1 Introducing the Minimum Description Length Principle

Peter Grünwald
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
The Netherlands
pdg@cwi.nl
www.grunwald.nl

This chapter provides a conceptual, entirely nontechnical introduction and overview of Rissanen's minimum description length (MDL) principle. It serves as a basis for the technical introduction given in Chapter 2, in which all the ideas discussed here are made mathematically precise.

1.1 Introduction and Overview

How does one decide among competing explanations of data given limited observations? This is the problem of *model selection.* It stands out as one of the most important problems of inductive and statistical inference. The minimum description length (MDL) principle is a relatively recent method for inductive inference that provides a generic solution to the model selection problem. MDL is based on the following insight: any regularity in the data can be used to *compress* the data, that is, to describe it using fewer symbols than the number of symbols needed to describe the data literally. The more regularities there are, the more the data can be compressed. Equating "learning" with "finding regularity," we can therefore say that the more we are able to compress the data, the more we have *learned* about the data. Formalizing this idea leads to a general theory of inductive inference with several attractive properties:

1. Occam's razor. MDL chooses a model that trades off goodness-of-fit on the observed data with 'complexity' or 'richness' of the model. As such, MDL embodies a form of Occam's razor, a principle that is both intuitively appealing and informally applied throughout all the sciences.

2. No overfitting, *automatically.* MDL procedures *automatically* and *inherently* protect against overfitting and can be used to estimate both the parameters and the structure (e.g., number of parameters) of a model. In contrast, to avoid overfitting when estimating the structure of a model, traditional methods such as maximum likelihood must be *modified* and extended with additional, typically ad hoc principles.

3. Bayesian interpretation. MDL is closely related to Bayesian inference, but avoids some of the interpretation difficulties of the Bayesian approach,[1] especially in the realistic case when it is known a priori to the modeler that none of the models under consideration is true. In fact:

4. No need for "underlying truth." In contrast to other statistical methods, MDL procedures have a clear interpretation independent of whether or not there exists some underlying "true" model.

5. Predictive interpretation. Because data compression is formally equivalent to a form of probabilistic prediction, MDL methods can be interpreted as searching for a model with good predictive performance on *unseen* data.

In this chapter, we introduce the MDL principle in an entirely nontechnical way, concentrating on its most important applications: model selection and avoiding overfitting. In Section 1.2 we discuss the relation between learning and data compression. Section 1.3 introduces model selection and outlines a first, 'crude' version of MDL that can be applied to model selection. Section 1.4 indicates how these crude ideas need to be refined to tackle small sample sizes and differences in model complexity between models with the same number of parameters. Section 1.5 discusses the philosophy underlying MDL, and considers its relation to Occam's razor. Section 1.7 briefly discusses the history of MDL. All this is summarized in Section 1.8.

1.2 The Fundamental Idea: Learning as Data Compression

We are interested in developing a method for *learning* the laws and regularities in data. The following example will illustrate what we mean by this and give a first idea of how it can be related to descriptions of data.

Regularity ... Consider the following three sequences. We assume that each sequence is 10000 bits long, and we just list the beginning and the end of each sequence.

```
00010001000100010001 ... 0001000100010001000100010001     (1.1)
01110100110100100110 ... 1010111010111011000101100010     (1.2)
00011000001010100000 ... 0010001000010000001000110000     (1.3)
```

The first of these three sequences is a 2500-fold repetition of `0001`. Intuitively, the sequence looks regular; there seems to be a simple 'law' underlying it; it might make sense to conjecture that future data will also be subject to this law, and to predict that future data will behave according to this law. The second sequence has been generated by tosses of a fair coin. It is, intuitively speaking, as "random as possible," and in this sense there is no regularity underlying it. Indeed, we cannot seem to find such a regularity either when we look at the data. The third sequence contains approximately four times as many 0s as 1s. It looks less regular, more random than the first, but it looks less random than the second. There is still some discernible regularity in these data, but of a statistical rather than of a deterministic kind. Again, noticing that such a regularity is there and predicting that future data will behave according to the same regularity seems sensible.

... and Compression We claimed that any regularity detected in the data can be used to *compress* the data, that is, to describe it in a short manner. Descriptions are always relative to some *description method* which maps descriptions D' in a unique manner to data sets D. A particularly versatile description method is a general-purpose computer language like C or PASCAL. A description of D is then any computer program that prints D and then halts. Let us see whether our claim works for the three sequences above. Using a language similar to PASCAL, we can write a program

```
for i = 1 to 2500; print "0001"; next; halt
```

which prints sequence (1.1) but is clearly a lot shorter. Thus, sequence (1.1) is indeed highly compressible. On the other hand, we show in Chapter 2, Section 2.1, that if one generates a sequence like (1.2) by tosses of a fair coin, then with extremely high probability, the shortest program that prints (1.2) and then halts will look something like this:

```
print "0111010011010000101010...1010111010111011000101100010"; halt
```

This program's size is about equal to the length of the sequence. Clearly, it does nothing more than repeat the sequence.

The third sequence lies in between the first two: generalizing $n = 10000$ to arbitrary length n, we show in Chapter 2, Section 2.1 that the first sequence can be compressed to $O(\log n)$ bits; with overwhelming probability, the second sequence cannot be compressed at all; and the third sequence can be compressed to some length αn, with $0 < \alpha < 1$.

