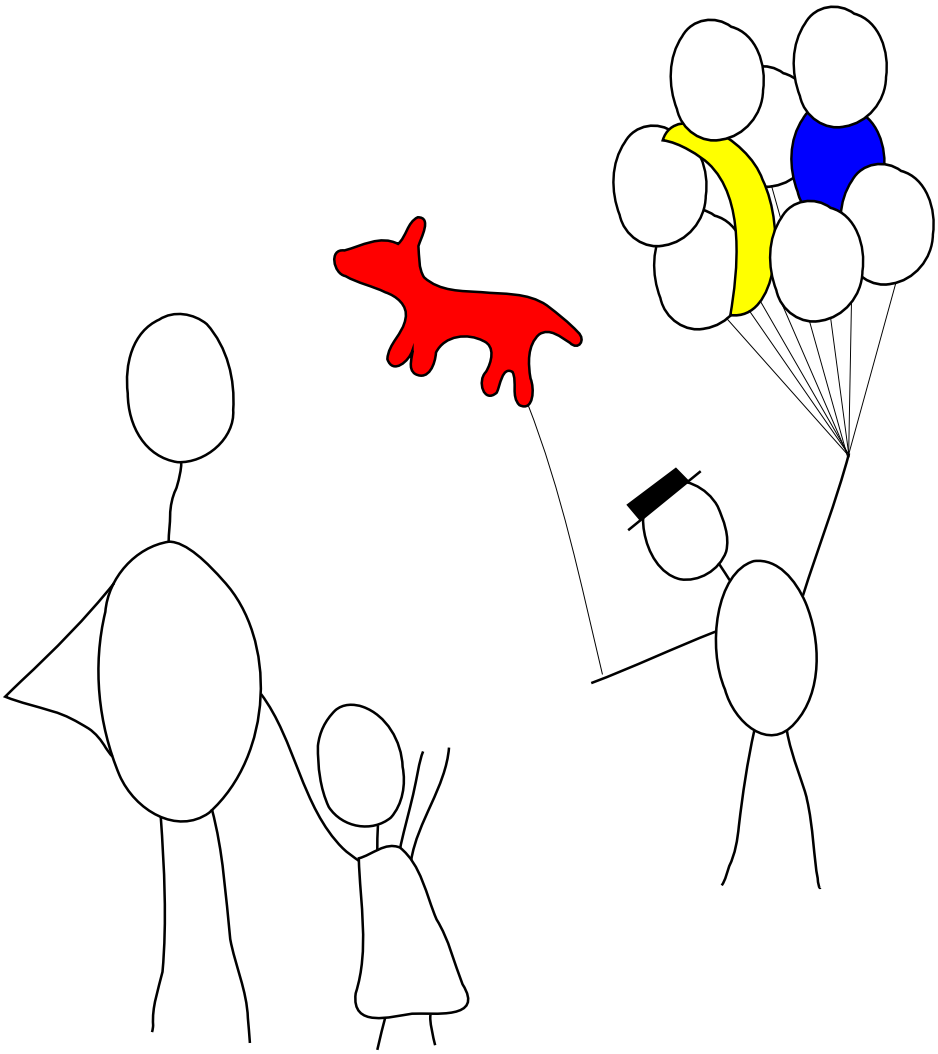
# Minimum Description Length Model Selection

## Problems and Extensions



Steven de Rooij

# Minimum Description Length Model Selection

## Problems and Extensions

ILLC Dissertation Series DS-2008-07

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

For further information about ILLC-publications, please contact

Institute for Logic, Language and Computation
Universiteit van Amsterdam
Plantage Muidergracht 24
1018 TV Amsterdam
phone: +31-20-525 6051
fax: +31-20-525 5206
e-mail: illc@science.uva.nl
homepage: http://www.illc.uva.nl/

# Minimum Description Length Model Selection

## Problems and Extensions

Academisch Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof.dr. D. C. van den Boom
ten overstaan van een door het college voor
promoties ingestelde commissie, in het openbaar
te verdedigen in de Agnietenkapel
op woensdag 10 september 2008, te 12.00 uur

door

Steven de Rooij

geboren te Amersfoort.

**Promotiecommissie:**

| | |
|---|---|
| Promotor: | prof.dr.ir. P.M.B. Vitányi |
| Co-promotor: | dr. P.D. Grünwald |
| Overige leden: | prof.dr. P.W. Adriaans |
| | prof.dr. A.P. Dawid |
| | prof.dr. C.A.J. Klaassen |
| | prof.dr.ir. R.J.H. Scha |
| | dr. M.W. van Someren |
| | prof.dr. V. Vovk |
| | dr.ir. F.M.J. Willems |

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The investigations were funded by the Netherlands Organization for Scientific Research (NWO), project 612.052.004 on Universal Learning, and were supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778.

Copyright © 2008 by Steven de Rooij

Cover inspired by Randall Munroe's xkcd webcomic (www.xkcd.org)
Printed and bound by PrintPartners Ipskamp (www.ppi.nl)
ISBN: 978-90-5776-181-2

Parts of this thesis are based on material contained in the following papers:

- *An Empirical Study of Minimum Description Length Model Selection with Infinite Parametric Complexity*
  Steven de Rooij and Peter Grünwald
  In: Journal of Mathematical Psychology, special issue on model selection. Vol. 50, pp. 180-190, 2006
  (Chapter 2)

- *Asymptotic Log-Loss of Prequential Maximum Likelihood Codes*
  Peter D. Grünwald and Steven de Rooij
  In: Proceedings of the 18th Annual Conference on Computational Learning Theory (COLT), pp. 652-667, June 2005
  (Chapter 3)

- *MDL Model Selection using the ML Plug-in Code*
  Steven de Rooij and Peter Grünwald
  In: Proceedings of the International Symposium on Information Theory (ISIT), 2005
  (Chapters 2 and 3)

- *Approximating Rate-Distortion Graphs of Individual Data: Experiments in Lossy Compression and Denoising*
  Steven de Rooij and Paul Vitányi
  Submitted to: IEEE Transactions on Computers
  (Chapter 6)

- *Catching up Faster in Bayesian Model Selection and Averaging*
  Tim van Erven, Peter D. Grünwald and Steven de Rooij
  In: Advances in Neural Information Processing Systems 20 (NIPS 2007)
  (Chapter 5)

- *Expert Automata for Efficient Tracking*
  W. Koolen and S. de Rooij
  In: Proceedings of the 21st Annual Conference on Computational Learning Theory (COLT) 2008
  (Chapter 4)

# Contents

# Acknowledgements

When I took prof. Vitányi's course on Kolmogorov complexity, I was delighted by his seemingly endless supply of anecdotes, opinions and wisdoms that ranged the whole spectrum from politically incorrect jokes to profound philosophical insights. At the time I was working on my master's thesis on data compression with Breanndán Ó Nualláin, which may have primed me for the course; at any rate, I did well enough and Paul Vitányi and Peter Grünwald (who taught a couple of guest lectures) singled me out as a suitable PhD candidate, which is how I got the job. This was a very lucky turn of events for me, because I was then, and am still, very much attracted to that strange mix of computer science, artificial intelligence, information theory and statistics that make up machine learning and learning theory.

While Paul's commonsense suggestions and remarks kept me steady, Peter has performed the job of PhD supervision to perfection: besides pointing me towards interesting research topics, teaching me to write better papers, and introducing me to several conferences, we also enjoyed many interesting discussions and conversations, as often about work as not, and he said the right things at the right times to keep my spirits up. He even helped me think about my future career and dropped my name here and there... In short, I wish to thank Paul and Peter for their excellent support.

Some time into my research I was joined by fellow PhD students Tim van Erven and Wouter Koolen. I know both of them from during college when I did some work as a lab assistant; Wouter and I in fact worked together on the Kolmogorov complexity course. The three of us are now good friends and we have had a lot of fun working together on the material that now makes up a large part of this thesis. Thanks to Wouter and Tim for all their help; hopefully we will keep working together at least occasionally in the future. Thanks also go to Daniel Navarro, who made my stay in Adelaide very enjoyable and with whom I still hope to work out the human psyche one day, and to my group members Rudi Cilibrasi, Łukasz Dębowski, Peter Harremoës and Alfonso Martinez. Regards also

to Harry Buhrman, and the people of his excellent quantum computing group.

In addition to my supervisors, I wish to thank the following people for making technical contributions to this thesis. Chapter 2 is inspired by the work of Aaron D. Lanterman. Thanks to Mark Herbster for a lively discussion and useful comments with respect to Chapter 4, and to Tim van Erven for his excellent suggestions. The ideas in Chapter 5 were inspired by a discussion with Yishay Mansour. We received technical help from Peter Harremoës and Wouter Koolen and very helpful insights from Andrew Barron.

Outside of work, I have been loved and supported by my wonderful girlfriend Nele Beyens. Without her and our son Pepijn, I might still have written a thesis, but I certainly would not have been as happy doing it! Also many thanks to my parents Anca and Piet and my brother Maarten, who probably do not have much of a clue what I have been doing all this time, but who have been appropriately impressed by it all.

Finally I wish to thank my friends Magiel Bruntink, Mark Thompson, Joeri van Ruth and Nienke Valkhoff, Wouter Vanderstede and Lieselot Decalf, Jan de Vos, Daniel Wagenaar and his family, and Thijs Weststeijn and Marieke van den Doel.

# Chapter 1

# The MDL Principle for Model Selection

Suppose we consider a number of different hypotheses for some phenomenon. We have gathered some data that we want to use somehow to evaluate these hypotheses and decide which is the "best" one. In case that one of the hypotheses exactly describes the true mechanism that underlies the phenomenon, then that is the one we hope to find. While this may already be a hard problem, available hypotheses are often merely approximations in practice. In that case the goal is to select a hypothesis that is useful, in the sense that it provides insight in previous observations, and matches new observations well. Of course, we can immediately reject hypotheses that are inconsistent with new experimental data, but hypotheses often allow for some margin of error; as such they are never truly inconsistent but they can vary in the degree of success with which they predict new observations. A quantitative criterion is required to decide between competing hypotheses. The Minimum Description Length (MDL) principle is such a criterion [67, 71]. It is based on the intuition that, on the basis of a useful theory, it should be possible to *compress* the observations, i.e. to describe the data in full using fewer symbols than we would need using a literal description. According to the MDL principle, the more we can compress a given set of data, the more we have learned about it. The MDL approach to inference requires that all hypotheses are formally specified in the form of *codes*. A code is a function that maps possible outcomes to binary sequences; thus the length of the encoded representation of the data can be expressed in bits. We can encode the data by first specifying the hypothesis to be used, and then specifying the data with the help of that hypothesis. Suppose that $L(H)$ is a function that specifies how many bits we need to identify a hypothesis $H$ in a countable set of considered hypotheses $\mathcal{H}$. Furthermore let $L_H(D)$ denote how many bits we need to specify the data $D$ using the code associated with hypothesis $H$. The MDL principle now tells us to select that hypothesis $H$ for which the total description length of the data, i.e. the length of the description of the hypothesis $L(H)$, plus the length of the description of data using that hypothesis $L_H(D)$, is shortest. The minimum

total description length as a function of the data is denoted $L_{\text{mdl}}$:

$$H_{\text{mdl}} := \arg\min_{H \in \mathcal{H}} \Big( L(H) + L_H(D) \Big), \tag{1.1}$$

$$L_{\text{mdl}}(D) := \min_{H \in \mathcal{H}} \Big( L(H) + L_H(D) \Big). \tag{1.2}$$

(In case that there are several $H$ that minimise (1.1), we take any one among those with smallest $L(H)$.) Intuitively, the term $L(H)$ represents the complexity of the hypothesis while $L_H(D)$ represents how well the hypothesis is able to describe the data, often referred to as the goodness of fit. By minimising the sum of these two components, MDL implements a tradeoff between complexity and goodness of fit. Also note that the selected hypothesis only depends on the *lengths* of the used code words, and not on the binary sequences that make up the code words themselves. This will make things a lot easier later on.

By its preference for short descriptions, MDL implements a heuristic that is widely used in science as well as in learning in general. This is the *principle of parsimony*, also often referred to as Occam's razor. A parsimonious learning strategy is sometimes adopted on the basis of a belief that the true state of nature is more likely to be simple than to be complex. We prefer to argue in the opposite direction: only if the truth is simple, or at least has a simple approximation, we stand a chance of learning its mechanics based on a limited data set [39]. In general, we try to avoid assumptions about the truth as much as possible, and focus on effective *strategies* for learning instead. That said, a number of issues still need to be addressed if we want to arrive at a practical theory for learning:

1. What code should be used to identify the hypotheses?
2. What codes for the data are suitable as formal representations of the hypotheses?
3. After answering the previous two questions, to what extent can the MDL criterion actually be shown to identify a "useful" hypothesis, and what do we actually mean by "useful"?

These questions, which lie at the heart of MDL research, are illustrated in the following example.

**Example 1** (Language Learning)**.** Ray Solomonoff, one of the founding fathers of the concept of Kolmogorov complexity (to be discussed in Section 1.1.3), introduced the problem of language inference based on only positive examples as an example application of his new theory [84]; it also serves as a good example for MDL inference.

Suppose we want to automatically infer a language based on a list of valid example sentences $D$. We assume that the vocabulary $\Sigma$, which contains all words

that occur in $D$, is background knowledge: we already know the words of the language, but we want to learn from valid sentences how the words may be ordered. For the set of our hypotheses $\mathcal{H}$ we take all context-free grammars (introduced as "phrase structure grammars" by Chomsky in [19]) that use only words from $\Sigma$ and and additional set $N$ of nonterminal symbols. $N$ contains a special symbol called the *starting symbol*. A grammar is a set of production rules, each of which maps a nonterminal symbol to a (possibly empty) sequence consisting of both other nonterminals and words from $\Sigma$. A sentence is grammatical if it can be produced from the starting symbol by iteratively applying a production rule to one of the matching nonterminal symbols.

We need to define the relevant code length functions: $L(H)$ for the specification of the grammar $H \in \mathcal{H}$, and $L_H(D)$ for the specification of the example sentences with the help of that grammar. In this example we use very simple code length functions; later in this introduction, after describing in more detail what properties good codes should have, we return to language inference with a more in-depth discussion.

We use *uniform codes* in the definitions of $L(H)$ and $L_H(D)$. A uniform code on a finite set $A$ assigns binary code words of equal length to all elements of the set. Since there are $2^l$ binary sequences of length $l$, a uniform code on $A$ must have code words of length at least $\lceil \log |A| \rceil$. (Throughout this thesis, $\lceil \cdot \rceil$ denotes rounding up to the nearest integer, and log denotes binary logarithm. Such notation is listed on page 201.) To establish a baseline, we first use uniform codes to calculate how many bits we need to encode the data literally, without the help of any grammar. Namely, we can specify every word in $D$ with a uniform code on $\Sigma \cup \diamond$, where $\diamond$ is a special symbol used to mark the end of a sentence. This way we need $|D| \lceil \log(|\Sigma| + 1) \rceil$ bits to encode the data. We are looking for grammars $H$ which allow us to compress the data beyond this baseline value.

We first specify $L(H)$ as follows. For each production rule of $H$, we use $\lceil \log(|N \cup \{\diamond\}|) \rceil$ bits to uniformly encode the initial nonterminal symbol, and $\lceil \log(|N \cup \{\diamond\} \cup \Sigma|) \rceil$ bits for each of the other symbols. The $\diamond$ symbol signals the end of each rule; two consecutive $\diamond$s signal the end of the entire grammar. If the grammar $H$ has $r$ rules, and the summed length of the replacement sequences is $s$, then we can calculate that the number of bits we need to encode the entire grammar is at most

$$(s + r) \lceil \log(|N| + |\Sigma| + 1) \rceil + (r + 1) \lceil \log(|N| + 1) \rceil. \tag{1.3}$$

If a context-free grammar $H$ is correct, i.e. all sentences in $D$ are grammatical according to $H$, then its corresponding code $L_H$ can help to compress the data, because it does not need to reserve code words for any sentences that are ungrammatical according to $H$. In this example we simply encode all words in $D$ in sequence, each time using a uniform code on the set of words that *could* occur next according to the grammar. Again, the set of possible words is augmented

with a $\diamond$ symbol to mark the end of each sentence and to mark the end of the data.

Now we consider two very simple correct grammars, both of which only need one nonterminal symbol S. The "promiscuous" grammar (terminology due to Solomonoff [84]) has rules $\mathrm{S} \to \mathrm{S}\ \mathrm{S}$ and $\mathrm{S} \to \sigma$ for each word $\sigma \in \Sigma$. This grammar generates *any* sequence of words as a valid sentence. It is very short: we have $r = 1 + |\Sigma|$ and $s = 2 + |\Sigma|$ so the number of bits $L(H_1)$ required to encode the grammar essentially depends only on the size of the dictionary and not on the amount of available data $D$. On the other hand, according to $H_1$ all words in the dictionary are allowed in all positions, so $L_{H_1}(D)$ requires as much as $\lceil \log(|\Sigma| + 1) \rceil$ bits for every word in $D$, which is equal to the baseline. Thus this grammar does not enable us to compress the data.

Second, we consider the "ad-hoc" grammar $H_2$. This grammar consists of a production rule $\mathrm{S} \to\ d$ for each sentence $d \in D$. Thus according to $H_2$, a sentence is only grammatical if it matches one of the examples in $D$ exactly. Since this severely restricts the number of possible words that can appear at any given position in a sentence given the previous words, this grammar allows for very efficient representation of the data: $L_{H_2}(D)$ is small. However, in this case $L(H_2)$ is at least as large as the baseline, since in this case the data $D$ appear literally in $H_2$!

Both grammars are clearly useless: the first does not describe any structure in the data at all and is said to *underfit* the data. In the second grammar random features of the data (in this case, the selection of valid sentences that happen to be in $D$) are treated as structural information; this grammar is said to *overfit* the data. Consequently, for both grammars $H \in \{H_1, H_2\}$, we can say that we do not compress the data at all, since in both cases the *total* code length $L(H) + L_H(D)$ exceeds the baseline. In contrast, by selecting a grammar $H_{\mathrm{mdl}}$ that allows for the greatest total compression as per (1.1), we avoid either extreme, thus implementing a natural tradeoff between underfitting and overfitting. Note that *finding* this grammar $H_{\mathrm{mdl}}$ may be a quite difficult search problem, but the algorithmic aspects of finding the best hypothesis in an enormous hypothesis space is mostly outside the scope of this thesis: only in Chapter 6 we address this search problem explicitly.

## The Road Ahead

In the remainder of this chapter we describe the basics of MDL inference, with special attention to *model selection*. Model selection is an instance of MDL inference where the hypotheses are sets of probability distributions. For example, suppose that Alice claims that the number of hairs on people's heads is Poisson distributed, while Bob claims that it is in fact geometrically distributed. After gathering data (which would involve counting lots of hair on many heads in this case), we may use MDL model selection to determine whose model is a better

description of the truth.

In the course of the following introduction we will come across a number of puzzling issues in the details of MDL model selection that warrant further investigation; these topics, which are summarised in Section 1.4, constitute the subjects of the later chapters of this thesis.

## 1.1 Encoding Hypotheses

We return to the three questions of the previous section. The intuition behind the MDL principle was that "useful" hypotheses should help compress the data. For now, we will consider hypotheses that are "useful to compress the data" to be "useful" in general. We postpone further discussion of the the third question: what this means exactly, to Section 1.3. What remains is the task to find out which hypotheses are useful, by using them to compress the data.

Given many hypotheses, we could just test them one at a time on the available data until we find one that happens to allow for substantial compression. However, if we were to adopt such a methodology in practice, results would vary from reasonable to extremely bad. The reason is that among so many hypotheses, there needs only be *one* that, by sheer force of luck, allows for compression of the data. This is the phenomenon of overfitting again, which we mentioned in Example 1. To avoid such pitfalls, we required that a *single* code $L_{\mathrm{mdl}}$ is proposed on the basis of all available hypotheses. The code has two parts: the first part, with length function $L(H)$, identifies a hypothesis to be used to encode the data, while the second part, with length function $L_H(D)$, describes the data using the code associated with that hypothesis.

The next section is concerned with the definition of $L_H(D)$, but for the time being we will assume that we have already represented our hypotheses in the form of codes, and we will discuss some of the properties a good choice for $L(H)$ should have. Technically, throughout this thesis we use only length functions that correspond to prefix codes; we will explain what this means in Section 1.2.1.

Consider the case where the best candidate hypothesis, i.e. the hypothesis $\hat{H} = \arg\min_{H \in \mathcal{H}} L_H(D)$, achieves substantial compression. It would be a pity if we did not discover the usefulness of $\hat{H}$ because we chose a code word with unnecessarily long length $L(\hat{H})$. The *regret* we incur on the data quantifies how bad this "detection overhead" can be, by comparing the total code length $L_{\mathrm{mdl}}(D)$ to the code length achieved by the best hypothesis $L_{\hat{H}}(D)$. A general definition, which also applies if $\hat{H}$ is undefined, is the following: the *regret* of a code $L$ on data $D$ with respect to a set of alternative codes $\mathcal{M}$ is

$$\mathcal{R}(L, \mathcal{M}, D) \;:=\; L(D) - \inf_{L' \in \mathcal{M}} L'(D). \tag{1.4}$$

The reasoning is now that, since we do not want to make a priori assumptions as to the process that generates the data, the code for the hypotheses $L(H)$ must be

chosen such that the regret $\mathcal{R}(L_{\text{mdl}}, \{L_H : H \in \mathcal{H}\}, D)$ is small, *whatever data we observe*. This ensures that whenever $\mathcal{H}$ contains a useful hypothesis that allows for significant compression of the data, we are able to detect this because $L_{\text{mdl}}$ compresses the data as well.

**Example 2.** Suppose that $\mathcal{H}$ is finite. Let $L$ be a uniform code that maps every hypothesis to a binary sequence of length $l = \lceil \log_2 |\mathcal{H}| \rceil$. The regret incurred by this uniform code is always exactly $l$, whatever data we observe. This is the best possible guarantee: all other length functions $L'$ on $\mathcal{H}$ incur a strictly larger regret for at least one possible outcome (unless $\mathcal{H}$ contains useless hypotheses $H$ which have $L(H) + L_H(D) > L_{\text{mdl}}(D)$ for all possible $D$.) In other words, the uniform code minimises the *worst-case regret* $\max_D \mathcal{R}L, \mathcal{M}, D$ among all code length functions $L$. (We discuss the exact conditions we impose on code length functions in Section 1.2.1.)

Thus, if we consider a finite number of hypotheses we can use MDL with a uniform code $L(H)$. Since in this case the $L(H)$ term is the same for all hypotheses, it cancels and we find $H_{\text{mdl}} = \hat{H}$ in this case. What we have gained from this possibly anticlimactic analysis is the following *sanity check*: since we equated learning with compression, we should not trust $\hat{H}$ to exhibit good performance on future data unless we were able to compress the data using $L_{\text{mdl}}$.

### 1.1.1 Luckiness

There are many codes $L$ on $\mathcal{H}$ that guarantee small regret, so the next task is to decide which we should pick. As it turns out, given any particular code $L$, it is possible to select a special subset of hypotheses $\mathcal{H}' \subset \mathcal{H}$ and modify the code such that the code lengths for these hypotheses are especially small, at the small cost of increasing the code lengths for the other hypotheses by a negligible amount. This can be desirable, because *if* a hypothesis in $\mathcal{H}'$ turns out to be useful, then we are lucky, and we can achieve superior compression. On the other hand, if all hypotheses in the special subset are poor, then we have not lost much. Thus, while a suitable code must always have small regret, there is quite a lot of freedom to favour such small subsets of special hypotheses.

Examples of this so-called *luckiness principle* are found throughout the MDL literature, although they usually remain implicit, possibly because it makes MDL inference appear subjective. Only recently has the luckiness principle been identified as an important part of code design. The concept of luckiness is introduced to the MDL literature in [39]; [6] uses a similar concept but not under the same name. We take the stance that the luckiness principle introduces only a mild form of subjectivity, because it cannot substantially harm inference performance. Paradoxically, it can only really be harmful *not* to apply the luckiness principle, because that could cause us to miss out on some good opportunities for learning!

**Example 3.** To illustrate the luckiness principle, we return to the grammar

learning of Example 1. To keep things simple we will not modify the code for the data $L_H(D)$ defined there; we will only reconsider $L(H)$ that intuitively seemed a reasonable code for the specification of grammars. Note that $L(H)$ is in fact a luckiness code: it assigns significantly shorter code lengths to shorter grammars. What would happen if we tried to avoid this "subjective" property by optimising the worst-case regret without considering luckiness?

To keep things simple, we reduce the hypothesis space to finite size by considering only context-free grammars with at most $|N| = 20$ nonterminals, $|\Sigma| = 500$ terminals, $r = 100$ rules and replacement sequences summing to a total length of $s = 2,000$. Using (1.3) we can calculate the luckiness code length for such a grammar as at most $\lceil 2100 \log(521) + 101 \log(21) \rceil = 19397$ bits.

Now we will investigate what happens if we use a *uniform* code on all possible grammars $\mathcal{H}'$ of up to that size instead. One may or may not want to verify that

$$|\mathcal{H}'| \;=\; \sum_{r=1}^{100} |N|^r \sum_{s=0}^{2000} (|N| + |\Sigma|)^s \binom{s+r-1}{s}.$$

Calculation on the computer reveals that $\lceil \log(|\mathcal{H}'|) \rceil = 19048$ bits. Thus, we compress the data 331 bits better than the code that we used before. While this shows that there is room for improvement of the luckiness code, the difference is actually not very large compared to the total code length. On the other hand, with the uniform code we *always* need 19048 bits to encode the grammar, even when the grammar is very short! Suppose that the best grammar uses only $r = 10$ and $s = 100$, then the luckiness code requires only $\lceil 110 \log(521) + 11 \log(21) \rceil = 1042$ bits to describe that grammar and therefore outcompresses the uniform code by 18006 bits: in that case we learn a lot more with the luckiness code.

Now that we have computed the minimal worst-case regret we have a target for improvement of the luckiness code. A simple way to combine the advantages of both codes is to define a third code that uses one additional bit to specify whether to use the luckiness code or the uniform code on the data.

Note that we do not mean to imply that the hypotheses which get special luckiness treatment are necessarily more likely to be true than any other hypothesis. Rather, luckiness codes can be interpreted as saying that this or that special subset *might* be important, in which case we should like to know about it!

### 1.1.2 Infinitely Many Hypotheses

When $\mathcal{H}$ is countably infinite, there can be no upper bound on the lengths of the code words used to identify the hypotheses. Since any hypothesis *might* turn out to be the best one in the end, the worst-case regret is typically infinite in this case. In order to retain MDL as a useful inference procedure, we are forced to embrace the luckiness principle. A good way to do this is to order the hypotheses such that

$H_1$ is luckier than $H_2$, $H_2$ is luckier than $H_3$, and so on. We then need to consider only codes $L$ for which the code lengths increase monotonically with the index of the hypothesis. This immediately gives a lower bound on the code lengths $L(H_n)$ for $n = 1, \ldots$, because the nonincreasing code that has the shortest code word for hypothesis $n$ is uniform on the set $\{H_1, \ldots, H_n\}$ (and unable to express hypotheses with higher indices). Thus $L(H_n) \geq \log |\{H_1, \ldots, H_n\}| = \log n$. It is possible to define codes $L$ with $L(H_n) = \log n + O(\log \log n)$, i.e. not much larger than this ideal. Rissanen describes one such code, called the "universal code for the integers", in [68] (where the restriction to monotonically increasing code word lengths is not interpreted as an application of the luckiness principle as we do here); in Example 4 we describe some codes for the natural numbers that are convenient in practical applications.

### 1.1.3 Ideal MDL

In MDL hypothesis selection as described above, it is perfectly well conceivable that the data generating process has a very simple structure, which nevertheless remains undetected because it is not represented by any of the considered hypotheses. For example, we may use hypothesis selection to determine the best Markov chain order for data which reads "110010010000111111...", never suspecting that this is really just the beginning of the binary expansion of the number $\pi$. In "ideal MDL" such blind spots are avoided, by interpreting *any* code length function $L_H$ that can be implemented in the form of a computer program as the formal representation of a hypothesis $H$. Fix a universal prefix Turing machine $U$, which can be interpreted as a language in which computer programs are expressed. The result of running program $T$ on $U$ with input $D$ is denoted $U(T, D)$. The *Kolmogorov complexity* of a hypothesis $H$, denoted $K(H)$, is the length of the shortest program $T_H$ that implements $L_H$, i.e. $U(T_H, D) = L_H(D)$ for all binary sequences $D$. Now, the hypothesis $H$ can be encoded by literally listing the program $T_H$, so that the code length of the hypotheses becomes the Kolmogorov complexity. For a thorough introduction to Kolmogorov complexity, see [60].

In the literature the term "ideal MDL" is used for a number of approaches to model selection based on Kolmogorov complexity; for more information on the version described here, refer to [8, 1]. To summarise, our version of ideal MDL tells us to pick

$$\min_{H \in \mathcal{H}} K(H) + L_H(D), \tag{1.5}$$

which is (1.1), except that now $\mathcal{H}$ is the set of *all* hypotheses represented by computable length functions, and $L(H) = K(H)$. (Note that $L_H(D) \approx K(D|H)$ iff $D$ is $P(\cdot|H)$-random.)

In order for this code to be in agreement with MDL philosophy as described above, we have to check whether or not it has small regret. It is also natural to wonder whether or not it somehow applies the luckiness principle. The following

property of Kolmogorov complexity is relevant for the answer to both questions. Let $\mathcal{H}$ be a countable set of hypotheses with computable corresponding length functions. Then for all computable length functions $L$ on $\mathcal{H}$, we have

$$\exists c > 0 : \forall H \in \mathcal{H} : K(H) \leq L(H) + c. \tag{1.6}$$

Roughly speaking, this means that ideal MDL is ideal in two respects: first, the set of considered hypotheses is expanded to include all computable hypotheses, so that any computable concept is learned given enough data. Second, it matches all other length functions up to a constant, including all length functions with small regret as well as length functions with *any* clever application of the luckiness principle.

On the other hand, performance guarantees such as (1.6) are not very specific, as the constant overhead may be so large that it completely dwarfs the length of the data. To avoid this, we would need to specify a particular universal Turing machine $U$, and give specific upper bounds on the values that $c$ can take for important choices of $\mathcal{H}$ and $L$. While there is some work on such a concrete definition of Kolmogorov complexity for individual objects [91], there are as yet no concrete performance guarantees for ideal MDL or other forms of algorithmic inference.

The more fundamental reason why ideal MDL is not practical, is that Kolmogorov complexity is uncomputable. Thus it should be appreciated as a theoretical ideal that can serve as an inspiration for the development of methods that can be applied in practice.

## 1.2 Using Models to Encode the Data

On page 2 we asked how the codes that formally represent the hypotheses should be constructed. Often many different interpretations are possible and it is a matter of judgement how exactly a hypothesis should be made precise. There is one important special case however, where the hypothesis is formulated in the form of a set of probability distributions. Statisticians call such a set a *model*. Possible models include the set of all normal distributions with any mean and variance, or the set of all third order Markov chains, and so on. The problem of model selection is central to this thesis, but before we can discuss how the codes to represent models are chosen, we have to discuss the close relationship between coding and probability theory.

### 1.2.1 Codes and Probability Distributions

We have introduced the MDL principle in terms of coding; here we will make precise what we actually mean by a code and what properties we require our codes to

have. We also make the connection to statistics by describing the correspondence between code length functions and probability distributions.

A *code* $C : \mathcal{X} \to \{0,1\}^*$ is an injective mapping from a countable source alphabet $\mathcal{X}$ to finite binary sequences called *code words*. We consider only *prefix codes*, that is, codes with the property that no code word is the prefix of another code word. This restriction ensures that the code is *uniquely decodable*, i.e. any concatenation of code words can be decoded into only one concatenation of source symbols. Furthermore, a prefix code has the practical advantage that no look-ahead is required for decoding, that is, given any concatenation $S$ of code words, the code word boundaries in any prefix of $S$ are determined by that prefix and do not depend on the remainder of $S$. Prefix codes are as efficient as other uniquely decodable codes; that is, for any uniquely decodable code with length function $L_C$ there is a prefix code $C'$ with $L_{C'}(x) \leq L_C(x)$ for all $x \in \mathcal{X}$, see [25, Chapter 5]. Since we never consider non-prefix codes, from now on, whenever we say "code", this should be taken to mean "prefix code".

Associated with a code $C$ is a *length function* $L : \mathcal{X} \to \mathbb{N}$, which maps each source symbol $x \in \mathcal{X}$ to the length of its code word $C(x)$.

Of course we want to use efficient codes, but there is a limit to how short code words can be made. For example, there is only one binary sequence of length zero, two binary sequences of length one, four of length three, and so on. The precise limit is expressed by the Kraft inequality:

**Lemma 1.2.1** (Kraft inequality)**.** *Let $\mathcal{X}$ be a countable source alphabet. A function $L : \mathcal{X} \to \mathbb{N}$ is the length function of a prefix code on $\mathcal{X}$ if and only if:*

$$\sum_{x \in \mathcal{X}} 2^{-L(x)} \quad \leq \quad 1.$$

*Proof.* See for instance [25, page 82]. □

If the inequality is strict, then the code is called *defective*, otherwise it is called *complete*. (The term "defective" is usually reserved for probability distributions, but we apply it to code length functions as well.)

Let $C$ be any prefix code on $\mathcal{X}$ with length function $L$, and define

$$\forall x \in \mathcal{X} : \qquad P(x) := 2^{-L(x)}. \tag{1.7}$$

Since $P(x)$ is always positive and sums to at most one, it can be interpreted as a probability mass function that defines a distribution corresponding to $C$. This mass function and distribution are called *complete* or *defective* if and only if $C$ is.

Vice versa, given a distribution $P$, according to the Kraft inequality there must be a prefix code $L$ satisfying

$$\forall x \in \mathcal{X} : \qquad L(x) := \left\lceil -\log P(x) \right\rceil. \tag{1.8}$$

To further clarify the relationship between $P$ and its corresponding $L$, define the *entropy* of distribution $P$ on a countable outcome space $\mathcal{X}$ by

$$H(P) := \sum_{x \in \mathcal{X}} -P(x) \log P(x). \tag{1.9}$$

(This $H$ should not be confused with the $H$ used for hypotheses.) According to Shannon's noiseless coding theorem [79], the mean number of bits used to encode outcomes from $P$ using the most efficient code is at least equal to the entropy, i.e. for all length functions $L'$ of prefix codes, we have $E_P[L'(X)] \geq H(P)$. The expected code length using the $L$ from (1.8) stays within one bit of entropy. This bit is a consequence of the requirement that code lengths have to be integers.

Note that apart from rounding, (1.7) and (1.8) describe a one-to-one correspondence between probability distributions and code length functions that are most efficient for those distributions.

Technically, probability distributions are usually more convenient to work with than code length functions, because their usage does not involve rounding. But conceptually, code length functions are often more intuitive objects than probability distributions. A practical reason for this is that the probability of the observations typically decreases exponentially as the number of observations increases, and such small numbers are hard to handle psychologically, or even to plot in a graph. Code lengths typically grow linearly with the number of observations, and have an analogy in the real world, namely the effort required to remember all the obtained information, or the money spent on storage equipment.

A second disadvantage of probability theory is the philosophical controversy regarding the interpretation of probability [76, 12]: for example, the frequentist school of thought holds that probability only carries any meaning in the context of a repeatable experiment. The frequency of a particular observation converges as more observations are gathered; this limiting value is then called the probability. According to the Bayesian school on the other hand, a probability can also express a degree of belief in a certain proposition, even outside the context of a repeatable experiment. In both cases, specifying the probability of an event usually means making a statement about (ones beliefs about) the true state of nature. This is problematic, because people of necessity often work with very crude probabilistic models, which everybody agrees have no real truth to them. In such cases, probabilities are used to represent beliefs/knowledge which one a priori knows to be false! Using code lengths allows us to avoid this philosophical can of worms, because code lengths do not carry any such associations. Rather, codes are usually judged on the basis of their performance in practice: a good code achieves a short code length on data that we observe in practice, whether it is based on valid assumptions about the truth or not. We embrace this "engineering criterion" because we feel that inference procedures should be motivated solely on the basis of guarantees that we can give with respect to their performance in practice.

In order to get the best of both worlds: the technical elegance of probability theory combined with the conceptual clarity of coding theory, we generalise the concept of coding such that code lengths are no longer necessarily integers. While the length functions associated with such "ideal codes" are really just alternative representations of probability mass functions, and probability theory is used under the hood, we will nonetheless call negative logarithms of probabilities "code lengths" to aid our intuition and avoid confusion. Since this difference between an ideal code length and the length using a real code is at most one bit, this generalisation should not require too large a stretch of the imagination. In applications where it is important to actually encode the data, rather than just compute its code length, there is a practical technique called arithmetic coding [REF] which can usually be applied to achieve the ideal code length to within a few bits; this is outside the scope of this thesis because for MDL inference we only have to compute code lengths, and not actual codes.

**Example 4.** In Section 1.1.2 we remarked that a good code for the natural numbers always achieves code length close to $\log n$. Consider the distribution $W(n) = f(n) - f(n+1)$. If $f : \mathbb{N} \to \mathbb{R}$ is a decreasing function with $f(1) = 1$ and $1/f \to 0$, then $W$ is an easy to use probability mass function that can be used as a prior distribution on the natural numbers. For $f(n) = 1/n$ we get code lengths $-\log W(n) = \log(n(n+1)) \approx 2\log n$, which, depending on the application, may be small enough. Even more efficient for high $n$ are $f(n) = n^{-\alpha}$ for some $0 < \alpha < 1$ or $f(n) = 1/\log(n+1)$.

### 1.2.2 Sequential Observations

In practice, model selection is often used in a *sequential* setting, where rather than observing one outcome from some space $\mathcal{X}$, we keep making more and more observations $x_1, x_2, \ldots$. Here we discuss how probability theory can be applied to such a scenario. There are in fact three different ways to do it; while the first is perhaps easiest to understand, the other two will eventually be used as well so we will outline all of them here. This section can be skipped by readers who find themselves unconfused by the technicalities of sequential probability. We consider only countable outcome spaces for simplicity; the required definitions for uncountable outcome spaces are given later as they are needed.

A first approach is to define a *sequence* of distributions $P^1, P^2, \ldots$ Each $P^n$ is then a distribution on the $n$-fold product space $\mathcal{X}^n$ which is used in reasoning about outcome sequences of length $n$. Such a sequence of distributions defines a random process if it is *consistent*, which means that for all $n \in \mathbb{N}$, all sequences $x^n \in \mathcal{X}^n$, the mass function satisfies $P^n(x^n) = \sum_{x \in \mathcal{X}} P^{n+1}(x^n, x)$. With some abuse of notation, we often use the same symbol for the mass function and the distribution; we will sometimes explicitly mention which we mean and sometimes consider it clear from context.

A mathematically more elegant solution is to define a distribution on infinite sequence of outcomes $\mathcal{X}^\infty$, called a *probability measure*. A sequence of outcomes $x^n \in \mathcal{X}^n$ can then be interpreted as an event, namely the set of all infinite sequences of which $x^n$ is a prefix; the marginal distributions on prefixes of length $0, 1, \ldots$ are automatically consistent. With proper definitions, a probability measure on $\mathcal{X}^\infty$ uniquely defines a random process and vice versa. Throughout this thesis, wherever we talk about "distributions" on sequences without further qualification we actually mean random processes, or equivalently, suitably defined probability measures.

A random process or probability measure defines a *predictive distribution* $P(X_{n+1}|x^n) = P(x^n; X_{n+1})/P(x^n)$ on the next outcome given a sequence of previous observations. A third solution, introduced by Dawid [26, 29, 28] is to turn this around and start by specifying the predictive distribution, which in turn defines a random process. Dawid defines a *prequential forecasting system* (PFS) as an algorithm which, when input any initial sequence of outcomes $x^n$, issues a probability distribution on the next outcome. The total "prequential" probability of a sequence is then given by the *chain rule*:

$$\begin{aligned} P(x^n) &= P(x^1) \cdot \frac{P(x^2)}{P(x^1)} \cdots \frac{P(x^n)}{P(x^{n-1})} \\ &= P(x_1) \cdot P(x_2|x^1) \cdots P(x_n|x^{n-1}). \end{aligned} \tag{1.10}$$

This framework is simple to understand and actually somewhat more powerful than that of probability measures. Namely, if a random process assigns probability zero to a sequence of observations $x^n$, then the predictive distribution $P(X_{n+1}|x^n) = P(x^n; X_{n+1})/P(x^n) = 0/0$ is undefined. This is not the case for a PFS which *starts* by defining the predictive distributions. This property of FPSs becomes important in Chapters 4 and 5, because there we consider predictions made by *experts*, who may well assign probability zero to an event that turns out to occur!

### 1.2.3 Model Selection and Universal Coding

Now that we have identified the link between codes and probability distributions it is time to address the question which codes we should choose to represent hypotheses. In case that the hypothesis is given in terms of a probability distribution or code from the outset, no more work needs to be done and we can proceed straight to doing MDL model selection as described at the start of this chapter. Here we consider instead the case where the hypothesis is given in the form of a *model* $\mathcal{M} = \{P_\theta : \theta \in \Theta\}$, which is a set of random processes or probability measures parameterised by a vector $\theta$ from some set $\Theta$ of allowed parameter vectors, called the *parameter space*. For example, a model could be the set of all fourth order Markov chains, or the set of all normal distributions. Now it is no longer immediately clear which single code should represent the hypothesis.

To motivate the code $L_{\mathcal{M}}$ that represents such a model, we apply the same reasoning as we used in Section 1.1. Namely, the code should guarantee a small regret, but is allowed to favour some small subsets of the model on the basis of the luckiness principle. To make this idea more precise, we first introduce a function $\hat{\theta} : \mathcal{X}^* \to \Theta$ called the *maximum likelihood estimator*, which is defined by:

$$\hat{\theta}(x^n) := \arg\max_{\theta \in \Theta} P_\theta(x^n). \tag{1.11}$$

(The following can be extended to the case where arg max is undefined, but we omit the details here.) Obviously, the maximum likelihood element of the model also minimises the code length. We prefer to abbreviate $\hat{\theta} = \hat{\theta}(x^n)$, if the sequence of outcomes $x^n$ is clear from context.

**Definition 1.2.2.** Let $\mathcal{M} := \{L_\theta : \theta \in \Theta\}$ be a (countable or uncountably infinite) model with parameter space $\Theta$. Let $f : \Theta \times \mathbb{N} \to [0, \infty)$ be some function. A code $L$ is called *$f$-universal* for a model $\mathcal{M}$ if, for all $n \in \mathbb{N}$, all $x^n \in \mathcal{X}^n$, we have

$$\mathcal{R}(L, \mathcal{M}, x^n) \leq f(\hat{\theta}, n).$$

This is quite different from the standard definition of universality [25], because it is formulated in terms of individual sequences rather than expectation. Also it is a very general formulation, with a function $f$ that needs further specification. Generality is needed because, as it turns out, different degrees of universality are possible in different circumstances. This definition allows us to express easily what we expect a code to live up to in all these cases.

For finite $\mathcal{M}$, a uniform code similar to the one described in Example 2 achieves $f$-universality for $f(\hat{\theta}, n) = \log |\mathcal{M}|$. Of course we incur a small overhead on top of this if we decide to use luckiness codes.

For countably infinite $\mathcal{M}$, the regret cannot be bounded by a single constant, but we can avoid dependence on the sample size. Namely, if we introduce a distribution $W$ on the parameter set, we can achieve $f$-universality for $f(\hat{\theta}, n) = -\log W(\hat{\theta})$ by using a Bayesian or two-part code (these are explained in subsequent sections).

Finally for uncountably infinite $\mathcal{M}$ it is often impossible to obtain a regret bound that does not depend on $n$. For parametric models however it is often possible to achieve $f$-universality for $f(\hat{\theta}, n) = \frac{k}{2} \log \frac{n}{2\pi} + g(\hat{\theta})$, where $k$ is the number of parameters of the model and $g : \Theta \to [0, \infty)$ is some continuous function of the maximum likelihood parameter. Examples include the Poisson model, which can be parameterised by the mean of the distribution so $k = 1$, and the normal model, which can be parameterised by mean and variance so $k = 2$. Thus, for parametric uncountable models a logarithmic dependence on the sample size is the norm.

We have now seen that in MDL model selection, universal codes are used on two levels: on one level, each model is represented by a universal code. Then

another universal code (a two-part code, see below) is used to combine them into a single code for the data.

We describe the four most common ways to construct universal codes; and we illustrate each code by applying it to the model of Bernoulli distributions $\mathcal{M} = \{P_\theta : \theta \in [0,1]\}$. This is the "biased coin" model, which contains the possible distributions on heads and tails when a coin is flipped. The distributions are parameterised by the bias of the coin: the probability that the coin lands heads. Thus, $\theta = \frac{1}{2}$ represents the distribution for an unbiased coin. The distributions in the model are extended to $n$ outcomes by taking the $n$-fold product distribution: $P_\theta(x^n) = P_\theta(x_1) \cdots P_\theta(x_n)$.

### Two-Part Codes

In Section 1.1 we defined a code $L_{\text{mdl}}$ that we now understand to be universal for the model $\{L_H : H \in \mathcal{H}\}$. This kind of universal code is called a two-part code, because it consists first of a specification of an element of the model, and second of a specification of the data using that element. Two-part codes may be defective: this occurs if multiple code words represent the same source symbol. In that case one must ensure that the encoding function is well-defined by specifying exactly which representation is associated with each source word $D$. Since we are only concerned with code *lengths* however, it suffices to adopt the convention that we always use one of the shortest representations.

**Example 5.** We define a two-part code for the Bernoulli model. In the first part of the code we specify a parameter value, which requires some discretisation since the parameter space is uncountable. However, as the maximum likelihood parameter for the Bernoulli model is just the observed frequency of heads, at a sample size of $n$ we know that the ML parameter is in the set $\{0/n, 1/n, \ldots, n/n\}$. We discretise by restricting the parameter space to this set. A uniform code uses $L(\theta) = \log(n+1)$ bits to identify an element of this set. For the data we can use the code corresponding to $\theta$. The total code length is minimised by ML distribution, so that we know that the regret is always exactly $\log(n+1)$; by using slightly cleverer discretisation we can bring this regret down even more such that it grows as $\frac{1}{2}\log n$, which, as we said, is usually achievable for uncountable single parameter models.

### The Bayesian Universal Distribution

Let $\mathcal{M} = \{P_\theta : \theta \in \Theta\}$ be a countable model; it is convenient to use mass functions rather than codes as elements of the model here. Now define a distribution with mass function $W$ on the parameter space $\Theta$. This distribution is called the *prior distribution* in the literature as it is often interpreted as a representation of a priori beliefs as to which of the hypotheses in $\mathcal{M}$ represents the "true state of the world". More in line with the philosophy outlined above would be the interpretation that

$W$ is a *code* which should be chosen for practical reasons to optimise inference performance. At any rate, the next step is to define a *joint distribution* $P$ on $\mathcal{X}^n \times \Theta$ by $P(x^n, \theta) = P_\theta(x^n)W(\theta)$. In this joint space, each outcome comprises a particular state of the world and an observed outcome.

In the field of Bayesian statistics, inference is always based on this joint distribution. We may, for example, calculate $P(\mathbf{x^n})$, where $\mathbf{x^n} = \{(x^n, \theta) : \theta \in \Theta\}$ denotes the event in the joint space that a particular sequence of outcomes $x^n$ is observed. Second, we can calculate how we should update our beliefs about the state of the world after observing outcome $x^n$. Let $\boldsymbol{\theta} = \{(x^n, \theta) : x^n \in \mathcal{X}^n\}$ denote the event in the joint space that $\theta$ is true. Then we have:

$$P(\boldsymbol{\theta}|\mathbf{x^n}) = \frac{P(\mathbf{x^n} \cap \boldsymbol{\theta})}{P(\mathbf{x^n})} = \frac{P(\mathbf{x^n}|\boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathbf{x^n})}. \tag{1.12}$$

This result is called *Bayes' rule*; its importance to inference stems from the idea that it can be used to update beliefs about the world $W$ on the basis of new observations $x^n$. The conditional distribution on the hypotheses is called the *posterior distribution*; with considerable abuse of notation it is often denoted $W(\theta \mid \mathbf{x^n})$.

Note that the marginal distribution satisfies:

$$-\log P(\mathbf{x^n}) = -\log \sum_{\theta \in \Theta} P_\theta(\mathbf{x^n}) w(\theta) \leq -\log P_{\hat{\theta}}(\mathbf{x^n}) - \log W(\hat{\theta}),$$

where $\hat{\theta}$ is the maximum likelihood estimator, the element of $\mathcal{M}$ that minimises the code length for the data. Thus we find that if we use a Bayesian universal code, we obtain a code length less than or equal to the code length we would have obtained with the two-part code with $L(\theta) = -\log W(\theta)$. Since we already found that two-part codes are universal, we can now conclude that Bayesian codes are at least as universal. On the flip side, the sum involved in calculating the Bayesian marginal distribution can be hard to evaluate in practice.

**Example 5** (continued)**.** Our definitions readily generalise to uncountable models with $\Theta \subseteq \mathbb{R}^k$, with the prior distribution given by a density $w$ on $\Theta$. Rather than giving explicit definitions we revisit our running example.

We construct a Bayesian universal code for the Bernoulli model. For simplicity we use a uniform prior density, $w(\theta) = 1$. Let $h$ and $t = n - h$ denote the number of heads and tails in $x^n$, respectively. Now we can calculate the Bayes marginal likelihood of the data:

$$P_{\text{bayes}}(x^n) = \int_0^1 P_\theta(x^n) \cdot 1 \ \mathrm{d}\theta = \int_0^1 \theta^h (1-\theta)^t \, \mathrm{d}\theta = \frac{h!\, t!}{(n+1)!}.$$

Using Stirling's approximation of the factorial function, we find that the corresponding code length $-\log P_{\text{bayes}}(x^n)$ equals $-\log P_{\hat{\theta}}(x^n) + \frac{1}{2}\log n + O(1)$. Thus we find roughly the same regret as for a well-designed two-part code.

### Prequential Universal Distributions

An algorithm that, given a sequence of previous observations $x^n$, issues a probability distribution on the next outcome $P(X_{n+1}|x^n)$, is called a *prequential forecasting system* (PFS) [26, 29, 28]. A PFS defines a random process by the chain rule (1.10). Vice versa, for any random process $P$ on $\mathcal{X}^\infty$, we can calculate the conditional distribution on the next outcome $P(X_{n+1} \mid X^n = x^n) = P(x^n; X_{n+1})/P(x^n)$, provided that $P(x^n)$ is positive. This so-called *predictive distribution* of a random process defines a PFS. We give two important prequential universal distributions here.

First, we may take a Bayesian approach and define a joint probability measure on $\mathcal{X}^\infty \times \Theta$ based on some prior distribution $W$. As before, this induces a marginal probability measure on $\mathcal{X}^\infty$ which in turn defines a PFS. In this way, the Bayesian universal distribution can be reinterpreted as a prequential forecasting system.

Second, since the Bayesian predictive distribution can be hard to compute it may be useful in practice to define a forecasting system that uses a simpler algorithm. Perhaps the simplest option is to predict an outcome $X_{n+1}$ using the maximum likelihood estimator for the previous outcomes $\hat{\theta}(x^n) = \arg\max_\theta P_\theta(x^n)$. We will use this approach in our running example.

**Example 5** (continued)**.** The ML estimator for the Bernoulli model parameterised by the probability of observing heads, equals the frequency of heads in the sample: $\hat{\theta} = h/n$, where $h$ denotes the number of heads in $x^n$ as before. We define a PFS through $P(X_{n+1}|x^n) := P_{\hat{\theta}}$. This PFS is ill-defined for the first outcome. Another impractical feature is that it assigns probability 0 to the event that the second outcome is different from the first. To address these problems, we slightly tweak the estimator: rather than $\hat{\theta}$ we use $\tilde{\theta} = (h+1)/(n+2)$.

Perhaps surprisingly, in this case the resulting PFS is equivalent to the Bayesian universal distribution approach we defined in the previous section: $P_{\tilde{\theta}}$ turns out to be the Bayesian predictive distribution for the Bernoulli model if a uniform prior density $w(\theta) = 1$ is used. In general, the distribution indexed by such a "tweaked" ML estimator may be quite different from the Bayesian predictive distribution.

Although the prequential ML code has been used successfully in practical inference problems, the model selection experiments in Chapter 2 show that usage of PFSs based on estimators such as the "tweaked" ML estimator from the example, leads to significantly worse results than those obtained for the other three universal codes. This is the subject of Chapter 3, where we show that in fact the prequential ML code does achieve the same asymptotic regret as the other universal codes, *provided* that the distribution that generates the data is an element of the model. Under misspecification the prequential ML code has different behaviour that had not been observed before.

### Normalised Maximum Likelihood

The last universal code we discuss is the one preferred in the MDL literature, because if we fix some sample size $n$ in advance, it provably minimises the worst-case regret. It turns out that the code minimising the worst-case regret must achieve equal regret for all possible outcomes $x^n$. In other words, the total code length must always be some constant longer than the code length achieved on the basis of the maximum likelihood estimator. This is precisely what the Normalised Maximum Likelihood (NML) distribution achieves:

$$P_{\mathrm{nml}}(x^n) := \frac{P_{\hat{\theta}(x^n)}(x^n)}{\sum_{y^n \in \mathcal{X}^n} P_{\hat{\theta}(y^n)}(y^n)}. \tag{1.13}$$

For all sequences $x^n$, the regret on the basis of this distribution is exactly equal to the logarithm of the denominator, called the *parametric complexity* of the model:

$$\inf_L \sup_{x^n} \mathcal{R}(L, \mathcal{M}, x^n) = \log \sum_{y^n \in \mathcal{X}^n} P_{\hat{\theta}(y^n)}(y^n) \tag{1.14}$$

Under some regularity conditions on the model it can be shown [72, 39] that there is a particular continuous function $g$ such that the parametric complexity is less than $\frac{k}{2} \log \frac{n}{2\pi} + g(\hat{\theta})$, as we required in Section 1.2.3. We return to our Bernoulli example.

**Example 5** (continued)**.** The parametric complexity (1.14) has exponentially many terms, but for the Bernoulli model the expression can be significantly simplified. Namely, we can group together all terms which have the same maximum likelihood estimator. Thus the minimal worst-case regret can be rewritten as follows:

$$\log \sum_{y^n \in \mathcal{X}^n} P_{\hat{\theta}(y^n)}(y^n) = \log \sum_{h=0}^{n} \binom{n}{h} \left(\frac{h}{n}\right)^h \left(\frac{n-h}{n}\right)^{n-h}. \tag{1.15}$$

This term has only linearly many terms and can usually be evaluated in practice. Approximation by Stirling's formula confirms that the asymptotic regret is $\frac{1}{2} \log n + O(1)$, the same as for the other universal distributions.

The NML distribution has a number of significant practical problems. First, it is often undefined, because for many models the numerator in (1.13) is infinite, even for such simple models as the model of all Poisson or geometric distributions. Second, $|\mathcal{X}^n|$ may well be extremely large, in which case it may be impossible to actually calculate the regret. In a sequential setting, the number of terms grows exponentially with the sample size so calculating the NML probability is hopeless except in special cases such as the Bernoulli model above, where mathematical trickery can be applied to get exact results (for the multinomial model, see [50].) Chapter 2 addresses the question of what can be done when NML is undefined or too difficult to compute.

## 1.3 Applications of MDL

While we have equated "learning" with "compressing the data" so far, in reality we often have a more specific goal in mind when we apply machine learning algorithms. This touches on the third question we asked on page 2: what do we mean by a "useful" hypothesis? In this section we will argue that MDL inference, which is designed to achieving compression of the data, is also suitable to some extent in settings with a different objective, so that, at least to some extent, MDL provides a reliable one-size-fits-all solution.

### 1.3.1 Truth Finding

We have described models as formal representations of hypotheses; truth finding is the process of determining which of the hypotheses is "true", in the sense that the corresponding model contains the data generating process. The goal is then to use the available data to identify this true model on the basis of as little data as possible.

Since the universal code for the true model achieves a code length not much larger than the code length of the best code in the model (it was designed to achieve small regret), and the best code in the model achieves code length at most as large as the data generating distribution, it seems reasonable to assume that, as more data are being gathered, a true model will eventually be selected by MDL. This intuition is confirmed in the form of the following consistency result, which is one of the pillars of MDL model selection. The result applies to Bayesian model selection as well.

**Theorem 1.3.1** (Model Selection Consistency)**.** *Let $\mathcal{H} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots\}$ be a countably infinite set of parametric models. For all $n \in \mathbb{Z}^+$, let $\mathcal{M}_n$ be 1-to-1 parameterised by $\Theta_n \subseteq \mathbb{R}^k$ for some $k \in \mathbb{N}$ and define a prior density $w_n$ on $\Theta_n$. Let $W(i)$ define a prior distribution on the model indices. We require for all integers $j > i > 0$ that with $w_j$-probability 1, a distribution drawn from $\Theta_j$ is mutually singular with all distributions in $\Theta_i$. Define the MDL model selection criterion based on Bayesian universal codes to represent the models:*

$$\delta(x^n) = \arg\min_i \left( -\log W(i) - \log \int_{\theta \in \Theta_i} P_\theta(x^n) w_i(\theta) \, \mathrm{d}\theta \right).$$

*Then for all $\delta^* \in \mathbb{Z}^+$, for all $\theta^* \in \Theta_{\delta^*}$, except for a subset of $\Theta_{\delta^*}$ of Lebesgue measure 0, it holds for $X_1, X_2, \ldots \sim P_{\theta^*}$ that*

$$\exists n_0 : \forall n \geq n_0 : \qquad P_{\theta^*}(\delta(X^n) = \delta^*) \;=\; 1.$$

*Proof.* Proofs of various versions of this theorem can be found in [5, 28, 4, 39] and others. □

The theorem uses Bayesian codes, but as conjectured in [39] and partially in Chapter 5, it can probably be extended to other universal codes such as NML. Essentially it expresses that *if* we are in the ideal situation where one of the models contains the true distribution, we are guaranteed that this model will be selected once we have accumulated sufficient data. This property of model selection criteria is called *consistency*. Unfortunately, this theorem says nothing about the more realistic scenario where the models are merely approximations of the true data generating process. If the true data generating process $P^*$ is not an element of any of the models, it is not known under what circumstances the model selection criterion $\delta$ selects the model that contains the distribution $P_{\hat{\theta}}$ minimising the Kullback-Leibler divergence $D(P^* \| P_{\hat{\theta}})$. On the other hand, some model selection criteria that are often used in practice (such as maximum likelihood model selection, or AIC [2]) are known *not* to be consistent, even in the restricted sense of Theorem 1.3.1. Therefore consistency of MDL and Bayesian model selection is reassuring.

