Maximally Informative Subspaces: Nonparametric Estimation for Dynamical Systems

by

Alexander T. Ihler

B.S. California Institute of Technology (1998)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2000

Author . . . .
Department of Electrical Engineering and Computer Science
August 9, 2000

Certified by . . . .
John W. Fisher III
Research Scientist, Laboratory for Information and Decision Systems
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

Maximally Informative Subspaces:
Nonparametric Estimation for Dynamical Systems

by

Alexander T. Ihler

Submitted to the Department of Electrical Engineering and Computer Science
on August 9, 2000, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

Modeling complex dynamical systems is a difficult problem with a wide range of applications in prediction, discrimination, and simulation. Classical stochastic models make a number of simplifying assumptions to improve tractability (e.g. linear dynamics, Gaussian uncertainty). While such assumptions lead to algorithms which are both fast and optimal under the assumptions, there are a great many real world problems for which these assumptions are false. Recently, computational power has increased to the point where another method becomes feasible – purely example-based, or "nonparametric", models. Yet these are limited because their computational requirements grow exponentially with the number of variables we observe about the system. For dynamical systems, in which we generally observe the past, this means that processes with any substantial past-dependence become intractable. In this thesis we present a novel dynamical system model making use of a nonparametric estimate of uncertainty, with an information-theoretic criterion for reducing the model's required dimension while preserving as much of the predictive power in the observations as possible. To explore its behavior, we apply this technique to three dynamical systems – a "toy" nonlinear system (random telegraph waves), a real-world time series from predictive literature (Santa Fe laser data), and a more cutting-edge application (on-line signature authentication). Each of these examples demonstrates techniques for improving the model's performance and evidence of its effectiveness.

Thesis Supervisor: John W. Fisher III
Title: Research Scientist, Laboratory for Information and Decision Systems

Maximally Informative Subspaces: Nonparametric Estimation for Dynamical Systems

by

Alexander T. Ihler

Submitted to the Department of Electrical Engineering and Computer Science
on August 9, 2000, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

Modeling complex dynamical systems is a difficult problem with a wide range of applications in prediction, discrimination, and simulation. Classical stochastic models make a number of simplifying assumptions to improve tractability (e.g. linear dynamics, Gaussian uncertainty). While such assumptions lead to algorithms which are both fast and optimal under the assumptions, there are a great many real world problems for which these assumptions are false. Recently, computational power has increased to the point where another method becomes feasible – purely example-based, or "nonparametric", models. Yet these are limited because their computational requirements grow exponentially with the number of variables we observe about the system. For dynamical systems, in which we generally observe the past, this means that processes with any substantial past-dependence become intractable. In this thesis we present a novel dynamical system model making use of a nonparametric estimate of uncertainty, with an information-theoretic criterion for reducing the model's required dimension while preserving as much of the predictive power in the observations as possible. To explore its behavior, we apply this technique to three dynamical systems – a "toy" nonlinear system (random telegraph waves), a real-world time series from predictive literature (Santa Fe laser data), and a more cutting-edge application (on-line signature authentication). Each of these examples demonstrates techniques for improving the model's performance and evidence of its effectiveness.

Thesis Supervisor: John W. Fisher III
Title: Research Scientist, Laboratory for Information and Decision Systems

Acknowledgments

Nos esse quasi nanos gigantum humeris insidientes
(We are as dwarfs, sitting on the shoulders of giants)
–Bernard of Chartres

First of all, I would like to thank my advisors, John Fisher and Alan Willsky, for all of their encouragement, guidance, and insight. John's ability to retain an idea's original excitement even when bogged down in the trenches of implementation has inspirational. In addition, he has always managed to make time to discuss my ideas or general progress even when beset by other claimants. Alan's superb insight into systems and estimation, and his intellectual curiosity for almost everything, has made his ideas, advice and analysis indispensable. I also have appreciated the intellectual camaraderie of the rest of the SSG. Such a talented group of students creates an atmosphere where interests overlap and ideas are freely shared, and I have learned as much from their research as anywhere else. To all of them, especially Junmo Kim, with whom I have had so many discussions on the material within, thank you.

I would like to thank the AI lab for funding my research, and for providing me with a "home away from home". I especially thank Paul Viola for all his time and help; conversations with him formed many of the seeds for this thesis. Paul often had a unique perspective which always put a new spin on ideas. He has gone out of his way to make me feel welcome there, and I appreciate it.

I also owe a great debt to Nick Matsakis, who provided me with the means to collect dynamic signature data. His instant willingness to lend his time, equipment, and code is what made Chapter 5 possible.

I wish that I had the space or the memory to properly thank everyone who has contributed to the creation of this thesis. None of the work presented in this thesis would have been possible without the support I have received from so many people. I would like to thank Michelle for her support; and Widget, my cat — he really knows how to listen. And most of all thanks to all my family and friends; I couldn't have done it without you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problems of Information Extraction

The problem of modeling complex systems is one of the oldest and most fundamental in science and engineering. The human mind is a formidable tool for finding patterns and hypothesizing structure within such systems; the development of the theory of mechanics, electricity, and so forth are testaments to the ability of humans to find an imposed structure within observations. However, in modern applications we often deal with systems which are either too complex to understand in such an analytic way, or in situations where we would like to be able to capture this structure automatically, without requiring human intervention.

This is the essential problem of data mining – how to extract knowledge, or find structure, within observations. Given a large number of observation variables, which are important to the quantity of interest? In what way do they relate? Often, even with complex systems, the brain is capable of discerning order; thus we know it can be done, and some researchers examine the brain searching for an idea as to how. Yet the answer remains elusive, and so for problems which do not behave according to "simple" models, it is of great interest to find an automated solution to these kinds of questions.

To put this another way, if we have a system which we believe to be inherently nonrandom (or rather, "not-very-random") controlled for instance by a relatively small number of "hidden" states which we can only observe indirectly, can we find these states within our observations of the system and relate the states to its behavior?

Some of the most prominent examples of such seemingly complex but intuitively nonrandom systems are found in the tasks necessary for human/computer interaction. Recognition tasks are a fundamental part of biometric authentication systems, which are gaining attention and support in security applications. Applications such as automatic dictation or pen-based text input use recognition of individual words and models of sentence structure to produce more natural interfaces to computers. Everyday systems such as handwriting generally exhibit a large variation between examples of the same word, yet are instantly recognizable to the human brain. Additionally, the dependency structure within such signals is quite long, and so the number of samples which may be relevant to estimating high-level variables such as the word or the identity of the writer is large.

In addition, the "necessary" relationships and "acceptable deviations" (randomness) are very difficult to describe. Other coordinate bases (Fourier, wavelets, etc) are an attempt to describe a simple coordinate system in which, it is hoped, important information is well-separated from the incidental. However, none of these achieve as much as we would like (though of course, some coordinate systems are better than others). It is simply that although our brains know how to detect the hidden relationships representing a word, or underlying biological motion, we are unable to consciously describe those relationships; this is probably a good indication that the relationships themselves are quite complex in nature. One possible explanation for this is that the mind is primarily a vast store of experience, and that our analysis of such systems is based mostly on a comparison to previously seen examples. Nonparametric approaches use exactly this sort of methodology – the model uses as large a store of examples as can be managed in the hopes that there are enough to be "close" to any future observation.

1.2 Simplicity versus Capacity

Such "difficult-to-describe" systems are often the bane of parametric modeling. We like to use simple models and simple relationships, such as linear functions with Gaussian uncertainties, in order to make a problem more tractable. Often, this is a good idea and a reasonable set of assumptions, and the algorithms which have been developed allow "optimal" (assuming that reality fits the model) performance at low computational costs.

Unfortunately when the relationships of the data become too complex, we are forced to turn to models with high capacity. Such models have a great deal of flexibility inherent in them; however, this comes at a price of increased number of parameters which must be selected (learned) and so an increased difficulty in use, often both in computation and in required observation data size.

For instance, it could be argued that given enough data and processing power, nearly any system can be modeled nonparametrically — by assembling a database of all the observations and the corresponding response of the system, we could simply predict the system's behavior by looking up instances of the past which are close to our current observations, and make a prediction based on them. This is nearly ideal in that it is a very flexible model – it is capable of modeling highly nonlinear relationships, capturing noise dynamics, etc – but has an extremely high computational cost associated with it. It is only recently that computing has become powerful enough to contemplate implementing such procedures, and even now there is never "enough" processing power for every task we would like to attempt. In part this is because as the number of variables we are allowed to observe rises, the dimension of the model grows; and the amount of data necessary to accurately model the observations' relationships grows exponentially with the dimension. We have a trade-off between our desire to utilize more observations for their information about the system and our ability to handle those observations. This tradeoff is made all the worse by the fact that we often do not know how helpful any particular type of observation is. We could at least handle this tradeoff intelligently if we knew which information we should keep.

In Section 2.7, we will present a model which does exactly this. We find the "most informative" of a parameterized family of subspaces in order to reduce our computation cost while retaining the most value for the dimensions we keep. We use a nonparametric estimate of the mutual information between the quantity of interest (in a dynamical system, the future of the process) and the retained subspace in order to determine what subspace to keep.

1.3 Many open questions

Unfortunately, there are some basic questions which we have not addressed above, much less proposed solutions to. For instance, how do we even go about measuring the performance of a particular model? The most standard answer is to choose mean-squared-error; many applications choose this simply because everyone else seems to, or because they do not question its appropriateness. However, in many cases it bears little resemblance to the metric we ourselves impose on a solution; solutions which minimize squared error need not be those which "look best". But human qualitative judgment is not a quantitative method, and we do not know how to adapt a model so that, in the end, it "looks good". Perhaps no quantitative method can maximize such an abstract criterion; but without addressing the question of a "good" quality metric we cannot proceed.

Another difficult question is posed by the very idea of "prediction" using a nonparametric uncertainty model; such an action is not well-defined. We could, for example, choose the "most likely" value; however, there can be many equally likely alternatives with no way to choose between them. Worse, there is no guarantee that always selecting "high likelihood" predictions will produce sample paths which meet any quality criterion described above. In fact, it makes little sense to select *any* single point as representative of a future sample; and yet prediction is one of the most basic concepts in system theory. A model which cannot produce *some* kind of prediction is lacking a basic and necessary quality.

The scope of such questions stretches well beyond that of this thesis; yet some kind of answer to them must and will be proposed. The difficulties (and sometimes even advantages!) caused by the differences between nonparametric and more canonical approaches will appear again in each of the three application chapters. In each, we will go into more detail about the nature of the question (how does one measure the quality of this process? how does prediction relate to a process of this type?) and discuss the appropriateness or inappropriateness of our solutions.

Chapter 2

Preliminary Information

The focus of this chapter is to present the background and concepts which are necessary for development of the thesis. Section 2.1 provides a background of entropy, likelihood and hypothesis testing. In Section 2.2 we discuss criteria for prediction and sample path synthesis in a generalized noise framework. Section 2.3 gives a brief introduction to the linear-quadratic-Gaussian assumption and its solution for stationary systems, the Wiener filter. We then cover the basics of nonparametric density estimation, the Parzen density, and kernel functions in Section 2.4, and discuss the application of these ideas to the problem of entropy estimation in Section 2.5. In Section 2.7 we state a more precise formulation of our model and problem, and discuss our technique for searching the model space for good representations.

2.1 Entropy and Likelihood

2.1.1 Entropy and Mutual Information

The contents of this thesis are heavily dependent on ideas from information theory, specifically the concepts of entropy and mutual information. Thus, we provide some of the definitions and results which will be made use of later. For a full development of the subject of entropy and information theory, see [4].

Entropy is a measure of randomness, or equivalently of uncertainty. For a (continuous) random variable X distributed according to a density function $p(x)$ with support S, the (differential) entropy is defined to be

$$H(p) = E_p[-\log p(x)] = -\int_S p(x) \log p(x) dx \tag{2.1}$$

For a process $\mathcal{X} = \{X_t\}$ (a time-indexed sequence of random variables) the *entropy rate* is defined as [4]

$$H(\mathcal{X}) = \lim_{N \to \infty} \frac{1}{N} H(X_1, \ldots, X_N)$$

when the limit exists. It is easy to show that stationarity of $\mathcal{X}$ is a sufficient condition for convergence of the limit.

The *relative entropy*, or *Kullback-Leibler divergence* between two probability distributions $p(x)$ and $q(x)$ is given by

$$D(p\|q) = E_p\left[\log \frac{p(x)}{q(x)}\right]$$

Although this quantity does not behave as a true distance metric (for instance, it is not symmetric), it does have some distance-like properties that make it useful to think of it in this way. Specifically, $D(p\|q)$ is always non-negative and equals zero if and only if $p = q$.

Mutual information provides us with a quantifiable measure of the information shared between two random variables, specifically the reduction in entropy when one of the two is known. Some useful equivalent forms of this are given:

$$\begin{aligned} I(X;Y) \quad &= I(Y;X) \quad &&= I(X; f(Y)) \qquad f(\cdot) \text{ any invertible function} \\ & &&= D(p(x,y)\|p(x)p(y)) \\ & &&= H(X) - H(X|Y) \\ & &&= H(X) + H(Y) - H(X,Y) \end{aligned}$$

A note with respect to these forms and our application: the third form $H(X) - H(X|Y)$ has an intuitive interpretation, namely the reduction in uncertainty of the "variable of interest" X given some observations Y. However, since we will usually be dealing with a continuous value for our conditioned variable Y, computationally speaking we will be manipulating the the last form, $H(X) + H(Y) - H(X,Y)$.

In addition, we will find the *data processing inequality* to be of great importance, as it describes the concept of information *loss*. It states that

$$I(X;Y) \geq I(X; f(Y)) \qquad \text{for any } f(\cdot)$$

with equality if (but not only if) $f(\cdot)$ is bijective. That is, we can at best preserve information by processing data; we can never have more than the original data, and if we are not careful we can destroy it.

In statistics, there is a notion of *sufficiency* which is related to these concepts. If we observe a random variable Y, related to a system controlled by the underlying variable X, any statistic $f(\cdot)$ forms a Markov chain:

$$X \to Y \to f(Y)$$

implying graphically that, conditioned on the value of Y, $f(Y)$ is independent of X. A *sufficient statistic* of Y for X is a function $f(\cdot)$ (generally not invertible) for which the following is also a Markov chain

$$X \to f(Y) \to Y$$

This is equivalent to an equality of mutual information:

$$I(X;Y) = I(X;f(Y))$$

So a sufficient statistic is one which has attained the absolute maximum mutual information.

This property is generally taken to be a boolean notion; either a statistic is sufficient or it is not. However, within non-sufficient statistics there are degrees of loss. This creates the notion of *relative sufficiency* [20]. Such a concept is useful since there are many cases when a true sufficient statistic of dimension less than the original Y is not known or does not exist, yet (perhaps due to data volume) the data must still be summarized by some function. In such cases it behooves us to use as “nearly sufficient” a statistic as we can. In the future we will sometimes refer to such “relatively sufficient” functions as “informative statistics”.

2.1.2 Likelihood & Hypothesis Testing

Later in the thesis we will be discuss using our models for discriminative purposes, meaning, selecting or rejecting a model as “fitting” new data. This is fundamentally a question of the likelihood $p(x)$. For a more complete treatment of the subject see e.g. [26].

Suppose we wish to decide between two hypotheses H_0 and H_1, given a vector of observations $\mathbf{X}$. If we define P_D, P_F to be the probabilities of detection (decide $\hat{H} = H_1$ when H_1 is true) and false-alarm ($\hat{H} = H_1$ when H_0 is true), respectively, we can ask to find the decision rule which maximizes P_D for $P_F \leq \alpha$. This is the Neyman-Pearson criterion.

Theorem 1 (Neyman-Pearson Rule) *Define $p_0(\mathbf{X}), p_1(\mathbf{X})$ to be the probability density function of $\mathbf{X}$ under H_0, H_1 respectively, and let $L(\mathbf{X}) \doteq \frac{p_1(\mathbf{X})}{p_0(\mathbf{X})}$. The decision rule which maximizes P_D for $P_F \leq \alpha$ has the form*

$$\hat{H} = \begin{cases} H_0 & L(\mathbf{X}) \leq \lambda \\ H_1 & L(\mathbf{X}) > \lambda \end{cases}$$

Proof. See [26].

This gives us the *likelihood ratio test*:

$$\frac{p_1(\mathbf{X})}{p_0(\mathbf{X})} \underset{\hat{H}=H_0}{\overset{\hat{H}=H_1}{\substack{\geq \\ <}}} \lambda$$

where λ has been chosen so that $P_F = \alpha$. We could also choose λ so that $P_F = 1 - P_D$, i.e. that we wish the probability of incorrectly deciding H_1 to equal the probability of incorrectly deciding H_0. In this case we find that $\lambda = 1$; this is also called the *maximum likelihood test*.

We can apply any monotonic transformation without changing the test; it is common to take the logarithm, giving

$$\log p_1(\mathbf{X}) - \log p_0(\mathbf{X}) \underset{\hat{H}=H_0}{\overset{\hat{H}=H_1}{\substack{\geq \\ <}}} \log \lambda$$

It is worth mentioning that if $\mathbf{X} = \{X_i\}_1^n$, with X_i independent, then the average log-likelihood,

$$n^{-1} \log L(\mathbf{X}) = n^{-1} \sum \log p(x_i|H_1) - n^{-1} \sum \log p(x_i|H_0)$$

provides an equivalent test. It should also be noted that this framework is easily extensible to M hypotheses.

The likelihood ratio test ties in directly to the Kullback-Leibler distance function as well, since

$$D(p_1\|p_0) = E_{p_1}[\log(\frac{p_1}{p_0})] \qquad \text{and`} D(p_0\|p_1) = -E_{p_0}[\log(\frac{p_1}{p_0})]$$

So the KL distance has an interpretation as the average log value of the likelihood ratio when H_1 is true, or as the negative of the average log likelihood ratio when H_0 is true; thus relating the distance between two hypotheses to our ability to discriminate between them.

Finally, we can even construct a test for a single hypothesis with unknown alternatives, e.g.

$$\begin{array}{ll} H_1: & \mathcal{X} = \mathcal{X}_1 \\ H_0: & \mathcal{X} \neq \mathcal{X}_1 \end{array}$$

In this case, without any sort of prior knowledge of the alternatives when $\mathcal{X} \neq \mathcal{X}_1$, we cannot calculate P_F, but given $p(\mathcal{X})$ we know the entropy rate of $\mathcal{X}$ and for e.g. a symmetric threshold test

$$|H(\mathcal{X}) + \frac{1}{N} \log p(X_1, \ldots, X_N)| \leq \eta \Rightarrow H_1$$

we can calculate η from a given P_D. We will discuss this further as it relates to one of our applications, in Chapter 5. A more detailed treatment of entropy rates and hypothesis testing can be found in [4].

