3rd Year Project Final Report

SleepAppnea

George Cochrane, Sachin Mylavarapu, Tuan Anh Le, Sophie Louth

April 21, 2014

Abstract

The abstract text goes here.

Introduction

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Medical Information

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Rationale

3.1 Economic Rationale

3.1.1 Current Solutions

As is evident, sleep apnea, particularly obstructive sleep apnea (OSA) affects a significant proportion of the population and is worthy of attention. The current available diagnosis options for sufferers or potential sufferers are limited, expensive, invasive and realtively inaccessible. We outline below the most common sources of help and diagnosis for potential sufferers:

- NHS - GP check-up, home devices and polysomnography (PSG)
- Commercial home-testing devices
- Other possible future options such as mobile sleep apps and non-contact health sensing technologies

NHS

According to the NHS guidelines, patients who believe they may be suffering from sleep apnea can schedule a visit to their GP. The GP will ask a few questions to determine the likelihood of apnea, along with a physical examination that includes a blood pressure (BP) test and a hypothyroidism test, to determine whether an underactive thyroid gland is the reason for the patient's tiredness. The patient can then choose if he/she would like to be observed for one night in a sleep centre, or would like to be given a monitoring device to wear at home when sleeping. Those who opt for a home sleep study are required to visit the sleep centre at a convenient time to collect and learn how to use the portable recording equipment. These include breathing sensors, heart rate monitors and oxygen sensors. Information from the device can be analysed on the next visit and further action, such as referring to a sleep centre can be taken.

Observation at a sleep centre is done through polysomnography (PSG). This involves electrodes being placed on the face, scalp and above the lips, and bands being placed around the chest and abdomen.

Additionally, sensors are placed on the legs and an oxygen sensor is attached to a finger. The tests carried out during a PSG include:

- Electro-encephalography (EEG)
- Electromyography (EMG)
- Recording of thoracoabdominal movements
- Recording of oronasal airflow
- Pulse Oximetry
- Electrocardiography (ECG)
- Sound and Video Recording

The data from these tests is used to positively diagnose obstructive sleep apnea and a treatment regimen is then decided upon and enforced by the healthcare professional. For the purposes of this report, we shall not delve into the treatment of OSA - we concern ourselves with the diagnosis process and how we can improve it.

Commercial home-testing devices

While these devices are more commmon in the United States, they are also available in the UK, and can be purchased if desired without visiting a GP. An example of such home-testing devices is the AccuSom® test from NovaSom Inc., pictured below:

Figure 3.1: AccuSom®

The prevalence of such tests in the U.S. points towards the increasing unfeasiblity of sleep centre tests for simple diagnosis of OSA, and is an indication of where the diagnosis process is headed in the future,

in the U.K. as well as around the world. The economic reasons for the trend are clear. On average, a single night at a sleep centre and the associated tests could cost from \$800 to \$3000 in the U.S. (figures for the NHS in the U.K. are more difficult to find, but should be comparable) The home tests can be administered at a fraction of that price, ranging from \$200 to \$600. Moreover, sleep centre tests are extremely inconvenient for the patients, and hence should be administered towards the later stages of the diagnosis process. The use of home-testing devices also allows a larger percentage of the population to be able to test for OSA, which is desirable given that around 1 in 5 suffer from OSA.

Other options

The trend towards greater user independence in OSA diagnosis is further observed through two major related breakthroughs - the rise of mobile sleep apps and the development of non-contact health sensing technologies.

The popularity of sleep monitoring apps in the iOS and Android App stores illustrates an increasing interest among smartphone users in their sleep patterns. Currently, such apps use the accelerometers and microphones to detect movements and noises from the users while they sleep, and use relatively simple algorithms to determine which stage of sleep the user is in. In addition, some apps come with additional headsets or headbands that track electrical impulses and measure the user's activity more accurately. The alarm clock function is activated only when the user is in light sleep (within a reasonable time window) to ensure he/she is woken up feeling energised. Some examples of such apps are:

- Sleep Cycle (iOS) - \$1
- Sleep Bot Tracker (Android) - Free
- Wakemate - \$60
- Lark - \$99
- Zeo Sleep Manager Mobile - \$99
- Sleep Tracker Elite - \$149

The development of non-contact health sensing technologies holds promise as well. Recently, Xerox and Manipal University Hospital in India announced a collaboration to develop such techonologies at Xerox's research centre in Bangalore. Using image and signal processing algorithms, the collaboration aims to determine health indicators in a non-invasive manner, and such that monitoring can take place remotely. The clear trend of health monitoring towards non-invasive methods and the increasing use

of algorithms to supplement health care and diagnosis efforts is set to gain even more momentum as healthcare professionals begin to embrace 'big data'.

3.1.2 Proposed App - Where it fits in

The need for an app that is able to diagnose OSA from a mobile platform, using non-invasive techniques is clear from the current diagnosis solutions and trends. While there has been a shift towards cheaper home-testing devices before necessitating a visit to a sleep centre, the fact remains that even at $200, these tests are not inexpensive. What if we could create a means for diagnosing OSA without the need for cumbersome and invasive sensors, that could be accessed by anyone with a smartphone, at a negligible cost compared to a home-testing system? Those who have doubts over their sleep habits, but are too busy for a GP visit and do not want to spend money unnecessarily on a home-testing system could try the app and move on to pursue the matter more seriously if the results from the app suggested a need to do so. Since machine learning algorithms will be used in the processing of the data, the app has the potential to continuously improve in accuracy and, eventually, could outperform the home-testing devices. The development of image processing technology that could remove the need for invasive monitoring, especially in neonatal care, by Manipal University Hospital and Xerox sets a precedent and raises the possibility of diagnosing OSA without the need for sensors being placed around the body.

3.1.3 Advantages of proposed app

The advantages of the proposed smartphone app, that will use machine learning algorithms to accurately diagnose OSA using non-invasive mobile phone sensors, are numerous. Firstly, such an app, if utilised as a precursor to home-testing systems and sleep centres, could result in significant cost savings for both the healthcare provider and the patient. From the point of view of the NHS, savings from performing polysomnographies only on those who really need it would be substantial. Moreover, some of the functions such as collecting information about the patient can be done through the app using questionnaires to save time for the doctors. From the point of view of the patients, not having to schedule visits to the GP for collecting and returning home-testing kits would result in a larger number of users for the app than would otherwise have been the case. This brings us to another advantage of using the mobile app - accessibility. Given that 65 % of people over 65 in the U.K. have OSA, along with a significant proportion of middle-aged men and women, and that the condition often goes undiagnosed, it is essential to reach out to as many users as possible. The best way to do so is through mobile phones. The number of smartphone

users is projected to increase globally from 1.75 billion in 2014 to almost 2.5 billion by 2017. The graph below highlights the growth of mobile phone users worldwide:
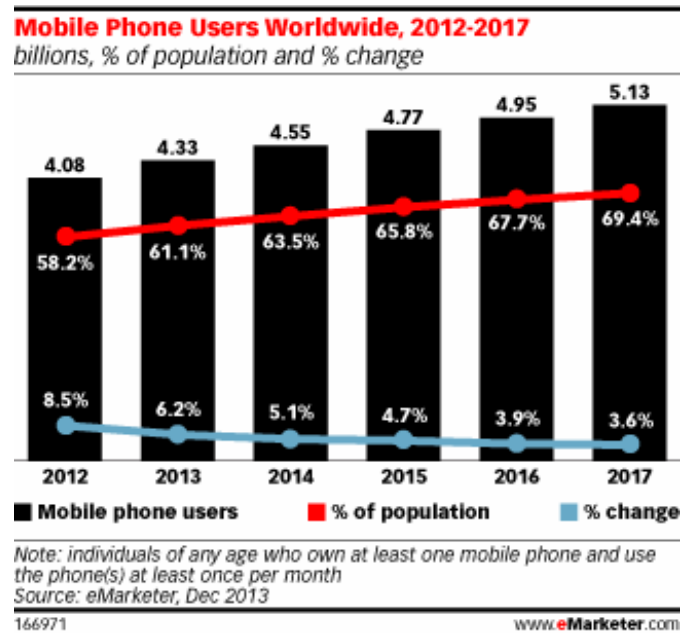

Figure 3.2: Mobile Phone Users 2012 - 2017

This highlights the potential for the app, and smartphone-aided diagnosis in general, not just in the U.K. but in other developing countries as well, where access to a sleep centre may not be as readily available.

The advantage that the proposed app holds over existing mobile phone apps is that it is specifically for diagnosing OSA - it is meant for medical purposes as opposed to general sleep cycle monitoring. This means that it does not compete with the above-mentioned apps, but serves a distinct purpose which is not possible in the other apps. Moreover, the use of machine learning algorithms in the app means that there is potential for the app to improve in accuracy over time as the amount of training data used is improved. With every update, the app can become better at diagnosing OSA using only non-invasive methods.