### 1.3.2 Prediction

The second application of model selection is prediction. Here the models are used to construct a *prediction strategy*, which is essentially the same as a prequential forecasting system: an algorithm that issues a probability distribution on the next outcome given the previous outcomes. There are several ways to define a prediction strategy based on a set of models; we mostly consider what is perhaps the most straightforward method, namely to apply a model selection criterion such as MDL to the available data, and then predict according to some estimator that issues a distribution on the next outcome based on the selected model and the sequence of previous outcomes.[1]

The performance of a prediction strategy based on a set of models is usually analysed in terms of the *rate of convergence*, which expresses how quickly the distribution issued by the prediction strategy starts to behave like the data generating process $P^*$. From the information inequality ($D(P^* \| Q) \geq 0$ with equality for $Q = P^*$) we know that $P^*$ itself is optimal for prediction. The expected discrepancy between the number of bits needed by the prediction strategy to encode the next outcome and the number of bits needed by the true distribution $P^*$ is called the *risk*; a prediction strategy has a good rate of convergence if the risk goes down quickly as a function of the sample size. A number of results that give explicit guarantees about the risk convergence rate for prediction strategies based on the MDL model selection criterion are given in [39]. Interestingly, the

[1] This strategy does not take into account the predictions of any of the other models, while at the same time we can never be sure that the selected model is actually the best one. It is better to take a weighted average of the predictions of the models in question; the most common practice is to weight the models by their Bayesian posterior probability, a procedure called *Bayesian model averaging*; see Chapter 5.

risk may converge to zero even if $P^*$ is not an element of any of the models under consideration. This is useful in practical applications: for example, if the models correspond to histogram densities with increasing numbers of bins, then the risk converges to zero if $P^*$ is described by any bounded and continuous density; see Chapter 5 for more details.

Roughly speaking, two groups of model selection criteria can be distinguished in the literature. Model selection criteria of the first group have often been developed for applications of prediction; criteria such as Akaike's Information Criterion (AIC) and Leave One Out Cross-Validation (LOO), exhibit a very good risk convergence rate, but they can be inconsistent, i.e. they keep selecting the wrong model regardless of the available amount of data. The second group contains criteria such as the Bayesian Information Criterion (BIC), as well as regular Bayesian/MDL model selection, which can be proven consistent but which are often found to yield a somewhat worse rate of convergence. It is a long standing question whether these two approaches can be reconciled [103].

An assumption that is implicitly made when MDL or Bayesian model selection are used for prediction, is that there is one model that exhibits the best predictive performance from the start, and our only task is to identify that model. However, in Chapter 5 we argue that this approach is in fact often too simplistic: in reality, which model can provide the best predictive performance may well *depend on the sample size*. In the histogram density estimation example from above, we typically do not expect any particular model to contain the true distribution; instead we expect to use higher and higher order models as we gain enough data to estimate the model parameters with sufficient accuracy to make good predictions.

In Chapter 5 we attempt to combine the strong points of the two different approaches to model selection by dropping the assumption that one model has best performance at all sample sizes. This results in a modification of MDL and Bayesian model selection/averaging, called the *switch-distribution*, that achieves better predictive performance, *without sacrificing consistency*. Note that the switch-distribution can be used to improve MDL model selection as well as Bayesian model selection, although the idea may not agree very well with a Bayesian mindset, as we will discuss.

The switch-distribution ties in with existing universal prediction literature on *tracking the best expert*, where the assumption that one predictor is best at all sample sizes has already been lifted [94, 46, 95]. In the past, that research has not been connected with model selection though. In Chapter 6 we describe how the switch-code lengths can be computed efficiently, and introduce a formalism that helps to define other practical methods of combining predictors. These ideas help understand the connection between many seemingly very different prediction schemes that are described in the literature.

### 1.3.3 Separating Structure from Noise

Algorithmic Rate-Distortion Theory is a generalisation of ideal MDL. Recall from 1.1.3 that in ideal MDL, Kolmogorov complexity is used as the code length function for the hypothesis $L(H) = K(H)$; model selection is then with respect to the set $\mathcal{H}$ of all computable hypotheses. This approach is generalised by introducing a parameter $\alpha$, called the *rate*, that is used to impose a maximum on the complexity of the hypothesis $K(H)$. The selected hypothesis $H$, and the code length it achieves on the data $L_H(D)$, now become functions of $\alpha$. The latter function is called the *structure function*:[2]

$$h_D(\alpha) = \min_{H \in \mathcal{H}, K(H) \leq \alpha} L_H(D) \tag{1.16}$$

We write $H(\alpha)$ for the hypothesis that achieves the minimum at rate $\alpha$. We use dotted symbols $\dot{=}$, $\dot{<}$ and $\dot{\leq}$ for (in)equality up to an independent additive constant; the dot may be pronounced "roughly".

A hypothesis $H$ is called a *sufficient statistic* if $K(H) + L_H(D) \dot{=} K(D)$. Intuitively, such hypotheses capture all simple structural properties of the data. Namely, if there are any easily describable properties of $D$ that are not captured by $H$, then we would have $K(D|H) \dot{<} L_H(D)$. But in that case $K(D) \dot{\leq} K(H) + K(D|H) \dot{<} K(H) + L_H(D)$ which is impossible for sufficient statistics $H$.

If the rate is high enough, certainly for $\alpha = K(D)$, the hypothesis $H(\alpha)$ is a sufficient statistic. The most thorough separation of structure and noise is obtained at the lowest rate at which $H(\alpha)$ is a sufficient statistic. At even lower rates, some structural properties of $D$ can no longer be represented; the structure function can then be used as an indicator of how much structure has been discarded.

*Algorithmic rate-distortion theory* generalises even further by introducing a distortion function, which allows expression of the kind of properties of the original object we consider important. It is an analogue of Shannon's classical rate-distortion theory [79, 37]. The novelty of algorithmic rate-distortion theory compared to classical rate-distortion theory is that it allows analysis of *individual objects* rather than expected properties of objects drawn from a source distribution. This is useful because it is often impossible to define a source distribution that is acceptable as a reasonable model of the process of interest. For example, from what source distribution was Tolstoy's War and Piece drawn?

In Chapter 6 we introduce this new theory of algorithmic rate-distortion in more detail, and then proceed to put it to the test. To obtain a practical method, we approximate Kolmogorov complexity by the compressed size under a general purpose data compression algorithm. We then compute the rate-distortion characteristics of four objects from very different domains and apply the results to

[2] In Kolmogorov's original formulation, the codes $L_H$ for $H \in \mathcal{H}$ were uniform on finite sets of possible outcomes.

denoising and lossy compression. The chapter is written from the point of view of algorithmic rate-distortion theory, but includes a discussion of how the approach relates to MDL as we describe it in this introduction.

## 1.4 Organisation of this Thesis

In this introduction we have outlined MDL model selection. An important ingredient of MDL model selection is universal coding (Section 1.2.3). We have described several approaches; the universal code preferred in the MDL literature is the code that corresponds to the Normalised Maximum Likelihood (NML) distribution (1.13), which achieves minimal regret in the worst case. However it turns out that the NML distribution is often undefined. In those cases, the way to proceed is not agreed upon. In Chapter 2, we evaluate various alternatives experimentally by applying MDL to select between the Poisson and geometric models. The results provide insight into the strengths and weaknesses of these various methods.

One important result of the experiments described in Chapter 2 is that the prequential forecasting system, one of the universal models introduced in Section 1.2.3), defined to issues predictions on the basis of the maximum likelihood distribution, achieves a regret very different from that achieved by other models. This leads to inferior model selection performance. This was unexpected, because in the literature, e.g. [71], the prequential ML universal model is described as a practical approximation of other universal codes, certainly suitable for model selection. In Chapter 3 we analyse the regret of the prequential ML universal distribution under misspecification, that is, the data are drawn from a distribution outside the model. The behaviour under misspecification is important to model selection, since there the data generating distribution is often in only one of the considered models (if that many). Together, chapters 2 and 3 form the first leg of this thesis.

The second leg involves the relationship between model selection and prediction (see Section 1.3). It is based on the observation that the model that allows for the best predictions may *vary with the sample size*. For example, simple models with only few parameters tend to predict reasonably well at small sample sizes while for large models often a lot of data is required before the parameters can be estimated accurately enough to result in good predictions. While this may seem obvious, we show that the codes that are typically used in Bayesian and MDL model selection do not take this effect into account and thus the achieved code lengths are longer than they need to be. In Chapter 4 we describe a number of alternative ways to combine the predictions of multiple codes. It is also shown how the resulting code lengths can still be computed efficiently. The discussion expands on results in the source coding and universal prediction. One of the described codes, called the *switch-code*, is specifically designed to improve model

selection results. In Chapter 5 we show how model selection based on the switch-code, while remaining consistent, achieves the best possible rate of convergence. This resolves a perceived distinction between consistent model selection criteria on the one hand, and criteria that are suitable for prediction, which achieve a fast rate of convergence, on the other hand.

In Chapter 6, the third leg of this thesis, we put the new algorithmic rate-distortion theory to the test. Algorithmic rate-distortion theory is introduced in Section 1.3.3, more thoroughly in Chapter 6 itself, and even more thoroughly in [92]. It is a generalisation of ideal MDL (Section 1.1.3) and a variation on classical rate-distortion theory [79]; it allows for analysis of the rate-distortion properties of individual objects rather than of objects drawn from some source distribution. Algorithmic rate-distortion theory uses Kolmogorov complexity which is uncomputable. To obtain a practical method, we approximate Kolmogorov complexity by the compressed size under a general purpose data compression algorithm. We then compute the rate-distortion characteristics of four objects from very different domains and apply the results to denoising and lossy compression.

# Chapter 2

# Dealing with Infinite Parametric Complexity

Model selection is the task of choosing one out of a set of hypotheses for some phenomenon, based on the available data, where each hypothesis is represented by a model, or set of probability distributions. As explained in the introductory chapter, MDL model selection proceeds by associating a so-called *universal distribution* with each model. The number of bits needed to encode the data $D$ using the universal code associated with model $\mathcal{M}$ is denoted $L_{\mathcal{M}}(D)$. We then pick the model that minimises this expression and thus achieves the best compression of the data.

Section 1.2.3 lists the most important such universal codes, all of which result in slightly different code lengths and thus to different model selection criteria. While any choice of universal code will lead to an asymptotically "consistent" model selection method (eventually the right model will be selected), to get good results for small sample sizes it is crucial that we select an efficient code.

The MDL literature emphasises that the used universal code should be efficient *whatever data are observed*: whenever the model contains a code or distribution that fits the data well in the sense that it achieves a small code length, the universal code length should not be much larger. More precisely put, the universal code should achieve small regret in the worst case over all possible observations, where the regret is the difference between the universal code length and the shortest code length achieved by an element of the model $\mathcal{M}$. The *normalised maximum likelihood* (NML) code (Section 1.2.3) minimises this worst-case regret. Moreover, the NML regret is a function of only the sample size $n$, and does not depend on the data. As such it can be viewed as an objective measure of model complexity: the *parametric complexity* of a model is the regret achieved by the NML code, as a function of the sample size $n$. For this reason NML is the preferred universal code in modern MDL.

However, it turns out that the parametric complexity is infinite for many models, so that NML-based model selection is not effective. Other universal codes

that do achieve finite code lengths can usually still be defined for such models, but those codes have a worst-case regret that depends not only on the sample size $n$, but also on the parameter of the best fitting distribution in the model $\hat{\theta}$; such codes are *luckiness codes* in the sense of Section 1.1.1. Model selection criteria based on such luckiness codes thus acquire an element of subjectivity.

Several remedies have been proposed for this situation: variations on NML-based model selection that can obviously not be worst case optimal, but that do achieve finite code length without explicitly introducing regret that depends on the element of the model that best fits the data. In this chapter we evaluate such alternative procedures empirically for model selection between the simple Poisson and geometric models. We chose these two models since they are just about the simplest and easiest-to-analyse models for which the NML code is undefined. We find that some alternative solutions, such as the use of BIC (or, equivalently, in our context, maximum likelihood testing) lead to relatively poor results. Our most surprising finding is the fact that the prequential maximum likelihood code – which was found to perform quite well in some contexts [61, 52] – exhibits poor behaviour in our experiments. We briefly discuss the reasons for this behaviour in Section 2.5; a full analysis is the subject of Chapter 3.

Since the codes used in these approaches no longer minimise the worst-case regret they are harder to justify theoretically. In fact, as explained in more detail in Section 2.3.6, the only method that may have an MDL-type justification closely related to that of the original NML code is the Bayesian code with the improper Jeffreys' prior. Perhaps not coincidentally, this also seems the most dependable selection criterion among the ones we tried.

In Section 2.1 we describe the code that achieves worst-case minimal regret. This code does not exist for the Poisson and geometric distributions. We analyse these models in more detail in Section 2.2.

In Section 2.3 we describe four different approaches to MDL model selection under such circumstances. We test these criteria by measuring error probability, bias and calibration, as explained in Section 2.4. The results are evaluated in Section 2.5. Our conclusions are summarised in Section 2.6.

## 2.1 Universal codes and Parametric Complexity

We first briefly review some material from the introductory chapter that is especially important to this chapter. The *regret* of a code $L$ with respect to a parametric model $\mathcal{M} = \{P_\theta : \theta \in \Theta\}$ with parameter space $\Theta$ on a sequence of outcomes $x^n = x_1, \ldots, x_n \in \mathcal{X}^n$ is

$$\mathcal{R}(L, \mathcal{M}, x^n) := L(x^n) - \inf_{\theta \in \Theta} - \log P_\theta(x^n),$$

which was first defined in (1.4), page 5. A function $\hat{\theta} : \mathcal{X}^n \to \Theta$ that maps outcomes $x^n$ to a parameter value that achieves this infimum is called a *maximum*

*likelihood estimator*. We sometimes abbreviate $\hat{\theta} = \hat{\theta}(x^n)$.

A code $L$ is *$f$-universal with respect to model $\mathcal{M}$* if $\mathcal{R}(L, \mathcal{M}, x^n) \leq f(\hat{\theta}, n)$ for all $n$ and all $x^n$ (Definition 1.2.2, page 14). The MDL philosophy [72, 38] has it that the best universal code minimises the regret in the worst case of all possible data sequences. This "minimax optimal" solution is called the "Normalised Maximum Likelihood" (NML) code, which was first described by Shtarkov, who also observed its minimax optimality properties. The NML-probability of a data sequence for a parametric model is defined as in (1.13), page 18:

$$P_{\text{nml}}(x^n) := \frac{P(x^n|\hat{\theta}(x^n))}{\sum_{y^n} P(y^n|\hat{\theta}(y^n))},$$

with corresponding code length

$$L_{\text{nml}}(x^n) := -\log P(x^n|\hat{\theta}(x^n)) + \log \sum_{y^n} P(y^n|\hat{\theta}(y^n)).$$

This code length is called the *stochastic complexity* of the data. The first term in the equation is the code length of the data using the maximum likelihood estimator; the second term is the $-\log$ of the normalisation factor of $P_{\text{nml}}$, called the *parametric complexity* of the model $\mathcal{M}$ at sample size $n$.

It is usually impossible to compute the parametric complexity analytically, but there exists a good approximation $L_{\text{anml}}$, due to Rissanen, Takeuchi and Barron [72, 88, 89, 87]:

$$L_{\text{anml}}(x^n) := L(x^n|\hat{\theta}) + \frac{k}{2}\log\frac{n}{2\pi} + \log\int_{\Theta}\sqrt{\det I(\theta)}\,\mathrm{d}\theta. \tag{2.1}$$

Here, $n$, $k$ and $I(\theta)$ denote the number of outcomes, the number of parameters and the Fisher information matrix respectively. If $\mathcal{M}$ is an exponential family model such as the Poisson and Geometric models considered in this chapter, and is parameterised by $\Theta \subseteq \mathbb{R}^k$ for some $k \in \mathbb{Z}^+$, and if both the parametric complexity and the integral in (2.1) are finite, then we have the following. For any $\Theta'$ with nonempty interior, and whose closure is a bounded subset of the interior of $\Theta$, we have

$$\lim_{n\to\infty} \sup_{x^n:\hat{\theta}(x^n)\in\Theta'} |L_{\text{nml}}(x^n) - L_{\text{anml}}(x^n)| = 0.$$

Thus, the approximation uniformly converges to the exact code length for all sequences of outcomes whose maximum likelihood estimator does not converge to the boundaries of the parameter space. For more information, see [39]. Since the last term in (2.1) does not depend on the sample size, it has often been disregarded and many people came to associate MDL only with the first two terms. But the third term can be quite large or even infinite, and it can substantially influence the inference results for small sample sizes. Interestingly, (2.1) also describes

the asymptotic behaviour of the Bayesian universal code where Jeffreys' prior is used: here MDL and an objective Bayesian method coincide even though their motivation is quite different.

As stated in the introduction, the problem is that for many models the parametric complexity is infinite. Many strategies have been proposed to deal with this, but most are somewhat ad-hoc. When Rissanen defines stochastic complexity as $L_{\mathrm{nml}}(x^n)$ in [72], he writes that he does so "thereby concluding a decade long search", but as Lanterman observes in [53], "in the light of these problems we may have to postpone concluding the search just a while longer".

## 2.2 The Poisson and Geometric Models

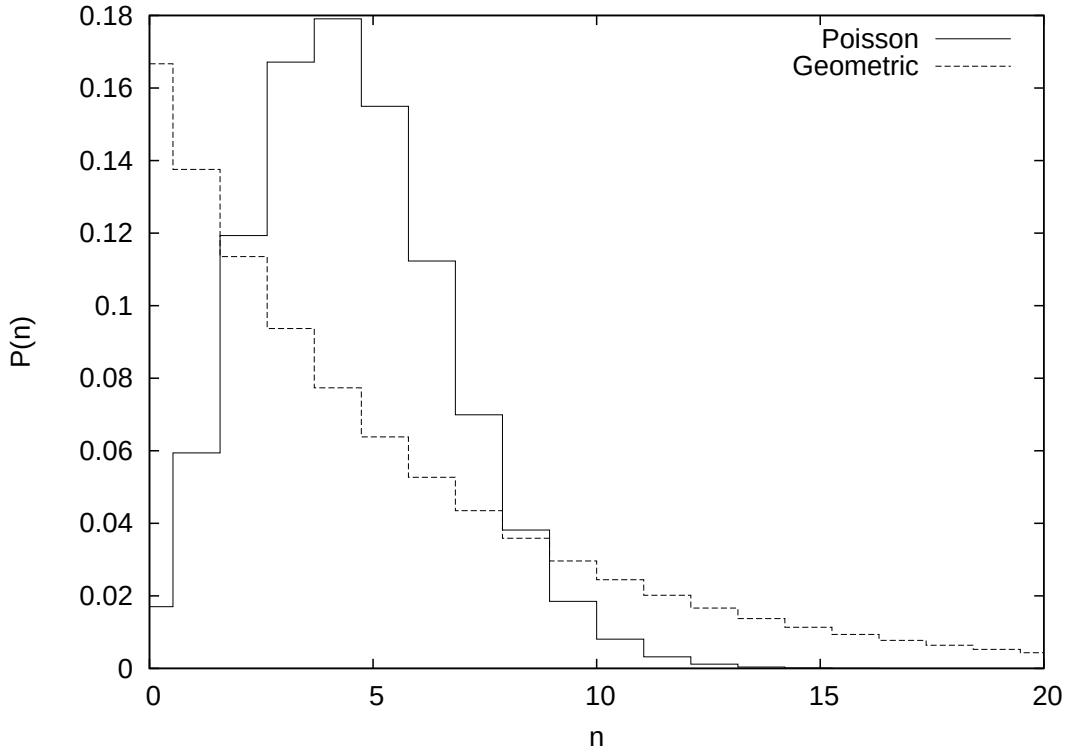
We investigate MDL model selection between the Poisson and Geometric models. Figure 2.1 may help form an intuition about the probability mass functions of the two distributions. One reason for our choice of models is that they are both single parameter models, so that the dominant $\frac{k}{2}\log\frac{n}{2\pi}$ term of (2.1) cancels. This means that at least for large sample sizes, simply picking the model that best fits the data should always work. We nevertheless observe that for small sample sizes, data generated by the geometric distribution are misclassified as Poisson much more frequently than the other way around (see Section 2.5). So in an informal sense, even though the number of parameters is the same, the Poisson distribution is more prone to "overfitting".

To counteract the bias in favour of Poisson that is introduced if we just select the best fitting model, we would like to compute the third term of (2.1), which now characterises the parametric complexity. But as it turns out, both models have an infinite parametric complexity! The integral in the third term of the approximation also diverges. So in this case it is not immediately clear how the bias should be removed. This is the second reason why we chose to study the Poisson and Geometric models. In Section 2.3 we describe a number of methods that have been proposed in the literature as ways to deal with infinite parametric complexity; in Section 2.5 they are evaluated empirically.

Reassuringly, all methods we investigate tend to compensate for this overfitting phenomenon by "punishing" the Poisson model. However, to what extent the bias is compensated depends on the used method, so that different methods give different results.

We parameterise both the Poisson and the Geometric family of distributions by the mean $\mu \in (0, \infty)$, to allow for easy comparison. This is possible because for both models, the empirical mean (average) of the observed data is a sufficient statistic. For Poisson, parameterisation by the mean is standard. For geometric, the reparameterisation can be arrived at by noting that in the standard parameterisation, $P(x|\theta) = (1-\theta)^x\theta$, the mean is given by $\mu = (1-\theta)/\theta$. As a notational reminder the parameter is called $\mu$ henceforth. Conveniently, the ML estimator

**Figure 2.1** The mean 5 Poisson and geometric distributions.




$\hat{\mu}$ for both distributions is the average of the data.

We will add a subscript P or G to indicate that code lengths are computed with respect to the Poisson model or the Geometric model, respectively. Furthermore, to simplify the equations in the remainder of this chapter somewhat we will express code lengths in nats ($-\ln$ probabilities) rather than bits ($-\log_2$ probabilities).

$$L_{\mathrm{P}}\left(x^n|\mu\right) = -\ln\prod_{i=1}^{n}\frac{e^{-\mu}\mu^{x_i}}{x_i!} = \sum_{i=1}^{n}\ln(x_i!) + n\mu - \ln\mu\sum_{i=1}^{n}x_i, \tag{2.2}$$

$$L_{\mathrm{G}}\left(x^n|\mu\right) = -\ln\prod_{i=1}^{n}\frac{\mu^{x_i}}{(\mu+1)^{x_i+1}} = n\ln(\mu+1) - \ln\left(\frac{\mu}{\mu+1}\right)\sum_{i=1}^{n}x_i. \tag{2.3}$$

## 2.3 Four Ways to deal with Infinite Parametric Complexity

In this section we discuss four general ways to deal with the infinite parametric complexity of the Poisson and Geometric models when the goal is to do model

selection. Each of these four methods leads to one, or sometimes more, concrete model selection criteria which we evaluate in Section 2.5.

### 2.3.1 BIC/ML

One way to deal with the diverging integral in the approximation is to just ignore it. The model selection criterion that results corresponds to only a very rough approximation of any real universal code, but it has been used and studied extensively. It was first derived by Jeffreys as an approximation to the Bayesian marginal likelihood [47], but it became well-known only when it was proposed by Rissanen [67] and Schwarz [78]. While Schwarz gave the same type of derivation as Jeffreys, Rissanen arrived at it in a quite different manner, as an approximation to a two-part code length. We note that Rissanen already abandoned the idea in the mid 1980's in favour of more sophisticated code length approximations. Because of its connection to the Bayesian marginal likelihood, it is best known as the BIC (Bayesian Information Criterion):

$$L_{\mathrm{BIC}}(x^n) = L(x^n|\hat{\mu}) + \frac{k}{2}\ln n.$$

Comparing BIC to the approximated NML code length we find that in addition to the diverging term, a $\frac{k}{2}\ln\frac{1}{2\pi}$ term has also been dropped. This curious difference can be safely ignored in our setup, where $k$ is equal to one for both models so the whole term cancels anyway. According to BIC, we must select the Geometric model if

$$0 < L_{\mathrm{P,BIC}}\left(x^n\right) - L_{\mathrm{G,BIC}}\left(x^n\right) = L_{\mathrm{P}}\left(x^n|\hat{\mu}\right) - L_{\mathrm{G}}\left(x^n|\hat{\mu}\right).$$

We are left with a generalised likelihood ratio test (GLRT). In such a test, the ratio of the probabilities under the two models, $P_{\mathrm{P}}\left(x^n|\hat{\mu}\right)/P_{\mathrm{G}}\left(x^n|\hat{\mu}\right)$ is compared against a fixed constant $\eta$; the BIC criterion thus reduces to a GLRT with $\eta = 0$, which is also known as maximum likelihood (ML) testing. As we remarked before, experience shows that this approach often leads to overfitting and a bias in favour of the "more complex" Poisson model. (On a side note, this equivalence of BIC and ML occurs when all models under consideration have the same numbers of parameters; if this is not the case, then BIC may or may not overfit and it usually gives better results than ML.)

### 2.3.2 Restricted ANML

One often used method of rescuing the NML approach to MDL model selection is to restrict the range of values that the parameters can take to ensure that the third term of (2.1) stays finite. Our approach is to impose a maximum on the allowed mean by setting $\Theta = (0, \mu^*)$.

To compute the approximated parametric complexity of the restricted models we need to establish the Fisher information first. We use $I(\theta) = -E_\theta[\frac{\mathrm{d}^2}{\mathrm{d}\theta^2} L\left(x|\theta\right)]$ to obtain

$$I_{\mathrm{P}}(\mu) = -E_\mu\left[-\frac{x}{\mu^2}\right] = \frac{1}{\mu}, \text{ and} \tag{2.4}$$

$$I_{\mathrm{G}}(\mu) = -E_\mu\left[-\frac{x}{\mu^2} + \frac{x+1}{(\mu+1)^2}\right] = \frac{1}{\mu(\mu+1)}. \tag{2.5}$$

Now we can compute the last term in the parametric complexity approximation (2.1):

$$\ln \int_0^{\mu^*} \sqrt{I_{\mathrm{P}}(\mu)}\,\mathrm{d}\mu = \ln \int_0^{\mu^*} \mu^{-\frac{1}{2}}\,\mathrm{d}\mu = \ln\left(2\sqrt{\mu^*}\right); \tag{2.6}$$

$$\ln \int_0^{\mu^*} \sqrt{I_{\mathrm{G}}(\mu)}\,\mathrm{d}\mu = \ln \int_0^{\mu^*} \frac{1}{\sqrt{\mu(\mu+1)}}\,\mathrm{d}\mu = \ln\left\{2\ln\left(\sqrt{\mu^*} + \sqrt{\mu^*+1}\right)\right\}. \tag{2.7}$$

Thus, the parametric complexities of the restricted models are both monotonically increasing functions of $\mu^*$. Let the function $\delta(\mu^*) := \ln(2\sqrt{\mu^*}) - \ln(2\ln(\sqrt{\mu^*} + \sqrt{\mu^*+1}))$ measure the difference between the parametric complexities. We obtain a model selection criterion that selects the Geometric model if

$$0 < L_{\mathrm{P,ANML}(\mu^*)}(x^n) - L_{\mathrm{G,ANML}(\mu^*)}(x^n) = L_{\mathrm{P}}\left(x^n|\hat{\mu}\right) - L_{\mathrm{G}}\left(x^n|\hat{\mu}\right) + \delta(\mu^*).$$

This is equivalent to a GLRT with threshold $\delta(\mu^*)$. We have experimented with restricted models where the parameter range was restricted to $(0, \mu^*)$ for $\mu^* \in \{10, 100, 1000\}$.

It is not hard to show that the parametric complexity of the restricted Poisson model grows *faster* with $\mu^*$ than the parametric complexity of the Geometric model: $\delta(\mu^*)$ is monotonically increasing in $\mu^*$, and grows unboundedly in $\mu^*$. This indicates that the Poisson model has more descriptive power, even though the models have the same number of parameters and both have infinite parametric complexity.

An obvious conceptual problem with restricted ANML is that the imposed restriction is quite arbitrary and requires a priori knowledge about the scale of the observations. But the parameter range can be interpreted as a hyper-parameter, which can be incorporated into the code using several techniques; two such techniques are discussed next.

### 2.3.3 Two-part ANML

Perhaps the easiest way to get rid of the $\mu^*$ parameter that determines the parameter range, and thereby the restricted ANML code length, is to use a two-part

code. The first part contains a specification of $\mu^*$, which is followed by an encoding of the rest of the data with an approximate code length given by restricted ANML for that $\mu^*$. To do this we need to choose some discretisation, such that whatever $\hat{\mu}$ is, it does not cost many bits to specify an interval that contains it. For a sequence with ML parameter $\hat{\mu}$, we choose to encode the integer $b = \lceil \log_2 \hat{\mu} \rceil$. A decoder, upon reception of such a number $b$, now knows that the ML parameter value must lie in the range $(2^{b-1}, 2^b]$ (for otherwise another value of $b$ would have been transmitted). By taking the logarithm we ensure that the number of bits used in coding the parameter range grows at a negligible rate compared to the code length of the data itself, but we admit that the code for the parameter range allows much more sophistication. We do not really have reason to assume that the best discretisation should be the same for the Poisson and Geometric models for example.

The two-part code is slightly redundant, since code words are assigned to data sequences of which the ML estimator lies outside the range that was encoded in the first part – such data sequences cannot occur, since for such a sequence we would have encoded a different range. Furthermore, the two-part code is no longer minimax optimal, so it is no longer clear why it should be better than other universal codes which are not minimax optimal. However, as argued in [38], whenever the minimax optimal code is not defined, we should aim for a code $L$ which is "close" to minimax optimal in the sense that, for any compact subset $\mathcal{M}'$ of the parameter space, the additional regret of $L$ on top of the NML code for $\mathcal{M}'$ should be small, e.g. $O(\log \log n)$. The two-part ANML code is one of many universal codes satisfying this "almost minimax optimality". While it may not be better than another almost minimax optimal universal code, it certainly is better than universal codes which do not have the almost minimax optimality property.

### 2.3.4 Renormalised Maximum Likelihood

Related to the two-part restricted ANML, but more elegant, is Rissanen's *renormalised maximum likelihood* (RNML) code, [73, 38]. This is perhaps the most widely known approach to deal with infinite parametric complexity. The idea here is that the NML distribution is well-defined *if* the parameter range is restricted to, say, the range $(0, \mu^*)$. Letting $P_{\text{NML},\mu^*}$ be the NML distribution relative to this restricted model, we can now define a new parametric model, with $\mu^*$ as the parameter and the corresponding restricted NML distributions $P_{\text{NML},\mu^*}$ as elements. For this new model we can again compute the NML distribution! To do this, we need to compute the ML value for $\mu^*$, which in this case can be seen to be as small as possible such that $\hat{\mu}$ still falls within the range, in other words, $\mu^* = \hat{\mu}$.

If this still leads to infinite parametric complexity, we define a hyper-hyper-parameter. We repeat the procedure until the resulting complexity is finite. Unfortunately, in our case, after the first renormalisation, both parametric complex-

ities are still infinite; we have not performed a second renormalisation. Therefore, the RNML code is not included in our experiments.

### 2.3.5 Prequential Maximum Likelihood

The *prequential maximum likelihood code*, which we will abbreviate PML-code, is an attractive universal code because it is usually a lot easier to implement than either NML or a Bayesian code. Moreover, its implementation hardly requires any arbitrary decisions. Here the outcomes are coded sequentially using the probability distribution indexed by the ML estimator for the previous outcomes [26, 69]; for a general introduction see [96] or [38].

$$L_{\mathrm{PIPC}}\left(x^n\right) = \sum_{i=1}^{n} L\left(x_i|\hat{\mu}(x^{i-1})\right),$$

where $L\left(x_i|\hat{\mu}(x^{i-1})\right) = -\ln P\left(x_i|\hat{\mu}(x^{i-1})\right)$ is the number of nats needed to encode outcome $x_i$ using the code based on the ML estimator on $x^{i-1}$. We further discuss the motivation for this code in Section 2.5.1.

For both the Poisson model and the Geometric model, the maximum likelihood estimator is not well-defined until after a nonzero outcome has been observed (since 0 is not inside the allowed parameter range). This means that we need to use another code for the first few outcomes. It is not really clear how we can use the model assumption (Poisson or geometric) here, so we pick a simple code for the nonnegative integers that does not depend on the model. This will result in the same code length for both models; therefore it does not influence which model is selected. Since there are usually only very few initial zero outcomes, we may reasonably hope that the results are not too distorted by our way of handling this start-up problem. We note that this start-up problem is an inherent feature of the PML approach [26, 71], and our way of handling it is in line with the suggestions in [26].

### 2.3.6 Objective Bayesian Approach

In the Bayesian framework we select a prior $w(\theta)$ on the unknown parameter and compute the marginal likelihood

$$P_{\mathrm{BAYES}}\left(x^n\right) = \int_{\Theta} P\left(x^n|\theta\right) w(\theta)\,\mathrm{d}\theta, \tag{2.8}$$

with universal code length $L_{\mathrm{BAYES}}\left(x^n\right) = -\ln P_{\mathrm{BAYES}}\left(x^n\right)$. Like NML, this can be approximated with an asymptotic formula. Under conditions similar to those for the NML approximation (2.1), we have [3]:

$$L_{\mathrm{ABAYES}}\left(x^n\right) := L\left(x^n|\hat{\theta}\right) + \frac{k}{2}\ln\frac{n}{2\pi} + \ln\frac{\sqrt{\det I(\theta)}}{w(\theta)}, \tag{2.9}$$

where the asymptotic behaviour is the same as for the approximation of the NML code length, roughly $L_{\text{ABAYES}}(x^n) - L_{\text{BAYES}}(x^n) \to 0$ as $n \to \infty$ (see below Eq. (2.1) for details). Objective Bayesian reasoning suggests we use Jeffreys' prior for several reasons; one reason is that it is uniform over all "distinguishable" elements of the model [3], which implies that the obtained results are independent of the parameterisation of the model [47]. It is defined as follows:

$$w(\theta) = \frac{\sqrt{\det I(\theta)}}{\int_\Theta \sqrt{\det I(\theta)}\,\mathrm{d}\theta}. \tag{2.10}$$

Unfortunately, the normalisation factor in Jeffreys' prior diverges for both the Poisson model and the Geometric model. But if one is willing to accept a so-called *improper* prior, which is not normalised, then it is possible to compute a perfectly proper Bayesian posterior, after observing the first outcome, and use that as a prior to compute the marginal likelihood of the rest of the data. Refer to [14] for more information on objective Bayesian theory. The resulting universal codes with lengths $L_{\text{BAYES}}\left(x_2, \ldots, x_n \mid x_1\right)$ are, in fact, *conditional* on the first outcome. Recent work by [58] suggests that, at least asymptotically and for one-parameter models, the universal code achieving the minimal *expected redundancy conditioned on the first outcome* is given by the Bayesian universal code with the improper Jeffreys' prior. Li and Barron only prove this for scale and location models, but their result does suggest that the same would still hold for general exponential families such as Poisson and geometric. It is possible to define MDL inference in terms of either the expected redundancy or of the worst-case regret. In fact, the resulting procedures are very similar, see [4]. Thus, we have a tentative justification for using Jeffreys' prior also from an MDL point of view, on top of its justification in terms of objective Bayes.

It can be argued that using the first outcome for conditioning rather than some other outcome is arbitrary while it does influence the results. On the other hand, the posterior after having observed all data will be the same whatever outcome is elected to be the special one that we refrain from encoding. It also seems preferable to let results depend on arbitrary properties of the data than to let it depend on arbitrary decisions of the scientist, such as the choice for a maximum value for $\mu^*$ in the case of the restricted ANML criterion. As advocated for instance in [13], arbitrariness can be reduced by conditioning on every outcome in turn and then using the mean or median code length one so obtains. We have not gone to such lengths in this study.

We compute Jeffreys' posterior after observing one outcome, and use it to find the Bayesian marginal likelihoods. We write $x_i^j$ to denote $x_i, \ldots, x_j$ and $\hat{\mu}(x_i^j)$ to indicate which outcomes determine the ML estimator, finally we abbreviate $s_n = x_1 + \ldots + x_n$. The goal is to compute $P_{\text{BAYES}}\left(x_2^n \mid x_1\right)$ for the Poisson and Geometric models. As before, the difference between the corresponding code lengths defines a model selection criterion. We also compute $P_{\text{ABAYES}}\left(x_2^n \mid x_1\right)$

for both models, the approximated version of the same quantity, based on approximation formula (2.9). Equations for the Poisson and Geometric models are presented below.

**Bayesian code for the Poisson model** We compute Jeffreys' improper prior and the posterior after observing one outcome:

$$w_{\mathrm{P}}(\mu) \propto \sqrt{I_{\mathrm{P}}(\mu)} = \mu^{-\frac{1}{2}}; \tag{2.11}$$

$$w_{\mathrm{P}}(\mu \mid x_1) = \frac{P_{\mathrm{P}}(x_1|\mu)\, w_{\mathrm{P}}(\mu)}{\int_0^\infty P_{\mathrm{P}}(x_1|\theta)\, w_{\mathrm{P}}(\theta)\, d\theta} = \frac{e^{-\mu}\mu^{x_1-\frac{1}{2}}}{\Gamma(x_1+\frac{1}{2})}. \tag{2.12}$$

From this we can derive the marginal likelihood of the rest of the data. The details of the computation are omitted for brevity.

$$P_{\mathrm{P,BAYES}}(x_2^n \mid x_1) = \int_0^\infty P_{\mathrm{P}}(x_2^n|\mu)\, w_{\mathrm{P}}(\mu \mid x_1)\, d\mu = \frac{\Gamma(s_n+\frac{1}{2})}{\Gamma(x_1+\frac{1}{2})} \Big/ \left( n^{s+\frac{1}{2}} \prod_{i=2}^{n} x_i! \right). \tag{2.13}$$

For the approximation (2.9) we obtain:

$$L_{\mathrm{P,ABAYES}}(x_2^n \mid x_1) = L_{\mathrm{P}}(x_2^n|\hat{\mu}(x_2^n)) + \tfrac{1}{2}\ln\frac{n}{2\pi} + \hat{\mu}(x_2^n) - x_1 \ln \hat{\mu}(x_2^n) + \ln\Gamma(x_1+\tfrac{1}{2}). \tag{2.14}$$

**Bayesian code for the Geometric model** We perform the same computations for the Geometric model. This time we get:

$$w_{\mathrm{G}}(\mu) \propto \mu^{-\frac{1}{2}}(\mu+1)^{-\frac{1}{2}}; \tag{2.15}$$

$$w_{\mathrm{G}}(\mu \mid x_1) = (x_1+\tfrac{1}{2})\mu^{x_1-\frac{1}{2}}(\mu+1)^{-x_1-\frac{3}{2}}; \tag{2.16}$$

$$P_{\mathrm{G,BAYES}}(x^n) = (x_1+\tfrac{1}{2})\frac{\Gamma(s+\frac{1}{2})\Gamma(n)}{\Gamma(n+s+\frac{1}{2})}. \tag{2.17}$$

For the approximation we obtain:

$$L_{\mathrm{G,ABAYES}}(x_2^n \mid x_1) = L_{\mathrm{G}}(x_2^n|\hat{\mu}(x_2^n)) + \tfrac{1}{2}\ln\frac{n}{2\pi} + x_1 \ln\left(1+\frac{1}{\hat{\mu}(x_2^n)}\right) + \tfrac{1}{2}\ln(\hat{\mu}(x_2^n)) - \ln(x_1+\tfrac{1}{2}). \tag{2.18}$$

## 2.4 Experiments

The previous section describes four methods to compute or approximate the length of a number of different universal codes, which can be used in an MDL model selection framework. The MDL principle tells us to select the model using

which we can achieve the shortest code length for the data. This coincides with the Bayesian maximum a-posteriori (MAP) model with a uniform prior on the models. In this way each method for computing or approximating universal code lengths defines a model selection criterion, which we want to compare empirically.

**Known $\mu$ criterion** In addition to the criteria that are based on universal codes, as developed in Section 2.3, we define one additional, "ideal" criterion to serve as a reference by which the others can be evaluated. The *known* $\mu$ criterion cheats a little bit: it computes the code length for the data with knowledge of the mean of the generating distribution. If the mean is $\mu$, then the known $\mu$ criterion selects the Poisson model if $L_{\mathrm{P}}\left(x^n|\mu\right) < L_{\mathrm{G}}\left(x^n|\mu\right)$. Since this criterion uses extra knowledge about the data, it should be expected to perform better than the other criteria. The theoretical analysis of the known $\mu$ criterion is helped by the circumstance that (1) one of the two hypotheses equals the generating distribution and (2) the sample consists of outcomes which are i.i.d. according to this distribution. In [25], Sanov's Theorem is used to show that in such a situation, the probability that the criterion prefers the wrong model ("error probability") decreases exponentially in the sample size. If the Bayesian MAP model selection criterion is used then the following happens: if the data are generated using Poisson[$\mu$] then the error probability decreases exponentially in the sample size, with some error exponent; if the data are generated with Geom[$\mu$] then the overall error probability is exponentially decreasing with the same exponent [25, Theorem 12.9.1 on page 312 and text thereafter]. Thus, we expect that the line for the "known $\mu$" criterion is straight on a logarithmic scale, with a slope that is equal whether the generating distribution is Poisson or geometric. This proves to be the case, as can be seen from Figure 2.2.

**Tests** We perform three types of test on the selection criteria, which are described in detail in the following subsections:

1. Error probability measurements.
2. Bias measurements.
3. Calibration testing.

### 2.4.1 Error Probability

The *error probability* for a criterion is the probability that it will select a model that does not contain the distribution from which the data were sampled. In our experiments, samples are always drawn from a Poisson[$\mu$] distribution with probability $p$, or from a Geom[$\mu$] distribution with probability $1-p$. We measure the error probability through repeated sampling; strictly speaking we thus obtain *error frequencies* which approximate the error probability.

Figures 2.2, 2.4, 2.5 and 2.6 plot the sample size against the error frequency, using different means $\mu$ and different priors $p$ on the generating distribution. We use a log-scale, which allows for easier comparison of the different criteria; as we pointed out earlier, for the known $\mu$ criterion we should expect to obtain a straight line.

In Figures 2.4, 2.5 and 2.6 the log of the error frequency of the known $\mu$ criterion is subtracted from the logs of the error frequencies of the other criteria. This brings out the differences in performance in even more detail. The known $\mu$ criterion, which has no bias, is perfectly calibrated (as we will observe later) and which also has a low error probability under all circumstances (although biased criteria can sometimes do better if the bias happens to work in the right direction), is thus treated as a baseline of sorts.

### 2.4.2 Bias

We define the level of evidence in favour of the Poisson model as:

$$\Delta\left(x^n\right) := L_{\mathrm{G}}\left(x^n|\mu\right) - L_{\mathrm{P}}\left(x^n|\mu\right), \tag{2.19}$$

which is the difference in code lengths according to the known $\mu$ criterion. The other criteria define estimators for this quantity: the estimator for a criterion $C$ is defined as:

$$\Delta_{\mathrm{C}}(x^n) := L_{\mathrm{G,C}}\left(x^n\right) - L_{\mathrm{P,C}}\left(x^n\right) \tag{2.20}$$

(Some people are more familiar with Bayes factors, of which this is the logarithm.) In our context the *bias* of a particular criterion is the expected difference between the level of evidence according to that criterion and the true level of evidence,

$$E\left[\Delta_{\mathrm{C}}(X^n) - \Delta\left(X^n\right)\right]. \tag{2.21}$$

The value of this expectation depends on the generating distribution, which is assumed to be some mixture of the Poisson and geometric distributions of the same mean.

We measure the bias through sampling. We measure the bias with generating distributions Poisson[8] and Geom[8]; as before we vary the sample size. The results are in Figure 2.3.

### 2.4.3 Calibration

The classical interpretation of probability is frequentist: an event has probability $p$ if in a repeated experiment the frequency of the event converges to $p$. This interpretation is no longer really possible in a Bayesian framework, since prior assumptions often cannot be tested in a repeated experiment. For this reason, calibration testing is avoided by some Bayesians who may put forward that it

is a meaningless procedure from a Bayesian perspective. On the other hand, we take the position that even with a Bayesian hat on, one would like one's inference procedure to be calibrated – in the *idealised* case in which identical experiments are performed repeatedly, probabilities should converge to frequencies. If they do not behave as we would expect even in this idealised situation, then how can we trust inferences based on such probabilities in the real world with all its imperfections?

In the introduction we have indicated the correspondence between code lengths and probability. If the universal code lengths for the different criteria correspond to probabilities that make sense in a frequentist way, then the Bayesian a posteriori probabilities of the two models should too. To test this, we generate samples with a fixed mean and a fixed sample size; half of the samples are drawn from a Poisson distribution and half from a geometric distribution. We then compute the a posteriori probability that it is generated by the Poisson model, for each of the selection criteria. The samples are distributed over 40 bins by discretising their a posteriori probability. For each bin we count the number of sequences that actually were generated by Poisson. If the a posteriori Bayesian probability that the model is Poisson makes any sense in a frequentist way, then the result should be a more or less straight diagonal.

The results are in Figure 2.7. We used mean 8 and sample size 8 because on the one hand we want a large enough sample size that the posterior has converged to something reasonable, but on the other hand if we choose the sample size even larger it becomes exceedingly unlikely that a sequence is generated of which the probability that it is Poisson is estimated near 0.5, so we would need to generate an infeasibly large number of samples to get accurate results. Note that the "known $\mu$" criterion is perfectly calibrated, because its implicit prior distribution on the mean of the generating distribution puts all probability on the actual mean, so the prior perfectly reflects the truth in this case. Under such circumstances Bayesian and frequentist probability become the same, and we get a perfect answer.

We feel that calibration testing is too often ignored, while it can safeguard against inferences or predictions that bear little relationship to the real world. Moreover, in the objective Bayesian branch of Bayesian statistics, one does emphasise procedures with good frequentist behaviour [11]. At least in restricted contexts [23, 22], Jeffreys' prior has the property that the Kullback-Leibler divergence between the true distribution and the posterior converges to zero quickly, no matter what the true distribution is. Consequently, after observing only a limited number of outcomes, it should already be possible to interpret the posterior as an almost "classical" distribution in the sense that it can be verified by frequentist experiments [23].

## 2.5 Discussion of Results

Roughly, the results of our tests can be summarised as follows:

- In this toy problem, as one might expect, all criteria perform extremely well even while the sample size is small. But there are also small but distinct differences that illustrate relative strengths and weaknesses of the different methods. When extrapolated to a more complicated model selection problem, our results should help to decide which criteria are appropriate for the job.
- As was to be be expected, the known $\mu$ criterion performs excellently on all tests.
- The PML and BIC/ML criteria exhibit the worst performance.
- The basic restricted ANML criterion yields results that range from good to very bad, depending on the chosen parameter range. Since the range must be chosen without any additional knowledge of the properties of the data, this criterion is rather arbitrary.
- The results for the two-part restricted ANML and Objective Bayesian criteria are reasonable in all tests we performed; these criteria thus display robustness.

In the following subsections we evaluate the results for each model selection criterion in turn.

### 2.5.1 Poor Performance of the PML Criterion

One feature of Figure 2.2 that immediately attracts attention is the unusual slope of the error rate line of the PML criterion, which clearly favours the geometric distribution. This is even clearer in Figure 2.3, where the PML criterion can be observed to become more and more favourable to the Geometric model as the sample size increases, regardless of whether the data were sampled from a Poisson or geometric distribution. This is also corroborated by the results on the calibration test, where the PML criterion most severely underestimates the probability that the data were sampled from a Poisson distribution: of those sequences that were classified as geometric with 80% confidence, in fact about 60% turned out to be sampled from a Poisson distribution.

While this behaviour may seem appropriate if the data really are more likely to come from a geometric distribution, there is actually a strong argument that *even under those circumstances it is not the most desirable behaviour*, for the following reason. Suppose that we put a fixed prior $p$ on the generating distribution, with nonzero probability for both distributions Poisson[$\mu$] and Geom[$\mu$]. The marginal

**Figure 2.2** The $\log_{10}$ of the error frequency.

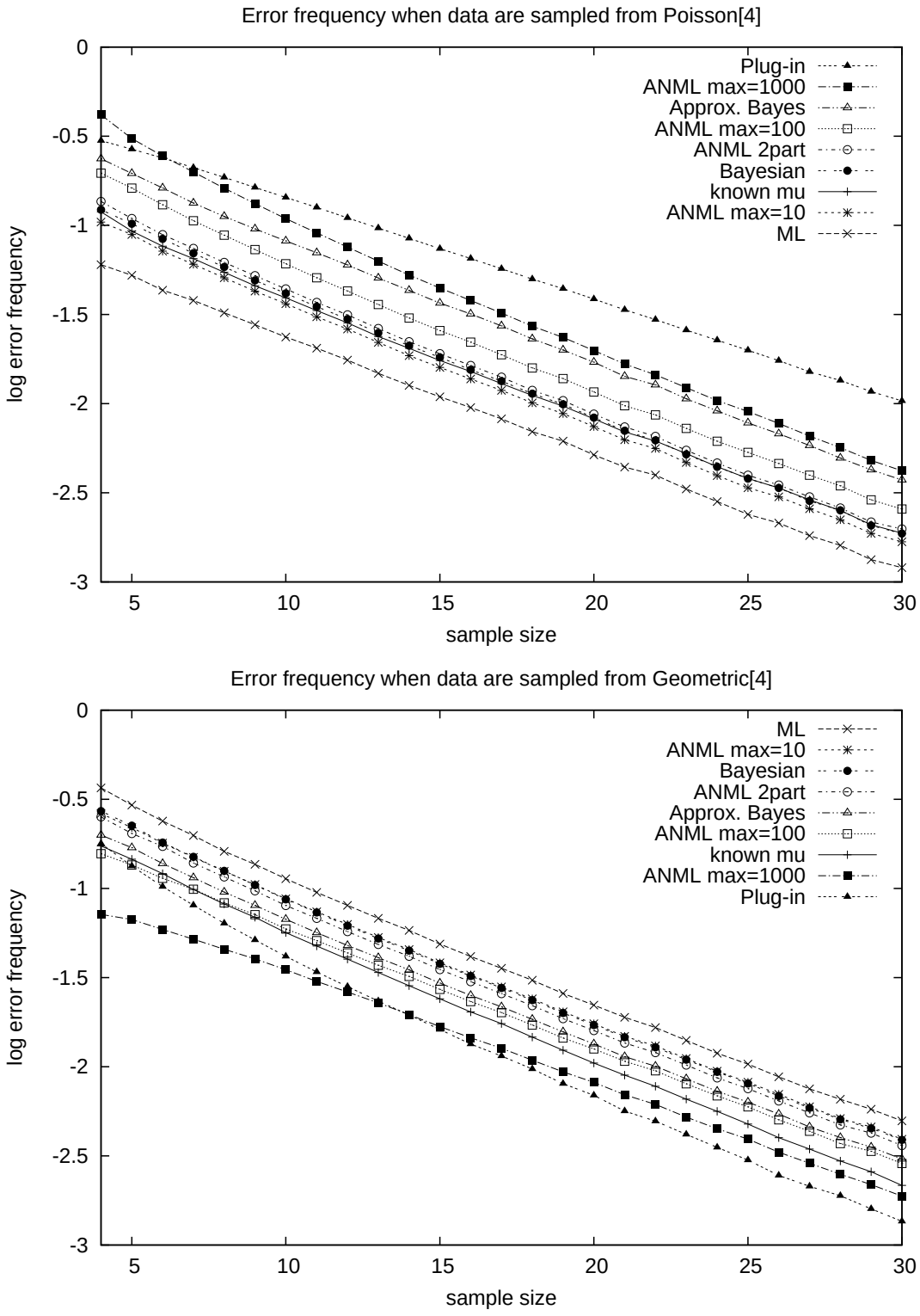


**Figure 2.3** The classification bias in favour of the Poisson model in bits.

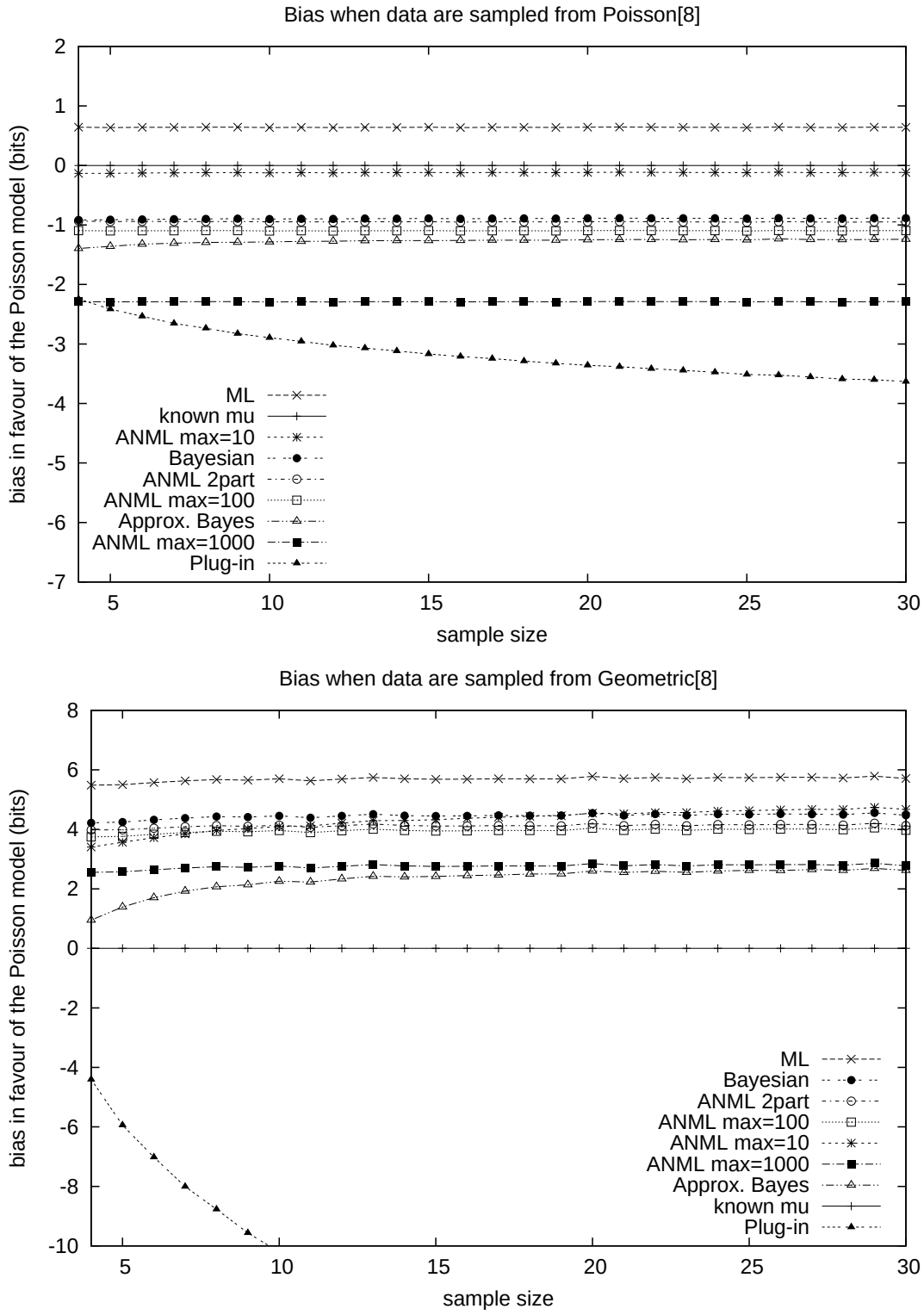


**Figure 2.4** The difference in the $\log_{10}$ of the frequency of error between each criterion and the known $\mu$ criterion. The mean is 4. In these graphs, data are sampled from one of the two models with unequal probabilities.

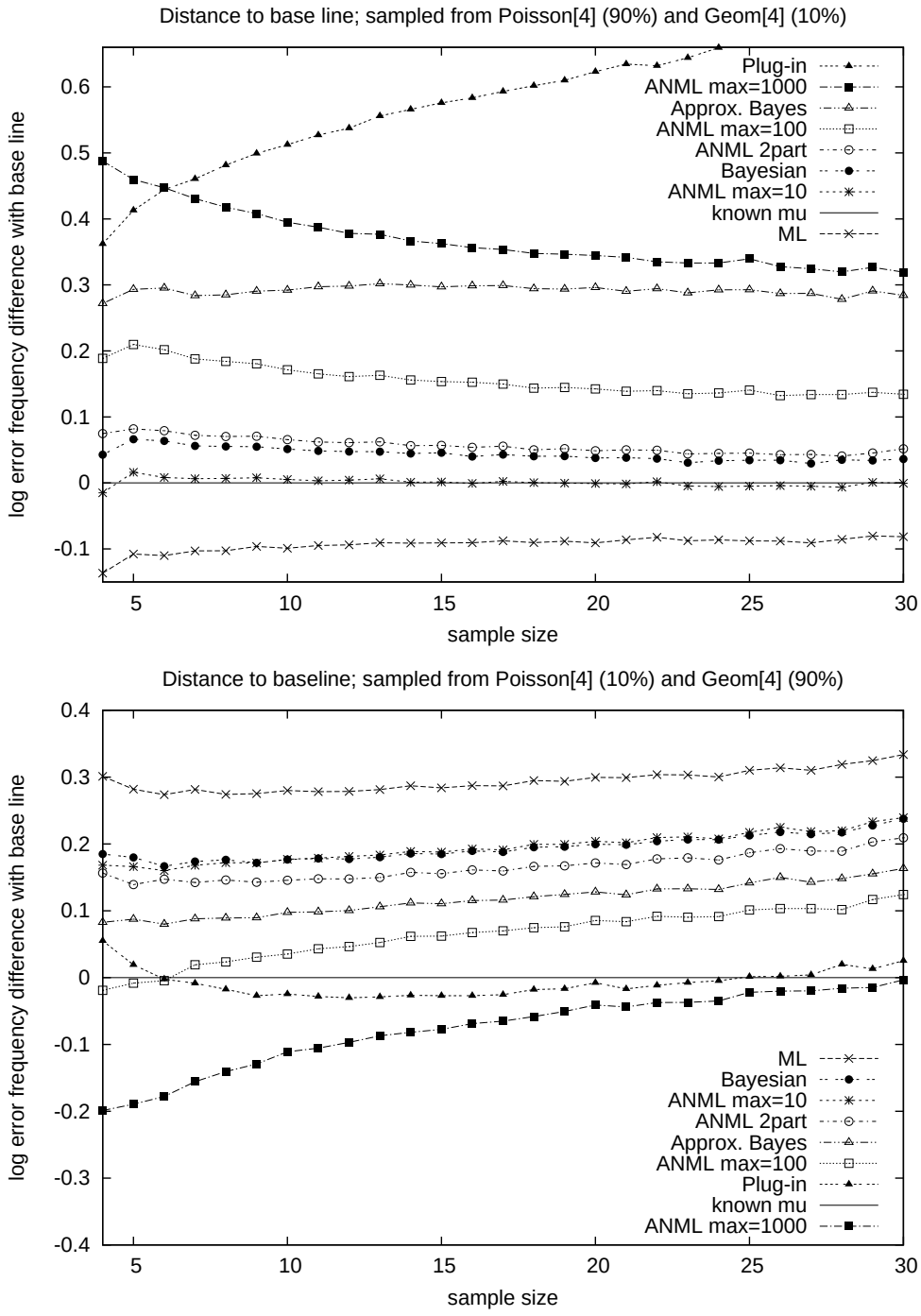


**Figure 2.5** The difference in the $\log_{10}$ of the frequency of error between each criterion and the known $\mu$ criterion. The mean is 4.

