THESIS

ANALYSIS OF TEMPORAL STRUCTURE AND NORMALITY IN EEG DATA

Submitted by

Artem Sokolov

Department of Computer Science

In partial fulfillment of the requirements

for the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Spring 2007

ABSTRACT OF THESIS

ANALYSIS OF TEMPORAL STRUCTURE AND NORMALITY IN EEG DATA

The thesis examines normality and temporal correlation of samples in EEG data. Presented in the context of Bayesian classification, empirical results quantify the advantages of modeling the temporal information. A connection to the theoretical background of the underlying classifiers is also discussed. Using a five-task six-channel dataset, the experiments demonstrate that an increase in performance can be observed by simple consideration of multiple samples. Exploitation of temporal structure leads to additional improvement, but not always. The measurement of Normality is used to demonstrate an alternative to cross-validation, where each class is considered independently of the others.

Artem Sokolov
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
Spring 2007

Chapter 1

Introduction

1.1 Brain-Computer Interfaces

One science fiction idea that is proving to become realized is a concept of controlling a computer with one's brain, completely bypassing the nervous system. The idea is known as a brain-computer interface (BCI) and has a variety of potential applications. A BCI provides an alternative means of communication and control for patient's with Amyotrophic Lateral Sclerosis (ALS), a disease which leads to a shutdown of the patients nervous system causing the person to lose all voluntary muscle control. Interfacing one's brain with a computer can also augment additional functionality to daily operation of healthy persons.

BCI equipment and technology can be coarsely divided into two categories: intrusive and non-intrusive. The intrusive kind employs insertion of electrodes below the skull, often onto the surface of the brain [34, 55], but occasionally deep within the cortex for purposes of monitoring one particular region [19, 55]. Non-intrusive methods generally involve measurements of electro-magnetic potentials from outside the head. This thesis is concerned with electroencephalography (EEG), which is a non-intrusive method of recording voltage measurements from the scalp surface.

Although the BCI field has recently seen a large increase in the use of intrusive methods, EEG classification still remains a relevant problem. As the dominant non-intrusive method of today, electroencephalography offers by far the most practicality than any other BCI method, making it appealing for use by healthy persons in military and space agencies. It is the safest, least expensive and easiest to use. The fact that technology necessary for capturing EEG signals can be equipped and unequipped outside of a laboratory within minutes inspires control augmentation in a wide range of environments. For example, NASA is interested in the use of EEG as control signals for aircraft and vehicles [53].

Other advantages of non-intrusive methods include the ability to monitor large portions of the brain, allowing to observe higher-level cognitive processes. Unfortunately, reception of mixed EEG signals from different parts of the brain is just as good at thwarting the classification effort as it is at providing an overall picture of brain activity. The signals from interesting (and potentially discriminative) regions of the brain are mixed in a non-trivial way with each other as well as time-lagged versions of themselves. The problem is further complicated by the fact that electrical signals travel through the brain tissue at a different speed than they do through the skull or the scalp, an issue that doesn't arise with other non-intrusive methods, such as magnetoencephalography (MEG) and functional magnetic resonance imaging (fMRI). These factors make EEG classification comparable to standing outside of a large hall and trying to understand what a crowd of people inside is talking about. In spite of this difficulty, the potential offered by EEG classification makes the problem interesting and worthwhile.

1.2 Electroencephalographic signals

An example of EEG, taken from the Keirn dataset [27], is presented in Figure 1.1. The subject was instructed to perform two mental tasks: to multiply two non-trivial numbers and to imagine writing a letter. The EEG was recorded at 250Hz and presented in the figure are two seconds from each task. The six channels were positioned according

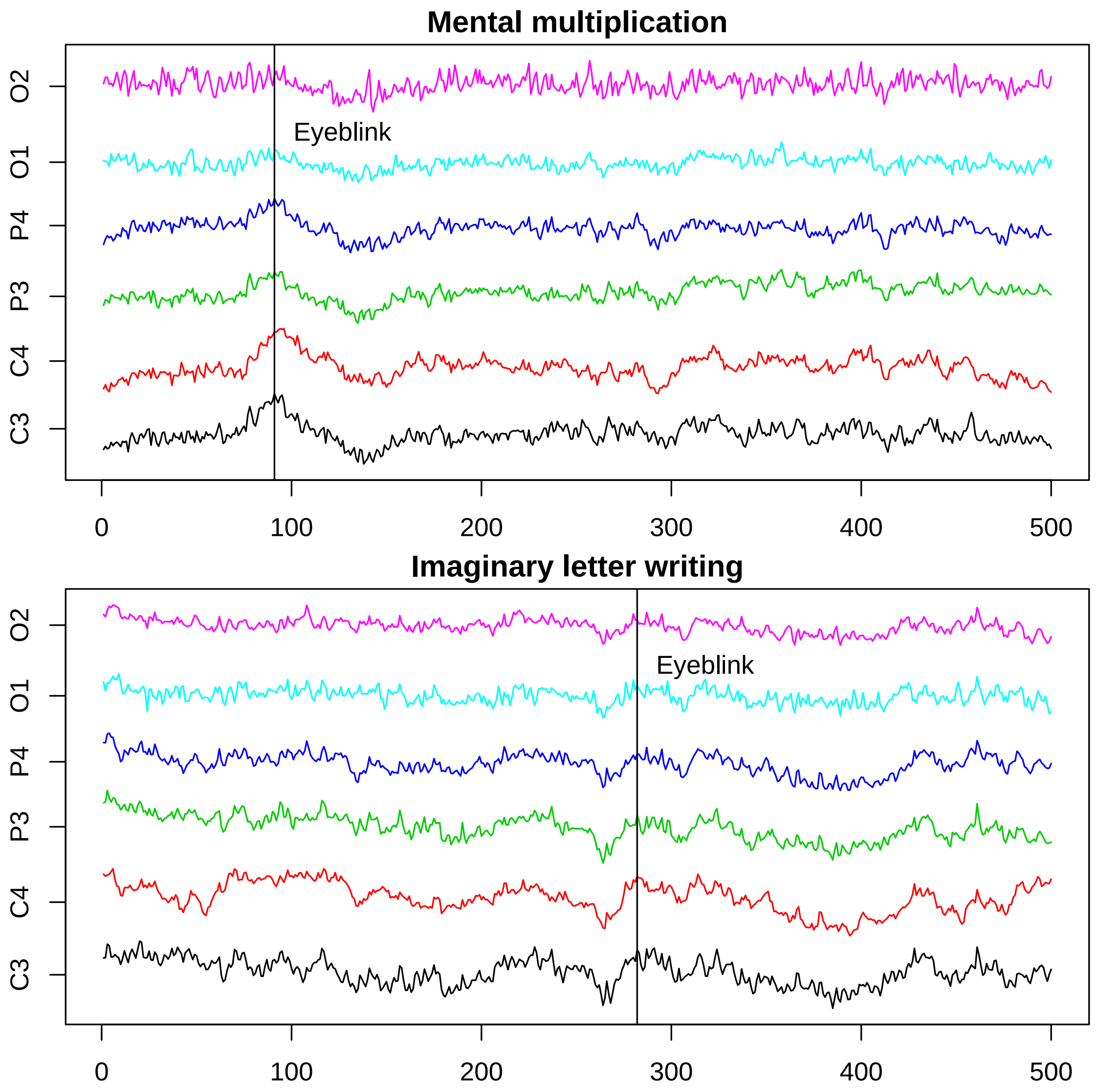

Figure 1.1: Two seconds of EEG recorded at 250 samples per second. The subject was mentally multiplying two numbers (above) and imagining writing a letter (below). Black vertical lines denote places where an eyeblink was detected.

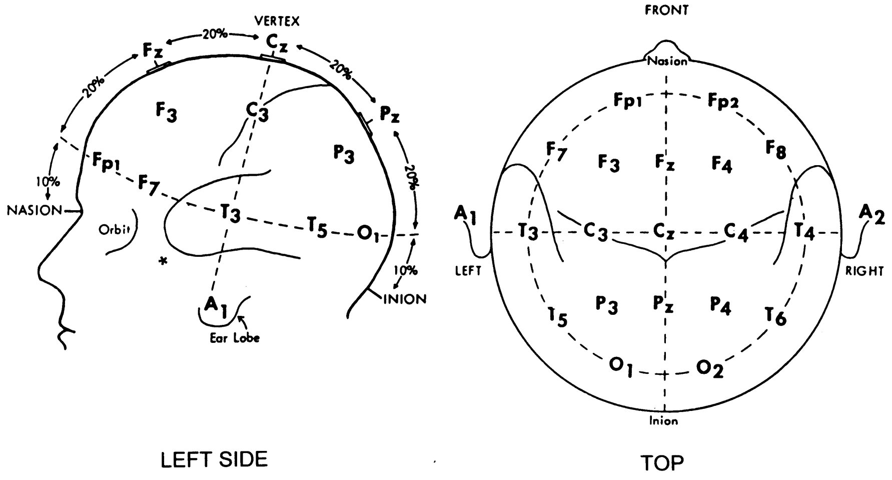

Figure 1.2: Placement of electrodes on the scalp according to the international 10-20 standard [25, 30].

to the international 10-20 standard, as displayed in Figure 1.2. Figure 1.1 also demonstrates that EEG is often contaminated by such artifacts as eyeblinks. Artifacts introduce additional noise and further complicate the problem of mental task classification.

Cognitive tasks such as multiplication and letter writing present one paradigm of mental tasks. Another paradigm of mental tasks involves imaginary movements of limbs. The imaginary movement of limbs appears in control applications and is often reported by subjects as being more intuitive for such tasks as control of a cursor, wheelchair, or robotic appendages [17]. It is also common to find combinations of imaginary movement with the other cognitive tasks, such as generation of words that begin with the same letter [17, 12]. The motivation behind such combinations arises from the fact that the evoked potential can be picked up from areas of the scalp that correspond to the lobe responsible for a particular cognitive task.

The choice of a dataset for this thesis has been primarily influenced by its previous use in BCI research at Colorado State University [5, 2, 4, 29, 3]. Analysis of temporal structure and normality can be viewed as providing an orthogonal view to this dataset, which complements the previous work. The analysis performed in this thesis can be easily carried over to other datasets.

1.3 BCI Evaluation

Performance of a BCI system is measured in several different ways. The most widespread measurement of success is *classification rate* (also known as *classification accuracy*), which is computed as the number of labels assigned correctly divided by the total number of label assignments. The best classification rate is, of course, 1 (or 100% if measured in percent). The worst classification rate is 1 over the number of classes (e.g, $1/2 = 50\%$ for two classes, $1/4 = 25\%$ for four classes, etc.) and is termed as *random classification rate*. If a classifier has accuracy lower than random, we can always construct a new classifier that takes the original output and inverts it by randomly picking a label that is different from it, thus increasing the expected accuracy above random.

The problem with using classification rate in the context of EEG is that the measure does not account for how often a decision is being made. From the application's point of view, a system that has 80% classification rate but assigns a label to each sample has by far higher utility than a system that has 90% classification rate but assigns a single label to a 10-second trial. An alternative measure of success that gets around this problem is *bitrate*. The bitrate measures the amount of information produced by the classification decision per second. The intuition behind bitrate arises from information theory and the actual measure can be computed as [54]:

$$V \left(\log_2 N + P \log_2 P + (1 - P) \log_2 \frac{1 - P}{N - 1} \right) \tag{1.1}$$

where V is the classification speed (i.e., the number of samples assigned per second), P is the classification rate, and N is the number of classes. The lowest bitrate is 0, which occurs when the classification accuracy is random, i.e., $P = 1/N$. On the other hand, the perfect classification $P = 1$ leads to the bitrate of $V \log_2 N$ which is the highest bitrate achievable for a given value of speed V.

This thesis uses classification rate as a measure of performance. We do, however, put an upper bound on the number of samples considered in every decision. This bound guarantees that a new label will be provided at least as often as 1 label/second. A BCI that makes a decision every second can be utilized to perform many basic tasks, such as controlling a cursor on the screen, and therefore presents a realistic system. Any real-time tasks (such as controlling a wheelchair), however, may require faster classification speeds. Classification speeds are stated for every experiment and together with classification accuracies allow one to calculate the corresponding bitrates, by simply using Equation (1.1).

1.4 State-of-the-art

The information presented at the BCI Meeting 2005 [1] suggests that the current state-of-the-art in the field has already progressed past the simple proof-of-concept experiments with many groups being able to successfully perform very basic control and communication experiments. The BCI technology is, nevertheless, still in its early stages and current directions of research involve design of better signal acquisition and signal processing methods, engineering faster and more robust hardware and software, and forming standards for interfaces between different modules in a BCI system [14, 32, 37, 55]. Chapter 2 goes into more detail on previous approaches to signal processing and data analysis in the context of EEG classification.

1.5 Objective

The objective of this thesis is to quantify two aspects of EEG data, temporal correlation and normality, and examine how they relate to classification accuracy and model selection, respectively. As described in the next chapter, most of the previous approaches to EEG classification consider multiple samples to make a single classification decision. This thesis examines how the classification rate changes when multiple samples are considered with the experiments aimed at determining whether the change in classification rate is due to temporal correlation of samples or due to there simply being more information available. Understanding how much temporal structure affects classification provides an important contribution to researches that design feature extraction and classification methods that consider multiple adjacent samples to make a classification decision.

This thesis also demonstrates how analysis of normality can be used to perform model selection. Models that assume Normality of the data are not uncommon in the literature and analysis of normality allows to determine the spatial distribution of the data in a way that is consistent with the model. Analysis of normality has an additional advantage in that it only requires data from a single class to perform model selection.

All experiments in this thesis consider raw EEG data. No feature extraction is performed because data transformation methods may take advantage of temporal structure introducing a bias into our evaluation. Instead, multiple samples are considered through time-lag embedding [43] and windowing [52]. Time-lag embedding allows us to consider multiple adjacent samples, evaluating *local temporal structure*. In addition, we also consider *global temporal structure* by modeling time-lag samples with a first-order Markov process. Analysis of normality is done by measuring Gaussian fit based on the empirical distribution function.

Presented empirical evidence suggests that a high level of accuracy can be achieved by simply considering multiple samples in time in the classification decision. Temporal structure of EEG yields additional increase in the classification rate, but not always. The best results are achieved when the time-lag embedding and windowing are combined.

1.6 Overview of Remaining Chapters

The remainder of this report is structured as follows. Chapter 2 describes previous work in the area of EEG classification, noting in particular the modeling techniques that rely on temporal structure of the data. Chapter 3 goes into the theoretical details of methods to be used. Description of the data is located in Chapter 4. Chapter 5 presents empirical results on real EEG data, followed by detailed discussion of said results. Concluding remarks can be found in Chapter 6.

Chapter 2

Background

This chapter extends the previous one by describing what approaches to EEG classification have been previously explored. Classification of EEG signals has been done for purposes other than brain-computer interface. A canonical example is learning to automatically distinguish between healthy subjects and, e.g., patients with Alzheimer disease [48, 28]. Because the context of this work is a BCI, only classification of EEG related to mental tasks is addressed in this chapter.

In general, EEG classification of mental tasks has been performed at two levels of granularity. First, there is single-trial classification where observation sequences (or trials) are recorded over some fixed period of time (e.g., 10 seconds) with the subject performing the same mental task during that period. A classifier is then free to analyze the whole sequence and must assign a single class label to the entire trial. Single-trial classification is very popular with offline analysis. On the other hand, a more realistic online BCI should allow the subject to make their own decision about when to switch to a different mental task. Thus, classification must be performed on a continuous stream of observations. Ideally, a classifier should be able to provide a decision after every new sample, but to gain temporal information we may allow the decision to be lagged by a few samples. This leads to the introduction of sliding time windows [52, 9]. Sliding time windows can be viewed as a way to limit a classification decision to a specific set of samples in the time series, and are discussed in more detail in Chapter 3.

Any approach to a classification problem consists of two major parts: extracting a set of features from the original data and applying a classification algorithm to learn and apply a mapping from these features to a set of labels. As a consequence, most of the work in EEG classification addresses one or both of these two major parts by either presenting a new algorithm, evaluating a well-established algorithm on the problem of EEG classification, or attempting to understand what works best and why. This thesis is related the closest to the last category because of its focus on measuring aspects of EEG data.

2.1 Feature Extraction

Not surprisingly, almost all feature extraction methods used with EEG consider multiple samples at a time. The two most popular approaches to feature extraction are auto-regression (AR) [42] and transformation into a frequency space (either by means of a Fourier transform or wavelet analysis) [36]. Both approaches make use of multiple raw samples to construct a single feature. Auto-regression models each sample in a time series as a (possibly non-linear) function of its predecessors in time. Coefficients of this function constitute the features. Frequency transformation estimates the frequency decomposition of a signal within some local time window. The corresponding features are composed of energy measurements in a set of frequency bands. Sometimes, frequency transforms are utilized only as a temporal filter (i.e., band-passing the original signal) with the actual feature extraction being performed using some other method.

In addition to band-pass filters, spatial filters are also sometimes employed, the most common two being principal components analysis (PCA) [26] and common spatial patterns (CSP) [38]. Spatial filters use multiple samples only for the purpose of computing covariance matrices between the channels. The output of a spatial filter is a projection to a lower-dimensional subspace with the original dimensionality being determined by the number of channels.

2.2 Classification

Classification algorithms fall into two paradigms: discriminative and generative. Generative classifiers create a model for each class and classify new samples according to the highest likelihood via the Bayes' rule. Discriminative classifiers learn a rule that maps samples to their labels without modeling the data from each class directly. In the context of EEG classification, generative approaches are almost exclusively limited to hidden Markov models (HMMs) with various topologies and extensions. HMMs typically model the data in each class with a finite number of Gaussian distributions and assume a first order Markov process over which Gaussians produce the samples in the series. The discriminative approaches used for EEG classification include neural networks (NN) and support vector machines (SVM).

2.3 Review of Recent Literature

Examining specific papers, we find that authors usually select a number of the methods described above and create a (filter data $\rightarrow$ extract features $\rightarrow$ classify) chain. These chains range in complexity but most authors reach as high as 85-90% classification accuracy. Examples from the recent literature are summarized in the following paragraph.

Several papers are worth noting in the discriminative classification paradigm. Xu, *et al.* [56] apply spatial and band-pass filters to raw EEG data, use radial basis functions (RBF) as features and perform classification of self-paced finger tapping using support vector machines, reaching 90% classification accuracy. Their work demonstrates that combining PCA and CSP to perform spatial filtering yields higher results than using either method separately. Garcia, *et al.* [20] combined frequency spectrum features and support vector machines to classify EEG from imaginary hand movement with 72-88% classification accuracy. The authors showed SVM outperforming Linear Discriminant Analysis, a simpler classifier that was applied to the same set of features. Haselstiner and Pfurtscheller performed a comparative study between the traditional multilayer perceptron (MLP) and a neural network that uses finite impulse response (FIR) units, achieving as high as 95% accuracy with one of their subjects [23]. Their work shows FIR-based network outperforming the more traditional MLP. Other variants of neural networks considered for EEG classification include Learning Vector Quantization [44, 33].