Example 1.1 (compressing various regular sequences) The regularities underlying sequences (1.1) and (1.3) were of a very particular kind. To illustrate that *any* type of regularity in a sequence may be exploited to compress that sequence, we give a few more examples:

The Number π Evidently, there exists a computer program for generating the first n digits of π — such a program could be based, for example, on an infinite series expansion of π. This computer program has constant size, except for the specification of n which takes no more than $O(\log n)$ bits. Thus, when n is very large, the size of the program generating the first n digits of π will be very small compared to n: the π-digit sequence is deterministic, and therefore extremely regular.

Physics Data Consider a two-column table where the first column contains numbers representing various heights from which an object was dropped. The second column contains the corresponding times it took for the object to reach the ground. Assume both heights and times are recorded to some finite precision. In Section 1.3 we illustrate that such a table can be substantially compressed by first describing the coefficients of the second-degree polynomial H that expresses Newton's law, then describing the heights, and then describing the deviation of the time points from the numbers predicted by H.

Natural Language Most sequences of words are not valid sentences according to the English language. This fact can be exploited to substantially compress English text, as long as it is syntactically mostly correct: by first describing a grammar for English, and then describing an English text D with the help of that grammar [Grünwald 1996], D can be described using many fewer bits than are needed without the assumption that word order is constrained.

1.2.1 Kolmogorov Complexity and Ideal MDL

To formalize our ideas, we need to decide on a description method, that is, a formal language in which to express properties of the data. The most general choice is a general-purpose[2] computer language such as C or PASCAL. This choice leads to the definition of the *Kolmogorov complexity* [Li and Vitányi 1997] of a sequence as the length of the shortest program that prints the sequence and then halts. The lower the Kolmogorov complexity of a sequence, the *more regular* it is. This notion seems to be highly dependent on the particular computer language used. However, it turns out that for every two general-purpose programming languages A and B and every data sequence D, the length of the shortest program for D written in language A and the length of the shortest program for D written in language B differ by no more than a constant c, which does not depend on the length of D. This so-called *invariance theorem* says that, *as long as the sequence D is long enough*, it is not essential which computer language one chooses, as long as it is general-purpose. Kolmogorov complexity was introduced, and the invariance theorem was proved, independently by Kolmogorov [1965], Chaitin [1969] and Solomonoff [1964]. Solomonoff's paper, called "A Formal Theory of Inductive Inference," contained the idea that the ultimate model for a sequence of data may be identified with the shortest program that prints the data. Solomonoff's ideas were later extended by several authors, leading to an 'idealized' version of MDL [Solomonoff 1978; Li and Vitányi 1997; Gács, Tromp, and Vitányi 2001]. This idealized MDL is very general in scope, but not practically applicable, for the following two reasons:

1. *Uncomputability***.** It can be shown that there exists no computer program that, for every set of data D, when given D as input, returns the shortest program that prints D [Li and Vitányi 1997].

2. *Arbitrariness/dependence on syntax***.** In practice we are confronted with small data samples for which the invariance theorem does not say much. Then the hypothesis chosen by idealized MDL may depend on arbitrary details of the syntax of the programming language under consideration.

1.2.2 Practical MDL

Like most authors in the field, we concentrate here on nonidealized, practical versions of MDL that deal explicitly with the two problems mentioned above. The basic idea is to scale down Solomonoff's approach so that it does become applicable. This is achieved by using description methods that are less expressive than general-purpose computer languages. Such description methods C should be restrictive enough so that for any data sequence D, we can always compute the length of the shortest description of D that is attainable using method C; but they should be general enough to allow us to compress many of the intuitively "regular" sequences. The price we pay is that, using the "practical" MDL principle, there will always be some regular sequences which we will not be able to compress. But we already know that there can be *no* method for inductive inference at all which will always give us all the regularity there is — simply because there can be no automated method which for any sequence D finds the shortest computer program that prints D and then halts. Moreover, it will often be possible to guide a suitable choice of C by a priori knowledge we have about our problem domain. For example, below we consider a description method C that is based on the class of all polynomials, such that with the help of C we can compress all data sets which can meaningfully be seen as points on some polynomial.

1.3 MDL and Model Selection

Let us recapitulate our main insights so far:

MDL: The Basic Idea
The goal of statistical inference may be cast as trying to find regularity in the data. “Regularity” may be identified with “ability to compress.” MDL combines these two insights by *viewing learning as data compression*: it tells us that, for a given set of hypotheses $\mathcal{H}$ and data set D, we should try to find the hypothesis or combination of hypotheses in $\mathcal{H}$ that compresses D most.

This idea can be applied to all sorts of inductive inference problems, but it turns out to be most fruitful in (and its development has mostly concentrated on) problems of *model selection* and, more generally, those dealing with *overfitting*. Here is a standard example (we explain the difference between “model” and “hypothesis” after the example).

Example 1.2 (Model Selection and Overfitting) Consider the points in Figure 1.1. We would like to learn how the y-values depend on the x-values. To this end, we may want to fit a polynomial to the points. Straightforward linear regression will give us the leftmost polynomial — a straight line that seems overly simple: it does not capture the regularities in the data well. Since for any set of n points there exists a polynomial of the $(n-1)$st degree that goes exactly through all these points, simply looking for the polynomial with the least error will give us a polynomial like the one in the second picture. This polynomial seems overly complex: it reflects the random fluctuations in the data rather than the general pattern underlying it. Instead of picking the overly simple or the overly complex polynomial, it seems more reasonable to prefer a relatively simple polynomial with a small but nonzero error, as in the rightmost picture. This intuition is confirmed by numerous experiments on real-world data from a broad variety of sources [Rissanen 1989; Vapnik 1998; Ripley 1996]: if one naively fits a high-degree polynomial to a small sample (set of data points), then one obtains a very good fit to the data. Yet if one *tests* the inferred polynomial on a second set of data coming from the same source, it typically fits these test data very badly in the sense that there is a large distance between the polynomial and the new data points. We say that the polynomial *overfits* the data. Indeed, all model selection methods that are used in practice either implicitly or explicitly choose a tradeoff between goodness-of-fit and

