Lectures on Statistical Modeling Theory

J. Rissanen
Helsinki Institute for Information Technology
and
Technical Universities of Tampere and Helsinki, Finland
and
Computer Learning Research Center
University of London, Royal Holloway, UK
jorma.rissanen@mdl-research.org

Contents

1 Introduction

Statistical modeling or model building is an activity aimed at learning rules and restrictions in a set of observed data, proverbially called 'laws' or "the go of it" as stated by Maxwell. The traditional approach, presumably influenced by physics, is to imagine or assume that the data have been generated as a sample from a population, originally of a parametrically defined probability distribution and later, more generally, a so-called nonparametric distribution. Then the so-imagined unknown distribution is estimated from the data by use of various principles, such as the least squares and maximum likelihood principles, or by minimization of some mean loss function, the mean taken with respect to an imagined distribution.

Although such a process can work well if the situation is similar to that in physics, namely, that there is a 'law' which is guessed correctly and which is capable of describing the data sufficiently well for one to be able to account for the unavoidable deviations to small random instrument noise. However, in statistical applications the 'law' assumed is a probability distribution, and if we sample any of its estimates we get data which certainly deviate from the observed data by more than 'small' instrument noise. And what is worse there is no reliable, let alone absolute, way to decide whether a given data set is a sample typically obtained from any suggested distribution. It is true that in some very special cases, such as coin flipping, the physical data generating mechanism suggests a good probability model, and we do have a reasonable assurance of a good match. In general, however, we do not have enough knowledge of the machinery that generates the data to convert it into a probability distribution, whose samples would be statistically similar to the observed data, and we end up in the impossible task of trying to estimate something that does not exist. And because we certainly can imagine more than one target distribution to be estimated, an assessment of the estimation error is meaningless. To put this another way, when the problem of statistics is viewed as estimating from data a probability distribution belonging to an assumed set, there is no rational way to compare different sets of the distributions assumed. Since in general a more complex distribution 'fits' the data better than a simpler one, where the degree of the 'fit' is measured by any reasonable standard, it is impossible within such a theory to rule out the inevitable conclusion that the best model is the most complex one assumed. The usual way to avoid this disastreous conclusion is to resort to judgment or adhoc procedures. Good practitioners, of course, know intuitively that one can never find the 'true' data generating distribution, so that it must be regarded only as an unreachable model. But this leads us to the awkward position that we must estimate this model, and since the estimates define the actual model we must live with we end up modeling a model!

There is a different approach to the problem, which is a step in the right direction. Starting with the reasonable idea that in general no model in a collection of parametric models can capture all the regular features in the data we should merely look for a model within the confines of a collection selected that does it best. In order to formalize this Akaike, [1], assumed still the 'true model' to exist, which captures *all* the properties in the data but which is not in the collection selected. He then

looks for the model in the collection that is closest to this 'true model' in the sense of the information theoretic Kullback-Leibler distance. Now, the model with the maximum number of parameters will minimize the distance, which certainly is not what is wanted. But since the distance between the unknown 'true model' and the ones in the collection cannot be computed anyway, the objective is changed to minimize the mean distance, the mean taken over all the estimated models. This can be estimated asymptotically, and the result is Akaike's AIC criterion, [1]. However, this cannot overcome the basic difficulty stemming from the fact that no density function, 'true' or 'false', exists that captures all the information in data. Hence, it is impossible to interpret Akaike's criterion or any other criterion derived from such a false premise in terms of the observed data alone, and we do not know what properties in the data we actually have learned.

We must admit that there is a remedy of sorts to the difficulties stemming from the 'true model' assumption, which is to devise a robust estimation method designed to work no matter which 'true' distribution in a large class would generate the data. Although one can avoid catastrophic results that way such a defeatist approach is unsatisfactory as a general principle, because it goes against the grain of the very intent in model building and statistical inference, which is to learn the properties of the data generating mechanism. There is another way aimed at escaping from the curse of the false assumption of a 'true' data generating mechanism, called 'cross-validation' or 'data-reuse'. This a technique in which one picks a portion of the available data to fit the models and then uses the rest to compare the performance of the fitted models. Often the two sets of models are picked randomly. This is a pure adhoc technique based on the fallacious idea that by repeated sampling one can extract more information from the existing data than what is there.

To summarize, the traditional dogmatic approach to statistical model building, in which probabilities are viewed as inherent properties of 'random' data and restricted to them, is resting on shaky logical foundations. This undoubtedly is the reason that it has not evolved into a general theory, which would provide insight including statements about limitations on statistical inquiry of the kind what can and cannot be done. The lack of a general theory also restricts creation of new methods for model construction required to understand data such as met in the field of data mining, DNA modeling, or image processing, whose complexity far exceeds anything considered in traditional statistics.

A different and popular Bayesian view of modeling, in which probabilities without much worry about their interpretation can be assigned to just about anything, is based on the seemingly attractive idea that the best hypothesis or a model in a collection is the one which has the highest probability of being 'true' in light of the observed data, given that a 'prior' probability was assigned to the hypotheses on some data independent grounds. Of course, the prior could have been determined from other data no longer available. If one of the hypotheses were true, it would indeed be meaningful to talk about such a probability, but this is the case only in simple modeling situations. The interpretation offered by Jeffreys, [12], an eminent Bayesian, is that the probability of a model is the degree of belief we place on it so that the statement 'the degree of belief of hypothesis i is .4'

is meaningful although perhaps not all that illuminating. However, the degrees of belief satisfy the usual axioms of probabilities so that the difference between the two names is just semantic. The so-called posterior probability, into which the data modifies the prior one, is obtained from Bayes' formula by normalization of the joint probability of the data and each hypothesis, which amounts to division by the marginal probability on the data.

Although one can define probability distributions on virtually any set of hypotheses, I do not see why a hypothesis which has the highest probability when none of the hypotheses is 'true' should be taken as the model that best captures the regular features in the data. Actually, Jeffreys himself offered the following justification for the case where the prior probabilities are equal, which is the most important case because of the desire to avoid prejudiced choice of the priors: *"We shall be most ready to accept the hypothesis that requires the fact that the observations have occurred to be the least remarkable coincidence"*. But this statement says nothing about probabilities on models even though the model that assigns the highest probability to the data also happens to have the highest posterior probability as defined by Bayes' formula. It then seems that whatever advantages maximization of the posterior probability has are to a great degree, and in fact often to a dominant degree, inherited from the more fundamental principle which is to seek models that maximize the probability on the *data*. As will be discussed below the real justification for the Bayesian techniques is the fact they approximate quite well a suitably generalized global maximum likelihood technique, especially for large amounts of data, as the subject matter of main interest in these lectures could also be called.

The algorithmic theory of information, introduced by Solomonoff,[30], which we discuss briefly later, provides a different foundation for statistical inquiry, which in principle is free from the problems discussed above. What is most important is that data need not be regarded as a sample from any distribution, and the idea of a model is simply a computer program that describes or encodes the data. This may seem as strange, but it can be made to be equivalent with a probability distribution, constructed from the length of the program as the number of binary digits required to describe the program. The length of the shortest program serves as a measure of the complexity of the data string, and there is even a construct, known as Kolmogorov's sufficient statistic, which provides a natural measure for the complexity of the best model for the data. All this would leave nothing wanting from statistical inquiry except for a fatal problem: The shortest program, which can be argued to provide the penultimate ideal model for the data, or even its length cannot be found by algorithmic means. For an extensive treatment of this theory we refer to [15].

There is, however, another development, based on coding and information theory, which follows in spirit the algorithmic theory but which avoids the computability problem; see [24]. It permits us to define first the *stochastic complexity* of a data set, relative to a collection of models, as the fewest number of bits in a probabilistic sense with which the data can be encoded using a code designed by help of the models. In other words, the set of all programs for a universal computer is replaced by a code as a binary tree whose leaves define the codewords for the data sequences of the same length as

the observed data. Quite nicely such a code will be equivalent with a probability distribution, and the length of the path from the root to the leaf, from which its data sequence can be decoded, is in essence given by the negative binary logarithm of the probability assigned to the data sequence by the distribution. But more is true. The universal model defined by the complexity lets us define the long-sought *information* in the data string that can be extracted with the collection of models at hand. In fact, we obtain a split of the code for the data into an information bearing part and another 'noise' having no useful information to contribute. At least for numerical data the same split extends to the data themselves. This means that we now can compare different classes of models, regardless of their structure, shape or size, simply by the amount of useful information they extract from the data, and we get a foundation for an authentic theory of model building if not for all statistics in general, [27], in which the real-valued parameters and their number both can be treated on equal footing.

The objective of these lectures is to carry out such a program for statistical model theory in an introductory way. The basic and relevant notions of information and coding theories are reviewed in Part I. In Part II the main theme, statistical modeling, is developed in a manner inspired by the theory of Kolmogorov complexity and its universal model, which are reviewed briefly.

Part I
INFORMATION AND CODING

A formal notion of *information* in the sense of *complexity* was introduced at about the same time around the late forties independently by both Norbert Wiener and Claude Shannon, [28], [36]. Despite the fact that a case could be made for this notion of information to be among the most influential ideas in recent history of science, Wiener did not seem to regard it worthwhile to elaborate it much further. He gave a brief sketch of its application to communication problems but that is about it. By contrast, it was Shannon who did provide such an elaboration, and the result was the beginning of an entirely new discipline, information theory, which has a growing impact in almost all branches of science. As will be clear later on, the basic information measure in a restricted form was already given by Hartley in 1928, [11], and in essence the same concept as a measure of disorder has been used in statistical mechanics much longer, 1895, by Boltzmann, [3].

Initially, information theory was synonymous with communication theory, but gradually the central concept of entropy with variations such as the Kolmogorov-Sinai entropy for chaotic processes started to find important applications in probability theory and other parts of mathematics. In the mid sixties an exciting new idea emerged, which added a new branch to information theory with goals and theorems completely different from communication problems. In this, the information in a string is formalized as the length of the shortest computer program that generates the string. The new measure is usually called the Kolmogorov complexity, even though it was introduced in a clear manner years earlier by Solomonoff. The same idea was rediscovered by Chaitin, [5], who has also made other important contributions to the theory.

2 Shannon-Wiener Information

2.1 Basics of Coding with Random Variables

In coding we want to transmit or store sequences of elements of a finite set $A = \{a_1, \ldots, a_m\}$ in terms of binary symbols 0 and 1. The set A is called the *alphabet* and its elements are called *symbols*, which can be of any kinds, often numerals. The sequences are called *messages*, or often just data, when the symbols are numerals. We begin by defining the code as a function $C : A \to B^*$ taking each symbol in the alphabet into a finite binary string, called *codeword*. We are interested in the codes as one-to-one maps so that they have an inverse. The code may be extended to sequences $x = x_1, \ldots, x_n$

$$C : A^* \to B^*$$

by the operation of *concatenation*: $C(xx_{n+1}) = C(x)C(x_{n+1})$, where xa denotes the string obtained when symbol a is appended to the end of the string x. We want the extended code, also written as C, to be not only invertible but also such that the codewords $C(i)$ of the symbols $i \in A$ (We

often write the symbols as i rather than x_i) can be separated and recognized in the code string $C(x)$ without a comma. This implies an important restriction to the codes, the so-called prefix property, which states that *no codeword is a prefix of another*. This requirement, making a code a *prefix code*, implies the important Kraft inequality

$$\sum_{i \in A} 2^{-n_i} \leq 1, \tag{1}$$

where $n_i = |C(i)|$ denotes the length of the codeword $C(i)$.

To prove this consider the leaves of any complete binary tree; ie, a tree where each node has either two sons or none. Any such tree can be obtained by starting with a 2-leaf tree and splitting successively the leaf nodes until the tree is obtained. For the 2-leaf tree the Kraft inequality holds with equality: $2^{-1} + 2^{-1} = 1$. Splitting a node w of length $|w|$ the equality $2^{-|w0|} + 2^{-|w1|} = 2^{-|w|}$ holds for the son nodes, just as the probabilities of a Bernoulli process, which by an easy induction implies that the Kraft inequality holds with equality for any complete binary tree. The codewords of a prefix code are leaves of a tree, which can be completed. Hence, the claim holds. It is easily extended even to countable alphabets, where the sum becomes the limit of strictly increasing finite sums bounded from above by unity.

The codeword lengths of a prefix code, then, define a probability distribution by $P(i) = 2^{-n_i}/K$, where K denotes the sum of the left hand side of the Kraft inequality. Even the converse is true in essence. Indeed, assume conversely that a set of integers $n_1, \ldots, n_m$ satisfies the Kraft inequality. We shall describe a prefix code with the lengths of the codewords given by these integers. Let $\hat{n}$ be the maximum of these integers. Write each term as

$$2^{-n_i} = 2^{r_i} \times 2^{-\hat{n}},$$

where $r_i = \hat{n} - n_i$, and

$$\sum_i 2^{-\hat{n}} 2^{r_i} \leq 1,$$

by the assumption. This is a sum of the leaf probabilities $2^{-\hat{n}}$ over a subset of the leaves $0...0, 0...01, \ldots$ of a balanced tree of depth $\hat{n}$. Hence, we get the required prefix code by partitioning this subset into m segments of $2^{r_1}, 2^{r_2}, \ldots$ adjacent leaves in each such that each segment has a common mother node of length n_i, which we take as the codeword.

An important practical coding problem results when we ask how to design a prefix code when the symbols are taken independently with probabilities $P(a)$, and we want the code to have as short a mean length

$$L^*(P) = \min_{\{n(a_i)\}} \sum_i P(a_i) n(a_i)$$

as possible, where the lengths $n(a_i)$ are required to satisfy the Kraft inequality. This may also be regarded as a measure of the *mean complexity* of the alphabet A or its symbols taken by sampling the distribution P. Indeed, intuitively, we consider an object 'complex' if it takes a large number of binary digits to describe (= encode).

The optimal lengths and a corresponding prefix code can be found by Huffman's algorithm, but far more important is the following remarkable property:

Theorem 1 *For any code for which the Kraft-inequality holds, the mean code length $L(P)$ satisfies*

$$(i)\ L(P) \geq -\sum_i P(a_i)\log_2 P(a_i) \equiv H(P)$$

$$(ii)\ L(P) = H(P) \Leftrightarrow n(a_i) \equiv -\log_2 P(a_i),$$

where $0\log_2 0 = 0$.

Proof. We give a very simple proof of this fundamental theorem usually credited to Shannon. Letting log stand for the binary logarithm and ln for the natural one, we have

$$\begin{aligned}\sum P(a_i)\log\frac{2^{-n(a_i)}}{P(a_i)} &= (\log e)\sum P(a_i)\ln\frac{2^{-n(a_i)}}{P(a_i)} \\ &\leq (\log e)\sum P(a_i)[\frac{2^{-n(a_i)}}{P(a_i)} - 1] \\ &= (\log e)[\sum 2^{-n(a_i)} - 1] \leq 0.\end{aligned}$$

The first inequality follows from $\ln x \leq x - 1$.

The lower bound $H(P)$ of the mean code lengths in the theorem is the famous *entropy,* which may be taken as a measure of the *ideal* mean complexity of A in light of P. Indeed, $H(P) \leq L^*(P) \leq H(P) + 1$, and for large alphabets the entropy is close to $L^*(P)$. The ideal code length $-\log P(a_i)$, the *Shannon-Wiener information*, may be taken as the *complexity* of the element a_i, relative to the given distribution P. The role of the distribution P is that of a *model* for the elements in the set A in that it describes a property of them, which, in turn, restricts the elements in a collective sense rather than individually. Entropy, then, measures the strength of such a restriction in an inverse manner. For instance, binary strings generated by sampling a Bernoulli distribution (or *source*) such that the probability of symbol 0 is much greater than that of 1 have a low entropy meaning a strong restriction and allowing a short mean code length. Conversely, coin flipping strings have little that constrains them, and they have the maximum entropy 1 per symbol.

There is already a hint in the theorem of the intimate connection between code length minimization and model selection, which will be the main theme in these lectures. To see this, consider the class of independent identically distributed (iid) processes over the alphabet A, defined by the parameters $P(a_i)$ for $a_i \in A$. Suppose we have a data sequence from A^n of length n generated by some unknown process in the family, and we consider encoding the data sequence with a prefix code. If we assign a binary string of length ℓ_i to the symbol a_i, the code length for the string would be $\sum_i n_i\ell_i$, where n_i denotes the number of times symbol a_i occurs in the string. The code length would be minimized for $\ell_i = \log(n/n_i)$, which can be seen by the theorem, except that ℓ_i will have to be an integer. We can take $\ell_i = \lceil\log(n/n_i)\rceil$, to satisfy the Kraft inequality, and we would get a near optimal code length, at least for large alphabets, because the extra length amounts at most to

one bit per symbol. However, if we do not worry about the integer length requirement and consider $\log(n/n_i)$ as an *ideal* code length, we see that the optimal ideal code length gives the maximum likelihood estimate of the unkown parameters $P(a_i)$ of the process that generated the data.

2.2 Basic Properties of Entropy and Related Quantities

Notation: when a random variable (r.v.) X has the distribution P, which is symbolized as $X \sim P$, we write frequently $H(P)$ as $H(X)$. Clearly,

(1) $H(X) \geq 0$,

because it is the sum of non-negative elements.

In the important binary case, where $P = (p, 1-p)$, we write $H(P) = h(p)$. The function is a symmetric concave function of p, reaching its maximum value 1 at $p = 1/2$, and vanishing at points 0 and 1.

(2) If $(X, Y) \sim p(x, y)$, then the *conditional* entropy is defined as

$$\begin{aligned} H(Y|X) &= \sum_{x,y} p(x,y) \log \frac{1}{p(y|x)} \\ &= \sum_{x} p(x) \sum_{y} p(y|x) \log \frac{1}{p(y|x)} \\ &= \sum_{x} p(x) H(Y|X=x). \end{aligned}$$

Notice that $H(X|X) = 0$ and $H(Y|X) = H(Y)$, if X and Y are independent.
We have

(3) $H(X, Y) = H(X) + H(Y|X)$.

Prove the following:

(4) $H(X, Y|Z) = H(X|Z) + H(Y|X, Z)$.

The *relative entropy* between two distributions p and q is defined as

$$D(p\|q) = \sum_{x} p(x) \log \frac{p(x)}{q(x)}.$$

By Shannon's theorem,

(5) $D(p\|q) \geq 0$,

equality iff $p(x) \equiv q(x)$.

The relative entropy is also called the Kullback-Leibler distance. Note that $D(p\|q) \neq D(q\|p)$ in general.

The important *mutual information* is defined as

$$I(X;Y) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)q(y)} = D(p(X,Y)\|p(X)q(Y)),$$

where the two marginals are denoted by $p(x)$ and $q(y)$, respectively.

Because $p(x,y) = p(y,x)$ the mutual information is symmetric: $I(X;Y) = I(Y;X)$.

Further,

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X).$$

To see this, write

$$\begin{aligned} I(X;Y) &= \sum_{x,y} p(x,y) \log \frac{p(x|y)q(y)}{p(x)q(y)} \\ &= \sum_x [\log \frac{1}{p(x)}] \sum_y p(x,y) - \sum_{x,y} p(x,y) \log \frac{1}{p(x|y)}. \end{aligned}$$

It follows from (5) that

(6) $I(X;Y) \geq 0$,

and the equality iff X and Y are independent. More generally, the *conditional mutual information* is defined as

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z),$$

which also satisfies

(7) $I(X;Y|Z) \geq 0$,

the equality holding iff X and Y are conditionally independent. Indeed,

$$I(X;Y|Z) = \sum_{x,y,z} p(x,y,z) \log \frac{p(x,y|z)}{p(x|z)p(y|z)},$$

which vanishes exactly when the numerator can be written as the denominator.

It further follows from (5) that

(8) $H(X) \leq \log |A|$,

the equality iff X has the uniform distribution $u(x) = 1/|A|$, where A denotes the range of X and $|A|$ its size. Indeed,

$$D(p\|u) = \sum_x p(x) \log \frac{p(x)}{u(x)} = \log |A| - H(X) \geq 0.$$

Equation (6) implies at once

(9) $H(X|Y) \leq H(X)$,

the equality holding iff X and Y are independent.

2.3 Channel Capacity

By a 'channel' information theorists mean a conditional probability distribution $p(y|x)$ to model a physical channel, where x denotes random symbols entering the channel and y the usually related but in general distorted symbols exiting the channel. By coding we have the chance of selecting the probability distribution by which the input symbols are put into the channel, say $w(x)$. The problem of utmost importance is to select the distribution $w(x)$ so that the mutual information is maximized

(10) $\max_w I_w(X;Y) = \max_w \sum_x w(x) D(p(y|x)\|p(y))$,

where, we realize, $p(y) = p_w(y) = \sum_z p(x,y) = \sum_z p(y|z)w(z)$ depends on w. This is what makes the maximization problem difficult. By studying the nonlinear maximization problem one can deduce an important necessary condition for the maximizing distribution $w^*(x)$:

(11) $D(p(y|x)\|p_{w^*}(y))$

is the same for all x, and hence the *channel capacity* C is given by

(12) $C = I_{w^*}(X;Y) = D(p(y|x)\|p_{w^*}(y))$.

There is an algorithm due to Arimoto and Blahut to calculate the channel capacity. First, it is not too hard to see that the capacity results from the double maximization

(13) $C = \max_{q(x|y)} \max_{w(x)} \sum_x \sum_y w(x)p(y|x) \log \frac{q(x|y)}{w(x)}$.

Then by starting with a guess for the maximizing $w(x)$, say the uniform, we find the maximizing conditional distribution, which can be seen to be

$$q(x|y) = \frac{w(x)p(y|x)}{\sum_u w(u)p(y|u)}.$$

For this conditional distribution we find the next iteration $w(x)$ by maximization, which can be done easily with Lagrange multipliers. The result is

$$w(x) = \frac{\prod_y q(x|y)^{p(y|x)}}{\sum_x \prod_y q(x|y)^{p(y|x)}},$$

and the cycle can be repeated. That it converges to the channel capacity follows from a general result due to Csiszar and Tusnady.

2.4 Chain Rules

Theorem 2 *Let $X_1, \ldots, X_n \sim p(x_1, \ldots, x_n)$. Then*

$$H(X_1, \ldots, X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, \ldots, X_1).$$

Proof. For $n = 2$ the claim follows from (3). For $n = 3$, we get first from (4)

$$H(X_1, X_2, X_3) = H(X_1) + H(X_2, X_3|X_1),$$

and then the claim with an application of (4) to the second term. By a simple induction the claim follows for any n.