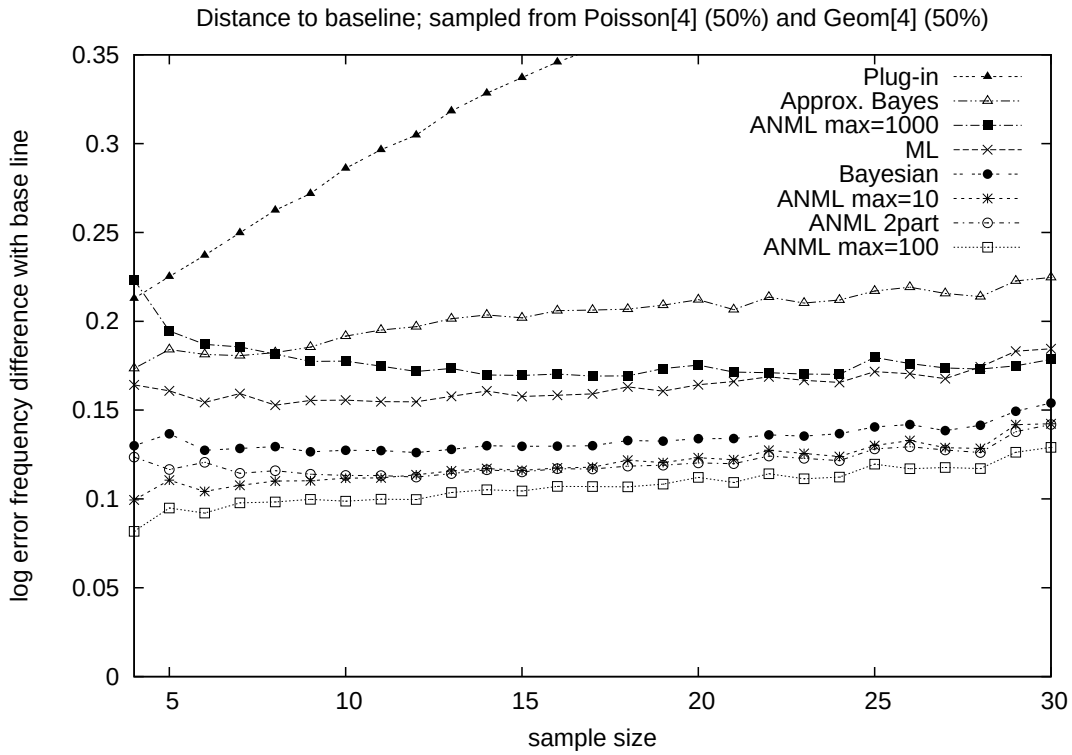
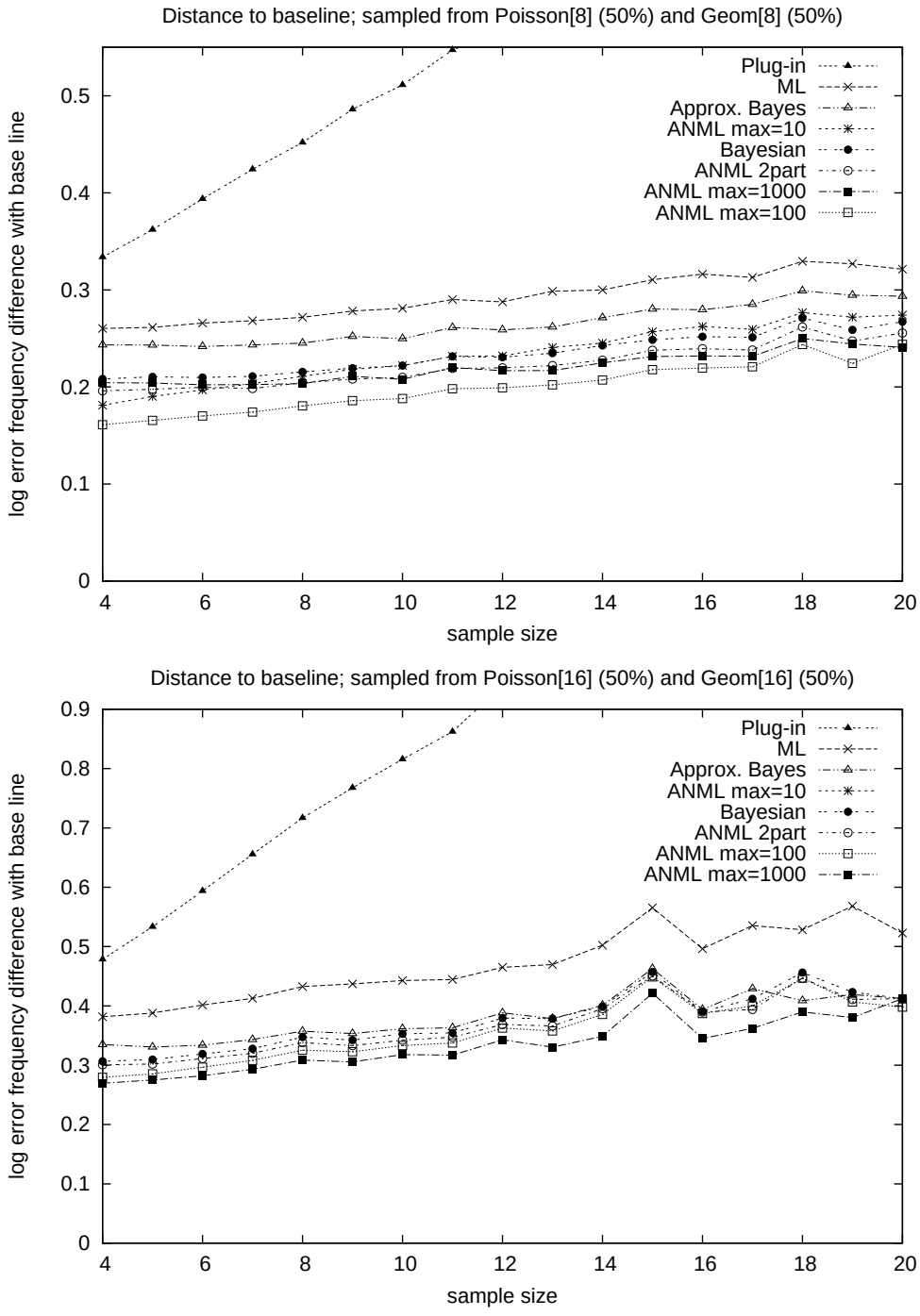


**Figure 2.6** The difference in the $\log_{10}$ of the frequency of error between each criterion and the known $\mu$ criterion. The mean is 8 in the top graph and 16 in the bottom graph.



error probability is a linear combination of the probabilities of error for the two generating distributions; as such it is dominated by the probability of error with the *worst* exponent. So if minimising the error probability is our goal, then we must conclude that the behaviour of the PML criterion is suboptimal. (As an aside, minimising the error probability with respect to a fixed prior is *not* the goal of classical hypothesis testing, since in that setting the two hypotheses do not play a symmetrical role.) To illustrate, the bottom graph in Figure 2.4 shows that, even if there is a 90% chance that the data are geometric, then the PML criterion still has a worse (marginal) probability of error than "known $\mu$" as soon as the sample size reaches 25. Figure 2.5 shows what happens if the prior on the generating distribution is uniform – using the PML criterion immediately yields the largest error probability of all the criteria under consideration. This effect only becomes stronger if the mean is higher.

This strangely poor behaviour of the PML criterion initially came as a complete surprise to us. Theoretical literature certainly had not suggested it. Rather the contrary: in [71] we find that "it is only because of a certain inherent singularity in the process [of PML coding], as well as the somewhat restrictive requirement that the data must be ordered, that we do not consider the resulting predictive code length to provide another competing definition for the stochastic complexity, but rather regard it as an approximation". There are also a number of results to the effect that the regret for the PML code grows as $\frac{k}{2}\ln n$, the same as the regret for the NML code, for a variety of models. Examples are [70, 35, 97]. Finally, publications such as [61, 52] show excellent behaviour of the PML criterion for model selection in regression and classification based on Bayesian networks, respectively. So, we were extremely puzzled by these results at first.

To gain intuition as to why the PML code should behave so strangely, note that the variance of a geometric distribution is much larger than the variance of the Poisson distribution with the same mean. This suggests that the penalty for using an estimate, $\hat{\mu}(x^{n-1})$ rather than the optimal $\mu$ to encode each outcome $x_n$ is higher for the Poisson model. The accumulated difference accounts for the difference in regret.

This intuition is made precise in the next chapter, where we prove that for single parameter exponential families, the regret for the PML code grows with $\frac{1}{2}\ln(n)\mathrm{var}_{P^*}(X)/\mathrm{var}_{P_{\theta^*}}(X)$, where $P^*$ is the generating distribution, while $P_{\theta^*}$ is the element of the model that minimises $D(P^*\|P_{\theta^*})$. The PML model has the same regret (to $O(1)$) as the NML model if and only if the variance of the generating distribution is the same as the variance of the best element of the model. The existing literature studies the case where $P^* = P_{\theta^*}$, so that the variances cancel.

Nevertheless, for some model selection problems, the PML criterion may be the best choice. For example, the Bayesian integral (2.8) cannot be evaluated analytically for many model families. It is then often approximated by, for example, Markov Chain Monte Carlo methods, and it is not at all clear whether the re-

sulting procedure will show better or worse performance than the PML criterion. Theoretical arguments [4] show that there are quite strong limits to how badly the PML criterion can behave in model selection. For example, whenever a finite or countably infinite set of parametric models (each containing an uncountable number of distributions) are being compared, and data are i.i.d. according to an element of one of the models, then the error probability of the PML criterion *must* go to 0. If the number of models is finite and they are non-nested, it must even go to 0 as $\exp(-cn)$ for some constant $c > 0$. The same holds for other criteria including BIC, but not necessarily for ML. The PML criterion may have slightly lower $c$ than other model selection procedures, but the ML criterion is guaranteed to fail (always select the most complex model) in cases such as regression with polynomials of varying degree, where the number of models being compared is nested and countably infinite. Thus, whereas in our setting the PML criterion performs somewhat worse (in the sense that more data are needed before the same quality of results is achieved) than the ML criterion, it is guaranteed to display reasonable behaviour over a wide variety of settings, in many of which the ML criterion fails utterly.

All in all, our results indicate that the PML criterion should be used with caution, and may exhibit worse performance than other selection criteria under misspecification.

### 2.5.2 ML/BIC

Beside known $\mu$ and PML, all criteria seem to share more or less the same error exponent. Nevertheless, they still show differences in bias. While we have to be careful to avoid over-interpreting our results, we find that the ML/BIC criterion consistently displays the largest bias in favour of the Poisson model. Figure 2.3 shows how the Poisson model is *always* at least $10^{0.7} \approx 5$ times more likely according to ML/BIC than according to known $\mu$, regardless whether data were sampled from a geometric or a Poisson distribution. Figure 2.7 contains further evidence of bias in favour of the Poisson model: together with the PML criterion, the ML/BIC criterion exhibited the worst calibration performance: when the probability that the data is Poisson distributed is assessed by the ML criterion to be around 0.5, the real frequency of the data being Poisson distributed is only about 0.2.

This illustrates how the Poisson model appears to have a greater descriptive power, even though the two models have the same number of parameters, an observation which we hinted at in Section 2.2. Intuitively, the Poisson model allows more information about the data to be stored in the parameter estimate. All the other selection criteria compensate for this effect, by giving a higher probability to the Geometric model.

### 2.5.3 Basic Restricted ANML

We have seen that the ML/BIC criterion shows the largest bias for the Poisson model. Figure 2.3 shows that the second largest bias is achieved by ANML $\mu^* = 10$. Apparently the correction term that is applied by ANML criterion is not sufficient if we choose $\mu^* = 10$. However, we can obtain any correction term we like since we observed in Section 2.3.2 that ANML is equivalent to a GLRT with a selection threshold that is an unbounded, monotonically increasing function of $\mu^*$. Essentially, by choosing an appropriate $\mu^*$ we can get *any* correction in favour of the Geometric model, even one that would lead to a very large bias in the direction of the Geometric model. We conclude that it does not really make sense to use a fixed restricted parameter domain to repair the NML model when it does not exist, unless prior knowledge is available.

### 2.5.4 Objective Bayes and Two-part Restricted ANML

We will not try to interpret the differences in error probability for the (approximated) Bayesian and ANML 2-part criteria. Since we are using different selection criteria we should expect at least some differences in the results. These differences are exaggerated by our setup with its low mean and small sample size.

The Bayesian criterion, as well as its approximation appear to be somewhat better calibrated than the two-part ANML but the evidence is too thin to draw any strong conclusions.

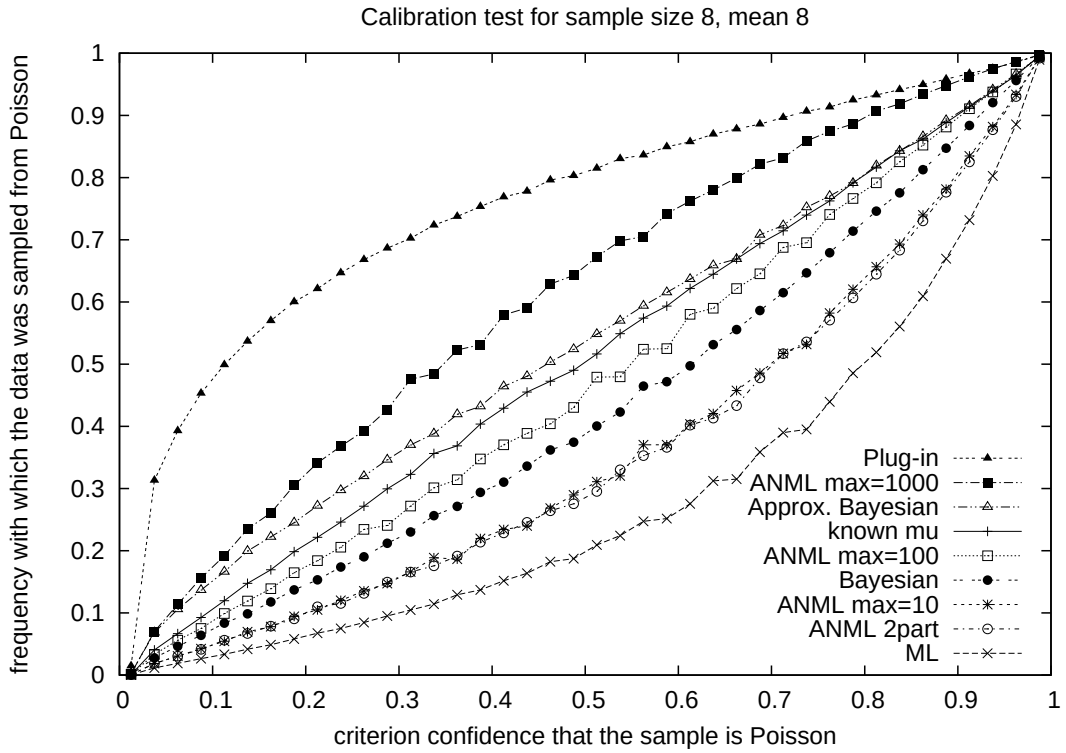
Figures 2.4–2.6 show that the error probability for these criteria tends to decrease at a slightly lower rate than for known $\mu$ (except when the prior on the generating distribution is heavily in favour of Poisson). While we do not understand this phenomenon well enough so as to prove it mathematically, it is of course consistent with the general rule that with more prior uncertainty, more data are needed to make the right decision. It may be that all the information contained within a sample can be used to improve the resolution of the known $\mu$ criterion, while for the other criteria some of that information has to be sacrificed in order to estimate the parameter value.

## 2.6 Summary and Conclusion

We have experimented with a number of model selection criteria which are based on the MDL philosophy and involve computing the code length of the data with the help of the model. There are several ways to define such codes, but the preferred method, the Normalised Maximum Likelihood (NML) code, cannot be applied since it does not exist for the Poisson and Geometric models that we consider.

We have experimented with the following alternative ways of working around this problem: (1) using BIC which is a simplification of approximated NML

**Figure 2.7** Calibration: probability that the model assigned to the data being Poisson against the frequency with which it actually was Poisson.




(ANML), (2) ANML with a restricted parameter range, this range being either fixed or encoded separately, (3) a Bayesian model using Jeffreys' prior, which is improper for the case at hand but which can be made proper by conditioning on the first outcome of the sample, (4) its approximation and (5) a PML code which always codes the new outcome using the distribution indexed by the maximum likelihood estimator for the preceding outcomes.

Only the NML code incurs the same regret for all possible data sequences $x^n$; the regret incurred by the alternatives we studied necessarily depends on the data. Arguably this dependence is quite weak for the objective Bayesian approach (see [39] for more details); how the regret depends on the data is at present rather unclear for the other approaches. As such they cannot really be viewed as "objective" alternatives to the NML code. However, new developments in the field make this distinction between "objective" and "subjective" codes seem a bit misleading. It is probably better to think in terms of "luckiness" (also see Section 1.1.1): while the NML code allows equally good compression (and thus learning speed) for all data sequences, other methods are able to learn faster from some data sequences than from others. This does not mean that conclusions drawn from inference with such luckiness codes are necessarily subjective.

We have performed error probability tests, bias tests and calibration tests to study the properties of the model selection criteria listed above and made the following observations.

Both BIC and ANML with a fixed restricted parameter range define a GLRT test and can be interpreted as methods to choose an appropriate threshold. BIC chooses a neutral threshold, so the criterion is biased in favour of the model which is most susceptible to overfitting. We found that even though both models under consideration have only one parameter, a GLRT with neutral threshold tends to be biased in favour of Poisson. ANML implies a threshold that counteracts this bias, but for every such threshold value there exists a corresponding parameter range, so it does not provide any more specific guidance in selecting that threshold. If the parameter range is separately encoded, this problem is avoided and the resulting criterion behaves competitively.

The Bayesian criterion displays reasonable performance both on the error rate experiments and the calibration test. The Bayesian universal codes for the models are not redundant and admit an MDL interpretation as minimising worst-case code length in an expected sense (Section 2.3.6).

The most surprising result is the behaviour of the PML criterion. It has a bias in favour of the Geometric model that depends strongly on the sample size. As a consequence, compared to the other model selection criteria its error rate decreases more slowly in the sample size if the data are sampled from each of the models with positive probability. This observation has led to a theoretical analysis of the code length of the PML code in the next chapter. It turns out that the regret of the PML code does not necessarily grow with $\frac{k}{2} \ln n$ like the NML and Bayesian codes do, if the sample is not distributed according to any element of the model.

# Chapter 3

# Behaviour of Prequential Codes under Misspecification

Universal coding lies at the basis Rissanen's theory of MDL (minimum description length) learning [4, 39] and Dawid's theory of prequential model assessment [26]. It also underlies on-line prediction algorithms for data compression and gambling purposes. In the introductory chapter (Section 1.2.3), we defined universality of a code in terms of its worst-case regret. Roughly, a code is universal with respect to a model $\mathcal{M}$ if it achieves small worst-case regret: it allows one to encode data using not many more bits than the optimal code in $\mathcal{M}$. We also described the four main universal codes: the *Shtarkov* or *NML* code, the *Bayesian mixture* code, the *2-part MDL code* and the *prequential maximum likelihood code* (PML), also known as the "ML plug-in code" or the "predictive MDL code" [4, 38]. This code was introduced independently by Rissanen [69] and by Dawid [26], who proposed it as a forecasting strategy rather than as a code. In this chapter we study the behaviour of the PML code if the considered model $\mathcal{M}$ does not contain the data-generating distribution $P^*$. We require that the model is a 1-parameter exponential family, but our results can possibly be extended to models with more parameters.

Instead of the worst-case regret, we analyse the *redundancy*, a closely related concept. We find that the redundancy of PML can be quite different from that of the other main universal codes. For all these codes, the redundancy is $\frac{1}{2}c \ln n + O(1)$ for some $c$, but while $c = 1$ for Bayes, NML and 2-part codes (under regularity conditions on $P^*$ and $\mathcal{M}$), we show here that for PML, any $c > 0$ is possible, depending on $P^*$ and $\mathcal{M}$.

There are a plethora of results concerning the redundancy and/or the regret for PML, for a large variety of models including multivariate exponential families, ARMA processes, regression models and so on. Examples are [70, 44, 97, 57]. In all these papers it is shown that either the regret or the redundancy grows as $\frac{k}{2} \ln n + o(\ln n)$, either in expectation or almost surely. Thus, these results already indicate that $c = 1$ for those models. The reason that these results do

not contradict ours, is that they invariably concern the case where the generating $P^*$ is in $\mathcal{M}$, so that automatically $\mathrm{var}_{M^*}(X) = \mathrm{var}_{P^*}(X)$.

As discussed in Section 3.4, the result has interesting consequences for parameter estimation and practical data compression, but the most important and surprising consequence is for MDL learning and model selection, where our result implies that PML may behave suboptimally *even if one of the models under consideration is correct!*

In Section 3.1 we informally state and explain our result. Section 3.2 contains the formal statement of our main result (Theorem 3.2.3), as well as a proof. In Section 3.3 we show that our results remain valid to some extent if "redundancy" is replaced by "expected regret" (Theorem 3.3.1). We discuss further issues, including relevance of the result, in Section 3.4. Section 3.5 states and proves various lemmas needed in the proofs of Theorems 3.2.3 and 3.3.1.

## 3.1 Main Result, Informally

Suppose $\mathcal{M} = \{P_\theta \ : \ \theta \in \Theta\}$ is a $k$-dimensional parametric family of distributions, and $Z_1, Z_2, \ldots$ are i.i.d. according to some distribution $P^* \in \mathcal{M}$. A code is universal for $\mathcal{M}$ if it is almost as efficient at coding outcomes from $P^*$ as the best element of $\mathcal{M}$. (As in previous chapters, we sometimes use codes and distributions interchangeably.) In the introductory chapter we measured the overhead incurred on the first $n$ outcomes $z^n = z_1, \ldots, z_n$ in terms of the worst-case regret

$$\max_{z^n} \mathcal{R}(L, \mathcal{M}, z^n) \quad = \quad \max_{z^n} \left( L(z^n) - \inf_{L' \in \mathcal{M}} L'(z^n) \right),$$

but in this chapter we consider the redundancy instead. We define the *redundancy* of a distribution $Q$ with respect to a model $\mathcal{M}$ as

$$\mathfrak{R}(P^*, Q, \mathcal{M}, n) \quad := \quad \underset{Z^n \sim P^*}{E}[-\ln Q(Z^n)] - \inf_{\theta \in \Theta} \underset{Z^n \sim P^*}{E}[-\ln P_\theta(Z^n)], \qquad (3.1)$$

where we use nats rather than bits as units of information to simplify equations. We omit the first three arguments of the redundancy if they are clear from context. These and other notational conventions are detailed in Section 3.2. The redundancy is a close lower bound on the *expected* regret, see Section 3.3. We do not know the exact relationship to worst-case regret.

The four major types of universal codes, Bayes, NML, 2-part and PML, all achieve redundancies that are (in an appropriate sense) close to optimal. Specifically, under regularity conditions on $\mathcal{M}$ and its parameterisation, these four types of universal codes all satisfy

$$\mathfrak{R}(P^*, Q, \mathcal{M}, n) = \frac{k}{2} \ln n + O(1), \qquad (3.2)$$

where the $O(1)$ may depend on $P^*$, $\mathcal{M}$ and the universal code $Q$ that is used. This is the famous "$k$ over 2 log $n$ formula", refinements of which lie at the basis of most practical approximations to MDL learning, see [38].

Often, the source distribution $P^*$ is assumed to be an element of the model. If such is the case, then by the information inequality [25] the second term of (3.1) is minimised for $P_\theta = P^*$, so that

$$\mathfrak{R}(P^*, Q, \mathcal{M}, n) = E_{P^*}[-\ln Q(Z^n)] - E_{P^*}[-\ln P^*(Z^n)]. \tag{3.3}$$

Thus, (3.3) can be interpreted as the expected number of additional nats one needs to encode $n$ outcomes if one uses the code corresponding to $Q$ instead of the optimal (Shannon-Fano) code with lengths $-\ln P^*(Z^n)$. For a good universal code this quantity is small for all or most $P^* \in \mathcal{M}$.

In this chapter we consider the case where the data are i.i.d. according to an *arbitrary* $P^*$ not necessarily in the model $\mathcal{M}$. It is now appropriate to rename the redundancy to *relative* redundancy, since we measure the number of nats we lose compared to the best element in the model, rather than compared to the generating distribution $P^*$. The definition (3.1) remains unchanged. It can no longer be rewritten as (3.3) however: Assuming it exists and is unique, let $P_{\theta^*}$ be the element of $\mathcal{M}$ that minimises KL divergence to $P^*$:

$$\theta^* \quad := \quad \arg\min_{\theta\in\Theta} D(P^*\|P_\theta) \quad = \quad \arg\min_{\theta\in\Theta} E_{P^*}[-\ln P_\theta(Z)],$$

where the equality follows from the definition of the KL divergence [25]. Then the relative redundancy satisfies

$$\mathfrak{R}(P^*, Q, \mathcal{M}, n) = E_{P^*}[-\ln Q(Z^n)] - E_{P^*}[-\ln P_{\theta^*}(Z^n)]. \tag{3.4}$$

It turns out that for the NML, 2-part MDL and Bayes codes, the relative redundancy (3.4) with $P^* \notin \mathcal{M}$, still satisfies (3.2), under some conditions on $\mathcal{M}$ and $P^*$ (Section 3.3). In this chapter we show that (3.2) does *not* hold for PML. The PML code with length function $L$ works by sequentially predicting $Z_{i+1}$ using a (slightly modified) ML or Bayesian MAP estimator $\hat\theta_i = \hat\theta(z^i)$ based on the past data, that is, the first $i$ outcomes $z^i$. The total code length $L(z^n)$ on a sequence $z^n$ is given by the sum of the individual "predictive" code lengths (log losses): $L(z^n) = \sum_{i=0}^{n-1}[-\ln P_{\hat\theta_i}(z_{i+1})]$. In our main theorem, we show that if and $\mathcal{M}$ is a regular one-parameter exponential family ($k = 1$), then

$$\mathfrak{R}(P^*, Q, \mathcal{M}, n) = \tfrac{1}{2}\frac{\mathrm{var}_{P^*}X}{\mathrm{var}_{P_{\theta^*}}X}\ln n + O(1), \tag{3.5}$$

where $X$ is the sufficient statistic of the family. Example 6 below illustrates the phenomenon. Note that if $P^* \in \mathcal{M}$, then $P_{\theta^*} = P^*$ and (3.5) becomes the familiar expression. The result holds as long as $\mathcal{M}$ and $P^*$ satisfy a mild condition that is stated and discussed in the next section. Section 3.4 discusses the consequences of

this result for compression, estimation and model selection, as well as its relation to the large body of earlier results on PML coding.

**Example 6.** Let $\mathcal{M}$ be the family of Poisson distributions, parameterised by their mean $\mu$. Since neither the NML universal code nor Jeffreys' prior are defined for this model it is attractive to use the PML code as a universal code for this model. The ML estimator $\hat{\mu}_i$ is the empirical mean of $z_1, \ldots, z_i$.

Suppose $Z$, $Z_1$, $Z_2$, ... are i.i.d. according to a degenerate $P$ with $P(Z = 4) = 1$. Since the sample average is a sufficient statistic for the Poisson family, $\hat{\mu}_i$ will be equal to 4 for all $i \geq 1$. On the other hand, $\mu^*$, the parameter (mean) of the distribution in $\mathcal{M}$ closest to $P$ in KL-divergence, will be equal to 4 as well. Thus the relative redundancy (3.4) of the PML code is given by

$$\mathfrak{R}(P^*, Q, \mathcal{M}, n) = -\ln P_{\hat{\mu}_0}(4) + \ln P_4(4) + \sum_{i=1}^{n-1}[-\ln P_4(4) + \ln P_4(4)] = O(1),$$

assuming an appropriate definition of $\hat{\mu}_0$. In the case of the Poisson family, we have $Z = X$ in (3.5). Thus, since $\mathrm{var}_{P^*} Z = 0$, this example agrees with (3.5).

Now suppose data are i.i.d. according to some $P_\tau$, with $P_\tau(Z = z) \propto (z+1)^{-3}$ for all $z$ smaller than $\tau$, and $P_\tau(Z = z) = 0$ for $z \geq \tau$. It is easy to check that, for $\tau \to \infty$, the entropy of $P_\tau$ converges to a finite constant, but the variance of $P_\tau$ tends to infinity. Thus, by choosing $\tau$ large enough, the redundancy obtained by the Poisson PML code can be made to grow as $c \log n$ for arbitrarily large $c$.

**Example 7.** The Hardy-Weinberg model deals with genotypes of which the alleles are assumed independently Bernoulli distributed according to some parameter $p$. There are four combinations of alleles, usually denoted "aa", "AA", "aA", "Aa"; but since "aA" and "Aa" result in the same genotype, the Hardy-Weinberg model is defined as a probability distribution on three outcomes. We model this by letting $X$ be a random variable on the underlying space, that maps "aA" and "Aa" to the same value: $X(aa) = 0$, $X(aA) = X(Aa) = \frac{1}{2}$ and $X(AA) = 1$. Then $P(X = 0) = (1-p)^2$, $P(X = \frac{1}{2}) = 2p(1-p)$ and $P(X = 1) = p^2$. The Hardy-Weinberg model is an exponential family with sufficient statistic $X$. To see this, note that for any parameter $p \in [0, 1]$, we have $EX = \mu = P(A) = p$, so we can parameterise the model by the mean of $X$. The variance of the distribution with parameter $\mu$ is $\frac{1}{2}\mu(1-\mu)$. Now suppose that we code data in a situation where the Hardy-Weinberg model is *wrong* and the genotypes are in fact distributed according to $P(X = \frac{1}{2}) = P(X = 1) = \frac{1}{2}$ and $P(X = 0) = 0$, such that mean and variance of $X$ are $\frac{3}{4}$ and $\frac{2}{32}$ respectively. The closest distribution in the model has the same mean (since the mean is a sufficient statistic), and variance $\frac{3}{32}$. Thus PML will achieve a redundancy of $\frac{1}{3}\ln n$ rather than $\frac{1}{2}\ln n$ (up to $O(1)$).

## 3.2 Main Result, Formally

In this section, we define our quantities of interest and we state and prove our main result. Throughout this text we use nats rather than bits as units of information. Outcomes are capitalised if they are to be interpreted as random variables instead of instantiated values. Let $P^*$ be a distribution on some set $\mathcal{Z}$, which can be either finite or countably infinite, or a subset of $k$-dimensional Euclidean space for some $k \geq 1$. A sequence of outcomes $z_1, \ldots, z_n$ is abbreviated to $z^n$. Let $X : \mathcal{Z} \to \mathbb{R}^k$ be a random vector. We write $E_{P^*}[X]$ as a shorthand for $E_{X \sim P^*}[X]$. When we consider a sequence of $n$ outcomes independently distributed $\sim P^*$, we even use $E_{P^*}$ as a shorthand for the expectation of $(X_1, \ldots, X_n)$ under the $n$-fold product distribution of $P^*$. Finally, $P^*(X)$ denotes the probability mass function of $P^*$ in case $X$ is discrete-valued, and the density of $P^*$ in case $X$ takes values in a continuum. When we write "density function of $X$", then, if $X$ is discrete-valued, this should be read as "probability mass function of $X$". Note however that in our main result, Theorem 3.2.3 below, we do not assume that the data-generating distribution $P^*$ admits a density.

We define the particular random vector $Z(z) := z$. Let $X : \mathcal{Z} \to \mathbb{R}$ be a random variable on $\mathcal{Z}$, and let $\mathcal{X} = \{x \in \mathbb{R} \ : \ \exists z \in \mathcal{Z} : X(z) = x\}$ be the range of $X$. Exponential family models are families of distributions on $\mathcal{Z}$ defined relative to a random variable $X$ (called "sufficient statistic") as defined above, and a function $h : \mathcal{Z} \to (0, \infty)$. Let $\mathrm{Z}(\eta) := \int_{z \in \mathcal{Z}} e^{-\eta X(z)} h(z) dz$ (the integral to be replaced by a sum for countable $\mathcal{Z}$), and $\Theta_\eta := \{\eta \in \mathbb{R} : \mathrm{Z}(\eta) < \infty\}$.

**Definition 3.2.1** (Exponential family)**.** The *single parameter exponential family* [48] with *sufficient statistic* $X$ and *carrier* $h$ is the family of distributions with densities $P_\eta(z) := \frac{1}{\mathrm{Z}(\eta)} e^{-\eta X(z)} h(z)$, where $\eta \in \Theta_\eta$. $\Theta_\eta$ is called the *natural parameter space*. The family is called *regular* if $\Theta_\eta$ is an open interval of $\mathbb{R}$.

In the remainder of this text we only consider single parameter, regular exponential families with a 1-to-1 parameterisation; this qualification will henceforth be omitted. Examples include the Poisson, geometric and multinomial families, and the model of all Gaussian distributions with a fixed variance or mean. In the first four cases, we can take $X$ to be the identity, so that $X = Z$ and $\mathcal{X} = \mathcal{Z}$. In the case of the normal family with fixed mean, $\sigma^2$ becomes the sufficient statistic and we have $\mathcal{Z} = \mathbb{R}$, $\mathcal{X} = [0, \infty)$ and $X = Z^2$.

The statistic $X(z)$ is sufficient for $\eta$ [48]. This suggests reparameterising the distribution by the expected value of $X$, which is called the *mean value parameterisation*. The function $\mu(\eta) = E_{P_\eta}[X]$ maps parameters in the natural parameterisation to the mean value parameterisation. It is a diffeomorphism (it is one-to-one, onto, infinitely often differentiable and has an infinitely often differentiable inverse) [48]. Therefore the mean value parameter space $\Theta_\mu$ is also an open interval of $\mathbb{R}$. We note that for some models (such as Bernoulli and Poisson), the parameter space is usually given in terms of the a non-open set

of mean-values (e.g., $[0, 1]$ in the Bernoulli case). To make the model a regular exponential family, we have to restrict the set of parameters to its own interior. Henceforth, whenever we refer to a standard statistical model such as Bernoulli or Poisson, we assume that the parameter set has been restricted in this sense.

We are now ready to define the PML distribution. This is a distribution on infinite sequences $z_1, z_2, \ldots \in \mathcal{Z}^\infty$, recursively defined in terms of the distributions of $Z_{n+1}$ conditioned on $Z^n = z^n$, for all $n = 1, 2, \ldots$, all $z^n = (z_1, \ldots, z_n) \in \mathcal{Z}^n$. In the definition, we use the notation $x_i := X(z_i)$.

**Definition 3.2.2** (PML distribution)**.** Let $\Theta_\mu$ be the mean value parameter domain of an exponential family $\mathcal{M} = \{P_\mu \mid \mu \in \Theta_\mu\}$. Given $\mathcal{M}$ and constants $x_0 \in \Theta_\mu$ and $n_0 > 0$, we define the *PML distribution* $U$ by setting, for all $n$, all $z^{n+1} \in \mathcal{Z}^{n+1}$:

$$U(z_{n+1} \mid z^n) = P_{\hat{\mu}(z^n)}(z_{n+1}),$$

where $U(z_{n+1} \mid z^n)$ is the density/mass function of $z_{n+1}$ conditional on $Z^n = z^n$,

$$\hat{\mu}(z^n) := \frac{x_0 \cdot n_0 + \sum_{i=1}^n x_i}{n + n_0},$$

and $P_{\hat{\mu}(z^n)}(\cdot)$ is the density of the distribution in $\mathcal{M}$ with mean $\hat{\mu}(z^n)$.

We henceforth abbreviate $\hat{\mu}(z^n)$ to $\hat{\mu}_n$. We usually refer to the PML distribution in terms of the corresponding code length function

$$L_U(z^n) = \sum_{i=0}^{n-1} L_U(z_{i+1} \mid z^i) = \sum_{i=0}^{n-1} -\ln P_{\hat{\mu}_i}(z_{i+1}).$$

To understand this definition, note that for exponential families, for any sequence of data, the ordinary maximum likelihood parameter is given by the average $n^{-1} \sum x_i$ of the observed values of $X$ [48]. Here we define our PML distribution in terms of a slightly modified maximum likelihood estimator that introduces a "fake initial outcome" $x_0$ with multiplicity $n_0$ in order to avoid infinite code lengths for the first few outcomes (a well-known problem called by Rissanen the "inherent singularity" of predictive coding [71, 36]) and to ensure that the probability of the first outcome is well-defined for the PML distribution. In practice we can take $n_0 = 1$ but our result holds for any $n_0 > 0$. The justification of our modification to the ML estimator is discussed further in Section 3.4.

**Theorem 3.2.3** (Main result)**.** *Let $X, X_1, X_2, \ldots$ be i.i.d. $\sim P^*$, with $E_{P^*}[X] = \mu^*$. Let $\mathcal{M}$ be a single parameter exponential family with sufficient statistic $X$ and $\mu^*$ an element of the mean value parameter space. Let $U$ denote the PML distribution with respect to $\mathcal{M}$. If $\mathcal{M}$ and $P^*$ satisfy Condition 3.2.4 below, then*

$$\mathfrak{R}(P^*, U, \mathcal{M}, n) = \tfrac{1}{2} \frac{\mathrm{var}_{P^*} X}{\mathrm{var}_{P_{\mu^*}(X)}} \ln n + O(1). \tag{3.6}$$

Comparing this to (3.5), note that $P_{\mu^*}$ is the element of $\mathcal{M}$ achieving smallest expected code length, i.e. it achieves $\inf_{\mu\in\Theta_\mu} D(P^*\|P_\mu)$ [48].

**Condition 3.2.4.** *We require that the following holds both for $T := X$ and $T := -X$:*

- *If $T$ is unbounded from above then there is a $k \in \{4, 6, \ldots\}$ such that the first $k$ moments of $T$ exist under $P^*$ and that $\frac{\mathrm{d}^4}{\mathrm{d}\mu^4} D(P_{\mu^*}\|P_\mu) = O\left(\mu^{k-6}\right)$.*
- *If $T$ is bounded from above by a constant $g$ then $\frac{\mathrm{d}^4}{\mathrm{d}\mu^4} D(P_{\mu^*}\|P_\mu)$ is polynomial in $1/(g-\mu)$.*

Roughly, this condition expresses a trade-off between the data generating distribution $P^*$ and the model. If the model is well-behaved, in the sense that the fourth order derivative of the KL divergence does not grow too fast with the parameter, then we do not require many moments of $P^*$ to exist. Vice versa if the model is not well-behaved, then the theorem only holds for very specific $P^*$, of which many moments exist.

The condition holds for most single-parameter exponential families that are relevant in practice. To illustrate, in Figure 3.2 we give the fourth derivative of the divergence for a number of common exponential families explicitly. All parameters beside the mean are treated as fixed values. Note that to interpret the mean 0 normal distributions as a 1-parameter exponential family the density we had to set $X(z) = z^2$, so that its mean $E[X]$ is actually the variance $E[Z^2]$ of the normal distribution. As can be seen from the figure, for these exponential families, our condition applies whenever at least the first four moments of $P^*$ exist: a quite weak condition on the data generating distribution.

*Proof of Theorem 3.2.3.* For exponential families, we have

$$\begin{aligned} & E_{P^*}[-\ln P_\mu(Z)] - E_{P^*}[-\ln P_{\mu'}(Z)] \\ = \;& \eta(\mu)E_{P^*}[X(Z)] + \ln \mathrm{Z}(\eta(\mu)) + E_{P^*}[-\ln h(Z)] \\ - \;& \eta(\mu')E_{P^*}[X(Z)] - \ln \mathrm{Z}(\eta(\mu')) - E_{P^*}[-\ln h(Z)] \\ = \;& E_{P^*}[-\ln P_\mu(X)] - E_{P^*}[-\ln P_{\mu'}(X)], \end{aligned}$$

so that $\mathfrak{R}(P^*, U, \mathcal{M}, n) = E_{P^*}[L_U(X^n)] - \inf_\mu E_{P^*}[-\ln P_\mu(X^n)]$. This means that relative redundancy, which is the sole quantity of interest in the proof, depends only on the sufficient statistic $X$, not on any other aspect of the outcome that may influence $Z$. Thus, in the proof of Theorem 3.2.3 as well as all the Lemmas and Propositions it makes use of, we will never mention $Z$ again. Whenever we refer to a "distribution" we mean a distribution of random variable $X$, and we also think of the data generating distribution $P^*$ in terms of the distribution it induces on $X$ rather than $Z$. Whenever we say "the mean" without further qualification, we refer to the mean of the random variable $X$. Whenever we

**Figure 3.1** $\frac{\mathrm{d}^4}{\mathrm{d}\mu^4} D(P_{\mu^*} \| P_\mu)$ for a number of exponential families.

| | $P_{\mu^*}(x)$ | $\frac{\mathrm{d}^4}{\mathrm{d}\mu^4} D(P_{\mu^*} \| P_\mu)$ |
|---|---|---|
| Bernoulli | $(\mu^*)^x (1-\mu^*)^{(1-x)}$ | $\frac{6\mu^*}{\mu^4} + \frac{6(1-\mu^*)}{(1-\mu)^4}$ |
| Poisson | $\frac{e^{\mu^*} \mu^{*x}}{x!}$ | $\frac{6\mu^*}{\mu^4}$ |
| Geometric | $\theta^x(1-\theta) = \frac{(\mu^*)^x}{(\mu^*+1)^{x+1}}$ | $\frac{6\mu^*}{\mu^4} - \frac{6(\mu^*+1)}{(\mu+1)^4}$ |
| Exponential | $\frac{1}{\mu^*} e^{-x/\mu^*}$ | $-\frac{6}{\mu^4} + \frac{24\mu^*}{\mu^5}$ |
| Normal (fixed mean = 0) | $\frac{1}{\sqrt{2\pi\mu^* x}} e^{-\frac{x}{2\mu^*}}$ | $-\frac{3}{\mu^4} + \frac{12\mu^*}{\mu^5}$ |
| Normal (fixed variance = 1) | $\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\mu^*)^2}$ | $0$ |
| Pareto | $\frac{ab^a}{x^{a+1}}$ for $b = \frac{a-1}{a}\mu^*$ | $\frac{6a}{\mu^4}$ |

refer to the Kullback-Leibler (KL) divergence between $P$ and $Q$, we refer to the KL divergence between the distributions they induce for $X$. The reader who is confused by this may simply restrict attention to exponential family models for which $Z = X$, and consider $X$ and $Z$ identical.

The proof refers to a number of theorems and lemmas which will be developed in Section 3.5. In the statement of all these results, we assume, as in the statement of Theorem 3.2.3, that $X, X_1, X_2, \ldots$ are i.i.d. $\sim P^*$ and that $\mu^* = E_{P^*}[X]$. If $X$ takes its values in a countable set, then all integrals in the proof should be read as the corresponding sums.

The redundancy can be rewritten further as the sum of the expected risk for each outcome (Lemma 3.5.6). We obtain

$$\mathfrak{R}(P^*, U, \mathcal{M}, n) = \sum_{i=0}^{n-1} \underset{\hat{\mu}_i \sim P^*}{E} \left[ D(P_{\mu^*} \| P_{\hat{\mu}_i}) \right]. \tag{3.7}$$

Here, the estimator $\hat{\mu}_i$ is a random variable that takes on values according to $P^*$, while the optimal parameter value $\mu^*$ is fixed (and determined by $P^*$). We will write $D(\mu^* \| \hat{\mu}_i)$ as shorthand notation for $P_{\mu^*}$ and $P_{\hat{\mu}_i}$.

We now first rewrite the divergence. We abbreviate $\delta_i := \hat{\mu}_i - \mu^*$ and $D^{(k)}(\mu) := \frac{d^k}{d\mu^k} D(P_{\mu^*} \| P_\mu)$. That is, $D^{(k)}(\mu)$ is the $k$-th derivative of the function $f(\mu) := D(P_{\mu^*} \| P_\mu)$. Taylor-expansion of the divergence around $\mu^*$ yields

$$D(P_{\mu^*} \| P_{\hat{\mu}_i}) = 0 + \delta_i D^{(1)}(\mu^*) + \frac{{\delta_i}^2}{2} D^{(2)}(\mu^*) + \frac{{\delta_i}^3}{6} D^{(3)}(\mu^*) + \frac{{\delta_i}^4}{24} D^{(4)}(\ddot{\mu}_i).$$

The last term is the remainder term of the Taylor expansion, in which $\ddot{\mu}_i \in [\mu^*, \hat{\mu}_i]$. The second term $D^{(1)}(\mu^*)$ is zero, since $D(\mu^* \| \mu)$ reaches its minimum at $\mu = \mu^*$. For the third term we observe that

$$D^{(2)}(\mu) = \frac{\mathrm{d}^2}{\mathrm{d}\mu^2} E[\ln P_{\mu^*}(X) - \ln P_{\mu}(X)] = -\frac{\mathrm{d}^2}{\mathrm{d}\mu^2} E[\ln P_{\mu}(X)],$$

which is equal to the Fisher information. Fisher information is defined as $I(\theta) := E\left[(\frac{\mathrm{d}}{\mathrm{d}\theta} \ln f(X \mid \theta))^2\right]$, but as is well known [48], for exponential families this is equal to $-\frac{\mathrm{d}^2}{\mathrm{d}\theta^2} E\left[\ln f(X \mid \theta)\right]$, which matches $D^{(2)}(\cdot)$ exactly. Furthermore, for the mean value parameterisation $I(\mu) = 1/\mathrm{var}_{P_\mu}(X)$. We obtain

$$D(P_{\mu^*} \| P_{\hat{\mu}_i}) = \frac{1}{2} {\delta_i}^2 / \mathrm{var}_{P_{\mu^*}}(X) + \frac{1}{6} {\delta_i}^3 D^{(3)}(\mu^*) + \frac{1}{24} {\delta_i}^4 D^{(4)}(\ddot{\mu}_i). \tag{3.8}$$

We plug this expression back into (3.7), giving

$$\mathfrak{R}(P^*, U, \mathcal{M}, n) = \frac{1}{2\mathrm{var}_{P_{\mu^*}}(X)} \sum_{i=0}^{n-1} E_{P^*}\left[{\delta_i}^2\right] + R(n), \tag{3.9}$$

where the remainder term $R(n)$ is given by

$$R(n) = \sum_{i=0}^{n-1} E_{P^*}\left[\frac{1}{6} {\delta_i}^3 D^{(3)}(\mu^*) + \frac{1}{24} {\delta_i}^4 D^{(4)}(\ddot{\mu}_i)\right], \tag{3.10}$$

and where $\mu$ and $\delta_i$ are random variables; note that although $\mu$ is not indexed it does depend on the index $i$. In Lemma 3.5.8 we show that $R(n) = O(1)$, giving:

$$\mathfrak{R}(P^*, U, \mathcal{M}, n) = O(1) + \frac{1}{2\mathrm{var}_{P_{\mu^*}}(X)} \sum_{i=0}^{n-1} E_{P^*}\left[(\hat{\mu}_i - \mu^*)^2\right]. \tag{3.11}$$

Note that $\hat{\mu}_i$ is almost the ML estimator. This suggests that each term in the sum of (3.11) should be almost equal to the variance of the ML estimator, which is $\mathrm{var}X/i$. Because of the slight modification that we made to the estimator, we get a correction term of $O(i^{-2})$ as established in Theorem 3.5.2:

$$\begin{aligned} \sum_{i=0}^{n-1} E_{P^*}\left[(\hat{\mu}_i - \mu^*)^2\right] &= \sum_{i=0}^{n-1} O\left((i+1)^{-2}\right) + \mathrm{var}_{P^*}(X) \sum_{i=0}^{n-1} (i+1)^{-1} \\ &= O(1) + \mathrm{var}_{P^*}(X) \ln n \end{aligned} \tag{3.12}$$

The combination of (3.11) and (3.12) completes the proof. □

## 3.3 Redundancy vs. Regret

The "goodness" of a universal code relative to a model $\mathcal{M}$ can be measured in several ways: rather than using redundancy (as we did here), one can also choose to measure code length differences in terms of *regret*, where one has a further choice between *expected regret* and *worst-case regret* [4]. Here we only discuss the implications of our result for the expected regret measure.

Let $\mathcal{M} = \{P_\theta \mid \theta \in \Theta\}$ be a family of distributions parameterised by $\Theta$. Given a sequence $z^n = z_1, \ldots, z_n$ and a universal code $U$ for $\mathcal{M}$ with lengths $L_U$, the *regret* of $U$ on sequence $z^n$ is defined as

$$L_U(z^n) - \inf_{\theta\in\Theta}[-\ln P_\theta(z^n)]. \tag{3.13}$$

Note that if the (unmodified) ML estimator $\hat{\theta}(z^n)$ exists, then this is equal to $L_U(z^n) + \ln P_{\hat{\theta}(z^n)}(z^n)$. Thus, one compares the code length achieved on $z^n$ by $U$ to the best possible that could have been achieved on that particular $z^n$, using any of the codes/distributions in $\mathcal{M}$. Assuming $Z_1, Z_2, \ldots$ are i.i.d. according to some (arbitrary) $P$, one may now consider the expected regret

$$E_{Z^n\sim P^*}[\mathcal{R}(P^*, U, \mathcal{M}, n)] = E_{P^*}[L_U(Z^n) - \inf_{\theta\in\Theta}[-\ln P_\theta(Z^n)]].$$

To quantify the difference with redundancy, consider the function

$$d(n) := \inf_{\theta\in\Theta} E_P[-\ln P_\theta(Z^n)] - E_P\,[\inf_{\theta\in\Theta}[-\ln P_\theta(Z^n)]],$$

and note that for any universal code, $\mathfrak{R} - E[\mathcal{R}] = d(n)$. In case $P^* \in \mathcal{M}$, then under regularity conditions on $\mathcal{M}$ and its parameterisation, it can be shown [23] that

$$\lim_{n\to\infty} d(n) = \frac{k}{2}, \tag{3.14}$$

where $k$ is the dimension of $\mathcal{M}$. In our case, where $P^*$ is not necessarily in $\mathcal{M}$, we have the following :

**Theorem 3.3.1.** *Let $\mathcal{X}$ be finite. Let $P^*$, $P_\mu$ and $\mu^*$ be as in Theorem 3.2.3. Then*

$$\lim_{n\to\infty} d(n) = \frac{1}{2}\frac{\mathrm{var}_{P^*}X}{\mathrm{var}_{P_{\mu^*}}X}. \tag{3.15}$$

Once we are dealing with 1-parameter families, in the special case that $P^* \in \mathcal{M}$, this result reduces to (3.14). We suspect that, under a condition similar to Condition 3.2.4, the same result still holds for general, not necessarily finite or countable or bounded $\mathcal{X}$, but we do not know the details. In any case, our result is sufficient to show that in some cases (namely, if $\mathcal{X}$ is finite), we have

$$\mathcal{R}(P^*, U, \mathcal{M}, n) = \frac{1}{2}\frac{\mathrm{var}_{P^*}X}{\mathrm{var}_{P_{\mu^*}}X}\ln n + O(1),$$

so that, up to $O(1)$-terms, the redundancy and the regret of the prequential ML code behave in the same way.

Incidentally, we can use Theorem 3.3.1 to substantiate the claim in Section 3.1, which stated that the Bayes (equipped with a strictly positive differentiable prior), NML and 2-part codes still achieve relative redundancy of $\frac{1}{2}\ln n$ if $P^* \neq \mathcal{M}$, at least if $\mathcal{X}$ is finite. Let us informally explain why this is the case. It is easy to show that Bayes, NML and (suitably defined) 2-part codes achieve regret $\frac{1}{2}\ln n + O(1)$ for *all* sequences $z_1, z_2, \ldots$ such that $\hat{\theta}(z^n)$ is bounded away from the boundary of the parameter space $\Theta_\mu$, for all large $n$ [4, 38]. It then follows using, for example, the Chernoff bound that these codes must also achieve expected regret $\frac{1}{2}\ln n + O(1)$ for *all* distributions $P^*$ on $\mathcal{X}$ that satisfy $E_{P^*}[X] = \mu^* \in \Theta_\mu$. Theorem 3.3.1 then shows that they also achieve relative redundancy $\frac{1}{2}\ln n + O(1)$ for *all* distributions $P^*$ on $\mathcal{X}$ that satisfy $E_{P^*}[X] = \mu^* \in \Theta_\mu$. We omit further details.

## 3.4 Discussion

### 3.4.1 A Justification of Our Modification of the ML Estimator

A prequential code cannot be defined in terms of the ordinary ML estimator ($n_0 = 0$ in Definition 3.2.2) for two reasons. First, the ML estimator is undefined until the first outcome has been observed. Second, it may achieve infinite code lengths on the observed data. A simple example is the Bernoulli model. If we first observe $z_1 = 0$ and then $z_2 = 1$, the code length of $z_2$ according to the ordinary ML estimator of $z_2$ given $z_1$ would be $-\ln P_{\hat{\mu}(z_1)}(z_2) = -\ln 0 = \infty$. There are several ways to resolve this problem. We choose to add an "fake initial outcome". Another possibility that has been suggested (e.g., [26]) is to use the ordinary ML estimator, but to postpone using it until after $m$ outcomes have been observed, where $m$ is the smallest number such that $-\ln P_{\hat{\mu}(z^m)}(Z_{m+1})$ is guaranteed to be finite, no matter what value $Z_{m+1}$ is realized. The first $m$ outcomes may then be encoded by repeatedly using some code $L_0$ on outcomes of $\mathcal{Z}$, so that for $i \leq m$, the code length of $z_i$ does not depend on the outcomes $z^{i-1}$. In the Bernoulli example, one could for example use the code corresponding to $P(Z_i = 1) = 1/2$, until and including the first $i$ such that $z^i$ includes both a 0 and a 1. It then takes $i$ bits to encode the first $z^i$ outcomes, no matter what they are. After that, one uses the prequential code with the standard ML estimator. It is easy to see (by slight modification of the proof) that our theorem still holds for this variation of prequential coding. Thus, our particular choice for resolving the startup problem is not crucial to obtaining our result. The advantage of our solution is that, as we now show, it allows us to interpret our modified ML estimator as a Bayesian MAP and Bayesian mean estimator as well, thereby showing that the same behaviour

can be expected for such estimators.

### 3.4.2 Prequential Models with Other Estimators

An attractive property of our generalisation of the ML estimator is that it actually also generalises two other commonly used estimators, namely the Bayesian maximum a-posteriori and Bayesian mean estimators.

The Bayesian maximum a-posteriori estimator can always be interpreted as an ML estimator based on the sample and some additional "fake data", provided that a conjugate prior is used ([12]; see also the notion of *ESS (Equivalent Sample Size) Priors* discussed in, for example, [51]). Therefore, the prequential ML model as defined above can also be interpreted as a prequential MAP model for that class of priors, and the whole analysis carries over to that setting.

For the Bayesian mean estimator, the relationship is slightly less direct. However, it follows from the work of Hartigan [42, Chapter 7] on the so-called "maximum likelihood prior", that by slightly modifying conjugate priors, we can construct priors such that the Bayesian mean also becomes of the form of our modified ML estimator.

Our whole analysis thus carries over to prequential codes based on these estimators. In fact, we believe that our result holds quite generally:

**Conjecture 3.4.1.** *Let $\mathcal{M}$ be a regular exponential family with sufficient statistic $X$ and let $\mathcal{P}$ be the set of distributions on $\mathcal{Z}$ such that $E_{P^*}[X^4]$ exists. There exists* no "in-model" estimator *such that the corresponding prequential code achieves redundancy $\frac{1}{2}\ln n + O(1)$ for all $P^* \in \mathcal{P}$.*

Here, by an *in-model estimator* we mean an algorithm that takes as input any sample of arbitrary length and outputs a $P_\mu \in \mathcal{M}$. Let us contrast this with "out-model estimators": fix some prior on the parameter set $\Theta_\mu$ and let $P(\mu \mid z_1, \ldots, z_{n-1})$ be the Bayesian posterior with respect to this prior and data $z_1, \ldots, z_{n-1}$. One can think of the Bayesian predictive distribution $P(z_n \mid z_1, \ldots, z_{n-1}) := \int_{\mu \in \Theta_\mu} P_\mu(z_n) P(\mu \mid z_1, \ldots, z_{n-1}) d\mu$ as an *estimate* of the distribution of $Z$, based on data $z_1, \ldots, z_{n-1}$. But unlike estimators as defined in the conjecture above, the resulting *Bayesian predictive estimator* will in general not be a member of $\mathcal{M}$, but rather of its convex closure: we call it an *out-model estimator*. The redundancy of the Bayesian universal model is equal to the *accumulated Kullback-Leibler (KL) risk* of the Bayesian predictive estimator [36]. Thus, the accumulated KL risk of the Bayesian predictive estimator is $\frac{1}{2}\ln n + O(1)$ even under misspecification. Thus, if our conjecture above holds true, then in-model estimators behave in a fundamentally different way from out-model estimators in terms of their asymptotic risk.

