it corresponds to the time information in the wavelet domain. The scale parameter λ provides the variation of information in the time series and is inversely related to the frequency. A large λ relates to low frequency and provides global information. A small λ relates to high frequency and gives detailed information.

Fig. 3.3 shows the CWT of a linear chirp. One can clearly see how the time and frequency resolutions change over the spectrum.

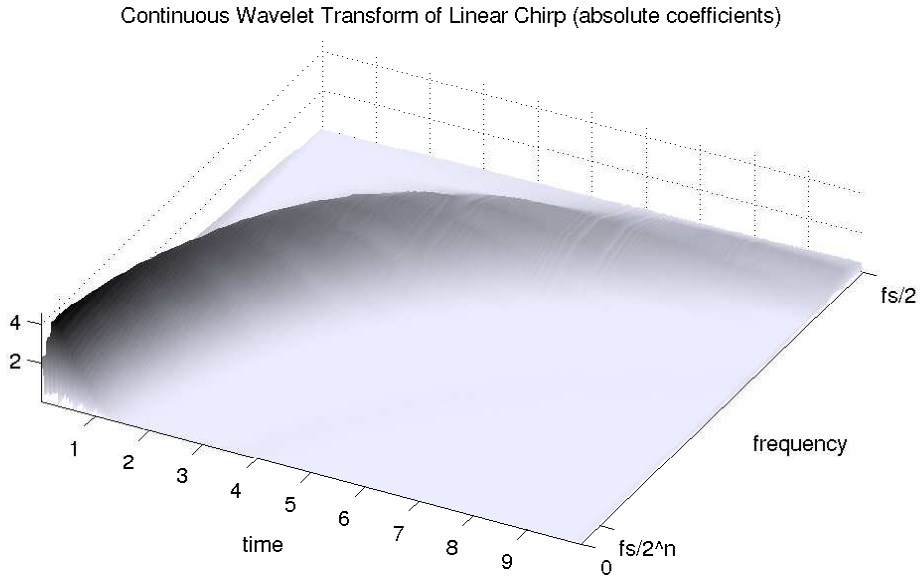


Figure 3.3: The CWT of a linear chirp.

3.3 Discrete Wavelet Transform

One drawback of the CWT is that it takes a fair amount of time and space to compute. It also contains information redundancy, i.e. a subsampled version of the CWT can still be used to perfectly reconstruct the original signal.

The discrete wavelet transform (DWT) was designed using digital filtering techniques with the aim of removing information redundancy from the CWT and increasing computational efficiency. This is achieved by using a set of filter banks to recursively subsample the signal up to a desired number of decomposition levels. At each recursion level, the scale doubles and the signal is filtered using a low- and high-pass filter. The high passed signal contains the detailed coefficients, and the low passed signal the approximation coefficients. The approximation coefficients are used as input for the next recursion level. A diagram of the procedure is shown in Fig. 3.4. Each decomposition level splits the coefficients into two orthonormal subspaces [41, 57], meaning that the DWT will not contain any information redundancy.

From the Nyquist sampling theorem, a signal can be perfectly reconstructed if the sampling rate is at least double that of the highest frequency present in a signal. This means that, at each decomposition level, the approximation coefficients can be halved in length without any loss of information. The combined length of all the DWT coefficients is therefore the same as the original time series. This saves space and has the added advantage of using fewer computations, leading to an increase in efficiency.

By inspecting the filter structure of the DWT in Fig. 3.4, one can see that it results in a fine time resolution at high frequencies and a fine frequency

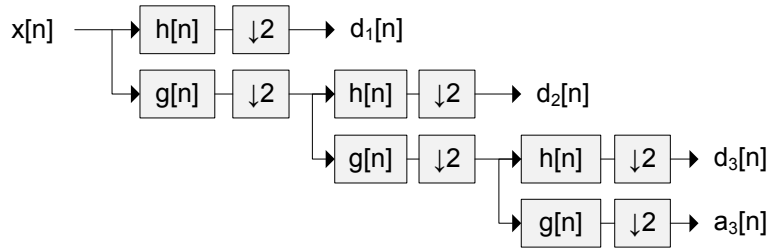


Figure 3.4: Calculating the discrete wavelet transform with a series of filters. h[n] and g[n] are high- and low-pass filters, d_1[n], d_2[n] and d_3[n] are the detailed coefficients and a_3[n] is the approximation coefficients of the signal.

resolution at low frequencies. Fig. 3.5 shows a breakdown of the spectra of the DWT. Take note how the time resolution halves and the frequency resolution doubles after each decomposition level. Fig. 3.6 shows the DWT spectra of the same linear chirp used to compute the CWT in Section 3.2. Note once again the change in time and frequency resolution at each decomposition level.

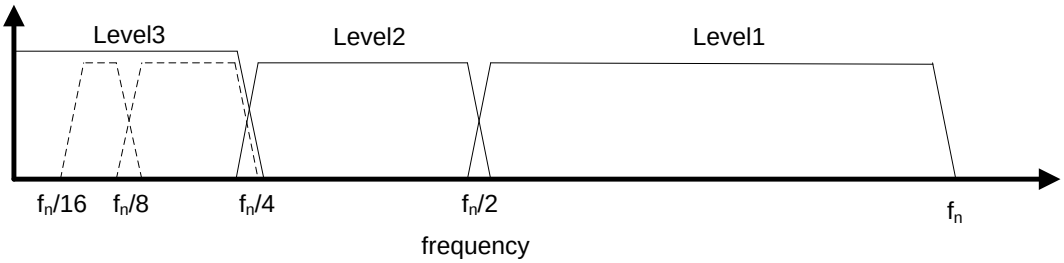


Figure 3.5: Frequency range at each level of the DWT. Example: Level 1 captures all frequencies between $f_n/2$ and f_n. $f_n = f_s/2$, where f_s is the sampling rate.

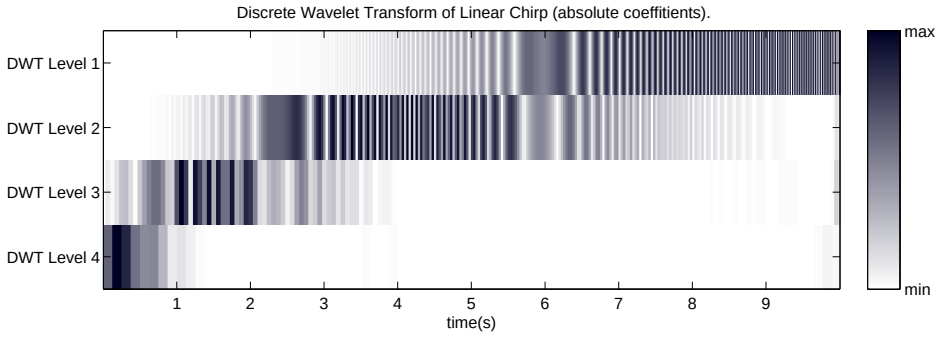


Figure 3.6: DWT of a linear chirp.

Reconstruction of the original signal from the DWT coefficients is the reverse process of the decomposition. The approximation and detailed coefficients are upsampled by two, passed through low pass and high pass synthesis

filters respectively, and then summed. The reconstruction process is shown in Fig. 3.7, however, reconstruction was not used in this study and will be omitted from the rest of this discussion.

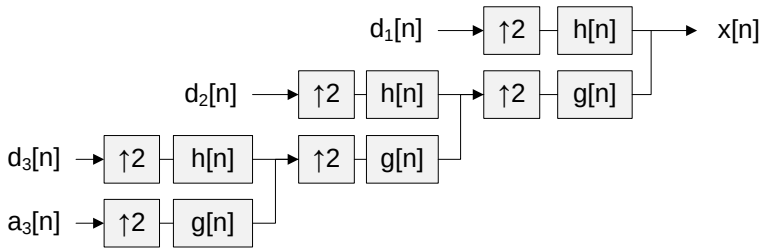


Figure 3.7: Signal reconstruction from approximation and detail coefficients.

3.4 Wavelet Packet Decomposition

Wavelet packet decomposition (WPD) uses the same principles as the DWT, but, instead of only decomposing the approximation coefficients at each level, WPD also decomposes the detail coefficients. The result is a higher frequency resolution at high frequencies, but at the cost of a decreased time resolution. WPD produces 2^n sets of coefficients (or nodes), where DWT only produces $n + 1$ sets. However, due to downsampling, the total number of coefficients remain the same. A diagram of WPD is shown in Fig. 3.8.

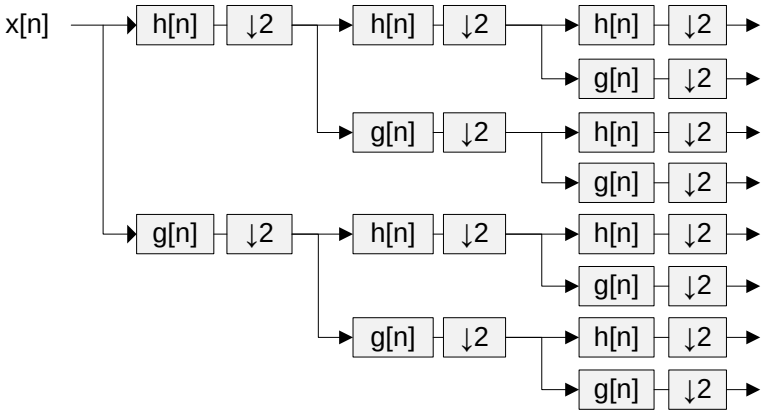


Figure 3.8: Wavelet Packet Decomposition.

The same principles described in Section 3.3 apply for decomposing the detail coefficients. This means that the WPD does not contain redundant information, whilst perfect signal reconstruction is possible from its coefficients. The WPD coefficients from the linear chirp used earlier is shown in Fig. 3.9.

Take note of the decreased time resolution and increased frequency resolution at high frequencies compared with the DWT in Fig. 3.6.

The WPD can often produce more desirable information than the DWT, and is often used in combination with the *best basis* algorithm [58, 59] to find a better set of bases to represent a signal. This helps to increase efficiency in compression techniques and improves classification rates in pattern recognition.

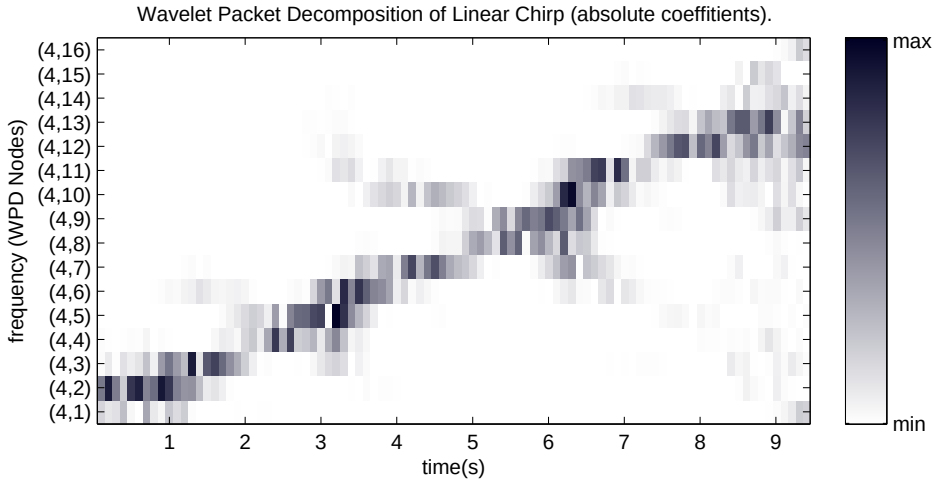


Figure 3.9: WPD of a linear chirp.

Chapter 4

Cepstrum and the Quefrency Domain

The cepstrum is used in applications ranging from speech processing to watermarking. It was first used by Bogert *et al.* [50] in 1963 as a technique for finding echoes in a composite signal. It was observed that a delayed echo will cause a ripple in the log spectrum of a signal, and by calculating a spectrum of the log spectrum, the "frequency" of that ripple is obtained. The ripple represents the harmonics of a signal, and because harmonics are periodic, they appear as peaks in the spectrum of the log spectrum. In a similar way, the cepstrum can also be used to find the fundamental frequency in a signal.

Other forms of cepstra have since been defined. To avoid confusion, the cepstrum defined originally by Bogert *et al.* is now referred to as the power cepstrum. Although a spectrum can be calculated in several ways, the Fourier transform is usually used when calculating the cepstrum. Alternative methods include the wavelet transform and discrete cosine transform [60, 61].

Using the Fourier transform and Bogert's definition, the power cepstrum is defined mathematically as [50]:

$$CEP_{pwr} = |\mathcal{F}\{log(|\mathcal{F}\{signal\}|^2)\}|^2, \tag{4.0.1}$$

and algorithmically as

$$signal \rightarrow |FT| \rightarrow square \rightarrow log \rightarrow |FT| \rightarrow square \rightarrow CEP_{pwr}. \tag{4.0.2}$$

Working with the spectrum of a log spectrum can lead to very confusing descriptions and definitions. Hence, a new set of terminology was proposed based on anagrams of existing terms used in signal analysis [50]. Although some terms are used more frequently than others, the terminology was generally accepted and has helped to avoid confusion. Some of the more frequently

used terms are:

frequency → quefrency
spectrum → cepstrum
phase → saphe
amplitude → gamnitude
filtering → liftering
harmonic → rahmonic
period → repiod

4.1 The Mel Scale and EEG Signals

When cepstral coefficients are used as features for voice or music, the power spectrum is usually transformed with the Mel scale. The Mel scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another, i.e. it provides a response that approximates the human auditory system more closely [62]. Transforming the spectrum with the Mel scale and then mapping the result to the quefrency domain yields Mel-Frequency Cepstral Coefficients, or MFCCs. Calculating MFCCs from EEG data has been investigated briefly, but EEG data is very different to audio signals and the human auditory system [63]. It is therefore unlikely that BCIs will benefit from the Mel scale itself. However, using the same concept and finding a scale relevant to motor-related EEG activity may potentially improve the features.

4.2 Processing Techniques

The power of the cepstrum lies in its ability to represent multiple signals that are convoluted in the time domain as a summation of components in the quefrency domain. For example, given a signal f consisting of a source f_{src} convoluted by a linear filter h_f:

$$f(t) = f_{src}(t) * h_f(t) \tag{4.2.1}$$

The Fourier transform of the signal is given as:

$$F(s) = F_{src}(s) \times H_f(s) \tag{4.2.2}$$

Applying the logarithm of the squared magnitude to (4.2.2) yields

$$\begin{aligned} g(t) &= log|F_{src}(s) \times H_f(s)|^2 \\ &= log|F_{src}(s)|^2 + log|H_f(s)|^2, \end{aligned} \tag{4.2.3}$$

thereby separating the source and filter into 2 separate components. Finding the squared magnitude of the Fourier transform of 4.2.3 yields the cepstrum of the recorded signal

$$g(t) = g_{src}(t) + g_f(t), \tag{4.2.4}$$

which is now represented as a summation of the source and filter in the quefrency domain. By having the two components separated in the quefrency domain, one can isolate g_f, estimate a model of the filter, and subtract it from the signal [64, 65].

The power cepstrum does not preserve its phase information, which means it is not invertible, and cannot be mapped back to the frequency domain or original time domain. The complex cepstrum introduced in Section 4.3 does however preserve its phase information and can be mapped back.

4.3 Other Forms of Cepstrum

Three other forms of cepstrum have also been defined, namely the complex, phase, and real cepstrum [50, 64]. Using the Fourier transform, the algorithmic definition of the complex cepstrum is:

$$signal \rightarrow |FT| \rightarrow log \rightarrow \text{phase unwrapping} \rightarrow FT \rightarrow CEP_{complex}. \tag{4.3.1}$$

Note that the logarithm used in calculating the complex cepstrum has to be defined for complex values. Real- and phase cepstrum are related to the power- and complex cepstrum in the following way:

$$(4 \times CEP_{real})^2 = CEP_{pwr}, \text{ and} \tag{4.3.2}$$

$$CEP_{phase} = (CEP_{complex} - \text{time reversal of } CEP_{complex})^2 \tag{4.3.3}$$

A major advantage of the complex cepstrum is that it retains its phase information. This makes it possible to map from the quefrency domain back to the frequency domain and from the frequency domain back to the time domain. From the example in Section 4.2, we can now separate the measured signal into g_{src} and g_f, remove g_f, and map g_{src} back to either the frequency or time domain for further processing. The same concept applies for removing echoes in a signal, as illustrated in the example in Section 4.5. Separating signals that are convoluted in time using cepstral analysis is known as homomorphic deconvolution, because a signal is mapped to the quefrency domain, processed, and then mapped back to the original domain.

4.4 Implementation Issues

Calculating the cepstrum has a few known issues, the first being aliasing. To avoid this, it is common practice to zero-pad the spectra before mapping to

the quefrency domain [64]. Another problem involves oversampling. Typically, signals have noise present in their spectra outside of their signal band. This is usually not a problem, because the noise components are very low in power. Calculating the cepstrum from the log spectrum can however be problematic because the noise components can potentially contribute as much to the cepstrum as the signal components themselves. When oversampling, the spectrum stretches, and the components in the frequency domain are spaced further apart, allowing more noise to slip in between the gaps. Because of this, the noise will have an increased contribution in the quefrency domain, which in turn degrades the quality of the cepstrum.

The last issue to consider is when windowing is used as a preprocessing step. This results in a convolution between the window and the measured signal in the frequency domain, which prevents the signals convoluted in time to be separated in the frequency domain (using the properties of the log function). Because of this, echo detection and other methods are greatly degraded. For applications with highly non-stationary data, however, windowing has proved useful because it allows us to work over a single pitch period [64].

4.5 Example: Echo Removal with Cepstral Analysis

To illustrate the power of quefrency analysis, we present an example where a series of echoes are removed from a signal with homomorphic deconvolution. In Fig. 4.1 a fading sine wave $x(t)$ contains echoes spaced 1.25 seconds apart. The aim is to remove these echoes without distorting the original sine wave.

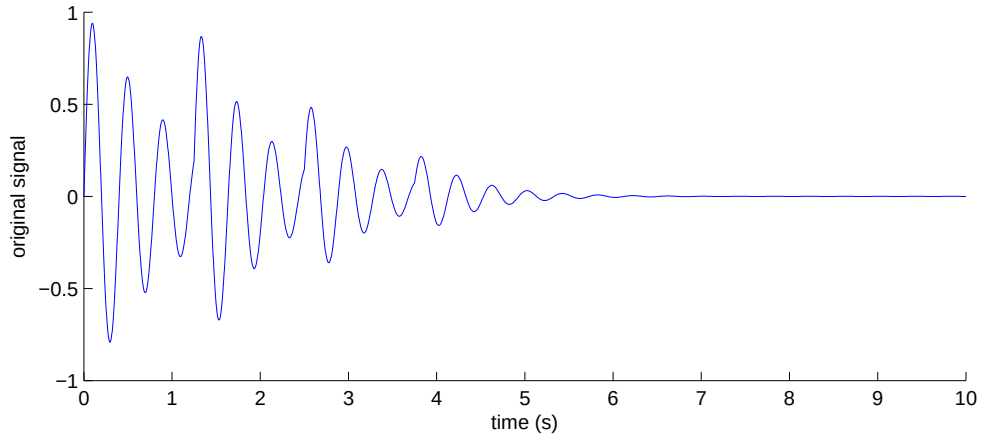


Figure 4.1: The original signal containing echoes.

Fig. 4.2 shows the spectrum of $x(t)$. We can see the ripples in the spectrum caused by the echoes, but there is no direct way of removing them in the

frequency domain without distorting the original sine wave.

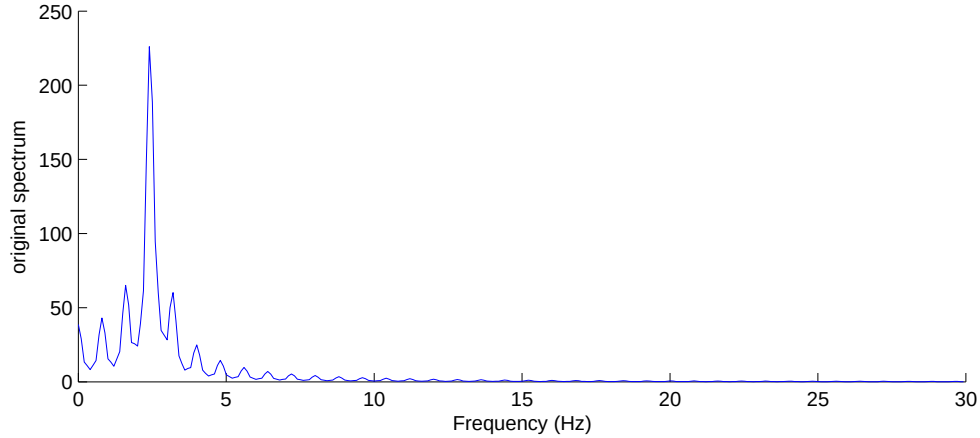


Figure 4.2: Spectrum of $x(t)$. The echoes cause ripples in the frequency domain.

The cepstrum of $x(t)$ is calculated, and is shown in Fig. 4.3. The echoes can be identified clearly as spikes in the quefrency domain.