Theorem 3

$$I(X_1, \ldots, X_n; Y) = \sum_{i=1}^{n} I(X_i; Y|X_{i-1}, \ldots, X_1).$$

Proof. The claim follows from the previous theorem and the definition of the conditional mutual information. For instance, for $n = 2$

$$\begin{aligned} I(X_1, X_2; Y) &= H(X_1, X_2) - H(X_1, X_2|Y) \\ &= H(X_1) + H(X_2|X_1) - H(X_1|Y) - H(X_2|X_1, Y), \end{aligned}$$

of which the claim follows by taking the first and the third terms and the second and the fourth, and adding them together.

Finally, define the conditional relative entropy as

$$D(p(Y|X)\|q(Y|X)) = \sum_x p(x) \sum_y p(y|x) \log \frac{p(y|x)}{q(y|x)}.$$

Theorem 4

$$D(p(Y, X)\|q(Y, X)) = D(p(X)\|q(X)) + D(p(Y|X)\|q(Y|X))$$

Proof. Immediate from the definitions:

$$D(p(Y, X)\|q(Y, X)) = \sum_{x,y} p(x, y) \log \frac{p(x)p(y|x)}{q(x)q(y|x)},$$

which gives the claim.

2.5 Jensen's Inequality

One of the most important tools in information theory is Jensen's inequality. A function $f(x)$ is called *convex* in an interval (a, b), if for x and y in the interval

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$$

for all $0 \leq \lambda \leq 1$; ie, if $f(x)$ does not lie above any cord. If it lies below, except for the two points, the function is *strictly convex*. A function $f(x)$ is concave if $-f(x)$ is convex.

Theorem 5 *If $f(x)$ is convex, then the mean, written as the operation 'E', satisfies*

$$Ef(X) \geq f(EX).$$

If f is strictly convex, the equality implies that X is constant.

Proof. (For discrete X.) Let the range of X be just the two points x_1 and x_2 with $p_i = P(x_i)$. Then by convexity

$$p_1 f(x_1) + p_2 f(x_2) \geq f(p_1 x_1 + p_2 x_2).$$

Let the theorem then hold for $k-1$ points. We have

$$\begin{aligned}
\sum_{i=1}^{k} p_i f(x_i) &= p_k f(x_k) + (1-p_k) \sum_{i=1}^{k-1} \frac{p_i}{1-p_k} f(x_i) \\
&\geq p_k f(x_k) + (1-p_k) f(\sum_{i=1}^{k-1} \frac{p_i}{1-p_k} x_i) \\
&\geq f[p_k x_k + (1-p_k) \sum_{i=1}^{k} \frac{p_i}{1-p_k} x_i] = f(\sum_{i=1}^{k} p_i x_i).
\end{aligned}$$

We conclude this section by demonstrating that the relative entropy is a convex function of its arguments, the two probability measures p and q. This further implies that the entropy is a concave function of its argument, the single probability measure. Here, the definitions of convexity and concavity are generalized from functions of the reals to functions of any arguments for which linear combinations may be formed.

Theorem 6 *$D(p\|q)$ is convex:*

$$D(\lambda p + (1-\lambda)p' \| \lambda q + (1-\lambda)q') \leq \lambda D(p\|q) + (1-\lambda) D(p'\|q'),$$

for $0 \leq \lambda \leq 1$.

Proof. First, for two sets of numbers $a_i > 0$ and $b_i > 0$,

$$(i) \quad \sum_i a_i \log \frac{a_i}{b_i} \geq (\sum_i a_i) \log \frac{\sum a_i}{\sum_i b_i}.$$

This is seen to be true by an application of Shannon's theorem to the distributions $a_i / \sum_j a_j$ and $b_i / \sum_j b_j$. Hence, by putting $a_1 = \lambda p(x)$, $a_2 = (1-\lambda)p'(x)$, $b_1 = \lambda q(x)$, and $b_2 = (1-\lambda)q'(x)$, applying (i) and summing over x we get the claim.

Theorem 7 *$H(p)$ is concave:*

$$H(\lambda p + (1-\lambda)p') \geq \lambda H(p) + (1-\lambda)H(p'),$$

for $0 \leq \lambda \leq 1$.

Proof. For the uniform distribution $u(x)$ we have

$$H(p) = \log |A| - D(p\|u),$$

and since $D(p\|u)$ as a function of p is convex by the previous theorem, the claim follows.

2.6 Theory of Types

We give a brief account of the important theory of types. Consider an iid process over the alphabet A.

Definition: The *type* of a sequence $x = x_1, \ldots, x_n$ is the empirical probability measure P_x on A

$$P_x = \{P_x(a) = \frac{n(a|x)}{n} : a \in A\},$$

where $n(a|x)$ denotes the number of times symbol a occurs in x.

Definition: $\mathcal{P}_n$ is the *set* of types in A^n.

As an example, for $A = B$, the binary alphabet,

$$\mathcal{P}_n = \{(0,1), (\frac{1}{n}, \frac{n-1}{n}), \ldots, (\frac{n-1}{n}, \frac{1}{n}), (1,0)\}.$$

Definition: type *class* of P_x is

$$\mathcal{T}(P_x) = \{y \in A^n : P_y = P_x\}$$

As an example, let $A = \{1,2,3\}$ and $x = 11321$. Then

$$\begin{aligned} P_x(1) &= 3/5, P_x(2) = P_x(3) = 1/5 \\ \mathcal{T}(P_x) &= \{11123,\ 11132,\ \ldots, 32111\} \\ |\mathcal{T}(P_x)| &= \binom{5}{3,1,1} = \frac{5!}{3!1!1!} = 20 \end{aligned}$$

Theorem 8

$$|\mathcal{P}_n| \leq (n+1)^{|A|}$$

Proof: Each P_x is a function $P_x : A \to \{0, 1/n, \ldots, n/n\}$, and there are at most $(n+1)^{|A|}$ functions.

Theorem 9 *Let $\{X_i\}$ be an iid process over A with distribution Q. We denote also by $Q(x) = \prod_{i=1}^n Q(x_i)$ the probability of a sequence $x = x^n = x_1, \ldots, x_n$.*

$$Q(x) = 2^{-n[H(P_x) + D(P_x\|Q)]}.$$

Proof:

$$\begin{aligned} Q(x) &= \prod_{i=1}^{n} Q(x_i) = \prod_{a\in A} Q^{n(a|x)}(a) \\ &= \prod_{a\in A} Q^{nP_x(a)}(a) = \prod_{a\in A} 2^{nP_x(a)\log Q(a)} \\ &= 2^{-n\sum_{a\in A} P_x(a)[\log \frac{P_x(a)}{Q(a)} - \log P_x(a)]}. \end{aligned}$$

Corollary: If $Q = P_x$ then

$$Q(x) = P_x(x) = 2^{-nH(P_x)}.$$

Example: For a Bernoulli process, $P(0) = p$:

$$-\log P(x|\frac{n_0}{n}) = nH(\frac{n_0}{n}) = n\log n - \sum_{i=0}^{1} n_i \log n_i.$$

Theorem 10 *For any $P_x \in \mathcal{P}_n$*

$$\frac{1}{(n+1)^{|A|}} 2^{nH(P_x)} \leq |\mathcal{T}(P_x)| \leq 2^{nH(P_x)}. \tag{2}$$

Proof: Upper bound:

$$1 \geq \sum_{x\in\mathcal{T}(P_x)} 2^{-nH(P_x)} = |\mathcal{T}(P_x)| 2^{-nH(P_x)}.$$

Lower bound: Since $A^n = \cup_{P_x\in\mathcal{P}_n} \mathcal{T}(P_x)$ we have

$$\max |\mathcal{T}(P_x)| \geq \frac{|A^n|}{|\mathcal{P}_n|} \geq \frac{|A^n|}{(n+1)^{|A|}}.$$

Also, $|A^n| \geq 2^{nH(P_x)}$, which follows from (8) above.

Theorem 11 *For any $P \in \mathcal{P}_n$ and any Q*

$$\frac{1}{(n+1)^{|A|}} 2^{-nD(P\|Q)} \leq Q(x) \leq 2^{-nD(P\|Q)}.$$

Proof:

$$Q(\mathcal{T}(P)) = |\mathcal{T}(P)| 2^{-n[H(P)+D(P\|Q)]},$$

which with (2) implies the claim.

Corollary: If $Q = P$ then

$$P(\mathcal{T}(P)) = |\mathcal{T}(P)| 2^{-nH(P)}.$$

2.7 Equipartition Property

The information theoretic inequalities for the entropy and the related quantities derived above, which all are properties in the mean, get strengthened when the range of the random variables involved is large, because then the results hold to a close approximation essentially for all values and not just in the mean. After all, that is what we intuitively want. For instance, we may then talk

about the complexity of individual strings in the sense that nearly all strings generated by a certain probabilistic source have the mean complexity, and so on. Although the alphabet or the range of a random variable often is not large, we get random variables with large range when instead of symbols we consider sequences of them.

Define the set of typical sequences of an iid process over A with distribution Q as follows

$$\mathcal{T}_Q^\epsilon = \{x^n = x : D(P_x \| Q) \leq \epsilon\},$$

where P_x is the type of x. Then

$$\begin{aligned} 1 - Q(\mathcal{T}_Q^\epsilon) &= \sum_{P_x : D(P_x\|Q) > \epsilon} Q(\mathcal{T}(P)) \\ &\leq \sum_{P_x : D(P_x\|Q) > \epsilon} 2^{-nD(P_x\|Q)} \\ &\leq \sum_{P_x : D(P_x\|Q) > \epsilon} 2^{-n\epsilon} \\ &\leq (n+1)^{|A|} 2^{-n\epsilon} = 2^{-(\epsilon - |A| \frac{\log(n+1)}{n})}, \end{aligned}$$

which goes to zero as $n \to \infty$. Hence, as n grows the probability of the set of typical sequences goes to one at the near exponential rate, no matter what ϵ is. Moreover, by Theorem 9 all typical sequences have just about equal probability

$$2^{-n(H(Q)+\epsilon)} \leq Q(x_1, \ldots, x_n) \leq 2^{-n(H(Q)-\epsilon)}$$

given by the entropy. This is called the asymptotic equipartition property.

Since the tail probabilities of the complement events of $\mathcal{T}_Q^\epsilon$ are summable over n the probability of the limsup of these events is zero by Borel-Cantelli lemma. Therefore the probability of the set of infinite strings, along which $D(P_{x^n} \| Q(x^n)) \leq \epsilon$ for all but finitely many times, is unity for all positive ϵ. This means that when n reaches some finite number the fluctuation of the type about the data generating probability Q does not exceed a fixed amount depending on ϵ, and it goes to zero if we let ϵ shrink to zero. In other words, $n_i(x^n)/n \to Q(i)$ almost surely, where $n_i(x^n)$ denotes the number of the occurrences of symbol i in x^n.

3 Coding with Random Processes

Hitherto we have considered data modeled as outcomes of random variables, defined by the alphabet and a single probability distribution, which when sampled repeatedly leads to an iid process. A much more general way to model data sequences is to consider them as samples from a random process.

3.1 Random Processes

Definition. A countable sequence of random variables $X_1, X_2, \ldots$ is said to form a *random (or stochastic) process*, r.p., if a (measurable) function $P : A^* \to [0, 1]$, where A^* is the set of all finite strings, exists satisfying the two axioms

1. $P(\lambda) = 1$, λ is the empty string

2. $\sum_{x_{n+1}} P(x_1, \ldots, x_{n+1}) = P(x_1, \ldots, x_n)$, all n.

Notice that the joint probability measure $P_n(x_1, \ldots, x_n)$ exists for each n, because the $X_i's$ are random variables. The second axiom requires that the marginal probability function obtained from P_{n+1} on the collection $x_1, \ldots, x_n$ agrees with the joint probability function P_n on the same collection; ie,

$$\sum_{x_{n+1}} P_{n+1}(x_1, \ldots, x_{n+1}) = P_n(x_1, \ldots, x_n), \tag{3}$$

which of course is not automatically satisfied. When it is true, we can safely omit the subindex n, as we did in the two axioms above, which notation actually conforms with the Kolmogorov extension theorem. This states that any set of probability measures P_n satisfying Equation 3 admit a unique extension P, defined for infinite strings. The same axiom permits us to define the conditional probabilities

$$P(x_{n+1}|x^n) = \frac{P(x^{n+1})}{P(x^n)}, \tag{4}$$

which by linking the past with the future is clearly needed to make meaningful predictions.

The idea of stationarity is captured by the axiom

$$\sum_{x_1} P(x_1, \ldots, x_n) = P(x_2, \ldots, x_n). \tag{5}$$

What this means is that in terms of the random variables

$$\sum_{x_1} Pr\{X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n\} = Pr\{X_1 = x_2, \ldots, X_{n-1} = x_n\},$$

which is just the statement of shift invariance. Finally, a process is independent if

$$P(x_1, \ldots, x_n) = \prod_i P(x_i). \tag{6}$$

3.2 Entropy of Stationary Processes

There are two ways to define the entropy of a random process:

1. $H_1(X) = \lim \frac{1}{n} H(X_1, \ldots, X_n)$

2. $H_2(X) = \lim H(X_n|X_{n-1}, \ldots, X_1)$

provided, of course, that the limits exist.

Theorem 12 *For stationary processes $H_1(X) = H_2(X)$.*

Proof. By Equation (9) in subsection 1.2,

$$\begin{aligned} H(X_{n+1}|X_n,\ldots,X_1) &\leq H(X_{n+1}|X_n,\ldots,X_2) \\ &= H(X_n|X_{n-1},\ldots,X_1), \end{aligned}$$

where the equality follows from stationarity. Hence, $H_2(X)$ as the limit of a nonincreasing sequence of positive numbers exists.

Consider then

$$\begin{aligned} \frac{1}{n}H(X_1,\ldots,X_n) &= \frac{1}{n}\sum_{i=1}^{n} H(X_i|X_{i-1},\ldots,X_1) \quad (7) \\ &= H_2(X) + \frac{1}{n}\sum_{i}[H(X_i|X_{i-1},\ldots,X_1) - H_2(X)]. \quad (8) \end{aligned}$$

The difference within the brackets is not greater than ϵ for $i \geq N_\epsilon$, which implies that

$$\begin{aligned} \frac{1}{n}H(X_1,\ldots,X_n) &\leq H_2(X) + \\ &+ \frac{1}{n}\sum_{i=1}^{N_\epsilon}[H(X_i|X_{i-1},\ldots,X_1) - H_2(X)] + \frac{n-N_\epsilon}{n}\epsilon. \end{aligned}$$

The second term goes to zero with increasing n and since the last term is less than ϵ, we certainly have

$$H_1(X) \leq H_2(X) + 2\epsilon.$$

Since we can take ϵ as small as we like $H_1(X) \leq H_2(X)$. To get the opposite inequality, notice that by stationarity and the fact that conditioning cannot increase the entropy $H(X_i|X_{i-1},\ldots,X_1) - H_2(X) \geq 0$. Hence by Equation (8), $\frac{1}{n}H(X_1,\ldots,X_n \geq H_2(X)$, and so is the limit H_1.

Because of the theorem we write $H_1(X) = H_2(X) = H(X)$.

3.3 Markov processes

It is clear that to describe a random process in a constructive way, for instance for the purpose of fitting it to data, we must consider special types of processes, for otherwise it would take an infinite number of conditional probabilities, Equation 4, to specify them. The most familiar and important subclasses of processes are those of finite memory:

Definition: A process is a *Finite Memory* process, if

$$P(x_{n+1}|x_n,x_{n-1},\ldots,x_1) = P(x_{n+1}|x_n,x_{n-1},\ldots,x_{n-k}) \equiv P(x_{n+1}|x_{n-k}^n).$$

We can then write the recursion

$$P(x^{n+1}) = P(x_{n+1}|x_{n-k}^n)P(x^n).$$

Such a process is also called a *Markov* process of order k. A first order Markov process, then, is one satisfying

$$P(x_1, \ldots x_n) = P(x_n|x_{n-1})P(x_1, \ldots, x_{n-1}), \tag{9}$$

where the symbols x_t are called *states*. If, in particular, the conditional probabilities $P(x_n|x_{n-1})$ do not change as a function of time, the process is called *time-invariant*. Letting x_t range over $\{1, \ldots, m\}$ we get from Equation (9) by summing first over $x_1, \ldots, x_{n-2}$ and then over x_{n-1},

$$P(x_n) = \sum_i P(x_n|i)P(x_{n-1} = i). \tag{10}$$

Due to the properties of the matrix $A = \{p_{ij} = P(i|j)\}$ of the state transition probabilities, the probabilities of the states converge to their stationary limits, written as a column vector $\bar{P} = col(P(1), \ldots, P(m))$. They add up to unity and by Equation (10) satisfy

$$\bar{P} = A\bar{P}. \tag{11}$$

Example. Let for a binary first order Markov process $p_{11} = 1-\alpha$, $p_{12} = \beta$, $p_{21} = \alpha$, and $p_{22} = 1-\beta$. Solving the stationary state probabilities with the additional requirement $P(1) + P(2) = 1$ gives $P(1) = \frac{\beta}{\alpha+\beta}$ and $P(2) = \frac{\alpha}{\alpha+\beta}$.

For stationary Markov processes

$$H(X) = H(X_2|X_1) = -\sum_{x_1} p(x_1) \sum_{x_2} p(x_2|x_1) \log p(x_2|x_1).$$

In case of the first order Markov process in the example we get

$$H(X) = \frac{\beta}{\alpha+\beta} h(\alpha) + \frac{\alpha}{\alpha+\beta} h(\beta),$$

where $h(\alpha)$ denotes the binary entropy function evaluated at α.

3.4 Tree Machines

There are two characteristics of particular importance in Markov processes: the order and the number of free parameters needed to define the process. A Markov process of order k with an alphabet size m has m^k states, each having $m-1$ probability parameters. However, there are important applications where the number of states is very large but the number of parameters required to specify the process is small, which makes their estimation relatively easy. Hence, it is worthwhile to separate the two characteristics, which can be achieved by *Tree Machines*, introduced in [21] and analyzed and developed further in [34]. We describe only binary tree machines.

First, take a complete binary tree, which is characterized by the property that each node has either two successors or none if the node is a leaf. At each node s a conditional probability $P(i = 0|s)$ is stored, which at the root node, λ, is written as $P(i = 0)$. In addition, there is defined a permutation of the indices, which we write as $\sigma(1, 2, \ldots, n) = n - \tau_1, n - \tau_2, \ldots, n - \tau_n$. The intent with this is to order the past symbols $x_n, x_{n-1}, ..., x_1$ by their importance in influencing the 'next' symbol x_{n+1}, or

at least what we think their importance is. Hence, for example, in Markov processes we implicitly assume that the most important symbol is x_n, the next most important x_{n-1} and so on, which is achieved by picking the permutation as the one that reverses the order. In modeling image data we often pick the order as the previous symbol x_n, the symbol x_{n-M} right above the symbol x_{n+1} and so on by the geometric nearness in two dimensions to the symbol x_{n+1}.

Let $s_t = x_{n-\tau_1}, x_{n-\tau_2}, \ldots$ be the deepest node in the tree, defined by the past permuted symbols. The tree defines the joint probability

$$P(x_1, \ldots, x_n) = \prod_{t=0}^{n-1} P(x_{t+1}|s_t),$$

to any string, where we take $s_0 = \lambda$. The additional assumption of stationarity implies that the leaf probabilities actually determine all the others, which means that the process gets defined by the conditional probabilities at the leaves. Since the tree need not be balanced, the number of parameters can be much smaller than the number of states. Notice that the set of all leaf nodes may not define the states. Here is a simple example showing this. Take the 4-leaf tree consisting of the leaves 1, 00, 010, and 011. If we consider the previous symbol $x_n = 1$ as a state, then the next symbol $x_{n+1} = 0$ moves the machine to the state 10, which is not a leaf. Hence in order to implement a tree machine a sufficient number of past symbols must be stored as states, but still only the parameters in the tree are needed to compute the probabilities of strings.

3.5 Tunstall's Algorithm

In the first section we discussed traditional coding, where the data sequences are modeled as samples from a random variable, which when extended to sequences amounts to an iid process. The prefix code with minimum mean length can be found with Huffman's algorithm, which is described in most textbooks on coding. Our main interest, however, is in codes and models of random processes, which requires a generalization of the traditional codes and even creation of completely different codes. We begin our discussion with a traditional algorithm due to Tunstall, which not only is more suitable for coding of random processes than Huffman's algorithm but which also is behind the important universal coding and modeling algorithm due to Lempel and Ziv.

Tunstall's algorithm constructs a binary code tree with a desired number of codewords, say m, by the rule: split the node with the maximum probability until m leaves have been generated. The segments of the source strings to be encoded are the paths to the leaves, each of which can be encoded with $\lceil \log m$ bits; namely, as the binary ordinal of the leaf when they are sorted from left to right. Such a coding is called 'Variable-to-Fixed' length coding.

To analyze Tunstall's codes we give a most convenient formula for the mean length of a finite complete tree, where the node probabilities satisfy the axioms of a random process; ie, the root has the probability unity and each internal node's probability is the sum of the children's probabilities.

Let T_i be such a tree with i leaves $S_i = \{s_1, \ldots, s_i\}$. Its mean length is defined to be

$$L(T_i) = \sum_{s \in S_i} P(s)|s|,$$

where $|s|$ is the depth of node s. If we fuse any pair of leaves with common father into the father node, say w, and denote the resulting complete subtree by T_{i-1}, we see that

$$L(T_i) = L(T_{i-1}) + P(w) = \sum_{w \in Int} P(w), \tag{12}$$

where w runs through all the internal nodes of the tree T_i. To simplify the subsequent discussion we consider only Bernoulli sources. The main results hold, however, for many types of dependent sources as well.

Theorem 13 *Tunstall's algorithm produces the tree with m leaves of maximum mean length.*

Proof: Let T_m^* be a tree with m leaves which has maximum mean length $L(T_m^*)$. The 2-leaf tree has length 1 and a 3-leaf Tunstall tree is clearly optimal. Arguing by induction, we see in (12) that for T_i to be optimal we must split the leaf of T_{i-1}^* that has the maximum probability, which completes the proof.