As already mentioned, hidden Markov models are the prevalent technique for EEG classification in the generative paradigm, although simpler modeling techniques such as Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) have also been applied [4]. HMMs have been considered for single-trial classification [41, 57] as well as part of an asynchronous system [13], including communication via a "virtual keyboard" [40]. The HMM topologies considered include coupled HMMs, multivariate HMMs, factorial HMMs and input-output HMMs with more complex topologies generally yielding better performance, occasionally as high as 90% classification accuracy [57, 13].

Among the more complex Markov models, an HMM with combined Gaussian and multinomial observation distributions has been applied to classify coefficients of an AR process extracted from EEG [50]. The authors used Markov Chain Monte Carlo posterior approximation for single-trial classification of left-vs.-right motor and auditory-vs.-navigation mental tasks, reaching 75-91% classificatio accuracy. Sykacek, *et al.* [51] later generalized the above approach by using a single joint density over all parameters involved in a model, obtaining 0.31-0.38 bits/second bitrate, up from 0.20-0.21 bits/second bitrate that results from separation of signal processing and classification.

An interesting combination of HMMs with an SVM classifier has been proposed by Lee and Choi [35]. The authors train a separate HMM for each channel/class pair and use output likelihoods as features for an SVM-based classifier. The classification accuracy they achieved was around 75% on a two-class problem.

2.4 Limitations of Current Literature and Contribution of Thesis

Today, there exist BCI systems that achieve classification accuracy close to 100% on a 2-5 task problem, although such high accuracy holds only for a specific subject or dataset [5, 10, 11]. The success of such systems is usually attributed to high density signal acquisition, advanced filtering, and hand-engineered machine learning approaches (hand-picking feature extraction and classification methods from the ones mentioned so far). The problem is, of course, that the success doesn't carry over from subject to subject and the same approach that gives 98% accuracy on EEG from one subject yields only 78% success on EEG from a different subject. Success of any BCI system is heavily dependent on the subject using the system, a fact that makes comparative assessment of performance somewhat difficult. This thesis demonstrates that a high level ($> 90\%$ on a five class problem) of classification accuracy can be obtained with very simple classifiers when multiple samples are considered to make a single classification decision. Realizing that this performance is dataset-dependent, we instead focus on understanding how multiple samples affect the classification rate. The nature of this thesis is, therefore, more investigative than comparative, although empirical comparisons are inherently present.

Previous analysis of spatial and temporal structure in EEG has lead to a number of methods in artifact removal [39, 31], spatial filtering [46], prediction [15], and isolation of features contributing to correct classification [9, 29]. The analysis and quantification of temporal structure and normality performed in this thesis complements the knowledge and understanding of EEG data and suggests how much improvement can be expected from multiple sample consideration. From a practical point of view, this allows to put an upper bound on the temporal window width exceeding which leads to a tiny increase in classification accuracy. On the other hand, such an upper bound leads to a better expectation of how fast the classification decisions can be made.

Chapter 3

Methods

3.1 Overview

This chapter describes the theoretical background of methods used in this thesis. The discussion starts off by describing methods used for analyzing local temporal structure. These include time-delay embedding and time windowing. This discussion is followed by an overview of spatial projection methods. All analysis in this thesis is done in the context of Bayesian classification, which is discussed after the spatial projection method. Individual Bayesian classifiers are described in order of growing complexity. Analysis of local temporal structure employs Linear Discriminant Analysis and Quadratic Discriminant Analysis, two very simple Bayesian classifiers. Global temporal structure analysis requires more complex models, Gaussian Mixture Models and Hidden Markov Models. The chapter is concluded with the description of how analysis of normality is performed.

3.2 Local Temporal Structure

3.2.1 Time-delay Embedding

Time-delay (sometimes also called time-lag) embedding [43] is used here as the primary method for analyzing the temporal structure in EEG data. The method effectively converts temporal structure to spatial feature representation. This allows one to consider multiple neighboring samples as a single observation in a high dimensional space.

Formally, time-delay embedding or lagging is defined as follows. Let X be a n by d matrix of n samples in d dimensions. Applying $k > 0$ lags results in a $n-k$ by $kd+d$ matrix Y_k, where the first d columns consist of samples 1 through $n-k$ from X, the second d columns consist of samples 2 through $n-k+1$ from X, and so on, with the last d columns of Y_k consisting of samples k through n from X. Here's an example:

$$X = \begin{bmatrix} 1.1 & 2.4 \\ 3.5 & 6.3 \\ 2.7 & 2.6 \\ 9.3 & 4.6 \end{bmatrix} \quad Y_1 = \begin{bmatrix} 1.1 & 2.4 & 3.5 & 6.3 \\ 3.5 & 6.3 & 2.7 & 2.6 \\ 2.7 & 2.6 & 9.3 & 4.6 \end{bmatrix}$$

$$Y_2 = \begin{bmatrix} 1.1 & 2.4 & 3.5 & 6.3 & 2.7 & 2.6 \\ 3.5 & 6.3 & 2.7 & 2.6 & 9.3 & 4.6 \end{bmatrix}$$

Because a large part of this thesis deals with normality of EEG data, we need to establish whether normality is preserved when the data is lagged.

Proposition 1. *If column vectors x_1 and x_2 are two independent samples from a multivariate Gaussian distribution with mean μ and covariance matrix Σ, then $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ has multivariate Gaussian distribution with mean $\begin{bmatrix} \mu \\ \mu \end{bmatrix}$ and covariance matrix $\begin{bmatrix} \Sigma & 0 \\ 0 & \Sigma \end{bmatrix}$.*

Proof. The distribution of d-dimensional samples x_1 and x_2 is defined by the density function

$$f(x|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right). \tag{3.1}$$

Using the fact that the two samples are independent we can compute their joint probability density as the product of the marginal densities:

$$f(x_1, x_2|\mu, \Sigma) = f(x_1|\mu, \Sigma)f(x_2|\mu, \Sigma) =$$

$$= \frac{1}{(2\pi)^d|\Sigma|} \exp\left(-\frac{1}{2}(x_1-\mu)^T\Sigma^{-1}(x_1-\mu) - \frac{1}{2}(x_2-\mu)^T\Sigma^{-1}(x_2-\mu)\right).$$

We can rewrite the exponent in matrix form as follows:

$$-\frac{1}{2}(x_1-\mu)^T\Sigma^{-1}(x_1-\mu) - \frac{1}{2}(x_2-\mu)^T\Sigma^{-1}(x_2-\mu) =$$

$$= -\frac{1}{2}\left[\begin{array}{cc} (x_1-\mu)^T\Sigma^{-1} & (x_2-\mu)^T\Sigma^{-1} \end{array}\right]\left[\begin{array}{c} x_1-\mu \\ x_2-\mu \end{array}\right]$$

$$= -\frac{1}{2}\left[\begin{array}{cc} (x_1-\mu)^T & (x_2-\mu)^T \end{array}\right]\left[\begin{array}{cc} \Sigma^{-1} & 0 \\ 0 & \Sigma^{-1} \end{array}\right]\left[\begin{array}{c} x_1-\mu \\ x_2-\mu \end{array}\right]$$

$$= -\frac{1}{2}\left(\left[\begin{array}{c} x_1 \\ x_2 \end{array}\right] - \left[\begin{array}{c} \mu \\ \mu \end{array}\right]\right)^T\left[\begin{array}{cc} \Sigma^{-1} & 0 \\ 0 & \Sigma^{-1} \end{array}\right]\left(\left[\begin{array}{c} x_1 \\ x_2 \end{array}\right] - \left[\begin{array}{c} \mu \\ \mu \end{array}\right]\right).$$

Finally, by noting that

$$|\Sigma| = \left(|\Sigma|^2\right)^{1/2} = \left|\begin{array}{cc} \Sigma & 0 \\ 0 & \Sigma \end{array}\right|^{1/2} \quad \text{and} \quad \left[\begin{array}{cc} \Sigma^{-1} & 0 \\ 0 & \Sigma^{-1} \end{array}\right] = \left[\begin{array}{cc} \Sigma & 0 \\ 0 & \Sigma \end{array}\right]^{-1},$$

we can express the joint distribution of $\bar{x} = \left[\begin{array}{c} x_1 \\ x_2 \end{array}\right]$ as

$$f\left(\bar{x}|\bar{\mu},\bar{\Sigma}\right) = \frac{1}{(2\pi)^{2d/2}|\bar{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}\left(\bar{x}-\bar{\mu}\right)^T\bar{\Sigma}^{-1}\left(\bar{x}-\bar{\mu}\right)\right).$$

We recognize this as the density function of a multivariate Gaussian distribution with mean $\bar{\mu} = \left[\begin{array}{c} \mu \\ \mu \end{array}\right]$ and covariance matrix $\bar{\Sigma} = \left[\begin{array}{cc} \Sigma & 0 \\ 0 & \Sigma \end{array}\right]$. $\square$

Although this result is derived for the number of lags $k = 1$, it can be easily extended to any $k > 0$ through simple mathematical induction. If the samples x_1 and x_2 are not independent, the joint density is no longer the product of marginal densities but rather depends on the conditional distribution $f(x_2|x_1,\mu,\Sigma)$:

$$f(x_1,x_2|\mu,\Sigma) = f(x_1|\mu,\Sigma)f(x_2|x_1,\mu,\Sigma)$$

If the conditional density is Normal, a calculation similar to the one above can be used to demonstrate preservation of normality. However, the normality of $f(x_2|x_1,\mu,\Sigma)$ is hard to verify and generally doesn't hold.

3.2.2 Time Windowing

Time windows were first proposed by Takens [52] and have previously been applied in the context of EEG by Blankertz, *et al.* [9]. Implicitly time windows are used by time-lag embedding and any other method that attempts to capture local temporal information. A time window refers to a set of consecutive samples in a time-series and is parameterized by two indices, that of the first and that of the last samples in the window (t_f and $t_l > t_f$), and spacing of samples δ. Techniques, such as time-lag embedding, take a time window as the input and produce a set of features associated with the collection of samples within that time window. Most algorithms that employ time windowing consider their first window by setting $t_f = 1$ (i.e., the first sample in the time series) and $t_l = t_f + w_l$ using some $w_l > 0$. All other windows considered by the algorithm are constructed by adding some $w_s > 0$ to t_f and t_l of the previous window. This is referred to as *sliding* time windowing and is completely specified by w_l and w_s. The spacing of samples δ is usually kept at 1 (as it is here) implying that every sample in a time window gets considered.

Analyzing temporal structure is not the only place where time windowing is a technique of interest. A hidden Markov model, one of the methods that will be employed for global structure analysis, considers transitions from one sample to the next and, therefore, requires a collection of multiple samples to make a classification decision. Rather than considering an entire trial (one extreme of the spectrum, where window size is equal to the trial length), the trial will be broken up into time windows and the results will be reported based on classification of individual windows.

3.3 Spatial Projection Methods

Due to the extraction of local temporal structure, we end up working with high-dimensional features which can lead to problems when the samples do not span all the dimensions.

This calls for projection of high-dimensional data to a lower dimensional subspace. Principal Component Analysis and Fisher's Projection are used here as two methods for performing such projection.

3.3.1 Principal Component Analysis

Principal Component Analysis (PCA) finds directions of most variance in the data by computing Singular Value Decomposition (SVD) on the mean-centered data matrix [26]. More specifically, if X is a n by d matrix of n mean-centered samples in d dimensions, then SVD of X yields

$$X = UDV^T,$$

where U and V^T are orthonormal bases for columns and rows of X, respectively. Since our samples are arranged as rows of X, the matrix V^T is of most interest. Entries of the diagonal matrix D are equal to the amount of variance in samples of X along the corresponding rows of V^T (or columns of V). Let k be the number of non-zero diagonal entries in D. Then the number of dimensions that X spans is equal to k and we can project the data to the corresponding subspace by taking $V^{(1:k)}$ to be the first k columns of V and computing $XV^{(1:k)}$.

3.3.2 Fisher's Projection

Fisher's projection, also known as reduced rank linear discriminant analysis (LDA), attempts to minimize within-class variance and maximize inter-class variance [24]. The solution to this problem results in a linear transformation that projects the original data into a subspace of maximal separation of class centroids, and minimal intra-class variance. Hastie, *et al.* outline the computations necessary for reduced rank LDA [24]:

1. Suppose there are K classes of data samples in d dimensions.

2. Compute M, the K by d matrix of class centroids.

3. Compute W, the common covariance matrix.

4. $M^* = MW^{-1/2}$.

5. Compute B^*, the covariance matrix of M^*.

6. Compute SVD of B^*: $B^* = V^* D_B V^{*T}$.

7. The first $K-1$ columns of V^* define the linear transformation that yields maximal separation of class centroids.

This computation can be seen as performing PCA on the whitened centroid matrix. The resulting columns of V^* are orthonormal in the whitened space, but not necessarily in the original data space. Note that the common covariance matrix must be non-singular, which implies that Fisher's Projection can only be applied to full rank data. Therefore, it might be necessary to perform PCA on the data prior to computing Fisher's Projection.

There is an alternative statistical perspective on Fisher's Projection which I discuss below.

3.4 Bayesian Classification

When each mental task is represented through a stochastic generative model, we can make use of a Bayesian framework to classify new data. Bayesian classification refers to constructing a probabilistic model for each class separately and assigning each new observation to the class with maximum posterior probability [22, 24, 7]. More specifically, assume we have K classes with labels l_1, l_2, ..., l_K. We propose that data in each class comes from some probabilistic model with parameters θ_k for class k. A training procedure is then used to estimate θ's and each new observation x is assigned label l_j with

$$j = \underset{k=1,2,\ldots,K}{\arg\max}\, P(L = l_k | x, \theta) \tag{3.2}$$

where L is a random variable that takes on value l_k when we get an observation from class k, and $\theta = (\theta_1, \theta_2, ..., \theta_K)$.

Posterior probabilities $P(L = l_k|x, \theta)$ of observation x belonging to class k can rarely be computed directly. The most common approach is to use Bayes' theorem [22], which arises directly from applying the definition of conditional probability twice:

$$P(L = l_k|x, \theta) = \frac{P(L = l_k, x|\theta)}{P(x|\theta)} = \frac{P(x|L = l_k, \theta)P(L = l_k|\theta)}{P(x|\theta)} = \frac{p_k P(x|\theta_k)}{P(x|\theta)},$$

where $P(x|L = l_k, \theta) = P(x|\theta_k)$ is readily available from the probabilistic model of class k. Priors $p_k = P(L = l_k|\theta) = P(L = l_k)$ are independent of model parameters and can be used to introduce domain knowledge regarding what classes are more likely to occur (under stochastic constraint $\sum_k p_k = 1$). The denominator $P(x|\theta)$ is nothing more than a normalization factor and can be interpreted as the probability that the observation occurs as a member of one of the classes, or mathematically $P(x|\theta) = \sum_{k=1}^{K} p_k P(x|\theta_k)$. Note that when the prior distribution across the classes is uniform, the same class will have the maximum posterior and class-conditional probabilities, i.e., $\arg\max_k P(L = l_k|x, \theta) = \arg\max_k P(x|\theta_k)$.

3.5 Modeling as a Single Gaussian

3.5.1 Linear Discriminant Analysis

The simplest approach in stochastic modeling is to model each class with a single Gaussian, using one common covariance matrix across all classes. The density of the data is then computed according to Equation (3.1) with the means for each class and the common covariance matrix estimated from the training data as

$$\mu_k = \sum_i x_i^{(k)} / n_k,$$

$$\Sigma = \sum_{k=1}^{K} \sum_{i} (x_i^{(k)} - \mu_k)(x_i^{(k)} - \mu_k)^T / (n - K),$$

where $x_i^{(k)}$ is used to denote the i^{th} sample from class k, with K being the total number of classes. The quantities n_k and n are used to denote the number of samples in class k and the total number of samples, respectively.

This approach is known as Linear Discriminant Analysis (LDA). After application of Bayes' theorem, the computation of Equation (3.2) can be reduced to comparing K linear discriminant functions [24] with the k^{th} function defined as

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log p_k.$$

The data sample x is assigned the label corresponding to the largest discriminant function value. The priors for Bayes' theorem are usually taken to be $p_k = n_k/n$.

The intersection of two linear discriminant functions is a hyperplane, with a sample being classified based on which side of the hyperplane it falls. This is equivalent to considering the closest class centroid in the whitened space which allows us to build a link to Fisher projection. When working with K class centroids, the problem of finding the closest centroid has up to $K - 1$ degrees of freedom (because K points can span at most $K - 1$ dimensions). This is why classification based on the closest centroid performed in the $K - 1$ dimensional space found by Fisher projection will yield exactly the same results as in the unprojected but whitened space.

3.5.2 Quadratic Discriminant Analysis

Quadratic Discriminant Analysis (QDA) removes the common covariance matrix constraint. The density is still computed using Equation (3.1) but the covariance matrix, Σ_k, for each class k is now estimated using only samples from that class as

$$\Sigma_k = \sum_{i} (x_i^{(k)} - \mu_k)(x_i^{(k)} - \mu_k)^T / (n - 1).$$

As a result of each class having its own covariance matrix, less cancellation can be performed in the derivation of discriminant functions [24]. The resulting discriminant functions are now quadratic in x and defined by

$$\delta_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k) + \log p_k.$$

3.6 Modeling with Latent Variables

To model transitions between neighboring samples, we turn to hidden Markov models (HMMs). HMMs fall into the category of modeling with latent variables. In addition to observations X and model parameters θ, we now assume the presence of an additional entity, the hidden (or latent) variables Z. The nature of hidden variables is model dependent but allows one to reason in terms of the joint probability distribution $P(X, Z|\theta)$. Given a model with latent variables, there are now two problems that need to be addressed: inference and learning.

3.6.1 Inference

The problem of inference is to compute $P(Z|X, \theta)$, i.e., the probability of hidden variables values given an observation sequence. Being able to compute this quantity allows one to answer the question of what is the most likely sequence of hidden variable values, i.e., find $\arg\max_Z P(Z|X, \theta)$.

Another related quantity of interest is the total likelihood of an observation sequence $P(X|\theta)$. The two quantities are related through:

$$P(Z|X,\theta) = \frac{P(Z,X|\theta)}{P(X|\theta)} = \frac{P(Z,X|\theta)}{\int_Z P(Z,X|\theta)dZ} \tag{3.3}$$

In general, the joint distribution $P(Z, X|\theta)$ is available from the model and the bulk of the work is in computing the integral.

3.6.2 Learning via Expectation Maximization

A very powerful technique for solving the problem of maximum likelihood learning is the Expectation Maximization (EM) algorithm [18, 8, 24]. The algorithm is naturally suited for cases where there are either missing observations or latent variables in the model.

In learning, we're interested in finding $\arg\max_\theta L(\theta|X)$, or parameters that yield the highest likelihood of our model given the data. By definition, a likelihood function is simply a reparameterization of the corresponding probability function:

$$L(\theta|X) = P(X|\theta),$$

where likelihood is considered to be a function of the model given the data, rather than a function of the data given the model (likelihood and probability have identical form with the only difference being in whether it is the data or the model that is treated as a variable).