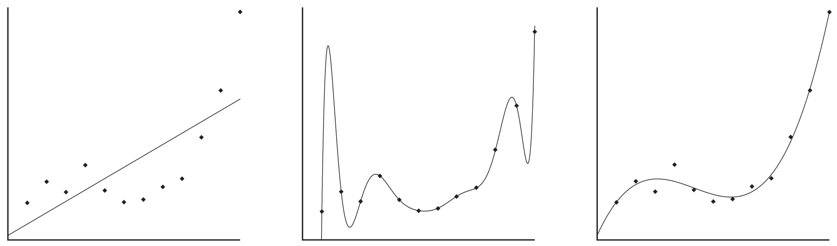

Figure 1.1 A simple, complex and tradeoff (third-degree) polynomial.

complexity of the models involved. In practice, such tradeoffs lead to much better predictions of test data than one would get by adopting the 'simplest' (one degree) or most "complex"[3] ($n-1$-degree) polynomial. MDL provides one particular means of achieving such a tradeoff.

It will be useful to make a precise distinction between "model" and "hypothesis":

Model vs. Hypothesis
We use the phrase *point hypothesis* to refer to a *single* probability distribution or function. An example is the polynomial $5x^2 + 4x + 3$. A point hypothesis is also known as a "simple hypothesis" in the statistical literature.
We use the word *model* to refer to a family (set) of probability distributions or functions with the same functional form. An example is the set of all second-degree polynomials. A model is also known as a "composite hypothesis" in the statistical literature.
We use *hypothesis* as a generic term, referring to both point hypotheses and models.

In our terminology, the problem described in Example 1.2 is a "hypothesis selection problem" if we are interested in selecting both the degree of a polynomial and the corresponding parameters; it is a "model selection problem" if we are mainly interested in selecting the degree.

To apply MDL to polynomial or other types of hypothesis and model selection, we have to make precise the somewhat vague insight "learning may be viewed as data compression." This can be done in various ways. In this section, we concentrate on the earliest and simplest implementation of the idea. This is the so-called *two-part code* version of MDL, see Figure 1.2.

Crude[4], Two-Part Version of MDL principle (Informally Stated)
Let $\mathcal{H}^{(1)}, \mathcal{H}^{(2)}, \ldots$ be a list of candidate models (e.g., $\mathcal{H}^{(k)}$ is the set of kth-degree polynomials), each containing a set of point hypotheses (e.g., individual polynomials). The best point hypothesis $H \in \mathcal{H}^{(1)} \cup \mathcal{H}^{(2)} \cup \ldots$ to explain the data D is the one which minimizes the sum $L(H) + L(D|H)$, where

- $L(H)$ is the length, in bits, of the description of the hypothesis; and
- $L(D|H)$ is the length, in bits, of the description of the data when encoded with the help of the hypothesis.

The best *model* to explain D is the smallest model containing the selected H.

Figure 1.2 The two-part MDL principle: first, crude implementation of the MDL ideas.

Example 1.3 (Polynomials, cont.) In our previous example, the candidate hypotheses were polynomials. We can describe a polynomial by describing its coefficients in a certain precision (number of bits per parameter). Thus, the higher the degree of a polynomial or the precision, the more[5] bits we need to describe it and the more 'complex' it becomes. A description of the data 'with the help of' a hypothesis means that the better the hypothesis fits the data, the shorter the description will be. A hypothesis that fits the data well gives us a lot of *information* about the data. Such information can always be used to compress the data (Chapter 2, Section 2.1). Intuitively, this is because we only have to code the *errors* the hypothesis makes on the data rather than the full data. In our polynomial example, the better a polynomial H fits D, the fewer bits we need to encode the discrepancies between the actual y-values y_i and the predicted y-values $H(x_i)$. We can typically find a very complex point hypothesis (large $L(H)$) with a very good fit (small $L(D|H)$). We can also typically find a very simple point hypothesis (small $L(H)$) with a rather bad fit (large $L(D|H)$). The sum of the two description lengths will be minimized at a hypothesis that is quite (but not too) "simple," with a good (but not perfect) fit.

1.4 Crude and Refined MDL

Crude MDL picks the H minimizing the sum $L(H) + L(D|H)$. To make this procedure well-defined, we need to agree on precise definitions for the codes (description methods) giving rise to lengths $L(D|H)$ and $L(H)$. We now discuss these codes in more detail. We will see that the definition of $L(H)$ is problematic, indicating that we somehow need to "refine" our crude MDL principle.

Definition of $L(D|H)$ Consider a two-part code as described above, and assume for the time being that all H under consideration define probability distributions. If H is a polynomial, we can turn it into a distribution by making the additional assumption that the Y-values are given by $Y = H(X) + Z$, where Z is a normally distributed noise term.

For each H we need to define a code with length $L(\cdot \mid H)$ such that $L(D|H)$ can be interpreted as "the code length of D when encoded with the help of H." It turns out that for probabilistic hypotheses, there is only one reasonable choice for this code. It is the so-called *Shannon-Fano code*, satisfying, for all data sequences D, $L(D|H) = -\log P(D|H)$, where $P(D|H)$ is the probability mass or density of D according to H – such a code always exists; see Chapter 2, Section 2.1.