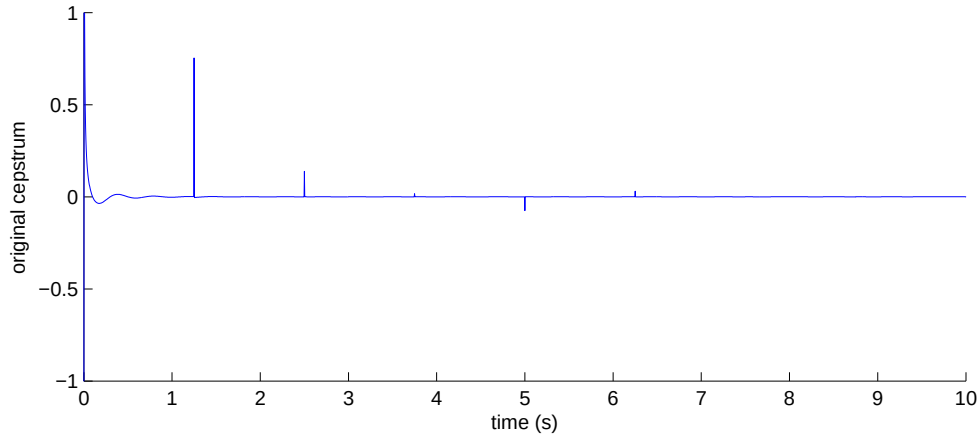


Figure 4.3: Cepstrum of $x(t)$ representing the echoes as spikes.

The spikes causing the echoes are removed from the cepstrum (shown in Fig. 4.4), and the spectrum is reconstructed and shown in Fig. 4.5. We can see that the ripples causing the echoes are not there anymore.

The signal is reconstructed from the newly acquired spectrum, and is shown in Fig. 4.6. The echoes have been removed without distorting the original sine wave.

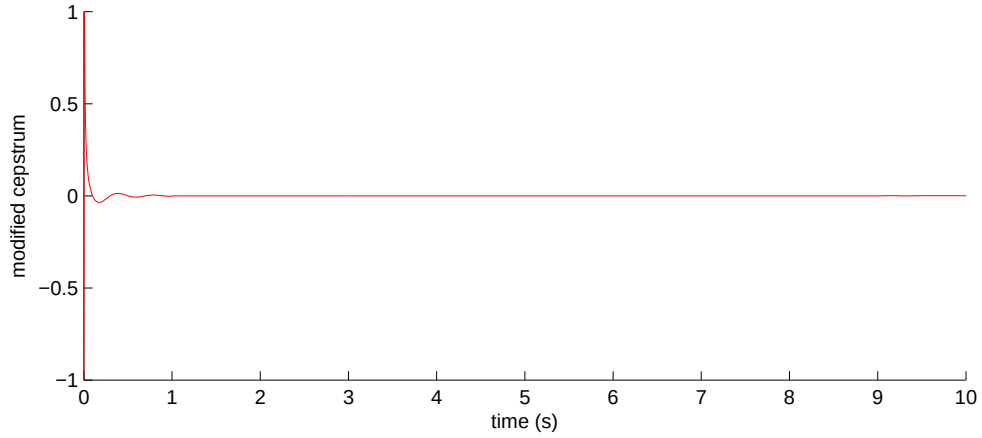


Figure 4.4: The modified cepstrum of $x(t)$ with the echoes removed.

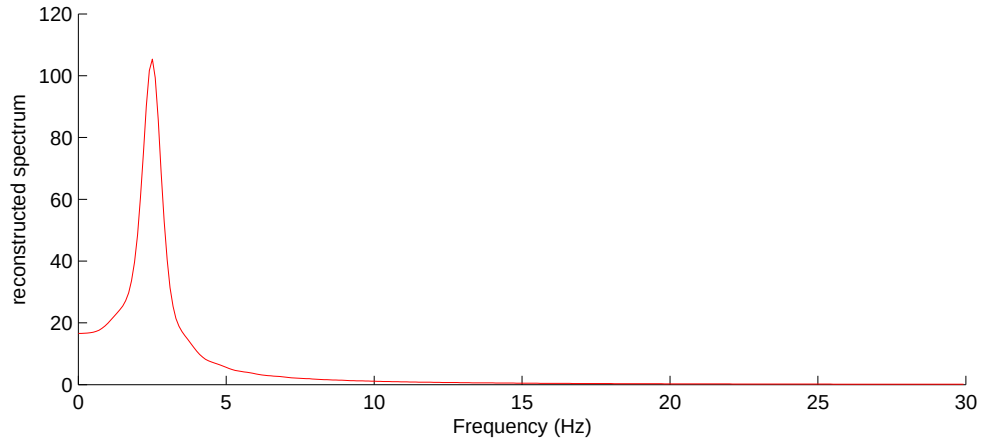


Figure 4.5: Spectrum of the modified cepstrum. The ripples seen in Fig. 4.2 are now removed.

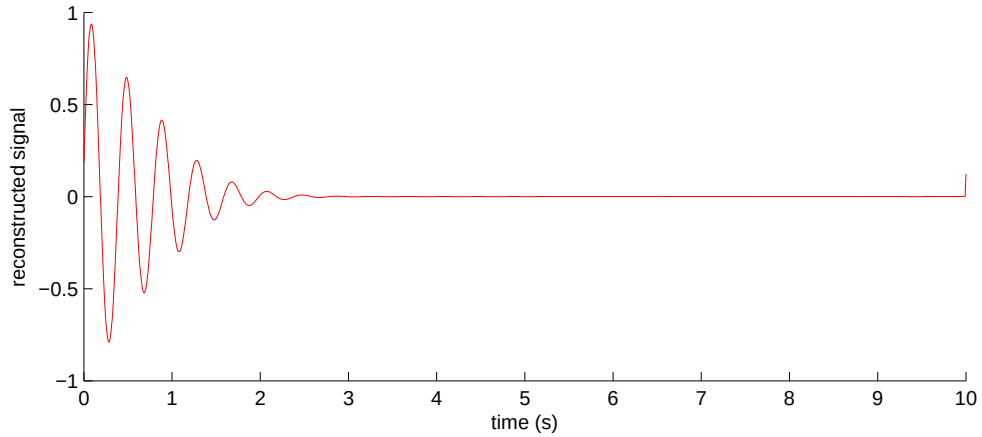


Figure 4.6: The result of homomorphic deconvolution. The echoes are removed from the signal without distorting the original fading sine wave.

Chapter 5

Feature Extraction

Electroencephalography (EEG) measures the electrical activity over different parts of the brain using special electrodes on the users scalp. For ease of use, the electrodes are mounted to a specifically designed cap whose positions are determined by a standardised referencing system. Several of these referencing systems exist, but the most popular for BCI research is the International 10-20 System [66] shown in Fig. 5.1. When used as the input to a BCI, EEG signals are processed into features, which are then classified to determine the performed mental task. From this, the classifier output can be used to produce an output for the BCI.

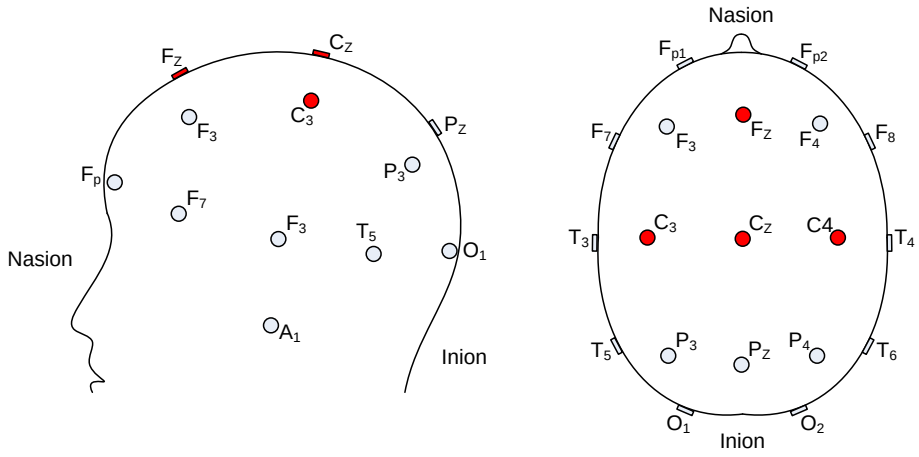


Figure 5.1: Electrode placement according to the international 10-20 system.

Although experiments on various types of brain activity have been done, those involving motor related tasks remain the most popular. Motor related tasks are divided into two main categories: motor execution and motor imagery. Motor execution involves the movement of a certain muscle group or body part, whereas motor imagery relates to the imagination of that movement, with no physical movement taking place.

One of the reasons why motor related tasks are so popular is because the associated brain activity resides in the motor cortex, an area in the brain that is easily accessible by EEG [2, 3, 4, 5, 10, 11]. Every brain is however unique, and although the activity of a certain task may reside very close to the scalp for one subject, the same task can easily execute in a deeper part of the brain for another, making it more difficult to measure by EEG. Because of this, one subject may perform exceptionally well in a given experiment while another may do considerably worse.

5.1 Simulating an Online System with Pre-Recorded Data

During this study, our BCI was evaluated with pre-recorded datasets (see Chapter 8). To simulate an online environment, a sliding window was used to represent the input buffer of an online system. A window length of l_w was used, and a step size of one sample caused the buffer to update at the same rate as the sampling frequency. Fig. 5.2 shows how the sliding window was used. Note that the datasets contained recordings from multiple EEG channels. A buffer was simulated for each of these channels.

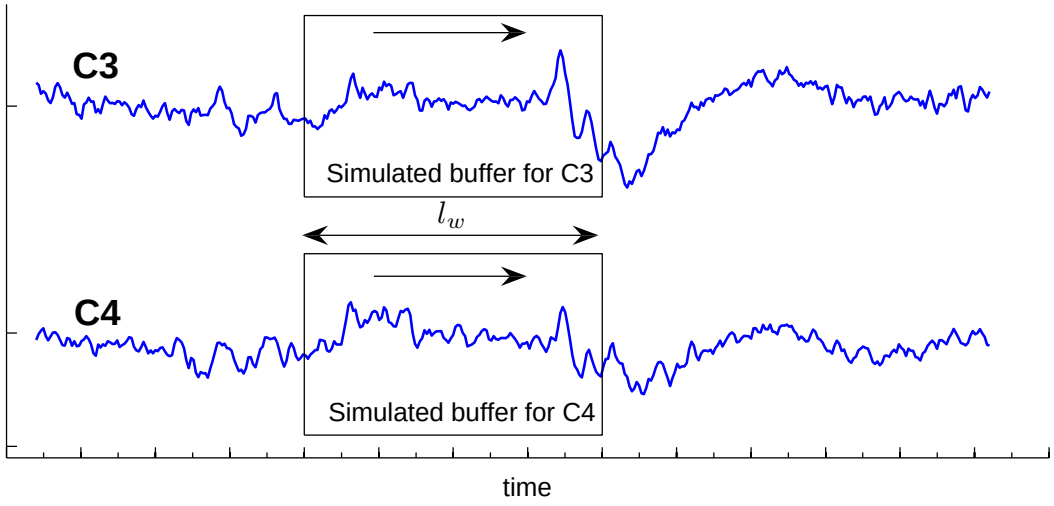


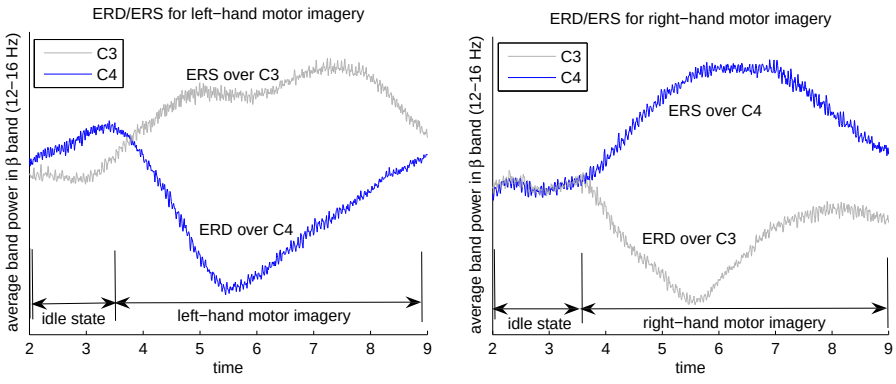
Figure 5.2: Sliding windows were used to simulate the input buffers of an online system.

5.2 Task-Related Information and ERD/ERS

So far we have explained how EEG is used to measure electrical activity in the brain, but we have not yet discussed the details of what we expect to find in these signals. Although the physiology of the brain plays an integral part in BCI research, it does not fall within the scope of this study. For the purposes of this document, brain function will only be discussed briefly at a higher level.

When the brain is awake and in an idle state, neurons fire in a rhythmic sequence. When a mental task is performed, a cluster of neurons synchronise in a specific part of the brain, and millions of tiny impulses fire in a uniform manner [1, 2, 3, 4, 5, 6, 7]. This forms a signal, which passes through the nervous system to the muscles where the desired task is then performed. The process is referred to as event-related synchronisation (ERS). For motor related tasks, ERS occurs in the ipsilateral hemisphere of the brain, e.g. a left hand movement will cause ERS (in the motor cortex) over the left hemisphere of the brain. Motor related tasks also cause an event in the contralateral hemisphere of the brain, referred to as event-related desynchronisation (ERD). This has the opposite effect to ERS: a cluster of neurons fire in a desynchronised manner, which then causes a decrease in rhythmic brain activity over a specific area of the brain.

Fig. 5.3 shows examples of ERD and ERS over electrode positions C3 and C4. In the examples, a subject performed left- and right hand motor imagery, and a time-frequency representation was obtained from the recorded signals. Due to the large amount of noise present in the recordings, it is difficult to clearly illustrate these effects using a single recorded trial. For the sake of a better illustration, 140 trials containing left hand motor imagery and 140 trials containing right hand motor imagery were averaged respectively and the averages of each class is presented in the examples.



(a) (b)

Figure 5.3: ERD and ERS over electrode positions C3 and C4 during the imagination of left and right hand movements. To better illustrate these effects, the noise was suppressed by averaging over 140 trials of each class at each time point.

An interesting and very important property of ERD and ERS is that they occur during both motor execution and motor imagery tasks. This is very beneficial for BCI technology, because it allows anyone with normal brain function to control a BCI, regardless of their ability to move their muscles.

5.3 Feature Extraction

All features were extracted on a single-trial basis, and a sliding window (as described in Section 5.1) was used to simulate the buffer of an online system. As the window moved across each trial, a single column vector was produced at each discrete time point. This provided a continuous stream of feature vectors which could then be classified to form a continuous control output.

Two feature types were used for this study. The first involved the signal band power, and was calculated with wavelet packet decomposition. The second type was constructed from the cepstral coefficients in the buffer. The methods were all causal and could be used in a real-time environment.

5.3.1 Wavelet Based Features

The band power features were constructed with the help of WPD, which was chosen because of its speed and flexibility in frequency analysis. After each update of the input buffer, a feature vector was constructed for that point in time. The vector consisted of the average band power in each frequency band for each available channel. The feature dimension was therefore determined by the number of channels and decomposition levels n_d. The frequency resolution for each band was given by

$$\left[\frac{jf_s}{2^{n_d+1}}, \frac{(j+1)f_s}{2^{n_d+1}}\right], \tag{5.3.1}$$

where $j = \{0, ..., 2^{n_d} - 1\}$, and f_s is the sample frequency. To keep the frequency bands narrow and the feature dimensions low, some datasets were downsampled before the features were extracted. Fig. 5.4 shows a summary of the feature construction process. In this example, channels C3 and C4 were available and n_d was chosen as three levels.

The band power in each dimension was determined by an average over the last l_{avg} milliseconds of each frequency band. When l_{avg} was too large, a smearing effect was caused which degraded the quality of the features, causing it not to reflect rapid changes in brain activity. When l_{avg} was however too small, the features became vulnerable to noise, and the error rate increased during the classification step.

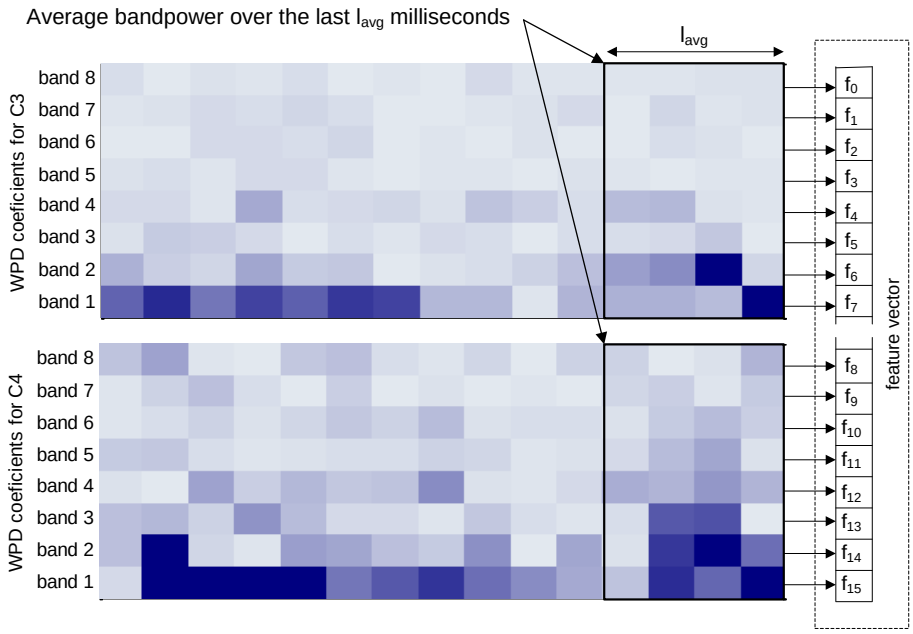


Figure 5.4: Feature construction from the band power of the WPD coefficients.

5.3.2 Cepstral Based Features

The second feature type investigated made use of the cepstral coefficients obtained from the power cepstrum. Due to a large number of coefficients, it was not feasible to use the entire cepstra as a feature vector. Instead, a segment was chosen where the largest differences between the two classes were noticed. The segment was chosen from visual inspection of the training set, however, an automated technique may have produced higher classification rates. The extracted segments from each channel were concatenated and a single feature vector was obtained for the signal at each point in time.

In this study, the power cepstrum was calculated using the Fourier transform to obtain the frequency spectrum. Other forms of cepstra (such as real- or complex cepstra) and other frequency analysis techniques (such as the wavelet transform and discrete cosine transform) could also have been used to calculate the coefficients.

Chapter 6

Classifiers

Three classifiers were investigated during this study. They were: linear discriminant analysis (LDA), support vector machines (SVMs) and logistic regression (LR). This chapter provides a detailed description for each of them.

6.1 Linear Discriminant Analysis

Linear discriminant analysis (LDA) can be used as a classifier or as a tool for dimension reduction on a dataset. When used as a classifier, LDA finds a linear hyperplane H_s to separate the vectors from different classes in their feature space. If the vectors reside in a d-dimensional feature space, a $(d-1)$-dimensional hyperplane will be used. To simplify matters, let us first consider

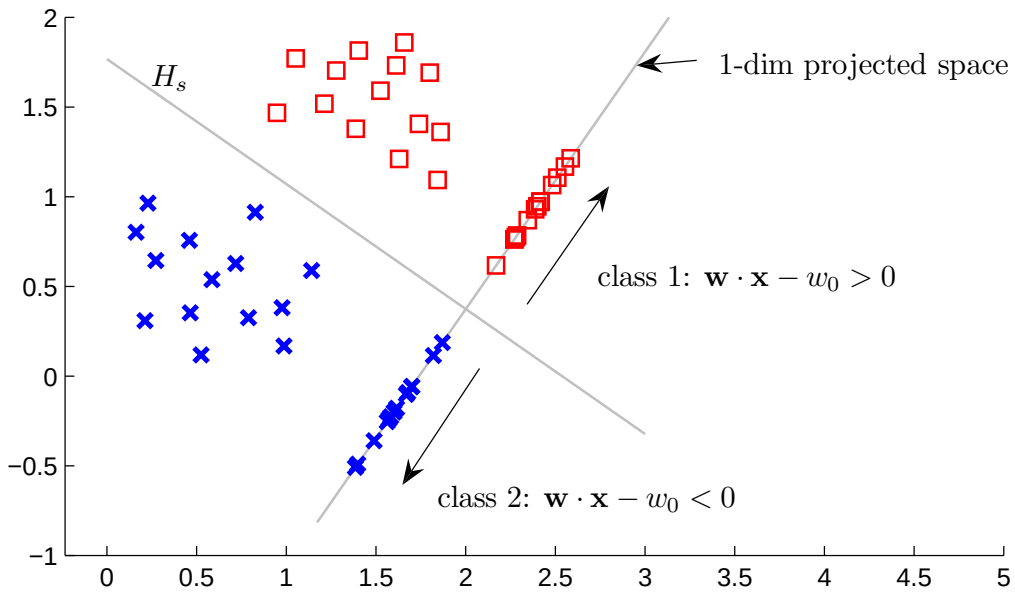


Figure 6.1: Finding a separating hyperplane in a two-class dataset and projecting the features to a one-dimensional decision space.

2-class problems. The classifier uses a transformation function $f(\mathbf{x})$ to project new input vectors from their input space to a one-dimensional decision space,

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + w_0, \tag{6.1.1}$$

where $\mathbf{x}$ is the input vector being projected by $\mathbf{w}$. The offset w_0 represents the distance of H_s to the origin, and is used as a threshold value to discriminate between the two classes. By subtracting w_0, the projection of H_s is shifted to the origin, thereby placing the threshold value at zero, and enabling us to use $f(\mathbf{x})$ as a signed distance function. The predicted class for an input vector $\mathbf{x}$ is then given by

$$c_{\mathbf{x}} = \begin{cases} class\ 1 & \text{if } f(\mathbf{x}) > 0 \\ class\ 2 & \text{if } f(\mathbf{x}) < 0 \end{cases} . \tag{6.1.2}$$

The perpendicular distance from $\mathbf{x}$ to the separating hyperplane H_s is represented by $|f(\mathbf{x})|$, which can be used as a confidence measure for the classified vector.