In conclusion, the rationale for the project to develop such an app is clear. If developed, the app is viable as a supplementary service by the NHS and can improve the diagnosis of OSA in adults in the U.K. dramatically by reaching out to a wider audience and simultaneously result in savings for the NHS. It can also be further developed for use internationally, by those in developing countries in the future.

The app

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Design Process

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

5.1 Questionnaire selection

5.2   Signal Analysis – simple method

5.3 Machine Learning – Theory

Machine learning, a branch of artificial intelligence, concerns the construction and study of systems that can learn from data [10]. The supervised classification problem in machine learning concerns finding the unknown target function that classifies certain input data into classes based on some set of training examples containing labelled input data.

It is exciting to use machine learning algorithms to aid medical diagnoses because it can dramatically save time for medical experts. These techniques should create a reasonably reliable, though not perfect, system on which the medical experts can base their decisions on.

In our case, given a sampled sound signal of a sleeper, we want to identify apnoea periods. In particular, we represent our input (sampled sound signal) as $\mathbf{S} = \{s_1, s_2, \ldots, s_T\} \equiv \{s_i\}_{i=1}^T$, and we want to output the classifiers for every K samples, $\mathbf{Y} = \{y_i\}_{i=1}^{T/K}$, where the classifier $y_i \in \{0, 1\}$ corresponds to samples $\{s_j\}_{j=(i-1)K+1}^{iK}$ of the signal. We assume that T is divisible by K, however if this is not the case, we discard an appropriate number of signal samples to fulfill this condition.

We have researched three models for our problem which we will discuss in this section. Firstly, we will discuss Support Vector Machines which are one of the most widely used algorithms in Machine Learning today, then we will discuss the State-Space Model and the Hidden Markov Models, which are more well-suited to our problem due to their temporal nature. Moreover, we discuss techniques to condition our data before using them as input data for our learning models.

5.3.1 Support Vector Machines

Here, we present the theory for Support Vector Machines (SVMs) based on the lecture notes from Prof. Andrew Ng [4]. SVMs are one of the most widely used and many argue among the best "off-the-shelf" supervised learning algorithms. This is mainly due to the sound theoretical framework, efficiency and good generalisation guarantees even for high-dimensional and linearly non-separable data.

Notation

Having m training examples, where

- $\mathbf{x}^{(i)} \in \mathbb{R}^d$ is the d-dimensional i-th training example.
- $y^{(i)} \in \{-1, 1\}$ is the i-th training label.

we want to find the parameter $\mathbf{w} \in \mathbb{R}^d$ which describes the hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ that separates our two classes. Thus, we can define our classifier as $h_{\mathbf{w},b}(\mathbf{x}) = g\left(\mathbf{w}^T\mathbf{x} + b\right)$, such that $g(z) = 1$ if $z \geq 0$ and $g(z) = -1$ otherwise. Note that this is a non-probabilistic learning model as we are not considering the probability of each class or the data.

Objectives

The essence of SVMs lies in finding the decision boundary (hyperplane) which maximises the gap between the closest training points to it. For the i-th training point $x^{(i)}$, we call the "gap", geometric margin $\gamma^{(i)}$, which is the distance to the decision hyperplane and can be found by considering a point $\mathbf{x}$ on it (note that $\mathbf{w}^T\mathbf{x} = -b$):

$$
\begin{aligned}
\gamma^{(i)} &= \left| \frac{\mathbf{w}^T}{\|\mathbf{w}\|} \left(\mathbf{x}^{(i)} - \mathbf{x}\right) \right| \\
&= \frac{1}{\|\mathbf{w}\|} y^{(i)} \left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right)
\end{aligned}
$$

However, we only consider the closest point $\mathbf{x}^{(i^*)} : i^* = \arg\min_i \gamma^{(i)}$ with the geometric margin of $\gamma = \min_i \gamma^{(i)}$. Since scaling $\mathbf{w}$ and b does not change the output of the classifier, nor the geometric margin $\gamma^{(i)}$, for convenience, we decide to scale $\mathbf{w}$ and b such that $\left|\mathbf{w}^T\mathbf{x}^{(i^*)} + b\right| = y^{(i^*)}\left(\mathbf{w}^T\mathbf{x}^{(i^*)} + b\right) = 1$. Thus, maximising the geometric margin of the closest point becomes

$$
\begin{aligned}
\max_{\mathbf{w},b} \quad & \frac{1}{\|\mathbf{w}\|} \\
\text{s.t.} \quad & y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad i = 1, \ldots, m.
\end{aligned}
$$

which is equivalent to

$$
\begin{aligned}
\min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{s.t.} \quad & y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad i = 1, \ldots, m.
\end{aligned}
\tag{5.1}
$$

The solution to the optimisation problem (5.1), which can be found using off-the-shelf Quadratic Programming (QP) packages, gives us the parameters required to do classification.

Lagrange duality

While solving the optimisation problem (5.1) using QP solves the problem, we move on to derive the dual form of the problem using the principle of Lagrange duality, which will help us to solve the problem using a
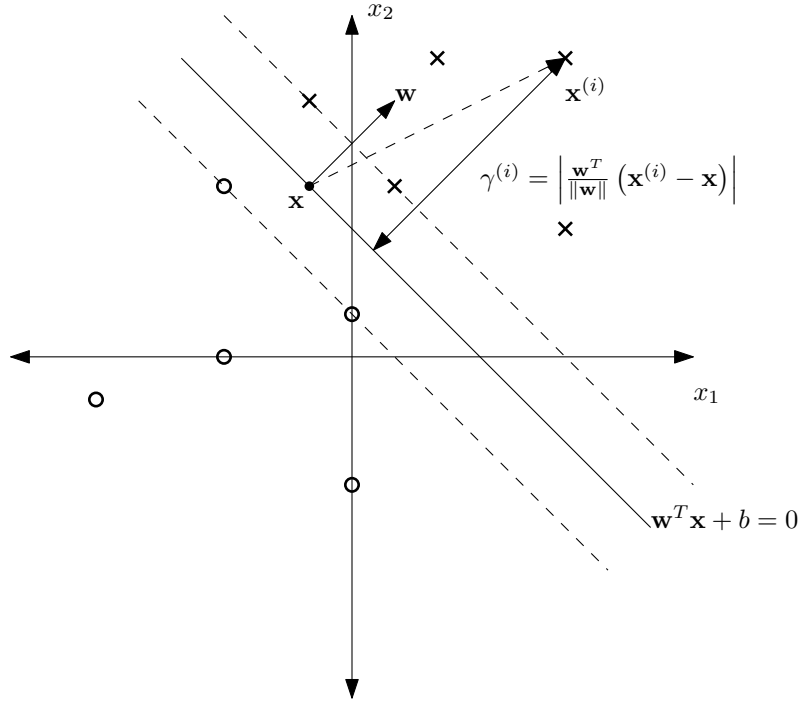
Figure 5.1: Illustration of the optimal margin classifier.

more efficient algorithm. More importantly, it will also allow us to efficiently transform the data points to high-dimensional spaces using the "kernel trick", capturing the non-linear nature of the decision boundary without losing the generalisation guarantees. The Lagrangian of our primal minimisation problem (5.1) (with parameters $\boldsymbol{\alpha} \in \mathbb{R}^m$) can be formed as

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)} \left(\mathbf{w}^T \mathbf{x}^{(i)} + b \right) - 1 \right] \tag{5.2}$$

To find the dual form of the problem, $W(\boldsymbol{\alpha}) = \min_{\mathbf{w}, b} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha})$, we find the gradients of (5.2) with respect to $\mathbf{w}$ and b and set them to zero to get

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y^{(i)} \mathbf{x}^{(i)} \tag{5.3}$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \tag{5.4}$$

Substituting (5.3) and (5.4) back to (5.2) gives us the dual form, $W(\boldsymbol{\alpha})$

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle \mathbf{x}^{(i)}, \mathbf{x}^{(j)} \right\rangle \tag{5.5}$$

Thus, under certain conditions which are in this case fulfilled (and omitted for clarity), our original optimisation problem (5.1) becomes equivalent to the dual optimisation problem

$$
\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) \\
\text{s.t.} \quad & \alpha_i \geq 0, \qquad\qquad i = 1, \ldots, m \\
& \sum_{i=1}^{m} \alpha_i y^{(i)} = 0, \quad i = 1, \ldots, m
\end{aligned}
\tag{5.6}
$$

