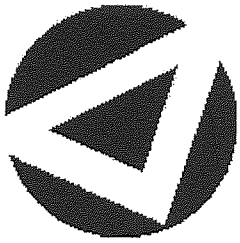
Hidden State Models for Time Series

MEHDI AZZOUZI

Doctor Of Philosophy



ASTON UNIVERSITY, BIRMINGHAM

October 1999

ASTON UNIVERSITY, BIRMINGHAM

Hidden State Models for Time Series

MEHDI AZZOUZI

Doctor Of Philosophy, 1999

Thesis Summary

Amongst all the objectives in the study of time series, uncovering the dynamic law of its generation is probably the most important. When the underlying dynamics are not available, time series modelling consists of developing a model which best explains a sequence of observations.

In this thesis, we consider hidden space models for analysing and describing time series. We first provide an introduction to the principal concepts of hidden state models and draw an analogy between hidden Markov models and state space models. Central ideas such as hidden state inference or parameter estimation are reviewed in detail.

A key part of multivariate time series analysis is identifying the delay between different variables. We present a novel approach for time delay estimating in a non-stationary environment. The technique makes use of hidden Markov models and we demonstrate its application for estimating a crucial parameter in the oil industry.

We then focus on hybrid models that we call dynamical local models. These models combine and generalise hidden Markov models and state space models. Probabilistic inference is unfortunately computationally intractable and we show how to make use of variational techniques for approximating the posterior distribution over the hidden state variables. Experimental simulations on synthetic and real-world data demonstrate the application of dynamical local models for segmenting a time series into regimes and providing predictive distributions.

Keywords: Time series, hidden Markov models, state space models, variational techniques

Acknowledgments

This thesis could not have been carried out without the support of many friends, colleagues and my family.

First and foremost I would like to thank my supervisor, Doctor Ian T. Nabney, for his criticism, supervision and freedom he gave me in deciding what research to pursue, which probably helped me most in developing my view of scientific research.

My work has benefited greatly from discussions with many of the members of the Neural Computing Research Group and in particular Professors Christopher M. Bishop and David Lowe.

Many thanks go to Mike Affleck of Thule Rigtech for his assistance. I would also like to acknowledge the financial support by Thule Rigtech, Shell, Agip and EPSRC (GR/L 08632).

Many thanks are due to all my fellow Ph.D. students. I very much enjoyed the collaborative and productive work environment. I thank Otman Belmahi, Lehel Csato, David J. Evans, Lars U. Hjorth (especially for reviewing one chapter of this thesis), Renato Vicente and Wei-Lee Woon.

Special thanks go to Ragnar Lesch for his helpful comments on my work and several arguments we had during three years.

I am particularly grateful to two good friends: Neil D. Lawrence and Francesco Vivarelli. Thank you, guys, for everything!

Thanks are due also to Palmarosa Cagna, Frederic Petiniot and Laurent Stein for their unceasing support. I am also deeply grateful to Sophie Desmartin and Stephane Touboul for hosting me several times in Paris. Thank you Stephane, for your help, criticism and long-standing friendship.

Obvious thanks go to Laurent Sauveur, a special friend. Thank you Laurent for your daily support, help and attitude towards life. I would never thank you enough, tocard!

Finally, I am very grateful to my family for unflagging moral and financial support and encouragement during all my studies. In particular, I would like to thank Christine, Marie-Claude, Abbas and Aissa.

Contents

List of Tables

List of Figures

Declaration

This thesis describes the work carried out between January 1997 and September 1999 in the Neural Computing Research Group at Aston University under the supervision of Dr. Ian T. Nabney.

The work reported in this thesis has been entirely executed by myself. This thesis has been composed by myself and has not, nor any similar dissertation, submitted in any previous application for a degree.

Publications

Azzouzi, M. and Nabney, I. T. (1999). Dynamical Local Models for Segmentation and Prediction of Financial Time Series. Submitted to *European Journal of Finance.*

Azzouzi, M. and Nabney, I. T. (1999). Time Delay Estimation with Hidden Markov Models. *Proceedings of the Ninth International Conference on Artificial Neural Networks, IEE*, pages 473–478, IEE.

Azzouzi, M. and Nabney, I. T. (1999). Modelling Financial Time Series with Switching State Space Models. *Proceedings of the IEEE/IAFE 1999 Conference on Computational Intelligence for Financial Engineering (CIFEr)*, pages 240–249, IEE/IAFE.

Azzouzi, M. and Nabney, I. T. (1998). Analysing Time Series Structure with Hidden Markov Models. *Neural Networks for Signal Processing*, volumes 8, pages 402–408, IEEE.

Chapter 1

Introduction

From physics to econometrics, our desire to predict the future motivates the search for laws that govern the dynamics of observed phenomena. Observations have been always recorded over time in order to help us understand the changing of our world. Examples range from weather forecasting to financial markets prediction.

A time series is a series of observations taken sequentially over time. In formal terms, a time series is a sequence of vectors $\boldsymbol{y}_t$, depending on time t. The components of the vector can be any observable variable, such as the temperature of the air, the price of a certain commodity, the daily electricity load demand, *etc.* In contrast to a standard regression problem where the order of the observations is irrelevant, it is this order property that is crucial and that distinguishes time series from non time series data.

In theory, it may possible for a time series to be purely deterministic, which means that it can be described by some deterministic equations governing the dynamics of an underlying state vector $\boldsymbol{s}_t$ which will regenerate the past and predict the future values without discrepancies. Unfortunately, it is often the case that the underlying theory is absent and we are left with the data themselves. Moreover, it is important to understand that an observed time series include components that are both deterministic as well as probabilistic in nature. Due to measuring errors, unknown or uncontrollable factors, hardly anything that happens in the real world is deterministic and one almost always has to assume that time series are stochastic in nature.

There are many reasons for wishing to model a time series. Forecasting is probably the most wide-spread application. There are many real-world problems where the aim is to accurately predict the value of a variable. From business to energy planning, investigators try to accurately predict future values in order to decide upon a trading strategy or optimise production.

The other motivation is to provide a description of a time series in terms of its components of interest. Time series analysis is a description of the long-term properties of a process. One may, for example, wish to examine the trend in order to see the main movements which have

taken place in the series.

A third motivation concerns the characterisation of a time series. One may wish to determine the fundamental properties of a time series, such as the number of degrees of freedom of a process or the amount of randomness.

These three goals often overlap. A model which has been developed on observed data points can be used as a description of the time series, its parameters being viewed as a kind of feature set of the series. A model used for description can also be used as the basis for forecasting. However these goals can differ: understanding the long-term behaviour of a process may not be the most reliable way for estimating the parameters of a model for short-term predictions.

Modern time series has been for many years dominated by the so-called linear Gaussian models. Yule (1927) developed the autoregressive models (AR) for predicting sunspots: the value of the observation at time t is a weighted sum of previous observations:

$$y_t = \sum_{i=1}^{p} \lambda_i y_{t-i} + \epsilon_t \tag{1.1}$$

where λ_i are constant, p is the order of the model and ϵ_t is a zero-mean, uncorrelated random variable, also called white noise. A more general case is the family of autoregressive moving average models ARMA(p,q), which consists of replacing the ϵ_t term by a weighted average of $\epsilon_t, \ldots \epsilon_{t-q}$:

$$\sum_{i=0}^{p} \lambda_j y_{t-i} = \sum_{j=0}^{q} \phi_j \epsilon_{t-j} \tag{1.2}$$

Linear Gaussian models have been successful for two main reasons. First, the linearity assumption allows a nice mathematical development and the theoretical properties of these models are now well understood. The traditional time series paradigm in which it is assumed that a time series can be reduced to stationarity[1] by differencing or detrending is another important motivation for considering ARMA models.

However, there are many real-world problems for which the assumptions of linearity and stationarity are not valid. For instance, one of the obstacles to the prediction of exchange rates in the capital markets is a non-constant conditional variance, known as heteroscedasticity. Many industrial plant exhibit also multiple discrete modes of behaviour under different operating conditions.

The development of powerful computers and the desire of uncovering the underlying dynamics of a time series have been essential in the last two decades. The state-space reconstruction by time-delay embedding provides a technique for recognising when a time series has been generated by deterministic equations. The simplest procedure consists of creating

[1] The essential feature of a stationary process is that its properties do not change over time. For example, a second-order stationary stochastic process y_t is one for which the mean and the variance are time independent.

delay vectors $\boldsymbol{y}_{t-k}^{t} = [y_{t-k}, \ldots, y_t]$ from a scalar time series y_t. Having reconstructed a vector time series, investigators try to develop a model which explains the scalar time series y_t in term of the delay vector dynamics. Viewed this way, time series modelling becomes a problem of function approximation and the emergence of adaptive models such as neural networks has been crucial in this sense.

There is actually a strong connection between delay vectors and underlying dynamics. This connection was originally proposed for noise free non-linear dynamical systems (Packard *et al.*, 1980). It turns out that the dynamics of the underlying state-space variable $\boldsymbol{s}_t$ are related to the dynamics of delay vectors. Takens (1981) gave the conditions for such a mapping between $\boldsymbol{s}_t$ and $\boldsymbol{y}_{t-k}^{t}$.

While the framework of embedology was formulated in the limit of infinite amounts of noise free data, hidden state space models have been developed from the desire of modelling hidden state stochastic dynamics $P(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})$ and noisy observations $P(y_t \mid \boldsymbol{s}_t)$. Hidden Markov models and state space models assume explicitly the existence of a discrete or a continuous hidden state, whose dynamics are governed by a Markov process. In the last decade, several descendant models have been suggested. Examples range from speech recognition, control, machine learning and finance. The need to account for non-linearity and non-stationarity has been the main motivation for considering hybrid models.

In this thesis, we consider hidden state models for modelling processes that exhibit a sequential changing behaviour. The underlying generator is believed to switch between different regimes. Within each regime, the time series satisfies (almost) the requirement of stationarity, but between them, it might have different noise level or different dynamics.

Chapter 2 presents the oil well drilling process. This process exhibits complex relationships between different time series and a highly non-stationary behaviour. A significant difficulty during exploration drilling is ensuring the drilling debris is removed from the bore. We present standard filtering techniques for time series and show how a description of the long-term properties of the process can be obtained.

Chapter 3 provides an introduction to the principal concepts of hidden state models. By drawing an analogy between hidden Markov models and state space models, the chapter introduces many of the central ideas, such as hidden state inference, parameter estimation and probability density prediction.

Chapter 4 deals with the problem of time delay estimation which represents an important task in time series analysis. We present a novel approach for estimating the lag or delay between two time series. The approach is based on using hidden Markov models and we show how to estimate a crucial parameter in the oil industry.

Chapter 5 presents variational techniques for probabilistic inference. For complex probabilistic models, inferring the value of some hidden variables is computationally intractable. Variational techniques provide a principled framework for solving this problem.

Chapter 6 introduces dynamical local models. These models combine and generalise hidden Markov models and state space models and therefore are capable of modelling discrete and continuous dynamics. We also present a new extension which incorporates local non-linearity. We demonstrate the application of these models for segmenting a time series into regimes and for providing predictive distributions.

Chapter 7 summarises the work presented in this thesis and provides some future directions of research.

Chapter 2

Monitoring the oil well drilling process

2.1 Introduction

Oil well drilling is a complex and highly skilled process. One significant and expensive aspect of exploration drilling is that of ensuring the drilling debris is effectively removed from the bore. In the case of vertical wells, an adequate velocity of the drilling fluid is generally sufficient to guarantee that most debris is brought to the surface. Current practice within the industry is to drill more 'high angle' or 'horizontally deviated' wells and offsets of several kilometres are quite common. These types of wells include angled or even near horizontal sections, in which gravity settlement of particles in the drilling fluid can occur and create stationary cuttings beds. This significantly increases the operating problem of 'hole cleaning'.

During the past two decades, many laboratory studies as well as field observations have been directed at the cuttings transport problem. These have resulted in a better understanding of the subject. Several models have also been developed which give a tool to improve the specification of the hydraulic requirements to clean the hole. However, due to the complexity of the problem, at present, no comprehensive and proven model exists to act as a monitor of hole cleaning status and the process still relies heavily on the experience of the drilling engineers.

Although several fluid mechanics based predictive models are used in the oil industry, a hole cleaning monitoring system needs to be developed that receives all the available and relevant data in real time for determining the hole status. As discussed in (Pilehvari *et al.*, 1996), it is indeed not prudent to rely only on predictions of such models, because of our poor understanding of downhole conditions.

A new device, the PD-50, developed by Thule Rigtech, is capable of monitoring on-line and in real time the fine particles of rock in drilling fluid. Our proposed approach to improving

the assessment of hole cleaning quality is to take advantage of downhole data and to develop suitable techniques to monitor the distributions of drilled solids. By following their trends and relating them to parameters that can be monitored at the surface, our goal is to obtain a better picture of downhole conditions.

In this chapter, we first describe the drilling process and present the different parameters that are collected for our purposes. In section 2.3, we review the hole cleaning problem and discuss several techniques that have been suggested in the literature. We then present in section 2.4 standard filtering techniques and show how they can be applied in a simple algorithm for stuck pipe detection. Our algorithm follows the trend of low gravity solids and is able to alert the user in advance of a future problem in the hole.

2.2 Oil well drilling

Drilling for oil and gas is a complex and highly skilled operation. If the find is promising, a field will be developed and brought into production. The drilling industry engages the services of thousands of men and a complicated array of machinery and materials and requires a vast network of transportation facilities. On-the-spot decisions which must be made in the course of drilling a well invariably concern the many thousands of pounds invested in the hole already drilled and in the drilling equipment.

Wells are drilled with rotary drilling tools. The cutting tool is the drilling bit which has tough metal, or sometimes diamond, teeth that can bore through the hardest rock. The bit is suspended on a drilling string consisting of lengths of pipe, which are added to as the bit goes deeper. The bit is turned either by a rotary table or, increasingly, by a downhole motor. In time, the bit gets worn and has to be replaced. The whole drilling string, sometimes weighing over 100 tons, must then be hauled to the surface and dismantled section by section as it emerges. The new bit is fitted and slowly lowered as the drill pipe sections are re-assembled.

One of the essential supplies for the drilling crew is *mud*, or drilling fluid. This is a special mixture of clay, various chemicals and water, which is constantly pumped down through the drill pipe and comes out through nozzles in the drilling bit. The stream of mud returns upwards through the space between the drilling string and the borehole, carrying with it rock fragments cut away by the bit. At the top, the returned mud is sieved and then recirculated through a pump. The cuttings left on the sieve indicate the kind of rock the drill is passing through and they may show traces of oil as the bit nears an oil-bearing formation. The drilling mud also keeps the bit cool and prevents the escape of gas or oil when the bit enters an oil trap. Figure 2.1 is a diagrammatic view of a drilling rig with its different elements and shows the circulation of the mud.

The rate of drilling or rate of progress (ROP) varies with the hardness of the rock. Sometimes the bit may cut through as much as 200 feet per hour, but in a very hard layer

Figure 2.1: A rig and its elements.

progress may be as little as 1 foot per hour. Most oil wells are between 3,000 and 16,500 feet deep but wells as deep as four or five miles are sometimes drilled.

Whenever possible, wells are drilled vertically, but sometimes, especially offshore, it is necessary to deviate from vertical in order to reach a wide spread of targets from a single platform. This is known as 'directional drilling'. Recent developments have made it possible to deviate as much as 90 degrees from the vertical. Known as *horizontal drilling*, this technique can, in some instances, increase the productivity of a well.

2.3 Hole cleaning

Ensuring that the drilling cuttings are effectively removed from the bore is a significant and expensive aspect of exploration drilling. The problem is more complicated when drilling deviated wells: the drill string can be lying eccentrically and gravity settlement can occur. The gradual build up of low gravity solids (LGS) increases the torque required to turn the drill string. In extreme cases, the drill pipe may get stuck or even fracture off. Retrieving a stuck pipe is a difficult and expensive operation. Costs of remedial treatment are always high, and may amount to many hundreds of thousands of pounds (Rabia, 1985). In near horizontal wells, the drilling fluid exhibits little or no velocity component in the vertical plane. The hole cleaning problem is more complicated when drilling relatively deviated wells (40° to 80°

from vertical) where the vertical velocity component becomes increasingly important.

Current operational methods for monitoring the drilling process are based on the experience of key personnel using subjective judgments of the appearance and volume of cuttings. Mathematical models have been also developed in order to predict the torque and drag anticipated at various depths of the well. A plot is usually kept to compare the actual figures with the predicted ones and significant variations, when they occur, are used to signal a developing problem.

Guild *et al.* (1995) proposed a hole cleaning program so as to improve drilling performance: in order to estimate the total amount of solids coming to the surface, a record is kept of the time taken to fill a viscosity cup with cuttings. This recording is made from the same place on one of the mud cleaning units, known as 'shale shakers', and requires a full time dedicated engineer to make manual measurements every 20 minutes.

Kenny *et al.* (1996) emphasize the drilling fluid properties and the relationships between the fluid rheology and the particle-settling velocity: for highly deviated wells, a balance must be struck between minimising particle-settling velocity and promotion of fluid velocity. The hole cleaning problem can be alleviated by identifying and adjusting the relevant rheological parameters of the drilling fluids.

Rasi (1994) focuses on hole problems resulting from accumulation of cuttings or cavings on the low side of wellbores with high inclination and large holes. In such wellbores, tall and stationary cuttings beds often form. Such beds do not usually cause problems while the drill bit is being rotated. Problems arise when the drill string is moved axially in or out the hole. They propose a hole cleaning design tool based on fluid mechanics first principles. This qualitative approach enables the optimisation of fluid rheology selection, drill string design and wellbore profile.

Zamora and Hanson (1991), based on laboratory observations and field experience, compiled 28 rules of thumb to improve high-angle hole cleaning.

Pilehvari *et al.* (1996) review in detail all the experimental and qualitative solutions that have been suggested in the literature of cuttings transport in horizontal wellbores. The authors point out the lack of quantitative statistical models and the deficiencies of fluid mechanics models. They conclude that an efficient hole cleaning monitoring system should take advantage of the relevant data in real time.

During drilling operations, valuable information about the field at various depths is collected by a procedure known as 'logging'. Drill cuttings which are returned to the surface are examined for traces of hydrocarbons and for their fossil content. Wireline logs measure the electrical, acoustic and radioactive properties of the rocks which give clues as to the rock type, its porosity and how much fluid it contains. Other quantitative parameters are measured at the surface. It is possible to collect and track a wide variety of drilling parameters, as the rate of progress, the drill string torque (*Torque*) and pressure (*STPP*), the bit depth, etc.

Statistical analysis of drilling data has been conducted by Sifferman and Becker (1992) who examined the effect of different parameters on cuttings transport. It was shown that there are interactions between various parameters, and thus a simple relationship could not be derived.

Thule Rigtech has recently developed a new device, the PD-50, which is capable of detecting and monitoring fine particles in drilling fluid. The principle of the PD is ultra sonic, supported by complex electronic processing. It operates in real time, on-line. The device is sensitive enough to detect material as small as 30 microns across. The operation of the device relies on the exchange of energy which occurs when a small particle impacts the sensor face at speed. The energy of the particle is proportional to the product of its mass and the square of its velocity. The energy lost by the particle on impact is converted by the sensor into an electrical discharge. Thus, a small particle travelling at high speed might produce a similar discharge to a larger particle travelling at a lower speed. To use the device as a measure of size, one of the variables needs to be fixed and it is easier to control the velocity. The PD-50 mud stream is accelerated by a positive displacement pump towards the sensor face. The pressure in the stream is monitored, and used as a proxy for flowrate. By tracking changes in the circulating volumes of fine LGS during active drilling, the PD-50 is also able to detect changes in the rate of progress (ROP). For a complete description of the PD-50, see (Thule Rigtech, 1995).

Using the PD-50, we can follow the trends in the volumes of drilled solids in order to obtain a better picture of downhole conditions with regard to drilled solids than has ever been possible before. Preliminary trials showed that there are empirical relationships between events in drilling and the particle size measurements and volumes (Thule Rigtech, 1995). If a numerical relationship can be developed between levels of fine LGS expected in normal drilling for a given ROP and levels circulating in a known 'clean hole', a numerically based specification of what constitutes a clean hole could be defined.

Other relationships might be explored: for example, where there is excess fine LGS relative to known ROP together perhaps with small fluid losses, this might suggest possible hole wash out[1] or steady reductions in LGS of a certain specification, which might map deteriorations in bit condition.

To improve the assessment of hole cleaning quality, Thule Rigtech gathered on-line information about the changes in LGS volumes in the drilling mud in order to analyse this information for correlations with other drilling parameters, for example, ROP and *Torque*. It has been decided to measure each parameter every 30 seconds. There are reasons to think that the resulting time series may be oversampled. The drilling process manifests a sequen-

[1] Alternating hard and soft formations cause offset ledges. Soft bands are easily drilled and may be washed out by the drilling fluid, an oversize hole being produced. Unwanted deviations are produced with the potential of a severe dog-leg.

tially changing behaviour and the properties of the process are usually held steady, for a certain period of time, and then, at certain instances, change to another set of properties. Using such a small sampling rate can produce highly autocorrelated time series, because of this period of time where the parameters do not fluctuate. It was however important to specify a reasonably small sample rate as data are gathered on-line. It was only possible to take measurements at a well site once; hence in order to avoid missing important information, it was better to oversample the time series.

The data have been collected on a rig in Holland. Unfortunately it often happens that some sensors are 'off' and therefore no measurement is possible for a certain time. These problems occur for example when the drilling fluid pumps have been switched off. Several drilling parameters are directly affected by such an event. Another significant problem occurs when the whole drilling string has to be hauled to the surface. In this case, a new section is added to the string and the whole string is then lowered. These events are quite common during a day and precautions must be taken in order to assess the quality of the collected data. The knowledge of an expert is thus required and fortunately a drilling engineer has produced a quality data control and has identified sections of relevant data. These sections of data points cannot be simply re-assembled in order to create a big dataset: the resulting time series would represent different drilling situations and the temporal dependencies would be destroyed.

For these reasons, it is difficult to have a reasonably large dataset and our statistical analysis has been affected by this problem. For example, it is often the case that the size of the time series is not larger than 400 points. Moreover, electrical certification requirements led to a redesign of equipment and several months were lost because of this.

2.4 Noise reduction and smoothing techniques

One of the major features of the data is the high level of noise and it is worth to distinguish here *white noise* (also called random or Gaussian noise) which occurs with similar amplitudes and *impulse noise* which is a momentary perturbation of the signal. It is also important to note that the relatively small sample rate of 30 seconds can produce a high level of *correlated noise* although no statistical investigation of it has been carried out in our work.

2.4.1 Removing outliers

It often happens that data points are affected by electrical noise. This is particularly significant for the ROP time series: several outliers can be identified, since a lot of these values are greater than 100 ft/hr (an ROP of 45-50 ft/hr is considered to be very good).

In order to remove outliers, a basic and robust approach is based on a median filter. If X denotes a random variable, then the median of the probability density function $P(x)$ is the

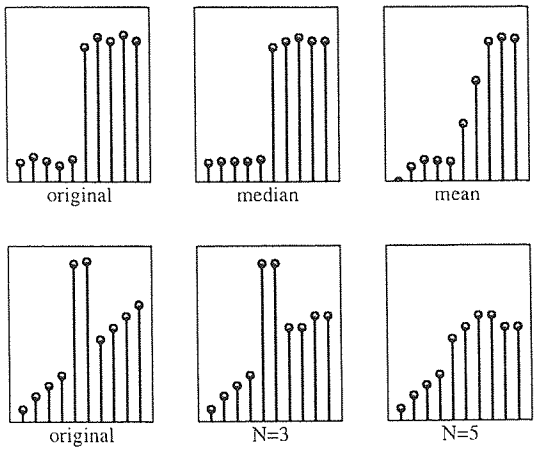


Figure 2.2: The first line shows how median and mean filters perform on a noisy step function. The second line shows the importance of the window size: a window of 3 points is not able to remove the outliers.

value x_{med} for which larger and smaller values of x are equally probable:

$$\int_{-\infty}^{x_{med}} P(x)dx = \frac{1}{2} = \int_{x_{med}}^{-\infty} P(x)dx. \tag{2.1}$$

In other words, the median of a distribution is estimated from a sample of values $x_1, \ldots, x_n$ by finding that value x_i which has equal numbers of values above it and below it. One useful characteristic of median filtering is its ability to preserve signal edges while filtering out impulses. Thus, instead of using the mean, which fails as an estimator of the central value for values drawn from a probability distribution with very broad 'tails', we can use the median of the distribution to estimate the central value. This is particularly useful when the sample contains a lot of outliers: the median will not take into account the values that are very far from the central value, contrary to the mean.

In practice, a median filter consists of sliding a window of an odd number N of elements along the signal, replacing the centre sample by the median of the samples in the window. Figure 2.2 shows how filters based on the mean and the median perform on a simple example. The first line represents a noisy step function and the outputs we can obtain with a window size of 3 points. The median filter preserves the edge of the function whereas the mean filter smooths it completely. The second line shows a noisy straight line with two significant outliers and how a correct choice of the window size is important for the median filter to perform well.

In order to filter out impulse noise, we therefore have performed a median filter on the different time series. As said previously, the value of N is important: a large value for N will give a very *smooth* time series, and in extreme cases the shape of the original time series will be lost. On the contrary, a small value for N will not help to remove outliers.

2.4.2 Moving averages

A moving average (MA) is a simple technique for calculating the average value of a time series. The term *moving* implies that the average changes or moves, There are many different types of moving averages: arithmetic, exponential and weighted. The only difference between these various types is the weight assigned to the recent data. We describe here the weighted MA which is the most common moving averaging technique.

When applying weighted moving averages, we basically fit a polynomial to the first set of terms, say $2m+1$, and use the polynomial to determine the trend value at the $(m+1)th$ point, the middle of the range of that set. We then fit the same order of polynomial to the $2nd, 3rd, \ldots, (2m+2)th$ observations and determine the trend value at the $(m+2)th$ point and so on. It can be shown that the procedure is simply equivalent to taking linear combinations of the observations with coefficients or weights which can be tabulated in a standard form:

$$\hat{x}_t = \sum_{j=-m}^{m} \alpha_j x_{t+j}. \tag{2.2}$$

This technique of averaging a time series is often used in econometrics in order to get the trend of a time series. We will see in the next section how to use this technique for monitoring the oil well drilling process.

2.5 Stuck pipe detection

The main motivation for the project was to use the PD-50 to measure the fine particles of rock and thus monitor downhole conditions. In particular, we hoped that analysis of PD-50 data would indicate when a stuck pipe is likely to occur. Figure 2.3 plots the LGS time series (total amount of low gravity solids) over one day. At 17h50, the rig was stuck in the hole and at 18h00 circulating operations started. It can be seen that the build up is gradual till the rig is stuck.

We propose a simple algorithm in order to detect such a crucial problem. As a significant and long lasting increase of the amount of low gravity solids is the key indicator of this fault, we propose to extract the trend from the data and apply elementary rules to determine if the trend is significant. The quality data control made by Thule Rigtech enables us to slice the data and remove sections of invalid data. A median filter is then used in order to remove outliers. We then apply moving averages to get the trend of the time series. In this work, we implemented a simple unweighted moving average ($\alpha_j = \frac{1}{2m+1}$[2]). The algorithm

[2]The value of m should be chosen in order to reflect the major trends in the time series. In this work, we chose $m = \frac{1}{2}$ hr although bigger values such as $m = 1$ hr or $m = 2$ hr gave simular results.

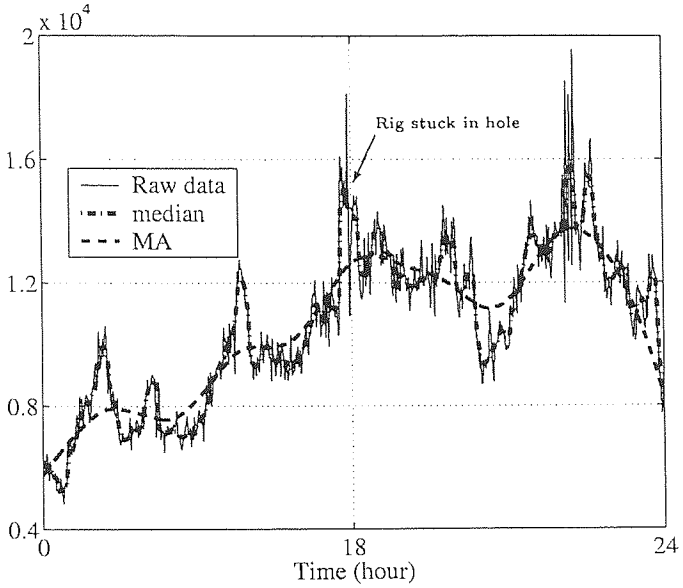


Figure 2.3: Trend of *LGS* before and after a stuck pipe. Sections of invalid data have been removed.

is essentially based on computing the fractional increase of low gravity solids over a certain period (algorithm 1).

The variable *occurrences* is used to detect 2 consecutive significant increasing trends of *LGS*. The rule based on the fractional increase allows an increase between the range [min, max][3] over 2 hours. When applied on the data we have collected, it is possible to detect substantial problems. We have tested our algorithm on the whole database. This represents twenty days of collection. Amongst all the data, the algorithm has detected precisely two significant increasing trends of *LGS*. For the first event, the code detects 2 consecutive gradual increases of *LGS* of 23% and 25% (i.e. 54% over 4 hours), informing the user at 15h00 (i.e. 3 hours before the fault) that the pipe may get stuck (see Figure 2.3). For the second event, although the pipe was not stuck at the end of the day, the code has detected an increase of 87% over 4 hours, which could have resulted in a stuck rig. More data are obviously needed for properly assessing the performance of the algorithm.

Finally, it should be noticed that the on-line implementation of this algorithm is straightforward and is computationally efficient. Also, although this algorithm gives good results on our database, an improved version can be obtained by using a smoother version of the fractional increase: instead of computing the fractional increase every 2 hours, we can compute the gradient at each time and average it over 2 hour.

[3] As we remove sections of invalid data, big jumps of PD-50 data can happen from one period to another one. Such an upper bound is meant to ignore them. In this work, we used min = 20% and max = 50% as suggested by a drilling engineer.

Algorithm 1 Stuck Pipe Detection

```
occurrences = 0
loop
  Filter the data using a median filter.
  Smooth the time series using moving averages.
  Compute the fractional increase every 2 hour:
  Δ(x̂_t) = (x̂_t − x̂_{t−1}) / x̂_{t−1}
  if Δ(x̂_t) > min and Δ(x̂_t) < max then
    occurrences = occurrences + 1
  else
    occurrences = 0
  end if
  if occurrences = 2 then
    Stop drilling: detecting a forthcoming problem!!!
  end if
end loop
```

2.6 Discussion

We have outlined some of the problems raised by the analysis of oil well drilling data. Drilling is a complex real-world process and many aspects of the hole cleaning need to be carefully studied and researched. In this chapter, we have presented the process and standard techniques for filtering the different time series. One of the main goals for the project was the stuck pipe detection and we have shown how to design a simple and efficient algorithm which can help the engineers on the rig for monitoring the drilling process.

Collecting oil well drilling data is a difficult operation. Because the data are collected during drilling operations, it is often the case that portions of the data are missing simply because the devices are not running or because the process is stopped for a specific reason. The amount of data used in our studies was very small and, as we saw, it is difficult to assess properly the performance of the algorithm we proposed.