Rather than working with the likelihood directly, consider the log-likelihood function $\mathscr{L}(\theta) = \log L(\theta|X) = logP(X|\theta)$. The $\log$ function is introduced for several reasons. First, it is a monotonically increasing function. Thus, maximizing log-likelihood is equivalent to maximizing the likelihood itself. Second, logarithmic transform turns all exponentials to products and all products to sums, significantly simplifying all calculations. The third reason for considering log-likelihood is machine precision. Probabilistic modeling deals with quantities that can be very tiny and at the same time differ by several orders of magnitude, a recipe for round-off errors. Logarithmic transform alleviates this problem by "compressing" a difference in orders of magnitude to the difference by a factor.

To understand EM algorithm, first recall that the likelihood of the data is a marginal distribution of the joint likelihood $P(Z, X|\theta)$. Marginal distributions are obtained by integrating out all variables except those of interest. Therefore, $\mathscr{L}(\theta) = \log P(X|\theta) = \log \int_Z P(Z, X|\theta) dZ$.

The EM algorithm makes use of Jensen's inequality to compute the lower bound on $\mathscr{L}(\theta)$. Jensen's inequality says that for any concave function g and random variable Z, inequality $E[g(Z)] \leq g(E[Z])$ holds. To apply this concept, we first have to turn the integral $\int_Z P(Z, X|\theta) dZ$ into an expectation. We do this by introducing some arbitrary distribution Q over the hidden variables:

$$\log \int_Z P(Z, X|\theta) dZ = \log \int_Z Q(Z) \frac{P(Z, X|\theta)}{Q(Z)} dZ. \tag{3.4}$$

Notice now that the right-hand side is logarithm of an expectation (the expectation is with respect to distribution $Q(Z)$). $\log$ is a concave function and we can now apply Jensen's inequality:

$$\mathscr{L}(\theta) = \log \int_Z Q(Z) \frac{P(Z, X|\theta)}{Q(Z)} dZ \geq \int_Z Q(Z) \log \frac{P(Z, X|\theta)}{Q(Z)} dZ \triangleq \mathscr{F}(Q, \theta). \tag{3.5}$$

The right-hand side of the inequality can be further separated into two parts: one that depends on Q and θ, and one that depends only on Q:

$$\int_Z Q(Z) \log \frac{P(Z, X|\theta)}{Q(Z)} dZ = \int_Z Q(Z) \log P(Z, X|\theta) dZ - \int_Z Q(Z) \log Q(Z) dZ. \tag{3.6}$$

To maximize the lower bound $\mathscr{F}(Q, \theta)$ in Equation (3.5), EM alternates between maximizing over Q and θ [47]. From here we have the two famous steps:

$$\text{E-Step} : Q_{k+1} \leftarrow \arg\max_Q \mathscr{F}(Q, \theta_k) \tag{3.7}$$

$$\text{M-Step} : \theta_{k+1} \leftarrow \arg\max_\theta \mathscr{F}(Q_{k+1}, \theta) \tag{3.8}$$

It turns out that the maximization problem in the E-Step can be solved analytically. By setting $Q(Z) = P(Z|X, \theta)$, the lower bound becomes

$$\int_Z P(Z|X,\theta)\log\frac{P(Z,X|\theta)}{P(Z|X,\theta)}dZ = \int_Z P(Z|X,\theta)\log P(X|\theta)dZ \tag{3.9}$$

$$= \log P(X|\theta)\int_Z P(Z|X,\theta)dZ \tag{3.10}$$

$$= \log P(X|\theta)\times 1 = \mathscr{L}(\theta). \tag{3.11}$$

In other words, the lower bound turns into an equality when Q is chosen to be the conditional distribution. This also implies that log-likelihood will not decrease as iterations progress. The E- and M- steps can now be simplified to

$$\text{E-Step} : \mathscr{G}(\theta,\theta_k) \triangleq \int_Z P(Z|X,\theta_k)\log P(Z,X|\theta)dZ \tag{3.12}$$

$$= E\left[\log P(Z,X|\theta)\right]_{Z|X,\theta_k} \tag{3.13}$$

$$\text{M-Step} : \theta_{k+1} \leftarrow \arg\max_\theta \mathscr{G}(\theta,\theta_k), \tag{3.14}$$

which is the form EM algorithm is usually presented in. Notice that the term $\int_Z Q(Z)\log Q(Z)dZ$ has been dropped from the computations since it does not depend on parameters θ.

The E-Step involves computing expectation of the joint log-likelihood with respect to the conditional distribution over the hidden variables. This is exactly the inference problem described above. Methods that apply this generic formulation to a specific model generally find $\mathscr{G}(\theta)$ analytically and then take derivatives with respect to specific parameters to come up with update rules for the M-step. The following sections discuss specific formulation of the EM algorithm for Gaussian mixture models and hidden Markov models.

3.6.3 Gaussian Mixture Models

A Gaussian mixture model attempts to capture all the data with a fixed number of Gaussian distributions, called *components* or *clusters*. Once the number of components P has been specified, parameters θ of a GMM are identified with P 3-tuples:

$\theta = (\mu_j, \Sigma_j, p_j), j = 1, ..., P$. Here, μ_j and Σ_j denote mean and covariance for component j and have the same dimensionality as the data. The mixture coefficients p_j provide a measure of relative component importance and are subject to the stochastic constraint $\sum_j p_j = 1$. The likelihood of GMM parameters, given a data point x is, by definition, equal to the density of the data points given those parameters:

$$L(\theta|x) = f(x|\theta) = \sum_{j=1}^{P} \frac{p_j}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_j)^T\Sigma_j^{-1}(x-\mu_j)\right), \tag{3.15}$$

where d denotes the dimensionality. This likelihood calculation can be recognized as a linear combination of P multivariate Gaussian probability density functions (pdfs).

There is a single hidden variable in GMMs, namely the index of the component that generated a sample. The conditional distribution $P(Z|X,\theta)$ can be computed as a posterior over hidden variables using the Bayes rule with p_j's being the prior multinomial distribution. More specifically, the E-Step computes component responsibilities for each pair of sample x_i and component j as

$$\gamma_{ij} = \frac{p_j f(x_i|\mu_j, \Sigma_j)}{\sum_k p_k f(x_i|\mu_k, \Sigma_k)}, \tag{3.16}$$

with $f(x_i|\mu_j, \Sigma_j)$ defined in Equation (3.15).

The M-Step reestimates parameters of the Gaussian mixture model using the component responsibilities from the E-step [8, 24]. The update rules are obtained by computing $\mathscr{G}(\theta, \theta_k)$ from Equation (3.12) analytically and setting its derivatives with respect to each parameter to zero [8]. The constraint $\sum_j p_j = 1$ is enforced by using a Lagrange multiplier. After all necessary computations are carried out, the resulting update rules come out to be

$$\mu_j = \frac{\sum_{i=1}^n \gamma_{ij} x_i}{\sum_{i=1}^n \gamma_{ij}} \quad \Sigma_j = \frac{\sum_{i=1}^n \gamma_{ij}(x_i-\mu_j)(x_i-\mu_j)^T}{\sum_{i=1}^n \gamma_{ij}} \quad p_j = \frac{\sum_{i=1}^n \gamma_{ij}}{n} \tag{3.17}$$

with n denoting the total number of samples.

We can make a GMM behave as QDA by setting the number of components to 1 for each class. A single-component GMM can be further constrained to behave as LDA, by using the same covariance matrix across classes.

Unlike QDA and LDA, training a Gaussian mixture model on a particular dataset will not always result in the same set of parameters θ. While Expectation Maximization itself is a deterministic algorithm, the final solution will depend on the initial assignment of values to the parameters. The implementation used in this thesis draws initial μ_j values from a multivariate Gaussian distribution with mean $\mu = \sum_{i=1}^{n} x_i/n$ and covariance matrix $\Sigma = \sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T/(n-1)$, i.e., using the mean and covariance of the entire dataset. The covariance matrices are initialized to be equal to the covariance matrix of the entire dataset: $\Sigma_j = \Sigma$. The prior distribution p over component responsibilities is taken to be uniform:

$$p_i = 1/P \quad \text{for} \quad i = 1, ..., P.$$

GMMs can be seen as HMMs with the underlying Markov process replaced by the mixture coefficients. As a result, GMMs are unable to capture temporal information and, given an observation sequence, will yield the same likelihood value regardless of the sample ordering in that sequence. GMMs are considered in this thesis to validate HMM results. The two modeling methods have the same spatial complexity and identical performance will indicate that no advantage is gained by introducing the state transition matrix. This in turn would imply that any temporal information present in an observation sequence is not related to the class.

3.6.4 Hidden Markov Models

As already suggested, hidden Markov models [45] introduce a first order Markov process in place of mixture coefficients. Each Gaussian component now becomes a state with p_j's representing the prior multinomial distribution over the state assignment to the first sample. Distribution for state assignment to successive samples comes from the new parameter, a state transition matrix A with element i, j denoting the probability of going from state i to state j.

Likelihood computation is performed in two steps. First, the observation probability matrix B is computed for each pair of sample x_t and state j, using again the multivariate Gaussian pdf:

$$B_{t,j} = \frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(x_t - \mu_j)^T \Sigma_j^{-1}(x_t - \mu_j)\right). \tag{3.18}$$

In theory, distributions other than Gaussian can be associated with the states, but Gaussian pdfs are chosen here to be consistent with the theme of this thesis. In particular, we are interested in investigating Gaussian fit to EEG data and comparison of hidden Markov models with other classifiers that model the data using one or more Gaussian distributions.

Once the observation probability matrix is available, the likelihood of a P-state model given an observation sequence **x** of length T is computed recursively using the *forward* step [45]. We define forward variables α to capture the likelihood of model parameters based on a partial observation sequence up to time t as follows:

$$\alpha_{1,i} = p_i B_{1,i}, \quad 1 \leq i \leq P,$$

$$\alpha_{t+1,j} = \left[\sum_{i=1}^{P} \alpha_{t,i} A_{i,j}\right] B_{t+1,j}, \quad 1 \leq t \leq T-1, \quad 1 \leq j \leq P,$$

$$L(\theta|\mathbf{x}) = P(\mathbf{x}|\theta) = \sum_{i=1}^{P} \alpha_{T,i}.$$

Because the true state assignment for data samples is unknown, the computation of $\alpha_{t+1,j}$ considers all possible state assignments for the previous sample x_t, weighted by the corresponding transition probabilities from state i to state j. Summing up the α values at the last timestep $t = T$ we obtain the probability density of the entire sequence.

A similar *backward* step can be used to compute the likelihood of model parameters based on a partial observation sequence from time t to time T [45]:

$$\beta_{T,i} = 1, \quad 1 \leq i \leq P,$$

$$\beta_{t,i} = \sum_{j=1}^{P} A_{i,j} B_{t+1,j} \beta_{t+1,j}, \quad T-1 \geq t \geq 1, \quad 1 \leq i \leq P,$$

The forward and backward steps together make up the *forward-backward procedure*. This procedure allows us to compute $\gamma_{t,i} = P(\text{being in state } i \text{ at time } t \,|\mathbf{x}, \theta)$ as

$$\gamma_{t,i} = \frac{\alpha_{t,i}\beta_{t,i}}{\sum_{i=1}^{P} \alpha_{t,i}\beta_{t,i}},$$

which is the first part of inference necessary for the E-step in the EM algorithm. The γ variables alone do not provide enough information to reestimate the transition probabilities. Therefore, the second part of inference involves computing $\xi_{t,i,j} = P($being in state i at time t and stat
1 $|\mathbf{x}, \theta)$, defined as

$$\xi_{t,i,j} = \frac{\alpha_{t,i} A_{i,j} B_{t+1,j} \beta_{t+1,j}}{\sum_{i=1}^{P} \sum_{j=1}^{P} \alpha_{t,i} A_{i,j} B_{t+1,j} \beta_{t+1,j}}.$$

Having γ and ξ variables allows for reestimation of transitional model parameter in the M-step [8] by

$$p_i = \gamma_{1,i} \qquad A_{i,j} = \frac{\sum_{t=1}^{T-1} \xi_{t,i,j}}{\sum_{t=1}^{T-1} \gamma_{t,i}}.$$

The spatial model parameters are reestimated in exactly the same way as in the case of Gaussian mixture models (i.e., according to Equation (3.17)):

$$\mu_i = \frac{\sum_{t=1}^{T} \gamma_{t,i} x_t}{\sum_{t=1}^{T} \gamma_{t,i}} \qquad \Sigma_i = \frac{\sum_{t=1}^{T} \gamma_{t,i} (x_t - \mu_i)(x_t - \mu_i)^T}{\sum_{t=1}^{T} \gamma_{t,i}}$$

for state i.

As is the case with GMMs, hidden Markov models produced by the EM algorithm are functions of the initial parameter value assignment. Parameters μ_j, Σ_j, and p_j are initialized the same way as in the Gaussian mixture model case. Rows of the transition matrix A are initialized to be random multinomial distributions with values $A_{i,j}$ representing representing probabilities of going from state i to state j.

The ability to compute γ values allows one to reason in terms of the most likely sequence of states. This inference problem is approached using the Viterbi algorithm [45] that consists of the following steps:

$$\delta_{1,i} = p_i B_{1,i} \quad \psi_{1,i} = 0, \qquad 1 \le i \le P$$

$$\delta_{t,j} = \max_i \left[\delta_{t-1,i} A_{i,j}\right] B_{t,j}, \qquad 1 \le i \le P, \qquad 2 \le t \le T$$

$$\psi_{t,j} = \arg\max_i \left[\delta_{t-1,i} A_{i,j}\right], \qquad 1 \le i \le P, \qquad 2 \le t \le T$$

$$q_T^* = \arg\max_i \left[\delta_{T,i}\right] \qquad q_t^* = \psi_{t+1,q_{t+1}^*}$$

with q^* being the most likely sequence of states. The Viterbi algorithm can be recognized as the forward calculation with the sum replaced by the $\max$ operation in the recursive part of the algorithm. The ψ variables are used to keep track of the indices at every timestep, allowing us to backtrack along the path of the highest likelihood.

3.6.5 Overfi tting, Validation and Model Selection

An iterative training algorithm such as EM will continue to increase likelihood of the model parameters given one particular data set. The problem is that the model may not generalize well to other data sets, particularly other sample sequence from the same class. This problem is known as overfitting and is alleviated by implementing an early stopping mechanism. Early stopping in EM can be implemented by withholding part of the training data and using it as validation to assess the model fit. When likelihood of the model given validation data starts to decrease, the training is terminated even though additional increase in likelihood on the training data is possible.

Early stopping is related to model selection. Model selection is motivated by a desire to have a system that automatically decides on the amount of complexity needed for each mental task. In GMMs and HMMs, the complexity is controlled through the number of components/states. Validation data can also be used to perform model and parameter selection. This is done by training multiple models and selecting the one that has the highest likelihood given the validation data.

When data from multiple classes is not intermixed, the only way to assess model fit on validation data is through log likelihood. On the other hand, when data from more than one class is used in validation, model fit can be assessed based on the classification rate.

3.7 Analysis of Normality

Gaussian mixture models and hidden Markov models assume that the spatial distribution of the data being modeled can be accurately captured with a finite number of Gaussian components. A way to quantify how well a particular dataset fits a mixture of Gaussians would provide an alternative to validation when it comes to selecting the number of Gaussian components. The approach taken here is based on the computations involved in G-means [21].

More specifically, suppose we have a d-dimensional dataset of n samples $\{x_1, x_2, ..., x_n\}$ and a Gaussian mixture model that has been trained on it. To measure the model fit using an analysis of normality, we first compute the component responsibilities for each sample using the Equation (3.16). Unlike the original work on G-means [21], where the authors performed a hard assignment of each sample to a single cluster, analysis of normality in this thesis uses the component responsibilities as soft weights for each sample's influence on each cluster. Using all n samples, the following set of computations is performed for every cluster j.

1. Compute the first principal component v of the cluster by performing SVD on the cluster covariance matrix Σ_j.

2. Obtain the one dimensional projection, $\hat{\mu}_j$, of the cluster mean μ_j, the projection, $\hat{\sigma}_j^2$, of the cluster covariance matrix Σ_j, and the projection y_i of all samples x_i

onto the first principal component by

$$\hat{\mu}_j = \mu_j^T v \quad \hat{\sigma}_j^2 = v^T \Sigma_j v \quad y_i = x_i v.$$

3. Compute the Probability Integral Transformation [16] of the resulting 1-D samples:

$$z_i = F(y_i; \hat{\mu}_j, \hat{\sigma}_j) = \int_{-\infty}^{y_i} f(t; \hat{\mu}_j, \hat{\sigma}_j) dt, \quad 1 \leq i \leq n,$$

   where f is the Gaussian probability density function defined by Equation (3.1). The function F can be recognized as the Gaussian cumulative distribution function (cdf).

4. Reorder z_i's and the corresponding responsibilities γ_{ij}, so that $0 \leq z_{(1)} \leq z_{(2)} \leq ... \leq z_{(n)} \leq 1$ and $\gamma_{(i)j}$ corresponds to $z_{(i)}$.

5. Compute the empirical distribution function (edf) of z_i's:

$$F_n(z) = \begin{cases} 0, & z < z_{(1)} \\ c_k, & z_{(k)} \leq z < z_{(k+1)} \\ 1, & z_{(n)} \leq z \end{cases} \quad \text{with} \quad c_k = \frac{\sum_{i=1}^{k} \gamma_{(i)j}}{\sum_{i=1}^{n} \gamma_{(i)j}} \tag{3.19}$$

   This distribution function parallels the more traditional definition of an edf, a step function where $F_n(z) = i/n$ for $z_{(i)} \leq z \leq z_{(i+1)}$. The problem with the traditional definition is that it does not take into account how well a sample $z_{(i)}$ is captured by the cluster j. As a result, the edf can highly deviate from an accurate representation of the cluster distribution. The function definition in Equation (3.19) automatically accounts for the contribution of each sample through the use of γ_{ij}. Samples that are poorly captured by cluster j will have little impact on the empirical distribution function. On the other hand, samples that are well captured by cluster j will attribute a step value closer to $1/n$, as desired.

6. Compute the test statistic Q_j as the deviation between the empirical and the true distribution functions by

$$Q_j = \left[\sum_{i=1}^{n} \gamma_{ij}\right] \int_0^1 \left[F_n(z) - z\right]^2 dz. \tag{3.20}$$

This is a version of the Cramer-von Mises statistic [16, 49], an edf based calculation that measures the sum of squared errors (SSE) between the true cdf and the edf estimated from the data. Equation (3.20) compares the probability integral transformations of the two. In case of the true cdf, the transformation yields a uniform $(0, 1)$ distribution [6] and, thus, $F(z) = z$. We use $F_n(z)$ defined above as the probability integral transformation of the edf of the actual samples y_i. The quantity $\sum_{i=1}^{n} \gamma_{ij} = \sum_{i=1}^{n} \gamma_{(i)j}$ is a continuous quantity, equivalent to the number of points captured by the cluster j.