2.2 Model-based synthesis

Whenever one talks about prediction, there is implicit an idea of the cost (or loss) function associated with that prediction. A predictor cannot be good or bad except in relation to a particular cost function. Typically this function is taken to be e.g. mean-squared-error. The MSE predictor, however, may lead to atypical predictions (in symmetric bimodal distributions, for instance, it will select the center). Unfortunately this simply highlights that often the loss function we would like to use is not so simple

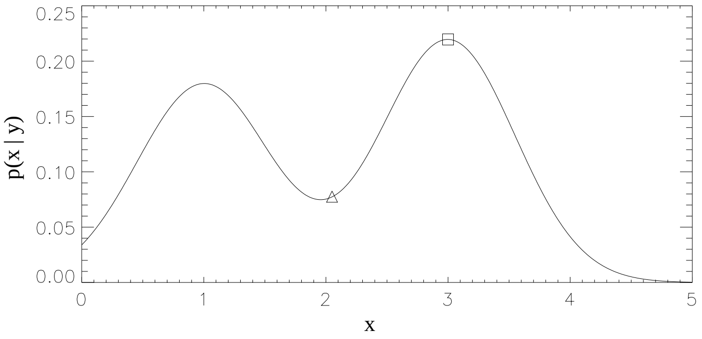


Figure 2.1: A bimodal density with ML estimate (triangle) and MSE estimate (square)

to evaluate, but instead corresponds to some more heuristic ideal. In cases like this, a more sensible predictor from a likelihood standpoint might be the max likelihood estimate (argmax $p(x)$). This carries with it its own difficulties, however, since the ML estimate may not even be unique, and the ML sample path may not be typical, and so may not bear resemblance to any observed sequences.

The idea of evaluating the quality of sample path realizations is even less well-defined. For a model to truly capture the dynamics of a system, it should be able to produce not just the most likely sample path, but a set of sample paths which (one hopes) approximates the true distribution of such paths. Again, evaluating the quality of these new sample paths implies more analytic knowledge of the system then we are likely to have. Consequently, this forms another open question arising from a methodology which does not conform to a traditional predictive representation. Fully exploring such questions is beyond the scope of this thesis, but we will address them empirically later in the thesis.

2.3 Linear methods & the Wiener Filter

In the past, a large body of the work on dynamical systems has been in the regime of the Linear-Quadratic-Gaussian (LQG) assumptions, namely linear system dynamics with gaussian noise, optimizing for a quadratic cost function. We briefly review some of this work here, as it will be used later for comparison.

Suppose we wish to predict a stationary process $\mathcal{X} = \{X_j\}$ from observations $\mathbf{y}_j$, where each $\mathbf{y}_j$ is a vector $\mathbf{y} = [y_1, \cdots, y_M]$. Let $R_{\mathbf{yy}}$ be the covariance matrix for $\mathbf{y}$, so

$$(R_{\mathbf{yy}})_{i,j} = E[y_i, y_j]$$

Denote the cross-covariance between X and $\mathbf{y}$ as

$$R_{x\mathbf{y}} = [E[x, y_1], \cdots, E[x, y_M]]$$

Then, an optimal α in the sense that it minimizes

$$E\left[(X_i - \alpha^T \mathbf{y})^2\right]$$

will solve the *normal equations*,

$$R_{x\mathbf{y}}^T = R_{\mathbf{yy}} \alpha$$

If $R_{\mathbf{yy}}$ is positive definite, then α is unique and given by

$$\alpha = R_{\mathbf{yy}}^{-1} R_{x\mathbf{y}}^T$$

A more detailed treatment and proofs can be found in [26].

2.4 Nonparametric Density Estimation

Sometimes, however, it is not desirable to make this assumption of Gaussianity for any randomness in the system. In some cases, when attributes of the system are well known, it may be judged that another type of parametric density is more suitable for fitting to the data. But other times it is more desirable to simply let the data observed about the system dictate the modeled randomness, unconstrained (or as loosely as possible) by an apriori form. This last is the goal of nonparametric density estimators.

2.4.1 Parzen Window Estimator

A common form of nonparametric density estimator is the Parzen window, or kernel, estimate. Given data observations $\{X_i\}_1^N$, we define

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} K_h(x - X_i) \tag{2.2}$$

where

$$K_h(x) = \frac{1}{h}K(\frac{x}{h}) \tag{2.3}$$

Here, $K(\cdot)$ is the *kernel function* and h represents the kernel size, or *bandwidth*. $K(\cdot)$ is usually taken to be a symmetric density function itself, so that $\hat{f}(\cdot)$ is also guaranteed to be a valid density function. However, we must select the form of the kernel function

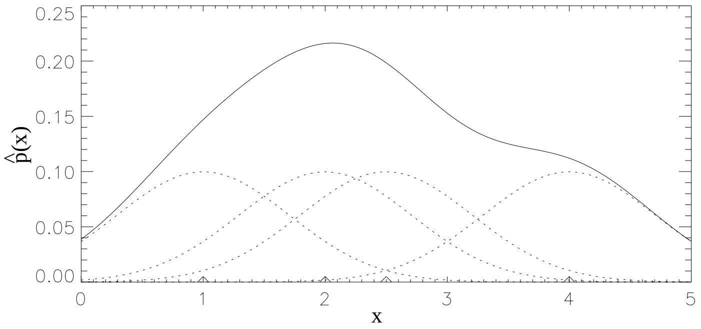


Figure 2.2: A kernel density estimate from 4 samples; kernels shown as dashed

(shape) $K(\cdot)$ and the bandwidth h.

Criteria of Fit

Before we can address how best to construct this estimator, we must try to describe a cost function for our estimate. Currently, the two most commonly used cost functions are the Mean Integrated Squared Error (MISE)

$$MISE = E\left[\int (\hat{f}(x) - f(x))^2 dx\right] \tag{2.4}$$

and the Kullback-Leibler divergence,

$$D(f\|\hat{f}) = \int f(x)(\log f(x) - \log \hat{f}(x))dx \tag{2.5}$$

In general, we will choose to base our methods on the second, due to its similarity to likelihood. KL divergence measures a kind of distance in likelihood, and so it is more plausibly applied to selecting density estimates for hypothesis testing and sampling from the density, both likelihood-based tasks, than a squared-error criterion. (The MISE criterion is generally considered a good choice for density *visualization*, however.)

Kernel Shape and Size

Selection of an optimal kernel shape and size is an important area of research on these techniques. Ideally, one would like to do both; however each is difficult even individually and a method for doing both well has been elusive. In general, we would like to choose our kernel so it is smooth (differentiable) and relatively efficient to evaluate. It may also be that we would like a kernel with "tails", as choosing a kernel with relatively small support may cause outliers to unduly influence the kernel size (see below). An asymptotic argument yields the (asymptotic) optimality (with respect to Mean Integrated Squared Error) of the Epanetchnikov kernel [23],

$$K_e = \begin{cases} \frac{3}{4\sqrt{5}}(1 - \frac{t^2}{5}) & |t| \leq \sqrt{5} \\ 0 & \text{otherwise} \end{cases}$$

Unfortunately, the improvement it offers over other kernel shapes is small, in terms of relative data requirements to achieve the same MISE [23]. Furthermore, the fact that it has finite support will be problematic for our purposes – when evaluating likelihood it would be possible for a single observation outside the support to force our estimate to zero. Often the small MISE improvement is not worth risking such concerns, and convenience or habit encourages us to simply choose a Gaussian shape.

Selection of kernel size is a crucial decision, however. The two criteria above lead to different schools of bandwidth estimators. There has been considerable work for the MISE case, for the "least-squares cross-validation" approach [2, 5], the "plug-in estimator" method [11, 22], and several others. The KL-distance case, however, has received less attention; the most common method is the leave-one-out maximum likelihood approach [5]. Bearing in mind that for the rest of the thesis these density estimates will be used for calculating likelihoods and sampling, it seems natural to select the KL-distance as our error measure, and so we will refrain from further discussion of the alternatives.

Another justification for our use of this metric can be found through comparison to parametric model estimation. It can be shown that for a parametric model $\hat{p}$, minimizing the KL divergence $D(p\|\hat{p})$ is equivalent to performing maximum likelihood parameter estimation. So KL divergence is widely applied as a cost function (albeit under another name) in parametric modeling; thus it is probably reasonable to use it for nonparametric as well.

To minimize $D(f\|\hat{f})$, we equivalently minimize the portion dependent on $\hat{f}$: $-\int f(x)\log\hat{f}(x)$. If we define the leave-one-out estimator

$$\hat{f}_j(x) = \frac{1}{N-1}\sum_{i\neq j} K_h(x - X_i) \tag{2.6}$$

then we can write as an estimate of this quantity the function

$$CV_{ML} = \frac{1}{N}\sum_{i=1}^{N}\log\hat{f}_i(X_i) \tag{2.7}$$

Notice that it is important to use the leave-one-out estimate for $\hat{f}$, since otherwise the ML bandwidth solution would be $h \to 0$, giving a sum of δ-functions at the observed data points.

Variable kernels

Although asymptotically we can expect good performance from the kernel estimation method, we would still like to find ways to improve its small-sample performance. One way this might be possible is to attempt to imbue the kernel size with a sense of the local structure; for instance, it makes sense that samples in the tails of a distribution be more "spread", while keeping samples in dense regions from being over-smoothed. A good method for doing so is given by [23], the "variable" or "adaptive" kernel size.

The basic idea is this: define $h_i = h \cdot \alpha_i$ to be the bandwidth at X_i. Given an estimate of the distribution, use that estimate to assign samples with low probability a high proportionality constant α, and samples with high probability a small α. For example, using the kth nearest neighbor estimate, we could define $h_i = h \cdot d_{i,k}$, where $d_{i,k}$ is the distance to the kth nearest neighbor of X_i. For the adaptive approach, this new estimate may be used to iterate the process. Later in the thesis we shall see examples of advantages gained by using a variable kernel size.

Multivariate densities

A few more issues arise when dealing with multivariate densities. Specifically, it is now possible to choose kernel shapes and sizes which have unequal width in various directions. Proposals for selecting kernel shape include using the covariance structure of the data [9] and iteratively searching over rotations and kernel sizes [19]. Certainly executing such a search can become computationally intensive, due to the coupled nature of all the quantities being estimated. Most of the efficient univariate estimators (such as the plug-in estimate [11, 22]) do not appear to be easily extensible to the multivariate case. Lastly, we note that what is really needed is a method of capturing a local structure property to both shape and bandwidth; but a solution to this has yet to be found. Further discussion will take place later in the thesis, in Section 4.2.

2.5 Nonparametric Entropy Estimation

2.5.1 Entropy

If we wish to use entropy or mutual information as a criterion, for example to search for nearly sufficient functions, we need to evaluate (or more commonly, estimate) the entropy of a random variable. When $p(x)$ is not known exactly, we must estimate H, which may or may not involve estimating p explicitly. In fact, there are many techniques for finding an estimate of the entropy H [18, 2]. Unfortunately most of these techniques are for estimating the entropy in a single dimension, and do not extend easily to multidimensional distributions. If, however, we have explicitly an estimate of the distribution $\hat{p}(x)$, from samples x_i, a simple solution presents itself:

$$\hat{H} = -\int \hat{p}(x) \log \hat{p}(x) dx$$

However, this is not at all easy to calculate; in order find something with more reasonable computational requirements we will be forced to make an approximation somewhere. We mention two possible estimates, which differ only in where the approximation takes place.

The first possibility presents itself from the expectation interpretation of entropy: using the law of large numbers to approximate the expectation, we can write

$$-\hat{H} = E[\log \hat{p}(x)] \approx \frac{1}{N} \sum_{i=1}^{N} \log \hat{p}_i(x_i) \tag{2.8}$$

where $\hat{p}_i$ is the estimated p.d.f. leaving out data point x_i (see e.g. [13, 23]). This method gives us an approximate integral of our estimated density function, easing the computational burden associated with the integration.

2.5.2 ISE Approximation

The second possibility is to approximate the integrand ($p \log p$) of Equation 2.1. It turns out that, expanding $p(x) \log p(x)$ in a Taylor expansion around a uniform density $u(x)$ and keeping terms out to second order, we have

$$-\hat{H} = -\int p(x) \log p(x) \approx \int (p(x) - u(x))^2 dx + \text{a constant}$$

This can give us a useful approximation of the gradient of $H(p)$; for more details see [8].

There is a good justification for this expansion in the case that we are maximizing entropy. For a finite region of support, the distribution which maximizes entropy is the uniform distribution. Therefore, as we get closer and closer to the maximum our approximation becomes better. Unfortunately, maximizing mutual information is not quite so simple – it involves both maximizing the marginal entropy while minimizing the joint. However, it turns out that even in this region, the approximation is still reasonable. Empirical trials have shown that gradient estimates using this estimate are accurate enough for our use.

As with the previous estimate, the computation involved in calculating this estimate is $\mathcal{O}(N^2)$ in the number of data points in the density estimate. It does, however, allow us a few more advantages. There is a slight computational advantage (of constant order), but the main draw of this approach is that it possesses an automatic saturation effect. Namely, if (as above) the function we are training is confined to a finite region of support, our gradient will cease before reaching the edge of that support. A true entropy estimate will continue to expand the region of support as large as possible, giving a nonzero gradient outward for points at the very edge of the region. A neural network (see below), with its saturating nonlinearity, will thus continue to increase its weights *ad infinitum.* Thus one cannot be assured that the network weights will converge; so one must use distance in the output space to determine a stopping time. By using ISE instead, we are assured that our network weights will not grow beyond a reasonable scale (see [8]).

2.6 Neural Networks and Backpropagation

Connectionist techniques, especially neural networks, have become widely used in the past few decades. A neural network, or alternatively an N-layer perceptron, is defined to be a function of the form

$$f(x_1, \ldots, x_M) = \sigma\left(\sum_{i_N} \alpha_{N,i_N} \sigma\left(\sum_{i_{N-1}} \alpha_{N-1,i_{N-1}} \sigma\left(\ldots\left(\sum_{i_1} \alpha_{1,i_1} x_{i_1}\right)\right)\right)\right)$$

where $\sigma(\cdot)$ is a saturating nonlinearity, in our case the hyperbolic tangent function.

They hold the promise of many desirable properties, but also have a number of drawbacks. Among the properties which are most useful in our context is the

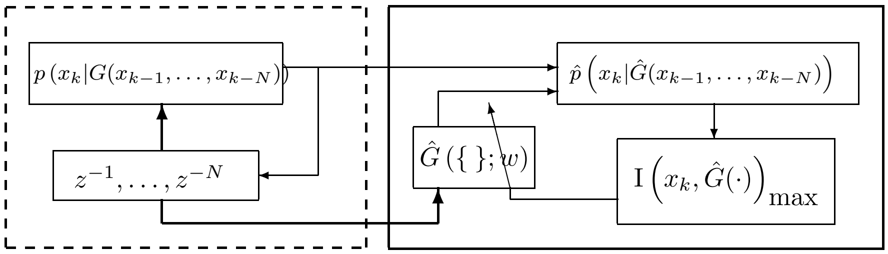

$p\left(x_k|G(x_{k-1},\ldots,x_{k-N})\right)$
$z^{-1},\ldots,z^{-N}$
$\hat{p}\left(x_k|\hat{G}(x_{k-1},\ldots,x_{k-N})\right)$
$\hat{G}\left(\{\};w\right)$
$\mathrm{I}\left(x_k,\hat{G}(\cdot)\right)_{\max}$

Figure 2.3: A Dynamical System Model

fact that, for a sufficiently large network structure, the output can be shown to be a universal approximator [1]. In addition, there exists an efficient method of updating the network weights based on gradient information at the output, generally referred to as back-propagation [3].

2.7 Learning in Dynamical Systems

In this thesis we will be dealing with the modeling of dynamical systems, and so we must ask the question of what class of models we shall attempt to incorporate. We would like the class of models to be large enough that it either contains the true system of our application(s), or a “good” approximation in some sense. However, the model must be limited enough to give us a structure to search over tractably.

We will consider a generalized dynamical system shown in Figure 2.3. Here, x_k denotes the time series we wish to model; $\mathbf{Y}$ denotes a random variable related (causally) to $\mathcal{X}$. The $\mathbf{y}_k$ are our observed values of $\mathbf{Y}$; and $G(\cdot)$ is a possibly nonlinear, possibly vector-valued function. The underling assumption in this diagram is that $G(\mathbf{y})$ is a sufficient statistic for $\mathbf{y}$, and that the conditional distribution, $p(x_k|G(\mathbf{y}_k))$ is constant over time. Here, $\mathbf{y}$ could include past values of the process $\mathbf{x}$, or side information about $\mathbf{x}$ from another observed process. If there is no side information available, this reduces to $p(x_k|G(x_{k-1}\ldots x_{k-N}))$; for example any stable auto-regression of the form $x_k = G(x_{k-1},\ldots,x_{k-N}) + \nu_k$ for ν_k i.i.d. falls into this category. It can be noted that these are equivalent to a Markov chain whose state includes all N of these past values [7].

To summarize, we assume the state of the system is (or is observable within) the vector $\mathbf{y}_k$; $G(\mathbf{y}_k)$ represents that portion of the state which is germane to the prediction of x_k, and $p(x_k, G(\mathbf{y}_k))$ describes the relation between the state and the signal observations. Thus the problem of modeling the process is equivalent to modeling the function $G(\cdot)$ and the distribution $p(x, G(\cdot))$.

In order to search over dynamical systems of this type and be able to model a system as such, we must further limit our model to a parameterized form for $\hat{G}$ and search over the set of parameters. In general, any differentiable function $\hat{G}$ will do; within the rest of this thesis we have selected $\hat{G}$ in the form of a neural network, because of its approximation power and efficient form for the differential parameter updates (see Section 2.6). Although all of the following experiments are restricted to a single layer, the methodology is also applicable to multiple layers.

Since by hypothesis, $G(\mathbf{y})$ is a sufficient statistic for $\mathbf{y}$ (for predicting x), we know that

$$I(x;\mathbf{y}) = I(x;G(\mathbf{y}))$$

and that for any $\hat{G}$,

$$I(x;\mathbf{y}) \geq I(x;\hat{G}(\mathbf{y}))$$

So if we wish to come as close to G as possible, in the sense that we reduce as much uncertainty about x as if we knew G exactly, we should maximize the quantity $I(x; \hat{G}(\mathbf{y}))$. Note that to achieve this bound with equality, we need not be able to represent G exactly – representing any invertible transformation of G is sufficient. Of course, some transformations of G may have simpler relationships $p(x, G(\cdot))$.

Throughout the rest of this thesis we will refer to such a functions $G(\cdot)$ (or approximations $\hat{G}(\cdot)$) alternately as statistics, or functionals of the data, and as subspaces of the data space, meaning the space induced by the function (and its density). $G(\cdot)$ can be thought of as a differentiable projection from $\mathbf{Y}$ whose image is of dimension less than or equal to G's. Therefore, G's inverse image in $\mathbf{Y}$ is of the same dimension, and G describes an equivalence between all points of $\mathbf{Y}$ mapping to the same $G(\mathbf{y})$. If our statistics are sufficient for x, equivalent points in $\mathbf{Y}$ contain exactly the same information about x.

Thus, the basic idea of searching over model space is as follows: choose an initial $\hat{G}$, and model $p(x, \hat{G}(\mathbf{y}))$ from observed data. Estimate $I(x; \hat{G}(\mathbf{y}))$ from the estimated $\hat{p}(\cdot)$ and use its gradient with respect to the parameters of $\hat{G}$ to increase $I(x; \hat{G}(\mathbf{y}))$.

