A Generative Approach to EEG Source Separation, Classification and Artifact Correction

Helle Henriksen

Kongens Lyngby 2012
IMM-MSc-2012-36

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk IMM-MSc-2012-36

Abstract

This thesis deals with the detection of right and left hand-pull stimuli in EEG data for five healthy subjects. This paradigm give rise to activation of motor cortex contra-lateral to stimuli side.
ICA components obtained from a Kalman filter based algorithm have been applied as features in the classification task and compared with time series features and Infomax ICA features. The Kalman ICA components have proven to be well-suited for separating the two classes in this thesis, and the Kalman features accomplished the lowest error rates when classifying left and right stimuli. Different classifiers have been tested on the three feature types, and the advanced SVM classifier performed best in all cases. The percentage of significant different features between the two classes showed to be strongly correlated to the classification performance. For the purpose of stimuli detection a visual inspection of the ICA components has been made. The visible distinction is not as pronounced as the difference in classification performance for the two ICA features.

Resumé

Målet for denne afhandling er at anvende EEG data for fem raske personer til at detektere to forskellige slags stimuli. Disse stimuli er trækken i hhv. højre og venstre hånd. Dette paradigme forårsager aktivering af motor cortex i den modsatte side end den hånd der blev trukket i.
ICA komponenter fra en Kalman filter baseret algoritme er blevet brugt som features og sammenlignet med tidsserie features og Infomax ICA features. Kalman ICA komponenterne har vist sig at være særdeles velegnede til at separere data i de to forskellige slags stimuli, og Kalman komponenterne er også de features der opnår lavest fejlrate i klassifikationsopgaven. Der er blevet afprøvet forskellige klassifikatorer på alle tre slags features og den avancerede SVM klassifikator har i alle tilfælde klaret sig bedst. Procentdelen af signifikant forskellige features mellem de to klasser har vist sig at være yderst korreleret med klassifikationspræstationen. Der er yderligere blevet lavet en visuel inspektion af ICA komponenterne med det formål at se om stimuli detekteringen er synlig. Det viser sig, at den visuelle forskel ikke er lige så udtalt som forskellen mellem klassifikationspræstationerne for de to slags ICA features.

Preface

This thesis was prepared at DTU Informatics at the Technical University of Denmark in fulfilment of a Master of Science degree in Biomedical Engineering. The work on this thesis was carried out in the period from September 6th 2011 to April 16th 2012 with a workload of 40 ECTS credits.

Lyngby, 16-April 2012

Helle Henriksen

Acknowledgements

I would like to especially thank my supervisors. Thanks to my main supervisor, associate professor, Ph.D. Ole Winther, for competent guidance, engagement, feedback and a positive approach. Thanks to professor, Ph.D. Lars Kai Hansen for supporting this thesis and making it happen. Furthermore I would like to thank Morten Mørup for help and information regarding the dataset and Sidse M. Arnfred for collecting the data. I also thank Simon Christian Hede and Ricardo Henao for consulting and guidance regarding methods and implementation. Additionally I would like to thank Melissa Larsen and Louise Mejdal Jeppesen for helping out, supporting me, and keeping me company. Finally my thanks goes to Julie A E Christensen for theoretical help and valuable feedback.

Contents

Abbreviations

BCI	Brain Computer Interface, 2
EEG	Electroencephalography, 1
ERP	Event Related Potential, 1
FIR	Finite Impulse Response, 30
fMRI	Functional Magnetic Resonance Imaging, 1
GP	Gaussian Process, 26
ICA	Independent Component Analysis, 2
IMM	DTU Informatics, 32
KNN	K Nearest Neighbour, 15
LM	Lagrange Multiplier, 20
MI	Mutual Information, 6
NBC	Naive Bayes Classifier, 16
PET	Positron Emission Tomography, 1
QP	Quadratic Programming, 20
SVM	Support Vector Machine, 17

Chapter 1

Introduction

Electroencephalography (EEG) is a recording method that measures the electrical activity of the brain, and electrodes placed on the scalp are widely used to record this. The electrical activity is caused by simultaneous electrical signals from a huge number of nerve cells, and the EEG recordings are used for both clinical and research purposes [39]. EEG is used as a diagnostic tool in certain neurophysiological disorders. An example is epilepsy, where the seizures result in very different electrical behaviour compared to normal activity [42]. In the research field EEG is e.g. applied to study Event Related Potentials (ERP's) , which is a response to a given internal or external stimulus, and Brain Computer Interface (BCI) that enables communication between human and computer only by means of brain activity [21]. Exploration of the brain can be done by other modalities as well, such as fMRI and PET, which indirectly measures the electrical activity and have a much higher spatial resolution than EEG. Some significant advantages of EEG are the high temporal resolution and the low cost of the examination compared to fMRI and PET. Combination of EEG with fMRI give possibility of both high spatial and temporal resolution [35]. Raw EEG signals contain both biological and environmental artifacts and furthermore drift, which makes the raw signals very hard to interpret, as well as time consuming [28]. In addition, the signals are summations of all brain activity and the separation between activity caused by background- and stimulus EEG can be quite difficult [2]. For these reasons various automated methods have been proposed to solve these challenges.

1.1 Modelling of EEG Signals

Modelling of EEG signals is one of the proposed solutions for the above mentioned challenges regarding raw EEG signals. Modelling provides a tool for tracking the underlying brain activity and dividing the signal into components caused by different brain processes, such as artifacts and stimuli [32]. This division of the signals into components is obviously an advantage as it enables selection and discarding of respectively valuable and useless signals. This is applicable in many different types of EEG data, such as sleep or BCI recordings, because it has the ability to act as a filter or classification tool. In sleep data it is especially important to correct for artifacts in the form of e.g. blinking and movements, and in BCI, feature extraction is the key to classification, which can be done by modelling. Independent Component Analysis (ICA) is one way to split a given signal into sources and thereby unmix the signal [32], [28], and in combination with a generative model it makes a well suited tool for EEG analysis [11].
Generative modelling has been explored in various ways and on many different types of data. In [36] a linear state space model is applied to divide a speech mixture into individual speech sources and in [24] a temporal Gaussian regression problem is reformulated as Kalman filtering of linear state space models. A generative ICA approach has been applied with success in [11] on mental task EEG data with the purpose of using the components in BCI classification.

1.2 Detection of Different Types of Stimuli in EEG Signals

Distinction between stimuli in EEG signals is conditional on a traceable difference between EEG caused by the different types of stimuli, and discrimination of background EEG and EEG related to stimuli. In the experiment used in this thesis, the subjects are pulled in their left and right hand respectively. It is expected that the left side of motor cortex is activated when the subjects are being pulled in the right hand and vice versa, see section 3.1 for further details. This should result in activation of different electrode areas for the two stimuli, which enables separation between these. In [3] detection of gamma waves contra-lateral to the stimulus side was observed 0.6 seconds after stimuli. The data in [3] is the same type of pull stimuli used in this thesis. In addition, the same paradigm was used in [4] to study if the EEG data for this particular stimulus is different between schizophrenic and healthy subjects.

1.3 Thesis Objective

The object of this thesis was originally formulated as a generative approach to modelling the multivariate EEG signal into underlying Brain processes using the Gaussian Process Kalman filter, and in addition apply the filter for classification by the use of the augmented binary probit node. This has been reformulated a little during the process, but the result is nevertheless almost the same.
The Kalman filter has been applied as an ICA algorithm to track the underlying components in the EEG data. These Kalman ICA components have been used as features in a classification task and compared to raw time series features and Infomax ICA features. The two simple classifiers K Nearest Neighbour (KNN) and Naive Bayes (NBC) plus the more advanced Support Vector Machine (SVM) have been tested on these three feature types to verify whether the Kalman filter provides stimuli related to components applicable for classification or not. A visual comparison of the ICA components has been carried out for inspection of the nature of the tracked components, meaning whether the components is related to noise, artifacts or stimuli. In addition filtering has been applied to correct for the drift in data. The EEG data used for this purpose originates from 5 subjects pulled in their left and right hand, respectively.
In [37] Kalman filter parameters are used as features, and in [11] generative ICA components are applied for classification, but using Kalman ICA components as features to EEG classification have to the best of my knowledge not been done before.

Chapter 2

Independent Component Analysis Applied to EEG

ICA can be applied to EEG signals to separate data into underlying components caused by e.g. artifacts and external stimuli [32]. This chapter is a general introduction to the ICA method.
The Infomax ICA theory is provided in the first section, and ICA applied to EEG through a concrete example is provided in the second section.

2.1 Infomax ICA

The general idea of ICA can be described as the process of separating an observed dataset, $\mathbf{x}$, into a set of independent components/sources, $\mathbf{s}$, by finding the unmixing matrix $\mathbf{W}$. In Infomax ICA the generative model is described as

$$\mathbf{x} = \mathbf{A}\mathbf{s} \ , \tag{2.1}$$

where it is assumed that the number of observations is equal to the number of components, e.i. the mixing matrix $\mathbf{A}$ is square and related to $\mathbf{W}$ by having it as its inverse, $\mathbf{A} = \mathbf{W}^{-1}$. The Infomax ICA algorithm, which was invented by Bell and Sejnowski in 1995 [6], is one method to perform ICA. It approximates

W by minimising the Mutual Information (MI) between the components[32]. Making the MI go to zero returns maximally independent components [30], and the MI objective can also be seen as a maximum likelihood inference problem [31]. The likelihood function for **A** is given by

$$\mathbf{P}(\mathbf{X}|\mathbf{A}) = \prod_{n=1}^{N} \mathbf{p}(\mathbf{x}^{(n)}|\mathbf{A}) \ , \tag{2.2}$$

for $n = 1, ..., N$ where N is the number of samples. The right-hand side in Eq. 2.2 is the product of the marginalised probabilities and a single factor in the likelihood can be written as

$$\mathbf{p}(\mathbf{x}^{(n)}|\mathbf{A}) = \int \mathbf{p}(\mathbf{x}^{(n)}|\mathbf{A}, \mathbf{s}^{(n)})\mathbf{p}(\mathbf{s}^{(n)})d\mathbf{s}^{(n)} \ . \tag{2.3}$$

Assuming noise-free data [6], Eq 2.3 can be rewritten, by marginalising over delta functions, yielding

$$\mathbf{p}(\mathbf{x}^{(n)}|\mathbf{A}) = \int \delta(\mathbf{x}^{(n)} - \mathbf{A}\mathbf{s}^{(n)})\mathbf{p}(\mathbf{s}^{(n)})d\mathbf{s}^{(n)} \ . \tag{2.4}$$

Now introducing a shift in variables $\mathbf{z} = \mathbf{A}\mathbf{s}^{(n)}$ and by the use of the Jacobian given as

$$\begin{aligned} d\mathbf{s}^{(n)} &= |\det(\frac{d\mathbf{s}^{(n)}}{d\mathbf{z}})|d\mathbf{z} \\ d\mathbf{s}^{(n)} &= |\det(\mathbf{A}^{-1})|d\mathbf{z} = \frac{1}{\det|\mathbf{A}|}d\mathbf{z} \ , \end{aligned} \tag{2.5}$$

the following is obtained by replacing Eq. 2.5 into Eq. 2.4, giving

$$\mathbf{p}(\mathbf{x}^{(n)}|\mathbf{A}) = \frac{1}{\det|\mathbf{A}|}\int \delta(\mathbf{x}^{(n)} - \mathbf{z})\mathbf{p}(\mathbf{A}^{-1}\mathbf{z})d\mathbf{z} \ , \tag{2.6}$$

and together with the property: $\int f(y)\delta(y - y_0)dy = f(y_0)$, the reduced expression is given by

$$\mathbf{p}(\mathbf{x}^{(n)}|\mathbf{A}) = \frac{1}{\det|\mathbf{A}|}\mathbf{p}(\mathbf{A}^{-1}\mathbf{x}^{(n)}) \ . \tag{2.7}$$

The log likelihood can be derived directly from Eq. 2.7 and inserting **W** result in the following expression for a single factor

$$\ln \mathbf{p}(\mathbf{x}^{(n)}|\mathbf{W}) = \ln|\det(\mathbf{W})| + \ln \mathbf{p}(\mathbf{W}\mathbf{x}^{(n)}) \ . \tag{2.8}$$

From now on $\mathbf{W}$ is assumed to be positive definite, and by finding the gradient of the log likelihood, the maximum likelihood algorithm will be obtained by

$$\frac{\partial \ln \mathbf{p}(\mathbf{x}^{(n)}|\mathbf{W})}{\partial \mathbf{W}} = [\mathbf{W}^T]^{-1} + \mathbf{y}\mathbf{x}^T \ , \tag{2.9}$$

where $\mathbf{y} = f(\mathbf{W}\mathbf{x}) = \frac{d \ln \mathbf{p}(\mathbf{s})}{d\mathbf{s}}\Big|_{s=\mathbf{W}\mathbf{x}}$ which is a non-linear mapping. Maximising the log likelihood and thereby minimising the MI can therefore be expressed by adjusting the weights according to the gradient in the following [31]

$$\Delta \mathbf{W} = [\mathbf{W}^T]^{-1} + \mathbf{y}\mathbf{x}^T \ . \tag{2.10}$$

If the prior distribution is defined as $\mathbf{p}(\mathbf{s}) = \frac{1}{\pi \cosh(\mathbf{s})}\Big|_{s=\mathbf{W}\mathbf{x}}$ then the function f is given by $f(\mathbf{s}) = -\tanh(\mathbf{s})|_{s=\mathbf{W}\mathbf{x}}$. This definition for f is often applied, because it assumes a more heavier tailed prior distribution than a Gaussian prior [31].
Adjusting the weights according to Eq. 2.10 is one way to create the learning algorithm, but the covariant algorithm is a simpler and faster alternative [31]. In this approach the weights are adjusted to the following gradient

$$\Delta \mathbf{W} = \mathbf{W} + \mathbf{y}\mathbf{x}\prime^T \ , \tag{2.11}$$

where $\mathbf{x}\prime = \mathbf{W}^T\mathbf{W}\mathbf{x}$. The maximum likelihood problem is in this approach solved by taking the second derivative (instead of the first) of the log likelihood with respect to $\mathbf{W}$, and the expression is advantageous because no inversion of $\mathbf{W}$ appears [31]. For further description of this approach see [31].

2.2 Practical Aspects of Infomax ICA

An EEG dataset containing signals from 72 electrodes from one subject, stimulated by 120 left and right hand pulls respectively, is used in this section. The sampling rate of the data is 512 Hz, and initially the dataset was high pass filtered at 3 Hz to correct for the offset in the data. The filtered signal is visualised in Fig. 2.1. The vertical lines indicates different events, where 64602 and 64603 is left and right hand pulls respectively. To investigate the ICA algorithm's capability to track stimuli, ICA is performed on the entire EEG signal, but since channel 65-72 are reference and artifact channels, these are not included in the analysis. The Infomax ICA algorithm, implemented in EEGlab, is applied to perform an investigation of the signal, and the algorithm provides a temporal and a spatial component. The spatial map can be derived from each column in the mixing matrix $\mathbf{A}$ and the temporal component from each row in the source

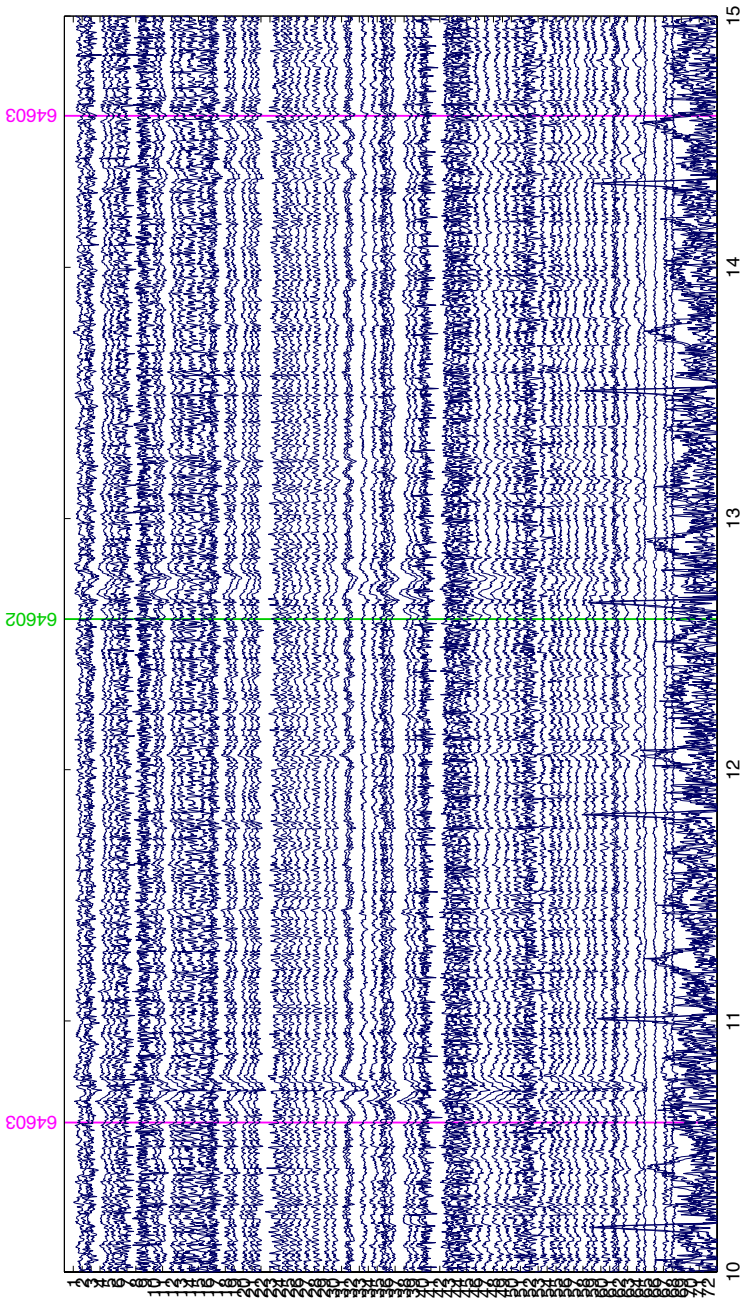


Figure 2.1: EEG signal filtered with a 3 Hz highpass filter.

matrix **S**. The temporal independent components are visualised in Fig. 2.2. It is clear from this figure that the ICA components are sorted according to energy and thereby importance, but it is difficult to conclude if the ICA algorithm has tracked the stimuli. The 64 temporal components are segmented into 240 epochs, holding 120 for left hand and 120 for right hand, and averaged with respect to epochs. Dividing into epochs and averaging is done to study if any differences, related to the two different stimuli, are detectable. The epochs consist of information from start of the stimuli to 1.5 seconds after. Different illustrations of the segmented averaged components are shown in Fig. 2.3 and 2.4. The corresponding spatial components are provided in Fig. 2.5.