Chapter 3

Hidden state models for time series

3.1 Introduction

A multivariate time series is a sequence of continuous d-dimensional random vectors $\boldsymbol{Y}$ indexed by a time variable t. Suppose that at time T we have seen a sequence of observations $\mathcal{Y}_1^T = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t, \ldots, \boldsymbol{y}_T]$[1] and that we wish to build a model from the sequence $\mathcal{Y}_1^T$ that enables us to predict the value of $\boldsymbol{Y}$ at time $T+1$. Unfortunately, in most realistic cases, the observations are not *deterministically* related. Moreover, because of the finite and limited size of the data, there will be undoubtedly a mismatch between our model and the true process.

Probability theory is a powerful tool for expressing uncertainty and randomness in our model. A statistical model is based on certain probabilistic assumptions that attempt to capture the essential characteristics of the data generation process. We do not believe that a model will represent exactly the true process but we hope we will be able to develop tools that enable us to make decisions for new data points. Given a model and a sequence of observations, the main goal of time series modelling is to estimate the parameters of the model by a statistical procedure, such as *maximum likelihood*, make point predictions and define error bars for unseen data. Other goals are possible: for example if the underlying generator is believed to *switch* between different regimes, it is quite important to develop techniques that *segment* the time series in order to identify which regime was responsible at a certain time t.

The classical theory of maximum likelihood estimation assumes that the observations $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T$ are independently and identically distributed, which allows us to write the joint

[1] Throughout this thesis, a sequence of observations will be denoted $\mathcal{Y}_i^j = [\boldsymbol{y}_i, \boldsymbol{y}_{i+1}, \ldots, \boldsymbol{y}_j]$. We will use the notation $P(\boldsymbol{y}_t)$ to denote both the probability density function for a continuous random vector and the probability $P(\boldsymbol{Y}_t = \boldsymbol{y}_t)$ that the discrete random variable $\boldsymbol{Y}_t$ takes the value of $\boldsymbol{y}_t$ at time t.

distribution as:

$$P(\mathcal{Y}_1^T) = \prod_{t=1}^{T} P(\boldsymbol{y}_t). \tag{3.1}$$

By definition, the main characteristic of a predictable time series is that the observations are *not* independent. Hence Equation (3.1) is not applicable. We can however always factorise the joint probability as:

$$P(\mathcal{Y}_1^T) = P(\boldsymbol{y}_1) \prod_{t=2}^{T} P(\boldsymbol{y}_t \,|\, \mathcal{Y}_1^{t-1}), \tag{3.2}$$

which makes explicit the relationship between the observation $\boldsymbol{y}_t$ at time t and the history of the sequence. The intractability of such a general model is clear: it is in general impossible to develop a model which takes into account the whole history of the time series.

In order to circumvent this problem, one must assume that the influence of the past observations is summarised by some function of previous observations:

$$P(\boldsymbol{y}_t \,|\, \mathcal{Y}_1^{t-1}) = P(\boldsymbol{y}_t \,|\, f(\mathcal{Y}_{t-k}^{t-1})), \tag{3.3}$$

which tells that the probability of observing $\boldsymbol{y}_t$ is a function of k previous observations $\boldsymbol{y}_{t-1}, \cdots, \boldsymbol{y}_{t-k}$.

The technique of *embedology* is inspired by non-linear dynamical system theory (Packard *et al.*, 1980). In this framework, we assume that the data are generated from a finite dimensional system whose dynamics are described in a d-dimensional manifold. d is the effective degree of freedom of the system. Unfortunately the whole variables describing the system are often not accessible; only one variable represented by a time series is observable. However, Takens' theorem allows us to reconstruct the underlying dynamical system or more precisely an *embedding* space where the variables exhibit a behaviour similar to the original ones. The technique is based on constructing delay vectors $[\boldsymbol{y}_{t-1}, \ldots, \boldsymbol{y}_{t-k}]$ and Takens' theorem gives the conditions for this to be an embedding (Takens, 1981). It is then possible to describe the observation $\boldsymbol{y}_t$ with respect to the previous observations $\boldsymbol{y}_{t-1}, \cdots, \boldsymbol{y}_{t-k}$. For example, an *autoregressive* (AR) model of order k assumes that the observation y_t is a linear function of k previous observations,

$$y_t = \sum_{i=1}^{k} \lambda_i y_{t-i} + \epsilon, \tag{3.4}$$

where ϵ is a white noise term ($\epsilon \sim \mathcal{N}(0, \sigma^2)$).

This technique has been successfully applied in non-linear dynamical systems and neural networks (Lowe and Webb, 1994). There are however several practical problems that need

to be addressed. First, the theory has been developed for deterministic dynamical systems, while most real-world data contain a high level of noise. Moreover, the number of delays that must be taken into account in order to describe the dynamics of the system is often not known *a priori*. Techniques have been however developed to circumvent this problem and estimate the intrinsic dimension of the system (Broomhead and King, 1986; Lowe and Hazarika, 1997).

A more general approach assumes that the past sequence can be summarised in a *state* variable which carries all the information contained in $\mathcal{Y}_1^{t-1}$:

$$P(\boldsymbol{y}_t \mid \mathcal{Y}_1^{t-1}) = P(\boldsymbol{y}_t \mid \boldsymbol{s}_t). \tag{3.5}$$

The state variable $\boldsymbol{s}_t$ is often called *hidden* because it is not visible. In this framework, we believe that the dynamics of the data generating process are described by the hidden state variable. We do not assume that the *observed* data points $\boldsymbol{y}_t$ follow the conditional independence property given by Equation (3.3); however another, *unobserved* variable is assumed to exist and to have this property.

In the simplest case of $k = 1$, the state variable follows a first-order Markov process[2],

$$P(\boldsymbol{s}_t \mid \boldsymbol{s}_1^{t-1}) = P(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}), \tag{3.6}$$

and is completely defined by the initial probability $P(\boldsymbol{s}_1)$ and the state *transition* probability $P(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})$. When the state variable is discrete, the model is called an *hidden Markov model* (HMM). In the case of a continuous variable, the model is known as a *state space model* (SSM).

Due to their flexibility and to the simplicity and efficiency of their parameter estimation algorithms, HMMs and SSMs have proven to be the most widely used tools for learning probabilistic models of time series data. In this chapter, we review both models, describe how they can be trained in a maximum likelihood approach and show their use for time series modelling.

3.2 Hidden Markov models

Given an observation sequence $\mathcal{Y}_1^T$, our goal is to model the probabilistic distribution from which the time series was generated. Let S be a discrete random variable taking values in the set $\{q_1, \ldots, q_N\}$ and assume that the system at any time t is in one and only one of the N

[2]Markov models quickly become intractable for large values of k. A k-order Markov model where the state $\boldsymbol{s}_t$ is a multinomial variable $\boldsymbol{s}_t \in \{q_1, \ldots, q_N\}$ would require N^{k+1} transition probabilities.

states $q_1, \ldots, q_N$. The random variable $\boldsymbol{Y}_t$ can be considered to be a probabilistic function of the underlying states, i.e. $\boldsymbol{y}_t$ is an observed measurement from the system but the underlying states are not themselves directly observable. Assuming that the state variable S_t is a stationary discrete-time first-order Markov process, the resulting model is a doubly stochastic process and is called a first-order hidden Markov model (HMM). It is called hidden because the state of the underlying process is *not* observable, but can only be inferred indirectly through another set of stochastic processes that produce the sequence of observations. Thus the model assumes two sets of conditional independence relations: that $\boldsymbol{Y}_t$ is independent of all other random variables given S_t and that S_t is independent of $S_1, \ldots, S_{t-2}$ given S_{t-1} (the Markov property). Using these independence relations, the joint probability for the sequence of states and observations can be written as

$$P(S_1^T, \mathcal{Y}_1^T) = P(s_1)P(\boldsymbol{y}_1 \mid s_1) \prod_{t=2}^{T} P(s_t \mid s_{t-1})P(\boldsymbol{y}_t \mid s_t). \tag{3.7}$$

To illustrate the concept of a hidden Markov model, let us consider the following example inspired from the speech recognition community: given a sequence of acoustic features (which have been obtained by preprocessing the speech signal), a speech recognition model attempts to recover the sequence of words that the speaker intended to pronounce. The usual approach is to define a language model which specifies the structural dependencies of the sequence of symbols. This is often defined by a graphical model which represents the structural relationships between the words and phonemes of the language. A simple Markovian interpretation can then be derived from this graphical model, namely by defining the transition probability from one word to another. Each word or phoneme (i.e. each hidden state) is characterised by a probability distribution of its acoustics features.

HMMs have been successfully applied in speech recognition (Bahl *et al.*, 1983; Rabiner, 1989), cryptography and more recently in other areas such protein classification, sequence alignment (Baldi *et al.*, 1993) and fault detection (Smyth, 1994).

In general, the parameters of a specific model are referred as $\boldsymbol{w} = \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\Pi}\}$, where $\boldsymbol{A} = \{a_{ij}\}$ denotes the state transition matrix, $\boldsymbol{B} = \{b_i(\boldsymbol{y}_t)\}$ the observation probability distribution in each state and $\boldsymbol{\Pi} = \{\pi_i\}$ the initial state distribution:

$$a_{ij} = P(S_t = q_j \mid S_{t-1} = q_i), \tag{3.8a}$$

$$b_i(\boldsymbol{y}_t) = P(\boldsymbol{y}_t \mid S_t = q_i), \tag{3.8b}$$

$$\pi_i = P(S_1 = q_i). \tag{3.8c}$$

The matrix $\boldsymbol{A}$ defines the structure or the topology of the model and is often summarised in

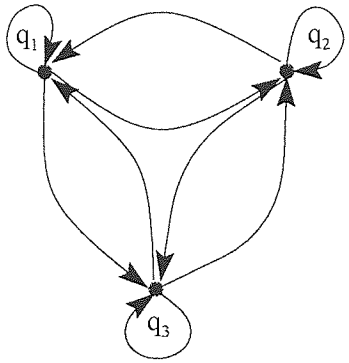


Figure 3.1: A Markov chain with 3 states. Each node represents a hidden state. The structure of the model is represented by the edges which specify the probability transition from one state to another.

a graph as the one in Figure 3.2. In this example, the states are interconnected is such a way that every state can be reached from any other state.

For time series modelling, the observation probability distributions $\boldsymbol{B}$ are often chosen to be a mixture of Gaussians, as such models can approximate, arbitrarily closely, any finite, continuous density function, provided that enough components are used:

$$b_i(\boldsymbol{y}_t) = \sum_{p=1}^{P} c_{ip} \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\mu}_{ip}, \boldsymbol{\Sigma}_{ip}), \tag{3.9}$$

where $\boldsymbol{y}_t$ is the multivariate observation to be modelled, c_{ip} is the mixture coefficient for the p-th mixture in state i and $\mathcal{N}$ is a Gaussian density function with mean $\boldsymbol{\mu}_{ip}$ and covariance matrix $\boldsymbol{\Sigma}_{ip}$.

$$\mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\mu}_{ip}, \boldsymbol{\Sigma}_{ip}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}_{ip}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{y}_t - \boldsymbol{\mu}_{ip})' \boldsymbol{\Sigma}_{ip}^{-1} (\boldsymbol{y}_t - \boldsymbol{\mu}_{ip})\right). \tag{3.10}$$

So far, we have presented the model without reviewing the main problems that must be solved for the model to be useful in real-world application. Because state space models are similar models, we prefer to present them in the next section. In section 3.4 and section 3.5, we will review the inference and parameter estimation problems for both models and show how these problems can been tackled in a similar framework.

3.3 State space models

The major difference between a hidden Markov model and a state space model relies on the property of the hidden state variable. Whereas the hidden state is *discrete* in an HMM, the variable s_t in a state space model belongs to a *continuous* space.

A state space model (SSM), also known as a linear dynamical system (LDS), is described

by the equations:

$$\boldsymbol{s}_t = \boldsymbol{F}_t \boldsymbol{s}_{t-1} + \boldsymbol{u}_t, \tag{3.11}$$

$$\boldsymbol{y}_t = \boldsymbol{G}_t \boldsymbol{s}_t + \boldsymbol{v}_t, \tag{3.12}$$

where $\boldsymbol{s}_t$ is an *unobserved* state vector, $\boldsymbol{s}_t \in \mathbb{R}^m$, $\boldsymbol{y}_t$ is the *observed* vector, $\boldsymbol{y}_t \in \mathbb{R}^d$ and $\boldsymbol{u}_t$ and $\boldsymbol{v}_t$ are uncorrelated zero-mean Gaussian vector processes with covariances $\boldsymbol{Q}_t$ and $\boldsymbol{R}_t$. These equations can be rewritten as:

$$P(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}) = \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{F}_t \boldsymbol{s}_{t-1}, \boldsymbol{Q}_t), \tag{3.13}$$

$$P(\boldsymbol{y}_t \mid \boldsymbol{s}_t) = \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{G}_t \boldsymbol{s}_t, \boldsymbol{R}_t). \tag{3.14}$$

The initial state is also assumed to be a Gaussian random variable, $P(\boldsymbol{s}_1) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$. Exogenous inputs can be included in the state equation but we shall omit such terms for the sake of simplicity as they are of the same form as the noise term.

The major motivation for such a model comes from the desire to model separately the uncertainties in the model defined by the noise $\boldsymbol{Q}_t$ and the uncertainties in the measurements expressed by the output noise matrix $\boldsymbol{R}_t$. In order to illustrate where a state space model might be used, consider the autoregressive (AR) model of second order, $y_t = \lambda_1 y_{t-1} + \lambda_2 y_{t-2} + \epsilon_t$. A simple transformation allows us to rewrite it as:

$$y_t = \begin{pmatrix} \lambda_1 & \lambda_2 \end{pmatrix} \begin{bmatrix} s^1 \\ s^2 \end{bmatrix}_t + u_t \tag{3.15}$$

$$\begin{bmatrix} s^1 \\ s^2 \end{bmatrix}_t = \begin{pmatrix} \lambda_1 & \lambda_2 \\ 1 & 0 \end{pmatrix} \begin{bmatrix} s^1 \\ s^2 \end{bmatrix}_{t-1} + v_{t-1}, \tag{3.16}$$

by introducing a 2-dimensional state vector $\boldsymbol{s}_t$ and assuming $v_t = u_t = \epsilon_t$. It can be shown actually that any ARMA model $\sum_{i=0}^{p} \lambda_i y_{t-i} = \sum_{j=0}^{p} \phi_j \epsilon_{t-j}$, where ϵ_t is a noise term, can be expressed in an equivalent state space form with constant matrices $\boldsymbol{F}$ and $\boldsymbol{G}$. In the reminder of this thesis, we will consider models where the transition and output matrices $\boldsymbol{F}_t$ and $\boldsymbol{G}_t$ are time invariant. Noise covariance matrices will also be considered as not depending on t.

There are many examples from econometrics (Pole *et al.*, 1994) to engineering (Anderson and Moore, 1979) where the state space model has been successfully applied. A good example comes from surveillance systems, where the goal is to track a large number of objects using measurement data from many diverse sensors. The goal of a typical tracking system is to estimate the current state of an object from noisy measurements. Here the state is the velocity

or the acceleration of the target. The measurements are noise-corrupted observations such as the position, the azimuth or signals from several sensors (Bar-Shalom and Li, 1993).

In these applications, the parameters of the model $\boldsymbol{w} = \{\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ are often known in advance. For instance, in the tracking problem, the system equation (Equation (3.11)) represents the dynamics of a physical phenomenon: the trajectory of an object. In this case, the goal is not to learn these parameters but rather to estimate the trajectory of the hidden state, which is known in control and engineering as *filtering*. This problem is very similar to the problem of recovering the sequence of hidden states in HMMs. In the next section, we will see how to solve this *inference* problem. We will also see that making *predictions* is a special case of this problem. We will then concentrate on the learning problem for both models and show how parameter estimation can be obtained in the same framework, *i.e.* by using the EM (Expectation-Maximisation) algorithm (Dempster *et al.*, 1977) which is a powerful algorithm for models involving *missing* variables.

3.4 Inference

Given a model with known parameters $\boldsymbol{w}$ and a sequence of observations $\mathcal{Y}_1^T$, the inference problem consists of estimating the posterior probabilities of the hidden variables.

For HMMs, two algorithms are commonly used to solve two different forms of the inference problem (Rabiner, 1989). The first computes the posterior probabilities of the hidden states $P(s_t \mid \mathcal{Y}_1^T)$, for $t \leq T$ using a recursive algorithm known as the *forward-backward* algorithm. The other inference problem is to find a sequence of hidden states which 'best' explains the observation sequence. The most widely used criterion is to find the single most likely path, *i.e.* to maximise $P(\mathcal{S}_1^T \mid \mathcal{Y}_1^T, \boldsymbol{w})$ which is equivalent to maximising the joint probability $P(\mathcal{S}_1^T, \mathcal{Y}_1^T \mid \boldsymbol{w})$. For this purpose, the *Viterbi* algorithm (Viterbi, 1967), a particular case of dynamic programming, offers an efficient solution. Typical uses might be to learn about the structure of the model and to segment the time series into different regimes if we believe that the system operates in multiple modes and switches its dynamics.

For SSMs, the inference problem is usually divided into special cases in the engineering literature: *filtering*, *smoothing* and *prediction* (Anderson and Moore, 1979). The aim of filtering is to compute the probability of the hidden state given the observations sequence up to time t, i.e. to compute $P(s_t \mid \mathcal{Y}_1^t, \boldsymbol{w})$. In the case of a linear dynamical system with Gaussian assumption there exists an optimal recursive algorithm: the *Kalman filter* (Kalman and Bucy, 1961). Smoothing differs from filtering in that the information made available after time t

is used for estimation the hidden state. Here we are interested in estimating $P(s_t \mid \mathcal{Y}_1^T)$, for $t \leq T$. There exists also a recursive algorithm similar to the forward-backward algorithm for HMM which is called the *Kalman smoother*, also known as the Rauch-Tung-Streibel smoother (Rauch, 1963). Finally, the goal of prediction is to compute the probability $P(s_t \mid \mathcal{Y}_1^T)$, for $t > T$. Here measurements up to time t can be used and the model is simulated in the forward direction.

Because HMMs and SSMs belong to the same family of probabilistic models, it is worth mentioning that the forward-backward and Kalman smoothing algorithms are special cases of exact inference for more general graphical probabilistic models (Pearl, 1988; Smyth *et al.*, 1997).

Recall that for a first-order Markov model the joint probability can be factored as:

$$P(\mathcal{S}_1^T, \mathcal{Y}_1^T) = P(s_1)P(y_1 \mid s_1) \prod_{t=2}^{T} P(s_t \mid s_{t-1})P(y_t \mid s_t). \tag{3.17}$$

By marginalising the distribution[3] $P(y_1, \ldots, y_T)$, we see that

$$P(\mathcal{Y}_1^T) = \int P(s_t, \mathcal{Y}_1^T) ds_t. \tag{3.18}$$

The inference algorithm is based on a recursive equation over the probability $P(s_t, y_1^T)$, instead of $P(s_t \mid y_1^T)$. This latter conditional probability can immediately be obtained by using the fact that:

$$P(s_t \mid \mathcal{Y}_1^T) = \frac{P(s_t, \mathcal{Y}_1^T)}{\int P(s_t, \mathcal{Y}_1^T) ds_t}. \tag{3.19}$$

Equation (3.18) and Equation (3.19) show that the joint distribution $P(s_t, \mathcal{Y}_1^T)$ is all we need to solve the inference problem. Moreover, as we will see, the likelihood of the observation sequence y_1^T can straightforwardly be computed as well. At each time t, because of the Markov property, we first note that the variable s_t separates the chain into three parts.

$$\begin{aligned} P(s_t, \mathcal{Y}_1^T) &= P(\mathcal{Y}_1^{t-1}, s_t)P(y_t \mid s_t)P(\mathcal{Y}_{t+1}^T, s_t) \\ &= P(\mathcal{Y}_1^t, s_t)P(\mathcal{Y}_{t+1}^T \mid s_t) \\ &= \alpha_t(s_t)\beta_t(s_t), \end{aligned} \tag{3.20}$$

where we have defined the parameters $\alpha_t(s_t) = P(\mathcal{Y}_1^t, s_t)$ and $\beta_t(s_t) = P(\mathcal{Y}_{t+1}^T \mid s_t)$, borrowing the notation from (Rabiner, 1989). *Forward* recursive equations for α can be obtained by

[3] In order to keep the development as general as possible, the random variable S_t is taken to be continuous. The framework applies for HMMs by simply replacing the integral ($\int$) by a sum ($\sum$) over the hidden states. Theoretical arguments for mixing discrete and continuous random variables can be derived by introducing *Lebesque-Stieltjes* integrals (Grimmett and Stirzaker, 1982).

observing that:

$$\begin{aligned}
\alpha_t(s_t) &= P(\mathcal{Y}_1^t, s_t) \\
&= P(\boldsymbol{y}_t \mid s_t) P(\mathcal{Y}_1^{t-1}, s_t) \\
&= P(\boldsymbol{y}_t \mid s_t) \int P(\mathcal{Y}_1^{t-1}, s_{t-1}, s_t) ds_{t-1} \\
&= P(\boldsymbol{y}_t \mid s_t) \int P(s_t \mid s_{t-1}) \alpha_{t-1}(s_{t-1}) ds_{t-1},
\end{aligned} \tag{3.21}$$

where again we have used the first-order Markov property. Similarly, *backward* recursive equations are obtained for β:

$$\begin{aligned}
\beta_{t-1}(s_{t-1}) &= P(\mathcal{Y}_t^T \mid s_{t-1}) \\
&= \int P(s_t \mid s_{t-1}) P(\mathcal{Y}_t^T \mid s_t) ds_t \\
&= \int P(s_t \mid s_{t-1}) P(\boldsymbol{y}_t \mid s_t) \beta_t(s_t) ds_t.
\end{aligned} \tag{3.22}$$

These recursive equations are initialised in the following way:

$$\alpha_1(s_1) = P(s_1) P(\boldsymbol{y}_1 \mid s_1), \tag{3.23}$$

$$\beta_T(s_T) = 1. \tag{3.24}$$

The α and β parameters contain all the information we need to compute the statistics of any order. For example, in the learning problem, we will see that we need the statistics of the second order $\xi_{t-1}(i,j) = P(s_{t-1}, s_t \mid \mathcal{Y}_1^T)$, *i.e.* the probability of being in state i at time $t-1$ and moving to state j at time t given the whole sequence $\mathcal{Y}_1^T$. This can be computed by observing that:

$$\begin{aligned}
P(s_{t-1}, s_t \mid \mathcal{Y}_1^T) &= \frac{P(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}, \boldsymbol{y}_t, \ldots, \boldsymbol{y}_T \mid s_{t-1}, s_t)}{P(\mathcal{Y}_1^T)} \\
&= \frac{P(\mathcal{Y}_1^{t-1}, s_{t-1}) P(s_t \mid s_{t-1}) P(\boldsymbol{y}_t \mid s_t) P(\boldsymbol{y}_{t+1}^T \mid s_t)}{P(\mathcal{Y}_1^T)} \\
&= \frac{\alpha_{t-1}(s_{t-1}) P(s_t \mid s_{t-1}) P(\boldsymbol{y}_t \mid s_t) \beta_t(s_t)}{P(\mathcal{Y}_1^T)}.
\end{aligned} \tag{3.25}$$

3.4.1 Inference for HMMs

In the case of an HMM, the hidden variable s_t is discrete and takes values from a set $\{q_1, \ldots, q_N\}$.

The forward-backward algorithm

The recursive equations for the α and β parameters are immediately obtained by:

$$\alpha_t(i) = P(\boldsymbol{y}_t \mid S_t = q_i) \sum_{j=1}^{N} a_{ji} \alpha_{t-1}(j), \tag{3.26}$$

$$\beta_{t-1}(i) = \sum_{j=1}^{N} a_{ij} P(\boldsymbol{y}_t \mid S_t = q_j) \beta_t(j). \tag{3.27}$$

Defining $\gamma_t(i) = P(S_t = q_i \mid \mathcal{Y}_1^T)$ and $\xi_{t-1}(i,j) = P(S_{t-1} = q_i, S_t = q_j \mid \mathcal{Y}_1^T)$, and using Equation (3.19), Equation (3.20) and Equation (3.25), we get:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_j \alpha_t(j)\beta_t(j)}, \tag{3.28}$$

$$\xi_{t-1}(i,j) = \frac{\alpha_{t-1}(i) a_{ij} b_j(\boldsymbol{y}_t) \beta_t(j)}{\sum_{i,j} \alpha_{t-1}(i) a_{ij} b_j(\boldsymbol{y}_t) \beta_t(j)}. \tag{3.29}$$

It is also worth noticing that the likelihood $P(\boldsymbol{y}_1^T)$ can be easily computed via the α parameters:

$$P(\mathcal{Y}_1^T) = \sum_{i=1}^{N} \alpha_T(i). \tag{3.30}$$

Predictions

Given an observation sequence $\mathcal{Y}_1^T$, the goal of the one-step ahead prediction is to estimate the predictive distribution $P(\boldsymbol{y}_{T+1} \mid \mathcal{Y}_1^T)$. This can be done by first estimating the predictive probability $\rho_{T+1}(i) = P(S_{T+1} = q_i \mid \mathcal{Y}_1^T)$ of reaching each state i at time $T+1$:

$$\rho_{T+1}(i) = \frac{P(S_{T+1} = q_i, \mathcal{Y}_1^T)}{P(\mathcal{Y}_1^T)} \tag{3.31}$$

$$= \frac{\sum_j P(S_T = q_j, \mathcal{Y}_1^T) P(S_{T+1} = q_i \mid S_T = q_j)}{P(\mathcal{Y}_1^T)} \tag{3.32}$$

$$= \frac{\sum_j \alpha_T(j) a_{ji}}{\sum_j \sum_i \alpha_T(j) a_{ji}}. \tag{3.33}$$

The predictive density distribution $P(\boldsymbol{y}_{T+1} \mid \mathcal{Y}_1^T)$ is then a weighted sum of each individual output density $P(\boldsymbol{y}_{T+1} \mid S_{T+1} = q_i)$:

$$P(\boldsymbol{y}_{T+1} \mid \mathcal{Y}_1^T) = \sum_{i=1}^{N} \rho_{T+1}(i) P(\boldsymbol{y}_{T+1} \mid S_{T+1} = q_i). \tag{3.34}$$

The Viterbi algorithm

Finally we mention another important inference problem for HMMs. In some applications, the hidden state is associated with a particular meaning (*e.g.* words or phonemes in speech

recognition). Here the goal is to infer the *most likely* sequence. This is done by finding the sequence $\mathcal{S}_1^{T*}$ of hidden states which maximises $P(\mathcal{S}_1^T, \mathcal{Y}_1^T)$:

$$\mathcal{S}_1^{T*} = \arg\max_{\mathcal{S}_1^T} P(\mathcal{S}_1^T, \mathcal{Y}_1^T). \tag{3.35}$$

The *Viterbi* algorithm (Viterbi, 1967) offers an efficient solution for this problem. The algorithm is very similar to the forward procedure; the difference is the maximisation over the previous states instead of the sum in Equation (3.26). The most likely sequence is obtained by backtracking at each time t the 'best' state. This is done by first defining:

$$\delta_t(i) = \max_{\mathcal{S}_1^{t-1}} P(\mathcal{S}_1^{t-1}, S_t = q_i, \mathcal{Y}_1^t), \tag{3.36}$$

and recursively computing:

$$\delta_t(i) = P(\boldsymbol{y}_t \mid S_t = q_i) \max_j [a_{ji}\delta_{t-1}(j)] \tag{3.37}$$

$$\psi_t(i) = \arg\max_j [a_{ji}\delta_{t-1}(j)]. \tag{3.38}$$

$\psi_i(t)$ records the 'best' previous state at time t and the optimal sequence of states is obtained in a backward recursion with $s_{t-1}^* = \psi_t(s_t^*)$ starting from $s_T^* = \arg\max_i \delta_T(i)$. In the next chapters, we will see how to use this algorithm for initialising HMMs and recovering a sequence of states (see Chapter 4).

3.4.2 Inference for SSMs

When the hidden variable $\boldsymbol{s}_t$ is continuous, the integral at each time step must be solved analytically for a tractable algorithm. This leads to restrictions on the transition $P(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})$ and output probability densities $P(\boldsymbol{y}_t \mid \boldsymbol{s}_t)$. The linear-Gaussian assumption circumvents this problem[4].

The Kalman smoother

The forward procedure is known as the Kalman filter, which is the optimal estimator of the state vector at each time step t based on the information available at time t. It can be shown that the mean of the conditional distribution $P(\boldsymbol{s}_t \mid \mathcal{Y}_1^t)$ is an optimal estimator of $\boldsymbol{s}_t$ in the sense that it minimises the mean square error (Anderson and Moore, 1979). Moreover, when the normality assumption is dropped[5], the Kalman filter is still an optimal estimator in the sense that it minimises the mean square error within the class of all linear estimators. Based

[4] Actually, the recursive equations are still tractable for any distribution belonging to the exponential family.
[5] In that case, there is no guarantee that the Kalman filter will give the conditional mean of the state vector.

on the assumption that the disturbances and the initial state are normally distributed, the recursion equations of α allow us to estimate the mean and the covariance of the Gaussian probability distribution $P(s_t \mid \mathcal{Y}_1^t) = \mathcal{N}(s_t; s_{t|t}, \boldsymbol{P}_{t|t})$.

The recursive equations decouple naturally into recursions for the mean $s_{t|t}$ and the covariance $\boldsymbol{P}_{t|t}$. Rewriting Equation (3.21) for the continuous case and taking the parameters α to be Gaussian leads to the well known Kalman filter equations (Kalman and Bucy, 1961). Denoting the prior $P(s_t \mid \mathcal{Y}_1^{t-1}) = \mathcal{N}(s_t; s_{t|t-1}, \boldsymbol{P}_{t|t-1})$, the posterior $P(s_t \mid \mathcal{Y}_1^t) = \mathcal{N}(s_t; s_{t|t}, \boldsymbol{P}_{t|t})$ and the predictive distribution $P(\boldsymbol{y}_t \mid \mathcal{Y}_1^{t-1}) = \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{y}_{t|t-1}, \boldsymbol{W}_t)$, we get:

$$s_{t|t-1} = \boldsymbol{F} s_{t-1|t-1} \tag{3.39}$$

$$\boldsymbol{P}_{t|t-1} = \boldsymbol{F}\boldsymbol{P}_{t-1|t-1}\boldsymbol{F}' + \boldsymbol{Q} \tag{3.40}$$

$$\boldsymbol{y}_{t|t-1} = \boldsymbol{G} s_{t|t-1} \tag{3.41}$$

$$\boldsymbol{W}_t = \boldsymbol{G}\boldsymbol{P}_{t|t-1}\boldsymbol{G}' + \boldsymbol{R} \tag{3.42}$$

$$s_{t|t} = s_{t|t-1} + \boldsymbol{K}_t(\boldsymbol{y}_t - \boldsymbol{y}_{t|t-1}) \tag{3.43}$$

$$\boldsymbol{P}_{t|t} = (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{G})\boldsymbol{P}_{t|t-1} \tag{3.44}$$

starting with $s_{1|0} = \boldsymbol{\mu}$ and $\boldsymbol{P}_{t|0} = \boldsymbol{\Sigma}$. The matrix $\boldsymbol{K}_t$ is known as the *Kalman gain*, $\boldsymbol{K}_t = \boldsymbol{P}_{t|t-1}\boldsymbol{G}'\boldsymbol{Q}^{-1}$ and determines how much of the error is signal rather than noise. Equation (3.42) allows us to estimate the reliability of our predictions.