For a step-by-step treatment of the learning algorithm, see Appendix A.

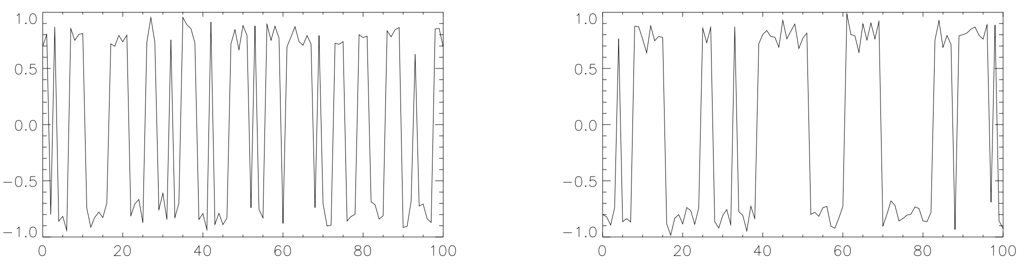
Chapter 3

Learning Random Telegraph Waves

The rest of this thesis will be devoted to applying the idea of mutual information-based learning of system dynamics to various problems. We begin by constructing a simple dynamical system to highlight a few of the areas in which more canonical methods can fail. Yet we take care to keep the system simple enough to allow us to analytically characterize the performance of the algorithm, a characterization which will be nearly impossible for any real-world systems.

3.1 Random Telegraph Waves

Although there is a common stochastic process which is normally known by this title, we will usurp the name for a signal as defined below. Let $\mathcal{X}_M = \{x_k\}$ be a random



(a) (M=4)

(b) (M=20)

Figure 3.1: Random Telegraph Waves of differing memory depths

telegraph wave with memory M (RTW$_M$). Then $x_k = \mu_k + \nu_k$, where

$$\mu_k = \begin{cases} \mu_{k-1} & \text{with probability } p_k \\ -\mu_{k-1} & \text{with probability } 1 - p_k \end{cases} \tag{3.1}$$

$$p_k = \max(\alpha \frac{1}{M} |\sum_{i=1}^{M} x_{k-i}|, 1) \tag{3.2}$$

$$\nu_k \sim N(0, \sigma^2), \quad \text{independent Gaussian noise} \tag{3.3}$$

where $\alpha < 1$ is a constant. We selected this process because its parameters are simple functions of the past, but their relation to the signal's future is nonlinear. Changing the process' memory (M, the length of dependence on the past) changes the observed dynamics of the system. Figure 3.1 shows a comparison of RTW$_M$ signals for different memory depths ($M = 4$ and $M = 20$).

For Figure 3.1 and for the rest of this chapter, we will use the parameters $\alpha = .75$, $\mu = |\mu_k| = .8$, and $\sigma = .1$.

3.2 Process Analysis

Before discussing our methodology or results, it behooves us to briefly discuss the characteristics of the signal defined above. This will give us some insight into how to proceed, and also some bounds and expectations about our possible performance on such a signal.

First of all we can ask the question, what are the true sufficient statistics of our signal? For an RTW$_M$, it is clear that the full state at time $k-1$ is at most $\{\mu_{k-1}, x_{k-1}, \ldots, x_{k-M}\}$, since p_k can be calculated from $\{x_{k-i}\}$ and ν_k is independent of any observation. We also note that $I(x_k; \{\mu_{k-1}, f(x_{past})\})$ is maximized for $f(x_{past}) = p_k$, since x_k is independent of the process' past given μ_k (it is Gaussian with mean μ_k) and μ_k is independent of the past given $\{\mu_{k-1}, p_k\}$ (it is Bernoulli). We can then calculate the entropy of the process given this state

$$\begin{aligned} H(\mathcal{X}_M) &= E_{\mu_{k-1},p_k} H(x_k|\mu_{k-1}, p_k) && (3.4) \\ &= -\sum_{\mu_{k-1}\in\{\pm\mu\}} \int_0^1 p(\mu_{k-1}, p_k) \int p(x_k|\mu_{k-1}, p_k) \log p(x_k|\mu_{k-1}, p_k) dx_k dp_k && (3.5) \end{aligned}$$

By symmetry of the definition, $p(\mu_k = \mu) = .5$ and $p(p_k|\mu_k) = p(p_k)$, and so we can write

$$H(\mathcal{X}_M) = \int_0^1 p(p_k) H(x_k|\mu_{k-1} = \mu, p_k) dp_k \qquad (3.6)$$

Here, $p(x_k|\mu_{k-1}, p_k)$ is a weighted sum of two Gaussians; but $p(p_k)$ is not so easy. Define $q_k = \frac{1}{M}\sum_{i=1}^{M} x_{k-i}$ and note that $I(x_k, \{\mu_{k-1}, p_k\}) = I(x_k, \{\mu_{k-1}, q_k\})$, because p_k is a function of q_k, and $I(x_k, \{\mu_{k-1}, p_k\})$ is maximal with respect to observations of past x. So, we can replace p_k above with q_k, and we know that

$$\begin{aligned} q_k &= \frac{1}{M}\sum_{i=1}^{M} x_{k-i} \\ &= \frac{1}{M}\sum_{i=1}^{M} \mu_{k-1} + \frac{1}{M}\sum_{i=1}^{M} \nu_{k-1} \end{aligned}$$

So q_k is distributed as the sum of $M+1$ Gaussians, each with variance $\frac{\sigma^2}{M}$. The weights of these Gaussians are unknown, but can be estimated from large sample sets. Then we may integrate (numerically in practice) to find

$$H(\mathcal{X}_M) = \int p(q_k) H(x_k|\mu_{k-1} = \mu, q_k) dq_k \tag{3.7}$$

giving us an estimate of the true entropy rate of an RTW$_M$. Estimates for $M \in \{4, 20\}$ can be found in Table 3.1.

We might also ask the question, how easily can one differentiate between two RTW$_M$'s? To answer this we attempt to calculate $D(p(\mathcal{X}_{M_1})\|p(\mathcal{X}_{M_2}))$, specifically for the example of $M_1, M_2 \in \{4, 20\}, M_1 \neq M_2$. We write

$$D(p_{\mathcal{X}_{M_1}}\|p_{\mathcal{X}_{M_2}}) = -H(p_{\mathcal{X}_{M_1}}) - \int\int\int p_{\mathcal{X}_{M_1}}(q_1, q_2) p_{\mathcal{X}_{M_1}}(x|q_1) p_{\mathcal{X}_{M_2}}(x|q_2) dx dq_1 dq_2 \tag{3.8}$$

Unfortunately again, exact calculation of these quantities would require the joint density

$$p(q_1, q_2) = p\left(\frac{1}{M_1}\sum_{i=1}^{M_1} x_{k-i}, \frac{1}{M_2}\sum_{i=1}^{M_2} x_{k-i}\right)$$

which is more difficult to estimate than its marginals described above. Therefore, we make another simplifying approximation – that the two quantities are independent (which they clearly are not, though the approximation improves with larger separation between M_1 and M_2). This allows us to define a random variable which is entropically equivalent to "first-order knowledge" of $\mathcal{X}_M$, namely

$$\zeta_M \sim \overline{p}_M * N(\mu, \sigma^2) + (1 - \overline{p}_M) * N(-\mu, \sigma^2)$$

(where $\overline{p}_M$ is the average probability of switching for an RTW$_M$) and use it to approximate

$$D(p_{\mathcal{X}_{M_1}}(x) \| p_{\mathcal{X}_{M_2}}(x)) \approx p_{\zeta_{M_1}}(x) \| D(p_{\mathcal{X}_{M_2}}(x)) \tag{3.9}$$

We know $H(\mathcal{X}_M)$, and the second term in (3.8) then becomes

$$\int p_{\mathcal{X}_{M_1}}(q_2) \int p_{\zeta_{M_1}}(x) \log p_{\mathcal{X}_{M_2}}(x|q_2) dx dq_2$$

We can approximate $p_{\mathcal{X}_{M_1}}(q_2)$ the same way we did $p_{\mathcal{X}_{M_1}}(q_1)$ above, and the other two distributions are weighted sums of two Gaussians. Again, in practice we numerically integrate to find the results in Table 3.1.

Table 3.1: Approximate Entropy rates and KL-distance for $\mathcal{X}_{M=4}$ and $\mathcal{X}_{M=20}$ (in bits)

M_1	$H(M_1)$	$D(M_1 \| M_2)$
4	-0.589	.556
20	-0.644	.760

Finally, we might wish to have some lower bound on our expected performance at estimating the entropy rate or at the task of differentiation. Because of the symmetry of the signal, an RTW$_M$ of any M will have the same 0^{th}-order statistics, meaning it will be $\pm\mu$ with probability .5, plus Gaussian noise. We might therefore naively attempt to differentiate on the basis of $p(x_k|\mu_{k-1})$. Since $I(p_k; \mu_{k-1}) \neq 0$, we find that $H(x_k|\mu_{k-1})$ is upper bounded by but approximately equal to $H(\zeta)$, where ζ is as defined above. (Except for very small M, this is a less strained approximation than the one in Equation 3.9.) As can be seen from the results in Table 3.2, the results are quite far from the true distributions, and are probably unsuitable for signal differentiation.

Table 3.2: First-order model approximate entropy & KL-distance (bits)

M_1	$H(\zeta_{M_1})$	$D(\zeta_{M_1} \| \zeta_{M_2})$
4	-.400	.054
20	-.548	.050

This indicates that not just any statistic will work; we need to capture information which is present not only in the prior mean, but also in the switching probability in order to discriminate between these waveforms. Intuitively, this is because both types of waveform have nearly the same average probability of switching; since the probability of switching is not influenced greatly by a single sample for large M, we observe periods of infrequent switching until the probability of doing so grows to significance as which point we will observe a number of rapid switches until p has once again fallen. So although given a long window of the past we can easily tell the difference, a myopic data window would be fruitless. This also provides a justification for the data summarization – we have the hopes of achieving good separability with only one statistic, whereas any single observation of the past would be unable to distinguish the two dynamics.

3.3 Learning Dynamics

In this section we will address a few practical concerns for the implementation. Specifically, there are a number of parameters alluded to in Section 2.7. Our notation as we discuss these issues will follow the notation of that section.

3.3.1 Subspaces & the Curse of Dimensionality

We know we can capture the dynamics of our system, and all the information we need, if we can reliably estimate the joint pdf $p(x_k, x_{k-1}, \ldots, x_{k-M})$. Unfortunately, the data required to do so nonparametrically goes up exponentially with M, and therefore the computation involved in evaluating or sampling from such a distribution also increases exponentially. So we must constrain the problem somehow, in order to assuage our computation or data-availability concerns. Normally, this is through selection of a parameterization for the *density* $p(\cdot)$; but as we have already said this may be overly constraining — there may be no parameterized density we are confident will capture the uncertainty. So instead, we parameterize the portion of the *data* used, in that we find a lower-dimensional description of it. This lower-dimensional form describes a subspace near which the data points cluster. Then we can perform our density estimate on this subspace, and achieve a better quality estimate with fewer data.

3.3.2 A form for $\hat{G}$

Before we can do anything else, we must select a parametric shape for the functions of the past we will allow, namely $\hat{G}$. We have already said that $\hat{G}$ will take the form of a multilayer perceptron; so the size of this perceptron is what we must decide. We will first decide the number of output nodes, since this also determines the size of the pdf we must estimate; then we will discuss the size of the network.

We know from the previous section that it is possible to access all relevant information about an RTW$_M$ from its past M data values (with the small exception that μ_{k-1} is not actually observable, but under the assumption that $\sigma \ll \mu$ it is practically unambiguous). We also know that the information about x_k in $\{x_{k-1}, \ldots\}$ is accessible in a compact form; explicitly in the form $\{\mu_{k-1}, p_k\}$. So, in the language of Section 2.7,

$$G(x_{k-1}, \ldots, x_{k-M}) = (\mu_{k-1}, p_k)$$

In the interest of characterization, we decide upon a form for $\hat{G}$ which cannot quite capture this exact statistic. The purpose of restricting ourselves to functions $\hat{G}$ which cannot completely represent the sufficient statistics of the system is to more closely resemble the situations we will deal with later, where the statistics we are searching for, their form, and even how many we require are not known. We wish to show that even in such cases, there is a reasonable expectation that, having simply guessed at these quantities, we can still learn $\hat{G}$'s which capture useful information about the system. By restricting ourselves in a known situation we can gauge our performance loss.

With the goal of demonstrating a degree of such robustness, we select a dimension for $\hat{G}(\cdot)$. The true sufficient statistic is two-dimensional; therefore we will only allow the output of $\hat{G}$ to be one-dimensional. This will put us close to, but still shy of, true sufficiency.

However, in analyzing the true sufficient statistic it can be noted that, since μ_{k-1} is a discrete random variable, and p_k has a limited range ($p_k \in [0, 1]$) if we had a complicated enough function for $\hat{G}$ it would be possible to represent both unambiguously in a single number (e.g. $\hat{G}(\cdot) = 2 * \mu_{k-1} + p_k$). Therefore we choose $\hat{G}$ so this cannot be achieved. It is possible to represent (a good approximation of) this function with a two-layer perceptron; so we again force ourselves to fall short by restricting ourselves to a single-layer network. Finally, we should note again that the true sufficient statistic is actually only *almost* observable – although $\text{sign}(x_{k-1})$ provides an excellent estimate of μ_{k-1}, there is always a small chance (a function of the values of μ and σ) that the noise term is large enough to obscure the value of μ_{k-1}.

So, our form for $\hat{G}$ is:

$$\hat{G}(x_{k-1}, \ldots, x_{k-M}) = s\left(\sum_{i=1}^{M} \alpha_i x_{k-i}\right)$$

where $s(\cdot)$ is a sigmoid function, specifically the hyperbolic tangent (see Section 2.6).

3.3.3 Regularization

Often, machine learning problems can be difficult or ill-posed, with a large number of degrees of freedom or a large number of local minima or maxima. Such problems are simply too unconstrained to ensure that good solutions are found. To assist the learning process, we can add regularization penalties. The concept of regularization is to penalize solutions which are "too complicated" [24]. Examples (for functions such as ours) include encouraging sparseness (few nonzero coefficients), or low-energy. Regularization, of course, introduces its own parameters: what type of penalty to apply and the tradeoff of the relative importance of simplicity to quality of fit. The former we choose to encourage sparseness – an L^1 penalty (fixed reduction amounts); the latter we choose based on experience and experimentation.

One important consequence of using regularization can be illustrated with a simple example. Suppose we have a process $\mathcal{X}$ with $E[X_k] = 0$, and there is no information to be gleaned from $\{X_{k-n_1}, \ldots, X_{k-n_2}\}$. In that case,

$$E\left[\sum_{i=n_1}^{n_2} \alpha_i X_{k-i}\right] = 0$$

and without weight decay, there is no reason for $\{\alpha_i\}_{n_1}^{n_2}$ to change at all, whereas the desirable thing would be for such "useless" weights to be zero. In expectation, of course, it doesn't matter; but such "false dependencies" will increase the variance of our estimator and so affect small-sample performance.

3.3.4 Dataset size

There are several data sets for which we must select required sizes. Each has various pros and cons associated with increasing or decreasing size, and so each must be evaluated separately.

First, there is the overall training data set. This is the set of all points which will be used to train our informative functions. If this set is too small, statistical variations within it can lead to selection of informative features which are actually anomalies; however, larger data sets may not be available, and we may wish to reserve some of our training data for later cross-validation, to estimate new data's likelihood under our model. In this particular case, the size of this set is not a problem; we can simply generate as much as we like.

Secondly, but related to the first, is the size of the dataset to be used at any one time in estimating the joint pdf $p(x_k, \hat{G}(x_{past}))$. Again, if this set is too small, we may train towards anomalously "informative" functions. If it is too large, however, we pay a heavy computational penalty – we evaluate a nonparametric pdf (or rather its gradient) on its own examples and so the cost is quadratic in the number of points used. This set can be chosen to be the complete training data set, or a random subset which changes over time as the learning progresses. In the first case, the best estimate of the information gradient given the examples is always used to update the statistic; in the latter, we use a stochastic estimate whose expectation is equal to the best estimate, but with a variance dependent on the size of the subset we use. It is interesting to note that in addition to the computational speedup gained by selecting the latter method, it may also give the learning process some robustness to local maxima, since there is a chance of mis-estimating the gradient and perturbing away from a local max. Of course, doing so may also slow the learning process.

Finally, there is the size of the data used in the estimate $\hat{p}(x, \hat{G}(x_{past}))$. Once again, each evaluation of the pdf is linear in this size (so that evaluating a length-M process implies MN operations), and all the data which forms it must be saved. This too could be stochastically estimated by only using a subset of the available data at any one evaluation, but the costs of finding a suitable kernel size alone would seem to make this a less desirable technique. Note as well that there is a minimum number of evaluations we can perform when we estimate the entropy rate of a process — we must evaluate a long enough data sequence that the process looks ergodic; else the likelihood will not converge.

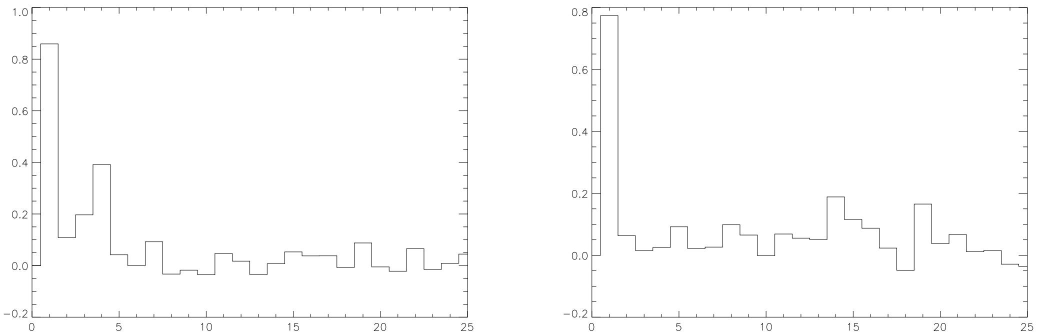
All of these concerns are problem-specific, with no way currently to automatically determine them such that we guarantee good performance. Each induces an intrinsic tradeoff between computational expense and accuracy. Therefore we decide the parameters based on experimental trials, and to some degree based on how long we can stand to wait for the calculations to run: a training data set of 1000 points, of which at any time step of the training process we will use 100 to estimate the information gradient; and a density-estimate data set of 200 points.

3.4 Learned Statistics

We cannot assume that we know, apriori, the length of the past dependence of a signal. In the case of the RTW$_M$, we know that M samples are sufficient, but without knowing M how do we choose a statistic length? In fact, in the future we may not even know that there exists a finite M with that trait. Therefore we would hope that our technique be robust to an incorrect choice of the dependence length N. To demonstrate this, we learn statistics for varying N, on two different RTW$_M$'s.