In Fig. 2.3 the first 16 and most important ICA components are shown separately. It is clear from the components that Fig. a and b comes from different stimuli, because the activation pattern between the two are visible differentiable. Especially component 10 and 16 are easy to distinguish from each-other, and when inspecting the spatial components in Fig. 2.5 it appears that the left and right motor cortex area is activated, respectively. In Sec. 3.1 the physiological background for this is explained. In Fig. 2.4 the two averaged components are plotted for the two stimuli in the same plot with errorbars to study if the difference between the stimuli is significant. The errorbars are calculated as the standard-deviation across epochs, and the bars are quite big, which makes the distinction between the two stimuli difficult, and classification based on temporal ICA components doubtful.

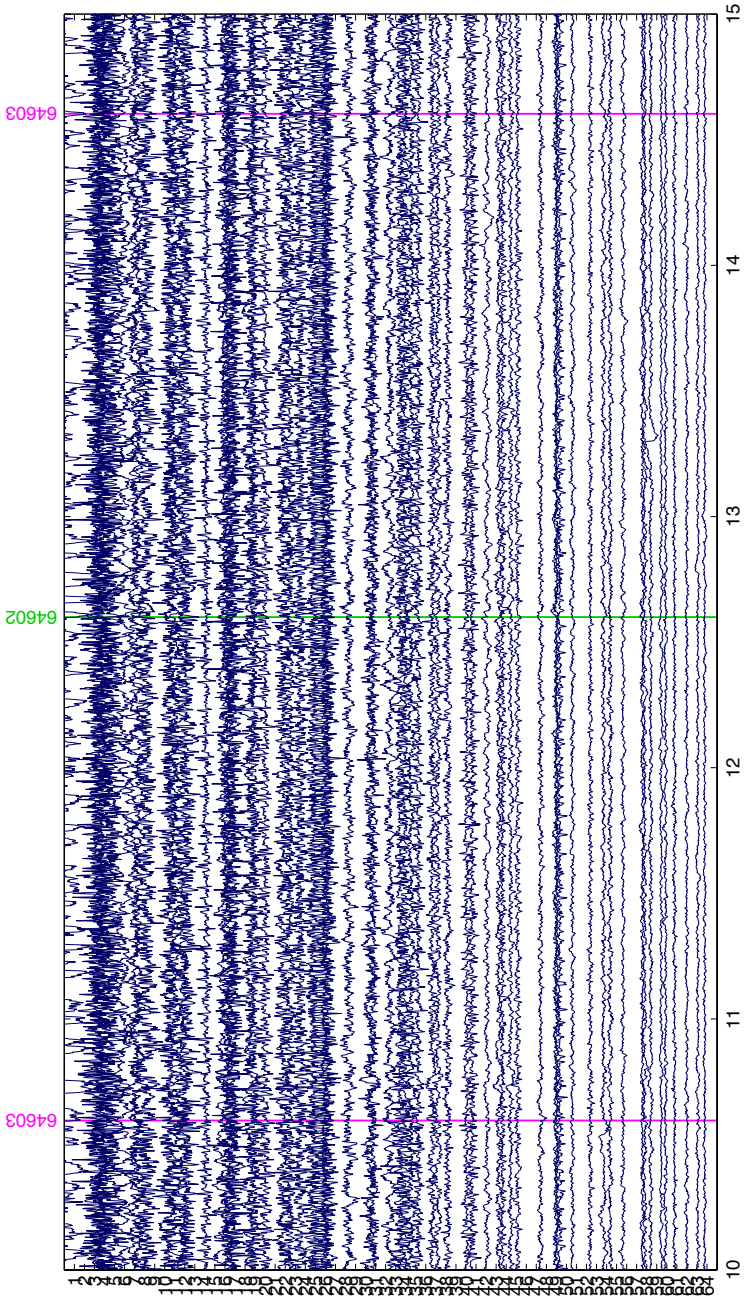
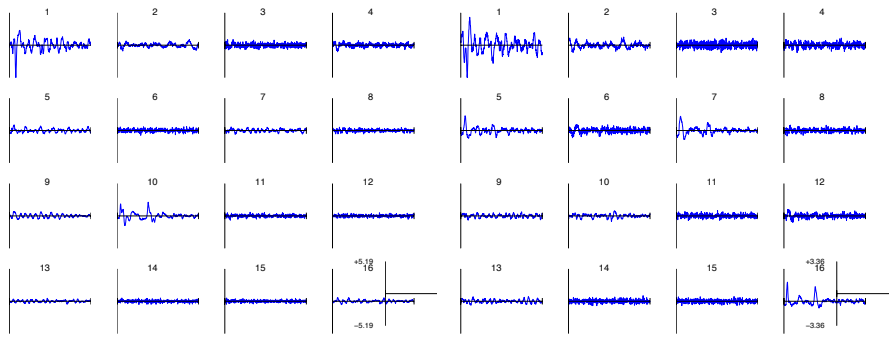
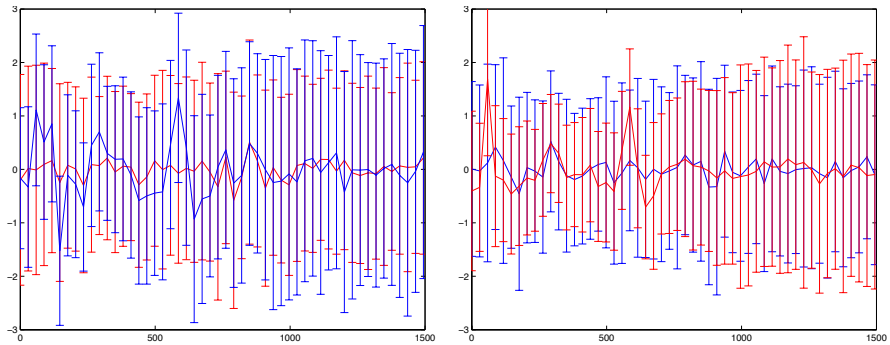


Figure 2.2: Temporal ICA components.



(a) (b)

Figure 2.3: Segmented averaged ICA components, left and right stimuli, respectively.



(a) (b)

Figure 2.4: Segmented averaged ICA component 10 and 16, respectively, with errorbars. The blue curve is left stimuli and the red curve is right stimuli.

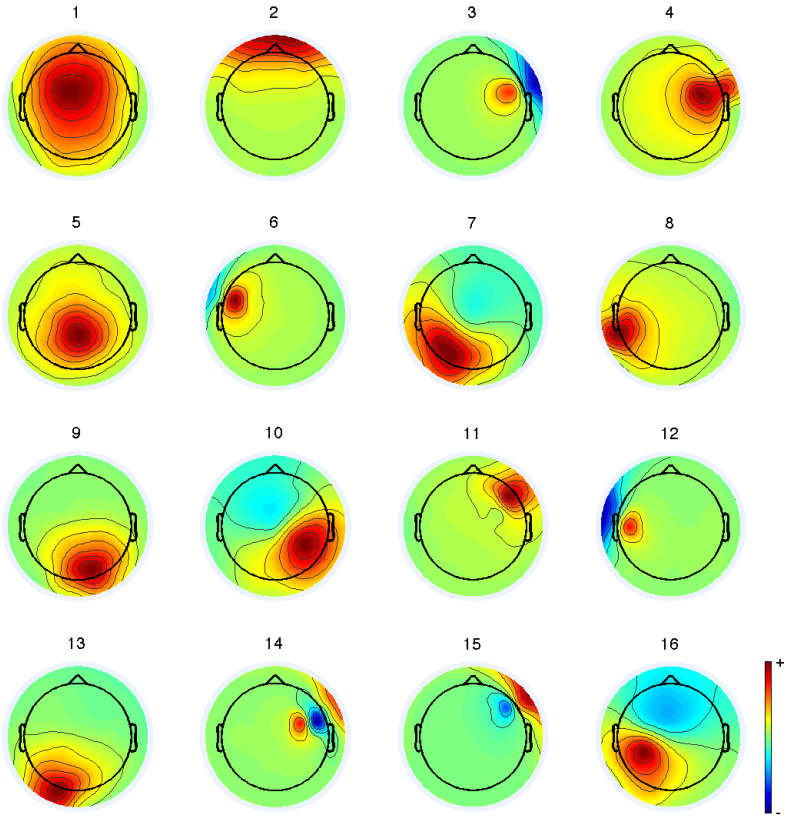


Figure 2.5: The 16 first spatial components.

Chapter 3

Theory

This chapter provides knowledge within the clinical and technical field relevant for this thesis. The first section concerns a basic introduction to EEG signals and how these are affected by external stimuli. The next three sections deal with the theoretical background for the classification methods (KNN, NBC and SVM) applied in this thesis. Finally, the last section provides the needed knowledge for the Kalman filter theory.

3.1 EEG Signals and Activation of Motor Cortex

EEG is a representation of the electrical brain activity [39], and the activity is recorded by electrodes either placed on the surface of the scalp or by sub dermal needles. The electrical activity is a measure of the voltage between an electrode placed in an active area and a reference electrode [25], and the activity is caused by electrical signals called action potentials that act as cell to cell communication and activation of intracellular processes [38]. Electrodes are not sensitive enough to measure individual action potentials, and the recorded electrical currents are generated by a large number of simultaneous action potentials originating from different neurons. The method was applied to humans for the first time in 1924 by Hans Berger, and in 1929 he reported on the subject, where

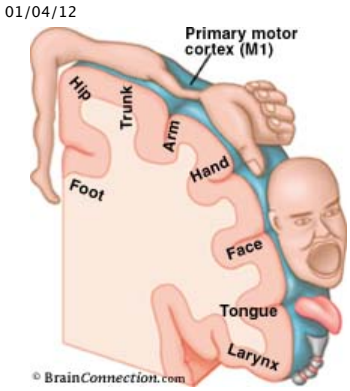


Figure 3.1: Homunculus model from [44] of the right hemisphere of motor cortex.

the terms alpha and beta waves were introduced as well [7]. EEG signals from a person that is awake and relaxed, has in general no specific pattern, because the electrical activity is not synchronous. At other mental stages such as sleep, certain low frequency patterns are dominating. Alpha waves (8-13 Hz) occurs when a person is awake with closed eyes and in quiet surroundings, and beta waves (above 13 Hz) are dominant at EEG recordings at intense mental activity[39]. Gamma wave activity (25-100 Hz) is likely to occur in neural communication, reflecting external input information to the brain [27], and the most pronounced frequency in this wave pattern is 40 Hz [13].
The brain can be divided into four main parts; the brainstem, the cerebellum, the diencephalon and the cerebrum. The outer surface of cerebrum is called cerebral cortex and is the part of the brain that contribute most to the EEG signals [39]. The motor area of cerebral cortex is called motor cortex and the action potential originating from this area mainly controls voluntary movements and especially movements performed by the hand are well represented [38]. In Fig. 3.1 a homunculus model of the right hemisphere of motor cortex is shown, and from this figure it is also illustrated, how big the part that controls hand movement is. For this reason hand movements should result in detectable variation in the EEG signals compared to background activity [3]. The brain consists of a right and a left hemisphere, and the left one controls the activity of muscles from the right half of the body and vice versa [38]. Since movement of left/right hand has a big region in the right and left hemisphere, respectively, difference in EEG recordings between stimuli of the two hands should be detectable [3].

3.2 K Nearest Neighbour

The K Nearest Neighbour (KNN) algorithm is a supervised classification method that was introduced for the first time in 1951 by Fix and Hodges [16]. KNN is one of the most simple machine learning algorithms, and the method requires a training set with known class labels to develop the classifier, and a test set to test the classification performance. The classification of a test point is determined by the euclidean distance from the test point to K training points. Assuming N and P are the number of training and test points respectively, $\mathbf{x}^{(n)}$ is the training set and $\mathbf{y}^{(l)}$ is the test set, where $n = 1...N$ and $l = 1...P$. The euclidean distance between one test point e.g. $\mathbf{y}^{(1)}$ and the entire training set, $\mathbf{x}^{(n)}$, is calculated by Eq. 3.1. The distances are sorted and the K nearest training points determines the classification of $\mathbf{y}^{(1)}$ [39].

$$\begin{aligned} d^{(1)} &= \sqrt{x^{(1)} - y^{(1)}}^2 \\ d^{(2)} &= \sqrt{x^{(2)} - y^{(1)}}^2 \\ &\vdots \\ d^{(n)} &= \sqrt{x^{(n)} - y^{(1)}}^2 \end{aligned} \tag{3.1}$$

The number of neighbours, K, is crucial for the classification result, and the optimal value of K is dependent on the specific dataset. If K is set too high there is a risk of over smoothing and difficulties in distinguishing between classes. On the other hand if K is to small there is a good change of over-fitting to the pattern of the specific dataset. Accordingly it is of great importance to find a value for K that is neither to high nor to small [8]. The identification of the optimal K can be done by applying the "nested cross-validation" method, which can be explained by the "leave one out" method only used on the training data, meaning all points from the training set in turn are used as a test point, where the distance from this point to its K neighbours are calculated, and the class of the point is predicted and compared to the known true class. A classification error for each number of K is thereby provided, and the optimal value of K is the one that results in the lowest training classification error [22]. The size of K is due to the leave-one-out method limited to the size of the training set minus one, $K_{max} = N_{train} - 1$.
The advantage of the KNN algorithm lies in the simplicity and that no prior knowledge about density function is needed, but the necessity of storing all samples and comparing each of them with unknown samples is quite a disadvantage as it is very computational expensive [18]. The KKN algorithm is illustrated in Fig. 3.2 where the number of classes is two and the total number of training points is ten. The black square indicates a test point that is classified by the K nearest neighbour from the training set. The figure illustrates the importance of

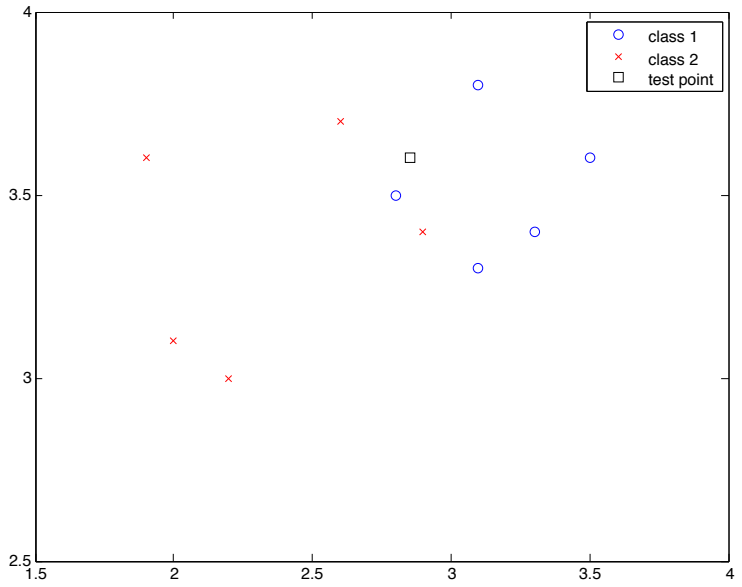


Figure 3.2: Illustration of the K nearest neighbour algorithm.

the value K for the classification of the test point. If $K = 1, 5, 6, 7$ the test point will have the shortest distance to a majority of class 1 points, but if $K = 2, 4$ the test point can not be classified because the closest training points are of equal numbers of class 1 and 2, and if $K = 3$ the point will be classified as class 2. To avoid equal amount of points from each class, K can be forced to be an odd number [18].