Definition of $L(H)$: A Problem for Crude MDL It is more problematic to find a good code for hypotheses H. Some authors have simply used 'intuitively reasonable' codes in the past, but this is not satisfactory: since the description length $L(H)$ of any fixed point hypothesis H can be very large under one code, but quite short under another, our procedure is in danger of becoming arbitrary.

Instead, we need some additional principle for designing a code for $\mathcal{H}$. In the first publications on MDL [Rissanen 1978, 1983], it was advocated to choose some sort of *minimax code* for $\mathcal{H}$, minimizing, in some precisely defined sense, the shortest worst-case total description length $L(H) + L(D|H)$, where the worst case is over all possible data sequences. Thus, the MDL principle is employed at a "metalevel" to choose a code for H. However, this code requires a cumbersome discretization of the model space $\mathcal{H}$, which is not always feasible in practice. Alternatively, Barron [1985] encoded H by the shortest computer program that, when input D, computes $P(D|H)$. While it can be shown that this leads to similar code lengths, it is computationally problematic. Later, Rissanen [1984] realized that these problems could be sidestepped by using a *one-part* rather than a *two-part code*. This development culminated in 1996 in a completely precise prescription of MDL for many, but certainly not all, practical situations [Rissanen 1996]. We call this modern version of MDL *refined MDL*:

Refined MDL In refined MDL, we associate a code for encoding D *not with a single* $H \in \mathcal{H}$, but with the full model $\mathcal{H}$. Thus, given model $\mathcal{H}$, we encode data not in two parts but we design a single *one-part code* with lengths $\bar{L}(D|\mathcal{H})$. This code is designed such that *whenever there is a member of (parameter in)* $\mathcal{H}$ *that fits the data well, in the sense that* $L(D \mid H)$ *is small, then the code length* $\bar{L}(D|\mathcal{H})$ *will also be small.* Codes with this property are called *universal codes* in the information-theoretic literature [Barron, Rissanen, and Yu 1998]. Among all such universal codes, we pick the one that is *minimax optimal* in a sense made precise in Chapter 2, Section 2.4. For example, the set $\mathcal{H}^{(3)}$ of third-degree polynomials is associated with a code with lengths $\bar{L}(\cdot \mid \mathcal{H}^{(3)})$ such that, the better the data D are fit by the best-fitting third-degree polynomial, the shorter the code length $\bar{L}(D \mid \mathcal{H})$. $\bar{L}(D \mid \mathcal{H})$ is called the *stochastic complexity* of the data given the model.

Parametric Complexity The second fundamental concept of refined MDL is the *parametric complexity* of a parametric model $\mathcal{H}$ which we denote by $\mathbf{COMP}(\mathcal{H})$. This is a measure of the 'richness' of model $\mathcal{H}$, indicating its ability to fit random data. This complexity is related to the degrees of freedom in $\mathcal{H}$, but also to the geometric structure of $\mathcal{H}$; see Example 1.4. To see how it relates to stochastic complexity, let, for given data D, $\hat{H}$ denote the distribution in $\mathcal{H}$ which maximizes the probability, and hence minimizes the code length $L(D \mid \hat{H})$ of D. It turns out that

$$\text{Stochastic complexity of } D \text{ given } \mathcal{H} = L(D \mid \hat{H}) + \mathbf{COMP}(\mathcal{H}).$$

Refined MDL model selection between two parametric models (such as the models of first- and second-degree polynomials) now proceeds by selecting the model such that the stochastic complexity of the given data D is smallest. Although we used a one-part code to encode data, refined MDL model selection still involves a tradeoff between two terms: a goodness-of-fit term $L(D \mid \hat{H})$ and a complexity term $\mathbf{COMP}(\mathcal{H})$. However, because we do not explicitly encode hypotheses H anymore, there is no arbitrariness anymore. The resulting procedure can be interpreted in several different ways, some of which provide us with rationales for MDL beyond the pure coding interpretation (see Chapter 2, Sections 2.5.1–2.5.4):

1. **Counting/differential geometric interpretation.** The parametric complexity of a model is the logarithm of the number of *essentially different*, *distinguishable* point hypotheses within the model.

2. **Two-part code interpretation.** For large samples, the stochastic complexity can be interpreted as a two-part code length of the data after all, where hypotheses H are encoded with a special code that works by first discretizing the model space $\mathcal{H}$ into a set of "maximally distinguishable hypotheses," and then assigning equal code length to each of these.

3. **Bayesian interpretation.** In many cases, refined MDL model selection coincides with Bayes factor model selection based on a *noninformative prior* such as *Jeffreys' prior* [Bernardo and Smith 1994].

4. **Prequential interpretation.** Refined MDL model selection can be interpreted as selecting the model with the best predictive performance when sequentially predicting *unseen* test data, in the sense described in Chapter 2, Section 2.5.4. This makes it an instance of Dawid's [1984] *prequential* model validation and also relates it to *cross-validation* methods.

Refined MDL allows us to compare models of different functional form. It even accounts for the phenomenon that different models with the same number of parameters may not be equally "complex":

Example 1.4 Consider two models from psychophysics describing the relationship between physical dimensions (e.g., light intensity) and their psychological counterparts (e.g., brightness) [Myung, Balasubramanian, and Pitt 2000]: $y = ax^b + Z$ (Stevens's model) and $y = a \ln(x + b) + Z$ (Fechner's model) where Z is a normally distributed noise term. Both models have two free parameters; nevertheless, it turns out that in a sense, Stevens's model is more *flexible* or *complex* than Fechner's. Roughly speaking, this means there are a lot more data patterns that can be *explained* by Stevens's model than can be explained by Fechner's model. Myung and co-workers [2000] generated many samples of size 4 from Fechner's model, using some fixed parameter values. They then fitted both models to each sample. In 67% of the trials, Stevens's model fitted the data better than Fechner's, even though the latter generated the data. Indeed, in refined MDL, the 'complexity' associated with Stevens's model is much larger than the complexity associated with Fechner's, and if both models fit the data equally well, MDL will prefer Fechner's model.