Note that, as the model converges to the optimal state, the prior will become more sharply peaked around this optimum state estimate. The covariance $\boldsymbol{P}_{t|t}$ will therefore become small, $\boldsymbol{W}_t$ will tend to $\boldsymbol{R}$, and the prediction error, $\boldsymbol{e}_t = \boldsymbol{y}_t - \boldsymbol{y}_{t|t-1}$, also known as the innovation, will converge toward the white noise $\boldsymbol{v}$.

In Equation (3.43) the first term contains the information gained from previous data, and can be thought of as representing the current model. The second term represents the contribution from the new data point. If $\boldsymbol{K}_t$ is small, then the new measurement will not alter the state vector significantly, implying that the current model is adequate. If $\boldsymbol{K}_t$ is large, then the new measurement will be making an important contribution, implying that the current model requires a significant adjustment. Problems arise when the Kalman gain is small, but the measurements contain important information. In this case, the filter is failing to adapt to this new information, and is said to be *diverging*. When using the Kalman filter, it is essential to check for divergence in order to determine the validity of the model (Anderson and Moore, 1979; Bar-Shalom and Fortmann, 1988).

Similarly, backward recursions can be obtained using the β parameters. Combining for-

ward and backward equations leads to equations known in the engineering community as the Rauch-Tung-Steibel smoother or the Kalman smoother (Rauch, 1963), estimating the mean and the covariance of the posterior distribution $P(s_t \mid \mathcal{Y}_1^T) = \mathcal{N}(s_t \mid s_{t|T}, \boldsymbol{P}_{t|T})$.

$$s_{t-1|T} = s_{t-1|t-1} + \boldsymbol{J}_{t-1}(s_{t|T} - \boldsymbol{F}s_{t-1|t-1}), \tag{3.45}$$

$$\boldsymbol{P}_{t-1|T} = \boldsymbol{P}_{t-1|t-1} + \boldsymbol{J}_{t-1}(\boldsymbol{P}_{t|T} - \boldsymbol{P}_{t|t-1})\boldsymbol{J}_{t-1}', \tag{3.46}$$

where $\boldsymbol{J}_{t-1} = \boldsymbol{P}_{t-1|t-1}\boldsymbol{F}'(\boldsymbol{P}_{t|t-1})^{-1}$ is a matrix similar to the Kalman gain.

Predictions

One-step ahead predictions are given by Equation (3.41) and Equation (3.42):

$$P(\boldsymbol{y}_{T+1} \mid \mathcal{Y}_1^T) = \mathcal{N}(\boldsymbol{G}s_{T+1|T}, \boldsymbol{G}\boldsymbol{P}_{T+1|T}\boldsymbol{G}' + \boldsymbol{R}). \tag{3.47}$$

Forecasting several steps ahead requires the prior information to be projected into the future through repeated applications of the system equations. Given the prior at time $t+1$, the prior at time $t+2$ is indeed given by the system equation. Linearity ensures again that the prior will be normal:

$$s_{T+N|T} = \boldsymbol{F}^{N-1}s_{T+1|T}, \tag{3.48}$$

$$\boldsymbol{P}_{T+N|T} = \boldsymbol{F}^{N-1}\boldsymbol{P}_{T+1|T}\boldsymbol{F}^{N-1\prime} + \sum_{i=T+1}^{T+N-1} \boldsymbol{F}^{N+t-i}\boldsymbol{Q}\boldsymbol{F}^{N+t-1\prime}, \tag{3.49}$$

where $s_{T+1|T}$ is the one step ahead prediction and is given by Equation (3.39). $s_{T+N|T}$ is the N step ahead estimate. Note however that if each eigenvalue of $\boldsymbol{F}$ is less than 1, $s_{T+N|T} \to 0$ as $N \to \infty$, irrespective to $s_{T+1|T}$. The output predictive distribution is easily obtained by:

$$P(\boldsymbol{y}_{T+N|T} \mid \mathcal{Y}_1^T) = \mathcal{N}(\boldsymbol{G}s_{T+N|T}, \boldsymbol{G}\boldsymbol{P}_{T+N|T}\boldsymbol{G}' + \boldsymbol{R}). \tag{3.50}$$

3.5 Parameter estimation

Given a sequence of observations $\mathcal{Y}_1^T$, the goal of *parameter estimation* or *learning*[6] is to estimate the parameters $\boldsymbol{w}$ of a model. We concentrate here on a maximum likelihood (ML) approach and show how both HMMs and SSMs can be trained in the same framework.

Direct maximum likelihood approaches like the Newton-Raphson technique or the method of scoring involve the use of non-linear optimisation techniques. These numerical techniques

[6] In the engineering community, this problem is known as system identification.

make use of the derivatives of the log-likelihood and estimate iteratively the parameter's vector w according to the equation:

$$w_{k+1} = w_k - \eta_k {M_k}^{-1} \frac{\partial \mathcal{L}(w_k)}{\partial w}, \tag{3.51}$$

where $M_k = \frac{\partial^2 \mathcal{L}(w_k)}{\partial^2 w}$ for the Newton-Raphson technique and $M_k = I(w_k)$ for the method of scoring. $I(w_k) = \mathrm{E}\left[\frac{\partial \mathcal{L}(w_k)}{\partial w} \frac{\partial \mathcal{L}(w_k)}{\partial w}'\right]$ is the Fisher information matrix and η_k is a scalar step size chosen in order to ensure an increase of the likelihood at each step.

For example, for a state space model, the (predictive) log-likelihood can be obtained by considering the predictive distributions $P(\boldsymbol{y}_t \mid \mathcal{Y}_1^{t-1})$ from time step 1 to T:

$$\mathcal{L}(\boldsymbol{w}) = -\frac{1}{2}\sum_{t=1}^{T}(\boldsymbol{y}_t - \boldsymbol{G}\boldsymbol{s}_{t|t-1})'{\boldsymbol{W}_t}^{-1}(\boldsymbol{y}_t - \boldsymbol{G}\boldsymbol{s}_{t|t-1}) - \frac{T}{2}\log(2\pi) - \frac{1}{2}\sum_{t=1}^{T}\log|\boldsymbol{W}_t|, \tag{3.52}$$

where $\boldsymbol{s}_{t|t-1}$ and $\boldsymbol{W}_t = \boldsymbol{G}\boldsymbol{P}_{t|t-1}\boldsymbol{G}' + \boldsymbol{R}$ are given by the Kalman filter equations. By taking the derivatives of Equation (3.52) with respect to the parameter's vector $\boldsymbol{w} = \{\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$, it is possible to derive an iterative algorithm which maximises the likelihood.

These techniques have been shown to work for several cases (Goodrich and Caines, 1979). Gupta and Mehra (1974) review also these techniques for linear dynamical systems. The authors note however that the Newton-Raphson method should not be used for several reasons. Firstly, this technique fails to converge whenever $\boldsymbol{M}_k$ has negative eigenvalues. Secondly, there might be numerical problems inverting $\boldsymbol{M}_k$ when this matrix is nearly singular and finally, the computation is very expensive especially if the number of parameters is significant. They propose methods for improving the method of scoring though they note that this technique can also run into several problems: for example, the step size may not lead to an increase of the likelihood. A monotonic increase of the likelihood is indeed not guaranteed.

The Expectation-Maximisation (EM) algorithm (Baum *et al.*, 1970; Dempster *et al.*, 1977) ensures, however, at each step an increase of the likelihood and convergence to a local maximum is always guaranteed. The algorithm is simpler to implement than second order methods and is often used when dealing with *missing* variables. Both HMMs and SSMs can be trained in this framework because the hidden state variables can be treated as missing variables. By integrating out all the hidden variables, the log-likelihood for a Markov model can be written as:

$$\mathcal{L}(\boldsymbol{w}) = \log \int P(\mathcal{S}_1^T, \mathcal{Y}_1^T \mid \boldsymbol{w})\, d\boldsymbol{s}_1 \ldots d\boldsymbol{s}_T. \tag{3.53}$$

If we introduce now a distribution Q over the hidden variables, we obtain a lower bound:

$$\begin{aligned}
\log \int P(\mathcal{S}_1^T, \mathcal{Y}_1^T \mid \boldsymbol{w})\, d\mathcal{S}_1^T &= \log \int Q(\mathcal{S}_1^T) \frac{P(\mathcal{S}_1^T, \mathcal{Y}_1^T \mid \boldsymbol{w})}{Q(\mathcal{S}_1^T)}\, d\mathcal{S}_1^T \\
&\geq \int Q(\mathcal{S}_1^T) \log \frac{P(\mathcal{S}_1^T, \mathcal{Y}_1^T, \mid \boldsymbol{w})}{Q(\mathcal{S}_1^T)}\, d\mathcal{S}_1^T \\
&= \langle \log P(\mathcal{S}_1^T, \mathcal{Y}_1^T \mid \boldsymbol{w}) \rangle_Q - \langle \log Q(\mathcal{S}_1^T) \rangle_Q \qquad (3.54) \\
&\equiv \mathcal{F}(Q, \boldsymbol{w}) \qquad (3.55)
\end{aligned}$$

where we have used Jensen's inequality[7]. The EM algorithm alternates between maximising the lower bound $\mathcal{F}(Q, \boldsymbol{w})$ with respect to Q and $\boldsymbol{w}$, respectively, holding the other fixed.

$$\begin{aligned}
\textbf{E-step} \qquad Q_{k+1} &= \arg\max_Q \mathcal{F}(Q, \boldsymbol{w}_k) \qquad (3.56) \\
\textbf{M-step} \qquad \boldsymbol{w}_{k+1} &= \arg\max_{\boldsymbol{w}} \mathcal{F}(Q_{k+1}, \boldsymbol{w}) \qquad (3.57)
\end{aligned}$$

As the E-step does not change $\boldsymbol{w}$, the likelihood after each combined EM step does not decrease. The maximum in the M-step is obtained by maximising the first term in Equation (3.54) since the entropy term $-\langle \log Q(\mathcal{S}_1^T) \rangle_Q$ does not depend on $\boldsymbol{w}$. It can be easily shown that for arbitrary Q the maximum in the E-step occurs when $Q(\mathcal{S}_1^T) = P(\mathcal{S}_1^T \mid \mathcal{Y}_1^T, \boldsymbol{w})$, at which point the bound becomes an equality. The M-step holds this distribution fixed and re-estimates the parameters $\boldsymbol{w}$ to maximise $\mathcal{F}(Q, \boldsymbol{w})$. This process is iterated until convergence to a local maximum is reached. Whereas the direct maximum likelihood techniques perform the optimisation in the marginal parameter space, the EM separates the optimisation in two spaces: the hidden variables and the model parameters spaces.

It is also worth noting that the algorithm can be partially implemented: in the generalised EM algorithm, the new estimate $\boldsymbol{w}$ increases the likelihood but does not necessarily maximises it. This is particularly useful when the M-step involves non-linear equations. In that case, we must resort to gradient based techniques. We will use this version for training HMMs in a maximum mutual information approach (Chapter 4).

For Markov models, we see that the E-step is nothing else than the inference problem where we have to estimate $P(s_t \mid \mathcal{Y}_1^T, \boldsymbol{w})$. An efficient algorithm for computing this has been described in section 3.4 for both HMMs and SSMs. The M-step is straightforward as it simply involves linear equations. Appendix A solves these equations and gives the re-estimation formulae for HMMs and SSMs.

The EM algorithm has found applications in several and different real-world problems: its version for HMMs, the *Baum-Welch* algorithm (Baum *et al.*, 1970) is the most commonly

[7] We use the notation $\langle f \rangle_P$ to denote the expectation of the function f under the distribution P, i.e. $\langle f \rangle_P = \int f(x) P(x) dx$.

used algorithm in speech recognition. Vermaak *et al.* (1998) compare the EM algorithm and numerical techniques for speech enhancement. Shumway and Stoffer (1982), Pole *et al.* (1994) suggest also the use of the EM algorithm for parameter estimation in state space modelling. Because of its efficiency and simplicity, the algorithm is also used in other fields like mixture models (Titterington *et al.*, 1985).

However, the EM algorithm is highly sensitive to initial values of the parameters. This is a common problem for mixture models because of the existence of a large number of local minima. It is therefore important to develop good initialisation techniques in order to obtain fast convergence to a good optimum. In Chapter 4 and Chapter 6, we will present such procedures. Unfortunately, another disadvantage of the EM is its slow convergence in the latter stages of training: it is often the case to notice a plateau in the likelihood after 20 iterations of the algorithm. Shumway and Stoffer (1982) suggest to switch to another algorithm like the second order methods at this stage.

3.6 Examples

In this section, we give two simple examples of how hidden state models can be used for analysing and forecasting time series.

3.6.1 Physiological data

In order to illustrate an important application of HMMs, namely the segmentation of a time series, we consider physiological data recorded from a patient who was tentatively diagnosed as suffering from sleep apnoea, which is a condition in which the subject temporarily stops breathing during sleep. This data has been taken from the repository of time series data sets of the Santa Fe Time Series Analysis and Prediction Competition (Weigend and Gershenfeld, 1993). The whole data set is actually a multivariate time series containing several physiological variables such as heart rate and blood oxygen saturation. We focus here on the univariate time series of the respiration of the patient. More details about this data are given in (Rigney *et al.*, 1994).

The dataset consists of 1000 data points[8]. Assuming that periods of apnoea are characterised by a relatively small activity of the chest, we train a simple 2-state HMM. The output probability density $P(y_t \mid s_t)$ is a single Gaussian, although this assumption may not be ap-

[8] As in (Ghahramani and Hinton, 1998), we used samples 6201-7200 for training. The dataset can be found in `http://www.cs.colorada.edu/~andreas/Time-Series/SantaFe.html#SetB`.

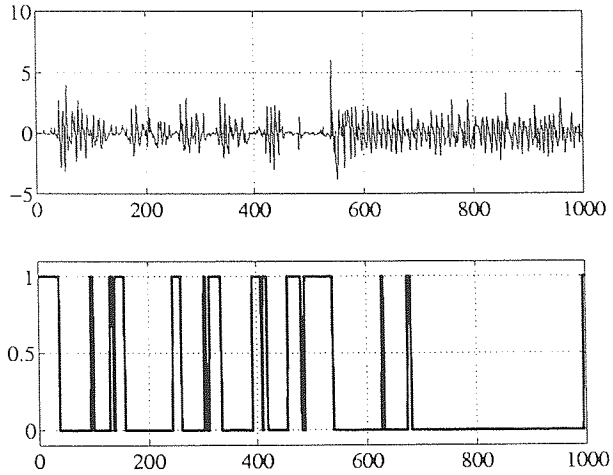


Figure 3.2: Segmentation of the chest volume data with the Viterbi algorithm. The sections of low activity, corresponding to periods of apnoea, are easily identifiable.

propriate for such physiological data. We used this data as a simple segmentation problem only, being aware that other models, such as autoregressive HMMS may fit the time series more accurately.

In no more than 10 iterations the EM algorithm converges. Figure 3.2 plots the chest volume and the segmentation obtained by applying the Viterbi algorithm. Although this example is quite simple, it shows how hidden Markov models can be applied for identifying underlying regimes. In many problems, it is indeed quite interesting to develop techniques that allow us to segment a time series. A simple predictive model, often linear, can then be fitted on each segment.

3.6.2 Sunspot data

Figure 3.3 plots the annual average[9] of sunspots from 1700 to 1920. The time series shows a strong seasonality with a time varying amplitude and has served as a benchmark in the statistical literature (Tong, 1995) namely because the underlying generator is not exactly known. The average time between maxima is 11 years and to date no principled theory exists for describing this behaviour, although it is known that sunspots are related to other activity of the sun. Our goal here is not to present a model that outperforms other techniques. We want to show how a simple linear dynamical system can be used and emphasize the dimension of the hidden state.

The training set contains 200 points from 1700 to 1920. We use the data from 1921

[9]The data are daily, monthly and annually reported by the Royal Observatory of Belgium and can be found at `http://www.oma.be/KSB-ORB/SIDC/sidc_txt.html`.

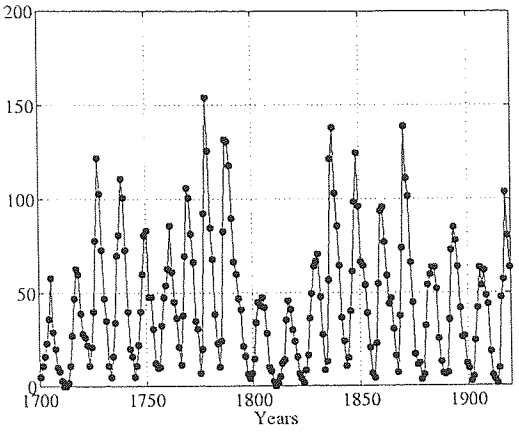


Figure 3.3: Annual average of sunspots from 1700 to 1920.

to 1998 for testing the models. In order to illustrate the importance of the dimension of the hidden state, we trained two linear dynamical systems with the EM algorithm: the difference between these two models is simply the dimension of the hidden state space. The one-dimensional SSM(1) ($s_t \in \mathbb{R}$) is not capable of predicting the data: the one-step ahead predictions are a delayed version of the target. On the contrary, the second model, SSM(2), which contains a two-dimensional hidden state, performs quite well (Figure 3.4). This is confirmed by comparing the root mean squared errors[10] for these two models: $RMSE(1) = 0.362$ and $RMSE(2) = 0.155$. Furthermore, it is interesting to note that higher dimensional hidden state space models do not outperform the results obtained with SSM(2), suggesting that the state space may lie in $\mathbb{R}^2$.

3.7 Discussion

In this chapter, we have reviewed two powerful tools for modelling time series in a probabilistic framework. We have shown how first-order Markov models can be viewed in the same framework. Because of the Markov property, the inference problem is exactly the same for both hidden state models and we derived recursive equations that enable us to estimate at each time the hidden variables.

We also reviewed the learning problem in a maximum likelihood framework and described the Expectation-Maximisation algorithm. As mentioned previously, the main disadvantage of the EM is its slow convergence. It is indeed important to have good starting values for the

[10] For assessing the performance of a model on a test set, it is common to define the root mean squared error: $RMS = \frac{\sum_{t=1}^{T} ||\hat{y}_t - y_t||^2}{\sum_{t=1}^{T} ||y_t - \bar{y}_t||^2}$ where $\hat{y}_t$ is the prediction of the model and $\bar{y}_t$ is the mean of the target. This value does not grow with the size of the test dataset and has a value of unity when the model is predicting the mean of the test data.

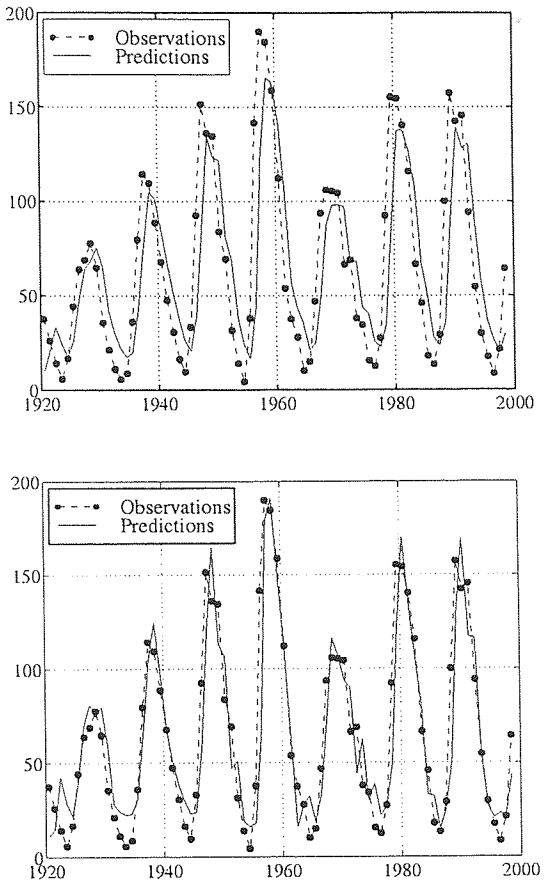


Figure 3.4: Predictions of state space models on the test set: the one-dimensional SSM(1) (top) does not manage to capture the dynamics of the time series and produces a prediction which is delayed with respect to target. The second SSM(2) (bottom) performs better.

parameters as it increases both the speed of convergence and the probability of converging to a good local optimum. We believe that this initialisation is crucial for any real-world problem. In the next chapters, we present initialisation techniques that alleviate this problem.

A Bayesian treatment of Markov models represents a challenging and promising field of research. Recently, MacKay (1998) suggested using techniques known as ensemble learning (Hinton and Van Camp, 1993) for discrete HMMs (discrete output probabilities). Whereas the maximum likelihood approach optimises a point estimate in the parameter space, the ensemble learning approach considers an *ensemble* in order to approximate the posterior probability of the parameters. At this stage no results have been presented. The extension to continuous HMMs is also promising, especially when dealing with a small amount of observations, where the ML approach may be susceptible to over-fitting. An important application of the Bayesian approach for HMMs would be to identify the relative importance of the hidden states. This could be carried out using techniques similar to *automatic relevance determination* (Neal, 1996; MacKay, 1995). For a real-world problem, it is often particularly

difficult to know in advance the number of hidden states the model should contain.

The same approach of ensemble learning can be applied in the field of linear dynamical systems. A full Bayesian treatment of state space models is impractical but approximations like the ensemble learning may give a better representation of the posterior distribution than its mode.

In our work, we use HMMs and SSMs for two important issues. The first one concerns *segmentation*: for example, the oil well drilling process is essentially characterised by different *regimes*: the properties of the process are usually held relatively steady, except for minor fluctuations, for a certain period of time, and then, at certain instances, change to another set of properties. Because the dynamics of the process are different from one regime to another, we show in Chapter 4 how to use a hidden Markov model for modelling this process. The second application concerns *prediction*. Because the Kalman Filter enables us to predict not only the mean of the time series but also to provide confidence intervals for unseen data points, we show in Chapter 6 how to combine HMMs and SSMs in models that we call dynamical local models.

There are several extensions of HMMs and SSMs. In Chapter 6, we will review in detail several hybrid models that have been proposed in different fields like econometrics, engineering and machine learning.

Chapter 4

Time delay estimation with hidden Markov models

4.1 Introduction

A key part of multivariate time series analysis is identifying the lags or delays between different variables. This differs from characterising the order or degree of freedom of a single time series, where the goal is to estimate the intrinsic dimensionality of the data in order to model the deterministic component of the data generator (Broomhead and King, 1986). In the latter case, the correlation of past values usually tails off gradually, so that the most recent samples have the largest impact on the current value. This is not the case where two time series X_t and Y_t are related by a lag λ. There will be no relationship between X_{t-l} and Y_t for $l < \lambda$.

Under the assumption of stationarity, cross-correlation is a powerful tool for measuring and modelling linear relationships between variables (Kendall and Ord, 1990). The cross-correlation can then be used in linear model identification procedures; it is often used as the basis for identifying the order of non-linear models, as it is fast to compute. However, in many real-world applications the assumptions of linear dependencies and stationarity are not valid.

In this chapter, we consider the problem of modelling processes which manifest a sequentially changing behaviour: the parameters of the data generator usually remain constant, except for minor fluctuations, and then, at certain times, change to another set of values.

Our approach is based on using hidden Markov models to model the distribution of the time series. More precisely, given two time series X_t and Y_t, related by a lag λ and generated

from a non-stationary underlying process which exhibits different regimes, we show how HMMs can be used to estimate the value of λ. An HMM is essentially a mixture model, in which information about the past is conveyed through a single discrete variable, the hidden state. In certain circumstances, this state can be viewed as a switching variable between different process regimes.

To estimate parameters for an HMM within a maximum likelihood framework, one can use the *Baum-Welch* algorithm (Baum *et al.*, 1970), which is the relevant version of the EM algorithm (Dempster *et al.*, 1977) (see Chapter 3). In section 4.2, we develop a novel procedure for training HMMs to maximise the mutual information (MMI) between two time series X_t and Y_t and compare it with maximum likelihood (ML) estimation. The Baum-Welch algorithm is a hill-climbing algorithm which does not require the cost function gradient. Unfortunately, no such method is known for MMI estimation and we must therefore resort to the use of traditional maximisation techniques that do use the cost function gradient.

We apply this approach to the analysis of the oil well drilling process. The process exhibits both complex time relationships between variables and highly non-stationary behaviour. A fluid called 'mud' carries the drilling cuttings up the hole to the surface. The time it takes for the cuttings to come up to the surface is called the *lag for return* and is a crucial parameter for modelling and understanding the process. This time-varying parameter depends not only on the depth of the hole and the pressure of the drilling fluid, but also on the geology of the surrounding rock formation and the drilling mode. In section 4.3 we analyse drilling data and compare our results with cross-correlations and the numerical models based on fluid mechanics which are currently used operationally.

4.2 Delay estimation

Consider the following problem: a sequence of observations $\mathcal{Z}_1^T = [(x_1, y_1), \ldots, (x_T, y_T)]$ is being generated by an underlying system. Unfortunately, we do not see the true sequence $\mathcal{Z}_1^T$ but a modified version where one variable is delayed: $\mathcal{Z}_1^T(\lambda) = [(x_{1-\lambda}, y_1), \ldots, (x_{T-\lambda}, y_T)]$. Our task is to estimate the value of λ. Given a delayed two-dimensional time series vector $\boldsymbol{Z}_1^T(\lambda) = (X_{t-\lambda}, Y_t)_{t=1\ldots T}$, we say that Y_t leads X_t by an unknown lag λ. For convenience and clarity, we define $X^l \equiv X_1^T(l) = [X_{1-l}, \ldots, X_{T-l}]$ to be the time series X_t delayed by l steps, omitting time indexes and $Z^l \equiv (X^l, Y)$.

Let S be a discrete random variable taking values in a set $\{q_1, \ldots, q_N\}$ and assume that the system at any time is in one and only one of the N states, such that the observation z_t

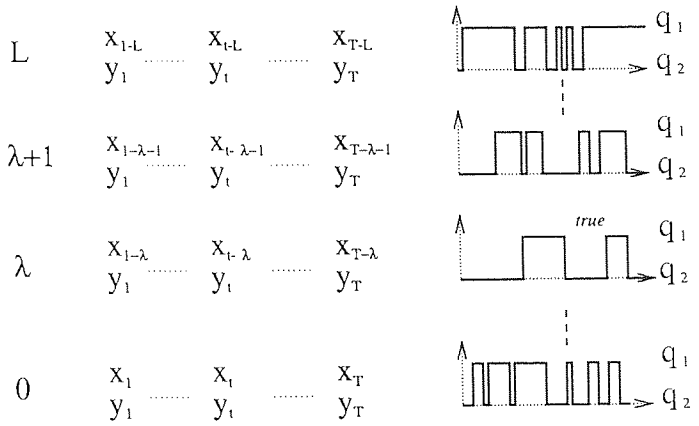


Figure 4.1: The synchronisation problem: we assume an underlying *true* state sequence S_{true}. This sequence is not observable but can be recovered by identifying the corresponding observation sequence $\mathcal{Z}^{\lambda}$, i.e. when the observations x_t and y_t are properly synchronised.

is a measurement from the system but the underlying states are not observable. Under this assumption, the problem of delay estimation can be viewed as a *synchronisation* problem, where the goal is to recover the *correct* sequence of hidden states. Figure 4.1 shows the underlying sequence $S(l)$ corresponding to different delayed time series (in this example, the system can switch between two states q_1 and q_2). The value λ corresponds to a *true* sequence of hidden states and the task is to recover this sequence by identifying the corresponding observation sequence $\mathcal{Z}^{\lambda}$.

4.2.1 Maximum likelihood

As mentioned in Chapter 3, the usual procedure for training HMMs is to find the parameter values of $\boldsymbol{w}$ that maximise the likelihood or the log-likelihood of the observed sequence $\mathcal{Z}_1^T$:

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \log P(\mathcal{Z}_1^T \mid \boldsymbol{w}). \tag{4.1}$$

The standard approach for doing this is to use the Baum-Welch algorithm, which is the relevant version of the EM algorithm (Baum *et al.*, 1970) (see Appendix A).

For a multivariate time series $\boldsymbol{Z} = (X, Y)$, the ML estimation procedure will maximise the joint distribution $P(X, Y \mid \boldsymbol{w})$. Using this approach, training an HMM denoted by $\boldsymbol{w}^l$ will allow us to obtain the most likely model parameters that describe the delayed sequence $\mathcal{Z}^l$. Assuming the existence of a *true* value $l = \lambda$ that relates $x_{t-\lambda}$ and y_t, we propose the following delay estimation procedure based on maximum likelihood. First, an HMM $\boldsymbol{w}^l$ is trained with the delayed sequence z^l and the likelihood of each model $\boldsymbol{w}^l$ is estimated. The

lag is then obtained by finding the most likely model, *i.e.*

$$\lambda = \arg\max_{l} \log \mathcal{L}^l, \tag{4.2}$$

where $\mathcal{L}^l = P(\mathcal{Z}^l \,|\, \boldsymbol{w})$ denotes the likelihood of the delayed sequence $\mathcal{Z}^l$ given the model $\boldsymbol{w}^l$. The approach is motivated by the fact that the sequence $\mathcal{Z}^l$ corresponds to a specific sequence of hidden states representing the dynamics of the process (Figure 4.1). Intuitively, we expect that $\mathcal{L}^l$ will always be less than $\mathcal{L}^\lambda$ $(l \neq \lambda)$. Indeed, assuming that for each time step t, $x_{t-\lambda}$ and y_t have been generated by a specific state q_t^*, the system will not be able to enter that *true* state if x_t and y_t are not properly synchronised.

4.2.2 Maximum mutual information

There are many very important properties of the maximum likelihood estimate but most of them stem from an implicit assumption of model correctness. The justifications for using ML to estimate the mean and the variance of a Gaussian distribution, for example, assume that the sample has indeed been generated by a Gaussian distribution. If, however, we do not know the 'correct' model which has generated the data and if there is no reason to believe that the sample has been generated from any particular model, then we can ask ourselves whether the use of ML is appropriate.

In previous work (Brown, 1987), the maximum mutual information (MMI) criterion for discriminative training of multiple HMMs has been introduced in order to alleviate problems that may occur when several HMMs are to be designed at the same time. This leads to an algorithm where the goal is to choose a correct model m amongst a set of M models that maximises the following expression:

$$I(\boldsymbol{Z}, \boldsymbol{w}_m) = \left[\log \frac{P(\mathcal{Z}_1^T | \boldsymbol{w}_m)}{\sum_{n=1}^{M} P(\mathcal{Z}_1^T | \boldsymbol{w}_n)} \right]. \tag{4.3}$$

In contrast, the motivation of our approach is not to maximise the mutual information between the observation sequence and a *complete* set of models $\boldsymbol{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M)$; instead we are trying to estimate the parameters of a single HMM that maximises the mutual information between two random variables.