In LDA, the following assumptions are made: the data from each class is normally distributed, and all classes have identical covariance matrices. To find the best separation between two classes, LDA maximizes the ratio of between-class variance to within-class variance. The variances are calculated as [29, 36]:

$$\sigma^2_{between} = \sum_{i=1}^{2} (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T, \text{ and} \tag{6.1.3}$$

$$\sigma^2_{within} = \sum_{i=1}^{2} \sum_{k=1}^{n_i} (\mathbf{x}^k - \boldsymbol{\mu}_i)(\mathbf{x}^k - \boldsymbol{\mu}_i)^T, \tag{6.1.4}$$

where n_i and $\boldsymbol{\mu}_i$ are the number of features and mean values for class i, respectively. The mean across all trials is represented by $\boldsymbol{\mu}$.

Several techniques are available to extend the LDA classifier to m-class problems where m represents more than two classes [29, 67]. Some of the techniques reduce the m-class problem to a set of 2-class problems, and then finds a global prediction from the individual outputs accordingly. The 2-class classifiers are trained in one of two ways: *one-versus-rest* or *one-versus-one*. For the *one-versus-rest* approach, each classifier compares the input between one class and the rest of the training set. The input is assigned to the class producing the highest value. For the *one-versus-one* approach, a classifier is trained for every combination of classes. The input is classified with each classifier, and the class obtaining the most assignments is chosen as the overall assigned class. The disadvantage of the two-class approach is that it forms ambiguous regions in the input space.

In [29], alternate ways to classify multi-class problems are discussed. However, this study only evaluates two class problems, and a further discussion into this matter is not required.

6.2 Support Vector Machines

Support vector machines (SVMs) have been used in many fields to solve classification problems and discriminate between multi-class input vectors. If given a p-dimensional set of training vectors with each vector assigned to one of two classes, the SVM will attempt to find a hyperplane that can separate the two classes from each other in the input space. It will then classify new input vectors based on their position relative to the separating hyperplane.

We start our discussion of SVMs with the simplest case where a two-class dataset is linearly separable in the input space, and then move to a scenario where the classes overlap and the features cannot be separated. After this, non-linear SVMs are introduced where kernel-based methods are used to separate the features in a higher-dimensional feature space, and lastly we extend the SVM to a multi-class classifier that can discriminate between any number of classes. In this chapter we cover the main concepts of SVMs, however, an in-depth discussion is not covered in this document. A full discussion can be found in [29, 68, 69].

6.2.1 A SVM for Linearly Separable Datasets

Let S represent a two-class, p-dimensional training set consisting of n feature vectors. Then,

$$S = \{(\mathbf{x}_i, c_i)|\ i = \{0, ..., n-1\}\,,\ c_i \in \{-1, 1\}\,,\ \mathbf{x}_i \in \mathbb{R}^p\}\,. \tag{6.2.1}$$

Now let us suppose that the two classes in S are separable in the input space with a $(p-1)$-dimensional linear hyperplane. There may be an infinite number of hyperplanes that can separate the classes, however, we want to find the one that will provide the highest classification accuracy. This hyperplane can be found by maximizing the margin between the two classes and is known as the *maximum-margin* hyperplane H_s.

To find H_s, we construct two parallel hyperplanes H_a and H_b and place them between the two classes. We then allow them to "push" against the borders of both classes and by doing this find the maximum margin between them. H_s is placed in the middle of the margin and has an equal distance to both H_a and H_b respectively. This concept is illustrated in Fig. 6.2 and the procedure is described below.

For a vector $\mathbf{w}$ that is normal to H_s, and a point $\mathbf{x}$ on the hyperplane H_s:

$$\mathbf{w} \cdot \mathbf{x} - b = 0, \tag{6.2.2}$$

where $\frac{b}{||\mathbf{w}||}$ is the normalized perpendicular distance from H_s to the origin.

The vectors intercepted by the parallel hyperplanes H_a and H_b are called the support vectors. If H_a and H_b both have a distance of r from H_s, then

$$\mathbf{w} \cdot \mathbf{x_i} - b = +r, \tag{6.2.3}$$

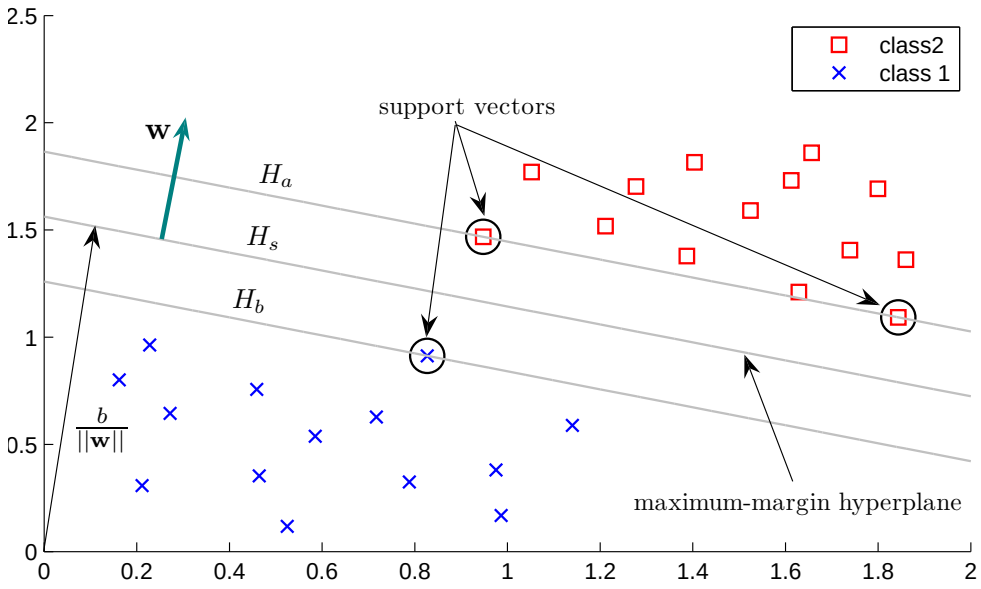


Figure 6.2: A dataset containing two-class feature vectors that are separable by a linear hyperplane. H_s provides the maximum linear separation between the two classes. It is found by constructing H_a and H_b which "push" against the borders of both classes.

for vectors intercepted by H_a and

$$\mathbf{w} \cdot \mathbf{x_i} - b = -r, \tag{6.2.4}$$

for vectors intercepted by H_b. To prevent any points from falling between H_a and H_b, the following constraints are put in place:

$$\mathbf{w} \cdot \mathbf{x_i} - b \geq +r \quad \text{for } c_i \ = +1, \text{ and} \tag{6.2.5}$$
$$\mathbf{w} \cdot \mathbf{x_i} - b \leq -r \quad \text{for } c_i \ = -1. \tag{6.2.6}$$

Equations (6.2.5) and (6.2.6) can be combined to form

$$c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq r \quad \text{for } i = \{0, ..., n-1\}\,. \tag{6.2.7}$$

The distance between H_a and H_b is $\frac{2r}{||\mathbf{w}||}$, so in order to maximize the margin, we need to minimize $||\mathbf{w}||$. This brings us to the optimization problem, described as:

$$minimize(||\mathbf{w}||), \text{ subject to (6.2.7)}. \tag{6.2.8}$$

To find a solution for this optimization problem would be very complex. For this reason it is simplified to an equivalent problem, and $\mathbf{w}$ and b are calculated accordingly. Take note that r is not needed in our final solution and can be discarded. To do this, we make use of scaling [29]: $\mathbf{w} \rightarrow r^{-1}\mathbf{w}$ and $b \rightarrow r^{-1}b$. The distance from any point $\mathbf{x}_i$ to H_s, given by $||\mathbf{w}||^{-1}(\mathbf{w} \cdot \mathbf{x}_i - b)$,

will remain unchanged. What we have gained from this however, is that the distance between H_a and H_b is now $\frac{2}{||\mathbf{w}||}$. Constraints (6.2.5) and (6.2.6) can be replaced with

$$\mathbf{w} \cdot \mathbf{x_i} - b \geq +1 \quad \text{for } c_i \ = +1, \text{ and} \tag{6.2.9}$$

$$\mathbf{w} \cdot \mathbf{x_i} - b \leq -1 \quad \text{for } c_i \ = -1, \tag{6.2.10}$$

and they can once again be combined to form

$$c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1 \ \text{ for } i = \{0, ..., n-1\}. \tag{6.2.11}$$

Minimizing $||\mathbf{w}||$ is equivalent to minimizing $\frac{1}{2}||\mathbf{w}||^2$, and by using the latter term instead, we can solve the problem with Quadratic Programming (QP) techniques.

The optimization problem is now defined as [29, 69]

$$minimize(\frac{1}{2}||\mathbf{w}||^2), \text{ subject to (6.2.11)} \tag{6.2.12}$$

In order for us to consider the constraints in this minimization, we assign them Lagrange multipliers $\boldsymbol{\alpha}$, where $\alpha_i \geq 0$ for $i = \{0, ..., n-1\}$. Hence,

$$\begin{aligned} L_P &= \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=0}^{n-1} \alpha_i \left[c_i(\mathbf{w} \cdot \mathbf{x}_i - b) - 1\right] \\ &= \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=0}^{n-1} \alpha_i c_i(\mathbf{w} \cdot \mathbf{x}_i - b) + \sum_{i=0}^{n-1} \alpha_i \end{aligned} \tag{6.2.13}$$

Equation (6.2.13) describes the problem in its primal form. For us to solve the optimization problem, we need to minimize with respect to $\mathbf{w}$ and b, and maximize with respect to $\boldsymbol{\alpha}$, all within the constraints of $\alpha_i \geq 0$ for $i = \{0, ..., n-1\}$. We do this by calculating the derivatives of L_P with respect to $\mathbf{w}$ and b, and setting them to zero [69]:

$$\begin{aligned} &\frac{dL_P}{d\mathbf{w}} = \frac{d}{d\mathbf{w}}\left[\frac{1}{2}||\mathbf{w}||^2 - \sum_{i=0}^{n-1} \alpha_i c_i(\mathbf{w} \cdot \mathbf{x}_i - b) + \sum_{i=0}^{n-1} \alpha_i\right] = 0 \\ &\Rightarrow \mathbf{w} - \sum_{i=0}^{n-1} \alpha_i c_i \mathbf{x}_i = 0 \\ &\Rightarrow \mathbf{w} = \sum_{i=0}^{n-1} \alpha_i c_i \mathbf{x}_i \end{aligned} \tag{6.2.14}$$

$$\begin{aligned} &\frac{dL_P}{db} = \frac{d}{db}\left[\frac{1}{2}||\mathbf{w}||^2 - \sum_{i=0}^{n-1} \alpha_i c_i(\mathbf{w} \cdot \mathbf{x}_i - b) + \sum_{i=0}^{n-1} \alpha_i\right] = 0 \\ &\Rightarrow \sum_{i=0}^{n-1} \alpha_i c_i = 0 \end{aligned} \tag{6.2.15}$$

By substituting (6.2.14) and (6.2.15) into (6.2.13), we obtain a formulation that is only dependent on $\boldsymbol{\alpha}$. This formulation is referred to as the Dual form and is denoted by L_D [69]:

$$\begin{aligned} L_D &= \frac{1}{2}\sum_{j=0}^{n-1}\sum_{i=0}^{n-1}\alpha_j\alpha_i c_j c_i \mathbf{x}_j \cdot \mathbf{x}_i - \sum_{j=0}^{n-1}\sum_{i=0}^{n-1}\alpha_j\alpha_i c_j c_i \mathbf{x}_j \cdot \mathbf{x}_i + \sum_{i=0}^{n-1}\alpha_i c_i b + \sum_{i=0}^{n-1}\alpha_i \\ &= -\frac{1}{2}\sum_{j=0}^{n-1}\sum_{i=0}^{n-1}\alpha_j\alpha_i c_j c_i \mathbf{x}_j \cdot \mathbf{x}_i + \sum_{i=0}^{n-1}\alpha_i, \\ &= -\frac{1}{2}\sum_{j=0}^{n-1}\sum_{i=0}^{n-1}\alpha_j\alpha_i c_j c_i k(\mathbf{x}_j, \mathbf{x}_i) + \sum_{i=0}^{n-1}\alpha_i, \quad k(\mathbf{x}_j, \mathbf{x}_i) = \mathbf{x}_j \cdot \mathbf{x}_i, \end{aligned} \tag{6.2.16}$$

subject to $\alpha_i \geq 0$ for $i = \{0, ..., n-1\}$ and $\sum_{i=0}^{n-1}\alpha_i c_i = 0$.

The structure of L_D is of particular interest to us, because it allows us to use kernel-based methods to perform non-linear classifications. Non-linear SVMs are discussed in Section 6.2.3, but for now, let us focus on the optimization problem at hand. L_D is only dependent on $\boldsymbol{\alpha}$, and because (6.2.16) is a convex quadratic optimization problem, we can use a QP-solver to maximize it and find $\boldsymbol{\alpha}$ [70]. If we have $\boldsymbol{\alpha}$, equation (6.2.14) can be used to find $\mathbf{w}$. All that remains is to calculate b.

To find b, we use the support vectors in the training set which are found at the indices where $\alpha_i > 0$:

$$S_{sv} = (\mathbf{x}_i, c_i) \tag{6.2.17}$$

where $\alpha_i > 0,\ i = \{0, ..., n-1\}$. From (6.2.9) and (6.2.10) we know that

$$c_i(\mathbf{w} \cdot \mathbf{x_i} - b) = 1,\ (\mathbf{x}_i, c_i)\ \in S_{sv}, \tag{6.2.18}$$

and substituting (6.2.14) into (6.2.18), we find that

$$c_i\left(\sum_{i=0}^{n-1}\alpha_i c_i \mathbf{x}_i - b\right) = 1,\ (\mathbf{x}_i, c_i)\ \in S_{sv}. \tag{6.2.19}$$

Multiplying (6.2.19) through by c_i and noting that $c_i^2 = 1$, we see that

$$b = c_i - \sum_{j \in S}\alpha_j c_j \mathbf{x}_j \cdot \mathbf{x}_i,\ (\mathbf{x}_i, c_i)\ \in S_{sv}. \tag{6.2.20}$$

Thus, any support vector can be used to calculate b. It is however better practice to calculate b as the mean value obtained over all the support vectors. Therefore, if S_{sv} has n_{sv} support vectors:

$$b = \frac{1}{n_{sv}}\sum_{(\mathbf{x}_i, c_i) \in S_{sv}}\left(c_i - \sum_{(\mathbf{x}_j, c_j) \in S_{sv}}\alpha_j c_j \mathbf{x}_j \cdot \mathbf{x}_i\right) \tag{6.2.21}$$

Now that we have $\mathbf{w}$ and b, we can classify a new input vector $\hat{\mathbf{x}}$ with the signed distance function

$$y = f(\hat{\mathbf{x}}) = \mathbf{w} \cdot \hat{\mathbf{x}} + b. \tag{6.2.22}$$

The predicted class for $\hat{\mathbf{x}}$ is given by

$$c_{\hat{\mathbf{x}}} = \begin{cases} class\ 1 & \text{if } y > 0 \\ class\ 2 & \text{if } y < 0 \end{cases} \tag{6.2.23}$$

and $|y|$ will give us the distance from $\hat{\mathbf{x}}$ to the separating hyperplane H_s, thereby providing a measure of confidence of the predicted class $c_{\hat{\mathbf{x}}}$.

6.2.2 Overlapping Datasets (Soft Margin SVM)

It happens often that the two classes in a dataset overlap. For this reason Cortes *et el.* suggested a way to allow mislabeled features in a training set and approached the problem by assigning a penalty to them [71]. The method, as shown in Fig. 6.3, uses positive slack variables $\xi_i \geq 0$ which measures the degree of misclassification. By adding slack variables to the constraints in (6.2.9) and (6.2.10), we now have

$$\mathbf{w} \cdot \mathbf{x_i} - b \geq +1 - \xi_i \quad \text{for } c_i \ = +1,\ \xi_i \geq 0, \text{ and} \tag{6.2.24}$$
$$\mathbf{w} \cdot \mathbf{x_i} - b \leq -1 + \xi_i \quad \text{for } c_i \ = -1\ \ \xi_i \geq 0. \tag{6.2.25}$$

Equations (6.2.24) and (6.2.25) can be combined to form

$$c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1 - \xi_i \ \text{ for } i = \{0, ..., n-1\},\ \xi_i \geq 0. \tag{6.2.26}$$

As (6.2.26) shows, the mislabeled features are penalised and our optimization problem becomes a trade off between a large margin and a small error penalty. Similar to (6.2.8), we need to solve the following optimization problem:

$$minimize(\frac{1}{2}||\mathbf{w}||^2), \text{ subject to (6.2.26).} \tag{6.2.27}$$

Again we can solve the problem by using Lagrange multipliers to minimize with respect to $\mathbf{w}$, b and $\boldsymbol{\xi}$ and to maximize with respect to $\boldsymbol{\alpha}$, where $\alpha_i \geq 0$ and $\xi_i \geq 0$ for $i = \{0, ..., n-1\}$. Describing the problem in primal form, we have [69]

$$\begin{aligned} L_P &= \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=0}^{n-1}\xi_i - \sum_{i=0}^{n-1}\alpha_i\left[c_i(\mathbf{w}\cdot\mathbf{x}_i - b) - 1 + \xi_i\right] - \sum_{i=0}^{n-1}\mu_i\xi_i \\ &= \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=0}^{n-1}\xi_i - \sum_{i=0}^{n-1}\alpha_i c_i(\mathbf{w}\cdot\mathbf{x}_i - b) + \sum_{i=0}^{n-1}\alpha_i - \sum_{i=0}^{n-1}\alpha_i\xi_i - \sum_{i=0}^{n-1}\mu_i\xi_i, \end{aligned} \tag{6.2.28}$$

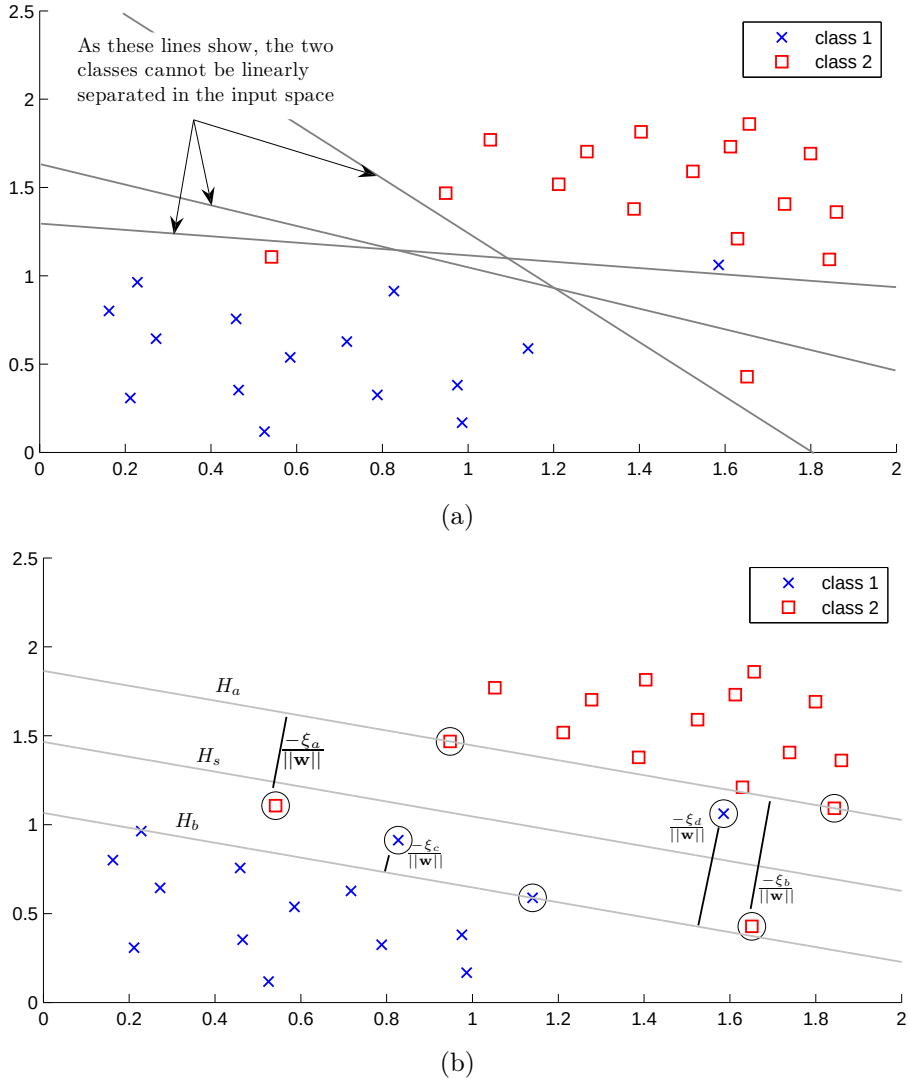


Figure 6.3: The two figures represent a two-class dataset. (a) A linear hyperplane cannot separate the two classes. (b) Slack variables are assigned to mislabelled features and the optimization problem becomes a trade-off between the separating margin and error penalty.