3.3 Naive Bayes Classifier

The Naive Bayes Classifier (NBC) is a simple classifier method, which is named "Naive" because of the assumption about independent features. The method is probabilistic and is based on Bayes' Theorem stating that the posterior probability can be calculated from the prior probability and the likelihood

$$p(c|\mathbf{x}) = \frac{p(\mathbf{x}|c)p(c)}{p(\mathbf{x})} \ . \tag{3.2}$$

In Eq. 3.2, $\mathbf{x} = x_1, x_2..., x_k$, where k is the number of features, $\mathbf{x}$ represent a feature vector and c is the value of the class variable C. The posterior distribution

is thereby the probability that the feature vector belongs to class c. Since the features are independent the posterior probability can be modified according to the conditional independence property

$$p(x_1, x_2, ..., n_k|c) = p(x_1|c)p(x_2|c)...p(x_k|c) = \prod_{i=1}^{k} p(x_i|c) \ . \tag{3.3}$$

The posterior probability is then expressed by Eq. 3.4, and the class that result in highest probability for a given feature vector gets this vector assigned.

$$p(c|x_1, x_2..., x_k) = \frac{p(c)\prod_{i=1}^{k} p(x_i|c)}{p(x_1, x_2..., x_k)} \ . \tag{3.4}$$

For a two class situation where the value of c is 1 and -1 for the two classes, respectively, the decision function can be described by the following

$$\begin{aligned} NBC &= \frac{p(C=1|\mathbf{x})}{p(C=-1|\mathbf{x})} \\ &= \frac{p(C=1)}{p(C=-1)} \prod_{i=1}^{k} \frac{p(x_i|C=1)}{p(x_i|C=-1)} \ , \end{aligned} \tag{3.5}$$

where the feature vector is assigned class 1 ($c = 1$) if $NBC > 1$ and class 2 ($c = -1$) if $NBC < 1$.
In real world applications the independence assumption appears rather unrealistic, but despite this fact the NBC shows satisfying results. In [45] it is proposed that the individual dependencies between features cancel out in the big picture, and what matters for the NBC performance instead is the distribution of the dependencies among all features over classes.

3.4 Support Vector Machine

The Support Vector Machine (SVM) algorithm was described for the first time in 1995 by Cortes and Vapnik [12]. SVM is a supervised classification method that aims to create a hyperplane that separates the classes in feature space in the most optimal way by the use of support vectors. SVM classification can be divided into creating an optimal hyperplane between three kinds of data:

1. Linear Separable
2. Not Fully Linear Separable

3. Nonlinear

and the classification of these three types using SVM is described in the following sections. The nonlinear approach is not applied in this thesis, and is accordingly not described in details.

3.4.1 Linear Separable Classification

For simplicity a two class classification problem with only two features is explained. Assuming $\mathbf{x}^{(n)}$ is the training data that belongs to either class $c^{(n)} = -1$ or $c^{(n)} = 1$, where $n = 1, ..., N$ denotes the sample number, the hyperplane for separating the two classes can be expressed by [12]:

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \ . \tag{3.6}$$

Assuming the number of features is $k = 2$ the data can be represented by:

$$\{x^{(n)}, c^{(n)}\} \quad \text{where} \quad n = 1, ..., N, \ c^{(n)} \in \{-1, 1\} \quad x^{(n)} \in \Re^2 \ . \tag{3.7}$$

In Eq. 3.7 the assumption about linear separability has been made, meaning a hyperplane is able to fully separate the classes.
In Eq. 3.6, $\mathbf{w}$ is the normal to the hyperplane and $\frac{b}{\|\mathbf{w}\|}$ is the perpendicular distance from the origin to the separating hyperplane. The two-dimensional two class example is illustrated in Fig. 3.3 and the samples closest to the separating hyperplane is called Support Vectors. The SVM algorithm aims to locate the hyperplane that has the longest distance to the closest observations of both classes[17]. The maximisation of the margin between the two classes are also referred to as the Maximum Margin Classifier.

The support vectors, marked by extra circles in Fig. 3.3 spans two lines (hyperplanes in higher dimensions), H_1 and H_2. These lines can be expressed by:

$$\begin{aligned} \mathbf{x}^{(n)} \cdot \mathbf{w} + b &= +1 \qquad \text{for} \quad H_1 \\ \mathbf{x}^{(n)} \cdot \mathbf{w} + b &= -1 \qquad \text{for} \quad H_2 \end{aligned} \tag{3.8}$$

The distances from these lines, d_1 and d_2, to the separating hyperplane are equal and in order to orientate the hyperplane with the longest distance to the support vectors, it is necessary to maximise the quantity $d_1 + d_2 = 2d_1$, since this is the distance between H_1 and H_2. The distance from the origin to the two

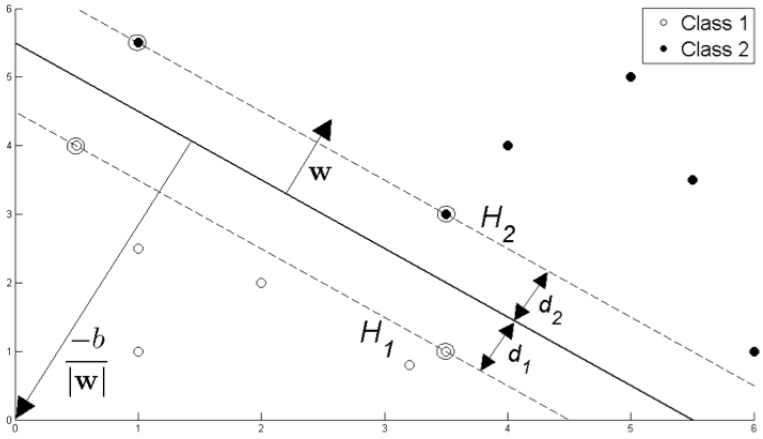


Figure 3.3: Illustration of the binary classification example with two features and six observations of each class. H_1 and H_2 are the lines spanned by the support vectors that are marked with extra circles, and d_1 and d_2 are the distances from the separating hyperplane to H_1 and H_2. Figure from [17].

hyperplanes spanned by the support vectors are given by $\frac{|1-b|}{\|\mathbf{w}\|}$ and $\frac{|-1-b|}{\|\mathbf{w}\|}$ [9], meaning $2d_1$ can be calculated as:

$$\begin{aligned} 2d_1 &= \frac{(1-b)}{\|\mathbf{w}\|} - \frac{(-1-b)}{\|\mathbf{w}\|} \Rightarrow \\ d_1 &= \frac{(1-b)-(-1-b)}{2\|\mathbf{w}\|} \\ &= \frac{(1-b+1+b)}{2\|\mathbf{w}\|} \\ &= \frac{1}{\|\mathbf{w}\|} \end{aligned} \tag{3.9}$$

Finding the optimal separating hyperplane by maximising the distance between the support vectors is therefore equivalent to minimising $\|\mathbf{w}\|$ and accordingly the training data can be described by [12]:

$$\begin{aligned} \mathbf{x}^{(n)} \cdot \mathbf{w} + b \geq +1 \qquad &\text{for} \quad c^{(n)} = +1 \\ \mathbf{x}^{(n)} \cdot \mathbf{w} + b \leq -1 \qquad &\text{for} \quad c^{(n)} = -1 \end{aligned} \tag{3.10}$$

which combined gives:

$$c^{(n)}(\mathbf{x}^{(n)} \cdot \mathbf{w} + b) - 1 \geq 0 \quad \forall_n \ . \tag{3.11}$$

Minimising $\|\mathbf{w}\|$ is equivalent to minimising $\frac{1}{2}\|\mathbf{w}\|^2$, which will turn out to be handy later on, because it enables Quadratic Programming (QP) optimization [17]. The problem formulation can therefore be summarised to:

$$\min(\frac{1}{2}\|\mathbf{w}\|^2) \qquad \text{such that} \qquad c^{(n)}(\mathbf{x}^{(n)} \cdot \mathbf{w} + b) - 1 \geq 0 \quad \forall_n \ . \tag{3.12}$$

In order to solve the constrained minimisation problem in 3.12 positive Lagrange Multiplier's (LM's), $\alpha^{(n)}$ for $n = 1, ...N$ are introduced, where $\alpha^{(n)} \geq 0 \ \forall_n$:

$$\begin{aligned} L(\mathbf{w}, b, \alpha) &\equiv \min \frac{1}{2}\|\mathbf{w}\|^2 - \alpha^{(n)}[c^{(n)}(\mathbf{x}^{(n)} \cdot \mathbf{w} + b) - 1 \ \forall_n] \\ &\equiv \min \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} \alpha^{(n)}[c^{(n)}(\mathbf{x}^{(n)} \cdot \mathbf{w} + b) - 1] \\ &\equiv \min \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} \alpha^{(n)} c^{(n)}(x^{(n)} \cdot \mathbf{w} + b) + \sum_{n=1}^{N} \alpha^{(n)} \ . \end{aligned} \tag{3.13}$$

The switch to Lagrangian formulation is done for two reasons [9]:

1. The original constraints in Eq. 3.11 are substituted by constraints on the LM.

2. The training data occurs only in the form of dot products, which is exploited in the *Kernel Trick* to be explained below.

To satisfy the constraint $\alpha^{(n)} \geq 0 \ \forall_n$, the Lagrangian is minimised by setting the derivatives with respect to $\mathbf{w}$ and b equal to zero [9], yielding:

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{n=1}^{N} \alpha^{(n)} c^{(n)} \mathbf{x}^{(n)} = 0 \\ &\Rightarrow \mathbf{w} = \sum_{n=1}^{N} \alpha^{(n)} c^{(n)} \mathbf{x}^{(n)} \end{aligned} \tag{3.14}$$

$$\frac{\partial L}{\partial b} = \sum_{n=1}^{N} \alpha^{(n)} c^{(n)} = 0 \tag{3.15}$$

The dual formulation is obtained by substituting Eq. 3.14 and 3.15 into Eq. 3.13, where dual refers to solving a different problem, where the solution is the same as the original problem.

$$\begin{aligned} L_{dual} &\equiv \sum_{n=1}^{N} \alpha^{(n)} - \frac{1}{2} \sum_{n=1}^{N} \alpha^{(n)} \alpha^{(n)\prime} c^{(n)} c^{(n)\prime} \mathbf{x}^{(n)} \cdot \mathbf{x}^{(n)\prime} \\ &\text{such that } \alpha^{(n)} \geq 0 \; \forall_n, \; \sum_{n=1}^{N} \alpha^{(n)} c^{(n)} = 0 \end{aligned} \tag{3.16}$$

The transformation from primal to dual formulation, and thereby making the formulation only dependent on $\alpha^{(n)}$, changes the problem from a minimisation of L to a maximisation of L_{dual}. The maximisation of L_{dual} is the objective of the support vector training [9], and can return a vector α by running the before mentioned QP solver. A description of this solver is beyond the scope of this thesis and will not be explained in details. α is substituted into Eq. 3.14 to find $\mathbf{w}$, and the support vectors are used to find b by substituting Eq. 3.14 in to Eq. 3.10. There exists a LM for all training observations, and the observations where $\alpha^{(n)} > 0$ are the support vectors and lie on the lines H_1 and H_2.
The classification of an unknown test observation $\mathbf{x}^{test}$, knowing the optimal separating hyperplane from $\mathbf{w}$ and b is done by evaluating the sign of the function given by:

$$s(\mathbf{x}^{test}) = \mathbf{w} \cdot \mathbf{x}^{test} + b \tag{3.17}$$

3.4.2 Not Fully Linear Separable Classification

Assuming data is fully linear separable is not always realistic, which encourage an extension of the method. This is done by introducing a positive slack variable, $\xi^{(n)}$, $n = 1, ..., N$ [12]. The slack variable induces a penalty to an observation that is on the wrong side of the separating hyperplane and the penalty increases with the distance. In Fig. 3.4 the not fully separable classification problem is illustrated. The introduction of $\xi^{(n)}$ modifies the training data in Eq. 3.11 to

$$c^{(n)}(\mathbf{x}^{(n)} \cdot \mathbf{w} + b) - 1 + \xi^{(n)} \geq 0 \qquad \text{where} \qquad \xi^{(n)} \geq 0 \quad \forall_n \; , \tag{3.18}$$

and from this, the minimisation problem in Eq. 3.12 is transformed to

$$\min(\frac{1}{2}\|\mathbf{w}\|^2 + T \sum_{n=1}^{N} \xi^{(n)}) \qquad \text{such that} \qquad c^{(n)}(\mathbf{x}^{(n)} \cdot \mathbf{w} + b) - 1 + \xi^{(n)} \geq 0 \quad \forall_n \; , \tag{3.19}$$

where T is the regularisation parameter and corresponds to the penalty assigned to the misclassifications. T is user-determined [12]. Switching to the Lagrangian

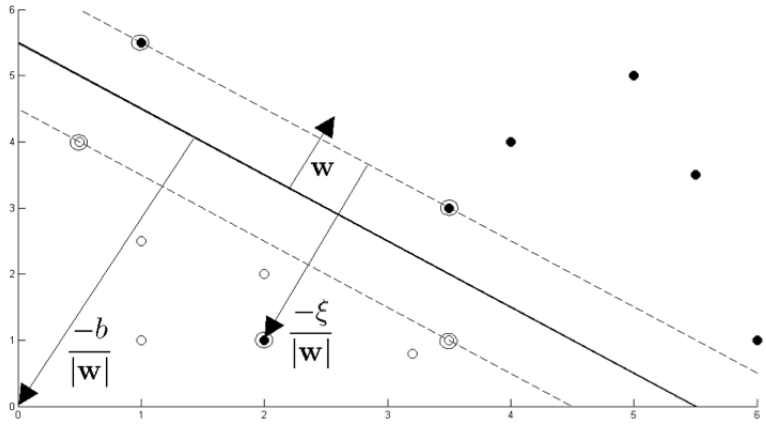


Figure 3.4: Illustration of the binary not fully separable classification example with two features and six observations of each class. The distance from the observation that is on the incorrect side of the separating hyperplane to the line spanned by the support vectors is $\frac{-\xi}{|w|}$. Figure from [17].

formulation gives:

$$L(\mathbf{w}, b, \alpha, \mu) \equiv \frac{1}{2}\|\mathbf{w}\|^2 + T\sum_{n=1}^{N}\xi^{(n)} - \sum_{n=1}^{N}\alpha^{(n)}[c^{(n)}(x^{(n)}\cdot\mathbf{w}+b) - 1 + \xi^{(n)}] - \sum_{n=1}^{N}\mu^{(n)}\xi^{(n)} \ , \tag{3.20}$$

where LM's, $\mu^{(n)} \geq 0$, forces $\xi^{(n)}$ to be positive. Setting the derivative of Eq. 3.20 with respect to $\mathbf{w}$, b and ξ^n equal to zero and substituting into Eq. 3.20 gives the same dual formulation as in Eq. 3.16. However the gradient of Eq. 3.20 with respect to ξ^n gives

$$\frac{\partial L}{\partial \xi^{(n)}} = T - \alpha^{(n)} - \mu^{(n)} = 0 \Rightarrow T = \alpha^{(n)} + \mu^{(n)} \ , \tag{3.21}$$

which together with the constraint $\mu^{(n)} \geq 0 \quad \forall_n$ gives the combined constraint $0 \leq \alpha^{(n)} \leq T$ [17]. The SVM for not fully linear separable classification can

therefore be summarized to:

$$L_{dual} \equiv \sum_{n=1}^{N} \alpha^{(n)} - \frac{1}{2} \sum_{n=1}^{N} \alpha^{(n)} \alpha^{(n)\prime} c^{(n)} c^{(n)\prime} \mathbf{x}^{(n)} \cdot \mathbf{x}^{(n)\prime}$$
$$\text{such that } 0 \leq \alpha^{(n)} \leq T \; \forall_n, \; \sum_{n=1}^{N} \alpha^{(n)} c^{(n)} = 0 \; , \tag{3.22}$$

where L_{dual} is maximised in the same way as previous by a QP solver and returns α that provides $\mathbf{w}$. Contrary to the separable classification problem, the support vectors used to find b now has to satisfy $0 < \alpha^{(n)} < T$.