This problem can be solved using a QP algorithm, however a more efficient Sequential Minimal Optimization (SMO) algorithm can be used. Once the optimal value $\boldsymbol{\alpha}^*$ is obtained, we can find the corresponding parameters of the SVM $\mathbf{w}^*$ using (5.3) and b^* using

$$
\begin{aligned}
b^* &= -\frac{\max_{i:y^{(i)}=-1} \mathbf{w}^{*T}\mathbf{x}^{(i)} + \min_{i:y^{(i)}=1} \mathbf{w}^{*T}\mathbf{x}^{(i)}}{2} \\
&= -\frac{\max_{i:y^{(i)}=-1} \sum_{j=1}^{m} \alpha_i^* y^{(i)} \left\langle \mathbf{x}^{(i)}, \mathbf{x}^{(j)} \right\rangle + \min_{i:y^{(i)}=1} \sum_{j=1}^{m} \alpha_i^* y^{(i)} \left\langle \mathbf{x}^{(i)}, \mathbf{x}^{(j)} \right\rangle}{2}
\end{aligned}
\tag{5.7}
$$

Thus the classification of a test point $\mathbf{x}$ can be done by evaluating the argument of $g(\mathbf{w}^{*T}\mathbf{x} + b^*)$

$$
\mathbf{w}^{*T}\mathbf{x} + b^* = \sum_{i=1}^{m} \alpha_i^* y^{(i)} \left\langle \mathbf{x}^{(i)}, \mathbf{x} \right\rangle + b^*
\tag{5.8}
$$

We note that due to certain conditions (Karush-Kuhn-Tucker conditions) the α_i^*'s that are non-zero correspond to the $\mathbf{x}^{(i)}$'s that lie on the margin. We call these points the Support Vectors (SVs).

Kernel trick

Note that according to (5.8) (and (5.7)), we only need to evaluate the inner products of $\mathbf{x}$ and the support vectors in order to classify the point $\mathbf{x}$. This fact is used in the "kernel trick", in which a kernel function $K(\mathbf{x}, \mathbf{z})$ is used as a proxy for the inner product $\langle \mathbf{x}, \mathbf{z} \rangle$. This effectively simulates transforming the data points to another, possibly unknown, space via a transformation function $\boldsymbol{\phi}(\cdot) : \mathbb{R}^d \to \mathcal{Z}$ as long as there exists such $\mathcal{Z}$, i.e. there exists $\boldsymbol{\phi}(\cdot)$ such that $K(\mathbf{x}, \mathbf{z}) = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{z})$. It can be shown that a kernel $K(\cdot, \cdot)$ is valid if and only if it satisfies certain conditions called the Mercer's condtitions. This allows us to transform our input domain into much higher dimensional domains without actually doing so which turns

out to improve the time complexity significantly.

$$K_{\text{poly}}(\mathbf{x}, \mathbf{z}) = \left(a\mathbf{x}^T\mathbf{z} + c\right)^k \tag{5.9}$$

$$K_{\text{rbf}}(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2}\right) \tag{5.10}$$

In our application, we are going to use the kernels above (5.9, 5.10). The first one, (5.9), is the polynomial kernel which effectively transforms the input domain space into a feature space whose features are products all possible permmutations of the input features of degrees up to k. The second one, (5.10), is a radial basis kernel (RBF), whose corresponding transformation function transforms the input domain space to an infinite dimensional one. This kernel gives a large inner product to two points if they are close to each other.

Soft margin SVMs

Soft margin SVMs are a slight modification of the original problem that allows small violations of the margin. The purpose is to guarantee that the algorithm doesn't fail because of the non-separability of the input data even after using the kernel trick. In practice, these two techniques are used simultaneously. It turns out that only minor changes arise and the SMO algorithm can still be used.

Our problem

In our problem, we are going to use the `SVM` package in `MATLAB`® in order to train our model and classify sleep apnoea. The package includes implementations of both QP and SMO algorithms, as well as various kernels, including the polynomial kernel and the RBF kernel. Recall that in our problem, we have the signal $\{s_i\}_{i=1}^{T}$ and the classifiers for every K samples $\{y_i\}_{i=1}^{T/K}$, where $y_i \in \{0, 1\}$ classifies the time period represented by $\{s_j\}_{j=(i-1)K+1}^{iK}$. When using SVMs, the input vectors and the classifiers become

$$\mathbf{x}^{(i)} = [s_{(i-1)K+1}, \ldots, s_{iK}]^T$$

$$y^{(i)} = \begin{cases} 1 & \text{if } y_i = 1 \\ -1 & \text{otherwise.} \end{cases}$$

5.3.2 State-Space Models

Here, we present the State-Space Models (SSMs) (also known as Linear Dynamical Systems) based on Kevin Murphy's book [3, Chapter 18] and Prof. Zoubin Ghahramani's paper [1]. SSMs allow us to model systems that are dynamic. Unlike the SVMs, SSMs will model dependencies of the current state on the previous ones.
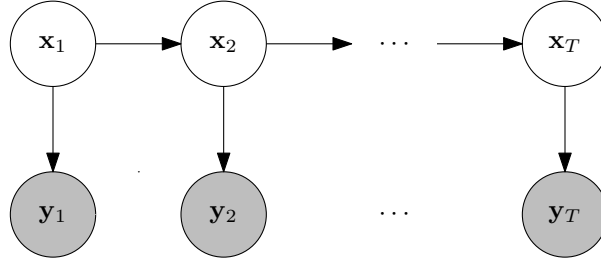

Figure 5.2: Probabilistic Graphical Model of a SSM and a HMM

We have two set of random variables, the hidden states $\mathbf{X} = \left\{\mathbf{x}_t, \mathbf{x}_t \in \mathbb{R}^d\right\}_{t=1}^T$ and the observed vectors $\mathbf{Y} = \left\{\mathbf{y}_t, \mathbf{y}_t \in \mathbb{R}^k\right\}_{t=1}^T$. We can represent SSMs graphically in a Bayesian netowrk in Figure 5.2, and write out the joint probability of the model as

$$p\left(\mathbf{X}, \mathbf{Y}\right) = p(\mathbf{x}_1)p(\mathbf{y}_1|\mathbf{x}_1)\prod_{t=2}^T p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{y}_t|\mathbf{x}_t) \tag{5.11}$$

In our case, we consider these particular transition and observation models with zero-mean Gaussian noises:

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{w}_t \tag{5.12}$$

$$\mathbf{y}_t = \mathbf{C}\mathbf{x}_t + \mathbf{v}_t \tag{5.13}$$

$$\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}) \tag{5.14}$$

$$\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \tag{5.15}$$

Since we assume that $\pi = p(\mathbf{x}_1)$ is also Gaussian, all our conditional probability distributions will be Gaussian due to the linearity of the transition and observation models. Once the parameters are obtained, the problem of inference and state estimation consists of

1. **Filtering.** We want to find $p(\mathbf{x}_t|\{\mathbf{y}_t\}_1^t)$. This is solved by the Kalman filtering algorithm.
2. **Smoothing.** We want to find $p(\mathbf{x}_t|\mathbf{Y})$. This is solved by the Kalman smoothing algorithm.

3. **Prediction.** We want to find $p(\mathbf{x}_{t+\tau}|\{\mathbf{y}_t\}_1^t)$. This is done by solving to 1. and then simulating using the state transition function.

We are mainly interested in filtering and smoothing.

Our problem

Although SSMs are well-suited for time-series data, they are not very well suited for our problem because the hidden state $\mathbf{x}_t$ in this model is continuous, whereas in our problem, the sleeper's state is binary. While there are techniques to map from a continuous domain to a discrete one, we move on to discuss a more promising model.

5.3.3 Hidden Markov Models

Here, we present the Hidden Markov Models (HMMs), one of the most popular statistical models in machine learning with applications in many fields including but not limited to Cryptanalysis, Speech recognition and Bioinformatics [9]. The material presented here is based on [5], [6] and [3]. We will first present HMMs with discrete observations, then we extend this to include models with continuous observations.

Discrete observations

Similarly to SSMs, we have two sets of random variables. Observed variables $\mathbf{Y} = \{y_t\}_{t=1}^T$ which are drawn from the observation alphabet $V = \{v_1, \ldots, v_M\}$, and hidden states $\mathbf{X} = \{x_t\}_{t=1}^T$ which are drawn from the hidden state alphabet $S = \{s_1, \ldots, s_N\}$. The probabilistic graphical model of the HMM is identical to the one in Figure 5.2. The hidden states obey Markov assumptions, i.e. $P(x_t|x_{t-1}, \ldots, x_1) = P(x_t|x_{t-1}) = $ const., $t = 2, \ldots, T$. Moreover, the observed variables are only dependent on the corresponding hidden state, i.e. $P(y_t|x_t, \ldots, x_1, y_{t-1}, \ldots, y_1) = P(y_t|x_t), t = 1, \ldots, T$.