subject to $\alpha_i \geq 0$ and $\xi_i \geq 0$ for $i = \{0, ..., n-1\}$. The cost parameter, C, controls the trade off between allowing training errors and forcing rigid margins. It can be adjusted to either increase the margin or decrease the error

penalty. Finding the derivatives with respect to $\mathbf{w}$, b and ξ_i:

$$\frac{dL_P}{d\mathbf{w}} = \mathbf{w} - \sum_{i=0}^{n-1} \alpha_i c_i \mathbf{x}_i = 0 \Rightarrow \ \mathbf{w} = \sum_{i=0}^{n-1} \alpha_i c_i \mathbf{x}_i \tag{6.2.29}$$

$$\frac{dL_P}{db} = \sum_{i=0}^{n-1} \alpha_i c_i = 0 \tag{6.2.30}$$

$$\frac{dL_P}{d\xi_i} = C - \xi_i - \mu_i = 0 \Rightarrow \ C = \xi_i + \mu_i \tag{6.2.31}$$

If we combine (6.2.31) with the constraint $\mu_i \geq 0$, we find that $0 \leq \alpha_i \leq C$, and by substituting (6.2.29), (6.2.30) and (6.2.31) into (6.2.28):

$$\begin{aligned} L_D &= \frac{1}{2} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_j \alpha_i c_j c_i \mathbf{x}_j \cdot \mathbf{x}_i + \sum_{i=0}^{n-1} \xi_i (\alpha_i + \mu_i) - \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_j \alpha_i c_j c_i \mathbf{x}_j \cdot \mathbf{x}_i \\ &\quad + \sum_{i=0}^{n-1} \alpha_i c_i b + \sum_{i=0}^{n-1} \alpha_i - \sum_{i=0}^{n-1} \xi_i \alpha_i - \sum_{i=0}^{n-1} \xi_i \mu_i \\ &= -\frac{1}{2} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_j \alpha_i c_j c_i \mathbf{x}_j \cdot \mathbf{x}_i + \sum_{i=0}^{n-1} \alpha_i, \\ &= -\frac{1}{2} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_j \alpha_i c_j c_i k(\mathbf{x}_j, \mathbf{x}_i) + \sum_{i=0}^{n-1} \alpha_i, \quad k(\mathbf{x}_j, \mathbf{x}_i) = \mathbf{x}_j \cdot \mathbf{x}_i, \end{aligned} \tag{6.2.32}$$

subject to $0 \leq \alpha_i \leq C$ and $\sum_{i=0}^{n-1} \alpha_i c_i = 0$. Note that L_D has the same form as (6.2.16) and we can find $\boldsymbol{\alpha}$ by solving the following optimization problem:

$$maximize \left(-\frac{1}{2} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_j \alpha_i c_j c_i k(\mathbf{x}_j, \mathbf{x}_i) + \sum_{i=0}^{n-1} \alpha_i \right) \tag{6.2.33}$$

such that $0 \leq \alpha_i \leq C$, $\sum_{i=0}^{n-1} \alpha_i c_i = 0$, and $k(\mathbf{x}_j, \mathbf{x}_i) = \mathbf{x}_j \cdot \mathbf{x}_i$. This can once again be solved with a QP-solver [70]. After finding $\boldsymbol{\alpha}$, $\mathbf{w}$ is calculated with (6.2.29) and b can be calculated in the same way as in (6.2.21) using the support vectors. The support vectors are found at the indices where $0 \leq \alpha_i \leq C$. Thus, for n_{sv} support vectors in the training set:

$$b = \frac{1}{n_{sv}} \sum_{(\mathbf{x}_i, c_i) \in S_{sv}} \left(c_i - \sum_{(\mathbf{x}_j, c_j) \in S_{sv}} \alpha_j c_j \mathbf{x}_j \cdot \mathbf{x}_i \right) \tag{6.2.34}$$

After finding $\mathbf{w}$ and b, the signed distance function

$$y = f(\hat{\mathbf{x}}) = \mathbf{w} \cdot \hat{\mathbf{x}} + b \tag{6.2.35}$$

can be used the same way as in Section 6.2.2 to classify new input vectors.

6.2.3 Non-Linear SVM

In Sections 6.2.1 and 6.2.2 we have seen how a linear hyperplane is used to separate two classes in the input space. Many feature sets cannot be separated this way, but with appropriate transformations they are separable in a higher dimension.

This brings us to kernel-based methods, where input vectors are projected to a different feature space and then classified using the maximum-margin hyperplane in that space, leading to a non-linear classification in the input space. To graphically illustrate the use of kernel methods, a two-dimensional feature set is given in Fig. 6.4(a) where the classes cannot be separated linearly. Fig. 6.4(b) shows how they are mapped to a three-dimensional feature space with the Radial Basis function where they are separable.

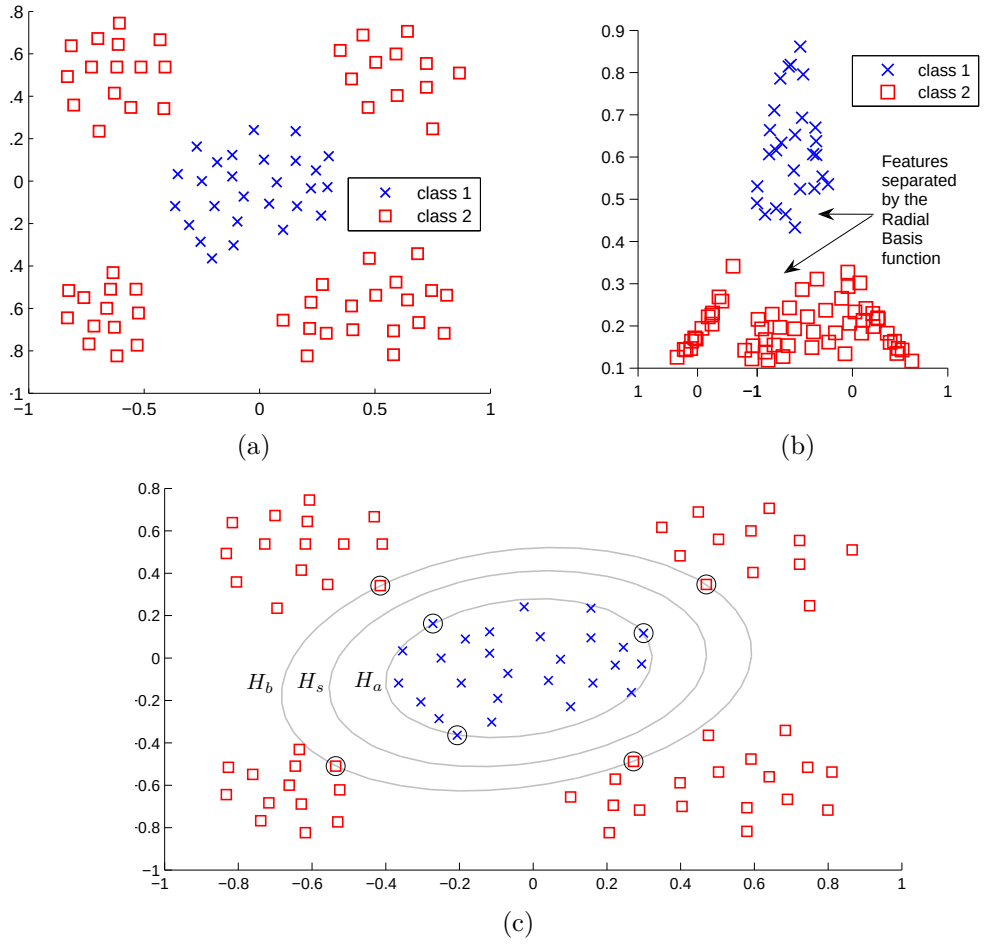


Figure 6.4: (a) The two-class, two-dimensional feature set cannot be separated linearly in the input space. (b) After mapping the features to a three-dimensional feature space with the Radial Basis function, the two classes are easily separable. (c) The separating hyperplane from the projected space mapped back to the original input space.

If we go back to equations (6.2.16) and (6.2.32) we find the kernel:

$$k(\mathbf{x}_j, \mathbf{x}_i) = \mathbf{x}_j \cdot \mathbf{x}_i. \tag{6.2.36}$$

This is referred to as the linear kernel, because it performs a dot product on the input features without a projection to a different feature space. We can replace this with a non-linear kernel and map the features to a new feature space where they are separated by a maximum-margin hyperplane. By doing this, the SVM is extended to a non-linear classifier. There are many kernel functions available and some of the popular ones are given in Table 6.1 [69]. Fig. 6.4(c) shows how the maximum-margin hyperplane in the projected space is mapped back to the original input space where it serves as a non-linear separating hyperplane.

Polynomial:	$k(\mathbf{x}_j, \mathbf{x}_i) = (\mathbf{x}_j \cdot \mathbf{x}_i)^d$
Sigmoidal:	$k(\mathbf{x}_j, \mathbf{x}_i) = tanh(a\mathbf{x}_j \cdot \mathbf{x}_i - b)$
Radial Basis:	$k(\mathbf{x}_j, \mathbf{x}_i) = e^{-\gamma\|\|\mathbf{x}_j - \mathbf{x}_i\|\|^2)}$, for $\gamma > 0$
Gaussian Radial Basis:	$k(\mathbf{x}_j, \mathbf{x}_i) = e^{-\frac{\|\|\mathbf{x}_j - \mathbf{x}_i\|\|^2}{2\sigma^2}}$

Table 6.1: Definitions for the Polynomial, Sigmoidal, Radial Basis and Gaussian Radial Basis kernel functions.

To use a non-linear SVM, we follow the same procedure as described in Sections 6.2.1 and 6.2.2, but first each feature vector $\mathbf{x}$ is projected to the new feature space with the kernel mapping $\mathbf{x} \mapsto \phi(\mathbf{x})$, where $\phi(\cdot)$ represents the chosen kernel function. Using the Soft Margin SVM, the optimization problem becomes

$$maximize \left(-\frac{1}{2} \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_j \alpha_i c_j c_i k(\mathbf{x}_j, \mathbf{x}_i) + \sum_{i=0}^{n-1} \alpha_i \right) \tag{6.2.37}$$

such that $0 \leq \alpha_i \leq C$, $\sum_{i=0}^{n-1} \alpha_i c_i = 0$, and $k(\mathbf{x}_j, \mathbf{x}_i) = \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_i)$. This can again be solved using a QP-solver. After finding $\boldsymbol{\alpha}$, $\mathbf{w}$ can be calculated as

$$\mathbf{w} = \sum_{i=0}^{n-1} \alpha_i c_i \phi(\mathbf{x}_i), \tag{6.2.38}$$

and b is found by

$$b = \frac{1}{n_{sv}} \sum_{(\mathbf{x}_i, c_i) \in S_{sv}} \left(c_i - \sum_{(\mathbf{x}_j, c_j) \in S_{sv}} \alpha_j c_j \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_i) \right). \tag{6.2.39}$$

A new input vector $\hat{\mathbf{x}}$ is then classified with the signed distance function

$$y = f(\hat{\mathbf{x}}) = \mathbf{w} \cdot \phi(\hat{\mathbf{x}}) + b \tag{6.2.40}$$

and the classlabel and classifier confidence can be extracted as described in Section 6.2.2.

6.2.4 Multi-Class SVM

So far SVMs have only been described for two-class problems, however m-class problems where $m > 2$ can also be solved. The same technique described for multi-class LDA classifiers can be used for multi-class SVMs, and a detailed discussion about this is provided in Section 6.1 and in [29]. Take note that only two-class data was used in this study, and there was no need for any of these techniques.

6.3 Logistic Regression

Logistic regression (LR) belongs to the group of classifiers known as generalised linear classifiers. It takes a single input z, fits it to a logistic curve, and produces a dichotomous result [29]:

$$f(z) = \frac{1}{1 + e^{-z}}. \tag{6.3.1}$$

The parameter z, also known as the logit, is formed by combining a set of weights $\boldsymbol{\beta} \in \mathbb{R}^{d+1}$ with an input vector $\mathbf{x} \in \mathbb{R}^d$:

$$z = \beta_0 + \beta_1 x_0 + ... + \beta_d x_{d-1}. \tag{6.3.2}$$

$\beta_0, .., \beta_d$ are called the regression coefficients, and $x_0, .., x_{d-1}$ the explanatory variables. The explanatory variables can be numerical or categorical, i.e. provide a quantity or indicate if the input data has a specific quality.

As we see in (6.3.2), LR uses the regression coefficients to model the relationship between the explanatory variables and the importance of their contribution to the model. A large regression coefficient indicates a strong influence by the explanatory variable on the outcome of the classification, and a near-zero coefficient indicates little or no influence. Similarly, positive regression coefficients will increase the probability of an outcome, whereas negative coefficients will decrease it.

The logistic curve is sigmoidal, meaning the classifier output will always lie between zero and one. This implies that $f(z)$ will tend to zero for small z, and to one for large z. Two classes can therefore be separated by estimating a meaningful $\boldsymbol{\beta}$ that will produce an output close to zero for one class, and close to one for the other. For the purposes of a BCI output, we can turn $f(z)$ into

a signed distance function by using the transformation $\hat{f}(z) = 2(f(z) - 0.5)$. Now, $\hat{f}(z)$ will produce a negative output for all features from one class, and a positive output for the other. The magnitude of $\hat{f}(z)$ can also be used as a confidence measure for the predicted class.

As mentioned above, LR only discriminates between two classes. This restriction can be lifted by training m parameter vectors $\{\boldsymbol{\beta}_{(0,:)}, ..., \boldsymbol{\beta}_{(m-1,:)}\} \in \mathbb{R}^{d+1}$ for data containing m classes where $m > 2$. Each vector $\boldsymbol{\beta}_{(c,:)}, c \in \{0, ..., m-1\}$ is trained individually by finding the best separation between class c and the rest of the training set. This is called the *one-versus-rest* approach. Other techniques are also available (see Chapter 6.1 and [29]). During classification, a new input vector is classified with the parameter vector of each class, and is then assigned to the class providing the highest probability.

The multinomial logistic model for a d-dimensional dataset containing m classes is given by [29]:

$$p(c|\mathbf{x}, \boldsymbol{\beta}) = \frac{\exp(\boldsymbol{\beta}_{(c,:)} \cdot \mathbf{x})}{Z_x}, c \in \{0, ..., m-1\}, \tag{6.3.3}$$

where Z_x is the normalising factor given by

$$Z_x = \sum_{c'=0}^{m-1} e^{\boldsymbol{\beta}_{(c',:)} \cdot \mathbf{x}}. \tag{6.3.4}$$

To find the optimal $\boldsymbol{\beta}$, we have to minimise the classification error for each class in the training set. This can be done by descending the error function Err_R of the logistic model until a global minimum is found.

Sections 6.3.1 and 6.3.2 define the error function and formulate the gradient of Err_R for a given point on the function. Section 6.3.3 will then describe how the error is minimised with gradient descent methods, and Section 6.3.4 will briefly discuss the implementation of gradient descent algorithms.

6.3.1 The Error Function

Let S represent an m-class, d-dimensional training set consisting of n feature vectors, and let $\boldsymbol{\beta}$ represent the parameter set containing the regression coefficients for each class in S:

$$S = \left\{(\mathbf{x}_j, c_j) |\ j = \{0, ..., n-1\},\ c_j \in \{0, ..., m-1\},\ \mathbf{x}_j \in \mathbb{R}^d\right\}, \tag{6.3.5}$$

$$\boldsymbol{\beta} = \left\{\boldsymbol{\beta}_{(0,:)}, ..., \boldsymbol{\beta}_{(m-1,:)}\right\} \in \mathbb{R}^{d+1}. \tag{6.3.6}$$

The error function is defined as [72]

$$\mathrm{Err}_R(\boldsymbol{\beta}, S, \sigma^2) = \mathrm{Err}_l(S, \boldsymbol{\beta}) + \mathrm{Err}_p(\boldsymbol{\beta}, \sigma^2), \tag{6.3.7}$$

where $\mathrm{Err}_l(\cdot)$ is the likelihood error and $\mathrm{Err}_p(\cdot)$ the prior error for a prior of type R. The likelihood error calculates the log likelihood of the prediction of

each input $\mathbf{x}_j$:

$$\begin{aligned}\text{Err}_l(S, \boldsymbol{\beta}) &= -\log p(S|\boldsymbol{\beta}) \\ &= -\log \prod_{i=0}^{n-1} p(c_j|\mathbf{x}_j, \boldsymbol{\beta}) \\ &= -\sum_{j=0}^{n-1} \log p(c_j|\mathbf{x}_j, \boldsymbol{\beta}),\end{aligned} \tag{6.3.8}$$

and the prior error calculates the log-likelihood for $\boldsymbol{\beta}$ at each dimension and class of S:

$$\begin{aligned}\text{Err}_p(\boldsymbol{\beta}, \sigma^2) &= -\log p_R(\boldsymbol{\beta}|\sigma^2) \\ &= -\log \left(\prod_{c=0}^{m-1} \prod_{i=0}^{d-1} f_R(\beta_{c,i}|\sigma_i^2) \right) \\ &= -\sum_{c=0}^{m-1} \sum_{i=0}^{d-1} \log f_R(\beta_{c,i}|\sigma_i^2),\end{aligned} \tag{6.3.9}$$

where $f_R(\cdot)$ is the density function for a prior of type R.

For Gaussian, Laplace, and Cauchy priors, the zero-centered density functions are [72]:

$$f_{\text{gaussian}}(\beta_{c,i}|\sigma_i^2) = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{\beta_{c,i}^2}{2\sigma_i^2}} \tag{6.3.10}$$

$$f_{\text{laplace}}(\beta_{c,i}|\sigma_i^2) = \frac{\sqrt{2}}{2\sigma_i} e^{-\frac{\sqrt{2}|\beta_{c,i}|}{\sigma_i}} \tag{6.3.11}$$

$$f_{\text{cauchy}}(\beta_{c,i}|\sigma_i^2) = \frac{\lambda}{\pi(\beta_{c,i}^2 + \lambda^2)} \tag{6.3.12}$$

In (6.3.12), $\lambda > 0$ is the scale parameter that determines the spread of the probability distribution. If we substitute (6.3.8) and (6.3.9) into (6.3.7), the error function can be written as

$$\begin{aligned}\text{Err}_R(\boldsymbol{\beta}, S, \sigma^2) &= \text{Err}_l(S, \boldsymbol{\beta}) + \text{Err}_p(\boldsymbol{\beta}, \sigma^2) \\ &= -\sum_{j=0}^{n-1} \log p(c_j|\mathbf{x}_j, \boldsymbol{\beta}) - \sum_{c=0}^{m-1} \sum_{i=0}^{d-1} \log f_R(\beta_{c,i}|\sigma_i^2).\end{aligned} \tag{6.3.13}$$

6.3.2 Calculating the Gradient of the Error Function

To find the optimal values for $\boldsymbol{\beta}$, we have to find the point in the error function where the gradient is zero. To do this we need a formulation for the gradient

at any given point (c, i), where $c \in \{0, ..., m-1\}$ and $i \in \{0, ..., n-1\}$. The error gradient can be obtained by deriving Err_R with respect to $\beta_{(c,i)}$ [72]:

$$\begin{aligned}\nabla_{c,i}\text{Err}_R(\boldsymbol{\beta}, S, \sigma^2) &= \frac{d}{d\beta_{c,i}}(\text{Err}_l(S, \boldsymbol{\beta}) + \text{Err}_p(\boldsymbol{\beta}, \sigma^2)) \\ &= \nabla_{c,i}\text{Err}_l(S, \boldsymbol{\beta}) + \nabla_{c,i}\text{Err}_p(\boldsymbol{\beta}, \sigma^2) \end{aligned} \tag{6.3.14}$$

The derivative of the likelihood error is:

$$\begin{aligned}\nabla_{c,i}\text{Err}_l(S, \boldsymbol{\beta}) &= \frac{d}{d\beta_{c,i}}\text{Err}_l(S, \boldsymbol{\beta}) \\ &= \frac{d}{d\beta_{c,i}}\left(-\sum_{j=0}^{n-1}\log p(c_j|\mathbf{x}_j, \boldsymbol{\beta})\right) \\ &= -\sum_{j=0}^{n-1}\frac{d}{d\beta_{c,i}}\log p(c_j|\mathbf{x}_j, \boldsymbol{\beta}) \end{aligned} \tag{6.3.15}$$

For a training example belonging to class c, the derivative at dimension i is:

$$\begin{aligned}\frac{d}{d\beta_{c,i}}\log p(c_j|\mathbf{x}_j, \boldsymbol{\beta}) &= \frac{d}{d\beta_{c,i}}\log\frac{e^{\boldsymbol{\beta}_{(c_j,:)}\cdot\mathbf{x}_j}}{\sum_{c'=0}^{m-1}e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j}} \\ &= \frac{d}{d\beta_{c,i}}\log e^{\boldsymbol{\beta}_{(c_j,:)}\cdot\mathbf{x}_j} - \frac{d}{d\beta_{c,i}}\log\sum_{c'=0}^{m-1}e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j} \\ &= \frac{d}{d\beta_{c,i}}(\boldsymbol{\beta}_{(c_j,:)}\cdot\mathbf{x}_j) - \frac{1}{\sum_{c'=0}^{m-1}e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j}}\sum_{c'=0}^{m-1}\frac{d}{d\beta_{c,i}}e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j} \\ &= x_{(i,j)}I_c(c = c_j) - \frac{1}{\sum_{c'=0}^{m-1}e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j}}\sum_{c'=0}^{m-1}\left(e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j}\frac{d}{d\beta_{c,i}}\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j\right) \\ &= x_{(i,j)}I_c(c = c_j) - \left(\frac{1}{\sum_{c'=0}^{m-1}e^{\boldsymbol{\beta}_{(c',:)}\cdot\mathbf{x}_j}}\right)(e^{\boldsymbol{\beta}_{(c,:)}\cdot\mathbf{x}_j})(x_{(i,j)}) \\ &= x_{(i,j)}I_c(c = c_j) - p(c|\mathbf{x}_j, \boldsymbol{\beta})x_{(i,j)} \\ &= x_{(i,j)}(I_c(c = c_j) - p(c|\mathbf{x}_j, \boldsymbol{\beta})) \end{aligned} \tag{6.3.16}$$

where $I_c(c = c_j)$ is an indicator function:

$$I_c(c = c_j) = \begin{cases} 1 & \text{if } c_j = c \\ 0 & \text{if } c_j \neq c \end{cases} \tag{6.3.17}$$