3.4.3 Nonlinear Support Vector Machine Classification

The problem of classifying a data set that is not linear separable in feature space is done by applying the *Kernel Trick* introduced for the first time by Aizerman et al. in 1964 [1]. In brief the concept of the *Kernel Trick* is to map a non-linear data classification problem into a high-dimensional (or even infinite) feature space where the mapped data classification issue becomes linear and can be handled by linear models[8], such as the linear SVM. The kernel function is given by

$$k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)\prime}) = \phi((\mathbf{x}^{(n)})^T) \phi(\mathbf{x}^{(n)\prime}) \; , \tag{3.23}$$

where $\phi(\mathbf{x}^{(n)})$ is the non-linear mapping. The simplest kernel is the linear kernel

$$k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)\prime}) = \mathbf{x}^{(n)} \cdot \mathbf{x}^{(n)\prime} = (\mathbf{x}^{(n)})^T \mathbf{x}^{(n)\prime} \; , \tag{3.24}$$

and the general idea of the kernel trick is to manipulate the data into only containing inner product between $\mathbf{x}^{(n)\prime}$ and $(\mathbf{x}^{(n)})^T$, meaning the explicit calculation of ϕ is unnecessary [17]. The scalar product can thereby be replaced with any other valid kernel of choice [8]. In the dual formulation of the Lagrangian in Eq. 3.16 the input data only appears as inner products, which makes it a perfect candidate for applying the *Kernel trick* to a non-linear dataset and classify it with the linear SVM. In Fig. 3.5 the *Kernel trick* is illustrated. The dataset is impossible to separate in the original feature space, but mapped into another feature space with higher dimensions, the data is now linear separable. Besides the linear kernel, other choices for kernels are:
Radial Basis Kernel:

$$k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)\prime}) = e^{-\frac{\|\mathbf{x}^{(n)} - \mathbf{x}^{(n)\prime}\|^2}{2\sigma^2}} \; , \tag{3.25}$$

Polynomial Kernel:

$$k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)\prime}) = (\mathbf{x}^{(n)} \cdot \mathbf{x}^{(n)\prime} + a)^2 \; , \tag{3.26}$$

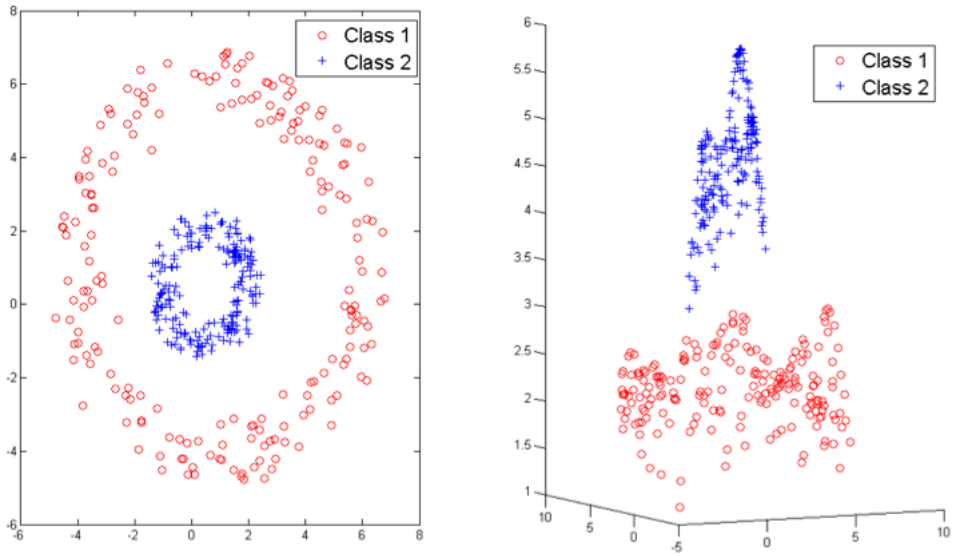


Figure 3.5: Illustration of the *Kernel trick*. Figure from [17].

Sigmoidal Kernel:

$$k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)\prime}) = \tanh(a\mathbf{x}^{(n)} \cdot \mathbf{x}^{(n)\prime} - b) \ , \tag{3.27}$$

and the Matern Kernel:

$$k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)\prime}) = \frac{2^{1-v}}{\Gamma(v)} \left(\frac{\sqrt{2v}|x^{(n)} - x^{(n)\prime}|}{\lambda} \right)^{v} K_v \left(\frac{\sqrt{2v}|x^{(n)} - x^{(n)\prime}|}{\lambda} \right) \tag{3.28}$$

where K_v is the modified Bessel function and the user defined parameters a, b, λ and v controls the behaviour of the kernels.

3.5 Kalman Filtering

The Kalman filter is a generative modelling approach that was invented by R.E. Kalman in 1960 [29]. It provides a set of inference equations to estimate the underlying state, $z \in \Re^m$, of a linear dynamical system, given some noisy measurements, $x \in \Re^n$, [33], [20]. Linear dynamic systems takes the dynamic evolution of the state into account and captures the temporal structure of the data[41].

3.5.1 State process estimation

The linear dynamical system with time step k is given by the generative model below

$$z_k = Hz_{k-1} + w_{k-1} \ , \tag{3.29}$$

and the noisy measurements, $x \in \Re^n$, are given by

$$x_k = Az_k + v_k \ , \tag{3.30}$$

where w_k and v_k are random variables and represents the process and measurement noise respectively. These variables are assumed to be independent, white and normally distributed with covariance Q and R, respectively [23].
H is a state transition model that relates the previous state to the current state and A is the observation model that relates the state, z, to the measurement x. By defining *a priori* state estimate, $\hat{z}_k^- \in \Re^n$, at time step k given knowledge of the process prior to step k, and *a posteriori* state estimate, $\hat{z}_k \in \Re^n$, at time step k given measurement x_k, the equation for calculating *a posteriori* state estimate from a linear combination of *a priori* state estimate is given by

$$\hat{z}_k = \hat{z}_k^- + K(x_k - A\hat{z}_k^-) \ . \tag{3.31}$$

K is the Kalman gain that controls the residual, given by $x_k - A\hat{z}_k^-$. The residual is a measure of the difference between the true measurement x_k and the predicted measurement $A\hat{z}_k^-$. The *a priori* and *a posteriori* estimate errors are defined by

$$e_k^- \equiv z_k - \hat{z}_k^- \tag{3.32}$$

$$e_k \equiv z_k - \hat{z}_k \ , \tag{3.33}$$

which entail that the *a priori* and *a posteriori* estimate error covariance are given by

$$P_k^- = E[e_k^-(e_k^-)^T] \tag{3.34}$$

$$P_k = E[e_k e_k^T] \ . \tag{3.35}$$

The Kalman gain, K, is chosen to be optimal when the *a posteriori* estimate error covariance, P_k, is minimised [43], which is accomplished by substituting Eq. 3.31 into Eq. 3.33

$$e_k = z_k - [\hat{z}_k^- + K(x_k - A\hat{z}_k^-)] \ , \tag{3.36}$$

and then differentiate P_k with respect to K, where Eq. 3.36 is inserted into Eq. 3.35. The expression for K is then obtained by setting the derivative equal to

zero and solving for K. An expression for K, when minimising P_k is then given by [5]:

$$K_{opt} = \frac{P_k^- A^T}{AP_k^- A^T + R} \ . \tag{3.37}$$

Inspecting Eq. 3.37 the weighting of the residual by the Kalman gain can be clarified. When the measurement error covariance goes to zero, K_{opt} approaches A^{-1}, meaning the residual is weighted more and x_k is trusted more. If P_k^- goes to zero the gain approach zero too, which will weight the residual less and the predicted measurement will be trusted more [43].

3.5.2 Kalman Filter Algorithm

The Kalman filter algorithm is a recursive estimator, and can be divided into two processes; prediction and correction [8]. The prediction phase estimates the process state at the current time-step from the previous time-step, and the correction phase provides feedback, the *a posteriori* estimate, from the knowledge of the values obtained in the prediction step, the *a priori* estimate. The prediction process can be described by

$$\hat{z}_k^- = H\hat{z}_{k-1}^- \qquad \text{and} \tag{3.38}$$

$$P_k^- = HP_{k-1}^- H^T + Q \ , \tag{3.39}$$

and the correction process by

$$K_{opt} = \frac{P_k^- A^T}{AP_k^- A^T + R} \ , \tag{3.40}$$

$$\hat{z}_k = \hat{z}_k^- + K_{opt}(x_k - A\hat{z}_k^-) \qquad \text{and} \tag{3.41}$$

$$P_k = (I - K_{opt}A)P_k^- \ . \tag{3.42}$$

To obtain the *a posteriori* state and error covariance estimate, the Kalman gain minimised according to the *a posteriori* estimate error covariance, P_k, is the first step in the correction phase, and the second step is to obtain the measurement x_k. The two values for K_{opt} and x_k are inserted in to Eq. 3.41 and 3.42. This is repeated for every time step.

3.5.3 Gaussian Process Model

The definition of a Gaussian Process (GP) is the probability distribution over functions, $\mathbf{f} = f^{(1)}, f^{(2)}, ..., f^{(N)}$, for which any subset of samples, $\mathbf{X} =$

$x^{(1)}, x^{(2)}, ..., x^{(N)}$, is normally distributed [8], meaning $\mathbf{p}(\mathbf{f}|\mathbf{X}) = \mathcal{N}(0, \mathbf{K})$, where $\mathbf{K}$ is the covariance matrix.
Applying ICA notation:

$$\mathbf{X} = \mathbf{AS} \ , \tag{3.43}$$

where $\mathbf{S}$ is the source matrix and s_i denotes the ith row in $\mathbf{S}$. Each source is then assumed to have a GP prior, yielding

$$\begin{aligned} \mathbf{p}(\mathbf{S}) &= \prod_{i=1}^{I} p(s_i) \ , \\ \text{where} \quad p(s_i) &= \mathcal{N}(0, K_i) \end{aligned} \tag{3.44}$$

In [40] it is stated that the optimisation problems for a GP classifier and a SVM are very similar, because they are both convex. GP's can accordingly be interpreted as a probabilistic version of SVM.

3.5.4 Kalman ICA

ICA is a method that tracks the underlying sources in a measured dataset. The Kalman filter can be viewed as a GP model [31] with independent sources, meaning the Kalman filter can be applied as an ICA algorithm to separate the data into different sources in a generative way. The GP source model to ICA has a cubic complexity and by mapping the GP to a Kalman filter this can be avoided, and replaced by a linear computational complexity instead [24]. In [24] it is shown that this mapping can be accomplished for specific choices of kernel functions such as the Matern Kernel, Eq. 3.28. The details of the mapping of the temporal GP to the Kalman model is quite complex and the details given in [24].
In the Kalman approach the mixing matrix is estimated via the Kalman gain and the sources/states are estimated by the two-step recursive Kalman filter, Eq. 3.38 to 3.42.

Chapter 4

Methods and Implementation

In this chapter the methods applied is this thesis, and how these are implemented are described. The first section introduces the data set including facts and how data is recorded. The next three sections concerns the feature extraction methods and how the feature matrix is obtained. Finally the last sections explains the implementation of the classifiers including parameter settings and cross-validation.

4.1 Software and Toolboxes

The following software and toolboxes has been applied to do the processing and analysis of the EEG data.

- Matlab version 7.12.0. All programming and toolbox used have been carried out in Matlab.
- EEGlab [14]. The toolbox has been used to pre-process data and extract ICA components.

- Modified K nearest neighbour script from [22], used to perform KNN training and testing.
- Naive Bayes Classifier toolbox in Matlab. The toolbox has been applied to create a Naive Bayes Classifier for both training and test.
- LIBSVM [10]. Support vector machine toolbox. Compiled and applied in Matlab to train data and test its performance with a linear classifier.
- Modified script for Kalman ICA, original made by Ricardo Henao. The code has been used to extract Kalman ICA components, used as features.

4.2 Data Acquisition and Preprocessing

The EEG data was obtained at Department of Psychiatry, Hvidovre Hospital, University Hospital of Copenhagen, Denmark, by Sidse M. Arnfred. The data was provided to analysis in this thesis by Morten Mørup. Recordings from five healthy persons with 72 electrodes have been applied, but it is only the first 64 scalp electrodes, that is actually used, because the last eight electrodes only contains noise and reference electrodes. The 64 scalp electrodes are located according to the 10-10 system, and channel 65 and 66 are reference electrodes located on the earlobes. The exact channel location of the 64 scalp electrodes can be obtained in Appendix A.
The five subjects are stimulated with two different stimuli; pulling of the left and right hand, respectively. These two stimuli are repeated 120 times each with a two seconds interval, starting with a right pull. This makes a total of 240 alternating left and right stimuli per subject [2]. The provided data is sampled in Labview with a passband from 0.1-160 Hz and a sampling frequency of 2048 Hz, that was down-sampled to 512 Hz [34]. To correct for drift in the data, EEGlab's FIR filter function was used to high pass the data at 3 Hz. The format of the data is .bdf, meaning channel location and epochs information is included in the datafile and can be imported into EEGlab.

4.3 Feature Extraction

The extraction of features from the raw EEG signal is essential when applying a classifier to separate data in to different stimuli. In this thesis three types of features are tested with the classifiers. These are:

- Features from raw time series.

- Features from Infomax ICA components
- Features from Kalman ICA components

The extraction of these three kinds of features is explained in the following section.

4.3.1 Raw Time Series

The raw time series features are applied in the classification task as a reference measure, because if the time features show better result than the ICA features, it seem rather unnecessary to even bother performing ICA. Using time series as features is initialised with dividing the data in to epochs in EEGlab, where one epoch contains information from one stimuli (either left or right). The epochs were chosen to contain information from stimuli start to 1.5 seconds after, and given a 512 Hz sampling frequency this yield a three dimensional dataset with the dimensions $240 \times 64 \times 768$ ($epochs \times channels \times frames$). The data is normalised over channels to avoid domination of features in greater numeric ranges over features with lower numeric ranges [26], and the features for each epoch are the information from $channels \times frames$. This results in 240 feature vectors, one for each epoch, with size $64 \times 768 = 49152$. The feature matrix ($epochs \times features$) is sorted by alternating right and left stimuli.

4.3.2 Infomax ICA Components

The extraction of Infomax ICA components is done by the *run ICA* option in EEGlab, and initially the function normalises the data by removing the mean for each channel [15]. The Infomax ICA algorithm, explained in Sec. 2.1, is done before the division of the data into epochs and it returns a weight vector, which inverted and multiplied with the original EEG data yields the ICA components. Finding the weight vectors is an iterative process, and the weights are adjusted according to Eq. 2.11, meaning the optimal weights are found when $\Delta \mathbf{W}$ is small enough. The algorithm stops when the user specified value for $\Delta \mathbf{W}$ is reached, or after 512 iterations, and the component is ordered according to how much of the data they account for, starting with the component that accounts for the most [15]. The number of output components is the same as the number of input channels, i.e. 64, but the number of components used for classification in this thesis is ten. The final feature matrix is similar to the one for time series data but with the component information instead of raw signal as features, meaning the feature matrix is given by $epochs \times features$, where

$features = components \times frames$. Illustrations of time and spatial components can be seen in Sec. 2.2.

4.3.3 Kalman ICA Components

The Kalman ICA components are obtained by a modified version of a Matlab script made by Ricardo Henao. The algorithm is performed on the continuous high pass filtered data that is not divided into epochs. The components are found by inference by the implementation of Gibbs sampling (forward filter) [19] and backward sampling. Since the method is very computational expensive the code was run on IMM's clusters, and the number of output components is only ten. The ten output components are used as features for classification by loading them into EEGlab and dividing them in to epochs. This results in a feature matrix with the dimensions $epochs \times features$, where $features = 10 \times 768 = 7680$.

4.4 Classification

The whole point of classification is to classify a dataset into different classes. In this thesis the number of classes is two, and three very different classifiers has been tested on the above described features to perform the task of classifying the data. The three classifiers are:

- K Nearest Neighbour
- Naive Bayes Classifier
- Support Vector Machine

The data set is divided into a test and a training set, and the test set consists of 20% (48 epochs) of the data and the training set of the remaining 80% (192 epochs). In order to use all data points as both training and test points [40], a 5-fold cross-validation algorithm is implemented, and the error rate is represented by the average of these five folds. The 5-fold cross-validation is illustrated in Fig. 4.1. The error rates are obtained by comparing the known true class label with the class labels predicted by the three classifiers. In the following sections the implementation of the classifiers is described. For the purpose of validation all of the classifiers have been tested on an artificial dataset with two very different stimuli, and all classifiers separated the stimuli perfectly.

Figure 4.1: The applied 5-fold cross-validation method.

4.4.1 K Nearest Neighbour

The KNN algorithm is a modified version of the script from [22]. The algorithm is provided with a training and test data with corresponding class labels, as described above, and initially calculates the optimal value for K by the nested cross-validation method. The optimal K with a maximum value of 191 (see Sec. 3.2), is applied when calculating the error rate for the test set.

4.4.2 Naive Bayes Classifier

The NBC algorithm is implemented by the use of the built-in toolbox in Matlab, and the function consists of a fitting and predicting part. In the fitting step the decision boundary is created by applying the training set, and additionally the function calculates the prior probability from the class labels. The only user specified option is the distribution used for fitting the data, which is chosen to be Gaussian. In the predicting step the decision boundary, obtained from the fitting step, is applied to the test set and thereby provides an error rate. A detailed description of the NBC algorithm is provided in Sec. 3.3.

4.4.3 Support Vector Machine

The LIBSVM package has been applied to implement a SVM for classification. The toolbox provides a model function that creates a hyperplane from the training set, and a test function that uses this hyperplane to classify the test set. The training step creates the optimal hyperplane by calculating the Support Vectors, $\mathbf{w}$ and b [12]. The SVM algorithm is explained in Sec. 3.4, where details regarding the calculation of the optimal hyperplane can be found. Different types of kernels can be applied in the LIBSVM package, but in this thesis only the linear kernel has been tested. The regularisation parameter, T, is set to default, because it is not a priority to test the effect of varying T. The default value of T is 1.

Chapter 5

Results

This chapter concerns the results obtained in this thesis. The first section presents a comparison of the performance of the three classifiers for each of the different types of features. In addition percentage and visualisation of the significant different features between the two classes are provided. The second section contains a visual inspection of the ICA components averaged over epochs.

5.1 Classification of Left and Right Stimuli

In the following the results for classification of left and right stimulation for five different subjects are represented. The results are, as mentioned previously, an average of five error rates, obtained by 5-fold cross-validation. The number of right and left stimulation is equal, and a random pick of an epoch is therefore 50% for both classes, meaning in order for the classifiers to be better than flipping a coin the error rates has to be below 50%.

5.1.1 Time series features

The error rates for normalised time features for the five subjects are shown in Tab. 5.1. In general the classifiers perform almost equally good for the five subjects, but for KNN and NBC the error rates are very high and in some cases close to 50%. The NBC seems to perform just a tiny bit better than KNN, but non of them show impressive results. The best performance is 43.75% and 41.25% for KNN and NBC, respectively. The SVM classifier clearly provides the lowest error rates compared to the other two, and the error rate accomplished for subject 5 is 29.17% and is the lowest seen.