We parameterise a HMM using the transition matrix $\mathbf{A}$, emission matrix $\mathbf{B}$, and the initial state distribution $\boldsymbol{\pi}$. The transition matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ describes the transitions between hidden states, $A_{ij} = P(x_{t+1} = s_j | x_t = s_i)$. The emission matrix $\mathbf{B} \in \mathbb{R}^{N \times M}$ describes the probability of an observation conditioned on a hidden state, $B_{jk} = B_j(v_k) = P(y_t = v_k | x_t = s_j)$. The initial state distribution $\boldsymbol{\pi} \in [0, 1]^N$ simply describes the initial probabilities of the hidden state, $\pi_i = P(x_1 = s_i)$. The model is fully described if we know these parameters, which we group into what is called a parameter set of the model, $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$.

18

The three main questions of a HMM are

1. Find the probability of observations given the model, $P(\mathbf{Y}; \lambda)$.

2. Find the most likely series of hidden states $\mathbf{X}$ to have generated the observations $\mathbf{Y}$, $\mathbf{X}^* = \arg\max_{\mathbf{X}} P(\mathbf{Y}|\mathbf{X}; \lambda)$.

3. Find the parameters λ to maximise $P(\mathbf{Y}; \lambda)$.

We will discuss algorithmic solution to these three problems in turn.

Solution to the first problem. To find the probability of an observed sequence, we use the dynamic programming algorithm, called the FORWARD PROCEDURE (outlined in Algorithm 1) which calculates the forward variable $\alpha_t(i) = P(\mathbf{Y}, x_t = s_i; \lambda)$. As we can see, the algorithm has a time complexity of $O(TN)$.

Algorithm 1 FORWARD PROCEDURE for computing $\alpha_t(i)$.

---

1. **Initialisation.**
$$\alpha_1(i) = \pi_i B_i(y_1), 1 \le i \le N$$

2. **Induction.**
$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) A_{ij} \right] B_j(y_{t+1}), \quad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix}$$

3. **Termination.** (solution to the first problem)

$$P(\mathbf{Y}; \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Solution to the second problem. To solve the problem of finding the most likely sequence of hidden states, we use the VITERBI ALGORITHM proposed by Andrew Viterbi in 1967. The most likely sequence of hidden states is also called the Viterbi path. Firstly, we define the quantity $\delta_t(i)$, which stores the highest probability along a single path ending at the state $x_t = s_i$:

$$\delta_t(i) = \max_{x_1,\dots,x_{t-1}} P(x_1, \dots, x_{t-1}, x_t = s_i, y_1, \dots, y_t; \lambda) \tag{5.16}$$

By induction, we have

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) A_{ij} \right] B_j(y_{t+1}) \tag{5.17}$$

We also keep track of the index of the hidden state that maximises this quantity in

$$\psi_{t+1}(j) = \arg\max_{i} \delta_t(i) A_{ij} \tag{5.18}$$

Having defined these quantities, we present the VITERBI ALGORITHM in Algorithm 2. As we can see, the

---

Algorithm 2 VITERBI ALGORITHM for computing the most likely sequence of hidden states.

1. **Initialisation.**

$$\delta_1(i) = \pi_i B_i(y_1), \qquad\qquad\qquad 1 \le i \le N$$
$$\psi_1(i) = 0, \qquad\qquad\qquad\qquad\quad 1 \le i \le N$$

2. **Recursion.**

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) A_{ij}] B_j(y_t), \qquad \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix}$$

$$\psi_t(j) = \arg\max_{1 \le i \le N} [\delta_{t-1}(i) A_{ij}], \qquad \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix}$$

3. **Termination.**

$$P^* = \max_{1 \le i \le N} \delta_T(i)$$
$$x_T^* = \arg\max_{1 \le i \le N} \delta_T(i)$$

4. **Path backtracking.**
$$x_t^* = \psi_{t+1}\left(x_{t+1}^*\right), T - 1 \ge t \ge 1$$

5. **Return** $\{x_t^*\}_1^T$.

---

algorithm has a time complexity of $O(TN^2)$.

Solution to the third problem. To learn the parameters of the model, $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, we use the FORWARD-BACKWARD ALGORITHM by Baum-Welch. We will need to introduce few quantities. Firstly, we introduce the backward variable $\beta_t(i) = P(y_{t+1}, \ldots, y_T | x_t = s_i; \lambda)$ which can be computed using a dynamic programming algorithm in Algorithm 3. We also define the variable $\gamma_t(i) = P(x_t = s_i | \mathbf{Y}; \lambda)$

Algorithm 3 BACKWARD ALGORITHM for computing $\beta_t(i)$.

1. **Initialisation.**
$$\beta_T(i) = 1, 1 \leq i \leq N$$

2. **Induction.**
$$\beta_t(i) = \sum_{j=1}^{N} A_{ij} B_j(y_{t+1}) \beta_{t+1}(j), \quad \begin{array}{l} T-1 \leq t \leq 1 \\ 1 \leq j \leq N \end{array}$$

---

which can be expressed as

$$
\begin{aligned}
\gamma_t(i) &= P\left(x_t = s_i | \mathbf{Y}; \lambda\right) \\
&= \frac{P\left(x_t = s_i, \{y_\tau\}_1^t; \lambda\right) P\left(\{y_\tau\}_{t+1}^T | x_t = s_i; \lambda\right)}{P\left(\mathbf{Y}; \lambda\right)} \\
&= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}
\end{aligned}
\tag{5.19}
$$

We also define the quantity $\xi_t(i, j)$ as

$$
\begin{aligned}
\xi_t(i, j) &= P\left(x_t = s_i, x_{t+1} = s_j | \mathbf{Y}; \lambda\right) \\
&= \frac{\alpha_t(i) A_{ij} B_j(y_{t+1}) \beta_{t+1}(j)}{\sum_{i,j=1}^{N} \alpha_t(i) A_{ij} B_j(y_{t+1}) \beta_{t+1}(j)}
\end{aligned}
\tag{5.20}
$$

Now, we are in a position to present the FORWARD-BACKWARD ALGORITHM by Baum-Welch in Algorithm 4. The algorithm belongs to a family of Expectation-maximisation (EM) algorithms for finding maximum likelihood (ML) or maximum a posteriori (MAP) estimates of parameters in statistical models [8].

Algorithm 4 FORWARD-BACKWARD ALGORITHM (BAUM-WELCH) for estimating HMM parameters λ.

1. **Initialisation.** Set $\mathbf{A}$, $\mathbf{B}$, $\boldsymbol{\pi}$ to be random valid probability matrices/vectors.
2. **Repeat until convergence:**

   - **E-step.** Run FORWARD and BACKWARD PROCEDURES to get $\alpha_t(i)$ and $\beta_t(i)$. Evaluate $\gamma_t(i)$ using (5.19).
   - **M-step.** Re-estimate parameters using

$$\pi_i = \gamma_1(i)$$
$$A_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T} \gamma_t(i)}$$
$$B_j(v_k) = \frac{\sum_{t=1, \text{s.t.} y_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Parameters estimation with known hidden states. In case the hidden states $\mathbf{X}$ are given to us, in addition to the observed variables $\mathbf{Y}$, the problem of parameters estimation simply reduces to counting transitions, i.e.

$$A_{ij} = \frac{\sum_{t=1}^{T-1} 1\{x_t = s_i \wedge x_{t+1} = s_j\}}{\sum_{t=1}^{T} 1\{x_t = s_i\}} \tag{5.21}$$

$$B_j(k) = \frac{\sum_{t=1}^{T} 1\{x_t = s_j \wedge y_t = v_k\}}{\sum_{t=1}^{T} 1\{x_t = s_j\}} \tag{5.22}$$

where $1(\cdot)$ is an indicator function (1 if the boolean argument is true, 0 otherwise).

Continuous observations

We now discuss the case in which the observations $\mathbf{Y} = \{y_t\}_1^T$ are not drawn from a finite set V, but are real-valued vectors $\{\mathbf{y}_t\}_1^T$, drawn from $\mathbb{R}^d$, i.e. $\mathbf{y}_t \in \mathbb{R}^d, 1 \leq t \leq T$ (Note: hidden states $\mathbf{X} = \{x_t\}_1^T$ are still drawn from a finite set S). In this case, we cannot describe the observations using an emission matrix anymore. Since the observation random variables are now continuous, they must be drawn from some probability distribution function (pdf). This function can be any arbitrary pdf, however in order to learn anything useful, we parameterise it. We assume a simple case and let it be a Gaussian, parameterised by its mean, and covariance. In particular, $B_j(\cdot)$ becomes a probability distribution function:

$$B_j(\mathbf{y}_t) = P\left(\mathbf{y}_t | x_t = s_j\right)$$

$$= \mathcal{N}(\mathbf{y}_t; \boldsymbol{\mu}_j, \mathbf{U}_j), \qquad \begin{array}{l} 1 \leq t \leq T \\ 1 \leq j \leq N \end{array} \tag{5.23}$$

This means that instead of the original emission matrix $\mathbf{B}$, we are now parameterising the emissions using the means $\boldsymbol{\mu} = \{\boldsymbol{\mu}_j \in \mathbb{R}^d\}_1^N$ and the covariances $\mathbf{U} = \{\mathbf{U}_j \in \mathbb{R}^{d \times d}\}_1^N$. Thus, our parameters set becomes $\lambda = \{\mathbf{A}, \boldsymbol{\mu}, \mathbf{U}, \boldsymbol{\pi}\}$.