A very fast way to encode the leaf segments in a Tunstall tree is to enumerate them from left to right and encode a leaf w by its ordinal, written in binary with $\log\lceil m$ bits. This Variable-to Fixed coding is also efficient. First, by the theorem just proved the ratio

$$r(m) = \frac{\lceil \log m}{L(T_m)}$$

is minimized by the Tunstall tree over all trees of the same size. Secondly, by Shannon's theorem the mean ideal code length for the leaves is the leaf entropy $H(T_m)$. Clearly $\log m$ is an upper bound for the entropy, and the excess of $r(m)$ over the absolute minimum $H(T_m)/L(T_m)$ depends on how much smaller the entropy is than the uniform $\log m$. But again, no tree with m leaves has greater leaf entropy than a Tunstall tree. To see this consider a Bernoulli process with the parameter $P(0) = p$, which we take to be not smaller than $q = 1 - p$. The entropy defined by the leaf probabilities of the complete subtree T_i is given by

$$H(T_i) = -\sum_{s \in S_i} P(s) \log P(s).$$

Consider again the subtree T_{i-1}, obtained by fusing two leaf sons, say $w0$ and $w1$, to their father node w. Clearly,

$$\begin{aligned} H(T_i) &= H(T_{i-1}) + P(w)\log P(w) - P(w)p\log(P(w)p) - P(w)q\log(P(w)q) \\ &= H(T_{i-1}) + P(w)h(p) \\ &= h(p)\sum_{w \in Int} P(w) = h(p)L(T_i) \end{aligned} \tag{13}$$

where $h(p)$ denotes the binary entropy function. This result generalizes to Markov sources where an alphabet extension tree is placed at each state.

Now, let $\hat{P}_m$ and $\tilde{P}_m$, respectively, be the maximum and the minimum leaf probabilities of a Tunstall tree with m leaves. Further, for $m = 2$ we have $\tilde{P}_m \geq \hat{P}_m q$. Then in terms of code lengths, which may be easier to argue with, $-\log \tilde{P}_m \leq -\log \hat{P}_m + \log 1/q$ for $m = 2$, which forms the induction base. We show by induction that the same inequality holds for all larger Tunstall trees. In fact, suppose it holds up to size n, $n \geq m$. Then by splitting the node with $\hat{P}_n$ we create two new nodes with probabilities $\hat{P}_n p$ and $\hat{P}_n q$, the latter of which will be the smallest $\tilde{P}_{n+1}$ in the $n+1$-leaf tree. By the induction assumption $-\log \tilde{P}_{n+1} \leq -\log P_n(s) + \log 1/q$ for all leaves s in T_n, and we need to check the condition for the new leaf with $\hat{P}_n p$. Indeed, $\hat{P}_n q = \tilde{P}_{n+1} \geq \hat{P}_n pq$, which completes the induction.

Clearly, for all n,

$$\log n \leq \log 1/\tilde{P}_n \leq \log 1/\hat{P}_n + \log 1/q,$$

while $H(T_n) \geq \log 1/\hat{P}_n$, and $H(T_n) \leq \log n$, which give

$$H(T_n) \leq \log n \leq H(T_n) + \log \frac{1}{q}.$$

All told, we see with (13) that

$$r(n) \to h(p)$$

as $n \to \infty$.

3.6 Arithmetic Codes

There is a different kind of code altogether, Arithmetic Code, introduced by me in [20] and developed further by Pasco, [18], Rissanen and Langdon, [19], and later by many others, which is very well suited for coding all kinds of random processes. Such a code is based on two ideas

1. the code is a cumulative probability on strings of length n, sorted alphabetically, and computed recursively in n.

2. for practical implementation all calculations are to be done in fixed size registers.

We consider first the case where no restriction is placed on the precision, and for the sake of simplicity we consider only binary alphabets. Letting $P(x^n)$ denote a probability function that defines a random process we take the cumulative probability $C(x^n)$ as the code of x^n. From a binary tree with $s0$ the left son of s we get immediately the recursion:

$$C(x^n 0) = C(x^n) \tag{14}$$

$$C(x^n 1) = C(x^n) + P(x^n 0) \tag{15}$$

$$P(x^n i) = P(x^n)P(i|x^n),\ i = 0, 1. \tag{16}$$

A minor drawback of this code is that it is one-to-one only for strings that end in 1. The main drawback is that even if we begin with conditional probabilities $P(i|x^n)$, written in a finite precision, the multiplication in the third equation will keep on increasing the precision, and soon enough we exceed any register we pick. What is needed is to satisfy the marginality condition in a random process without increasing the precision. There are several ways to do it. The most straightforward way is to replace the exact multiplication in the third equation above by the following:

$$\bar{P}(x^n i) = \lfloor \bar{P}(x^n)\bar{P}(i|x^n) \rfloor_q, \ i = 0, 1,$$

where $\lfloor z \rfloor_q$ denotes the truncation of a fractional binary number $z = 0.0 \ldots 01...$ such that there are just q digits following the first occurrence of 1. It is clear then that the addition in the update of the code string can also be done in a register of width $q + 1$, except for a possible overflow, which is dealt with later. We shall show presently that the code is one-to-one on strings that do not end in a zero, and hence decoding can be done.

Consider the two strings which differ for the first symbol x_t:

$x^n = u1w$

$y^m = u0v.$

Unless empty, the strings w and v end in the symbol 1. In case nonempty the code strings can be written as

$C(x^n) = C(u) + \bar{P}(u0) + \bar{P}(u10^r0) + \ldots$

$C(y^m) = C(u) + \bar{P}(u00^s0) + \ldots,$

where 0^r and 0^s denote strings of 0 of length $r \geq 0$ and $s \geq 0$, respectively.

We see that regardless of what w and v are,

$\bar{P}(u0) \leq C(x^n) - C(u)$, for $x_t = 1$,

$\bar{P}(u0) > C(y^m) - C(u)$, for $x_t = 0$.

Obviously the first is true and so is the second if v is empty. But even when it is nonempty, we have

$$\bar{P}(u0) \geq \sum_i \bar{P}(u0i) > \bar{P}(u00) + \bar{P}(u010) + \bar{P}(u0110) + \ldots,$$

where the right hand side results from $v = 11 \ldots 1$ which gives the largest possible sum of the addends.

We conclude that the decoding can be done with the rule:

$$x_t = 1 \iff \bar{P}(u0) \leq C(x^n) - C(u),$$

which, in turn, can be decided by looking at the $q + 1$ leading bits of the code string.

The problem of a possible overflow can be solved by 'bit stuffing'. In fact, when the register where the code is being updated becomes full, a further addition will propagate a '1' in the already decoded portion of the string. In order to prevent this from happening an additional bit 0 is added to the code string in the bit position immediately to the left of the register. An overflow will turn this into a '1', which can be detected by the decoder and taken into account.

We make an estimate of the code length. We see in (16) that the code length is given in essence by the number of leading zeros in $\bar{P}(x^n 0)$ plus the size of the register q needed to write down the probability itself. If the data have been generated by a process $P(x^n)$ we have $\bar{P}(i)/P(i) \geq 1 - 2^{r-q}$, where r is the smallest integer such that $\min_i P(i) \geq 2^{-r}$, and q is taken larger than r. We then have

$$-\log \bar{P}(x^n) \leq -\log P(x^n) + n \log(1 - 2^{r-q}).$$

Since further also $\bar{P}(x^n i)/\bar{P}(x^n)\bar{P}(i) \geq 1 - 2^{r-q}$ we get

$$\frac{1}{n} L(x^n) \leq -\frac{1}{n} \log P(x^n) + q/n + 2^{r-q},$$

which for large enough n can be made as close to the ideal code length as desired by taking q large enough. In this estimate we have excluded the effect of the bit stuffing, which in general extends the code length by an ignorable amount.

The quantity $\bar{P}(x^n)$ no longer defines a probability measure and a random process; rather, it defines a semi measure. It still requires a multiplication, which sometimes is undesirable. It is possible, however, to remedy these defects with a tolerable increase in the code length. Consider the following recursive definition of a process $p(x^n)$: First, let $p(0)$ denote a probability of the (binary) symbol being 0, written as a binary fraction with w fractional digits. We take it to be less than or equal to 1/2; this low probability symbol could of course be 1, and in fact in general it is sometimes 0 sometimes 1 depending on the past string. Now define $a(x^n) = 2^{L(x^n)} p(x^n)$, where $L(x^n)$ is the integer that satisfies $.75 \leq a(x^n) < 1.5$. In other words, for a string $s = x^n$, $a(s)$ is the decreasing probability $p(s)$ normalized to a fixed size register. Then calculate recursively

1. $p(s0) = 2^{-L(s)} p(0)$
2. $p(s1) = p(s) - 2^{-L(s)} p(0)$.

We see that $p(s)$ has no more than w digits following the first 1 and that it defines a random process.

Encoding Algorithm:

Initialize register C with 0's and set A to 10...0.

1. Read next symbol. If none exists, shift contents of C left w positions to obtain the final code string.

2. If the next symbol is the low probability symbol, say x, replace the contents of A by $p(x)$ ($p(x) \leq 1/2$) and go to 4.

3. If the next symbol is the high probability symbol, add to register C the number $p(x)$ and subtract from A the same number.

4. Shift the contents of both registers left as many positions as required to bring A to the range $[.75, 1.5)$. Go to 1.

Decoding Algorithm:

Initialize register C with w leftmost symbols of the code string and set A to 10...0.

1. Form an auxiliary quantity T by subtracting $p(x)$ from the contents of C. Test if $T < 0$.

2. If $T < 0$, decode the symbol as x. Load A with $p(x)$.

3. If $T \geq 0$, decode the symbol as the high probability symbol. Load C with T, subtract $p(x)$ from A.

4. Shift the contents of both registers left as many positions as required to bring A to the range $[.75, 1.5)$ and read the same number of symbols from the code string in the vacated positions in C. If none exist, stop. Else, go to 1.

Much as in the case of random variables, where prefix code lengths and probability distributions are in one-to-one correspondence, we can also equate certain codes and random processes.

A code C for a random process is a one-to-one map

$$C : A^* \to B^*.$$

The prefix condition suggests the following generalization:

$$L(i|x^n) \equiv L(x^n i) - L(x^n) \tag{17}$$

$$\sum_i 2^{-L(i|x^n)} \leq 1, \tag{18}$$

for all x^n, where $L(x^n)$ denotes the length of the code string $C(x^n)$. Hence, if the Kraft inequality holds with equality the code lengths define a random process. This time the integer length requirement for the difference $L(i|x^n)$ would prevent us from representing processes over small alphabets accurately as codes; for instance, for the binary Bernoulli process the smallest positive integer value

1 for $L(0|x^n) = L(0)$ would give the maximum value $P(0) = 1/2$, which is not an accurate model for a process where $P(0) = .9$, say. However, with arithmetic coding we can accurately represent any process, because the codewords themselves represent cumulative probabilities defining a random process. Hence, for some symbols there is no increase in the code length from $L(x^n)$ to $L(x^n i)$, which would make the Kraft-inequality to fail. The appropriate generalization of the Kraft-inequality is simply the statement that

$$\sum_i \bar{P}(i|x^n) \leq 1,$$

where $\bar{P}(i|x^n)$ is the quantity with which the arithmetic code is constructed recursively, and which defines the conditional probability generating a random process. We call such a code *regular*. Hence, with help of arithmetic codes we can state again that a regular code is equivalent with a random process.

4 Universal Coding

In practice the data compression problem is to encode a given data string when no probability distributions are given. For a long time the problem was tackled by designing codes on intuitive basis, until it was realized that such codes always incorporate some type of a model which is 'universal' in an entire class $\mathcal{M} = \{P_i(x^n)\}$ of stationary processes, where i ranges over a countable set Ω. Since we know by Shannon's theory how to design a code with each member in the class so that the mean length is close to the corresponding entropy, a reasonable requirement of any code which claims to represent the entire class and be *universal*, relative to the class, is that it has the following property:

> The mean per symbol code length should approach the per symbol entropy in the limit, no matter which process in the class generates the data:
>
> $$\frac{1}{n} E_i L(X^n) \to H_i(X), \tag{19}$$
>
> where the expectation is with respect to P_i and $H_i(X)$ is the corresponding per symbol entropy.

If we can construct or even define codes with such a property, we may start asking more: We would like the mean per symbol length to approach the entropy at the fastest possible rate. A little reflection will reveal that this is too much to ask, for by Shannon's theorem we could achieve the entropy instantly if we guess correctly the data generating process. However, if there are a countable number of index values we cannot expect to be right with our guess, and the exceptional case does not seem to be of much interest in practice. Nevertheless, for conceptual reasons we must worry even about such remote events, and we need to be a bit careful in specifying in what sense a code can have the fastest rate. We do this in a later subsection and turn next to code designs that will satisfy the first requirement.

4.1 Lempel-Ziv - and Ma - Algorithms

An elegant algorithm for universal coding is due to Lempel and Ziv, [14]. The algorithm parses recursively the data string x^n into nonoverlapping segments by the rule:

- Starting with the empty segment, each new one added to the collection is one symbol longer segment than the longest match so far found.

Lempel and Ziv encoded each segment as the pair (i, y), where i, written as a binary number, gives the index of the longest earlier found match in the list, and y is the last added symbol. The code length for the string x is then given by

$$L_{LZ}(x) = m(x) + \sum_{j=1}^{m(x)} \lceil \log j,$$

where $m(x)$ denotes the number of parsed segments.

It may appear that there is no probability model behind the algorithm. However, the parsing actually incorporates occurrence counts of symbols, which in effect define a random process and a sequence of Tunstall trees permiting a Variable-to-Fixed length coding (more or less, because the tree keeps on growing!). To see this consider the recursive construction of a binary tree from a given binary string $x_1, \ldots, x_n$:

1. Start with a 2-node tree, the root node marked with a counter initialized to 2, and each son's counter initialized to 1.
2. Recursively, use the previous tree to parse the next segment off the remaining string as the path from the root to the deepest node in the tree. While climbing the tree, increment by one the count of each node visited.
3. If the last visited node is a leaf, split it to create two new son nodes, and initialize their counts to 1.

If we denote the parsed segments by σ_i, we can write the string in these terms as: $x^n = \sigma_1, \sigma_2, \ldots, \sigma_m$, which corresponds exactly to the parses found with the LZ algorithm; the last segment σ_m may not be complete. Notice that the leaves of the tree always have the count of unity, and each father node's count is the sum of the sons' counts. Hence, if we divide all the node counts $n(s)$ by the root count $n(\lambda)$, each node s gets marked with the probability $P(s) = n(s)/n(\lambda)$. Also, the leaf with the highest count gets split - just as in Tunstall's algorithm.

Write $x^t = \sigma_1 \ldots \sigma_i z_j$, where z_j denotes a prefix of the full segment σ_{i+1}. By identifying z_j with the corresponding node in the so far constructed tree we can define the conditional probability of the symbol x_{t+1}, given the past string, as

$$P(x_{t+1}|x^t) = n(z_j x_{t+1})/n(z_j).$$

When we multiply all these conditional probabilities along the next fully parsed segment σ_{i+1} we get its conditional probability as

$$P(\sigma_{i+1}|\sigma^i) = 1/n(\lambda),$$

where $n(\lambda) = i + 2$. This last equality follows from the fact that each parsed segment adds 1 to the root count, which is initialized to 2. Finally, the probability of a string x with $m(x)$ full segments is

$$P(x) = 1/(m(x) + 1)!.$$

This gives the *ideal* code length as $-\log P(x)$, which is somewhat better than the Lempel-Ziv code length, because of the fairly crude way the coding is done in that code.

The Lempel-Ziv code is universal in a very large class of processes $\{P\}$, namely, the class of stationary ergodic processes, in the sense that

$$\frac{1}{n} E_P L_{LZ}(X^n) \to H(P),$$

no matter which process P generates the data. The same holds also in the almost sure sense. The notation E_P denotes the expectation with respect to the process P. Hence, it does satisfy the minimum optimality requirement stated above. The rate of convergence can be shown to be

$$\frac{1}{n}[E_\theta L(X^n) - H(X^n)] = O(\frac{1}{\log n}),$$

which in the case of the class of Markov and tree machine processes $P(x^n; T)$ will be seen not to be too good.

4.2 Universal Context Coding

Elegant as the LZ - algorithm is, it has the defect that 'it doesn't know' when it has learned enough of the constraints to stop searching for more by an increase in the complexity; ie, the block length. This is reflected by the slow approach to the ideal, the entropy, if the data in fact were generated by some Markov process or one approximately like it. We shall describe below another algorithm which has such a learning power. But first we must discuss the basic case of encoding binary strings generated by some Bernoulli process, whose symbol probability $P(x = 0) = p$ is not known. In other words, the class of processes considered is $\mathcal{B} = \{P(x^n; p)\}$, where

$$P(x^n; p) = p^{n_0}(1 - p)^{n-n_0}$$

and n_0 denotes the number of 0's in the string x^n.

Begin with a coding method, called *predictive*, for an obvious reason: Encode the very first symbol x_1 with the probability 1/2. Knowing now something about the symbol occurrences, encode the next symbol x_2 with the probability $P(x_2|x_1) = 2/3$, if $x_2 = x_1$, and, of course, with 1/3, otherwise. In general, then, put

$$P(x_{t+1} = 0|x^t) = \frac{n_0(x^t) + 1}{t + 2}. \tag{20}$$

where $n_0(x^t)$ denotes the number of times the symbol 0 occurs in the string x^t. Such a scheme was invented by Laplace, when he was asked for the probability that the sun would rise tomorrow. It is an easy exercise for the reader to show that this scheme defines the following probability for a sequence x^n

$$P(x^n) = \frac{n_0!(n-n_0)!}{(n+1)!} = \binom{n}{n_0}^{-1}(n+1)^{-1}, \tag{21}$$

where we now wrote $n_0 = n_0(x^n)$. Hence the code length we get is

$$L(x^n) = -\log P(x^n) = \log \binom{n}{n_0} + \log(n+1). \tag{22}$$

Exactly the same probability results from the formula for the marginal distribution

$$P(x^n) = \int_0^1 P(x^n; p)dp = \int_0^1 p^{n_0}(1-p)^{n-n_0}dp, \tag{23}$$

where we pick a uniform prior for the probability $p = P(0)$ that the symbol is 0. Such an integral is a Dirichlet's integral with the value given in (21).

In particular for strings where n_0/n is close to zero or unity, there is a better estimate for the conditional 'next' symbol probability than in (20), namely

$$P(x_{t+1} = 0|x^t) = \frac{n_0(x^t) + 1/2}{t+1}. \tag{24}$$

The reason for this will be explained in a later section.

We are now ready to describe an algorithm, called Algorithm Context, introduced in [21] and analyzed in [34], which is universal in a large class of Markov processes. In particular, it provides coding of any string over a given alphabet, say binary, in such a manner that not only does the per symbol code length approach that of the data generating Markov process, whatever that process is, but the approach is the fastest possible in a sense made precise later. The algorithm provides a better compression than the LZ - algorithm for strings that have Markovian type properties.

The algorithm has two stages, which can be combined but which we describe separately, namely, Algorithm A for growing a tree, and algorithms for, in effect, tree pruning, or more accurately, for Choice of the Encoding Node. For simplicity we also use the reverse ordering of the past symbols.

Algorithm A

1. Begin with 1-node tree, marked with counts $(c_0, c_1) = (1, 1)$ and code length $L(\lambda) = 0$.

2. Read next symbol $x_{t+1} = i$. If none exists, exit. Otherwise, climb the tree by reading the past string backwards $x_t, x_{t-1}, \ldots$ and update the count c_i by unity and the code length $L(s)$ by $-\log P(i|s)$ obtained from (20) or (24) at every node s met until one of the two following conditions are satisfied:

3. If the node whose count c_i becomes two after the update is an internal node, go to 2. But if it is a leaf, create two new nodes and initialize their counts to $c_0 = c_1 = 1$. Go to 2.

This portion of the algorithm creates an in general lopsided tree, because the tree grows along a path which is traveled frequently. Further, each node s represents a 'context' in which symbol i occurs about c_i times in the entire string. In fact, since the path from the root to the node s is a binary string $s = i_1, i_2 \ldots, i_k$ the substring $i_k, \ldots, i_1, i$ occurs in the entire string x^n very close to $c(i|s)$ times, where $c(i|s)$ is the count of i at the node s. Notice that the real occurrence count may be a little larger, because the substring may have occurred before node s was created. Also, the following important condition is satisfied

$$c(i|s0) + c(i|s1) = c(i|s), \tag{25}$$

for all nodes whose counts are greater than 1. Finally, the tree created is complete. There exist versions of the algorithm which create incomplete trees.

Choice of Encoding Nodes

Since each symbol x_{t+1}, even the first, occurs at least in one node s, we can encode it with the ideal code length given by Equation (21), where the counts are those at the node s. If a symbol occurs in node s, other than the root, it also occurs in every shorter node on the path to s, which raises the question which node we should pick. There are more than one way to select this 'encoding' node, and we'll describe two.

We use the notation $x^t(s)$ for the substring of the symbols of the 'past' string x^t that have occurred at the node s and $L(x^t(s))$ for its code length, which is computed predictively by use of the conditional probabilities (20) or (24).

Rule 1.

For x_{t+1} pick the first node s such that

$$L(x^t(s)) \leq \sum_{i=0}^{1} L(x^t(si).$$

Notice that both sides refer to the code length for the same symbols, the left hand side when the symbols occur at the father node and the right hand side when they occur at the son nodes. The rule, then, finds the first node where the son nodes' code length is no longer better than the father node's code length. Clearly, because the algorithm does not search the entire path to the leaf such a strategy may not find the best node.

Rule 2.

It is possible to find an optimal subtree of any complete tree, where each node is marked with a code length for the symbols that occur at it, and where the symbols that occur at the node also occur at the son nodes. For the final tree, say $\mathcal{T}$, obtained by Algorithm A this is insured by Equation (25). We write now the code lengths $L(x^n(s))$ more simply as $L(s)$.

Algorithm PRUNE:

1. Initialize: For the tree $\mathcal{T}$ put $\bar{S}$ as the set of the leaves. Calculate $I(s) = L(s)$ at the leaves.
2. Recursively, starting at the leaves compute at each father node s
$$I(s) = \min\{L(s), \sum_{j=0}^{1} I(sj)\}. \tag{26}$$
If the first element is smaller than or equal to the second, replace the sons in the set $\bar{S}$ by the father; otherwise, leave $\bar{S}$ unchanged.
3. Continue until the root is reached.

It is easy to convert Algorithm Context into a universal code. All we need is to have an arithmetic coding unit, which receives as the input the 'next' symbol x_{t+1} and the predictive probability (24) at its encoding node, say $s^*(t)$, rewritten here
$$P(x_{t+1} = 0|s^*(t)) = \frac{c(0|x^t(s^*(t))) + 1/2}{|x^t(s^*(t))| + 1}. \tag{27}$$
Notice that only one coding unit is needed - rather than one for every encoding node. This is because an arithmetic code works for any set of conditional probabilities defining a random process, and, clearly, the conditional probabilities of Algorithm Context define a process by
$$P(x^n) = \prod_{t=0}^{n-1} P(x_{t+1}|s^*(t)), \tag{28}$$
where $s^*(0) = \lambda$, the root node.