In order to measure the amount of information with respect to Y that we may expect to obtain by observing X^l, it is useful to introduce the concept of mutual information $I(X^l, Y \,|\, \boldsymbol{w})$ between X^l and Y:

$$I(X^l, Y) = \mathcal{H}(X^l) + \mathcal{H}(Y) - \mathcal{H}(X^l, Y), \tag{4.4}$$

where $\mathcal{H}(X) = -\langle \log P(X) \rangle_{P(X)}$ is the entropy of the random variable X. As the joint probability $P(X^l, Y)$ can be rewritten as $P(X^l, Y) = P(Y \mid X^l)P(X^l)$, we have

$$I(X^l, Y) = \mathcal{H}(Y) - \mathcal{H}(Y \mid X^l). \tag{4.5}$$

In our problem, the goal is to maximise the information with respect to Y we may get by observing a delayed time series X^l. We notice however that the first term of Equation (4.5) does not depend on l and can be discarded in the optimisation procedure. Indeed, for two different values of l, *i.e.* for two different time series X^{l_1} and X^{l_2}, the entropy of Y does not affect the change in the mutual information $I(X^{l_1}, Y) - I(X^{l_2}, Y)$. Thus, maximising the conditional distribution $P(Y \mid X^l)$ is equivalent to maximising the mutual information between Y and X^l.

$$\arg\max_l I(X^l, Y) = \arg\max_l \log P(Y \mid X^l). \tag{4.6}$$

Comparing Equation (4.6) to Equation (4.2), we see that ML and MMI estimations differ in the objective function. In ML, we are interested in estimating parameters that maximise the joint probability. The MMI approach leads to maximising the conditional probability.

In terms of previous work, our approach resembles that of (Viola and Wells, 1995) who used mutual information for image matching. It is also interesting to point out the link between *supervised* learning and *discriminant* learning. One of the major drawbacks of HMMs is their poor discriminant power due to unsupervised learning. That is why the MMI approach has been introduced for improving discrimination (Bahl *et al.*, 1983; Brown, 1987). Bridle (1989) showed that supervised and discriminant learning are strongly related. In our work, we suggest to model the *output* time series Y as a function of the delayed *input* time series X^l instead of modelling the joint distribution. That leads to a discriminant supervised learning algorithm. This approach is very similar to the Input-Output HMM of (Bengio and Frasconi, 1995), although in that work the hidden state is also a function of the input time series X.

The Baum-Welch algorithm is a hill-climbing algorithm for ML estimation which does not require the model gradient. Unfortunately, the exact M-step for MMI estimation is not possible. Suppose that the sample vector $\boldsymbol{z}_t$ is composed of two multi-dimensional vectors $\boldsymbol{z}_t = (\boldsymbol{x}_t, \boldsymbol{y}_t)$ with mean $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ and covariance matrix $\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$ then the conditional density $P(\boldsymbol{y}_t \mid \boldsymbol{x}_t)$ is Gaussian with mean $\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_t - \boldsymbol{\mu}_1)$ and covariance $\Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}$, *i.e.*

$$P(\boldsymbol{y}_t \mid \boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_t - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}). \tag{4.7}$$

For an HMM with Gaussian observation densities, the output density of each hidden state i is parameterised by a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$. For example, taking the derivatives of the expected log-likelihood $\langle \log P(\mathcal{S}_1^T, \mathcal{Y}_1^T \mid \mathcal{X}_1^T, \boldsymbol{w}) \rangle_Q$ with respect to the means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and setting them to zero leads to 'inter-related' equations involving both means. This implies that re-estimation formulae cannot be obtained for the parameters of the HMM.

In Chapter 3 however we saw that a generalised EM algorithm can be derived by using traditional gradient-based minimisation techniques: at each M-step, a new estimate $\boldsymbol{w}$ is obtained by taking the derivatives of the expected log-likelihood with respect to $\boldsymbol{w} = \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\Pi}\}$. These derivatives involve the forward-backward equations (E-step) and can be used either in a simple gradient descent algorithm or a nonlinear optimisation algorithm such as conjugate gradients (Press *et al.*, 1992), which uses the gradient of the objective function. Each forward and backward recursion requires on the order of N^2T calculations, where N is the number of hidden states and T is the length of the sequence. This can lead to a computationally expensive algorithm, especially if the HMM contains a large number of parameters. This is not a big issue for the problem we are interested in, as the size of the HMM we consider is relatively small (the models in our simulations do not contain more than 3 hidden states). It should also be noticed that contrary to the ML approach where the constraints of the model[1] are satisfied at each iteration, the implementation of the MMI approach requires a re-parameterisation of the model to ensure that the constraints are satisfied. Appendix B reviews the derivations and the technical points of the MMI procedure.

4.3 Initialisation

As mentioned in Chapter 3, the choice of initial conditions is crucial for the EM algorithm since the re-estimation equations give values of the HMM parameters which correspond to a local maximum of the likelihood. Good initial conditions will increase the speed of convergence. For this reason, it is important to develop good initialisation procedures. Rabiner *et al.* (1985) emphasized the strong sensitivity of the Baum-Welch algorithm to the initial model parameters. The authors clearly showed that the accuracy of the estimates depends on the initial conditions from which the iterative procedure starts. In particular, a good initialisation of the means of a mixture of Gaussians can lead to significant improvement in the convergence speed and enhances the quality of the local maximum found.

[1] At each iteration, we must ensure that the new estimates a_{ij} can be interpreted as transition probabilities. Another constraint concerns the output probabilities $b_i(\boldsymbol{y}_t)$: if we consider a Gaussian distribution for example, a symmetric positive definite covariance matrix is required at each step.

Depending on the application, prior knowledge will of course allow the investigator to specify initial estimates. For example, a knowledge about the average state duration is important since it permits us to initialise the transition matrix $\boldsymbol{A}$. Rabiner (1989) notes that either random or uniform initial estimates for Π and $\boldsymbol{A}$ parameters can lead to good results. The most important parameters are the output density probabilities $\boldsymbol{B}$, especially when dealing with a mixture of components. For this purpose, a *segmental K-means* algorithm has been derived in the speech recognition community in order to initialise hidden Markov models. This procedure is a variant of the well-known K-means algorithm. Starting from an initial estimate of the HMM parameters, the observation sequence is segmented into N states. This segmentation is obtained by finding the most likely state sequence given the current model and can be carried out with the Viterbi algorithm (Viterbi, 1967) (see Chapter 3). The parameters of each hidden state are then updated via a K-means clustering procedure. This results in a maximum likelihood estimate of the data that occur within each state of the model. This procedure is particular useful when dealing with high-dimensional data space and when the output density probabilities $\boldsymbol{B}$ are modelled by a mixture of Gaussians. Moreover, it is possible to obtain good initial estimates for the transition probabilities a_{ij} by simply enumerating the number of transitions from state q_i to state q_j.

In our simulations, however, we noticed that a simple K-means algorithm without performing a segmentation via the Viterbi algorithm is sufficient, because the size of our models is small and because each hidden state is associated with a single Gaussian distribution. This allows us to implement a fast initialisation procedure which leads to rapid convergence of the EM algorithm. We illustrate the efficiency of the procedure by training HMMs on synthetic data. We first generate two sequences of 1000 observations (training and test sequences) from an HMM containing $N = 3$ hidden states. The observation sequence is a two-dimensional time series and the parameters of the HMM are as follows:

$$\boldsymbol{A} = \begin{pmatrix} 0.95 & 0.03 & 0.02 \\ 0.02 & 0.95 & 0.03 \\ 0.03 & 0.02 & 0.95 \end{pmatrix} \quad \begin{array}{llll} \boldsymbol{\mu}_1 = & (0.5, 0.0) & \boldsymbol{\Sigma}_1 = & \mathrm{diag}(1.0, 1.0) \\ \boldsymbol{\mu}_2 = & (-1.0, 0.5) & \boldsymbol{\Sigma}_2 = & \mathrm{diag}(2.0, 0.5) \\ \boldsymbol{\mu}_3 = & (2.0, 2.0) & \boldsymbol{\Sigma}_3 = & \mathrm{diag}(0.5, 0.2) \end{array} \tag{4.8}$$

We trained 100 models with different initial conditions on the first observation sequence. Each model is then evaluated on the test set. Table 4.1 summarises the performance of the initialisation procedure and compares it to a simple random initialisation.

The convergence is faster on average when the model has been initialised with the K-means algorithm. Moreover, the standard deviation is significantly lower than the random

Technique	# iterations	$\mathcal{L}_{\text{train}}$	$\mathcal{L}_{\text{test}}$
random	29 ± 16	-2.715 ± 0.04	-2.752 ± 0.12
K-means	18 ± 3	-2.702 ± 0.01	-2.739 ± 0.04

Table 4.1: Performance of the K-means algorithm as an initialisation procedure for HMMs.

initialisation. This suggests that different initial estimates will lead to a similar model after a small number of iterations. On the contrary, we see that a model with a random initialisation may need several iterations of the EM before converging. The likelihood per datum on the test set of the true model is -2.722 and both techniques lead to models with good generalisation capabilities. Concerning the maximum likelihood estimates, we give here the mean of the transition matrix over the 100 models for both initialisation procedures:

$$A_{\text{rand}} = \begin{pmatrix} 0.92 & 0.04 & 0.04 \\ 0.05 & 0.91 & 0.04 \\ 0.04 & 0.04 & 0.92 \end{pmatrix} \quad A_{\text{K-means}} = \begin{pmatrix} 0.95 & 0.03 & 0.02 \\ 0.02 & 0.95 & 0.03 \\ 0.03 & 0.02 & 0.95 \end{pmatrix} \tag{4.9}$$

Compared to the true parameters, we see that the K-means procedure leads to better estimates.

4.4 Experimental results

In this section, we first present the results of the maximum likelihood approach on synthetic data in order to demonstrate the HMM approach and show that it is more general than the classic standard linear techniques for lag detection. We then introduce the drilling process and demonstrate the application of our methods for estimating a crucial parameter in the oil industry.

4.4.1 Synthetic Data

In order to illustrate our approach, we consider the following problem: we generated a synthetic two dimensional sequence of 1000 observations $\boldsymbol{z}^* = (x_t, y_t)_{t=1\ldots T}$ from a two state continuous HMM denoted by $\boldsymbol{w}^* = \{\boldsymbol{A}^*, \boldsymbol{B}^*, \boldsymbol{\Pi}^*\}$. In this case $\lambda = 0$. We then train two state continuous HMMs Θ^l with delayed sequences $\boldsymbol{z}^l = (x_{t-l}, y_t)_{t=1\ldots T}, -L \leq l \leq L$ using the ML approach. We choose a transition matrix $\boldsymbol{A}^*$ which allows balanced transitions from one state to another, *i.e.* $a^*_{ij} = a^*_{ji} (= 0.6)$. The output probability density associated with each state is a Gaussian with a diagonal covariance matrix and does not affect the simulations significantly.

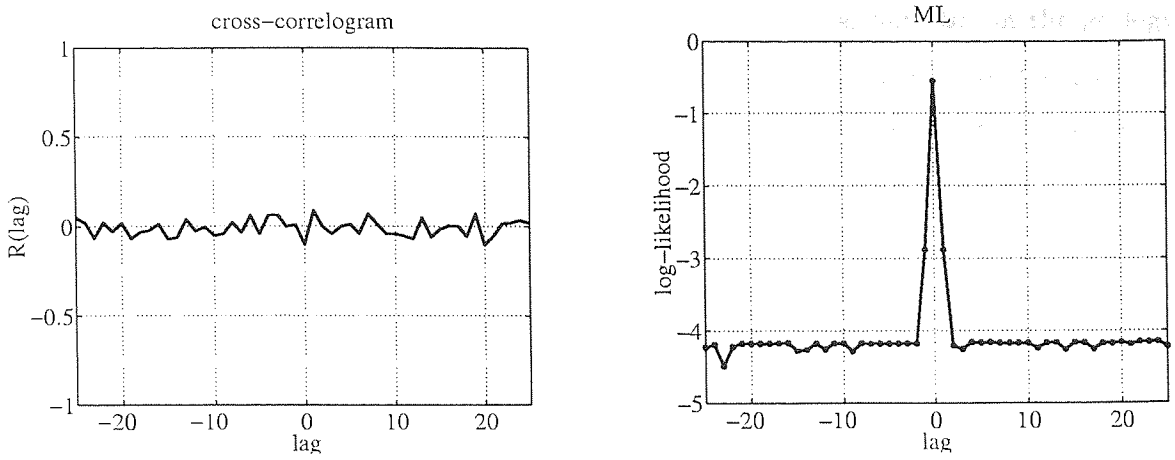


Figure 4.2: The left hand plot shows the cross-correlogram of the two time series generated by the 2 state continuous HMM Θ^*. The right hand plot shows the log-likelihood against the lag when training HMMs with ML approach. A significant peak appears for $l = 0$, which corresponds to the true lag $\lambda = 0$.

The results are shown in Figure 4.2. It can be seen that the cross-correlogram is not capable of detecting any relationship between the two time series, even though they are generated by the same HMM. This is simply because we chose a diagonal covariance matrix, which means that there is no linear relation between the two time series. On the other hand, plotting the log-likelihood of each model Θ^l against l shows a significant peak for $l = 0$, which corresponds to Θ^*, *i.e.* the true model which was used to generate the time series. The sharpness of the peak shows that we cannot use the shape of the likelihood curve to search for an optimal value of l, as the curve is practically flat for $|l| > 1$.

4.4.2 Drilling data

A significant difficulty during exploration drilling is ensuring the drilling debris is effectively removed from the bore; this is known as the 'hole cleaning' problem (Pilehvari *et al.*, 1996). At present, no equipment exists to act as a monitor of hole cleaning status. In the case of vertical wells, an adequate velocity in the mud circulation is generally sufficient to guarantee that most debris are brought to the surface. The problem is more complicated when drilling deviated wells since gravity settlement can occur. The gradual build up of low gravity solids (LGS) increases the torque required to turn the drill string. In extreme cases, the drill pipe may get stuck or even fracture off. Retrieving a stuck pipe is a difficult and expensive operation: if the pipe breaks and cannot be recovered, the well may even be abandoned.

The time it takes for the cuttings to come to the surface is called the *lag for return* and is a crucial parameter in early stuck pipe detection and modelling the drilling process.

Indeed this parameter depends not only on the depth of the hole, but also on the geology of the surrounding rock formation and the rheology[2] of the mud. The current algorithms used on rigs to compute the lag for return are physical models based on fluid mechanics but are believed to have an accuracy in the order of several minutes, mainly because of assumptions on the nature of the fluid and the flow. Fluids are divided into two general classes, namely the *Newtonian* and *non-Newtonian* fluids[3]. Generally speaking, there are two types of flow: laminar flow and turbulent flow[4]. During drilling operations, numerical models take into account non-Newtonian nature of drilling mud, but do assume that the flow is laminar, because of the poor understanding of downhole conditions.

Recently a new device, capable of detecting fine particulate solids in drilling fluids, has been developed by Thule Rigtech Ltd. (Thule Rigtech, 1995). A drilling engineer has gathered a large dataset of drilling variables and low gravity solid information from a rig in Holland. As all the data are collected on the surface, if λ represents the lag for return, then Y_t, which is the quantity of low gravity solids (LGS) measured at time t, is effectively the amount of solids that has been generated by the bit at time $t - \lambda$. Thus, assuming that Y_t is related to other drilling parameters $X_{t-\lambda}$, we propose to compute the lag of return by using hidden Markov models and the procedures described in Section 4.2.

Our motivation is essentially based on the fact that the drilling process is actually a process that manifests a sequentially changing behaviour: the properties of the process are usually held steady, except for minor fluctuations, for a certain period of time, and then, at certain instances, change to another set of properties (caused by action of the drilling engineers anticipating a problem, change of geology, etc). The opportunity for more efficient modelling can be exploited if we can first identify these periods of rather steady behaviour, and then are willing to assume that the temporal variations within each of these steady periods are stochastic. A more efficient representation may then be obtained by using a common short time model for each of the steady, or well-behaved parts of the model, along with some characterisation of how one such period evolves to the next.

Even if the process cannot be considered as stationary, there are strong reasons to believe that λ remains relatively constant over a 2 hour time scale. Typically, X_t represents one relevant drilling parameter (although we have also considered models with more than one

[2]The term rheology defines the chemical properties of the mud: the most important rheological properties of mud are its plastic viscosity, its yield point and its gel strength (Rabia, 1985).

[3]A Newtonian Fluid is defined by a constant viscosity, which is only influenced by changes in temperature and pressure.

[4]In turbulent flow, the flow pattern is random in both time and place. The chaotic and disordered motion of fluid particles results in two components of velocity: a longitudinal and a transverse component.

parameter): for instance, the pressure of the circulating fluid inside the pipe (*STPP*) or the torque of the pipe (*Torque*). The total force applied on the drilling system in order to hold the drill pipe in the rig (*Hook Load*) and the rate of progress (*ROP*) are other important parameters.

Normal drilling conditions

This data set represents the 'normal' drilling conditions as no special event was identified by the drilling engineers. The data set contains 450 data points and represents a period of 4 hours of drilling (each time step corresponding to $30s$). Figure 4.3 plots the original *LGS* and *Hook Load* time series. It is very difficult to estimate the value of λ by a simple visual inspection of the time series and the numerical models suggest a value of 31 min for the lag for return.

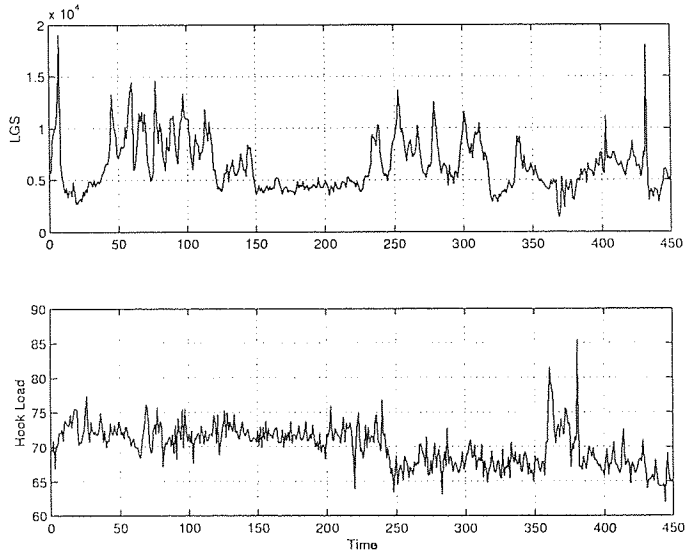


Figure 4.3: *LGS* and *Hook Load* time series for normal drilling conditions.

Figure 4.4(top) plots the cross-correlogram between *LGS* and two important drilling parameters, namely *STPP* and *Hook Load*. No correlation significantly different from zero can be detected.

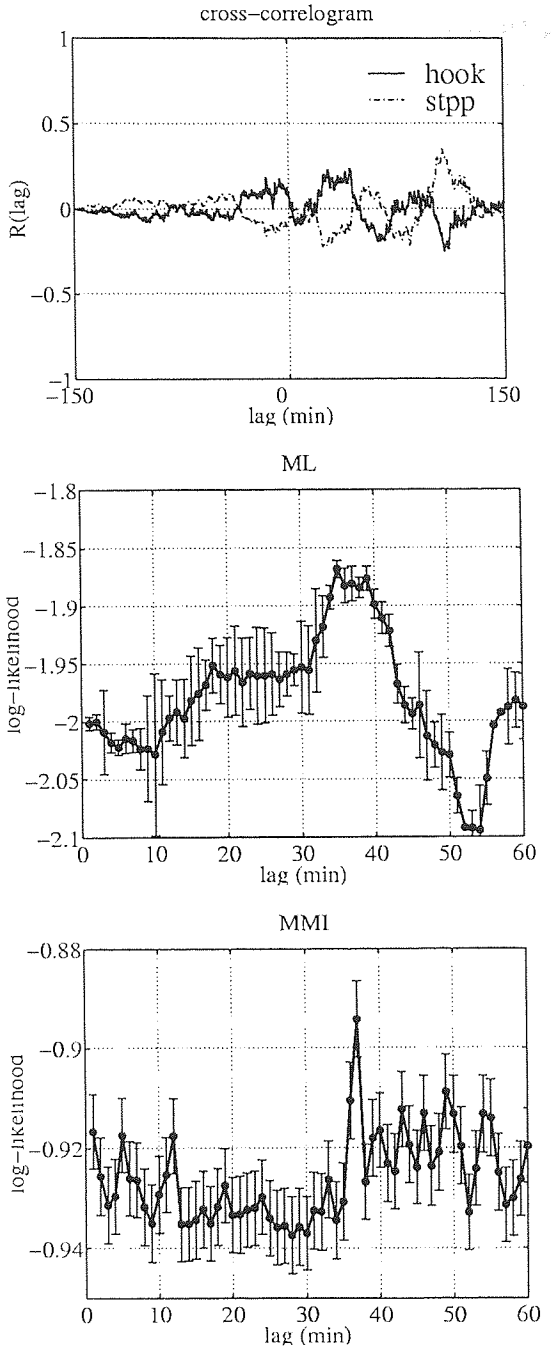


Figure 4.4: Cross-correlogram (top) and log-likelihood of the observation sequences $\mathcal{Z}^l$ given the model HMM^l for ML (middle) and MMI (bottom) approaches.

For each value of l, 100 HMMs with different initial parameters have been trained using ML and MMI approaches. A K-means procedure was used to initialise each model. Figure 4.4(middle) and Figure 4.4(bottom) plot the mean and the two standard deviation error bars. The ML approach suggests a value between 36 and 40 min whereas the MMI approach is more confident and suggests a sharper peak at 37 min, which is statistically significant.

The results reported here have been obtained by training 3 state HMMs using *Hook Load*

parameter for the time series X_t. Simulations with other drilling parameters did not give significant results. As we shall see later, depending on the drilling conditions, selecting the correct variable X_t is a key problem with this approach. However, when using other data sets corresponding to normal drilling conditions, we have found that *Hook Load* always gives good results.

It is also important to note that, for our problem, there is no automatic technique to specify in advance the number of states the model should contain. The major drawback of HMMs is the significance of hidden states and without any prior knowledge, the investigator needs to run several simulations with different topologies. In our application, we could speculate that each hidden state is associated to a specific formation or rock structure but this knowledge can hardly be used in advance.

Thus, other simulations with different numbers of states have been carried out in order to assess the results obtained with an HMM containing 3 states. The same training procedure was applied to models containing from 1 to 10 hidden states. For example, Figure 4.5 plots the results obtained by the ML approach with HMMs containing 2 and 4 states. It can be seen that the peak is less pronounced for an HMM containing 2 states[5] (Figure 4.5a). By increasing the number of hidden states, we have noticed that the global shape of the plot remained similar to the one obtained with 3 states. This can be seen in Figure 4.5b which plots the results obtained with 4 states: the figure is very similar to Figure 4.4(middle) and also suggests a peak around 35 min. It is however important to note that higher complexity in the model increases the number of local minima and this is reflected in the error bars. The same conclusions were drawn from models containing a greater number of states.

[5]The same remark applies to an HMM containing only one state, namely a simple Gaussian distribution for which, no lag could be detected.

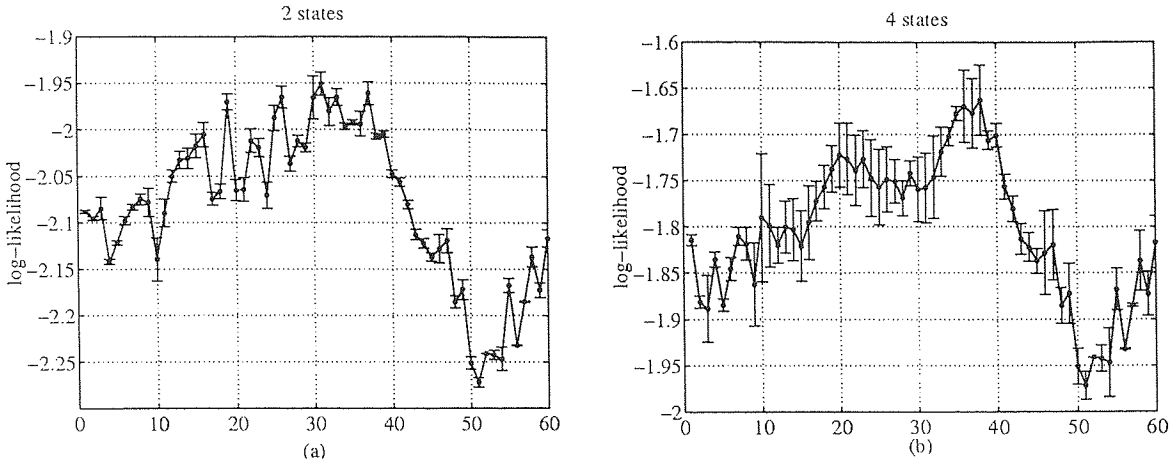


Figure 4.5: ML results obtained with 2 and 4 state HMMs on normal drilling conditions time series.

Formation change

When exploring different types of rock structure, it is sometimes possible to estimate the lag for return by visually monitoring relevant time series where a transition occurs between two geological formations.

Figure 4.6 plots the rate of progress and the quantity of low gravity solids for a specific event. At 08h04 the formation changed from soft to hard rock, which can be easily seen in the first plot where the rate of progress decreases suddenly.

The second figure plots the quantity of solids and shows a significant regime transition at 08h27: the amount of particles decreases as well. Note the noisy signal for LGS. For this day, the numerical models based on fluid mechanics suggested a value of λ of 43 min, which seems to be wrong as visual inspection of the graphs gives a value of 23 min.

Figure 4.7 shows our results: we trained 2 state continuous HMMs with $\mathcal{Z}^l = (x_{t-l}, y_t)_{t=1...T}$ where x_t and y_t denote respectively LGS and $STPP$[6]. Again, the cross-correlogram of the two time series shows clearly that this approach does not suggest any value for λ. The second figure plots the results obtained with the maximum likelihood approach. A reasonably significant peak around 23 min can be seen. Moreover, by applying the Viterbi algorithm to the sequence $\mathcal{Z}^l = (x_{t-l}, y_t)$ with $l = 23$ min, we can see that the HMM stays in one state before the event, jumps to the other state precisely when the formation change occurs and then remains in the second state. Such a clear sequence could not be obtained with other HMM^{l}s trained with a different value for l, confirming the computed value for λ. The third figure plots the results obtained with the maximum mutual information approach. Again,

[6]The hook load parameter does not give good results for this data set.

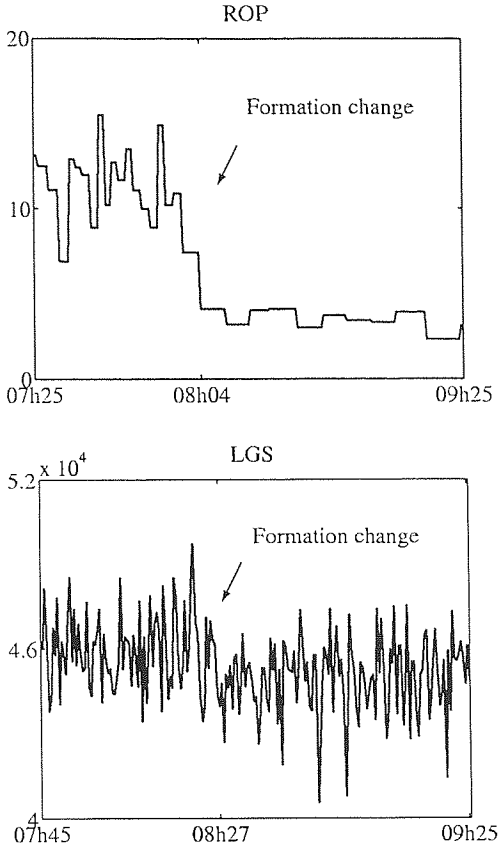


Figure 4.6: Evolution of *ROP* and *LGS* for a particular event in drilling operations. The formation changes from soft to hard rock at a certain time. The lag of return is visually identifiable in such a situation.

this method suggests a sharper peak and confirms the value of $\lambda = 23$ min.

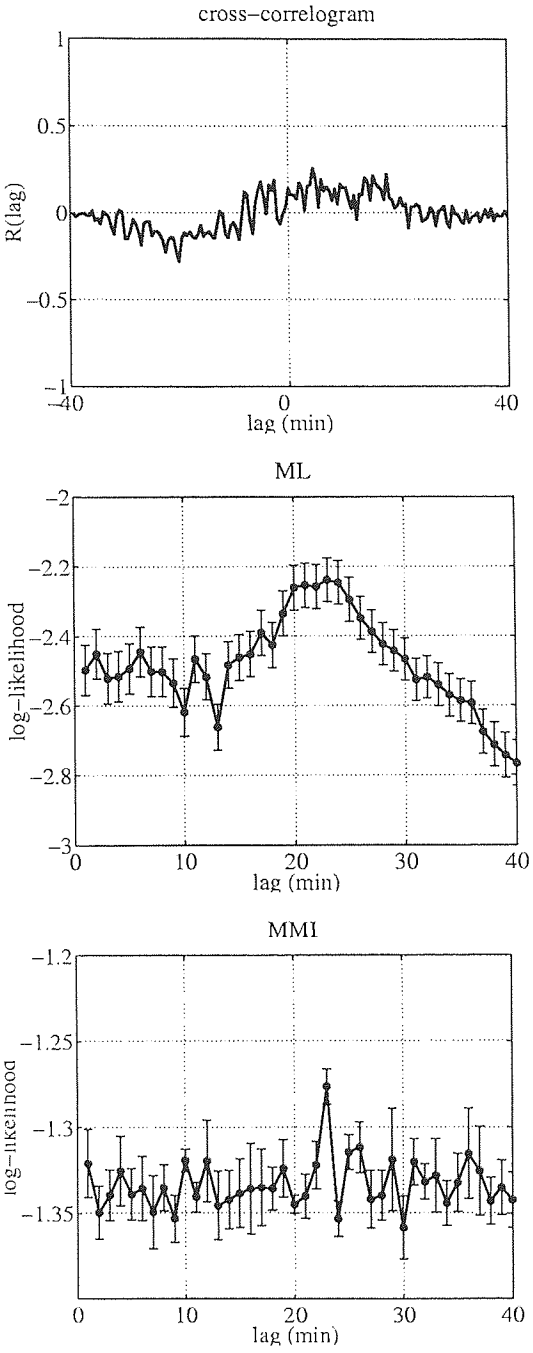


Figure 4.7: Formation change: cross-correlogram (top) and log-likelihood of the observation sequences $\mathcal{Z}^l$ given the model HMM^l for ML (middle) and MMI (bottom) approaches.

To conclude this section, Table 4.2 reports the results of our simulations on 4 different data sets and shows how they differ from the ones obtained from the fluid mechanics models. In no case was it possible to identify the lag from the cross-correlogram.

Technique	Dataset A	Dataset B	Dataset C	Dataset D
Fluid Mech.	68 min	31 min	36 min	43 min
ML	72-75 min	36-40 min	40-43 min	22-25 min
MMI	74 min	37 min	42 min	23 min

Table 4.2: Our results compared to the ones obtained by fluid mechanics models.

4.5 Discussion

In this chapter, we have shown how hidden Markov models can be used for identifying relationships between variables in a non-stationary environment. We proposed two approaches for training the models for this application. The first one uses the usual maximum likelihood estimation and consists of maximising the joint probability of the two variables. The second approach uses a novel mutual information estimation approach, which maximises the conditional probability of one variable with respect to the other.

We tested both techniques on data from a real-world process and clearly demonstrated their ability to outperform traditional cross-correlation methods. We focussed on the estimation of a crucial parameter of the drilling process and obtained better results than the numerical models based on fluid mechanics used in the oil industry. When comparing the ML and the MMI approaches, the latter seems consistently to estimate the lag more precisely.

Whereas the ML estimation can be implemented with an exact EM algorithm, the MMI needs gradient information of the expected log-likelihood which leads to a generalised EM algorithm. As indicated in Section 4.2, the MMI implementation can be time consuming and we have noticed that it is more sensitive to initialisation than the ML.