Summarizing, refined MDL removes the arbitrary aspect of crude, two-part code MDL and associates parametric models with an inherent 'complexity' that does not depend on any particular description method for hypotheses. We should, however, warn the reader that we only discussed a special, simple situation in which we compared a finite number of parametric models that satisfy certain regularity conditions. Whenever the models do not satisfy these conditions, or if we compare an infinite number of models, then the refined ideas have to be extended. We then obtain a "general" refined MDL principle, which employs a combination of one-part and two-part codes.

1.5 The MDL Philosophy

The first central MDL idea is that every regularity in data may be used to compress those data; the second central idea is that learning can be equated with finding regularities in data. Whereas the first part is relatively straightforward, the second part of the idea implies that *methods for learning from data must have a clear interpretation independent of whether any of the models under consideration is "true" or not.* Quoting Rissanen [1989], the main originator of MDL:

'We never want to make the false assumption that the observed data actually were generated by a distribution of some kind, say Gaussian, and then go on to analyze the consequences and make further deductions. Our deductions may be entertaining but quite irrelevant to the task at hand, namely, to learn useful properties from the data.'
- Jorma Rissanen, 1989

Based on such ideas, Rissanen has developed a radical philosophy of learning and statistical inference that is considerably different from the ideas underlying mainstream statistics, both frequentist and Bayesian. We now describe this philosophy in more detail:

1. Regularity as Compression. According to Rissanen, the goal of inductive inference should be to 'squeeze out as much regularity as possible' from the given data. The main task for statistical inference is to distill the meaningful information present in the data, that is, to separate structure (interpreted as the regularity, the 'meaningful information') from noise (interpreted as the 'accidental information'). For the three sequences of Example 1.2, this would amount to the following: the first sequence would be considered as entirely regular and "noiseless." The second sequence would be considered as entirely random — all information in the sequence is accidental, there is no structure present. In the third sequence, the structural part would (roughly) be the pattern that 4 times as many 0s than 1s occur; given this regularity, the description of exactly which of all sequences with four times as many 0s than 1s occurs is the accidental information.

2. Models as Languages. Rissanen interprets models (sets of hypotheses) as nothing more than languages for describing useful properties of the data — a model $\mathcal{H}$ is *identified* with its corresponding universal code $\bar{L}(\cdot \mid \mathcal{H})$. Different individual hypotheses within the models express different regularities in the data, and may simply be regarded as *statistics*, that is, summaries of certain regularities in the data. *These regularities are present and meaningful independently of whether some $H^* \in \mathcal{H}$ is the "true state of nature" or not.* Suppose that the model $\mathcal{H}$ under consideration is probabilistic. In traditional theories, one typically assumes that some $P^* \in \mathcal{H}$ generates the data, and then 'noise' is defined as a random quantity relative to this P^*. In the MDL view 'noise' is defined relative to the model $\mathcal{H}$ as the residual number of bits needed to encode the data once the model $\mathcal{H}$ is given. Thus, noise is *not* a random variable: it is a function only of the chosen model and the *actually observed data*. Indeed, there is no place for a "true distribution" or a "true state of nature" in this view — there are only models and data. To bring out the difference to the ordinary statistical viewpoint, consider the phrase 'these experimental data are quite noisy.' According to a traditional interpretation, such a statement means that the data were generated by a distribution with high variance. According to the MDL philosophy, such a phrase means only that the data are not compressible with the currently hypothesized model — as a matter of principle, it can *never* be ruled out that there exists a different model under which the data are very compressible (not noisy) after all!

3. We Have Only the Data. Many (but not all[6]) other methods of inductive inference are based on the idea that there exists some "true state of nature," typically a distribution assumed to lie in some model $\mathcal{H}$. The methods are then designed as a means to identify or approximate this state of nature based on as little data as possible. According to Rissanen,[7] such methods are fundamentally flawed. The main reason is that the methods are designed under the assumption that the true state of nature is in the assumed model $\mathcal{H}$, which is often not the case. Therefore, *such methods only admit a clear interpretation under assumptions that are typically violated in practice.* Many cherished statistical methods are designed in this way — we mention hypothesis testing, minimum-variance unbiased estimation, several non-parametric methods, and even some forms of Bayesian inference — see Example 2.22. In contrast, MDL has a clear interpretation which *depends only on the data*, and not on the assumption of any underlying "state of nature."

Example 1.5 (Models That Are Wrong, Yet Useful) Even though the models under consideration are often wrong, they can nevertheless be very *useful*. Examples are the successful 'naive Bayes' model for spam filtering, hidden Markov models for speech recognition (is speech a stationary ergodic process? probably not), and the use of linear models in econometrics and psychology. Since these models are evidently wrong, it seems strange to base inferences on them using methods that are designed under the assumption that they contain the true distribution. To be fair, we should add that domains such as spam filtering and speech recognition are not what the fathers of modern statistics had in mind when they designed their procedures – they were usually thinking about much simpler domains, where the assumption that some distribution $P^* \in \mathcal{H}$ is "true" may not be so unreasonable.