By carrying out the integration in Equation (3.20), one can arrive at

$$Q_j = \frac{1}{3}\left[\sum_{i=1}^{n} \gamma_{ij}\right] \left[z_{(1)}^3 + \sum_{k=1}^{n-1} \left[\left(z_{(k+1)} - c_k\right)^3 - \left(z_{(k)} - c_k\right)^3\right] + \left(1 - z_{(n)}\right)^3\right] \tag{3.21}$$

with c_k's defined by Equation (3.19).

Carrying out this computation for each cluster yields a vector of statistic values $Q_1, Q_2, ..., Q_K$, where K is the number of clusters in the model. Larger values of Q_j imply greater deviation of the empirical distribution from the true cdf, which in turn presents more evidence that the cluster is not well-fit with a Gaussian distribution. To assess the fit of the entire model, we take the maximum Q_j value: $\hat{Q} = \max_{j=1...K} Q_j$. The maximum Q_j value corresponds to a cluster that has the worst fit.

Recall that a Gaussian mixture model solution is dependent on the initial value assignment to the parameters. To perform a statistically sound comparison a sample of 30 GMMs is created for each value of the number of clusters. The comparison is then performed using the thirty $\hat{Q}$ values available from each model fit.

Although the test statistics are computed only for the first principal component of each cluster, further normality analysis can be performed by considering the dataset projection onto other principal components. This is outside the scope of this thesis because testing along the first principal component already provides some comparative information regarding the Gaussian fit of GMMs with different number of clusters. The first principal component captures the maximum direction of variance in each cluster and is independent of other principal components if the cluster is indeed Gaussian [26].

Chapter 4

Data

The dataset used in the following experiments was collected by Zak Keirn for his master's work [27]. The dataset contains EEG data recorded from channels C3, C4, P3, P4, O1 and O2 (positioned according to the international 10-20 standard [25, 30], as presented in Figure 4.1) from 7 subjects. Keirn recorded the data at a sampling frequency of 250Hz and filtered it to 0.1 - 100Hz frequency band. This dataset has been used extensively in brain-computer interface experiments conducted at Colorado State University [5, 2, 4, 29, 3].

The entire dataset was visually inspected and narrowed down to subjects 1, 2, 6, and 7. This subset was used in the experiments described in the following chapter.

The data for each subject consists of multiple 10-second trials. For each trial, the subject was asked to perform one of the following five mental tasks for 10 seconds [27]:

- **Baseline:** The subject was asked to relax without performing any particular mental task.

- **Math:** The subject was instructed to multiply two numbers in their head with the numbers chosen such that the answer was not trivial.

- **Letter:** The subject was asked to compose a letter in their head. In consecutive trials, the subject was to continue composing from where he/she left off.

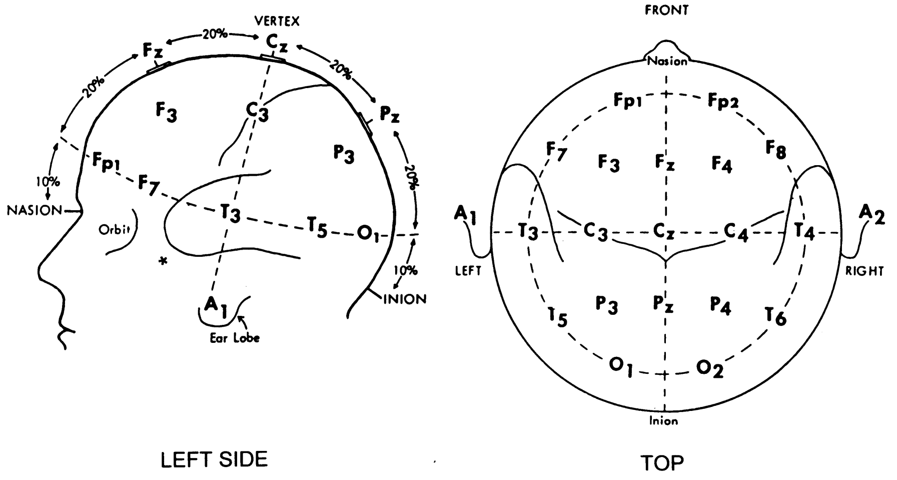

Figure 4.1: Placement of electrodes on the scalp according to the international 10-20 standard [25, 30].

- **Counting:** The subject was instructed to imagine a whiteboard with numbers being written on it sequentially.
- **Rotation:** The subject was asked to imagine rotation of a three-dimensional figure.

The first trial of each task was recorded (in random ordering of the tasks) before moving on to record the second trial. Similarly the second trial for each task was recorded prior to the third trial, and so on. A single recording session consisted of 5 trials for each task. There were two sessions recorded, each session on a different day, for subjects 1 and 6, yielding a total of 10 trials/task for these subjects. Subjects 2 and 7 participated in a single recording session yielding 5 trials/task of available data.

Chapter 5

Results

5.1 Spatial Structure

Prior to measuring temporal structure at any level, we investigate how time-lag embedding affects the data spatially. In particular, we would like to understand what happens when there is no temporal correlation between neighboring samples in observation sequences. By using the independence of neighboring samples as the base case, we can then measure additional effects of temporal structure.

The data in this experiment was generated from three highly overlapping observation sequences with each observation sequence consisting of 1000 samples drawn independently from a 2-D Gaussian distribution. The distribution parameters are given by the means $\mu_1 = [-0.5, 0]^T, \mu_2 = [0.5, 0]^T, \mu_3 = [0, 0.5]^T$, and three identity covariance matrices. The generated data is displayed in the upper left corner of Figure 5.1. The remainder of Figure 5.1 shows the Fisher projection [24] of the same data after application of time-lag embedding. The three sequences become linearly separable as the number of lags increases.

Recall that Fisher projection gives a visual representation of the classification accuracy one would expect to achieve with Linear Discriminant Analysis. As the number of lags increases, so does the dimensionality of the corresponding LDA solution. A rather trivial but important fact is that the likelihood of a high-dimensional LDA solu-

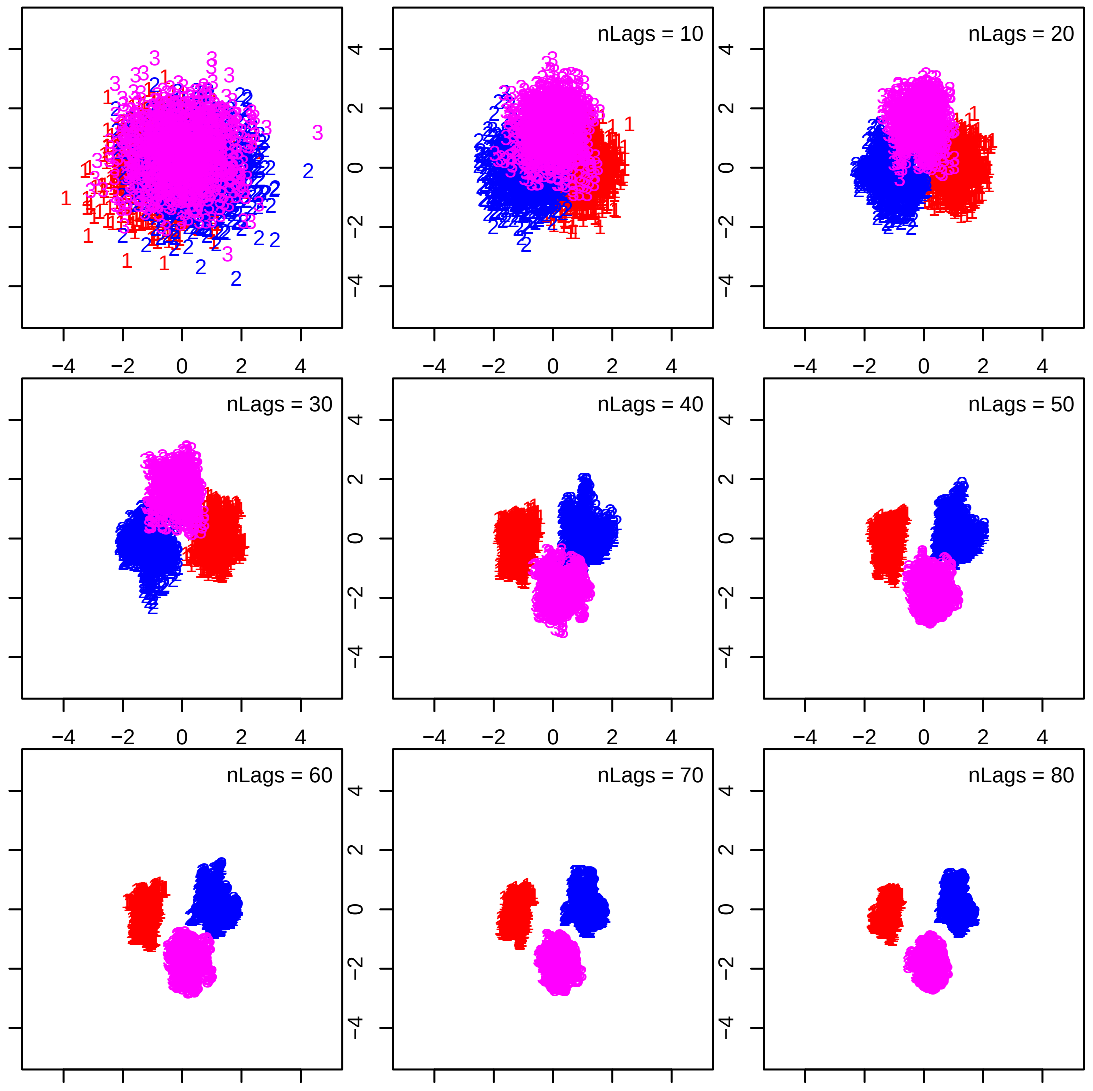

Figure 5.1: Fisher projection of time-lag data from three observation sequences.

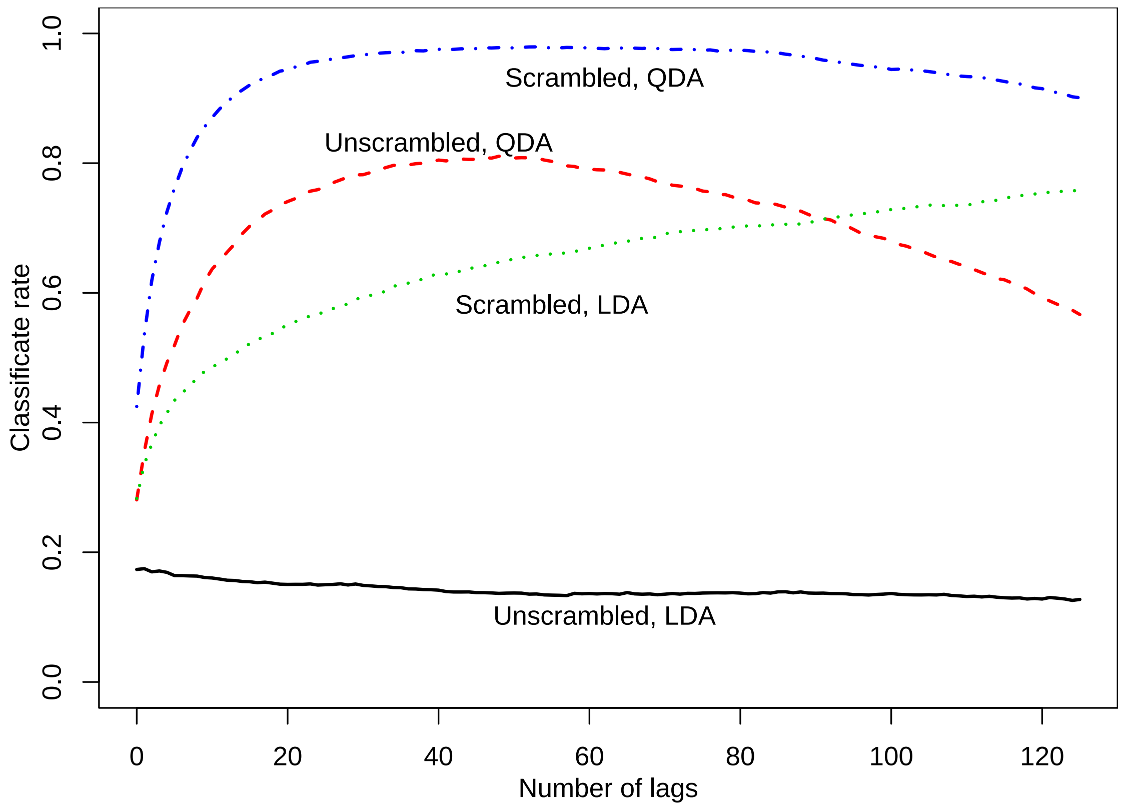

Figure 5.2: Five-fold cross-validation classification accuracy plotted against the number of lags for LDA applied to unscrambled data, LDA applied to scrambled data, QDA applied to unscrambled data, and QDA applied to scrambled data. Training and testing is performed on the same trial.

tion given a time-lag sample is the product of likelihoods of the low-dimensional LDA solution given the original samples that make up the time-lag sample. This fact holds true as long as the original samples are independent and tells us that we can achieve the same classification accuracy either by lagging the data or by multiplying posterior probabilities of adjacent samples.

5.2 Local Temporal Structure

The question of interest is what happens when the samples are not independent. To answer this question, we turn to the Keirn dataset and consider the first trial of each of the five tasks from the first subject. This gives a total of 2500 samples for each class. To assess the dependence of adjacent samples, a second dataset is created. It consists of the same five 2500-sample observation sequences but with the samples randomly reordered in each sequence. The reordering is done by taking a random permutation of the original samples. The second dataset has exactly the same spatial distribution but any temporal dependence of adjacent samples is destroyed through this reordering.

Figure 5.2 displays the classification rate obtained by running LDA and QDA on the two datasets, plotted against the number of lags. The classification rate is computed as the fraction of test samples classified correctly divided by the total number of test samples. Perfect performance is given by a value of 1.0, while a classifier that assigns a random label to each test sample will yield (in expectation) accuracy of 0.2 on this five-class problem.

Each curve in Figure 5.2 displays five-fold cross-validation results, where the fold division was performed at every 2 seconds of a 10 second trial. In cross-validation, each fold in turn is used as the test set with the remaining folds being treated as the training set. The results for cross-validation are reported as the average classification rate across the five folds.

In the interest of brevity, we use labels `ulda` and `uqda` to refer to LDA and QDA classification performed on the original, unscrambled dataset. Similarly, the labels `slda` and `sqda` will refer to LDA and QDA applied to the scrambled dataset, created through reordering of samples in the first dataset.

The following observations can be made regarding Figure 5.2.

- `ulda` performance steadily decreases as the number of lags increases.
- `slda` performance steadily increases as the number of lags increases.
- As the number of lags increases, the QDA curves increase then decrease.

- Classification is more accurate for the scrambled datasets, even without lagging.

Prior to explaining these observations, we check to see how well they generalize to other trials and subjects in the Keirn dataset. The same experiment was repeated for all available trials and performance measured at 0, 42, 83, and 125 lags. These correspond to lagging the data 0, 1/6, 1/3, and 1/2 of a second, respectively. Applying a large number of lags results in a very high dimensional space, which in turn leads to problems with singular covariance matrices. For these reasons, any number of lags higher than 125 was not considered. Table 5.1 and Table 5.2 present the corresponding classification rates for five-fold cross-validation. The first row in these tables is seen to be consistent with Figure 5.2.

We can now revisit the observations made about Figure 5.2. First, LDA always performs better with the scrambled dataset. Moreover, the difference in LDA performance on scrambled and unscrambled datasets grows with the number of lags. Recall, that LDA makes its classification decision based solely on the closest centroid in the whitened space. By considering more samples, we are able to determine the closest centroid more accurately. However, local temporal information in EEG consists of high correlation between neighboring samples. As a result, considering more samples in the unscrambled dataset yields less information needed for centroid approximation than it does in the scrambled case. The consistent decrease in `ulda` performance suggests that the sample mean drifts throughout the trial and doesn't generalize well from one 1/5th of the trial to the next. Figure 5.3 shows that this is indeed the case.

The drift of the sample mean also explains why just scrambling the data without lagging helps to increase the classification rate. Scrambling the data eliminates this problem and reduces the classification problem to the case that was already depicted in Figure 5.1 during our spatial structure discussion in the previous section.

Second observation is that, overall, QDA yields higher performance than LDA, at

Subj./	# lags = 0				# lags = 42			
Trial	ulda	uqda	slda	sqda	ulda	uqda	slda	sqda
1/1	0.173	0.281	0.285	0.426	0.139	0.805	0.628	0.958
1/2	0.107	0.196	0.261	0.368	0.089	0.806	0.585	0.880
1/3	0.117	0.252	0.265	0.414	0.083	0.866	0.541	0.941
1/4	0.157	0.295	0.304	0.441	0.136	0.843	0.663	0.980
1/5	0.090	0.252	0.251	0.423	0.051	0.870	0.603	0.972
1/6	0.175	0.249	0.243	0.336	0.140	0.832	0.517	0.786
1/7	0.143	0.248	0.251	0.348	0.107	0.858	0.526	0.810
1/8	0.120	0.286	0.244	0.391	0.082	0.861	0.438	0.927
1/9	0.153	0.251	0.240	0.339	0.122	0.852	0.485	0.801
1/10	0.136	0.290	0.242	0.386	0.066	0.942	0.486	0.926
2/1	0.296	0.332	0.343	0.384	0.314	0.809	0.723	0.781
2/2	0.269	0.308	0.341	0.384	0.275	0.889	0.644	0.732
2/3	0.265	0.320	0.313	0.394	0.302	0.908	0.637	0.797
2/4	0.284	0.325	0.331	0.375	0.281	0.900	0.629	0.722
2/5	0.219	0.270	0.265	0.335	0.183	0.867	0.612	0.772
6/1	0.166	0.308	0.233	0.351	0.126	0.786	0.412	0.851
6/2	0.181	0.298	0.245	0.365	0.172	0.766	0.518	0.882
6/3	0.173	0.277	0.240	0.340	0.164	0.758	0.448	0.800
6/4	0.130	0.269	0.226	0.337	0.070	0.882	0.364	0.866
6/5	0.161	0.332	0.236	0.394	0.124	0.842	0.400	0.946
6/6	0.172	0.259	0.230	0.327	0.147	0.922	0.486	0.693
6/7	0.146	0.247	0.236	0.337	0.082	0.839	0.496	0.835
6/8	0.162	0.273	0.254	0.341	0.100	0.845	0.466	0.813
6/9	0.173	0.273	0.229	0.330	0.152	0.794	0.456	0.810
6/10	0.135	0.275	0.230	0.351	0.060	0.803	0.412	0.874
7/1	0.148	0.277	0.262	0.372	0.131	0.843	0.560	0.834
7/2	0.175	0.303	0.268	0.378	0.238	0.816	0.623	0.810
7/3	0.137	0.308	0.254	0.409	0.106	0.874	0.576	0.872
7/4	0.150	0.243	0.256	0.361	0.117	0.817	0.559	0.821
7/5	0.181	0.270	0.284	0.388	0.186	0.763	0.791	0.936

Table 5.1: Five-fold cross-validation results (as fraction classified correctly) for the entire Keirn dataset using 0 and 42 lags. Training and testing is performed on the same trial.