Our first concern is that we be able to overspecify the dependence length $N > M$ and, in training, discern the fact that there is no more information to be gleaned from past samples. The presence of a weight-decay term will encourage such unimportant data values to be de-emphasized. Figure 3.2 shows the weights of learned networks of size $N = 25$ for RTW$_M$'s with $M \in \{4, 20\}$. As can be seen, there is a strong dependence on the previous value x_{k-1}, which is expected since it provides the most information about the state μ_{k-1}. For small M, $M = 4$, we see that the recent past has much more information relative to the past beyond M; and although for the larger $M = 20$ the dependence length is less clear-cut, the information is obviously more uniformly distributed across a large region of the past than for $M = 4$. So we are not unjustified in hoping to be robust to the presence of some extraneous data.

We are also concerned with our ability to extract information out of too little data, for example the situation where we have under-specified the dependence and chosen $N < M$. In a situation like this, it is difficult to read anything into the selection of weights themselves, since it is difficult to analytically describe the dependence that might be exploited. Thus this situation will instead be examined in Section 3.5.



(a) (M=4)

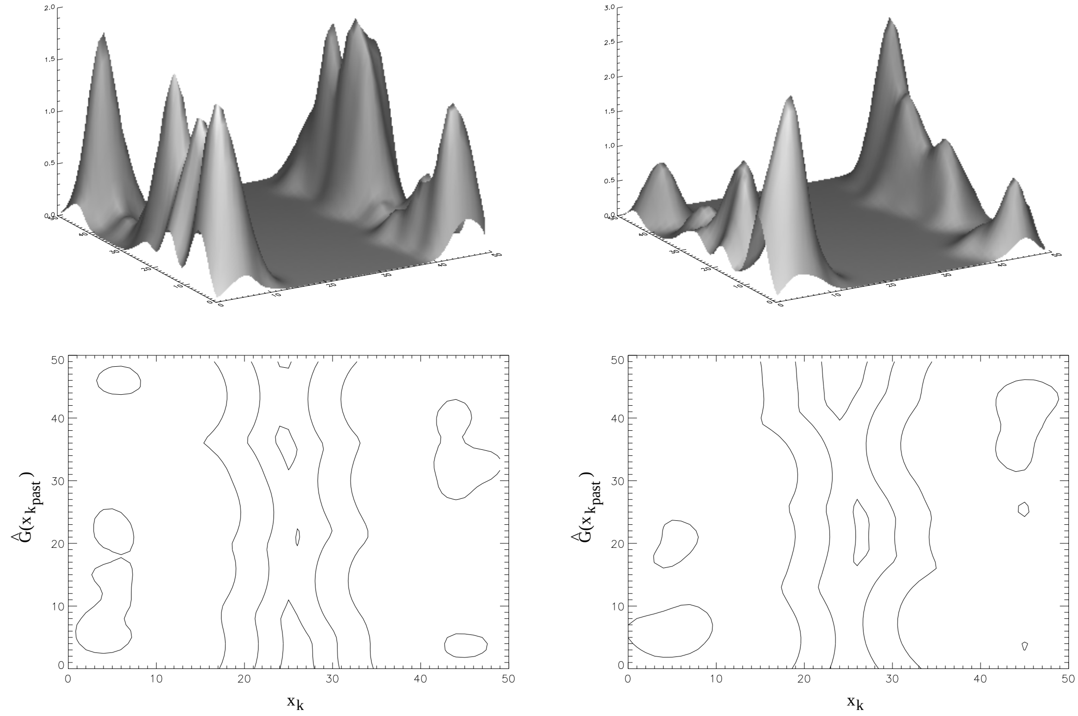
(b) (M=20)

Figure 3.2: Informative functions of the past for RTW$_M$; weights as a function of delay

We can, however, verify that the learned functions do indeed induce low-entropy densities. Figure 3.3 shows the joint densities between the learned statistics and the next data point (the data point being only found near $\pm\mu$. Low entropy, in this case, simply means that for a given statistic value, most of the probability is located on one side or the other, and there are few if any modes directly across from one another.

3.5 Empirical Entropy Rates and Likelihood

In evaluating the performance of our model, it is natural for us to use the likelihood of random telegraph waves as a measure of quality. What we should see is that the likelihood of a matching RTW$_M$ corresponds to our analytically estimated entropy rate for such an RTW$_M$, and that mismatched RTW$_M$ s correspond to our estimated KL divergence between the two processes.



(a) (M=4)

(b) (M=20)

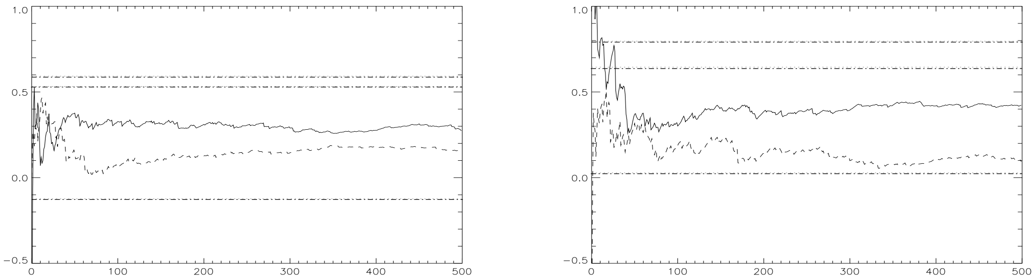
Figure 3.3: Estimated densities $p(x_k, \hat{G}(x_{k-1}, \ldots))$

One way in which we might estimate our model's entropy rate is to simply calculate the entropy of the pdf we model with. Similar to our joint and marginal entropy estimates during training, we can find the leave-one-out conditional entropy estimate of our Parzen window density. In general we should expect this method to underestimate entropy (overestimate likelihood). This is because we are reusing the training data for this estimate, and our training procedure minimizes the conditional entropy for the training set. Also, our kernel size is chosen to maximize leave-one-out likelihood. If we had an abundance of data we might want to improve our estimate by choosing kernel size to maximize the likelihood of a cross-validation data set instead. If our access to new data is limited, however, our original training set may be the best estimate available.

A better estimate of entropy rate can be found by using cross-validation data and taking its likelihood under our model. We can also then take data from another process, and find its likelihood under our model; their difference will be the estimated KL divergence. It also gives a graphical view of an maximum likelihood hypothesis test.

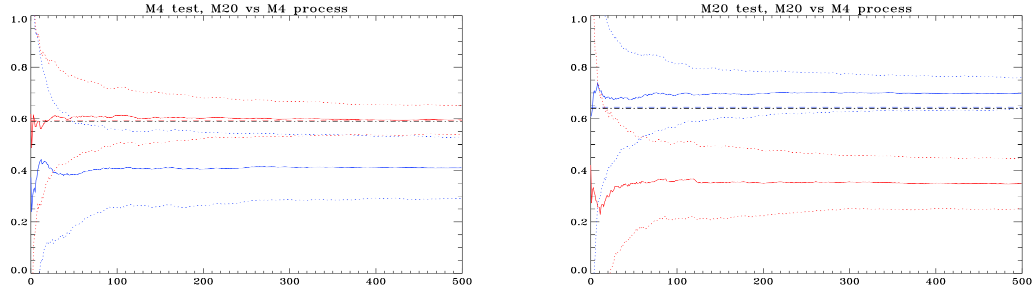
Examples of such estimates are shown in Figure 3.4. These two plots show the accumulated log-likelihood of new data from RTW_M processes with $M = 4$ and $M = 20$, under one of the models $\hat{p}_{\mathcal{X}_{M=4},N=25}$ or $\hat{p}_{\mathcal{X}_{M=20},N=25}$. Dashed lines indicate the negative entropy rate of each process estimated analytically (middle and lowest dashed lines) and the distribution's leave-one-out conditional entropy estimate (top-most dashed line)

Figure 3.4 shows a comparison of likelihoods between different processes under a given model; but if we wished to test which of these two process classes a new sample path belonged to, we would instead compare the likelihood of that path under each



(a) (M=4 model) (b) (M=20 model)

Figure 3.4: Compare process likelihoods under fixed model

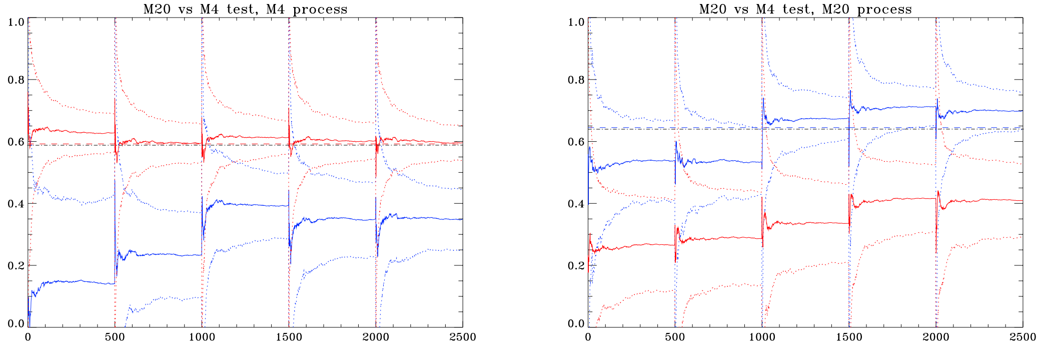


(a) (M=4 process) (b) (M=20 process)

Figure 3.5: Compare a process likelihood under different models

of our two models. In such a case, we would also be concerned with the variance of our likelihood with respect to our data sets, specifically the evaluation data (the samples to be classified) and the data set used for modeling $\hat{p}$. Such a test would take the form of a likelihood ratio, or a difference of log-likelihoods. Figure 3.5 shows the average and standard deviation of 100 of such tests.

Figures 3.4 and 3.5 show the performance of models based on learned statistics when the statistic's past dependence length was overestimated. We would also like to have some idea of the variation of performance as that dependence is changed,



(a) (M=4 process)

(b) (M=20 process)

Figure 3.6: Compare a process likelihood under different models

especially when it is underestimated. To illustrate this, Figure 3.6 shows a composite plot of the same quantities as Figure 3.5, but in five sections as N varies over $\{4, 5, 10, 20, 25\}$, with changes in N occurring at the obvious discontinuities.

3.6 Generative Models

Another use of such a model is as a generator of new sample paths. If our model has truly captured both the dynamics of the system and its uncertainty, we should be able to generate new, plausible data simply by sampling from our conditional distribution.

3.6.1 Comparison to linear filter

We can first get a feel for how well some of our competition might capture the dynamics; we do so by modeling the system in the traditional LQG regime. A straightforward and common method is the Wiener filter (Section 2.3). The underlying model becomes that of a linear autoregression with additive iid Gaussian noise, that is

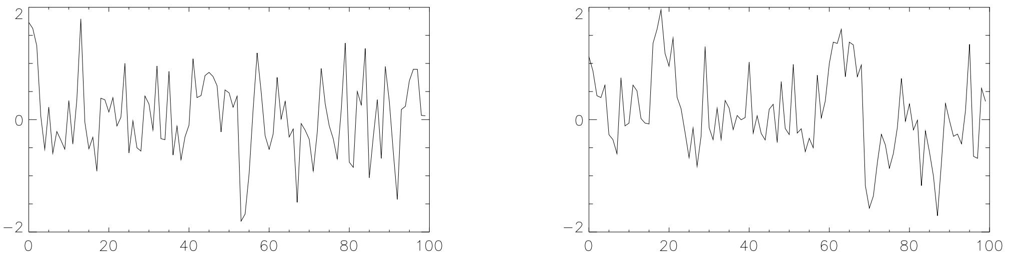
$$y_k = A \cdot [y_{k-1}, \ldots, y_{k-N}] + v_k$$

This bears some similarity to the method above, namely using a linear function of the past observations to give information about the future sample value, and choosing A to minimize the variance of v_k is equivalent to minimizing the entropy of v_k under the assumption that v_k is a iid Gaussian process. The additional capacity we expect from the nonparametric approach is therefore twofold: first, that a nonparametric v_k will be capable of describing much more complex uncertainty than a unimodal Gaussian, and secondly that v_k need not be iid, but can vary its statistics based on (in our case) the value of $A \cdot [y_{past}]$. This allows not only a more flexible description of the uncertainty (which for a RTW is obviously not identical at all time steps) but also frees us from the role of $A \cdot [y_{past}]$ as a *predictor* of y_k.

Because we know that this system does not exhibit a linear *predictive* dynamic, we do not expect to be able to do as well under such assumptions. In fact, we can expect predictions which are in fact highly unlikely; for instance when the probability of switching is .5, an MSE predictor will select x_k near 0, a value which will rarely occur in the true waveform. Such problems are the cause of synthesis paths which are atypical and thus “visually dissimilar” to the true system (see Figure 3.7).

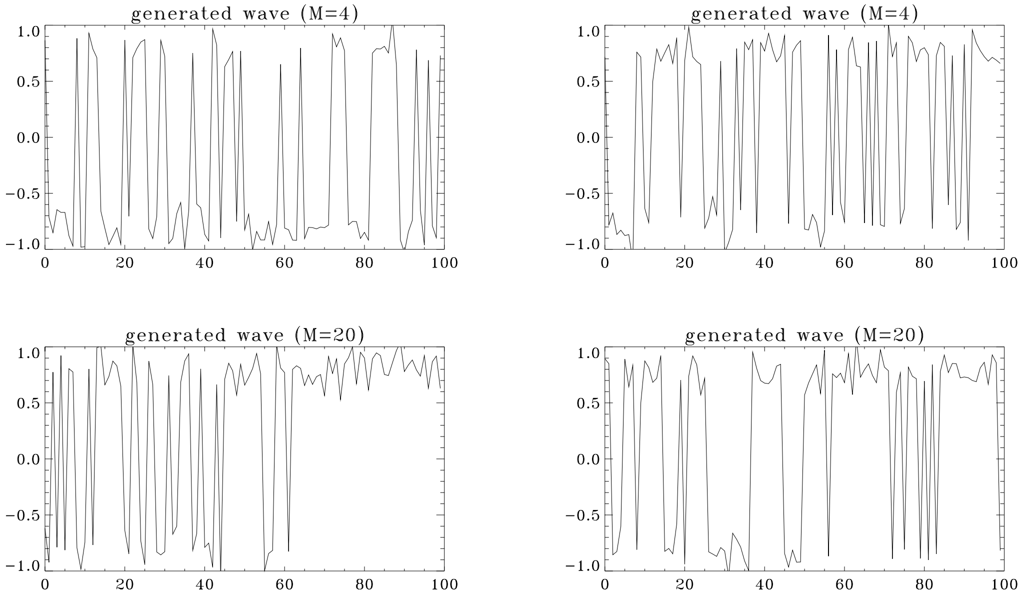
3.6.2 Synthesis paths from a nonparametric density model

As was discussed in Section 2.2, analytic characterization of the quality of synthesis is a difficult prospect. Therefore we present example telegraph waves sampled from our



(M=4) (M=20)

Figure 3.7: Sample paths realized from the Wiener filter



(N=4) (N=25)

Figure 3.8: Synthesized sample paths with over- and under-complete information

learned models in Figure 3.8 for visual inspection and evaluation. These examples show a comparison between paths where the model's length was over- or under-estimated.

As we would expect, the quality of the synthesized waveform is higher in the case when $N > M$, meaning that all the information present in the past of the signal was available for use. Notably, even the $M = 20$, $N = 4$ case possesses qualities associated with a true $M = 20$ random telegraph wave, although it also possesses sequences which we are unlikely to see in true random telegraph waves (e.g. $\mu_k = +\mu$ for many more than M samples). This is simply another indication that even when not all possible information is accessible, useful information is still contained in the partial past, and is extracted by the statistic.

3.7 Conclusions

We can see that, at least for our synthetic example, we are able to extract informative statistics from the past. Given a sufficient window of the past, we are capable of capturing most of the information present, even though we are not capable of representing the system's sufficient statistics exactly. Some confidence in this will be necessary later, when we are not sure of the form or even existence of sufficient statistics. We also seem to have some robustness to over- and under-estimating the required window size; when overestimated, the extraneous inputs were de-emphasized, and when underestimated the performance suffered (as it must) but still, clearly some useful information was extracted. This will be important in later problems, when we do not know the scope of the future's dependence on the past.

With respect to this particular dataset, we are able to show that we have captured much of the nature of the uncertainty in the system, despite its nonlinear relationship. The hypothesis tests constructed using the models' likelihood showed good differentiation between the two types of processes, and the entropy rate of the process could be estimated giving the possibility of a single-hypothesis test (accept or reject as a process $\mathcal{X}_M$).

Chapter 4

Real-World Data: Laser Intensity

We now turn to a real-world time-series, for which it will be more difficult to characterize the results of our model but which will highlight a number of issues which were not apparent in the simpler random telegraph wave process. In this case we are unable to write down a sufficient statistic for the system, nor is its intrinsic dimension known. Therefore, we use entropy estimates to gauge our performance as we increase the number of learned statistics and so estimate the process dimension. Synthesis results are shown and discussed, and the versatility of the nonparametric density for capturing varying uncertainty is demonstrated.

4.1 Time series description

The data set in question is from the Santa Fe Time Series Competition; specifically data set A. It is a discretization of NH_3 laser intensity measurements. This system of

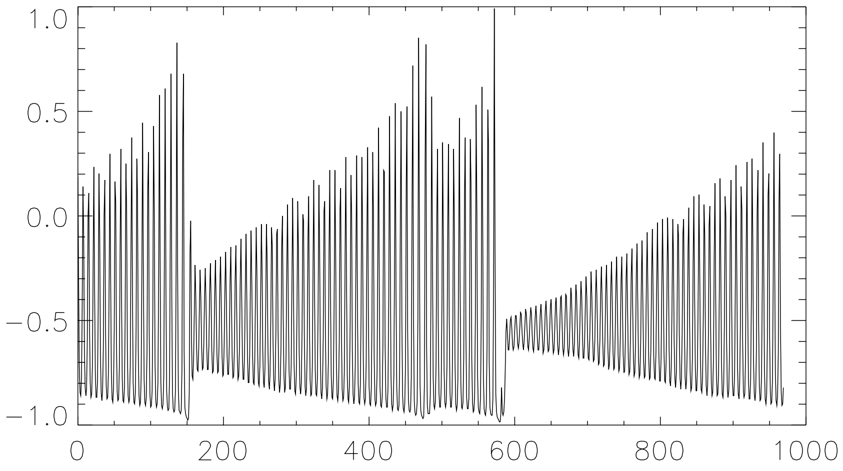


Figure 4.1: Laser intensity time series training data

"spontaneous periodic and chaotic pulsations" is best approximated analytically using nonlinear differential equations, specifically the Lorenz equations and the Lorenz-Haken model [25]. The data available for training and model estimation consists of 1000 points of this time series, shown in Figure 4.1. Features immediately of note to a human observer are the linearly amplifying, near-periodic oscillation with "random" collapses. We have chosen to restrict ourselves to the same available data used in the actual competition in order to preserve the standard of the original challenge. Much of the difficulty in this time series comes from the relatively small volume of data available in comparison to the scale on which interactions take place. Therefore, to deviate from the confines of the competition by e.g. synthesizing more data for training by simulating the Lorenz-Haken model might undermine the quality of the data set for evaluative purposes. In addition, situations with small data volumes pose an important test for nonparametric estimators, which will fail if not given a sufficient number of examples.