What is the code length of this universal code? To answer that question we assume that the data are generated by a stationary Markov source of some order K, where all the $k = 2^K$ parameters, one at each state, namely $P(x = 0|s)$, are irreducible. One can show, [34], that the mean ideal code length resulting from the universal code defined by Equation 27 satisfies
$$\frac{1}{n} E L_{AC}(X^n) = H(X) + \frac{k}{2n} \log n + o(n^{-1} \log n),$$
where $H(X)$ is the entropy of the source. Moreover, we see later that no universal code exists where the mean per symbol code length approaches the entropy faster. We further see that the LZ-code is not optimal in this stronger sense.

Part II
STATISTICAL MODELING

Statistical modeling is about finding general laws from observed data, which amounts to extracting information from the data. Despite the creation of information theory half a century ago with its formal measures of information, the entropy and the Kullback-Leibler distance or the relative entropy, there have been serious difficulties in applying them to make exact the idea of information extraction for model building. The main problem is that these measures refer to information in probability distributions rather than in data. It is perhaps an indication of the strength of dogma in statistical thinking that only relatively recently the information extraction process was formalized by Akaike in 1973, [1], as one of searching for the model in a proposed collection that is closest to the 'true model' as a probability distribution in terms of the Kullback-Leibler distance. The 'true model' is assumed to lie outside the collection, and it is known only through the observed data from which the distance must be estimated. This turns out to be difficult, and Akaike's criterion AIC amounts to an asymptotic estimate of the mean Kullback-Leibler distance, the mean taken over the estimated models in the class, each having the same number of parameters.

Although Akaike's work represents an important deviation in thinking from tradition in avoiding the need to add artificial terms to the criterion to penalize the model complexity, there are difficulties with the criterion. The first, of course, is the need to assume the existence of the 'true' underlying model. As a matter of fact, even in such a paradigm the Kullback-Leibler distance gets reduced as the number of parameters in the fitted models is increased, which is the very problem we are trying to avoid. That this gets overcome by the estimated mean distance is not a great consolation for how can we explain the paradox that the ideal criterion, where everything is known, fails but the estimation process, which one would expect to make things worse, produces a criterion that works better than the ideal. Besides, it is known that if we assume the 'true' model to be in the set of the fitted models, the AIC criterion will not find it no matter how large the data set is, which suggests a lack of self consistency.

There is another quite different way to formalize the problem of extracting information from data, which appears to be much more natural. It is based on the idea, inspired by the theory of Kolmogorov complexity which will be reviewed below, that the complexity of a data set is measured by the fewest number of bits with which it can be encoded when advantage is taken of a proposed class of models. Hence, the complexity measure is relative to the class of models, which then act like a language allowing us to express the properties in the data, and, as we shall see, the information in the data. This makes intuitive sense, for if the language is poor we expect to be able to learn only gross properties. If again the language is very rich we can express a large number of properties, including spurious 'random' quirks. This raises the thorny issue of deciding how much and which properties in data we want to and can learn. Our solution will be based on the idea that the portion in the data that cannot be compressed with the class of models available will be *defined* to be uninteresting

'noise', and the rest is what we want to learn, the useful learnable *information*. We may state that to achieve such a decomposition of data is the purpose of all modeling.

When we formalize the two fundamental notions, the complexity and information, relative to a class of models, there is no need to assume any metaphysical 'true model', which somehow would represent not only the information in the given data set but even in all future data sets generated by sampling the 'true model'. Although our formalization does provide a solid foundation for a theory of modeling it, of course, does not solve all the model selection problems, for the selection of a good class of models remains. In fact, there can be a lot of prior information that others have gathered from data generated under similar conditions. However, this we cannot formalize in any other way than saying that such information should be used to suggest good or at least promising classes of models. The noncomputability of the Kolmogorov complexity (see below) implies that the process of selecting the optimal model and model class will always have to be done by informal means where human intelligence and intuition will play a dominant role. No other currently used methodology of model selection has revealed similar limits to what can and cannot be done.

5 Kolmogorov Complexity

For a string (x^n), generated by sampling a probability distribution $P(x^n)$, we have already suggested the ideal code length $-\log P(x^n)$ to serve as its complexity, the Shannon complexity, with the justification that its mean is for large alphabets a tight lower bound for the mean prefix code length. The problem, of course, arises that this measure of complexity depends very strongly on the distribution P, which in the cases of interest to us is not given. Nevertheless, we feel intuitively that a measure of complexity ought to be linked with the ease of its description. For instance, consider the following three types of data strings of length $n = 20$, where the length actually ought to be taken large to make our point:

1. 01010101010101010101

2. 00100010000000010000

3. generate a string by flipping a coin 20 times

We would definitely regard the first string as 'simple', because there is an easy rule permitting a short description. Indeed, we can describe the string by telling its length, which takes about $\log n$ bits, and giving the rule. If the length n were actually much greater than 20, the description length of the rule: "alternate the symbols starting with 0", encoded in some computer language as a binary string, could be ignored in comparison with $\log n$. The amount of information we can extract from the rule is also small, because there is not much information to be extracted.

The second string appears more complex, because we cannot quite figure out a good rule. However, it has $n_0 = 17$ zeros and only $n - n_0 = 3$ ones, which may be taken advantage of. In fact, there are just $N = 20!/(3!17!)$ such strings, which is quite a bit less than the number of all strings of

length 20. Hence, if we just sort all such strings alphabetically and encode each string as its ordinal, written in binary, we can encode the given string with about $\log N \cong 14$ bits. This suggests that we are modeling the string more or less by the class of Bernoulli models. The information that can be extracted that way is a bit more than in the previous case but the amount is not all that much.

The third string is maximally complex; no rule helps to shorten the code length, about $n = 20$, which results if we write it down symbol for symbol. However, the information is small; after all, there is nothing to be learned other that the string looks random.

In the preceding discussion the language in which the objects are to be described was left unspecified and vague. An ingenious way to specify the language is to take it as a universal programming language where each program in the machine language form is a binary string. All the usual programming languages are universal in that any recursive function of any number of arguments can be programmed in them. Let U be a computer that can execute programs $p_U(x^n)$, each delivering the desired binary data string $x^n = x_1, \ldots, x_n$ as the output. There are clearly a countable number of such programs for each data string, because we can add any number of instructions canceling each other to create the same string. When the program $p_U(x^n)$ is fed into the computer, the machine after a while prints out x^n and stops. We may then view the computer as defining a many-to-one map decoding the binary string x^n from any of its programs. In the terminology above, a program $p_U(x^n)$ is a codeword of length $|p_U(x^n)|$ for the string x^n. Notice that when the machine stops it will not start by itself, which means that no program that generates x^n and stops can be a prefix of a longer program that does the same. Hence, if we place the set of all programs with the said property in a binary tree, where they appear as leaves, they satisfy the Kraft-inequality and we have an infinite prefix code.

The *Kolmogorov complexity* of a string x^n, relative to a universal computer U, is defined as

$$K_U(x^n) = \min_{p_U(x^n)} |p_U(x^n)|.$$

In words, it is the length of the shortest program in the language of U; ie, in the set of its programs, that generates the string, [30], [13].

The set of all programs for U may be ordered, first by the length and then the programs of the same length alphabetically. Hence, each program p has an index $i(p)$ in this list. Importantly, this set has a generating grammar, which can be programmed in another universal computer's language with the effect that each universal computer can execute another computer's programs by use of this translation or compiler program, as it is also called. Hence, in the list of the programs for U there is in some place a shortest program $p_U(V)$, which is capable of translating all the programs of another universal computer V. But this then means that

$$K_V(x^n) \leq K_U(x^n) + |p_V(U)| = K_U(x^n) + C_U,$$

where C_U does not depend on the string x^n. By exchanging the roles of U and V we see that the dependence of the complexity of long strings x^n on the particular universal computer used is diluted.

We cannot, of course, claim that the dependence has been eliminated, because the constant C_U can be as large as we wish by picking a 'bad' universal computer. However, by recognizing that for a reasonably long string x^n the shortest program in a universal computer, designed without knowledge of the particular string x^n, will have to capture essentially all the regular features in the string, we can safely regard such a U as fixed, and the Kolmogorov complexity $K_U(x^n)$ provides us a virtually absolute measure of the string's complexity.

5.1 Universal Algorithmic Model

The Kolmogorov complexity provides a universal model par excellence, which we can even equate with a probability measure, namely,

$$P_K(x^n) = C2^{-K_U(x^n)}, \tag{29}$$

where

$$C = 1/\sum_{y\in B^*} 2^{-K_U(y)},$$

the sum being finite by the Kraft-inequality. With this we can construct a universal process defined recursively as follows

$$P(x^t i) = P(x^t)P(i|x), \tag{30}$$

where $x^t i$ is the string x^t followed by the symbol i, and $P(i|x) = P_K(x^t i)/\sum_j P_K(x^t j)$. The universal distribution in (29) has the remarkable property of being able to mimic *any* computable probability distribution $Q(x^t)$ in the sense that

$$P_K(x^t) \geq AQ(x^t), \tag{31}$$

where the constant A does not depend on t. The same holds about the universal process.

5.2 Kolmogorov Sufficient Statistics

Although we have identified the shortest program for a string x with its 'ideal' model, because it will have to capture all the regular features in the string, it does not quite correspond to intuition. In fact, we associate with a model only the regular features rather than the entire string generating machinery. For instance, a purely random string has no regular features, and we would like its 'model' to be empty having no complexity. We outline next the construction due to Kolmogorov by which the desired idea of a model is captured in the algorithmic theory.

The Kolmogorov complexity is immediately extended to the conditional complexity $K(x|y)$ as the length of the shortest program that generates string x from another string y and causes the computer to stop. One sees readily that

$$\begin{aligned} K(x,y) &\stackrel{\cdot}{\leq} K(x) + K(y|x) &(32)\\ &\stackrel{\cdot}{\leq} K(y) + K(x|y); &(33)\end{aligned}$$

we mean by $\doteq$ 'almost equality', say equality up to a constant not depending on the length of the strings x and y, and in the same sense we denote by $\dot{<}$ and $\dot{\leq}$ approximate inequalities. We now take y to be a program that describes 'summarizing properties' of string $x = x^n$. A 'property' of data may be formalized as a subset A where the data belongs along with other sequences sharing this property. The amount of properties is in inverse relation to the size of the set A. Hence, the smallest singelton set $A = \{x^n\}$ represents *all* the conceivable properties of x^n, while the set consisting of all strings of length n assigns no particular properties to x^n (other than the length n). We may now think of programs consisting of two parts, where the first part describes optimally such a set A with the number of bits given by the Kolmogorov complexity $K(A)$, and the second part merely describes x^n in A with about $\log |A|$ bits, $|A|$ denoting the number of elements in A. The sequence x^n gets then described in $K(A) + \log |A|$ bits. We consider now sets A for which

$$K(A) + \log |A| = K(A) + \max_{y \in A} K(y|A) \doteq K(x^n). \tag{34}$$

Suppose for a string x^n we ask for the maximal set $\hat{A}$ for which (34) holds. Equivalently, it is a set with the smallest complexity. Such a set $\hat{A}$, or its defining program, is called Kolmogorov's *minimal sufficient statistic* for the description of x^n. We define $K(\hat{A})$ to be Kolmogorov or the *algorithmic information* in the string x^n.

There is a generalization of Kolmogorov's *minimal sufficient statistic*, also due to him, [32]. Consider the function

$$h_{x^n}(\alpha) = \min_A \{\log |A| : x^n \in A, K(A) \leq \alpha\}. \tag{35}$$

Clearly, $\alpha < \alpha'$ implies $h_\alpha(x^n) \geq h'_\alpha(x^n)$ so that $h_{x^n}(\alpha)$ is nonincreasing function of α, and $h_{x^n}(\alpha) = \log\{x^n\} = 0$ for $\alpha \doteq K(x^n)$. Because $K(x^n|A) + K(A) \dot{>} K(x^n) \doteq \min_A\{K(x^n|A) + K(A)\}$, $h_{x^n}(\alpha) + \alpha \dot{>} K(x^n)$, and $h_{x^n}(\alpha)$ is above the line $L(\alpha) = K(x^n) - \alpha$ to within a constant. They are equal at the smallest α

$$\min\{\alpha : h_{x^n}(\alpha) + \alpha \doteq K(x^n)\}, \tag{36}$$

and we get the Kolmogorov's *minimal sufficient statistic* decomposition of x^n.

All these findings while very interesting and conceptually important will not solve our universal coding and modeling problems. This is because the Kolmogorov complexity is noncomputable. What this means is that there is no program such that, when given the data string x^n, the computer will calculate the complexity $K_U(x^n)$ as a binary integer. The usual proof of this important theorem due to Kolmogorov appeals to the undecidability of the so-called halting problem. We give here another proof due to Ragnar Nohre, [17].

Suppose to the contrary that a program q_U with the said property exists. We can then write a program p_U, using q_U as a subroutine, which finds the shortest string $z^n = p_U(\lambda)$ such that

$$K_U(z^n) > |p_U| = n.$$

In essence, the program p_U examines the strings x^t, sorted alphabetically by nondecreasing length, computes $K_U(x^t)$ and checks with q_U if this inequality holds. It is clear that such a shortest string

exists, because $K_U(x^t)$ has no upper bound and $|p_U|$ has some fixed finite length. But by the definition of Kolmogorov complexity $K_U(z^n) \leq |p_U(\lambda)|$, which contradicts the inequality shown. Hence, no program q_U with the assumed property exists.

It is possible to estimate the complexity from above with increasing accuracy, but we cannot have an idea of how close to the complexity we are. Despite the negative content of the proved result, it is, or should be, of utmost importance in statistical inference, because it clearly sets a limit for what can be meaningfully asked. Any statistical effort trying to search for the 'true' underlying data generating distribution by systematic (= mechanical) means is hopeless - even if we by the 'truth' mean only the best model. Therefore, human intuition and intelligence in this endeavor are indispensable. It is also clear in this light that, while many physical phenomena are simple, because their data admit laws (or, as Einstein put it, God is good), to find the laws is inherently difficult! It has generally taken geniuses to discover even some of the simplest laws in physics. In the algorithmic theory most strings are maximally complex, although we cannot prove a single one to be so, which raises the intriguing question whether most data strings arising in the physical world are also maximally complex or nearly so; ie, obeying *no* laws.

6 Stochastic Complexity and Information in Data

As we discussed above the problem of main interest for us is to obtain a measure of both the complexity and the (useful) information in a data set. As in the algorithmic theory the complexity is the primary notion, which then allows us to define the more intricate notion of information. Our plan is to define the complexity in terms of the shortest code length when the data is encoded with a class of models as codes. In the previous section we saw that this leads into the noncomputability problem if we let the class of models include the set of all computer programs, a 'model' identified with a computer program (code) that generates the given data. However, if we select a smaller class the noncomputability problem can be avoided but we have to overcome another difficulty: How are we to define the shortest code length? It seems that in order not to fall back to the Kolmogorov complexity we must spell out exactly how the distributions as models are to be used to restrict the coding operations. In universal coding we did just that by doing the coding in a predictive way, in Lempel-Ziv code by the index of the leaf in the tree of the segments determined by the past data, and in Context Coding by applying an arithmetic code to each 'next' symbol, conditioned on a context defined by the algorithm as a function of the past data. Here we adopt a different strategy: we define the idea of shortest code length in a probabilistic sense, which turns out to satisfy practical requirements. To do that we must be more formal about models.

6.1 Models

For data sets of type $(y^n, x^n) = (y_1, x_1), \ldots, (y_n, x_n)$, where $y_t \in Y$ and $x_t \in X$ are data of any kind, consider a decomposition

$$y^n = \hat{y}^n + e^n \tag{37}$$

$$\hat{y}_t = F(x_t; \theta) \tag{38}$$

where F is a parametric function defining a part of the *model*. The remainder e^n is viewed as noise, for which we need something to measure its size. This can be done by an error function $\delta(y, \hat{y})$, which is extended to sequences by

$$\delta(y^n, \hat{y}^n) = \sum_t \delta(y_t, \hat{y}_t).$$

An important error function is a conditional probability or density function

$$\delta(y_t, \hat{y}_t) = -\log f(y_t|\hat{y}_t),$$

which is seen to define a parametric class of probability models $\mathcal{M} = \{f(y^n|x^n; \theta)\}$ by independence.

There is a definite sense in which virtually all types of loss functions can be represented by the ideal code length and hence a probability model. In fact, given a distance function define the density function

$$f(y|x; \lambda, \theta) = Z^{-1}(\lambda, \theta) e^{-\lambda \delta(y, \hat{y})}, \tag{39}$$

where $Z(\lambda, \theta) = \int e^{-\lambda \delta(y, \hat{y})} dy$. The existence of the integral clearly puts a constraint on the distance function, which, however, is not too severe. For instance, $\delta(y - \hat{y}) = |y - \hat{y}|^\alpha$ for $\alpha > 0$ gives a finite integral. Then extending by independence we get

$$-\ln f(y^n|x^n; \lambda, \theta) = n \ln Z(\lambda, \theta) + \lambda \sum_t \delta(y_t, \hat{y}_t)). \tag{40}$$

We often consider unconditional probability models $f(y^n; \theta)$, in which the data variable then is often written as x^n.

Important classes of probability models are the exponential families

$$f(x^n; \lambda) = Z^{-1}(\lambda) e^{-\lambda' \phi(x^n)}, \tag{41}$$

where $\lambda = \lambda_1, \ldots, \lambda_k$ denotes a parameter vector and $\phi(x^n) = \phi_1(x^n), \ldots, \phi_k(x^n)$ functions of x^n. Let $A(\lambda) = \ln Z(\lambda)$ and let $\dot{A}(\lambda)$ denote the vector of the derivatives $dA(\lambda)/d\lambda_i$. Then by taking the derivatives with respect to λ in the equation

$$\int f(x^n; \lambda) dx^n = Z(\lambda) \tag{42}$$

we get

$$\dot{A} = \frac{\dot{Z}(\lambda)}{Z(\lambda)} = -E_\lambda \phi(X^n), \tag{43}$$

which holds for all λ. The maximum likelihood estimate $\hat{\lambda} = \hat{\lambda}(x^n)$ is seen to satisfy

$$\dot{A}(\hat{\lambda}) + \phi(x^n) = 0, \tag{44}$$

giving

$$-\ln f(x^n; \hat{\lambda}(x^n)) = A(\hat{\lambda}) + \hat{\lambda}'\phi(x^n). \tag{45}$$

Next, consider the entropy H_λ

$$-E_f \ln f(X^n; \lambda) = A(\lambda) + \lambda' E_\lambda \phi(X^n). \tag{46}$$

If we evaluate this at $\hat{\lambda}$ we get with (43) and (44) the remarkable equality

$$H_{\hat{\lambda}} = -\ln f(x^n; \hat{\lambda}). \tag{47}$$

This shows among other things that the maximized likelihood depends only on $\hat{\lambda}$; ie, it is the same for all y^n such that $\hat{\lambda}(y^n) = \hat{\lambda}$. Hence for all such y^n we have the factorization

$$\begin{aligned} f(y^n; \hat{\lambda}(x^n)) &= f(y^n, \hat{\lambda}(x^n)) & (48)\\ &= f(y^n|\hat{\lambda}(x^n))h(\hat{\lambda}(x^n)). & (49) \end{aligned}$$

An important related class of probability models are induced by loss functions

$$-\ln f(y^n|x^n; \lambda, \theta) = n \ln Z(\lambda) + \lambda \sum_t \delta(y_t, \hat{y}_t)), \tag{50}$$

where the normalizing coefficient $Z(\lambda)$ does not depend on θ. Such loss functions are called *simple*, [9].

We see that if we minimize the ideal code length with respect to λ we get

$$\begin{aligned} nd\ln Z/d\lambda &+ \sum_t \delta(y_t, \hat{y}_t) = 0 & (51)\\ d\ln Z/d\lambda &= -\int f(y|x; \lambda, \theta)\delta(y, \hat{y})dy = -E_{\lambda,\theta}\delta(y, \hat{y}), & (52) \end{aligned}$$

the latter holding for all λ and θ. As above the minimizing value $\hat{\lambda}(y^n, x^n) = \hat{\lambda}$ also minimizes the mean $-E_{\lambda,\theta} \ln f(y^n|x^n; \lambda, \theta)$, and for $\lambda = \hat{\lambda}$

$$E_{\lambda,\theta} \sum_t \delta(y_t, \hat{y}_t) = \sum_t \delta(y_t, \hat{y}_t). \tag{53}$$

Hence, having minimized the sum of the prediction errors with respect both to λ and θ we know this to be the mean performance, the mean taken with the distribution defined by the minimizing parameter value $\hat{\lambda}$ and any θ. As an example consider the quadratic error function. The normalizing integral is given by

$$\int e^{-\lambda(y-F(x,\theta))^2} dy = \sqrt{\pi/\lambda},$$

which does not depend on θ. We see that we get a normal distribution with mean $F(x,\theta)$ and variance $\sigma^2 = 1/(2\lambda)$. If we extend this to sequences by independence and minimize the negative logarithm with respect to θ and the variance, we see that the minimizing variance is given by the minimized quadratic loss.

The quadratic error function is but a special case of loss functions of type $\delta(y-\hat{y}) = |y-\hat{y}|^\alpha$ for $\alpha > 0$, which we call the $\alpha-$class. The normalizing coefficient is given by

$$\int e^{-\lambda|y-F(x,\theta|^\alpha} dy = Z_\lambda = \frac{2}{\alpha}\lambda^{-1/\alpha}\Gamma(1/\alpha), \tag{54}$$

where $\Gamma(x)$ is the gamma-function. Such loss functions are then seen to be simple.

We show next that the distributions

$$p(y|x;\lambda,\theta) = Z^{-1}(\lambda)e^{-\lambda\delta(y,\hat{y})},$$

where $\hat{y} = h(x,\theta)$, are maximum entropy distributions. Consider the problem

$$\max_g E_g \ln 1/g(Y), \tag{55}$$

where the maximization is over all g such that $E_g\delta(Y,h(\mathbf{x};\theta)) \leq E_p\delta(Y,h(\mathbf{x};\theta))$. We have first by this restriction on the density functions g

$$E_g \ln 1/p(Y|\mathbf{x};\theta,\lambda) = \lambda E_g\delta(Y,h(\mathbf{x};\theta)) + \ln Z_\lambda \leq H(p),$$

where $H(p)$ denotes the entropy of $p(y|\mathbf{x};\theta,\lambda)$. Then, by Shannon's inequality the entropy $H(g)$ of g satisfies $H(g) \leq E_g \ln 1/p(Y|\mathbf{x};\theta,\lambda)$, the right hand side being upperbounded by $H(p)$. The equality is reached with $g = p$. This result generalizes the familiar fact that the normal distribution with variance σ^2 has the maximum entropy among all distributions whose variance does not exceed σ^2.