The derivative of the prior error can also be distributed through its parameters:

$$\begin{aligned}\nabla_{c,i}\mathrm{Err}_p(\boldsymbol{\beta},\sigma^2) &= \frac{d}{d\beta_{c,i}}\mathrm{Err}_p(\boldsymbol{\beta},\sigma^2)\\ &= -\frac{d}{d\beta_{c,i}}\sum_{c'=0}^{k-1}\sum_{i'=0}^{d-1}\log f_R(\beta_{c',i'}|\sigma_{i'}^2)\\ &= -\frac{d}{d\beta_{c,i}}\log f_R(\beta_{c,i}|\sigma_i^2)\end{aligned} \tag{6.3.18}$$

For the Gaussian prior:

$$\begin{aligned}-\frac{d}{d\beta_{c,i}}\log\left(\frac{1}{\sqrt{2\pi{\sigma_i}^2}}e^{-\frac{\beta_{c,i}^2}{2\sigma_i^2}}\right) &= \frac{1}{2}\frac{d}{d\beta_{c,i}}\log(2\pi\sigma_i^2) - \frac{d}{d\beta_{c,i}}\log e^{-\frac{\beta_{c,i}^2}{2\sigma_i^2}}\\ &= 0 - \frac{d}{d\beta_{c,i}}\left(-\frac{\beta_{c,i}^2}{2\sigma_i^2}\right)\\ &= \frac{\beta_{c,i}}{\sigma_i^2}\end{aligned} \tag{6.3.19}$$

For the Laplace prior:

$$\begin{aligned}-\frac{d}{d\beta_{c,i}}\log\left(\frac{\sqrt{2}}{2\sigma_i}e^{-\frac{\sqrt{2}|\beta_{c,i}|}{\sigma_i}}\right) &= -\frac{d}{d\beta_{c,i}}\log\frac{\sqrt{2}}{2\sigma_i} - \frac{d}{d\beta_{c,i}}\log e^{\frac{-\sqrt{2}|\beta_{c,i}|}{\sigma i}}\\ &= 0 - \frac{d}{d\beta_{c,i}}\left(-\frac{\sqrt{2}|\beta_{c,i}|}{\sigma_i}\right)\\ &= \begin{cases}\frac{\sqrt{2}}{\sigma_i} & \text{if } \beta_{c,i} > 0\\ -\frac{\sqrt{2}}{\sigma_i} & \text{if } \beta_{c,i} < 0\end{cases}\end{aligned} \tag{6.3.20}$$

For the Cauchy prior:

$$\begin{aligned}-\frac{d}{d\beta_{c,i}}\log\left(\frac{\lambda_i}{\pi(\beta_{c,i}^2+\lambda_i^2)}\right) &= \frac{d}{d\beta_{c,i}}\log(\pi) - \frac{d}{d\beta_{c,i}}\log(\lambda_i) + \frac{d}{d\beta_{c,i}}\log(\beta_{c,i}^2+\lambda_i^2)\\ &= 0 - 0 + \left(\frac{1}{\beta_{c,i}^2+\lambda_i^2}\right)\left(\frac{d}{d\beta_{c,i}}(\beta_{c,i}^2+\lambda_i^2)\right)\\ &= \frac{2\beta_{c,i}}{\beta_{c,i}^2+\lambda_i^2}\end{aligned} \tag{6.3.21}$$

With the above equations the error gradient can be calculated at every dimension $i \in \{0,...,n-1\}$ for every class $c \in \{0,...,m-1\}$.

6.3.3 Finding the Optimal Parameters from the Error Function

In [72] it is shown that for finite priors such as Gaussian, Laplace, and Cauchy density functions, the error function is concave, and a unique solution exists. The optimal parameters $\hat{\boldsymbol{\beta}}$ are found at the lowest point on the error function, and at this point the gradient of the error function will be zero in every dimension $i \in \{0, ..., n-1\}$ for every class $c \in \{0, ..., m-1\}$ [29, 72]:

$$\nabla_{c,i}\text{Err}_R(S, \hat{\boldsymbol{\beta}}, \sigma^2) = \nabla_{c,i}\text{Err}_l(S, \hat{\boldsymbol{\beta}}) + \nabla_{c,i}\text{Err}_p(\hat{\boldsymbol{\beta}}, \sigma^2) = 0 \tag{6.3.22}$$

Substituting the gradient of the likelihood error from (6.3.15) and (6.3.18) into (6.3.22):

$$\begin{aligned} &\nabla_{c,i}\text{Err}_l(S, \hat{\boldsymbol{\beta}}) + \nabla_{c,i}\text{Err}_p(\hat{\boldsymbol{\beta}}, \sigma^2) = 0 \\ -\sum_{j=0}^{n-1} &\Big(x_{(i,j)}(I_c(c = c_j) - p(c|\mathbf{x}_j, \hat{\boldsymbol{\beta}})) \Big) + \nabla_{c,i}\text{Err}_p(\hat{\boldsymbol{\beta}}, \sigma^2) = 0 \\ \Rightarrow \sum_{j=0}^{n-1} x_{(i,j)} p(c|\mathbf{x}_j, \hat{\boldsymbol{\beta}}) &= \sum_{j=0}^{n-1} x_{(i,j)} I_c(c = c_j) - \nabla_{c,i}\text{Err}_p(\hat{\boldsymbol{\beta}}, \sigma^2), \end{aligned} \tag{6.3.23}$$

at every dimension $i \in \{0, ..., n-1\}$ for every class $c \in \{0, ..., m-1\}$. For the various prior types, (6.3.19), (6.3.20) or (6.3.21) can be substituted into (6.3.23) as the prior error gradient. For example, if we use the Gaussian prior:

$$\sum_{j=0}^{n-1} x_{(i,j)} p(c|\mathbf{x}_j, \hat{\boldsymbol{\beta}}) = \sum_{j=0}^{n-1} x_{(i,j)} I_c(c = c_j) - \frac{\hat{\beta}_{c,i}}{\sigma_i^2} \tag{6.3.24}$$

6.3.4 Gradient Descent Algorithms

To find $\hat{\boldsymbol{\beta}}$, we have to locate the lowest point on the error function. Too many points exist to do this with a brute force method, so, instead, we use a gradient descent method for this task. The gradient descent algorithm requires an error function to be concave and to have a unique solution. If we use one of the priors mentioned in the sections above, these conditions are met, and the algorithm can be used.

The algorithm initiates by choosing an arbitrary point on Err_R, and gradually moves down the slope of the function until the minimum error is reached. Gradient descent is an iterative process. At each iteration $\nabla_{c,i}\text{Err}_R$ is calculated and the position of the point is updated accordingly. The point reaches the global minimum when it has a zero-gradient.

The gradient descent algorithm can be implemented in several ways. Two popular methods are batch- and stochastic gradient descent [29, 72]. Batch gradient descent finds the true gradient, i.e. the sum of the gradients from

each individual training example, and then updates $\boldsymbol{\beta}$ accordingly. Stochastic (or online) gradient descent uses a single training example at a time to update $\boldsymbol{\beta}$. This allows stochastic gradient descent to find $\hat{\boldsymbol{\beta}}$ faster than batch gradient descent for large training sets. A combination of the two methods ("mini-batches") is often used to produce a method that is more robust than the stochastic method, but faster than batch gradient descent.

An important factor in gradient descent is the step size, i.e. the amount by which the point is moved along the gradient after each iteration. If the step is large, the point will move in a "zig-zag" pattern towards the global minimum. If it is too small, it will take very long to find the global minimum. For these reasons, a dynamic step size is preferred. As the point moves in the same direction during consecutive iterations, it builds momentum and the step size increases. After each iteration, the error is compared to the previous point and, if it is higher, the step size is decreased and the point is rolled back to its previous position.

A maximum number of iterations are usually specified to prevent the process from taking too long to find a solution. A threshold can also be used to stop the process when the change in error is very small, indicating that the current parameter set is very close to the optimal solution $\hat{\boldsymbol{\beta}}$.

Chapter 7

Performance Quantification

There are many aspects to consider in a BCI. The processes range from feature extraction to classification, and each of them can be implemented with a variety of techniques. As is to be expected, BCI evaluation is not a straightforward task, and no method takes into account all the significant characteristics of such a system. To give an example, a BCI with perfect classification accuracy will not have much use if it has a response time of one minute.

Several ways have been proposed to measure the performance of a BCI. To test our system, we used three of them: the error rate (ERR), Cohen's kappa coefficient, and the mutual information (MI) [73, 74]. Only two-class datasets were chosen for this study. It was therefore possible to produce a control output with a time-varying signed distance (TSD). This is described in Section 7.1 below.

7.1 Time-varying Signed Distance

First, let us consider the classifiers used in this study. They were: linear discriminant analysis (LDA), support vector machines (SVMs), and logistic regression (LR). To summarise the models used by each of them, their transformation functions are presented in Table 7.1.

Table 7.1: LDA, SVM, and LR transformation functions.

LDA	$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + w_0$
SVM	$f(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b$
LR	$f(\mathbf{x}) = \frac{1-e^{-z}}{1+e^{-z}}$, where $z = \beta_0 + \beta_1 x_0 + ... + \beta_d x_{d-1}$

Now, if we define a feature set X that contains n trials, with each trial

having a duration of τ discrete time points:

$$X = \left\{ \mathbf{x}_t^{(i)} | i = \{0, ..., n-1\}, t = \{0, ..., \tau - 1\} \right\} \tag{7.1.1}$$

The TSD of a trial i in X at time t can be described by:

$$D_t^{(i)} = f(\mathbf{x}_t^{(i)}), \tag{7.1.2}$$

where $i \in \{0, ..., n-1\}$ and $t \in \{0, ..., \tau - 1\}$. Using the TSD, the trial is then assigned a class label using:

$$c_t^{(i)} = \begin{cases} class\ 1 & \text{if } D_t^{(i)} > 0 \\ class\ 2 & \text{if } D_t^{(i)} < 0 \end{cases}, \tag{7.1.3}$$

where $i \in \{0, ..., n-1\}$ and $t \in \{0, ..., \tau - 1\}$. The TSD includes the confidence of the classifier into the class assignment. This is determined by the magnitude $|D_t^{(i)}|$.

7.2 Accuracy, Error Rate, and Cohen's Kappa Coefficient

The classification accuracy ACC and error rate ERR for a given system are easy to calculate, but they do not take into account the probability of the correct classification by chance. ACC is given by

$$ACC = p_0 = \frac{number\ of\ correct\ classifications}{number\ of\ trials} \tag{7.2.1}$$

and $ERR = 1 - p_0$. The chance agreement p_e gives the estimated accuracy of a classifier if it was to randomly assign classes. For a problem with two-classes, a random classifier will have a chance agreement of 50%. To calculate p_e for a m-class problem with equal class probability, we can use the following formula:

$$p_e = \frac{100\%}{m}. \tag{7.2.2}$$

Cohen's kappa coefficient κ is also a measure of the classification accuracy, but it includes the prior and posterior probability in the calculation. The result is a coefficient of zero if the predicted classes have no correlation with the actual classes, and a coefficient of one if perfect classification is obtained. Kappa is calculated as [73]:

$$\kappa = \frac{p_0 - p_e}{1 - p_e}. \tag{7.2.3}$$

7.3 Mutual Information and System Response

The mutual information (MI) estimates the information transfer rate of a BCI and is measured in bits. It is based on Shannon's theorem of information transfer, and is calculated using the signal-to-noise ratio (SNR).

The SNR is calculated from the TSD, which is used to represent the classifier output of each trial at each time point. For two-class problems, the TSD would ideally be positive for all trials belonging to one class, and negative for the other. This is however not the case, because processes uncorrelated to the motor imagery task are also captured and influence the outcome of the classification. Using the notation defined in Section 7.1, the SNR at time point t is defined as [74]:

$$SNR_t = \frac{2 * var\left\{D_t^{\{\mathbf{L},\mathbf{R}\}}\right\}}{var\left\{D_t^{\{\mathbf{L}\}}\right\} + var\left\{D_t^{\{\mathbf{R}\}}\right\}} - 1, \tag{7.3.1}$$

where $var\{\cdot\}$ is the variance and $\{\mathbf{L}\}$, $\{\mathbf{R}\}$, and $\{\mathbf{L},\mathbf{R}\}$ are the indices for the trials of the left class, right class, and both classes respectively. The MI is easily calculated from the SNR as [74]:

$$I_t = 0.5 * log_2(1 + SNR_t) \tag{7.3.2}$$

and for two-class problems without trial rejection, the maximum MI is one bit.

The BCI's response to changes in mental activity also plays an important part in the information transfer rate. One way to measure the response is by finding the maximum steepness of the MI. The steepness of the MI (STMI) is calculated as [73]:

$$STMI_t = \frac{I_t}{t - t_{cue}}, \tag{7.3.3}$$

where t_{cue} is the time point at which the cue was presented. STMI is measured in bits/s.

Chapter 8

Experimental Investigation

To evaluate the methods investigated in this study, a BCI was developed for both of the feature extraction algorithms described in Chapter 5. For easy reference, the BCI using wavelet packet based features will be referred to as BCI_{wp} and the one using cepstral based features as BCI_{cep}.

A BCI has many aspects that require assessment. This experimental investigation focused on four of them: classification accuracy, robustness over multiple subjects, information transfer rate, and the response time of the system to changes in mental activity. Due to time constraints, new recordings of EEG data could not be made. Instead, a well known data repository was used that consisted of competitions that were specifically aimed at evaluating BCIs. The results of the past competitions were also published, and this provided us with a platform to compare BCI_{wp} and BCI_{cep} to the systems of other groups.

Each competition consisted of several challenges, and each challenge evaluated a different aspect of the BCI. Two challenges were chosen as the basis for our evaluation, and are represented by Experiment A and Experiment B respectively. During Experiment A, the overall accuracy of BCI_{wp} and BCI_{cep} was measured, and their robustness over multiple subjects were determined. In Experiment B, focus was placed on the information transfer rate and the system response. Both experiments used EEG signals as input, and the subjects were asked to use motor imagery to perform the required tasks. Different recording paradigms were used for the two experiments, and a detailed description of each is provided in Appendix A. It is important to note that the subjects were not allowed to move during the recording sessions, meaning that they could only use motor imagery as a means of control.

An open source toolbox referred to as BioSig [75] provided a set of functions that were specifically designed to assess the performance of a BCI. This toolbox was used in the competition to evaluate the submitted BCIs, and also in Experiments A and B to ensure that the conditions of the evaluation remained the same.

8.1 Experiment A: Robustness and Accuracy

Two very important aspects of any BCI are its classification accuracy and its ability to produce consistent results over multiple subjects. As mentioned in Chapter 5, the brain function of every person is different, and a BCI requires training for each individual subject. It is however desirable to keep the parameters of the underlying processes of a BCI the same (eg. feature extraction and classification), so that the system can adapt quickly to a new subject without the need for special calibration.

This experiment evaluates the robustness and accuracy of BCI_{wp} and BCI_{cep} over a set of nine subjects using a two-class dataset consisting of left and right hand motor imagery. The same set of training parameters were used for all subjects. The objectives are listed in Section 8.1.1 below.

8.1.1 Objectives

- *BCI Competition IV* provides a dataset containing the EEG data of nine subjects. Use the wavelet and cepstral based techniques to construct feature sets for each of the nine subjects.
- Split each feature set into train and test sets and use the three classifiers discussed in Chapter 6 (LDA, SVM, LR) to evaluate them.
- Calculate the ERR of the test data over the duration of the trial. Compare the results between BCI_{wp} and BCI_{cep}, as well as the results obtained between different classifiers. Also make a comparison between the different subjects.
- Compare BCI_{wp} and BCI_{cep} to the BCIs used by the other groups in the competition.

8.1.2 Dataset Details

The dataset was provided by the *BCI Competition IV* repository [55], and consisted of nine subjects with five recording sessions for each subject. Each session contained between 120 and 160 trials, giving a total of 720 to 800 trials per subject. The first two recording sessions did not contain any feedback. Instead, they were used to train a BCI that provided feedback in the last three sessions. Paradigm $P1$ (described in Appendix A) was used for the recording runs without feedback, and paradigm $P3$ for the trials with feedback. The signals were sampled at a rate of 250Hz and filtered between 0.5Hz and 1kHz. A notch filter was also applied at 50Hz. EEG recordings were made at electrode positions C3, Cz and C4 (see Fig. 5.1 for reference) and the parameters t_{start}, t_{ready}, t_{cue}, and t_{end} were $0s$, $2s$, $3s$ and $7.5s$, respectively.

8.1.3 Parameters and Setup

To test BCI_{wp} and BCI_{cep} on each subject, the trials from the five recording sessions were pooled together and 10-fold cross validation was performed on the combined feature sets. To ensure an unbiased result, the evaluations took place under double blind conditions, and the class labels of the trials were kept hidden from the feature extraction and classification methods.

BCI_{wp} extracted its features with a FE window length (l_w) of $2s$, and each window segment was decomposed to a depth of three levels. A fourth order *Daubechies* wavelet was used, and the average bandpower in the last second was used to generate the feature vectors ($l_{avg} = 1s$). BCI_{cep} calculated the power cepstrum from a FE window of $0.25s$ in length, and constructed its features from the first 16 cepstral coefficients. Both BCI_{wp} and BCI_{cep} used three classifiers (LDA, SVM, and LR) for the evaluation of the feature sets. A Soft Margin SVM (with $C = 1$) was used with a non-linear kernel (Radial Basis function with $\gamma = 0.25$), and for the LR classifier a Gaussian prior was used.

After feature extraction, BCI_{wp} and BCI_{cep} were trained using a segment in the train features between t_{start} and t_{end}. The segment was chosen based on the assumption that the optimal train data was where the motor imagery task had its maximum strength. Parameters t_{start} and t_{end} were therefore chosen as 4000ms and 4500ms accordingly (between 1s and 1.5s after cue onset). It was also realised that channel Cz did not make any relevant contribution in the constructed feature set. It was therefore discarded, which meant that only channels C3 and C4 were used. The feature vectors were thereby 16-dimensional for BCI_{wp} and 32-dimensional for BCI_{cep}.

Normalisation of the raw EEG signals and constructed feature sets did not show any significant difference to the outcome of the experiment. It was therefore not included in the experiment.

8.1.4 Results

An LDA, SVM, and LR classifier was used to evaluate the constructed feature sets, and for each subject, 10-fold cross validation was performed with each of the three classifiers. A time course of the ERR over all test trials was obtained for each fold, and the mean time course for each subject was calculated across the ten folds. The time courses of the classifications with the LDA classifier is presented in Fig. 8.1, and the time courses for the SVM and LR are provided in Appendix B. The subfigures in Fig.8.1 consist of two plots over time. One plot shows the ERR for BCI_{wp}, and the other a time course for BCI_{cep}. The time courses show how the error rate drops after cue presentation (t=3s) and continues to do so until a minimum point is reached where the motor imagery is at it's strongest. A short delay is noted before the ERR begins to improve after cue presentation.

The minimum ERR from each of the ten folds were found, and their average is presented in Table 8.1. The second, third, and fourth columns show the minimum ERR for BCI_{wp}, and the last three columns show the minimum ERR for BCI_{cep}. The last row in the table shows the mean ERR from each column. It should be noted that the time plots provide the average time course over ten folds, whereas Table 8.1 uses the mean value calculated from the minimum ERR obtained in each individual fold.