4.2 Learning multidimensional statistics

For this process (like any real-world system) we do not know the true underlying dynamics, and so we cannot determine the form or even existence of a sufficient representation of the past. We do not even know the minimum dimension of such a representation, and so we will be required to estimate it. Since the process clearly involves an oscillation of varying amplitude, we can hypothesize that at least two statistics will be necessary to capture the information in both quantities, and so it will be necessary to discuss the learning of a vector-valued $\hat{G}$ and how one might select its dimension.

As it turns out there are some difficulties in simply performing the learning step simultaneously with multiple statistics. First of all, doing so increases the number of local minima which any gradient-based algorithm can become trapped by. Also, it was observed that two statistics would sometimes appear to "compete" for the same information, reversing gradients often and slowing convergence. Thus for the moment we chose to implement a greedy, sequential training algorithm in order to further constrain the learning problem. This algorithm adapts the first dimension to capture as much information as possible, at which point the next dimension attempts to absorb only the remaining information. At each step we train the i^{th} dimension of $\hat{G}(\cdot)$, denoted $\hat{G}_i(\cdot)$, to maximize the conditional information $I(x_k; \hat{G}_i(x_{past})|\hat{G}_{1\ldots i-1}(x_{past}))$. In theory this procedure of conditioning on already learned dimensions when training each subsequent dimension will eventually capture all available information. However, it may not do so with the minimum dimension possible. Just as we elect to use a simple function for $\hat{G}(\cdot)$, trading some representational power for ease of training, we risk some suboptimality to decrease the ill-posedness of the learning process.

Use of such a sequential method may also change the form of $\hat{G}$. In the case that each output of $\hat{G}$ is a single-layer network there is no difference; but if $\hat{G}$ has multiple layers it must take the form of several parallel multi-layer perceptrons (increasing the total number of parameters in the structure). Alternatively one could restrict subsequent statistic training to updates of their final layer only; but if the required information has already been lost as irrelevant to the original output in a previous layer such a system will be unable to retrieve it (this is the data processing inequality at work). In our case, however, we have restricted ourselves to single-layer networks and so we need not address these ramifications in full.

In Chapter 3 it was demonstrated that even when the true dependence of the signal on the past is longer than the statistic allows, useful information can nevertheless be extracted. In the absence of more information about the signal's dependency we simply hypothesize a length to capture all the necessary information. Figure 4.2 shows the results of sequential learning of statistics; the weights are plotted as a function of delay time. They support our original intuition — the first two statistics capture the oscillatory nature of the process, since together they are capable of discerning both phase and amplitude. All of the functions are orthogonal, as well. But there is some subtlety here as well — although the weights are regularized, they do not utilize only one oscillatory period; this is probably related to the fact that the signal is not perfectly periodic. Additionally, as we shall see later, two statistics are *not* enough to capture the full dynamics.

How many of such statistics do we need in order to "reasonably" capture the system dynamics? The answer to this question depends strongly on the desired application for which the model will be used. However, to give us an idea of the amount of information we have captured, we can graph the resulting entropy rate estimates (using Equation 2.8) after learning each statistic. Such an estimate can be seen in

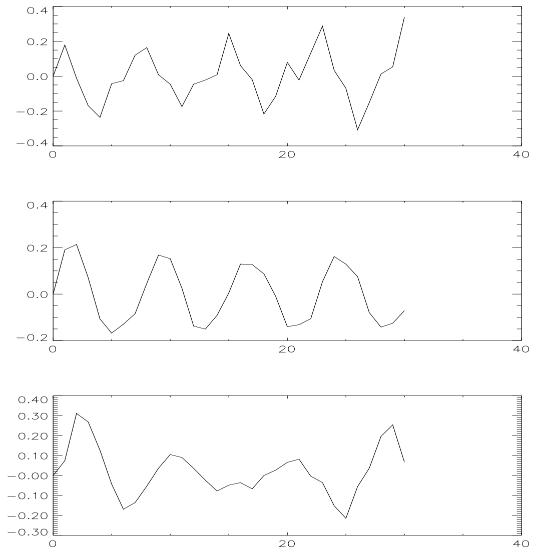


Figure 4.2: First three learned statistics for the Santa Fe laser data (weights versus delay time)

Figure 4.3. There is an evident knee in this curve, indicating that at three statistics we have ceased to improve our model. Although this does not necessarily mean that our model has *fully* captured the system dynamics, it does indicate that the training algorithm has ceased to be able to extract more information. As this is real-world data, neither we nor any method can definitively determine the intrinsic dimension of the process; but the plot certainly gives good indication of the dimensionality our method will require.

4.3 Multivariate Density Estimation

Because we have determined to learn multiple statistics, we must now perform our density estimate on a higher-dimensional p.d.f., a considerably more difficult task. In

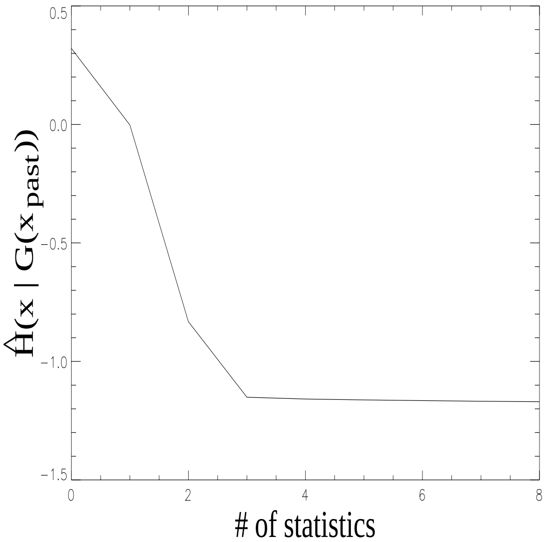


Figure 4.3: Conditional entropy with increasing statistic dimension

this section we will address some of the problems which can arise and the application of techniques which were not necessary in the preceding chapter.

The problem of kernel size selection becomes much more difficult in high dimension. Although in the "large-data limit" a global, spherical kernel will accurately estimate the distribution function, the computation costs associated with nonparametric estimators means that often we cannot afford to use the volume of data which would be necessary. In many cases, including this one, the limited availability of data precludes the attempt even if the cost were acceptable. Because of this, any inhomogeneities of the distribution function can be best accounted for by our choice of kernel size.

In general, optimal kernel choice (size and shape) is an ill-posed problem. Finding good methods and evaluating their performance is an open area for future research. In order to use kernel density estimates we must choose a method, and so we evaluate a few empirically.

The first possible solution is to use a kernel with a directional shape and size, as discussed in Section 2.4.1. Normalizing the variance in each dimension as [9] may help, but in truly inhomogeneous densities, for example multimodal densities, this approach may not address the problem. A multivariable search over the kernel size in each direction may be able to capture such shape-related estimate improvements, but is a relatively computationally intensive task. It may also be that the optimal kernel shape is regionally dependent; but finding such regional shapes is an open problem [19].

As was mentioned in Section 2.4.1, another possibility is to use a fixed shape but vary size over the space. Using such a variable kernel, with each point affecting an area proportional to its neighborhood's density, can avoid oversmoothing which would otherwise adversely affect the estimate. Again, required quality depends on the use to which this will be put; it may be that the likelihood estimate will be relatively unaffected by oversmoothing (although its usefulness for discrimination might not be!) but for synthesis, accuracy in low-entropy regions may be more critical. The advantages of a local kernel size can be demonstrated by observing the marginal distributions over two dimensions of the statistic learned on the laser intensity data. The data points themselves can be seen in the scatter-plot of Figure 4.4; notice that the regions of the interior of the plot exhibit low-entropy, while the outer ring's entropy appears to be higher. The ML estimates of kernel size, both global (left) or variable (right), are shown in Figure 4.5. Notice that the presence of randomness in the outer region has caused considerable oversmoothing of the inner, low entropy region for the global case. A complete comparison of all pairs can be found in Figures 4.7 and 4.8.

It turns out that these spirals are related to the phase of the time series, and that at low amplitude there is very little ambiguity in the phase of the next point. Therefore any synthesized waveform should display the same kind of structure, and oversmoothing will produce a spurious randomness effect. (It also appears to drift slightly after each collapse; this and the fact that we have only two complete examples causes the visible bifurcation along the spiral arms of Figures 4.4 and 4.5(b). However, it seems unlikely that any distribution estimate would capture this drift from so few examples.)

One test of whether this visual improvement in the kernel density estimate has, in fact, improved our model is evaluation of the likelihood of the sequence's true continuation (which was not used for training). We compare the accumulated one-step-ahead log-likelihood of this data in Figure 4.6. The likelihood under a local kernel model is shown as solid; the global kernel as dashed. By improving our estimate of uncertainty to more accurately reflect the process, we have increased the likelihood of the evaluation data. This agrees with our visual comparison, that the locally tightened estimate agrees with the true system's distribution better than the globally determined version.

It is also worth noting that, although the kernel size is known to be of critical importance in terms of estimating likelihood or sampling, empirically it appears that it is of less importance for estimating the entropy *gradient* for learning purposes. It does have an effect, in that learning from identical initial conditions and data with slightly different fixed kernel sizes do not always result in the same feature selection, but did not appear to produce statistics which were significantly more or less informative. Philosophically speaking, relatively larger kernels for learning may be better, since intuitively it seems that a larger kernel size should correspond to "larger-scale" similarities and differences. Still, it is difficult to speculate on the exact effects of a change in kernel size. An in-depth analysis of the exact effects of

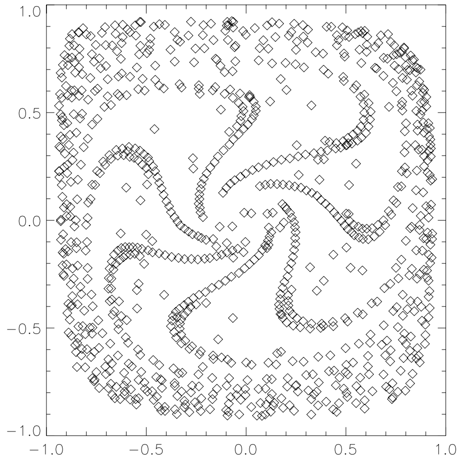
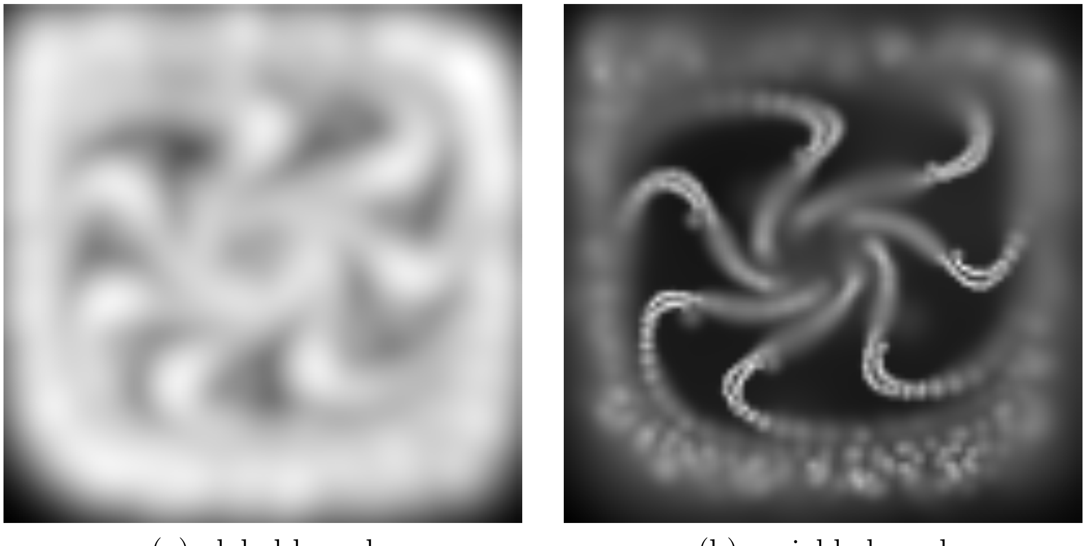


Figure 4.4: Scatterplot of training data 2-d marginal



(a) global kernel

(b) variable kernel

Figure 4.5: Effect of kernel size choice on density

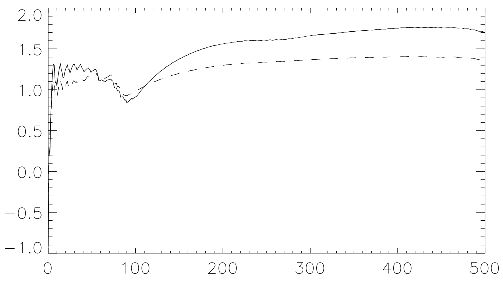


Figure 4.6: Accumulated 1-step-ahead log likelihood for true continuation under local (solid) and global (dashed) kernel size density estimates

this choice on functions of the density, such as the information gradient, is an open area of research and beyond the scope of this thesis.

4.4 Synthesizing new sample paths

A visual test of whether we have captured the dependencies and randomness of a signal is the similarity of a generated sample path and our human expectations. Failing to capture the true dependency structure or ascribing too much randomness to the system will result in sample paths which fail to capture the long-term patterns of the signal; ascribing too little randomness results in repetitious patterns with little or no deviation from the observed data set.

A sample path generated with three statistics and the ML variable kernel size density estimate can be compared to the true continuation and to other synthesis results in Figure 4.9. Notice that statistics of sufficient dimension (3) have captured the long term structure (overall shape, and increasing oscillation with collapses) of

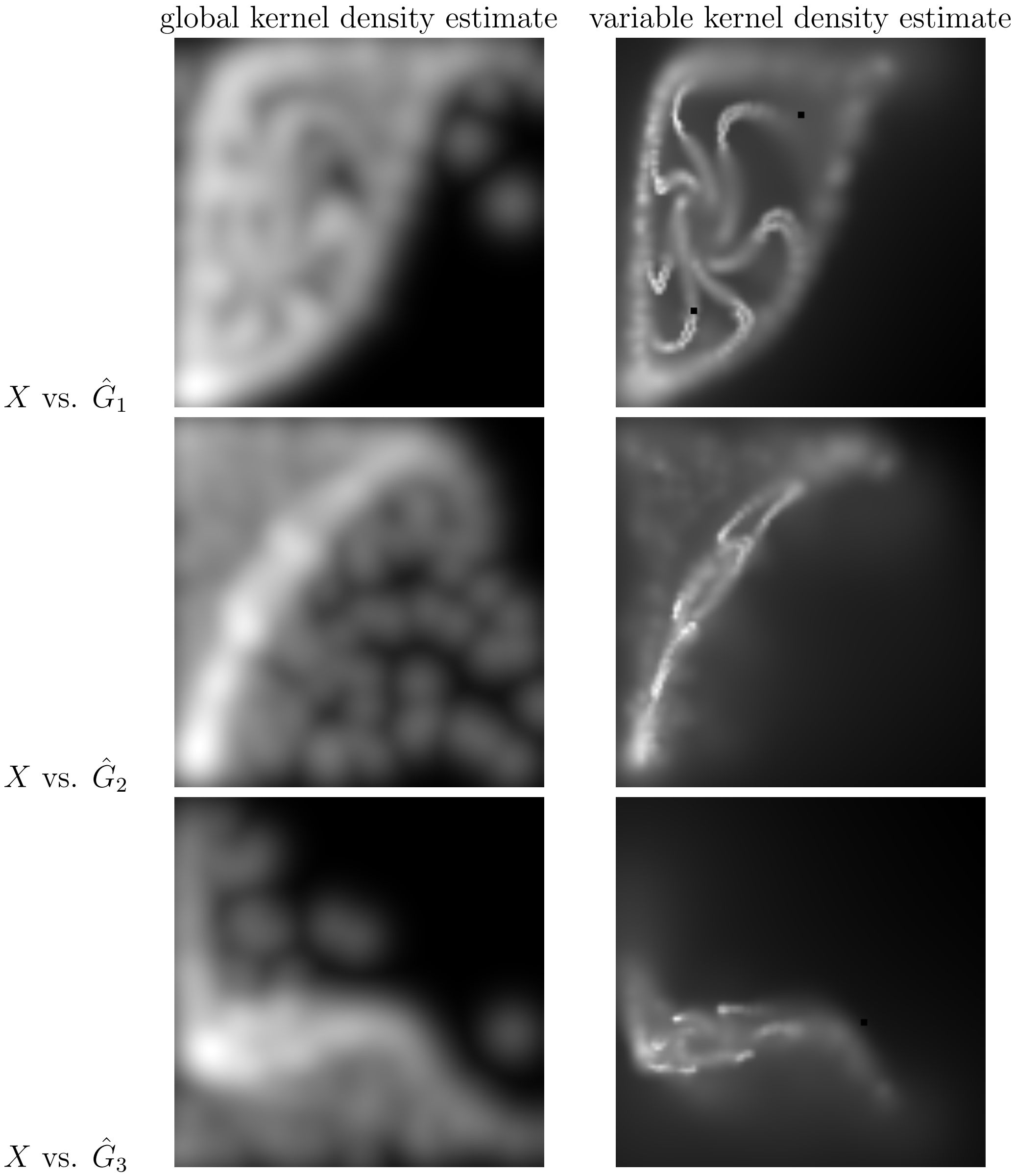


Figure 4.7: Joint densities of x and each dimension of $\hat{G}$, with global vs. local kernel size choice

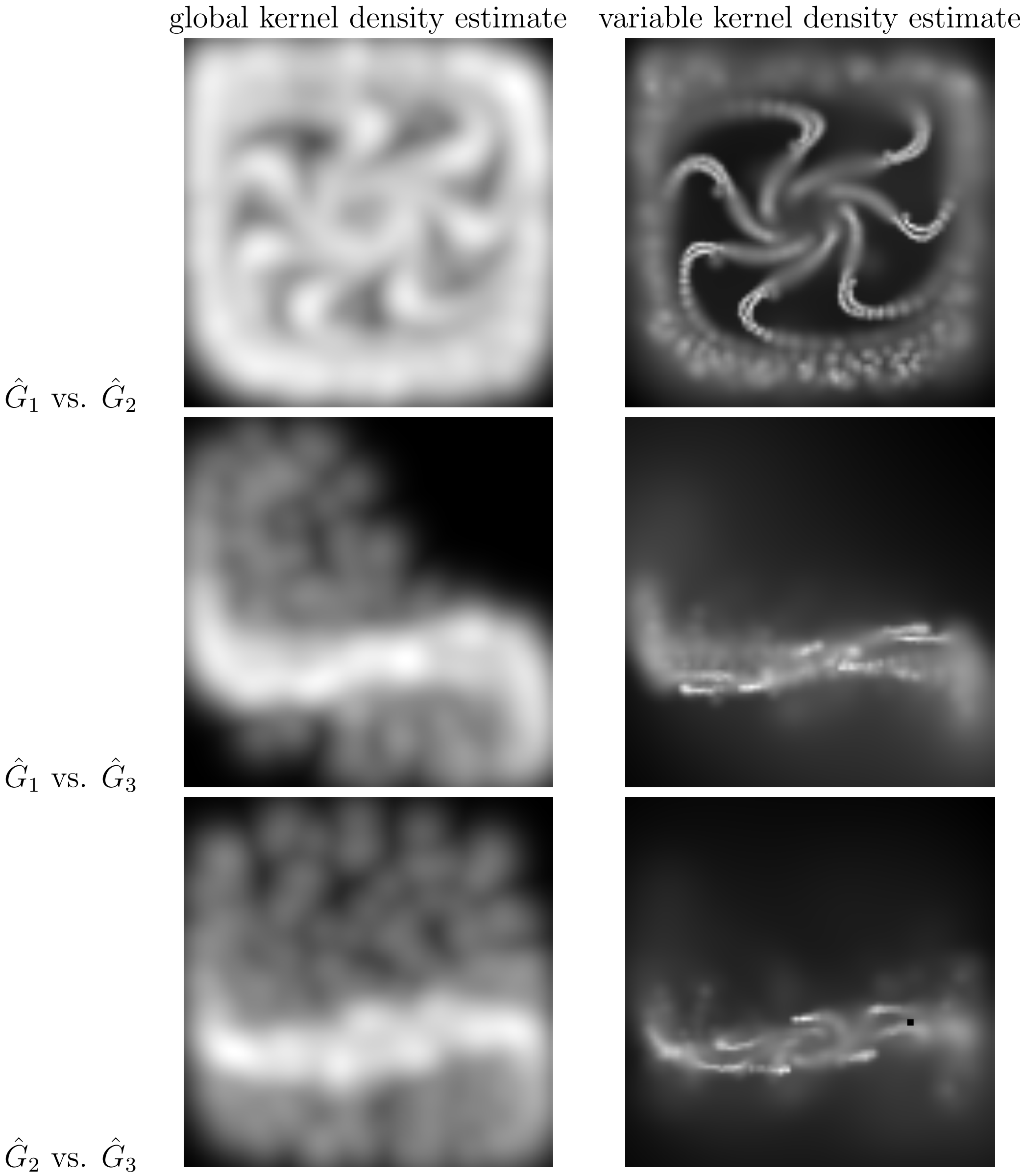


Figure 4.8: Joint densities between dimensions of $\hat{G}$, with global vs. local kernel size choice

the signal, in contrast to methods with less relatively sufficient statistics (model of dimension 2 or linear model).