Table 5.1: Error rates for classification with three different classifiers for the five subjects with normalised time series features.

Classifier/Subjects	1	2	3	4	5
KNN	0.4458	0.4375	0.4750	0.4542	0.4667
NBC	0.4208	0.4750	0.4292	0.4125	0.4292
SVM	0.3167	0.3375	0.3250	0.3083	0.2917

5.1.2 Infomax ICA Components

The error rates, obtained by applying the Infomax ICA components, for the five subjects are shown in Tab. 5.2. The features are not normalised, because this is done as a part of the ICA algorithm in EEGlab. The error rates are obtained by applying the ten components that account for most of the data in the classification process despite the fact that the algorithm provides 64. This is done to make the results comparable to the Kalman ICA components. Results for 16, 30 and 64 components are provided in Appendix B.

Table 5.2: Error rates for classification with three different classifiers for the five subjects with 10 Infomax ICA components as features.

Classifier/Subjects	1	2	3	4	5
KNN	0.4292	0.4417	0.4500	0.3708	0.4417
NBC	0.3042	0.4250	0.4625	0.2625	0.4958
SVM	0.2125	0.3417	0.3750	0.2833	0.3750

The error rates for KNN is in general very high, whereas the results for NBC are rather varying between subjects, spanning from 26.25% to 49.58%. The best

performance is accomplished by the SVM, and the lowest error rate on 21.25% is obtained for subject 1 with the SVM classifier.

5.1.3 Kalman ICA Components

The classifiers are tested on both normalised and non-normalised Kalman ICA components, and the obtained error rates are listed in Tab. 5.4 and 5.3, respectively. The general performance is much better than the performance for time series, and a little better than Infomax ICA, except for a few outliers.

Table 5.3: Error rates for classification with three different classifiers for the five subjects with non-normalised Kalman ICA components as features.

Classifier/Subjects	1	2	3	4	5
KNN	0.5625	0.4375	0.3750	0.2917	0.4167
NBC	0.3667	0.4542	0.3542	0.3667	0.3292
SVM	0.2083	0.1917	0.1333	0.2458	0.2125

The effect of normalising is ambiguous, but in most cases the performance is better or the same with the exception of the two highlighted values in Tab. 5.4. The lowest error rate for KNN is 29.17% obtained with the non-normalised features, and the best performance for NBC and SVM obtained with normalised components is 32.92% and 12.92%, respectively. Again the SVM seems to perform the best.

Table 5.4: Error rates for classification with three different classifiers for the five subjects with normalised Kalman ICA components as features.

Classifier/Subjects	1	2	3	4	5
KNN	0.4583	0.3542	0.3750	**0.4375**	**0.4583**
NBC	0.3667	0.4542	0.3542	0.3667	0.3292
SVM	0.1333	0.1792	0.1292	0.2458	0.2167

5.1.4 Comparison of Classifiers and Features

An average of the error rates has been calculated to compare the features and classifiers. The pattern in Tab. 5.5 is pretty clear; the best classifier is the SVM

and the best features for classification is the normalised Kalman ICA components. The overall lowest error rate is 12.92% for subject 3 with normalised Kalman ICA components, classified by SVM, see Tab. 5.4. From Tab. 5.5

Table 5.5: Error rates averaged over subjects for all three classifiers and features. The Kalman ICA components is normalised.

Classifier/Features	Time series	Infomax ICA	Kalman ICA
KNN	0.4558	0.4267	0.4167
NBC	0.4333	0.3900	0.3742
SVM	0.3158	0.3175	0.1808

it is also evident that the ten Kalman ICA components are more well suited for classification than the ten Infomax ICA components for the data applied in this thesis. In Fig. 5.1 20 right and left epochs for subject 3 are shown for two random Kalman features in feature-space. Even though this is only two out of 7680 features, distinction between the two classes in the two dimensional feature-space is visible.

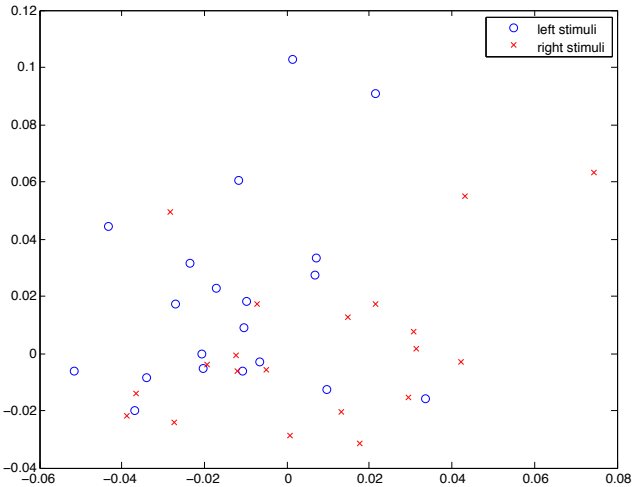


Figure 5.1: 20 right and left stimulation for subject 3 shown in feature space for two random features.

5.1.5 Significant Different Features between Left and Right Stimuli

Another way to illustrate the suitability of the three feature types for classification, is to calculate the amount of features that shows significant difference between the two classes. This is done by a simple two-sample t-test that reveals how many and which features that show significant difference between the two stimuli. In Tab. 5.6 the percentage of significant different features with a significance level at 1% for all subjects is shown. Hence the higher the value the more significant difference is seen for the features.

Table 5.6: Percentage of significant different features.

Classifier/Subjects	1	2	3	4	5
Time series	0.4	0.5	0.4	1.0	0.6
Infomax ICA	1.9	1.1	0.7	1.7	0.5
Kalman ICA	2.0	1.5	2.0	2.0	1.5

The percentage of significant different features in Tab. 5.6 is very low and the highest value is 2%, but the pattern is almost unambiguous, and corresponds to the classification performance yield by the three types of features, meaning the highest percentages are obtained by using the Kalman features, and the lowest by using the time features. In Fig. 5.2, 5.3 and 5.4 the distribution according to channels/components and time after stimuli of the significant different features for subject 3 is visualised. Figures for the other subjects are provided in Appendix C. These figures show that the discrimination between features is more pronounced right after the start of the stimuli. Especially around 0.1 and 0.6 seconds after stimuli in Fig. 5.2 and 5.4, the significant different features are in the majority. In Fig. 5.4 for the Kalman features component two, seven and nine seem to be contributing most to the significant features, and for the Infomax features in Fig. 5.3, five is the most dominant component. However the time pattern around 0.1 and 0.6 seconds is not as pronounced for these features. In Fig. 5.2 it is verified that the channels covering motor cortex are the channels that contribute with most feature difference.

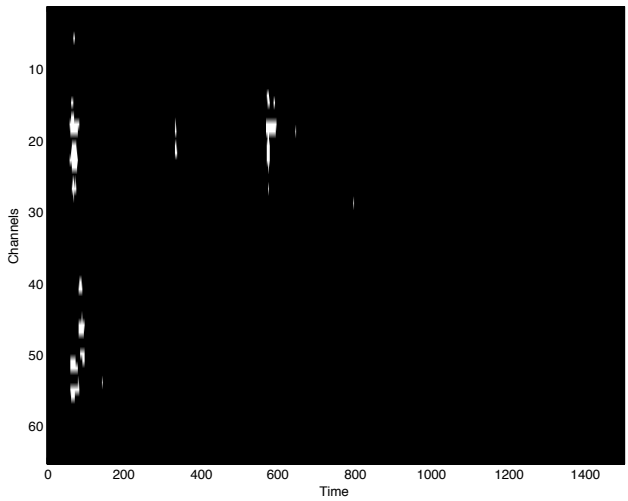


Figure 5.2: Visualisation of significant different features for time series for subject 3.

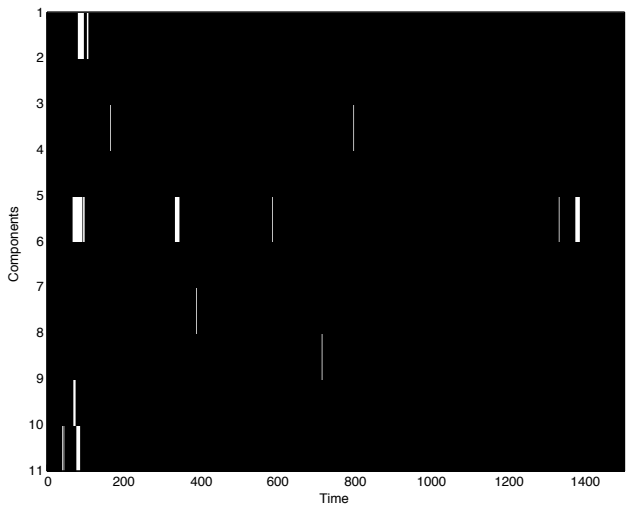


Figure 5.3: Visualisation of significant different features for Infomax ICA components for subject 3.

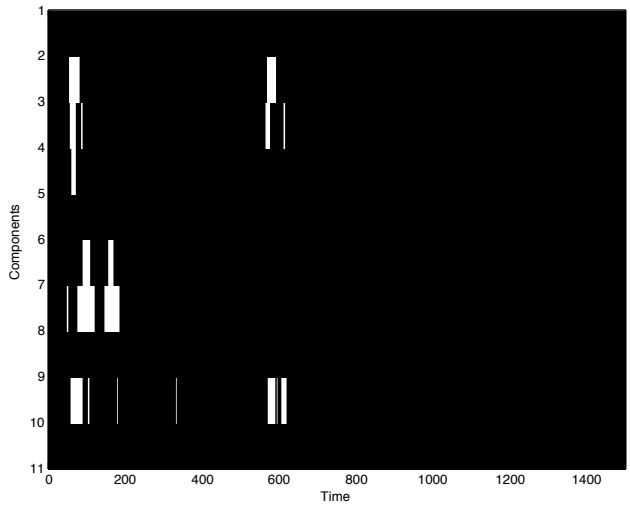


Figure 5.4: Visualisation of significant different features for Kalman ICA components for subject 3.

5.2 Visualisation of ICA Components

In Sec. 5.1.4 it was established that the ten Kalman ICA components shows better results than the ten Infomax ICA components in the classification task, and especially for subject 3 the performance difference is evident. Accordingly, a visual inspection of the ICA components for subject 3 is provided in this section. Besides, the visualisation can be applied for artifact detection. To study the general pattern of the left and right stimuli an average over epochs has been calculated. Fig. 5.5 is the average for Infomax ICA components and Fig. 5.6 is the averaged Kalman ICA components. Figures for the other subjects is provided in Appendix D. The Kalman and Infomax ICA components are visual very different from each other. The Infomax components contain in general more high frequencies in the ten components, whereas the Kalman components shows lower frequency content in some components. Discriminating between left versus right stimuli for Infomax components in Fig. 5.5 is a little difficult partly because of the high frequent nature of the components and furthermore the majority of the components is very similar.
The distinction between the left and right stimuli for the Kalman components in Fig. 5.6 is a little more pronounced. In Fig. 5.6 it is evident that most of the components shows activity at 0.1 and 0.6 seconds after stimuli start. The third component is almost identical for the two stimuli, whereas component two, seven and nine show distinction in the nature of the activation between the two

stimuli. This indicates that component two, seven and nine might be related to stimuli, whereas component three probably is caused by an artifact. The visualisation in Fig. 5.4 showed significant different features between the two classes at the same time and components as in Fig. 5.6, and therefore suggests that these activations are stimuli related.
In Fig. 5.3 five was the most dominant component, which corresponds to the visualisation of the component in Fig. 5.5, meaning the difference between left and right stimuli is conspicuous. The averaged Infomax ICA component five and Kalman ICA component two with errorbars are provided In Fig. 5.7 and 5.8, respectively. These are examples of visual distinguishable components in Fig. 5.5 and 5.6, but the size of the error bars indicates that the obvious visual difference should be taken with precautions.
The activation at 0.1 and 0.6 seconds is visible in the Infomax ICA components as well and especially component six in Fig. 5.5 illustrates this phenomena. Component six for the two stimuli is very similar and this could be the identification of an artifact, since the activation occurs independently of stimuli type.

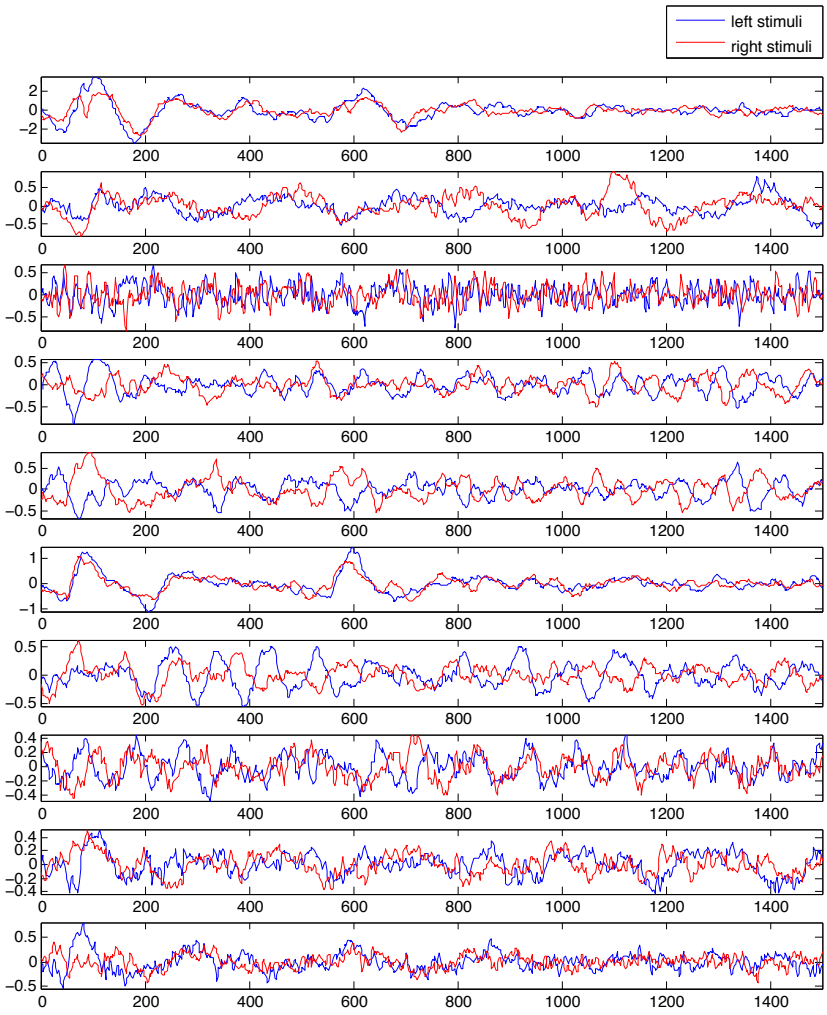


Figure 5.5: Epoch-averaged Infomax ICA components for both left and right stimuli for subject 3.

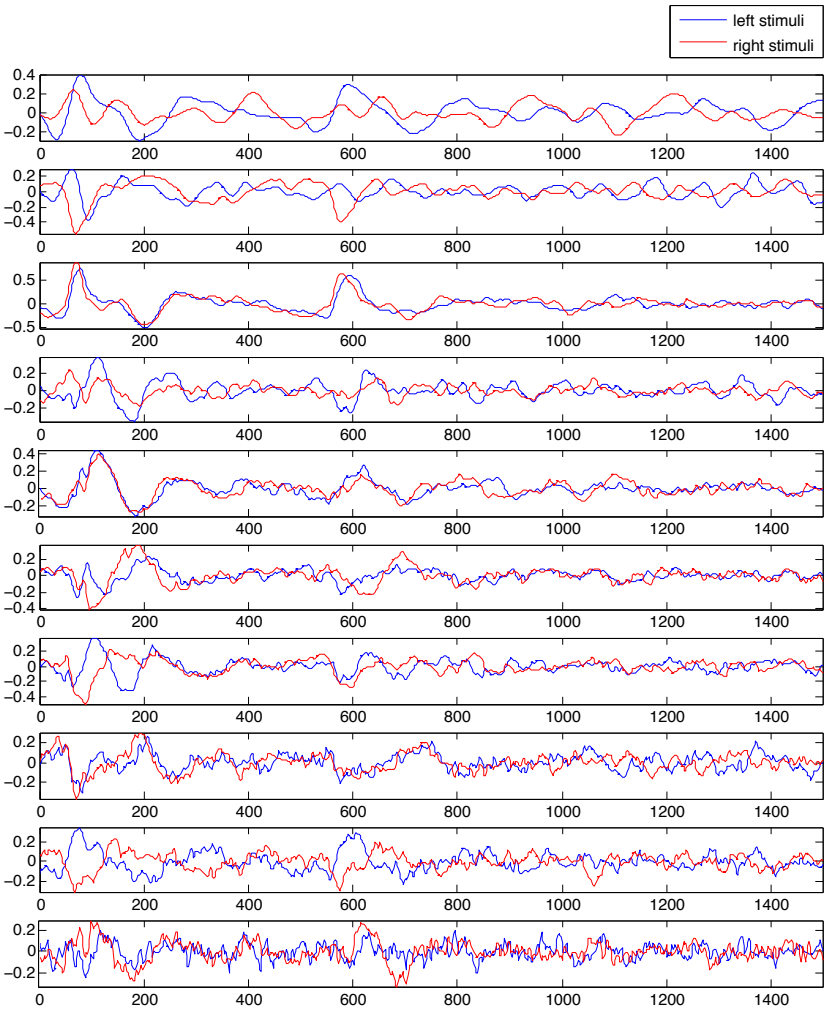


Figure 5.6: Epoch-averaged normalised Kalman ICA components for both left and right stimuli for subject 3.