Subj./	# lags = 83				# lags = 125			
Trial	ulda	uqda	slda	sqda	ulda	uqda	slda	sqda
1/1	0.138	0.738	0.680	0.960	0.127	0.567	0.796	0.930
1/2	0.086	0.769	0.755	0.824	0.075	0.588	0.793	0.677
1/3	0.071	0.798	0.673	0.877	0.072	0.589	0.730	0.772
1/4	0.126	0.790	0.722	0.954	0.126	0.658	0.735	0.812
1/5	0.063	0.846	0.703	0.957	0.078	0.733	0.806	0.607
1/6	0.151	0.658	0.659	0.640	0.134	0.361	0.743	0.413
1/7	0.096	0.732	0.689	0.670	0.098	0.589	0.672	0.503
1/8	0.063	0.788	0.502	0.843	0.064	0.580	0.535	0.721
1/9	0.112	0.811	0.573	0.697	0.112	0.670	0.673	0.584
1/10	0.061	0.889	0.624	0.874	0.066	0.736	0.702	0.724
2/1	0.311	0.748	0.851	0.774	0.310	0.529	0.841	0.616
2/2	0.294	0.793	0.726	0.649	0.289	0.435	0.794	0.588
2/3	0.277	0.816	0.719	0.743	0.275	0.526	0.768	0.586
2/4	0.298	0.875	0.683	0.733	0.305	0.698	0.730	0.604
2/5	0.177	0.824	0.740	0.711	0.179	0.628	0.783	0.594
6/1	0.098	0.737	0.478	0.771	0.094	0.600	0.561	0.499
6/2	0.158	0.817	0.659	0.716	0.168	0.716	0.723	0.519
6/3	0.163	0.748	0.490	0.708	0.164	0.689	0.582	0.503
6/4	0.077	0.889	0.407	0.788	0.078	0.826	0.426	0.721
6/5	0.113	0.796	0.430	0.893	0.112	0.691	0.493	0.708
6/6	0.144	0.883	0.551	0.632	0.150	0.736	0.661	0.501
6/7	0.073	0.798	0.572	0.796	0.081	0.739	0.626	0.702
6/8	0.078	0.820	0.536	0.802	0.081	0.730	0.530	0.661
6/9	0.164	0.751	0.494	0.623	0.131	0.671	0.556	0.489
6/10	0.061	0.835	0.511	0.826	0.052	0.809	0.546	0.649
7/1	0.122	0.752	0.659	0.732	0.105	0.549	0.689	0.517
7/2	0.230	0.667	0.739	0.761	0.216	0.362	0.767	0.546
7/3	0.093	0.796	0.745	0.763	0.081	0.499	0.739	0.500
7/4	0.125	0.830	0.713	0.755	0.127	0.722	0.732	0.676
7/5	0.188	0.691	0.891	0.866	0.180	0.513	0.930	0.650

Table 5.2: Five-fold cross-validation results (as fraction classified correctly) for the entire Keirn dataset using 83 and 125 lags. Training and testing is performed on the same trial.

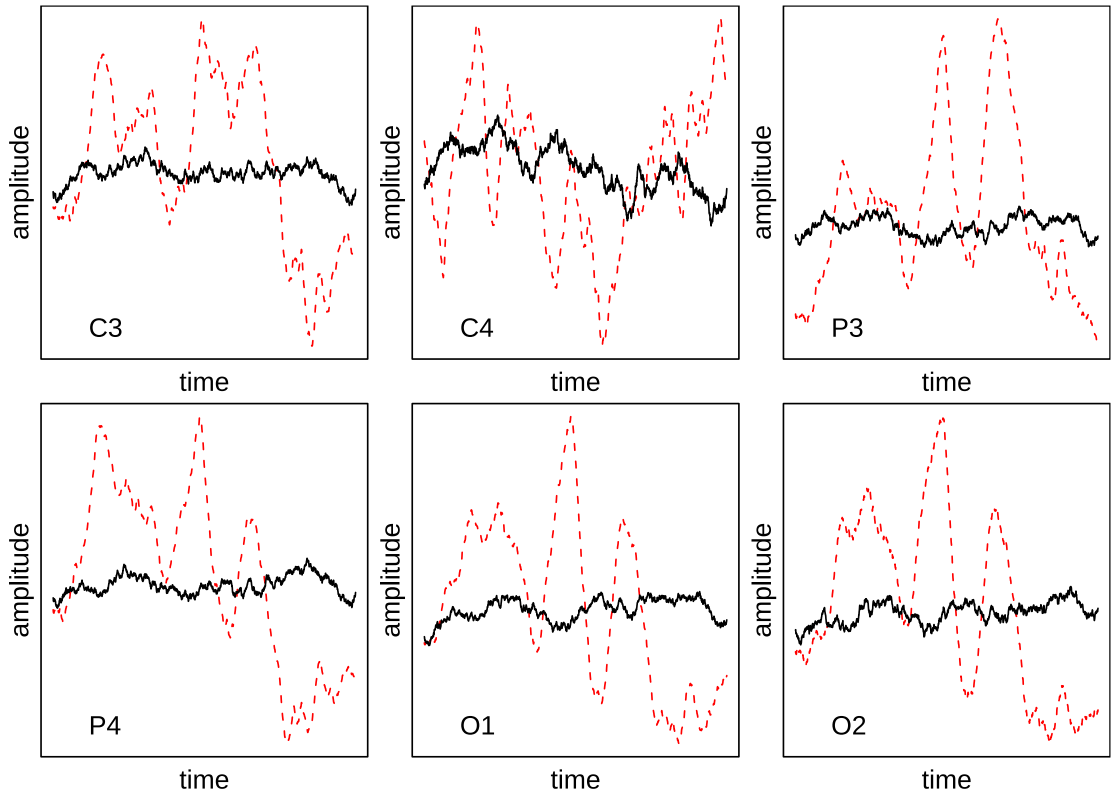

Figure 5.3: Mean amplitude computed for samples in a sliding one-second window for all 6 channels of the first trial. Data from the first mental task performed by Subject 1 was used. Plotted are the mean amplitudes vs. time for scrambled (—) and unscrambled (- -) datasets.

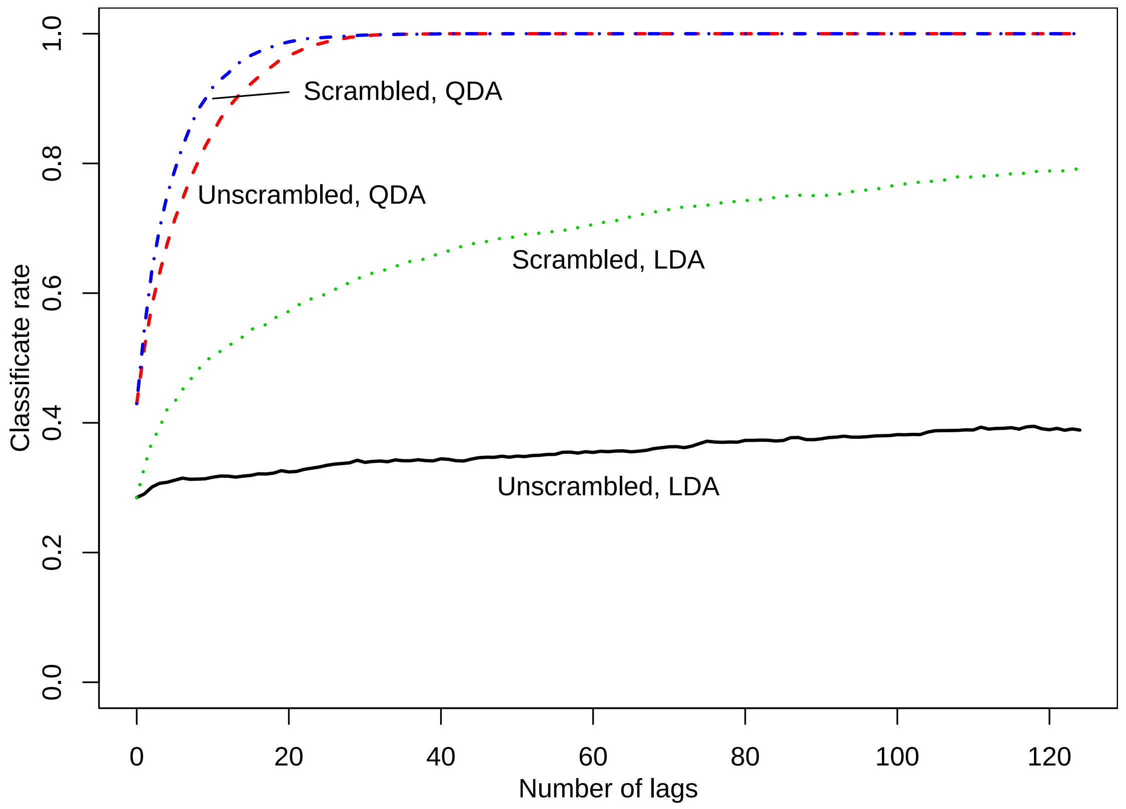

Figure 5.4: Classification accuracy on the first trial of the first subject. Same data was used for training and testing.

times reaching as high as 90% accuracy. This is expected, since QDA uses the first two moments of variance and provides more flexibility to the Bayesian decision boundary than LDA, which uses only the first moment of variance. Unlike LDA, however, QDA performance peaks for some number of lags and then begins to decrease. From Table 5.2, we see that QDA performance is lower for the half a second delay (last four columns) than it is for 1/3 of a second delay (columns 2 through 6) for all subjects and all trials. This is rather surprising because one would expect that once the classifier finds a good separating boundary in d dimensions, that boundary will still be present in $p > d$ dimensions. The consequence of such invariance should be a monotonically increasing classification accuracy function. This is what is being observed in Figure 5.4, which displays classification accuracy on the training data. The fact that the classification accuracy on the test data begins to go down suggests that we are dealing with overfitting, where a very large number of dimensions provides enough flexibility for the training data to become quadratically separable. Unfortunately, the quadratic boundary begins to separate the training data so well that it fails to generalize to the test data.

It also turns out that the intuition regarding boundary invariance is not entirely accurate. Suppose that an LDA model for two classes is defined by μ_1, p_1, the mean and prior of the first class, and μ_2, p_2, the same set of parameters of the second class. Suppose also that Σ defines the common covariance matrix. The boundary between two classes is then defined to be all points x, for which the two Bayesian posterior probabilities are equal [24]. After application of the Bayes rules, the boundary can be expressed in terms of the Gaussian probability density functions of the two classes:

$$f_1(x)p_1 = f_2(x)p_2$$

$$p_1 \exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right) = p_2 \exp\left(-\frac{1}{2}(x-\mu_2)^T\Sigma^{-1}(x-\mu_2)\right)$$

$$\log\frac{p_1}{p_2} + \left(x - \frac{\mu_1}{2} - \frac{\mu_2}{2}\right)^T \Sigma^{-1}(\mu_1 - \mu_2) = 0 \tag{5.1}$$

where we took the log of both sides and canceled out the quadratic term $x^T\Sigma^{-1}x$ after some simplification. Suppose now that this boundary is defined for d dimensions and let's see what happens when we bump the number of dimensions to $d + 1$. For the sake of argument, suppose that we are working with the same number of samples n as in the original d-dimensional space. The only difference is that each sample now has an additional $(d + 1)^{th}$ coordinate appended to it. Increasing dimensionality by 1 is equivalent to increasing the numbers of lags by 1 when working with a 1-D time series. Understanding what happens in $d + 1$ dimensions would then allow us to inductively generalize to an arbitrary $p > d$ number of dimensions.

Because estimating the first two moments of variance involves component-wise computations, the new samples $\breve{x}_i$ and the new model parameters computed from those samples can be expressed using the original quantities. More specifically,

$$\breve{x}_i = \begin{bmatrix} x_i \\ a_i \end{bmatrix} \quad \breve{\mu}_j = \begin{bmatrix} \mu_j \\ m_j \end{bmatrix} \quad \breve{\Sigma} = \begin{bmatrix} \Sigma & v \\ v^T & \sigma \end{bmatrix} \quad \text{for} \quad \begin{array}{l} i = 1, 2, ..., n, \\ j = 1, 2, \end{array}$$

where a_i's, m_j's, σ are the new scalars and v is a new d by 1 column vector introduced by the increase in dimensionality. The LDA boundary for $d + 1$ dimensions can now be defined as

$$\log \frac{p_1}{p_2} + \left(\breve{x} - \frac{\breve{\mu}_1}{2} - \frac{\breve{\mu}_2}{2} \right)^T \breve{\Sigma}^{-1}(\breve{\mu}_1 - \breve{\mu}_2) = 0, \tag{5.2}$$

where $\breve{x}$ was introduced to emphasize that the solution will be a $d + 1$-dimensional (infinite) set of points. We would like to compare the projection of this new boundary with the original d-dimensional boundary defined by the Equation (5.1). To do so, we set the $(d + 1)^{th}$ coordinate of $\breve{x}$, $\breve{\mu}_1$, $\breve{\mu}_2$ to 0, producing

$$\log \frac{p_1}{p_2} + \begin{bmatrix} x - \mu_1/2 - \mu_2/2 \\ 0 \end{bmatrix}^T \begin{bmatrix} \Sigma & v \\ v^T & \sigma \end{bmatrix}^{-1} \begin{bmatrix} \mu_1 - \mu_2 \\ 0 \end{bmatrix} = 0. \tag{5.3}$$

Note that the 0 entries will cancel out any covariance matrix content in the $(d+1)^{th}$ row and the $(d + 1)^{th}$ column. Thus, we are only interested in the upper-left d by d entries, which are given by

$$\left[\begin{array}{cc} \Sigma & v \\ v^T & \sigma \end{array}\right]^{-1} = \left[\begin{array}{cc} \Sigma^{-1} + \Sigma^{-1} v(\sigma - v^T \Sigma^{-1} v)^{-1} v^T \Sigma^{-1} & \ldots \\ \ldots & \ldots \end{array}\right].$$

We label $C = \Sigma^{-1} v(\sigma - v^T \Sigma^{-1} v)^{-1} v^T \Sigma^{-1}$, plug it into the Equation (5.3) and simplify to get

$$\log \frac{p_1}{p_2} + \left(x - \frac{\mu_1}{2} - \frac{\mu_2}{2}\right)^T (\Sigma^{-1} + C)(\mu_1 - \mu_2) = 0.$$

This result can be separated into a term that depends on C and terms that do not, resulting in

$$\log \frac{p_1}{p_2} + \left(x - \frac{\mu_1}{2} - \frac{\mu_2}{2}\right)^T \Sigma^{-1}(\mu_1 - \mu_2) + \left(x - \frac{\mu_1}{2} - \frac{\mu_2}{2}\right)^T C(\mu_1 - \mu_2) = 0.$$

Comparing this result to Equation (5.1), we note that the two are equivalent when C is 0. Going back to the definition of C, we see that this happens only when v is 0, which (due to the Normality assumption) would indicate independence between the $(d+1)^{th}$ coordinate and the other d coordinates. Moreover, when v is not 0, the new $(d+1)$-dimensional boundary will not go through the original boundary and it will not be parallel to it. In other words, the new $(d+1)$-dimensional boundary will contain the original d-dimensional boundary if and only if the new $(d+1)^{th}$ dimension is independent of the original d dimensions.

A similar computation can be carried out for the QDA decision boundary. The d dimensional boundary is defined by

$$\frac{1}{2} \log \frac{|\Sigma_2|}{|\Sigma_1|} + \log \frac{p_1}{p_2} + \frac{1}{2} \left[(x - \mu_2)^T \Sigma_2^{-1} (x - \mu_2) - (x - \mu_1)^T \Sigma_1^{-1} (x - \mu_1)\right] = 0, \quad (5.4)$$

where class-specific covariance matrices Σ_1 and Σ_2 are now used in place of the former common covariance matrix Σ. Suppose that the $(d+1)^{th}$ dimension is independent of the other d dimensions. This implies

$$\breve{\Sigma}_i = \left[\begin{array}{cc} \Sigma_i & 0 \\ 0 & \sigma \end{array}\right] \quad \breve{\Sigma}_i^{-1} = \left[\begin{array}{cc} \Sigma_i^{-1} & 0 \\ 0 & \sigma^{-1} \end{array}\right] \quad |\breve{\Sigma}_i| = \sigma_i |\Sigma_i| \qquad i = 1, 2.$$

Plugging this into the equation for the $d + 1$-dimensional boundary and computing the projection onto the original d dimensions, we get

$$\frac{1}{2}\log\frac{\sigma_2|\Sigma_2|}{\sigma_1|\Sigma_1|} + \log\frac{p_1}{p_2} + \frac{1}{2}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2) - (x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right] = 0,$$

which is very similar to the Equation (5.4) but not quite identical. From here, we can conclude that, unlike in the case with LDA, the $d + 1$-dimensional quadratic decision boundary will not contain the original d dimensional quadratic boundary when the new dimension is independent. Instead, its projection into the d dimensions will be offset by $\frac{1}{2}\log\frac{\sigma_2}{\sigma_1}$ from the original boundary. Note, however, that the quadratic curvature is preserved if the $(d + 1)^{th}$ dimension is independent.

The analysis above demonstrates that the Bayesian decision boundary found in d dimensions by LDA is invariant to the increase in dimensionality if and only if the additional dimensions are independent of these d dimensions. In our experiments so far, `slda` is the only place where this occurs. Because the data is scrambled and assumed to come from a Gaussian distribution, we can apply Proposition 1 to show that the new dimensions introduced through lagging will be independent of the original dimensions. In case of the other three curves (i.e., `ulda`, `uqda`, `sqda`), the dimensions are either correlated or the decision boundary is quadratic leading to the lack of invariance in the decision boundary.