**Example 8.** The well-known Laplace and Krichevsky-Trofimov estimators for the Bernoulli model [38] define PML distributions according to Definition 3.2.2:

they correspond to $x_0 = 1/2, n_0 = 2$, and $x_0 = 1/2, n_0 = 1$ respectively. Yet, they also correspond to Bayesian predictive distributions with uniform prior or Jeffreys' prior respectively. This implies that the code length achieved by the Bayesian universal model with Jeffreys' prior and the PML distribution with $x_0 = 1/2, n_0 = 1$ must *coincide*. We claimed before that the expected regret for a Bayesian universal model is $\frac{1}{2} \log n + O(1)$ if data are i.i.d. $\sim P^*$, for essentially all distributions $P^*$. This may seem to contradict our result which says that the expected regret of the PML distribution can be $0.5c \log n + O(1)$ with $c \neq 1$ if $P^* \notin \mathcal{M}$. But there really is no contradiction: since the Bernoulli model happens to contain *all* distributions $P^*$ on $\{0, 1\}$, we cannot have $P^* \notin \mathcal{M}$ so Theorem 1 indeed says that $c = 1$ no matter what $P^*$ we choose. But with more complicated models such as the Poisson or Hardy-Weinberg model, it is quite possible that $P^* \notin \mathcal{M}$. Then the Bayesian predictive distribution will *not* coincide with any PML distribution and we can have $c \neq 1$.

### 3.4.3 Practical Significance for Model Selection

As mentioned in the introduction, there are many results showing that in various contexts, if $P^* \in \mathcal{M}$, then the prequential ML code achieves optimal redundancy. These results strongly suggest that it is a very good alternative for (or at least approximation to) the NML or Bayesian codes in MDL model selection. Indeed, quoting Rissanen [71]:

> "If the encoder does not look ahead but instead computes the best parameter values from the past string, only, using an algorithm which the decoder knows, then no preamble is needed. The result is a *predictive* coding process, one which is quite different from the sum or integral formula in the stochastic complexity.[1] And it is only because of a certain inherent singularity in the process, as well as the somewhat restrictive requirement that the data must be ordered, that we do not consider the resulting predictive code length to provide another competing definition for the stochastic complexity, but rather regard it as an approximation."

Our result however shows that the prequential ML code may behave quite differently from the NML and Bayes codes, thereby strengthening the conclusion that it should not be taken as a definition of stochastic complexity. Although there is only a significant difference if data are distributed according to some $P^* \notin \mathcal{M}$, the difference is nevertheless very relevant in an MDL model selection context with disjoint models, even if one of the models under consideration *does* contain the "true" $P^*$. To see this, suppose we are comparing two models $\mathcal{M}_1$ and $\mathcal{M}_2$ for

[1] The stochastic complexity is the code length of the data $z_1, \ldots, z_n$ that can be achieved using the NML code.

the same data, and in fact, $P^* \in \mathcal{M}_1 \cup \mathcal{M}_2$. For concreteness, assume $\mathcal{M}_1$ is the Poisson family and $\mathcal{M}_2$ is the geometric family. We want to decide which of these two models best explains the data. According to the MDL Principle, we should associate with each model a universal code (preferably the NML code). We should then pick the model such that the corresponding universal code length of the data is minimised. Now suppose we use the prequential ML code lengths rather than the NML code lengths. Without loss of generality suppose that $P^* \in \mathcal{M}_1$. Then $P^* \notin \mathcal{M}_2$. This means that the code length relative to $\mathcal{M}_1$ behaves essentially like the NML code length, but the code length relative to $\mathcal{M}_2$ behaves differently – at least as long as the variances do not match (which for example, is forcibly the case if $\mathcal{M}_1$ is Poisson and $\mathcal{M}_2$ is geometric). This introduces a bias in the model selection scheme. In the previous chapter we found experimentally that the error rate for model selection based on the prequential ML code decreases more slowly than when other universal codes are used. Even though in some cases the redundancy grows *more slowly* than $\frac{1}{2} \ln n$, so that the prequential ML code is more efficient than the NML code, we explained that model selection based on the prequential ML codes must nevertheless always behave worse than Bayesian and NML-based model selection. The practical relevance of this phenomenon stems from the fact that the prequential ML code lengths are often a lot easier to compute than the Bayes or NML codes. They are often used in applications [61, 52], so that is important to determine when this can be done safely.

### 3.4.4 Theoretical Significance

The result is also of theoretical-statistical interest: our theorem can be re-interpreted as establishing bounds on the asymptotic *Kullback-Leibler risk* of density estimation using ML and Bayes estimators under misspecification ($P^* \notin \mathcal{M}$). Our result implies that, under misspecification, the KL risk of estimators such as ML, which are required to lie in the model $\mathcal{M}$, behaves in a fundamentally different way from the KL risk of estimators such as the Bayes predictive distribution, which are not restricted to lie in $\mathcal{M}$. Namely, we can think of every universal model $U$ defined as a random process on infinite sequences as an *estimator* in the following way: define, for all $n$,

$$\breve{P}_n := \Pr_U(Z_{n+1} = \cdot \mid Z_1 = z_1, \ldots, Z_n = z_n),$$

a function of the sample $z_1, \ldots, z_n$. $\breve{P}_n$ can be thought of as the "estimate of the true data generating distribution upon observing $z_1, \ldots, z_n$". In case $U$ is the prequential ML model, $\breve{P}_n = P_{\hat{\mu}_n}$ is simply our modified ML estimator. However, universal models other than PML, $\breve{P}_n$ does not always lie in $\mathcal{M}$. An example is the Bayesian universal code defined relative to some prior $w$. This code has lengths $L'(z^n) := -\ln \int P_\mu(z^n) w(\mu) \, \mathrm{d}\mu$ [38]. The corresponding estimator is the *Bayesian posterior predictive distribution* $P_{\mathrm{Bayes}}(z_{i+1} \mid z^i) := \int P_\mu(z_{i+1}) w(\mu \mid z^i) \, \mathrm{d}\mu$ [38].

The Bayesian predictive distribution is a mixture of elements of $\mathcal{M}$. We will call standard estimators like the ML estimator, which are required to lie in $\mathcal{M}$, *in-model* estimators. Estimators like the Bayesian predictive distribution will be called *out-model*.

Let now $\breve{P}_n$ be any estimator, in-model or out-model. Let $\breve{P}_{z^n}$ be the distribution estimated for a particular realized sample $z^n$. We can measure the closeness of $\breve{P}_{z^n}$ to $P_{\mu^*}$, the distribution in $\mathcal{M}$ closest to $P^*$ in KL-divergence, by considering the *extended KL divergence*

$$D^*(P_{\mu^*} \| \breve{P}_{z^n}) = E_{Z \sim P^*}[-\ln \breve{P}_{z^n}(Z) - [-\ln P_{\mu^*}(Z)]].$$

We can now consider the *expected* KL divergence between $P_{\mu^*}$ and $\breve{P}_n$ after observing a sample of length $n$:

$$E_{Z_1,\ldots,Z_n \sim P^*}[D^*(P_{\mu^*} \| \breve{P}_n)]. \tag{3.16}$$

In analogy to the definition of "ordinary" *KL risk* [4], we call (3.16) the *extended KL risk*. We recognise the redundancy of the PML distribution as the accumulated expected KL risk of our modified ML estimator (see Proposition 3.5.7 and Lemma 3.5.6). In exactly the same way as for PML, the redundancy of the Bayesian code can be re-interpreted as the accumulated KL risk of the Bayesian predictive distribution. With this interpretation, our Theorem 3.2.3 expresses that under misspecification, the cumulative KL risk of the ML estimator differs from the cumulative KL risk of the Bayes estimator by a term of $\Theta(\ln n)$. If our conjecture that *no* in-model estimator can achieve redundancy $\frac{1}{2}\ln n + O(1)$ for all $\mu^*$ and all $P^*$ with finite variance is true (Section 3.4.2), then it follows that the KL risk for in-model estimators behaves in a fundamentally different way from the KL risk for out-model estimators, and that out-model estimators are needed to achieve the optimal constant $c = 1$ in the redundancy $\frac{1}{2}c\ln n + O(1)$.

## 3.5 Building Blocks of the Proof

The proof of Theorem 3.2.3 is based on Lemma 3.5.6 and Lemma 3.5.8. These Lemmas are stated and proved in Sections 3.5.2 and 3.5.3, respectively. The proofs of Theorem 3.2.3 and Theorem 3.3.1, as well as the proof of both Lemmas, are based on a number of generally useful results about probabilities and expectations of deviations between the average and the mean of a random variable. We first list list these deviation-related results.

### 3.5.1 Results about Deviations between Average and Mean

**Lemma 3.5.1.** *Let $X, X_1, X_2, \ldots$ be i.i.d. with mean 0. Then $E\left[\left(\sum_{i=1}^n X_i\right)^2\right] = n\mathrm{var}(X)$.*

*Proof.* The lemma is clearly true for $n = 0$. Suppose it is true for some $n$. Abbreviate $S_n := \sum_{i=1}^n X_i$. We have $E[S_n] = \sum EX = 0$. Now we find $E\left[S_{n+1}^2\right] = E\left[(S_n + X)^2\right] = E\left[S_n^2\right] + 2E[S_n]EX + E[X^2] = (n+1)\mathrm{var}(X)$. The proof follows by induction. □

**Theorem 3.5.2.** *Let $X, X_1, \ldots$ be i.i.d. random variables, define $\hat{\mu}_n := (n_0 \cdot x_0 + \sum_{i=1}^n X_i)/(n + n_0)$ and $\mu^* = E[X]$. If $\mathrm{var}X < \infty$, then $E\left[(\hat{\mu}_n - \mu^*)^2\right] = O\left((n+1)^{-2}\right) + \mathrm{var}(X)/(n+1)$.*

*Proof.* We define $Y_i := X_i - \mu^*$; this can be seen as a new sequence of i.i.d. random variables with mean 0 and $\mathrm{var}Y = \mathrm{var}X$. We also set $y_0 := x_0 - \mu^*$. Now we have:

$$\begin{aligned}
E\left[(\hat{\mu}_n - \mu^*)^2\right] &= E\left[\left(n_0 \cdot y_0 + \sum_{i=1}^n Y_i\right)^2\right](n+n_0)^{-2} \\
&= E\left[(n_0 \cdot y_0)^2 + 2n_0 \cdot y_0 \sum_{i=1}^n Y_i + \left(\sum_{i=1}^n Y_i\right)^2\right](n+n_0)^{-2} \\
&= O\left((n+1)^{-2}\right) + E\left[\left(\sum_{i=1}^n Y_i\right)^2\right](n+n_0)^{-2} \\
&\overset{(*)}{=} O\left((n+1)^{-2}\right) + n\mathrm{var}(Y)(n+n_0)^{-2} \\
&= O\left((n+1)^{-2}\right) + \mathrm{var}(X)/(n+1),
\end{aligned}$$

where $(*)$ follows by Lemma 3.5.1. □

The following theorem is of some independent interest.

**Theorem 3.5.3.** *Suppose $X, X_1, X_2, \ldots$ are i.i.d. with mean 0. If the first $k \in \mathbb{N}$ moments of $X$ exist, then we have Then $E\left[\left(\sum_{i=1}^n X_i\right)^k\right] = O\left(n^{\lfloor \frac{k}{2} \rfloor}\right)$.*

**Remark** It follows as a special case of Theorem 2 of [98] that $E\left[|\sum_{i=1}^n X_i|^k\right] = O(n^{\frac{k}{2}})$ which almost proves this lemma and which would in fact be sufficient for our purposes. We use this lemma instead which has an elementary proof.

*Proof.* We have:

$$E\left[\left(\sum_{i=1}^n X_i\right)^k\right] = E\left[\sum_{i_1=1}^n \cdots \sum_{i_k=1}^n X_{i_1} \cdots X_{i_k}\right] = \sum_{i_1=1}^n \cdots \sum_{i_k=1}^n E\left[X_{i_1} \cdots X_{i_k}\right].$$

We define the *frequency sequence* of a term to be the sequence of exponents of the different random variables in the term, in decreasing order. For a frequency sequence $f_1, \ldots, f_m$, we have $\sum_{i=1}^{m} f_i = k$. Furthermore, using independence of the different random variables, we can rewrite $E[X_{i_1} \cdots X_{i_k}] = \prod_{i=1}^{m} E[X^{f_i}]$ so the value of each term is determined by its frequency sequence. By computing the number of terms that share a particular frequency sequence, we obtain:

$$E\left[\left(\sum_{i=1}^{n} X_i\right)^k\right] = \sum_{f_1+\ldots+f_m=k} \binom{n}{m} \binom{k}{f_1, \ldots, f_m} \prod_{i=1}^{m} E[X^{f_i}].$$

To determine the asymptotic behaviour, first observe that the frequency sequence $f_1, \ldots, f_m$ of which the contribution grows fastest in $n$ is the longest sequence, since for that sequence the value of $\binom{n}{m}$ is maximised as $n \to \infty$. However, since the mean is zero, we can discard all sequences with an element 1, because the for those sequences we have $\prod_{i=1}^{m} E[X^{f_i}] = 0$ so they contribute nothing to the expectation. Under this constraint, we obtain the longest sequence for even $k$ by setting $f_i = 2$ for all $1 \leq i \leq m$; for odd $k$ by setting $f_1 = 3$ and $f_i = 2$ for all $2 \leq i \leq m$; in both cases we have $m = \left\lfloor \frac{k}{2} \right\rfloor$. The number of terms grows as $\binom{n}{m} \leq n^m/m! = O(n^m)$; for $m = \left\lfloor \frac{k}{2} \right\rfloor$ we obtain the upper bound $O(n^{\lfloor \frac{k}{2} \rfloor})$. The number of frequency sequences is finite and does not depend on $n$; since the contribution of each one is $O\left(n^{\lfloor \frac{k}{2} \rfloor}\right)$, so must be the sum. □

**Theorem 3.5.4.** *Let $X, X_1, \ldots$ be i.i.d. random variables, define $\hat{\mu}_n := (n_0 \cdot x_0 + \sum_{i=1}^{n} X_i)/(n + n_0)$ and $\mu^* = E[X]$. If the first $k$ moments of $X$ exist, then $E[(\hat{\mu}_n - \mu^*)^k] = O(n^{-\lceil \frac{k}{2} \rceil})$.*

*Proof.* The proof is similar to the proof for Theorem 3.5.2. We define $Y_i := X_i - \mu^*$; this can be seen as a new sequence of i.i.d. random variables with mean 0, and $y_0 := x_0 - \mu^*$. Now we have:

$$\begin{aligned} E\left[(\hat{\mu}_n - \mu^*)^k\right] &= E\left[\left(n_0 \cdot y_0 + \sum_{i=1}^{n} Y_i\right)^k\right] (n + n_0)^{-k} \\ &= O\left(n^{-k}\right) \sum_{p=0}^{k} \binom{k}{p} (n_0 \cdot y_0)^p E\left[\left(\sum_{i=1}^{n} Y_i\right)^{k-p}\right] \\ &= O\left(n^{-k}\right) \sum_{p=0}^{k} \binom{k}{p} (n_0 \cdot y_0)^p \cdot O\left(n^{\lfloor \frac{k-p}{2} \rfloor}\right). \end{aligned}$$

In the last step we used Theorem 3.5.3 to bound the expectation. We sum $k+1$ terms of which the term for $p = 0$ grows fastest in $n$, so the expression is $O(n^{-\lceil \frac{k}{2} \rceil})$ as required. □

Theorem 3.5.4 concerns the *expectation* of the deviation of $\hat{\mu}_n$. We also need a bound on the *probability* of large deviations. To do that we have the following separate theorem:

**Theorem 3.5.5.** *Let $X, X_1, \ldots$ be i.i.d. random variables, define $\hat{\mu}_n := (n_0 \cdot x_0 + \sum_{i=1}^{n} X_i)/(n + n_0)$ and $\mu^* = E[X]$. Let $k \in \{0, 2, 4, \ldots\}$. If the first $k$ moments exists then $P(|\hat{\mu}_n - \mu^*| \geq \delta) = O\left(n^{-\frac{k}{2}} \delta^{-k}\right)$.*

*Proof.*

$$\begin{aligned} P^*(|\hat{\mu}_n - \mu^*| \geq \delta) &= P^*\left((\hat{\mu}_n - \mu^*)^k \geq \delta^k\right) \\ &\leq E\left[(\hat{\mu}_n - \mu^*)^k\right] \delta^{-k} \quad \text{(by Markov's inequality)} \\ &= O\left(n^{-\frac{k}{2}} \delta^{-k}\right) \quad \text{(by Theorem 3.5.4)} \end{aligned}$$

□

## 3.5.2 Lemma 3.5.6: Redundancy for Exponential Families

**Lemma 3.5.6.** *Let $U$ be a PML distribution model and $\mathcal{M}$ be an exponential family as in Theorem 3.2.3. We have*

$$\mathfrak{R}(P^*, U, \mathcal{M}, n) = \sum_{i=0}^{n-1} \underset{\hat{\mu}_i \sim P^*}{E} \left[ D(P_{\mu^*} \parallel P_{\hat{\mu}_i}) \right].$$

*(Here, the notation $\hat{\mu}_i \sim P^*$ means that we take the expectation with respect to $P^*$ over data sequences of length $i$, of which $\hat{\mu}_i$ is a function.)*

*Proof.* We have:

$$\arg\min_{\mu} E_{P^*}\left[-\ln P_\mu(X^n)\right] = \arg\min_{\mu} E_{P^*}\left[\ln \frac{P_{\mu^*}(X^n)}{P_\mu(X^n)}\right] = \arg\min_{\mu} D(P_{\mu^*} \parallel P_\mu).$$

In the last step we used Proposition 3.5.7 below. The divergence is minimised when $\mu = \mu^*$ [48], so we find that:

$$\begin{aligned} &\mathfrak{R}(P^*, U, \mathcal{M}, n) = E_{P^*}[-\ln U(X^n)] - E_{P^*}[-\ln P_{\mu^*}(X^n)] = E_{P^*}\left[\ln \frac{P_{\mu^*}(X^n)}{U(X^n)}\right] \\ &= E_{P^*}\left[\sum_{i=0}^{n-1} \ln \frac{P_{\mu^*}(X_i)}{P_{\hat{\mu}_i}(X_i)}\right] = \sum_{i=0}^{n-1} E_{P^*}\left[\ln \frac{P_{\mu^*}(X_i)}{P_{\hat{\mu}_i}(X_i)}\right] = \sum_{i=0}^{n-1} \underset{\hat{\mu}_i \sim P}{E} \left[ D(P_{\mu^*} \parallel P_{\hat{\mu}_i}) \right]. \end{aligned} \tag{3.17}$$

Here, the last step again follows from Proposition 3.5.7. □

**Proposition 3.5.7.** *Let $X \sim P^*$ with mean $\mu^*$, and let $P_\mu$ index an exponential family with sufficient statistic $X$, so that $P_{\mu^*}$ exists. We have:*

$$E_{P^*}\left[-\ln \frac{P_{\mu^*}(X)}{P_\theta(X)}\right] = D(P_{\mu^*} \parallel P_\theta)$$

*Proof.* Let $\eta(\cdot)$ denote the function mapping parameters in the mean value parameterisation to the natural parameterisation. (It is the inverse of the function $\mu(\cdot)$ which was introduced in the discussion of exponential families.) By working out both sides of the equation we find that they both reduce to:

$$\eta(\mu^*)\mu^* + \ln \mathrm{Z}(\eta(\mu^*)) - \eta(\theta)\mu^* - \ln \mathrm{Z}(\eta(\theta)). \qquad \square$$

### 3.5.3 Lemma 3.5.8: Convergence of the sum of the remainder terms

**Lemma 3.5.8.** *Let $R(n)$ be defined as in (3.10). Then*

$$R(n) = O(1).$$

*Proof.* We omit irrelevant constants. We abbreviate $\frac{\mathrm{d}^k}{\mathrm{d}\mu^k} D(P_{\mu^*} \| P_\mu) = D^{(k)}(\mu)$ as in the proof of Theorem 3.2.3. First we consider the third order term. We write $E_{\delta_i \sim P^*}$ to indicate that we take the expectation over data which is distributed according to $P^*$, of which $\delta_i$ is a function. We use Theorem 3.5.4 to bound the expectation of ${\delta_i}^3$; under the condition that the first three moments exist, which is assumed to be the case, we obtain:

$$\sum_{i=0}^{n-1} \underset{\delta_i \sim P^*}{E}\left[{\delta_i}^3 D^{(3)}(\mu^*)\right] = D^{(3)}(\mu^*) \sum_{i=0}^{n-1} E[\delta_i^3] = D^{(3)}(\mu^*) \sum_{i=0}^{n-1} O((i+1)^{-2}) = O(1).$$

(The constants implicit in the big-ohs are the same across terms.)

The fourth order term is more involved, because $D^{(4)}(\mu)$ is not necessarily constant across terms. To compute it we first distinguish a number of regions in the value space of $\delta_i$: let $\Delta_- = (-\infty, 0)$ and let $\Delta_0 = [0, a)$ for some constant value $a > 0$. If the individual outcomes $X$ are bounded on the right hand side by a value $g$ then we require that $a < g$ and we define $\Delta_1 = [a, g)$; otherwise we define $\Delta_j = [a + j - 1, a + j)$ for $j \geq 1$. Now we must establish convergence of:

$$\sum_{i=0}^{n-1} \underset{\delta_i \sim P^*}{E}\left[{\delta_i}^4 D^{(4)}(\ddot{\mu}_i)\right] \;=\; \sum_{i=0}^{n-1} \sum_j P^*(\delta_i \in \Delta_j) \underset{\delta_i \sim P^*}{E}\left[{\delta_i}^4 D^{(4)}(\ddot{\mu}_i) \mid \delta_i \in \Delta_j\right]$$

If we can establish that the sum converges for all regions $\Delta_j$ for $j \geq 0$, then we can use a symmetrical argument to establish convergence for $\Delta_-$ as well, so it suffices if we restrict attention to $j \geq 0$. First we show convergence for $\Delta_0$. In

this case, the basic idea is that since the remainder $D^{(4)}(\ddot{\mu}_i)$ is well-defined over the interval $\mu^* \leq \mu < \mu^* + a$, we can bound it by its extremum on that interval, namely $m := \sup_{\mu \in [\mu^*, \mu^* + a)} \left| D^{(4)}(\ddot{\mu}_i) \right|$. Now we get:

$$\left| \sum_{i=0}^{n-1} P^*(\delta_i \in \Delta_0) E\left[ {\delta_i}^4 D^{(4)}(\ddot{\mu}_i) \mid \delta_i \in \Delta_0 \right] \right| \leq \left| \sum_{i=0}^{n-1} 1 \cdot E\left[ {\delta_i}^4 \left| D^{(4)}(\ddot{\mu}_i) \right| \right] \right| \leq \left| m \sum_i E\left[ {\delta_i}^4 \right] \right|. \quad (3.18)$$

Using Theorem 3.5.4 we find that $E[{\delta_i}^4]$ is $O((i+1)^{-2})$ of which the sum converges. Theorem 3.5.4 requires that the first four moments of $P^*$ exist, but this is guaranteed to be the case: either the outcomes are bounded from both sides, in which case all moments necessarily exist, or the existence of the required moments is part of the condition on the main theorem.

Now we have to distinguish between the unbounded and bounded cases. First we assume that the $X$ are unbounded from above. In this case, we must show convergence of:

$$\sum_{i=0}^{n-1} \sum_{j=1}^{\infty} P^*(\delta_i \in \Delta_j) E\left[ {\delta_i}^4 D^{(4)}(\ddot{\mu}_i) \mid \delta_i \in \Delta_j \right].$$

To show convergence, we bound the absolute value of this expression from above. The $\delta_i$ in the expectation is at most $a + j$. Furthermore $D^{(4)}(\ddot{\mu}_i) = O(\mu^{k-6})$ by assumption on the main theorem, where $\mu \in [a + j - 1, a + j)$. Depending on $k$, both boundaries could maximise this function, but it is easy to check that in both cases the resulting function is $O(j^{k-6})$. So we get:

$$\left| \ldots \right| \leq \sum_{i=0}^{n-1} \sum_{j=1}^{\infty} P^*(|\delta_i| \geq a + j - 1)(a + j)^4 O(j^{k-6}).$$

Since we know from the condition on the main theorem that the first $k \geq 4$ moments exist, we can apply Theorem 3.5.5 to find that $P(|\delta_i| \geq a + j - 1) = O(i^{-\lceil \frac{k}{2} \rceil}(a + j - 1)^{-k}) = O(i^{-\frac{k}{2}})O(j^{-k})$ (since $k$ has to be even); plugging this into the equation and simplifying we obtain $\sum_i O(i^{-\frac{k}{2}}) \sum_j O(j^{-2})$. For $k \geq 4$ this expression converges.

Now we consider the case where the outcomes are bounded from above by $g$. This case is more complicated, since now we have made no extra assumptions as to existence of the moments of $P^*$. Of course, if the outcomes are bounded from both sides, then all moments necessarily exist, but if the outcomes are unbounded from below this may not be true. We use a trick to remedy this: we map all outcomes

into a new domain in such a way that all moments of the transformed variables are guaranteed to exist. Any constant $x^-$ defines a mapping $g(x) := \max\{x^-, x\}$. Furthermore we define the random variables $Y_i := g(X_i)$, the initial outcome $y_0 := g(x_0)$ and the mapped analogues of $\mu^*$ and $\hat{\mu}_i$, respectively: $\mu^\dagger$ is defined as the mean of $Y$ under $P$ and $\tilde{\mu}_i := (y_0 \cdot n_0 + \sum_{j=1}^{i} Y_j)/(i + n_0)$. Since $\tilde{\mu}_i \geq \hat{\mu}_i$, we can bound:

$$\begin{aligned} & \left| \sum_i P(\delta_i \in \Delta_1) E\left[ {\delta_i}^4 D^{(4)}(\ddot{\mu}_i) \mid \delta_i \in \Delta_1 \right] \right| \\ \leq & \sum_i P(\hat{\mu}_i - \mu^* \geq a) \sup_{\delta_i \in \Delta_1} \left| {\delta_i}^4 D^{(4)}(\ddot{\mu}_i) \right| \\ \leq & \sum_i P(|\tilde{\mu}_i - \mu^\dagger| \geq a + \mu^* - \mu^\dagger) g^4 \sup_{\delta_i \in \Delta_1} \left| D^{(4)}(\ddot{\mu}_i) \right|. \end{aligned}$$

By choosing $x^-$ small enough, we can bring $\mu^\dagger$ and $\mu^*$ arbitrarily close together; in particular we can choose $x^-$ such that $a + \mu^* - \mu^\dagger > 0$ so that application of Theorem 3.5.5 is safe. It reveals that the summed probability is $O(i^{-\frac{k}{2}})$. Now we bound $D^{(4)}(\ddot{\mu}_i)$ which is $O((g-\mu)^{-m})$ for some $m \in \mathbb{N}$ by the condition on the main theorem. Here we use that $\ddot{\mu}_i \leq \hat{\mu}_i$; the latter is maximised if all outcomes equal the bound $g$, in which case the estimator equals $g - n_0(g - x_0)/(i + n_0) = g - O(i^{-1})$. Putting all of this together, we get $\sup \left| D^{(4)}(\ddot{\mu}_i) \right| = O((g-\mu)^{-m}) = O(i^m)$; if we plug this into the equation we obtain:

$$\ldots \leq \sum_i O(i^{-\frac{k}{2}}) g^4 O(i^m) = g^4 \sum_i O(i^{m-\frac{k}{2}})$$

This converges if we choose $k \geq m/2$. As the construction of the mapping $g(\cdot)$ ensures that all moments exist, the first $m/2$ moments certainly must exist. This completes the proof. □

## 3.6 Proof of Theorem 3.3.1

We use the same conventions as in the proof of Theorem 3.2.3. Specifically, we concentrate on the random variables $X_1, X_2, \ldots$ rather than $Z_1, Z_2, \ldots$, which is justified by Equation (3.7). Let $f(x^n) = -\ln P_{\mu^*}(x^n) - [\inf_{\mu \in \Theta_\mu} - \ln P_\mu(x^n)]$. Within this section, $\hat{\mu}(x^n)$ is defined as the ordinary ML estimator. Note that, if $x^n$ is such that its ML estimate is defined, then $f(x^n) = -\ln P_{\mu^*}(x^n) + \ln P_{\hat{\mu}(x^n)}(x^n)$.

Note $d(n) = E_{P^*}[f(X^n)]$. Let $h(x)$ be the carrier of the exponential family under consideration (see Definition 3.2.1). Without loss of generality, we assume

$h(x) > 0$ for all $x$ in the finite set $\mathcal{X}$. Let $a_n^2 = n^{-1/2}$. We can write

$$\begin{aligned} d(n) \;=\; E_{P^*}[f(X^n)] \;=\; & \pi_n \quad E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 \geq a_n^2] \\ + \; & (1-\pi_n) \quad E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2], \end{aligned} \tag{3.19}$$

where $\pi_n = P^*((\mu^* - \hat{\mu}_n)^2 \geq a_n^2)$. We determine $d(n)$ by bounding the two terms on the right of (3.19). We start with the first term. Since $X$ is bounded, all moments of $X$ exists under $P^*$, so we can bound $\pi_n$ using Theorem 3.5.5 with $k = 8$ and $\delta = a_n = n^{-1/4}$. (Note that the theorem in turn makes use of Theorem 3.5.4 which remains valid when we use $n_0 = 0$.) This gives

$$\pi_n = O(n^{-2}). \tag{3.20}$$

Note that for all $x^n \in \mathcal{X}^n$, we have

$$0 \leq f(x^n) \leq \sup_{x^n \in \mathcal{X}^n} f(x^n) \leq \sup_{x^n \in \mathcal{X}^n} -\ln P_{\mu^*}(x^n) \leq nC, \tag{3.21}$$

where $C$ is some constant. Here the first inequality follows because $\hat{\mu}$ maximises $\ln P_{\hat{\mu}(x^n)}(x^n)$ over $\mu$; the second is immediate; the third follows because we are dealing with discrete data, so that $P_{\hat{\mu}}$ is a probability mass function, and $P_{\hat{\mu}}(x^n)$ must be $\leq 1$. The final inequality follows because $\mu^*$ is in the interior of the parameter space, so that the natural parameter $\eta(\mu^*)$ is in the interior of the natural parameter space. Because $X$ is bounded and we assumed $h(x) > 0$ for all $x \in \mathcal{X}$, it follows by the definition of exponential families that $\sup_{x \in \mathcal{X}} -\ln P_{\mu^*}(x) < \infty$.

Together (3.20) and (3.21) show that the expression on the first line of (3.19) converges to 0, so that (3.19) reduces to

$$d(n) = (1 - \pi_n) E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2] + O(n^{-1}). \tag{3.22}$$

To evaluate the term inside the expectation further we first Taylor approximate $f(x^n)$ around $\hat{\mu}_n = \hat{\mu}(x^n)$, for given $x^n$ with $(\mu^* - \hat{\mu}_n)^2 < a_n^2 = 1/\sqrt{n}$. We get

$$f(x^n) = -(\mu^* - \hat{\mu}_n) \frac{d}{d\mu} \ln P_{\hat{\mu}_n}(x^n) + n \frac{1}{2} (\mu^* - \hat{\mu}_n)^2 I(\mu_n), \tag{3.23}$$

where $I$ is the Fisher information (as defined in the proof of the main theorem) and $\mu_n$ lies in between $\mu^*$ and $\hat{\mu}$, and depends on the data $x^n$. Since the first derivative of $\mu$ at the ML estimate $\hat{\mu}$ is 0, the first-order term is 0. Therefore $f(x^n) = \frac{1}{2} n (\mu^* - \hat{\mu}_n)^2 I(\mu_n)$, so that

$$\frac{1}{2} n g(n) \inf_{\mu \in [\mu^* - a_n, \mu^* + a_n]} I(\mu) \leq E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2] \leq \frac{1}{2} n g(n) \sup_{\mu \in [\mu^* - a_n, \mu^* + a_n]} I(\mu),$$

where we abbreviated $g(n) := E_{P^*}[(\mu^* - \hat{\mu}_n)^2 \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2]$. Since $I(\mu)$ is smooth and positive, we can Taylor-approximate it as $I(\mu^*) + O(n^{-\frac{1}{4}})$, so we obtain the bound:

$$E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2] = n g(n) \left( \frac{1}{2} I(\mu^*) + O(n^{-\frac{1}{4}}) \right). \tag{3.24}$$

To evaluate $g(n)$, note that we have

$$E_{P^*}[(\mu^* - \hat{\mu}_n)^2] = \pi_n E_{P^*}[(\mu^* - \hat{\mu}_n)^2 \mid (\mu^* - \hat{\mu}_n)^2 \geq a_n^2] + (1 - \pi_n)g(n). \quad (3.25)$$

Using Theorem 3.5.2 with $n_0 = 0$ we rewrite the expectation on the left hand side as $\mathrm{var}_{P^*}X/n$. Subsequently reordering terms we obtain:

$$g(n) = \frac{(\mathrm{var}_{P^*}X)/n - \pi_n E_{P^*}[(\mu^* - \hat{\mu}_n)^2 \mid (\mu^* - \hat{\mu}_n)^2 \geq a_n^2]}{1 - \pi_n}. \quad (3.26)$$

Plugging this into bound (3.24), and multiplying both sides by $1 - \pi_n$, we get:

$$(1 - \pi_n)E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2] = \\ \left(\mathrm{var}_{P^*}X - n\pi_n E_{P^*}[(\mu^* - \hat{\mu}_n)^2 \mid (\mu^* - \hat{\mu}_n)^2 \geq a_n^2]\right)\left(\frac{1}{2}I(\mu^*) + O(n^{-\frac{1}{4}})\right).$$

Since $X$ is bounded, the expectation on the right must lie between 0 and some constant $C$. Using $\pi_n = O(n^{-2})$ and the fact that $I(\mu^*) = 1/\mathrm{var}_{P_{\mu^*}}X$, we get

$$(1 - \pi_n)E_{P^*}[f(X^n) \mid (\mu^* - \hat{\mu}_n)^2 < a_n^2] = \frac{1}{2}\frac{\mathrm{var}_{P^*}X}{\mathrm{var}_{P_{\mu^*}}X} + O(n^{-\frac{1}{4}}).$$

The result follows if we combine this with (3.22).

## 3.7 Conclusion and Future Work

In this paper we established two theorems about the relative redundancy, defined in Section 3.1:

1. A particular type of universal code, the *prequential ML code* or *ML plug-in code*, exhibits behaviour that we found unexpected. While other important universal codes such as the NML/Shtarkov and Bayesian codes, achieve a regret of $\frac{1}{2}\ln n$, where $n$ is the sample size, the prequential ML code achieves a relative redundancy of $\frac{1}{2}\frac{\mathrm{var}_{P^*}X}{\mathrm{var}_{P_{\mu^*}}X}\ln n$. (Sections 3.1 and 3.2.)

2. At least for finite sample spaces, the relative redundancy is very close to the expected regret, the difference going to $\frac{1}{2}\frac{\mathrm{var}_{P^*}X}{\mathrm{var}_{P_{\mu^*}}X}$ as the sample size increases (Section 3.3, Theorem 3.3.1). In future work, we hope to extend this theorem to general 1-parameter exponential families with arbitrary sample spaces.

There is a substantial amount of literature in which the regret for the prequential ML code is proven to grow with $\frac{1}{2}\ln n$. While this may seem to contradict our results, in fact it does not: In those articles, settings are considered where

$P^* \in \mathcal{M}$, and under such circumstances our own findings predict precisely that behaviour.

The first result is robust with respect to slight variations in the definition of the prequential ML code: in our framework the so-called "start-up problem" (the unavailability of an ML estimate for the first few outcomes) is resolved by introducing fake initial outcomes. Our framework thus also covers prequential codes that use other point estimators such as the Bayesian MAP and mean estimators defined relative to a large class of reasonable priors. In Section 3.4.2 we conjecture that no matter what in-model estimator is used, the prequential model cannot yield a relative redundancy of $\frac{1}{2} \ln n$ independently of the variance of the data generating distribution.

# Chapter 4

# Interlude: Prediction with Expert Advice

We cannot predict exactly how complicated processes such as the weather, the stock market, social interactions and so on, will develop into the future. Nevertheless, people do make weather forecasts and buy shares all the time. Such predictions can be based on formal models, or on human expertise or intuition. An investment company may even want to choose between portfolios on the basis of a combination of these kinds of predictors. In such scenarios, predictors typically cannot be considered "true". Thus, we may well end up in a position where we have a whole collection of prediction strategies, or *experts*, each of whom has *some* insight into *some* aspects of the process of interest. We address the question how a given set of experts can be combined into a single predictive strategy that is as good as, or if possible even better than, the best individual expert.

The setup is as follows. Let $\Xi$ be a finite set of experts. Each expert $\xi \in \Xi$ issues a distribution $P_\xi(\boldsymbol{x}_{n+1}|x^n)$ on the next outcome $\boldsymbol{x}_{n+1}$ given the previous observations $x^n := x_1, \ldots, x_n$. Here, each outcome $x_i$ is an element of some countable space $\mathcal{X}$, and random variables are written in bold face. The probability that an expert assigns to a sequence of outcomes is given by the chain rule: $P_\xi(x^n) = P_\xi(x_1) \cdot P_\xi(x_2|x_1) \cdot \ldots \cdot P_\xi(x_n|x^{n-1})$.

A standard Bayesian approach to combine the expert predictions is to define a prior $w$ on the experts $\Xi$ which induces a joint distribution with mass function $P(x^n, \xi) = w(\xi)P_\xi(x^n)$. Inference is then based on this joint distribution. We can compute, for example: (a) the *marginal probability* of the data $P(x^n) = \sum_{\xi\in\Xi} w(\xi)P_\xi(x^n)$, (b) the *predictive distribution* on the next outcome $P(\boldsymbol{x}_{n+1}|x^n) = P(x^n, \boldsymbol{x}_{n+1})/P(x^n)$, which defines a prediction strategy that combines those of the individual experts, or (c) the *posterior distribution* on the experts $P(\boldsymbol{\xi}|x^n) = P_{\boldsymbol{\xi}}(x^n)w(\boldsymbol{\xi})/P(x^n)$, which tells us how the experts' predictions should be weighted. This simple probabilistic approach has the advantage that it is computationally easy: predicting $n$ outcomes using $|\Xi|$ experts requires only $O(n \cdot |\Xi|)$ time. Additionally, this Bayesian strategy guarantees that the overall

probability of the data is only a factor $w(\hat{\xi})$ smaller than the probability of the data according to the best available expert $\hat{\xi}$. On the flip side, with this strategy we never do any *better* than $\hat{\xi}$ either: we have $P_{\hat{\xi}}(x^n) \geq P(x^n) \geq P_{\hat{\xi}}(x^n)w(\hat{\xi})$, which means that potentially valuable insights from the other experts are not used to our advantage!

More sophisticated combinations of prediction strategies can be found in the literature under various headings, including (Bayesian) statistics, source coding and universal prediction. In the latter the experts' predictions are not necessarily probabilistic, and scored using an arbitrary loss function. Here we consider only logarithmic loss, although our results can undoubtedly be generalised to the framework described in, e.g. [95].

The three main contributions of this paper are the following. First, we introduce prior distributions on *sequences* of experts, which allows unified description of many existing models. Second, we show how HMMs can be used as an intuitive graphical language to describe such priors and obtain computationally efficient prediction strategies. Third, we use this new approach to describe and analyse several important existing models, as well as one recent and one completely new model for expert tracking.

## Overview

In Section 4.1 we develop a new, more general framework for combining expert predictions, where we consider the possibility that the *optimal* weights used to mix the expert predictions may *vary over time*, i.e. as the sample size increases. We stick to Bayesian methodology, but we define the prior distribution as a probability measure on *sequences of experts* rather than on experts. The prior probability of a sequence $\xi_1, \xi_2, \ldots$ is the probability that we rely on expert $\xi_1$'s prediction of the first outcome and expert $\xi_2$'s prediction of the second outcome, etc. This allows for the expression of more sophisticated models for the combination of expert predictions. For example, the nature of the data generating process may evolve over time; consequently different experts may be better during different periods of time. It is also possible that not the data generating process, but the experts themselves change as more and more outcomes are being observed: they may learn from past mistakes, possibly at different rates, or they may have occasional bad days, etc. In both situations we may hope to benefit from more sophisticated modelling.

Of course, not all models for combining expert predictions are computationally feasible. Section 4.2 describes a methodology for the specification of models that allow efficient evaluation. We achieve this by using hidden Markov models (HMMs) on two levels. On the first level, we use an HMM as a formal specification of a distribution on sequences of *experts* as defined in Section 4.1. We introduce a graphical language to conveniently represent its structure. These graphs help to understand and compare existing models and to design new ones.

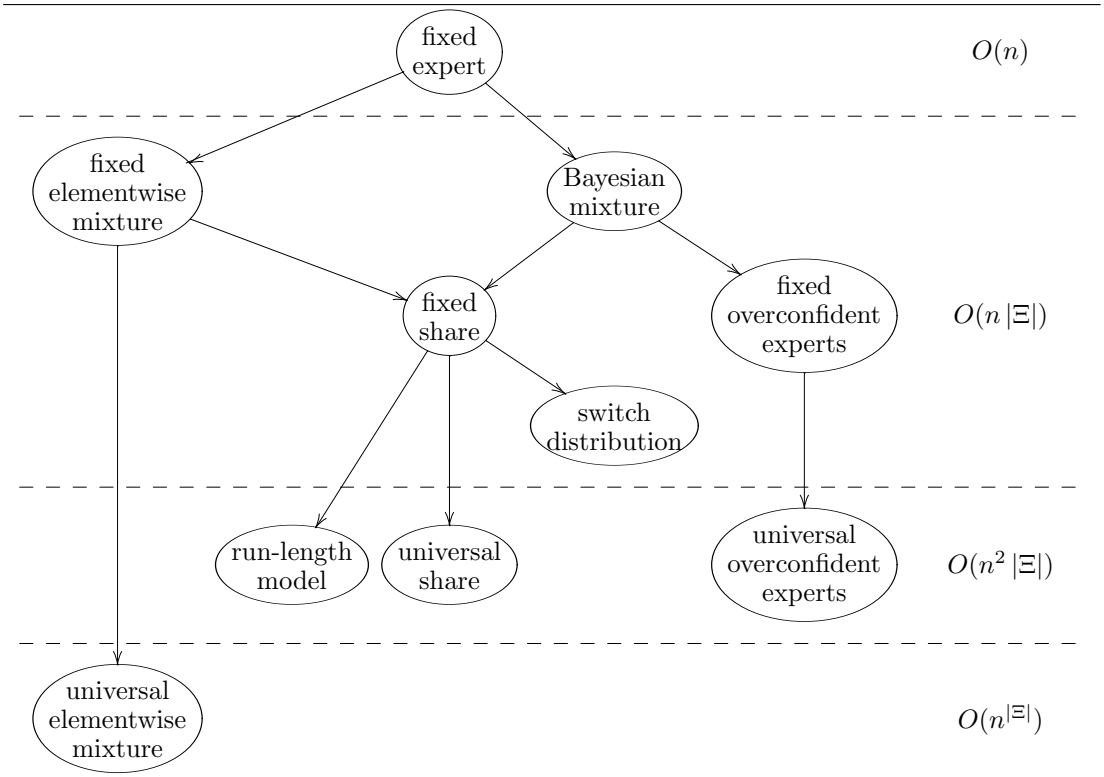
We then modify this first HMM to construct a second HMM that specifies the distribution on sequences of *outcomes*. Subsequently, we can use the standard dynamic programming algorithms for HMMs (forward, backward and Viterbi) on both levels to efficiently calculate most relevant quantities, most importantly the marginal probability of the observed outcomes $P(x^n)$ and posterior weights on the next expert given the previous observations $P(\boldsymbol{\xi_{n+1}}|x^n)$.

It turns out that many existing models for prediction with expert advice can be specified as HMMs. We provide an overview in Section 4.3 by giving the graphical representations of the HMMs corresponding to the following models. First, universal elementwise mixtures (sometimes called mixture models) that learn the optimal mixture parameter from data. Second, Herbster and Warmuth's fixed share algorithm for tracking the best expert [45, 46]. Third, universal share, which was introduced by Volf and Willems as the "switching method" [94] and later independently proposed by Bousquet [15]. Here the goal is to learn the optimal fixed-share parameter from data. The last considered model safeguards against overconfident experts, a case first considered by Vovk in [95]. We render each model as a prior on sequences of experts by giving its HMM. The size of the HMM immediately determines the required running time for the forward algorithm. The generalisation relationships between these models as well as their running times are displayed in Figure 4.1. In each case this running time coincides with that of the best known algorithm. We also give a loss bound for each model, relating the loss of the model to the loss of the best competitor among a set of alternatives in the worst case. Such loss bounds can help select between different models for specific prediction tasks.

Besides the models found in the literature, Figure 4.1 also includes two new generalisations of fixed share: the switch distribution and the run-length model. These models are the subject of Section 4.4. In Chapter 5 the switch distribution is used to improve Bayes/Minimum Description Length prediction to achieve the optimal rate of convergence in nonparametric settings. Here we give the concrete HMM that allows for its linear time computation, and we prove that it matches its parametric definition. The run-length model is based on a distribution on the number of successive outcomes that are typically well-predicted by the same expert. Run-length codes are typically applied directly to the data, but in our novel application they define the prior on expert sequences instead. Again, we provide the graphical representation of their defining HMMs as well as loss bounds.

Then in Section 4.5 we discuss a number of extensions of the above approach, such as approximation methods to speed up calculations for large HMMs.

**Figure 4.1** Expert sequence priors: generalisation relationships and run time




## 4.1 Expert Sequence Priors

In this section we explain how expert tracking can be described in probability theory using expert sequence priors (ES-priors). These ES-priors are distributions on the space of infinite sequences of experts that are used to express regularities in the development of the relative quality of the experts' predictions. As illustrations we render Bayesian mixtures and elementwise mixtures as ES-priors. In the next section we show how ES-priors can be implemented efficiently by hidden Markov models.

**Notation** For $n \in \mathbb{N}$, we abbreviate $\{1, 2, \ldots, n\}$ by $[n]$, with the understanding that $[0] = \emptyset$. We also define $[\infty] = \mathbb{Z}^+$. For any natural number $n$, we let the variable $q^n$ range over the $n$-fold Cartesian product $Q^n$, and we write $q^n = \langle q_1, \ldots, q_n \rangle$. We also let $q^\infty$ range over $Q^\infty$ — the set of infinite sequences over $Q$ — and write $q^\infty = \langle q_1, \ldots \rangle$. We read the statement $q^\lambda \in Q^{\leq \infty}$ to first bind $\lambda \leq \infty$ and subsequently $q^\lambda \in Q^\lambda$. If $q^\lambda$ is a sequence, and $\kappa \leq \lambda$, then we denote by $q^\kappa$ the prefix of $q^\lambda$ of length $\kappa$.

**Forecasting Systems** Let $\mathcal{X}$ be a countable outcome space. We use the notation $\mathcal{X}^*$ for the set of all finite sequences over $\mathcal{X}$ and let $\triangle(\mathcal{X})$ denote the set of

all probability mass functions on $\mathcal{X}$. A *(prequential) $\mathcal{X}$-forecasting system* (PFS) is a function $P : \mathcal{X}^* \to \triangle(\mathcal{X})$ that maps sequences of previous observations to a predictive distribution on the next outcome. Prequential forecasting systems were introduced by Dawid in [27].

**Distributions** We also require probability measures on spaces of infinite sequences. In such a space, a basic event is the set of all continuations of a given prefix. We identify such events with their prefix. Thus a distribution on $\mathcal{X}^\infty$ is defined by a function $P : \mathcal{X}^* \to [0, 1]$ that satisfies $P(\epsilon) = 1$, where $\epsilon$ is the empty sequence, and for all $n \geq 0$, all $x^n \in \mathcal{X}^n$ we have $\sum_{x \in \mathcal{X}} P(x_1, \ldots, x_n, x) = P(x^n)$. We identify $P$ with the distribution it defines. We write $P(x^n|x^m)$ for $P(x^n)/P(x^m)$ if $0 \leq m \leq n$.

Note that forecasting systems continue to make predictions even after they have assigned probability 0 to a previous outcome, while distributions' predictions become undefined. Nonetheless we use the same notation: we write $P(x_{n+1}|x^n)$ for the probability that a forecasting system $P$ assigns to the $n + 1$st outcome given the first $n$ outcomes, as if $P$ were a distribution.

**ES-Priors** The slogan of this chapter is, *we do not understand the data*. Instead of modelling the data, we work with experts. We assume that there is a fixed set of experts $\Xi$, and that each expert $\xi \in \Xi$ predicts using a forecasting system $P_\xi$.

We are interested in switching between different forecasting systems at different sample sizes. For a sequence of experts with prefix $\xi^n$, the combined forecast, where expert $\xi_i$ predicts the $i$th outcome, is denoted

$$P_{\xi^n}(x^n) := \prod_{i=1}^{n} P_{\xi_i}(x_i|x^{i-1}).$$

Adopting Bayesian methodology, we impose a prior $\pi$ on infinite sequences of experts; this prior is called an *expert sequence prior* (ES-prior). Inference is then based on the distribution on the joint space $(\mathcal{X} \times \Xi)^\infty$, called the *ES-joint*, which is defined as follows:

$$P\Big(\langle \xi_1, x_1 \rangle, \ldots, \langle \xi_n, x_n \rangle\Big) := \pi(\xi^n) P_{\xi^n}(x^n). \tag{4.1}$$

We adopt shorthand notation for events: when we write $P(S)$, where $S$ is a subsequence of $\xi^n$ and/or of $x^n$, this means the probability under $P$ of the set of sequences of pairs which match $S$ exactly. For example, the marginal probability of a sequence of outcomes is:

$$P(x^n) = \sum_{\xi^n \in \Xi^n} P(\xi^n, x^n) = \sum_{\xi^n} P\Big( \langle \xi_1, x_1 \rangle, \ldots, \langle \xi_n, x_n \rangle \Big). \tag{4.2}$$

Compare this to the usual Bayesian statistics, where a model $\{P_\theta \mid \theta \in \Theta\}$ is also endowed with a prior distribution $w$ on $\Theta$. Then, after observing outcomes $x^n$, inference is based on the posterior $P(\boldsymbol{\theta}|x^n)$ on the parameter, which is never actually observed. Our approach is exactly the same, but we always consider $\Theta = \Xi^\infty$. Thus as usual our predictions are based on the posterior $P(\boldsymbol{\xi}^\infty|x^n)$. However, since the predictive distribution of $\boldsymbol{x}_{n+1}$ only depends on $\xi_{n+1}$ (and $x^n$) we always marginalise as follows:

$$P(\xi_{n+1}|x^n) = \frac{P(\xi_{n+1}, x^n)}{P(x^n)} = \frac{\sum_{\xi^n} P(\xi^n, x^n) \cdot \pi(\xi_{n+1}|\xi^n)}{\sum_{\xi^n} P(\xi^n, x^n)}. \tag{4.3}$$

At each moment in time we predict the data using the posterior, which is a mixture over our experts' predictions. Ideally, the ES-prior $\pi$ should be chosen such that the posterior coincides with the optimal mixture weights of the experts at each sample size. The traditional interpretation of our ES-prior as a representation of belief about an unknown "true" expert sequence is tenuous, as normally the experts do not generate the data, they only predict it. Moreover, by mixing over different expert sequences, it is often possible to predict significantly better than by using any single sequence of experts, a feature that is crucial to the performance of many of the models that will be described below and in Section 4.3. In the remainder of this chapter we motivate ES-priors by giving performance guarantees in the form of bounds on running time and loss.

### 4.1.1 Examples

We now show how two ubiquitous models can be rendered as ES-priors.

**Example 9** (Bayesian Mixtures)**.** Let $\Xi$ be a set of experts, and let $P_\xi$ be a PFS for each $\xi \in \Xi$. Suppose that we do not know which expert will make the best predictions. Following the usual Bayesian methodology, we combine their predictions by conceiving a prior $w$ on $\Xi$, which (depending on the adhered philosophy) may or may not be interpreted as an expression of one's beliefs in this respect. Then the standard Bayesian mixture $P_{\text{bayes}}$ is given by

$$P_{\text{bayes}}(x^n) = \sum_{\xi \in \Xi} P_\xi(x^n) w(\xi), \quad \text{where} \quad P_\xi(x^n) = \prod_{i=1}^{n} P_\xi(x_i|x^i). \tag{4.4}$$

The Bayesian mixture is not an ES-joint, but it can easily be transformed into one by using the ES-prior that assigns probability $w(\xi)$ to the identically-$\xi$ sequence for each $\xi \in \Xi$:

$$\pi_{\text{bayes}}(\xi^n) = \begin{cases} w(k) & \text{if } \xi_i = k \text{ for all } i = 1, \ldots, n, \\ 0 & \text{o.w.} \end{cases}$$

We will use the adjective "Bayesian" generously throughout this paper, but when we write *the standard Bayesian ES-prior* this always refers to $\pi_{\text{bayes}}$. $\Diamond$

**Example 10** (Elementwise Mixtures)**.** The *elementwise mixture*[1] is formed from some mixture weights $\alpha \in \triangle(\Xi)$ by

$$P_{\mathrm{mix},\alpha}(x^n) := \prod_{i=1}^{n} P_\alpha(x_i|x^{i-1}), \quad \text{where} \quad P_\alpha(x_i|x^{i-1}) = \sum_{\xi\in\Xi} P_\xi(x_i|x^{i-1})\alpha(\xi).$$

In the preceding definition, it may seem that elementwise mixtures do not fit in the framework of ES-priors. But we can rewrite this definition in the required form as follows:

$$\begin{aligned} P_{\mathrm{mix},\alpha}(x^n) &= \prod_{i=1}^{n}\sum_{\xi\in\Xi} P_\xi(x_i|x^{i-1})\alpha(\xi) = \sum_{\xi^n\in\Xi^n}\prod_{i=1}^{n} P_{\xi_i}(x_i|x^{i-1})\alpha(\xi_i) \\ &= \sum_{\xi^n} P_{\xi^n}(x^n)\pi_{\mathrm{mix},\alpha}(\xi^n), \end{aligned} \tag{4.5a}$$

which is the ES-joint based on the prior

$$\pi_{\mathrm{mix},\alpha}(\xi^n) := \prod_{i=1}^{n}\alpha(\xi_i). \tag{4.5b}$$

Thus, the ES-prior for elementwise mixtures is just the multinomial distribution with mixture weights $\alpha$. ◊

We mentioned above that ES-priors cannot be interpreted as expressions of belief about individual expert sequences; this is a prime example where the ES-prior is crafted such that its posterior $\pi_{\mathrm{mix},\alpha}(\xi_{n+1}|\xi^n)$ exactly coincides with the desired *mixture* of experts.

## 4.2 Expert Tracking using HMMs

We explained in the previous section how expert tracking can be implemented using expert sequence priors. In this section we specify ES-priors using hidden Markov models (HMMs). The advantage of using HMMs is that the complexity of the resulting expert tracking procedure can be read off directly from the structure of the HMM. We first give a short overview of the particular kind of HMMs that we use throughout this chapter. We then show how HMMs can be used to specify ES-priors. As illustrations we render the ES-priors that we obtained for Bayesian mixtures and elementwise mixtures in the previous sections as HMMs. We conclude by giving the forward algorithm for our particular kind of HMMs. In Section 4.3 we provide an overview of ES-priors and their defining HMMs that are found in the literature.

[1]These mixtures are sometimes just called mixtures, or predictive mixtures. We use the term elementwise mixtures both for descriptive clarity and to avoid confusion with Bayesian mixtures.

### 4.2.1 Hidden Markov Models Overview

Hidden Markov models (HMMs) are a well-known tool for specifying probability distributions on sequences with temporal structure. Furthermore, these distributions are very appealing algorithmically: many important probabilities can be computed efficiently for HMMs. These properties make HMMs ideal models of expert sequences: ES-priors. For an introduction to HMMs, see [66]. We require a slightly more general notion that incorporates silent states and forecasting systems as explained below.

We define our HMMs on a generic set of outcomes $\mathcal{O}$ to avoid confusion in later sections, where we use HMMs in two different contexts. First in Section 4.2.2, we use HMMs to define ES-priors, and instantiate $\mathcal{O}$ with the set of experts $\Xi$. Then in Section 4.2.4 we modify the HMM that defines the ES-prior to incorporate the experts' predictions, whereupon $\mathcal{O}$ is instantiated with the set of observable outcomes $\mathcal{X}$.

**Definition 4.2.1.** Let $\mathcal{O}$ be a finite set of outcomes. We call a quintuple

$$\mathbb{A} = \left\langle Q, Q_{\mathrm{p}}, \mathrm{P}_\circ, \mathrm{P}, \left\langle P_q \right\rangle_{q \in Q_{\mathrm{p}}} \right\rangle$$

a *hidden Markov model* on $\mathcal{O}$ if $Q$ is a countable set, $Q_{\mathrm{p}} \subseteq Q$, $\mathrm{P}_\circ \in \triangle(Q)$, $\mathrm{P} : Q \to \triangle(Q)$ and $P_q$ is an $\mathcal{O}$-forecasting system for each $q \in Q_{\mathrm{p}}$.

**Terminology and Notation** We call the elements of $Q$ *states*. We call the states in $Q_{\mathrm{p}}$ *productive* and the other states *silent*. We call $\mathrm{P}_\circ$ the *initial distribution*, let $I$ denote its support (i.e. $I := \left\{q \in Q \mid \mathrm{P}_\circ(q) > 0\right\}$) and call $I$ the set of *initial states*. We call P the *stochastic transition function*. We let $S_q$ denote the support of $\mathrm{P}(q)$, and call each $q' \in S_q$ a *direct successor* of $q$. We abbreviate $\mathrm{P}(q)(q')$ to $\mathrm{P}(q \to q')$. A finite or infinite sequence of states $q^\lambda \in Q^{\leq \infty}$ is called a *branch* through $\mathbb{A}$. A branch $q^\lambda$ is called a *run* if either $\lambda = 0$ (so $q^\lambda = \epsilon$), or $q_1 \in I$ and $q_{i+1} \in S_{q_i}$ for all $1 \leq i < \lambda$. A finite run $q^n \neq \epsilon$ is called *a run to* $q_n$. For each branch $q^\lambda$, we denote by $q^\lambda_{\mathrm{p}}$ its subsequence of productive states. We denote the elements of $q^\lambda_{\mathrm{p}}$ by $q^{\mathrm{p}}_1$, $q^{\mathrm{p}}_2$ etc. We call an HMM *continuous* if $q^\infty_{\mathrm{p}}$ is infinite for each infinite run $q^\infty$.

**Restriction** In this chapter we only work with continuous HMMs. This restriction is necessary for the following to be well-defined.