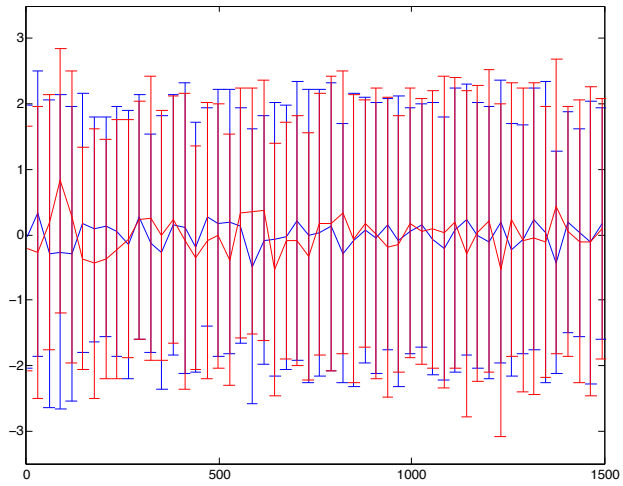


Figure 5.7: Epoch-averaged Infomax ICA component five for both left and right stimuli with errorbars for subject 3.

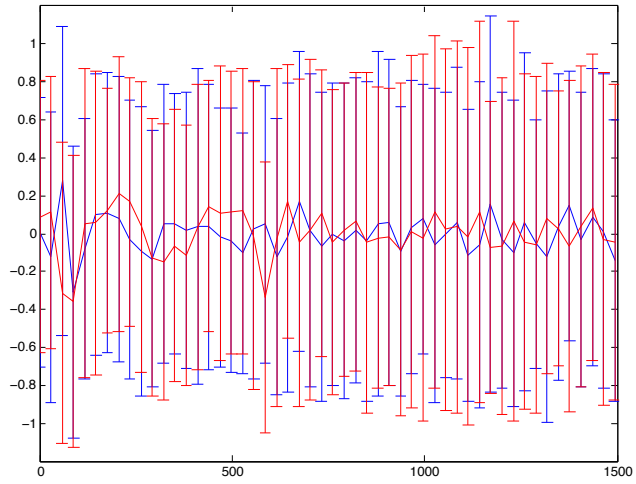


Figure 5.8: Epoch-averaged Kalman ICA component two for both left and right stimuli with errorbars for subject 3.

Chapter 6

Discussion

This chapter contains a discussion of the result obtained in Chapter 5. The first section concerns possible explanation for high and low performance for both classifiers and feature extraction methods. The second section is an analysis of the visualisation of the components, and a comparison of the visualisations with the classification performance. Finally, the last section provides suggestions for areas to improve and explore in continuation of this thesis.

6.1 Classification Performance

The results of classification of left and right stimuli are very dependent on both classification and feature extraction method. In general the highest error rates are provided by the time series features and on average over subjects, Tab. 5.5, the time series features show the worst performance. This tendency is not unexpected, because even though the amount of information is bigger than for the ICA features, no attempt to concentrate or separate the features has been performed, and it is likely that most of the information is contributing with noise instead of valuable information related to stimuli. The Infomax ICA algorithm provides on average the second best type of feature for classification, which is likely to be related to the concentration of the informative features in the ten components. However Tab. 5.2 and 5.1 showed that in some cases the Infomax

ICA features are outmatched by the time series features. This can be explained by the lack of tracking stimuli related components in the ten components and loss of valuable information instead of concentration. This is consistent with Tab. 5.6, since low percentage of significant different features is correlated with high error rate. The Kalman ICA components are evidently providing the best features for classification on the dataset used in this thesis, and the lowest obtained error rate is 13%. This suggests that the Kalman filtering approach is more capable of detecting the temporal stimuli than the Infomax ICA algorithm. The better performance is probably caused by the temporal aspect of the Kalman filter that facilitates capturing of the temporal evolution of the data. Furthermore, the percentage of significant different features is the highest, and accordingly a concentration of more of the important information most be collected in the ten components than in the ten ICA components.
From a general perspective the two simple classifiers, KNN and NBC, is clearly performing worse than the SVM classifier, which is probably because of the similarity between the two types of stimuli in the EEG signal. Accordingly the classifiers are not able to create a decision boundary that completely separates the two classes. The SVM classifier accomplish the lowest error rates, and this is likely to be caused by the more advanced nature of this classifier compared to KNN and NBC. The t-test reveals that only around 2% of the Kalman ICA features is different between the two stimuli, but the SVM classifier is able to find a hyperplane that classifies the data with an accuracy of 87% for some subjects.

6.2 Visual Comparison of ICA Components

The visual inspection of the ICA components reveals that the Kalman and Infomax algorithms divide the EEG data into very different components, and this is probably the reason for the variation between the classification performance. Furthermore, the visual distinction between the two stimuli is harder to track for the Infomax components than for the Kalman components. Consequently the visualisation of the ICA components corresponds fairly good to the classification results, discussed above.
The averaged components showed peaks/valleys at 0.1 and 0.6 seconds after stimuli start, but it is difficult to conclude if it originates from stimuli, artifacts or a combination. The difference between left and right stimuli in component two for the averaged Kalman components suggest the peak being caused by stimuli, but the size of the error bars, obtained in figure 5.8 limits the credibility of the visual distinction. However the obvious similarity in component three suggests that the peaks being caused by artifact. Comparing the concentration of significant different features at 0.1 and 0.6 seconds after stimuli with the ICA

components makes a strong indication of the peaks/valleys being stimuli related. Finally, the visual inspection of component three, seven and nine for the kalman algorithm in Fig. 5.4 and 5.6 is consistent.

6.3 Future Work

The Kalman ICA algorithm applied to EEG data shows promising results, and a further investigation of the application of this could be interesting. The algorithm is currently very heavy and an optimisation would accordingly be desirable in the long run. In addition it could be attractive to reformulate the algorithm to a plug-in, which could be used in e.g. EEGlab, since the Kalman ICA algorithm returns a different result than the Infomax ICA. Finally, further development of the Kalman algorithm to perform the original object of this thesis could be of great interest.

Chapter 7

Conclusion

Classification of left and right hand-pull stimuli by applying EEG data from five subjects has been carried out. By using the ten temporal Kalman ICA components as features the lowest error rate on 13% was accomplished. The best results for time series and ten temporal Infomax ICA features were 29% and 21%, respectively. All of the three error rates were obtained by applying the SVM classifier, which in general performs way better than the KNN and NBC classifiers. The paradigm prepare the ground for temporal distinction between the two classes, and the Kalman features classified by SVM prove that this discrimination indeed can be obtained. Even though the percentage of significant different features between the two stimuli is low for all three features with a maximum of 2%, it corresponds to the classification performance and provides a verification of the results.

The visual inspection of the ten ICA components together with the visualisation of the significant different features between the two stimuli showed that some components are related to stimuli, whereas others might be caused by artifacts. In addition activation around 0.1 and 0.6 seconds after stimuli was observed and the components with significant different features showed visual distinction as well.

It can be concluded that the Kalman ICA components for the data used in this thesis captures the stimuli in the EEG signal despite the fact that some components are most likely to be noise and artifact related. Accordingly, the components are well suited as features in a classification task.

Appendix A

Channel Locations

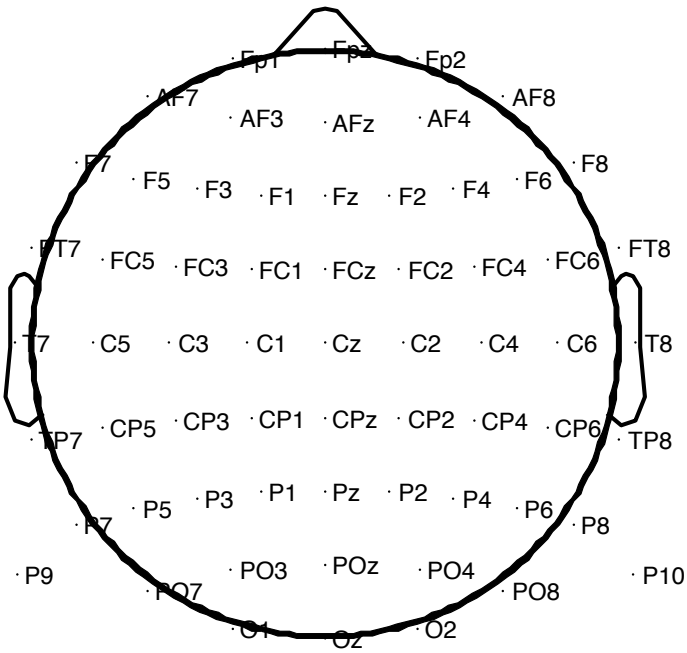


Figure A.1: Channel location for the 64 scalp electrode, placed according to the 10-10 system

Appendix B

Error Rates for Infomax ICA

Table B.1: Error rates for classification with three different classifiers for the five subjects with 16 Infomax ICA components as features.

Classifier/Subjects	1	2	3	4	5
KNN	0.3667	0.3917	0.5125	0.4083	0.4042
NBC	0.2625	0.4042	0.4250	0.2458	0.4750
SVM	0.2042	0.3083	0.3500	0.2083	0.3917

Table B.2: Error rates for classification with three different classifiers for the five subjects with 30 Infomax ICA components as features.

Classifier/Subjects	1	2	3	4	5
KNN	0.4417	0.3750	0.4625	0.3792	0.3833
NBC	0.2292	0.4542	0.3083	0.1875	0.4208
SVM	0.1667	0.2250	0.2667	0.1750	0.3042

Table B.3: Error rates for classification with three different classifiers for the five subjects with 64 Infomax ICA components as features.

Classifier/Subjects	1	2	3	4	5
KNN	0.4458	0.3958	0.4625	0.3625	0.4667
NBC	0.1875	0.3375	0.2375	0.4000	0.3083
SVM	0.1583	0.2042	0.2542	0.1708	0.2833

Appendix C

Visualisation of Significant Different Features

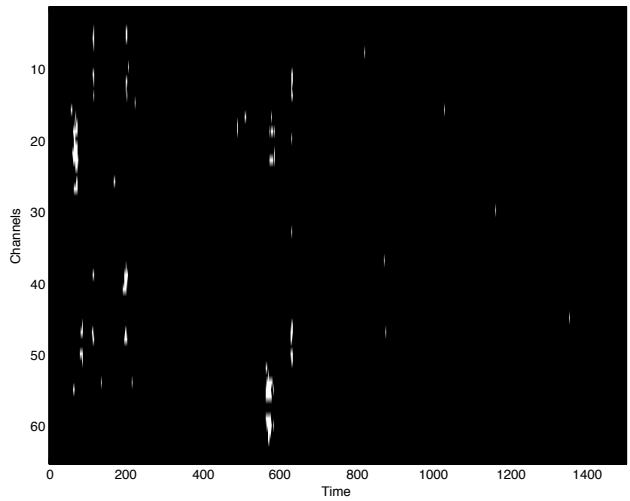


Figure C.1: Visualisation of significant different features for time series. Subject 1.

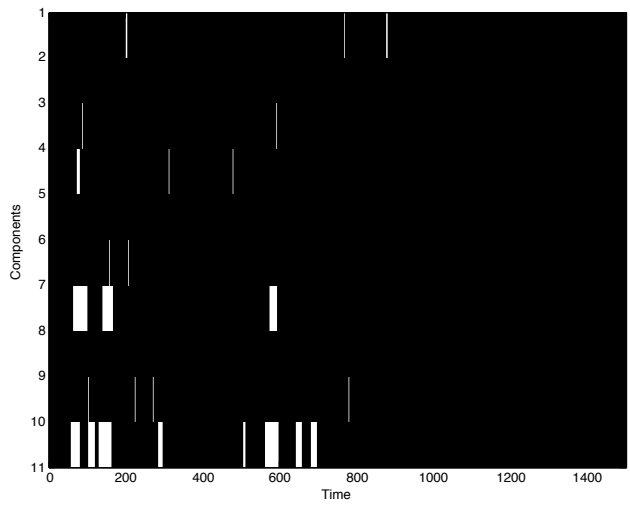


Figure C.2: Visualisation of significant different features for Infomax ICA components. Subject 1.

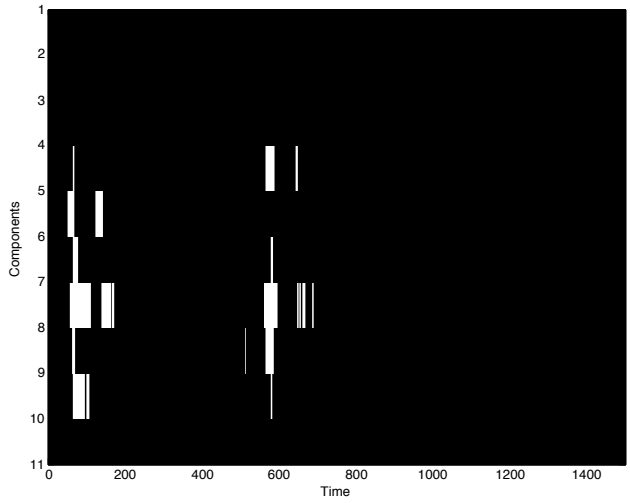


Figure C.3: Visualisation of significant different features for Kalman ICA components. Subject 1.

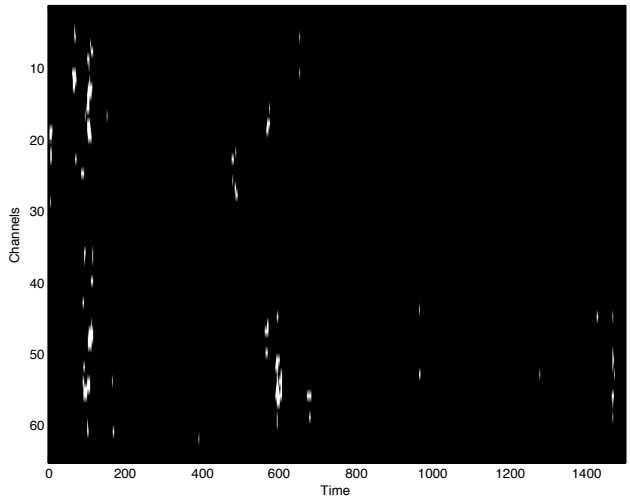


Figure C.4: Visualisation of significant different features for time series. Subject 2.

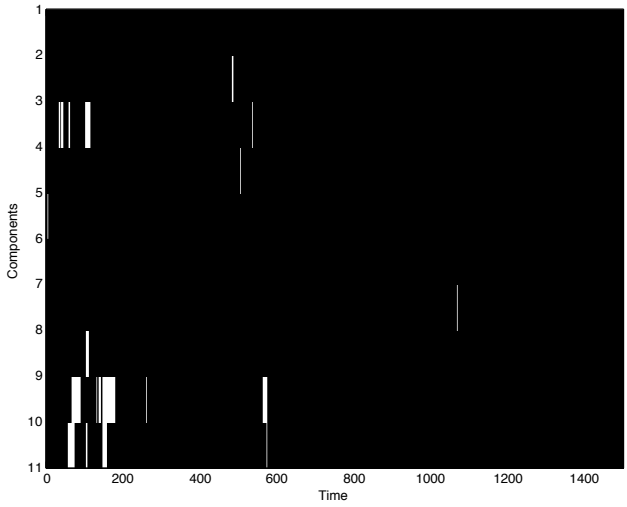


Figure C.5: Visualisation of significant different features for Infomax ICA components. Subject 2.

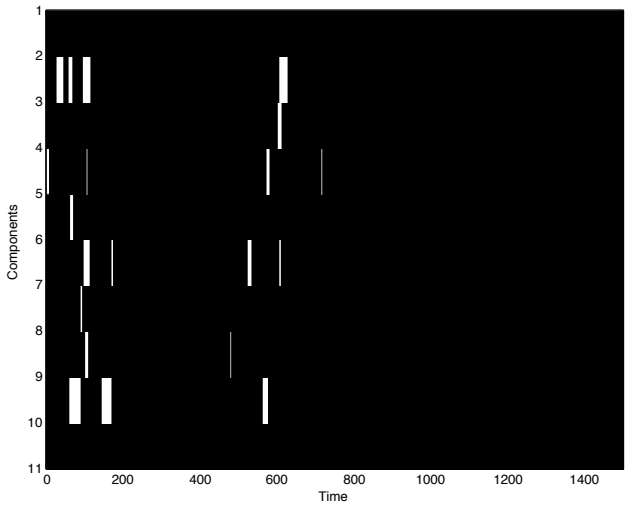


Figure C.6: Visualisation of significant different features for Kalman ICA components. Subject 2.