Changes to the algorithms. While the three main questions for the HMM remain the same, we must change the algorithms used in the discrete observations case. It turns out that in the FORWARD PROCE-DURE (Algorithm 1), BACKWARD PROCEDURE (Algorithm 3), and the VITERBI ALGORITHM (Algorithm 2), we don't need to change anything except the interpretation of $B_j(\cdot)$. Whereas before, we had a single value for this quantity, now we must evaluate it using the parameters of the pdf (in this case mean and covariance) in (5.23). The problem of estimating parameters, given the observations becomes slightly

different and will be left untouched in our report. However, we discuss parameters estimation, given both the observations and the hidden states, which simply becomes fitting a Gaussian, i.e.

$$\boldsymbol{\mu}_j = \frac{\sum_{t=1}^{T} 1\{x_t = s_j\}\mathbf{y}_t}{\sum_{t=1}^{T} 1\{x_t = s_j\}} \tag{5.24}$$

$$\mathbf{U}_j = \frac{\sum_{t=1}^{T} 1\{x_t = s_j\}(\mathbf{y}_t - \boldsymbol{\mu}_j)(\mathbf{y}_t - \boldsymbol{\mu}_j)^T}{\sum_{t=1}^{T} 1\{x_t = s_j\}} \tag{5.25}$$

Different probability density functions. We have assumed that the conditional distribution of the observations is Gaussian. One disadvantage of this method is that it may be too simple to capture the real pdf the observations are drawn from. One of the alternatives is the Gaussian Mixture Model (GMM), however fine-tuning the number of modes can be difficult. It turns out that finding the optimal MLE for a GMM is intractable [3].

Our problem

In our problem, we model the apnoeatic states as the hidden states $\{x_t\}_1^T$ drawn from a binary set $\{0,1\}$. Although the observed signal is a sampled one-dimensional signal, since we only have annotations every K samples, we stack all K samples to a vector and treat it as a K-dimensional, real-valued observed variable $\mathbf{y}_t \in \mathbb{R}^d$, $(d = K)$. Since we assume that we have the annotated signal, we can do the training offline. Thus we are interested in the third problem (with known hidden states), for which the solution is just fitting the parameters; and the second problem, finding the most likely sequence of the apnoeatic diagnoses, given the model and the observations, using the VITERBI ALGORITHM. We will train the model using the annotated data and once trained, we will use the trained model to diagnose sleep apnoea from new data. Implementations of the algorithms for our applications are available for MATLAB® in the packages `pmtk3` (`https://github.com/probml/pmtk3`) and `HMM Toolbox` (`http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html`) from Kevin Murphy.

5.3.4 Conditioning input data

Working in the time domain, the data is usually transformed to the frequency domain to analyse where the pattern is found more easily. Also, ergardless of the chosen model, we will work in high dimensions which means we need a lot of input data in order to find the pattern. In the case of SVMs, it will be high-dimensional training points $\{\mathbf{x}^{(i)}\}_1^m$; in the case of SSMs and HMMs, it will be the high-dimensional observed variables $\{\mathbf{y}_t\}_1^T$. We will present two ways to condition the input data, before analysing it using

the learning algorithms above, namely frequenc-domain analysis and Principal Components Analysis.

Frequency-domain analysis

Discrete Fourier Transform.   Discrete Fourier Transform (DFT) is used to transform a finite list of equally spaced samples of a function into the list of coefficients of a finite combination of complex sinusoids [7]. It can be said to convert the sampled function from its original domain to the frequency domain [7]. The DFT, $\mathcal{F}$, can be defined to transform $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$ to $\mathbf{X} = \{X_1, X_2, \ldots, X_N\}$ as

$$X_k = \sum_{n=1}^{N} x_n \exp\left(-i2\pi kn/N\right), 1 \leq k \leq N \tag{5.26}$$

and can be performed using the Fast Fourier Transform (FFT) algorithm.

Power Spectral Density.   It is common to take the Power Spectral Density (PSD) instead of the DFT in order to remove the imaginary components in the frequency domain. The PSD of a continuous signal $x(t)$ can be defined as

$$S_{xx}(f) = \lim_{T \to \infty} \frac{1}{T} |X(f)|^2 \tag{5.27}$$

where $X(f)$ is the Fourier transform ($\mathcal{FT}$) of $x(t)$ in the interval $-T/2 < t < T/2$ [2]. It can be shown that in the discrete case where $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$, the PSD is obtained by

$$S_{xx}(k) = \lim_{T \to \infty} \frac{(\Delta t)^2}{T} |X_k|^2 \tag{5.28}$$

$$\approx \frac{(\Delta t)^2}{T} |X_k|^2 \tag{5.29}$$

where T is the actual recording time, N is number of samples and $1/\Delta t$ is the sampling frequency (note that $T = N(\Delta t)$) [11].

Spectrograms.   To create a spectrogram of a signal $\mathbf{x} = \{x_1, x_2, \ldots, x_N\} = \{x(\Delta t), x(2\Delta t), \ldots, x(N\Delta t)\}$, we must choose a window size T_{window} and an offset T_{offset}. Then, we keep shifting the window by T_{offset} and each time, we transform and store the window's content (in time domain) to the frequency domain using the DFT. This way, we will have a frequency domain data of the size T_{window} every T_{offset}, which we will call $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_\tau$. We usually represent this frequency domain data using colour intensities and plot them as τ columns of colour intensities with time on the horizontal axis and frequency on the vertical

axis. Figure 5.3 illustrates the process of creating a spectrogram. However, it is common to find the PSDs instead of the DFTs of the sliding windows (the principle stays the same). Hence we will refer to the process of creating the spectrogram with PSDs (instead of DFTs) of the sliding windows as *freqency analysis of the signal*.
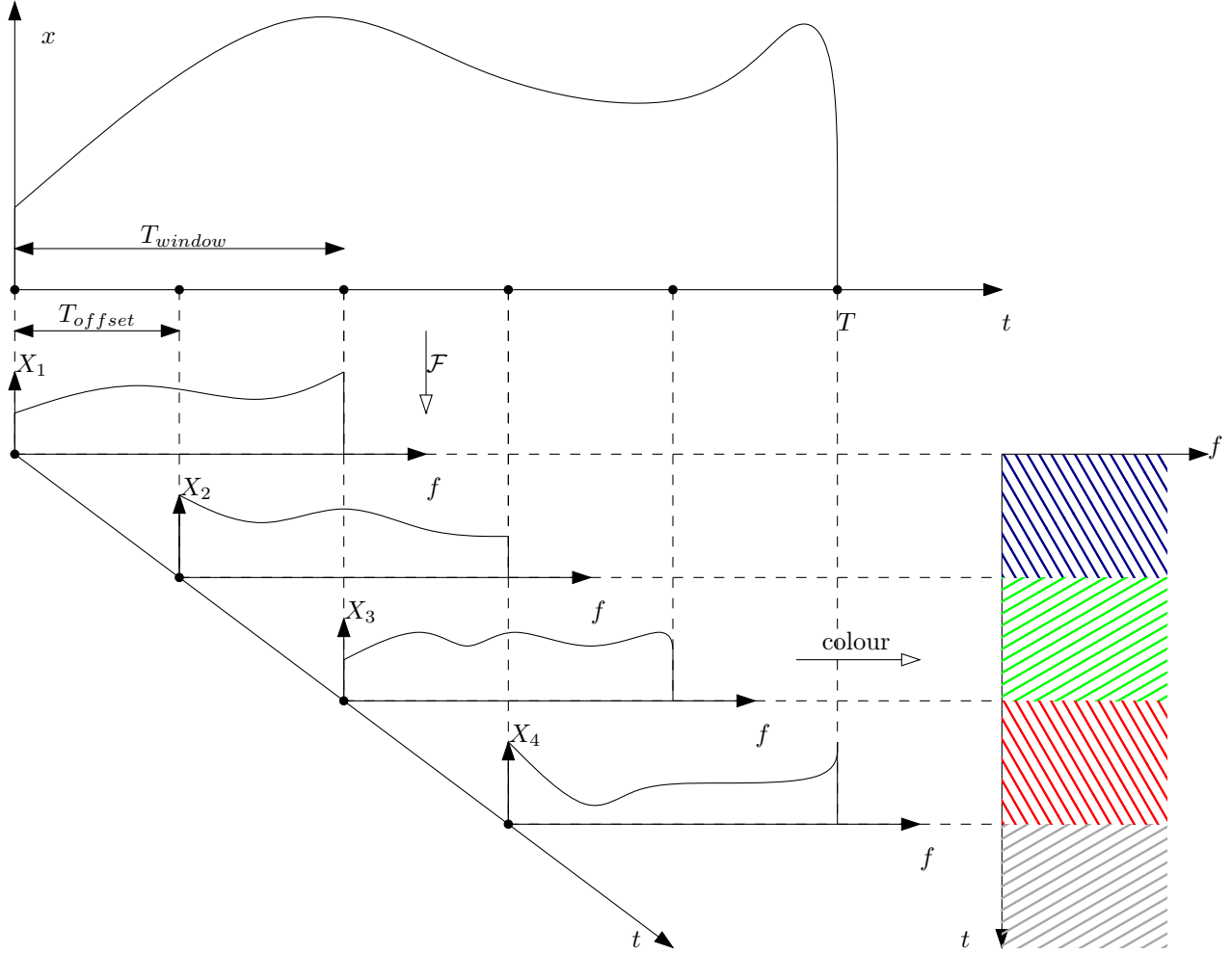

Figure 5.3: The process of creating a spectrogram.