**Definition 4.2.2.** An HMM $\mathbb{A}$ defines the following distribution on sequences of states. $\pi_{\mathbb{A}}(\epsilon) := 1$, and for $\lambda \geq 1$

$$\pi_{\mathbb{A}}(q^\lambda) := \mathrm{P}_\circ(q_1) \prod_{i=1}^{\lambda-1} \mathrm{P}(q_i \to q_{i+1}).$$

Then via the PFSs, $\mathbb{A}$ induces the joint distribution $P_{\mathbb{A}}$ on runs and sequences of outcomes. Let $o^n \in \mathcal{O}^n$ be a sequence of outcomes and let $q^\lambda \neq \epsilon$ be a run with at least $n$ productive states, then

$$P_{\mathbb{A}}(o^n, q^\lambda) := \pi_{\mathbb{A}}(q^\lambda) \prod_{i=1}^{n} P_{q_i^{\mathrm{P}}}(o_i | o^{i-1}).$$

The value of $P_{\mathbb{A}}$ at arguments $o^n, q^\lambda$ that do not fulfil the condition above is determined by the additivity axiom of probability.

**Generative Perspective** The corresponding generative viewpoint is the following. Begin by sampling an initial state $q_1$ from the initial distribution $\mathrm{P}_\circ$. Then iteratively sample a direct successor $q_{i+1}$ from $\mathrm{P}(q_i)$. Whenever a productive state $q_i$ is sampled, say the $n^{\text{th}}$, also sample an outcome $o_n$ from the forecasting system $P_{q_i}$ given all previously sampled outcomes $o^{n-1}$.

**The Importance of Silent States** Silent states can always be eliminated. Let $q'$ be a silent state and let $R_{q'} := \{q \mid q' \in S_q\}$ be the set of states that have $q'$ as their direct successor. Now by connecting each state $q \in R_{q'}$ to each state $q'' \in S_{q'}$ with transition probability $\mathrm{P}(q \to q')\,\mathrm{P}(q' \to q'')$ and removing $q'$ we preserve the induced distribution on $Q^\infty$. Now if $|R_{q'}| = 1$ or $|S_{q'}| = 1$ then $q'$ deserves this treatment. Otherwise, the number of successors has increased, since $|R_{q'}| \cdot |S_{q'}| \geq |R_{q'}| + |S_{q'}|$, and the increase is quadratic in the worst case. Thus, silent states are important to keep our HMMs small: they can be viewed as shared common subexpressions. It is important to keep HMMs small, since the size of an HMM is directly related to the running time of standard algorithms that operate on it. These algorithms are described in the next section.

### Algorithms

There are three classical tasks associated with hidden Markov models [66]. To give the complexity of algorithms for these tasks we need to specify the input size. Here we consider the case where $Q$ is finite. The infinite case will be covered in Section 4.2.5. Let $m := |Q|$ be the number of states and $e := \sum_{q \in Q} |S_q|$ be the number of transitions with nonzero probability. The three tasks are:

1. Computing the marginal probability $P(o^n)$ of the data $o^n$. This task is performed by the forward algorithm. This is a dynamic programming algorithm with time complexity $O(ne)$ and space requirement $O(m)$.

2. MAP estimation: computing a sequence of states $q^\lambda$ with maximal posterior weight $P(q^\lambda | o^n)$. Note that $\lambda \geq n$. This task is solved using the Viterbi algorithm, again a dynamic programming algorithm with time complexity $O(\lambda e)$ and space complexity $O(\lambda m)$.

3. Parameter estimation. Instead of a single probabilistic transition function P, one often considers a collection of transition functions $\langle \mathrm{P}_\theta \mid \theta \in \Theta \rangle$ indexed by a set of parameters $\Theta$. In this case one often wants to find the parameter $\theta$ for which the HMM using transition function $\mathrm{P}_\theta$ achieves highest likelihood $P(o^n|\theta)$ of the data $o^n$.

   This task is solved using the Baum-Welch algorithm. This is an iterative improvement algorithm (in fact an instance of Expectation Maximisation (EM)) built atop the forward algorithm (and a related dynamic programming algorithm called the backward algorithm).

Since we apply HMMs to sequential prediction, we are mainly concerned with Task 1 and occasionally with Task 2. Task 3 is outside the scope of this study.

We note that the forward and backward algorithms actually compute more information than just the marginal probability $P(o^n)$. They compute $P(q_i^{\mathrm{p}}, o^i)$ (forward) and $P(o^n|q_i^{\mathrm{p}}, o^i)$ (backward) for each $i = 1, \ldots, n$. The forward algorithm can be computed incrementally, and can thus be used for on-line prediction. Forward-backward can be used together to compute $P(q_i^{\mathrm{p}}|o^n)$ for $i = 1, \ldots, n$, a useful tool in data analysis.

Finally, we note that these algorithms are defined e.g. in [66] for HMMs without silent states and with simple distributions on outcomes instead of forecasting systems. All these algorithms can be adapted straightforwardly to our general case. We formulate the forward algorithm for general HMMs in Section 4.2.5 as an example.

### 4.2.2 HMMs as ES-Priors

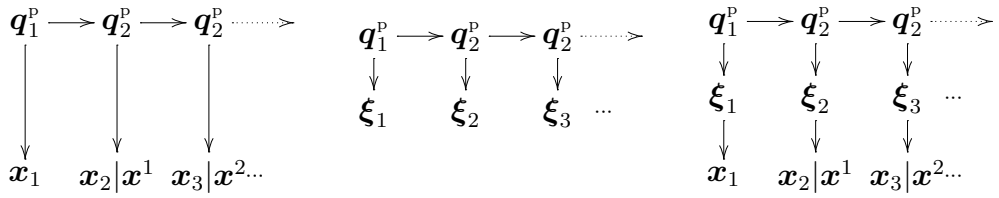
In applications HMMs are often used to model data. This is a good idea whenever there are local temporal correlations between outcomes. A graphical model depicting this approach is displayed in Figure 4.2(a).

Here we take a different approach; we use HMMs as ES-priors, that is, to specify temporal correlations between the performance of our experts. Thus instead of concrete observations our HMMs will produce sequences of experts, that are never actually observed. Figure 4.2(b). illustrates this approach.

Using HMMs as priors allows us to use the standard algorithms of Section 4.2.1 to answer questions about the prior. For example, we can use the forward algorithm to compute the prior probability of the sequence of one hundred experts that issues the first expert at all odd time-points and the second expert at all even moments.

Of course, we are often interested in questions about the data rather than about the prior. In Section 4.2.4 we show how joints based on HMM priors (Figure 4.2(c)) can be transformed into ordinary HMMs (Figure 4.2(a)) with at most a $|\Xi|$-fold increase in size, allowing us to use the standard algorithms of Section 4.2.1 not only for the experts, but for the data as well, with the same

**Figure 4.2** HMMs. $\boldsymbol{q}_i^{\mathrm{p}}$, $\boldsymbol{\xi}_i$ and $\boldsymbol{x}_i$ are the $i^{\text{th}}$ productive state, expert and observation.

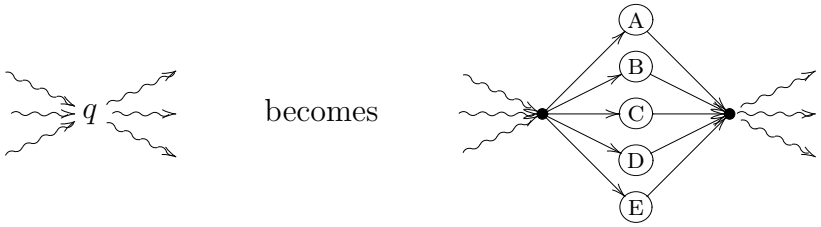



(a) Standard use of HMM (b) HMM ES-prior (c) Application to data

increase in complexity. This is the best we can generally hope for, as we now need to integrate over all possible expert sequences instead of considering only a single one. Here we first consider properties of HMMs that represent ES-priors.

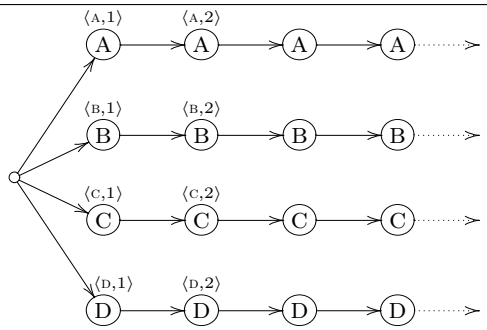
**Restriction** HMM priors "generate", or define the distribution on, sequences of experts. But contrary to the data, which are observed, no concrete sequence of experts is realised. This means that we cannot condition the distribution on experts in a productive state $q_n^{\mathrm{p}}$ on the sequence of previously produced experts $\xi^{n-1}$. In other words, we can only use an HMM on $\Xi$ as an ES-prior if the forecasting systems in its states are simply distributions, so that all dependencies between consecutive experts are carried by the state. This is necessary to avoid having to sum over all (exponentially many) possible expert sequences.

**Deterministic** Under the restriction above, but in the presence of silent states, we can make any HMM deterministic in the sense that each forecasting system assigns probability one to a single outcome. We just replace each productive state $q \in Q_{\mathrm{p}}$ by the following gadget:




In the left diagram, the state $q$ has distribution $P_q$ on outcomes $\mathcal{O} = \{\mathrm{A}, \ldots, \mathrm{E}\}$. In the right diagram, the leftmost silent state has transition probability $P_q(o)$ to a state that deterministically outputs outcome $o$. We often make the functional relationship explicit and call $\langle Q, Q_{\mathrm{p}}, \mathrm{P}_{\mathrm{o}}, \mathrm{P}, \Lambda \rangle$ a *deterministic HMM* on $\mathcal{O}$ if $\Lambda : Q_{\mathrm{p}} \to \mathcal{O}$. Here we slightly abuse notation; the last component of a (general) HMM assigns a *PFS* to each productive state, while the last component of a deterministic HMM assigns an *outcome* to each productive states.

**Figure 4.3** Combination of four experts using a standard Bayesian mixture.




Sequential prediction using a general HMM or its deterministic counterpart costs the same amount of work: the $|\mathcal{O}|$-fold increase in the number of states is compensated by the $|\mathcal{O}|$-fold reduction in the number of outcomes that need to be considered per state.

**Diagrams** Deterministic HMMs can be graphically represented by pictures. In general, we draw a node $N_q$ for each state $q$. We draw a small black dot, e.g. •, for a silent state, and an ellipse labelled $\Lambda(q)$, e.g. Ⓓ, for a productive state. We draw an arrow from $N_q$ to $N_{q'}$ if $q'$ is a direct successor of $q$. We often reify the initial distribution $\mathrm{P}_\circ$ by including a virtual node, drawn as an open circle, e.g. ∘, with an outgoing arrow to $N_q$ for each initial state $q \in I$. The transition probability $P(q \to q')$ is not displayed in the graph.

### 4.2.3 Examples

We are now ready to give the deterministic HMMs that correspond to the ES-priors of our earlier examples from Section 4.1.1: Bayesian mixtures and elementwise mixtures with fixed parameters.

**Example 11** (HMM for Bayesian Mixtures)**.** The Bayesian mixture ES-prior $\pi_{\text{bayes}}$ as introduced in Example 9 represents the hypothesis that a single expert predicts best for all sample sizes. A simple deterministic HMM that generates the prior $\pi_{\text{bayes}}$ is given by $\mathbb{A}_{\text{bayes}} = \langle Q, Q_{\text{p}}, \mathrm{P}, \mathrm{P}_\circ, \Xi, \Lambda \rangle$, where

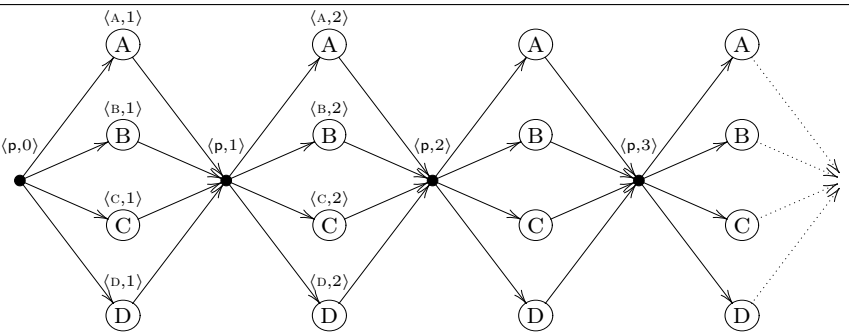
$$Q = Q_{\text{p}} = \Xi \times \mathbb{Z}^+ \qquad \mathrm{P}\left(\langle \xi, n \rangle \to \langle \xi, n+1 \rangle\right) = 1 \tag{4.6a}$$

$$\Lambda(\xi, n) = \xi \qquad \mathrm{P}_\circ(\xi, 1) = w(\xi) \tag{4.6b}$$

The diagram of (4.6) is displayed in Figure 4.3. From the picture of the HMM it is clear that it computes the Bayesian mixture. Hence, using (4.4), the loss of the HMM with prior $w$ is bounded for all $x^n$ by

$$-\log P_{\mathbb{A}_{\text{bayes}}}(x^n) + \log P_\xi(x^n) \leq -\log w(\xi) \qquad \text{for all experts } \xi. \tag{4.7}$$

**Figure 4.4** Combination of four experts using a fixed elementwise mixture




In particular this bound holds for $\hat{\xi} = \arg\max_\xi P_\xi(x^n)$, so we predict as well as the single best expert with *constant* overhead. Also $P_{\mathbb{A}_{\text{bayes}}}(x^n)$ can obviously be computed in $O(n\,|\Xi|)$ using its definition (4.4). We show in Section 4.2.5 that computing it using the HMM prior above gives the same running time $O(n\,|\Xi|)$, a perfect match. ◊

**Example 12** (HMM for Elementwise Mixtures)**.** We now present the deterministic HMM $\mathbb{A}_{\text{mix},\alpha}$ that implements the ES-prior $\pi_{\text{mix},\alpha}$ of Example 10. Its diagram is displayed in Figure 4.4. The HMM has a single silent state per outcome, and its transition probabilities are the mixture weights $\alpha$. Formally, $\mathbb{A}_{\text{mix},\alpha}$ is given using $Q = Q_{\text{s}} \cup Q_{\text{p}}$ by

$$Q_{\text{s}} = \{\mathsf{p}\} \times \mathbb{N} \quad Q_{\text{p}} = \Xi \times \mathbb{Z}^+ \quad \text{P}_\circ(\mathsf{p}, 0) = 1 \quad \Lambda(\xi, n) = \xi \tag{4.8a}$$

$$\text{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \xi, n+1\rangle \\ \langle \xi, n\rangle \to \langle \mathsf{p}, n\rangle \end{pmatrix} = \begin{pmatrix} \alpha(\xi) \\ 1 \end{pmatrix} \tag{4.8b}$$

The vector-style definition of P is shorthand for one P per line. We show in Section 4.2.5 that this HMM allows us to compute $P_{\mathbb{A}_{\text{mix},\alpha}}(x^n)$ in time $O(n\,|\Xi|)$. ◊

### 4.2.4 The HMM for Data

We obtain our model for the data (Figure 4.2(c)) by composing an HMM prior on $\Xi^\infty$ with a PFS $P_\xi$ for each expert $\xi \in \Xi$. We now show that the resulting marginal distribution on data can be implemented by a single HMM on $\mathcal{X}$ (Figure 4.2(a)) *with the same number of states as the HMM prior.* Let $P_\xi$ be an $\mathcal{X}$-forecasting system for each $\xi \in \Xi$, and let the ES-prior $\pi_{\mathbb{A}}$ be given by the deterministic HMM $\mathbb{A} = \langle Q, Q_{\text{p}}, \text{P}_\circ, \text{P}, \Lambda\rangle$ on $\Xi$. Then the marginal distribution of the data (see (4.1)) is given by

$$P_{\mathbb{A}}(x^n) = \sum_{\xi^n} \pi_{\mathbb{A}}(\xi^n) \prod_{i=1}^{n} P_{\xi_i}(x_i|x^{i-1}).$$

The HMM $\mathbb{X} := \left\langle Q, Q_{\text{p}}, \text{P}_\circ, \text{P}, \left\langle P_{\Lambda(q)}\right\rangle_{q\in Q_{\text{p}}}\right\rangle$ on $\mathcal{X}$ induces the same marginal distribution (see Definition 4.2.2). That is, $P_{\mathbb{X}}(x^n) = P_{\mathbb{A}}(x^n)$. Moreover, $\mathbb{X}$ contains

only the forecasting systems that also exist in $\mathbb{A}$ and it retains the structure of $\mathbb{A}$. In particular this means that the HMM algorithms of Section 4.2.1 have the *same* running time on the prior $\mathbb{A}$ as on the marginal $\mathbb{X}$.

### 4.2.5 The Forward Algorithm and Sequential Prediction

We claimed in Section 4.2.1 that the standard HMM algorithms could easily be extended to our HMMs with silent states and forecasting systems. In this section we give the main example: the forward algorithm. We will also show how it can be applied to sequential prediction. Recall that the forward algorithm computes the marginal probability $P(x^n)$ for fixed $x^n$. On the other hand, sequential prediction means predicting the next *observation* $\boldsymbol{x}_{n+1}$ for given data $x^n$, i.e. computing its distribution. For this it suffices to predict the next *expert* $\boldsymbol{\xi}_{n+1}$; we then simply predict $\boldsymbol{x}_{n+1}$ by averaging the expert's predictions accordingly: $P(x_{n+1}|x^n) = \mathrm{E}[P_{\boldsymbol{\xi}_{n+1}}(x_{n+1}|x^n)]$.

We first describe the preprocessing step called *unfolding* and introduce notation for nodes. We then give the forward algorithm, prove its correctness and analyse its running time and space requirement. The forward algorithm can be used for prediction with expert advice. We conclude by outlining the difficulty of adapting the Viterbi algorithm for MAP estimation to the expert setting.

**Unfolding** Every HMM can be transformed into an equivalent HMM in which each productive state is involved in the production of a unique outcome. The single node in Figure 4.6(a) is involved in the production of $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ In its unfolding Figure 4.6(b) the $i^{\text{th}}$ node is only involved in producing $\boldsymbol{x}_i$. Figures 4.6(c) and 4.6(d) show HMMs that unfold to the Bayesian mixture shown in Figure 4.3 and the elementwise mixture shown in Figure 4.4. In full generality, fix an HMM $\mathbb{A}$. The *unfolding* of $\mathbb{A}$ is the HMM

$$\mathbb{A}^{\mathrm{u}} := \left\langle Q^{\mathrm{u}}, Q^{\mathrm{u}}_{\mathrm{p}}, \mathrm{P}^{\mathrm{u}}_{\circ}, \mathrm{P}^{\mathrm{u}}, \left\langle P^{\mathrm{u}}_q \right\rangle_{q \in Q^{\mathrm{u}}} \right\rangle,$$

where the states and productive states are given by:

$$Q^{\mathrm{u}} := \left\{ \langle q_\lambda, n\rangle \mid q^\lambda \text{ is a run through } \mathbb{A} \right\}, \quad \text{where } n = \left| q^\lambda_{\mathrm{p}} \right| \tag{4.9a}$$

$$Q^{\mathrm{u}}_{\mathrm{p}} := Q^{\mathrm{u}} \cap (Q_{\mathrm{p}} \times \mathbb{N}) \tag{4.9b}$$

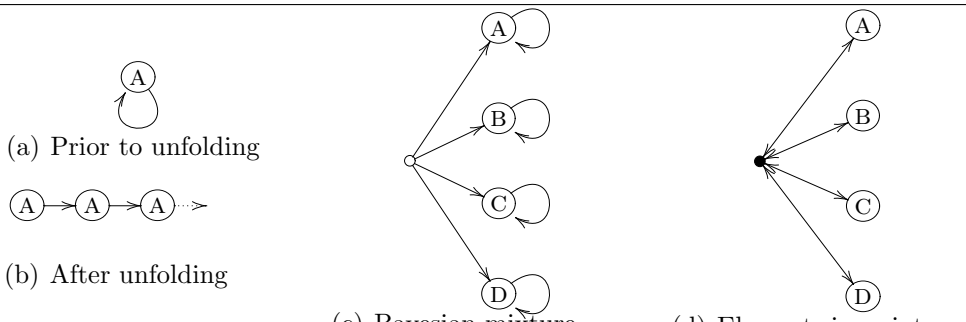
and the initial probability, transition function and forecasting systems are:

$$\mathrm{P}^{\mathrm{u}}_{\circ}\left(\langle q, 0\rangle\right) := \mathrm{P}_{\circ}(q) \tag{4.9c}$$

$$\mathrm{P}^{\mathrm{u}}\begin{pmatrix} \langle q, n\rangle \to \langle q', n+1\rangle \\ \langle q, n\rangle \to \langle q', n\rangle \end{pmatrix} := \begin{pmatrix} \mathrm{P}(q \to q') \\ \mathrm{P}(q \to q') \end{pmatrix} \tag{4.9d}$$

$$P^{\mathrm{u}}_{\langle q,n\rangle} := P_q \tag{4.9e}$$

**Figure 4.5** Unfolding example




First observe that unfolding preserves the marginal: $P_{\mathbb{A}}(o^n) = P_{\mathbb{A}^{\mathrm{u}}}(o^n)$. Second, unfolding is an idempotent operation: $(\mathbb{A}^{\mathrm{u}})^{\mathrm{u}}$ is isomorphic to $\mathbb{A}^{\mathrm{u}}$. Third, unfolding renders the set of states infinite, but for each $n$ it preserves the number of states reachable in exactly $n$ steps.

**Order** The states in an unfolded HMM have earlier-later structure. Fix $q, q' \in Q^{\mathrm{u}}$. We write $q < q'$ iff there is a run to $q'$ through $q$. We call $<$ the *natural order* on $Q^{\mathrm{u}}$. Obviously $<$ is a partial order, furthermore it is the transitive closure of the reverse direct successor relation. It is well-founded, allowing us to perform induction on states, an essential ingredient in the forward algorithm (Algorithm 4.1) and its correctness proof (Theorem 4.2.3).

**Interval Notation** We introduce interval notation to address subsets of states of unfolded HMMs, as illustrated by Figure 4.6. Our notation associates each productive state with the sample size at which it produces its outcome, while the silent states fall in between. We use intervals with borders in $\mathbb{N}$. The interval contains the border $i \in \mathbb{N}$ iff the addressed set of states includes the states where the $i^{\text{th}}$ observation is produced.

$$Q^{\mathrm{u}}_{[n,m)} := Q^{\mathrm{u}} \cap (Q \times [n,m)) \qquad Q^{\mathrm{u}}_{[n,m]} := Q^{\mathrm{u}}_{[n,m)} \cup Q^{\mathrm{u}}_{\{m\}} \tag{4.10a}$$

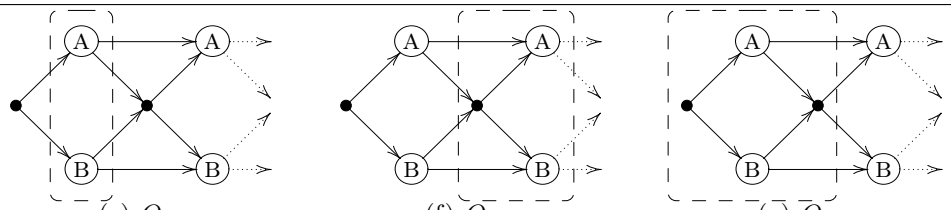
$$Q^{\mathrm{u}}_{\{n\}} := Q^{\mathrm{u}} \cap (Q_{\mathrm{p}} \times \{n\}) \qquad Q^{\mathrm{u}}_{(n,m)} := Q^{\mathrm{u}}_{[n,m)} \setminus Q^{\mathrm{u}}_{\{n\}} \tag{4.10b}$$

$$Q^{\mathrm{u}}_{(n,m]} := Q^{\mathrm{u}}_{[n,m]} \setminus Q^{\mathrm{u}}_{\{n\}} \tag{4.10c}$$

Fix $n > 0$, then $Q^{\mathrm{u}}_{\{n\}}$ is a non-empty $<$-anti-chain (i.e. its states are pairwise $<$-incomparable). Furthermore $Q^{\mathrm{u}}_{(n,n+1)}$ is empty iff $Q^{\mathrm{u}}_{\{n+1\}} = \bigcup_{q \in Q^{\mathrm{u}}_{\{n\}}} S_q$, in other words, if there are no silent states between sample sizes $n$ and $n+1$.

**The Forward Algorithm** Fix an unfolded deterministic HMM prior $\mathbb{A} = \langle Q, Q_{\mathrm{p}}, \mathrm{P}_{\circ}, \mathrm{P}, \Lambda \rangle$ on $\Xi$, and an $\mathcal{X}$-PFS $P_\xi$ for each expert $\xi \in \Xi$. The input consists of a sequence $x^\infty$ that arrives sequentially. Then the forward algorithm for sequential prediction on models with silent states can be rendered as follows.

**Figure 4.6** Interval notation




(e) $Q_{\{1\}}$ (f) $Q_{(1,2]}$ (g) $Q_{(0,2)}$

**Analysis** Consider a state $q \in Q$, say $q \in Q_{[n,n+1)}$. Initially, $q \notin \mathrm{dom}(w)$. Then at some point $w(q) \leftarrow \mathrm{P}_\circ(q)$. This happens either in the second line because $q \in I$ or in Forward Propagation because $q \in S_u$ for some $u$ (in this case $\mathrm{P}_\circ(q) = 0$). Then $w(q)$ accumulates weight as its direct predecessors are processed in Forward Propagation. At some point all its predecessors have been processed. If $q$ is productive we call its weight at this point (that is, just before Loss Update) $\mathrm{Alg}(\mathbb{A}, x^{n-1}, q)$. Finally, Forward Propagation removes $q$ from the domain of $w$, never to be considered again. We call the weight of $q$ (silent or productive) just before removal $\mathrm{Alg}(\mathbb{A}, x^n, q)$.

Note that we associate *two* weights with each productive state $q \in Q_{\{n\}}$: the weight $\mathrm{Alg}(\mathbb{A}, x^{n-1}, q)$ is calculated *before* outcome $n$ is observed, while on the other hand $\mathrm{Alg}(\mathbb{A}, x^n, q)$ denotes the weight *after* the loss update incorporates outcome $n$.

**Theorem 4.2.3.** *Fix an HMM prior $\mathbb{A}$, $n \in \mathbb{N}$ and $q \in Q_{[n,n+1]}$, then*

$$\mathrm{Alg}(\mathbb{A}, x^n, q) = P_{\mathbb{A}}(x^n, q).$$

Note that the theorem applies twice to productive states.

*Proof.* By $<$-induction on states. Let $q \in Q_{(n,n+1]}$, and suppose that the theorem holds for all $q' < q$. Let $B_q = \left\{q' \mid \mathrm{P}(q' \to q) > 0\right\}$ be the set of direct predecessors of $q$. Observe that $B_q \subseteq Q_{[n,n+1)}$. The weight that is accumulated by Forward Propagation$(n)$ onto $q$ is:

$$\begin{aligned}\mathrm{Alg}(\mathbb{A}, x^n, q) &= \mathrm{P}_\circ(q) + \sum_{q' \in B_q} \mathrm{P}(q' \to q)\, \mathrm{Alg}(\mathbb{A}, x^n, q') \\ &= \mathrm{P}_\circ(q) + \sum_{q' \in B_q} \mathrm{P}(q' \to q) P_{\mathbb{A}}(x^n, q') \;=\; P_{\mathbb{A}}(x^n, q).\end{aligned}$$

The second equality follows from the induction hypothesis. Additionally if $q \in Q_{\{n\}}$ is productive, say $\Lambda(q) = \xi$, then after Loss Update$(n)$ its weight is:

$$\begin{aligned}\mathrm{Alg}(\mathbb{A}, x^n, q) &= P_\xi(x_n | x^{n-1})\, \mathrm{Alg}(\mathbb{A}, x^{n-1}, q) \\ &= P_\xi(x_n | x^{n-1}) P_{\mathbb{A}}(x^{n-1}, q) \;=\; P_{\mathbb{A}}(x^n, q).\end{aligned}$$

□

The second inequality holds by induction on $n$, and the third by Definition 4.2.2.

**Complexity** We are now able to sharpen the complexity results as listed in Section 4.2.1, and extend them to infinite HMMs. Fix $\mathbb{A}$, $n \in \mathbb{N}$. The forward algorithm processes each state in $Q_{[0,n)}$ once, and at that point this state's weight is distributed over its successors. Thus, the running time is proportional to $\sum_{q \in Q_{[0,n)}} \left|S_q\right|$. The forward algorithm keeps $\left|\mathrm{dom}(w)\right|$ many weights. But at each sample size $n$, $\mathrm{dom}(w) \subseteq Q_{[n,n+1]}$. Therefore the space needed is at most proportional to $\max_{m<n} \left|Q_{[m,m+1]}\right|$. For both Bayes (Example 11) and elementwise mixtures (Example 12) one may read from the figures that $\sum_{q \in Q_{[n,n+1)}} \left|S_q\right|$ and $\left|Q_{[n,n+1)}\right|$ are $O(|\Xi|)$, so we indeed get the claimed running time $O(n\,|\Xi|)$ and space requirement $O(|\Xi|)$.

**MAP Estimation** The forward algorithm described above computes the probability of the data, that is

$$P(x^n) \;=\; \sum_{q^\lambda : q_\lambda \in Q_{\{n\}}} P(x^n, q^\lambda).$$

Instead of the entire sum, we are sometimes interested in the sequence of states $q^\lambda$ that contributes most to it:

$$\arg\max_{q^\lambda} P(x^n, q^\lambda) \;=\; \arg\max_{q^\lambda} P(x^n|q^\lambda)\pi(q^\lambda).$$

The Viterbi algorithm [66] is used to compute the most likely sequence of states for HMMs. It can be easily adapted to handle silent states. However, we may also write

$$P(x^n) \;=\; \sum_{\xi^n} P(x^n, \xi^n),$$

and wonder about the sequence of experts $\xi^n$ that contributes most. This problem is harder because in general, a single sequence of experts can be generated by many different sequences of states. This is unrelated to the presence of the silent states, but due to different states producing the same expert simultaneously (i.e. in the same $Q_{\{n\}}$). So we cannot use the Viterbi algorithm as it is. The Viterbi algorithm can be extended to compute the MAP expert sequence for general HMMs, but the resulting running time explodes. Still, the MAP $\xi^n$ can be sometimes be obtained efficiently by exploiting the structure of the HMM at hand. The first example is the unambiguous HMMs. A deterministic HMM is *ambiguous* if it has two runs that agree on the sequence of *experts* produced, but not on the sequence of *productive states*. The straightforward extension of the Viterbi algorithm works for unambiguous HMMs. The second important example is the (ambiguous) switch HMM that we introduce in Section 4.4.1. We show how to compute its MAP expert sequence in Section 4.4.1.

## 4.3 Zoology

Perhaps the simplest way to predict using a number of experts is to pick one of them and mirror her predictions exactly. Beyond this "fixed expert model", we have considered two methods of combining experts so far, namely taking Bayesian mixtures, and taking elementwise mixtures as described in Section 4.2.3. Figure 4.1 shows these and a number of other, more sophisticated methods that fit in our framework. The arrows indicate which methods are generalised by which other methods. They have been partitioned in groups that can be computed in the same amount of time using HMMs.

We have presented two examples so far, the Bayesian mixture and the elementwise mixture with fixed coefficients (Examples 11 and 12). The latter model is parameterised. Choosing a fixed value for the parameter beforehand is often difficult. The first model we discuss learns the optimal parameter value on-line, at the cost of only a small additional loss. We then proceed to discuss a number of important existing expert models.

### 4.3.1 Universal Elementwise Mixtures

A distribution is "universal" for a family of distributions if it incurs small additional loss compared to the best member of the family. A standard Bayesian mixture constitutes the simplest example. It is universal for the fixed expert model, where the unknown parameter is the used expert. In (4.7) we showed that the additional loss is at most $\log|\Xi|$ for the uniform prior.

In Example 12 we described elementwise mixtures with fixed coefficients as ES-priors. Prior knowledge about the mixture coefficients is often unavailable. We now expand this model to learn the optimal mixture coefficients from the data. To this end we place a prior distribution $w$ on the space of mixture weights $\triangle(\Xi)$. Using (4.5) we obtain the following marginal distribution:

$$P_{\text{umix}}(x^n) = \int_{\triangle(\Xi)} P_{\text{mix},\alpha}(x^n) w(\alpha)\,\mathrm{d}\alpha = \int_{\triangle(\Xi)} \sum_{\xi^n} P_{\xi^n}(x^n)\pi_{\text{mix},\alpha}(\xi^n) w(\alpha)\,\mathrm{d}\alpha$$
$$= \sum_{\xi^n} P_{\xi^n}(x^n)\pi_{\text{umix}}(\xi^n), \quad \text{where} \quad \pi_{\text{umix}}(\xi^n) = \int_{\triangle(\Xi)} \pi_{\text{mix},\alpha}(\xi^n) w(\alpha)\,\mathrm{d}\alpha. \quad (4.11)$$

Thus $P_{\text{umix}}$ is the ES-joint with ES-prior $\pi_{\text{umix}}$. This applies more generally: parameters $\alpha$ can be integrated out of an ES-prior regardless of which experts are used, since the expert predictions $P_{\xi^n}(x^n)$ do not depend on $\alpha$.

We will proceed to calculate a loss bound for the universal elementwise mixture model, showing that it really is universal. After that we will describe how it can be implemented as a HMM.

### A Loss Bound

In this section we relate the loss of a universal elementwise mixture with the loss obtained by the maximum likelihood elementwise mixture. While mixture models occur regularly in the statistical literature, we are not aware of any appearance in universal prediction. Therefore, to the best of our knowledge, the following simple loss bound is new. Our goal is to obtain a bound in terms of properties of the prior. A difficulty here is that there are many expert sequences exhibiting mixture frequencies close to the maximum likelihood mixture weights, so that each individual expert sequence contributes relatively little to the total probability (4.11). The following theorem is a general tool to deal with such situations.

**Theorem 4.3.1.** *Let $\pi$, $\rho$ be ES-priors s.t. $\rho$ is zero whenever $\pi$ is. Then for all $x^n$, reading $0/0 = 0$,*

$$-\log \frac{P_\pi(x^n)}{P_\rho(x^n)} \;\leq\; E_{P_\rho}\left[-\log \frac{\pi(\boldsymbol{\xi}^n)}{\rho(\boldsymbol{\xi}^n)}\middle| x^n\right] \;\leq\; -\log \max_{\xi^n} \frac{\pi(\xi^n)}{\rho(\xi^n)}.$$

*Proof.* For non-negative $a_1, \ldots a_m$ and $b_1, \ldots b_m$:

$$\left(\sum_{i=1}^m a_i\right) \log \frac{\sum_{i=1}^m a_i}{\sum_{i=1}^m b_i} \;\leq\; \sum_{i=1}^m a_i \log \frac{a_i}{b_i} \;\leq\; \left(\sum_{i=1}^m a_i\right) \max_i \log \frac{a_i}{b_i}. \tag{4.12}$$

The first inequality is the log sum inequality [25, Theorem 2.7.1]. The second inequality is a simple overestimation. We now apply (4.12) substituting $m \mapsto |\Xi^n|$, $a_{\xi^n} \mapsto P_\rho(x^n, \xi^n)$ and $b_{\xi^n} \mapsto P_\pi(x^n, \xi^n)$ and divide by $\sum_{i=1}^m a_i$ to complete the proof. □

Using this theorem, we obtain a loss bound for universal elementwise mixtures that can be computed prior to observation and without reference to the experts' PFSs.

**Corollary 4.3.2.** *Let $P_{\text{umix}}$ be the universal elementwise mixture model defined using the $(\frac{1}{2}, \ldots, \frac{1}{2})$-Dirichlet prior (that is, Jeffreys' prior) as the prior $w(\alpha)$ in (4.11). Let $\hat{\alpha}(x^n)$ maximise the likelihood $P_{\text{mix},\alpha}(x^n)$ w.r.t. $\alpha$. Then for all $x^n$ the additional loss incurred by the universal elementwise mixture is bounded thus*

$$-\log P_{\text{umix}}(x^n) + \log P_{\text{mix},\hat{\alpha}(x^n)}(x^n) \;\leq\; \frac{|\Xi| - 1}{2} \log \frac{n}{\pi} + c$$

*for a fixed constant $c$.*

*Proof.* By Theorem 4.3.1

$$-\log P_{\text{umix}}(x^n) + \log P_{\text{mix},\hat{\alpha}(x^n)}(x^n) \;\leq\; \max_{\xi^n} \left(-\log \pi_{\text{umix}}(\xi^n) + \log \pi_{\text{mix},\hat{\alpha}(x^n)}(\xi^n)\right). \tag{4.13}$$

We now bound the right hand side. Let $\hat{\alpha}(\xi^n)$ maximise $\pi_{\mathrm{mix},\alpha}(\xi^n)$ w.r.t. $\alpha$. Then for all $x^n$ and $\xi^n$

$$\pi_{\mathrm{mix},\hat{\alpha}(x^n)}(\xi^n) \;\leq\; \pi_{\mathrm{mix},\hat{\alpha}(\xi^n)}(\xi^n). \tag{4.14}$$

For the $\left(\frac{1}{2},\ldots,\frac{1}{2}\right)$-Dirichlet prior, for all $\xi^n$

$$-\log \pi_{\mathrm{umix}}(\xi^n) + \log \pi_{\mathrm{mix},\hat{\alpha}(\xi^n)}(\xi^n) \;\leq\; \frac{|\Xi|-1}{2}\log\frac{n}{\pi} + c$$

for some fixed constant $c$ (see e.g. [100]) Combination with (4.14) and (4.13) completes the proof. □

Since the overhead incurred as a penalty for not knowing the optimal parameter $\hat{\alpha}$ in advance is only logarithmic, we find that $P_{\mathrm{umix}}$ is strongly universal for the fixed elementwise mixtures.

**HMM**

While universal elementwise mixtures can be described using the ES-prior $\pi_{\mathrm{umix}}$ defined in (4.11), unfortunately any HMM that computes it needs a state for each possible count vector, and is therefore huge if the number of experts is large. The HMM $\mathbb{A}_{\mathrm{umix}}$ for an arbitrary number of experts using the $\left(\frac{1}{2},\ldots,\frac{1}{2}\right)$-Dirichlet prior is given using $Q = Q_{\mathrm{s}} \cup Q_{\mathrm{p}}$ by

$$Q_{\mathrm{s}} = \mathbb{N}^{\Xi} \quad Q_{\mathrm{p}} = \mathbb{N}^{\Xi} \times \Xi \quad \mathrm{P}_{\circ}(\mathbf{0}) = 1 \quad \Lambda(\vec{n},\xi) = \xi \tag{4.15}$$

$$\mathrm{P}\begin{pmatrix} \langle \vec{n} \rangle \to \langle \vec{n},\xi \rangle \\ \langle \vec{n},\xi \rangle \to \langle \vec{n} + \mathbf{1}_\xi \rangle \end{pmatrix} = \begin{pmatrix} \frac{1/2+n_\xi}{|\Xi|/2+\sum_\xi n_\xi} \\ 1 \end{pmatrix} \tag{4.16}$$

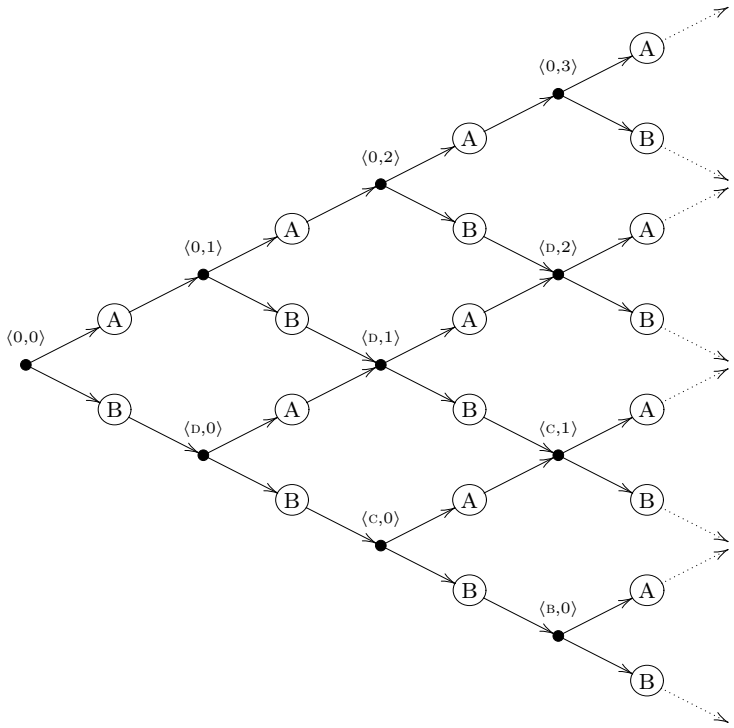
We write $\mathbb{N}^{\Xi}$ for the set of assignments of counts to experts; $\mathbf{0}$ for the all zero assignment, and $\mathbf{1}_\xi$ marks one count for expert $\xi$. We show the diagram of $\mathbb{A}_{\mathrm{umix}}$ for the practical limit of two experts in Figure 4.7. In this case, the forward algorithm has running time $O(n^2)$. Each productive state in Figure 4.7 corresponds to a vector of two counts $(n_1, n_2)$ that sum to the sample size $n$, with the interpretation that of the $n$ experts, the first was used $n_1$ times while the second was used $n_2$ times. These counts are a sufficient statistic for the multinomial model: per (4.5b) and (4.11) the probability of the next expert only depends on the counts, and these probabilities are exactly the successor probabilities of the silent states (4.16).

Other priors on $\alpha$ are possible. In particular, when all mass is placed on a single value of $\alpha$, we retrieve the elementwise mixture with fixed coefficients.

### 4.3.2 Fixed Share

The first publication that considers a scenario where the best predicting expert may change with the sample size is Herbster and Warmuth's paper on *tracking*

**Figure 4.7** Combination of two experts using a universal elementwise mixture




*the best expert* [45, 46]. They partition the data of size $n$ into $m$ segments, where each segment is associated with an expert, and give algorithms to predict almost as well as the best *partition* where the best expert is selected per segment. They give two algorithms called fixed share and dynamic share. The second algorithm does not fit in our framework; furthermore its motivation applies only to loss functions other than log-loss. We focus on fixed share, which is in fact identical to our algorithm applied to the HMM depicted in Figure 4.8, where all arcs *into* the silent states have fixed probability $\alpha \in [0, 1]$ and all arcs *from* the silent states have some fixed distribution $w$ on $\Xi$.[2] The same algorithm is also described as an instance of the Aggregating Algorithm in [95]. Fixed share reduces to fixed elementwise mixtures by setting $\alpha = 1$ and to Bayesian mixtures by setting $\alpha = 0$. Formally:

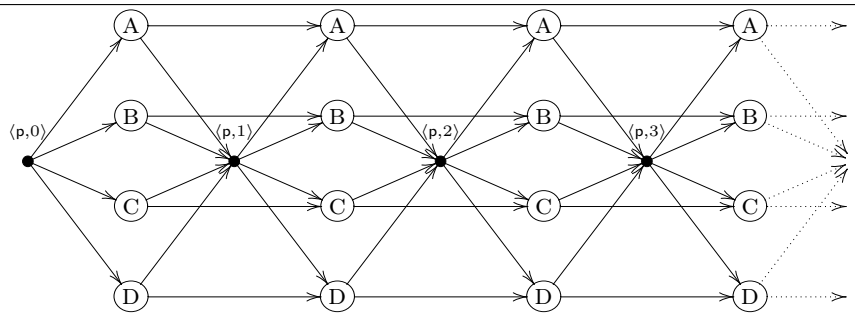
$$\begin{aligned} Q &= \Xi \times \mathbb{Z}^+ \cup \{\mathsf{p}\} \times \mathbb{N} & \mathrm{P}_\circ(\mathsf{p}, 0) &= 1 \\ Q_{\mathrm{p}} &= \Xi \times \mathbb{Z}^+ & \Lambda(\xi, n) &= \xi \end{aligned} \tag{4.17a}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \xi, n+1\rangle \\ \langle \xi, n\rangle \to \langle \mathsf{p}, n\rangle \\ \langle \xi, n\rangle \to \langle \xi, n+1\rangle \end{pmatrix} = \begin{pmatrix} w(\xi) \\ \alpha \\ 1-\alpha \end{pmatrix} \tag{4.17b}$$

Each productive state represents that a particular expert is used at a certain

[2] This is actually a slight generalisation: the original algorithm uses a uniform $w(\xi) = 1/|\Xi|$.

**Figure 4.8** Combination of four experts using the fixed share algorithm




sample size. Once a transition to a silent state is made, all history is forgotten and a new expert is chosen according to $w$.[3]

Let $\hat{L}$ denote the loss achieved by the best partition, with switching rate $\alpha^* := m/(n-1)$. Let $L_{\mathrm{fs},\alpha}$ denote the loss of fixed share with uniform $w$ and parameter $\alpha$. Herbster and Warmuth prove[4]

$$L_{\mathrm{fs},\alpha} - \hat{L} \;\leq\; (n-1)H(\alpha^*,\alpha) + (m-1)\log(|\Xi|-1) + \log|\Xi|\,,$$

which we for brevity loosen slightly to

$$L_{\mathrm{fs},\alpha} - \hat{L} \;\leq\; nH(\alpha^*,\alpha) + m\log|\Xi|\,. \tag{4.18}$$

Here $H(\alpha^*,\alpha) = -\alpha^*\log\alpha - (1-\alpha^*)\log(1-\alpha)$ is the cross entropy. The best loss guarantee is obtained for $\alpha = \alpha^*$, in which case the cross entropy reduces to the binary entropy $H(\alpha)$. A drawback of the method is that the optimal value of $\alpha$ has to be known in advance in order to minimise the loss. In Sections Section 4.3.3 and Section 4.4 we describe a number of generalisations of fixed share that avoid this problem.

### 4.3.3 Universal Share

Volf and Willems describe universal share (they call it *the switching method*) [94], which is very similar to a probabilistic version of Herbster and Warmuth's fixed share algorithm, except that they put a prior on the unknown parameter, with the result that their algorithm adaptively learns the optimal value during prediction.

In [15], Bousquet shows that the overhead for not knowing the optimal parameter value is equal to the overhead of a Bernoulli universal distribution. Let $L_{\mathrm{fs},\alpha} = -\log P_{\mathrm{fs},\alpha}(x^n)$ denote the loss achieved by the fixed share algorithm with parameter $\alpha$ on data $x^n$, and let $L_{\mathrm{us}} = -\log P_{\mathrm{us}}(x^n)$ denote the

[3] Contrary to the original fixed share, we allow switching to the same expert. In the HMM framework this is necessary to achieve running-time $O(n\,|\Xi|)$. Under uniform $w$, non-reflexive switching with fixed rate $\alpha$ can be simulated by reflexive switching with fixed rate $\beta = \frac{\alpha|\Xi|}{|\Xi|-1}$ (provided $\beta \leq 1$). For non-uniform $w$, the rate becomes expert-dependent.

[4] This bound can be obtained for the fixed share HMM using the previous footnote.

loss of universal share, where $P_{\text{us}}(x^n) = \int P_{\text{fs},\alpha}(x^n) w(\alpha)\,\mathrm{d}\alpha$ with Jeffreys' prior $w(\alpha) = \alpha^{-1/2}(1-\alpha)^{-1/2}/\pi$ on $[0,1]$. Then

$$L_{\text{us}} - \min_{\alpha} L_{\text{fs},\alpha} \leq 1 + \tfrac{1}{2}\log n. \tag{4.19}$$

Thus $P_{\text{us}}$ is universal for the model $\{P_{\text{fs},\alpha} \mid \alpha \in [0,1]\}$ that consists of all ES-joints where the ES-priors are distributions with a fixed switching rate.

Universal share requires quadratic running time $O(n^2\,|\Xi|)$, restricting its use to moderately small data sets.

In [63], Monteleoni and Jaakkola place a discrete prior on the parameter that divides its mass over $\sqrt{n}$ well-chosen points, in a setting where the ultimate sample size $n$ is known beforehand. This way they still manage to achieve (4.19) up to a constant, while reducing the running time to $O(n\sqrt{n}\,|\Xi|)$. In [16], Bousquet and Warmuth describe yet another generalisation of expert tracking; they derive good loss bounds in the situation where the best experts for each section in the partition are drawn from a small pool.

The HMM for universal share with the $(\frac{1}{2},\frac{1}{2})$-Dirichlet prior on the switching rate $\alpha$ is displayed in Figure 4.9. It is formally specified (using $Q = Q_{\text{s}} \cup Q_{\text{p}}$) by:

$$\begin{aligned} Q_{\text{s}} &= \{\mathsf{p},\mathsf{q}\} \times \{\langle m,n\rangle \in \mathbb{N}^2 \mid m \leq n\} \\ Q_{\text{p}} &= \quad \Xi \quad \times \{\langle m,n\rangle \in \mathbb{N}^2 \mid m < n\} \end{aligned} \tag{4.20a}$$

$$\Lambda(\xi,m,n) = \xi \qquad \mathrm{P}_\circ(\mathsf{p},0,0) = 1 \tag{4.20b}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p},m,n\rangle \to \langle \xi,m,n+1\rangle \\ \langle \mathsf{q},m,n\rangle \to \langle \mathsf{p},m+1,n\rangle \\ \langle \xi,m,n\rangle \to \langle \mathsf{q},m,n\rangle \\ \langle \xi,m,n\rangle \to \langle \xi,m,n+1\rangle \end{pmatrix} = \begin{pmatrix} w(\xi) \\ 1 \\ (m+\frac{1}{2})/\,n \\ (n-m-\frac{1}{2})/\,n \end{pmatrix} \tag{4.20c}$$

Each productive state $\langle \xi,n,m\rangle$ represents the fact that at sample size $n$ expert $\xi$ is used, while there have been $m$ switches in the past. Note that the last two lines of (4.20c) are subtly different from the corresponding topmost line of (4.16). In a sample of size $n$ there are $n$ possible positions to use a given expert, while there are only $n-1$ possible switch positions.

The presence of the switch count in the state is the new ingredient compared to fixed share. It allows us to adapt the switching probability to the data, but it also renders the number of states quadratic. We discuss reducing the number of states without sacrificing much performance in Section 4.5.1.

### 4.3.4 Overconfident Experts

In [95], Vovk considers overconfident experts. In this scenario, there is a single unknown best expert, except that this expert sometimes makes wild (over-categorical) predictions. We assume that the rate at which this happens is a

**Figure 4.9** Combination of four experts using universal share

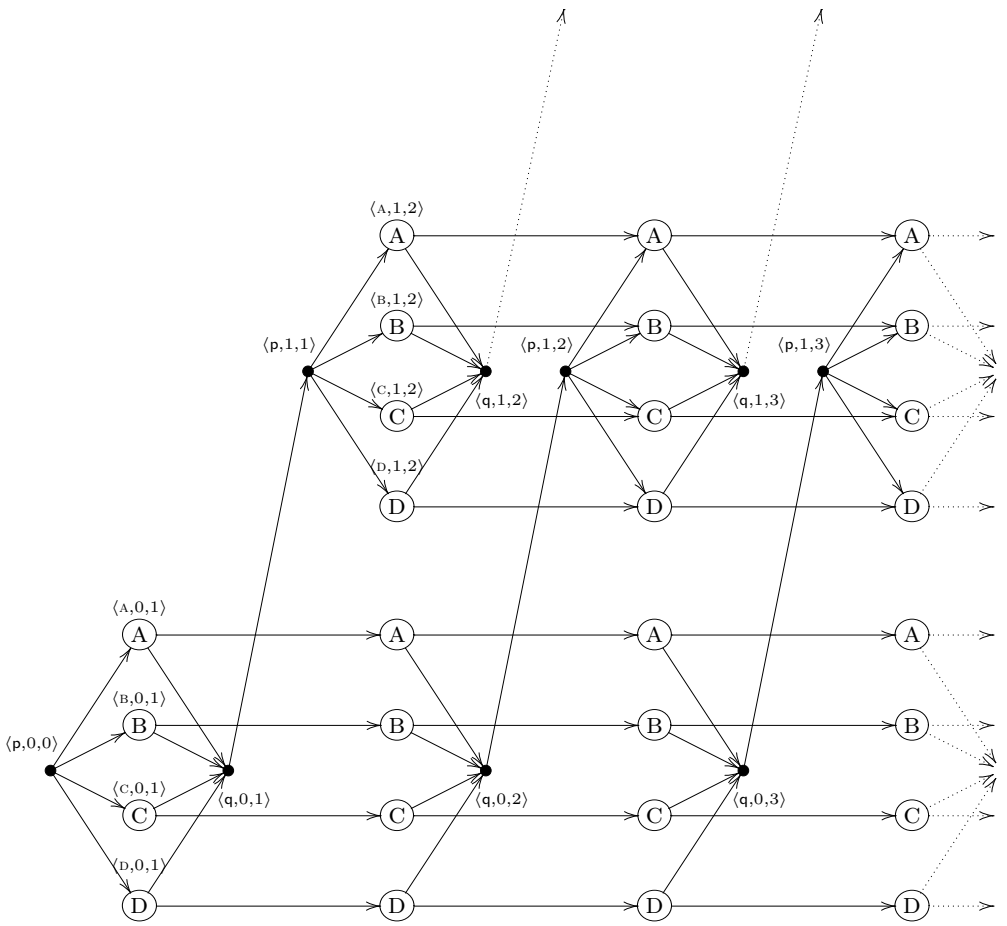
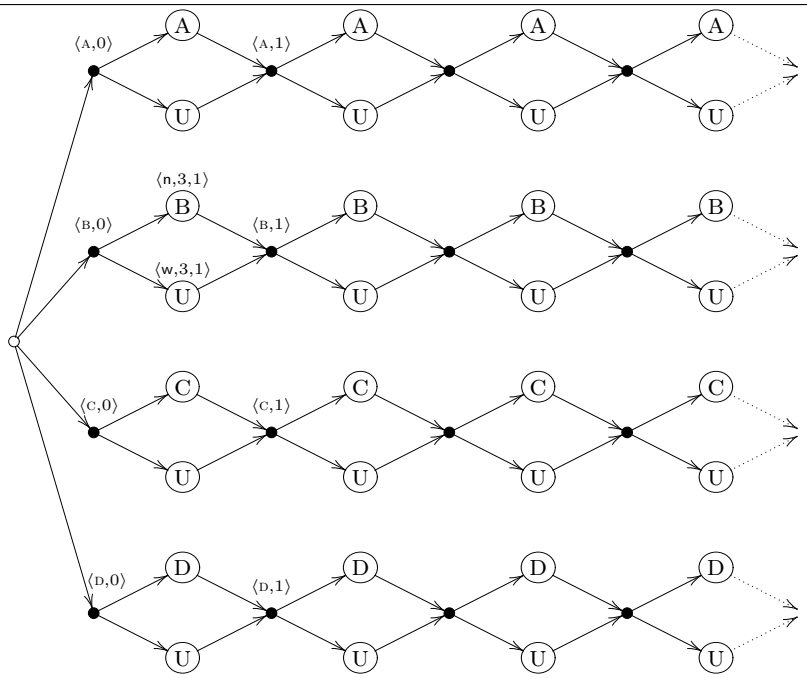


**Figure 4.10** Combination of four overconfident experts




known constant $\alpha$. The overconfident expert model is an attempt to mitigate the wild predictions using an additional "safe" expert $\text{U} \in \Xi$, who always issues the uniform distribution on $\mathcal{X}$ (which we assume to be finite for simplicity here). Using $Q = Q_\text{s} \cup Q_\text{p}$, it is formally specified by:

$$\begin{aligned} Q_\text{s} &= \Xi \times \mathbb{N} & \Lambda(\mathsf{n}, \xi, n) &= \xi \quad \text{P}_\circ(\xi, 0) = w(\xi) \\ Q_\text{p} &= \{\mathsf{n}, \mathsf{w}\} \times \Xi \times \mathbb{Z}^+ & \Lambda(\mathsf{w}, \xi, n) &= u \end{aligned} \tag{4.21a}$$

$$\text{P} \begin{pmatrix} \langle \xi, n \rangle \to \langle \mathsf{n}, \xi, n+1 \rangle \\ \langle \xi, n \rangle \to \langle \mathsf{w}, \xi, n+1 \rangle \\ \langle \mathsf{n}, \xi, n \rangle \to \langle \xi, n \rangle \\ \langle \mathsf{w}, \xi, n \rangle \to \langle \xi, n \rangle \end{pmatrix} = \begin{pmatrix} 1-\alpha \\ \alpha \\ 1 \\ 1 \end{pmatrix} \tag{4.21b}$$

Each productive state corresponds to the idea that a certain expert is best, and additionally whether the current outcome is normal or wild.

Fix data $x^n$. Let $\hat{\xi}^n$ be the expert sequence that maximises the likelihood $P_{\xi^n}(x^n)$ among all expert sequences $\xi^n$ that switch between a single expert and U. To derive our loss bound, we underestimate the marginal probability $P_{\text{oce},\alpha}(x^n)$ for the HMM defined above, by dropping all terms except the one for $\hat{\xi}^n$.

$$P_{\text{oce},\alpha}(x^n) \quad = \sum_{\xi^n \in \Xi^n} \pi_{\text{oce},\alpha}(\xi^n) P_{\xi^n}(x^n) \quad \geq \quad \pi_{\text{oce},\alpha}(\hat{\xi}^n) P_{\hat{\xi}^n}(x^n). \tag{4.22}$$

(This first step is also used in the bounds for the two new models in Section 4.4.) Let $\alpha^*$ denote the frequency of occurrence of U in $\hat{\xi}^n$, let $\xi_\text{best}$ be the other expert

that occurs in $\xi^n$, and let $\hat{L} = -\log P_{\hat{\xi}^n}(x^n)$. We can now bound our worst-case additional loss:

$$-\log P_{\mathrm{oce},\hat{\alpha}}(x^n) - \hat{L} \leq -\log \pi_{\mathrm{oce},\alpha}(\hat{\xi}^n) = -\log w(\xi_{\mathrm{best}}) + nH(\alpha^*, \alpha).$$

Again $H$ denotes the cross entropy. From a coding perspective, after first specifying the best expert $\xi_{\mathrm{best}}$ and a binary sequence representing $\hat{\xi}^n$, we can then use $\hat{\xi}^n$ to encode the actual observations with optimal efficiency.

The optimal misprediction rate $\alpha$ is usually not known in advance, so we can again learn it from data by placing a prior on it and integrating over this prior. This comes at the cost of an additional loss of $\frac{1}{2}\log n + c$ bits for some constant $c$ (which is $\leq 1$ for two experts), and as will be shown in the next subsection, can be implemented using a quadratic time algorithm.