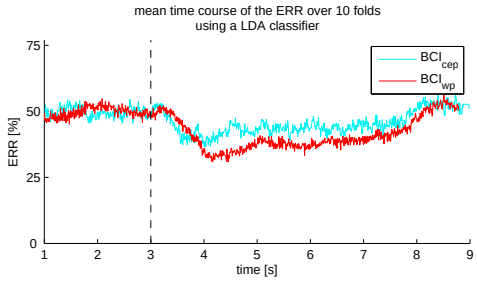
Table 8.1: Mean error rates obtained by finding the lowest point in the time course of each fold (after cue presentation). Columns 2, 3, and 4 show the ERR for BCI_{wp}, and columns 5, 6, and 7 for BCI_{cep}, using each of the three classifiers respectively.

	minimum ERR for BCI_{wp}			minimum ERR for BCI_{cep}		
Subject	LDA	LR	SVM_{RBF}	LDA	LR	SVM_{RBF}
S1	23.06	23.89	27.36	26.53	26.67	25.83
S2	29.56	27.94	36.03	30.88	31.18	29.85
S3	31.25	32.64	36.67	30.42	31.11	31.25
S4	3.38	4.50	4.88	9.37	9.50	9.25
S5	12.57	13.11	20.27	21.49	20.95	21.76
S6	20.69	23.82	35.83	27.64	26.11	27.36
S7	19.58	20.56	36.11	25.56	24.31	25.83
S8	16.18	15.79	20.79	17.50	17.37	20.39
S9	18.33	17.22	20.69	23.47	23.61	22.78
mean	**19.40**	**19.94**	**26.51**	**23.65**	**23.42**	**23.81**

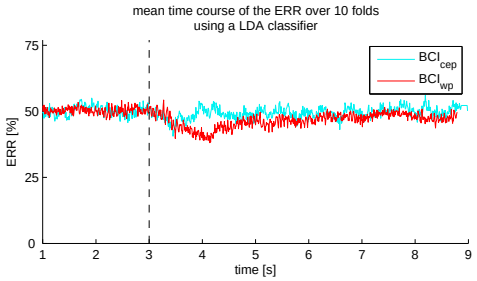
The overall performances of the subjects were very different. Subject S4 obtained the highest classification rate and had an overall ERR below 10%. Another four subjects (S5, S7, S8, and S9) achieved error rates below 20%; three more (S1, S2, and S6) below 30%; and one (S3) above 30%.

The minimum ERRs for the wavelet based BCIs were significantly lower than those for the cepstral based BCIs. BCI_{cep} did however show a slightly faster response after cue presentation than BCI_{wp}. The patterns of the time courses in BCI_{wp} and BCI_{cep} were very similar on subject level. Both types either obtained good results for a given subject or both of them performed badly. This would indicate that similar features were obtained from the EGG signals by both BCI types.

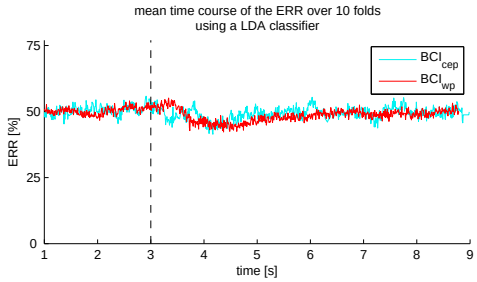
If we compare the three classifiers, we see that LDA and LR achieved very similar results. The SVM performed worse, however, it should be noted that this may be due to the chosen kernel and not the classifier itself.



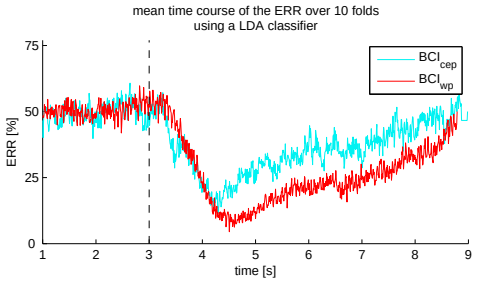
(a) S1



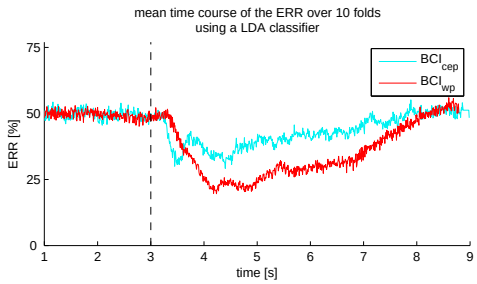
(b) S2



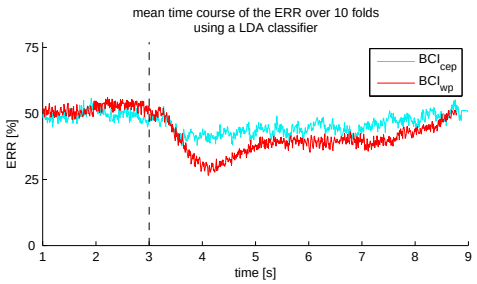
(c) S3



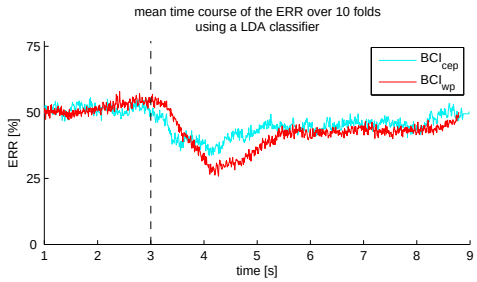
(d) S4



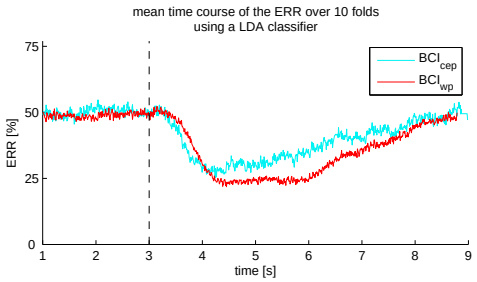
(e) S5



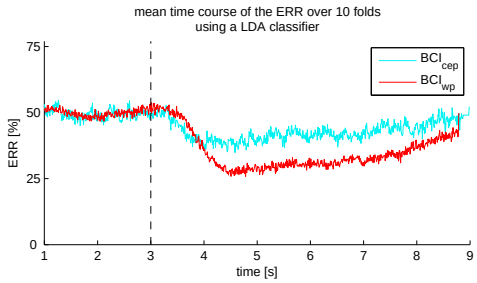
(f) S6



(g) S7



(h) S8



(i) S9

Figure 8.1: Time courses of the ERR for each subject calculated from the mean TSD of the ten test folds. An LDA classifier was used by both BCI_{wp} and BCI_{cep} in this figure. The dotted line at $t = 3s$ indicates the cue onset.

8.1.5 Results Compared to Competition Outcome

For the competition, the contestants were given all the EEG data from the nine subjects, but the true class labels of the last two sessions were kept unknown. The competition objective was to train a BCI with the first three sessions and then classify the trials of the remaining two. A time course of the kappa was calculated from the test data, and the winner was the BCI with the highest maximum kappa over all subjects.

To compare BCI_{wp} and BCI_{cep} to the BCIs of the competition, the third recording session was chosen for training, and sessions four and five were classified accordingly (sessions one and two were discarded). The same training parameters were used as those presented in Section 8.1.4, and an LDA classifier was selected for both BCI_{wp} and BCI_{cep}.

A time course of the kappa was obtained for each test trial and the mean values across the time courses were calculated at each time point. These results, together with those of the competition, are presented in Table 8.2. The results were obtained the same way as in the competition, using the functions provided by the BioSig [75] toolbox. The table shows the maximum kappa of each subject, and the first row indicates the type of features used (CSP: Common Spatial Patterns, WP: Wavelet Packets, CEP: Cepstrum, and BP: band power calculated with unspecified method). The rows and columns each represent a subject and contestant respectively, and the mean kappa of each contestant is shown in the last row.

An earlier version of BCI_{wp} was entered into the competition when it was held in 2008. This entry is represented by contestant D. The BCI entered used the same methods as BCI_{wp}, however, development was still in progress at the time of submission and further improvements were pending. Even so, it achieved an overall fourth place.

On subject level, BCI_{wp} and BCI_{cep} obtained acceptable results except for two subjects, S2 and S3, who could only obtain kappa values of 0.21 and 0.14 for BCI_{wp}, and 0.16 and 0.17 for BCI_{cep}. These two subjects were also the worst performers for most of the other groups, which leads us to assume that they may not have been capable of executing the motor imagery tasks in the required manner.

A mean kappa of 0.51 over all subjects was obtained by BCI_{wp}, which is 0.09 lower than the best performer and 0.08 higher than our original entry. BCI_{cep} could only obtain a mean kappa of 0.41 over the nine subjects, however, it still outperformed groups E and F, who had a mean kappa of 0.37 and 0.25 respectively.

By comparing the feature extraction methods used, we see that the first three contestants used CSP techniques, and the fifth and sixth contestants used the signal bandpower (however they did not specify their method).

Table 8.2: Results from *BCI Competition IV* (2008) showing the mean kappa for each subject. The last two columns provide the results of BCI_{wp} and BCI_{cep}, and the last row shows the mean kappa across all subjects. *Contestant D represents our entry to the competition with an earlier version of BCI_{wp}.

	A	B	C	D*	E	F	BCI_{wp}	BCI_{cep}
feature type	CSP	CSP	CSP	WPD	BP	BP	WPD	CEP
S1	0.40	0.42	0.19	0.23	0.20	0.02	0.41	0.26
S2	0.21	0.21	0.12	0.31	0.16	0.09	0.21	0.16
S3	0.22	0.14	0.12	0.07	0.16	0.07	0.14	0.17
S4	0.95	0.94	0.77	0.91	0.73	0.43	0.90	0.79
S5	0.86	0.71	0.57	0.24	0.21	0.25	0.60	0.48
S6	0.61	0.62	0.49	0.42	0.19	0.00	0.48	0.31
S7	0.56	0.61	0.38	0.41	0.39	0.14	0.49	0.34
S8	0.85	0.84	0.85	0.74	0.86	0.76	0.82	0.70
S9	0.74	0.78	0.61	0.53	0.44	0.47	0.55	0.47
mean	**0.60**	**0.59**	**0.46**	**0.43**	**0.37**	**0.25**	**0.51**	**0.41**

8.1.6 Discussion

In sections 8.1.4 and 8.1.5, BCI_{wp} and BCI_{cep} showed that both wavelet and cepstral based features are capable of discriminating between different classes of motor imagery. Some subjects (especially S2 and S3) did however not perform very well. Table 8.1 shows that subject S4 achieved the best results and had error rates below 5% for BCI_{wp} and below 10% for BCI_{cep}. Subjects S2 and S3 performed the worst, but still achieved error rates below 35% for both BCI_{wp} and BCI_{cep}. The remaining subjects obtained error rates between 10% and 30%, which indicated that they were also capable of controlling a device with the use of motor imagery.

As mentioned in Section 8.1.3, the same parameters were used for all the subjects. We see, however, that the techniques described are not yet robust enough for this, and individual calibration is required to provide higher classification rates for the lower performing subjects.

The SVM was implemented with a RBF kernel, which resulted in worse results than the other two classifiers. A different kernel could have produced better results, and the classifier itself was not necessarily unsuitable for classifying this feature type. The outcome does however show that the RBF is not a suitable kernel for either of these feature types.

To summarise this experiment, we see that wavelet-packet and cepstral based techniques are capable of producing results on the same level as more commonly used techniques (such as CSP), but further work is needed to im-

prove their robustness.

8.2 Experiment B: Information Transfer Rate and System Response

In Experiment A the accuracy and robustness of BCI_{wp} and BCI_{cep} was evaluated. The accuracy plays a very important part in a BCI, however, perfect classification rates will mean nothing if the system takes very long to produce a control output. The response of a BCI has a direct impact on its information transfer rate, and ultimately, this is the most important aspect of any BCI.

The goal of this experiment was to calculate the system response and information transfer rate of BCI_{wp} and BCI_{cep} in order to determine the quantity and speed at which information can be transferred by the proposed methods. A challenge from the *BCI Competition II* [37] was used for the evaluation, and the objectives of the experiment is listed in Section 8.2.1 below.

8.2.1 Objectives

- Using the data provided, construct a feature set with the wavelet and cepstral based techniques described in Chapter 5.
- Use the three classifiers (LDA, SVM, and LR) to evaluate each feature set and calculate the time course of the mean ERR and MI for each of them. Also find the point in each time course where the maximum MI occurs.
- Determine the system response by calculating the maximum steepness of the mutual information (STMI) for each feature type between $t = 4s$ and $t = 9s$.
- An outcome of the competition is provided in [76]. Use this to compare BCI_{wp} and BCI_{cep} to the BCIs of the competition.

8.2.2 Dataset Details

The dataset was obtained from the *BCI Competition II* online repository [37], and contains the EEG data of a single subject recorded over several runs on the same day. A two-class problem is presented and the subject was asked to perform either left or right hand imagined movements. Paradigm $P2$ (described in Appendix A) was used for the recording runs. There were 280 trials in total: 140 for tasks involving left hand motor imagery and 140 for right hand motor imagery. The signals were sampled at a rate of 128Hz and filtered between 0.5Hz and 30Hz. EEG recordings were made at positions C3, Cz, and

C4 (see Fig. 5.1), and the parameters t_{start}, t_{ready}, t_{cue}, and t_{end} were $0s$, $2s$, $3s$, and $9s$ respectively.

8.2.3 Parameters and Setup

The contestants received both train and test datasets, but the class labels for the test data were kept unknown. The competition had however already ended by the time of this experiment, and all the labels were made available. Therefore, to ensure unbiased results the dataset was shuffled and new train and test sets were assigned during the evaluation. The ratio between train and test data was kept the same (140 trails for training and 140 trials for testing) and each set contained an equal number of trials for left and right hand motor imagery. Feature extraction and testing were performed under double blind conditions, and the evaluation was repeated five times to ensure a fair comparison. The mean result from the five evaluations was used for comparisons and the results were obtained the same way as in the competition, using the functions provided by the BioSig [75] toolbox.

The wavelet based BCI had a FE window length (l_w) of $2s$, and l_{avg} was chosen as $1s$. A fourth order *Daubechies* wavelet was chosen for the extraction, and the signal was decomposed to a depth of three levels. BCI_{cep} calculated the power cepstrum of the signal over a $0.25s$ window and constructed its features from the first 16 cepstral coefficients. BCI_{wp} and BCI_{cep} used three classifiers each to train and test their feature sets accordingly. LDA, SVM, and LR classifiers were used. A Soft Margin SVM (with $C = 1$) was used with a non-linear kernel (Radial Basis Function with $\gamma = 0.25$), and for the LR classifier a Gaussian prior was used.

To train the BCIs, a time segment was chosen where the motor imagery was assumed to cause maximum ERD over the motor cortex. From visual inspection of the training data, t_{start} and t_{end} were selected as $4s$ and $4.5s$ respectively. Only channels C3 and C4 were used, resulting in 16-dimensional features for BCI_{wp} and 32-dimensional features for BCI_{cep}.

Normalisation of both EEG signals and constructed features were investigated, but a significant difference could not be seen in the final outcome of the evaluation. For the sake of simplicity, normalisation was therefore excluded from the experiment.

8.2.4 Results

Time courses of the ERR and MI for each classifier and feature type combination are shown in Fig. 8.2(a)-(f). In Fig. 8.2(a), (c), and (e) we see how the error rate drops a short time after cue presentation ($t_{cue} = 3s$) and continues to drop until reaching a point where the mental task is performed at its maximum strength. Time courses of the MI show the same effect. In Fig. 8.2(b), (d), and (f) the MI remains around zero until cue presentation where it climbs

steadily until reaching a peak at the same time when the error rate reaches its lowest point.

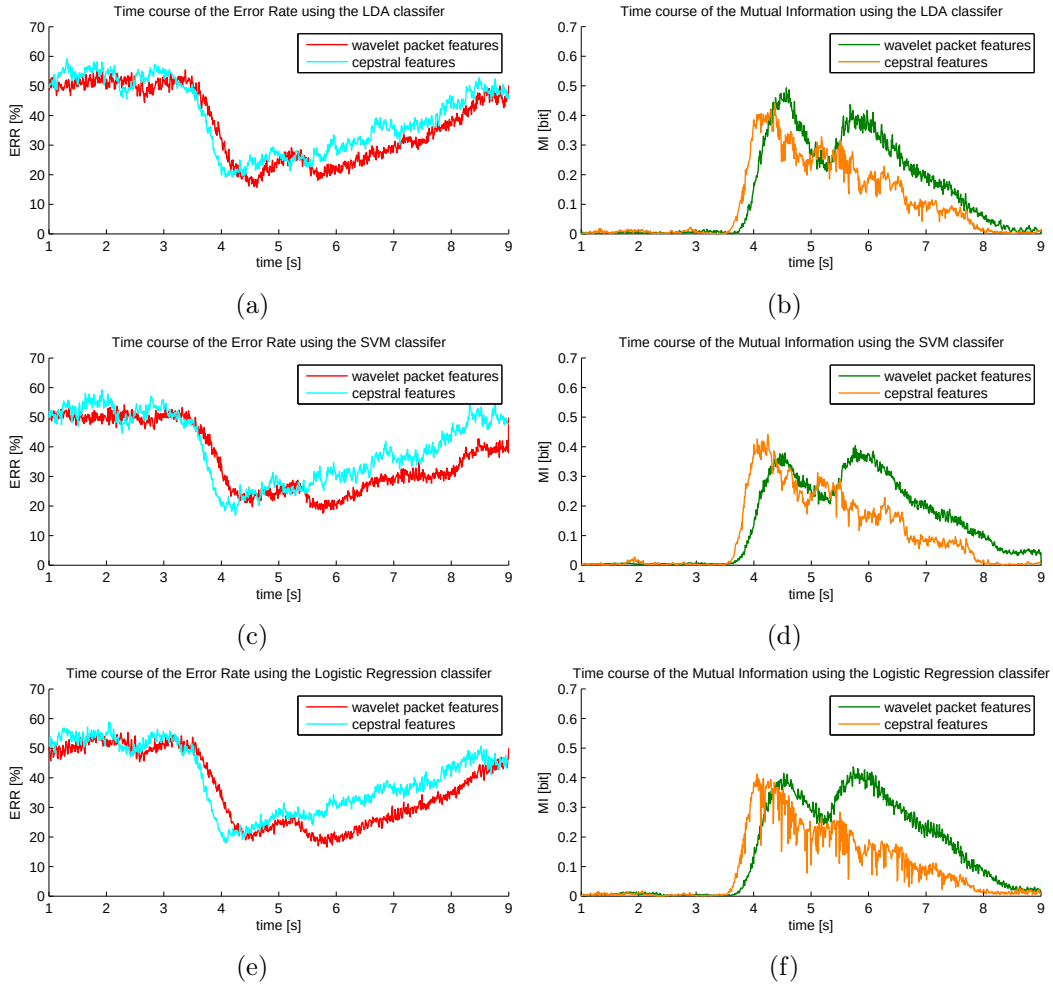


Figure 8.2: Time courses of the ERR and MI of each classifier and feature type combination. Three classifiers were used by BCI_{wp} and BCI_{cep}, namely: LDA, SVM, and LR.

Table 8.3 provides the minimum ERR and maximum MI for each time course, as well as the maximum STMI for each combination of classifier and feature type. The minimum ERR for each combination ranges from 13.14% to 16.29%, and the maximum MI from 0.42 to 0.52 bits. The cepstral features showed a faster response to changes in mental activity than the wavelet features; however, the latter achieved a better overall accuracy. The lowest ERR was found using an LDA classifier for BCI_{wp} and a SVM for BCI_{cep}. The classification rates in general were very close across all combinations (ERR = 15.14% $\pm \sim$ 2% to either side), and a similar pattern over time can be seen for all time courses.

Table 8.3: Summary of the results obtained during the evaluation of each classifier/feature type combination. Columns three to six present the minimum ERR, maximum MI, time of maximum MI, and maximum steepness of the MI.

Classifier	Feature Type	Minimum ERR [%]	Maximum MI [bit]	time [s]	Maximum STMI [bit/s]
LDA	wavelet	13.14	0.52	4.58	0.33
SVM	wavelet	15.57	0.42	5.56	0.26
LR	wavelet	14.57	0.46	5.64	0.28
LDA	cepstral	16.14	0.46	4.26	0.39
SVM	cepstral	15.14	0.45	4.15	0.40
LR	cepstral	16.29	0.45	4.14	0.39

A noticeable difference exists, however, between the system response of the two feature types. The response, measured by the STMI, shows that cepstral features produced an average bit rate of 0.39 $bits/s$, whereas wavelet features could only achieve 0.29 $bits/s$.

8.2.5 Results Compared to Outcome of the Competition

During the competition, the MI was used to calculate the information transfer rate, and participants were ranked according to the maximum MI that they could produce. Fig. 8.3 was obtained from [76], and shows the time course of the MI for each group from the competition. Nine groups took part; however, one of them (group H) did not provide the required information or produce the correct output. Due to this, their results will not be considered. The results of the eight remaining groups ranged from ERR=32.14%, MI=0.09 bits, and STMI=0.05 $bits/s$ to ERR=10.71%, MI=0.61 bits, and STMI=0.39 $bits/s$. A summary of the results for each group is presented in Table 8.4.

The MI of groups B and C climbs steadily and does not fall like the other groups; however, they reach their peak much later than the other groups. Group C has the lowest ERR and the highest MI, but, due to a slower response, it only reaches a maximum STMI of 0.24 $bits/s$. Groups A and F achieve a maximum STMI of 0.39 and 0.28 $bits/s$ respectively. For most groups, the MI starts climbing around $4s$ and peaks between $4.5s$ and $6.5s$. Group A peaks earlier at $4.18s$, whereas groups B and C only reach their peak MI at $6.70s$ and $7.59s$ respectively. The pattern over time for A and F are similar, but a delay of roughly $0.5s$ exist between them. The same pattern over time is also observed in Fig. 8.2 with BCI_{wp}.