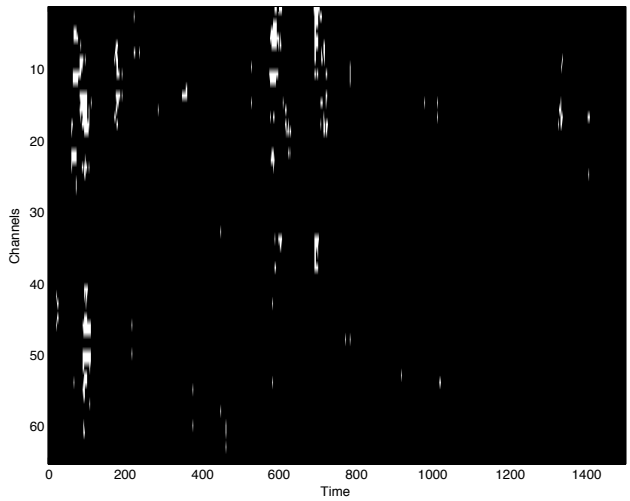


Figure C.7: Visualisation of significant different features for time series. Subject 4.

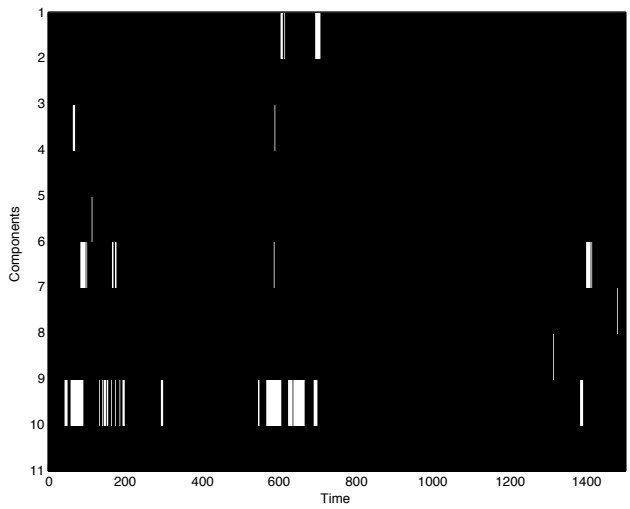


Figure C.8: Visualisation of significant different features for Infomax ICA components. Subject 4.

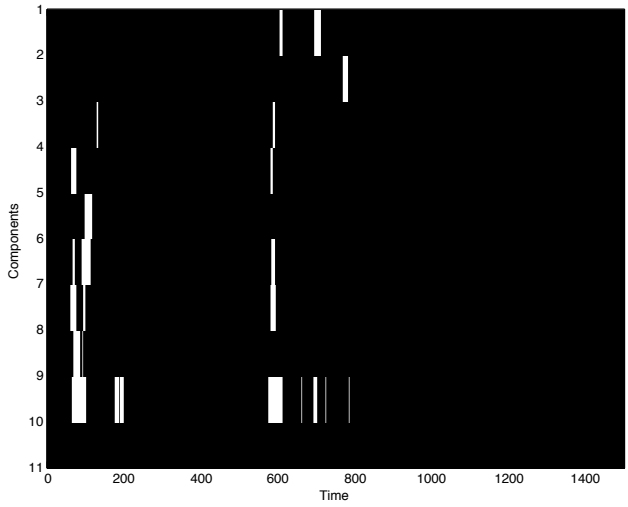


Figure C.9: Visualisation of significant different features for Kalman ICA components. Subject 4.

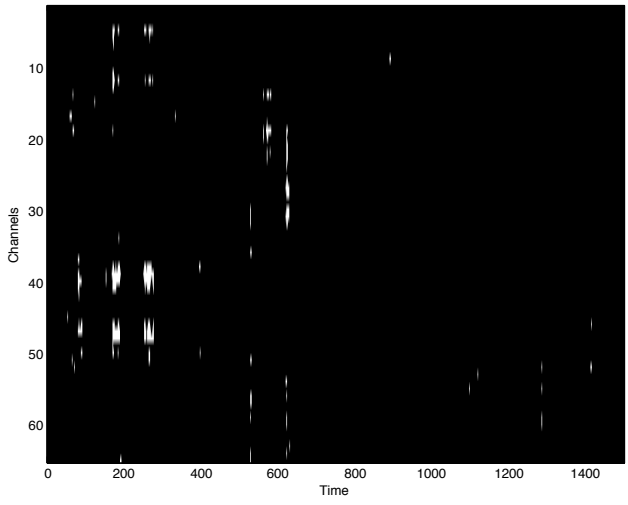


Figure C.10: Visualisation of significant different features for time series. Subject 5.

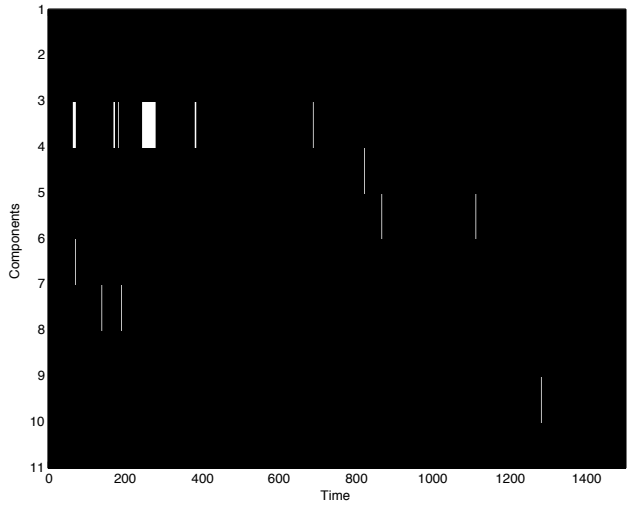


Figure C.11: Visualisation of significant different features for Infomax ICA components. Subject 5.

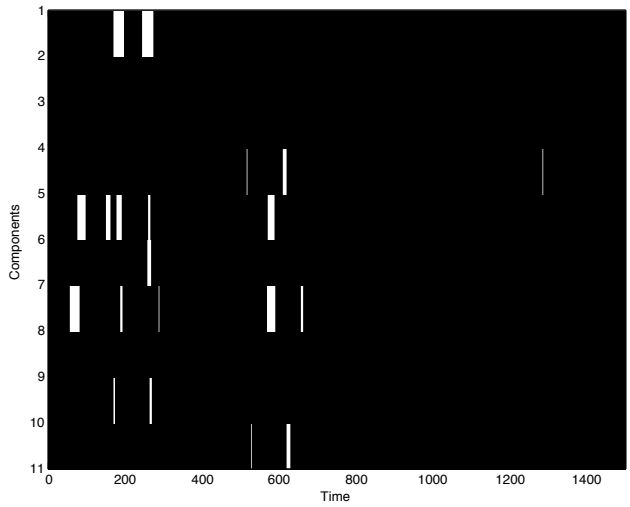


Figure C.12: Visualisation of significant different features for Kalman ICA components. Subject 5.

Appendix D

Averaged Components over Epochs

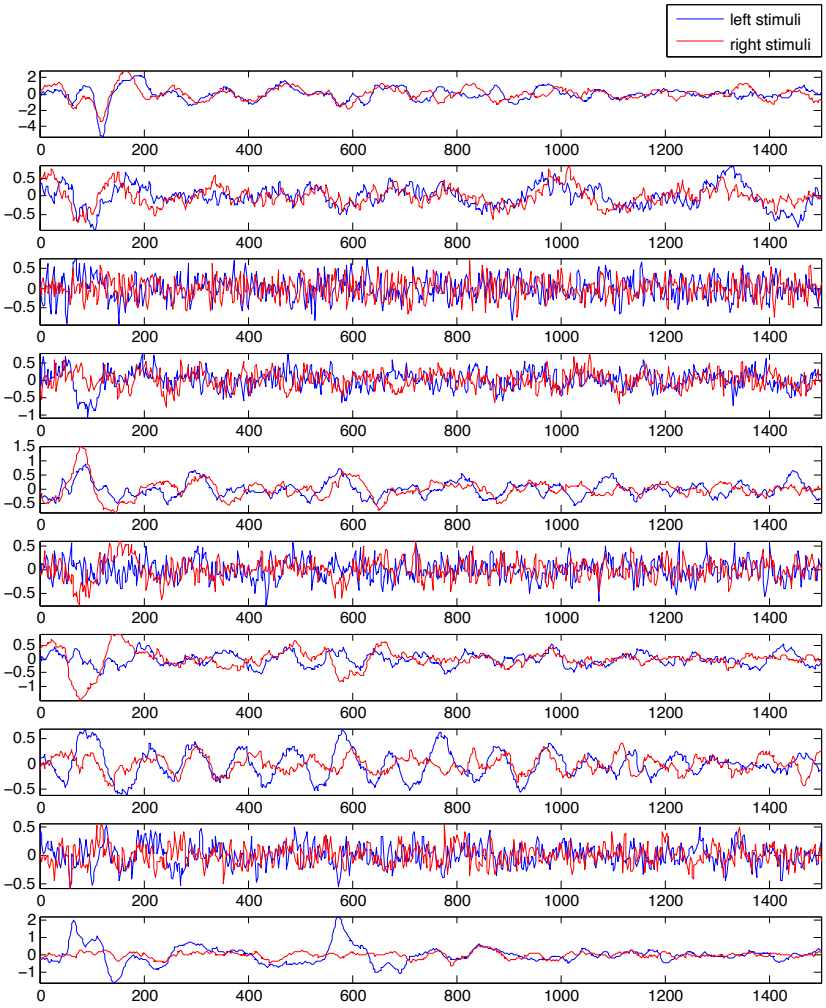


Figure D.1: Averaged Infomax ICA components for both left and right stimuli. Subject 1.

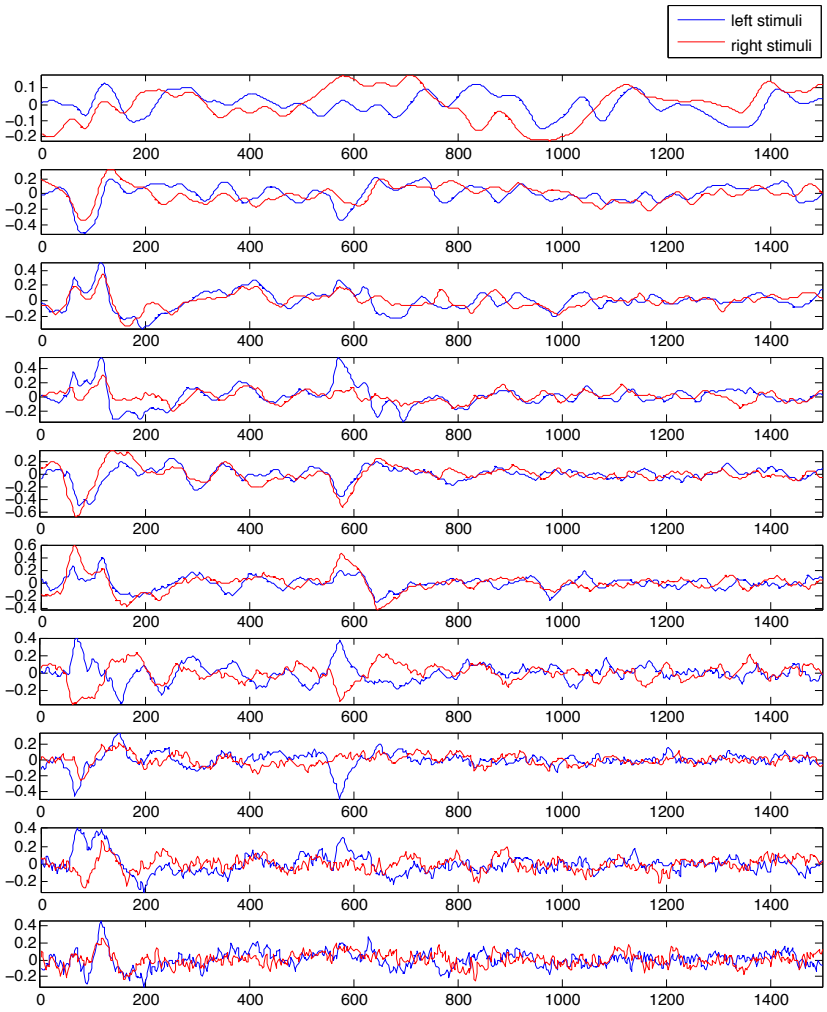


Figure D.2: Averaged normalised Kalman ICA components for both left and right stimuli. Subject 1.

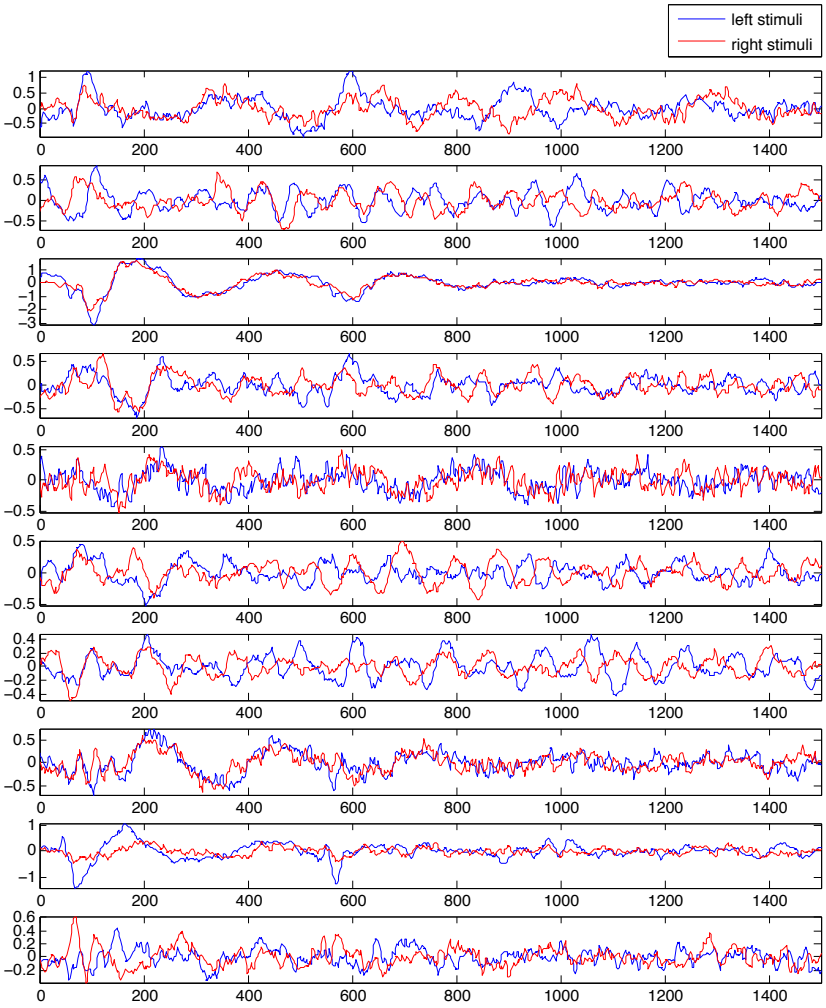


Figure D.3: Averaged Infomax ICA components for both left and right stimuli. Subject 2.

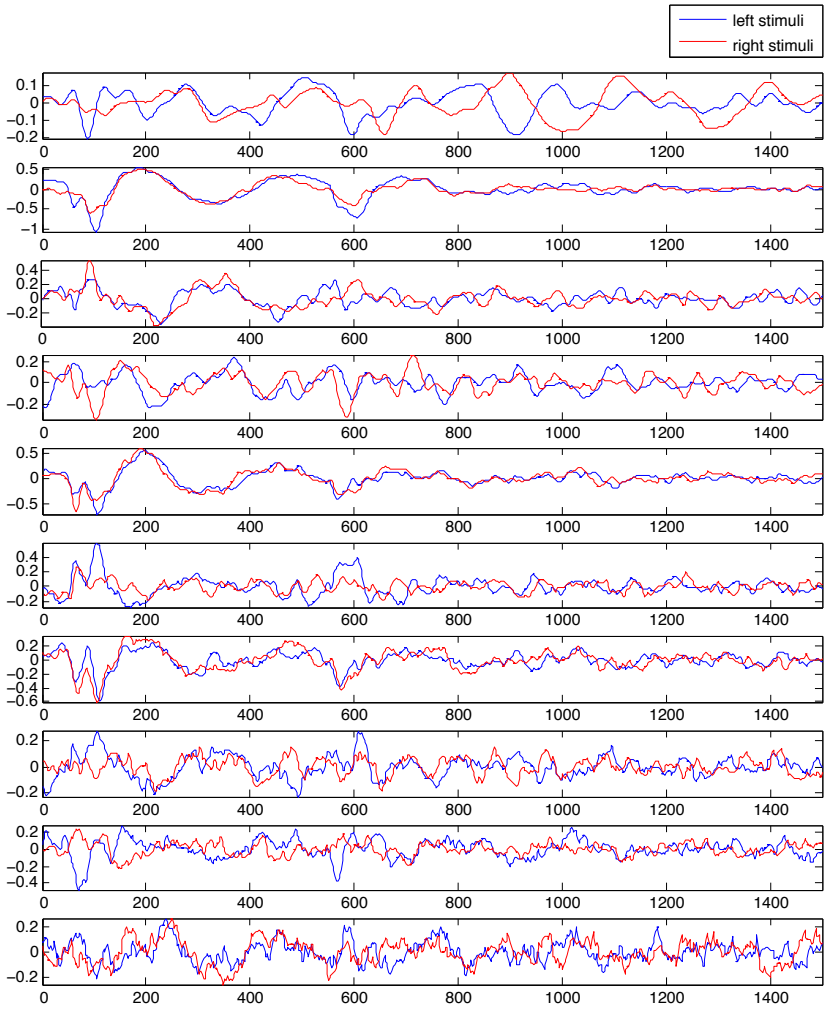


Figure D.4: Averaged normalised Kalman ICA components for both left and right stimuli. Subject 2.