Although no connection to monotonically increasing classification rate is provided, the above analysis gives an intuition regarding what happens to a decision boundary as dimensionality increases. A non-stationary decision boundary does not necessarily lead to a potential decrease in classification because, as observed in Figure 5.4, the training error keeps decreasing as dimensionality increases and the decrease of the classification rate observed in curves `uqda` and `sqda` in Figure 5.2 is mostly due to overfitting. One thing that the above analysis does suggest is that the decision boundary found for p dimensions in the `slda` experiment will have to contain the decision boundary found for

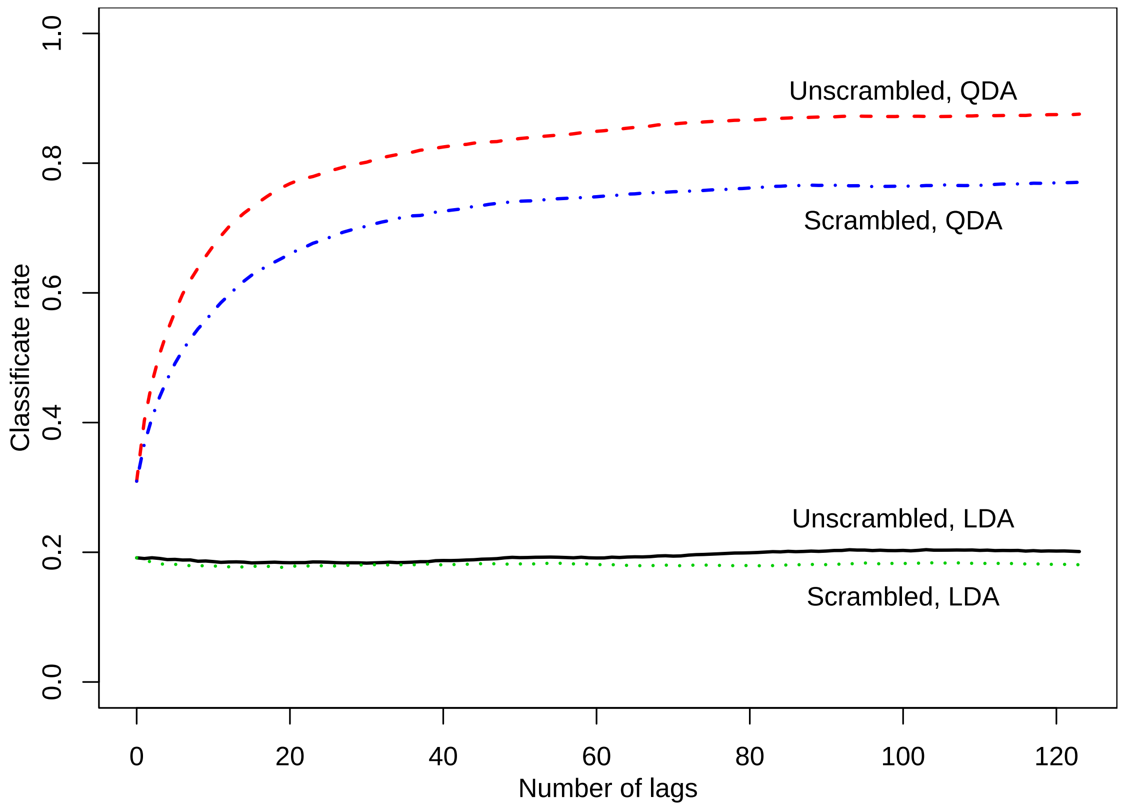

Figure 5.5: Five-fold cross-validation classification accuracy plotted against the number of lags for LDA applied to unscrambled data (ulda), LDA applied to scrambled data (slda), QDA applied to unscrambled data (uqda), and QDA applied to scrambled data (sqda). Training is performed on four trials in the session. Testing is performed on the remaining fifth trial.

$d < p$ dimensions. While this may result in less flexibility when it comes to separating the data, the problem of overfitting ends up being less pronounced. This reasoning is related to the flexibility versus overfitting trade-off (also known as the bias-variance trade-off [24]) that is often associated with the choice of the classifier complexity.

The drift of the sample mean throughout the trial motivates the use of multiple trials for training. This is examined next.

Subj./	# lags = 0				# lags = 42			
Sess.	ulda	uqda	slda	sqda	ulda	uqda	slda	sqda
1/1	0.191	0.310	0.191	0.310	0.187	0.828	0.183	0.727
1/2	0.205	0.297	0.205	0.297	0.219	0.808	0.253	0.677
2/1	0.297	0.333	0.297	0.333	0.333	0.778	0.457	0.654
6/1	0.215	0.285	0.215	0.285	0.240	0.594	0.279	0.671
6/2	0.199	0.278	0.199	0.278	0.214	0.568	0.190	0.601
7/1	0.222	0.306	0.222	0.306	0.225	0.691	0.336	0.549

Table 5.3: Five-fold cross-validation results (as fraction classified correctly) for the entire Keirn dataset using 0 and 42 lags. Training is performed on four trials in the session. Testing is performed on the remaining fifth trial.

Subj./	# lags = 83				# lags = 125			
Sess.	ulda	uqda	slda	sqda	ulda	uqda	slda	sqda
1/1	0.201	0.868	0.184	0.762	0.201	0.875	0.169	0.776
1/2	0.230	0.846	0.261	0.716	0.231	0.855	0.272	0.726
2/1	0.343	0.801	0.487	0.677	0.347	0.808	0.501	0.686
6/1	0.249	0.625	0.283	0.718	0.267	0.638	0.299	0.732
6/2	0.204	0.593	0.209	0.636	0.198	0.603	0.202	0.672
7/1	0.233	0.705	0.380	0.553	0.235	0.704	0.396	0.534

Table 5.4: Five-fold cross-validation results (as fraction classified correctly) for the entire Keirn dataset using 83 and 125 lags. Training is performed on four trials in the session. Testing is performed on the remaining fifth trial.

5.3 Temporal Structure Across Trials

The set of experiments performed here is similar to those in the previous section. The main difference now is that the training is performed using four trials in a session and the resulting classifier is tested on the remaining fifth trial. Samples are scrambled only within each trial but are not inter-mixed across trials.

By considering multiple trials, we are ensuring that we are classifying mental tasks and not just times of recording. Figure 5.5 displays detailed results for the first session of the first subject and Tables 5.3 and 5.4 are presented in order to show how well these results generalize to other subjects and sessions.

Comparing these results to the single-trial experiments, we note the following differences. First, `slda` no longer vastly outperforms `ulda`. In fact, the two algorithms tend to be competitive with each other. The reason for this behavior is related to the previous discussion on the sample mean drift. More specifically, training on four trials builds a linear boundary that best separates the sample means of those four trials. The problem is that the sample mean is different in the fifth trial and the constructed LDA classifier fails to capture this difference. Because the samples are not allowed to be inter-mixed across trials, scrambling the data does not alleviate this problem. This is in contrast to the earlier experiments where scrambling annihilated any difference in the sample means across the different folds in the cross-validation. Failure to eliminate this difference here also explains why the results are identical between scrambled and unscrambled versions of the algorithms when the number of lags is zero (i.e., columns 2 through 6 of Table 5.3).

The second difference between single-trial and multi-trial experiments is the apparent lack of overfitting. The curves `uqda` and `sqda` in Figure 5.5 maintain monotonic increase between 0 and 125 lags. From Table 5.4 we also observe that, except for the first session of subject 7, the performance of QDA classifiers was higher with 125 lags

Subj./	# lags = 250			
Sess.	ulda	uqda	slda	sqda
1/1	0.234	0.850	0.176	0.766
1/2	0.219	0.816	0.284	0.700
2/1	0.367	0.827	0.576	0.645
6/1	0.287	0.658	0.338	0.724
6/2	0.203	0.633	0.206	0.631
7/1	0.235	0.681	0.430	0.537

Table 5.5: Five-fold cross-validation results (as fraction classified correctly) for the entire Keirn dataset using 250 lags. Training is performed on four trials in the session. Testing is performed on the remaining fifth trial.

($1/2$ of a second) than with 83 lags ($1/3$ of a second). It turns out that overfitting still occurs but it occurs for a much higher number of dimensions. Table 5.5 displays the classification results when 250 lags were used. The classification rates are seen to be lower than those corresponding to 125 lags. The fact that a higher number of dimensions is required to observe overfitting is expected because we are dealing with a large number of samples. Overfitting a larger number of training samples requires higher flexibility of a classifier, which in this case is controlled through the dimensionality. This, again, is related to the bias-variance trade-off mentioned earlier.

The final difference between single-trial and multi-trial experiments is the seemingly inconclusive results regarding whether the scrambling helps to increase the classification rate. Based on the results presented in Tables 5.1, 5.2, 5.3, 5.4, and 5.5 we are unable to say whether `sqda` or `uqda` performs universally better than the other. While scrambling an EEG trial is not a practical thing to do, it is important to understand how and why it affects the classification rate. From the experiments thus far, we were able to observe that applying a classifier to a time-lag version of the data increases the classification rate. By examining the scrambled version of the dataset, we would like to conclude whether this increase in performance is due to some geometrical structure that is created from the temporal structure through lagging or the increase in performance is simply attributed to considering multiple samples at once. Scrambling destroys any correlation between neighboring samples in a time series. More specifically, the mean and the variance become more uniform across the samples that get scrambled (in our case, across each of the trials). The `sqda` curve provides a quantitative measure of how the classification accuracy is affected based solely on considering more samples at a time. The deviation of `uqda` curve from `sqda` shows how the classification accuracy is affected by the temporal information. When `uqda` is higher than `sqda`, we can conclude that lagging transforms temporal correlation of neighboring samples into a certain geometrical structure that is then exploited by QDA to enhance the classification accuracy. When `uqda` is lower than `sqda`, we can conclude one of two things. First, perhaps the geometrical structure does not generalize well between the training and the test data and that we are, again, dealing with overfitting. However, Figure 5.4 demonstrates lower performance on the unscrambled dataset even for the training data, suggesting that overfitting alone does not explain what is happening. The second, more intuitive conclusion may be that the created geometrical structure presents more of a challenge for such simple classifiers as LDA and QDA. Further investigation needs to consider a larger number of sessions for each subject to determine whether the difficulty of geometric structure separation is subject-dependent.

Later, Gaussian mixture models (GMMs) are considered. Besides being a baseline for evaluating performance of hidden Markov models, GMMs also arise as a generalization of LDA and QDA due to modeling each class with more than a single Gaussian.

5.4 Overfitting and early stopping

Prior to investigating temporal information with GMMs and HMMs, we discuss some issues related to model complexity and overfitting in these two techniques. Recall

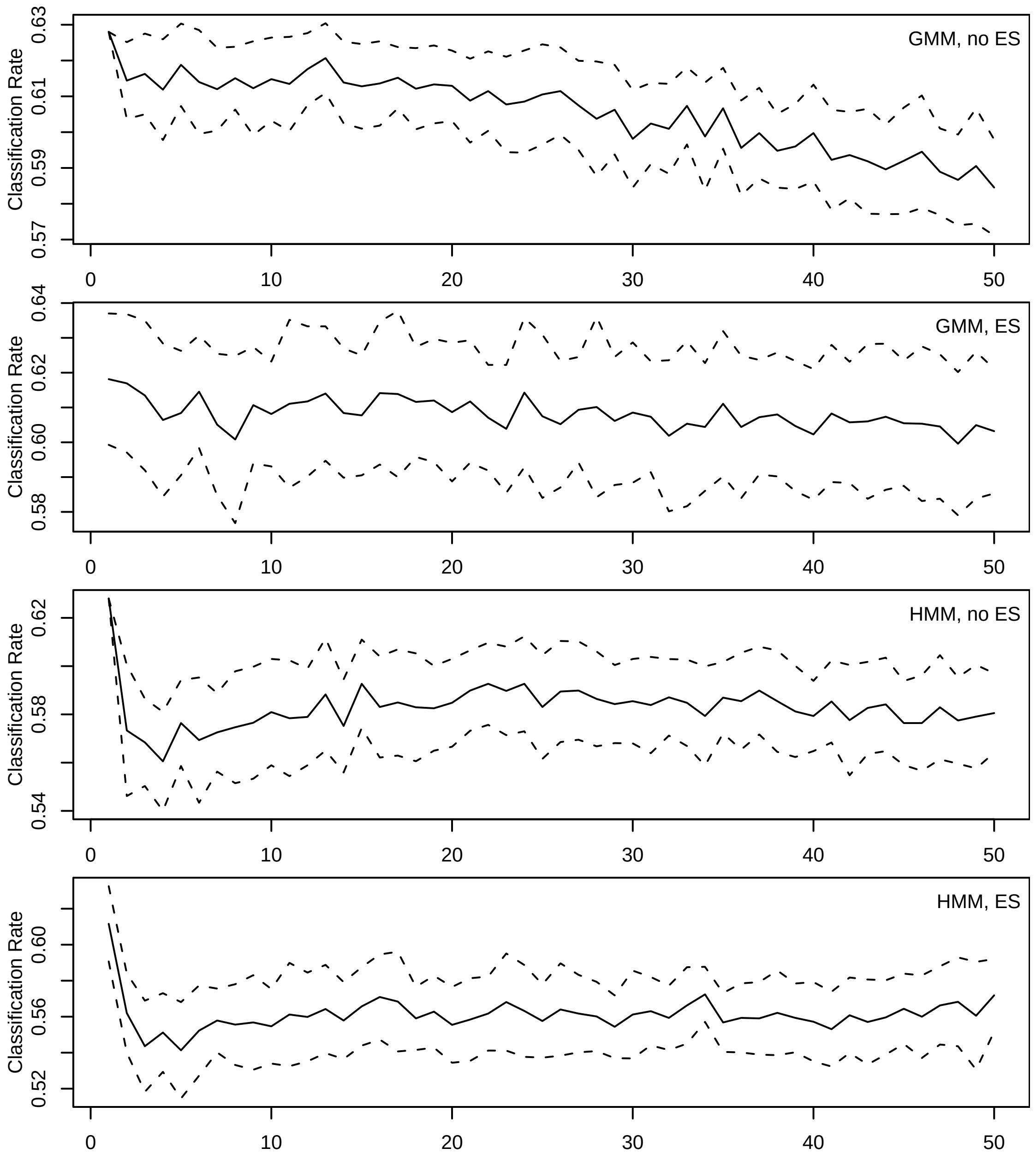

Figure 5.6: Five-fold cross-validation classification accuracy plotted versus the number of clusters for Bayesian classifiers based on Gaussian mixture models (GMM) and hidden Markov models (HMM) with and without early stopping (ES). The solid lines display the mean across 30 runs and the dashed lines are 1 standard deviation away from them.

that learning a model with latent variables involves running Expectation Maximization (EM), which is an iterative algorithm. The EM algorithm guarantees that the likelihood of the model given the training data will be non-decreasing as the iterations progress, potentially leading to overfitting. In addition to the number of EM iterations the final model fit also depends on the number of states (or clusters). A larger number of states leads to more flexibility but also to a larger potential for overfitting. In this section, I investigate the effects of early stopping in an attempt to address the overfitting problem.

Figure 5.6 displays five-fold cross-validation results on EEG data from the first recording session of the first subject. Training was performed on four of the trials in the session and the resulting classifier was applied to the fifth trial. Rather than working with the samples directly, each trial was split up into ten one-second windows. Each window is then treated as a single observation sequence, yielding a total of 200 observation sequences (5 tasks $\times$ 4 trials $\times$ 10 windows) for training and 50 observations sequences for testing in each of the cross-validation folds. A single label is assigned to each observation sequence in the test data and the classification rate is computed as the number of sequences classified correctly. The primary reason for working with observation sequences, or windows, rather than with individual observations is that hidden Markov models take into consideration transitions between adjacent samples. If individual samples are modeled, the hidden Markov model representation is no different from the Gaussian mixture models because no transitions are being considered. To investigate the differences between the two modeling techniques one really has to consider multiple samples. Furthermore, the GMM modeling does not consider transitions between subsequent one-second windows.

The first graph in Figure 5.6 displays results in the form mean $\pm$ st.d (computed over 30 runs) for Gaussian mixture models with the number of Gaussians in the mixture varying between 1 and 50. Five-fold cross-validation was performed in each of the 30 runs. There is a steady decrease in the average classification accuracy as the number of clusters increases, which is a clear evidence of overfitting. Modeling with a larger number of Gaussians allows one to approximate arbitrary data distributions more accurately. In the limit, when the number of Gaussian clusters is equal to the number of data points, the training data is captured perfectly. The down side of working with a large number of clusters is the lack of generalization of the resulting model to other data from the same class. Since we are interested in a good general description of each class, we become limited to using a small number of clusters. In the case depicted by the first graph of Figure 5.6, using more than 13 clusters/class appears to be undesirable.

The second graph in Figure 5.6 demonstrates the effects of early stopping in the context of Gaussian mixture models. To implement early stopping, 8 of the 40 training observation sequences are chosen uniformly at random for each task and are withheld for validation. The EM algorithm terminates when the log-likelihood measure given the validation data begins to decrease. As seen from the graph, early stopping effectively reduces the amount of overfitting, particularly in the case with a higher number of clusters. The classification accuracy now tends to remain more uniform as the number of Gaussian clusters varies between 1 and 50. This suggests that, regardless of what the number of clusters is chosen to be, the use of the validation data ensures that the model remains general enough to accurately describe the entire class, not just one particular instance of a dataset from that class. Recall, from the “Methods” chapter, that the covariance matrices for each cluster are initialized to be the covariance matrix of the entire dataset. As iterations of the EM algorithm progress, entries in the covariance matrices decrease and each Gaussian cluster “zooms in” on one particular part of the training data. Early stopping prematurely terminates the training and results in the covariance matrices remaining large. While this results in lower likelihood of the model given the training data, early stopping allows the resulting model to capture new data that does not quite overlap with the training data but instead lies in its proximity. This behavior is invariant to the number of clusters, hence the nearly horizontal line in the second graph of Figure 5.6.

The bottom two graphs in Figure 5.6 display the effects of early stopping on training hidden Markov models. Comparing these graphs with the GMM results, we note a few striking differences. First, the best classification accuracy is achieved by using a single hidden state. The behavior of a single-state HMM is identical to a single-cluster GMM, which in turn is identical to using QDA. Moreover, as the number of states is increased to 2, we observe a dramatic negative jump in performance. The classification accuracy related to a two-state HMM is significantly lower than its two-cluster GMM counter part. This suggests that by learning the state transitions in the training data, an HMM actually overfits that data. Such a result is quite surprising as one would expect that state transitions somehow encapsulate local temporal information which, based on our earlier experiments, is potentially useful to accurate classification. Observing a decrease in the classification accuracy we conclude that a first order Markov process may not be the right tool for capturing the dependence of adjacent samples (either in the original or in time-lag space). The second difference between GMM and HMM results lies in the apparent lack of overfitting when a large number of states is used in an HMM. The curve in the third graph (HMM, no early stopping) appears, for the most part, horizontal while the decrease in the classification accuracy can be clearly observed in the first graph (GMM, no early stopping). Moreover, employment of early stopping appears to hurt more than it helps, an observation that becomes the third difference between GMM and HMM results. The average performance drops by about 2% when early stopping is introduced. Both the apparent lack of overfitting for models with a high number of states and the decrease in performance attributed to early stopping, are currently very puzzling and require further investigation.

Lower classification rates in the bottom two graphs of Figure 5.6 do not speak in favor of hidden Markov models. Based on the results of this section, we are more inclined to use Gaussian mixture models. Not only do GMMs yield higher classification rate, the EM algorithm calculations are also simpler because we don't need to reestimate a transition matrix. In the following section, I investigate how these classification rates are affected when time-lag embedding is introduced into the picture. Because early stopping seems to hurt HMM performance, it was not employed in the following experiments. To offset the effects of overfitting, however, I used a small number of Gaussian clusters/states.