Principal Components Analysis

Here we present the theory for Principal Components Analysis (PCA) based on Andrew Ng's lecture notes [4] and Kevin Murphy's book [3]. The objective is to take m n-dimensional input data points $\{\mathbf{x}^{(i)} \in \mathbb{R}^n\}_1^m$ and transform them into m k-dimensional data points $(k < n)$ $\{\mathbf{y}^{(i)} \in \mathbb{R}^m\}_1^m$, where $\mathbf{y}^{(i)}$'s are projections of $\mathbf{x}^{(i)}$'s onto k orthonormal basis vectors $\{\mathbf{u}_i\}_1^k$ while "preserving the most variance". We assume that over all m points, their mean $\left[\{x_j^{(i)}\}_{i=1}^m\right] = 0$ and their var $\left[\{x_j^{(i)}\}_{i=1}^m\right] = 1$ for all features of the input points $1 \leq j \leq n$. We normalise the data first if this is not the case. For $n = 2$, $k = 1$, Figure

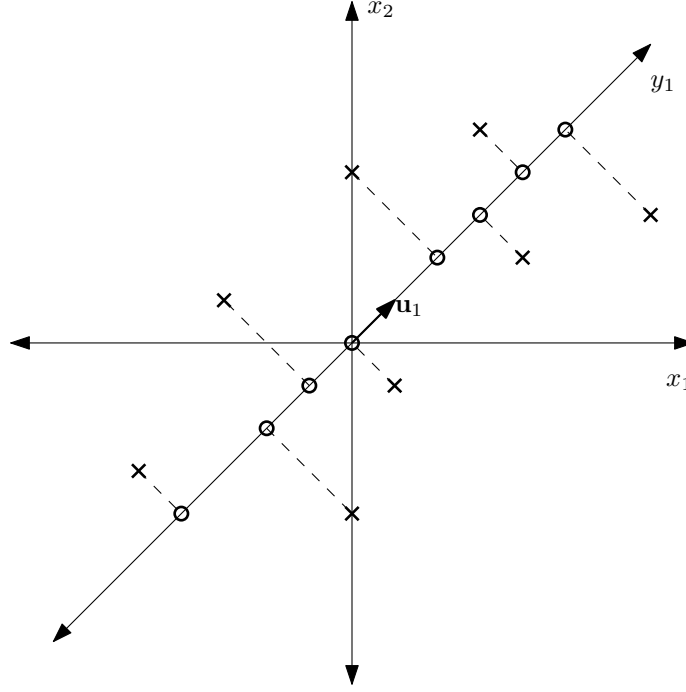5.4 shows the projections to the new axis.

Figure 5.4: Illustration of PCA.

We can see that the transformed points can be expressed as

$$
\mathbf{y}^{(i)} = \begin{bmatrix} \mathbf{u}_1^T \mathbf{x}^{(i)} \\ \mathbf{u}_2^T \mathbf{x}^{(i)} \\ \vdots \\ \mathbf{u}_k^T \mathbf{x}^{(i)} \end{bmatrix}, 1 \le i \le m \tag{5.30}
$$

It can be shown that in order to maximise $\sum_{i=1}^m \left\| \mathbf{y}^{(i)} \right\|^2$, conditioned on orthonormality of the bases, we must choose normalised eigenvectors corresponding to the k largest eigenvalues of the variance matrix $\boldsymbol{\Sigma} = \frac{1}{m} \sum_{i=1}^m \mathbf{x}^{(i)} \mathbf{x}^{(i)^T}$ as the bases. These bases are also called the *principal components*. The eigenvalues are proportional to the actual variances of data in the new bases. Hence, we can compare the relative "importance" of the bases and use this fact to decide how many principal components to use.

5.4 Machine Learning – Experiments

We obtained the data from the ECG database from the PhysioNet database located at `http://physionet.org/physiobank/database/apnea-ecg/`. The data consists of 35 labelled training records and 35 unlabelled records (used for the CinC Challenge 2000 competition). The recordings vary from less than 7 hours to 10 hours each and include continuous digitised ECG signal and, in the case of the training data, a set of apnoea annotations derived by human experts. The continuous signal is sampled at the rate of 100 Hz and the annotations are available at every 6000 samples (i.e. every minute) of the signal indicating the presence of apnoea at that time. One such record is shown in Figure 5.5.
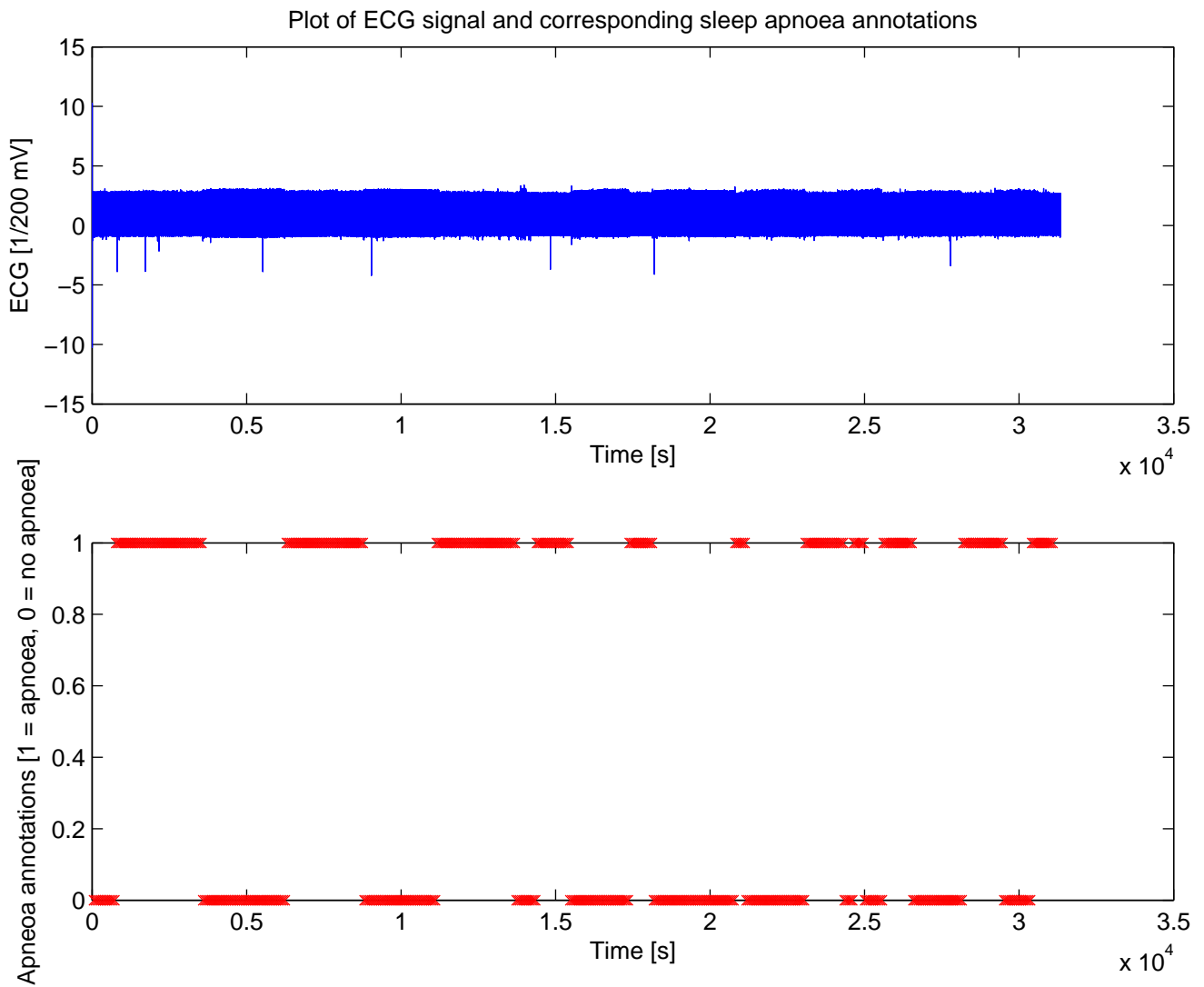
Figure 5.5: One training record

5.4.1   Conditioning input data

Frequency Analysis

We use MATLAB$^{\circledR}$'s function `spectrogram`, which takes the following parameters

- `X` – our signal $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$.
- `WINDOW` – length (in number of samples) of the window T_{window}. The windows are automatically filtered using a Hamming window. We arbitrarily choose the window to be the number of signal samples corresponding to one annotation (6000 in our case).
- `NOVERLAP` – number of overlapping samples between two consecutive windows. Effectively $T_{\text{window}} - T_{\text{offset}}$. We arbitrarily choose the offset to be 1000 such that we have six bins of PSD's for each annotation.
- `NFFT` – number of frequency points used to calculate the discrete Fourier transforms. We use default.
- `Fs` – sampling frequency in Hz. In our case 100 Hz.