4. MDL and Consistency. Let $\mathcal{H}$ be a probabilistic model, such that each $P \in \mathcal{H}$ is a probability distribution. Roughly, a statistical procedure is called *consistent* relative to $\mathcal{H}$ if, for all $P^* \in \mathcal{H}$, the following holds: suppose data are distributed according to P^*. Then given enough data, the learning method will learn a good approximation of P^* with high probability. Many traditional statistical methods have been designed with consistency in mind (Chapter 2, Section 2.2).

The fact that in MDL, we do not assume a true distribution may suggest that we do not care about statistical consistency. But this is not the case: we would still like our statistical method to be such that in the *idealized* case, where one of the distributions in one of the models under consideration actually generates the data, our method is able to identify this distribution, given enough data. If even in the idealized special case where a 'truth' exists within our models, the method fails to learn it, then we certainly cannot trust it to do something reasonable in the more general case, where there may not be a "true distribution" underlying the data at all. So: consistency *is* important in the MDL philosophy, but it is used *as a sanity check (for a method that has been developed without making distributional assumptions) rather than as a design principle.*

In fact, mere consistency is not sufficient. We would like our method to converge to the imagined true P^* *fast*, based on as small a sample as possible. Two-part code MDL with 'clever' codes achieves good rates of convergence in this sense (Barron and Cover [1991], complemented by Zhang [2004], show that in many situations, the rates are *minimax optimal*). The same seems to be true for refined one-part code MDL [Barron et al. 1998], although there is at least one surprising exception where inference based on the normalized maximum likelihood (NML) and Bayesian universal model behaves abnormally — see Csiszár and Shields [2000] for the details.

Summarizing this section, the MDL philosophy is quite agnostic about whether any of the models under consideration is "true", or whether something like a "true distribution" even exists. Nevertheless, it has been suggested [Webb 1996; Domingos 1999] that MDL embodies a naive belief that "simple models are a priori more likely to be true than complex models." Below we explain why such claims are mistaken.

1.6 MDL and Occam's Razor

When two models fit the data equally well, MDL will choose the one that is the "simplest" in the sense that it allows for a shorter description of the data. As such, it implements a precise form of Occam's razor – *even though as more and more data become available, the model selected by MDL may become more and more 'complex'!* Occam's razor is sometimes criticized for being either (1) arbitrary or (2) false [Webb 1996; Domingos 1999]. Do these criticisms apply to MDL as well?

"1. Occam's Razor (and MDL) Is Arbitrary" Because "description length" is a syntactic notion it may seem that MDL selects an arbitrary model: different codes would have led to different description lengths, and therefore, to different models. By changing the encoding method, we can make 'complex' things 'simple' and vice versa. This overlooks the fact we are not allowed to use just any code we like! 'Refined' MDL tells us to use a specific code, independent of any specific parameterization of the model, leading to a notion of complexity that can also be interpreted without any reference to 'description lengths' (see also Chapter 2, Section 2.9.1).

"2. Occam's Razor Is False" It is often claimed that Occam's razor is false — we often try to model real-world situations that are arbitrarily complex, so why should we favor simple models? In the words of Webb [1996], "What good are simple models of a complex world?" [8]

The short answer is: even if the true data-generating machinery is very complex, it may be a good strategy to prefer simple models for small sample sizes. Thus, MDL (and the corresponding form of Occam's razor) is a *strategy* for inferring models from data ("choose simple models at small sample sizes"), not a statement about how the world works ("simple models are more likely to be true") — indeed, a strategy cannot be true or false; it is "clever" or "stupid." And the strategy of preferring simpler models is clever even if the data-generating process is highly complex, as illustrated by the following example:

Example 1.6 ("Infinitely" Complex Sources) Suppose that data are subject to the law $Y = g(X) + Z$ where g is some continuous function and Z is some noise term with mean 0. If g is not a polynomial, but X only takes values in a finite interval, say $[-1, 1]$, we may still approximate g arbitrarily well by taking polynomials of higher and higher degree. For example, let $g(x) = \exp(x)$. Then, if we use MDL to learn a polynomial for data $D = ((x_1, y_1), \ldots, (x_n, y_n))$, the degree of the polynomial $\ddot{f}^{(n)}$ selected by MDL at sample size n will increase with n, and with high probability $\ddot{f}^{(n)}$ converges to $g(x) = \exp(x)$ in the sense that $\max_{x \in [-1,1]} |\ddot{f}^{(n)}(x) - g(x)| \rightarrow 0$. Of course, if we had better prior knowledge about the problem we could have tried to learn g using a model class $\mathcal{M}$ containing the function $y = \exp(x)$. But in general, both our imagination and our computational resources are limited, and we may be forced to use imperfect models.

If, based on a small sample, we choose the best-fitting polynomial $\hat{f}$ within the set of *all* polynomials, then, even though $\hat{f}$ will fit the data very well, it is likely to be quite unrelated to the "true" g, and $\hat{f}$ may lead to disastrous predictions of future data. The reason is that, for small samples, the set of all polynomials is very large compared to the set of possible data patterns that we might have observed. Therefore, any particular data pattern can only give us very limited information about which high-degree polynomial best approximates g. On the other hand, if we choose the best-fitting $\hat{f}^{\circ}$ in some much smaller set such as the set of second-degree polynomials, then it is highly probable that the prediction quality (mean squared error) of $\hat{f}^{\circ}$ on future data is about the same as its mean squared error on the data we observed: the size (complexity) of the contemplated model is relatively small compared to the set of possible data patterns that we might have observed.

Therefore, the particular pattern that we do observe gives us a lot of information on what second-degree polynomial best approximates g.