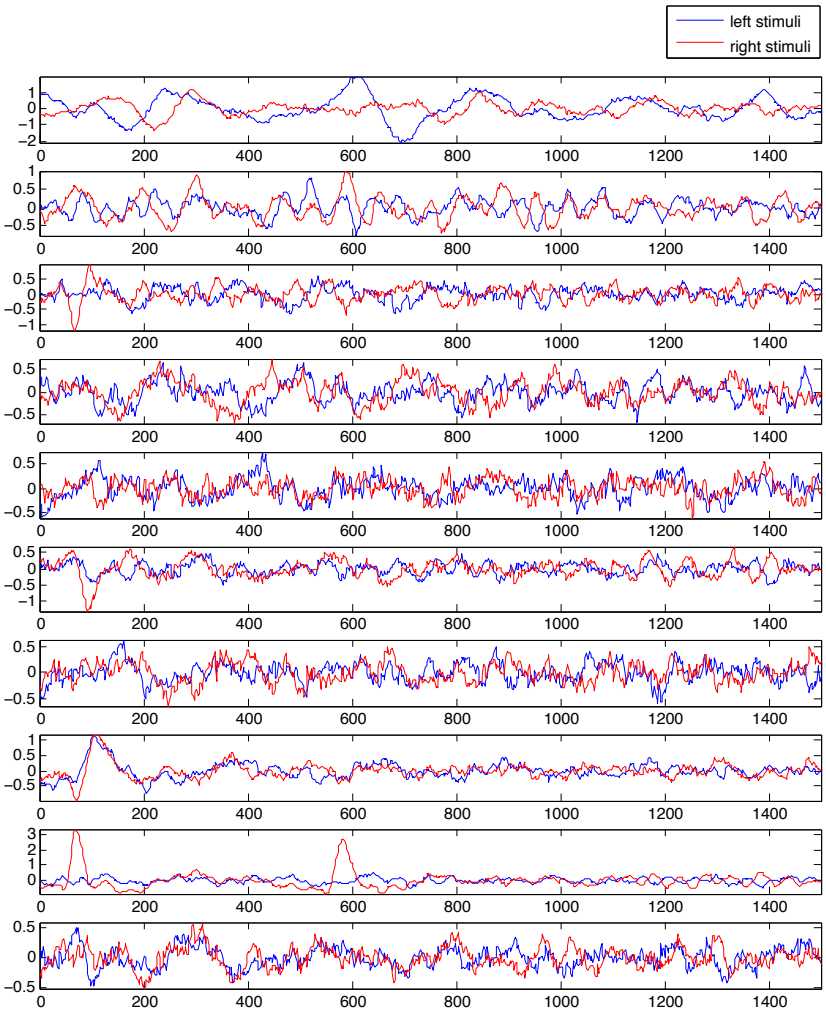


Figure D.5: Averaged Infomax ICA components for both left and right stimuli. Subject 4.

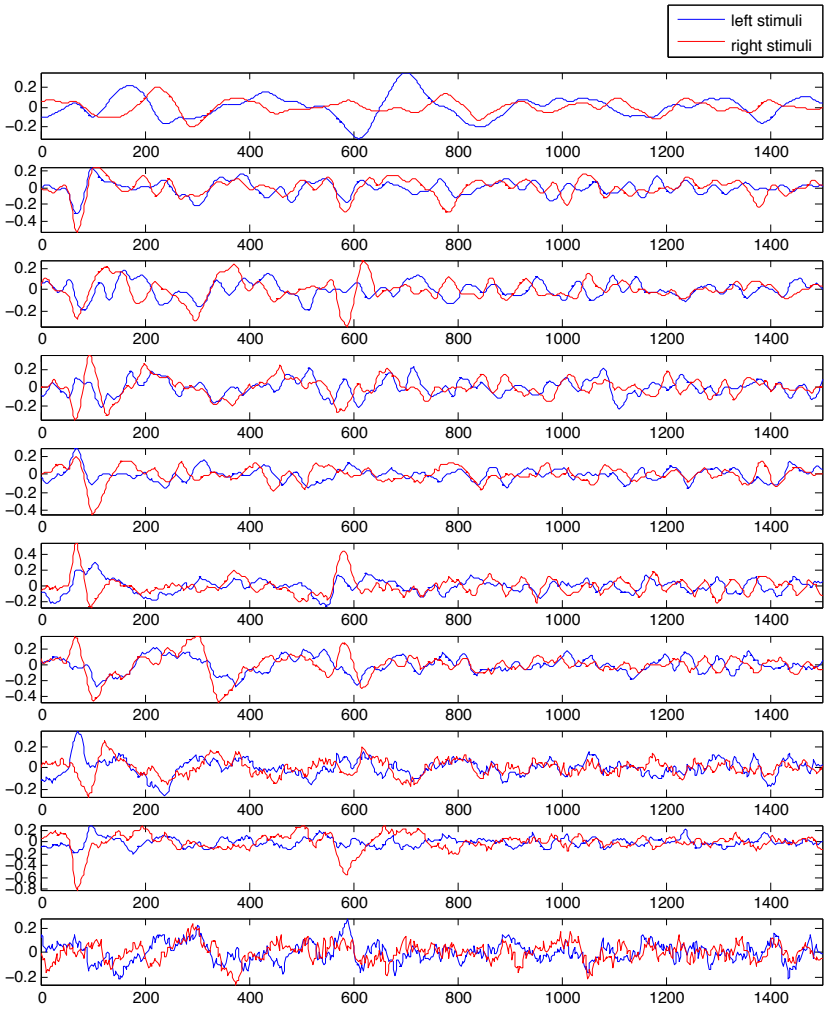


Figure D.6: Averaged normalised Kalman ICA components for both left and right stimuli. Subject 4.

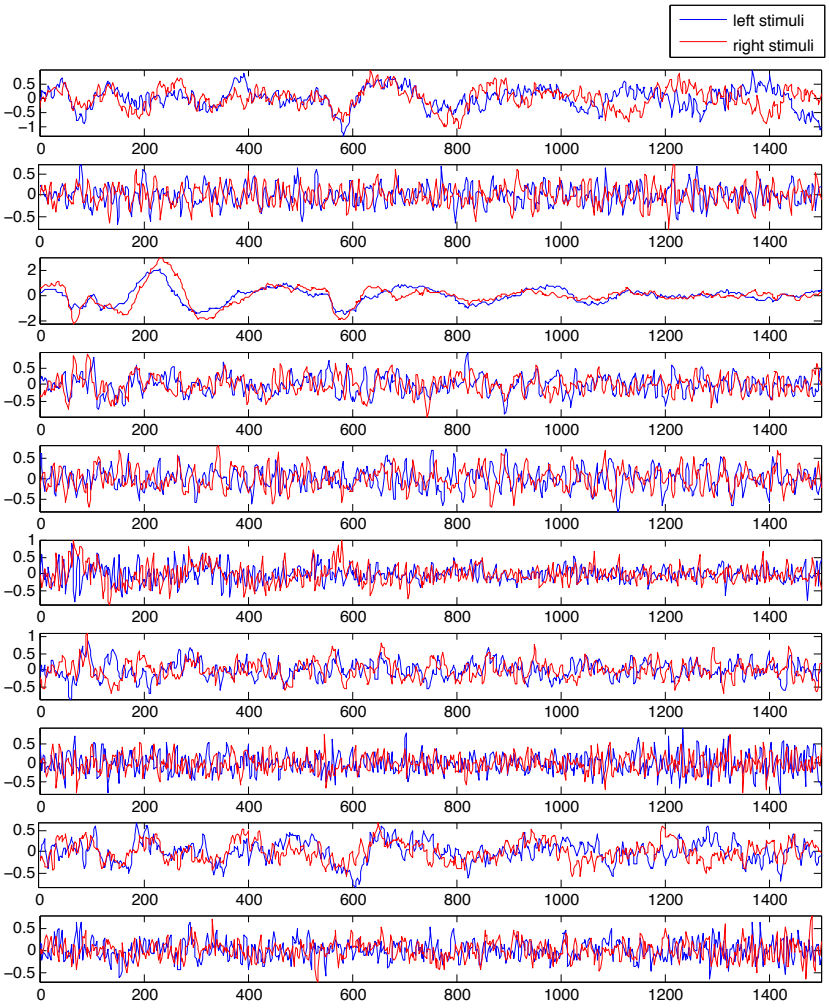


Figure D.7: Averaged Infomax ICA components for both left and right stimuli. Subject 5.

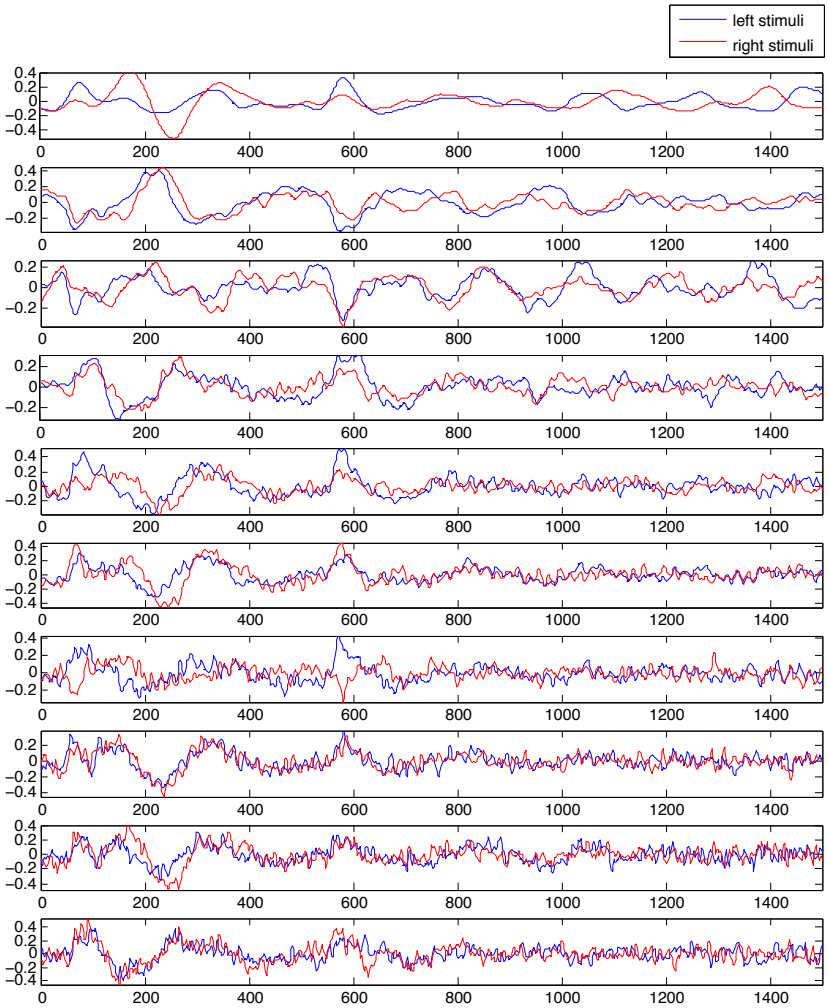


Figure D.8: Averaged normalised Kalman ICA components for both left and right stimuli. Subject 5.

Bibliography

[1] A. Aizerman, E.M. Braverman, and LI Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and remote control*, 25:821–837, 1964.

[2] S. Arnfred, A.C.N. Chen, D. Eder, B. Glenthųj, and R. Hemmingsen. Proprioceptive evoked potentials in man: cerebral responses to changing weight loads on the hand. *Neuroscience letters*, 288(2):111–114, 2000.

[3] S.M. Arnfred, L.K. Hansen, J. Parnas, and M. Mørup. Proprioceptive evoked gamma oscillations. *Brain research*, 1147:167–174, 2007.

[4] S.M. Arnfred, R.P. Hemmingsen, and J. Parnas. Delayed early proprioceptive information processing in schizophrenia. *The British Journal of Psychiatry*, 189(6):558–559, 2006.

[5] K.J. Åström. *Introduction to stochastic control theory*, volume 70. Elsevier Science, 1970.

[6] A.J. Bell and T.J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7(6):1129–1159, 1995.

[7] H. Berger. Über das elektrenkephalogramm des menschen. *European Archives of Psychiatry and Clinical Neuroscience*, 87(1):527–570, 1929.

[8] C.M. Bishop and SpringerLink (Service en ligne). *Pattern recognition and machine learning*, volume 4. Springer New York, 1st edition, 2006.

[9] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[10] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[11] S. Chiappa and D. Barber. EEG classification using generative independent component analysis. *Neurocomputing*, 69(7):769–777, 2006.

[12] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[13] F. Crick and C. Koch. Towards a neurobiological theory of consciousness. In *Seminars in the Neurosciences*, volume 2, page 203, 1990.

[14] A. Delorme and S. Makeig. EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. *Journal of neuroscience methods*, 134(1):9–21, 2004.

[15] A. Delorme and S. Makeig. Eeglab wikitorial, 2009.

[16] E. Fix and J.L. Hodges. Discriminatory analysis. Nonparametric discrimination: Consistency properties. *Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas*, 1951.

[17] T. Fletcher. Support vector machines explained. *Tutorial paper., Mar*, 2009.

[18] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Academic Pr, 1972.

[19] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–741, 1984.

[20] M.S. Grewal and A.P. Andrews. Kalman filtering: theory and practice using MATLAB. 2001.

[21] C. Guger, H. Ramoser, and G. Pfurtscheller. Real-time EEG analysis with subject-specific spatial patterns for a brain-computer interface (BCI). *Rehabilitation Engineering, IEEE Transactions on*, 8(4):447–456, 2000.

[22] Lars Kai Hansen. Course 02457 non-linear signal processing, exercise 9. 2007.

[23] PJ Hargrave. A tutorial introduction to Kalman filtering. In *Kalman Filters: Introduction, Applications and Future Developments, IEE Colloquium on*, pages 1–1. IET, 1989.

[24] J. Hartikainen and S. Särkkä. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *Proceedings of IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 379–384, 2010.

[25] Lise-Lotte Hergel. *Gyldendals Store Lægebog*. Nordisk Forlag, 2005.

[26] C.W. Hsu, C.C. Chang, C.J. Lin, et al. A practical guide to support vector classification, 2003.

[27] J.R. Hughes. Gamma, fast, and ultrafast waves of the brain: their relationships with epilepsy and behavior. *Epilepsy & Behavior*, 13(1):25–31, 2008.

[28] T.P. Jung, S. Makeig, C. Humphries, T.W. Lee, M.J. Mckeown, V. Iragui, and T.J. Sejnowski. Removing electroencephalographic artifacts by blind source separation. *Psychophysiology*, 37(02):163–178, 2000.

[29] R.E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[30] D. Langlois, S. Chartier, and D. Gosselin. An introduction to Independent Component Analysis: Infomax and FastICA algorithms. *Tutorials in Quantitative Methods for Psychology*, 6(1):31–38, 2010.

[31] D.J.C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge Univ Pr, 2003.

[32] S. Makeig, A.J. Bell, T.P. Jung, T.J. Sejnowski, et al. Independent Component Analysis of electroencephalographic data. *Advances in neural information processing systems*, pages 145–151, 1996.

[33] P.S. Maybeck. *Stochastic models, estimation and control*, volume 1. Academic Pr, 1979.

[34] M. Mørup, L.K. Hansen, S.M. Arnfred, L.H. Lim, and K.H. Madsen. Shift-invariant multilinear decomposition of neuroimaging data. *NeuroImage*, 42(4):1439–1450, 2008.

[35] C. Mulert, L. Jäger, R. Schmitt, P. Bussfeld, O. Pogarell, H.J. Möller, G. Juckel, and U. Hegerl. Integration of fMRI and simultaneous EEG: towards a comprehensive understanding of localization and time-course of brain activity in target detection. *Neuroimage*, 22(1):83–94, 2004.

[36] R.K. Olsson and L.K. Hansen. Linear state-space models for blind source separation. *The Journal of Machine Learning Research*, 7:2585–2602, 2006.

[37] A.H. Omidvarnia, F. Atry, S.K. Setarehdan, and B.N. Arabi. Kalman filter parameters as a new EEG feature vector for bci applications. In *Proceedings of the 13th European Signal Processing Conference Eusipco2005. Citeseer*, 2005.

[38] T. D. Stephens R. R. Seeley and P. Tate. *Anatomy and Physiology*. Mc Graw Hill, 7th edition, 2005.

[39] R.M. Rangayyan. *Biomedical signal analysis*. IEEE press, 2002.

[40] C.E. Rasmussen and CKI Williams. *Gaussian processes for machine learning*. The MIT Press, Cambridge, MA, USA, 2006.

[41] S. Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural computation*, 11(2):305–345, 1999.

[42] T.V. Schroeder. *Basisbog i medicin og kirurgi*. Munksgaard Danmark, 2005.

[43] G. Welch and G. Bishop. An introduction to the Kalman filter. *University of North Carolina at Chapel Hill, Chapel Hill, NC*, 7(1), 1995.

[44] www.BrainConnection.com. http://brainconnection.positscience.com/topics/?main=anat/motor-anat, retrieved 1st of april 2012.

[45] H. Zhang. The optimality of naive Bayes. *A A*, 1(2):3, 2004.