A spectrogram for one record can be seen in Figure 5.6. Just by looking at the spectrogram, we can spot different regimes of the time series (most clearly seen at 0.5×10^4, 1×10^4, 2×10^4, and 2.7×10^4 seconds). Comparing this to the apnoea annotations in Figure 5.5, these regimes are actually the non-apnoeatic regimes. Also, we can see that most of the "action" is happening below 20 Hz. For this reason, we cut off the frequencies above 25 Hz for subsequent analysis. We combine the corresponding six bins of PSD's per annotation in to a large feature vector corresponding to that annotation. Next, we will reduce the dimensionality of these feature vectors using PCA.
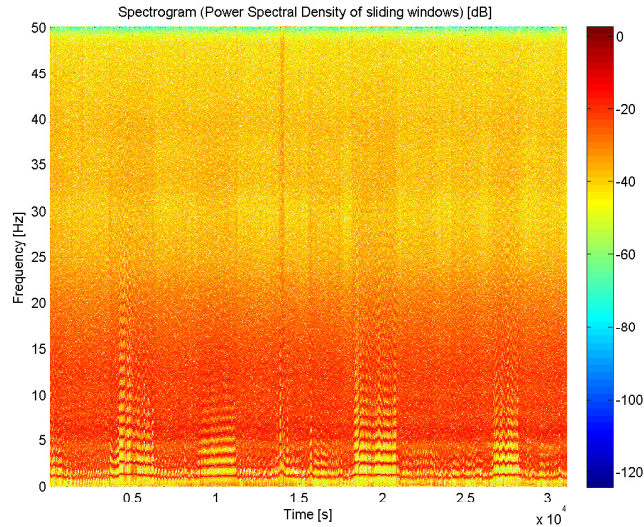

Figure 5.6: Spectrogram of one record

PCA

For this, we use the function `pca` from the package `pmtk3`. Once we get the principal components (orthogonal bases) and their corresponding principal coefficients (variances of the data, projected onto that base, correct to a constant factor) $\{\lambda_1, \lambda_2, \ldots, \lambda_D\}$, we will plot the graph of their cumulative sum over their total sum $\frac{\sum_{i=1}^{n} \lambda_i}{\sum_{j=1}^{N} \lambda_j}$ in order to decide on the number of principal components needed. The plot is in the Figure 5.7. Performed on the first 10 records, we can see that we will need to include at least 100 to capture half of the variance but at least 3000 components to capture the whole variance.
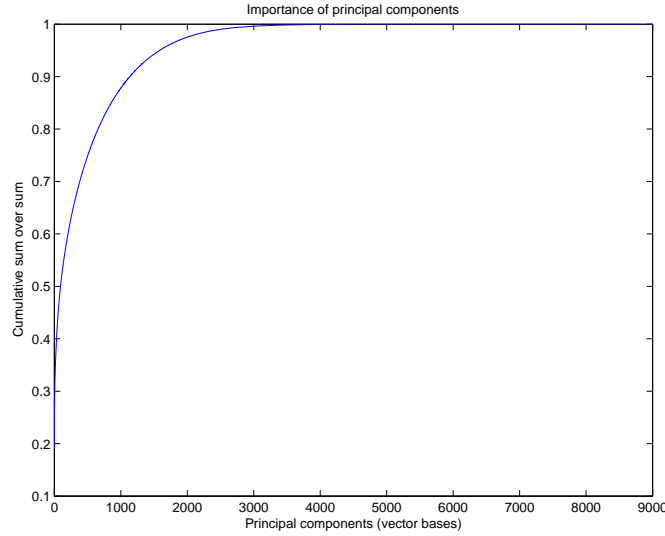
Figure 5.7: PCA of the first 10 records

5.4.2 Data Visualisation

It is helpful to visualise our data to confirm the presence of any detectable patterns and/or to help improve our learning algorithms. Since our Principal Component Analysis revealed that we need at least 100 principal components to capture at least half of the variance, it is not illustrative to visualise our data by plotting the data points on a 2D plane using only the first two principal components. Instead, we take the six corresponding bins of spectrograms of the two classes of annotations (apnoea, no apnoea) and take the average for each class as shown in Figure 5.8. We can notice the differences in the spectrograms which is reassuring because we can see that there exist some patterns to be found and that the spectrogram contains enough information to detect sleep apnoea.
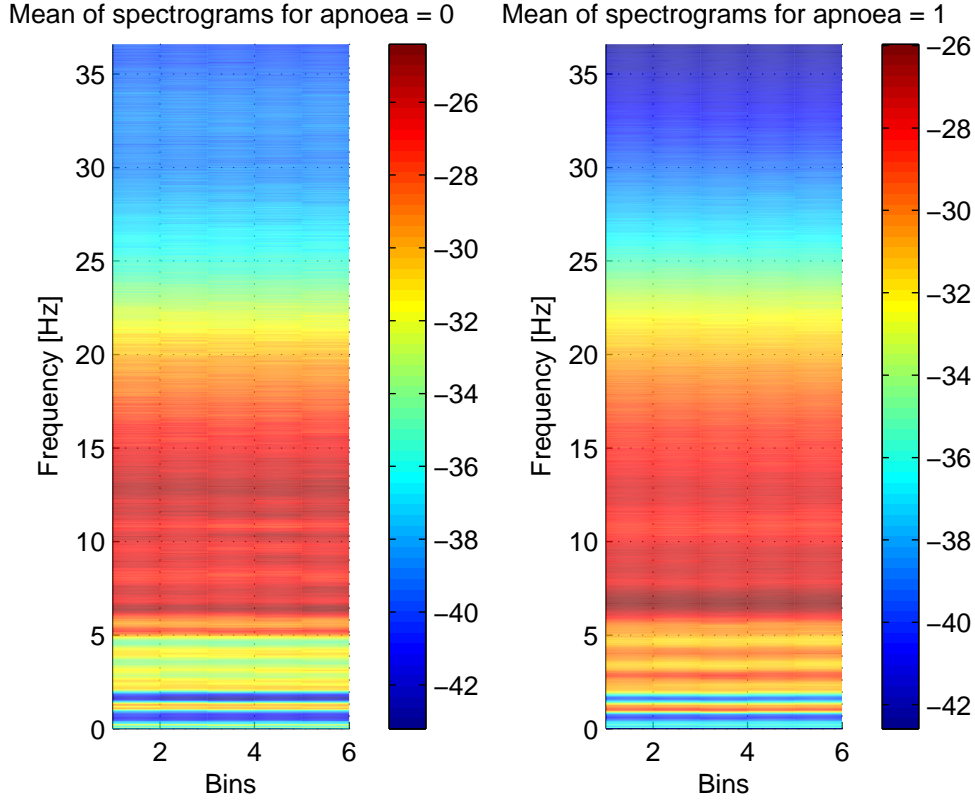
Figure 5.8: Comparison of the spectrograms of the two classes based on their average spectrogram from the first five records.

5.4.3 Support Vector Machines

Firstly, we investigate the case when no kernels are used. Then we will investigate the use of two possible kernels – polynomial (degree 3) and radial basis function. We will train on the first ten records using the MATLAB®'s function svmtrain. Then we will test our trained parameters on the next five records, where we record the accuracy as the ratio of the number of correctly classified annotations over the total number of annotations. We will also record the number of Support Vectors (SVs) because they indicate the generalisation performance.