Notice that the two-dimensional statistic is unable to reproduce synthesis paths which conform to our expectations of the signal. This is an effect of the insufficiency of that statistic. Because we have learned the statistics sequentially, both features of this model are present as $\hat{G}_1$ and $\hat{G}_2$ of the 3-dimensional model. Except for a difference in kernel size (which empirically is slight) the distribution used is the same as $p(X, \hat{G}_1, \hat{G}_2)$ as depicted by the marginals in Figures 4.7 and 4.8. Thus, the additional information gained by $\hat{G}_3$ can be gauged by the remaining plots: $p(X, \hat{G}_3)$,$p(\hat{G}_1, \hat{G}_3$, and $p(\hat{G}_2, \hat{G}_3)$. These plots indicate that the increase in information from $\hat{G}_3$ is probably less than that from $\hat{G}_1$ or $\hat{G}_2$, since most of the probability appears to be clustered in the a small range of values of $(X, \hat{G}_1, \hat{G}_2)$. However, there is some visible structure present, indicating that there is still some information in $\hat{G}_3$. This intuition is corroborated by both the entropy estimates in Figure 4.3 and the synthesis results of Figure 4.9.

If instead of new sample path synthesis we wished to pursue a predictive approach, we could select the ML prediction of each point given the previous points. Note that such an approach is not necessarily the ML sample path, since the entire path is not selected in a joint manner; such a joint selection would be quite computationally costly. It is also not the most likely sample k steps in the future, since it makes use of the previous selections for a one-step-ahead prediction; this too would be very costly. A point by point, one-step-ahead ML prediction is feasible, if somewhat more costly than mere sampling (see Section A); such a sample path is shown in Figure 4.10. Notice that it does *not* capture the structure of the observed data; this indicates that in the trained model its collapsing structure has been (perhaps correctly) attributed to the randomness inherent in the model's form. This also illus-

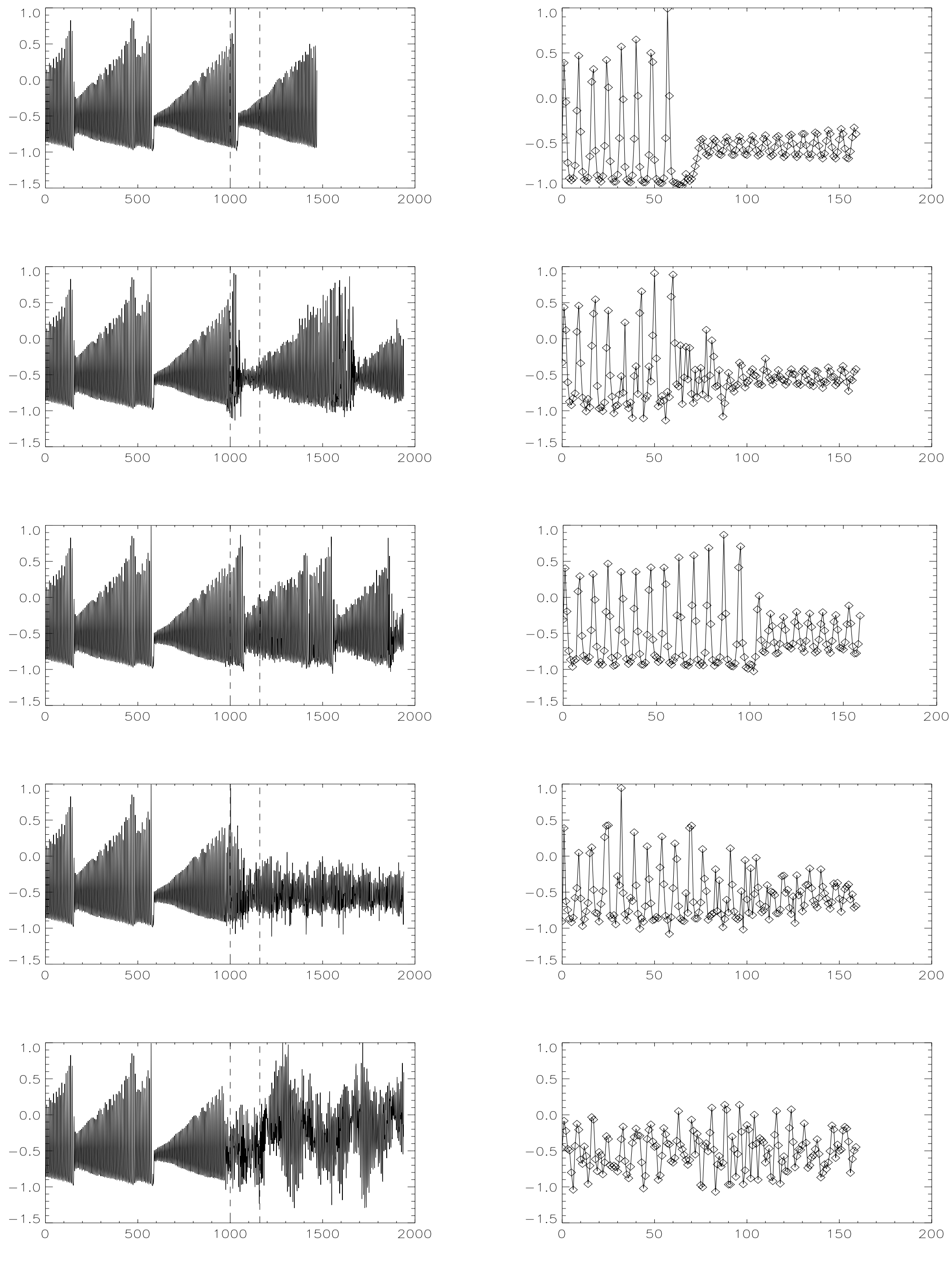


Figure 4.9: Comparison of true continuation data (top) with synthesized paths from (in order): 3-dimensional statistic with local kernel, 3d stat with global kernel, 2d stat with local kernel, and a linear filter.

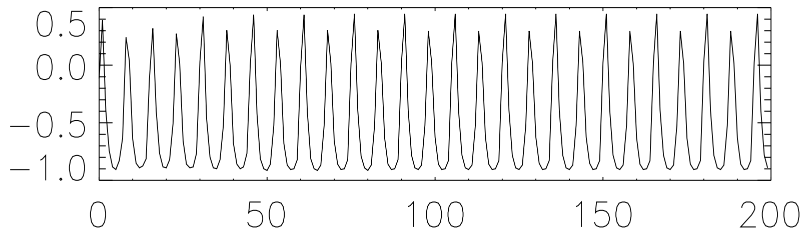
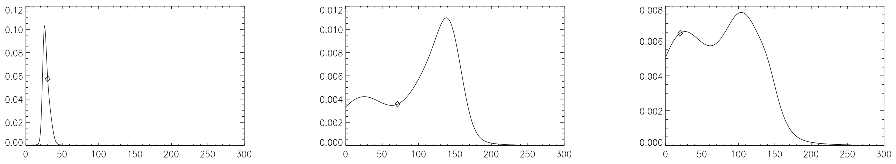


Figure 4.10: One-step-ahead ML prediction path

trates one of the points made earlier, in Section 2.2 – although this is the most likely path, it is atypical, and does not exhibit the dynamics we expect.

We can see some of the flexibility of the nonparametric density by examining the conditional distribution $p(\mathcal{X}_t|\mathcal{X}_{past})$. Specifically, we show the conditional distributions induced at various points by the continuation data (as if we were observing the continuation one sample at a time). Such densities provide us with a view of the variable entropy over portions of the signal. Examples of the conditional distribution at three different times are shown in Figure 4.11. These illustrate several different possibilities for the distribution: relatively certain (low-entropy, (a)), very uncertain (high-entropy, (b)), and bimodalities (c). The true datum associated with each of these three distributions is also shown. The third plot (c) actually shows the conditional distribution just as the collapse occurs. At this time sample, a normal oscillation would continue in an upward swing, but instead the waveform collapses. We can see in the distribution that this has been described by a bimodality, indicating that model believes a collapse is not guaranteed, but neither is it completely unlikely.

We also can look at the progression of such distributions as a function of time.



(a) low-entropy (b) high-entropy (c) at the collapse

Figure 4.11: Example conditional distributions of the continuation data; diamond indicates true datum

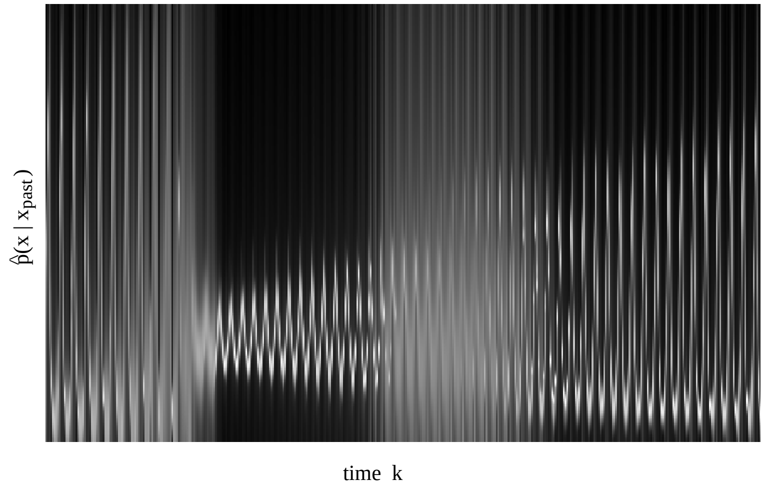


Figure 4.12: Conditional distribution of continuation data over time

Figure 4.12 shows $\log p(\mathcal{X}_t|\mathcal{X}_{past})$, represented as an intensity image (black = low probability). From this image it is clear that the model has attributed a high degree of uncertainty to those regions with high amplitude, and much less in the lower part of each oscillation. This makes sense, because as was mentioned above, the model has associated the collapse with a random event, and so indicates (through high-entropy conditional distributions) that it cannot accurately predict the next sample in this region. After the oscillation has settled into a low-amplitude pattern the uncertainty shrinks again.

4.5 Conclusions

We have shown that it is possible to apply our framework to model a real-world system, either estimating or simply guessing such quantities as past dependence length and minimum required dimension. The fact that these quantities will be unknown in almost all real-world systems makes it very important that any method be able to cope with their possible misestimation, and perform as well as possible under the circumstances.

We showed that it was possible to learn multi-dimensional statistics, in order to capture the behavior of systems with more complex behavior than that seen in the previous chapter. We also demonstrated that we can use the estimated conditional entropy of the model to gauge improvements through increasing subspace dimension. This gave us a clear indication of the information loss due to a given level of approximation, and a stopping criterion for learning additional statistics. We also demonstrated the improvement in synthesis quality which resulted from using a statistic which was closer to sufficiency.

We discussed the importance of the density estimator for likelihood-based discrimination and especially for sample synthesis. We then implemented a technique which appeared to improve the density estimator considerably, through using a local adaptive kernel size to capture anisotropies in the data. This improved density estimator also resulted in increased likelihood for the continuation data, indicating that the improvement was indicative of the true uncertainty structure.

Finally, we evaluated how well we had managed to capture the dynamics of the system through new sample path generation. It was seen that, for a sufficient dimension of the statistic (3), we were able to capture both the short and long-term structures of the signal, as compared to a simple linear filter or even a nonparametric model with fewer than the required number of dimensions.

Chapter 5

Handwriting and Signatures

A person's handwriting, and especially their signature, is very consistent — so consistent it can be recognized with a high degree of accuracy by most people, given only a small set of examples. As such it has comprised one of the world's oldest "secure" forms of identification.

We can consider the time-series of a stroke of handwriting as a two-dimensional dynamical system. For simplicity, we will remove the uncertainty of the text from this description, and limit ourselves to fixed-text strokes, of which signatures are the most obviously applicable subset. It is possible to imagine that such a system is locally linear, but likely not be globally linear. Its uncertainty is hard to describe analytically, but there is no reason to think it Gaussian — for example, our signature variations are not a "mean signature" and some additive independent randomness; there is uncertainty in the overall shape of each letter. The uncertainty at any given point may easily be bimodal — for example, near a reversal of direction there is probability in both the forward and reverse directions but no probability of simply stopping. In fact, it is arguable that such a system is not even stationary. We will discuss each of these points in more detail later, but suffice it to say that to attack this problem with a conventional, linear approach would be quite difficult.

To this end, we apply the same techniques we have been developing in the past chapters. By learning features of the data which are maximally informative about the state, and nonparametric descriptions of the inherent uncertainty in that state, we attempt to capture the dynamics well enough to synthesize plausible new signatures, or develop an accurate test of the plausibility of a new signature. The models in this section were trained using only eight example signatures, thus showing that the method can be effective without a large database for comparison.

5.1 Signature Synthesis

Although the commercial applications of a method for synthesizing signatures may be more limited than discrimination, we can still find uses for sample path generation. For instance, our ability to use the model to generate plausible new signatures will be indicative of how completely we have captured the dynamics. Also, in many discrimination applications it is desirable to have an underlying generative model so as to be able to overcome challenges such as missing data or variable-length sequences [12].

5.1.1 Sampling versus Prediction

New signature path realization is a situation where the differences between sampling and predictive methods such as a nearest-training-example or max-likelihood selection becomes clear. Given a set of example signatures, if we merely choose the most likely sample, the example whose past most closely resembles the current past without adding new uncertainty, it is quite likely that we will simply regenerate one of the signatures from our database. Perhaps this produces a plausible new signature, but it will usually produce the *same* signatures, much the same as simply selecting one of our original examples. The ML path, while perhaps different from any example, will also be deterministic. Clearly, while both these are by definition plausible signatures, this does not meet the goal of signature synthesis. Conditional distribution sampling, then, is a more logical course of action.

5.1.2 Coordinate System Choice

Inherent in every algorithm but often overlooked is the fundamental question of a coordinate system for the data. For example, audio recognition algorithms may be simpler in the frequency domain than as a sampled time series. Indeed, we have successfully overlooked this issue on both previous examples; but in this case it behooves us to examine it.

This is not to say that we must search for the perfect coordinate system. Indeed, there is a coordinate system in which any given problem becomes trivial — for example, suppose we have a solution and we use it to define a coordinate system. Then, in these coordinates, our task becomes simple. But such a statement is hardly helpful in finding a solution in the first place. We merely note that some coordinate systems may be better than others, and would like to ensure that we use a "good" one.

Essentially, good coordinate systems are those in which we have thrown away irrelevant portions of the data, and collected together helpful information. For example, transformation to a frequency domain collects together information about frequency bands (which was of course present in the original data, but less easily accessible) and allows us, if we wish, to throw away portions we consider unimportant, for example high frequencies. We want a transformation which will collect useful information – just like our formulation of finding informative statistics. So why worry about choosing a coordinate system – why not simply let our learning algorithm find such statistics for us?

The answer is simple – one should always use any apriori information available to simplify before turning a machine-learning algorithm loose. The parametric form of our statistics determines in part the shape of all reachable functions of the data. In order to include a particular kind of transformation (e.g. Fourier coefficients) the form for $\hat{G}$ might need to be very complicated indeed. The more complicated $\hat{G}$ becomes, the more difficult it is to learn — more local maxima, slower training, etc. If we can simplify this form to a fixed transformation followed by a $\hat{G}$ with fewer free parameters, we make the learning procedure easier.

Use of such a transformation allows us to apply any apriori knowledge of what information is *useless* and expunge it, collecting useful portions of the data together as best we can. In many problems, we enter with some ideas and intuition about the problem, and our own expectations of what will work and how well. A good algorithm should provide means of incorporating such prescience in a reasonably principled way;

and one such way is through this choice of coordinate systems.