We consider next models defined by probability or density functions on single data sequences, which we now denote by x^n rather than y^n. We actually need to have a double parameter γ,θ, where $\gamma \in \Gamma$ is a structure index and $\theta = \theta_1,\ldots,\theta_{k(\gamma)}$ a vector of real-valued parameters ranging over a subset Ω_γ of the k-dimensional Euclidean space. We consider then parametric distributions defined either by discrete probability functions $\mathcal{M}_\gamma = \{P(x^n;\theta,\gamma)\}$ or density functions $\mathcal{M}_\gamma = \{f(x^n;\theta,\gamma)\}$ when the data items x_t range over the real numbers. We get a bigger class of models by taking the union $\mathcal{M} = \bigcup_\gamma \mathcal{M}_\gamma$.

Example 1. The class of tree machines with a complete tree T defining the set of contexts is determined by γ as the set of leaves, and $\theta = P(0|\gamma_1),\ldots,P(0|\gamma_m)$. As described above these parameters define a probability assignment to any string $P(x^n;T)$, which, moreover, satisfies the axioms for a random process.

Example 2. Take the class of ARMA processes

$$x_n + a_1x_{n-1} + \ldots + a_px_{n-p} = b_0e_n + \ldots + b_{q-1}e_{n-q+1}, \tag{56}$$

where the pair (p, q) plays the role of γ and the collection of the coefficients forms θ. These equations define the required probability distribution if we model the process e_n as an iid process, each random variable having the normal distribution of zero mean and unit variance.

6.2 Universal Models

The fundamental idea with a universal model for a given model class $\mathcal{M}_\gamma = \{P(x^n; \theta, \gamma) : \theta \in \Omega \subset R^k\}$ is that it should assign as large a probability or density to the data strings as we can get with help of the given model class. Equivalently, the ideal code length for the data strings should be minimized. Hence, if we are given a particular data string x^n the universal model must not depend on this string, for we could just as well have another data string of the same length and the same model class. One way to obtain universal models is to construct a universal coding system by some intuitively appealing coding method, which, in fact, was the way universal coding systems were constructed - and still is to some extend. However, it will be particularly informative and useful to construct optimization problems whose solutions will define universal models.

6.2.1 Minmax Problems

A prototype of optimization problems for codes is Shannon's theorem, expressed as

$$\min_Q \sum_{x^n} P(x^n) \log \frac{P(x^n)}{Q(x^n)} = D(P\|Q),$$

which is solved by $Q = P$. Since we are interested in codes for a class of models instead of just one we may look for distributions $Q(x^n)$ whose ideal code length is the shortest in the mean for the worst case 'opponent' model that generates the data. The first such minmax problem is obtained as follows. Define the 'redundancy', [7], as the excess of the mean code length, obtained with a distribution Q, over the entropy of $P_\theta(x^n) = P(x^n; \theta, \gamma)$:

$$R_n(Q, \theta) = \sum_{x^n} P(x^n; \theta, \gamma) \log \frac{P(x^n; \theta, \gamma)}{Q(x^n)} = D(P_\theta\|Q). \tag{57}$$

We may then ask for a distribution Q as the solution to the minimax problem

$$R_n^+ = \min_Q \max_\theta R_n(Q, \theta). \tag{58}$$

This can be tackled more easily by embedding it within a wider problem by considering the mean redundancy

$$R_n(Q, w) = \int w(\theta) R_n(Q, \theta) d\theta,$$

where $w(\theta)$ is a density function for the parameters in a space Ω_γ. There is really no conflict in notations, because the limit of the distributions $w_i(\eta)$ which place more and more of their probability mass in a shrinking neighborhood of θ, call it $w_\theta(\eta)$, will make $R_n(Q, w_\theta) = R_n(Q, \theta)$. For each w the minimizing distribution of $R_n(Q, w)$ is by Shannon's theorem the *mixture* distribution

$$P_w(x^n) = \int P(x^n; \theta, \gamma) w(\theta) d\theta, \tag{59}$$

which gives the minimized value

$$R_n(w) = \min_Q R_n(Q, w) = \sum_{x^n} \int w(\theta) P(x^n; \theta, \gamma) \log \frac{P(x^n; \theta, \gamma)}{P_w(x^n)} d\theta = I_w(\Theta; X^n). \tag{60}$$

This is seen to be the mutual information between the random variables Θ and X^n, defined by the Kullback-Leibler distance between the joint distribution $w(\theta)P(x^n; \theta, \gamma)$ and the product of the marginals $w(\theta)$ and $P_w(x^n)$.

Further, we may ask for the worst case w as follows

$$\sup_w R_n(w) = \sup_w I_w(\Theta; X^n) = K_n(\gamma), \tag{61}$$

which is seen to be the capacity of the channel $\Theta \to X^n$. By (11)-(12) in Subsection 1.3 the distance $D(P_\theta \| P_{\bar{w}})$ for the maximizing prior is the same for all θ, which means that $r_n^+ = K_n(\gamma)$ and the model family lies on the surface of a hyperball with the special mixture as the center.

Recently the minmax problem (58) has been generalized to the minmax *relative* redundancy problem defined as follows: First, define θ_g as the unique parameter value in Ω^k such that

$$\min_\theta E_g \log \frac{g(X^n)}{P(X^n; \theta, \gamma)} = E_g \log \frac{g(X^n)}{P(X^n; \theta_g, \gamma)}. \tag{62}$$

In words, θ_g picks the model in the class that is nearest to the data generating model $g(x^n)$ in the Kullback-Leibler distance, which need not lie within the model class $\mathcal{M}_\gamma$. Then consider

$$\min_q \max_g E_g \log \frac{P(x^n; \theta_g, \gamma)}{q(X^n)}. \tag{63}$$

This time the solution is not easy to find, but asymptotically it is a modification of the mixture solving (58), [31].

There is a third minmax problem based on the very smallest ideal code length

$$\min_\theta - \log P(x^n; \theta, \gamma) \tag{64}$$

for the data obtainable with the model class given. The minimizing parameter $\hat{\theta} = \hat{\theta}(x^n)$ is the *ML* (*Maximum Likelihood*) estimator. Clearly, the model $P(y^n; \hat{\theta}(x^n), \gamma)$ does not qualify as a universal model, because it depends on the data x^n. In coding theoretic terms x^n could not be decoded, because its codeword depends on $\hat{\theta}(x^n)$. Curiously enough, in the previous case the best model is $P(x^n; \theta_g, \gamma)$ but it cannot be computed since g is not known, while here the best code length is obtained with $P(x^n; \hat{\theta}(x^n), \gamma)$, which can be computed but it is not a model since it does not integrate to unity. This suggests looking for the nearest model as the solution to the minmax problem

$$\min_q \max_g E_g \log \frac{P(X^n; \hat{\theta}(X^n), \gamma)}{q(X^n)}, \tag{65}$$

where g and q may be virtually any distributions.

Theorem 14 *If q and g run through any sets that include the* NML *(*Normalized Maximum Likelihood*) model*

$$\hat{P}(x^n;\gamma) = \frac{P(x^n;\hat{\theta}(x^n),\gamma)}{C_n(\gamma)} \tag{66}$$

$$C_n(\gamma) = \sum_{y^n} P(y^n;\hat{\theta}(y^n),\gamma), \tag{67}$$

both the minimizing q and the maximizing g are given by $\hat{P}(x^n;\gamma)$, and the minmax value is $\log C_n(\gamma)$.

Proof: We have

$$\min_q \max_g E_g \log \frac{P(X^n;\hat{\theta}(X^n),\gamma)}{q(X^n)} \geq \max_g \min_q \{D(g\|q) - D(g\|\hat{P}) + \log C_n(\gamma)\} \tag{68}$$

$$= \log C_n(\gamma), \tag{69}$$

which is true, because only the first term depends on q and is minimized by the choice $q = g$ while the second term is maximized by $q = \hat{P}$. With these choices the equality is reached.

It is important that the optimal q is definable in terms of the model class $\mathcal{M}_\gamma$ selected and hence it can be computed.

The two minmax problems (58) and (65) are clearly related and their solutions should be close. This can be illustrated by the following theorem, which in essence is due to P. Grünwald and H. White, [9], [35],

Theorem 15 *Let the data be generated by an iid process $g(\cdot)$. Then with probability unity under g*

$$\hat{\theta}(x^n) \to \theta_g. \tag{70}$$

Further, if $q(x^n) = q_n(x^n)$ is a family of density functions, then for every n,

$$\frac{1}{m}\sum_{t=0}^{m-1} \log \frac{f(x_{tn+1}^{(t+1)n};\hat{\theta}(nm))}{q(x_{tn+1}^{(t+1)n})} \to E_g \log \frac{f(X^n;\theta_g)}{q(X^n)} \tag{71}$$

in $g-$probability 1, as $m \to \infty$, where $\hat{\theta}(nm)$ denotes the ML estimate evaluated from x_1^{nm}.

Sketch of Proof: We have first

$$E_g \frac{\partial}{\partial\theta} \log f(X;\theta_g) = 0, \tag{72}$$

where the argument θ_g indicates the point where the derivative is evaluated. Similarly,

$$\frac{1}{n}\sum_{t=1}^{n} \frac{\partial}{\partial\theta} \log f(x_t;\hat{\theta}(n)) = 0.$$

For each fixed value $\hat{\theta}(n) = \theta$ this sum converges by the strong law of large numbers to the mean $E_g \log f(X;\theta)$, in $g-$probability 1, and the sum is always zero, which in light of (72) means that the limiting mean is zero. This proves (70). We can again apply the strong law of large numbers to (71), and in view of (70) conclude the second claim.

Yet another minmax problem involving the nonreachable ideal code length was defined by Shtarkov, [29], as follows

$$\min_q \max_{x^n} \log \frac{P(x^n; \hat{\theta}(x^n), \gamma)}{q(x^n)}. \tag{73}$$

The minimizing worst case excess is obtained again by the *NML* distribution. This can be seen by considering the equivalent minmax problem

$$\min_q \max_{x^n} \log \frac{\hat{P}(x^n; \gamma)}{q(x^n)} + \log C_n(\gamma).$$

The ratio of two different distributions will exceed unity at some point, and hence the maximum ratio is minimized to unity only when the distributions are the same. The claim follows, and the minmax value of the original problem is $\log C_n(\gamma)$.

We can write the *NML* model in another form. Put first

$$P_n(\theta) = \sum_{y^n : \hat{\theta}(y^n) = \theta} P(y^n; \theta, \gamma), \tag{74}$$

and then

$$\pi_n(\theta) = \frac{P_n(\theta)}{\sum_{\hat{\theta}} P_n(\hat{\theta})}, \tag{75}$$

where θ and $\hat{\theta}$ range over the discrete ML-parameters. Because of its construction we call $\pi_n(\theta)$ the *canonical* prior. Finally,

$$\hat{P}(x^n; \gamma) = \frac{P(x^n; \hat{\theta}(x^n), \gamma)}{P_n(\hat{\theta}(x^n)} \pi_n(\hat{\theta}(x^n)). \tag{76}$$

Because the *NML* model depends on the ML-estimates $\hat{\theta}(x^n)$, which, in turn, depend on n, it does not define a random process. However, since the ML-estimates in many cases converge so that for large t, $\hat{\theta}(x^{t+1}) \cong \hat{\theta}(x^t)$ we obtain

$$P(x_{t+1} | x^t, \hat{\theta}(x^t), \gamma) \cong \frac{\hat{P}(x^{t+1}; \gamma)}{\sum_u \hat{P}(x^t, u; \gamma))},$$

where the left hand side is seen to be a *predictive* process defining a predictive universal model. For small values of t the ML-estimates may define singular probabilities, and to avoid that we must modify the estimates appropriately, as was done in the Context Algorithm for universal coding. Clearly, there will be an initialization problem, because to estimate k parameters we need at least k data points. However, the predictive process is an important way to obtain universal models in particular for cases where the data are ordered and where the data set is not too small.

How about the *NML* distribution for the class $\mathcal{M} = \bigcup_\gamma \mathcal{M}_\gamma$? It turns out that although we cannot meaningfully maximize $P(x^n; \hat{\theta}(x^n), \gamma)$ over γ in its range Γ we can maximize $\hat{P}(x^n; \gamma)$, which leads to the *NML* distribution

$$\hat{P}(x^n) = \hat{P}(x^n; \Gamma) = \frac{\hat{P}(x^n; \hat{\gamma}(x^n))}{\sum_{y^n : \hat{\gamma}(y^n) \in \Gamma} \hat{P}(y^n; \hat{\gamma}(y^n))}. \tag{77}$$

We can write the denominator in the following form

$$\sum_{y^n} \hat{P}(y^n; \hat{\gamma}(y^n)) = \sum_{\gamma} Q(\gamma) \tag{78}$$

$$Q(\gamma) = \sum_{\hat{\gamma}(y^n)=\gamma} \hat{P}(y^n; \hat{\gamma}(y^n)) \tag{79}$$

and

$$\hat{P}(x^n) = \frac{\hat{P}(x^n; \hat{\gamma}(x^n))}{Q_n(\hat{\gamma}(x^n))} q_n(\hat{\gamma}(x^n)), \tag{80}$$

where

$$q_n(\gamma) = \frac{Q_n(\gamma)}{\sum_{\gamma' \in \Gamma} Q_n(\gamma')} \tag{81}$$

is seen to sum up to unity.

Is there anything we can say about the distribution $q_n(\gamma)$? In fact, we have the theorem

Theorem 16 *Let for each $\gamma \in \Gamma_f$ $Prob(\hat{\gamma}(y^n) \neq \gamma) \to 0$, where Γ_f is a finite subset of Γ and $\theta \in \Omega^k(\gamma)$. Then*

$$Q_n(\gamma) \to 1 \tag{82}$$

$$q_n(\gamma) \to 1/|\Gamma_f|. \tag{83}$$

Proof: We have

$$1 = \sum_{y^n} \hat{P}(y^n; \gamma) = Q_n(\gamma) + \sum_{\hat{\gamma}(y^n) \neq \gamma} \hat{P}(y^n; \hat{\gamma}(y^n)), \tag{84}$$

where the second sum converges to zero by assumption. The claim follows.

6.2.2 Strong Optimality

The solutions to the various minmax problems considered, while certainly optimal for the worst case opponent, leave open the nagging question whether their ideal code length would be shorter for all the other opponents or perhaps for most of them. We give a theorem, [22], which may be viewed as an extension of Shannon's noiseless coding theorem, and it shows that the worst case performance is the rule rather than an exception. For this we consider a class of parametric probability density functions $\mathcal{M}_\gamma = \{f(x^n; \theta, \gamma)\}$, where the parameter vector $\theta = \theta_1, \ldots, \theta_k$ ranges over an open bounded subset $\Omega = \Omega^k$ of the $k-$dimensional euclidian space. The parameters are taken to be 'free' in the sense that any two distinct values for θ specify distinct probability measures.

Theorem 17 *Assume that there exist estimates $\hat{\theta}(x^n)$ (such as the ML estimates) which satisfy the central limit theorem at each interior point of Ω^k, where $k = k(\gamma)$, such that $\sqrt{n}(\hat{\theta}(x^n) - \theta)$ converges in probability to a normally distributed random variable. If $q(x^n)$ is any density function defined on the observations, then for all positive numbers ϵ and all $\theta \in \Omega^k$, except in a set whose volume goes to zero as $n \to \infty$,*

$$E_\theta \log \frac{f(x^n; \theta, \gamma)}{q(x^n)} \geq \frac{k - \epsilon}{2} \log n.$$

The mean is taken relative to $f(x^n; \theta, \gamma)$.

We give the original proof, which can be generalized to the proof of a stronger version of the theorem, [23]. There exists an elegant proof of an extension of the theorem in [16].

Proof:

Consider a partition of the set Ω^k into $k-$ dimensional hypercubes of edge length $\Delta_n = c/\sqrt{n}$, where c is a constant. Let the, say m_n, centers of these hypercubes form the set $\Omega(\Delta_n) = \{\theta^1, \theta^2, \ldots\}$, and write $C_n(\theta)$ for the cube with center at θ, $\theta \in \Omega(\Delta_n)$. We need to construct a corresponding partition of a subset of X_n, the set of all sequences of length n. For this we use the estimator $\hat{\theta}(x^n)$, and we define $X_n(\theta) = \{x^n | \hat{\theta}(x^n) \in C_n(\theta)\}$. Let the probability of this set under the distribution $f(x^n; \theta, \gamma)$ be $P_n(\theta)$. Because of the assumed consistency of the estimator the probability of the set $X_n(\theta)$, namely $P_n(\theta)$, satisfies the inequality

$$P_n(\theta) \geq 1 - \delta(c) \tag{85}$$

for all n greater than some number, say n_c, that depends on c. Moreover, $\delta(c)$ can be made as small as we please by selecting c large enough.

Consider next a density function $q(x^n)$, and let $Q_n(\theta)$ denote the probability mass it assigns to the set $X_n(\theta)$. The ratio $f(x^n; \theta, \gamma)/P_n(\theta)$ defines a distribution on $X_n(\theta)$, as does of course $q(x^n)/Q_n(\theta)$. By Shannon's Theorem, applied to these two distributions, we get

$$\int_{X_n(\theta)} f(x^n; \theta, \gamma) \log \frac{f(x^n; \theta, \gamma)}{q(x^n)} dx^n \geq P_n(\theta) \log \frac{P_n(\theta)}{Q_n(\theta)}. \tag{86}$$

For a positive number ϵ, $\epsilon < 1$, let $A_\epsilon(n)$ be the set of θ such that the left hand side of (86), denoted $T_n(\theta)$, satisfies the inequality

$$T_n(\theta) < (1 - \epsilon) \log n^{k/2}. \tag{87}$$

¿From (86) and (87) we get

$$-\log Q_n(\theta) < [(1 - \epsilon)/P_n(\theta) - \frac{\log P_n(\theta)}{\log n^{k/2}}] \log n^{k/2}, \tag{88}$$

which holds for $\theta \in A_\epsilon(n)$. Replace $P_n(\theta)$ by its lower bound $1 - \delta(c)$ in (85), which does not reduce the right hand side. Pick c so large that $\delta(c) \leq \epsilon/2$ for all n greater than some number n_ϵ. The first term within the bracket is then strictly less than unity, and since the second term is bounded from above by $(-\log(1 - \epsilon/2))/\log n^{k/2}$, and hence converges to zero with growing n, the expression within the brackets is less than some α such that $0 < \alpha < 1$, for all sufficiently large n, say, larger than some n'_ϵ. Therefore,

$$Q_n(\theta) > n^{-\alpha k/2} \tag{89}$$

for $\theta \in A_\epsilon(n)$ and n larger than n'_ϵ. Next, let $B_\epsilon(n)$ be the smallest set of the centers of the hypercubes which cover $A_\epsilon(n)$, and let ν_n be the number of the elements in $B_\epsilon(n)$. Then the Lebesgue volume

V_n of $A_\epsilon(n)$ is bounded by the total volume of the ν_n hypercubes, or

$$V_n \leq \nu_n c^k / n^{k/2}. \tag{90}$$

¿From (89) and the fact that the sets $X_n(\theta)$ are disjoint we get further

$$1 \geq \sum_{\theta \in B_\epsilon(n)} Q_n(\theta) \geq \nu_n n^{-\alpha k/2}, \tag{91}$$

which gives an upper bound for ν_n. ¿From (90) we then get the desired inequality

$$V_n \leq c^k n^{(\alpha-1)k/2}, \tag{92}$$

which shows that $V_n \to 0$ as n grows to infinity.

Using the inequality $\log z \geq 1 - 1/z$ for $z = f(x^n;\theta,\gamma)/q(x)$ we get

$$\int_{\bar{X}_n(\theta)} f(x^n;\theta,\gamma) \log[f(x^n;\theta,\gamma)/q(x^n)]dx^n \geq Q_n(\theta) - P_n(\theta) > -1, \tag{93}$$

where $\bar{X}$ denotes the complement of X. To finish the proof let $\theta \in \Omega^k - A_\epsilon(n)$. Then the opposite inequality, $\geq$, in (87) holds. By adding the left hand sides of (86) and (93) we get

$$E_\theta \log[f(x^n;\theta,\gamma)/q(x^n)] > (1-\epsilon)\log n^{k/2} - 1,$$

which concludes the proof.

We add that with further and not too sever smoothness conditions on the models one can also prove (Dawid) a similar theorem in the almost sure sense: Let $Q(x^n)$ define a random process. Then for all $\theta \in \Omega^k$, except in a set of measure 0,

$$-\log Q(x^n) \geq -\log P(x^n;\theta,\gamma) + \frac{k-\epsilon}{2}\log n \tag{94}$$

for all but finitely many n in $P(x^\infty;\theta)$ - probability 1. Here, $P(x^\infty;\theta)$ denotes the unique measure on infinite sequences defined by $P(x^n;\theta,\gamma)$.

These results show that the right hand side bound cannot be beaten by any code, except one in the necessary loophole of Lebesgue measure zero, while reachability results immediately from (96). Clearly, such an exception must be provided, because the data generating model itself is the best, but it, of course, could not be found except with a wild guess. There exists also a stronger version of the theorem about the minmax bound $\log C_n(k)$, which in effect states that this worst case bound is not a rare event but that it cannot be beaten in essence even when we assume that the data were generated by the most 'benevolent' opponents, [27].

6.2.3 Universal Sufficient Statistics

We begin by sketching the derivation of a quite accurate formula for the ideal code length of the fundamental *NML* model, which we write for density models. If we overlook some details, [25], the

derivation is quite simple. The main condition required is that the ML-estimates satisfy the Central Limit Theorem in that $\sqrt{n}(\hat{\theta}(x^n) - \theta)$ converges in distribution to the normal distribution of mean zero and covariance $J^{-1}(\theta)$, where

$$J(\theta) = \lim_{n\to\infty} -n^{-1}\{E_\theta \frac{\partial^2 \log f(X^n;\theta,\gamma)}{\partial\theta_i \partial\theta_j}\} \tag{95}$$

is a generalization of the so-called Fisher information matrix, defined ordinarily for iid processes. For the required integral to be finite the space $\Omega = \Omega_\gamma$ will be chosen as an open bounded set, which gives the two-part ideal normalized code length in the form

$$\begin{aligned} -\log \hat{f}(x^n;\gamma) &= -\log f(x^n;\hat{\theta}(x^n),\gamma) + \log C_n(\gamma) && (96)\\ &= -\log \frac{f(x^n;\hat{\theta}(x^n),\gamma)}{(2\pi/n)^{-k/2}|J(\hat{\theta}(x^n))|^{1/2}} - \log \pi(\hat{\theta}(x^n)) + o(1) && (97)\\ &= -\log f(x^n;\hat{\theta}(x^n),\gamma) + \frac{k}{2}\log\frac{n}{2\pi} + \log\int_\Omega |J(\theta)|^{1/2} d\theta + o(1), && (98) \end{aligned}$$

where the canonical prior $\pi(\hat{\theta}(x^n))$ is given by

$$\pi(\theta) = \frac{|J(\theta)|^{1/2}}{\int_\Omega |J(\theta')|^{1/2} d\theta'}. \tag{99}$$

The argument in $J(\hat{\theta}(x^n))$ is regarded as the fixed parameter $\hat{\theta} = \hat{\theta}(x^n)$. The term $o(1)$, which goes to zero as $n \to \infty$, takes care of the rate of convergence by the Central Limit Theorem and other details. For iid processes one can show that

$$-E_\theta \log \hat{f}(x^n;\gamma) = -E_\theta \log f_{\bar{w}}(x^n;\gamma) + o(1), \tag{100}$$

where $f_{\bar{w}}(x^n;\gamma$ is the mixture that achieves the channel capacity.