We also showed how a fast and efficient K-means procedure can be used in order to initialise hidden Markov models. This technique was applied to a simple synthetic problem and gave significant results compared to random initialisation.

Concerning the estimation of the lag for return for the oil drilling data, as mentioned above, we have noticed difficulties associated with the variable selection, as no general characterisation of the most relevant drilling parameters is available at this stage. One way to tackle this problem would be to use the qualitative relationships given by numerical models.

One direct and promising extension of this work would be to provide an on-line estimate of the lag for return for the purposes of early stuck pipe detection. The current algorithms are batch algorithms. By considering only a forward pass in the forward-backward algorithm, it is indeed possible to derive sequential learning algorithms.

Chapter 5

Variational techniques for probabilistic inference

5.1 Introduction

Probability theory provides a principled framework for quantifying uncertainty and its benefits in many applied fields has become increasingly apparent (Pearl, 1988). However, probabilistic solutions often require integrating over distributions of unobserved variables, and for many models, these integrals are computationally intractable. One way of making further progress is through the development of approximations.

A probabilistic model defines a joint distribution over a set of random variables. In many models, such as hidden Markov models or linear dynamical systems, it is often the case that some variables are not observed whereas others are visible. In that case, we need to compute

$$P(\boldsymbol{s} \,|\, \boldsymbol{y}, \boldsymbol{w}) = \frac{P(\boldsymbol{s}, \boldsymbol{y} \,|\, \boldsymbol{w})}{P(\boldsymbol{y} \,|\, \boldsymbol{w})}, \tag{5.1}$$

where $\boldsymbol{w}$ denotes the adaptive parameters of the model, $\boldsymbol{S}$ the set of hidden random variables and $\boldsymbol{Y}$ represents the set of observable random variables. The denominator $P(\boldsymbol{y} \,|\, \boldsymbol{w}) = \int P(\boldsymbol{s}, \boldsymbol{y} \,|\, \boldsymbol{w})\, d\boldsymbol{s}$ is another important quantity known as the *likelihood* and is closely related to the calculation of the posterior distribution $P(\boldsymbol{s} \,|\, \boldsymbol{y}, \boldsymbol{w})$. Learning algorithms that maximise the likelihood often make use of the calculation of $P(\boldsymbol{s} \,|\, \boldsymbol{y}, \boldsymbol{w})$. For example, in Chapter 3 we saw how to train hidden state models with the EM algorithm: the E-step consists of estimating the posterior probability $P(\boldsymbol{s} \,|\, \boldsymbol{y}, \boldsymbol{w})$. The M-step makes use of this distribution for maximising the expected log-likelihood with respect to the parameters $\boldsymbol{w}$ of the model.

For complex models, the integration over the hidden variables $\boldsymbol{s}$ is computationally not

feasible. For such probabilistic models, it is necessary to develop approximation procedures in order to render the inference problem tractable.

Variational techniques involve the introduction of an approximating distribution $Q(s)$. Consider the following expression

$$\begin{aligned} \log P(y \mid w) &= \log \int P(s, y \mid w)\, ds \\ &= \log \int Q(s) \frac{P(s, y \mid w)}{Q(s)}\, ds \\ &\geq \int Q(s) \log \frac{P(s, y \mid w)}{Q(s)}\, ds \\ &\equiv \mathcal{F}(Q, w) \end{aligned} \tag{5.2}$$

where we have used Jensen's inequality. This forms a rigorous lower bound $\mathcal{F}(Q, w)$ on the true log-likelihood. By choosing a judicious distribution $Q(s)$, the computation of the lower bound may be tractable. It is easy to see that the difference between the left hand side and the right hand side of Equation (5.2) is nothing else than the KL-divergence between the approximating distribution $Q(s)$ and the true posterior $P(s \mid y, w)$:

$$\begin{aligned} \log P(y \mid w) - \mathcal{F}(Q, w) &= \int Q(s) \left(\log P(y \mid w) - \log \frac{P(s, y \mid w)}{Q(s)} \right) ds \\ &= \int Q(s) \log \frac{P(s, y \mid w)}{P(s \mid y, w)} \frac{Q(s)}{P(s, y \mid w)}\, ds \\ &= \int Q(s) \log \frac{Q(s)}{P(s \mid y, w)}\, ds \\ &= \mathrm{KL}(Q \| P). \end{aligned} \tag{5.3}$$

The KL-divergence between Q and P is a non-negative expression and is minimised if and only if $Q = P$ in which case it is zero and the bound becomes exact. However, this would not lead to any simplification of the problem.

The goal of variational techniques is to choose a suitable form of the approximating distribution which is sufficiently simple to compute the lower bound on the true log-likelihood and sufficiently flexible to keep this bound tight. For this purpose, one can consider a parametric family $\mathcal{Q}$ of distributions $Q(s \mid \Phi)$ governed by a set of *variational* parameters Φ. From the family of distributions, we choose a particular one $Q(s \mid \Phi^*)$ which minimises the KL-divergence:

$$\Phi^* = \arg\min_{\Phi} \mathrm{KL}\left(Q(s \mid \Phi) || P(s \mid y, w) \right). \tag{5.4}$$

In section 5.2, we will see that graphical models provide a natural framework for choosing a suitable parametric family.

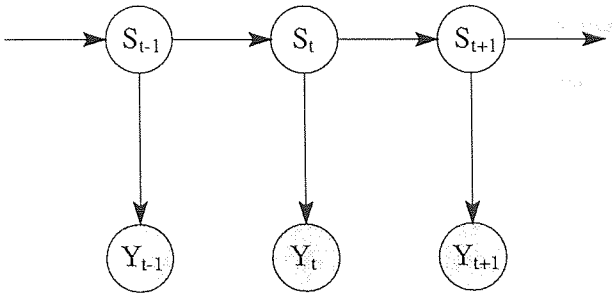


Figure 5.1: A Bayesian network specifying the independence relations for a hidden Markov model.

As discussed in Neal and Hinton (1998), there is an interesting link between variational techniques and the EM algorithm. Recall that the E-step consists of inferring the posterior distribution $P(\boldsymbol{s} \mid \boldsymbol{y}, \boldsymbol{w})$. The above derivations show that the E-step can still be applied when the true posterior is not tractable. It is actually possible to generalise the E-step by using a tractable distribution Q and minimise the KL-divergence with respect to this approximation. In Chapter 6, we will make use of this remark for models where the E-step is computationally intractable. Note also that it is not necessary to fully maximise $\mathcal{F}(Q, \boldsymbol{w})$ over Q in the E-step. A partial E-step, where the function $\mathcal{F}$ is locally optimised, can be implemented.

This chapter is intended to be a brief review of variational techniquse. A good overview of these techniques can be found in (Jordan *et al.*, 1998). Because we will make use of variational techniques in the next chapter, we prefer to present the fundamental ideas in this introductory chapter. In Chapter 6, we will see how to use variational methods for training hybrid models that combine HMMs and SSMs.

5.2 Variational techniques in graphical models

Bayesian networks (Pearl, 1988) are graphical models for representing conditional dependencies between a set of random variables. Figure 5.2 shows a Bayesian network for an HMM. The figure shows a directed acyclic graph in which each node corresponds to a random variable. A directed arc is drawn from node $\boldsymbol{U}$ to node $\boldsymbol{V}$ if $\boldsymbol{V}$ is conditioned on $\boldsymbol{U}$ in the factorisation of the joint distribution. $\boldsymbol{U}$ is a *parent* of $\boldsymbol{V}$ and $\boldsymbol{V}$ is a *child* of $\boldsymbol{U}$. The absence of an edge from node $\boldsymbol{U}$ to node $\boldsymbol{V}$ implies an unconditional independence between $\boldsymbol{U}$ and $\boldsymbol{V}$. For example we draw an arc between $\boldsymbol{S}_t$ and $\boldsymbol{S}_{t+1}$ but not from $\boldsymbol{Y}_t$ to $\boldsymbol{Y}_{t+1}$.

A Bayesian network allows the user not only to understand the relationships between the random variables but is also crucial for computing marginal and conditional probabilities required in the inference and learning problems (Pearl, 1988; Heckerman, 95; Smyth *et al.*,

1997; Jordan *et al.*, 1998). Such models have been proven to be useful for modelling the causal structure of complex systems involving several interacting variables. The problem of probabilistic inference in graphical models is to compute the conditional probability distribution over the values of some of the random variables (represented as nodes) given the value of other variables.

There are many cases where exact inference is not computationally feasible. A good example is the QMR-DT database which is a large-scale probabilistic database intended to be used as a diagnostic aid. The corresponding graphical model is bipartite in which the upper layer nodes represent diseases and the lower layer nodes are symptoms. Because of the huge number of variables, exact inference is not computationally feasible (Shwe *et al.*, 1991).

The key question in variational techniques is how to pick a tractable parameterisation for Q. Graphical models are actually of great help. The idea is to identify a simplified structure in the graphical model which renders the inference problem tractable. This is done by eliminating edges in the Bayesian network and choosing Q from the family of distributions defined by the simplified graph. Of course, the simplified graph must be rich enough to provide an approximate distribution Q close to the true distribution P. The simplest approximation consists of deleting all the existing edges of the graph. This leads to a completely factorised distribution Q where all the variables are independent. This approximation, known as the *mean field approximation*, is often used in statistical mechanics (Parisi, 1988) and has been applied in the neural computing community for the case of Boltzmann machine (Peterson and Anderson, 1987) and sigmoid belief networks (Saul *et al.*, 1996). *Structured* approximation is another type of approximation where some but not all of the edges are removed. The idea is to preserve substructures of the original graphical model for which exact inference is computationally tractable (Saul and Jordan, 1996).

In order to illustrate these two approximations, consider the following example inspired by (Ghahramani, 1997). Figure 5.2 shows a Bayesian network with T hidden variables and one visible variable. The corresponding joint probability is given by:

$$P(s_1, \ldots, s_T, \boldsymbol{y} \mid \boldsymbol{w}) = P(s_1 \mid \boldsymbol{w}) \prod_{t=2}^{T} P(s_t \mid s_{t-1}, \boldsymbol{w}) P(\boldsymbol{y} \mid s_1, \ldots, s_T, \boldsymbol{w}). \tag{5.5}$$

Consider the simple case where the hidden variables are discrete and let represent the variable

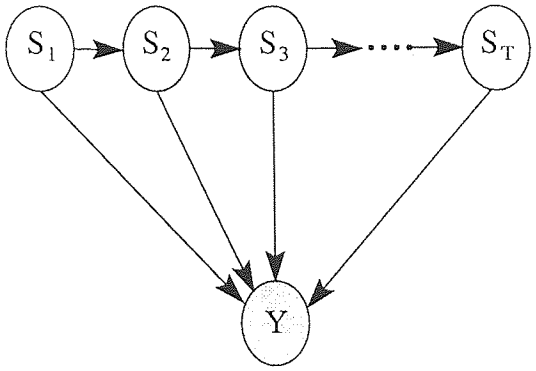


Figure 5.2: A Bayesian network for which inference is not computationally feasible.

s_t by a N-dimensional vector $s_t = [s_t^1, \ldots, s_t^N]$ where $s_t^i \in \{0, 1\}$ so that

$$P(s_1 \,|\, \pi) = \prod_{i=1}^{N} \pi_i^{s_1^i}, \tag{5.6}$$

$$P(s_t \,|\, s_{t-1}, A) = \prod_{i=1}^{N} \prod_{j=1}^{N} (a_{ij})^{s_{t-1}^i s_t^j}, \tag{5.7}$$

where π represents the initial probabilities and $\boldsymbol{A} = \{a_{ij}\}$ the transition probabilities matrix. We will assume, for simplicity, that the output variable $\boldsymbol{y}$ is one-dimensional and normally distributed with a mean specified by a linear combination of the hidden variables:

$$P(y \,|\, s_1, \ldots, s_T) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2} (y - \sum_{t=1}^{T} \sum_{i=1}^{N} w_t^i s_t^i)^2 \right\}. \tag{5.8}$$

In order to compute the posterior probability of a specific hidden variable, say s_t, given the visible observation y, we need to sum over all the other hidden variables:

$$\begin{aligned} P(s_t \,|\, y, \boldsymbol{w}) &= \sum_{s_1, \ldots, s_{t-1}, s_{t+1}, \ldots, s_T} P(s_1, \ldots, s_T \,|\, y, \boldsymbol{w}) \\ &= \frac{\sum_{s_1, \ldots, s_{t-1}, s_{t+1}, \ldots, s_T} P(s_1, \ldots, s_T, y \,|\, \boldsymbol{w})}{\sum_{s_1, \ldots, s_T} P(s_1, \ldots, s_T, y \,|\, \boldsymbol{w})}. \end{aligned} \tag{5.9}$$

In the case of discrete variables with N possible values, the sum in the denominator contains N^T terms. In order to circumvent this problem, a simple approximation consists of defining a distribution Q where all the hidden variables s are independent given y, that is

$$Q(s_1, \ldots, s_T \,|\, \Phi) = Q(s_1 \,|\, \phi_1) \ldots Q(s_T \,|\, \phi_T). \tag{5.10}$$

This approximation corresponds to the simple mean field approximation (Figure 5.3a) and a vector of variational parameter Φ is associated with Q:

$$Q(s_1, \ldots, s_T \,|\, \Phi) = \prod_{t=1}^{T} \prod_{i=1}^{N} (\phi_t^i)^{s_t^i}, \tag{5.11}$$

(a) (b)

Figure 5.3: a) The mean field approximation assumes that all the hidden states $\boldsymbol{S}_t$ are independent. b) The structured approximation keeps the Markov chain.

in which the hidden variables $\boldsymbol{s}_t$ appear as independent variables with adjustable means $\boldsymbol{\phi}_t$. The values of these variational parameters are obtained by minimising the KL-divergence (Equation (5.3)), or similarly, maximising the lower bound on the log-likelihood (Equation (5.2)). For our example, this gives:

$$\begin{aligned}
\mathcal{F}(Q, \boldsymbol{w}) &= \langle \log P(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T, y \mid \boldsymbol{w}) \rangle_Q - \langle \log Q \rangle_Q \\
&= \sum_{i=1}^{N} \phi_1^i \log \pi_i + \sum_{t=2}^{T} \sum_{i,j=1}^{N} \phi_{t-1}^i \phi_t^j \log a_{ij} \\
&\quad - \frac{1}{2\sigma} \left(y^2 + \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j \neq i}^{N} w_t^i w_t^j \phi_t^i \phi_t^j + \sum_{t=1}^{T} \sum_{i=1}^{N} w_t^{i2} \phi_t^i - 2y \sum_{t=1}^{T} \sum_{i=1}^{N} w_t^i \phi_t^i \right) \\
&\quad - \sum_{t=1}^{T} \sum_{i=1}^{N} \phi_t^i \log \phi_t^i - \frac{1}{2} \log(2\pi\sigma^2),
\end{aligned} \tag{5.12}$$

where we have used the fact that under Q, $\langle s_t^i \rangle_Q = \phi_t^i$ and $\langle s_t^i s_t^i \rangle_Q = \phi_t^i$. By taking the derivatives of Equation (5.12) and setting them to zero, we get a set of fixed point equations for the variational parameters:

$$\phi_t^i = \rho \left(\sum_{j=1}^{N} (\phi_{t-1}^j \log a_{ji} + \phi_{t+1}^j \log a_{ij}) + \frac{w_t^i}{\sigma^2} (y - \sum_{j \neq i}^{N} w_t^j \phi_t^j - \frac{1}{2} w_t^i) \right), \tag{5.13}$$

where $\rho(x_i) = \frac{\exp(x_i)}{\sum_i exp(x_i)}$ is the softmax function. We see that the equation couples the variational parameter of each node with the parameters of its predecessor and successor. Starting with some initial values for the mean field parameters ϕ_t^i, the fixed point equations are iterated until convergence of the KL-divergence.

A structured variational approximation is an approximation in which the variables are not completely factorised and is obtained by removing some edges of the graph and keeping others. In this example, we can keep the structure amongst the hidden variables as these variables define a Markov chain and an exact algorithm exists for solving the corresponding inference problem (see Chapter 3). In this case, the corresponding variational approximation

is:

$$Q(s_1, \ldots, s_T \mid \Phi) = \frac{1}{Z_Q} Q(s_1 \mid \phi_1) \prod_{t=2}^{T} Q(s_t \mid s_{t-1}, \phi_t), \tag{5.14}$$

where we have introduced a variational parameters vector Φ which scales the probabilities of the hidden variables. Z_Q is a normalisation factor ensuring that Q integrates to one.

$$Q(s_1 \mid \phi_1) = \prod_{i=1}^{N} (\pi_i \phi_1^i)^{s_1^i}, \tag{5.15}$$

$$Q(s_t \mid s_{t-1}, \phi_t) = \prod_{i=1}^{N} \prod_{j=1}^{N} (a_{ij} \phi_t^j)^{s_{t-1}^i s_t^j}. \tag{5.16}$$

Figure 5.3b shows the corresponding graphical model. Comparing Equation (5.14)-(5.16) to Equation (5.5), we see that each variational parameter ϕ_t^i plays the role of a probability $P(y \mid s_t^i)$: we have indeed replaced the interaction of N hidden variables by N identical visible variables $\boldsymbol{Y}$. When expressing the lower bound on the log-likelihood, several terms disappear to give

$$\begin{aligned} \mathcal{F}(Q, \boldsymbol{w}) = & -\frac{1}{2} \left(y^2 + \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j \neq i}^{N} w_t^i w_t^j \langle s_t^i \rangle_Q \langle s_t^j \rangle_Q + \sum_{t=1}^{T} \sum_{i=1}^{N} w_t^{i2} \langle s_t^i \rangle_Q - 2y \sum_{t=1}^{T} \sum_{i=1}^{N} w_t^i \langle s_t^i \rangle_Q \right) \\ & - \sum_{t=1}^{T} \sum_{i=1}^{N} \langle s_t^i \rangle_Q \log \phi_t^i + \log Z_Q - \frac{1}{2} \log(2\pi\sigma^2). \end{aligned} \tag{5.17}$$

Taking the derivatives of Equation (5.17) with respect to ϕ_t^i and setting them to zero gives a set of fixed point equations for the variational parameters:

$$\phi_t^i \propto \exp \left\{ \frac{w_t^i}{\sigma^2} (y - \sum_{j \neq i}^{N} w_t^j \langle s_t^j \rangle_Q - \frac{1}{2} w_t^i) \right\}. \tag{5.18}$$

The equations make use of the expectations $\langle s_t^i \rangle_Q$. These expectations are obtained by running the forward-backward algorithm on an HMM where the output probability ϕ_t^i is associated to each hidden state. In section 3.4, we saw how to compute $\gamma_t(i) = \langle s_t^i \rangle_Q$. This defines an iterative procedure where the forward-backward algorithm is used as a subroutine for estimating $\langle s_t^i \rangle_Q$.

5.3 Discussion

In this chapter we have briefly reviewed the general framework of variational techniques for probabilistic inference and shown its application in graphical models. Variational methods

provide a rigorous lower bound on the likelihood and lead to optimising the KL divergence between the approximation and the true posterior distribution.

As discussed in (Jordan *et al.*, 1998), a key issue of variational techniques concerns the accuracy of the approximation. The choice of the approximation is often a matter of judgement and the tightness of the resulting bound is obviously affected by this choice. It is also important to note that, while the distribution $Q(s)$ might be easy to compute, the likelihood $Q(y \mid s, \Phi)$ under this distribution might still be intractable. As in Equation (5.14), the approximation often introduces a normalisation factor which renders the computation of the likelihood intractable. This can represent a problem for model comparison.

Other interesting problems concern the development of upper bounds on the log-likelihood and mixture models as approximating distributions. Research in variational techniques is recent and there are many open problems.

Chapter 6

Dynamical local models for time series

6.1 Introduction

Most forecasting approaches try to predict the next value of a time series by assuming stationarity: *i.e.* the *underlying* generator of the data is globally time invariant. In many real world applications, this assumption is not valid. Even non-linear regressors like neural networks are not effective in modelling changing temporal structure in the time series. For instance, one of the obstacles to the prediction of exchange rates in the capital markets is a non-constant conditional variance, known as heteroscedasticity. GARCH models have been developed to estimate a time-dependent variance (Bollerslev, 1986).

A special form of non-stationarity, where the underlying generator switches between (approximately) stationary regimes, seems a reasonable assumption for many practical problems. In the last decade, hybrid approaches have been developed in order to model this behaviour. One example is the mixture of experts (Jacobs *et al.*, 1991; Cacciatore and Nowlan, 1994; Weigend *et al.*, 1995) which decomposes the global model into several (linear or non-linear) local models known as *experts*, as each specialises in modelling a small region of input space. One limitation of these models for time series analysis is that the gating network which combines the local models has no dynamics. It is controlled only by the current value of the time series.

One way to address this limitation is to use a hidden Markov model (which does have dynamics) to switch between local models. For example, autoregressive hidden Markov models (ARHMMs) switch between autoregressive models, where the predictions are a linear com-

bination of past values (Poritz, 1982). ARHMMs have been reintroduced in the machine learning community under the name of hidden filter HMMs (Fraser and Dimitriadis, 1994) and have been recently applied to financial engineering in order to model high frequency foreign exchange data (Shi and Weigend, 1997).

From econometrics to control, several similar hybrid models have been proposed. Their main characteristic is the mixing of discrete and continuous hidden variables (Chang and Athans, 1977; Hamilton, 1989; Shumway and Stoffer, 1991; Bar-Shalom and Li, 1993). A linear system with Markovian coefficients, also called a jump-linear system, assumes the existence of a linear dynamical system of the general form:

$$\boldsymbol{x}_t = \boldsymbol{F}(s_t)\boldsymbol{x}_{t-1} + \boldsymbol{u}_t \tag{6.1}$$

$$\boldsymbol{y}_t = \boldsymbol{G}(s_t)\boldsymbol{x}_t + \boldsymbol{v}_t \tag{6.2}$$

where $\boldsymbol{x}_t$ is the state vector, $\boldsymbol{y}_t$ the measurement vector, and s_t the unknown time-varying parameter. s_t is restricted to take values from a finite set $\{q_1, \ldots, q_N\}$. In the simplest case, this parameter follows a first-order Markov process. The transition matrix governing the Markov chain and the parameters of the model are usually assumed to be known. The main problem consists thus of estimating the hidden state $\boldsymbol{x}_t$.

Chang and Athans (1977) focus on the state estimation problem for a system where the output matrix $\boldsymbol{G}$ is time independent. They show that estimation of the exact distribution of the state requires a bank of elemental estimators whose size grows exponentially in time. Mazor *et al.* (1998) review the state estimation problem for the most general case where both $\boldsymbol{F}$ and $\boldsymbol{G}$ are allowed to depend on a switch variable s_t. They also show why an optimal solution is not computationally tractable and present techniques known as 'interacting multiple models' that consist of a bank of cooperating Kalman filters: at each time step t the state estimate is computed under each possible current model, with each filter using a different combination of the previous model-conditioned estimates (see also (Blom and Bar-Shalom, 1988; Bar-Shalom and Li, 1993)).

Shumway and Stoffer (1991) consider the problem of learning the parameters of a state space model with a switching output matrix $\boldsymbol{G}(s_t)$ which is known in advance. They proposed an approximate EM algorithm where the E-step, which would require the computation of a mixture of Gaussians with an exponentially increasing number of components, is approximated at each time step t by a single Gaussian.

In this chapter, we investigate switching state space models (SSSMs). These models consist of N multiple linear/non-linear state space models controlled by a dynamic switch

and, in this sense are a generalisation of jump-linear systems. They assume that the behaviour of the system can be characterised by a finite number of dynamical systems with hidden states, each of which tracks the data in a different regime. As discussed in (Ghahramani and Hinton, 1998), SSSMs can also be seen as a generalisation of the mixture of experts model.

A long-standing limitation for training these models is that the complexity of the exact training algorithm grows exponentially with order N^T, where N is the number of models and T is the length of the time sequence. Various *ad hoc* and not completely satisfactory approximations have been proposed, *e.g.* (Shumway and Stoffer, 1991). Recently, Ghahramani and Hinton (1998) reintroduced linear switching state space models in the machine learning community and proposed an efficient and principled approximate algorithm for training these models in a maximum likelihood framework.

In section 6.2 we first present linear switching state space models (SSSMs) and show how to train these models using variational techniques. In section 6.3 we present a new extension which incorporates non-linear state space models using radial basis function (RBF) networks. Although linear SSSMs enable us to model piece-wise stationarity, they may have difficulties in modelling non-linear dependencies in the time series. As the initialisation step is crucial for training mixture models due to the large number of local minima, we present a novel algorithm which addresses this problem in section 6.4. We then show how to use these models for time series segmentation and probabilistic density prediction. The models are finally tested on different datasets and we compare their performance with other standard techniques.

6.2 Linear switching state space models

In Chapter 3, we reviewed two probabilistic models for time series: hidden Markov models and state space models. A linear switching state space model (linear SSSM) is a model that combines HMMs and SSMs. More precisely, N different linear dynamical systems compete in order to describe the observation $\boldsymbol{y}_t \in \mathbb{R}^d$. Each real-valued state vector $\boldsymbol{x}_t^{(i)} \in \mathbb{R}^m$ evolves between time steps according to the system equation:

$$\boldsymbol{x}_t^{(i)} = \boldsymbol{F}_i \boldsymbol{x}_{t-1}^{(i)} + \boldsymbol{u}_i, \tag{6.3}$$

where $\boldsymbol{F}_i$ is the state transition matrix and $\boldsymbol{u}_i \sim \mathcal{N}(0, \boldsymbol{Q}_i)$ is a zero mean Gaussian noise associated to model i. The initial state vector is also assumed to be Gaussian: $P(\boldsymbol{x}_1^{(i)}) = \mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$.

A discrete variable $S_t \in \{q_1, \ldots, q_N\}$, also represented by a vector $\boldsymbol{S}_t = [S_t^{(1)}, \ldots, S_t^{(N)}]$,

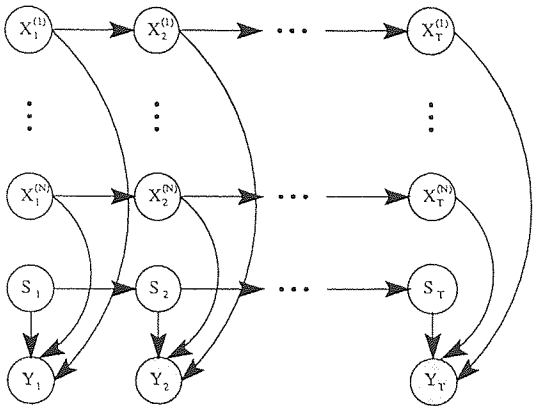


Figure 6.1: Graphical representation of a switching state space model. All the hidden variables have Markovian dynamics. At each time t, N real-valued hidden variables compete in order to explain the observation $\boldsymbol{y}_t$ and the discrete variable $\boldsymbol{s}_t$ plays the role of a gate.

where $S_t^{(i)} \in \{0,1\}$, plays the role of a gate. When the system enters a specific state i, *i.e.* $S_t = q_i$ (or $S_t^{(i)} = 1$), the observation is Gaussian and is given by:

$$\boldsymbol{y}_t = \boldsymbol{G}_i \boldsymbol{x}_t^{(i)} + \boldsymbol{v}_i, \tag{6.4}$$

where $\boldsymbol{G}_i$ is the output matrix which maps the hidden state to the observation. The random variable $\boldsymbol{v}_i \sim \mathcal{N}(0, \boldsymbol{R}_i)$ is also a zero mean Gaussian noise. The discrete state variable S_t evolves according to Markovian dynamics that can be represented by a discrete transition matrix $\boldsymbol{A} = \{a_{ij}\}$,

$$a_{ij} = P(S_t = q_j \mid S_{t-1} = q_i). \tag{6.5}$$

Therefore, an SSSM is essentially a mixture model, in which information about the past is captured in two types of random variables: one continuous and one discrete. Using the Markov dependence relations, the joint probability for the sequence of states and observations can be written as

$$\begin{aligned} P(\mathcal{S}_1^T, \mathcal{X}_1^{T(1)}, \ldots, \mathcal{X}_1^{T(N)}, \mathcal{Y}_1^T) &= P(\boldsymbol{s}_1) \prod_{t=2}^{T} P(\boldsymbol{s}_t | \boldsymbol{s}_{t-1}) \prod_{i=1}^{N} \left(P(\boldsymbol{x}_1^{(i)}) \prod_{t=2}^{T} P(\boldsymbol{x}_t^{(i)} | \boldsymbol{x}_{t-1}^{(i)}) \right) \\ & \prod_{t=1}^{T} P(\boldsymbol{y}_t | \boldsymbol{x}_t^{(1)}, \ldots, \boldsymbol{x}_t^{(N)}, \boldsymbol{s}_t). \end{aligned} \tag{6.6}$$

The corresponding graphical model is shown in Figure 6.1.

Given a sequence of observations $\mathcal{Y}_1^T$, the learning problem consists of estimating the parameters $\boldsymbol{w} = \{\boldsymbol{F}_i, \boldsymbol{Q}_i, \boldsymbol{G}_i, \boldsymbol{R}_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{1 \leq i \leq N}$ of each Kalman filter and the transition matrix $\boldsymbol{A}$ of the discrete state Markov process in order to maximise the likelihood of the observations. An exact procedure to solve this maximum likelihood estimation could be derived from the

Expectation-Maximisation algorithm (Dempster *et al.*, 1977). As discussed in Chapter 3, the E-step consists of computing the posterior probabilities $P(\mathcal{S}_1^T, \mathcal{X}_1^{T^{(1)}}, \ldots, \mathcal{X}_1^{T^{(N)}} | \mathcal{Y}_1^T, \boldsymbol{w})$ of the hidden states. The M-step uses the expected values to re-estimate the parameters of the model.

Unfortunately, it can be shown that exact inference is not computationally tractable, since it scales as N^T. Even if $P(\boldsymbol{x}_1^{(i)} | \boldsymbol{y}_1, \boldsymbol{w})$ is Gaussian, then $P(\boldsymbol{x}_t^{(i)} | \boldsymbol{y}_1^t, \boldsymbol{w})$ is in general a mixture of Gaussians with an exponentially increasing number of terms. Like the other models described in section 6.1, the posterior distribution of the state variables $\boldsymbol{x}_t^{(i)}$ is a mixture of Gaussians with N^t components. Although these variables are marginally independent, they become conditionally dependent when the variable $\boldsymbol{y}_t$ is observed, namely because of the discrete variable $\boldsymbol{S}_t$ which couples all the real-valued state variables $\boldsymbol{x}_t^{(1)}, \ldots, \boldsymbol{x}_t^{(i)}$ at time step t.

Several approximations have been proposed to circumvent this difficulty. For example, in (Shumway and Stoffer, 1991), a pseudo-EM algorithm is derived for learning a single hidden state space model with switching output matrices: at each step, the mixture of Gaussians is approximated by a single Gaussian. Recently Ghahramani and Hinton (1998) proposed a principled generalised EM algorithm. The idea is to make use of variational techniques in order to approximate the intractable true posterior distribution by a tractable distribution Q, and to maximise the lower bound on the log-likelihood (see Chapter 5):

$$\mathcal{F}(Q, \boldsymbol{w}) = \sum_{\mathcal{S}_1^T} \int Q(\mathcal{S}_1^T, \mathcal{X}_1^T) \log \frac{P(\mathcal{S}_1^T, \mathcal{X}_1^T, \mathcal{Y}_1^T \,|\, \boldsymbol{w})}{Q(\mathcal{S}_1^T, \mathcal{X}_1^T)} d\mathcal{X}_1^T, \tag{6.7}$$

where $\mathcal{X}_1^T$ denotes the whole sequence of hidden states: $\mathcal{X}_1^T = [\mathcal{X}_1^{T^{(1)}}, \ldots, \mathcal{X}_1^{T^{(N)}}]$.