No kernels

The results of this experiment are shown in Figure 5.9 and in Table 5.1. The accuracy is high, however so is the number of SV's. This indicates that the generalisation guarantee is poor.
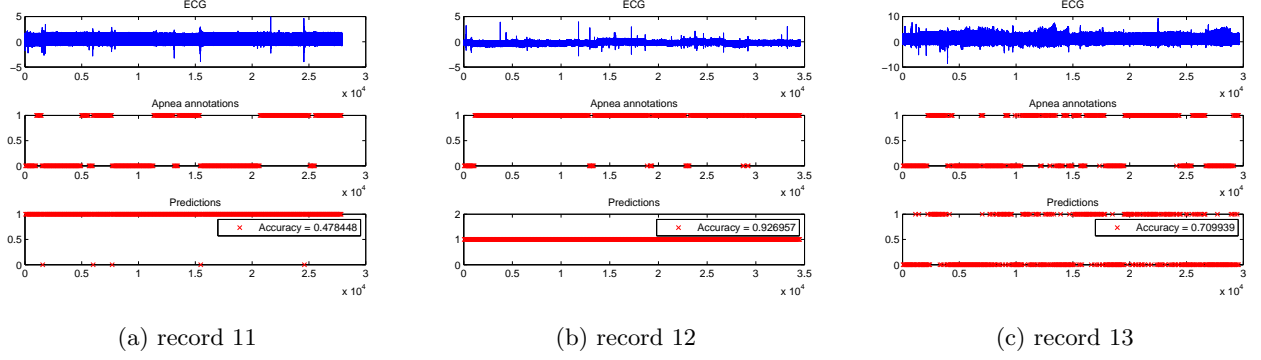
(a) record 11   (b) record 12   (c) record 13

(d) record 14   (e) record 15

Figure 5.9: Performance of no kernels on the test records 11 to 15
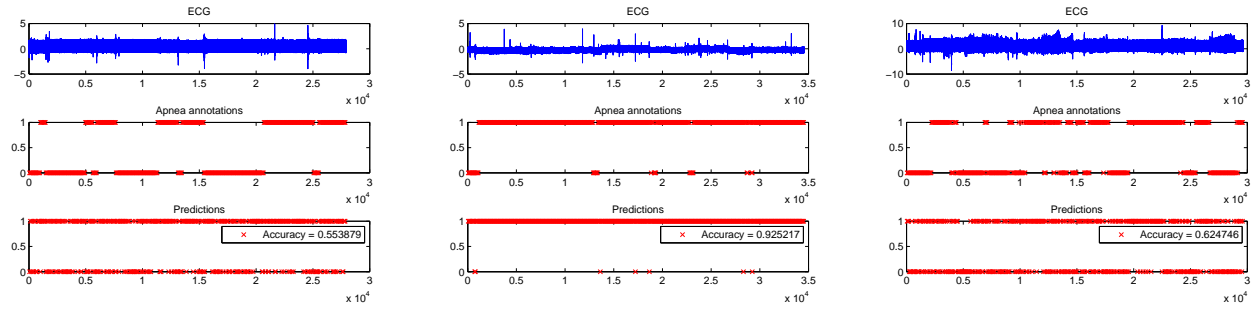
Polynomial kernel

The results of this experiment are shown in Figure 5.10 and in Table 5.1. We can see that there is a low number of SVs and high accuracy. This means that the polynomial kernel is likely to perform well on unseen data. If we had more computational resources, we could have used more features (and correspondingly more training data) in order to represent the features better.

Radial basis function kernel

The results of this experiment are shown in Figure 5.11 and in Table 5.1. Similarly to the case of no kernels, the accuracy is high, but so is the number of SV's indicating poor generalisation performance.
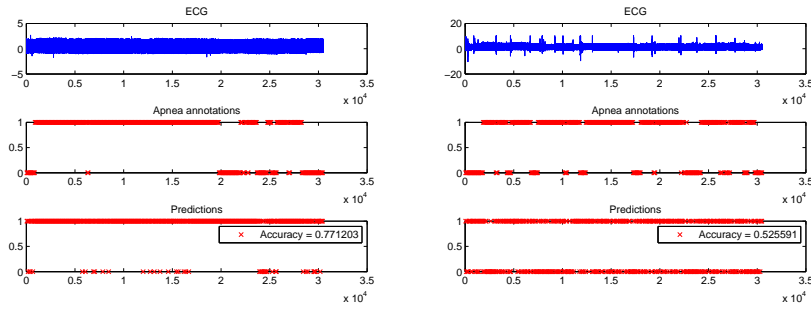
Summary

Summary of the performances of the different kernels is shown in Table 5.1. The polynomial kernel with degree 3 is the most promising in terms of generalisation. All three kernels have similar accuracy. If we had more computational resources, we could have done deeper analysis and tests, e.g. use more principal components to capture as much variance as possible and use more training data (we only used 10 out of

(a) record 11   (b) record 12   (c) record 13

(d) record 14   (e) record 15

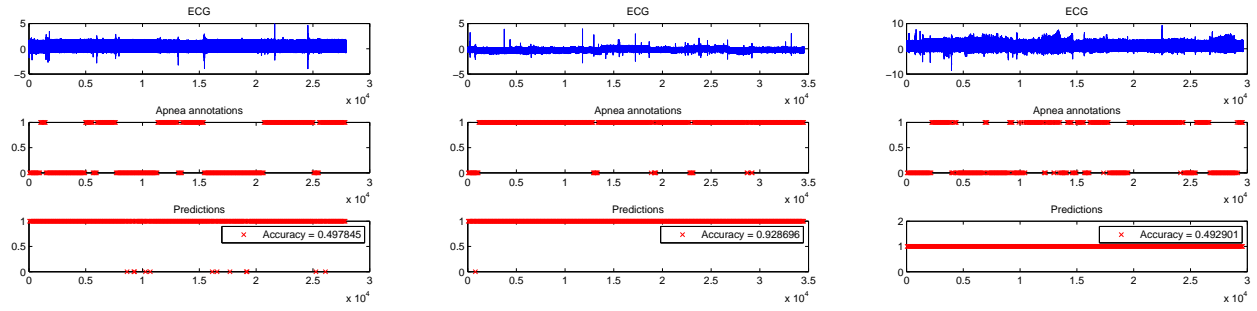Figure 5.10: Performance of the polynomial kernel on the test records 11 to 15

35).

	# SV's (5005 data points)	Average accuracy
No kernel	3679	0.7116
Poly kernel	1879	0.6801
Rbf kernel	4033	0.6814

Table 5.1: Summary of performances of different kernels
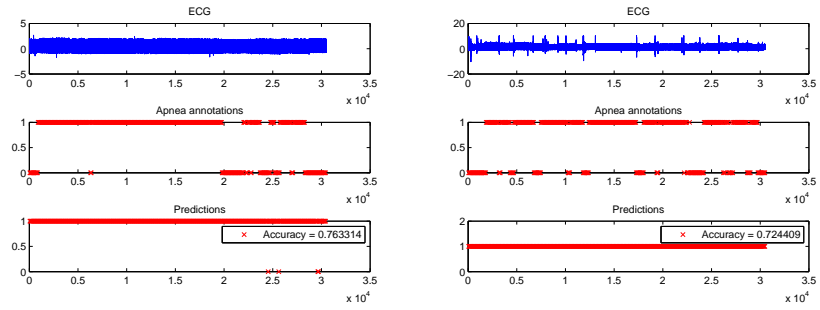
5.4.4 Hidden Markov Models

5.4.5 Summary

(a) record 11   (b) record 12   (c) record 13

(d) record 14   (e) record 15

Figure 5.11: Performance of the Radial Basis Function kernel on the test records 11 to 15

5.5 Machine Learning – Java

5.6   Android

Conclusion

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Bibliography

[1] Zoubin Ghahramani and Geoffrey E Hinton. Variational learning for switching state-space models. *Neural computation*, 12(4):831–864, 2000.

[2] A.M. Howatson, P.G. Lund, J.D. Todd, P.D. McFadden, P.J.P. Smith, and University of Oxford. Department of Engineering Science. *Engineering Tables and Data.* University of Oxford, Department of Engineering Science, 2008.

[3] Kevin P Murphy. *Machine learning: a probabilistic perspective.* Cambridge, MA, 2012.

[4] Andrew Ng. Cs229 - lecture notes. Course website, 2013.

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

[6] Daniel Ramage. Cs229 - section notes. Course website, 2007.

[7] Wikipedia. Discrete fourier transform — Wikipedia, the free encyclopedia, 2014. [Online; accessed 28-March-2014].

[8] Wikipedia. Expectation-maximization algorithm — Wikipedia, the free encyclopedia, 2014. [Online; accessed 25-March-2014].

[9] Wikipedia. Hidden markov model — Wikipedia, the free encyclopedia, 2014. [Online; accessed 24-March-2014].

[10] Wikipedia. Machine learning — Wikipedia, the free encyclopedia, 2014. [Online; accessed 18-March-2014].

[11] Wikipedia. Spectral density — Wikipedia, the free encyclopedia, 2014. [Online; accessed 14-April-2014].