In light of the minmax Theorem 14 and the strong optimality theorem in the previous subsection we are justified to regard $-\log \hat{f}(x^n;\gamma)$ as the *stochastic complexity* of x^n, given the model class $\mathcal{M}_\gamma$. We are interested in extending Kolmogorov's sufficient statistics decomposition (34) to the model classes $\mathcal{M}_\gamma$ and $\mathcal{M} = \cup_\gamma \mathcal{M}_\gamma$. Although the decompositions in Equations (96)-(98) are suggestive it is not clear that they achieve the intent. It is true that the first term in the right hand side of the second version (97) represents a conditional code length for the data, given the optimal model specified by $\hat{\theta}(x^n)$, but the canonical prior is a density function on the parameters, and its negative logarithm does not specify the optimal model, which should be a finitely describable object. Neither is it clear how $\log C_n(\gamma)$ in the first version could be viewed as the code length for the optimal model nor how the first term, the negative logarithm of $f(x^n;\hat{\theta}(x^n),\gamma)$ could be viewed as the conditional code length for the data, given the optimal model, because it does not integrate to unity. That we have in these equations density functions on the data does not matter, because the level of quantization of the data unlike that of the parameters is irrelevant.

It was shown by Balasubramanian, [2], that $\log C_n(\gamma)$ for a fixed γ with k parameters, gives the maximal number of *distinguishable* models from data of size n. His idea of distinguishability is

based on differential geometry and somewhat intricate. Moreover it is defined in terms of covers of the parameter space rather than partitions, which makes it difficult to visualize how the universal sufficient statistics should be defined such that the intent in Kolmogorov's sufficient statistics is achieved. For these reasons we study the issue in a different manner, which does not require the explicitly defined notion of distinguishability, although one that is similar to Balasubramanian's can be defined.

Consider a hyper ellipsoid $D_{i,n}$ in the bounded parameter space, centered at θ_i, defined by

$$(\theta - \theta_i)' J(\theta_i)(\theta - \theta_i) = d/n, \tag{101}$$

where d is a parameter. Let $B_{i,n} = B_{i,n}(d)$ be the maximal rectangle within $D_{i,n}$, and let $\Pi_n = \{B_{i,n}\}$ denote a partitioning of Ω with these rectangles. (Actually, these rectangles will have to be curvilinear, but when we let n grow the difference will disappear.) The reason for using the norm defined by (101) rather than the Euclidian norm is that the Kullback-Leibler distance between the models $f(y^n;\theta_i,\gamma)$ and $f(y^n;\theta_j,\gamma)$, where θ_i and θ_j are the centers of two adjacent cells in Π_n, is the same for any pair of adjacent cells. In other words, to the partition Π_n there corresponds an symptotically equi-distance partition on the class of models $\mathcal{M}_k$. This can be seen by applying Taylor's expansion, evaluated at θ_i, to the two adjacent models, written as f_i and f_j,

$$D(f_i\|f_j) = \frac{n}{2}(\theta_j - \theta_i)' J(\tilde{\theta})(\theta_j - \theta_i),$$

where $\tilde{\theta}$ is a point between θ_i and θ_j. When the n grows the difference $\tilde{\theta} - \theta_i$ shrinks to zero, and we see that the Kulback-Leibler distance d/n becomes the same for all pairs of adjacent models.

Next, pick $d = \hat{d}$ such that

$$\int_{\hat{\theta}(y^n)\in B_{i,n}} f(y^n;\hat{\theta}(y^n),\gamma)dy^n = \int_{\hat{\theta}\in B_{i,n}} g(\hat{\theta};\hat{\theta})d\hat{\theta} = 1, \tag{102}$$

where $g(\hat{\theta};\theta)$ is the density function on the statistic $\hat{\theta}(y^n) = \hat{\theta}$ induced by $f(y^n;\theta,\gamma)$. Then the number of elements in the partitioning is given by

$$|\Pi_n| = \sum_{i=1}^{|\Pi_n|} 1 = C_n(\gamma). \tag{103}$$

Define next for $B_{i,n}(\hat{d})$ that includes $\hat{\theta}(x^n)$ having θ_i as the center

$$f(y^n|\theta_i,\gamma) = \begin{cases} f(y^n;\hat{\theta}(y^n),\gamma) & \text{if } y^n \in B_{i,n} \\ 0 & \text{otherwise.} \end{cases} \tag{104}$$

We then get a decomposition

$$-\log \hat{f}(x^n;\gamma) = -\log f(x^n|\theta_i,\gamma) + \log C_n(\gamma). \tag{105}$$

One reason why this decomposition fails to be a satisfactory analog of Kolmogorov's sufficient statistics decomposition is the lack of justification of that $\log C_n(\gamma)$ represents the code length of the model that captures the minimal amount of properties for which the stochastic complexity is obtained; i.e.

the code length for the model should be minimal. The trouble stems from the fact that regardless of the value of the parameter d determining the volume of the cells $B_{i,n}(d)$, we reach the stochastic complexity, and hence the maximal cell size Ω itself would minimize the code length for the model leaving all the data as 'noise'! In fact, suppose we pick a value for d such that the volume of the cell $B_{i,n}$ becomes α times what it is for $d = \hat{d}$. Then the number of models would be $C_n(\gamma)/\alpha$, and the conditional density function defining the 'noise' would be

$$f(y^n|\theta_i,\gamma) = \begin{cases} f(y^n;\hat{\theta}(y^n),\gamma)/\alpha) & \text{if } y^n \in B_{i,n} \\ 0 & \text{otherwise.} \end{cases} \tag{106}$$

However, we still get the same decomposition (105), defined by the unique ratio

$$\hat{f}(x^n;\gamma) = \frac{f(x^n;\hat{\theta}(x^n),\gamma)}{C_n(\gamma)}, \tag{107}$$

where the numerator and the denominator have no common factors. The effect of the arbitrary α is simply to remove some of the code length for the noise into the code length for the optimal model, while $\log C_n(\gamma)$ alone gives the number of 'distinguishable' models.

Another and perhaps the more troublesome aspect of the decomposition (105) is that it is based on models of type $f(x^n;\hat{\theta}(x^n),\gamma)$ with support $B_{i,n}(\hat{d})$, which is not quite what we would like to have. Rather, the models we would like to deal with are of type $f(y^n;\theta_i,\gamma)$, where θ_i is the maximum likelihood estimate quantized to the center of $B_{i,n}(d)$ for some d, and y^n ranges over all sequences of length n. Our objective is to obtain a decomposition for such models. We consider first model classes $\mathcal{M}_\gamma$ of a fixed structure γ.

How should we replace the set A, formalizing the properties of the string x^n in the algorithmic theory? There the intent is that all the strings in A are equal sharing the properties specified by A, and hence being 'typical' of this set. This suggests that we should replace it by a set of 'typical' strings of the model $f(y^n;\theta_i,\gamma)$, namely, a set $B_{i,n}(d)$ for some d. Then the complexity of this set itself will be the code length needed to describe θ_i. As will be seen there is no code length needed for d.

Just as $\log|A|$ is the code length of the worst case sequence in A, we need the code length of the worst case sequence y^n in $B_{i,n}(d)$, which is obtained by the Taylor series expansion as follows

$$-\log f(y^n;\theta_i,\gamma) = -\log f(y^n;\hat{\theta}(y^n),\gamma) + \frac{1}{2}d, \tag{108}$$

where y^n denotes a sequence for which

$$n(\hat{\theta}(y^n) - \theta_i)'\hat{J}(\tilde{\theta}_i)(\hat{\theta}(y^n) - \theta_i) = d.$$

Here $\hat{J}(\hat{\theta})$ is the empirical Fisher information matrix

$$\hat{J}(\hat{\theta}) = -n^{-1}\{\frac{\partial^2 \log f(y^n;\hat{\theta},\gamma)}{\partial\hat{\theta}_j\partial\hat{\theta}_k}\}, \tag{109}$$

and $\tilde{\theta}_i$ is a point between θ_i and $\hat{\theta}(y^n)$. We also assume that for all data sequences such that $\hat{\theta}(y^n)$ falls within $B_{i,n}$ the empirical $\hat{J}(\hat{\theta}(y^n))$ converges to $\hat{J}(\hat{\theta})$ as $\hat{\theta}(y^n) \to \hat{\theta}$.

Consider the probability distribution for the quantized parameters, obtained from the canonical prior, (99),

$$\pi_d(\theta_i) = \frac{\int_{\hat{\theta}\in B_{i,n}} h(\hat{\theta};\hat{\theta})d\hat{\theta}}{C_n(\gamma)}.$$

We get more convenient asymptotically valid formulas provided that the Central Limit Theorem holds. The Fisher information is taken to be continuous, and

$$\int_{B_{i,n}} h(\hat{\theta};\hat{\theta})d\hat{\theta} \to (n/(2\pi))^{k/2}|J(\theta_i)|^{1/2}|B_{i,n}|,$$

where $|B_{i,n}|$ denotes the volume of $B_{i,n}$, given by

$$|B_{i,n}| = (\frac{4d}{nk})^{k/2}|J(\theta_i)|^{-1/2}. \tag{110}$$

With these approximations the numerator of π_d converges to

$$(\frac{n}{2\pi})^{k/2}|J(\theta_i)|^{1/2}(\frac{4d}{nk})^{k/2}|J(\theta_i)|^{-1/2} = (\frac{2d}{\pi k})^{k/2}.$$

By replacing $-\log f(y^n;\hat{\theta}(y^n),\gamma)$ by $-\log f(x^n;\hat{\theta}(x^n),\gamma)$ with an insignificant error, we then set the analog of Kolmogorov's structure function $h_{x^n}(\alpha)$ as follows

$$h_{x^n}(\alpha) = \min_d\{-\log f(x^n;\hat{\theta}(x^n),\gamma) + \frac{1}{2}d : -\log \pi_d(\theta_i) \le \alpha\}. \tag{111}$$

For the minimizing d the inequality will have to be satisfied with equality,

$$\alpha = \frac{k}{2}\log\frac{\pi k}{2d} + \log C_n(\gamma),$$

and with the asymptotic approximations we get

$$d_\alpha = \frac{\pi k}{2}C_n(\gamma)^{2/k}e^{-2\alpha/k},$$

and from (108)

$$h_{x^n}(\alpha) = -\log f(y^n;\theta_i,\gamma) = -\log f(x^n;\hat{\theta}(x^n),\gamma) + d_\alpha/2. \tag{112}$$

With this we get further

$$h_{x^n}(\alpha) + \alpha = -\log \hat{f}(x^n;\gamma) + d_\alpha/2 - \frac{k}{2}\log d_\alpha + \frac{k}{2}\log\frac{\pi k}{2}. \tag{113}$$

Recall Equation (36), rewritten here

$$\min\{\alpha : h_{x^n}(\alpha) + \alpha \doteq K(x^n)\}, \tag{114}$$

which gives the crucial and optimal way to separate the complexity of the data into the complexity α of the model and the complexity $h_{x^n}(\alpha)$ of the 'noise', as it were. However, we do not want to ignore constants, which means that instead of the approximate equality $\doteq$ we want to write an equality. This poses a bit of a problem, because for no value of α could we reach the stochastic complexity $-\log \hat{f}(x^n;\gamma)$. We resolve this problem by asking for the smallest value of α for which the 2-part

complexity on the left hand side reaches its minimum value. In our case then we ask for the smallest, and in fact the only, value for α that minimizes the 2-part code length (113),

$$\min_{\alpha}\{h_{x^n}(\alpha)+\alpha\} = -\log \hat{f}(x^n;\gamma) + \frac{k}{2}\log\frac{\pi k e}{2}. \tag{115}$$

The minimum is reached for $d_\alpha = k$ at the point

$$\bar{\alpha} = \frac{k}{2}\log\frac{\pi}{2} + \log C_n(\gamma).$$

For large n this logarithm of the number of optimal models is not much larger than the number obtained by Balasubramanian, namely, $\log C_n(\gamma)$. We interpret a bit later $\bar{\alpha}$ as the number of 'distinguishable' models in a sense similar to Balasubramanian's.

Next, consider the line, analogous to the *sufficiency line* in the algorithmic theory, [32],

$$L(\alpha) = -\log \hat{f}(x^n;\gamma) + \frac{k}{2}\log\frac{\pi k e}{2} - \alpha. \tag{116}$$

By (113) and (115) the curve $h_{x^n}(\alpha)$ lies above the line $L(\alpha)$ for $0 \le \alpha \le \alpha_{max}$, where

$$\alpha_{max} = -\log \hat{f}(y^n;\gamma) + \frac{k}{2}\log\frac{\pi k e}{2},$$

except for the point $\bar{\alpha}$, where the line is its tangent. Indeed,

$$\frac{dh_{x^n}(\alpha)}{d\alpha} = -\frac{d_\alpha}{k}.$$

We see that $h_{x^n}(\alpha) - L(\alpha) = d_\alpha/2$ grows more rapidly for $\alpha < \bar{\alpha}$ than for α exceeding $\bar{\alpha}$. Hence, if we measure the amount of statistical properties in the data by the complexity of the model, α, we see that too complex models leave smaller amounts as 'noise', namely $h_{x^n}(\alpha)$, than too simple models for the same amount of deviation from the optimal $\bar{\alpha}$. In other words, it is better to have overly complex models than too simple ones which do not capture all the statistical properties in the data. In algorithmic theory, all too complex models still reach the complexity $K(x^n)$ up to a constant.

Because we set the target for the 2-part code length as its minimum value rather than the stochastic complexity we can shorter the code length for the data by the process of normalization. In fact, define

$$\bar{h}_{x^n}(\alpha) = -\log \bar{f}(x^n|\hat{\theta}(x^n),\gamma) + d_\alpha/2,$$

where

$$\bar{f}(y^n|\theta_i,\gamma) = \begin{cases} \frac{f(y^n;\theta_i,\gamma)}{P_i} & \text{if } y^n \in B_{i,n}(d) \\ 0 & \text{otherwise,} \end{cases} \tag{117}$$

and

$$P_i = \int_{\hat{\theta}(y^n)\in B_{i,n}} f(y^n;\theta_i,\gamma)dy^n \tag{118}$$

$$= \int_{\hat{\theta}\in B_{i,n}(d)} g(\hat{\theta};\theta_i)d\hat{\theta}. \tag{119}$$

We get then the universal density function $\bar{f}(y^n; \mathcal{M}_\gamma)$ defined by

$$-\log \bar{f}(y^n; \mathcal{M}_\gamma) = -\log \bar{f}(y^n|\theta_i, \gamma) + \log C_n(\gamma) + \frac{k}{2}\log\frac{\pi}{2}. \tag{120}$$

This has the same number of optimal models as the 2-part code but the the code length for noise, as defined by the first term, is reduced. We regard (120) as the *universal sufficient statistics decomposition* of the model class $\mathcal{M}_\gamma$. The probability P_i is clearly upper bounded by the product of the peak $g(\theta_i; \theta_i)$ and the volume of the cell $B_{i,n} = B_{i,n}(k)$, or $(\frac{2}{\pi})^{k/2}$, and lower bounded by the upper bound times $e^{-k/2}$, so that the inequalities

$$-\log \hat{f}(y^n; \mathcal{M}_\gamma) - k/2 < -\log \bar{f}(y^n; \mathcal{M}_\gamma) < -\log \hat{f}(y^n; \mathcal{M}_\gamma) + k/2$$

hold. Since the decomposition (120) differs from the stochastic complexity by less than the constant $k/2$ we can confidently say that virtually all the regular features in the data are captured by the optimal model $f(y^n; \theta_i, \gamma)$, where θ_i is the center of $B_{i,n}(k)$ in the partition Π_n. Notice, however, that we cannot do the normalization first and then find the optimal decomposition, because the optimum would be reached for $d = 0$, which is uninteresting.

We next study the universal sufficient statistics decomposition for the model class $\mathcal{M}$. We need a prior $q(\gamma)$ for the structure index, which by Theorem 16 is taken as the uniform $1/|\Gamma|$, where $|\Gamma|$ is the number of the structures, often just n. The code length for the models is then increased from the previously discussed case by adding $\log |\Gamma|$ to it. The structure function is now

$$h_{x^n}(\alpha) = \min_{d,\gamma}\{-\log f(x^n; \hat{\theta}(x^n), \gamma) + \frac{1}{2}d : -\log \pi_d(\theta_i) + \log |\Gamma| \leq \alpha\}. \tag{121}$$

For each γ the minimizing value for d is

$$d_{\alpha,\gamma} = \frac{\pi k}{2}(|\Gamma| C_n(\gamma))^{2/k} e^{-2\alpha/k},$$

and it is reached when the codelength for the optimal model is α. To get an idea of the behavior of $d_{\alpha,\gamma}$ when $\gamma = k$, we use the asymptotic formula for $C_n(\gamma)$, (98), which gives

$$d_{\alpha,k} = nk|\Gamma|^{2/k} e^{-2\alpha/k}.$$

For a fixed α this is seen to shrink as k gets smaller. The structure function then is given by

$$h_{x^n}(\alpha) = \min_{\gamma}\{-\log f(x^n; \hat{\theta}(x^n), \gamma) + \frac{1}{2}d_{\alpha,\gamma}\} = -\log f(x^n; \hat{\theta}(x^n), \hat{\gamma}) + \frac{1}{2}d_{\alpha,\hat{\gamma}}, \tag{122}$$

where $\bar{\gamma}$ with $\bar{k}$ parameters is the minimizing value dependent on α. There is a well defined minimum, because the first term increases with decreasing k while the second gets smaller.

The minimum of the 2-part code length $h_{x^n}(\alpha) + \alpha$ over α is found by

$$\min_{d,\gamma}[-\log f(x^n; \hat{\theta}(x^n), \gamma) + \frac{1}{2}d - \log \pi_d(\theta_i) + \log |\Gamma|]\}. \tag{123}$$

For each γ the minimizing value for d is $\hat{d} = k$, as before, and we are left with the minimization

$$\min_{\gamma}\{-\log \hat{f}(y^n; \gamma) + \log |\Gamma| + \frac{k}{2}\log\frac{\pi e}{2}\}.$$

Letting $\hat{\gamma}$ denote the minimizing structure and $\hat{k}$ the number of parameters in it, we get

$$h_{x^n}(\hat{\alpha}) = -\log f(y^n;\theta_i,\hat{\gamma}) = -\log f(x^n;\hat{\theta}(x^n),\hat{\gamma}) + \hat{k}/2, \tag{124}$$

where

$$\hat{\alpha} = \frac{\hat{k}}{2}\log\frac{\pi}{2} + \log(C_n(\hat{\gamma})|\Gamma|). \tag{125}$$

This gives

$$h_{x^n}(\hat{\alpha}) + \hat{\alpha} = -\log\hat{f}(x^n;\hat{\gamma}) + \log|\hat{\Gamma}| + \frac{\hat{k}}{2}\log\frac{\pi e}{2}. \tag{126}$$

The volume of the $\hat{k}-$dimensional cells $B_{i,n}(\hat{k})$ is given by

$$(\frac{4}{n})^{\hat{k}/2}|J(\theta_i)|^{-1/2}. \tag{127}$$

This means that the number of the cells corresponding to $d = \hat{k}$ is $(\pi/2)^{\hat{k}/2}C_n(\hat{\gamma})$. As above, $h_{\mathbf{x}^n}(\alpha)$ stays above the sufficiency line $L(\alpha) = -\log\hat{f}(x^n;\hat{\gamma}) + d_{\alpha,\gamma}/2$, except at the point $\hat{\alpha}$, where they are equal.

We convert the optimal 2-part code length (126) into the universal density function $\bar{f}(y^n;\mathcal{M})$ by normalization

$$-\log\bar{f}(y^n;\mathcal{M}) = -\log\bar{f}(y^n|\theta_i) + \log C_n(\hat{\gamma})|\Gamma|) + \frac{k}{2}\log\frac{\pi}{2}, \tag{128}$$

where

$$\bar{f}(y^n|\theta_i,\gamma) = \begin{cases} \frac{f(y^n;\theta_i,\gamma)}{P_i(\hat{k})} & \text{if } y^n \in B_{i,n} \\ 0 & \text{otherwise,} \end{cases} \tag{129}$$

and

$$P_i(\hat{k}) = \int_{\hat{\theta}(y^n)\in B_{i,n}} f(y^n;\theta_i,\hat{\gamma})dy^n \tag{130}$$

$$= \int_{\hat{\theta}\in B_{i,n}} h(\hat{\theta};\theta_i)d\hat{\theta}. \tag{131}$$

In conclusion we discuss the idea of *distinguishable* models for the model class $\mathcal{M}_\gamma$ in the spirit of Balasubramanian. We consider the models defined by the parameters within each cell $B_{i,n}(k)$ as *indistinguishable* from each other, and the corresponding sequences y^n such that $\hat{\theta}(y^n) \in B_{i,n}$, equivalent, in that each define the same optimal model $f(y^n;\theta_i,\gamma)$. We may consider these strings as most 'typical' for the optimal model $f(y^n;\theta_i,\gamma)$, and they have virtually the same code length as given by the first term in the decomposition (120). The two remaining terms give the logarithm of the number of optimally *distinguishable* models, which we define to be the amount of *information* in the data sequence x^n. This definition of distinguishability can be linked to the asymptotic probability of error, which Balasubramanian did by appeal to Stein's lemma. Indeed, if we let the cell sizes shrink at a rate slower than d/n, say $d/o(n)$, and we take an unbounded parameter space, then both of the probabilities $P_i(d/o(n))$ and $P_j(d/o(n))$ of adjacent models $f(y^n;\theta_i,\gamma)$ and $f(y^n;\theta_j,\gamma)$ converge to unity, and the models would be perfectly separated in the limit. If again we let the cell sizes shrink at a rate $o(d/n)$, which is faster than d/n, then the Kullback-Leibler distance between the adjacent

models will go to zero, and hence the error probability defined in any of the usual ways would not go to zero. Hence, the rate d/n is critical, and among all partitions such that their cells shrink at this critical rate, the partition defined by the rate k/n is optimal giving the smallest ideal code length.