Using a judicious structured variational approximation, the inference step can become tractable (Saul and Jordan, 1996). Because linear SSSMs are hybrid models combining HMMs and SSMs for which the E-step can be solved exactly, it is best to use an approximation that makes use of the forward-backward and Kalman smoother algorithms (see Chapter 3). The authors suggest the following approximation:

$$Q(\mathcal{S}_1^T, \mathcal{X}_1^{T^{(1)}}, \ldots, \mathcal{X}_1^{T^{(i)}}) = \frac{1}{Z} \Phi(s_1) \prod_{t=2}^{T} \Phi(s_{t-1}, s_t) \prod_{i=1}^{N} \Phi(\boldsymbol{x}_1^{(i)}) \prod_{t=1}^{T} \Phi(\boldsymbol{x}_{t-1}^{(i)}, \boldsymbol{x}_t^{(i)}) \tag{6.8}$$

which corresponds to the graphical model shown in Figure 6.2. Z is a normalisation factor ensuring that Q integrates to one.

The motivation of such an approximation is to destroy the interaction between the hidden variables which makes the inference problem computationally intractable. Each deleted edge

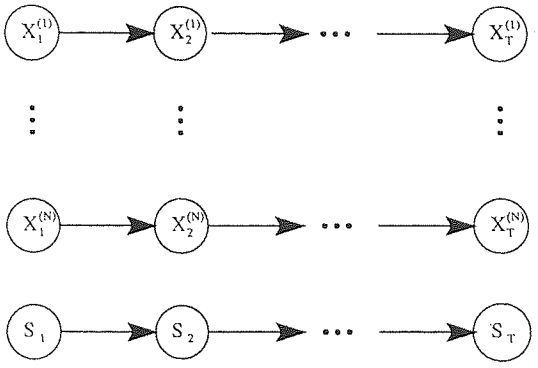


Figure 6.2: Structured variational approximation of a switching state space model. We have uncoupled the state space models but kept the Markov chain for each hidden variables. Exact inference for each hidden variable is now tractable.

in the graph is replaced by a variational parameter:

$$\Phi(s_1^{(i)}) = P(s_1^{(i)})q_1^{(i)} \tag{6.9}$$

$$\Phi(s_{t-1}^{(j)}, s_t^{(i)}) = P(s_t^{(i)} \mid s_{t-1}^{(j)})q_t^{(i)} \tag{6.10}$$

$$\Phi(\boldsymbol{x}_1^{(i)}) = P(\boldsymbol{x}_1^{(i)}) \left[P(\boldsymbol{y}_1 \mid \boldsymbol{x}_t^{(i)}, s_1^{(i)})\right]^{h_1^{(i)}} \tag{6.11}$$

$$\Phi(\boldsymbol{x}_{t-1}^{(i)}, \boldsymbol{x}_t^{(i)}) = P(\boldsymbol{x}_t^{(i)} \mid \boldsymbol{x}_{t-1}^{(i)}) \left[P(\boldsymbol{y}_t \mid \boldsymbol{x}_t^{(i)}, s_t^{(i)})\right]^{h_t^{(i)}}. \tag{6.12}$$

By introducing these variational parameters, we decouple the state space models but keep the Markov chain assumption for each of them.

The variational parameters $q_t^{(i)}$ and $h_t^{(i)}$ are obtained by minimising the KL-divergence between P and Q, which corresponds to the E-step. Ghahramani and Hinton (1998) derived the fixed point equations for these parameters. The parameters $q_t^{(i)}$ play exactly the same role as the output probabilities $P(\boldsymbol{y}_t \mid \boldsymbol{x}_t^{(i)})$ would play in a regular hidden Markov model, and are obtained by computing the expected error under the distribution Q if state space model i were used to generate the observation y_t:

$$q_t^{(i)} = \exp\left\{-\frac{1}{2}\langle(\boldsymbol{y}_t - \boldsymbol{G}_i\boldsymbol{x}_t^{(i)})' \boldsymbol{R}_i(\boldsymbol{y}_t - \boldsymbol{G}_i\boldsymbol{x}_t^{(i)})\rangle_Q\right\} \tag{6.13}$$

We can see that this parameter is a function of $\boldsymbol{x}_{t|T}^{(i)} \equiv \langle \boldsymbol{x}_t^{(i)} \rangle_Q$ and $\boldsymbol{V}_{t|T}^{(i)} \equiv \langle \boldsymbol{x}_t^{(i)} \boldsymbol{x}_t^{(i)'} \rangle_Q$. These expectations can be computed by running the Kalman smoother on state space model i with the observation $\boldsymbol{y}_t$ weighted by $h_t^{(i)}$ (see Equation (6.11) and Equation (6.12)). The parameters $h_t^{(i)}$ can be viewed as being the responsibility assigned to state space model i at time t, and are obtained by computing the expected probability of being in state i at time t under the approximating distribution Q.

$$h_t^{(i)} = \langle s_t^{(i)} \rangle_Q \tag{6.14}$$

They have exactly the same interpretation as the parameters $\gamma_t(i)$ in a regular HMM (see section 3.4).

We therefore see that the variational parameters are inter-related: the calculation of $q_t^{(i)}$ needs $h_t^{(i)}$ and vice-versa. Starting from some initial values for q and h, the E-step consists of running a Kalman smoother for each state space model with the output noise covariance matrix $\boldsymbol{R}_i$ weighted by $1/h_t^{(i)}$. This allows us to compute $q_t^{(i)}$ according to Equation (6.13) and the required expectations of each real-valued variable $\boldsymbol{x}_t^{(i)}$ needed in the M-step. The $h_t^{(i)}$ parameters are obtained by running a forward-backward algorithm, where each hidden state is associated to the output probability density $q_t^{(i)}$. The process is iterated until convergence of the KL-divergence. In practice, this is achieved in no more than 10 iterations.

The M-step consists of re-estimating the parameters $\boldsymbol{w}$ of the model and is straightforward. Like in HMMs and SSMs, the parameters can be re-estimated analytically. Appendix C gives the re-estimation equations.

The whole process (E and M steps) is iterated until convergence of the lower bound on the log-likelihood. This bound is a function of the variational parameters (see Appendix C):

$$\mathcal{F}(Q, \boldsymbol{w}) = \sum_{t=1}^{T} \sum_{i=1}^{N} h_t^{(i)} \log q_t^{(i)} + \log Z. \tag{6.15}$$

Note that the normalisation factor Z is intractable to compute.

6.3 Non-linear switching state space models

Although linear SSSMs are capable of modelling multi-modality, they may have difficulties in modelling non-linear dependencies in the time series. We present here a new extension of dynamical local models which takes into account non-linearity in the output:

$$\boldsymbol{y}_t = g_i(\boldsymbol{x}_t^{(i)}) + \boldsymbol{v}_i, \tag{6.16}$$

where g_i denotes now a non-linear function from the hidden state space to the observation space. By introducing this non-linearity, the posterior $P(\boldsymbol{x}_t^{(i)}|\boldsymbol{y}_1^T)$ is no longer Gaussian and optimal smoothing cannot be achieved analytically.

In order to circumvent this problem, one solution could be derived from sequential Monte Carlo integration techniques (Kitagawa, 1987; Gordon *et al.*, 1993; Kitagawa, 1996). These techniques have been applied for the inference problem in non-linear state space models, and the extension to the case of non-linear switching state space models could be investigated. In

these methods also known as bootstrap filter or sequential important sampling, arbitrary non-Gaussian densities are approximated by many particles that can be considered realisations from the distribution. It is then possible to derive a learning algorithm which makes use of these particles to fit the non-linear functions. However, these techniques are computationally expensive as a huge number of particles are needed at each time step t to be representative of the posterior distribution.

If the function g_i is sufficiently smooth, a suboptimal smoothing algorithm can be derived by considering the *linearisation* of the non-linear system. At every point $\boldsymbol{x}_{t|T}^{(i)}$, the function g_i is expanded as a first-order Taylor series:

$$g_i(\boldsymbol{x}) \approx g_i(\boldsymbol{x}_{t|T}^{(i)}) + \nabla_{\boldsymbol{x}} g_i(\boldsymbol{x}_{t|T}^{(i)})(\boldsymbol{x} - \boldsymbol{x}_{t|T}^{(i)}). \tag{6.17}$$

This approximate solution through linearisation around the current state estimate recovers the Gaussian structure and leads to the first-order *extended* Kalman smoother which is the exact Kalman smoother for the linearised model: the equations of the Kalman smoother derived in section 3.5 are still valid except those involving the output matrix $\boldsymbol{G}_i$ which is replaced by the Jacobian matrix $\mathcal{J}_{t|T}^{(i)} = \nabla_{\boldsymbol{x}} g_i(\boldsymbol{x}_{t|T}^{(i)})$.

The second complication arises in the M-step. In the case of a linear model, it is easy to obtain an exact re-estimation formulae for the parameters. If the functions g_i are not linear, it may be computationally difficult to re-estimate exactly the parameters of the function. For example, if g_i is represented by a multilayer neural network, exact re-estimation cannot be done and we must resort to non-linear optimisation methods.

To solve these two problems, we propose to model each non-linear function with a radial basis function network:

$$\boldsymbol{y}_t = \sum_{k=1}^{K} w_k^{(i)} \psi_k^{(i)}(\boldsymbol{x}_t^{(i)}) + \boldsymbol{v}_i = \boldsymbol{W}^{(i)} \boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_t^{(i)}) + \boldsymbol{v}_i, \tag{6.18}$$

where $\boldsymbol{W}^{(i)} = [w_1^{(i)}, \ldots, w_K^{(i)}]$ are the weights (including the bias) and $\{\psi_k^{(i)}\}_{2 \leq k \leq K}$ denote the $(K-1)$ Gaussian basis functions associated to model i (the bias is associated to a basis function whose activation is equal to 1):

$$\psi_k^{(i)}(\boldsymbol{x}_t^{(i)}) = \exp\left(-\frac{||\boldsymbol{x}_t^{(i)} - \boldsymbol{m}_k^{(i)}||^2}{2{\sigma_k^{(i)}}^2} \right). \tag{6.19}$$

Note that non-Gaussian basis functions could be used although we did not investigate their implementation in this work.

In that case, with fixed basis functions, the M-step is still tractable since the output function is linear with respect to the weight matrix $\boldsymbol{W}^{(i)}$. A good initialisation enables us

to keep the centres and widths of the basis functions fixed during the learning algorithm and to re-estimate only the weights, for which a fast and efficient algorithm exists[1]. Appendix C gives the re-estimation formulae for the weight matrix $\boldsymbol{W}^{(i)}$.

The number of basis functions K controls the smoothness of the output function g_i for each state space model. It is therefore possible to implement a non-linear SSSM with a number of basis functions that are different from one state space model to another. This can be quite useful if we believe, for example, that the underlying system is switching from a piecewise linear regime to a highly non-linear regime.

In terms of previous work, our model resembles that of (Kadirkamanathan and Kadirkamanathan, 1996), where the authors used modular RBF networks for learning multiple modes. Given input-output observations $\boldsymbol{z}_1^T = \{\boldsymbol{x}_1^T, \boldsymbol{y}_1^T\}$, their algorithm uses the Kalman filter for supervised recursive estimation of the weight vectors $\boldsymbol{W}^{(i)}$, which plays the role of the real-valued hidden state:

$$\boldsymbol{W}_t^{(i)} = \boldsymbol{W}_{t-1}^{(i)} + \boldsymbol{u}_i \tag{6.20}$$

$$\boldsymbol{y}_t = \boldsymbol{W}_t^{(i)} \boldsymbol{\Psi}_i(\boldsymbol{x}_t) + \boldsymbol{v}_i. \tag{6.21}$$

It is assumed that each model i has an associated score of being the current underlying model for the given observation $\boldsymbol{y}_t$. The parameters of the global model, for example the output noise covariance matrices $\boldsymbol{R}_i$ or the transition matrix $\boldsymbol{A}$, are not learned but are assumed to be known in advance. Our non-linear model differs from the modular RBF network on two major points. Firstly, in our approach, the parameters of each expert are learned in a maximum likelihood framework. Secondly, whereas the weight vectors $\boldsymbol{W}^{(i)}$ play the role of the hidden states in their model, they are considered as proper adaptive parameters of each RBF network in our work. This leads to a system where the hidden state is an input to the RBF network and therefore keeps its intuitive interpretation of representing the underlying dynamics we are trying to recover.

6.4 Initialisation

Mixture models trained using the EM algorithm are guaranteed to reach a local maximum likelihood solution. Because there are many local maxima, experience has shown that SSSMs are particularly sensitive to the initialisation. Therefore, the choice of initial conditions is

[1] If we want to learn these parameters, a generalised EM can be implemented.

crucial and we prefer to initialise the model carefully rather than a simple random initialisation.

For switching state space models, the initialisation is an important part of the learning algorithm, as both the HMM and the dynamical systems must be initialised. The key point is to start with a good segmentation of the data set, where by segmentation we mean a partition of the data, with each part modelled by a dynamical system. To address this problem, we have developed an efficient initialisation procedure.

For the linear case, we quickly[2] train a continuous hidden Markov model with as many discrete states as our SSSM on the data set and run the *Viterbi* algorithm in order to obtain the *most likely* path, *i.e.* the sequence of hidden states which 'best' explains the observation sequence (see section 3.4). Each data point is assigned to the most probable hidden state and thus gives us a segmentation of the data. A simple linear dynamical system is then initialised for each segment. This second phase can be done by estimating the covariance of the observations which allows us to initialise the output covariance $\boldsymbol{R}_i$. The system noise covariance $\boldsymbol{Q}_i$ can be, without any restriction, considered as a diagonal matrix and is simply initialised to the identity matrix. Values for $\boldsymbol{F}_i$ and $\boldsymbol{G}_i$ are then obtained by inverting the system.

For the non-linear case, it is crucial to initialise properly the centres and the widths of each radial basis function, as these parameters will not be learned during the training algorithm. We first perform the initialisation for a linear SSSM. For each segment of the data where a linear dynamical system has been initialised, a corresponding sequence of hidden continuous states $\boldsymbol{x}_t$ can be recovered by running the Kalman filter. A Gaussian Mixture Model is fitted to each sequence, which enables us to initialise the centres and the widths of each RBF network.

The parameters a_{ij} of the discrete transition matrix $\boldsymbol{A}$ can also be initialised by counting the number of transitions from state i to state j and dividing it by the number of transitions from state i to any other state.

We have noticed that such an initialisation procedure alleviates problems occurring during the E-step. The KL-divergence can have several local minima corresponding to different values of the variational parameters. This means that two significantly different segmentations can lead to a similar lower bound on the log-likelihood. Ghahramani and Hinton (1998) addressed this problem and modified the training algorithm by using the technique of *deterministic* annealing (Ueda and Nakano, 1995): the approximation distribution Q is broadened

[2]In practice, 5 iterations of the EM algorithm are sufficient.

with a *temperature* parameter that is annealed over time. However, with this method a large portion of training runs still converge to poor local minima.

In order to illustrate how our procedure can lead to a significant improvement, we consider the following synthetic problem involving a 2-state linear switching state space model:

$$x_t^{(1)} = 0.99x_{t-1}^{(1)} + u_1, \qquad u_1 \sim \mathcal{N}(0,1) \tag{6.22}$$

$$x_t^{(2)} = 0.90x_{t-1}^{(2)} + u_2, \qquad u_2 \sim \mathcal{N}(0,10) \tag{6.23}$$

The probability transition matrix $\boldsymbol{A}$ is such that $a_{11} = 0.99$ and $a_{22} = 0.98$. The output observation is identical for each model:

$$y_t = x_t^{(i)} + v, \qquad v \sim \mathcal{N}(0, 0.1) \qquad \forall i \tag{6.24}$$

We generate a sequence of $T = 1000$ points from this model and train linear SSSMs with the EM algorithm, considering three different learning techniques: our initialisation procedure, random initialisation without deterministic annealing and, random initialisation with deterministic annealing. For the deterministic annealing version, we follow Ueda and Nakano (1995): the variational parameters q and h are weighted by a decreasing temperature $\mathcal{T}$: starting with a relatively big value for $\mathcal{T}$, say $\mathcal{T} = 100$, the temperature is iteratively updated, $\mathcal{T}_i = \frac{1}{2}\mathcal{T}_{i-1} + \frac{1}{2}$, during the E-step. For each technique, 20 linear SSSMs corresponding to different random initial conditions were trained. We then evaluated the average mutual information between the true segmentation and the one obtained by each technique. Because the variational parameters h are real ($h_t^{(i)} \in [0,1]$), we first need to place a threshold on these values to obtain a *hard* segmentation[3].

Table 6.1 reports the results. Comparing the two random initialisations, on average, the deterministic annealing procedure performs slightly better. Our initialisation significantly outperforms both methods. We also report the average log-likelihood (lower bound) per data point for each technique. Compared to the likelihood obtained with the true model, each technique performs reasonably well. This shows the difficulty of comparing models when the exact computation of the likelihood is not tractable.

Figure 6.3 plots the time series and typical segmentations we obtain with the three approaches. Finding the true segmentation is actually very difficult. Even when the inference is performed with the true model, an underestimation of the switching can occur, leading to a segmentation where only one state space model is activated.

[3] $h_t^{(i)} = 1$ if $h_t \geq 0.5$, $h_t^{(i)} = 0$ otherwise.

Technique	Mutual Info	Log-likelihood
No annealing	0.42	−2.26
Annealing	0.49	−2.26
Initialisation	0.77	−2.21
True model	1.73	−2.17

Table 6.1: Average mutual information and log-likelihood (lower bound) per data point when training linear dynamical model with and without initialisation. For information, we report the results obtained with the true model: the entropy of the true segmentation is 1.73.

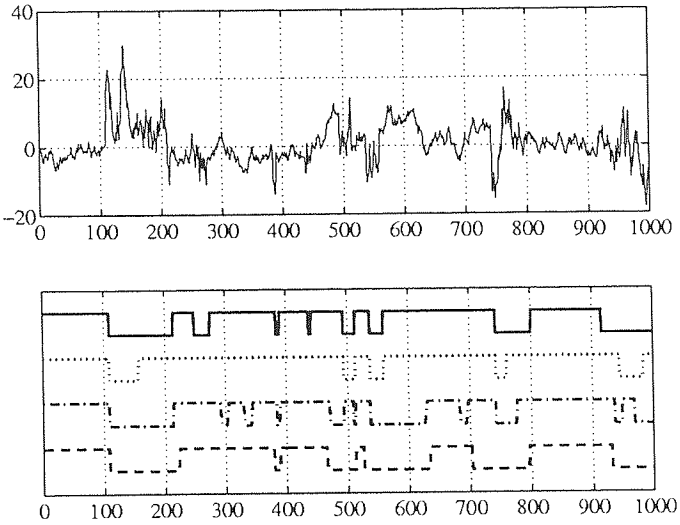


Figure 6.3: Synthetic time series (top) and segmentations (bottom) obtained with linear SSSMs compared to the true one (solid line): random initialisation without annealing (dotted line), random initialisation with annealing (dash dotted line) and initialisation (dashed line).

6.5 Predictions and on-line model selection

In this section we show how to make one-step ahead predictions with dynamical local models. The algorithm makes use of Bayes' theorem at each time step t and is known as the multiple model approach (Bar-Shalom and Li, 1993).

At each time step t, we note that each model contributes to the explanation of the observation $\boldsymbol{y}_t$ in the following way:

$$P(\boldsymbol{y}_t \mid s_t, \boldsymbol{x}_t^{(1)}, \ldots, \boldsymbol{x}_t^{(N)}) = \prod_{i=1}^{N} [P(\boldsymbol{y}_t \mid \boldsymbol{x}_t^{(i)})]^{s_t^{(i)}} \tag{6.25}$$

Unfortunately the value of the switching variable is not known in advance, but an expected value can be derived by using Bayes' theorem:

$$\mathrm{E}[S_t = q_i \mid \mathcal{Y}_1^t] = \frac{P(\boldsymbol{y}_t \mid \boldsymbol{y}_1^{t-1}, S_t = q_i) P(S_t = q_i \mid \boldsymbol{y}_1^{t-1})}{P(\boldsymbol{y}_t \mid \mathcal{Y}_1^{t-1})} \tag{6.26}$$

The first term in the numerator is given by Equation (6.4). The second term represents the predicted probability of model i at time t given all the earlier observations. As the discrete

state S_t is a first-order Markov process, this probability is given by:

$$\rho_t(i) \equiv P(S_t = q_i \mid \mathcal{Y}_1^{t-1}) = \sum_{j=1}^{N} a_{ji} P(S_{t-1} = q_j \mid \boldsymbol{y}_1^{t-1}) \tag{6.27}$$

The initial prior probabilities are assigned to be equal to $1/N$. The denominator is the normalising term (also known as the *evidence*) and is given by:

$$P(\boldsymbol{y}_t \mid \mathcal{Y}_1^{t-1}) = \sum_{i=1}^{N} \rho_t(i) P(\boldsymbol{y}_t \mid \mathcal{Y}_1^{t-1}, S_t = q_i) \tag{6.28}$$

Thus on-line estimations for each model decouple naturally. The Kalman filter recursive equations hold for each model i with the only modification that the likelihood of the observation $\boldsymbol{y}_t$ is weighted by $\rho_t(i)$.

Depending on the context, *hard* and *soft* competition can be implemented (Kadirkamanathan and Kadirkamanathan, 1996). In *hard* competition, it is believed that only one model is responsible for describing the observation at time t. This is done by considering only the model i with the highest predicted probability $\rho_t(i)$. In that case, $\rho_t(i) = 1$ and $\rho_t(j) = 0$ for the other models. In *soft* competition, $\rho_t(i) = P(S_t = q_i \mid \boldsymbol{y}_1^{t-1})$ and each model is allowed to adapt its parameters. This obviously leads to two different types of segmentations.

Thus the model inherits the properties from both HMMs and SSMs: the first-order Markov assumption for the discrete variable allows us to do on-line model selection. The state space model plays the role of the predictive model within each regime. As the mean and the covariance of the hidden states are updated on-line, the models allow us to obtain a full description of the predictive distribution.

6.6 Experimental results

We have assessed the performance of dynamical local models on different problems. We first show how linear switching state space models can be useful for modelling some time series relevant to the oil well drilling process. We then run simulations on synthetic data in order to evaluate and compare the performances of linear and non-linear local dynamical models on data which exhibit local non-linearity. We finally show promising results of both models for modelling financial time series.

6.6.1 Drilling data

Figure 6.4 (top) plots the low gravity solids time series (LGS) over a period of two and a half hours (350 points). This time series corresponds to normal drilling conditions. The data

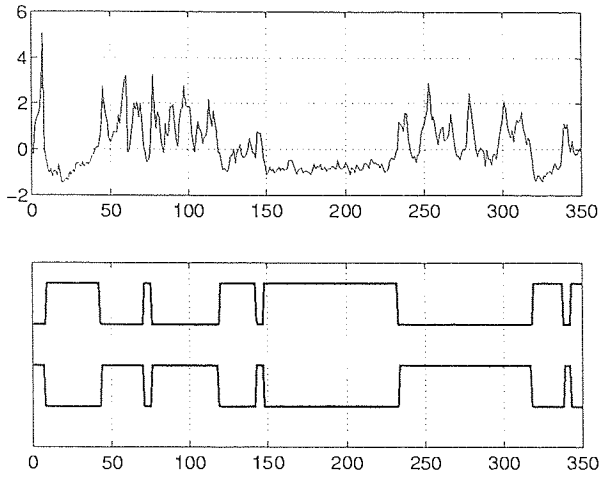


Figure 6.4: Low gravity solids time series (top) corresponding to typical normal conditions (the data have been normalised to zero mean and unit covariance). Below, segmentation performed by the variational inference.

exhibits interesting time scale effects such as a variability in the amplitude and the mean. Also note the noise signal. We propose to model such a time series with a linear switching model. By tracking the trends in the volumes of drilled solids, we might obtain a better picture of downhole conditions. For example, if relationships between the regimes found by the model and events in drilling can be found, a specification of what constitutes a clean hole could be defined. In a simplistic view, each regime could represent the type of formation currently drilled. Within each regime, the state space model could represent the physical dynamics of the drilling process.

In order to illustrate how dynamical local models can be used for monitoring the drilling process, we present here some interesting results obtained on this small dataset. As mentioned previously (Chapter 2), data collection is a difficult task and we do not possess a reasonable number of data points for assessing properly the performances of our approach. The time windows are very small, representing no more than 500 non-continuous valid data points over one day.

We trained different linear SSSMs, varying the dimension of the hidden state space and the number of local models. We present in this section results obtained with a model containing $N = 2$ state space models, each being of dimension $m = 2$. Figure 6.4 (bottom) plots a typical segmentation obtained with variational inference on the training set. The picture shows how each local model is specialised: one local models seems to take care of low variability and the second tracks high variability. We have noticed that an increase of the number of hidden states leads to a similar segmentation. For example, an added state is only activated when a

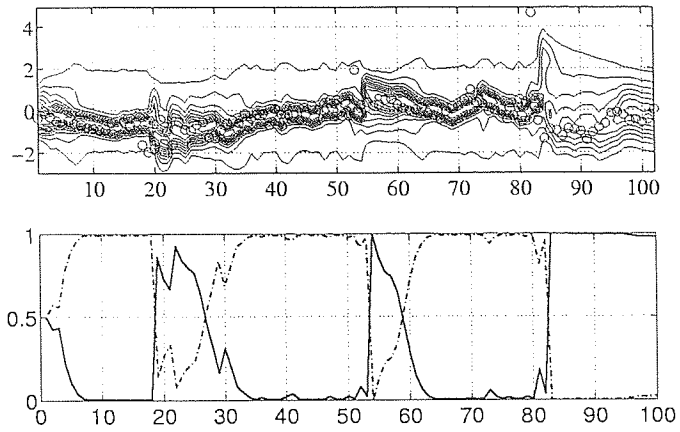


Figure 6.5: Contour plot (top) of the predictive distribution $P(y_t \mid Y_{t-1})$ and segmentation of low gravity solids (bottom) with linear switching state space models. Note how the the model switches and how the confidence is affected by these transitions. Note also sections where soft competition takes place. The targets are represented by circles in the contour plot.

transition occurs between the two other states.

One of the major advantages of dynamical local models is their capability of providing a full predictive distribution. Whereas autoregressive hidden Markov models assume a local constant variance, dynamical local models provide a time dependent probability density. Figure 6.5 plots the contour plot of the predictive distribution for a small time window of 100 points. Soft competition was used and the picture shows how the distribution varies with time. Each regime is associated to a Gaussian with a specific variance. It can be seen that the model switches from regions of high predictive probability $P(y_t \mid Y_{t-1})$ to low probability. This is particularly significant in the last section (last 20 points of this window) where the first state is being activated: the contour plot is sparser, reflecting a low confidence in the prediction compared to sections where the second state (corresponding to a smaller variance) is responsible for predicting the output.

6.6.2 Synthetic data

We generated data from a bimodal process (Weigend *et al.*, 1995):

$$y_{t+1} = \begin{cases} 2(1 - y_t^2) - 1 & \text{if } s_t = 0, \\ \tanh(-1.2 y_t + \epsilon) & \text{if } s_t = 1. \end{cases} \tag{6.29}$$

where $\epsilon \sim \mathcal{N}(0, 0.1)$. The first mode is a deterministic chaotic process whereas the second mode is a noisy non-chaotic process. The switching obeys a first order Markov process with diagonal entries $a_{ii} = 0.98$. Both training and test datasets contain 500 points.

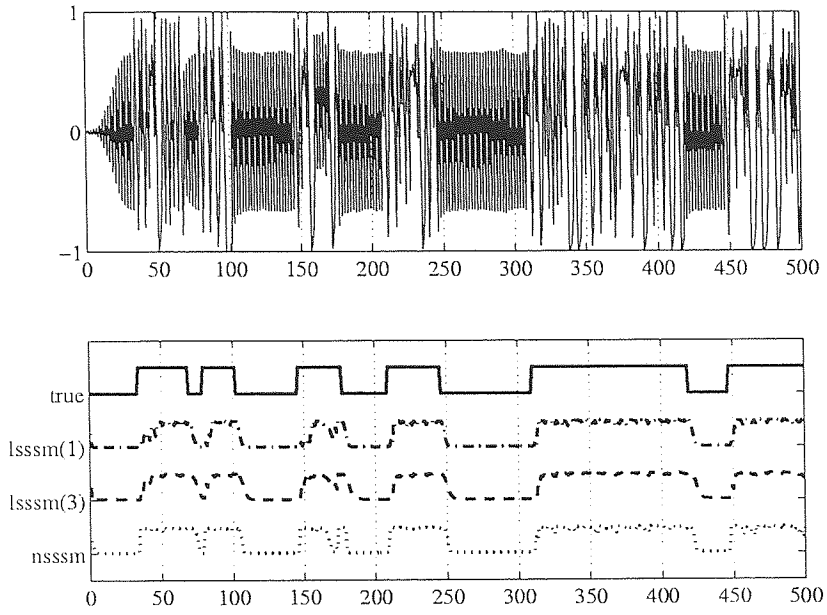


Figure 6.6: Test data and model probabilities for true (solid), linear SSSM of dimension 1 (dash dotted), linear SSSM of dimension 3 (dashed) and non-linear SSSM (dotted).

In order to obtain a valid comparison between linear and non-linear SSSMs, we trained two linear SSSMs of different hidden state dimension and a non-linear SSSM on this data. The dimensions of the hidden states $\boldsymbol{x}_t^{(i)}$ for the linear models have been taken to be $m = 1$ and $m = 3$. The dimension of the hidden state for the non-linear model is $m = 1$ and a RBF network with $K = 3$ hidden units has been used.

Figure 6.6 plots the test dataset and the corresponding segmentations obtained by the three models. Compared to the true segmentation, we can see that all three models capture the underlying regime well, but that the linear SSSM of dimension 3 and the non-linear SSSM are slightly more successful than the simple linear SSSM. Indeed, the correlations between the true segmentation and the ones obtained by the two linear SSSMs are respectively 0.78 and 0.81. The non-linear SSSMs outperform the linear models and gives rise to a correlation of 0.85.

Figure 6.7 plots the accuracy of the linear SSSM of hidden state dimension 3 and the non-linear SSSM under the deterministic chaotic regime. Although the linear SSSM is able to capture the non-linearity, the non-linear SSSM seems to be more accurate[4]. This is particularly significant in the central region where there is a perfect match between the true underlying function and the output of the non-linear model.

[4]This is more obvious in the next table which reports the log-likelihood and the normalised mean squared error on the test dataset.