Thus, (a) $\hat{f}^{\circ}$ typically leads to better predictions of future data than $\hat{f}$; and (b) unlike $\hat{f}$, $\hat{f}^{\circ}$ is *reliable* in that it gives a correct impression of how good it will predict future data *even if the "true" g is 'infinitely' complex*. This idea does not just appear in MDL, but is also the basis of Vapnik's [1998] structural risk minimization approach and many standard statistical methods for nonparametric inference. In such approaches one acknowledges that the data-generating machinery can be infinitely complex (e.g., not describable by a finite-degree polynomial). Nevertheless, it is still a good strategy to approximate it by simple hypotheses (low-degree polynomials) as long as the sample size is small. Summarizing:

The Inherent Difference between Under- and Overfitting
If we choose an overly simple model for our data, then the best-fitting point hypothesis within the model is likely to be almost the best predictor, within the simple model, of future data coming from the same source. If we overfit (choose a very complex model) and there is noise in our data, then, *even if the complex model contains the "true" point hypothesis*, the best-fitting point hypothesis within the model is likely to lead to very bad predictions of future data coming from the same source.

This statement is very imprecise and is meant more to convey the general idea than to be completely true. As will become clear in Chapter 2, Section 2.9.1, it becomes provably true if we use MDL's measure of model complexity; we measure prediction quality by logarithmic loss; and we assume that one of the distributions in $\mathcal{H}$ actually generates the data.

1.7 History

The MDL principle has mainly been developed by Jorma Rissanen in a series of papers starting with [Rissanen 1978]. It has its roots in the theory of *Kolmogorov* or *algorithmic* complexity [Li and Vitányi 1997], developed in the 1960s by Solomonoff [1964], Kolmogorov [1965], and Chaitin [1966, 1969]. Among these authors, Solomonoff (a former student of the famous philosopher of science, Rudolf Carnap) was explicitly interested in inductive inference. The 1964 paper contains explicit suggestions on how the underlying ideas could be made practical, thereby foreshadowing some of the later work on two-part MDL. Although Rissanen was not aware of Solomonoff's work at the time, Kolmogorov's [1965] paper did serve as an inspiration for Rissanen's [1978] development of MDL.

Another important inspiration for Rissanen was Akaike's [1973] information criterion (AIC) method for model selection, essentially the first model selection method based on information-theoretic ideas. Even though Rissanen was inspired by AIC, both the actual method and the underlying philosophy are substantially different from MDL.

MDL is much more closely related to the *minimum message length (MML) principle*, developed by Wallace and his co-workers in a series of papers starting with the groundbreaking [Wallace and Boulton 1968]; other milestones were [Wallace and Boulton 1975] and [Wallace and Freeman 1987]. Remarkably, Wallace developed his ideas without being aware of the notion of Kolmogorov complexity. Although Rissanen became aware of Wallace's work before the publication of [Rissanen 1978], he developed his ideas mostly independently, being influenced rather by Akaike and Kolmogorov. Indeed, despite the close resemblance of both methods in practice, the underlying philosophy is quite different (Chapter 2, Section 2.8).

The first publications on MDL only mention two-part codes. Important progress was made by Rissanen [1984], in which prequential codes are employed for the first time and [Rissanen 1987], introducing the Bayesian mixture codes into MDL. This led to the development of the notion of stochastic complexity as the shortest code length of the data given a model [Rissanen 1986, 1987]. However, the connection to Shtarkov's *normalized maximum likelihood code* was not made until 1996, and this prevented the full development of the notion of "parametric complexity." In the meantime, in his impressive Ph.D. thesis, Barron [1985] showed how a specific version of the two-part code criterion has excellent frequentist statistical consistency properties. This was extended by Barron and Cover [1991] who achieved a breakthrough for two-part codes: they gave clear prescriptions on how to design codes for hypotheses, relating codes with good minimax code length properties to rates of convergence in statistical consistency theorems. Some of the ideas of Rissanen [1987] and Barron and Cover [1991] were, as it were, unified when Rissanen [1996] introduced a new definition of stochastic complexity based on the *normalized maximum likelihood (NML) code* (Chapter 2, Section 2.4). The resulting theory was summarized for the first time by Barron and co-workers [1998], and is called 'refined MDL' in the present overview.

1.8 Summary and Outlook

We have discussed how regularity is related to data compression, and how MDL employs this connection by viewing learning in terms of data compression. One can make this precise in several ways; in *idealized* MDL one looks for the shortest program that generates the given data. This approach is not feasible in practice, and here we concern ourselves with *practical* MDL. Practical MDL comes in a crude version based on two-part codes and in a modern, more refined version based on the concept of *universal coding*. The basic ideas underlying all these approaches can be found in the boxes spread throughout the text.

These methods are mostly applied to model selection but can also be used for other problems of inductive inference. In contrast to most existing statistical methodology, they can be given a clear interpretation irrespective of whether or not there exists some "true" distribution generating data — inductive inference is seen as a search for regular properties in (interesting statistics of) the data, and there is no need to assume anything outside the model and the data. In contrast to what is sometimes thought, there is *no* implicit belief that 'simpler models are more likely to be true' — MDL does embody a preference for 'simple' models, but this is best seen as a strategy for inference that can be useful even if the environment is not simple at all.

In the next chapter, we make precise both the crude and the refined versions of practical MDL. For this, it is absolutely essential that the reader familiarizes him- or herself with two basic notions of coding and information theory: the relation between code length functions and probability distributions, and (for refined MDL), the idea of universal coding — a large part of Chapter 2 is devoted to these.