5.5 Global Temporal Structure

Time-lag embedding introduces a grouping to adjacent samples in a time series. This grouping allows us to analyze the local temporal structure present in the original time series by computing the correlation of dimensions in the new transformed space. In this section, I investigate whether it is beneficial to model transitions from one time-lag sample to the next one. Modeling these transitions is a way to capture global temporal structure, i.e., temporal dependence among groups that already encapsulate local temporal information.

More specifically, I model the time embedded samples with a mixture of Gaussians and then introduce a first order Markov process to model the switching from one Gaussian in the mixture to another as samples are being observed. As described in the "Methods" chapter, this is done using Gaussian mixture models and hidden Markov models. Similar to the previous experiments with these techniques, the training and classification is performed on one-second windows rather than individual samples (again, due to the fact that an HMM is modeling the transitions between samples and therefore requires multiple samples to work with). The lagging is applied on a per-window basis. In other

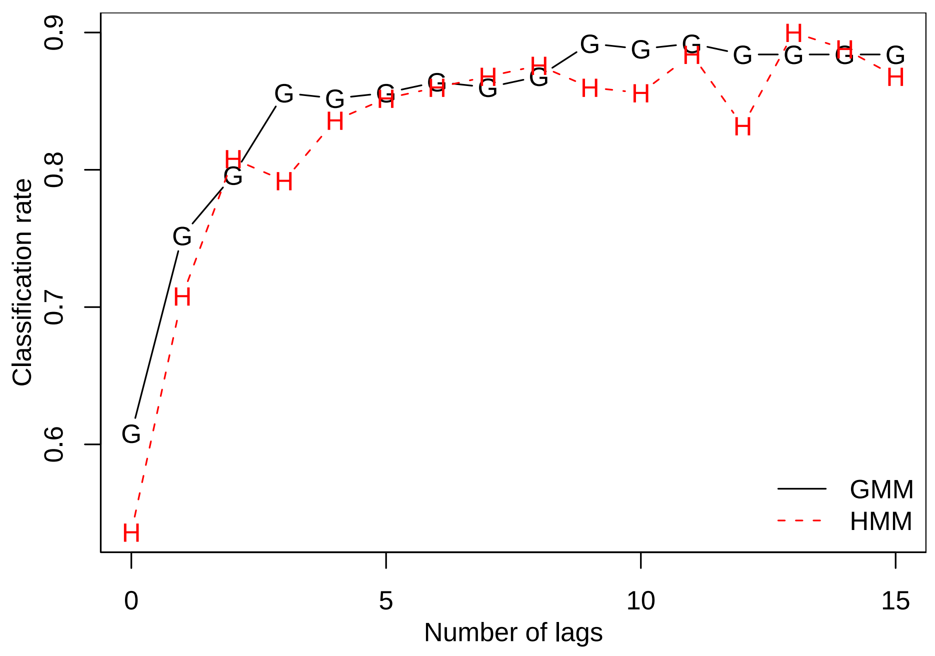

Figure 5.7: Classification accuracy plotted against the number of lags for classifiers based on five-state hidden Markov models (HMM) and five-cluster Gaussian mixture models (GMM). The results were computed as five-fold cross-validation using the first recording session of the first subject.

words, when a delay of k lags is requested each $250 \times p$ window of observations is transformed into a $(250 - k) \times (kp + p)$ observation sequence with no samples intermixed across windows. The classification rate is computed as the percent of windows, or observation sequences, being classified correctly.

Figure 5.7 demonstrates how the classification rate is affected by the number of lags for GMMs and HMMs. These results are for the first session of the first subject. As before, the classifier was trained on windows of four trials in the session and classification was performed on windows in the remaining fifth trial. A model with five Gaussian clusters/states was used for all GMMs and HMMs with no early stopping employed.

Based on the results presented in Figure 5.7, using a hidden Markov model offers no clear advantage to its counterpart algorithm that only models spatial structure. To validate this conclusion, we examine performance of GMMs and HMMs on the data

Subj./	GMM		HMM	
Sess.	0 lags	5 lags	0 lags	5 lags
1/1	0.608	0.888	0.536	0.856
1/2	0.712	0.936	0.636	0.920
2/1	0.592	0.856	0.552	0.836

Table 5.6: Five-fold cross-validation results for classifiers based on five-cluster Gaussian mixture models (GMM) and five-state hidden Markov models (HMM). Data from subjects 1 and 2 was used.

from other recording sessions and other subjects. Table 5.6 presents classification rates for the data from subjects 1 and 2. Unfortunately, time-lag data from subjects 6 and 7 contains nasty outliers that result in numerical instability of the EM algorithm when attempting to learn a GMM or an HMM. The likelihood of a model when given such an outlier comes out to be 0 resulting in a division by 0 in the M step of the algorithm. Removing outliers from the dataset could potentially result in a bias when learning an HMM because the model relies on transitions from one sample to the next. For this reason, results for subjects 6 and 7 are not included.

Nevertheless, Table 5.6 confirms our earlier conclusion that hidden Markov models offer no clear improvement in performance over Gaussian mixture models. In fact, it appears that classification rates for HMMs are lower but multiple runs would be required to conclude whether this difference is statistically significant. A potential explanation for this lack of performance increase is that a first order Markov process does not capture information specific to a particular mental task and the transitions learned during training do not offer any additional discriminative information in the Bayesian framework. There is a very particular structure to the transitions between neighboring time-delay samples. More specifically, suppose that k lags were applied to a d-dimensional signal resulting in samples of dimensionality $kd + d$. Then, by construction, entries $d + 1$ through $kd + d$ in a time embedded sample will be identical to entries 1 through kd in the immediately following sample. Thus, the transitions between neighboring samples can be modeled as a reflection about a 45-degree plane in kd dimensions followed by a translation in the other d dimensions. For example, consider four 2-D samples $[x_1, y_1]^T$, $[x_2, y_2]^T$, $[x_3, y_3]^T$, and $[x_4, y_4]^T$. Apply two lags results in two samples embedded in $2\times2+2 = 6$ dimensional space, $[x_1, y_1, x_2, y_2, x_3, y_3]^T$ and $[x_2, y_2, x_3, y_3, x_4, y_4]^T$, with the following transitional relationship:

$$\begin{bmatrix} x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ x_4 \\ y_4 \end{bmatrix}, \tag{5.5}$$

which can be recognized as a reflection about a 45-degree plane in 4-D and an additional translation in dimensions 5 and 6. Once the number of lags is specified, the matrix in Equation (5.5) is fixed not only across samples in one particular sequence but also across sequences, including across different mental tasks. Therefore, learning the matrix will not aid in classification and translation is the only place where any discriminative information may be present. However, the translation vector simply contains a sample of the original time series (in our example, the fourth sample of the original 2-D series). Learning the transition information from translation is, therefore, equivalent to learning the transition information from the original time series, which, as we saw from Figure 5.6 results in overfitting. This explains the apparent lower performance of HMM-based classifiers in Table 5.6.

The surprising result from Figure 5.7 and Table 5.6 is that 5 lags seems to yield a much higher increase in performance when compared to our earlier experiments with LDA and QDA. This may be due to the following two reasons. First, a move from a single Gaussian to using a mixture of five Gaussians to represent each class yields more flexibility in the Bayesian decision boundary. Second, by considering one-second windows rather than individual samples we are able to make a classification decision

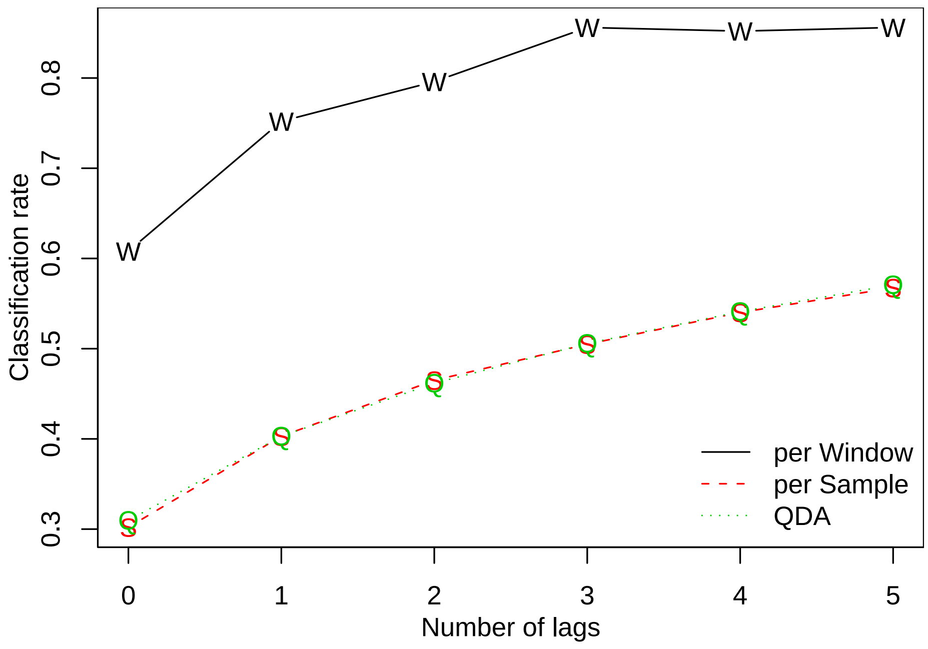

Figure 5.8: Classification accuracy plotted against the number of lags for a GMM-based classifier applied to one-second windows (W) and individual samples (S). Five clusters were used. QDA performance is included for comparison. The data used to generate this plot was the first five trials from subject 1.

based on multiple samples at once.

Figure 5.8 shows that using five Gaussians instead of one to model each class yields virtually no increase in performance, while using one-second windows doubles the classification accuracy, even when no lagging is performed. Recall, from our discussion in Section 5.1, that when adjacent samples are independent, the product of probabilities of those samples is equal to probability of the time-lag vector that they make up. The Bayesian posterior probability of each sample in a one-second window is calculated separately but the classification decision is made based on the product of those probabilities, achieving an effect similar to time-lag embedding. This result demonstrates an alternative to time-lag embedding when multiple samples need to be considered to make a Bayesian classification decision.

The advantage of windowing is the ability to work in the original lower-dimensional space without worrying about having enough samples to get a full-rank covariance matrix for every Gaussian. The disadvantage is the independence assumption that allows us to simply multiply posterior probabilities without considering any potential correlation between neighboring samples. Classification accuracy of the windowing method is further improved when it is combined with lagging. Recall that in the experiments described here, lagging was performed on a per-window basis, implying that no additional samples need to be included in each window and, if working with an online BCI, the decision can still be made every second. Lagging the samples in a window, however, drastically improves the classification accuracy (as demonstrated in Figure 5.7 and Table 5.6) even with as few as five lags. Combining windowing and lagging allows us to capture correlation between neighboring samples while maintaining low dimensionality of our computations.

5.6 Analysis of Normality

The single most important assumption about the spatial distribution of data in all the modeling techniques discussed in this thesis is the assumption of Normality. The data is modeled with one or more Gaussian components, which may or may not result in accurate abstraction. The set of experiments in this section attempts to measure the Normality of EEG data, specifically in the context of Gaussian mixture models. All calculations necessary to obtain these results have already been described in the "Methods" chapter.

Prior to experimenting with mixtures of Gaussians, we measure how well a single Gaussian fits the data. Table 5.7 displays the Q values (computed using Equation (3.21)) for the entire Keirn dataset. All five trials in a session were used to compute the Gaussian fit for each task. Recall that Q is the mean of squared deviations between the empirical distribution function and the true Gaussian cumulative distribution function. Therefore,

Subj./Sess.	Task				
Sess.	1	2	3	4	5
1/1	0.192	8.126	0.637	0.229	0.594
1/2	0.328	1.782	0.070	0.131	0.699
2/1	0.633	0.866	0.193	0.138	1.406
6/1	3.075	2.017	1.054	0.116	0.660
6/2	3.121	2.149	0.616	0.099	0.427
7/1	0.178	0.578	0.970	1.072	0.061

Table 5.7: Mean squared deviation between empirical and true distribution functions for the entire Keirn dataset.

lower values of Q signify better fit of a single Gaussian to the data.

There are some trends that can be observed in Table 5.7. There appears to be some consistency across multiple recording sessions of the same subject. For instance, tasks 1 and 2 are poorly fit with a single Gaussian in both sessions of subject 6 data. On the other hand, a single Gaussian captures task 1 fairly well for subject 1. We are also able to conclude that some tasks are inherently easier to model with one Gaussian that others. In particular, the Q values for task 4 are lower than for any other task and 8.126 is the highest value in the table, suggesting that it is more difficult to capture task 2 with one Gaussian than task 4.

Having measured the model fit for a single Gaussian, we move on to investigate how the fit changes as the number of Gaussian components is increased. Figure 5.9 displays the Q measure for the first session of the subject 1 data as the number of components ranges between 2 and 20. Recall that to assess Gaussian fit of the entire mixture model, we compute Q values for each cluster individually and then report the maximum of those values. The individual Q values depend on the Gaussian mixture model learned from the data and, therefore, will vary depending on the initialization of the EM algorithm. Due to this non-determinism, the curves presented in Figure 5.9 are the mean $\pm$ standard deviation computed across 30 runs, with a separate GMM trained for each trial. The

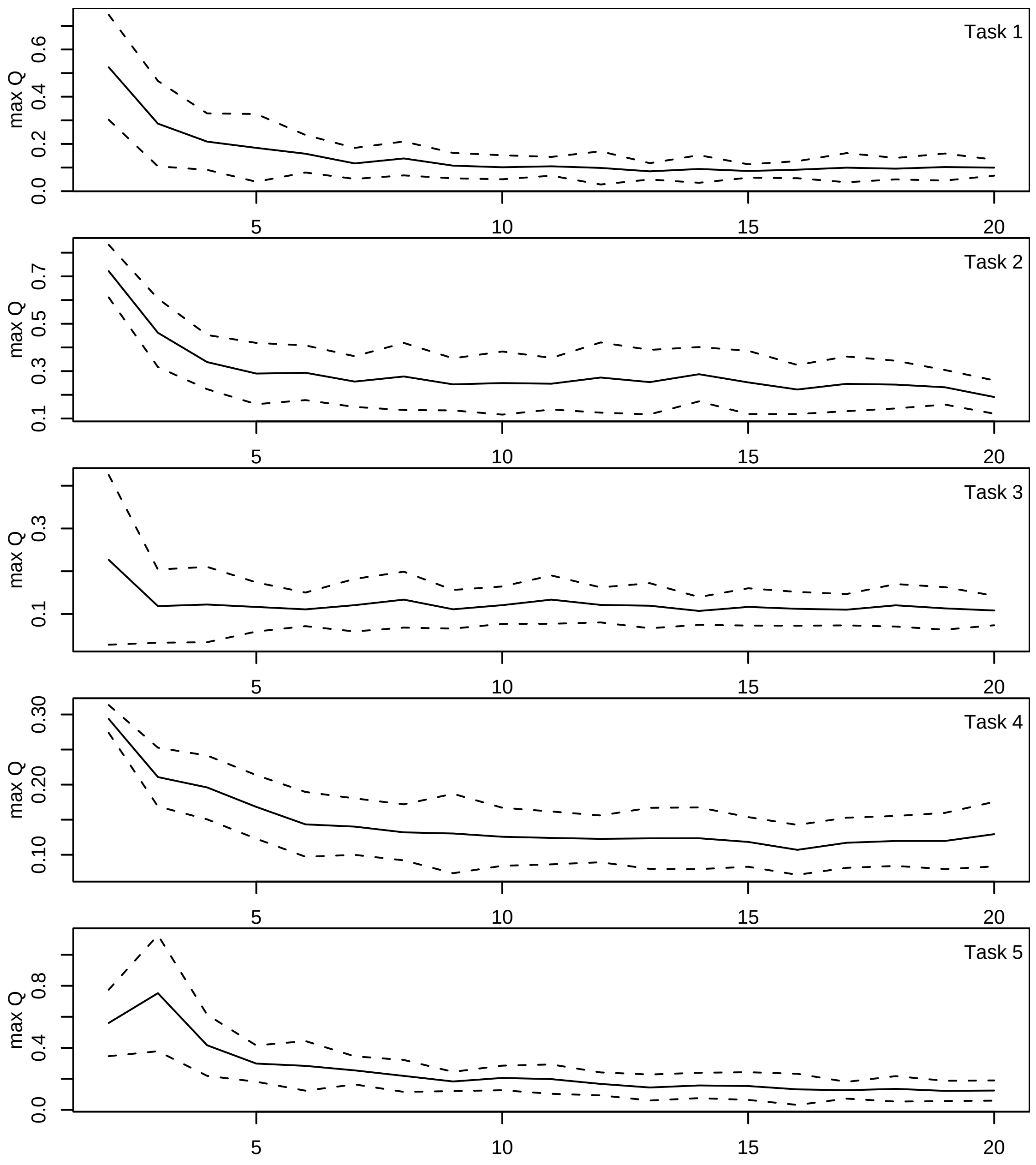

Figure 5.9: Mean squared deviation between empirical and true distribution functions computed across 30 runs. A separate GMM was trained in each run.

Number of clusters	Classification accuracy	
	0 lags	5 lags
(5,16,2,3,9)	0.622 ± 0.011	0.839 ± 0.006
2	0.615 ± 0.012	0.861 ± 0.006
3	0.616 ± 0.013	0.861 ± 0.006
5	0.621 ± 0.011	0.861 ± 0.007
9	0.620 ± 0.010	0.868 ± 0.007
16	0.613 ± 0.012	0.864 ± 0.009

Table 5.8: Dependence of classification rates on the number of clusters used to model each mental task. Results were computed across 30 runs and are presented in (mean ± st.d.) form. Five fold cross-validation was performed in each of the 30 runs. Data from the first five trials of subject 1 was used. Presented are results for 0 and 5 lags.

model fit becomes better as the number of Gaussians is increased, which is expected since any spatial distribution can be approximated with a mixture of Gaussian with arbitrary accuracy by simply introducing more Gaussians into the mixture. Figure 5.9 shows that accuracy of approximation grows at different rates for different mental tasks. For instance, the model fit for task 3 improves much faster than the model fit for task 4 as the number of Gaussian is increased from 2 to 5.

The analysis of normality presented in this section demonstrates an alternative to using a validation set when it comes to model selection. To make use of this analysis, one first has to decide on the desired accuracy level (e.g., $Q = 0.2$) and then determine the number Gaussians necessary to model each mental task with that accuracy. For example, using Figure 5.9 we may conclude that we want to use five clusters to model task 1, sixteen clusters to model task 2, two clusters to model task 3, three clusters to model task 4, and nine clusters to model task 5, all with accuracy $Q = 0.2$.