### Recursive Combination

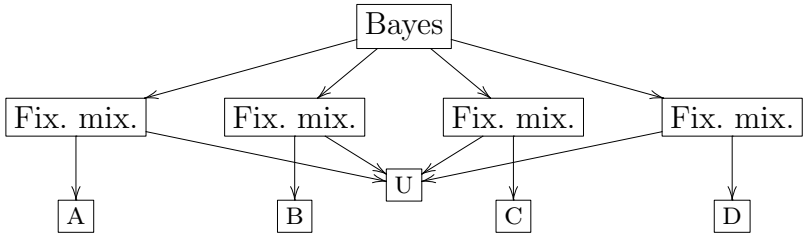
In Figure 4.10 one may recognise two simpler HMMs: it is in fact just a Bayesian combination of a set of fixed elementwise mixtures with some parameter $\alpha$, one for each expert. Thus two models for combining expert predictions, the Bayesian model and fixed elementwise mixtures, have been recursively combined into a single new model. This view is illustrated in Figure 4.11.

More generally, any method to combine the predictions of multiple experts into a single new prediction strategy, can itself be considered an expert. We can apply our method recursively to this new "meta-expert"; the running time of the recursive combination is only the *sum* of the running times of all the component predictors. For example, if all used individual expert models can be evaluated in quadratic time, then the full recursive combination also has quadratic running time, *even though it may be impossible to specify using an HMM of quadratic size.*

Although a recursive combination to implement overconfident experts may save some work, the same running time may be achieved by implementing the HMM depicted in Figure 4.10 directly. However, we can also obtain efficient generalisations of the overconfident expert model, by replacing any combinator by a more sophisticated one. For example, rather than a fixed elementwise mixture, we could use a universal elementwise mixture for each expert, so that the error frequency is learned from data. Or, if we suspect that an expert may not only make incidental slip-ups, but actually become completely untrustworthy for longer stretches of time, we may even use a fixed or universal share model.

One may also consider that the fundamental idea behind the overconfident expert model is to combine each expert with a uniform predictor using a misprediction model. In the example in Figure 4.11, this idea is used to "smooth" the expert predictions, which are then used at the top level in a Bayesian combination. However, the model that is used at the top level is completely orthogonal to the model used to smooth expert predictions; we can safeguard against overconfident experts not only in Bayesian combinations but also in other models such as

**Figure 4.11** Implementing overconfident experts with recursive combinations.




the switch distribution or the run-length model, which are described in the next section.

## 4.4 New Models to Switch between Experts

So far we have considered two models for switching between experts: fixed share and its generalisation, universal share. While fixed share is an extremely efficient algorithm, it requires that the frequency of switching between experts is estimated a priori, which can be hard in practice. Moreover, we may have prior knowledge about how the switching probability will change over time, but unless we know the ultimate sample size in advance, we may be forced to accept a linear overhead compared to the best parameter value. Universal share overcomes this problem by marginalising over the unknown parameter, but has quadratic running time.

The first model considered in this section, called the switch distribution, avoids both problems. It is parameterless and has essentially the same running time as fixed share. It also achieves a loss bound competitive to that of universal share. Moreover, for a bounded number of switches the bound has even better asymptotics.

The second model is called the run-length model because it uses a run-length code (c.f. [62]) as an ES-prior. This may be useful because, while both fixed and universal share model the distance between switches with a geometric distribution, the real distribution on these distances may be different. This is the case if, for example, the switches are highly clustered. This additional expressive power comes at the cost of quadratic running time, but we discuss a special case where this may be reduced to linear. We compare advantages and drawbacks of the run-length model compared to the switch distribution.

### 4.4.1 Switch Distribution

The switch distribution is a new model for combining expert predictions. Like fixed share, it is intended for settings where the best predicting expert is expected to change as a function of the sample size, but it has two major innovations. First, we let the probability of switching to a different expert decrease with the

sample size. This allows us to derive a loss bound close to that of the fixed share algorithm, without the need to tune any parameters.[5] Second, the switch distribution has a special provision to ensure that in the case where the number of switches remains bounded, the incurred loss overhead is $O(1)$.

The switch distribution is the subject of the next chapter, which addresses a long standing open problem in statistical model selection known as the "AIC vs BIC dilemma". Some criteria for model selection, such as AIC, are efficient when applied to sequential prediction of future outcomes, while other criteria, such as BIC, are "consistent": with probability one, the model that contains the data generating distribution is selected given enough data. Using the switch distribution, these two goals (truth finding vs prediction) can be reconciled. Refer to the paper for more information.

Here we disregard such applications and treat the switch distribution like the other models for combining expert predictions. We describe an HMM that corresponds to the switch distribution; this illuminates the relationship between the switch distribution and the fixed share algorithm which it in fact generalises.

The equivalence between the original definition of the switch distribution and the HMM is not trivial, so we give a formal proof. The size of the HMM is such that calculation of $P(x^n)$ requires only $O(n\,|\Xi|)$ steps.

We provide a loss bound for the switch distribution in Section 4.4.1. Then in Section 4.4.1 we show how the sequence of experts that has maximum a posteriori probability can be computed. This problem is difficult for general HMMs, but the structure of the HMM for the switch distribution allows for an efficient algorithm in this case.

### Switch HMM

Let $\sigma^\infty$ and $\tau^\infty$ be sequences of distributions on $\{0,1\}$ which we call the *switch probabilities* and the *stabilisation probabilities*. The switch HMM $\mathbb{A}_{\text{sw}}$, displayed

[5] The idea of decreasing the switch probability as $1/(n+1)$, which has not previously been published, was independently conceived by Mark Herbster and the authors.

in Figure 4.12, is defined below using $Q = Q_{\mathrm{s}} \cup Q_{\mathrm{p}}$:

$$\begin{aligned} Q_{\mathrm{s}} &= \{\mathsf{p}, \mathsf{p_s}, \mathsf{p_u}\} \times \mathbb{N} \qquad \mathrm{P_o}(\mathsf{p}, 0) = 1 \quad \Lambda(\mathsf{s}, \xi, n) = \xi \\ Q_{\mathrm{p}} &= \{\mathsf{s}, \mathsf{u}\} \times \Xi \times \mathbb{Z}^+ \qquad\qquad\quad \Lambda(\mathsf{u}, \xi, n) = \xi \end{aligned} \tag{4.23a}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \mathsf{p_u}, n\rangle \\ \langle \mathsf{p}, n\rangle \to \langle \mathsf{p_s}, n\rangle \\ \langle \mathsf{p_u}, n\rangle \to \langle \mathsf{u}, \xi, n+1\rangle \\ \langle \mathsf{p_s}, n\rangle \to \langle \mathsf{s}, \xi, n+1\rangle \\ \langle \mathsf{s}, \xi, n\rangle \to \langle \mathsf{s}, \xi, n+1\rangle \\ \langle \mathsf{u}, \xi, n\rangle \to \langle \mathsf{u}, \xi, n+1\rangle \\ \langle \mathsf{u}, \xi, n\rangle \to \langle \mathsf{p}, n\rangle \end{pmatrix} = \begin{pmatrix} \tau_n(0) \\ \tau_n(1) \\ w(\xi) \\ w(\xi) \\ 1 \\ \sigma_n(0) \\ \sigma_n(1) \end{pmatrix} \tag{4.23b}$$

This HMM contains two "expert bands". Consider a productive state $\langle \mathsf{u}, \xi, n\rangle$ in the bottom band, which we call the *unstable* band, from a generative viewpoint. Two things can happen. With probability $\sigma_n(0)$ the process continues horizontally to $\langle \mathsf{u}, \xi, n+1\rangle$ and the story repeats. We say that *no switch occurs*. With probability $\sigma_n(1)$ the process continues to the silent state $\langle \mathsf{p}, n\rangle$ directly to the right. We say that *a switch occurs*. Then a new choice has to be made. With probability $\tau_n(0)$ the process continues rightward to $\langle \mathsf{p_u}, n\rangle$ and then branches out to some productive state $\langle \mathsf{u}, \xi', n+1\rangle$ (possibly $\xi = \xi'$), and the story repeats. With probability $\tau_n(1)$ the process continues to $\langle \mathsf{p_s}, n\rangle$ in the top band, called the *stable* band. Also here it branches out to some productive state $\langle \mathsf{s}, \xi', n+1\rangle$. But from this point onward there are no choices anymore; expert $\xi'$ is produced forever. We say that the process has *stabilised*.

By choosing $\tau_n(1) = 0$ and $\sigma_n(1) = \theta$ for all $n$ we essentially remove the stable band and arrive at fixed share with parameter $\theta$. The presence of the stable band enables us to improve the loss bound of fixed share in the particular case that the number of switches is bounded; in that case, the stable band allows us to remove the dependency of the loss bound on $n$ altogether. We will use the particular choice $\tau_n(0) = \theta$ for all $n$, and $\sigma_n(1) = \pi_{\mathrm{T}}(\boldsymbol{Z} = n | \boldsymbol{Z} \geq n)$ for some fixed value $\theta$ and an arbitrary distribution $\pi_{\mathrm{T}}$ on $\mathbb{N}$. This allows us to relate the switch HMM to the parametric representation that we present next.

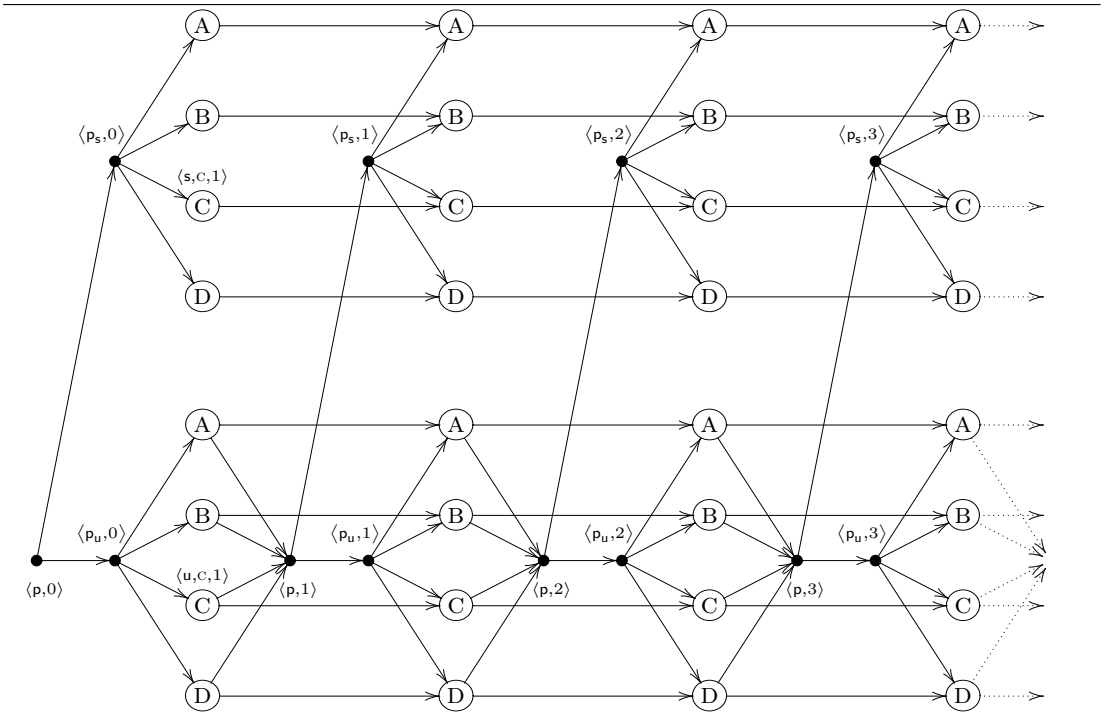
### Switch Distribution

In Chapter 5, we give a parametric definition of the switch distribution, and provide an algorithm that computes it efficiently, i.e. in time $O(n\,|\Xi|)$, where $n$ is the sample size and $|\Xi|$ is the number of considered experts. Here we show that this algorithm can really be interpreted as the forward algorithm applied to the switch HMM of Section 4.4.1.

**Definition 4.4.1.** We first define the countable set of *switch parameters*

$$\Theta_{\mathrm{sw}} := \left\{ \langle t^m, k^m \rangle \mid m \geq 1, k \in \Xi^m, t \in \mathbb{N}^m \text{ and } 0 = t_1 < t_2 < t_3 \ldots \right\}.$$

**Figure 4.12** Combination of four experts using the switch distribution




The *switch prior* is the discrete distribution on switch parameters given by

$$\pi_{\text{sw}}(t^m, k^m) := \pi_{\text{M}}(m)\pi_{\text{K}}(k_1)\prod_{i=2}^{m}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k_i),$$

where $\pi_{\text{M}}$ is geometric with rate $\theta$, $\pi_{\text{T}}$ and $\pi_{\text{K}}$ are arbitrary distributions on $\mathbb{N}$ and $\Xi$. We define the mapping $\xi : \Theta_{\text{sw}} \to \Xi^\infty$ that interprets switch parameters as sequences of experts by

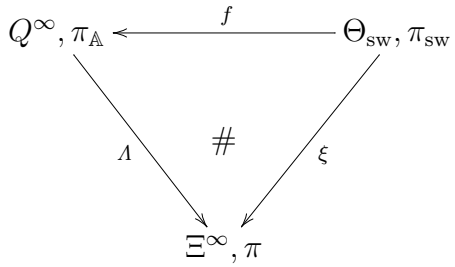
$$\xi(t^m, k^m) := k_1^{[t_2-t_1]} \frown k_2^{[t_3-t_2]} \frown \ldots \frown k_{m-1}^{[t_m-t_{m-1}]} \frown k_m^{[\infty]},$$

where $k^{[\lambda]}$ is the sequence consisting of $\lambda$ repetitions of $k$. This mapping is not 1-1: infinitely many switch parameters map to the same infinite sequence, since $k_i$ and $k_{i+1}$ may coincide. The *switch distribution* $P_{\text{sw}}$ is the ES-joint based on the ES-prior that is obtained by composing $\pi_{\text{sw}}$ with $\xi$.

### Equivalence

In this section we show that the HMM prior $\pi_{\mathbb{A}}$ and the switch prior $\pi_{\text{sw}}$ define the same ES-prior. During this section, it is convenient to regard $\pi_{\mathbb{A}}$ as a distribution on sequences of states, allowing us to differentiate between distinct sequences of states that map to the same sequence of experts. The function $\Lambda : Q^\infty \to \Xi^\infty$, that we call *trace*, explicitly performs this mapping; $\Lambda(q^\infty)(i) := \Lambda(q_i^{\text{P}})$. We cannot

**Figure 4.13** Commutativity diagram




relate $\pi_{\text{sw}}$ to $\pi_{\mathbb{A}}$ directly as they are carried by different sets (switch parameters vs state sequences), but need to consider the distribution that both induce on sequences of experts via $\xi$ and $\Lambda$. Formally:

**Definition 4.4.2.** If $f : \Theta \to \Gamma$ is a random variable and $P$ is a distribution on $\Theta$, then we write $f(P)$ to denote the distribution on $\Gamma$ that is induced by $f$.

Below we will show that $\Lambda(\pi_{\mathbb{A}}) = \xi(\pi_{\text{sw}})$, i.e. that $\pi_{\text{sw}}$ and $\pi_{\mathbb{A}}$ induce the same distribution on the expert sequences $\Xi^\infty$ via the trace $\Lambda$ and the expert-sequence mapping $\xi$. Our argument will have the structure outlined in Figure 4.13. Instead of proving the claim directly, we create a random variable $f : \Theta_{\text{sw}} \to Q^\infty$ mapping switch parameters into runs. Via $f$, we can view $\Theta_{\text{sw}}$ as a reparameterisation of $Q^\infty$. We then show that the diagram commutes, that is, $\pi_{\mathbb{A}} = f(\pi_{\text{sw}})$ and $\Lambda \circ f = \xi$. This shows that $\Lambda(\pi_{\mathbb{A}}) = \Lambda(f(\pi_{\text{sw}})) = \xi(\pi_{\text{sw}})$ as required.

**Proposition 4.4.3.** *Let $\mathbb{A}$ be the HMM as defined in Section 4.4.1, and $\pi_{\text{sw}}, \xi$ and $\Lambda$ as above. If $w = \pi_{\kappa}$ then*

$$\xi(\pi_{\text{sw}}) = \Lambda(\pi_{\mathbb{A}}).$$

*Proof.* Recall (4.23) that

$$Q = \{\mathsf{s}, \mathsf{u}\} \times \Xi \times \mathbb{Z}^+ \quad \cup \quad \{\mathsf{p}, \mathsf{p}_\mathsf{s}, \mathsf{p}_\mathsf{u}\} \times \mathbb{N}.$$

We define the random variable $f : \Theta_{\text{sw}} \to Q^\infty$ by

$$\begin{aligned} f(t^m, k^m) &:= \langle \mathsf{p}, 0 \rangle \frown u_1 \frown u_2 \frown \ldots \frown u_{m-1} \frown s, \qquad \text{where} \\ u_i &:= \left\langle \langle \mathsf{p}_\mathsf{u}, t_i \rangle, \langle \mathsf{u}, k_i, t_i + 1 \rangle, \langle \mathsf{u}, k_i, t_i + 2 \rangle, \ldots, \langle \mathsf{u}, k_i, t_{i+1} \rangle, \langle \mathsf{p}, t_{i+1} \rangle \right\rangle \\ s &:= \left\langle \langle \mathsf{p}_\mathsf{s}, t_m \rangle, \langle \mathsf{s}, k_m, t_m + 1 \rangle, \langle \mathsf{s}, k_m, t_m + 2 \rangle, \ldots \right\rangle. \end{aligned}$$

We now show that $\Lambda \circ f = \xi$ and $f(\pi_{\text{sw}}) = \pi_{\mathbb{A}}$, from which the theorem follows directly. Fix $p = \langle t^m, k^m \rangle \in \Theta_{\text{sw}}$. Since the trace of a concatenation equals the

concatenation of the traces,

$$\begin{aligned}\Lambda \circ f(p) &= \Lambda(u_1) \frown \Lambda(u_2) \frown \ldots \frown \Lambda(u_{m-1}) \frown \Lambda(s) \\ &= k_1^{[t_2-t_1]} \frown k_2^{[t_3-t_2]} \frown \ldots \frown k_2^{[t_m-t_{m-1}]} \frown k_m^{[\infty]} = \xi(p).\end{aligned}$$

which establishes the first part. Second, we need to show that $\pi_{\mathbb{A}}$ and $f(\pi_{\text{sw}})$ assign the same probability to all events. Since $\pi_{\text{sw}}$ has countable support, so has $f(\pi_{\text{sw}})$. By construction $f$ is injective, so the preimage of $f(p)$ equals $\{p\}$, and hence $f(\pi_{\text{sw}})(\{f(p)\}) = \pi_{\text{sw}}(p)$. Therefore it suffices to show that $\pi_{\mathbb{A}}(\{f(p)\}) = \pi_{\text{sw}}(p)$ for all $p \in \Theta_{\text{sw}}$. Let $q^\infty = f(p)$, and define $u_i$ and $s$ for this $p$ as above. Then

$$\pi_{\mathbb{A}}(q^\infty) = \pi_{\mathbb{A}}(\langle \mathsf{p}, 0 \rangle) \left( \prod_{i=1}^{m-1} \pi_{\mathbb{A}}(u_i | u^{i-1}) \right) \pi_{\mathbb{A}}(s | u^{m-1})$$

Note that

$$\begin{aligned}\pi_{\mathbb{A}}(s|u^{m-1}) &= (1-\theta)\pi_{\text{K}}(k_i) \\ \pi_{\mathbb{A}}(u_i|u^{i-1}) &= \theta \pi_{\text{K}}(k_i) \left( \prod_{j=t_i+1}^{t_{i+1}-1} \pi_{\text{T}}(\boldsymbol{Z} > j | \boldsymbol{Z} \geq j) \right) \pi_{\text{T}}(\boldsymbol{Z} = t_{i+1} | \boldsymbol{Z} \geq t_{i+1}).\end{aligned}$$

The product above telescopes, so that

$$\pi_{\mathbb{A}}(u_i|u^{i-1}) = \theta \pi_{\text{K}}(k_i) \pi_{\text{T}}(\boldsymbol{Z} = t_{i+1} | \boldsymbol{Z} \geq t_{i+1}).$$

We obtain

$$\begin{aligned}\pi_{\mathbb{A}}(q^\infty) &= 1 \cdot \theta^{m-1} \left( \prod_{i=1}^{m-1} \pi_{\text{K}}(k_i) \pi_{\text{T}}(t_{i+1} | t_{i+1} > t_i) \right) (1-\theta) \pi_{\text{K}}(k_m) \\ &= \theta^{m-1}(1-\theta)\pi_{\text{K}}(k_1) \prod_{i=2}^{m} \pi_{\text{K}}(k_i) \pi_{\text{T}}(t_i | t_i > t_{i-1}) \\ &= \pi_{\text{sw}}(p),\end{aligned}$$

under the assumption that $\pi_{\text{M}}$ is geometric with parameter $\theta$. □

### A Loss Bound

We derive a loss bound of the same type as the bound for the fixed share algorithm (see Section 4.3.2).

**Theorem 4.4.4.** *Fix data $x^n$. Let $\hat{\theta} = \langle t^m, k^m \rangle$ maximise the likelihood $P_{\xi(\hat{\theta})}(x^n)$ among all switch parameters of length $m$. Let $\pi_{\text{M}}(n) = 2^{-n}$, $\pi_{\text{T}}(n) = 1/(n(n+1))$*

*and* $\pi_{\text{K}}$ *be uniform. Then the loss overhead* $-\log P_{\text{sw}}(x^n) + \log P_{\xi(\hat{\theta})}(x^n)$ *of the switch distribution is bounded by*

$$m + m \log |\Xi| + \log \binom{t_m + 1}{m} + \log(m!).$$

*Proof.* We have

$$\begin{aligned} & -\log P_{\text{sw}}(x^n) + \log P_{\xi(\hat{\theta})}(x^n) \\ \leq\ & -\log \pi_{\text{sw}}(\hat{\theta}) \\ =\ & -\log \left( \pi_{\text{M}}(m) \pi_{\text{K}}(k_1) \prod_{i=2}^{m} \pi_{\text{T}}(t_i | t_i > t_{i-1}) \pi_{\text{K}}(k_i) \right) \\ =\ & -\log \pi_{\text{M}}(m) + \sum_{i=1}^{m} -\log \pi_{\text{K}}(k_i) + \sum_{i=2}^{m} -\log \pi_{\text{T}}(t_i | t_i > t_i - 1). \end{aligned} \quad (4.24)$$

The considered prior $\pi_{\text{T}}(n) = 1/(n(n+1))$ satisfies

$$\pi_{\text{T}}(t_i | t_i > t_{i-1}) = \frac{\pi_{\text{T}}(t_i)}{\sum_{i=t_{i-1}+1}^{\infty} \pi_{\text{T}}(i)} = \frac{1/(t_i(t_i+1))}{\sum_{i=t_{i-1}+1}^{\infty} \frac{1}{i} - \frac{1}{i+1}} = \frac{t_{i-1}+1}{t_i(t_i+1)}.$$

If we substitute this in the last term of (4.24), the sum telescopes and we are left with

$$\underbrace{-\log(t_1+1)}_{=\,0} + \log(t_m+1) + \sum_{i=2}^{m} \log t_i. \quad (4.25)$$

If we fix $t_m$, this expression is maximised if $t_2, \ldots, t_{m-1}$ take on the values $t_m - m + 2, \ldots, t_m - 1$, so that (4.25) becomes

$$\sum_{i=t_m-m+2}^{t_m+1} \log i = \log \left( \frac{(t_m+1)!}{(t_m-m+1)!} \right) = \log \binom{t_m+1}{m} + \log(m!).$$

The theorem follows if we also instantiate $\pi_{\text{M}}$ and $\pi_{\text{K}}$ in (4.24). □

Note that this loss bound is a function of the index of the last switch $t_m$ rather than of the sample size $n$; this means that in the important scenario where the number of switches remains bounded in $n$, the loss compared to the best partition is $O(1)$.

The bound can be tightened slightly by using the fact that we allow for switching to the same expert, as also remarked in Footnote 3 on page 96. If we take this into account, the $m \log |\Xi|$ term can be reduced to $m \log(|\Xi| - 1)$. If we take this into account, the bound compares quite favourably with the loss bound for the fixed share algorithm (see Section 4.3.2). We now investigate how much

worse the above guarantees are compared to those of fixed share. The overhead of fixed share (4.18) is bounded from above by $nH(\alpha) + m\log(|\Xi| - 1)$. We first underestimate this worst-case loss by substituting the optimal value $\alpha = m/n$, and rewrite

$$nH(\alpha) \;\geq\; nH(m/n) \;\geq\; \log \binom{n}{m}.$$

Second we overestimate the loss of the switch distribution by substituting the worst case $t_m = n - 1$. We then find the maximal difference between the two bounds to be

$$\left(m + m\log(|\Xi| - 1) + \log\binom{n}{m} + \log(m!)\right) - \left(\log\binom{n}{m} + m\log(|\Xi| - 1)\right)$$
$$= m + \log(m!) \;\leq\; m + m\log m. \quad (4.26)$$

Thus using the switch distribution instead of fixed share lowers the guarantee by at most $m + m\log m$ bits, which is significant only if the number of switches is relatively large. On the flip side, using the switch distribution does not require any prior knowledge about any parameters. This is a big advantage in a setting where we desire to maintain the bound sequentially. This is impossible with the fixed share algorithm in case the optimal value of $\alpha$ varies with $n$.

### MAP Estimation

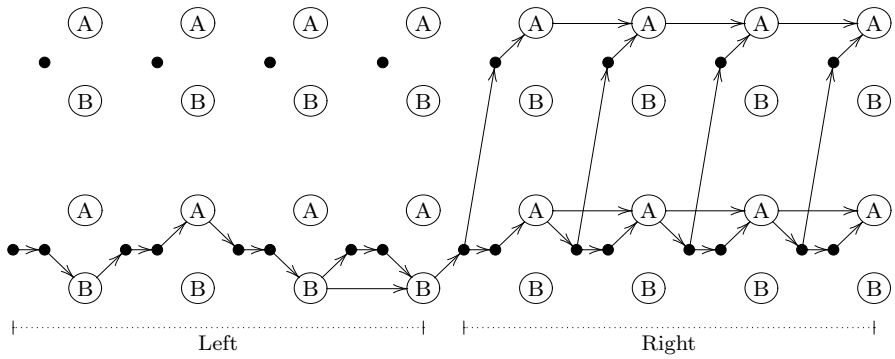
The particular nature of the switch distribution allows us to perform MAP estimation efficiently. The MAP sequence of experts is:

$$\arg\max_{\xi^n} P(x^n, \xi^n).$$

We observed in Section 4.2.5 that Viterbi can be used on unambiguous HMMs. However, the switch HMM is ambiguous, since a single sequence of experts is produced by multiple sequences of states. Still, it turns out that for the switch HMM we can jointly consider all these sequences of states efficiently. Consider for example the expert sequence ABAABBBB. The sequences of states that produce this expert sequence are exactly the runs through the pruned HMM shown in Figure 4.14. Runs through this HMM can be decomposed in two parts, as indicated in the bottom of the figure. In the right part a single expert is repeated, in our case expert D. The left part is contained in the unstable (lower) band. To compute the MAP sequence we proceed as follows. We iterate over the possible places of the transition from left to right, and then optimise the left and right segments independently.

In the remainder we first compute the probability of the MAP expert sequence instead of the sequence itself. We then show how to compute the MAP sequence from the fallout of the probability computation.

**Figure 4.14** MAP estimation for the switch distribution. The sequences of states that can be obtained by following the arrows are exactly those that produce expert sequence ABAABBBB.




To optimise both parts, we define two functions $L$ and $R$.

$$L_i := \max_{\xi^i} P(x^i, \xi^i, \langle \mathsf{p}, i \rangle) \tag{4.27}$$

$$R_i(\xi) := P(x^n, \xi_i = \ldots = \xi_n = \xi | x^{i-1}, \langle \mathsf{p}, i-1 \rangle) \tag{4.28}$$

Thus $L_i$ is the probability of the MAP expert sequence of length $i$. The requirement $\langle \mathsf{p}, i \rangle$ forces all sequences of states that realise it to remain in the unstable band. $R_i(\xi)$ is the probability of the tail $x_i, \ldots, x_n$ when expert $\xi$ is used for all outcomes, starting in state $\langle \mathsf{p}, i-1 \rangle$. Combining $L$ and $R$, we have

$$\max_{\xi^n} P(x^n, \xi^n) = \max_{i \in [n], \xi} L_{i-1} R_i(\xi).$$

**Recurrence** $L_i$ and $R_i$ can efficiently be computed using the following recurrence relations. First we define auxiliary quantities

$$L'_i(\xi) := \max_{\xi^i} P(x^i, \xi^i, \langle \mathsf{u}, \xi, i \rangle) \tag{4.29}$$

$$R'_i(\xi) := P(x^n, \xi_i = \ldots = \xi_n = \xi | x^{i-1}, \langle \mathsf{u}, \xi, i \rangle) \tag{4.30}$$

Observe that the requirement $\langle \mathsf{u}, \xi, i \rangle$ forces $\xi_i = \xi$. First, $L'_i(\xi)$ is the MAP probability for length $i$ under the constraint that the last expert used is $\xi$. Second, $R'_i(\xi)$ is the MAP probability of the tail $x_i, \ldots, x_n$ under the constraint that the same expert is used all the time. Using these quantities, we have (using the $\gamma_{(\cdot)}$ transition probabilities shown in (4.34))

$$L_i = \max_{\xi} L'_i(\xi) \gamma_1 \qquad R_i(\xi) = \gamma_2 R'_i(\xi) + \gamma_3 P_\xi(x^n | x^{i-1}). \tag{4.31}$$

For $L'_i(\xi)$ and $R'_i(\xi)$ we have the following recurrences:

$$L_{i+1}(\xi) = P_\xi(x_{i+1}|x^i) \max\left\{L'_i(\xi)(\gamma_4 + \gamma_1\gamma_5), L_i\gamma_5\right\} \tag{4.32}$$
$$R'_i(\xi) = P_\xi(x_i|x^{i-1})\left(\gamma_1 R_{i+1}(\xi) + \gamma_4 R'_{i+1}(\xi)\right). \tag{4.33}$$

The recurrence for $L$ has border case $L_0 = 1$. The recurrence for $R$ has border case $R_n = 1$.

$$\begin{aligned}
\gamma_1 &= \mathrm{P}\left(\langle \mathsf{u}, \xi, i\rangle \to \langle \mathsf{p}, i\rangle\right)\\
\gamma_2 &= \mathrm{P}\left(\langle \mathsf{p}, i-1\rangle \to \langle \mathsf{p}_\mathsf{u}, i-1\rangle \to \langle \mathsf{u}, \xi, i\rangle\right)\\
\gamma_3 &= \mathrm{P}\left(\langle \mathsf{p}, i-1\rangle \to \langle \mathsf{p}_\mathsf{s}, i-1\rangle \to \langle \mathsf{s}, \xi, i\rangle\right)\\
\gamma_4 &= \mathrm{P}\left(\langle \mathsf{u}, \xi, i\rangle \to \langle \mathsf{u}, \xi, i+1\rangle\right)\\
\gamma_5 &= \mathrm{P}\left(\langle \mathsf{p}, i\rangle \to \langle \mathsf{p}_\mathsf{u}, i\rangle \to \langle \mathsf{u}, \xi, i+1\rangle\right)
\end{aligned} \tag{4.34}$$

**Complexity** A single recurrence step of $L_i$ costs $O(|\Xi|)$ due to the maximisation. All other recurrence steps take $O(1)$. Hence both $L_i$ and $L'_i(\xi)$ can be computed recursively for all $i = 1, \ldots, n$ and $\xi \in \Xi$ in time $O(n\,|\Xi|)$, while each of $R_i, R'_i(\xi)$ and $P_\xi(x^n|x^{i-1})$ can be computed recursively for all $i = n, \ldots, 1$ and $\xi \in \Xi$ in time $O(n\,|\Xi|)$ as well. Thus the MAP probability can be computed in time $O(n\,|\Xi|)$. Storing all intermediate values costs $O(n\,|\Xi|)$ space as well.

**The MAP Expert Sequence** As usual in Dynamic Programming, we can retrieve the final solution — the MAP expert sequence — from these intermediate values. We redo the computation, and each time that a maximum is computed we record the expert that achieves it. The experts thus computed form the MAP sequence.

### 4.4.2 Run-length Model

Run-length codes have been used extensively in the context of data compression, see e.g. [62]. Rather than applying run length codes directly to the observations, we reinterpret the corresponding probability distributions as ES-priors, because they may constitute good models for the distances between consecutive switches.

The run length model is especially useful if the switches are clustered, in the sense that some blocks in the expert sequence contain relatively few switches, while other blocks contain many. The fixed share algorithm remains oblivious to such properties, as its predictions of the expert sequence are based on a Bernoulli model: the probability of switching remains the same, regardless of the index of the previous switch. Essentially the same limitation also applies to the universal share algorithm, whose switching probability normally converges as the sample size increases. The switch distribution is efficient when the switches are clustered toward the beginning of the sample: its switching probability decreases

in the sample size. However, this may be unrealistic and may introduce a new unnecessary loss overhead.

The run-length model is based on the assumption that the *intervals* between successive switches are independently distributed according to some distribution $\pi_{\text{T}}$. After the universal share model and the switch distribution, this is a third generalisation of the fixed share algorithm, which is recovered by taking a geometric distribution for $\pi_{\text{T}}$. As may be deduced from the defining HMM, which is given below, we require quadratic running time $O(n^2 |\Xi|)$ to evaluate the run-length model in general.

### Run-length HMM

Let $\mathbb{S} := \big\{\langle m, n\rangle \in \mathbb{N}^2 \mid m < n\big\}$, and let $\pi_{\text{T}}$ be a distribution on $\mathbb{Z}^+$. The specification of the run-length HMM is given using $Q = Q_{\text{s}} \cup Q_{\text{p}}$ by:

$$\begin{aligned} Q_{\text{s}} &= \{\mathsf{q}\} \times \mathbb{S} \;\cup\; \{\mathsf{p}\} \times \mathbb{N} & \Lambda(\xi, m, n) &= \xi \\ Q_{\text{p}} &= \Xi \times \mathbb{S} & \mathrm{P}_\circ(\mathsf{p}, 0) &= 1 \end{aligned} \tag{4.35a}$$

$$\mathrm{P}\begin{pmatrix} \langle \mathsf{p}, n\rangle \to \langle \xi, n, n+1\rangle \\ \langle \xi, m, n\rangle \to \langle \xi, m, n+1\rangle \\ \langle \xi, m, n\rangle \to \langle \mathsf{q}, m, n\rangle \\ \langle \mathsf{q}, m, n\rangle \to \langle \mathsf{p}, n\rangle \end{pmatrix} = \begin{pmatrix} w(\xi) \\ \pi_{\text{T}}(\boldsymbol{Z} > n | \boldsymbol{Z} \geq n) \\ \pi_{\text{T}}(\boldsymbol{Z} = n | \boldsymbol{Z} \geq n) \\ 1 \end{pmatrix} \tag{4.35b}$$

### A Loss Bound

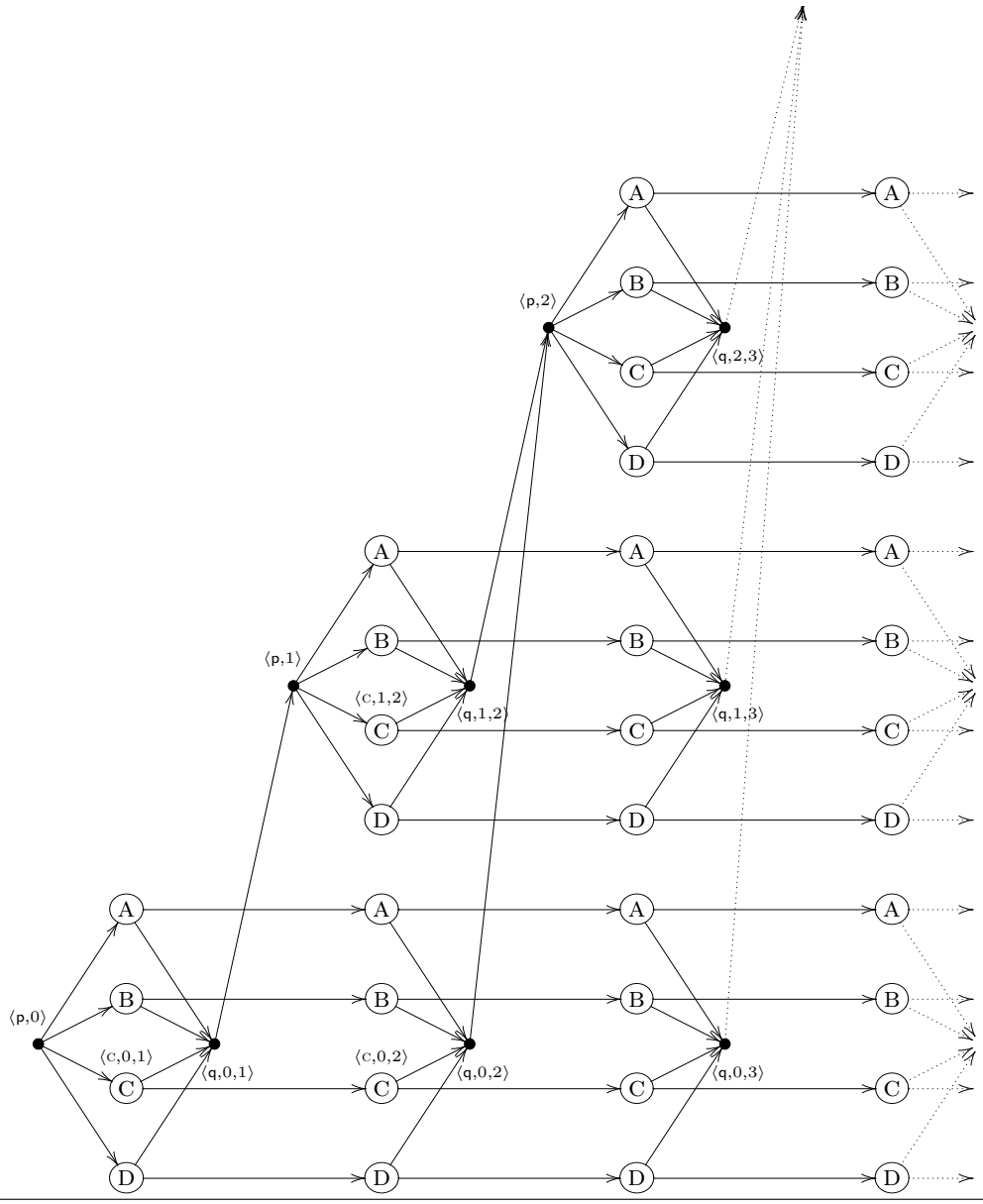
Fix an expert sequence $\xi^n$ with $m$ blocks. For $i = 1, \ldots, m$, let $\delta_i$ and $k_i$ denote the length and expert of block $i$. From the definition of the HMM above, we obtain that $\pi_{\text{rl}}(\xi^n)$ equals

$$\sum_{i=1}^{m} -\log w(k_i) + \sum_{i=1}^{m-1} -\log \pi_{\text{T}}(\boldsymbol{Z} = \delta_i) - \log \pi_{\text{T}}(\boldsymbol{Z} \geq \delta_m).$$

**Theorem 4.4.5.** *Fix data $x^n$. Let $\xi^n$ maximise the likelihood $P_{\xi^n}(x^n)$ among all expert sequences with $m$ blocks. Let $w$ be the uniform distribution on experts, and let $\pi_{\text{T}}$ be log-convex. Then the loss overhead is bounded thus*

$$-\log P_{\text{rl}}(x^n) + \log P_{\xi^n}(x^n) \;\leq\; m\left(\log |\Xi| - \log \pi_{\text{T}}\left(\tfrac{n}{m}\right)\right).$$

**Figure 4.15** HMM for the run-length model



*Proof.* Let $\delta_i$ denote the length of block $i$. We overestimate

$$\begin{aligned} &-\log P_{\text{rl}}(x^n) + \log P_{\xi^n}(x^n) \;\leq\; -\log \pi_{\text{rl}}(\xi^n) \\ &= m \log |\Xi| + \sum_{i=1}^{m-1} -\log \pi_{\text{T}}(\boldsymbol{Z} = \delta_i) - \log \pi_{\text{T}}(\boldsymbol{Z} \geq \delta_m) \\ &\leq m \log |\Xi| + \sum_{i=1}^{m} -\log \pi_{\text{T}}(\delta_i). \end{aligned} \tag{4.36}$$

Since $-\log \pi_{\text{T}}$ is concave, by Jensen's inequality we have

$$\sum_{i=1}^{m} \frac{-\log \pi_{\text{T}}(\delta_i)}{m} \leq -\log \pi_{\text{T}} \left( \sum_{i=1}^{m} \frac{\delta_i}{m} \right) = -\log \pi_{\text{T}} \left( \frac{n}{m} \right).$$

In other words, the block lengths $\delta_i$ are all equal in the worst case. Plugging this into (4.36) we obtain the theorem. □

**Finite Support**

We have seen that the run-length model reduces to fixed share if the prior on switch distances $\pi_{\text{T}}$ is geometric, so that it can be evaluated in linear time in that case. We also obtain a linear time algorithm when $\pi_{\text{T}}$ has finite support, because then only a constant number of states can receive positive weight at any sample size. For this reason it can be advantageous to choose a $\pi_{\text{T}}$ with finite support, even if one expects that arbitrarily long distances between consecutive switches may occur. Expert sequences with such longer distances between switches can still be represented with a truncated $\pi_{\text{T}}$ using a sequence of switches from and to the same expert. This way, long runs of the same expert receive exponentially small, but positive, probability.

### 4.4.3 Comparison

We have discussed two models for switching: the recent switch distribution and the new run-length model. It is natural to wonder which model to apply. One possibility is to compare asymptotic loss bounds. To compare the bounds given by Theorems 4.4.4 and 4.4.5, we substitute $t_m + 1 = n$ in the bound for the switch distribution, and use a prior $\pi_{\text{T}}$ for the run-length model that satisfies $-\log \pi_{\text{T}}(n) \leq \log n + 2 \log \log(n+1) + 3$ (for instance an Elias code [32]). The next step is to determine which bound is better depending on how fast $m$ grows as a function of $n$. It only makes sense to consider $m$ non-decreasing in $n$.

**Theorem 4.4.6.** *The loss bound of the switch distribution (with $t_n = n$) is asymptotically lower than that of the run-length model (with $\pi_{\text{T}}$ as above) if*

$m = o\big((\log n)^2\big)$, *and asymptotically higher if* $m = \Omega\big((\log n)^2\big)$.[6]

*Proof sketch.* After eliminating terms common to both loss bounds, it remains to compare

$$m + m \log m \quad \text{to} \quad 2m \log \log \left( \frac{n}{m} + 1 \right) + 3.$$

If $m$ is bounded, the left hand side is clearly lower for sufficiently large $n$. Otherwise we may divide by $m$, exponentiate, simplify, and compare

$$m \quad \text{to} \quad (\log n - \log m)^2,$$

from which the theorem follows directly. □

For finite samples, the switch distribution can be used in case the switches are expected to occur early on average, or if the running time is paramount. Otherwise the run-length model is preferable.

## 4.5 Extensions

The approach described in Sections 4.1 and 4.2 allows efficient evaluation of expert models that can be defined using small HMMs. It is natural to look for additional efficient models for combining experts that cannot be expressed as small HMMs in this way.

In this section we describe a number of such extensions to the model as described above. In Section 4.5.1 we outline different methods for approximate, but faster, evaluation of large HMMs. The idea behind Section 4.3.4 is to treat a *combination* of experts as a single expert, and subject it to "meta" expert combination. Then in Section 4.5.2 we outline a possible generalisation of the considered class of HMMs, allowing the ES-prior to depend on observed data. Finally we propose an alternative to MAP expert sequence estimation that is efficiently computable for general HMMs.

### 4.5.1 Fast Approximations

For some applications, suitable ES-priors do not admit a description in the form of a small HMM. Under such circumstances we might require an exponential amount of time to compute quantities such as the predictive distribution on the next expert (4.3). For example, although the size of the HMM required to describe the elementwise mixtures of Section 4.3.1 grows only polynomially in $n$, this is still not feasible in practice. Consider that the transition probabilities at sample size $n$ must depend on the number of times that each expert has occurred previously.

[6] Let $f, g : \mathbb{N} \to \mathbb{N}$. We say $f = o(g)$ if $\lim_{n\to\infty} f(n)/g(n) = 0$. We say $f = \Omega(g)$ if $\exists c > 0 \exists n_0 \forall n \geq n_0 : f(n) \geq cg(n)$.

The number of states required to represent this information must therefore be at least $\binom{n+k-1}{k-1}$, where $k$ is the number of experts. For five experts and $n = 100$, we already require more than four million states! In the special case of mixtures, various methods exist to efficiently find good parameter values, such as expectation maximisation, see e.g. [59] and Li and Barron's approach [55]. Here we describe a few general methods to speed up expert sequence calculations.

### Discretisation

The simplest way to reduce the running time of Algorithm 4.1 is to reduce the number of states of the input HMM, either by simply omitting states or by identifying states with similar futures. This is especially useful for HMMs where the number of states grows in $n$, e.g. the HMMs where the parameter of a Bernoulli source is learned: the HMM for universal elementwise mixtures of Figure 4.7 and the HMM for universal share of Figure 4.9. At each sample size $n$, these HMMs contain states for count vectors $(0, n), (1, n-1), \ldots, (n, 0)$. In [63] Monteleoni and Jaakkola manage to reduce the number of states to $\sqrt{n}$ when the sample size $n$ is known in advance. We conjecture that it is possible to achieve the same loss bound by joining ranges of well-chosen states into roughly $\sqrt{n}$ super-states, and adapting the transition probabilities accordingly.

### Trimming

Another straightforward way to reduce the running time of Algorithm 4.1 is by run-time modification of the HMM. We call this *trimming*. The idea is to drop low probability transitions from one sample size to the next. For example, consider the HMM for elementwise mixtures of two experts, Figure 4.7. The number of transitions grows linearly in $n$, but depending on the details of the application, the probability mass may concentrate on a subset that represents mixture coefficients close to the optimal value. A speedup can then be achieved by always retaining only the smallest set of transitions that are reached with probability $p$, for some value of $p$ which is reasonably close to one. The lost probability mass can be recovered by renormalisation.

### The ML Conditioning Trick

A more drastic approach to reducing the running time can be applied whenever the ES-prior assigns positive probability to all expert sequences. Consider the desired marginal probability (4.2) which is equal to:

$$P(x^n) = \sum_{\xi^n \in \Xi^n} \pi(\xi^n) P(x^n \mid \xi^n). \tag{4.37}$$

In this expression, the sequence of experts $\xi^n$ can be interpreted as a parameter. While we would ideally compute the Bayes marginal distribution, which means

integrating out the parameter under the ES-prior, it may be easier to compute a point estimator for $\xi^n$ instead. Such an estimator $\xi(x^n)$ can then be used to find a lower bound on the marginal probability:

$$\pi(\xi(x^n))P(x^n \mid \xi(x^n)) \quad \leq \quad P(x^n). \tag{4.38}$$

The first estimator that suggests itself is the Bayesian maximum a-posteriori:

$$\xi_{\text{map}}(x^n) := \arg\max_{\xi^n \in \Xi^n} \pi(\xi^n) P(x^n \mid \xi^n).$$

In Section 4.2.5 we explain that this estimator is generally hard to compute for ambiguous HMMs, and for unambiguous HMMs it is as hard as evaluating the marginal (4.37). One estimator that is much easier to compute is the maximum likelihood (ML) estimator, which disregards the ES-prior $\pi$ altogether:

$$\xi_{\text{ml}}(x^n) := \arg\max_{\xi^n \in \Xi^n} P(x^n \mid \xi^n).$$

The ML estimator may correspond to a much smaller term in (4.37) than the MAP estimator, but it has the advantage that it is extremely easy to compute. In fact, letting $\hat{\xi}^n := \xi_{\text{ml}}(x^n)$, each expert $\hat{\xi}_i$ is a function of only the corresponding outcome $x_i$. Thus, calculation of the ML estimator is cheap. Furthermore, if the goal is not to find a lower bound, but to predict the outcomes $x^n$ with as much confidence as possible, we can make an even better use of the estimator if we use it sequentially. Provided that $P(x^n) > 0$, we can approximate:

$$\begin{aligned} P(x^n) = \prod_{i=1}^{n} P(x_i|x^{i-1}) &= \prod_{i=1}^{n} \sum_{\xi_i \in \Xi} P(\xi_i|x^{i-1}) P_{\xi_i}(x_i|x^{i-1}) \\ &\approx \prod_{i=1}^{n} \sum_{\xi_i \in \Xi} \pi(\xi_i|\hat{\xi}^{i-1}) P_{\xi_i}(x_i|x^{i-1}) =: \tilde{P}(x^n). \end{aligned} \tag{4.39}$$

This approximation improves the running time if the conditional distribution $\pi(\xi_n|\xi^{n-1})$ can be computed more efficiently than $P(\xi_n|x^{n-1})$, as is often the case.

**Example 13.** As can be seen in Figure 4.1, the running time of the universal elementwise mixture model (cf. Section 4.3.1) is $O(n^{|\Xi|})$, which is prohibitive in practice, even for small $\Xi$. We apply the above approximation. For simplicity we impose the uniform prior density $w(\alpha) = 1$ on the mixture coefficients. We use the generalisation of Laplace's Rule of Succession to multiple experts, which states:

$$\pi_{\text{ue}}(\xi_{n+1}|\xi^n) \;=\; \int_{\triangle(\Xi)} \alpha(\xi_{n+1}) w(\alpha|\xi^n)\, \mathrm{d}\alpha \;=\; \frac{\left|\left\{j \leq n \mid \xi_j = \xi_{n+1}\right\}\right| + 1}{n + |\Xi|}. \tag{4.40}$$

Substitution in (4.39) yields the following predictive distribution:

$$\begin{aligned}\tilde{P}(x_{n+1}|x^n) &= \sum_{\xi_{n+1}\in\Xi} \pi(\xi_{n+1} \mid \hat{\xi}^n) P_{\xi_{n+1}}(x_{n+1} \mid x^n) \\ &= \sum_{\xi_{n+1}} \frac{|\{j \le n \mid \hat{\xi}_j(x^n) = \xi_{n+1}\}| + 1}{n + |\Xi|} P_{\xi_{n+1}}(x_{n+1}|x^n).\end{aligned} \tag{4.41}$$

By keeping track of the number of occurrences of each expert in the ML sequence, this expression can easily be evaluated in time proportional to the number of experts, so that $\tilde{P}(x^n)$ can be computed in the ideal time $O(n\,|\Xi|)$ (which is a lower bound because one has to consider all experts at all sample sizes). ♢

The difference between $P(x^n)$ and $\tilde{P}(x^n)$ is difficult to analyse in general, but the approximation does have two encouraging properties. First, the lower bound (4.38) on the marginal probability, instantiated for the ML estimator, also provides a lower bound on $\tilde{P}$. We have

$$\tilde{P}(x^n) \quad \geq \quad \prod_{i=1}^{n} \pi(\hat{\xi}_i \mid \hat{\xi}^{i-1}) P_{\hat{\xi}_i}(x_i \mid x^{i-1}) \;=\; \pi(\hat{\xi}^n) P(x^n \mid \hat{\xi}^n).$$

To see why the approximation gives higher probability than the bound, consider that the bound corresponds to a defective distribution, unlike $\tilde{P}$.

Second, the following information processing argument shows that even in circumstances where the approximation of the posterior $\tilde{P}(\xi_i \mid x^{i-1})$ is poor, the approximation of the predictive distribution $\tilde{P}(x_i \mid x^{i-1})$ might be acceptable.

**Lemma 4.5.1.** *Let $\pi, \rho$ be ES-priors. Then for all $n \in \mathbb{N}$,*

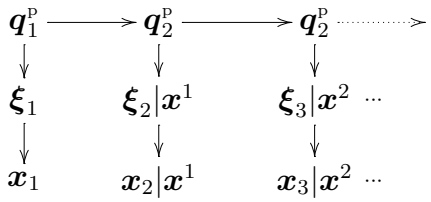
$$D\left(P_\rho(\boldsymbol{x}^n) \| P_\pi(\boldsymbol{x}^n)\right) \;\leq\; D\left(\rho(\boldsymbol{\xi}^n) \| \pi(\boldsymbol{\xi}^n)\right).$$

*Proof.* The claim follows from taking an expectation of Theorem 4.3.1 under $P_\rho$:

$$E_{P_\rho}\left[-\log \frac{P_\pi(\boldsymbol{x}^n)}{P_\rho(\boldsymbol{x}^n)}\right] \;\leq\; E_{P_\rho} E_{P_\rho}\left[-\log \frac{\pi(\boldsymbol{\xi}^n)}{\rho(\boldsymbol{\xi}^n)}\middle| \boldsymbol{x}^n\right] \;=\; E_{P_\rho}\left[-\log \frac{\pi(\boldsymbol{\xi}^n)}{\rho(\boldsymbol{\xi}^n)}\right]. \quad \square$$

We apply this lemma to the predictive distribution on the single next outcome after observing a sequence $x^n$. Setting $\pi$ to $P_\pi(\boldsymbol{\xi}_{n+1}|\xi(x^n))$ and $\rho$ to $P_\pi(\boldsymbol{\xi}_{n+1}|x^n)$, we find that the divergence between the predictive distribution on the next *outcome* and its approximation, is at most equal to the divergence between the posterior distribution on the next *expert* and its approximation. In other words, approximation errors in the posterior tend to cancel each other out during prediction.

**Figure 4.16** Conditioning ES-prior on past observations for free




### 4.5.2 Data-Dependent Priors

To motivate ES-priors we used the slogan *we do not understand the data*. When we discussed using HMMs as ES-priors we imposed the restriction that for each state the associated $\Xi$-PFS was independent of the previously produced experts. Indeed, conditioning on the *expert history* increases the running time dramatically as all possible histories must be considered. However, conditioning on the *past observations* can be done *at no additional cost*, as the data are *observed*. The resulting HMM is shown in Figure 4.16. We consider this technical possibility a curiosity, as it clearly violates our slogan. Of course it is equally feasible to condition on some function of the data. An interesting case is obtained by conditioning on the vector of losses (cumulative or incremental) incurred by the experts. This way we maintain ignorance about the data, while extending expressive power: the resulting ES-joints are generally not decomposable into an ES-prior and expert PFSs. An example is the Variable Share algorithm introduced in [46].

### 4.5.3 An Alternative to MAP Data Analysis

Sometimes we have data $x^n$ that we want to analyse. One way to do this is by computing the MAP sequence of experts. Unfortunately, we do not know how to compute the MAP sequence for general HMMs. We propose the following alternative way to gain in sight into the data. The forward and backward algorithm compute $P(x^i, q_i^{\mathrm{p}})$ and $P(x^n|q_i^{\mathrm{p}}, x^i)$. Recall that $q_i^{\mathrm{p}}$ is the productive state that is used at time $i$. From these we can compute the a-posteriori probability $P(q_i^{\mathrm{p}}|x^n)$ of each productive state $q_i^{\mathrm{p}}$. That is, the posterior probability taking the entire future into account. This is a standard way to analyse data in the HMM literature. [66] To arrive at a conclusion about experts, we simply project the posterior on states down to obtain the posterior probability $P(\xi_i|x^n)$ of each expert $\xi \in \Xi$ at each time $i = 1, \ldots, n$. This gives us a sequence of mixture weights over the experts that we can, for example, plot as a $\Xi \times n$ grid of gray shades. On the one hand this gives us mixtures, a richer representation than just single experts. On the other hand we lose temporal correlations, as we treat each time instance separately.

## 4.6 Conclusion

In prediction with expert advice, the goal is to formulate prediction strategies that perform as well as the best possible expert (combination). Expert predictions can be combined by taking a weighted mixture at every sample size. The best combination generally evolves over time. In this chapter we introduced expert sequence priors (ES-priors), which are probability distributions over infinite sequences of experts, to model the trajectory followed by the best expert combination. Prediction with expert advice then amounts to marginalising the joint distribution constructed from the chosen ES-prior and the experts' predictions.

We employed hidden Markov models (HMMs) to specify ES-priors. HMMs' explicit notion of current state and state-to-state evolution naturally fit the temporal correlations we seek to model. For reasons of efficiency we use HMMs with silent states. The standard algorithms for HMMs (Forward, Backward, Viterbi and Baum-Welch) can be used to answer questions about the ES-prior as well as the induced distribution on data. The running time of the forward algorithm can be read off directly from the graphical representation of the HMM.

Our approach allows unification of many existing expert models, including mixture models and fixed share. We gave their defining HMMs and recovered the best known running times. We also introduced two new parameterless generalisations of fixed share. The first, called the switch distribution, was recently introduced to improve model selection performance. We rendered its parametric definition as a small HMM, which shows how it can be evaluated in linear time. The second, called the run-length model, uses a run-length code in a novel way, namely as an ES-prior. This model has quadratic running time. We compared the loss bounds of the two models asymptotically, and showed that the run-length model is preferred if the number of switches grows like $(\log n)^2$ or faster, while the switch distribution is preferred if it grows slower. We provided graphical representations and loss bounds for all considered models.

Finally we described a number of extensions of the ES-prior/HMM approach, including approximating methods for large HMMs, as well as recursive combinations of expert models.

**Algorithm 4.1** Forward($\mathbb{A}$). Fix an unfolded deterministic HMM prior $\mathbb{A} = \langle Q, Q_{\mathrm{p}}, \mathrm{P}_\circ, \mathrm{P}, \Lambda \rangle$ on $\Xi$, and an $\mathcal{X}$-PFS $P_\xi$ for each expert $\xi \in \Xi$. The input consists of a sequence $x^\omega$ that arrives sequentially.

Declare the weight map (partial function) $w : Q \to [0,1]$.
$w(v) \leftarrow \mathrm{P}_\circ(v)$ **for all** $v$ s.t. $\mathrm{P}_\circ(v) > 0$.
$\triangleright$ $\mathrm{dom}(w) = I$
**for** $n = 1, 2, \ldots$ **do**
  Forward Propagation($n$)
  Predict next expert: $P(\boldsymbol{\xi}_n = \xi | x^{n-1}) = \dfrac{\sum_{v \in Q_{\{n\}} : \Lambda(v) = \xi} w(v)}{\sum_{v \in Q_{\{n\}}} w(v)}$.
  Loss Update($n$)
  Report probability of data: $P(x^n) = \sum_{v \in Q_{\{n\}}} w(v)$.
**end for**

Procedure Forward Propagation($n$):
**while** $\mathrm{dom}(w) \neq Q_{\{n\}}$ **do**
  $\triangleright$ $\mathrm{dom}(w) \subseteq Q_{[n-1,n]}$
  Pick a $<$-minimal state $u$ in $\mathrm{dom}(w) \setminus Q_{\{n\}}$.
  $\triangleright$ $u \in Q_{[n-1,n)}$
  **for** $v \in S_u$ **do**
    $\triangleright$ $v \in Q_{(n-1,n]}$
    $w(v) \leftarrow 0$ **if** $v \notin \mathrm{dom}(w)$.
    $w(v) \leftarrow w(v) + w(u)\,\mathrm{P}(u \to v)$.
  **end for**
  Remove $u$ from the domain of $w$.
**end for**

Procedure Loss Update($n$):
**for** $v \in Q_{\{n\}}$ **do**
  $\triangleright$ $v \in Q_{\mathrm{p}}$
  $w(v) \leftarrow w(v) P_{\Lambda(v)}(x_n | x^{n-1})$.
**end for**

# Chapter 5

# Slow Convergence: the Catch-up Phenomenon

We consider inference based on a countable set of models (sets of probability distributions), focusing on two tasks: model selection and model averaging. In model selection tasks, the goal is to select the model that best explains the given data. In model averaging, the goal is to find the weighted combination of models that leads to the best prediction of future data from the same source.