Notes

1. See Section 2.8.2, Example 2.22.
2. By this we mean that a universal Turing machine can be implemented in it [Li and Vitányi 1997].
3. Strictly speaking, in our context it is not very accurate to speak of "simple" or "complex" polynomials; instead we should call the *set* of first-degree polynomials "simple,' and the *set* of 100th-degree polynomials "complex."
4. The terminology 'crude MDL' is not standard. It is introduced here for pedagogical reasons, to make clear the importance of having a single, unified principle for designing codes. It should be noted that Rissanen's and Barron's early theoretical papers on MDL already contain such principles, albeit in a slightly different form than in their recent papers. Early practical applications [Quinlan and Rivest 1989; Grünwald 1996] often do use *ad hoc* two-part codes which really are 'crude' in the sense defined here.
5. See the previous endnote.
6. For example, cross-validation cannot easily be interpreted in such terms of 'a method hunting for the true distribution.'
7. My own views are somewhat milder in this respect, but this is not the place to discuss them.
8. Quoted with permission from *KDD Nuggets 96,28*, 1996.

References

Akaike, H. (1973). Information theory as an extension of the maximum likelihood principle. In B. N. Petrov and F. Csaki (Eds.), *Second International Symposium on Information Theory*, pp. 267–281. Budapest: Akademiai Kiado.

Barron, A.R. and T. Cover (1991). Minimum complexity density estimation. *IEEE Transactions on Information Theory*, *37*(4), 1034–1054.

Barron, A.R. (1985). *Logically Smooth Density Estimation.* Ph. D. thesis, Department of Electrical Engineering, Stanford University, Stanford, CA.

Barron, A.R., J. Rissanen, and B. Yu (1998). The Minimum Description Length Principle in coding and modeling. Special Commemorative Issue: Information Theory: 1948-1998. *IEEE Transactions on Information Theory*, *44*(6), 2743–2760.

Bernardo, J., and A. Smith (1994). *Bayesian theory.* New York: Wiley.

Chaitin, G. (1966). On the length of programs for computing finite binary sequences. *Journal of the ACM*, *13*, 547–569.

Chaitin, G. (1969). On the length of programs for computing finite binary sequences: Statistical considerations. *Journal of the ACM*, *16*, 145–159.

Csiszár, I., and P. Shields (2000). The consistency of the BIC Markov order estimator. *Annals of Statistics*, *28*, 1601–1619.

Dawid, A. (1984). Present position and potential developments: Some personal views, statistical theory, the prequential approach. *Journal of the Royal Statistical Society, Series A*, *147*(2), 278–292.

Domingos, P. (1999). The role of Occam's razor in knowledge discovery. *Data Mining and Knowledge Discovery*, *3*(4), 409–425.

Gács, P., J. Tromp, and P. Vitányi (2001). Algorithmic statistics. *IEEE Transactions on Information Theory*, *47*(6), 2464–2479.

Grünwald, P.D. (1996). A minimum description length approach to grammar inference. In G.Scheler, S. Wermter, and E. Riloff (Eds.), *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, Volume 1040 in Springer Lecture Notes in Artificial Intelligence, pp. 203–216. Berlin: Springer-Verlag

Kolmogorov, A. (1965). Three approaches to the quantitative definition of information. *Problems of Information Transmission*, *1*(1), 1–7.

Li, M., and P. Vitányi (1997). *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd edition. New York: Springer-Verlag.

Myung, I.J., V. Balasubramanian, and M.A. Pitt (2000). Counting probability distributions: Differential geometry and model selection. *Proceedings of the National Academy of Sciences USA*, *97*, 11170–11175.

Quinlan, J., and R. Rivest (1989). Inferring decision trees using the minimum description length principle. *Information and Computation*, *80*, 227–248.

Ripley, B. (1996). *Pattern Recognition and Neural Networks.* Cambridge, UK: Cambridge University Press.

Rissanen, J. (1978). Modeling by the shortest data description. *Automatica*, *14*, 465–471.

Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, *11*, 416–431.

Rissanen, J. (1984). Universal coding, information, prediction and estimation. *IEEE Transactions on Information Theory*, *30*, 629–636.

Rissanen, J. (1986). Stochastic complexity and modeling. *Annals of Statistics*, *14*, 1080–1100.

Rissanen, J. (1987). Stochastic complexity. *Journal of the Royal Statistical Society, Series B*, *49*, 223–239. Discussion: pp. 252–265.

Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry.* Singapore: World Scientific.

Rissanen, J. (1996). Fisher information and stochastic complexity. *IEEE Transactions on Information Theory*, *42*(1), 40–47.

Solomonoff, R. (1964). A formal theory of inductive inference, part 1 and part 2. *Information and Control*, *7*, 1–22, 224–254.

Solomonoff, R. (1978). Complexity-based induction systems: Comparisons and convergence theorems. *IEEE Transactions on Information Theory*, *24*, 422–432.

Vapnik, V. (1998). *Statistical Learning Theory.* New York: John Wiley.

Wallace, C., and D. Boulton (1968). An information measure for classification. *Computer Journal*, *11*, 185–195.

Wallace, C., and D. Boulton (1975). An invariant Bayes method for point estimation. *Classification Society Bulletin*, *3*(3), 11–34.

Wallace, C., and P. Freeman (1987). Estimation and inference by compact coding. *Journal of the Royal Statistical Society, Series B*, *49*, 240–251. Discussion: pp. 252–265.

Webb, G. (1996). Further experimental evidence against the utility of Occam's razor. *Journal of Artificial Intelligence Research*, *4*, 397–417.

Zhang, T. (2004). On the convergence of MDL density estimation. In *Proceedings of the Seventeenth Annual Conference on Computational Learning Theory (COLT' 04).* Berlin: Springer-Verlag.