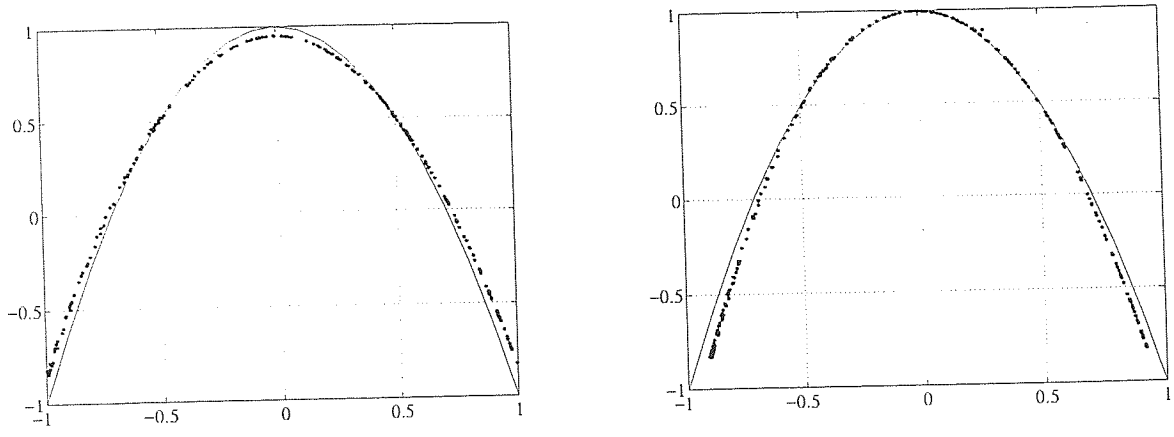


Figure 6.7: Accuracy of the linear (a) and the non-linear (b) SSSMs in the chaotic regime. The solid line is the true function.

We have also trained linear and non-linear dynamical systems on this dataset and we end this section by comparing linear and non-linear SSSMs with these single mode systems. The hidden state dimension of the linear models and the number of RBF units for non-linear models have been taken to be 3. Table 6.6.2 reports the log-likelihood per datum and the normalised mean squared error (NMSE) on the test set and shows the significant improvement of the switching models. For each model, we report the average and the spread over 10 different initial conditions. It is interesting to note that an LDS of hidden state dimension 3 does not outperform the simple LDS with an hidden state of dimension 1. This remark does not apply to linear switching state space models: a linear SSSM of hidden state dimension 1 gives rise on average to a likelihood of -0.60 and a NMSE of 0.025.

Model	Log-likelihood		NMSE	
	mean	std	mean	std
LDS	-0.8601	0.0001	0.0339	0.0001
NLDS	-0.8020	0.0040	0.0292	0.0003
LSSSM	-0.5667	0.0107	0.0228	0.0004
NLSSSM	-0.4523	0.0221	0.0183	0.0013

Table 6.2: Average log-likelihood and NMSE on the test set for a simple linear dynamical system (LDS), a non-linear dynamical system (NLDS), a 2-state linear SSSM ($m = 3$) and a 2-state non-linear SSSM.

6.6.3 Financial data

Because of the capability of state space models for tracking quasi-stationarity and the power of HMMs for uncovering the hidden switching between regimes, we investigate their performance on financial data. An advantage of viewing the model in a probabilistic framework is that we can also attach confidence intervals to the predictions, as the covariance matrix of the random variable $\boldsymbol{X}_t$ is also estimated at each time step t. One immediate and important application in financial engineering is risk estimation. In addition, the value of the discrete hidden variable S_t can be viewed as indicating the regime that the market is in at time t: this gives us a segmentation of the data, which is of value in its own right.

We present here results of our simulations on DEM/USD and GBP/USD foreign exchange rate daily returns:

$$r_t = \log p_t - \log p_{t-1} \approx \frac{p_t - p_{t-1}}{p_{t-1}} \tag{6.30}$$

where p_t is the closing daily exchange rate at time t. This quantity can be seen as the logarithm of the geometric growths and is known in finance as continuous compounded returns.

Figure 6.8 plots the datasets. The DEM/USD training set contains 3000 points from 29/09/1977 to 15/09/1989. The test set contains 1164 points from 16/09/1989 to 05/11/1994. The GBP/USD training set contains 2000 points from 01/06/73 to 29/01/81 and the test set contains 1164 from 30/01/81 to 21/05/87.

The first application of the model is to uncover underlying regimes. As an example, Figure 6.9 plots the segmentation obtained on the DEM/USD test set with a simple 3-state non-linear SSSM ($N = 2$). The dimension of each state space has simply been taken to $m = 1$ and the number of radial basis functions is $K = 5$. The figure shows how the model is capable of detecting abrupt changes in the time series. It is however difficult for us to give an interpretation of such a segmentation. The knowledge of an expert would be of great help. In a simplistic view, we could imagine that the underlying regimes are related to some macro-economical variables.

Another important application of dynamical local models in finance is the possibility of obtaining on-line estimates of the covariance of our prediction. Figure 6.10a shows a time window of 60 points where a regime transition occurred at time $t = 35$. Figure 6.10b is the corresponding contour plot. The model moves progressively from a high volatility region to a relatively low volatility region and the predictions are affected by this change. We clearly see how the predictive distribution $P(\boldsymbol{y}_t|\mathcal{Y}_{t-1})$ is sensitive to this transition.

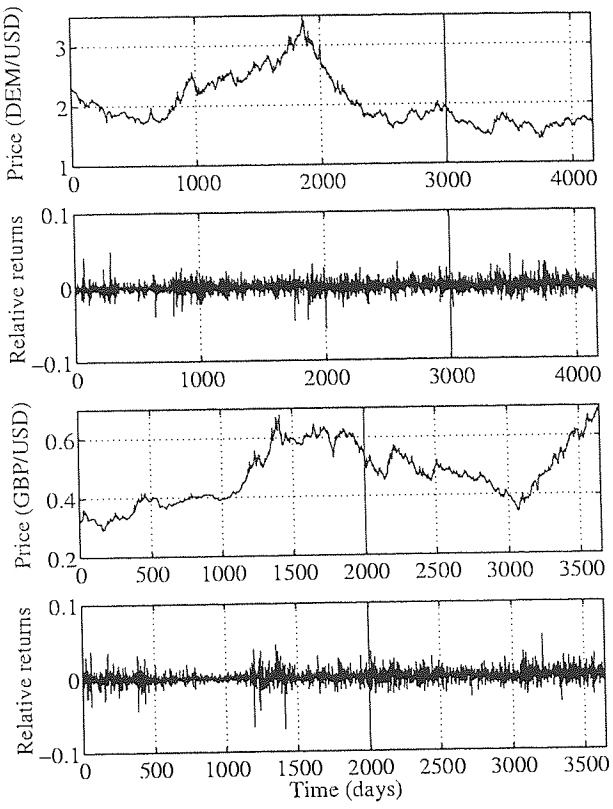


Figure 6.8: DEM/USD and GBP/USD training and test datasets used for evaluating dynamical local models.

Figure 6.10c shows the contour plot obtained by running a Kalman filter with adaptive state noise covariance matrix. Rather then learning the state noise covariance matrix $\boldsymbol{Q}_t$ from the data (see Chapter 3) and keep it fixed on unseen data, it is possible to estimate it on-line. This technique is due to (Jazwinski, 1969) and is useful for detecting transitions from a stationary regime to another. Assuming a diagonal state noise covariance matrix, $\boldsymbol{Q}_t = q_t I$, an update of the coefficients q_t can take the following form:

$$q_t = \alpha q_{t-1} + (1-\alpha) h\left(\frac{e_t^2 - w_0^2}{\boldsymbol{F}\boldsymbol{F}'}\right) \tag{6.31}$$

where α is a parameter which controls the smoothness of the update, $e_t = y_t - y_{t|t-1}$ is the prediction error and $w_0 = R + \boldsymbol{F}\boldsymbol{P}_{t-1|t-1}\boldsymbol{F}'$ is the estimated prediction variance. This technique relies heavily on the α parameter[5]. Figure 6.10c shows how the predictive distribution is affected by such an on-line update of $\boldsymbol{Q}_t$ (in this simulation, we used $\alpha = 0.1$). It is interesting to note the similarity of both techniques: the distribution becomes more sharply peaked after $t = 35$ and confirms the regime transition obtained by our dynamical local models. Note however that the predictive distribution obtained with switching state

[5]More details on this technique can be found in (Penny and Roberts, 1999).

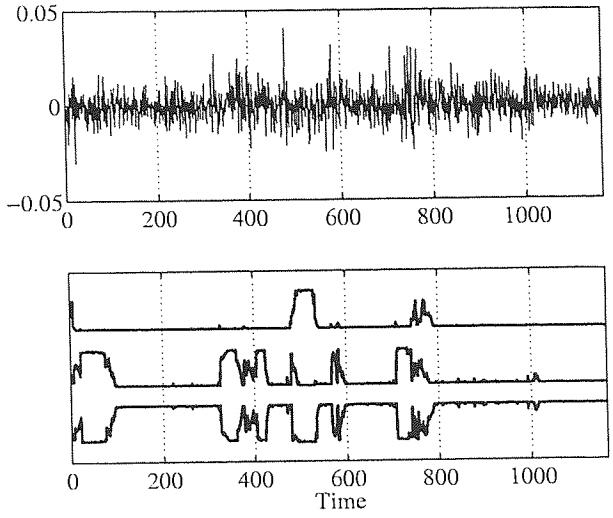


Figure 6.9: Predictive model probabilities $P(s_t \mid \mathcal{Y}_1^{t-1})$ obtained by a non-linear dynamical local model on the DEM/USD test set.

space models is smoother.

We end this section by evaluating the performance of dynamical local models using objective measures and compared them with other models. We trained autoregressive models (AR), GARCH models, MLP neural networks (NN) and autoregressive hidden Markov models (ARHMM) on the same data sets. A GARCH model (Bollerslev, 1986) consists of a linear AR model for the conditional mean and an exponential AR model for the conditional variance. They are very often used in finance engineering for modelling quasi-stationarity. For AR, NN and ARHMM models, the input dimension has been simply taken to be 5 lagged values of the observations (which represent the history of the previous week), although no careful analysis of the input dimension has been carried out. Similarly, the neural network contains 10 hidden non-linear nodes and the ARHMM contains, like our models, 3 hidden states.

We have computed the log-likelihood per datum and the normal mean squared error (NMSE). For each model, we report the average and the spread over 10 different initial conditions. Dynamical local models have been initialised by the procedure we presented in Section 6.4.

Table 6.3 reports the results. On average, the NLSSSM seems to be the best model to describe the data, as the likelihood suggests it. When comparing the NMSE, we see that none of these models seem to outperform the naive prediction, which would consist of making predictions based on the mean of the training set. Note, for example, that the log-likelihood for such a naive model is equal to -1.1575 on the DEM/USD dataset.

DEM/USD

Model	Log-likelihood		NMSE	
	mean	std	mean	std
AR	−2.3957	—	1.0002	—
GARCH	−1.1488	—	1.0000	—
NN	−1.1950	0.0149	1.0190	0.0094
ARHMM	−1.0456	0.0020	0.9998	0.0000
LDS	−1.1574	0.0000	0.9997	0.0000
NLDS	−1.1366	0.0030	0.9997	0.0001
LSSSM	−1.1045	0.0154	0.9995	0.0004
NLSSSM	−1.0361	0.0111	0.9995	0.0003

GBP/USD

Model	likelihood		NMSE	
	mean	std	mean	std
AR	−2.5268	—	1.0020	—
GARCH	−1.2174	—	0.9994	—
NN	−1.2191	0.0316	1.0720	0.0188
ARHMM	−1.0730	0.0000	1.0030	0.0000
LDS	−1.2500	0.0000	0.9999	0.0000
NLDS	−1.2214	0.0020	0.9999	0.0001
LSSSM	−1.1362	0.0283	0.9996	0.0002
NLSSSM	−1.0581	0.0121	0.9996	0.0002

Table 6.3: Average log-likelihood and normalised mean squared errors on the DEM/USD and GBP/USD test sets over 10 runs corresponding to different initial conditions.

These simulations were intended to compare dynamical local models with other standard techniques used in computational finance and confirm the fact that predicting the daily return is a very difficult task. A better understanding of financial markets could be obtained by considering high frequency data. For example, Shi and Weigend (1997) modelled high frequency foreign exchange data with autoregressive hidden Markov models and showed promising results.

6.7 Discussion

In this chapter we have reviewed hybrid models that combine hidden Markov models and state space models. These models have emerged from different scientific communities because of the necessity of modelling processes where the assumption of global stationarity does not hold.

We reviewed linear switching state space models and proposed a new extension which incorporates local non-linearity. This is done by using a local RBF network which maps the

hidden state space to the observation space[6]. The structured variational approach allows us to perform a principled approximate maximum likelihood estimation of the parameters. The inference decouples nicely into the inference algorithms for HMMs and SSMs. In the case of non-linear dynamical models, a linearisation of the local function leads to the extended Kalman filter.

We also proposed an efficient and fast initialisation algorithm which alleviates problems of multiple local minima during the variational inference. This procedure leads to a significant improvement in the reliability of training compared to the deterministic version.

In contrast to other hybrid models such as mixture of experts or autoregressive HMMs, dynamic local models provide a full description of the predictive distribution. This is an important issue, especially in finance where robust error bars need to be developed.

We evaluated the performance of the models on different data sets and compared them to other standard techniques. This was done by evaluating the log-likelihood per datum over a test set, as this measure allows direct comparisons between different models. Another evaluation of the density forecasts, based on the cumulative probability distribution, could complement our comparisons. This technique was proposed by Diebold *et al.* (1998) and consists of estimating the following random variable:

$$Z_{t+1} = \int_{-\infty}^{y^{t+1}} P(\eta \mid \mathcal{Y}_1^t)\, d\eta. \tag{6.32}$$

In order to assess the quality of the prediction, the random variable is tested against the hypothesis of a uniform distribution, which would correspond to a good model for the true predictive distribution $P^*(y_{t+1} \mid \mathcal{Y}_1^t)$. To test whether Z is uniformly distributed, Diebold *et al.* (1998) points out standard techniques. The simplest one consists of plotting the histogram.

The variational inference approach maximises a lower bound on the log-likelihood. An interesting problem concerns the quality of this bound which is a current open question. Empirical simulations could be done to evaluate this quality. This could be done by considering a dynamical local model containing a relatively small number of state space models, say $N = 2$ for example and a short time series. In that case, the exact estimation of the true posterior distribution of the hidden states can be performed and compared to the variational approximation.

Another comparison could be done by considering Monte Carlo integration techniques,

[6]It must be emphasized that a Radial Basis Function network can be hardly seen as a 'true' generative model.

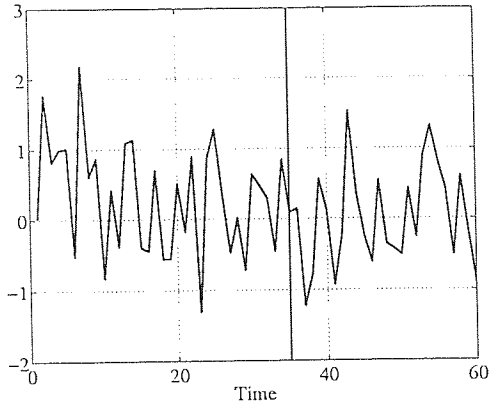
such as Gibbs sampling, which provide a more accurate representation of the true posterior. This would also help us to evaluate the performance of the extended Kalman filter for highly local non-linear dynamics.

Obviously, our models can be extended into several directions. In our work we did not consider exogenous variables as only a single time series is modelled. An immediate and straightforward extension consists of considering previous values of the time series as inputs in the dynamics of the hidden states:

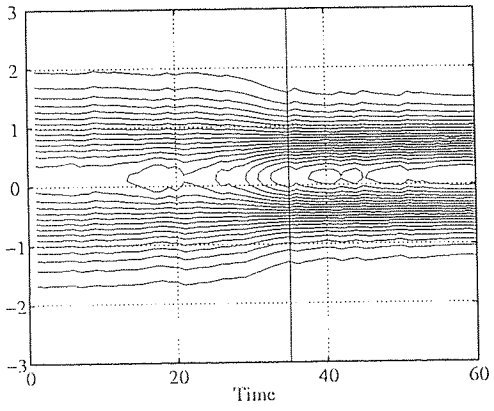
$$\boldsymbol{x}_t^{(i)} = \boldsymbol{F}_i \boldsymbol{x}_{t-1}^{(i)} + \boldsymbol{H}_i \boldsymbol{y}_{t-q}^{t-1} + \boldsymbol{u}_i, \tag{6.33}$$

where the vector $\boldsymbol{y}_{t-q}^{t-1} = [y_{t-q}, \ldots, y_{t-1}]$ contains, for example, the last $q-1$ observations. We also did not consider non-linearities for the system equation. This is also an immediate extension of the non-linear dynamical local models, although we believe that the resulting algorithm would be too cumbersome for practical application.

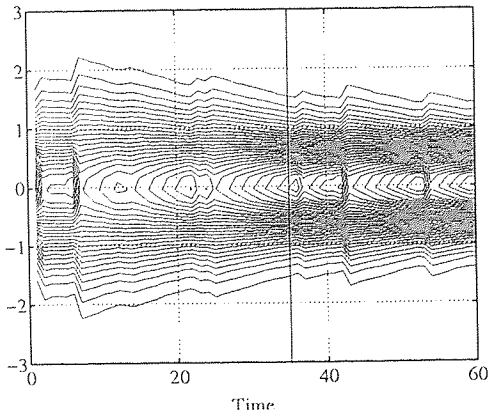
In this work, we assumed local stationarity in the variances. This was done by assuming that both output covariance and state noise covariance matrices are locally constant, which allows us to estimate these parameters through a maximum likelihood approach by segmenting the data. An alternative procedure for detecting non-stationarity is to use, as we mentioned it in Section 6.6, a single model with adaptive output and state noise covariances. Penny and Roberts (1999) mentioned the problems that may arise from such an approach, as an on-line update of both covariance matrices may be redundant for explaining the error in the predictions. The authors suggest a on-line estimate of either the state noise covariance or the output covariance, the other being learned from the data. These techniques rely on some parameters that control the smoothness of the on-line covariance estimates and the user has to specify them in advance. Our models take into account the non-stationarity in the covariances by specifying the regimes and allocating a model to each regime. The segmentation on unseen data arises naturally and does not require the user to specify threshold values for non-stationarity detection. Moreover, as we saw in our simulations, the predictive distribution obtained by dynamical local models is smoother and is not sensitive to some irrelevant abrupt changes.



(a)



(b)



(c)

Figure 6.10: Contour plot of the predictive distribution $P(y_t|\mathcal{Y}_{t-1})$ for a dynamical local model (b) and a Kalman filter with adaptive noise variance (c).

Chapter 7

Conclusions

From econometrics to engineering, time series analysis and forecasting is an important issue since many problems are concerned with predicting the value of a variable in the future. Probably, the most important objective in the study of time series is to uncover the dynamic law of its generation. When the underlying dynamics are not available, time series modelling consists of developing a model which best explains a set of data points. This thesis tried to give a unified view of hidden state models for time series. We presented several probabilistic models and algorithms and demonstrated their application on several synthetic and real-world datasets.

In Chapter 3, we presented hidden Markov models and state space models and showed how these probabilistic models can be viewed in the same framework. Both models assume explicitly the existence of *hidden state* variables which describe the underlying dynamics. The inference problem consists of uncovering the underlying dynamics and we showed how the Markov property allows us to treat these two models in a unified way. The parameter estimation problem has been also presented within a maximum likelihood framework and we mentioned how a Bayesian treatment of Markov models could be carried out using the ensemble learning technique. This represents a promising field of research especially when the data is relatively sparse compared to the number of parameters in the model. For example, an important application of the Bayesian framework for hidden Markov models would be to determine automatically which of the hidden states are relevant.

Time delay estimation is an important part of time series analysis. In Chapter 4, we suggested the use of hidden Markov models for identifying relationships between two time series. We derived two algorithms; one based on a maximum likelihood approach, the second on a maximum mutual information framework. Both techniques were successfully applied on

real-world data and we demonstrated their usefulness by estimating a crucial parameter for the oil industry. The mutual information approach seems to be more accurate while more computationally expensive and sensitive to initial conditions. Although the latter methodology is similar to what has been used previously for discriminant learning in the speech community, to our knowledge, this is the first time that hidden Markov models have been used for identifying relationships between two time series in a non-stationary environment within an information-theoretic approach. A further step in this work should carry out the on-line estimation of the delay when dealing with sequential data. This is an important issue for several monitoring applications, like the oil industry, where the goal is to prevent a major problem.

Variational techniques for probabilistic inference are new tools and we tried to demonstrate their application for modelling time series within a probabilistic framework. Variational techniques can be regarded as a principled and more controlled approximation than that provided by other solutions. One drawback is their computational cost.

Perhaps the key issue is the accuracy of the approximation. Although there are some well-studied cases where the properties of the variational approximation can be examined, there are many other graphical models for which such an evaluation is not possible.

In Chapter 6, we concentrated on the issue of modelling time series that exhibit a quasi-stationary behaviour. We reviewed several hybrid models that have been suggested in different communities and focussed the parameter estimation problem of switching state space models. We showed how variational techniques allow us to perform a principled maximum likelihood estimation which leads to a rigorous lower bound on the likelihood of the data given the model. For problems in which we have a prior belief that the underlying generator switches between stationary regime, dynamical local models are a natural tool.

In order to model non-linearity, we suggested the use of a dynamical model which combines a hidden Markov model and a non-linear dynamical system. We modelled the non-linearity with a radial basis function network and made use of the extended Kalman filter for the inference problem. Monte Carlo simulations should be however carried out for assessing the accuracy of this approximation.

We also proposed an efficient initialisation algorithm in order to alleviate problems of multiple local minima during the variational inference. This initialisation procedure was tested on synthetic data and we demonstrated a significant improvement in the variational inference compared to the deterministic annealing technique.

The models were tested on both synthetic and real-word data and we showed promising results compared to other standard techniques. More simulations should be carried out in order to compare non-linear dynamical model with other non-linear models such as Markovian mixture of experts. We focussed on the difficult task of forecasting financial time series and showed how interesting features such as segmentation and error bars for prediction are embedded within the framework. These two features are the major advantages of dynamical local models and can be of great help in financial applications.

Further works concern time delay estimation with hidden Markov models using other output distributions than the Gaussian distribution[1]. By making use of variational techniques, it is indeed possible to model other distributions. For example, the maximum mutual information approach, as it stands, cannot be applied for a mixture of Gaussians. In (Lawrence and Azzouzi, 2000), the authors show how to make use of a mixture of Gaussians for a Bayesian treatment of neural networks. A similar framework can be derived for hidden Markov models.

Another related promising field of research concerns the application of variational techniques for non-Gaussian switching state space models. This represents an interesting direction for future research and an exciting application in finance where the Gaussian assumption is far from being valid.

[1] I am grateful to Stephen J. Roberts for suggesting me this idea.

References

Anderson, B. D. O. and J. B. Moore (1979). *Optimal Filtering.* Prentice-Hall, Englewood Cliffs, NJ.

Bahl, L. R., F. Jelinek, and R. L. Mercer (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **5** (2), 179–190.

Baldi, P., Y. Chauvin, T. Hunkapiller, and M. A. McClure (1993). Hidden Markov models in molecular biology: New algorithms and application. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 5, pp. 11–18. San Mateo: Morgan Kaufmann.

Bar-Shalom, Y. and T. E. Fortmann (1988). *Tracking and Data Association*, Volume 179 of *Mathematics in Science and Engineering.* Academic Press, Inc.

Bar-Shalom, Y. and X. R. Li (1993). *Estimation and Tracking.* Artech House, Boston, MA.

Baum, L., T. Petrie, G. Soules, and N. Weiss (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics* **41**, 164–171.

Bengio, Y. and P. Frasconi (1995). An input-output HMM architecture. In G. Tesauro, D. S. Touretsky, and T. K. Leen (Eds.), *Advances in Neural Information Processing Systems*, Volume 7, pp. 427–234. Cambridge, MA: MIT Press.

Blom, H. A. P. and Y. Bar-Shalom (1988). The interactive multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transaction on Automatic Control* **33** (8), 780–783.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics* **31**, 307–327.

Bridle, J. S. (1989). Training stochastic model recognition algorithms as networks can

lead to maximum mutual information estimation of parameters. In D. S. Touretsky (Ed.), *Advances in Neural Information Processing Systems*, Volume 2, pp. 211–217. San Mateo: Morgan Kaufmann.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fogelman and J. Herault (Eds.), *Neurocomputing: Algorithms, Architectures and Applications*, pp. 227–236. Springer-Verlag.

Broomhead, D. S. and G. P. King (1986). Extracting qualitative dynamics from experimental data. *Physica* **20D**, 217–236.

Brown, P. F. (1987). *The Acoustic-Modeling Problem in Automatic Speech Recognition.* Ph.D. thesis, IBM T J Watson Research Center.

Cacciatore, T. W. and S. J. Nowlan (1994). Mixtures of controllers for jump linear and non-linear plants. In J. D. Cowan, G. Tesauro, and J. Alspector (Eds.), *Advances in Neural Information Processing System*, Volume 6, pp. 719–726. San Francisco: Morgan Kaufmann.

Chang, C. B. and M. Athans (1977). State estimation for discrete systems with switching parameters. *IEEE Transactions on Aerospace and Electronic Systems* **14** (2), 418–424.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* **39** (1), 1–38.

Diebold, F. X., T. A. Gunther, and A. S. Tay (1998). Evaluating density forecasts, with evaluation to risk management. *International Economic Review*.

Fraser, A. M. and A. Dimitriadis (1994). Forecasting probability densities by using hidden Markov models with mixed states. In A. S. Weigend and N. A. Gershenfeld (Eds.), *Time Series Prediction: Forecasting the Future and Understanding the Past*, pp. 264–281. Addison-Wesley.

Ghahramani, Z. (1997). On structured variational approximations. Technical Report CRG-TR-97-1, Department of Computer Science, University of Toronto.

Ghahramani, Z. and G. E. Hinton (1996). Parameter estimation for linear dynamical systems. Technical report, Department of Computer Science, University of Toronto.

Ghahramani, Z. and G. E. Hinton (1998). Switching state-space models. Technical report, Department of Computer Science, University of Toronto.

Goodrich, R. L. and P. E. Caines (1979), June. Linear system identification from nonstationary cross-sectional data. *IEEE Transactions on Automatic Control* **24** (3), 403–411.

Gordon, N. J., D. J. Salmon, and A. F. M. Smith (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *Proceedings of the IEEE*, Volume 140, pp. 107–113.

Grimmett, G. R. and D. R. Stirzaker (1982). *Probability and random processes*. Clarendon Press, Oxford.

Guild, G. J., I. M. Wallace, and M. J. Wassenborg (1995), February. Hole cleaning program for extended reach wells. In *Proceedings of the SPE/IADC Drilling Conference*, pp. 425–433. SPE/IADC 29381.

Gupta, N. K. and R. K. Mehra (1974), December. Computational aspects of maximum likelihood estimation and reduction in sensitivity function calculations. *IEEE Transactions on Automatic Control* **19** (6), 774–782.

Hamilton, J. D. (1989). A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica* **57**, 357–384.

Heckerman, D. (95). A tutorial on learning with Bayesian networks. Technical report, Mircrosoft Research.

Hinton, G. E. and D. Van Camp (1993). Keeping neural networks simple by minimizing the description length of the weight. In *Proceedings of the 6th Annual Workshop on Computational Learning Theory*, pp. 5–13. ACM Press.

Jacobs, R. A., M. I. Jordan, S. J. Nowlan, and G. E. Hinton (1991). Adaptive mixture of experts. *Neural Computation* **3**, 79–87.

Jazwinski, A. H. (1969). Adaptive filtering. *Automatica* **5**, 475–485.

Jordan, M. I., Z. Ghahramani, T. S. Jaakkola, and L. K. Saul (1998). *Learning in Graphical Models*, Chapter An introduction to variational methods for graphical models. MIT Press.

Kadirkamanathan, V. and M. Kadirkamanathan (1996). Recursive estimation of dynamic modular RBF networks. In G. Tesauro, D. S. Touretsky, and T. K. Leen (Eds.), *Advances in Neural Information Processing Systems*, Volume 8, pp. 239–245. MIT Press.

Kalman, R. E. and R. S. Bucy (1961). New results in linear filtering and prediction. *Journal of Basic Engineering (ASME)* **83D**, 95–108.

Kendall, M. and J. K. Ord (1990). *Time Series*. Edward Arnold.

Kenny, P., E. Sunde, and T. Hemphill (1996), March. What's 'n' got to do with it? In *Proceedings of the SPE/IADC Drilling Conference*. SPE/IADC 35099.

Kitagawa, G. (1987), December. Non-Gaussian state-space modeling of nonstationary time series. *Journal of the American Statistical Association* **82** (400), 1032–1063.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics* **5**, 1–25.

Lawrence, N. D. and M. Azzouzi (2000). A variational Bayesian committee of neural networks. *Neural Networks*. To appear.

Lowe, D. and N. Hazarika (1997), July. Complexity modelling and stability characterisation for long term iterated time series prediction. In *Proceedings of the fifth International IEE Conference on Artificial Neural Networks*, Volume 440, pp. 53–58.

Lowe, D. and A. R. Webb (1994). Time series prediction by adaptive networks: a dynamical systems perspective. In *Artificial Neural Networks: Forecasting time series*, pp. 12–19. V R Vemuri and R D Rogers, IEEE Computer Society Press.

MacKay, D. J. C. (1995). Developments in probabilistic modelling with neural networks - ensemble learning. In *Neural Networks: Artificial Intelligence and Industrial Applications. Proceedings of the 3rd Annual Symposium on Neural Networks, Nijmegen, Netherlands, 14-15 September 1995*, pp. 191–198. Berlin: Springer.

MacKay, D. J. C. (1998). Ensemble learning for hidden Markov models.

Mazor, E., A. Averbush, Y. Bar-Shalom, and J. Dayan (1998), January. Interacting multiple model methods in target tracking: a survey. *IEEE Transactions on Aerospace and Electronic Systems* **38** (1), 103–123.

Neal, R. M. (1996). *Bayesian learning for neural networks*. New York: Springer-Verlag.

Neal, R. M. and G. E. Hinton (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Kluwer Academic Press.

Packard, N. H., J. P. Crutchfield, J. D. Farmer, and R. S. Shaw (1980). Geometry from a time series. *Physical Review Letters* **45**, 712–716.

Parisi, G. (1988). *Statistical Field Theory*. Redwood City, CA: Addison-Wesley.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.

Penny, W. D. and S. J. Roberts (1999). Dynamic models for nonstationary signal segmentation. *Computers and Biomedical Research* **32** (6), 483–502.

Peterson, C. and J. R. Anderson (1987). A mean field theory learning algorithm for neural networks. *Complex Systems* **1**, 995–1019.

Pilehvari, A. A., J. J. Azar, and S. A. Shrazi (1996). State-of-the-art cuttings transport in horizontal wellbores. In *Proceedings of the SPE International Conference on Horizontal Well Technology*, pp. 389–393. SPE 37079.

Pole, A., M. West, and J. Harrison (1994). *Applied Bayesian forecasting and Time Series Analysis*. Chapman & Hall.

Poritz, A. B. (1982), May. Linear predictive hidden Markov models and the speech signal. In *Proceedings of ICASSP*, pp. 1291–1294.

Press, W., S. Teukolsky, W. Vetterling, and B. Flannery (1992). *Numerical Recipes in C. The Art of Scientific Computing* (second ed.). Cambridge University Press.

Rabia, H. (1985). *OilWell Drilling Engineering: Principles and Practice*. Graham & Trotman.

Rabiner, L. R. (1989), February. A tutorial on hidden Markov models and selected application in speech recognition. In *Proceedings of the IEEE*, Volume 77-2, pp. 257–286.