The universal sufficient statistics decomposition has the immediate implication that it resolves the puzzling anomaly in the traditional statistics that the maximum likelihood estimate $\hat{\theta}(x^n)$ of the values of the parameters is acceptable, but the same estimate of their number $\hat{k}(x^n)$ is not. Yet, both are just parameters, and one and the same principle of estimation should be applicable to both. The explanation is that the maximum likelihood principle should be rejected in both cases, because we are not then distinguishing between noise and the learnable part, the information. In case of $\hat{\theta}(x^n)$ the damage is minor for large amounts of data, because of the convergence properties of these estimates. However, in case of $\hat{k}(x^n)$ the damage is devastating. In light of the decomposition above we now see that in every case we should separate the noise part from the information and fit parameters only such that we capture the information; in other words, to the precision given by the quantization defined by the cells $B_{i,n}$ of volume (127) with $d = k$. As argued in the subsection 6.4 below their number should be estimated by minimizing the stochastic complexity, which amounts to the same, namely, to capture the learnable information.

Example. We derive the universal sufficient statistics decomposition for the Bernoulli class $\mathcal{B}$, with $P(x = 0) = \theta$ as the parameter. The *NML* distribution is given by

$$\hat{P}(x^n) = \frac{P(x^n; \hat{\theta}(x^n))}{\sum_m \binom{n}{m} (\frac{m}{n})^m (\frac{n-m}{n})^{n-m}}, \tag{132}$$

where $\hat{\theta} = \hat{\theta}(x^n) = n_0(x^n)/n$, $n_0(x^n)$ denoting the number of 0's in x^n, and $P(x^n; \hat{\theta}(x^n)) = \hat{\theta}^{n_0(x^n)}(1-\hat{\theta})^{n-n_0(x^n)}$. Write this in the form

$$\hat{P}(x^n) = \frac{1}{\binom{n}{n_0(x^n)}} \times \pi_n(\hat{\theta}(x^n)), \tag{133}$$

where

$$\pi_n(\hat{\theta}(x^n)) = \frac{\binom{n}{n_0(x^n)} P(x^n; \hat{\theta}(x^n))}{\sum_m \binom{n}{m} (\frac{m}{n})^m (\frac{n-m}{n})^{n-m}}. \tag{134}$$

In order to evaluate the resulting ideal code length $-\ln \hat{P}(x^n)$ we use the important and ubiquitous Stirling's approximation formula in the form refined by Robbins:

$$\ln n! = (n + 1/2) \ln n - n + \ln \sqrt{2\pi} + R(n), \tag{135}$$

where

$$\frac{1}{12(n+1)} \leq R(n) \leq \frac{1}{12n}.$$

This permits us to evaluate the terms in the sum in Equation 134 to a sufficient accuracy for us as:

$$\ln \binom{n}{m} \cong nh(m/n) - \frac{1}{2} \ln[(m/n)(n-m)/n] - \frac{1}{2} \ln n - \ln \sqrt{2\pi},$$

where $h(p)$ is the binary entropy at p. This gives

$$\binom{n}{m}(\frac{m}{n})^m(\frac{n-m}{n})^{n-m} \cong \frac{1}{\sqrt{2\pi n}}[(m/n)(n-m)/n]^{-1/2}.$$

Recognizing the sum in the denominator of Equation 134 (with step length $1/n$ rather than $1/\sqrt{n}$) as an approximation of a Riemann integral, we get it approximately as

$$\sqrt{n/(2\pi)}\int_0^1 \frac{1}{\sqrt{p(1-p)}}dp = \sqrt{n\pi/2},$$

where the integral of the square root of the Fisher information $J(p) = 1/(p(1-p))$ is a Dirichlet integral with the value π. Finally,

$$-\log \hat{P}(x^n) = nh(n_0/n) + \frac{1}{2}\log\frac{n\pi}{2} + o(1). \tag{136}$$

The added term $o(1)$, which goes to zero at the rate $O(1/n)$ as $n \to \infty$, takes care of the errors made by the application of Stirling's formula. The length of the intervals $B_{i,n}$ giving the optimal quantization is by (127)

$$|B_{i,n}| = (\frac{4}{n})^{1/2}((m/n)(1-m/n))^{-1/2},$$

where by the approximations made m should not be too close to zero nor unity. We also see that the cells are not the equivalence classes of strings of length n with m ones, whose number is n. Rather, the number of distinguishable models is $O(\sqrt{n})$

6.3 Prediction Bound for α-Loss Functions

Consider the $\alpha-$loss function $\delta(y-\hat{y}) = |y-\hat{y}|^\alpha$ for $\alpha > 0$, where $\hat{y}_t = f_t(y^{t-1}, \mathbf{x}^n)$, $f_1(y^0, \mathbf{x}^n) = 0$, is any predictor. The normalizing coefficient is given by (54)

$$\int e^{-\lambda|y-\mu|^\alpha}dy = Z_\lambda = \frac{2}{\alpha}\lambda^{-1/\alpha}\Gamma(1/\alpha). \tag{137}$$

For $\mu = \theta'\mathbf{x}_t$ we have the induced parametric models

$$p(y^n|\mathbf{x}^n;\lambda,\theta) = Z^{-n}(\lambda)e^{-\lambda L(Y^n|\mathbf{x}^n;\theta)},$$

where

$$L(Y^n|\mathbf{x}^n;\theta) = \sum_t \delta(y_t, \theta'\mathbf{x}_t)$$

and which we write p_θ for short. Let $\mathcal{M}_{\lambda,\alpha}$ denote the class of such models for $\theta \in \Omega \subset R^k$.

For the predictor $\mu = f_t(y^{t-1}, \mathbf{x}^n)$ we put

$$L_f(y^n|\mathbf{x}^n) = \sum_t \delta(y_t, \hat{y}_t), \tag{138}$$

and we get the induced model p_f.

Consider the minmax problem

$$\min_f \max_{p_\theta} E_{p_\theta}[L_f(Y^n|\mathbf{x}^n) - L(Y^n|\mathbf{x}^n;\hat{\theta}(Y^n,\mathbf{x}^n))]. \tag{139}$$

Theorem 18 *Let $\hat{y}_t = f_t(y^{t-1}, \mathbf{x}^n)$ be any predictor. Then for all α in the interval $[1, 2]$ and all positive ϵ the inequality*

$$\frac{1}{n} E_{p_\theta} L_f(Y^n|\mathbf{x}^n) \geq \frac{1}{\alpha\lambda}(1 + (k - \epsilon)\frac{\alpha}{2n} \ln n) \tag{140}$$

holds for n large enough and for all $\theta \in \Omega$, except in a set whose volume goes to zero as n grows to infinity; the expectation is with respect to p_θ.

Proof:

Consider

$$E_{p_\theta} \ln \frac{p_\theta}{p_f} = \lambda E_{p_\theta}(L_f(Y^n|\mathbf{x}^n) - L(Y^n|\mathbf{x}^n; \theta)).$$

For $\alpha \in [1, 2]$ one can show that the ML estimates satisfy the central limit theorem and the conditions in Theorem 15 are satisfied. The right hand side exceeds $\frac{k-\epsilon}{2} \ln n$ with the quantifications given. As in (52) we have

$$E_{p_\theta} L(Y^n|\mathbf{x}^n; \theta) = \frac{n}{\alpha\lambda},$$

and the claim follows.

6.4 The MDL Principle

We have shown in the preceding section that the universal *NML* models $\hat{f}(x^n; \mathcal{M}_\gamma)$ and $\hat{f}(x^n; \mathcal{M})$, respectively, permit us to extract all the useful information in the data that can be extracted with these model classes. Consider two model classes $\hat{f}(x^n; \mathcal{M}_\gamma)$ and $\hat{f}(x^n; \mathcal{M}'_{\gamma'})$ and their universal sufficient statistics decompositions. Suppose the stochastic complexity of the former is smaller than that of the latter. Because the code length for the optimal model, the information, is proportional to $\log n$, and that of the noise is proportional to n, it is the dominant part in the stochastic complexity. Hence, a part of what looks like noise in the model class $\hat{f}(x^n; \mathcal{M}'_{\gamma'})$ has been explained as useful information by the other model class. In other words, the model class $\hat{f}(x^n; \mathcal{M}_\gamma)$ has been able to extract more properties from the data rendering less as meaningless noise. To take a simple example, consider the alternating binary string 010101.... In light of the Bernoulli class with the probability of each symbol one half this string is all noise, and the extracted useful information is zero. On the other hand, a first order Markov model with the conditional probability of symbol 1 unity at state 0, and of symbol 0 unity at state 1, will render the entire string as useful information leaving nothing unexplained as noise. Hence, we see that it is the code length for noise rather than that for the optimal model that is decisive. After all, two different model classes may extract entirely different properties from the data and impose very different constraints, the learning of which is the very purpose of all modeling. These considerations suggest:

- The best model/model class among a collection of tentatively suggested ones is the one that gives the smallest stochastic complexity to the given data.

We call this the *MDL* for (Minimum Description Length) principle. It represents a drastically different foundation for model selection and, in fact, statistical inference in general. It has a number of distinctive features such as

- There is no need to assume anything about how the existing data were generated. In particular, unlike in traditional statistics, the data need not be assumed to form a sample from a population with some probability law.

- Hence, the objective is not to estimate an assumed but 'unknown' distribution, be it inside or outside the proposed class of models, but to find good models for the data.

- Most importantly, the principle permits comparison of any two models/model classes, regardless of their type. Hence, it provides a vastly more general criterion than *AIC*, *BIC*, and others that depend only on the number of parameters.

The application of the principle requires, of course, the calculation of the stochastic complexity, which for complex model classes can be a difficult task, and in practice we may have to settle for an upper bound approximation. A reasonable way to proceed is to decompose a complex model class into smaller ones, for which the formula derived above is valid. The code length required for the links needed to put these together can be estimated, usually by visualizing a concrete coding scheme. An alternative is to replace the stochastic complexity formula by the ideal code length obtained with the predictive universal model, which despite its own problems, mostly the initialization difficulty, is often applicable.

It is important to realize that the *MDL* principle has nothing to say about how to select the suggested family of model classes. In fact, this is a problem that cannot be adequately formalized. In practice their selection is based on human judgement and prior knowledge of the kinds of models that have been used in the past, perhaps by other researchers.

We would like to mention another important issue related to the restriction of the use of models, for instance for prediction. The issue has apparently been discussed for the first time in [9]. Since in the MDL view no model in general captures all the relevant regular features in the data we cannot expect to be able to predict reliably just any properties of the data we wish. Rather, we should predict only those properties we have captured by our model. Notice, that if the model is regarded as an approximation of an imagined 'true' underlying distribution, as the case is in traditional statistics, there are no restrictions on the properties one can and should predict. To be concrete, suppose we find that the MDL model in a class of all Markov models turns out to be a Bernoulli-model even though the data are generated by a first order Markov process. This can be the case only if there is not sufficient amount of data to justify the correct model. However, the prediction $\hat{p}(x^n) = m/n$ of the probability that the symbol is 0, where m denotes the number of zeros in the data string x^n, is still reliable. In fact, even if you had fitted a Markov model of the first order, you would get the same estimate of the probability that the state equals 0. By contrast, the prediction of the probability $P(00) = \hat{p}^2(x^n)$ with the optimal Bernoulli model would be completely unreliable and would differ from the estimate of the relative number of occurrences of two consecutive zeros in the future data that are typical for the correct first order Markov process. In conclusion one can say that the MDL

models can be used to provide reliable estimates of all the properties captured by the model - even though the models are imperfect; compare with Theorem 14.

7 Applications

7.1 Linear Regression

The linear least squares regression problem, such as polynomial fitting, is a fundamental modeling problem, for which we can give an exact formula for the stochastic complexity and hence applicable even for small data sets. Due to the importance of the problem we discuss it in some detail; for a full treatment see [26].

We consider the basic linear regression problem, where we have data of type $(y_t, x_{1t}, x_{2t}, \ldots)$ for $t = 1, 2, \ldots, n$, and we wish to learn how the values x_{it}, $i = 1, 2, \ldots, K$, of the *regressor* variables $\{x_i\}$ influence the corresponding values y_t of the *regression* variable y.

There may be a large number of the regressor variables, and the problem of interest is to find out which subset of them may be regarded to be the most important. This is clearly a very difficult problem, because we must be able to compare the performance of subsets of different sizes. We fit a linear model of type

$$y_t = \beta' \underline{x}_t + \epsilon_t = \sum_{i \in \gamma} \beta_i x_{it} + \epsilon_t, \tag{141}$$

where $\gamma = \{i_1, \ldots, i_k\}$ denotes a subset of the indices of the regressor variables; the prime denotes transposition, and for the computation of the required code lengths the deviations ϵ_t are modeled as samples from an iid Gaussian process of zero mean and variance $\tau = \sigma^2$, also as a parameter. In such a model the response data $y^n = y_1, \ldots, y_n$, are also normally distributed with the density function

$$f(y^n; \gamma, \beta, \tau) = \frac{1}{(2\pi\tau)^{n/2}} e^{-\frac{1}{2\tau} \sum_t (y_t - \beta' \underline{x}_t)^2}, \tag{142}$$

where $X'_\gamma = \{x_{it} : i \in \gamma\}$ is the $k \times n$ matrix defined by the values of the regressor variables with indices in γ. Write $Z_\gamma = X'_\gamma X_\gamma = n\Sigma_\gamma$, which is taken to be positive definite. The development for a while will be for a fixed γ, and we drop the subindex γ in the matrices above as well as in the parameters. The maximum likelihood solution of the parameters is given by

$$\hat{\beta}(y^n) = Z^{-1} X' y^n \tag{143}$$

$$\hat{\tau}(y^n) = \frac{1}{n} \sum_t (y_t - \hat{\beta}'(y^n) \underline{x}_t)^2. \tag{144}$$

We next consider the *NML* density function

$$\hat{f}(y^n; \gamma) = \frac{f(y^n; \gamma, \hat{\beta}(y^n), \hat{\tau}(y^n))}{\int_{Y(\tau_0, R)} f(z^n; \gamma, \hat{\beta}(z^n), \hat{\tau}(z^n)) dz^n}, \tag{145}$$

where y^n is restricted to the set

$$Y(\tau_0, R) = \{z^n : \hat{\tau}(z^n) \geq \tau_0,\ \hat{\beta}'(y^n) \Sigma \hat{\beta}(y^n) \leq R\} \tag{146}$$

Further, $Y(\tau_0, R)$ is to include x^n.

The numerator in Equation (145) has a very simple form

$$f(y^n; \gamma, \hat{\beta}(y^n), \hat{\tau}(y^n)) = 1/(2\pi e\hat{\tau}(y^n))^{n/2}, \tag{147}$$

and the problem is to evaluate the integral in the denominator. We can do it by using the facts that $\hat{\beta}$ and $\hat{\tau}$ are sufficient statistics for the family of normal models given, and that they are independent by Fisher's lemma. Hence if we with $\theta = (\beta, \tau)$ rewrite $f(y^n; \gamma, \beta, \tau) = f(y^n; \gamma, \theta)$, then we have first the factorization of the joint density function for y^n and $\hat{\theta}(y^n)$, which of course is still $f(y^n; \gamma, \theta)$, as the product of the marginal density of $\hat{\theta}$ and the conditional density of y^n given $\hat{\theta}$

$$f(y^n; \gamma, \theta) = f(y^n|\hat{\theta}(y^n); \gamma, \theta)p(\hat{\theta}(y^n); \gamma, \theta). \tag{148}$$

By the sufficiency of the statistic $\hat{\theta}$ we also have

$$f(y^n; \gamma, \theta) = h(y^n)p(\hat{\theta}(y^n); \gamma, \theta), \tag{149}$$

which shows that $f(y^n|\hat{\theta}(y^n); \gamma, \theta) = h(y^n)$ is actually independent of θ. Moreover,

$$p(\hat{\theta}; \gamma, \theta) = p_1(\hat{\beta}; \theta)p_2(\hat{\tau}; \tau), \tag{150}$$

where p_1 is normal with mean β and covariance $\frac{\tau}{n}\Sigma^{-1}$ while p_2 is obtained from the χ^2 distribution for $n\hat{\tau}/\tau$ with $n - k$ degrees of freedom.

Integrating the conditional $f(y^n|\hat{\theta}(y^n); \gamma, \theta) = h(y^n)$ over y^n such that $\hat{\theta}(y^n)$ equals any fixed value $\hat{\theta}$ yields unity. Therefore with $p(\hat{\theta}; \hat{\theta}) \equiv g(\hat{\tau}) = p_1(\hat{\beta}; \hat{\theta})p_2(\hat{\tau}; \hat{\theta})$ we get from the expression for the χ^2 density function in (150),

$$\begin{aligned} C(\tau_0, R) &= \int_{Y(\tau_0,R)} f(y^n; \hat{\theta}(y^n))dy^n & (151)\\ &= \int_{\tau_0}^{\infty} \hat{\tau}^{-\frac{k+2}{2}} d\hat{\tau} \int_{B_R} d\beta & (152)\\ &= \frac{2^{2-n/2}(\frac{n-k}{e})^{\frac{n-k}{2}}}{k\Gamma(\frac{k}{2})\Gamma(\frac{n-k}{2})}(\frac{R}{\tau_0})^{k/2} & (153)\\ &= A_{n,k}V_k\frac{2}{k}\left(\frac{R}{\tau_0}\right)^{k/2}, & (154) \end{aligned}$$

where $B_R = \{\beta : \beta'\Sigma\beta \leq R\}$ is an ellipsoid,,

$$V_k R^{k/2} = |\Sigma|^{-1/2}\frac{2\pi^{k/2}R^{k/2}}{k\Gamma(k/2)} \tag{155}$$

its volume, and

$$A_{n,k} = \frac{|\Sigma|^{1/2}}{(n\pi)^{k/2}}\frac{(\frac{n}{2e})^{\frac{n}{2}}}{\Gamma(\frac{n-k}{2})}. \tag{156}$$

We then have the *NML* density function itself for $0 < k < n$ and $y = y^n$

$$-\log \hat{f}(y; \gamma, \tau_0, R) = \frac{n}{2}\ln\hat{\tau} + \frac{k}{2}\ln\frac{R}{\tau_0} - \ln\Gamma(\frac{n-k}{2}) - \ln\Gamma(\frac{k}{2}) + \ln\frac{4}{k^2} + \frac{n}{2}\ln(n\pi). \tag{157}$$

We wish to get rid of the two parameters R and τ_0, which clearly affect the criterion in an essential manner, or rather we replace them with other parameters which do not influence the relevant criterion. We can set the two parameters to the values that minimize (157): $R = \hat{R}$, and $\tau_0 = \hat{\tau}$, where $\hat{R} = \hat{\beta}'(y)\Sigma\hat{\beta}(y)$. However, the resulting $\hat{f}(y;\gamma,\hat{\tau}(y),\hat{R}(y))$ is not a density function. We can of course correct this by multiplying it by a prior $w(\hat{\tau}(y),\hat{R}(y))$, but the result will be a density function on the triplet $y,\hat{\tau}(y),\hat{R}(y)$, which is not quite right. We do it instead by the same normalization process as above:

$$\hat{f}(y;\gamma) = \frac{\hat{f}(y;\gamma,\hat{\tau}(y),\hat{R}(y))}{\int_Y \hat{f}(z;\gamma,\hat{\tau}(z),\hat{R}(z))dz}, \tag{158}$$

where the range Y will be defined presently. By (148) and the subsequent equations we also have the factorization

$$\hat{f}(y;\gamma,\tau_0,R) = f(y|\gamma,\hat{\beta},\hat{\tau})p(\hat{\tau})/C(\tau_0,R) = f(y|\gamma,\hat{\beta},\hat{\tau})\frac{k}{2}\hat{\tau}^{-k/2-1}V_k^{-1}\left(\frac{\tau_0}{R}\right)^{k/2}. \tag{159}$$

As above we can now integrate the conditional while keeping $\hat{\beta}$ and $\hat{\tau}$ constant, which gives unity. Then by setting $\tau_0 = \hat{\tau}$ and $R = \hat{R}$ we integrate the resulting function $1/\hat{\tau}$ of $\hat{\tau}$ over a range $[\tau_1,\tau_2]$. Finally, noticing that the remaining function of $\hat{\beta}$ is constant, $\hat{R}$, on the surface of the ellipsoid $\hat{\beta}'\Sigma\hat{\beta} = \hat{R}$, its integral amounts to the integration of the surface area multiplied by $\hat{R}^{-k/2}$ with respect to $\hat{R}$ over a range $[R_1,R_2]$. All told we get

$$\int_Y \hat{f}(z;\gamma,\hat{\tau}(z),\hat{R}(z))dz = \frac{k}{2}\ln\frac{\tau_2 R_2}{\tau_1 R_1}, \tag{160}$$

where the parameters are such that the ranges they define include $\hat{\tau}(y)$ and $\hat{R}(y)$, respectively. The negative logarithm of $\hat{f}(y;\gamma)$ is then given by

$$-\ln\hat{f}(y;\gamma) = \frac{n-k}{2}\ln\hat{\tau} + \frac{k}{2}\ln\hat{R} - \ln\Gamma(\frac{n-k}{2}) - \ln\Gamma(\frac{k}{2}) + \ln\frac{2}{k} + \frac{n}{2}\ln(n\pi) + \ln\ln\frac{\tau_2 R_2}{\tau_1 R_1}. \tag{161}$$

Because the last term does not depend on γ nor k, we do not indicate the dependence of $\hat{f}(y;\gamma)$ on the new parameters. It is most interesting that a very closely related formula is obtained by fitting the hyperparameters in a mixture density to data, [10].