Both feature types were compared to the results provided by the outcome of the competition, and the optimal classifier was used for the BCI of each feature type (LDA for wavelet and SVM for cepstrum). By comparing the MI of BCI_{wp} and BCI_{cep} to those of the other groups, we see that BCI_{wp} (max

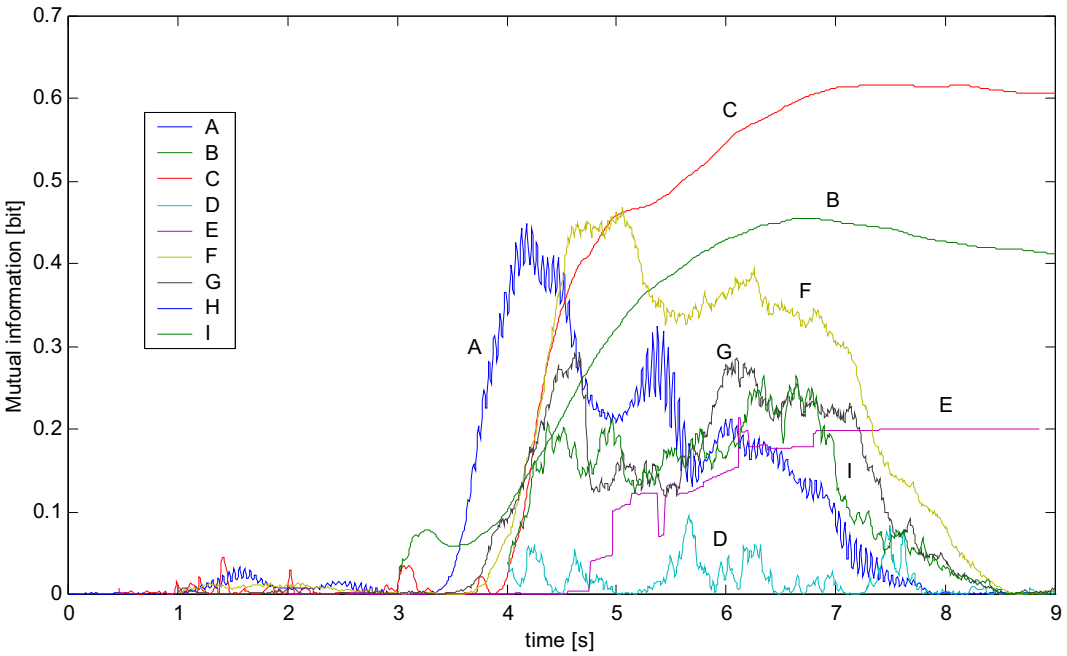


Figure 8.3: Time courses of the MI for each contestant in the *BCI Competition II* challenge (figure provided by [76]).

Table 8.4: Outcome of the *BCI Competition II* challenge. Columns three to six present the minimum ERR, maximum MI, time of maximum MI, and maximum steepness of the MI for each contestant.

Group	Minimum ERR [%]	Maximum MI [bit]	time [s]	Maximum STMI [bit/s]
C	10.71	0.61	7.59	0.24
F	15.71	0.46	5.05	0.28
B	17.14	0.45	6.70	0.16
A	13.57	0.44	4.18	0.39
G	17.14	0.29	4.66	0.19
I	23.57	0.26	6.34	0.15
E	17.14	0.21	6.13	0.07
D	32.14	0.09	5.66	0.05
~~H~~	~~49.29~~	~~0.00~~	~~9.00~~	~~0.00~~

MI=0.52 bits) outperformed all participants except for group C which had a maximum MI of 0.61 bits. BCI_{cep} achieved a maximum MI of only 0.45 bits, but it still managed to achieve better results than six of the other groups.

8.2.6 Discussion

BCI_{wp} showed a lower ERR than BCI_{cep} (average of 14.43% compared to 15.86%), but the classification rates of the evaluations were very close, and one feature type cannot be favoured above the other. It should be noted that the parameters were chosen manually, and it is possible that higher classification rates could have been obtained if a better parameter set was found for either of the feature types.

Although the error rates were similar for BCI_{wp} and BCI_{cep}, a clear difference is seen in their response time. From Fig. 8.2 we see that BCI_{cep} responded faster to the motor imagery than BCI_{wp}. This can be explained by the difference in window lengths used by BCI_{wp} and BCI_{cep}. The cepstral based features were calculated with a FE window length of $0.25s$, whereas BCI_{wp} used the average band power over $1s$. The longer window was required by BCI_{wp} to counteract the influence of noise in the frequency bands.

The difference in response has a big influence on the STMI, as we can see from Table 8.3. BCI_{cep} showed a much higher STMI than BCI_{wp} (avg. of 0.38 $bits/s$ over all classifiers compared to avg. of 0.22 $bits/s$ for BCI_{wp}), even though it had a worse ERR.

If we compare our results to those of the competition, we see that three contestants (A, F, and G) showed time courses similar to those of BCI_{wp} in Fig.8.2. It is likely that the feature extraction methods were similar between us and these groups.

The maximum STMI produced in the competition was 0.39 $bits/s$, achieved by group A. BCI_{wp} and BCI_{cep} produced a maximum STMI of 0.33 $bits/s$ and 0.40 $bits/s$ respectively. The importance of the system response is clearly illustrated by these results. Although having a lower classification rate and MI, BCI_{cep} was still capable of producing a higher bit rate than BCI_{wp}.

Chapter 9

Conclusions and Recommendations

9.1 Summary

This study investigated two feature extraction methods and three classification techniques that can be used to discriminate between different classes of motor imagery. The first feature extraction technique made use of the signal band power to extract features, and used the wavelet packet decomposition to obtain the spectrum accordingly. The wavelet based features could effectively be used to classify different classes of motor imagery; however, some subjects performed better than others, and all subjects did not achieve a desirable level of results. None the less, error rates below 5% and information transfer rates of up to 0.33 $bits/s$ were obtained by some.

The second feature extraction technique involved the cepstral analysis of a signal. The power cepstrum was chosen to calculate the cepstra over a localised time segment in the signal, and the quefrency coefficients were used as feature vectors. Although a conclusive test was not done to directly compare the two feature types, cepstral based features could not achieve the same classification rates as wavelet based features. They did however show a faster response to changes in mental activity, which resulted in them obtaining a higher information transfer rate than the wavelet based features. Error rates below 10% and information transfer rates of up to 0.4 $bits/s$ were obtained by the cepstral features.

To evaluate the constructed features, three machine learning techniques were investigated. They were: linear discriminant analysis (LDA), logistic regression (LR), and support vector machines (SVM). The SVM used the Radial Basis function as a kernel, and was the only non-linear classifier in this study. Although being the simplest technique of the three, the LDA classifier achieved the lowest error rates across both data sets. It was however closely followed by the LR classifier. The SVM obtained low classification rates in the

first experiment, but achieved results similar to the LDA and LR classifiers in the second. The lower performance of the SVM could have been a result of the chosen kernel, and the classifier may have done better if something different was chosen.

9.2 Recommendations for Future Work

Several techniques for each aspect of a BCI were investigated. It was however not possible or feasible to perform an in-depth implementation of each method, and many improvements can be made on the existing system. A few of them are provided here.

9.2.1 Automated Calibration

Probably the most important aspect to consider in work following from this study is an automated calibration system. Parameters for the experiments were chosen manually based on a set of assumptions. Manual selection was a tedious process and did not necessarily produce the best results. An automated calibration system was planned for, but due to time constraints one could not be developed. A calibration system would save time during evaluation, and would have the added benefit of most likely producing better results. The system would ideally optimise the following parameters:

- The train segment defined by t_{start} and t_{end}.
- Number decomposition levels for the WPD.
- Feature extraction window length l_w and l_{avg}.
- Cepstral coefficients to select from the quefrency domain.

9.2.2 Wavelet Packet Best Basis

The Best Basis algorithm has been found to provide an optimal representation of a (discrete time) signal in its frequency domain, and has successfully been applied to the processes of a BCI [47, 48, 49]. The algorithm makes use of wavelet packets to find the best combination of decomposition levels. Adding the algorithm to the wavelet based methods discussed in this study could therefore result in a possible increase to the overall performance of the system.

9.2.3 Wavelet Based Cepstrum

The cepstrum obtained in this study made use of the Fourier transform during its calculations. However, as mentioned in Chapter 4, other frequency analysis techniques can also be used. An interesting experiment in future work would

be to investigate the wavelet based cepstrum and construct features from the coefficients accordingly.

This and other forms of cepstrum could also be investigated as a filtering step before the actual feature extraction.

9.2.4 Dimension Reduction and DSLVQ

The feature vectors created by BCI_{wp} and BCI_{cep} varied from 16-dimensional to 32-dimensional in size depending on the parameters selected. However, many of these dimensions were redundant and could have been discarded (the classifiers assigned weights of zero to them). Applying dimension reduction to the feature vectors would increase the speed of the evaluations dramatically. Computational speed is a very important requirement in real-time systems, and although dimension reduction would not necessarily improve the classification rate, this aspect is just as crucial as any other. LDA is a popular technique and could serve as an option for dimension reduction in the preprocessing steps.

Another suggested method would be to use a Distinction Sensitive Learning Vector Quantiser (DSLVQ). This technique selects a subset of features from the feature set based on their relevance to the discrimination of different classes. DSLVQ has already been investigated in BCI research and has proved to be very useful [77]. DSLVQ also has another added advantage; different combinations of the available information can be fused together and incorporated into the feature vectors. Although this means that the resulting feature set will most likely contain redundant dimensions, the DSLVQ algorithm will discard them and the possibility of finding new information from the data will be greatly increased.

9.3 Conclusions

During this study, two feature types and three classifiers were investigated with the aim of finding effective techniques for the use in online Brain Computer Interfaces. The wavelet and cepstral based BCIs, referred to as BCI_{wp} and BCI_{cep}, both showed that they were capable of discriminating between left- and right-hand motor imagery.

BCI_{wp} and BCI_{cep} were tested over a multiple of subjects and the results were very subject specific. Some subjects performed exceptionally well (mean error rates below 5% and 10% for BCI_{wp} and BCI_{cep} respectively); however, acceptable classification rates could not be found for all of them. This could be due to the selection of the training parameters, or the BCIs simply did not posses the ability to find sufficient information in the EEG signals. Whatever the reason, more work is required to improve the robustness and accuracy of both systems on a subject level.

The linear classifiers (LDA and LR) obtained similar results in both experiments, but the same performance levels could not be achieved by the SVM in the first experiment. In Experiment B, however, the SVM achieved similar results to the LDA and LR classifiers. This may indicate that it had been impaired by its selected parameters in Experiment A. Either way, we can conclude that the Radial Basis function as kernel did not provide any better results for the described feature types than the linear classifiers. A more suitable kernel may however lead to the SVM outperforming the linear classifiers.

The main conclusion that we can make from this study is that both the wavelet-packet and cepstral based techniques are capable of producing results of similar nature to those of other widely used techniques (such as Common Spatial Patterns), but more work is required to refine them. The systems implemented were very simple, and with further development they are likely to produce better results.

Appendices

Appendix A

EEG Recording Paradigms

The paradigms in this appendix describe how the recordings are made in order to obtain the EEG datasets used by BCIs. References to experiments using similar paradigms are given in [52, 53, 74, 78, 79, 80, 81]. All paradigms described present two-class data, and are applied to single-trial classification problems. Data is captured over several sessions (usually one session per day), and each session contains several runs with several trials per run.

A.1 *P1*: Cue-Based Without Feedback

This cue-based paradigm without feedback is used for training a BCI when no subject data is available. Consequently, no feedback is provided. The subject sits in a relaxing chair with a computer screen placed at eye level roughly one meter in front of him. Each trial starts with a blank screen, and at t_{ready} a fixation cross appears in the middle of the screen and a short acoustic beep warns the subject that the cue is about to be presented. At t_{cue} the cue is presented. It can either be displayed as an arrow pointing left or right, or be a low/high acoustic tone representing the left/right class. Depending on the cue, the subject has to imagine moving his left or right hand for as long as possible. It is very important that no physical movement takes place. At t_{end}, the trial ends and the screen goes blank. A random break is added before the next trial starts to prevent adaptation.

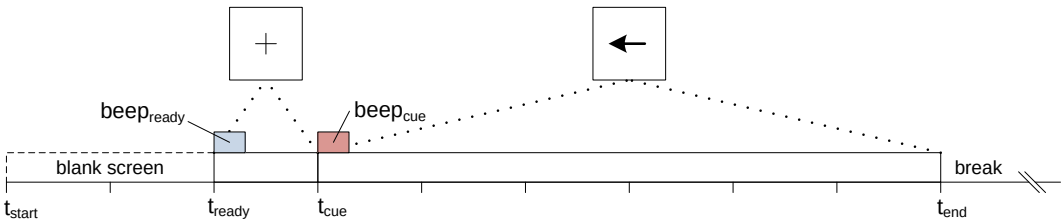


Figure A.1: Cue-based paradigm without feedback.

A.2 *P2*: Horizontal Bar

This paradigm is similar to *P1*, with the difference that feedback is provided during sessions. The subject is instructed to move a bar left or right in a horizontal direction based on a presented cue. The subject sits in a relaxing chair with armrests, and a screen is placed approximately one meter in front of him at eye level. The trial starts with a blank screen. At t_{ready}, a short acoustic beep is sounded and a fixation cross is displayed. At t_{cue}, an arrow pointing either left or right acts as a visual cue instructing the subject which side the control bar should be moved to. The subject then has to move the bar using left and right hand motor imagery accordingly, and keep it on the requested side of the screen for as long as possible. At t_{end} the trail ends and the screen becomes blank. A random break is added before the next trial starts. Feedback is provided by constructing features from the recorded data and classifying them with a trained classifier.

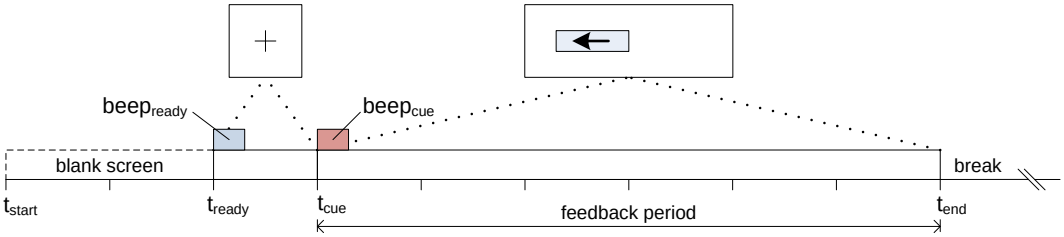


Figure A.2: Cue-based paradigm with horizontal feedback.

A.3 *P3*: Smiley

This paradigm is based on a smiley face image moving in the horizontal direction of imagined left and right hand movements. The subject sits in a relaxing armchair with a flat screen placed approximately one meter in front of him. At t_{start}, a gray smiley appears in the middle of the screen, followed by a warning beep at t_{ready}. At t_{cue}, a visual cue in the form of a green vertical bar on either the left or right side of the screen is presented to indicate the requested class. The subject then has to move the smiley in the direction of the cue for as long as possible by using left or right hand motor imagery. The integrated magnitude of the classifier output over the last two seconds determines the horizontal offset from the centre of the screen. The smiley becomes green if it moves towards the cue and red if it does not. At t_{end} the screen goes blank and a random break is added to the trial before the next one starts.

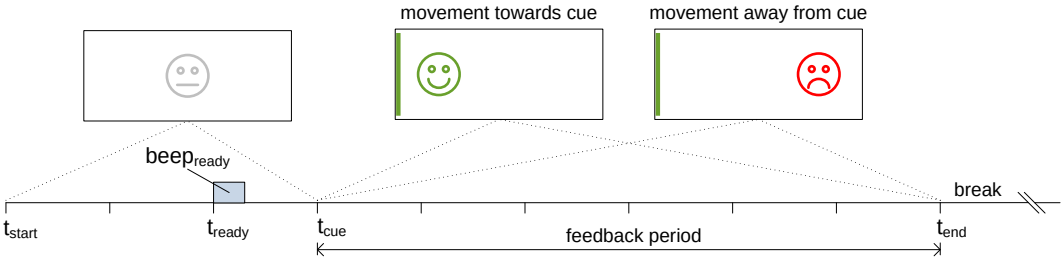
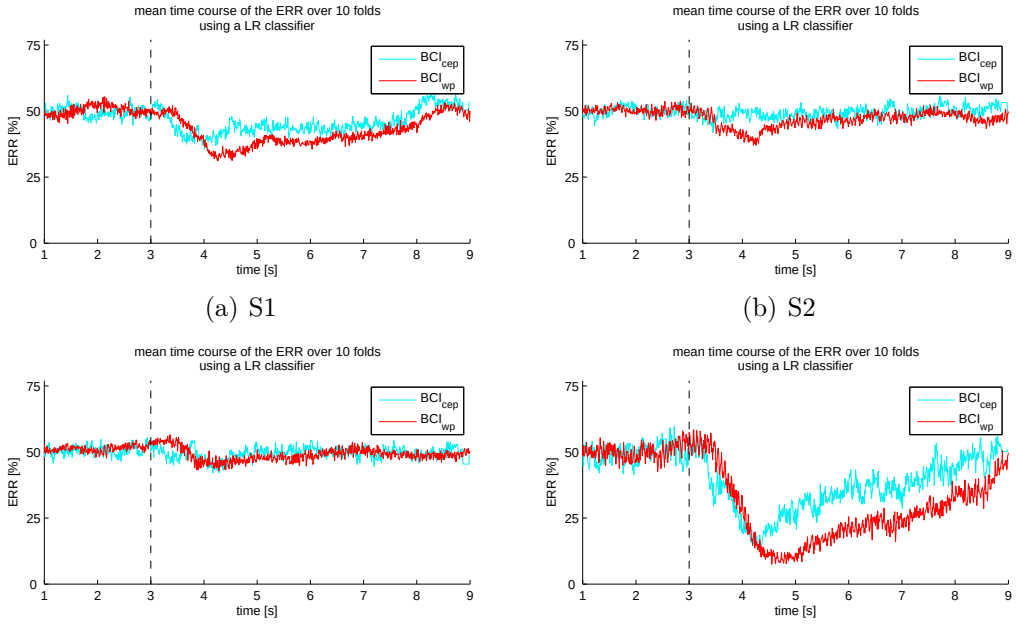


Figure A.3: Smiley paradigm

Appendix B

Additional Results: Experiment A

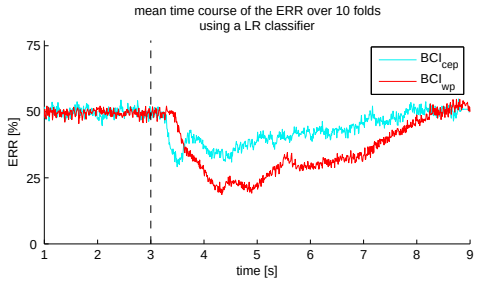
During Experiment A, a SVM and LR classifier was used to evaluate the constructed feature sets for each subject. Ten-fold cross validation was performed and the mean time course across the ten folds was obtained for each subject. Time courses for the BCIs using the LR and SVM classifiers are presented in Fig. B.1, Fig. B.2, and Fig. B.3. The subfigures consist of two plots over time. One plot shows the ERR for BCI_{wp} and the other a time course for BCI_{cep}.



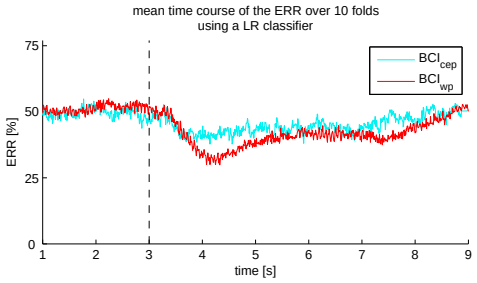
(a) S1 (b) S2

(c) S3 (d) S4

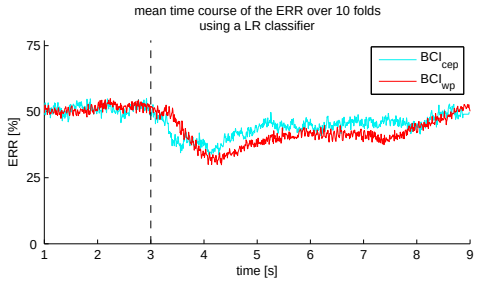
Figure B.1: Time courses of the ERR for subjects S1 to S4 calculated from the mean TSD of the ten test folds. A logistic regression classifier was used by both BCI_{wp} and BCI_{cep} in this figure. The dotted line at $t = 3s$ indicates the cue onset.



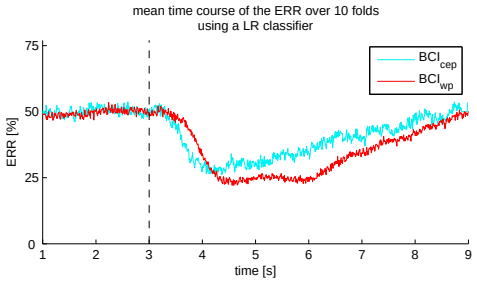
(a) S5



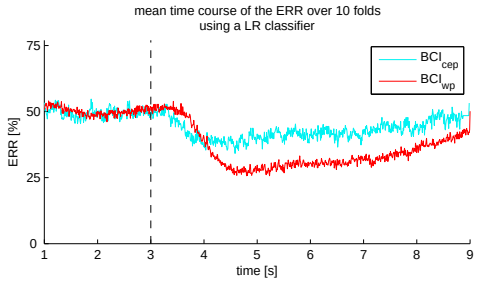
(b) S6



(c) S7

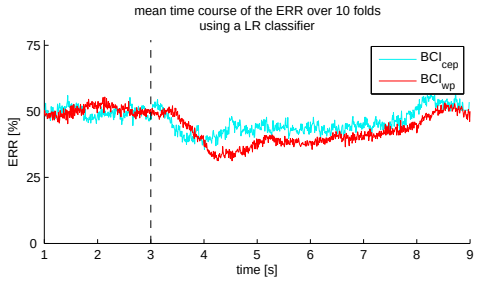


(d) S8

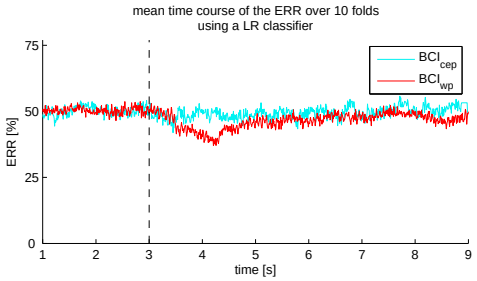


(e) S9

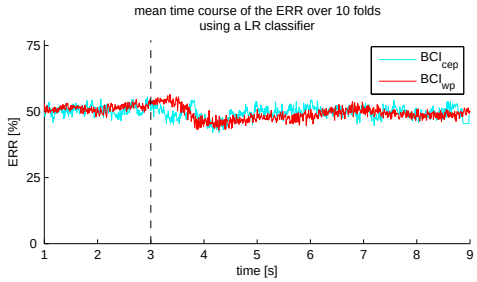
Figure B.2: Time courses of the ERR for subjects S5 to S9 calculated from the mean TSD of the ten test folds. A logistic regression classifier was used by both BCI_{wp} and BCI_{cep} in this figure. The dotted line at $t = 3s$ indicates the cue onset.