Of course, the possibilities for such a choice are nearly endless, and so in the interest of brevity and our general desire to retain a degree of automation to the system, we will restrict ourselves to a few simple coordinate choices. They will stem from the most obvious features of the data — the (x, y) coordinate system, a differential $(\Delta x, \Delta y)$ system, and the feature of fraction of time elapsed. In the interest of preserving some degree of smoothness, we will use these features to predict a differential update; predicting actual (x, y) positions behaved similarly but appeared less “plausible” due to discontinuities. Specifically, the three coordinate systems we shall use for observations are:

- a time-series of standard (x, y) locations

$$(\Delta x_k, \Delta y_k) \sim p((\Delta x_k, \Delta y_k)|G([x_{k-1}, y_{k-1}, x_{k-2}, \ldots, y_{k-N}]))$$

- a differential time-series $(\Delta x, \Delta y)$

$$(\Delta x_k, \Delta y_k) \sim p((\Delta x_k, \Delta y_k)|G([\Delta x_{k-1}, \Delta y_{k-1}, \Delta x_{k-2}, \ldots, \Delta y_{k-N}]))$$

- and a differential system augmented by the time elapsed thus far,

$$(\Delta x_k, \Delta y_k) \sim p((\Delta x_k, \Delta y_k)|G([\Delta x_{k-1}, \Delta y_{k-1}, \Delta x_{k-2}, \ldots, \Delta y_{k-N}, k/L]))$$

where L is the total number of samples in the signature, and

$$(\Delta x_k, \Delta y_k) = (x_k - x_{k-1}, y_k - y_{k-1})$$

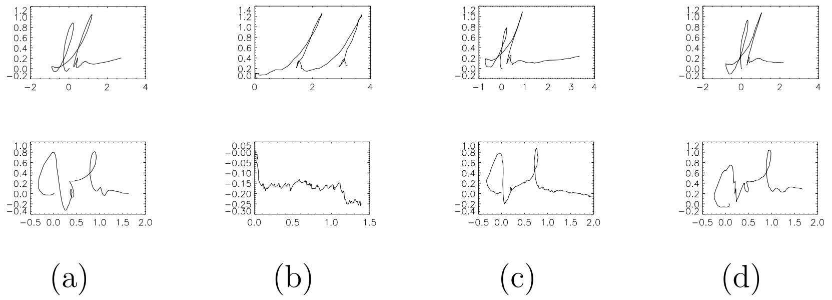


Figure 5.1: (a) Example signature; (b) synthesis using only local (dx, dy) information; (c) local (x, y) information; (d) local (dx, dy) augmented by time

In our implementation we chose $N = 10$; most signatures were 150-200 samples total in length, so features of only 10 samples means they encapsulate only local stroke information. An example signature and synthesis results for each of these three cases can be seen in Figure 5.1. A more sizeable data set is found in Figure 5.2.

One can see the effect of the coordinate system choice in these synthesis paths. The first synthesized path (column b) used only differential information and was usually of unacceptable quality. Sometimes it was unrecognizable, and at other times it would seem to begin a signature but lacked enough context to unambiguously discern its current position within the word, and can repeat sections (such as the double-h in Figure 5.1(b)). This indicates that transformation into that coordinate system has lost essential information, and the model is attributing deterministic elements of the dynamics to uncertainty. The missing context can be restored in more than one way — when we give the statistics access to the exact (x, y) position, it is much more capable of discerning its current position within the stroke; or, one can add nearly the same information by forcibly augmenting the statistics by a value indicating the percentage of time elapsed. The former approach tends to do well at the beginning, but becomes less plausible toward the end of the stroke. This is because we always begin at the origin, but do not always end a signature near the same point. Thus, the accumulation of drift within the stroke has caused there to be less data available, or more precisely larger data variation, near the end of the signature than at the beginning. It is also possible for the (x, y) data to become confused when a stroke crosses over itself with a similar dynamic, such as the 'o' in the name "John". Such problems do not occur when given elapsed time, since the beginning and end of the 'o' are clearly differentiated in time, but there is still some context missing, which will cause effects such as not following a straight line.

This "missing context" is actually a deep issue, stemming from the basic fact that we have violated our original assumptions with this data set. Namely, we have assumed that our time series is a stationary process, but a signature is not — its dynamics are very dependent on the written text and on the current position within it. In a signature, each value of the state will only occur exactly *once*. Yet, by using several examples we can still pretend that our process is stationary, and attempt to isolate a version of context so that matching signature regions have matching (or close) state values. Thus, our statistics are actually trying to extract enough information that, given their value, the process *looks* stationary; and finding a good statistic is similar to automatic "continuous segmentation" and registration of the signatures.

Similar segmentation issues form of the inherent challenges in most other model-based signature matching algorithms. Generally, they are forced by the non-stationarity of the process to segment the stroke discretely and model each segment separately [16, 14]. Within the context of a single, short *segment* of signature, then, they are able to apply models which require stationarity. Yet any such discrete segmentation must be artificial, since the state of a continuous stroke is intrinsically a continuous variable. In our method, however, we are able to recapture the local, stationary dynamics without resorting to external or discrete segmentation algorithms. Even when we augment the features by a discrete time value, the smoothness of the density estimate ensures that there is continuity of the state.

Of course, manually augmenting the feature set carries with it its own questions and issues. For instance, if we are using a spherical kernel, the bandwidth parameter will be influenced by the scale of the new feature. By expanding this scale we increase the relative importance of the feature, in this example to the limiting case that only examples of the k^{th} time-sample influence the model at time k. Decreasing scale approaches the limit wherein the feature provides no effective information at all. In an attempt to deal with this in a principled manner, we chose to normalize the scale of the new feature to be approximately the same as the features we had previously learned. If our learned features fill the perceptron's finite region of support, this will be approximately the variance of a uniform distribution over that region.

5.2 Signature Verification

More so than synthesizing new signatures, people are interested in verifying them. With very little experience, we as humans can confirm to a reasonable accuracy whether a new signature matches an example set or not. Notably, we do so without ever seeing examples of forgery attempts. This ability is the basis of one of our oldest methods of authentication, and one which it seems we are still trying to hold on to. In order to do so it is necessary to find ways to correctly and automatically perform such verification. We shall demonstrate that the information-theoretic concepts of entropy rates and typicality form a framework well-suited to such tasks, while most other discriminative techniques applied to this problem have more difficulty. A brief comparison to existing methods will then show the unique suitability of our framework for the treatment of this problem.

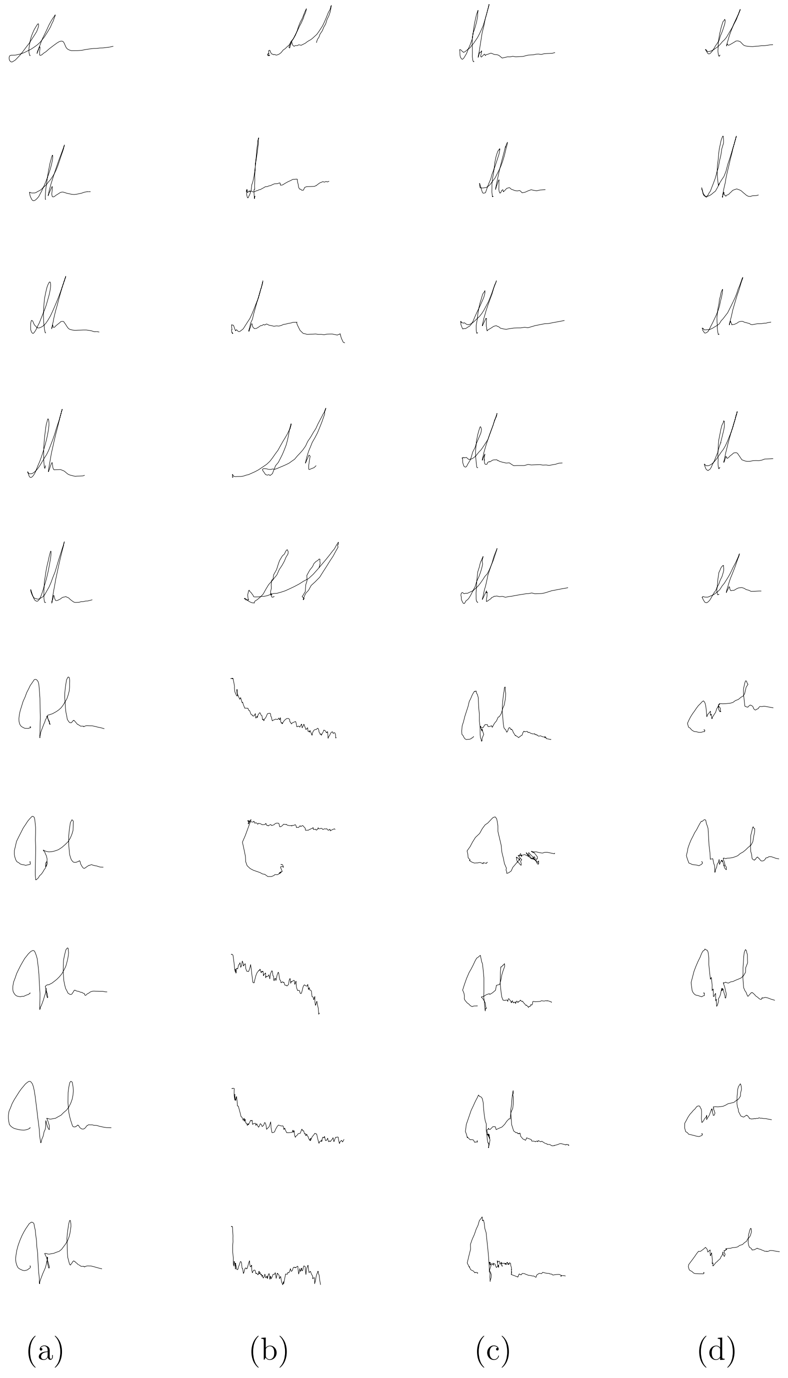


Figure 5.2: More synthesized signatures: (a) Training examples; (b) (dx, dy); (c) (x, y); (d) $(dx, dy) + t$

5.2.1 A Single-Hypothesis Test

In some problems it is the case that we wish to distinguish between a single hypothesis and unknown alternatives. This could be the case when the alternative hypotheses are too numerous to be modeled, for instance a hypothesis test versus "everything else", or it could be that the alternatives are small but difficult to model, perhaps because data about them is simply unavailable.

The task of signature verification is a perfect example of such a system. Although we presumably have data representing the true signature, we have no way to model an entire world full of possible forgers. One might try to circumvent this by asking many people to forge a particular signature, and then building a model of "the generic forger", but this is undesirable for several reasons. First, it requires more manpower than we might like, since it involves many people practicing and then forging each signature we want to verify. And such a model should be unnecessary — humans can discriminate without any examples of forgeries. Even with a number of examples, there is no reason to believe that such a model would aid discrimination with a particular, *specific* forger, whose exact style cannot possibly have been seen by the universal model. Each verification task is in fact a two-hypothesis test between acceptance and rejection in favor of an unknowable, and so unmodelable, forger. Therefore we would like to use a test in which the second, alternative model is unnecessary.

It is possible to construct such a "one-sided" hypothesis test in a similar manner to the Neyman-Pearson multi-hypothesis test formulation. Suppose that we have a stationary process $\{\mathcal{X}_t\}$ distributed according to a known distribution $p(\{\mathcal{X}_t\})$. We know that the average log-likelihood converges to a limit, namely:

$$\frac{1}{N}\sum_{t=1}^{N}\log p(\mathcal{X}_t|\mathcal{X}_{t-1},\ldots) \to \int p(\mathcal{X})\log p(\mathcal{X})d\mathcal{X}$$

in probability as $N \to \infty$ by the A.E.P. More formally, this is

$$\forall\epsilon > 0, \quad \alpha \in (0,1] \qquad \exists N_0 : \forall N > N_0,$$

$$P\left(|H(\mathcal{X}) + \frac{1}{N}\log p(\mathcal{X}_0, \mathcal{X}_1, \ldots, \mathcal{X}_N)| < \epsilon\right) > (1-\alpha)$$

The A.E.P tells us, therefore, that there is a fixed relationship between a bound $[-H(\mathcal{X}) - \epsilon, -H(\mathcal{X}) + \epsilon]$, the number of evaluated samples N_0, and the probability α that the average log-likelihood of those samples will fall within the bound. Therefore given any two of these quantities it is possible to compute the third. The quantity α represents P_D, the probability that we will correctly accept a true sequence from this process. We can further note that our upper and lower bounds around $-H(\mathcal{X})$ need not be symmetric to have this convergence property; later it will be useful to think of our bound asymmetrically, i.e. $[-H(\mathcal{X}) - \epsilon_0, -H(\mathcal{X}) + \epsilon_1]$

The difficulty arises because of the unknowability of the divergence between $\mathcal{X}$ and the alternative(s). Without any further information about an alternative $\mathcal{Y}$ it is impossible to determine the probability of incorrectly accepting a process of that type. We cannot even say whether the log likelihood of $\mathcal{Y}$ under $\mathcal{X}$'s model will approach a value larger, smaller, or even the same as $\mathcal{X}$ itself.

However, when we consider this problem in a practical light, with the application of signature dynamics in mind, things are a little more reassuring. In general signatures of different dynamics should always have lower log-likelihood than true signatures. This is because of the rather special circumstances that lead to the other possibility. If a process is to have *higher* likelihood than another in this formulation, it means that the signature must not only have very similar dynamics (so that the modes of both distributions are near *at almost all time samples*) but also have *less* deviation from those modes than the typical example of a true signature. So, in order for this to occur in our application, a forger would need to be in some sense *better* at signing than the owner himself. It seems unlikely that such a scenario could occur; its occurance would also seem to indicate an end of the usefulness of signatures for authentication.

So, practically speaking, we can apply a test which accepts the signature if its log-likelihood is higher than $-H(\mathcal{X}) - \epsilon_0$, and rejects otherwise. We can find (or, for unknown $p(\cdot)$, estimate) the probability of rejecting a correct signature given a fixed number of samples N_0. However, we are still unable to characterize the probability of false acceptance — but the better our model, the lower this probability will be.

This kind of likelihood-based evaluation appears to be unique in the online signature verification community. There are an abundance of methods which have been applied to select features of signature dynamics for discrimination, including neural networks [16], genetic algorithms [10], and stochastic modeling techniques from linear autoregressions [17, 15] to Hidden Markov Models [21, 6]. Yet it is surprising that even of those methods which construct a model of the signature, that model is never evaluated; instead, its parameters are used as features for some other comparison metric (maximum acceptable distance from a template, hash tables, neural networks, etc). Methods which relied on training a metric in general could not deal with a lack of example forgeries; and fixed metrics were generally lacking in theoretical justifications. Indeed, our own likelihood formulation could be interpreted in terms of distance in a feature space, since likelihood is a function of the distance of observed examples. However, our formulation has a principled interpretation to that distance and thus makes clear the implications of lacking a forgery model.

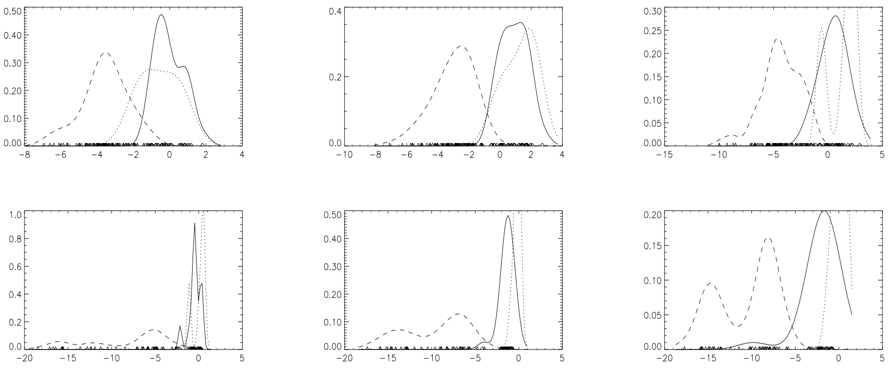
5.2.2 Model Inaccuracies

As we noted in Section 5.1, it may be that our model does not completely capture the dynamics of the system. This is generally disastrous for synthesis; however, for hypothesis testing its effects are not necessarily so unacceptable.

The reason for this is simple. Suppose that we have failed to disambiguate between two states S_1 and S_2 within the process $\mathcal{X}$. Then, instead of $p(x|S_1)$ and $p(x|S_2)$, we possess only $p(x|S_1 \textit{ or } S_2)$. If we attempt to synthesize with this distribution, we could select x while in state S_1 such that $p(x|S_1) = 0$, leading to a sample path which does not inherit the observed characteristics for the simple reason that not all of those characteristics have been detected. However, if we only mean to use $p(\cdot)$ to perform a hypothesis test, the effect may be less. This is because a true signal of this type will still have the reasonable likelihood

$$p(x|S_1 \textit{ or } S_2) \geq p(S_1|S_1 \textit{ or } S_2)p(x|S_1)$$

and a similar process $\mathcal{Y}$ will only have higher likelihood under the ambiguous model than the correct model if it matches the ambiguity. That is to say, if $\mathcal{Y}$ exhibits behavior while thought to be in S_1 similar to to $\mathcal{X}$'s behavior in S_2 or vice versa. So unless the model's inaccuracy *actually* happens to correspond to a characteristic of $\mathcal{Y}$, we suffer little penalty for the unmodeled dynamics.



(x, y) (dx, dy) $(dx, dy) + t$

Figure 5.3: Estimated PDF of average likelihoods for test signatures (solid), training (dotted), and forgeries (dashed) for "ihler" and "john"

5.2.3 Verification Results

We now show some results of using our previously learned models to find the likelihood of various signatures. We will use these likelihoods to attempt to discriminate between true signatures and a set of forgeries. As stated earlier, we cannot say anything definitive about the likelihood of rejecting a new forger's attempts. However, to show that faking such dynamics is at least reasonably difficult, we enlisted a number of "skilled" forgers. This means simply that the forgers had as much time as they liked to practice each signature, knew that dynamics will play a part in the evaluation, and had access to the same training set of example signatures used by the verification algorithm. Each of the three models (differing in coordinate systems), for both synthesized signatures above, are shown.

Our framework of a single hypothesis test requires that we not only have the means to calculate likelihood of new data, but that our model also provides us an estimate of the distribution of the overall likelihood, so that we may select a region of acceptance. Due to our general nonparametric emphasis and framework, we have opted to simply calculate the likelihood of each example signature with respect to the model achieved by simply removing that signature from the examples. These values can then be used to nonparametrically estimate the distribution of such likelihoods. Their estimated distribution is shown by the dotted line in Figure 5.3.

We then calculate and plot the estimated pdfs of the likelihood of a cross-validation set of true signatures and a set of forgeries on the full model. Approximately 30 signatures of each case were evaluated for the estimate. These distributions are shown by the solid and dashed lines in Figure 5.3.

As can be seen, the likelihood of the true signatures is reasonably separated from that of the forgery attempts. In addition, the distribution estimated from the training set and that estimated from the new data appear relatively close, still separated from that of the forgeries, suggesting that even the small sample set used for training can provide a complete enough description of the signature dynamics to allow signature authentication.

We note that many signature-authentication algorithms make use of additional "side-observations", quantities such as pen angle or pen pressure which were not used here. As was discussed early on, however, it is straightforward to add any type of observation to the vector input of $\hat{G}(\cdot)$; we need not be confined to the time series itself. One would expect, then, that if there is consistency in these variables for a given signer the algorithm would incorporate it. Alternatively, additional observations could be added directly to the feature vector (output of $\hat{G}(\cdot)$) just as we added elapsed time. Furthermore, it is then easy to deal with situations where we *lack* that observation, since we can simply marginalize over the unobservable variable.

5.3 Conclusions

In applying our modeling technique to the dynamics of signatures, we have demonstrated that it is capable of capturing the nonlinear relationships and nongaussian uncertainty despite the fact that signatures are inherently nonstationary. We have shown that the model can capture the dynamics completely enough to be used in a generative application, which is a good indication that we have captured most of the information we might need for discriminative purposes.