Rabiner, L. R., B. H. Juang, S. E. Levinson, and M. M. Sondhi (1985), July-August. Some properties of continuous hidden Markov model representation. *AT&T Techincal Journal* **64** (6), 1251–1270.

Rasi, M. (1994), February. Hole cleaning in large, high-angle wellbores. In *Proceedings of the IADC/SPE Drilling Conference*, pp. 299–310. IADC/SPE 27494.

Rauch, H. E. (1963). Solutions to the linear smoothing problem. *IEEE Transactions on Automatic Control* **8**, 371–372.

Rigney, D. R., A. L. Goldberger, W. C. Ocasio, Y. Ichimaru, G. B. Moody, and R. G. Mark (1994). Multi-channel physiological data: description and analysis. In A. S. Weigend and N. A. Gershenfeld (Eds.), *Time Series Prediction: Forecasting the Future and Understanding the Past*, pp. 105–129. Reading, MA: Addison-Wesley.

Saul, L. K., T. S. Jaakkola, and M. I. Jordan (1996). Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research* **4**, 61–76.

Saul, L. K. and M. I. Jordan (1996). Exploiting tractable substructures in intractable networks. In D. S. Touretsky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems*, Volume 8, pp. 486–492. Cambridge, MA: MIT Press.

Shi, S. and A. S. Weigend (1997), March. Taking time seriously: hidden Markov experts applied to financial engineering. In *Proceedings of the IEEE/IAFE Conference on Computational Intelligenge for Financial Engineering.*

Shumway, R. H. and D. S. Stoffer (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis* **3** (4), 253–264.

Shumway, R. H. and D. S. Stoffer (1991). Dynamic linear models with switching. *Journal of the American Statistical Association* **86**, 763–769.

Shwe, M. A., B. Middleton, D. E. Heckerman, M. Henrion, E. J. Horvitz, and H. P. Lehmann (1991). Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. *Meth. Inform. Med.* **9**, 241–255.

Sifferman, T. R. and T. E. Becker (1992), June. Hole cleaning in full-scale inclined wellbores. *SPEDE*, 115–120.

Smyth, P. (1994). Hidden Markov models for fault detection in dynamic systems. *Pattern Recognition* **27** (1), 149–164.

Smyth, P., D. Heckerman, and M. I. Jordan (1997). Probabilistic independence networks for hidden Markov probability models. *Neural Computation* **9**, 227–269.

Takens, F. (1981). Detecting strange attractors in turbulence. In D. A. Rand and L. S. Young (Eds.), *Dynamical Systems and Turbulence (Warwick 1980) (Lecture Notes in Mathematics)*, Volume 898, pp. 366–381. Berlin: Springer.

Thule Rigtech (1995). Personal communication.

Titterington, D. M., A. F. M. Smith, and U. E. Makov (1985). *Statistical Analysis of Finite Mixture Distributions.* John Wiley.

Tong, H. (1995). *Non-linear Time Series: a Dynamical System Approach*, Volume 6 of *Oxford Statistical Science Series.* Oxford: Clarendon Press.

Ueda, N. and R. Nakano (1995). Deterministic annealing variant of the EM algorithm. In G. Tesauro, D. S. Touretsky, and J. Alspector (Eds.), *Advances in Neural Information Processing Systems*, Volume 7, pp. 545–552. Morgan Kaufmann.

Vermaak, K., M. Niranjan, and S. J. Godsill (1998). Comparing solution methodologies for speech enhancement within a Bayesian framework. Technical Report CUED/F-INFENG/TR.329, Speech, Vision and Robotics Group, Department of Engineering, University of Cambridge.

Viola, P. and N. Wells (1995), June. Alignment by maximization of mutual information. In I. C. S. Press (Ed.), *Proceedings of the International Conference in Computer Vision*, pp. 16–23.

Viterbi, A. J. (1967), April. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory* **13**, 260–269.

Weigend, A. and N. Gershenfeld (1993). *Time Series Prediction: Forecasting the Future and Understanding the Past*, Volume XV of *Santa Fe Institute Studies in the Sciences of Complexity*. Reading, MA: Addisn-Wesley.

Weigend, A. S., M. Mangeas, and A. N. Srivastava (1995). Nonlinear gated experts for time series. *International Journal of Neural Systems* **3**, 373–399.

Yule, G. U. (1927). On a method of investigating periodicities in disturbed series with special reference to Wolfer's sunspot numbers. *Philosophical Transactions Royal Society London Series A* **226**, 267–298.

Zamora, M. and P. Hanson (1991), January. Rules of thumb to improve high-angle hole cleaning. *Petroleum Engineer International* **48** (51), 44–46.

Appendix A

Maximum likelihood estimation for HMM and SSM

A.1 The Baum-Welch algorithm

The E-step

We represent the variable s_t as a N-dimensional vector $s_t = [s_t^1, \ldots, s_t^N]$, where $s_t^i \in \{0, 1\}$. For example $s_1 = [0, 1, 0, \ldots, 0]$ means that at time $t = 1$, the system is in state 2. Using this representation, each term of the joint probability $P(\mathcal{S}_1^T, \mathcal{Y}_1^T)$ (Equation (3.7)) can be rewritten as:

$$P(s_1) = \prod_{i=1}^{N} \pi_i^{s_1^i}, \tag{A.1}$$

$$P(s_t \mid s_{t-1}) = \prod_{i=1}^{N} \prod_{j=1}^{N} (a_{ij})^{s_{t-1}^i s_t^j}, \tag{A.2}$$

$$P(y_t \mid s_t) = \prod_{i=1}^{N} [P(y_t \mid s_t^i)]^{s_t^i}, \tag{A.3}$$

where, again, we represent the initial state probabilities as a vector $\pi = [\pi_1, \ldots, \pi_N]$. This allows us to write each term of the logarithm of Equation (3.7) as:

$$\log P(\mathcal{S}_1^T, \mathcal{Y}_1^T) = \sum_{i=1}^{N} s_1^i \log \pi_i + \sum_{t=2}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} s_{t-1}^i s_t^j \log a_{ij} + \sum_{t=1}^{T} \sum_{i=1}^{N} s_t^i \log P(y_t \mid s_t^i). \tag{A.4}$$

The function to be evaluated in the E-step is $\mathcal{Q} = \langle \log P(\mathcal{S}_1^T, \mathcal{Y}_1^T, \mid \Theta) \rangle_Q$, *i.e.* the expectation of Equation (A.4) under the posterior distribution of the hidden states $Q(\mathcal{S}_1^T) = P(\mathcal{S}_1^T \mid \mathcal{Y}_1^T, \boldsymbol{w})$. This expectation is a function of $\langle s_t^i \rangle_Q$ and $\langle s_{t-1}^i s_t^j \rangle_Q$. The first term represents the posterior probability of being in state i at time t given the current parameters and the observation

sequence. This is therefore equal to $\gamma_t(i)$ (Equation (3.28)). The second term is the posterior probability of being in state i at time $t-1$ and in state j at time t: this is $\xi_{t-1}(i,j)$ (Equation (3.29)). We therefore get:

$$\mathcal{Q} = \sum_{i=1}^{N} \gamma_1(i) \log \pi_i + \sum_{t=1}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \xi_t(i,j) \log a_{ij} + \sum_{t=1}^{T} \sum_{i=1}^{N} \gamma_t(i) \log P(\boldsymbol{y}_t \mid s_t^i). \tag{A.5}$$

The M-step

A typical initial probability π_i^{new} is the solution of

$$\frac{\partial}{\partial \pi_i^{new}} \left\{ \mathcal{Q} - \lambda_i \left(\sum_{j=1}^{N} \pi_i^{new} - 1 \right) \right\} = 0, \tag{A.6}$$

$$\tag{A.7}$$

where λ_i is a Lagrange multiplier. This gives:

$$\pi_i^{new} = \gamma_1(i). \tag{A.8}$$

Similarly, for a typical transition a_{ij}^{new}, we get the re-estimation formula:

$$a_{ij}^{new} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}. \tag{A.9}$$

Depending on the parametrisation of the output observation distribution $P(\boldsymbol{y}_t \mid s_t^i)$, re-estimates can be easily found. We give here the re-estimation equations for a full covariance Gaussian density function $P(\boldsymbol{y}_t \mid s_t^i) = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$:

$$\boldsymbol{\mu}_i^{new} = \frac{\sum_{t=1}^{T} \gamma_t(i) \boldsymbol{y}_t}{\sum_{t=1}^{T} \gamma_t(i)}, \tag{A.10a}$$

$$\boldsymbol{\Sigma}_i^{new} = \frac{\sum_{t=1}^{T} \gamma_t(i) (\boldsymbol{y}_t - \boldsymbol{\mu}_i^{new})(\boldsymbol{y}_t - \boldsymbol{\mu}_i^{new})'}{\sum_{t=1}^{T} \gamma_t(i)}. \tag{A.10b}$$

A.2 The EM algorithm for state space models

The E-step

For a linear dynamical system, the joint log-likelihood is given by:

$$\begin{aligned} \log P(\mathcal{S}_1^T, \mathcal{Y}_1^T) = & -\frac{(m+d)T}{2} \log(2\pi) - \frac{1}{2} \log |\boldsymbol{\Sigma}| - \frac{(T-1)}{2} \log |\boldsymbol{Q}| - \frac{T}{2} \log |\boldsymbol{R}| \\ & -\frac{1}{2} (\boldsymbol{s}_1 - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\boldsymbol{s}_1 - \boldsymbol{\mu}) \\ & -\frac{1}{2} \sum_{t=2}^{T} (\boldsymbol{s}_t - \boldsymbol{F}\boldsymbol{s}_{t-1})' \boldsymbol{Q}^{-1} (\boldsymbol{s}_t - \boldsymbol{F}\boldsymbol{s}_{t-1}) \\ & -\frac{1}{2} \sum_{t=1}^{T} (\boldsymbol{y}_t - \boldsymbol{G}\boldsymbol{s}_t)' \boldsymbol{R}^{-1} (\boldsymbol{y}_t - \boldsymbol{G}\boldsymbol{s}_t). \end{aligned} \tag{A.11}$$

If all the variables were observed, then the ML parameters could be easily solved by taking the derivatives of Equation (A.11). Since the state vectors s_t are not observable, we use expected values wherever we do not have access to the actual observed values. We therefore require the expectation of s_t, $s_t s_t'$ and $s_t s_{t-1}'$ with respect to the posterior distribution $Q(\mathcal{S}_1^T) = P(\mathcal{S}_1^T \mid \mathcal{Y}_1^T, \boldsymbol{w})$:

$$s_{t|T} = \langle s_t \rangle_Q \tag{A.12}$$

$$\boldsymbol{V}_{t|T} = \langle s_t {s_t}' \rangle_Q \tag{A.13}$$

$$\boldsymbol{V}_{t,t-1|T} = \langle s_t {s_{t-1}}' \rangle_Q \tag{A.14}$$

As seen in Chapter 3, the Kalman smoother computes $s_{t|T}$ and $\boldsymbol{P}_{t|T} = \mathrm{cov}[s_t \mid \mathcal{Y}_1^T]$ recursively. What we need is $\boldsymbol{V}_{t|T} = \boldsymbol{P}_{t|T} + s_{t|T} {s_{t|T}}'$ and $\boldsymbol{V}_{t,t-1|T} = \boldsymbol{P}_{t,t-1|T} + s_{t|T} {s_{t-1|T}}'$, where $\boldsymbol{P}_{t,t-1|T} = \mathrm{cov}[s_t, s_{t-1} \mid \mathcal{Y}_1^T]$. The latter can be calculated with the backward recursion (Shumway and Stoffer, 1982):

$$\boldsymbol{P}_{t-1,t-2|T} = \boldsymbol{P}_{t-1|t-1} {\boldsymbol{J}_{t-2}}' + \boldsymbol{J}_{t-1}(\boldsymbol{P}_{t,t-1|T} - \boldsymbol{F}\boldsymbol{P}_{t-1|t-1}){\boldsymbol{J}_{t-2}}'. \tag{A.15}$$

starting with $\boldsymbol{P}_{T,T-1|T} = (\boldsymbol{I} - \boldsymbol{K}_T \boldsymbol{G})\boldsymbol{F}\boldsymbol{P}_{t-1|t-1}$.

The M-step

By taking the derivatives of the expected log-likelihood (Equation (A.11)) with respect to each parameter, we have immediately the re-estimation equations (Shumway and Stoffer, 1982; Ghahramani and Hinton, 1996):

$$\boldsymbol{F}^{new} = \left(\sum_{t=2}^{T} \boldsymbol{V}_{t,t-1|T} \right) \left(\sum_{t=2}^{T} \boldsymbol{V}_{t-1|T} \right)^{-1} \tag{A.16}$$

$$\boldsymbol{G}^{new} = \left(\sum_{t=1}^{T} \boldsymbol{y}_t s_{t|T}' \right) \left(\sum_{t=1}^{T} \boldsymbol{V}_{t|T} \right)^{-1} \tag{A.17}$$

$$\boldsymbol{Q}^{new} = \frac{1}{T-1} \left(\sum_{t=1}^{T} \boldsymbol{P}_{t|T} - \boldsymbol{F}^{new} \sum_{t=2}^{T} \boldsymbol{V}_{t-1,t|T} \right) \tag{A.18}$$

$$\boldsymbol{R}^{new} = \frac{1}{T} \sum_{t=1}^{T} (\boldsymbol{y}_t \boldsymbol{y}_t' - \boldsymbol{G}^{new} s_{t|T} \boldsymbol{y}_t') \tag{A.19}$$

$$\mu^{new} = s_{1|T} \tag{A.20}$$

$$\Sigma^{new} = \boldsymbol{P}_{1|T} \tag{A.21}$$

Appendix B

Maximum Mutual Information Estimation

In this appendix we derive the equations used in the maximum mutual information estimation procedure. For simplicity, we consider a 2-dimensional Gaussian distribution $P(x_t, y_t) = \mathcal{N}(\mu, \Sigma)$ with mean and covariance matrix

$$\boldsymbol{\mu} = (\mu_1, \mu_2), \tag{B.1a}$$

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix}. \tag{B.1b}$$

In this case, the conditional density is also Gaussian

$$P(y_t \,|\, x_t) = \mathcal{N}\left(\mu_2 + \frac{\sigma_{12}}{\sigma_1^2}(x_t - \mu_1), \sigma_2^2 - \frac{\sigma_{12}}{\sigma_1^2}\right). \tag{B.2}$$

Setting

$$m_t = \mu_2 + \frac{\sigma_{12}}{\sigma_{11}}(x_t - \mu_1), \tag{B.3}$$

$$\sigma = \sigma_2^2 - \frac{\sigma_{12}}{\sigma_1^2}, \tag{B.4}$$

it is straightforward to calculate the derivatives with respect to m_t and σ:

$$\frac{\partial}{\partial m_t} \log P(y_t \,|\, x_t) = \frac{y_t - m_t}{\sigma}, \tag{B.5}$$

$$\frac{\partial}{\partial \sigma} \log P(y_t \,|\, x_t) = -\frac{1}{2\sigma}\left(1 - \frac{(y - m_t)^2}{\sigma}\right). \tag{B.6}$$

Derivatives of the expected log-likelihood

Given an observation sequence $\mathcal{Z}_1^T = (\mathcal{X}_1^T, \mathcal{Y}_1^T)$, we need to compute the derivatives of the expected log-likelihood (Equation (A.5))

$$\mathcal{Q} = \sum_{i=1}^{N} \gamma_1(i) \log \pi_i + \sum_{t=1}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \xi_t(i,j) \log a_{ij} + \sum_{t=1}^{T} \sum_{i=1}^{N} \gamma_t(i) \log P(y_t \mid s_t^i, x_t) \tag{B.7}$$

with respect to θ, a component of $\boldsymbol{w}$. For clarity, we will omit the term x_t, as for each hidden state i we model the conditional probability density $b_i(z_t) = P(y_t \mid x_t)$. We get immediately:

$$\frac{\partial \mathcal{Q}}{\partial \pi_i} = \frac{\gamma_1(i)}{\pi_i}, \tag{B.8}$$

$$\frac{\partial \mathcal{Q}}{\partial a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{a_{ij}}, \tag{B.9}$$

$$\frac{\partial \mathcal{Q}}{\partial m_t^i} = \gamma_t(i) \frac{(y_t - m_t^i)}{\sigma^i}, \tag{B.10}$$

$$\frac{\partial \mathcal{Q}}{\partial \sigma^i} = -\frac{\gamma_t(i)}{2\sigma^i} \left(1 - \frac{(y - m_t^i)^2}{\sigma^i}\right). \tag{B.11}$$

Parameterisation

For ML we incorporated Lagrange multipliers in the M-step in order to ensure constraint satisfaction. For MMI, we use general unconstrained optimisation algorithms. For instance, in order to ensure that the parameters a_{ij} can be interpreted as transition probabilities, they must satisfy

$$\sum_{j=1}^{N} a_{ij} = 1, \tag{B.12a}$$

$$0 \leq a_{ij} \leq 1. \tag{B.12b}$$

These constraints can be satisfied by choosing a_{ij} to be a *softmax* version of an unconstrained variable w_{ij} (Bridle, 1990)

$$a_{ij} = \frac{e^{w_{ij}}}{\sum_{k=1}^{N} e^{w_{ik}}}. \tag{B.13}$$

Differentiating Equation (B.13) and using the chain rule, we have

$$\frac{\partial a_{ij}}{\partial w_{ik}} = \delta_{jk} a_{ik} - a_{ij} a_{ik}, \tag{B.14}$$

$$\frac{\partial \mathcal{Q}}{\partial w_{ij}} = \sum_k \frac{\partial \mathcal{Q}}{\partial a_{ik}} \frac{\partial a_{ik}}{\partial w_{ij}}. \tag{B.15}$$

Substitution in Equation (B.9) gives the gradient of the expected log-likelihood with respect to the new parameters w_{ij}[1].

[1] A similar constraint applies also for the initial probabilities π.

We must also ensure that the covariance matrix Σ is positive definite and symmetric. This can be done by defining an upper triangular matrix U which represents the Cholesky decomposition of Σ. U must have positive diagonal entries, but upper triangular entries are arbitrary.

$$\boldsymbol{U} = \begin{pmatrix} e^{u_1} & u_3 \\ 0 & e^{u_2} \end{pmatrix}. \tag{B.16}$$

So, writing $\boldsymbol{\Sigma} = \boldsymbol{U}^{'}\boldsymbol{U}$, we obtain

$$\boldsymbol{\Sigma} = \begin{pmatrix} e^{2u_1} & u_3 e^{u_3} \\ u_3 e^{u_3} & u_3^2 + e^{2u_2} \end{pmatrix}. \tag{B.17}$$

Again using the chain rule in Equation (B.5) and Equation (B.6), we can easily compute the derivatives of the output probabilities $b_i(z_t)$ with respect to the new parameters and therefore the derivatives of $\mathcal{Q}$:

$$\frac{\partial \mathcal{Q}}{\partial \mu_1^i} = -u_3 e^{-u_1} \sum_{t=1}^{T} \gamma_t(i) \frac{(y_t - m_t^i)}{\sigma^i} \tag{B.18}$$

$$\frac{\partial \mathcal{Q}}{\partial \mu_2^i} = \sum_{t=1}^{T} \gamma_t(i) \frac{(y_t - m_t^i)}{\sigma^i} \tag{B.19}$$

$$\frac{\partial \mathcal{Q}}{\partial u_1^i} = -u_3 e^{-u_1} \sum_{t=1}^{T} \gamma_t(i) \frac{(x_t - \mu_1^i)(y_t - m_t^i)}{\sigma^i} \tag{B.20}$$

$$\frac{\partial \mathcal{Q}}{\partial u_2^i} = -\sum_{t=1}^{T} \gamma_t(i)(1 - \frac{(y_t - m_t^i)^2)}{\sigma^i} \tag{B.21}$$

$$\frac{\partial \mathcal{Q}}{\partial u_3^i} = e^{-u_1} \sum_{t=1}^{T} \gamma_t(i) \frac{(x_t - \mu_1^i)(y_t - m_t^i)}{\sigma^i} \tag{B.22}$$

These derivatives can then be used with a nonlinear optimisation algorithm, such as conjugate gradients, in order to minimise the objective function.

Appendix C

Implementation of dynamical local models

In this appendix we first derive the lower bound on the log-likelihood which is maximised with variational parameters in the generalised EM algorithm for dynamical local models. We then give the equations needed for training linear and non-linear switching state space models.

C.1 Lower bound on the log-likelihood

We represent the N-valued discrete random variable $\boldsymbol{S}_t$ at time t as a N-dimensional vector $\boldsymbol{S}_t = [S_t^{(1)}, \ldots, S_t^{(N)}]$ where $S_t^{(i)} \in \{0, 1\}$. The variational techniques approach consists of maximising the following quantity:

$$\mathcal{F}(Q, \boldsymbol{w}) = \sum_{\mathcal{S}_1^T} \int Q(\mathcal{S}_1^T, \mathcal{X}_1^T) \log \frac{P(\mathcal{S}_1^T, \mathcal{X}_1^T, \mathcal{Y}_1^T \,|\, \boldsymbol{w})}{Q(\mathcal{S}_1^T, \mathcal{X}_1^T)} \, d\mathcal{X}_1^T. \tag{C.1}$$

This quantity is useful for comparing different models as it represents the lower bound on the log-likelihood. It is also important for assessing convergence of the EM algorithm. Recall that each state space model is defined by the following equations:

$$P(\boldsymbol{x}_1^{(i)}) = \mathcal{N}(\boldsymbol{x}_1^{(i)}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{C.2}$$

$$P(\boldsymbol{x}_t^{(i)} \,|\, \boldsymbol{x}_{t-1}^{(i)}) = \mathcal{N}(\boldsymbol{x}_t^{(i)}; \boldsymbol{F}_i \boldsymbol{x}_{t-1}^{(i)}, \boldsymbol{Q}_i), \tag{C.3}$$

$$P(\boldsymbol{y}_t \,|\, \boldsymbol{x}_t^{(i)}) = \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{G}_i \boldsymbol{x}_t^{(i)}, \boldsymbol{R}_i), \tag{C.4}$$

where $\boldsymbol{x}_t^{(i)} \in \mathbb{R}^m$ and $\boldsymbol{y}_t \in \mathbb{R}^d$.

The true joint log-likelihood is:

$$\begin{aligned}
\log P(\mathcal{S}_1^T, \mathcal{X}_1^T, \mathcal{Y}_1^T \,|\, \boldsymbol{w}) \;=\; & \sum_{i=1}^{N} s_1^{(i)} \log \pi_i + \sum_{t=2}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N} s_{t-1}^{(j)} s_t^{(i)} \log a_{ji} \\
& -\frac{mN}{2}\log 2\pi - \frac{1}{2}\sum_{i=1}^{N}\log|\boldsymbol{\Sigma}_i| \\
& -\frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{x}_1^{(i)} - \boldsymbol{\mu}_i)'(\boldsymbol{\Sigma}_i)^{-1}(\boldsymbol{x}_1^{(i)} - \boldsymbol{\mu}_i) \\
& -\frac{mN(T-1)}{2}\log 2\pi - \frac{(T-1)}{2}\sum_{i=1}^{N}\log|\boldsymbol{Q}_i| \\
& -\frac{1}{2}\sum_{t=2}^{T}\sum_{i=1}^{N}(\boldsymbol{x}_t^{(i)} - \boldsymbol{F}_i\boldsymbol{x}_{t-1}^{(i)})'(\boldsymbol{Q}_i)^{-1}(\boldsymbol{x}_t^{(i)} - \boldsymbol{F}_i\boldsymbol{x}_{t-1}^{(i)}) \\
& -\frac{d}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} s_t^{(i)}\log 2\pi - \frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} s_t^{(i)}\log|\boldsymbol{R}_i| \\
& -\frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} s_t^{(i)}(\boldsymbol{y}_t - \boldsymbol{G}_i\boldsymbol{x}_t^{(i)})'(\boldsymbol{R}_i)^{-1}(\boldsymbol{y}_t - \boldsymbol{G}_i\boldsymbol{x}_t^{(i)}). \qquad \text{(C.5)}
\end{aligned}$$

Similarly, the logarithm of the variational approximating distribution is given by:

$$\begin{aligned}
\log Q(\mathcal{S}_1^T, \mathcal{X}_1^T) \;=\; & \sum_{i=1}^{N} s_1^{(i)} \log \pi_i + \sum_{t=2}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N} s_{t-1}^{(q)} s_t^{(i)} \log a_{ji} \\
& +\sum_{t=1}^{T}\sum_{i=1}^{N} s_t^{(i)}\log q_t^{(i)} \\
& -\frac{mN}{2}\log 2\pi - \frac{1}{2}\sum_{i=1}^{N}\log|\boldsymbol{\Sigma}_i| \\
& -\frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{x}_1^{(i)} - \boldsymbol{\mu}_i)'(\boldsymbol{\Sigma}_i)^{-1}(\boldsymbol{x}_1^{(i)} - \boldsymbol{\mu}_i) \\
& -\frac{mN(T-1)}{2}\log 2\pi - \frac{(T-1)}{2}\sum_{i=1}^{N}\log|\boldsymbol{Q}_i| \\
& -\frac{1}{2}\sum_{t=2}^{T}\sum_{i=1}^{N}(\boldsymbol{x}_t^{(i)} - \boldsymbol{F}_i\boldsymbol{x}_{t-1}^{(i)})'(\boldsymbol{Q}_i)^{-1}(\boldsymbol{x}_t^{(i)} - \boldsymbol{F}_i\boldsymbol{x}_{t-1}^{(i)}) \\
& -\frac{d}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} h_t^{(i)}\log 2\pi - \frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} h_t^{(i)}\log|\boldsymbol{R}_i| \\
& -\frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} h_t^{(i)}(\boldsymbol{y}_t - \boldsymbol{G}_i\boldsymbol{x}_t^{(i)})'(\boldsymbol{R}_i)^{-1}(\boldsymbol{y}_t - \boldsymbol{G}_i\boldsymbol{x}_t^{(i)}) - \log Z, \qquad \text{(C.6)}
\end{aligned}$$

where Z is a normalisation factor. By taking the difference between $\langle \log P(\mathcal{S}_1^T, \mathcal{X}_1^T, \mathcal{Y}_1^T \,|\, \boldsymbol{w})\rangle_Q$

and $\langle \log Q(s_1^T, \boldsymbol{x}_1^T) \rangle_Q$, several terms disappear. We get:

$$\mathcal{F}(Q, \boldsymbol{w}) = \sum_{t=1}^{T} \sum_{i=1}^{N} h_t^{(i)} \log q_t^{(i)} + \log Z. \tag{C.7}$$

This corresponds to the lower bound on the log-likelihood that should be computed at each iteration of the EM algorithm in order to assess convergence. Unfortunately, because of the presence of the normalisation factor Z which is intractable to compute, it may happen that this bound decreases from one iteration to the next one.

C.2 The EM algorithm for linear switching state space models

The E-step

The E-step involves the Kalman smoother for each state space model i where the output covariance matrix $\boldsymbol{R}_i$ is weighted by $1/h_t^{(i)}$ at each time step t. This allows to compute the variational parameters $q_t^{(i)}$ (Equation (6.13)). These parameters are then used in the forward-backward algorithm as output density probabilities, and this enables us to estimate the responsibility $h_t^{(i)}$ of each model. The whole process is repeated until convergence of the KL divergence, or similarly convergence of the lower bound (Equation (C.7)).

The M-step

For the M-step, we make use of the re-estimations formulae given in Appendix A for HMMs and SSMs. The re-estimation equations for the transition matrix A and the initial probabilities Π are exactly the same as those obtained for an HMM (see section A.1). Concerning the re-estimation equations of each linear dynamical filter (see section A.2), the equations are also the same except for the output matrices $\boldsymbol{G}_i$ and the output noise covariance matrices $\boldsymbol{R}_i$. We must indeed take into account the responsibility of each state space model. This responsibility is given by the value of the variational parameters $h_t^{(i)}$. It is easy to obtain:

$$\boldsymbol{G}_i^{new} = \left(\sum_{t=1}^{T} h_t^{(i)} \boldsymbol{y}_t \boldsymbol{x}_{t|T}^{(i)\prime} \right) \left(\sum_{t=1}^{T} h_t^{(i)} \boldsymbol{V}_{t|T}^{(i)} \right)^{-1} \tag{C.8}$$

$$\boldsymbol{R}_i^{new} = \sum_{t=1}^{T} h_t^{(i)} \left(\boldsymbol{y}_t \boldsymbol{y}_t' - \boldsymbol{F}_i^{new} \boldsymbol{x}_{t|T}^{(i)} \boldsymbol{y}_t' \right) / \sum_{t=1}^{T} h_t^{(i)}, \tag{C.9}$$

where $\boldsymbol{x}_{t|T}^{(i)}$ and $\boldsymbol{V}_{t|T}^{(i)}$ are obtained by running the Kalman smoother on each state space model.

C.3 The EM algorithm for non-linear switching state space models

The E-step

The E-step involves the linearisation of each output function g_i. This function is approximated by an RBF network:

$$g_i(\boldsymbol{x}_t) = \boldsymbol{W}^{(i)}\boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_t^{(i)}), \tag{C.10}$$

where $\boldsymbol{W}^{(i)} = [w_1^{(i)}, \ldots, w_K^{(i)}]$ are the weights (including the bias) and $\boldsymbol{\Psi}^{(i)} = [\psi_1^{(i)}, \ldots, \psi_K^{(i)}]$ are the basis. By linearising each basis function $\psi_k^{(i)}$, we get:

$$\begin{aligned}
\langle \boldsymbol{\Psi}^{(i)}(x_t^{(i)})\rangle_Q &= \boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)}) \\
\langle \boldsymbol{\Psi}^{(i)}(x_t^{(i)})\boldsymbol{\Psi}^{(i)}(x_t^{(i)})'\rangle_Q &= \boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)})\boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)})' + \mathcal{J}_{t|T}^{(i)}\boldsymbol{P}_{t|T}\mathcal{J}_{t|T}^{(i)'}
\end{aligned}$$

where $\mathcal{J}_{t|T}^{(i)} \equiv \left.\frac{\partial \psi_k^{(i)}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{t|T}^{(i)}}$ is the Jacobian matrix.

The M-step

By taking the derivatives of the expected log-likelihood and setting them to zero, re-estimation formulae for the parameters are easily obtained. Because we just introduce non-linearity in the output function, the equations are the same as the ones for a linear switching state space model, except the output covariance matrix $\boldsymbol{R}_i$. We get:

$$\begin{aligned}
\boldsymbol{W}^{(i)^{new}} &= \left(\sum_{t=1}^{T} h_t^{(i)} \boldsymbol{y}_t \boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)})'\right)\boldsymbol{\Lambda}_i^{-1} \\
\boldsymbol{R}_i^{new} &= \sum_{t=1}^{T}\left[h_t^{(i)}\left(\boldsymbol{y}_t - \boldsymbol{W}^{(i)^{new}}\boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)})\right)\boldsymbol{y}_t'\right] / \sum_{t=1}^{T} h_t^{(i)}
\end{aligned}$$

with $\boldsymbol{\Lambda}_i = \sum_{t=1}^{T} h_t^{(i)}\left[\boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)})\boldsymbol{\Psi}^{(i)}(\boldsymbol{x}_{t|T}^{(i)})' + \mathcal{J}_{t|T}^{(i)}\boldsymbol{P}_{t|T}^{(i)}\mathcal{J}_{t|T}^{(i)'}\right]$.