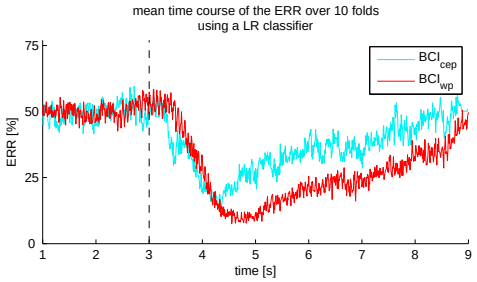
(a) S1



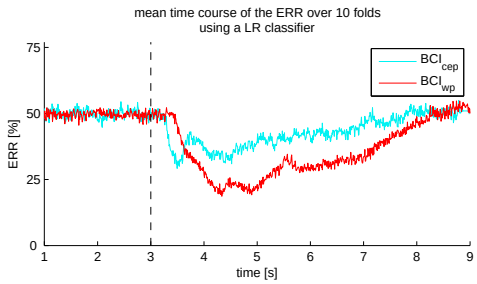
(b) S2



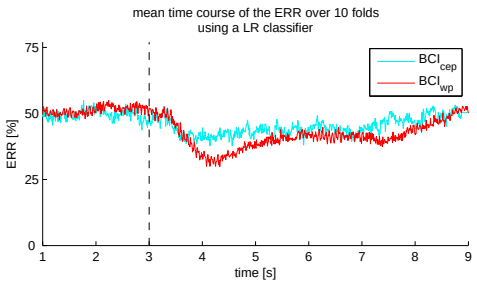
(c) S3



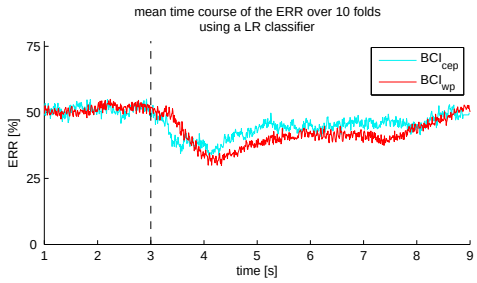
(d) S4



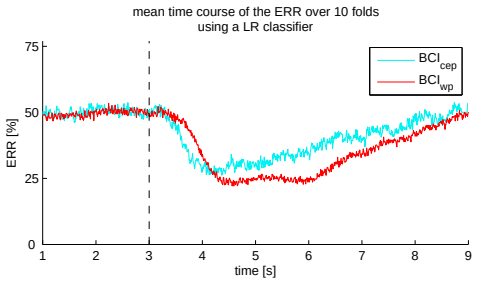
(e) S5



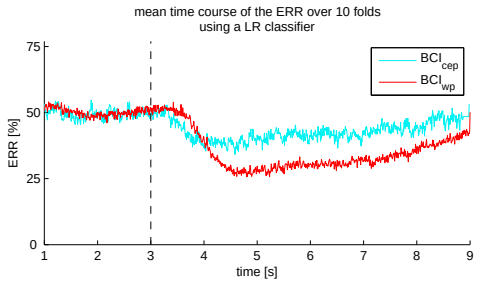
(f) S6



(g) S7



(h) S8



(i) S9

Figure B.3: Time courses of the ERR for each subject calculated from the mean TSD of the ten test folds. A support vector machine was used by both BCI_{wp} and BCI_{cep} to classify the features. The dotted line at $t = 3s$ indicates the cue onset.

List of References

[76] B. Blankertz, K.-R. Müller, G. Curio, T. M. Vaughan, G. Schalk, J. R. Wolpaw, J. R, A. Schlgl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schrder, and N. Birbaumer, "The BCI Competition 2003: Progress and Perspectives in Detection and Discrimination of EEG Single Trials," 2004.

[1] A. Li, F. Yetkin, R. Cox, and V. Haughton, "Ipsilateral hemisphere activation during motor and sensory tasks," *American Journal of Neuroradiology*, vol. 17, pp. 651–655, 1996.

[2] C. Neuper and G. Pfurtscheller, "Event-related desynchronization (ERD) of the Rolandic EEG with imagination of hand movement," *Biomedizinische Technik*, vol. 42 (1), pp. 213–215, 1997.

[3] M. Lotze, P. Montoya, M. Erb, E. Hülsmann, H. Flor, U. Klose, N. Birbaumer, and W. Grodd, "Activation of Cortical and Cerebellar Motor Areas during Executed and Imagined Hand Movements: An fMRI Study," *J. Cognitive Neuroscience*, vol. 11, no. 5, pp. 491–501, 1999.

[4] G. Pfurtscheller and C. Neuper, "Motor imagery and direct brain-computer communication," *Proceedings of the IEEE*, vol. 89, no. 7, pp. 1123–1134, Jul 2001.

[5] L. Qin, L. Ding, and B. He, "Motor imagery classification by means of source analysis for brain-computer interface applications," *Journal of Neural Engineering*, vol. 1, no. 3, pp. 135–141, 2004.

[6] G. Pfurtscheller, C. Neuper, C. Guger, W. Harkam, H. Ramoser, A. Schlogl, B. Obermaier, and M. Pregenzer, "Current trends in Graz brain-computer interface (BCI) research," *Rehabilitation Engineering, IEEE Transactions on [see also IEEE Trans. on Neural Systems and Rehabilitation]*, vol. 8, no. 2, pp. 216–219, 2000.

[7] B. Blankertz, G. Dornhege, M. Krauledat, K. R. Müller, and G. Curio, "The non-invasive Berlin Brain-Computer Interface: fast acquisition of effective performance in untrained subjects." *NeuroImage*, vol. 37, no. 2, pp. 539–550, August 2007.

[8] E. Arbabi, M. Shamsollahi, and R. Sameni, "Comparison between Effective Features Used for the Bayesian and the SVM Classifiers in BCI," in *Engineering in*

Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the, Jan. 2005, pp. 5365–5368.

[9] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for EEG-based brain-computer interfaces." *Journal of neural engineering*, vol. 4, no. 2, June 2007.

[10] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain-computer interfaces for communication and control," *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767 – 791, 2002.

[11] D. Coyle, G. Prasad, and T. M. McGinnity, "A time-frequency approach to feature extraction for a brain-computer interface with a comparative analysis of performance measures," *EURASIP J. Appl. Signal Process.*, vol. 2005, pp. 3141–3151, 2005.

[12] J. J. Vidal, "Toward Direct Brain-Computer Communication," *Annual Review of Biophysics and Bioengineering*, vol. 2, no. 1, pp. 157–180, 1973.

[13] J. Vidal, "Real-time detection of brain events in EEG," *Proceedings of the IEEE*, vol. 65, no. 5, pp. 633–641, 1977.

[14] 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 Trans Rehabil Eng*, vol. 8, no. 2, pp. 164–173, June 2000.

[15] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, J. S. Biggs, M. A. Srinivasan, and M. A. Nicolelis, "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates," *Nature*, vol. 408, no. 6810, pp. 361–365, November 2000.

[16] M. A. Lebedev, J. M. Carmena, J. E. O'Doherty, M. Zacksenhouse, C. S. Henriquez, J. C. Principe, and M. A. L. Nicolelis, "Cortical Ensemble Adaptation to Represent Velocity of an Artificial Actuator Controlled by a Brain-Machine Interface," *J. Neurosci.*, vol. 25, no. 19, pp. 4681–4693, 2005.

[17] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, "Cortical control of a prosthetic arm for self-feeding," *Nature*, vol. 453, no. 7198, pp. 1098–1101, June 2008.

[18] G. Bauer, F. Gerstenbrand, and E. Rumpl, "Varieties of the locked-in syndrome," *Journal of Neurology*, vol. 221, pp. 77–91, August 1979.

[19] N. Birbaumer, "Brain-computer-interface research: Coming of age," *Clinical Neurophysiology*, vol. 117, no. 3, pp. 479–483, March 2006.

[20] G. Pfurtscheller, G. Muller-Putz, A. Schlogl, B. Graimann, R. Scherer, R. Leeb, C. Brunner, C. Keinrath, F. Lee, G. Townsend, C. Vidaurre, and C. Neuper, "15 years of BCI research at graz university of technology: current projects,"

Neural Systems and Rehabilitation Engineering, IEEE Transactions on, vol. 14, no. 2, pp. 205–210, June 2006.

[21] J. Kronegg, "Goals, Problems and Solutions in Brain-Computer Interfaces: a Tutorial," PowerPoint Presentation, November 2005. [Online]. Available: www.bciinfo.tugraz.at

[22] R. Scherer, A. Schloegl, F. Lee, H. Bischof, J. Janša, and G. Pfurtscheller, "The self-paced Graz brain-computer interface: methods and applications," *Intell. Neuroscience*, vol. 2007, pp. 9–9, 2007.

[23] H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial EEG during imagined hand movement," *IEEE Trans. Rehab. Eng*, vol. 8, pp. 441–446, 1998.

[24] G. Dornhege, B. Blankertz, G. Curio, and K.-R. Müller, "Boosting bit rates in non-invasive EEG single-trial classifications by feature combination and multi-class paradigms," 2004.

[25] M. Doppelmayr, W. Klimesch, T. Pachinger, and B. Ripper, "Individual differences in brain dynamics: important implications for the calculation of event-related band power." *Biological Cybernetics*, pp. (1):49–57, 1998.

[26] S. Lemm, B. Blankertz, G. Curio, and K.-R. Müller, "Spatio-Spectral Filters for Improving the Classification of Single Trial EEG," *IEEE Trans. Biomed. Eng*, vol. 52, pp. 1541–1548, 2005.

[27] G. Dornhege, B. Blankertz, M. Krauledat, F. Losch, G. Curio, and K. robert Müller, "Combined optimization of spatial and temporal filters for improving brain-computer interfacing," *IEEE Trans. Biomed. Eng*, 2006.

[28] Q. Novi, C. Guan, T. H. Dat, and P. Xue, "Sub-band Common Spatial Pattern (SBCSP) for Brain-Computer Interface," pp. 204–207, May 2007.

[29] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, August 2006.

[30] A. Schlögl, "The Electroencephalogram and the Adaptive Autoregressive Model: Theory and Applications," Ph.D. dissertation, Shaker Verlag, Aachen, Germany, 2000.

[31] A. Schlögl, G. Müller, C. Neuper, G. Krausz, B. Graimann, and P. G., "Adaptive autoregressive parameters used in BCI research." Dec 2001.

[32] G. Dornhege, *Toward Brain-Computer Interfacing (Neural Information Processing)*. The MIT Press, September 2007.

[33] A. Schlögl, D. Flotzinger, and G. Pfurtscheller, "Adaptive autoregressive modeling used for single-trial EEG classification." *Biomedizinische Technik. Biomedical engineering*, vol. 42, no. 6, pp. 162–167, June 1997.

[34] M. Arnold, X. Milner, H. Witte, R. Bauer, and C. Braun, "Adaptive AR modeling of nonstationary time series by means of Kalman filtering," *Biomedical Engineering, IEEE Transactions on*, vol. 45, no. 5, pp. 553–562, May 1998.

[35] S.-M. Zhou, J. Q. Gan, and F. Sepulveda, "Classifying mental tasks based on features of higher-order statistics from EEG signals in brain-computer interface," *Inf. Sci.*, vol. 178, no. 6, pp. 1629–1640, 2008.

[36] V. Jeyabalan, S. Andrews, and C. K. Loo, "Motor Imaginary Signal Classification Using Adaptive Recursive Bandpass Filter and Adaptive Autoregressive Models for Brain Machine Interface Designs," vol. 3, pp. v3–4–32, 2008.

[37] "BCI Competition II homepage and data repository," 2003. [Online]. Available: http://www.bbci.de/competition/ii/

[38] M. Akin, "Comparison of Wavelet Transform and FFT Methods in the Analysis of EEG Signals," *J. Med. Syst.*, vol. 26, no. 3, pp. 241–247, 2002.

[39] J. Allen, "Applications of the short time Fourier transform to speech processing and spectral analysis," vol. 7, pp. 1012–1015, May 1982.

[40] G. Florian and G. Pfurtscheller, "Dynamic spectral analysis of event-related EEG data," *Electroencephalography and Clinical Neurophysiology*, vol. 95, no. 5, pp. 393 – 396, 1995.

[41] A. Graps, "An introduction to wavelets," *IEEE Computational Science and Engineering*, vol. 2, pp. 50–61, 1995.

[42] M. Canal, "Comparison of Wavelet and Short Time Fourier Transform Methods in the Analysis of EMG Signals," *Journal of Medical Systems*, 2008.

[43] M. Akay, *Time Frequency and Wavelets in Biomedical Signal Processing*, M. Akay, Ed. John Wiley & Sons, 1998.

[44] L. Qin and B. He, "A wavelet-based time–frequency analysis approach for classification of motor imagery for brain–computer interface applications," *Journal of Neural Engineering*, vol. 2, no. 4, pp. 65–72, 2005.

[45] B. Graimann, J. Huggins, S. Levine, and G. Pfurtscheller, "Toward a direct brain interface based on human subdural recordings and wavelet-packet analysis," *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 954–962, June 2004.

[46] J. Z. Xue, H. Zhang, C. X. Zheng, and Y. X. G., "Wavelet packet transform for feature extraction of EEG during mental tasks," *Machine Learning and Cybernetics, 2003 International Conference on*, vol. 1, pp. 360–363, 2003.

[47] B. hua Yang, G. zheng Yan, R. guo Yan, and T. Wu, "Feature extraction for EEG-based brain-computer interfaces by wavelet packet best basis decomposition," *Journal of Neural Engineering*, vol. 3, no. 4, pp. 251–256, 2006.

[48] D. Vautrin, X. Artusi, M. Lucas, and D. Farina, "A Novel Criterion of Wavelet Packet Best Basis Selection for Signal Classification With Application to Brain-Computer Interfaces." *Biomedical Engineering, IEEE Transactions on*, vol. [Epub ahead of print], 2009.

[49] B. hua Yang, G. zheng Yan, R. guo Yan, and T. Wu, "Adaptive subject-based feature extraction in brain-computer interfaces using wavelet packet best basis decomposition," *Medical Engineering & Physics*, vol. 29, no. 1, pp. 48 – 53, 2007.

[50] B. P. Bogert, M. J. R. Healy, and J. W. Tukey, "The quefrency analysis of time series for echoes: Cepstrum, pseudo-autocovariance cross-cepstrum and Saphe-cracking," in *Proceedings of Symposium on Time Series Analysis*, M. Rosenblatt, Ed. Wiley, New York, 1963, pp. 209–243.

[51] B. Gold and N. Morgan, "Speech and Audio Signal Processing: Processing and Perception of Speech and Music," 1999.

[52] R. Leeb, R. Scherer, F. Lee, H. Bischof, and G. Pfurtscheller, "Navigation in virtual environments through motor imagery," in *Proceedings of the CVWW'04*, February, 2004, pp. 99–108.

[53] R. Leeb, F. Lee, C. Keinrath, R. Scherer, H. Bischof, and G. Pfurtscheller, "Brain-Computer Communication: Motivation, Aim, and Impact of Exploring a Virtual Apartment," *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 15, no. 4, pp. 473–482, Dec. 2007.

[54] "BCI Competition III homepage and data repository," 2005. [Online]. Available: http://www.bbci.de/competition/iii/

[55] "BCI Competition IV homepage and data repository," 2008. [Online]. Available: http://www.bbci.de/competition/iv/

[56] J. Farquhar, J. Hill, T. N. Lal, and B. Schölkopf, "Regularised CSP for Sensor Selection in BCI," in *3rd International Brain-Computer Interface Workshop and Training Course*, Graz, Austria, Sept 2006.

[57] C. Heil and D. F. Walnut, "Continuous and discrete wavelet transforms," *SIAM Rev.*, vol. 31, no. 4, pp. 628–666, 1989.

[58] D. Marpe, H. Cycon, and W. Li, "Complexity-constrained best-basis wavelet packet algorithm for image compression," *IEE Proceedings - Vision, Image, and Signal Processing*, vol. 145, no. 6, pp. 391–398, 1998.

[59] S. Aroutchelvame and K. Raahemifar, "An Architecture Design of Threshold-Based Best-Basis Algorithm," *Multimedia and Expo, IEEE International Conference on*, vol. 0, pp. 1273–1276, 2006.

[60] H. Wassner and G. Chollet, "New Cepstral Representation Using Wavelet Analysis And Spectral Transformation For Robust Speech Recognition," in *Proc. of ICSLP-96*, 1996.

[61] R. Muralishankar and A. Ramakrishnan, "Pseudo Complex Cepstrum Using Discrete Cosine Transform," *International Journal of Speech Technology*, vol. 8, pp. 181–191(11), June 2005.

[62] J. Volkmann, S. S. Stevens, and E. B. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 208–208, 1937.

[63] W. Abdul and J. Wong, "Cortical activities pattern recognition for the limbs motor action," in *Intelligent Environments, 2008 IET 4th International Conference on*, July 2008, pp. 1–7.

[64] D. Childers, D. Skinner, and R. Kemerait, "The cepstrum: A guide to processing," *Proceedings of the IEEE*, vol. 65, no. 10, pp. 1428–1443, Oct. 1977.

[65] R. Kemerait and D. Childers, "Signal detection and extraction by cepstrum techniques," *Information Theory, IEEE Transactions on*, vol. 18, no. 6, pp. 745–759, Nov 1972.

[66] G. H. Klem, H. O. Lüders, H. H. Jasper, and C. Elger, "The ten-twenty electrode system of the International Federation. The International Federation of Clinical Neurophysiology." *Electroencephalography and clinical neurophysiology. Supplement*, vol. 52, pp. 3–6, 1999.

[67] K. Fukunaga, *Introduction to Statistical Pattern Recognition, Second Edition (Computer Science and Scientific Computing Series)*. Academic Press, September 1990.

[68] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[69] T. Fletcher, "Support Vector Machines Explained," 2009. [Online]. Available: http://www.tristanfletcher.co.uk/SVM%20Explained.pdf

[70] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, August 1999.

[71] C. Cortes and V. Vapnik, *Support Vector Networks*, 1995, vol. 20.

[72] B. Carpenter, "Lazy Sparse Stochastic Gradient Descent for Regularized Mutlinomial Logistic Regression," 2008, Alias-i, Inc.

[73] A. Schlögl, J. Kronegg, J. E. Huggins, and S. G. Mason, *Evaluation Criteria for BCI Research*. MIT Press, 2007, ch. 19, pp. 327–342.

[74] A. Schlögl, C. Neuper, and G. Pfurtscheller, "Estimating the mutal information of an EEG-based brain-computer interface," *Biomed. Technik*, vol. 47, pp. 3–8, 2002.

[75] A. Schlögl and C. Brunner, "BioSig: a free and open source software library for BCI research," *Computer*, vol. 41, no. 10, pp. 44–50, Oct. 2008. [Online]. Available: http://biosig.sourceforge.net/

[77] M. Pregenzer and G. Pfurtscheller, "Distinction Sensitive Learning Vector Quantization (DSLVQ) - application as a classifier based feature selection method for a brain computer interface," *IEE Conference Publications*, vol. 1995, no. CP409, pp. 433–436, 1995.

[78] A. Schlögl, K. Lugger, and G. Pfurtscheller, "Using adaptive autoregressive parameters for a brain-computer-interface experiment," *Engineering in Medicine and Biology Society, 1997. Proceedings of the 19th Annual International Conference of the IEEE*, vol. 4, pp. 1533–1535 vol.4, 1997.

[79] C. Neuper, A. Schlögl, and G. Pfurtscheller, "Enhancement of left-right sensorimotor EEG differences during feedback-regulated motor imagery." *J Clin Neurophysiol*, vol. 16, no. 4, pp. 373–382, July 1999.

[80] G. Pfurtscheller, C. Neuper, A. Schlogl, and K. Lugger, "Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters," *Rehabilitation Engineering, IEEE Transactions on*, vol. 6, no. 3, pp. 316–325, Sep 1998.

[81] C. Vidaurre, A. Schlogl, R. Cabeza, R. Scherer, and G. Pfurtscheller, "A fully on-line adaptive BCI," *Biomedical Engineering, IEEE Transactions on*, vol. 53, no. 6, pp. 1214–1219, 2006.