We wish to extend the density function $\hat{f}(y;\gamma)$ to the larger class of models, defined as the union over all submatrices Z of the fixed W, and to obtain a criterion for finding the optimal index set γ and the associated optimal model. We begin with the MDL estimator $\hat{\gamma}(\cdot)$, obtained by minimizing the ideal code length for the data $-\ln\hat{f}(y;\gamma)$ with respect to γ. Although the result $\hat{f}(y;\hat{\gamma}(y))$ is not a density function we get one by the normalization process

$$\hat{f}(y;\Omega) = \frac{\hat{f}(y;\hat{\gamma}(y))}{\int_{\hat{\gamma}(z)\in\Omega}\hat{f}(z;\hat{\gamma}(z))dz}, \tag{162}$$

where Ω is a set of indices such that it includes $\hat{\gamma}(y)$. The denominator, call it C, is given by

$$C = \sum_{\gamma\in\Omega}\hat{P}_n(\gamma), \tag{163}$$

where

$$\hat{P}_n(\gamma) = \int_{\{y:\hat{\gamma}(y)=\gamma\}} \hat{f}(y;\hat{\gamma}(y))dy. \tag{164}$$

This defines the *canonical* prior

$$\hat{\pi}_n(\gamma) = \frac{\hat{P}_n(\gamma)}{\sum_{\alpha\in\Omega}\hat{P}_n(\alpha)}, \tag{165}$$

and we get the factorization

$$\hat{f}(y;\Omega) = \frac{\hat{f}(y;\hat{\gamma}(x))}{\hat{P}_n(\hat{\gamma}(y))}\hat{\pi}_n(\hat{\gamma}(y)). \tag{166}$$

In analogy with $\hat{f}(y;\gamma)$ we call $\hat{f}(y;\Omega)$ the *NML* density function for the model class with the index sets in Ω. It then gives the final decomposition

$$-\ln\hat{f}(y;\Omega) = \frac{n-\hat{k}}{2}\ln\hat{\tau} + \frac{\hat{k}}{2}\ln\hat{R} - \ln\Gamma(\frac{n-\hat{k}}{2}) - \ln\Gamma(\frac{\hat{k}}{2}) + \ln\frac{1}{\hat{k}} + \ Const, \tag{167}$$

where we include in *Const* all the terms that do not depend on the optimal index set $\hat{\gamma}$ of size $\hat{k}$. Notice that the terms that depend on $\hat{k}$ differ slightly from (161). The terms other than the first define the length of a code from which the optimal normal model, defined by the ML parameters, in the subclass specified by the term *Const* can be decoded, while the first term represents the code length of the part of the data that adds no further information about the optimal model. It may be viewed as noise. Hence this decomposition is similar to Kolmogorov's sufficient statistics decomposition in the algorithmic theory of information, and it is also seen to extend the ordinary sufficient statistics, as defined for certain parametric families, to parameter free *universal* sufficient statistics.

By applying Stirling's approximation to the $\Gamma-$functions we get the *NML* criterion for $0 < k < n$

$$\min_{\gamma\in\Omega}\{(n-k)\ln\hat{\tau} + k\ln(n\hat{R}) + (n-k-1)\ln\frac{n}{n-k} - (k+1)\ln k\}, \tag{168}$$

where k denotes the number of elements in γ.

7.2 MDL Denoising

An important special case of the linear regression problem is the so-called 'denoising' problem, which is to remove noise from a data sequence $x' = x^n = x_1,\ldots,x_n$, taken as a row vector, so that the remaining 'smooth' signal $\bar{x}' = \bar{x}^n = \bar{x}_1,\ldots,\bar{x}_n$ represents the real information bearing data:

$$x_t = \bar{x}_t + \epsilon_t, \ \ t = 1,\ldots,n. \tag{169}$$

Ordinarily this is viewed as the problem of estimating the 'true' signal $\bar{x}_t$, expressed in an orthonormal basis $\underline{w}_i' = w_{i1},\ldots w_{in}$ for $i = 1,\ldots,n$, to which independent normally distributed noise of 0-mean and variance τ is added. The estimation is done by minimizing the risk criterion $R = \sum_t E(x_t - \bar{x}_t)^2 = n\tau$, where the expectation with respect to the assumed normal distribution is to be estimated from the data.

The trouble with this is the estimation of τ. In fact, the regression matrix $W' = \{w_{ij}\}$, defined by the rows $\underline{w}_i'$, has the transpose W as the inverse, and it defines the 1-1 transformation

$$\begin{aligned} x &= Wc \\ c &= W'x, \end{aligned} \tag{170}$$

where x and c denote the column vectors of the strings of the data $x' = x_1, \ldots, x_n$ and the coefficients $c' = c_1, \ldots, c_n$, respectively. Because of orthonormality Parseval's equality $c'c = \sum_t c_t^2 = x'x = \sum_t x_t^2$ holds. Accordingly, by taking $\hat{x} = x$ gives the natural estimate $n\hat{\tau} = (x - \hat{x})'(x - \hat{x}) = 0$, which is a meaningless minimum. Hence, the estimation must be done by some arbitrary scheme using only a portion of the extracted smooth signal.

By contrast, the *MDL* solution to the regression problem poses no difficulty, and the universal sufficient statistics decomposition (167) provides a natural definion of noise as the part of the data that cannot be compressed with the selected normal model class, while the rest, defined by the optimal model, gives the desired information bearing signal. Instead of (169) we have the decomposition

$$x_t = \hat{x}_t + e_t, \quad t = 1, \ldots, n, \tag{171}$$

where $\hat{x}$ is defined by the optimal model in (167). The fact that W is orthonormal permits a dramatic simplification, which we can take advantage of even if for n very large we only want to use the regression matrix defined by some m, $m \leq n$, of the basis vectors.

Let, then, W' denote the so-defined $m \times n$-matrix. The transform equations get now replaced by

$$\begin{aligned} \tilde{x} &= Wc \\ c &= W'\tilde{x} = W'x, \end{aligned} \tag{172}$$

where c denotes the ML-solution obtained with all the m available basis vectors; ie, with the entire $m \times n$-matrix W', and $\tilde{x}$ the corresponding orthogonal projection of x onto the space spanned by the basis vectors. Let $\gamma = \{i_1, \ldots, i_k\}$ denote a set of indices, $0 < k \leq m$, but $k < n$ if $m = n$, and let $\hat{c}'$ denote the row vector of components $\{\delta_i c_i\}$, where $\delta_i = 1$ for $i \in \gamma$, and zero otherwise. In words, the non-zero components in $\hat{c}'$ with indices in γ are the corresponding components of c'. These define the ML-estimates $\hat{\beta}$ in the notations of the previous section. The ML-estimate $\hat{\tau}$ is now given by

$$n\hat{\tau} = \sum_{t=1}^{n} (x_t - \hat{x}_t)^2 = c'c - \hat{c}'\hat{c}. \tag{173}$$

The criterion (168) for finding the best subset γ, including the number k of its elements, is then equivalent with

$$\min_{\gamma} C_\gamma(x) = \min_{\gamma} \{(n-k) \ln \frac{c'c - \hat{S}_k}{n-k} + k \ln \frac{\hat{S}_k}{k} - \ln \frac{k}{n-k}\}, \tag{174}$$

where

$$\hat{S}_k = \hat{x}'\hat{x} = \hat{c}'\hat{c}. \tag{175}$$

Because of Parseval's equality the sum of the squared deviations $\hat{\tau}$ is minimized by the k largest coefficients in absolute value. This has been utilized in the earlier solutions to the denoising problem in terms of a threshold λ for the coefficients such that all coefficients whose absolute value is less than the threshold are set to zero. Moreover, a threshold which has a certain minmax property for orthonormal transformations defined by wavelets was derived by Donoho and Johnstone to be just $\lambda = \sqrt{2\tau \ln n}$, where, as said, the variance τ has to be estimated. Because $\hat{S}$ in the second term of the criterion $C_\gamma(x)$ is maximized by the k largest squared coefficients, it is not clear at all that any threshold exits, and that the search for the minimizing index set $\hat{\gamma}$ through all the 2^m possible index sets can be avoided. However, we have the theorem

Theorem 19 *For orthonormal regression matrices the index set $\hat{\gamma}$ that minimizes the criterion (174) is given either by the indices $\hat{\gamma} = \{(1), \ldots, (k)\}$ of the k largest coefficients in absolute value or the indices $\hat{\gamma} = \{(m-k+1), \ldots, (m)\}$ of the k smallest ones for some $k = \hat{k}$.*

Proof: Let γ be an arbitrary collection of a fixed number of indices k, and let $\hat{S}_k$ be the corresponding sum of the squared coefficients. Let $u_i = c_i^2$ be a term in $\hat{S}_k$. The derivative of $C_\gamma(\mathbf{x})$ with respect to u_i is then

$$\frac{dC_\gamma(\mathbf{x})}{du_i} = \frac{k}{\hat{S}_k} - \frac{n-k}{\mathbf{c}'\mathbf{c} - \hat{S}_k}, \tag{176}$$

which is nonpositive when $\hat{S}_k/k \geq \hat{T}_k/(n-k)$, where $\hat{T}_k = n\hat{\tau}_k = \mathbf{c}'\mathbf{c} - \hat{S}_k$, and positive otherwise. The second derivative is always negative, which means that $C_\gamma(\mathbf{x})$ as a function of u_i is concave.

If for some γ, $\hat{S}_k/k > \hat{T}_k/(n-k)$, we can reduce $C_\gamma(\mathbf{x})$ by replacing, say, the smallest square c_i^2 in γ by a larger square outside of γ, and get another γ for which $\hat{S}_k$ is larger and $\hat{T}_k$ smaller. This process is possible until $\gamma = \{(1), \ldots, (k)\}$ consists of the indices of the k largest squared coefficients. Similarly, if for some γ, $\hat{S}_k/k < \hat{T}_k/(n-k)$, we can reduce $C_\gamma(\mathbf{x})$ until γ consists of the indices of the k smallest squared coefficients. Finally, if for some γ, $\hat{S}_k/k = \hat{T}_k/(n-k)$, then all the squared coefficients must be equal, and the claim holds trivially.

Remarks:

It may seem weird that the threshold defined by $\hat{k}$ could require setting coefficients that exceed the threshold to zero. However, we have made no assumptions about the data to which the criterion is to be applied, and it can happen that the signal $\hat{x} = \hat{x}^n$ recovered is defined by a model which is more complex than the noise $x^n - \hat{x}^n$, relative to the two classes of distributions considered, the normal one for the noise and the uniform for the models. Hence, in such a case it may pay to reverse the roles of the information bearing signal and the noise.

For the denoising problem, where $m = n$, we obtain by direct calculation

$$\bar{C}_{(k)}(x) = C_{(n-k)}(x) + 2 \ln \frac{n-k}{k}. \tag{177}$$

In the case of main interest the signal recovered is simple in the sense that the index set $\hat{\gamma}$ minimizing the criterion (174) has fewer than $n/2$ elements. Indeed, it follows from (177) that $\bar{C}_{(k)}(x)$ cannot

be the minimum. In fact, $C_{\hat{\gamma}}(x) \leq C_{(n-i)}(x) < \bar{C}_{(i)}(x)$ for all $i < n/2$, which in view of Theorem 2 implies $C_{\hat{\gamma}}(x) = C_{(\hat{k})}(x)$. For denoising, then, we should simply optimize the criterion (174) over the k largest coefficients in absolute value thus

$$\min_k C_{(k)}(x) = \min_k \{(n-k) \ln \frac{c'c - \hat{S}_{(k)}}{n-k} + k \ln \frac{\hat{S}_{(k)}}{k} - \ln \frac{k}{n-k}. \tag{178}$$

With $\hat{c}^n$ denoting the column vector defined by the coefficients $\hat{c}_1, \ldots, \hat{c}_n$, where $\hat{c}_i = c_i$ for $i \in \{(1), \ldots, (\hat{k})\}$ and zero, otherwise, the signal recovered is given by $\hat{x}^n = W\hat{c}^n$.

7.3 Examples

We calculate two examples using wavelets defined by Daubechies' N=6 scaling function. The first example is a case where the *MDL* threshold is close to those obtained with traditional techniques, as well as to the threshold, obtained with a rather complicated cross-validation technique. The second example is about real speech data,

where the information bearing signal, recovered by the *NML* criterion (178), differs a lot from the result obtained with the universal threshold due to Donoho and Johnstone.

In the first example the mean signal $\bar{x}_i$ consists of 512 equally spaced samples of the following function defined by three piecewise polynomials

$$x(t) = \begin{cases} 4t^2(3-4t) & \text{for } t \in [0, .5] \\ \frac{4}{3}t(4t^2 - 10t + 7) - \frac{3}{2} & \text{for } t \in [.5, .75] \\ \frac{16}{3}t(t-1)^2 & \text{for } t \in [.75, 1] \end{cases}$$

To the data points $\bar{x}_i$ were added pseudorandom normal 0-mean noise with standard deviation of 0.1, which defined the data sequence x_i.

The threshold obtained with the *NML* criterion is $\lambda = 0.246$. This is between the two thresholds called VisuShrink $\lambda = 0.35$ and GlobalSure $\lambda = 0.14$, both of the type derived by Donoho and Johnstone. It is also close to the threshold $\lambda = 0.20$, obtained with the much more complex cross-validation procedure by Nason.

In the second example the data sequence consists of 128 samples from a voiced portion of speech. The *NML* criterion retains 42 coefficients exceeding the threshold $\lambda = 7.3$ in absolute value. It gives the value $\hat{\tau} = 5.74$ for the noise variance. The estimation procedure by Donoho and Johnstone gives $\hat{\tau} = 10.89$ and the threshold $\lambda = \sqrt{2\hat{\tau} \ln 128} = 10.3$. Figure 1 shows the original signal together with the information bearing signal extracted by the *NML* criterion and the much smoother DJ-signal obtained with the threshold of Donoho and Johnstone. We see that the latter is far too smooth and fails to capture the important large pulses in the original signal. We also see that the *NML* criterion has not removed the sharp peaks in the large pulses despite the fact that they have locally high frequency content. They simply can be compressed by use of the retained coefficients, and by the general principle behind the criterion they are not regarded as noise.

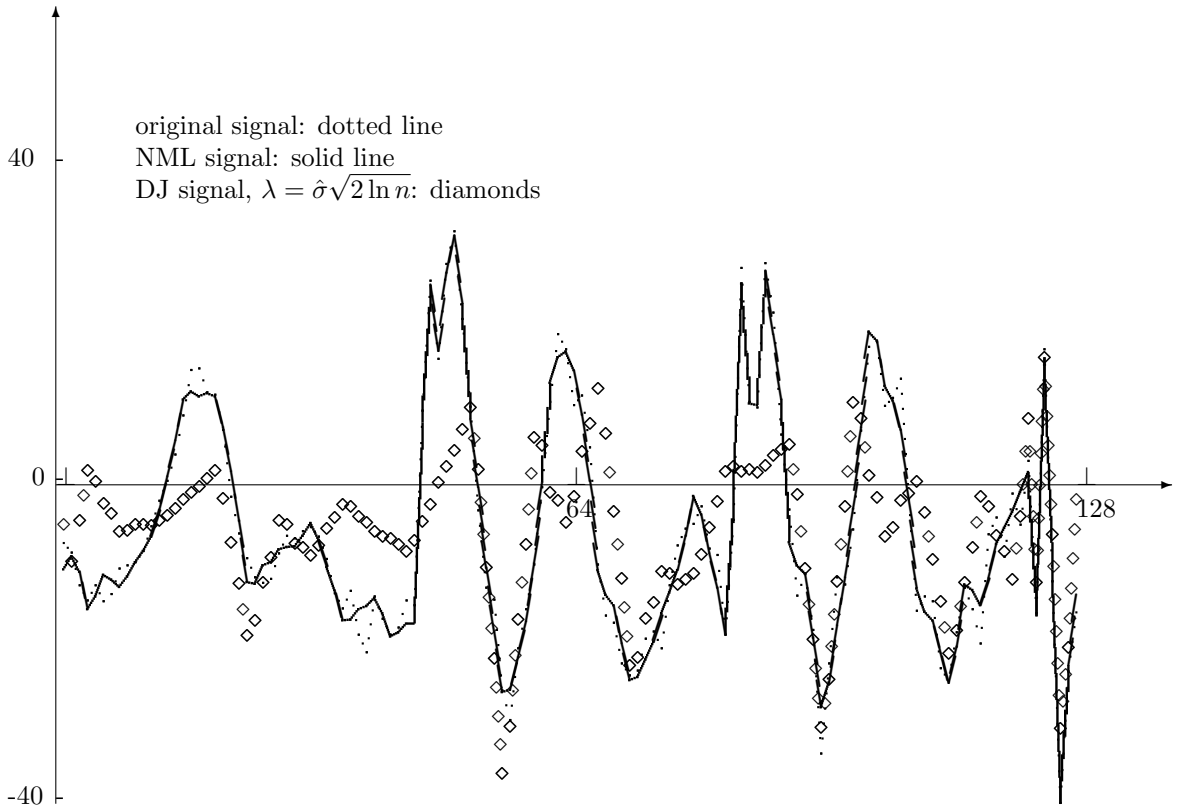

Figure 1. Speech signal smoothed with Daubechies' N=6 wavelet

References

[1] Akaike, H. (1973), 'Information theory as an extension of the maximum likelihood principle', pages 267-281 in B.N. Petrov and F. Csaki, (eds) *Second International Symposium on Information Theory.* Akademiai Kiado, Budapest

[2] Balasubramanian, V. (1996),'Statistical Inference, Occam's Razor and Statistical Mechanics on the Space of Probability Distributions', *Neural Computation*, **9**, No. 2, 349-268, 1997 http://arxiv.org/list/nlin/9601

[3] Boltzmann, L. (1895), *Vorlesungen uber Gastheorie*, 1er Theil, Leibzig, Barth.

[4] Barron, A.R., Rissanen, J., and Yu, B. (1998), 'The MDL Principle in Modeling and Coding', special issue of *IEEE Trans. Information Theory* to commemorate 50 years of information theory, Vol. **IT-44**, No. 6, October 1998, pp 2743-2760

[5] Chaitin, G.J. (1969), 'On the Length of Programs for Computing Finite Binary Sequences: Statistical Considerations", *JACM,* **16**, 145-159

[6] Cover, T. and Thomas, J. (1991), *Elements of Information Theory*, John Wiley and Sons, Inc., New York, 542 pages

[7] Davisson, L. D. (1973), 'Universal Noiseless Coding', *IEEE Trans. Information Theory*, Vol. **IT-19**, Nr 6, 783-795, November 1973

[8] Gradshteyn, I.S. and Ryzhik, I.M., *Table of Integrals, Series and Products*, Academic Press, New York, 1980, 1160 pages

[9] Grünwald, P.D. (1998), *The Minimum Description Length Principle and reasoning under Uncertainty*, PhD thesis, Institute for Logic, Language and Computation, Universiteit van Amsterdam, 296 pages

[10] Hansen, M.H. and Yu, B. (2001), 'Model Selection and the Principle of Minimum Description Length', JASA (to appear)

[11] Hartley, R.V. (1928), 'Transmission of Information', Bell System Technical Journal, **7**, 535-563

[12] Jeffreys, H. (1961) *Theory of Probability*, Clarendon Press, Oxford, 447 pages, (third edition)

[13] Kolmogorov, A.N. (1965), 'Three Approaches to the Quantitative Definition of Information', Problems of Information Transmission **1**, 4-7

[14] Lempel, A. and Ziv, J., 'Compression of Individual Sequences via Variable Rate Coding', *IEEE Trans. Information Theory*, Vol. **IT-24**, 530-536, September 1978

[15] Li, M. and Vitanyi, P. (1993), *An Introduction to Kolmogorov Complexity and Its Applications,* Springer-Verlag, 546 pages

[16] Merhav, N. and Feder, M. (1995), 'A Strong Version of the Redundancy-Capacity Theorem of Universal Coding', *IEEE Trans. Information Theory*, Vol. **IT-41**, No. 3, pp 714-722, May 1995.

[17] Nohre, R. (1994), *Some Topics in Descriptive Complexity*, PhD Thesis, Linkoping University, Linkoping, Sweden

[18] Pasco, R. (1976), *Source coding algorithms for fast data compression*, PhD thesis, Stanford University

[19] Rissanen, J. and Langdon, G.G. Jr, 'Universal Modeling and Coding', *IEEE Trans. Information Theory*, Vol. **IT-27**, Nr. 1, 12-23

[20] Rissanen, J. (1976), 'Generalized Kraft Inequality and Arithmetic Coding', IBM J. Res. Dev. **20**, Nr 3, 198-203

[21] Rissanen, J. (1983), 'A Universal Data Compression System', *IEEE Trans. Information Theory*, Vol. **IT-29**, Nr. 5, 656-664

[22] Rissanen, J. (1984), 'Universal Coding, Information, Prediction, and Estimation', *IEEE Trans. Information Theory*, Vol. **IT-30**, Nr. 4, 629-636

[23] Rissanen, J. (1986), 'Stochastic Complexity and Modeling', Annals of Statistics, Vol **14**, 1080-1100

[24] Rissanen, J. (1989), *Stochastic Complexity in Statistical Inquiry*, World Scientific Publ. Co., Suite 1B, 1060 Main Street, River Edge, New Jersey, (175 pages)

[25] Rissanen, J. (1996), 'Fisher Information and Stochastic Complexity', *IEEE Trans. Information Theory*, Vol. **IT-42**, No. 1, pp 40-47

[26] Rissanen, J. (2000), 'MDL Denoising', *IEEE Trans. on Information Theory*, Vol. **IT-46**, Nr. 7, November 2000.

[27] Rissanen, J. (2001), 'Strong Optimality of the Normalized ML Models as Universal Codes and Information in Data', *IEEE Trans. Information Theory*, Vol. **IT-47**, Nr. 5, July 2001.

[28] Shannon, C.E. (1948), 'A Mathematical Theory of Communication', Bell System Technical Journal, **27**, 379-423, 623-656

[29] Shtarkov, Yu. M. (1987), 'Universal Sequential Coding of Single Messages', Translated from Problems of Information Transmission, Vol. 23, No. 3, 3-17, July-September 1987.

[30] Solomonoff, R.J. (1964), 'A Formal Theory of Inductive Inference', Part I, Information and Control **7**, 1-22; Part II, Information and Control **7**, 224-254

[31] Takeuchi, Jun-ichi and Barron, Andrew R. (1998), 'Robustly Minimax Codes for Universal Data Compression', *The 21'st Symposium on Information Theory and Its Applications*, Gifu, Japan, December 2-5, 1998.

[32] Vereshchagin, N. and Vitanyi, P. (2002), 'Kolmogorov's Structure Functions with an Application to the Foundation of Model Selection', (personal communication)

[33] Vovk, V.G. (1990), 'Aggregating Strategies', Proceedings of 3'rd Annual Workshop on Computational Learning Theory, Morgan Kauffman, pp 371-386.

[34] Weinberger, M.J., Rissanen, J., and Feder, M. (1995), 'A Universal Finite Memory Source', *IEEE Trans. on Information Theory*, Vol. **IT-41**, No. 3, pp 643-652, May 1995

[35] White, H. (1994), *Estimation, inference, and specification analysis*, Cambridge University press, Cambridge, UK, 349 pages

[36] Wiener, N. (1961b), *Cybernetics*, (First edition, 1948), Second edition (revisions and two additional chapters), The MIT Press and Wiley, New York, 1961.

[37] Yamanishi, Kenji (1998), 'A Decision-Theoretic Extension of Stochastic Complexity and Its Application to Learning', *IEEE Trans. on Information Theory*, Vol. **IT-44**, No. 4, July 1998.