Being able to perform model selection for each class independently is an advantage specific to the analysis of normality. From this point of view, cross-validation is a worse alternative due to the combinatorial explosion that arises when each class demands its own value of a parameter. On a dogmatic note, model selection via analysis of normality falls under the generative approach to classification (such as Bayesian classification), where training considers data from one class at a time. To examine the advantage of each class having its own number of clusters, consider Table 5.8. Presented classification rates compare having a variable number of clusters for each class (row 1 in Table 5.8) to the case where the number of clusters is fixed across all classes (rows 2 through 6 in Table 5.8). For the first experiment, the number of clusters for mental tasks 1 through 5 was five, sixteen, two, three, and nine, respectively. These were chosen by observing which number of clusters yields an average Q value of 0.2 in Figure 5.9. The number of clusters for all other experiments was fixed across classes with the value displayed in the first column of Table 5.8. The statistics presented in Table 5.8 were computed over 30 runs and are presented in (mean $\pm$ st.d.) form. Table 5.8 demonstrates that using a different number of clusters to model each task yields higher classification accuracy (although not statistically significant) than keeping the number of clusters fixed across classes.

The last column in Table 5.8 displays results from a similar set of experiments, but with 5 lags applied to the data. While the classification rates go up due to consideration of multiple samples by the classifier, the number of Gaussians for each GMM selected by the analysis of Normality no longer yields a competitive result. This demonstrates the need to perform analysis of Normality in the lagged space if the lagged samples are to be modeled.

An interesting empirical result can be obtained using analysis of normality. Proposition 1 states that when two samples are independent, their time-lag embedding will preserve normality. Figure 5.10 shows that it is impossible to generate two independent samples using a computer. The Q measure is plotted for ten thousand samples generated "independently" from a Gaussian distribution, lagged, and projected onto the main principal component. The lagging picks up on temporal structure of these "independent"

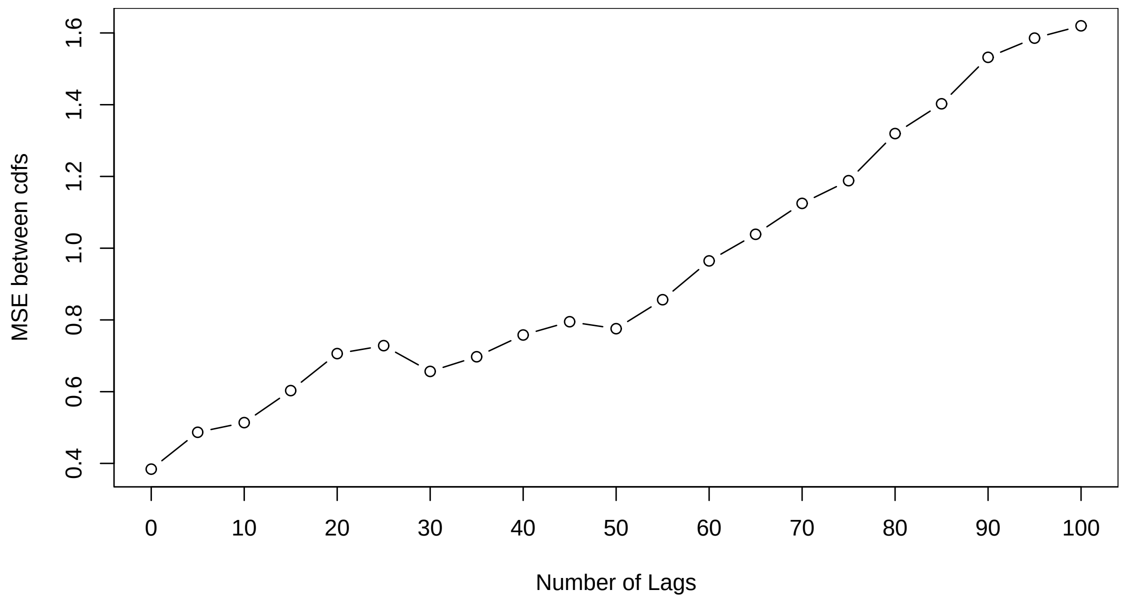

Figure 5.10: Mean squared deviation between empirical and true distribution functions computed across 100 runs and plotted against the number of lags. For each run, 10000 “independent” samples were generated from a Normal distribution.

samples by creating a geometric representation of the temporal correlation of neighboring samples. As demonstrated by Figure 5.10, this geometric representation loses the original normality as more and more samples are used in the embedding, suggesting that we are picking up on the bias of the pseudo-random number generator.

To further support this result, we measure correlation between dimensions of the lagged space. Figure 5.11 displays the correlation matrix computed across 9900 samples in 101 dimensions. The samples were obtained by applying 100 lags to 10000 one-dimensional “independent” samples generated from the standard Normal distribution. The diagonal of the matrix was set to 0 to better demonstrate relative magnitudes of the off-diagonal entries. Darker values in Figure 5.11 correspond to higher correlation (either positive or negative). The diagonal patterns represent correlation between samples i and $i + k$, where k is the offset of the pattern from the main diagonal. The largest correlation is observed between samples i and $i+30$ and is measure to be 3.45%.

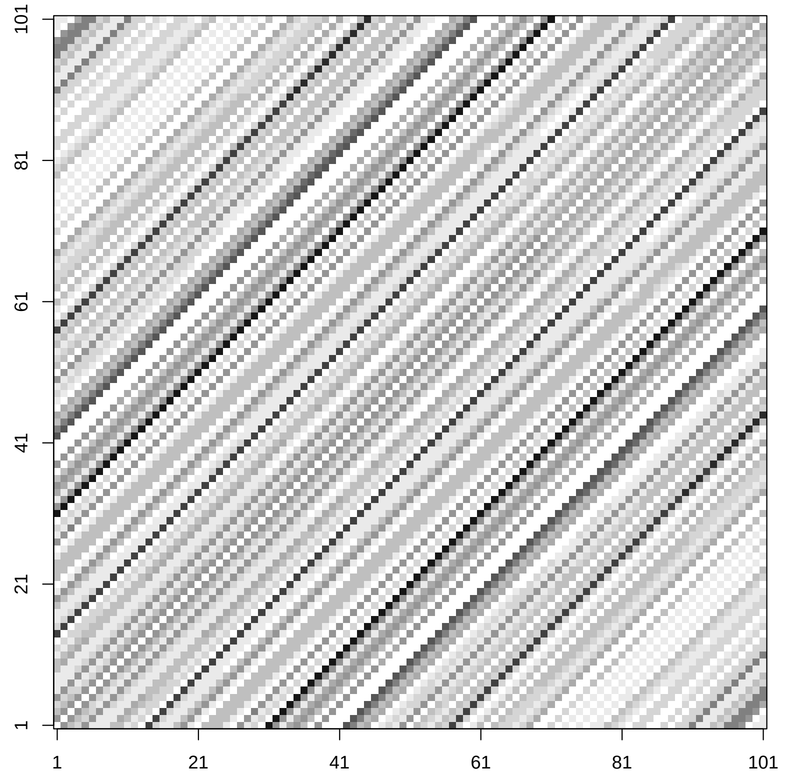

Figure 5.11: Absolute value of correlation between dimensions in a 101-dimensional space created through lagging a series of samples. The samples were generated "independently" from a standard Normal distribution.

This suggests that there is indeed some correlation between samples in the series.

Chapter 6

Conclusion

this thesis explored temporal structure and normality of EEG data in the context of Bayesian classification. By using time-lag embedding of a time series, it was demonstrated that classification accuracy increases when multiple adjacent samples are used to make the classification decision. It was also shown that correlation between neighboring samples creates geometry in a high-dimensional space that either increases or decreases the classification accuracy when compared to the case where the samples are independent. More recording sessions per subject are required to determine if the geometric difficulty is subject-dependent.

The second part of this thesis demonstrated that adjacent samples in a time-lag space are related through a reflection about a 45-degree plane, followed by a translation. Empirical results showed that a first order Markov process is not able to capture this relationship in such a way that it can be used for mental task discrimination. We are unable to conclude whether there is discriminative information in this correlation and other models need to be explored. It was also demonstrated that by windowing, i.e., multiplying posterior probabilities of adjacent samples to make the classification decision, we are able to obtain higher classification accuracy. Combining windowing with lagging yielded even further improvement.

The final part of this thesis demonstrated how to make use of normality assessment to perform model selection without having validation data. Different mental tasks require a different number of Gaussians to achieve the same accuracy of the model fit and an example was given regarding how to perform model selection on EEG data using analysis of normality. Analysis of normality was argued to fit better with the Bayesian framework due to its consideration of training data from one class at a time.

The theoretical derivations in this report showed what happens to the Bayesian decision boundary in LDA and QDA classifiers as the number of dimensions increases. Further derivation may be necessary to show that when the $(p+1)$-dimensional linear decision boundary contains the p-dimensional boundary, the classification rate will not decrease.

REFERENCES

[1] *The Third International Meeting on Brain-Computer Interface Technology: Making a Difference*, Rensselaerville, NY, June 2005.

[2] C. Anderson and Z. Sijercic. Classification of EEG signals from four subjects during five mental tasks. In *Intl. Conf. on Engineering Applications of Neural Networks*, pages 407–414, 1996.

[3] C. W. Anderson, E. A. Stolz, and S. Shamsunder. Multivariate autoregressive models for classification of spontaneous electroencephalogram during mental tasks. *IEEE Transactions on Biomedical Engineering*, 45(3):277–286, 1998.

[4] Charles Anderson and Michael Kirby. EEG subspace representations and feature selection for brain-computer interfaces. In *CVPR*, 2003. CD-ROM.

[5] Charles W. Anderson, James N. Knight, Tim O'Connor, Michael J. Kirby, and Artem Sokolov. Geometric subspace methods and time-delay embedding for EEG artifact removal and classification. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):142–146, 2006.

[6] John E. Angus. The probability integral transform and related results. *SIAM Review*, 36(4):652–654, December 1994.

[7] Jose M. Bernardo and Adrian F. Smith. *Bayesian Theory*. New York: Wiley, 1994.

[8] Jeff A. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report TR-97-021, U.C. Berkeley, 1998.

[9] Benjamin Blankertz, Gabriel Curio, and Klaus-Robert Muller. Classifying single trial EEG: Towards brain computer interfacing. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems, vol. 14*, pages 157–164, Cambridge, MA, 2002. MIT Press.

[10] Benjamin Blankertz, Guido Dornhege, Matthias Krauledat, Klaus-Robert Muller, Volker Kunzmann, Florian Losch, and Gabriel Curio. The berlin brain-computer interface: EEG-based communication without subject training. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):147–152, 2006.

[11] Benjamin Blankertz, Klaus-Robert Muller, Dean J. Krusienski, Gerwin Schalk, Jonathan R. Wolpaw, Alois Schlogl, Gert Pfurtscheller, Jose del R. Millan, Michael Schroder, and Niels Birbaumer. The BCI competition iii: Validating alternative approaches to actual BCI problems. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):147–152, 2006.

[12] Silvia Chiappa and David Barber. Generative independent component analysis for EEG classification. In *European Symposium on Artificial Neural Networks*, pages 297–302, 2005.

[13] Silvia Chiappa and Samy Bengio. HMM and IOHMM modeling of EEG rhythms for asynchronous BCI systems. In *European Symposium on Artificial Neural Networks, ESANN*, pages 193–204, 2004.

[14] Febo Cincotty, Luigi Bianchi, Gary Birch, Christoph Guger, Jurgen Mellinger, Reinhold Scherer, Robert N. Schmidt, Oscar Yanez Suarez, and Gerwin Schalk. BCI meeting 2005 — workshop on technology: Hardware and software. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):128–131, 2006.

[15] Stephen D. Cranstoun, Hernando C. Ombao, Rainer von Sachs, Wensheng Guo, and Brian Litt. Time-frequency spectral estimation of multichannel EEG using the Auto-SLEX method. *IEEE Transactions on Biomedical Engineering*, 49(9):988–996, September 2002.

[16] R.B. D'Agostino and M.A. Stephens. *Goodness-of-Fit Techniques*. Marcel Dekker, Inc., 1986.

[17] J. del R. Millan, F. Renkens, J. Mourino, and W. Gerstner. Brain-actuated interaction. *Artifical Intelligence*, 159(1-2):241–259, 2004.

[18] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[19] A.K. Engel, C.K. Moll, I. Fried, and G.A. Ojemann. Invasive recordings from the human brain: clinical insights and beyond. *Nature Reviews Neuroscience*, 6(1):35–47, January 2005.

[20] Gary N. Garcia, T. Ebrahimi, and J-M. Vesin. Support vector EEG classification in the fourier and time-frequency correlation domains. In *Proceedings of the IEEE-EMBS First International Conference on Neural Engineering*, pages 591– 594, 2003.

[21] G. Hamerly and C. Elkan. Learning the k in k-means. In L.K Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17, Cambridge, MA, USA, 2003. MIT Press.

[22] Robin Hanson, John Stutz, and Peter Cheeseman. Bayesian classification theory. Technical Report FIA-90-12-7-01, NASA Ames Research Center, 1991.

[23] Ernst Haselsteiner and Gert Pfurtscheller. Using time-dependent neural networks for EEG classification. *IEEE Transactions on Rehabilitation Engineering*, 8(4):457–463, December 2000.

[24] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[25] Herbert H. Jasper. Report of the committee on methods of clinical examination in electroencephalography. *Electroencephalography and clinical neurophysiology*, 10(2):370–375, May 1958.

[26] I. T. Jolliffe. *Principal Component Analysis*. New York, Springer-Verlag, 1986.

[27] Zachary A. Keirn. Alternative modes of communication between man and machine. Master's thesis, Purdue University, 1988.

[28] Hyun Taek Kim, Bo Yeon Kim, Eun Hye Park, Jong Woo Kim, Eui Whan Hwang, Seung Kee Han, and Sunyoung Cho. Computerized recognition of alzheimer disease-EEG using genetic algorithms and neural network. *Future Generation Computer Systems*, 21(7):1124–1130, 2005.

[29] Michael Kirby and Charles W. Anderson. Geometric analysis for the characterization of nonstationary time-series. *Springer Applied Mathematical Sciences Series Celebratory Volume for the Occasion of the 70th Birthday of Larry Sirovich*, pages 263–292, 2003.

[30] G.H. Klem, H.O. Luders, H.H. Jasper, and C. Elger. The ten-twenty electrode system of the international federation. *Electroencephalogr Clin Neurophysiol Suppl.*, 52:3–6, 1999.

[31] James N. Knight. Signal fraction analysis and artifact removal in EEG. Master's thesis, Colorado State University, Department of Computer Science, 2003.

[32] A. Kubler, V. K. Mushahwar, L. R. Hochberg, and J. P. Donoghue. BCI meeting 2005 — workshop on clinical issues and applications. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):131–134, 2006.

[33] Mauricio Kugler and Heitor Silverio Lopes. Using a chain of LVQ neural networks for pattern recognition of EEG signals related to intermittent photic-stimulation. In *Proceedings of VII Brazilian Symposium on Neural Networks*, pages 173–177, 2002.

[34] T.N. Lal, T. Hinterberger, G. Widman, M. Schroder, N.J. Hill, W. Rosenstiel, C.E. Elger, B. Scholkopf, and N. Birbaumer. Methods towards invasive human brain computer interfaces. In L.K Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 737–744, Cambridge, MA, 2005. MIT Press.

[35] Hyekyung Lee and Seungjin Choi. PCA+HMM+SVM for EEG pattern classification. In *Proceedings IEEE International Symposium on Signal Processing and its Applications*, pages 541–544, Paris, France, 2003.

[36] Stephane Mallat. *A Wavelet Tour of Signal Processing, by Stphane Mallat*. Academic Press, 1999.

[37] Dennis J. McFarland, Charles W. Anderson, Klaus-Robert Muller, Alois Schlogl, and Dean J. Krusienski. BCI meeting 2005 — workshop on bci signal processing: Feature extraction and translation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):135–138, 2006.

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

[39] N. Nicolaou and S.J. Nasuto. Temporal independent component analysis for automatic artefact removal from EEG. In *2nd International Conference on Medical Signal and Information Processing*, 2004.

[40] B. Obermaier, G.R. Muller, and G. Pfurtsheller. ”virtual keyboard” controlled by spontaneous EEG activity. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11(4):422–426, 2003.

[41] Bernhard Obermaier, C. Guger, Ch. Neuper, and Gert Pfurtscheller. Hidden markov models for online classification of single trial EEG data. *Pattern Recognition Letters*, 22(12):1299–1309, 2001.

[42] Marius Ooms. *Empirical Vector Autoregressive Modeling*. Springer-Verlag, 1994.

[43] H. Peitgen, H. Jurgens, and D. Saupe. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, 1992.

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

[45] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

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

[47] Sam Roweis and Zoubin Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–345, February 1999.

[48] Vitaly Schetinin. Polynomial neural networks learnt to classify EEG signals. Technical Report NIMIA-SC2001, NATO Advanced Study Institute on Neural Networks for Instrumentation, Measurement, and Related Industrial Applications, 2001.

[49] M.A. Stephens. EDF statistics for goodness of fit and some comparisons. *American Statistical Association*, 69(347):730–737, September 1974.

[50] Peter Sykacek and Stephen Roberts. Bayesian time series classification. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 937–944, Cambridge, MA, 2002. MIT Press.

[51] Peter Sykacek, Stephen Roberts, M. Stokes, E. Curran, M. Gibbs, and L. Pickup. Probabilistic methods in BCI research. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11(2):192–195, June 2003.

[52] Floris Takens. Detecting strange attractors in turbulence, dynamical systems and turbulence. In D. A. Rand & L. S. Young, editor, *Lecture Notes in Mathematics*, volume 898, pages 366–381. Springer, 1981.

[53] Leonard J. Trejo, K.R. Wheeler, C.C. Jorgensen, R. Rosipal, S.T. Clanton, B. Matthews, A.D. Hibbs, R. Matthews, and M. Krupka. Multimodal neuroelectric interface development. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11(2):199–204, 2003.

[54] J. R. Wolpaw, H. Ramoser, D. J. McFarland, and G. Pfurtscheller. EEG-based communication: improved accuracy by response verification. *IEEE Transactions Rehabilitation Engineering*, 6(3):326–333, 1998.

[55] Jonathan R. Wolpaw, Gerald E. Loeh, Brendan Z. Allison, Emanuel Donchin, Omar Feix do Nascimento, William J. Heetderks, Femke Nijboer, William G. Shain, and James N. Turner. BCI meeting 2005 — workshop on signals and recording methods. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14(2):138–141, 2006.

[56] Wenjie Xu, Cuntai Guan, Chng Eng Siong, S. Ranganatha, M. Thulasidas, and Jiankang Wu. High accuracy classification of EEG signal. In *Proceedings of the 17th International Conference on Pattern Recognition*, pages 391 – 394, 2004.

[57] Shi Zhong and Joydeep Ghosh. HMMs and coupled HMMs for multi-channel EEG classification. In *Proc. of IEEE Int. Joint Conf. on Neural Networks*, pages 1154–1159, Honolulu, Hawaii, 2002.