We then applied the trained model to the task of signature authentication, using the framework of entropy rates and likelihood. Most existing literature on signatures has difficulty coping with single-model discriminative problems or lacks a principled interpretation of acceptance criteria. Our model, on the other hand, implies an appealing and principled interpretation to signature distance (likelihood-based), the single-hypothesis discrimination problem and its intrinsic challenges, and the implications of a given acceptance criterion. We explored the implications of this framework, and then evaluated the likelihood of cross-validation signatures (both real and "skilled" forgery examples) on our previously trained models to give an indication of the performance of the estimator in discriminative purposes. Results were very encouraging; further research will have to be done to fully explore the effectiveness of this methodology.

Finally we would like to state that although all the previous work has been for single-stroke signatures, it is certainly possible to extend it to multiple-stroke, either by simply learning models for each stroke separately and combining their likelihoods, or by artificially stringing together each stroke as a single long time-series and learning features of the entire sequence.

Chapter 6

Conclusions

In this thesis, we have presented and explored a novel approach to modeling dynamical systems. This model combines simple feature statistics of the past with a nonparametric estimate of the relation and uncertainty of these features to the future. The features are selected using an information-theoretic criterion, allowing control of the model dimension while retaining access to large numbers of past observations for their information about the future.

This technique is an attempt to answer a fundamental research question, the question behind tasks of data mining or feature extraction: how do we find and model relationships between our observations and a random variable? Yet it brings up its own fundamental questions as well; questions such as, what is the role of prediction when our model of uncertainty is no longer unimodal? How should one evaluate a model's performance — prediction, likelihood, or something else entirely? These types of questions have been addressed empirically throughout the thesis.

We have shown the application of our technique to modeling a number of different processes. Each application was used to highlight different aspects of the technique itself, giving examples of its strengths and weaknesses, and ways to improve its performance. We will reiterate some of these findings here.

First of all, we showed examples of extremely simple processes with which more traditional descriptions of dynamics and uncertainty were unable to cope. Even if the exact dependence on the past is known, there can still be advantages to using a nonparametric uncertainty description. In the case that the parametric model is *correct*, of course, there are great advantages in computation and quality which can be achieved by a parametric approach. If we believe a parametric description *can* capture the system's behavior, we should always choose to use it. However, there are many occasions when we cannot describe the system parametrically, and we can improve our model using nonparametric techniques. In that case, we must find ways to restrain the exponential growth of complexity. Yet we must do so without compromising the information present in our observations. Section 2.7 discusses our approach to this problem, and the dynamical system model employed.

We applied this technique first to a very simple dynamical system, a concocted random telegraph-like waveform. This system was chosen to give a simple, analytically comprehensible data set with which to evaluate performance. We used this scenario to find results for various possible situations. When we had access to enough past observations to produce statistics which were sufficient for the entire past, i.e. containing all available information, we were able to learn functions which were close to sufficient in a likelihood sense – that the negative log-likelihood given our statistic could be close to the actual entropy rate. When somewhat more than enough data was available, we continued to produce nearly-sufficient functionals; and when too little data was available we nevertheless produced statistics which, while not sufficient, were informative about the state of the system.

An important outcome of this experiment was to highlight the importance of using regularization to constrain the function's training. Using an L^1 penalty to cull unhelpful variables improved the reliability of a function's informativeness. This acts as a complexity control for what is essentially an ill-posed problem.

We also showed that this system can be used for both discriminative or generative models. The trained system was capable of both discerning to which of two different modeled dynamics new data corresponded, and of producing reasonable new sample path realizations. Possessing such a generative model is important not only for synthesis, but also for discrimination when portions of a sequence are missing, or we wish to use sequences of variable length.

We next applied the technique to a data set from the Santa Fe Time Series Competition. The dynamical system was quite structured to the human eye, and we showed that, even using a relatively short window of the past, informative subspaces were capable of capturing this dynamic well enough to produce new realizations which carried the same long-term dependencies.

This second data set was the first to require learning a statistic of more than one dimension. Two or more dimensional subspaces carried with them a number of new problems, including the increased difficulty in estimating a density and increased number of parameters in training. To simplify the situation we again chose a simple functional form, and elected to learn each dimension's parameters sequentially so as to decouple them. We also demonstrated the importance of kernel size in the density estimate, primarily in the estimate used to model behavior *after* learning. A likelihood-maximizing non-uniform kernel size based on nearest-neighbor distance earned our recommendation, although several other techniques were attempted and briefly discussed.

Lastly, we applied the technique to a very challenging real-world data set — treating signatures as dynamical systems with the aim of using a likelihood based verification technique. This problem is of great interest due to the inherent advantages of biometric authentication — means of proving ones identity in some inherent way (other examples include fingerprints, face recognition, etc). These allow security methods which cannot be stolen and are (hopefully) very difficult to fake.

We first learned functionals of the signature dynamics, using synthesis results to highlight the importance of coordinate system transformations, and to discuss the assumption of stationarity and nonetheless modeling an unstationary process. Despite the more limited commercial applications of signature synthesis, a generative model is still of use, since it enables us to perform discrimination in ways which are robust to variations of sequence length and to deletions within the sequence.

We applied these models to the task of online signature verification. This showed considerable promise: entropy rate analysis gives us a unique perspective on a hypothesis test against unknown alternatives. This outlook precisely describes one of the difficulties of the task, and makes clear what we can and cannot expect from any such test. In particular, although we cannot estimate the probability of false acceptance without resorting to an example (a single sample from an immense set of possible forgers), we *can* evaluate how well we have modeled the signature with the knowledge that the better the model we possess, the more difficult forgery becomes; and we can estimate the probability of false rejection in order to determine a region of acceptance. This gives us a method of gauging performance even without a single example forgery. Of course, should forgery examples also be available, we can produce an estimate of the false acceptance rate, but *no method* can guarantee that rate for any given forger.

Experimental trials indicate that the signature models are reasonably accurate descriptions and that even from a small number of signatures a model can be trained and the entropy rate estimated in order to produce a description which is consistent with the cross-validation set and inconsistent with any of our would-be forgers.

In the future, it is our hope that this or similar techniques will be applied to more applications requiring difficult information-extraction — image or speech recognition, for instance. The signature modeling appears to be adept at discrimination, and it is quite possible that a biometric verification scheme could make use of it. Of course, any improvements in training technique, density estimation, or the computational complexity would contribute directly toward improving the available quality of the model, increasing its usefulness and viability. Finally, many uncommon perspectives have appeared as a result of using such a complex description of uncertainty, most notably the lack of a predictive method and lack of a better quality metric than likelihood. Further examination of these ideas could lead to better answers for the questions we were forced to address empirically. All of these issues are open questions for future research.

Appendix A

Algorithms

We will use this appendix to present two of the less well-known algorithms which have been used in the thesis directly, in a more detailed fashion than they have received in the body of the thesis. The code implementation of these algorithms was written in PV-WAVE, and is available through the author.

A.1 InfoMax Network Training

This is an overview of the algorithm we use to adapt a neural network to maximize mutual information between the output features of the network and another (set of) measurement(s). We shall use the following notation:

X, x_k : the process we wish to model, and its value a time k

Y, y_k : observations about the process, e.g. $y_k = [x_{k-1} \dots x_{k-N}]$

$\hat{G}$: a neural network function; input size equal to $\dim(Y)$

$\hat{G}_{1\dots m}$: the *fixed* portion of G; e.g. features already found

$\hat{G}_{m+1\dots n}$: the portion of G which is to be trained by this procedure

Z : the conjoined data: $[X, G(Y)]$. $z_k = (x_k, G(y_k))$

Z^P : $= X$; P refers to the *process* subspace of Z

Z^F : $= \hat{G}_{1\dots m}(Y)$; F refers to the *fixed* subspace of G's outputs

Z^T : $= \hat{G}_{m+1\dots n}(Y)$; T refers to the *trained* portion of the space

We begin by initializing $\hat{G}_{m+1\dots n}$ to something, possibly randomly. In general we should avoid degenerate cases of mutual information, e.g. $\hat{G}_i = \hat{G}_j$ for $i \neq j$ or $\hat{G}_i \equiv 0$. We must also choose an initial kernel size (and shape if desired). Experience has indicated that small kernel sizes are very slow to converge to any solution, and that an oversmoothed estimate still returns reasonable entropy gradients (in that informative solutions are discovered). Therefore in practice we choose the bandwidth as simply as possible – a fixed, global size which was larger than any we later saw in ML bandwidth estimates.

We then repeat the following steps until we have converged:

1) It is possible that we wish to only use a random subset of our full data set $\{z_k\}$. In the interest of reducing notation we will continue using $\{z_k\}$ to refer to the data *at this step*, but we can and often do reselect the data in this set from a larger pool every N iterations. As discussed in Section 3.3.4, this has the effect of giving us a faster but noisier gradient estimate below.

2) Find entropy gradients in "output space" Z. We shall use the identity

$$\mathrm{I}(Z^P; Z^{F+T}) = H(Z^P) + H(Z^{F+T}) - H(Z)$$

where $H(Z^P)$ is fixed, so we must evaluate the gradient of $H(Z^{F+T})$ and $H(Z)$ at each point z_k:

$$\nabla \mathrm{I}|_{z_k} = \nabla H(Z^{F+T})|_{z_k} - \nabla H(Z)|_{z_k}$$

3) In practice, we evaluate ∇H using the ISE approximation. Define $J = \int(\hat{p}(z) - u(z))^2 dz$ where u is the uniform density over a hypercube with sides of length d centered at the origin. Minimizing this criterion J will maximize (approximate) entropy. Closer analysis [8] in the case that the kernel function $K(\cdot)$ is a Gaussian with variance σ^2 yields the following gradient estimate in terms of the network parameters α:

$$-\frac{\partial H}{\partial \alpha} = \frac{\partial J}{\partial \alpha} = \frac{1}{N}\sum_k \epsilon_k \frac{\partial}{\partial \alpha} g(y_k, \alpha)$$

where $\frac{\partial g}{\partial \alpha}$ is the network sensitivity and ϵ_k are the vector-valued error *directions* of z_k:

$$\epsilon_k = f_r(z_k) - \frac{1}{N}\sum_{i \neq k} \kappa_a\left(z_k - z_i\right)$$

and the functions $f_r(z_k)$ and $\kappa_a(z_k - z_i)$ have interpretations (for entropy maximization) of a boundary repulsion vector and a repulsion vector from the other samples. These are given by

$$\begin{aligned} f_r(z_k)_i &\approx \frac{1}{d^{n+1}} \prod_{j \neq i} \left((K * u)\left(y_{ki} + \frac{d}{2}\right) - (K * u)\left(y_{ki} - \frac{d}{2}\right)\right) && \text{(A.1)} \\ \kappa_a(z) &= K(z) * \nabla K(z) && \text{(A.2)} \\ &= -\frac{\exp\left(-\frac{z^T z}{4\sigma^2}\right)}{\left(2^{n+2}\pi^{(n+1)/2}\sigma^{n+3}\right)} z && \text{(A.3)} \end{aligned}$$

4) We now have the gradient of I with respect to the output space $\{z_k\}$ (an $n+1$ dimensional vector for each z_k); however, we only need or want the gradient for the directions in which we will train. Thus we zero out the contributions in dimensions we cannot or do not wish to alter: $(\nabla \mathrm{I})^P = 0$ and $(\nabla \mathrm{I})^F = 0$.

5) We next propagate this gradient information back to the network weights of $\hat{G}_{m+1\ldots n}$. This is a well-known algorithm and will not be discussed further here; for more information see e.g. [3].

After sufficient convergence or a fixed number of iterations, we cease updating the network and take $\hat{G}_{m+1\ldots n}$ to be fixed from this point forth. We then keep some set of data, either a subset of the training data (possibly the entire set), or perhaps a new cross-validation data set, to form the examples for our Parzen kernel density. We use one of the techniques described in Section 2.4.1 to determine a kernel size for this density, and our model is complete.

A.2 Sampling from the conditional of a kernel density estimate

At many points in the thesis, we generate new synthesis paths by sampling from our estimated distribution conditioned on the observed variables. For completeness we present here the algorithm for such sampling.

Sampling can be done efficiently from a Parzen density estimate because of its form as a summation of density functions. That is, suppose we have a kernel density estimate

$$\hat{p}(X) = \sum K_i(X - X_i)$$

where the subscript i on $K_i(\cdot)$ indicates that the kernel function may vary from example to example, perhaps by kernel size or shape, and the notation $X = [X^S, X^C]$ is used to differentiate between those dimensions of X which we wish to sample (S) and those which we condition on (C). Then,

$$\begin{aligned} \hat{p}(X|X^C) &= \frac{\hat{p}(X^S, X^C)}{\hat{p}(X^C)} && \text{(A.4)} \\ &= \frac{\sum_i K_i^C(X^C - X_i^C) \cdot K_i^S(X^S - X_i^S|X^C - X_i^C)}{\sum K_i^C(X^C - X_i^C)} && \text{(A.5)} \end{aligned}$$

We shall make a convenient name change — denote

$$q(i) = \frac{K_i^C(X^C - X_i^C)}{\sum K_i^C(X^C - X_i^C)} \tag{A.6}$$

and

$$q(X^S|i) = K_i^S(X^S - X_i^S|X^C - X_i^C) \tag{A.7}$$

so we can rewrite Equation A.4 as

$$q(X^S) = \sum_I q(X^S|I)q(I)$$

implying that

$$(x^S, i) \sim q(X^S, I) \Rightarrow x^S \sim q(X^S)$$

Now, the form for $q(X^S|I)$ is quite simple, since by Equation A.7 we have

$$x^S \sim X_i^S + \nu_i^S \qquad where \qquad \nu_i^S \sim K_i^S(\nu^S|X^C - X_i^C)$$

and K_i in our case is Gaussian. (In fact, in all our experiments the kernel sizes were uncorrelated between dimensions, making this even easier – but this is not necessary.) Similarly, it is easy to sample from $q(I)$ since this is a discrete distribution with weights given in Equation A.6. So we first sample $i \sim q(I)$, then $\nu \sim K_i^S$, and

$$x = [x^S, x^C] = [(X_i^S + \nu), x^C] \sim \hat{p}(X|X^C) \tag{A.8}$$

A.2.1 A brief note on ML sampling

True maximum-likelihood sampling for such a continuous distribution is a computationally intensive task. A straightforward approach is to discretize the distribution at a fine enough scale to approximate the distribution, and then select the maximum value. Such a scale can be determined by the minimum kernel size in the density estimate, since all features of the distribution will be at least as smooth as the most peaked kernel. If a more exact value for the ML estimate is still desired, it can then be found through gradient ascent without risk of local extrema.

This method will provide an ML estimate of the next sample. However, to find the ML estimate k samples ahead, we would need to propagate these distribution of values through our model, an extremely intensive computation. Another possibility would be to generate many sample paths and use their values at time k to estimate a distribution. Finally, one could use the training data itself to estimate the joint density between X_{t+k} and $\hat{G}(X_{t_{past}})$.

In practice, we implemented an ML estimate on only one data set, in Chapter 4. We used only a one-step-ahead prediction, and in that instance the data itself was discretized, and so we simply found the most likely sample at the data's own quantization level.

Bibliography

[1] A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. on Information Theory*, 39:930–945, 1993.

[2] J. Beirlant, E. J. Dudewicz, L. Györfi, and E. C. van der Meulen. Nonparametric entropy estimation: An overview. *International Journal of Math. Stat. Sci.*, 6(1):17–39, June 1997.

[3] C. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, 1995.

[4] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley and Sons, Inc., New York, 1991.

[5] Luc Devroye. *A Course in Density Estimation*, volume 14 of *Progress in Probability and Statistics.* Birkhauser, Boston, 1987.

[6] J.G.A. Dolfing, E.H.L. Aarts, and J.J.G.M van Oosterhout. On-line signature verification with hidden markov models. In *Proc. of Fourteenth I.C. on Pattern Recognition*, volume 2, pages 1309–1312, 1998.

[7] J. W. Fisher, A. T. Ihler, and P. Viola. Learning informative statistics: A nonparametric approach. In S. A. Solla, T. K. Leen, and K-R. Mller, editors, *Proceedings of 1999 Conference on Advances in Neural Information Processing Systems 12*, 1999.

[8] J. W. Fisher and J.C. Principe. Information preserving transformations. *In submission*, 2000.

[9] K. Fukanaga. *Introduction to Statistical Pattern Recogntion.* 2nd edition, 1990.

[10] G.S.K. Fung, R.W.H. Lau, and J.N.K. Liu. A signature based password authentication method. *Systems, Man, and Cybernetics*, 1:631–636, 1997.

[11] Peter Hall, Simon J. Sheather, M. C. Jones, and J.S. Marron. On optimal data-based bandwidth selection in kernel density estimation. *Biometrika,*, 78(2):263–269, Jun 1991.

[12] Jaakkola and Haussler. Exploiting generative models in discriminative classifiers. In *Proc. of 1998 Advances in Neural Information Processing Systems 11*, 1998.

[13] Harry Joe. Estimation of entropy and other functionals of a multivariate density. *Ann. Inst. Statist. Math.*, 41(4):683–697, 1989.

[14] R. Martens and L. Claesen. Dynamic programming optimisation for on-line signature verification. In *Proc. of the Fourth I. C. on Document Analysis and Recognition*, volume 2, pages 653–656, 1997.

[15] T. Matsuura and H. Sakai. On stochastic system representation of handwriting process and its application to signature verification. In *Signal Processing; 3rd I.C.*, volume 2, pages 1330–1333, 1996.

[16] M. Mohankrishnan, Wan-Suck Lee, and Mark J. Paulik. A performance evaluation of a new signature verification algorithm using realistic forgeries. *Proc. I.C. on Image Processing*, 1:575–579, 1999.

[17] M. Mohankrishnan, M.J. Paulik, and M. Khalil. On-line signature verification using a nonstationary autoregressive model representation. *Circuits and Systems*, 4:2303 –2306, 1993.

[18] Abdelkader Mokkadem. Estimation of the entorpy and infoamation of absolutely continuous random variables. *IEEE Transactions on Information Theory*, 35(1):193–196, Jan 1989.

[19] D. Ormoneit and T. Hastie. Optimal kernel shapes for local linear regression. *Proc., Advances in Neural Information Processing Systems 12*, pages 540–546, 2000.

[20] R.F. Popoli and J.M. Mendel. Relative sufficiency. *IEEE Transactions on Automatic Control*, 38:826–828, 1993.

[21] G. Rigoll and A. Kosmala. A systematic comparison between on-line and off-line methods for signature verification with hidden markov models. In *Proc. of Fourteenth I.C. on Pattern Recognition*, volume 2, pages 1755–1757, 1998.

[22] S. J. Sheather and M. C. Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B*, 53(3):683–690, 1991.

[23] B.W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, New York, 1986.

[24] A.N. Tikhonov and V.Y. Arsenin. *Solutions to Ill-Posed Problems*. Wiley, New York, 1977.

[25] Andreas S. Weigend and Neil A. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley, 1994.

[26] A. Willsky, G. Wornell, and J. Shapiro. *6.432 Course Notes: Stochastic Processes, Estimation, and Detection*. 1998.