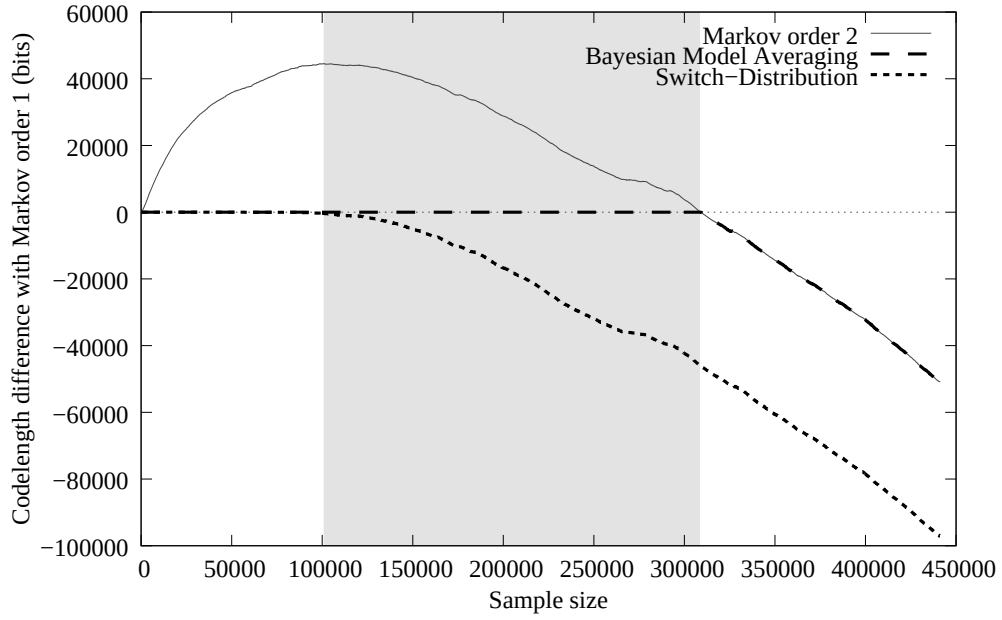
An attractive property of some criteria for model selection is that they are consistent under weak conditions, i.e. if the true distribution $P^*$ is in one of the models, then the $P^*$-probability that this model is selected goes to one as the sample size increases. BIC [78], Bayes factor model selection [49], Minimum Description Length (MDL) model selection [4] and prequential model validation [27] are examples of widely used model selection criteria that are usually consistent. However, other model selection criteria such as AIC [2] and leave-one-out cross-validation (LOO) [86], while often inconsistent, do typically yield better predictions. This is especially the case in nonparametric settings of the following type: $P^*$ can be arbitrarily well-approximated by a sequence of distributions in the (parametric) models under consideration, but is not itself contained in any of these. In many such cases, the predictive distribution converges to the true distribution at the optimal rate for AIC and LOO [80, 56], whereas in general BIC, the Bayes factor method and prequential validation only achieve the optimal rate to within an $O(\log n)$ factor [74, 34, 101, 39]. In this paper we reconcile these seemingly conflicting approaches [103] by improving the rate of convergence achieved in Bayesian model selection without losing its consistency properties. First we provide an example to show why Bayes sometimes converges too slowly.

Given priors on models $\mathcal{M}_1$, $\mathcal{M}_2$, ... and parameters therein, Bayesian inference associates each model $\mathcal{M}_k$ with the marginal distribution $p_k$, given in (5.1), obtained by averaging over the parameters according to the prior. In Bayes factor model selection the preferred model is the one with maximum a posteriori probability. By Bayes' rule this is $\arg\max_k p_k(x^n)w(k)$, where $w(k)$

denotes the prior probability of $\mathcal{M}_k$. We can further average over model indices, a process called Bayesian Model Averaging (BMA). The resulting distribution $p_{\text{bma}}(x^n) = \sum_k p_k(x^n)w(k)$ can be used for prediction. In a sequential setting, the probability of a data sequence $x^n := x_1, \ldots, x_n$ under a distribution $p$ typically decreases exponentially fast in $n$. It is therefore common to consider $-\log p(x^n)$, which we call the *code length* of $x^n$ achieved by $p$. We take all logarithms to base 2, allowing us to measure code length in *bits*. The name code length refers to the correspondence between code length functions and probability distributions based on the Kraft inequality, but one may also think of the code length as the accumulated log loss that is incurred if we sequentially predict the $x_i$ by conditioning on the past, i.e. using $p(\cdot|x^{i-1})$ [4, 39, 27, 69]. For BMA, we have $-\log p_{\text{bma}}(x^n) = \sum_{i=1}^n -\log p_{\text{bma}}(x_i|x^{i-1})$. Here the $i$th term represents the loss incurred when predicting $x_i$ given $x^{i-1}$ using $p_{\text{bma}}(\cdot|x^{i-1})$, which turns out to be equal to the posterior average: $p_{\text{bma}}(x_i|x^{i-1}) = \sum_k p_k(x_i|x^{i-1})w(k|x^{i-1})$.

Prediction using $p_{\text{bma}}$ has the advantage that the code length it achieves on $x^n$ is close to the code length of $p_{\hat{k}}$, where $\hat{k}$ is the best of the marginals $p_1, p_2, \ldots$, i.e. $\hat{k}$ achieves $\min_k -\log p_k(x^n)$. More precisely, given a prior $w$ on model indices, the difference between $-\log p_{\text{bma}}(x^n) = -\log(\sum_k p_k(x^n)w(k))$ and $-\log p_{\hat{k}}(x^n)$ must be in the range $[0, -\log w(\hat{k})]$, whatever data $x^n$ are observed. Thus, using BMA for prediction is sensible if we are satisfied with doing essentially as well as the best model under consideration. However, it is often possible to combine $p_1, p_2, \ldots$ into a distribution that achieves smaller code length than $p_{\hat{k}}$! This is possible if the index $\hat{k}$ of the best distribution *changes with the sample size in a predictable way*. This is common in model selection, for example with nested models, say $\mathcal{M}_1 \subset \mathcal{M}_2$. In this case $p_1$ typically predicts better at small sample sizes (roughly, because $\mathcal{M}_2$ has more parameters that need to be learned than $\mathcal{M}_1$), while $p_2$ predicts better eventually. Figure 5.1 illustrates this phenomenon. It shows the accumulated code length difference $-\log p_2(x^n) - (-\log p_1(x^n))$ on "The Picture of Dorian Gray" by Oscar Wilde, where $p_1$ and $p_2$ are the Bayesian marginal distributions for the first-order and second-order Markov chains, respectively, and each character in the book is an outcome. Note that the example models $\mathcal{M}_1$ and $\mathcal{M}_2$ are very crude; for this particular application much better models are available. However, in more complicated, more realistic model selection scenarios, the models may still be wrong, but it may not be known how to improve them. Thus $\mathcal{M}_1$ and $\mathcal{M}_2$ serve as a simple illustration only (see the discussion in Section 5.7.1). We used uniform priors on the model parameters, but for other common priors similar behaviour can be expected. Clearly $p_1$ is better for about the first 100 000 outcomes, gaining a head start of approximately 40 000 bits. Ideally we should predict the initial 100 000 outcomes using $p_1$ and the rest using $p_2$. However, $p_{\text{bma}}$ only starts to behave like $p_2$ when it *catches up* with $p_1$ at a sample size of about 310 000, when the code length of $p_2$ drops below that of $p_1$. Thus, in the shaded area $p_{\text{bma}}$ behaves like $p_1$ while $p_2$ is making better predictions of those outcomes:

**Figure 5.1** The Catch-up Phenomenon




since at $n = 100\,000$, $p_2$ is 40 000 bits behind, and at $n = 310\,000$, it has caught up, in between it must have outperformed $p_1$ by 40 000 bits! The general pattern that first one model is better and then another occurs widely, both on real-world data and in theoretical settings. We argue that failure to take this effect into account leads to the suboptimal rate of convergence achieved by Bayes factor model selection and related methods. We have developed an alternative method to combine distributions $p_1$ and $p_2$ into a single distribution $p_{\rm sw}$, which we call the *switch-distribution*, defined in Section 5.1. Figure 5.1 shows that $p_{\rm sw}$ behaves like $p_1$ initially, but in contrast to $p_{\rm bma}$ it starts to mimic $p_2$ *almost immediately* after $p_2$ starts making better predictions; it essentially does this *no matter what sequence $x^n$ is actually observed*. $p_{\rm sw}$ differs from $p_{\rm bma}$ in that it is based on a prior distribution on *sequences of models* rather than simply a prior distribution on models. This allows us to avoid the implicit assumption that there is one model which is best at all sample sizes. After conditioning on past observations, the posterior we obtain gives a better indication of which model performs best *at the current sample size*, thereby achieving a faster rate of convergence. Indeed, the switch-distribution is very closely related to earlier algorithms for *tracking the best expert* developed in the universal prediction literature; see also Section 5.6 [46, 95, 94, 63]; however, the applications we have in mind and the theorems we prove are completely different.

The remainder of the paper is organised as follows. In Section 5.1 we introduce our basic concepts and notation, and we then define the switch-distribution. While in the example above, we switched just between two models, the general definition allows switching between elements of any finite or countably infinite

set of models. In Section 5.2 and 5.3 we show that model selection based on the switch-distribution is consistent (Theorem 5.2.1). Then in Section 5.3 we show that the switch-distribution achieves a rate of convergence that is never significantly worse than that of Bayesian model averaging, and we develop a number of tools that can be used to bound the rate of convergence "in sum" compared to other model selection criteria. Using these results we show that, in particular, the switch-distribution achieves the *worst-case optimal* rate of convergence when it is applied to histogram density estimation. In Section 5.4 we provide additional discussion, where we compare our "in sum" convergence rates to the standard definition of convergence rates, and where we motivate our conjecture that the switch-distribution in fact achieves the worst-case optimal rate of convergence in a very wide class of problems including regression using mean squared error. In Section 5.5 we give a practical algorithm that computes the switch-distribution. Theorem 5.5.1 shows that the run-time for $k$ predictors is $\Theta(n \cdot k)$ time. In Sections 5.6 and Section 5.7 we put our work in a broader context and explain how our results fit into the existing literature. Specifically, Section 5.7.1 describes a strange implication of the catch-up phenomenon for Bayes factor model selection. The proofs of all theorems are in Section 5.9.

## 5.1 The Switch-Distribution for Model Selection and Prediction

### 5.1.1 Preliminaries

Suppose $X^\infty = (X_1, X_2, \ldots)$ is a sequence of random variables that take values in sample space $\mathcal{X} \subseteq \mathbb{R}^d$ for some $d \in \mathbb{Z}^+ = \{1, 2, \ldots\}$. For $n \in \mathbb{N} = \{0, 1, 2, \ldots\}$, let $x^n = (x_1, \ldots, x_n)$ denote the first $n$ outcomes of $X^\infty$, such that $x^n$ takes values in the product space $\mathcal{X}^n = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$. (We let $x^0$ denote the empty sequence.) Let $\mathcal{X}^* = \bigcup_{n=0}^\infty \mathcal{X}^n$. For $m > n$, we write $X_{n+1}^m$ for $(X_{n+1}, \ldots, X_m)$, where $m = \infty$ is allowed. We sometimes omit the subscript when $n = 0$ and write $X^m$ rather than $X_0^m$.

Any distribution $P(X^\infty)$ may be defined in terms of a sequential *prediction strategy* $p$ that predicts the next outcome at any time $n \in \mathbb{N}$. To be precise: Given the previous outcomes $x^n$ at time $n$, this prediction strategy should issue a conditional density $p(X_{n+1}|x^n)$ with corresponding distribution $P(X_{n+1}|x^n)$ for the next outcome $X_{n+1}$. Such sequential prediction strategies are sometimes called *prequential forecasting systems* [27]. An instance is given in Example 14 below. We assume that the density $p(X_{n+1}|x^n)$ is taken relative to either the usual Lebesgue measure (if $\mathcal{X}$ is continuous) or the counting measure (if $\mathcal{X}$ is countable). In the latter case $p(X_{n+1}|x^n)$ is a probability mass function. It is natural to define the joint density $p(x^m|x^n) = p(x_{n+1}|x^n) \cdots p(x_m|x^{m-1})$ and let $P(X_{n+1}^\infty|x^n)$ be the unique distribution on $\mathcal{X}^\infty$ such that, for all $m > n$, $p(X_{n+1}^m|x^n)$ is the density of

its marginal distribution for $X_{n+1}^m$. To ensure that $P(X_{n+1}^\infty|x^n)$ is well-defined even if $\mathcal{X}$ is continuous, we will only allow prediction strategies satisfying the natural requirement that for any $k \in \mathbb{Z}^+$ and any fixed measurable event $A_{k+1} \subseteq \mathcal{X}_{k+1}$ the probability $P(A_{k+1}|x^k)$ is a measurable function of $x^k$. This requirement holds automatically if $\mathcal{X}$ is countable.

### 5.1.2 Model Selection and Prediction

In *model selection* the goal is to choose an explanation for observed data $x^n$ from a potentially infinite list of candidate models $\mathcal{M}_1$, $\mathcal{M}_2$, ... We consider *parametric models*, which we define as sets $\{p_\theta : \theta \in \Theta\}$ of prediction strategies $p_\theta$ that are indexed by elements of $\Theta \subseteq \mathbb{R}^d$, for some smallest possible $d \in \mathbb{N}$, the number of degrees of freedom. A model is more commonly viewed as a set of distributions, but since distributions can be viewed as prediction strategies as explained above, we may think of a model as a set of prediction strategies as well. Examples of model selection are regression based on a set of basis functions such as polynomials ($d$ is the number of coefficients of the polynomial), the variable selection problem in regression [80, 56, 101] ($d$ is the number of variables), and histogram density estimation [74] ($d$ is the number of bins minus 1). A *model selection criterion* is a function $\delta : \mathcal{X}^* \to \mathbb{Z}^+$ that, given any data sequence $x^n \in \mathcal{X}^*$, selects the model $\mathcal{M}_k$ with index $k = \delta(x^n)$.

With each model $\mathcal{M}_k$ we associate a single prediction strategy $\bar{p}_k$. The bar emphasises that $\bar{p}_k$ is a meta-strategy based on the prediction strategies in $\mathcal{M}_k$. In many approaches to model selection, for example AIC and LOO, $\bar{p}_k$ is defined using some estimator $\hat{\theta}_k$, which maps a sequence $x^n$ of previous observations to an estimated parameter value that represents a "best guess" of the true/best distribution in the model. Prediction is then based on this estimator: $\bar{p}_k(X_{n+1} \mid x^n) = p_{\hat{\theta}_k(x^n)}(X_{n+1} \mid x^n)$, which also defines a joint density $\bar{p}_k(x^n) = \bar{p}_k(x_1) \cdots \bar{p}_k(x_n|x^{n-1})$. The Bayesian approach to model selection or model averaging goes the other way around. It starts out with a prior $w$ on $\Theta_k$, and then defines the Bayesian marginal density

$$\bar{p}_k(x^n) = \int_{\theta \in \Theta_k} p_\theta(x^n) w(\theta) \, \mathrm{d}\theta. \tag{5.1}$$

When $\bar{p}_k(x^n)$ is non-zero this joint density induces a unique conditional density $\bar{p}_k(X_{n+1} \mid x^n) = \bar{p}_k(X_{n+1}, x^n)/\bar{p}_k(x^n)$, which is equal to the mixture of $p_\theta \in \mathcal{M}_k$ according to the posterior, $w(\theta|x^n) = p_\theta(x^n)w(\theta)/\int p_\theta(x^n)w(\theta)\,\mathrm{d}\theta$, based on $x^n$. Thus the Bayesian approach also defines a prediction strategy $\bar{p}_k(X_{n+1}|x^n)$, whose corresponding distribution may be thought of as an estimator. From now on we sometimes call the distributions induced by $\bar{p}_1, \bar{p}_2, \ldots$ "estimators", even if they are Bayesian. We may usually think of the estimators $\bar{p}_k$ as universal codes relative to $\mathcal{M}_k$ [39]. This unified view is known as the *prequential approach to statistics* or *predictive MDL* [27, 69].

**Example 14.** Suppose $\mathcal{X} = \{0, 1\}$. Then a prediction strategy $\bar{p}$ may be based on the Bernoulli model $\mathcal{M} = \{p_\theta \mid \theta \in [0, 1]\}$ that regards $X_1, X_2, \ldots$ as a sequence of independent, identically distributed Bernoulli random variables with $P_\theta(X_{n+1} = 1) = \theta$. We may predict $X_{n+1}$ using the maximum likelihood (ML) estimator based on the past, i.e. using $\hat{\theta}(x^n) = n^{-1} \sum_{i=1}^{n} x_i$. The prediction for $x_1$ is then undefined. If we use a smoothed ML estimator such as the Laplace estimator, $\hat{\theta}'(x^n) = (n + 2)^{-1}(\sum_{i=1}^{n} x_i + 1)$, then all predictions are well-defined. It is well-known that the predictor $\bar{p}'$ defined by $\bar{p}'(X_{n+1} \mid x^n) = p_{\hat{\theta}'(x^n)}(X_{n+1})$ equals the Bayesian predictive distribution based on a uniform prior. Thus in this case a Bayesian predictor and an estimation-based predictor coincide!

In general, for a $k$-dimensional parametric model $\mathcal{M}_k$, we can define $\bar{p}_k(X_{n+1} \mid x^n) = p_{\hat{\theta}'_k(x^n)}(X_{n+1})$ for some smoothed ML estimator $\hat{\theta}'_k$. The joint distribution with density $\bar{p}_k(x^n)$ will then resemble, but in general not be precisely equal to, the Bayes marginal distribution with density $\bar{p}_k(x^n)$ under some prior on $\mathcal{M}_k$ [39].

### 5.1.3 The Switch-Distribution

Suppose $p_1, p_2, \ldots$ is a list of prediction strategies for $X^\infty$. (Although here the list is infinitely long, the developments below can with little modification be adjusted to the case where the list is finite.) We first define a family $\mathcal{Q} = \{q_{\mathbf{s}} : \mathbf{s} \in \mathbb{S}\}$ of combinator prediction strategies that switch between the original prediction strategies. Here the parameter space $\mathbb{S}$ is defined as

$$\mathbb{S} = \{(t_1, k_1), \ldots, (t_m, k_m) \in (\mathbb{N} \times \mathbb{Z}^+)^m \mid m \in \mathbb{Z}^+, 0 = t_1 < \ldots < t_m\}. \tag{5.2}$$

The parameter $\mathbf{s} \in \mathbb{S}$ specifies the identities of $m$ constituent prediction strategies and the sample sizes, called *switch-points*, at which to switch between them. For $\mathbf{s} = ((t'_1, k'_1), \ldots, (t'_{m'}, k'_{m'}))$, we define $t_i(\mathbf{s}) = t'_i$, $k_i(\mathbf{s}) = k'_i$ and $m(\mathbf{s}) = m'$. We omit the argument when the parameter $\mathbf{s}$ is clear from context, e.g. we write $t_3$ for $t_3(\mathbf{s})$. For each $\mathbf{s} \in \mathbb{S}$ the corresponding $q_{\mathbf{s}} \in \mathcal{Q}$ is defined as:

$$q_{\mathbf{s}}(X_{n+1}|x^n) = \begin{cases} p_{k_1}(X_{n+1}|x^n) & \text{if } n < t_2, \\ p_{k_2}(X_{n+1}|x^n) & \text{if } t_2 \le n < t_3, \\ \vdots & \vdots \\ p_{k_{m-1}}(X_{n+1}|x^n) & \text{if } t_{m-1} \le n < t_m, \\ p_{k_m}(X_{n+1}|x^n) & \text{if } t_m \le n. \end{cases} \tag{5.3}$$

Switching to the same predictor multiple times is allowed. The extra switch-point $t_1$ is included to simplify notation; we always take $t_1 = 0$, so that $k_1$ represents the strategy that is used in the beginning, before any actual switch takes place. Using (5.3), we may now define the switch-distribution as a Bayesian mixture of the elements of $\mathcal{Q}$ according to a prior $\pi$ on $\mathbb{S}$:

**Definition 5.1.1** (Switch-Distribution)**.** Let $\pi$ be a probability mass function on $\mathbb{S}$. Then the switch-distribution $P_{\mathrm{sw}}$ with prior $\pi$ is the distribution for $X^\infty$ such that, for any $n \in \mathbb{Z}^+$, the density of its marginal distribution for $X^n$ is given by

$$p_{\mathrm{sw}}(x^n) = \sum_{\mathbf{s}\in\mathbb{S}} q_{\mathbf{s}}(x^n) \cdot \pi(\mathbf{s}). \tag{5.4}$$

Although the switch-distribution provides a general way to combine prediction strategies (see Section 5.6), in this paper it will only be applied to combine prediction strategies $\bar{p}_1$, $\bar{p}_2$, ... that correspond to parametric models. In this case we may define a corresponding model selection criterion $\delta_{\mathrm{sw}}$. To this end, let $K_{n+1} : \mathbb{S} \to \mathbb{Z}^+$ be a random variable that denotes the strategy/model that is used to predict $X_{n+1}$ given past observations $x^n$. Formally, $K_{n+1}(\mathbf{s}) = k_i(\mathbf{s})$ iff $t_i(\mathbf{s}) \leq n$ and $i = m(\mathbf{s}) \vee n < t_{i+1}(\mathbf{s})$. Now note that by Bayes' theorem, the prior $\pi$, together with the data $x^n$, induces a posterior $\pi(\mathbf{s} \mid x^n) \propto q_{\mathbf{s}}(x^n)\pi(\mathbf{s})$ on switching strategies $\mathbf{s}$. This posterior on switching strategies further induces a posterior on the model $K_{n+1}$ that is used to predict $X_{n+1}$. Algorithm 5.1, given in Section 5.5, efficiently computes the posterior distribution on $K_{n+1}$ given $x^n$:

$$\pi(K_{n+1} = k \mid x^n) = \frac{\sum_{\{\mathbf{s}:K_{n+1}(\mathbf{s})=k\}} \pi\big(\mathbf{s}\big) q_{\mathbf{s}}(x^n)}{p_{\mathrm{sw}}(x^n)}, \tag{5.5}$$

which is defined whenever $p_{\mathrm{sw}}(x^n)$ is non-zero. We turn this into a model selection criterion

$$\delta_{\mathrm{sw}}(x^n) = \arg\max_k \pi(K_{n+1} = k | x^n)$$

that selects the model with maximum posterior probability.

## 5.2 Consistency

If one of the models, say with index $k^*$, is actually true, then it is natural to ask whether $\delta_{\mathrm{sw}}$ is *consistent*, in the sense that it asymptotically selects $k^*$ with probability 1. Theorem 5.2.1 states that, if the prediction strategies $\bar{p}_k$ associated with the models are Bayesian predictive distributions, then $\delta_{\mathrm{sw}}$ is consistent under certain conditions which are only slightly stronger than those required for standard Bayes factor model selection consistency. Theorem 5.2.2 extends the result to the situation where the $\bar{p}_k$ are not necessarily Bayesian.

Bayes factor model selection is consistent if for all $k, k' \neq k$, $\bar{P}_k(X^\infty)$ and $\bar{P}_{k'}(X^\infty)$ are mutually singular, that is, if there exists a measurable set $A \subseteq \mathcal{X}^\infty$ such that $\bar{P}_k(A) = 1$ and $\bar{P}_{k'}(A) = 0$ [4]. For example, this can usually be shown to hold if (a) the models are nested and (b) for each $k$, $\Theta_k$ is a subset of $\Theta_{k+1}$ of $w_{k+1}$-measure 0. In most interesting applications in which (a) holds, (b) also holds [39]. For consistency of $\delta_{\mathrm{sw}}$, we need to strengthen the mutual singularity-condition to a "conditional" mutual singularity-condition: we require that, for all

$k' \neq k$ and all $x^n \in \mathcal{X}^*$, the distributions $\bar{P}_k(X_{n+1}^\infty \mid x^n)$ and $\bar{P}_{k'}(X_{n+1}^\infty \mid x^n)$ are mutually singular. For example, if $X_1, X_2, \ldots$ are independent and identically distributed (i.i.d.) according to each $P_\theta$ in all models, but also if $\mathcal{X}$ is countable and $\bar{p}_k(x_{n+1} \mid x_n) > 0$ for all $k$, all $x^{n+1} \in \mathcal{X}^{n+1}$, then this conditional mutual singularity is automatically implied by ordinary mutual singularity of $\bar{P}_k(X^\infty)$ and $\bar{P}_{k'}(X^\infty)$.

Let $E_{\mathbf{s}} = \{\mathbf{s}' \in \mathbb{S} \mid m(\mathbf{s}') > m(\mathbf{s}), (t_i(\mathbf{s}'), k_i(\mathbf{s}')) = (t_i(\mathbf{s}), k_i(\mathbf{s})) \text{ for } i = 1, \ldots, m(\mathbf{s})\}$ denote the set of all possible extensions of $\mathbf{s}$ to more switch-points. Let $\bar{p}_1$, $\bar{p}_2$, ... be Bayesian prediction strategies with respective parameter spaces $\Theta_1$, $\Theta_2$, ... and priors $w_1$, $w_2$, ..., and let $\pi$ be the prior of the corresponding switch-distribution.

**Theorem 5.2.1** (Consistency of the Switch-Distribution)**.** *Suppose $\pi$ is positive everywhere on $\{\mathbf{s} \in \mathbb{S} \mid m(\mathbf{s}) = 1\}$ and such that for some positive constant $c$, for every $\mathbf{s} \in \mathbb{S}$, $c \cdot \pi(\mathbf{s}) \geq \pi(E_{\mathbf{s}})$. Suppose further that $\bar{P}_k(X_{n+1}^\infty \mid x^n)$ and $\bar{P}_{k'}(X_{n+1}^\infty \mid x^n)$ are mutually singular for all $k, k' \in \mathbb{Z}^+$, $k \neq k'$, $x^n \in \mathcal{X}^*$. Then, for all $k^* \in \mathbb{Z}^+$, for all $\theta^* \in \Theta_{k^*}$ except for a subset of $\Theta_{k^*}$ of $w_{k^*}$-measure 0, the posterior distribution on $K_{n+1}$ satisfies*

$$\pi(K_{n+1} = k^* \mid X^n) \stackrel{n\to\infty}{\longrightarrow} 1 \qquad \textit{with } P_{\theta^*}\textit{-probability } 1. \tag{5.6}$$

The requirement that $c \cdot \pi(\mathbf{s}) \geq \pi(E_{\mathbf{s}})$ is automatically satisfied if $\pi$ is of the form:

$$\pi(\mathbf{s}) = \pi_{\text{M}}(m)\pi_{\text{K}}(k_1)\prod_{i=2}^{m}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k_i), \tag{5.7}$$

where $\pi_{\text{M}}$, $\pi_{\text{K}}$ and $\pi_{\text{T}}$ are priors on $\mathbb{Z}^+$ with full support, and $\pi_{\text{M}}$ is geometric: $\pi_{\text{M}}(m) = \theta^{m-1}(1-\theta)$ for some $0 \leq \theta < 1$. In this case $c = \theta/(1-\theta)$.

We now extend the theorem to the case where the universal distributions $\bar{p}_1, \bar{p}_2, \ldots$ are not necessarily Bayesian, i.e. they are not necessarily of the form (5.1). It turns out that the "meta-Bayesian" universal distribution $P_{\text{sw}}$ is still consistent, as long as the following condition holds. The condition essentially expresses that, for each $k$, $\bar{p}_k$ must not be too different from a Bayesian predictive distribution based on (5.1). This can be verified if all models $\mathcal{M}_k$ are exponential families (as in, for example, linear regression problems), and the $\bar{p}_k$ represent ML or smoothed ML estimators (see Theorem 2.1 and 2.2 of [57]). We suspect that it holds as well for more general parametric models and universal codes, but we do not know of any proof.

**Condition** There exist Bayesian prediction strategies $\bar{p}_1^{\text{B}}, \bar{p}_2^{\text{B}}, \ldots$ of form (5.1), with continuous and strictly positive priors $w_1, w_2, \ldots$ such that

1. The conditions of Theorem 5.2.1 hold for $\bar{p}_1^{\text{B}}, \bar{p}_2^{\text{B}}, \ldots$ and the chosen switch-distribution prior $\pi$.

2. For all $k \in \mathbb{N}$, for each compact subset $\Theta'$ of the interior of $\Theta_k$, there exists a $K$ such that for all $\theta \in \Theta'$, with $\theta$-probability 1, for all $n$

$$-\log \bar{p}_k(X^n) + \log \bar{p}_k^{\mathrm{B}}(X^n) \leq K.$$

3. For all $k, k' \in \mathbb{N}$ with $k \neq k'$, $\bar{p}_k^{\mathrm{B}}$ and $\bar{p}_{k'}$ are mutually singular.

**Theorem 5.2.2** (Consistency of the Switch-Distribution, Part 2)**.** *Let $\bar{p}_1, \bar{p}_2, \ldots$ be prediction strategies and let $\pi$ be the prior of the corresponding switch distribution. Suppose that the condition above holds relative to $\bar{p}_1, \bar{p}_2, \ldots$ and $\pi$. Then, for all $k^* \in \mathbb{N}$, for all $\theta^* \in \Theta_{k^*}$ except for a subset of $\Theta_{k^*}$ of Lebesgue-measure 0, the posterior distribution on $K_{n+1}$ satisfies*

$$\pi(K_{n+1} = k^* \mid X^n) \stackrel{n\to\infty}{\longrightarrow} 1 \qquad \textit{with } P_{\theta^*}\textit{-probability } 1. \tag{5.8}$$

## 5.3 Optimal Risk Convergence Rates

In this section we investigate how well the switch-distribution is able to predict future data in terms of its accumulated KL-risk, which will be formally defined shortly. We first compare predictive performance of the switch-distribution to that achieved by Bayesian model averaging in Section 5.3.1, showing that, reassuringly, the summed risk achieved by $P_{\mathrm{sw}}$ is never more than a small constant higher than that achieved by $P_{\mathrm{bma}}$. Then in Section 5.3.2 we describe the general setup and establish a lemma that is used as a general tool in the analysis. Section 5.3.3 treats the case where the data are sampled from a density $p^*$ which is an element of one of the considered models $\mathcal{M}_{k^*}$ for some $k^* \in \mathbb{Z}^+$. In this case we already know from the previous section that the switch-distribution is typically consistent; here we show that it will also avoid the catch-up phenomenon as described in the introduction. Then in Section 5.3.4, we look at the situation where $p^*$ is not in any of the considered models. For this harder, nonparametric case we compare the loss of the switch distribution to that of any other model selection criterion, showing that under some conditions that depend on this reference criterion, the two losses are of the same order of magnitude. Finally in Section 5.3.5 we apply our results to the problem of histogram density estimation, showing that for the class of densities that are (uniformly) bounded away from zero and infinity, and have bounded first derivatives, the switch-distribution (based on histogram models with Bayesian estimators that have uniform priors) predicts essentially as well as any other procedure whatsoever.

The setup is as follows. Suppose $X_1, X_2, \ldots$ are distributed according to a distribution $P^*$ with density $p^* \in \mathcal{M}^*$, where $\mathcal{M}^*$ is an arbitrary set of densities on $\mathcal{X}^\infty$. Specifically, $X_1, X_2, \ldots$ do not have to be i.i.d. We abbreviate "$P^*$ described by density $p^* \in \mathcal{M}^*$" to "$P^* \in \mathcal{M}^*$".

For prediction we use a sequence of parametric models $\mathcal{M}_1, \mathcal{M}_2, \ldots$ with associated estimators $\bar{P}_1, \bar{P}_2, \ldots$ as before. We write $\mathcal{M} = \cup_{i=1}^\infty \mathcal{M}_i$. In Section 5.3.3

we assume that $P^* \in \mathcal{M}$, while in Section 5.3.4 we assume that this is not the case, i.e. $P^* \in \mathcal{M}^* \setminus \mathcal{M}$.

Given $X^{n-1} = x^{n-1}$, we will measure how well any estimator $\bar{P}$ predicts $X_n$ in terms of the Kullback-Leibler (KL) divergence $D(P^*(X_n = \cdot \mid x^{n-1}) \| \bar{P}(X_n = \cdot \mid x^{n-1}))$ [6]. Suppose that $P$ and $Q$ are distributions for some random variable $Y$, with densities $p$ and $q$ respectively. Then the KL divergence from $P$ to $Q$ is

$$D(P\|Q) = E_P\left[\log \frac{p(Y)}{q(Y)}\right]. \tag{5.9}$$

KL divergence is never negative, and reaches zero if and only if $P$ equals $Q$. Taking an expectation over $X^{n-1}$ leads to the following (standard) definition of the *risk* of estimator $\bar{P}$ at sample size $n$ relative to KL divergence:

$$R_n(P^*, \bar{P}) = \underset{X^{n-1}\sim P^*}{E}\left[D\big(P^*(X_n = \cdot \mid X^{n-1}) \| \bar{P}(X_n = \cdot \mid X^{n-1})\big)\right]. \tag{5.10}$$

The following identity connects accumulated statistical KL-risk to the information-theoretic redundancy (see e.g. [6] or [39, Chapter 15]) : for all $n$ we have

$$\sum_{i=1}^{n} R_i(P^*, \bar{P}) = \sum_{i=1}^{n} E\left[\log \frac{p^*(X_i \mid X^{i-1})}{\bar{p}(X_i \mid X^{i-1})}\right] = E\left[\log \frac{p^*(X^n)}{\bar{p}(X^n)}\right] = D\left(P^{*(n)} \| \bar{P}^{(n)}\right), \tag{5.11}$$

where the superscript $(n)$ denotes taking the marginal of the distribution on the first $n$ outcomes.

### 5.3.1 The Switch-distribution vs Bayesian Model Averaging

Here we show that the summed risk achieved by switch-distribution is never much higher than that of Bayesian model averaging, which is itself never much higher than that of any of the estimators $\bar{P}_k$ under consideration.

**Lemma 5.3.1.** *Let $P_{\mathrm{sw}}$ be the switch-distribution for $\bar{P}_1, \bar{P}_2, \ldots$ with prior $\pi$ of the form (5.7). Let $P_{\mathrm{bma}}$ be the Bayesian model averaging distribution for the same estimators, defined with respect to the same prior on the estimators $\pi_{\kappa}$. Then, for all $n \in \mathbb{Z}^+$, all $x^n \in \mathcal{X}^n$, and all $k \in \mathbb{Z}^+$,*

$$p_{\mathrm{sw}}(x^n) \quad \geq \quad \pi_{\mathrm{M}}(1) p_{\mathrm{bma}}(x^n) \quad \geq \quad \pi_{\mathrm{M}}(1)\pi_{\mathrm{K}}(k)\bar{p}_k(x^n).$$

*Consequently, for all $P^* \in \mathcal{M}^*$ we have*

$$\begin{aligned} &\sum_{i=1}^{n} R_i(P^*, P_{\mathrm{sw}}) \\ \leq &\sum_{i=1}^{n} R_i(P^*, P_{\mathrm{bma}}) - \log \pi_{\mathrm{M}}(1) \\ \leq &\sum_{i=1}^{n} R_i(P^*, \bar{P}_k) \quad - \log \pi_{\mathrm{M}}(1) - \log \pi_{\mathrm{K}}(k). \end{aligned}$$

*Proof.* For the first part we underestimate sums:

$$\begin{aligned} p_{\mathrm{sw}}(x^n) &= \sum_{m \in \mathbb{Z}^+} \sum_{\mathbf{s} \in \mathbb{S}: m(\mathbf{s})=m} q_{\mathbf{s}}(x^n)\pi(\mathbf{s}) \;\geq\; \pi_{\mathrm{M}}(1) \cdot \sum_{k' \in \mathbb{Z}^+} \pi_{\mathrm{K}}(k')\bar{p}_{k'}(x^n) \\ &= \pi_{\mathrm{M}}(1) \cdot p_{\mathrm{bma}}(x^n), \\ p_{\mathrm{bma}}(x^n) &= \sum_{k' \in \mathbb{Z}^+} \bar{p}_{k'}(x^n)\pi_{\mathrm{K}}(k') \;\geq\; \pi_{\mathrm{K}}(k)\bar{p}_k(x^n). \end{aligned}$$

We apply (5.11) to obtain the difference in summed risk:

$$\begin{aligned} \sum_{i=1}^{n} R_i(P^*, P_{\mathrm{sw}}) &= E\left[\log \frac{p^*(X^n)}{p_{\mathrm{sw}}(X^n)}\right] \\ &\leq E\left[\log \frac{p^*(X^n)}{\pi_{\mathrm{M}}(1)p_{\mathrm{bma}}(X^n)}\right] = \sum_{i=1}^{n} R_i(P^*, P_{\mathrm{bma}}) - \log \pi_{\mathrm{M}}(1), \\ \sum_{i=1}^{n} R_i(P^*, P_{\mathrm{bma}}) &= E\left[\log \frac{p^*(X^n)}{p_{\mathrm{bma}}(X^n)}\right] \\ &\leq E\left[\log \frac{p^*(X^n)}{\pi_{\mathrm{K}}(k)\bar{p}_k(X^n)}\right] = \sum_{i=1}^{n} R_i(P^*, \bar{P}_k) - \log \pi_{\mathrm{K}}(k). \end{aligned}$$

□

As mentioned in the introduction, one advantage of model averaging using $p_{\mathrm{bma}}$ is that it always predicts almost as well as the estimator $\bar{p}_k$ for *any* $k$, including the $\bar{p}_k$ that yields the best predictions overall. Lemma 5.3.1 shows that this property is shared by $p_{\mathrm{sw}}$, which multiplicatively dominates $p_{\mathrm{bma}}$. In the following sections, we will investigate under which circumstances the switch-distribution may achieve a *lower* summed risk than Bayesian model averaging.

### 5.3.2 Improved Convergence Rate: Preliminaries

Throughout our analysis of the achieved rate of convergence we will require that the prior of the switch-distribution, $\pi$, can be factored as in (5.7), and is chosen to satisfy

$$-\log \pi_{\mathrm{M}}(m) = O(m), \quad -\log \pi_{\mathrm{K}}(k) = O(\log k), \quad -\log \pi_{\mathrm{T}}(t) = O(\log t). \tag{5.12}$$

Thus $\pi_{\text{M}}$, the prior on the total number of distinct predictors, is allowed to decrease either exponentially (as required for Theorem 5.2.1) or polynomially, but $\pi_{\text{T}}$ and $\pi_{\text{K}}$ cannot decrease faster than polynomially. For example, we could set $\pi_{\text{T}}(t) = 1/(t(t+1))$ and $\pi_{\text{K}}(k) = 1/(k(k+1))$, or we could take the universal prior on the integers [68].

As competitors to the switch-distribution we introduce a slight generalisation of model selection criteria:

**Definition 5.3.2** (Oracle)**.** An *oracle* is a function $\sigma : \mathcal{M}^* \times \mathcal{X}^* \to \mathbb{Z}^+$ that is given not only the observed data $x^n \in \mathcal{X}^*$, but also the generating distribution $P^* \in \mathcal{M}^*$, which it may use to choose a model index $\sigma(P^*, x^n) \in \mathbb{Z}^+$ for all $n \in \mathbb{Z}^+$.

Given an oracle $\sigma$, for any $P^*$ and $n, x^{n-1}$, we abbreviate $\sigma_i = \sigma(P^*, x^{i-1})$ for $1 \leq i \leq n$. We define $P_\sigma$ as the distribution on $X^\infty$ with marginal densities $p_\sigma(x^n) = \prod_{i=1}^n p_{\sigma_i}(x_i|x^{i-1})$ for all $n, x^n$. Furthermore, we may split the sequence $\sigma_1, \ldots, \sigma_n$ into segments where the same model is chosen. Now let $m_n(\sigma)$ be the maximum number of such distinct segments over all $P^*$ and all $x^{n-1} \in \mathcal{X}^{n-1}$. That is, let

$$m_n(\sigma) = \max_{P^*} \max_{x^{n-1} \in \mathcal{X}^{n-1}} |\{1 \leq i \leq n-1 : \sigma_i \neq \sigma_{i+1}\}| + 1. \tag{5.13}$$

(The maximum always exists, because for any $P^*$ and $x^{n-1}$ the number of segments is at most $n$.)

The following lemma expresses that any oracle $\sigma$ that does not select overly complex models, can be approximated by the switch-distribution with a maximum overhead that depends on $m_n(\sigma)$, its maximum number of segments. We will typically be interested in oracles $\sigma$ such that this maximum is small in comparison to the sample size, $n$. The lemma is a tool in establishing the convergence rate of $P_{\text{sw}}$, both in the parametric and the nonparametric contexts considered below.

**Lemma 5.3.3.** *Let $P_{\text{sw}}$ be the switch-distribution, defined with respect to a sequence of estimators $\bar{P}_1, \bar{P}_2, \ldots$ as introduced above, with any prior $\pi$ that satisfies the conditions in* (5.12) *and let $P^* \in \mathcal{M}^*$. Suppose $\tau$ is a positive real number and $\sigma$ is an oracle such that*

$$\sigma(P^*, x^{i-1}) \leq i^\tau \tag{5.14}$$

*for all $i \in \mathbb{Z}^+$, all $x^{i-1} \in \mathcal{X}^{i-1}$. Then*

$$\sum_{i=1}^n R_i(P^*, P_{\text{sw}}) = \sum_{i=1}^n R_i(P^*, P_\sigma) + m_n(\sigma) \cdot O(\log n), \tag{5.15}$$

*where the multiplicative constant in the big-O notation depends only on $\tau$ and the constants implicit in* (5.12)*.*

*Proof.* Using (5.11) we can rewrite (5.15) into the equivalent claim

$$E\left[\log \frac{p_\sigma(X^n)}{p_{\text{sw}}(X^n)}\right] = m_n(\sigma) \cdot O(\log n), \tag{5.16}$$

which we proceed to prove. For all $n$, $x^n \in \mathcal{X}^n$, there exists a $\mathbf{s} \in \mathbb{S}$ with $m(\mathbf{s}) \leq m_n(\sigma)$ that represents the same sequence of models as $\sigma$, so that $q_{\mathbf{s}}(x^i \mid x^{i-1}) = p_{\sigma_i}(x^i \mid x^{i-1})$ for $1 \leq i \leq n$. Consequently, we can bound

$$p_{\text{sw}}(x^n) \;=\; \sum_{\mathbf{s}' \in \mathbb{S}} q_{\mathbf{s}'}(x^n) \cdot \pi(\mathbf{s}') \;\geq\; q_{\mathbf{s}}(x^n)\pi(\mathbf{s}) \;=\; p_\sigma(x^n)\pi(\mathbf{s}). \tag{5.17}$$

By assumption (5.14) we have that $\sigma$, and therefore $\mathbf{s}$, never selects a model $\mathcal{M}_k$ with index $k$ larger than $i^\tau$ to predict the $i$th outcome. Together with (5.12) this implies that

$$\begin{aligned}
&-\log \pi(\mathbf{s}) \\
&= -\log \pi_{\text{M}}(m(\mathbf{s})) - \log \pi_{\text{K}}(k_1(\mathbf{s})) + \sum_{j=2}^{m(\mathbf{s})} \left(-\log \pi_{\text{T}}(t_j(\mathbf{s}) \mid t_{j-1}(\mathbf{s})) - \log \pi_{\text{K}}(k_j(\mathbf{s}))\right) \\
&= O(m(\mathbf{s})) + \sum_{j=1}^{m(\mathbf{s})} O(\log t_j(\mathbf{s})) + O(\log k_j(\mathbf{s})) \\
&= O(m(\mathbf{s})) + \sum_{j=1}^{m(\mathbf{s})} O(\log t_j(\mathbf{s})) + O\left(\log\left((t_j(\mathbf{s})+1)^\tau\right)\right) = m_n(\sigma) \cdot O(\log n),
\end{aligned} \tag{5.18}$$

where the multiplicative constant in the big-O in the final expression depends only on $\tau$ and the multiplicative constants in (5.12). Together (5.17) and (5.18) imply (5.16), which was to be shown. □

In the following subsections, we compare the accumulated risk of $P_{\text{sw}}$ to that of $P_\sigma$ for various oracles $\sigma$; in all these cases our results are independent of the data generating distribution $P^* \in \mathcal{M}^*$. For that reason it will be convenient to define the worst-case summed risk of the switch-distribution and of oracle $\sigma$:

$$G_{\text{sw}}(n) := \sup_{P^* \in \mathcal{M}^*} \sum_{i=1}^{n} R_i(P^*, P_{\text{sw}}), \text{ and} \tag{5.19}$$

$$G_\sigma(n) := \sup_{P^* \in \mathcal{M}^*} \sum_{i=1}^{n} R_i(P^*, P_\sigma). \tag{5.20}$$

We will also compare the accumulated risk of $P_{\text{sw}}$ to the *minimax risk in sum*, defined as

$$G_{\text{mm-fix}}(n) := \inf_{\bar{P}} \sup_{P^* \in \mathcal{M}^*} \sum_{i=1}^{n} R_i(P^*, \bar{P}). \tag{5.21}$$

Here the infimum is over all prequential forecasting systems $\bar{P}$ for which, for each $n$, $x^{n-1} \in \mathcal{X}^{n-1}$, $\bar{P}(X_n = \cdot \mid X^{n-1} = x^{n-1})$ admits a density. Equivalently, the infimum is over all sequences of $n$ estimators $\bar{P}(X_1), \bar{P}(X_2 \mid X^1), \ldots, \bar{P}(X_n \mid X^{n-1})$. Note that there is no requirement that $\bar{P}$ maps $x^n$ to a distribution in $\mathcal{M}^*$ or $\mathcal{M}$; we are looking at the worst case over all possible estimators, irrespective of the model $\mathcal{M}$ used to approximate $\mathcal{M}^*$. Thus, we may call $\bar{P}$ an "out-model estimator" [39]. The notation $G_{\text{mm-fix}}$ will be clarified in Section 5.4, where we compare convergence in sum with more standard notions of convergence.

### 5.3.3 The Parametric Case

Here we assume that $P^* \in \mathcal{M}_{k^*}$ for some $k^* \in \mathbb{Z}^+$, but we also consider that if $\mathcal{M}_1, \mathcal{M}_2, \ldots$ are of increasing complexity, then the catch-up phenomenon may occur, meaning that at small sample sizes, some estimator $\bar{P}_k$ with $k < k^*$ may achieve lower risk than $\bar{P}_{k^*}$. The following lemma shows that the predictive performance of the switch-distribution is never much higher than the predictive performance of the best oracle that iterates through the models in order of increasing complexity.

**Lemma 5.3.4.** *Let $P_{\text{sw}}$ be the switch distribution, defined with respect to a sequence of estimators $\bar{P}_1, \bar{P}_2, \ldots$ as above, with prior $\pi$ satisfying* (5.12)*. Let $k^* \in \mathbb{Z}^+$, and let $\sigma$ be any oracle such that for all $P^*$, all $x^\infty$, we have that $\sigma(P^*, x^n)$ is monotonically nondecreasing in $n$; furthermore for sufficiently large $n$, we have $\sigma(P^*, x^n) = k^*$. Then*

$$G_{\text{sw}}(n) - G_\sigma(n) \leq \sup_{P^* \in \mathcal{M}^*} \left( \sum_{i=1}^{n} R_i(P^*, P_{\text{sw}}) - \sum_{i=1}^{n} R_i(P^*, P_\sigma) \right) = k^* \cdot O(\log n).$$

*In particular, if for some $c' > 0$, for all sufficiently large $n$, $G_\sigma(n) \geq c \log n$ (i.e. $G_\sigma = \Omega(\log n)$), then there is a $c$ such that*

$$\limsup_{n \to \infty} \frac{G_{\text{sw}}(n)}{G_\sigma(n)} \leq c.$$

The additional risk compared to $P_\sigma$ is of order $\log n$. In the parametric case, we often have $G_{\text{mm-fix}}(n)$ proportional to $\log n$ (Section 5.4.1). If that is the case, and if, as seems reasonable, there is an oracle $\sigma$ that satisfies the given restrictions and that achieves summed risk proportional to $G_{\text{mm-fix}}(n)$, then also the switch-distribution achieves a proportional summed risk.

*Proof.* The first inequality is a consequence of the general rule that for two functions $f$ and $g$, we have $\sup_x f(x) - \sup_x g(x) \leq \sup_x (f(x) - g(x))$. We proceed

to show the asymptotics of the second term, which has the supremum on the outside. Let $\mathbf{s} = (0, k^*)$. We have, uniformly for all $n$, $x^n \in \mathcal{X}^n$,

$$-\log p_{\mathrm{sw}}(x^n) \;=\; -\log \sum_{\mathbf{s}' \in \mathbb{S}} q_{\mathbf{s}'}(x^n)\pi(\mathbf{s}') \;\leq\; -\log q_{\mathbf{s}}(x^n) + \pi(\mathbf{s}). \tag{5.22}$$

Since $\sigma$ satisfies (5.14) for suitably chosen $\tau$, and the properties of $\sigma$ ensure $m_n(\sigma) \leq k^*$, we can apply Lemma 5.3.3. The first part of the lemma is then obtained by taking the supremum over $P^*$. To establish the second part, we can choose a $c$ such that

$$\limsup_{n\to\infty} \frac{G_{\mathrm{sw}}(n)}{G_\sigma(n)} \leq \limsup_{n\to\infty} \frac{G_\sigma(n) + k^* \cdot O(\log n)}{G_\sigma(n)} \leq 1 + \limsup_{n\to\infty} \frac{k^* \cdot O(\log n)}{c' \log n} = c. \tag{5.23}$$

□

### 5.3.4 The Nonparametric Case

In this section we develop an analogue of Lemma 5.3.4 for the nonparametric case, where there is no $k$ such that $P^* \in \mathcal{M}_k$. It is then applied to the problem of histogram density estimation.

**Lemma 5.3.5.** *Let $P_{\mathrm{sw}}$ be the switch-distribution, defined with respect to a sequence of estimators $\bar{P}_1, \bar{P}_2, \ldots$ as above, with any prior $\pi$ that satisfies the conditions in (5.12). Let $f : \mathbb{Z}^+ \to [0, \infty)$ and let $\mathcal{M}^*$ be a set of distributions on $X^\infty$. Suppose there exist an oracle $\sigma$ and positive constants $\tau$ and $c$ such that*

*(i)* $\sigma(P^*, x^{i-1}) \leq i^\tau$ *for all* $i \in \mathbb{Z}^+$, *all* $x^{i-1} \in \mathcal{X}^{i-1}$,

*(ii)* $m_n(\sigma) \log n = o(f(n))$, *and*

*(iii)* $\limsup_{n\to\infty} \frac{G_\sigma(n)}{f(n)} \leq c$.

*Then*

$$\limsup_{n\to\infty} \frac{G_{\mathrm{sw}}(n)}{f(n)} \leq c. \tag{5.24}$$

*Proof.* By (i) we can apply Lemma 5.3.3 to $\sigma$. Using conditions (ii) and (iii), a derivation similar to (5.23) completes the proof:

$$\limsup_{n\to\infty} \frac{G_{\mathrm{sw}}(n)}{f(n)} \leq$$
$$\limsup_{n\to\infty} \frac{G_\sigma(n) + m_n(\sigma)O(\log n)}{f(n)} \leq c + \limsup_{n\to\infty} \frac{m_n(\sigma)O(\log n)}{f(n)} = c. \qquad \square$$

Lemma 5.3.5 can be used to show minimax rates of convergence relative to specific nonparametric model classes $\mathcal{M}^*$. The general idea is to apply the lemma with $f(n)$ equal to the minimax risk in sum $G_{\text{mm-fix}}(n)$ (see (5.21)). It will be seen that in many standard nonparametric settings, one can exhibit an oracle $\sigma$ that only switches sporadically (Condition (ii) of the lemma) and that achieves $G_{\text{mm-fix}}(n)$ (Condition (iii)). The lemma then implies that $P_{\text{sw}}$ achieves the minimax risk as well. As a proof of concept, we now show this in detail for histogram density estimation. In Section 5.4.2, we discuss possibilities for extending the reasoning to more general settings.

### 5.3.5 Example: Histogram Density Estimation

Rissanen, Speed and Yu [74] consider density estimation based on histogram models with equal-width bins relative to a restricted set $\mathcal{M}^*$ of "true" distributions, identified in this section by their densities on the unit interval $\mathcal{X} = [0, 1]$. The restriction on $\mathcal{M}^*$ is that there should exist constants $0 < c_0 < 1 < c_1$ such that for every density $p \in \mathcal{M}^*$, for all $x \in \mathcal{X}$, $c_0 \leq p(x) \leq c_1$ and $|p'(x)| \leq c$, where $p'$ denotes the first derivative of $p$; unlike in the paper we require a uniform bound $c$ on the derivative which may not depend on $p \in \mathcal{M}^*$. The densities are extended to sequences by independence: $p(x^n) \equiv p^n(x^n) = \prod_{i=1}^n p(x_i)$ for $x^n \in \mathcal{X}^n$.

The histogram model $\mathcal{M}_k$ is the set of densities on $\mathcal{X} = [0, 1]$ that are constant within the $k$ bins $[0, a_1]$, $(a_1, a_2]$, ..., $(a_{k-1}, 1]$, where $a_i = i/k$, i.e. $\mathcal{M}_k$ contains all densities with $p_\theta$ such that, for all $x, x' \in [0, 1]$, if $x$ and $x'$ lie in the same bin, then $p_\theta(x) = p_\theta(x')$. The $k-1$-dimensional vector $\theta = (\theta_1, \ldots, \theta_{k-1})$ denotes the probability masses of the first $k-1$ bins. The last bin then gets the remaining mass, $1 - \sum_{j=1}^{k-1} \theta_k$. Note that the number of free parameters is one less than the number of bins.

Here we model densities from $\mathcal{M}^*$ by sequences of densities based on histogram models of an increasing number of bins as more data become available. Rissanen, Speed and Yu define prediction strategies $\bar{p}_k$ on $\mathcal{X}^\infty$ relative to each histogram model $\mathcal{M}_k$. For model $\mathcal{M}_k$, the conditional predictions are given by

$$\bar{p}_k(x_{n+1} \mid x^n) := \frac{n_{x_{n+1}}(x^n) + 1}{n + k} \cdot k, \tag{5.25}$$

where $n_{x_{n+1}}(x^n)$ denotes the number of outcomes in $x^n$ that fall into the same bin as $x_{n+1}$. These $\bar{p}_k$ correspond to Bayesian estimators relative to a uniform prior on the set of parameters, $\{\theta\}$.

Yu [107] shows that, relative to the $\mathcal{M}^*$ defined above, the minimax-risk in sum satisfies

$$\limsup_{n \to \infty} \frac{G_{\text{mm-fix}}(n)}{n^{1/3}} = C,$$

where $C$ is a constant that depends on the constants $c, c_0, c_1$ used in the definition of $\mathcal{M}^*$. In [74] it is shown that the simple strategy that uses the histogram model

with $\left\lceil n^{1/3} \right\rceil$ bins to predict the $n$th outcome achieves this minimax risk in sum up to a constant multiplicative factor:

**Theorem 5.3.6** (Theorem 1 from [74]). *For all $p^* \in \mathcal{M}^*$*

$$\limsup_{n\to\infty} \frac{\sum_{i=1}^{n} R_i(P^*, \bar{P}_{\lceil n^{1/3} \rceil})}{n^{1/3}} \leq A_{p^*}, \tag{5.26}$$

*where $A_{p^*}$ is a constant that depends only on $c_{p^*}$, the bound on the first derivative of $p^*$.*

We will now show that the switch-distribution also achieves the minimax risk in sum up to the same constant factor. The idea is to view $\bar{P}_{\lceil n^{1/3} \rceil}$ as an oracle. Even though it makes no use of $P^*$ in its selection, $\bar{P}_{\lceil n^{1/3} \rceil}$ clearly satisfies Definition 5.3.2, the definition of an oracle. We would like to apply Lemma 5.3.5 to oracle $\bar{P}_{\lceil n^{1/3} \rceil}$, but we cannot do so, since the oracle switches prediction strategies polynomially often in $n$. To prove that $P_{\mathrm{sw}}$ achieves a rate of $n^{1/3}$, we first need to consider a cruder version of $\bar{P}_{\lceil n^{1/3} \rceil}$ that still achieves the minimax rate, but only switches logarithmically often. This is the key to the proof of Theorem 5.3.7.

**Theorem 5.3.7.** *Let $p_{\mathrm{sw}}$ denote the switch-distribution with prior $\pi$ that satisfies the conditions in* (5.12)*, relative to histogram prediction strategies $\bar{p}_1$, $\bar{p}_2$, ... For all $p^* \in \mathcal{M}^*$ this switch-distribution satisfies*

$$\limsup_{n\to\infty} \frac{\sum_{i=1}^{n} R_i(P^*, P_{\mathrm{sw}})}{n^{1/3}} \leq A_{p^*}, \tag{5.27}$$

*where $A_{p^*}$ is the same constant as in* (5.26)*.*

We first note that in [74], Theorem 5.3.6 is proved from the following more general theorem, which gives an upper bound on the risk of any prediction strategy that uses a histogram model with approximately $\left\lceil n^{1/3} \right\rceil$ bins to predict outcome $n$:

**Theorem 5.3.8.** *For any $\alpha \geq 1$, any $k \in [\left\lceil (n/\alpha)^{1/3} \right\rceil, \left\lceil n^{1/3} \right\rceil]$, and any $p^* \in \mathcal{M}^*$,*

$$R_n(P^*, \bar{P}_k) \leq \alpha^{2/3} C_{p^*} n^{-2/3}, \tag{5.28}$$

*where $C_{p^*}$ depends only on the upper bound $c$ on the first derivative of $p^*$.*

In [74] the theorem is only proven for $\alpha = 1$, but the proof remains valid unchanged for any $\alpha > 1$. From this, Theorem 5.3.6 follows by summing (5.28), and approximating $\sum_{i=1}^{n} i^{-2/3}$ by an integral. We will now apply it to prove Theorem 5.3.7 as well.

*Proof of Theorem 5.3.7.* Choose any $p^* \in \mathcal{M}^*$, and let $\alpha > 1$ be arbitrary. Let $t_j = \lceil \alpha^{j-1} \rceil - 1$ for $j \in \mathbb{Z}^+$ be a sequence of switch-points, and define an oracle $\sigma_\alpha(P^*, x^{n-1}) := \lceil (t_j+1)^{1/3} \rceil$ for any $x^{n-1} \in \mathcal{X}^{n-1}$, where $j$ is chosen such that $n \in [t_j + 1, t_{j+1}]$. By applying Lemma 5.3.5 to $\sigma_\alpha$ with $f(n) = n^{1/3}$, $\tau = 1$ and $c = \alpha^{2/3} A_{p^*}$, we immediately obtain

$$\limsup_{n\to\infty} \frac{\sum_{i=1}^n R_i(P^*, P_{\mathrm{sw}})}{n^{1/3}} \leq \alpha^{2/3} A_{p^*} \tag{5.29}$$

for any $\alpha > 1$. Theorem 5.3.7 then follows, because the left-hand side of this expression does not depend on $\alpha$. We still need to show that conditions (i)–(iii) of Lemma 5.3.5 are satisfied.

Note that $\sigma_\alpha$ uses approximately $\left\lceil n^{1/3} \right\rceil$ bins to predict the $n$th outcome, but has relatively few switch-points: it satisfies $m_n(\sigma_\alpha) \leq \lceil \log_\alpha n \rceil + 2$. Thus, conditions (i) and (ii) are comfortably satisfied. To verify (iii), note that the selected number of bins is close to $\left\lceil n^{1/3} \right\rceil$ in the sense of Theorem 5.3.8: For $n \in [t_j + 1, t_{j+1}]$ we have

$$\left\lceil (t_j+1)^{1/3} \right\rceil = \left\lceil \left( \frac{n}{n/(t_j+1)} \right)^{1/3} \right\rceil \in \left[ \left\lceil (n/\alpha)^{1/3} \right\rceil, \left\lceil n^{1/3} \right\rceil \right] \tag{5.30}$$

using $n \leq t_{j+1}$ and $(t_{j+1})/(t_j+1) \leq \alpha$. We can now apply Theorem 5.3.8 to obtain

$$\limsup_{n\to\infty} \frac{\sum_{i=1}^n R_i(P^*, \sigma_\alpha)}{n^{1/3}} \leq \limsup_{n\to\infty} \frac{\sum_{i=1}^n \alpha^{2/3} C_{p^*} i^{-2/3}}{n^{1/3}} \leq \alpha^{2/3} A_{p^*}, \tag{5.31}$$

showing that (iii) is satisfied and Lemma 5.3.5 can be applied to prove the theorem. □

Theorem 5.3.7 shows that the switch distribution obtains the optimal convergence rate in sum relative to $\mathcal{M}^*$. In [74] it is also shown that standard two-part MDL does *not* achieve this rate; it is slower by an $O(\log n)$ factor. The analysis leading to this result also strongly suggests that Bayesian model averaging based on a discrete prior on the Bayesian marginal distributions $\bar{p}_1, \bar{p}_2, \ldots$ given by (5.25) is also a factor $O(\log n)$ slower compared to the minimax rate [39]. By Theorem 5.3.6, $\delta(x^n) := \lceil n^{1/3} \rceil$ defines a very simple model selection criterion which does achieve the optimal rate in sum relative to $\mathcal{M}^*$, but, in contrast to the switch distribution, it is inconsistent. Moreover, if we are lucky enough to be in a scenario where $p^*$ actually allows a *faster* than minimax optimal in-sum convergence by letting the number of bins grow as $n^\gamma$ for some $\gamma \neq \frac{1}{3}$, the switch-distribution would be able to take advantage of this whereas $\delta$ cannot.

## 5.4 Further Discussion of Convergence in the Nonparametric Case

In Section 5.4.1 we analyse the relation between our convergence rates in sum and standard convergence rates. In Section 5.4.2, we explore possible future applications of Lemma 5.3.5 to establish minimax convergence rates for model classes $\mathcal{M}_1, \mathcal{M}_2, \ldots$ beyond histograms.

### 5.4.1 Convergence Rates in Sum

Let $g : \mathbb{N} \to \mathbb{R}^+$ and $h : \mathbb{N} \to \mathbb{R}^+$ be two functions that converge to 0 with increasing $n$. We say that $g$ converges to 0 at rate $h$ if $\limsup_{n\to\infty} \frac{g(n)}{h(n)} \leq 1$. We say that $g$ converges to 0 *in sum* at rate $h$ if $\limsup_{n\to\infty} \frac{\sum_{i=1}^n g(i)}{\sum_{i=1}^n h(i)} \leq 1$. This notion of convergence has been considered by, among others, Rissanen, Speed and Yu [74], Barron [6], Poland and Hutter [65], and was investigated in detail by Grünwald [39]. Note that, if $g$ converges to 0 at rate $h$, and $\lim_{n\to\infty} \sum_{i=1}^n h(n) = \infty$, then $g$ also converges in sum at rate $h$. Conversely, suppose that $g$ converges in sum at rate $h$. Does this also imply that $g$ converges to 0 at rate $h$ in the ordinary sense? The answer is "almost": as shown in [39], $g(n)$ may be strictly greater than $h(n)$ for some $n$, but the *gap* between any two $n$ and $n' > n$ at which $g$ is larger than $h$ must become infinitely large with increasing $n$.

We will now informally compare the "convergence in sum" results of the previous section with more standard results about individual risk convergence. We will write $h(n) \asymp g(n)$ if for some $0 < c_1 < c_2$, for all large $n$, $c_1 g(n) < h(n) < c_2 g(n)$. The minimax convergence rate relative to a set of distributions $\mathcal{M}^*$ is defined as

$$h_{\text{mm}}(n) = \inf_{\bar{P}} \sup_{P^* \in \mathcal{M}} R_n(P^*, \bar{P}), \tag{5.32}$$

where $\bar{P}$ is any estimator that allows a density, it is not required to lie in $\mathcal{M}^*$ or $\mathcal{M}$. If a sequence of estimators achieves (5.32) to within a constant factor, we say that it converges at the "minimax optimal rate." Such a sequence of estimators will also achieve the minimax rate in sum, defined as

$$G_{\text{mm-var}}(n) = \sum_{i=1}^n h_{\text{mm}}(i) = \inf_{\bar{P}} \sum_{i=1}^n \sup_{P^* \in \mathcal{M}^*} R_i(P^*, \bar{P}), \tag{5.33}$$

where $\bar{P}$ now ranges over all prequential forecasting systems (i.e. sequences of estimators). In many nonparametric density estimation and regression problems, the minimax risk $h_{\text{mm}}(n)$ is of order $n^{-\gamma}$ for some $1/2 < \gamma < 1$ (see, for example, [105, 106, 9]), i.e. $h_{\text{mm}}(n) \asymp n^{-\gamma}$, where $\gamma$ depends on the smoothness assumptions on the densities in $\mathcal{M}^*$. We call the situation with $\mathcal{M}^*$ such that $h_{\text{mm}}(n) \asymp n^{-\gamma}$

the "standard nonparametric case." In this standard case, we have

$$G_{\text{mm-var}}(n) \asymp \sum_{i=1}^{n} i^{-\gamma} \asymp \int_1^n x^{-\gamma}\,\mathrm{d}x \asymp n^{1-\gamma}. \tag{5.34}$$

Similarly, in standard parametric problems, the minimax risk $h_{\text{mm}}(n) \asymp 1/n$. In that case, analogously to (5.34), we see that the minimax risk in sum $G_{\text{mm-var}}$ is of order $\log n$.

Note, however, that our result for histograms (and, more generally, for any rate-of-convergence result that may be based on Lemmata 5.3.3, 5.3.4 and 5.3.5), is based on a scenario where $P^*$, while allowed to depend on $n$, is kept fixed over the terms in the sum from 1 to $n$. Indeed, in Theorem 5.3.7 we showed that $P_{\text{sw}}$ achieves the minimax rate in sum $G_{\text{mm-fix}}(n)$ as defined in (5.21). Comparing to (5.33), we see that the supremum is moved outside of the sum. Fortunately, $G_{\text{mm-fix}}$ and $G_{\text{mm-var}}$ are usually of the same order: in the parametric case, e.g. $\mathcal{M}^* = \bigcup_{k \le k^*} \mathcal{M}_k$, both $G_{\text{mm-fix}}$ and $G_{\text{mm-var}}$ are of order $\log n$. For $G_{\text{mm-var}}$, we have already seen this. For $G_{\text{mm-fix}}$, this is a standard information-theoretic result, see for example [23]. In the standard nonparametric case, when the standard minimax rate is of order $n^{-\gamma}$ and therefore $G_{\text{mm-var}} \asymp n^{1-\gamma}$, it also holds that $G_{\text{mm-fix}}(n) \asymp n^{1-\gamma}$ [106, page 1582]. To see this, let $P_{\text{mm-fix}}$ be any prequential forecasting system that achieves $G_{\text{mm-fix}}$ as defined in (5.21) (if such a $P_{\text{mm-fix}}$ does not exist, take any $P$ that, for each $n$, achieves the infimum to within $\varepsilon_n$ for some sequence $\varepsilon_1, \varepsilon_2, \ldots$ tending to 0). Now define the prequential forecasting system

$$P_{\text{Césaro}}(x_n \mid x^{n-1}) := \frac{1}{n} \sum_{i=1}^{n} P_{\text{mm-fix}}(x_n \mid x^{i-1}).$$

Thus, $P_{\text{Césaro}}$ is obtained as a time ("Césaro"-) average of $P_{\text{mm-fix}}$. It now follows by applying Jensen's inequality as in Proposition 15.2 of [39] (or the corresponding results in [102] or [106]) that

$$\sup_{P^*} R_n(P^*, P_{\text{Césaro}}) \le \sup_{P^*} \frac{1}{n} \sum_{i=1}^{n} R_i(P^*, P_{\text{mm-fix}}) = n^{-1} O(n^{1-\gamma}) = O(n^{-\gamma}), \tag{5.35}$$

so that $\sum_{j=1}^{n} \sup_{P^*} R_j(P^*, P_{\text{Césaro}}) = O(\sum_{j=1}^{n} j^{-\gamma}) = O(n^{1-\gamma})$, and it follows that

$$G_{\text{mm-var}}(n) = O(n^{1-\gamma}) = O(G_{\text{mm-fix}}(n)). \tag{5.36}$$

Since, trivially, $G_{\text{mm-fix}}(n) \le G_{\text{mm-var}}(n)$, it follows that $G_{\text{mm-var}}(n) \asymp n^{1-\gamma}$. The underlying idea of basing predictive distributions on Césaro-averages is not new; see for example [43, Section 9] and [102]. It is described in detail in [39].

Summarising, both in standard parametric and nonparametric cases, $G_{\text{mm-fix}}$ and $G_{\text{mm-var}}$ are of comparable size. Therefore, Lemma 5.3.4 and 5.3.5 do suggest

that, both in standard parametric and nonparametric cases, $P_{\text{sw}}$ achieves the minimax convergence rate $G_{\text{mm-fix}}$ (and Theorem 5.3.7 shows that it actually does so in a specific nonparametric case). One caveat is in order though: the fact that $G_{\text{mm-fix}}$ and $G_{\text{mm-var}}$ are of comparable size does *not* imply that $P_{\text{sw}}$ also achieves the varying-$P^*$-minimax rate $G_{\text{mm-var}}$. We cannot prove the analogue of Lemma 5.3.4 and Lemma 5.3.5 with the supremum over $P^*$ inside the sum in $G_{\text{sw}}$ and $G_\sigma$. Therefore, we cannot prove even for histogram density estimation, that $P_{\text{sw}}$ achieves $G_{\text{mm-var}}$. Nevertheless, we do suspect that in the standard nonparametric case, $G_{\text{mm-fix}}(n) \asymp G_{\text{mm-var}}(n) \asymp n^{1-\gamma}$, whenever $P_{\text{sw}}$ achieves the fixed-$P^*$ minimax rate $G_{\text{mm-fix}}$, it also achieves the varying-$P^*$ minimax rate $G_{\text{mm-var}}$. The reason for this conjecture is that, if we can assume that the data are i.i.d. under all $P^* \in \mathcal{M}^*$, then whenever $P_{\text{sw}}$ achieves $G_{\text{mm-fix}}$, a small modification of $P_{\text{sw}}$ will achieve $G_{\text{mm-var}}$. Indeed, define the *Césaro-switch distribution* as

$$P_{\text{Césaro-sw}}(x_n \mid x^{n-1}) := \frac{1}{n} \sum_{i=1}^{n} P_{\text{sw}}(x_n \mid x^{i-1}).$$

Applying (5.35) to $P_{\text{Césaro-sw}}$ rather than $P_{\text{Césaro}}$, we see that $P_{\text{Césaro-sw}}$ achieves the varying-$P^*$-minimax rate whenever $P_{\text{sw}}$ achieves the fixed-$P^*$-minimax rate. Since, intuitively, $P_{\text{Césaro-sw}}$ learns "slower" than $P_{\text{sw}}$, we suspect that $P_{\text{sw}}$ itself then achieves the varying-$P^*$-minimax rate as well. However, in the parametric case, $h_{\text{mm}}(n) \asymp 1/n$ whereas $G_{\text{mm-fix}}(n)/n \asymp (\log n)/n$. Then the reasoning of (5.35) does not apply any more, and $P_{\text{Césaro-sw}}$ may not achieve the minimax rate for varying $P^*$. We suspect that this is not a coincidence — a recent result by Yang [103] suggests that, in the parametric case, *no* model selection/averaging criterion can achieve both consistency and minimax optimal varying-$P^*$ rates $G_{\text{mm-var}}$ (Section 5.6.1).

### 5.4.2 Beyond Histograms

The key to proving Theorem 5.3.7, the minimax convergence result for histogram density estimation, is the existence of an oracle $\sigma_\alpha$ which achieves the minimax convergence rate, but which, at the same time, switches only a logarithmic number of times. The theorem followed by applying Lemma 5.3.5 with this oracle. It appears that the same technique can be applied in many other standard nonparametric settings (with $h_{\text{mm}}(n) \asymp n^{-\gamma}$) as well. Important examples include linear regression based on full approximation sets of functions such as polynomials [106, 101] or splines, with random i.i.d. design and i.i.d. normally distributed noise with known variance $\sigma^2$. The development in Section 6.2 of [101] indicates that an analogue of Theorem 5.3.7 can be proven for such cases. Here the models $\mathcal{M}_k$ are families of conditional distributions $P_\theta$ for $Y \in \mathbb{R}$ given $X \in [0,1]^d$ for some $d > 0$, where $\theta = (\theta_1, \ldots, \theta_k)$ and $P_\theta$ expresses that $Y_i = \sum_{j=1}^{k} \theta_j \phi_j(X_i) + U$, with $\phi_j$ being the $j$-th basis function in the approximation set, and $U$, the noise,

is a zero-mean Gaussian random variable. The forecasting systems $\bar{p}_1, \bar{p}_2, \ldots$ are now based on maximum likelihood estimators rather than Bayes predictive distributions.

Another candidate is density estimation based on sequences of exponential families as introduced by Barron and Sheu [9], when the estimators $\bar{p}_1, \bar{p}_2, \ldots$ are based on Bayesian MAP estimators defined with respect to $k$-dimensional exponential families, and $\mathcal{M}^*$ is taken to be the set of densities $p$ such that $\log p$ is contained in some *Sobolev space* with smoothness parameter $r$ [7]. Preliminary investigations suggest that $P_{\rm sw}$ achieves the minimax convergence rates in both the linear regression and the density estimation setting, but, at the time of writing, we have not yet proven any formal statements.

## 5.5 Efficient Computation

For priors $\pi$ as in (5.7), the posterior probability on predictors $p_1, p_2, \ldots$ can be efficiently computed sequentially, provided that $\pi_{\rm T}(Z = n \mid Z \geq n)$ and $\pi_{\rm K}$ can be calculated quickly and that $\pi_{\rm M}(m) = \theta^m(1-\theta)$ is geometric with parameter $\theta$, as is also required for Theorem 5.2.1 and permitted in the theorems and lemma's of Section 5.3 that require (5.12). The algorithm resembles FIXED-SHARE [46], but whereas FIXED-SHARE implicitly imposes a geometric distribution for $\pi_{\rm T}$, we allow general priors by varying the shared weight with $n$, and through the addition of the $\pi_{\rm M}$ component of the prior, we ensure that the additional loss compared to the best prediction strategy does not grow with the sample size, a crucial property for consistency.

To ensure finite running time, rather than $P_{\rm sw}$ the algorithm uses a potentially defective distribution $P$ that assigns smaller or equal probability to all events. It is obtained by restricting $P_{\rm sw}$ to using only a finite nonempty set $\mathcal{K}_n$ of prediction strategies at sample size $n$. Then, analogously to (5.4), for any $n$ the density of the marginal distribution of $P$ on $X^n$ is given by $p(x^n) = \sum_{\mathbf{s} \in \mathbb{S}'} q_{\mathbf{s}}(x^n) \cdot \pi(\mathbf{s})$, where $\mathbb{S}' := \{\mathbf{s} \in \mathbb{S} \mid \forall n \in \mathbb{Z}^+ : K_n(\mathbf{s}) \in \mathcal{K}_n\}$ denotes the parameter space restricted to those prediction strategies that are considered at each sample size. We use the indicator function, $\mathbf{1}_A(x) = 1$ if $x \in A$ and 0 otherwise. Here is the algorithm:

This algorithm can be used to obtain fast convergence in the sense of Section 5.3, and, as long as $\pi$ does not vary with $n$, consistency in the sense of Theorem 5.2.1. Note that the running time $\Theta(\sum_{n=1}^{N} |\mathcal{K}_n|)$ is typically of the same order as that of fast model selection criteria like AIC and BIC: for example if the number of considered prediction strategies is fixed at $K$ then the running time is $\Theta(K \cdot N)$. In the interest of clarity and simplicity we only prove the theorem below, which assumes $P = P_{\rm sw}$, but the reader may verify that the algorithm remains valid for defective $P$.

**Theorem 5.5.1.** *If $P = P_{\rm sw}$, then Algorithm 5.1 correctly reports $P(K_{n+1}, x^n)$.*

**Algorithm 5.1** Switch($x^N$)

**for** $k \in \mathcal{K}_1$ **do** initialise $w_k^a \leftarrow \theta \cdot \pi_{\text{K}}(k)$; $w_k^b \leftarrow (1-\theta) \cdot \pi_{\text{K}}(k)$ **end for**
**for** $n=1,\ldots,N$ **do**
  Report $P(K_n, x^{n-1}) = w^a_{K_{n+1}} + w^b_{K_{n+1}}$ (a $K$-sized array)
  **for** $k \in \mathcal{K}_n$ **do** $w_k^a \leftarrow w_k^a \cdot p_k(x_n|x^{n-1})$; $w_k^b \leftarrow w_k^b \cdot p_k(x_n|x^{n-1})$ **end for**
  $\texttt{pool} \leftarrow \pi_{\text{T}}(Z = n \mid Z \geq n) \cdot \sum_{k \in \mathcal{K}_n} w_k^a$
  **for** $k \in \mathcal{K}_{n+1}$ **do**
    $w_k^a \leftarrow w_k^a \cdot \mathbf{1}_{\mathcal{K}_n}(k) \cdot \pi_{\text{T}}(Z \neq n \mid Z \geq n) \; + \; \texttt{pool} \cdot \pi_{\text{K}}(k) \cdot \theta$
    $w_k^b \leftarrow w_k^b \cdot \mathbf{1}_{\mathcal{K}_n}(k) \; + \; \texttt{pool} \cdot \pi_{\text{K}}(k) \cdot (1-\theta)$
  **end for**
**end for**
Report $P(K_{N+1}, x^N) = w^a_{K_{N+1}} + w^b_{K_{N+1}}$

The proof is given in Section 5.9.4.

## 5.6 Relevance and Earlier Work

### 5.6.1 AIC-BIC; Yang's Result

Over the last 25 years or so, the question whether to base model selection on AIC or BIC type methods has received a lot of attention in the theoretical and applied statistics literature, as well as in fields such as psychology and biology where model selection plays an important role (googling "AIC *and* BIC" gives 355000 hits) [85, 41, 40, 10, 33, 30, 81]. It has even been suggested that, since these two types of methods have been designed with different goals in mind (optimal prediction vs. "truth hunting"), one should not expect procedures that combine the best of both types of approaches to exist [81]. Our Theorem 5.2.1 and our results in Section 5.3 show that, at least in some cases, one can get the best of both worlds after all, and model averaging based on $P_{\text{sw}}$ achieves the minimax optimal convergence rate. In typical parametric settings ($P^* \in \mathcal{M}$), model selection based on $P_{\text{sw}}$ is consistent, and Lemma 5.3.4 suggests that model averaging based on $P_{\text{sw}}$ is within a constant factor of the minimax optimal rate in parametric settings. Superficially, our results may seem to contradict the central conclusion of Yang [103]. Yang shows that there are scenarios in linear regression where no model selection or model combination criterion can be both consistent and achieve the minimax rate of convergence.

Yang's result is proved for a variation of linear regression in which the estimation error is measured on the previously observed design points. This setup cannot be directly embedded in our framework. Also, Yang's notion of model combination is somewhat different from the model averaging that is used to compute $P_{\text{sw}}$. Thus, formally, there is no contradiction between Yang's results and

ours. Still, the setups are so similar that one can easily imagine a variation of Yang's result to hold in our setting as well. Thus, it is useful to analyse how these "almost" contradictory results may coexist. We suspect (but have no proof) that the underlying reason is the definition of our minimax convergence rate in sum (5.21) in which $P^*$ is allowed to depend on $n$, but then the risk with respect to that same $P^*$ is summed over all $i = 1, \ldots, n$. Yang's result holds in a parametric scenario, where there are two nested parametric models, and data are sampled from a distribution in one of them. Then both $G_{\text{mm-fix}}$ and $G_{\text{mm-var}}$ are of the same order $\log n$, but it may of course be possible that there does exist a minimax optimal procedure that is also consistent, relative to the $G_{\text{mm-fix}}$-game, in which $P^*$ is kept fixed once $n$ has been determined, while there does not exist a minimax optimal procedure that is also consistent, relative to the $G_{\text{mm-var}}$-game, in which $P^*$ is allowed to vary. Indeed, while in Section 5.4.1 we have established that $P_{\text{Césaro-sw}}$, a slight variation of $P_{\text{sw}}$, achieves the minimax optimal convergence rates $G_{\text{mm-var}}$ and $h_{\text{mm}}$ for some nonparametric $\mathcal{M}^*$, which suggests that $P_{\text{sw}}$ achieves these rates as well, we do not have such a result for parametric $\mathcal{M}^*$. Yang's results indicate that such an analogue may not exist.

Several other authors have provided procedures which have been designed to behave like AIC whenever AIC is better, and like BIC whenever BIC is better; and which empirically seem to do so; these include *model meta-selection* [30, 21], and Hansen and Yu's *gMDL* version of MDL regression [41]; also the "mongrel" procedure of [99] has been designed to improve on Bayesian model averaging for small samples. Compared to these other methods, ours seems to be the first that *provably* is both consistent and minimax optimal in terms of risk, for some classes $\mathcal{M}^*$. The only other procedure that we know of for which somewhat related results have been shown, is a version of cross-validation proposed by Yang [104] to select between AIC and BIC in regression problems. Yang shows that a particular form of cross-validation will asymptotically select AIC in case the use of AIC leads to better predictions, and BIC in the case that BIC leads to better predictions. In contrast to Yang, we use a single paradigm rather than a mix of several ones (such as AIC, BIC and cross-validation) – essentially our paradigm is just that of universal individual-sequence prediction, or equivalently, the individual-sequence version of predictive MDL, or equivalently, Dawid's prequential analysis applied to the log scoring rule. Indeed, our work has been heavily inspired by prequential ideas; in Dawid [29] it is already suggested that model selection should be based on the *transient* behaviours in terms of sequential prediction of the estimators within the models: one should select the model which is optimal at the given sample size, and this will change over time. Although Dawid uses standard Bayesian mixtures of parametric models as his running examples, he implicitly suggests that other ways (the details of which are left unspecified) of combining predictive distributions relative to parametric models may be preferable, especially in the nonparametric case where the true distribution is outside any of the parametric models under consideration.

### 5.6.2 Prediction with Expert Advice

Since the switch-distribution has been designed to perform well in a setting where the optimal predictor $\bar{p}_k$ changes over time, our work is also closely related to the algorithms for *tracking the best expert* in the universal prediction literature [46, 95, 94, 63]. However, those algorithms are usually intended for data that are sequentially generated by a mechanism whose behaviour changes over time. In sharp contrast, our switch distribution is especially suitable for situations where data are sampled from a *fixed* (though perhaps non-i.i.d.) source after all; the fact that one model temporarily leads to better predictions than another is caused by the fact that each "expert" $\bar{p}_k$ has itself already been designed as a universal predictor/estimator relative to some large set of distributions $\mathcal{M}_k$. The elements of $\mathcal{M}_k$ may be viewed as "base" predictors/experts, and the $\bar{p}_k$ may be thought of as meta-experts/predictors. Because of this two-stage structure, which meta-predictor $\bar{p}_k$ is best changes over time, even though the optimal base-predictor $\arg\min_{p\in\mathcal{M}} R_n(p^*, p)$ does not change over time.

If one of the considered prediction strategies $\bar{p}_k$ makes the best predictions eventually, our goal is to achieve consistent model selection: the total number of switches should also remain bounded. To this end we have defined the switch distribution such that positive prior probability is associated with switching finitely often and thereafter using $\bar{p}_k$ for all further outcomes. We need this property to prove that our method is consistent. Other dynamic expert tracking algorithms, such as the FixedShare algorithm [46], have been designed with different goals in mind, and as such they do not have this property. Not surprisingly then, our results do not resemble any of the existing results in the "tracking"-literature.

## 5.7 The Catch-Up Phenomenon, Bayes and Cross-Validation

### 5.7.1 The Catch-Up Phenomenon is Unbelievable! (according to $p_{\mathrm{bma}}$)

On page 122 we introduced the marginal Bayesian distribution $p_{\mathrm{bma}}(x^n) := \sum_k w(k)\bar{p}_k(x^n)$. If the distributions $\bar{p}_k$ are themselves Bayesian marginal distributions as in (5.1), then $p_{\mathrm{bma}}$ may be interpreted as (the density corresponding to) a distribution on the data that reflects some prior beliefs about the domain that is being modelled, as represented by the priors $w(k)$ and $w_k(\theta)$. If $w(k)$ and $w_k(\theta)$ truly reflected some decision-maker's a priori beliefs, then it is clear that the decision-maker would like to make sequential predictions of $X_{n+1}$ given $X^n = x^n$ based on $p_{\mathrm{bma}}$ rather than on $p_{\mathrm{sw}}$. Indeed, as we now show, the catch-up phenomenon as depicted in Figure 5.1 is exceedingly unlikely to take place under $p_{\mathrm{bma}}$, and *a priori* a subjective Bayesian should be prepared to bet a lot of money that it does not

occur. To see this, consider the *no-hypercompression inequality* [39], versions of which are also known as "Barron's inequality" [5] and "competitive optimality of the Shannon-Fano code" [25]. It states that for any two distributions $P$ and $Q$ for $X^\infty$, the $P$-probability that $Q$ outperforms $P$ by $k$ bits or more when sequentially predicting $X_1, X_2, \ldots$ is exponentially small in $k$: for each $n$,

$$P(-\log q(X^n) \leq -\log p(X^n) - k) \leq 2^{-k}.$$

Plugging in $p_{\rm bma}$ for $p$, and $p_{\rm sw}$ for $q$, we see that what happened in Figure 5.1 ($p_{\rm sw}$ outperforming $p_{\rm bma}$ by about 40000 bits) is an event with probability no more than $2^{-40000}$ according to $p_{\rm bma}$. Yet, in many practical situations, the catch-up phenomenon does occur and $p_{\rm sw}$ gains significantly compared to $p_{\rm bma}$. This can only be possible because either the models are wrong (clearly, The Picture of Dorian Gray has not been drawn randomly from a finite-order Markov chain), or because the priors are "wrong" in the sense that they somehow don't match the situation one is trying to model. For this reason, some subjective Bayesians, when we confronted them with the catch-up phenomenon, have argued that it is just a case of "garbage in, garbage out" (GIGO): when the phenomenon occurs, then, rather than using the switch-distribution, one should reconsider the model(s) and prior(s) one wants to use, and, once one has found a superior model $\mathcal{M}'$ and prior $w'$, one should use $p_{\rm bma}$ relative to $\mathcal{M}'$ and $w'$. Of course we agree that *if* one can come up with better models, one should of course use them. Nevertheless, we strongly disagree with the GIGO point of view: We are convinced that in practice, "correct" priors may be impossible to obtain; similarly, people are forced to work with "wrong" models all the time. In such cases, rather than embarking on a potentially never-ending quest for better models, the hurried practitioner may often prefer to use the imperfect – yet still useful – models that he has available, *in the best possible manner*. And then it makes sense to use $p_{\rm sw}$ rather than the Bayesian $p_{\rm bma}$: the best one can hope for in general is to regard the distributions in one's models as prediction strategies, and try to predict as well as the best strategy contained in any of the models, and $p_{\rm sw}$ is better at this than $p_{\rm bma}$. Indeed, the catch-up phenomenon raises some interesting questions for Bayes factor model selection: no matter what the prior is, by the no-hypercompression inequality above with $p = p_{\rm bma}$ and $q = p_{\rm sw}$, when comparing two models $\mathcal{M}_1$ and $\mathcal{M}_2$, before seeing any data, a Bayesian *always* believes that the switch-distribution will not substantially outperform $p_{\rm bma}$, which implies that a Bayesian *cannot* believe that, with non-negligible probability, a complex model $\bar{p}_2$ can at first predict substantially worse than a simple model $\bar{p}_1$ and then, for large samples, can predict substantially better. Yet in practice, this happens all the time!

### 5.7.2 Nonparametric Bayes

A more interesting subjective Bayesian argument against the switch distribution would be that, in the nonparametric setting, the data are sampled from some

$P^* \in \mathcal{M}^* \setminus \mathcal{M}$, and is not contained in any of the parametric models $\mathcal{M}_1, \mathcal{M}_2, \ldots$ Yet, under the standard hierarchical prior used in $p_{\text{bma}}$ (first a discrete prior on the model index, then a density on the model parameters), we have that with prior-probability 1, $P^*$ is "parametric", i.e. $P^* \in \mathcal{M}_k$ for some $k$. Thus, our prior distribution is not really suitable for the situation that we are trying to model in the nonparametric setting, and we should use a nonparametric prior instead. While we completely agree with this reasoning, we would immediately like to add that the question then becomes: what nonparametric prior *should* one use? Nonparametric Bayes has become very popular in recent years, and it often works surprisingly well. Still, its practical and theoretical performance strongly depends on the type of priors that are used, and it is often far from clear what prior to use in what situation. In some situations, some nonparametric priors achieve optimal rates of convergence, but others can even make Bayes inconsistent [31, 39]. The advantage of the switch-distribution is that it does not require any difficult modelling decisions, but nevertheless under reasonable conditions it achieves the optimal rate of convergence in nonparametric settings, and, in the special case where one of the models on the list in fact approximates the true source extremely well, this model will in fact be identified (Theorem 5.2.1). In fact, one may think of $p_{\text{sw}}$ as specifying a very special kind of nonparametric prior, and under this interpretation, our results are in complete agreement with the nonparametric Bayesian view.

### 5.7.3 Leave-One-Out Cross-Validation

From the other side of the spectrum, it has sometimes been argued that consistency is irrelevant, since in practical situations, the true distribution is never in any of the models under consideration. Thus, it is argued, one should use AIC-type methods such as leave-one-out cross-validation, because of their predictive optimality. We strongly disagree with this argument, for several reasons: first, in practical model selection problems, one is often interested in questions such as "does $Y$ depend on feature $X_k$ or not?" For example, $\mathcal{M}_{k-1}$ is a set of conditional distributions in which $Y$ is independent of $X_k$, and $\mathcal{M}_k$ is a superset thereof in which $Y$ can be dependent on $X_k$. There are certainly real-life situations where some variable $X_j$ is truly completely irrelevant for predicting $Y$, and it may be the primary goal of the scientist to find out whether or not this is the case. In such cases, we would hope our model selection criterion to select, for large $n$, $\mathcal{M}_{k-1}$ rather than $\mathcal{M}_k$, and the problem with the AIC-type methods is that, because of their inconsistency, they sometimes do not do this. In other words, we think that consistency does matter, and we regard it as a clear advantage of the switch-distribution that it is consistent.

A second advantage over leave-one-out cross-validation is that the switch-distribution, like Bayesian methods, satisfies Dawid's *weak prequential principle* [29, 39]: the switch-distribution assesses the quality of a predictor $\bar{p}_k$ only in

terms of the quality of predictions *that were actually made*. To apply LOO on a sample $x_1, \ldots, x_n$, one needs to know the prediction for $x_i$ given $x_1, \ldots, x_{i-1}$, but also $x_{i+1}, \ldots, x_n$. In practice, these may be hard to compute, unknown or even unknowable. An example of the first are non-i.i.d. settings such as time series models. An example of the second is the case where the $\bar{p}_k$ represent, for example, weather forecasters, or other predictors which have been designed to predict the future given the past. Actual weather forecasters use computer programs to predict the probability that it will rain the next day, given a plethora of data about air pressure, humidity, temperature etc. and the pattern of rain in the past days. It may simply be impossible to apply those programs in a way that they predict the probability of rain today, given data about tomorrow.

## 5.8 Conclusion and Future Work

We have identified the catch-up phenomenon as the underlying reason for the slow convergence of Bayesian model selection and averaging. Based on this, we have defined the switch-distribution $P_{\text{sw}}$, a modification of the Bayesian marginal distribution which is consistent, but also under some conditions achieves a minimax optimal convergence rate, a significant step forward in resolving the the AIC/BIC dilemma. Different strands of future work suggest themselves:

1. Lemma 5.3.5 provides a tool to prove minimax optimal in-sum convergence of the switch-distribution for particular nonparametric model classes $\mathcal{M}^*$. However, because of time constraints we have currently only applied this to histogram density estimation. We hope to eventually show that the switch-distribution actually achieves the minimax optimal convergence rate for a wide class of nonparametric problems.

2. Since $p_{\text{sw}}$ can be computed in practice, the approach can readily be tested with real and simulated data in both density estimation and regression problems. Initial results on simulated data, on which we will report elsewhere, give empirical evidence that $p_{\text{sw}}$ behaves remarkably well in practice. Model selection based on $p_{\text{sw}}$, like for $p_{\text{bma}}$, typically identifies the true distribution at moderate sample sizes. Prediction and estimation based on $P_{\text{sw}}$ is of comparable quality to leave-one-out cross-validation (LOO) and generally, in no experiment did we find that it behaved substantially worse than either LOO or AIC.

3. It is an interesting open question whether analogues of Lemma 5.3.5 and Theorem 5.3.7 exist for model *selection* rather than averaging. In other words, in settings such as histogram density estimation where model averaging based on the switch distribution achieves the minimax convergence rate, does model selection based on the switch distribution achieve it as

well? For example, in Figure 5.1, sequentially predicting by the $\bar{p}_{K_{n+1}}$ that has maximum a posteriori probability (MAP) under the switch-distribution given data $x^n$, is only a few bits worse than predicting by model averaging based on the switch-distribution, and still outperforms standard Bayesian model averaging by about 40 000 bits. In the experiments mentioned above, we invariably found that predicting by the MAP $\bar{p}_{K_{n+1}}$ empirically converges at the same rate as using model averaging, i.e. predicting by $P_{\rm sw}$. However, we have no proof that this really must always be the case. Analogous results in the MDL literature suggest that a theorem bounding the risk of switch-based model selection, if it can be proven at all, would bound the squared Hellinger rather than the KL risk ([39], Chapter 15).

4. The way we defined $P_{\rm sw}$, it does not seem suitable for situations in which the number of considered models or model combinations is exponential in the sample size. Because of condition (i) in Lemma 5.3.5, our theoretical results do not cover this case either. Yet this case is highly important in practice, for example, in the subset selection problem [101]. It seems clear that the catch-up phenomenon can and will also occur in model selection problems of that type. Can our methods be adapted to this situation, while still keeping the computational complexity manageable? And what is the relation with the popular and computationally efficient $L_1$-approaches to model selection? [90]

## 5.9 Proofs

### 5.9.1 Proof of Theorem 5.2.1

Let $U_n = \{\mathbf{s} \in \mathbb{S} \mid K_{n+1}(\mathbf{s}) \neq k^*\}$ denote the set of "bad" parameters $\mathbf{s}$ that select an incorrect model. It is sufficient to show that

$$\lim_n \frac{\sum_{\mathbf{s}\in U_n} \pi\big(\mathbf{s}\big) q_{\mathbf{s}}(X^n)}{\sum_{\mathbf{s}\in\mathbb{S}} \pi\big(\mathbf{s}\big) q_{\mathbf{s}}(X^n)} = 0 \qquad \text{with } \bar{P}_{k^*}\text{-probability 1.} \tag{5.37}$$

To see this, suppose the theorem is false. Then there exists a $\Phi \subseteq \Theta_{k^*}$ with $w_{k^*}(\Phi) := \int_\Phi w_{k^*}(\theta) d\theta > 0$ such that (5.6) does not hold for any $\theta^* \in \Phi$. But then by definition of $\bar{P}_{k^*}$ we have a contradiction with (5.37).

Now let $A = \{\mathbf{s} \in \mathbb{S} : k_m(\mathbf{s}) \neq k^*\}$ denote the set of parameters that are bad for sufficiently large $n$. We observe that for each $\mathbf{s}' \in U_n$ there exists at least one element $\mathbf{s} \in A$ that uses the same sequence of switch-points and predictors on the first $n+1$ outcomes (this implies that $K_i(\mathbf{s}) = K_i(\mathbf{s}')$ for $i = 1, \ldots, n+1$) and has no switch-points beyond $n$ (i.e. $t_m(\mathbf{s}) \leq n$). Consequently, either $\mathbf{s}' = \mathbf{s}$ or $\mathbf{s}' \in E_{\mathbf{s}}$. Therefore

$$\sum_{\mathbf{s}'\in U_n} \pi(\mathbf{s}') q_{\mathbf{s}'}(x^n) \;\leq\; \sum_{\mathbf{s}\in A} \big(\pi(\mathbf{s}) + \pi(E_{\mathbf{s}})\big)\, q_{\mathbf{s}}(x^n) \;\leq\; (1+c) \sum_{\mathbf{s}\in A} \pi(\mathbf{s}) q_{\mathbf{s}}(x^n). \tag{5.38}$$

Defining the mixture $r(x^n) = \sum_{\mathbf{s}\in A} \pi(\mathbf{s}) q_{\mathbf{s}}(x^n)$, we will show that

$$\lim_n \frac{r(X^n)}{\pi(\mathbf{s} = (0, k^*)) \cdot \bar{p}_{k^*}(X^n)} = 0 \qquad \text{with } \bar{P}_{k^*}\text{-probability 1.} \tag{5.39}$$

Using (5.38) and the fact that $\sum_{\mathbf{s}\in\mathbb{S}} \pi(\mathbf{s}) q_{\mathbf{s}}(x^n) \geq \pi(\mathbf{s} = (0, k^*)) \cdot \bar{p}_{k^*}(x^n)$, this implies (5.37).

For all $\mathbf{s} \in A$ and $x^{t_m(\mathbf{s})} \in \mathcal{X}^{t_m(\mathbf{s})}$, by definition $Q_{\mathbf{s}}(X_{t_m+1}^{\infty} | x^{t_m})$ is equal to $\bar{P}_{k_m}(X_{t_m+1}^{\infty} | x^{t_m})$, which is mutually singular with $\bar{P}_{k^*}(X_{t_m+1}^{\infty} | x^{t_m})$ by assumption. If $\mathcal{X}$ is a separable metric space, which holds because $\mathcal{X} \subseteq \mathbb{R}^d$ for some $d \in \mathbb{Z}^+$, it can be shown that this conditional mutual singularity implies mutual singularity of $Q_{\mathbf{s}}(X^{\infty})$ and $\bar{P}_{k^*}(X^{\infty})$. To see this for countable $\mathcal{X}$, let $B_{x^{t_m}}$ be any event such that $Q_{\mathbf{s}}(B_{x^{t_m}} | x^{t_m}) = 1$ and $\bar{P}_{k^*}(B_{x^{t_m}} | x^{t_m}) = 0$. Then, for $B = \{y^{\infty} \in \mathcal{X}^{\infty} \mid y_{t_m+1}^{\infty} \in B_{y^{t_m}}\}$, we have that $Q_{\mathbf{s}}(B) = 1$ and $\bar{P}_{k^*}(B) = 0$. In the uncountable case, however, $B$ may not be measurable. In that case, the proof follows by Corollary 5.9.2 proved in Section 5.9.3. Any countable mixture of distributions that are mutually singular with $P_{k^*}$, in particular $R$, is mutually singular with $P_{k^*}$. This implies (5.39) by Lemma 3.1 of [5], which says that for any two mutually singular distributions $R$ and $P$, the density ratio $r(X^n)/p(X^n)$ goes to zero as $n \to \infty$ with $P$-probability 1. □

## 5.9.2 Proof of Theorem 5.2.2

The proof is almost identical to the proof of Theorem 5.2.1. Let $U_n = \{\mathbf{s} \in \mathbb{S} \mid K_{n+1}(\mathbf{s}) \neq k^*\}$ denote the set of "bad" parameters $\mathbf{s}$ that select an incorrect model. It is sufficient to show that

$$\lim_n \frac{\sum_{\mathbf{s}\in U_n} \pi\big(\mathbf{s}\big) q_{\mathbf{s}}(X^n)}{\sum_{\mathbf{s}\in\mathbb{S}} \pi\big(\mathbf{s}\big) q_{\mathbf{s}}(X^n)} = 0 \qquad \text{with } \bar{P}_{k^*}^{\mathrm{B}}\text{-probability 1.} \tag{5.40}$$

Note that the $q_{\mathbf{s}}$ in (5.40) are defined relative to the non-Bayesian estimators $\bar{p}_1, \bar{p}_2, \ldots$, whereas the $\bar{P}_{k^*}^{\mathrm{B}}$ on the right of the equation is the probability according to a *Bayesian* marginal distribution $\bar{P}_{k^*}^{\mathrm{B}}$, which has been chosen so that the theorem's condition holds. To see that (5.40) is sufficient to prove the theorem, suppose the theorem is false. Then, because the prior $w_{k^*}$ is mutually absolutely continuous with Lebesgue measure, there exists a $\Phi \subseteq \Theta_{k^*}$ with nonzero prior measure under $w_{k^*}$, such that (5.8) does not hold for any $\theta^* \in \Phi$. But then by definition of $\bar{P}_{k^*}^{\mathrm{B}}$ we have a contradiction with (5.40).

Using exactly the same reasoning as in the proof of Theorem 5.2.1, it follows that, analogously to (5.39), we have

$$\lim_n \frac{r(X^n)}{\pi(\mathbf{s} = (0, k^*)) \cdot \bar{p}_{k^*}^{\mathrm{B}}(X^n)} = 0 \qquad \text{with } \bar{P}_{k^*}^{\mathrm{B}}\text{-probability 1.} \tag{5.41}$$

This is just (5.39) with $r$ now referring to a mixture of switch distributions defined relative to the non-Bayesian estimators $\bar{p}_1, \bar{p}_2, \ldots$, and the $\bar{p}_{k^*}^{\mathrm{B}}$ in the denominator

and on the right referring to the Bayesian marginal distribution $\bar{P}^{\mathrm{B}}_{k^*}$. Using (5.38) and the fact that $\sum_{\mathbf{s}\in\mathbb{S}} \pi(\mathbf{s}) q_{\mathbf{s}}(x^n) \geq \pi(\mathbf{s} = (0, k^*)) \cdot \bar{p}_{k^*}(x^n)$, and the fact that, by assumption, for some $K$, for all large $n$, $\bar{p}_{k^*}(X^n) \geq \bar{p}^{\mathrm{B}}_{k^*}(X^n) 2^{-K}$ with $\bar{P}^{\mathrm{B}}_{k^*}$-probability 1, (5.41) implies (5.40). □

### 5.9.3 Mutual Singularity as Used in the Proof of Theorem 5.2.1

Let $Y^2 = (Y_1, Y_2)$ be random variables that take values in separable metric spaces $\Omega_1$ and $\Omega_2$, respectively. We will assume all spaces to be equipped with Borel $\sigma$-algebras generated by the open sets. Let $p$ be a prediction strategy for $Y^2$ with corresponding distributions $P(Y_1)$ and, for any $y^1 \in \Omega_1$, $P(Y_2|y^1)$. To ensure that $P(Y^2)$ is well-defined, we impose the requirement that for any fixed measurable event $A_2 \subseteq \Omega_2$ the probability $P(A_2|y^1)$ is a measurable function of $y^1$.

**Lemma 5.9.1.** *Suppose $p$ and $q$ are prediction strategies for $Y^2 = (Y_1, Y_2)$, which take values in separable metric spaces $\Omega_1$ and $\Omega_2$, respectively. Then if $P(Y_2|y^1)$ and $Q(Y_2|y^1)$ are mutually singular for all $y^1 \in \Omega_1$, then $P(Y^2)$ and $Q(Y^2)$ are mutually singular.*

The proof, due to Peter Harremoës, is given below the following corollary, which is what we are really interested in. Let $X^\infty = X_1, X_2, \ldots$ be random variables that take values in the separable metric space $\mathcal{X}$. Then what we need in the proof of Theorem 5.2.1 is the following corollary of Lemma 5.9.1:

**Corollary 5.9.2.** *Suppose $p$ and $q$ are prediction strategies for the sequence of random variables $X^\infty = X_1, X_2, \ldots$ that take values in respective separable metric spaces $\mathcal{X}_1, \mathcal{X}_2, \ldots$ Let $m$ be any positive integer. Then if $P(X^\infty_{m+1}|x^m)$ and $Q(X^\infty_{m+1}|x^m)$ are mutually singular for all $x^m \in \mathcal{X}^m$, then $P(X^\infty)$ and $Q(X^\infty)$ are mutually singular.*

*Proof.* The product spaces $\mathcal{X}_1 \times \cdots \times \mathcal{X}_m$ and $\mathcal{X}_{m+1} \times \mathcal{X}_{m+2} \times \cdots$ are separable metric spaces [64, pp. 5,6]. Now apply Lemma 5.9.1 with $\Omega_1 = \mathcal{X}_1 \times \cdots \times \mathcal{X}_m$ and $\Omega_2 = \mathcal{X}_{m+1} \times \mathcal{X}_{m+2} \times \cdots$. □

*Proof of Lemma 5.9.1.* For each $\omega_1 \in \Omega_1$, by mutual singularity of $P(Y_2|\omega_1)$ and $Q(Y_2|\omega_1)$ there exists a measurable set $C_{\omega_1} \subseteq \Omega_2$ such that $P(C_{\omega_1}|\omega_1) = 1$ and $Q(C_{\omega_1}|\omega_1) = 0$. As $\Omega_2$ is a metric space, it follows from [64, Theorems 1.1 and 1.2 in Chapter II] that for any $\epsilon > 0$ there exists an open set $U^\epsilon_{\omega_1} \supseteq C_{\omega_1}$ such that

$$P(U^\epsilon_{\omega_1}|\omega_1) = 1 \quad \text{and} \quad Q(U^\epsilon_{\omega_1}|\omega_1) < \epsilon. \tag{5.42}$$

As $\Omega_2$ is a separable metric space, there also exists a countable sequence $\{B_i\}_{i\geq 1}$ of open sets such that every open subset of $\Omega_2$ ($U^\epsilon_{\omega_1}$ in particular) can be expressed as the union of sets from $\{B_i\}$ [64, Theorem 1.8 in Chapter I].

Let $\{B'_i\}_{i\geq 1}$ denote a subsequence of $\{B_i\}$ such that $U^\epsilon_{\omega_1} = \bigcup_i B'_i$. Suppose $\{B'_i\}$ is a finite sequence. Then let $V^\epsilon_{\omega_1} = U^\epsilon_{\omega_1}$. Suppose it is not. Then $1 = P(U^\epsilon_{\omega_1}|\omega_1) = P(\bigcup_{i=1}^\infty B'_i|\omega_1) = \lim_{n\to\infty} P(\bigcup_{i=1}^n B'_i|\omega_1)$, because $\bigcup_{i=1}^n B'_i$ as a function of $n$ is an increasing sequence of sets. Consequently, there exists an $N$ such that $P(\bigcup_{i=1}^N B'_i|\omega_1) > 1-\epsilon$ and we let $V^\epsilon_{\omega_1} = \bigcup_{i=1}^N B'_i$. Thus in any case there exists a set $V^\epsilon_{\omega_1} \subseteq U^\epsilon_{\omega_1}$ that is a union of a finite number of elements in $\{B_i\}$ such that

$$P(V^\epsilon_{\omega_1}|\omega_1) > 1-\epsilon \quad \text{and} \quad Q(V^\epsilon_{\omega_1}|\omega_1) < \epsilon. \tag{5.43}$$

Let $\{D\}_{i\geq 1}$ denote an enumeration of all possible unions of a finite number of elements in $\{B_i\}$ and define the disjoint sequence of sets $\{A^\epsilon_i\}_{i\geq 1}$ by

$$A^\epsilon_i = \{\omega_1 \in \Omega_1 : P(D_i|\omega_1) > 1-\epsilon, Q(D_i|\omega_1) < \epsilon\} \setminus \bigcup_{j=1}^{i-1} A^\epsilon_j \tag{5.44}$$

for $i = 1, 2, \ldots$ Note that, by the reasoning above, for each $\omega_1 \in \Omega_1$ there exists an $i$ such that $\omega_1 \in A^\epsilon_i$, which implies that $\{A^\epsilon_i\}$ forms a partition of $\Omega_1$. Now, as all elements of $\{A^\epsilon_i\}$ and $\{D_i\}$ are measurable, so is the set $F^\epsilon = \bigcup_{i=1}^\infty A^\epsilon_i \times D_i \subseteq \Omega_1 \times \Omega_2$, for which we have that $P(F^\epsilon) = \sum_{i=1}^\infty P(A^\epsilon_i \times D_i) > (1-\epsilon)\sum_{i=1}^\infty P(A_i) = 1-\epsilon$ and likewise $Q(F^\epsilon) < \epsilon$.

Finally, let $G = \bigcap_{n=1}^\infty \bigcup_{k=n}^\infty F^{2^{-k}}$. Then $P(G) = \lim_{n\to\infty} P(\bigcup_{k=n}^\infty F^{2^{-k}}) \geq \lim_{n\to\infty} 1 - 2^{-n} = 1$ and $Q(G) = \lim_{n\to\infty} Q(\bigcup_{k=n} F^{2^{-k}}) \leq \lim_{n\to\infty} \sum_{k=n}^\infty 2^{-k} = \lim_{n\to\infty} 2^{-n+1} = 0$, which proves the lemma. □

### 5.9.4 Proof of Theorem 5.5.1

Before we prove Theorem 5.5.1, we first need to establish some additional properties of the prior $\pi$ as defined in (5.7). Define, for all $n \in \mathbb{N}$ and $\mathbf{s} = ((t_1, k_1), \ldots, (t_m, k_m)) \in \mathbb{S}$:

$$S_n(\mathbf{s}) := \mathbf{1}_{\{t_1,\ldots,t_m\}}(n-1); \tag{5.45}$$
$$M_n(\mathbf{s}) := \mathbf{1}_{\{t_m,t_m+1,\ldots\}}(n-1); \tag{5.46}$$
$$K_n(\mathbf{s}) := k_i \text{ for the unique } i \text{ such that } t_i < n \text{ and } i = m \vee t_{i+1} \geq n. \tag{5.47}$$

These functions denote, respectively, whether or not a switch occurs just before outcome $n$, whether or not the last switch occurs somewhere before outcome $n$ and which prediction strategy is used for outcome $n$. The prior $\pi$ determines the distributions of these random variables. We also define $E_n(\mathbf{s}) := (S_n(\mathbf{s}), M_n(\mathbf{s}), K_n(\mathbf{s}))$ as a convenient abbreviation. Every parameter value $\mathbf{s} \in \mathbb{S}$ induces an infinite sequence of values $E_1, E_2, \ldots$ The advantage of these new variables is that they allow us to reformulate the prior as a strategy for prediction of the value of the next random variable $E_{n+1}$ (which in turn determines the distribution on $X_{n+1}$ given $x^n$), given all previous random variables $E^n$. Therefore, we first calculate

the conditional probability $\pi(E_{n+1}|E^n)$ before proceeding to prove the theorem. As it turns out, our prior has the nice property that this conditional probability has a very simple functional form: depending on $E_n$, it is either zero, or a function of only $E_{n+1}$ itself. This will greatly facilitate the analysis.

**Lemma 5.9.3.** *Let* $\pi(\mathbf{s}) = \theta^{m-1}(1-\theta)\pi_{\text{K}}(k_1)\prod_{i=2}^{m}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k)$ *as in* (5.7). *For* $\pi(\mathbf{s}) > 0$ *we have*

$$\pi(E_1) := \pi_{\text{K}}(K_1)\begin{cases}\theta & \text{if } M_1 = 0\\ 1-\theta & \text{if } M_1 = 1\end{cases} \tag{5.48}$$

$$\pi(E_{n+1}|E^n) := \begin{cases}\pi_{\text{T}}(Z > n|Z \geq n) & \text{if } S_{n+1} = 0 \text{ and } M_{n+1} = 0\\ 1 & \text{if } S_{n+1} = 0 \text{ and } M_{n+1} = 1\\ \pi_{\text{T}}(Z = n|Z \geq n)\pi_{\text{K}}(K_{n+1})\theta & \text{if } S_{n+1} = 1 \text{ and } M_{n+1} = 0\\ \pi_{\text{T}}(Z = n|Z \geq n)\pi_{\text{K}}(K_{n+1})(1-\theta) & \text{if } S_{n+1} = 1 \text{ and } M_{n+1} = 1.\end{cases} \tag{5.49}$$

*Proof.* To check (5.48), note that we must have either $E_1 = (1, 1, k)$, which corresponds to $\mathbf{s} = (0, k)$ which has probability $\pi_{\text{K}}(k)(1-\theta)$ as required, or $E_1 = (1, 0, k)$. The latter corresponds to the event that $m > 1$ and $k_1 = k$, which has probability $\pi_{\text{K}}(k)\theta$.

We proceed to calculate the conditional distribution $\pi(E_{n+1}|E^n)$. We distinguish the case that $M_n(\mathbf{s}) = 1$ (the last switch defined by $\mathbf{s}$ occurs before sample size $n$), and $M_n(\mathbf{s}) = 0$ (there will be more switches). First suppose $M_n(\mathbf{s}) = 0$, and let $a_n = \max\{i \mid t_i < n\} = \sum_{i=1}^{n} S_i$. Then

$$\begin{aligned}\pi(E^n) &= \sum_{m=a_n+1}^{\infty}\sum_{t_{a_n+1}=n}\sum_{\substack{t_{a_n+2},\ldots,t_m\\ k_{a_n+1},\ldots,k_m}}\pi_{\text{M}}(m)\prod_{i=1}^{m}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k_i)\\ &= \sum_{m=a_n+1}^{\infty}\pi_{\text{M}}(m)\left(\prod_{i=1}^{a_n}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k_i)\right)\sum_{t_{a_n+1}=n}^{\infty}\pi_{\text{T}}(t_{a_n+1}|t_{a_n+1} > t_{a_n})\\ &\cdot\sum_{t_{a_n+2},\ldots,t_m}\left(\prod_{i=a_n+2}^{m}\pi_{\text{T}}(t_i|t_i > t_{i-1})\right)\sum_{k_{a_n+1},\ldots,k_m}\prod_{i=a_n+1}^{m}\pi_{\text{K}}(k_i)\\ &= \pi_{\text{M}}(Z > a_n)\left(\prod_{i=1}^{a_n}\pi_{\text{T}}(t_i|t_i > t_{i-1})\pi_{\text{K}}(k_i)\right)\pi_{\text{T}}(t_{a_n+1} \geq n)\cdot 1\cdot 1.\end{aligned} \tag{5.50}$$

If $M_n(\mathbf{s}) = 1$, then there is only one $\mathbf{s}$ that matches $E^n$, which has probability

$$\pi(E^n) = \pi_{\text{M}}(Z = a_n) \prod_{i=1}^{a_n} \pi_{\text{T}}(t_i | t_i > t_{i-1}) \pi_{\text{K}}(k_i). \tag{5.51}$$

From (5.50) and (5.51) we can compute the conditional probability $\pi(E_{n+1}|E^n)$. We distinguish further on the basis of the possible values of $S_{n+1}$ and $M_{n+1}$, which together determine $M_n$ (namely, if $M_{n+1} = 0$ then $M_n = 0$ and if $M_{n+1} = 1$ then $M_n = 1 - S_{n+1}$). Also note that $S_{n+1} = 0$ implies $a_{n+1} = a_n$ and $S_{n+1} = 1$ implies $a_{n+1} = a_n + 1$ and $t_{a_n+1} = n$. Conveniently, most factors cancel out, and we obtain

$$\begin{aligned}
&\pi(E_{n+1}|E^n) \\
&= \begin{cases}
\pi_{\text{T}}(t_{a_{n+1}} + 1 \geq n+1) / \pi_{\text{T}}(t_{a_n} \geq n) & \text{if } S_{n+1} = 0,\ M_{n+1} = 0 \\
1 & \text{if } S_{n+1} = 0,\ M_{n+1} = 1 \\
\frac{\pi_{\text{M}}(Z > a_n+1)}{\pi_{\text{M}}(Z > a_n)} \pi_{\text{T}}(t_{a_n+1} | t_{a_n+1} > t_{a_n}) \pi_{\text{K}}(k_{a_n+1}) \frac{\pi_{\text{T}}(t_{a_n+2} \geq n+1)}{\pi_{\text{T}}(t_{a_n+1} \geq n)} & \text{if } S_{n+1} = 1,\ M_{n+1} = 0 \\
\frac{\pi_{\text{M}}(Z = a_n+1)}{\pi_{\text{M}}(Z > a_n)} \frac{\pi_{\text{T}}(t_{a_n+1} | t_{a_n+1} > t_{a_n})}{\pi_{\text{T}}(t_{a_n+1} \geq n)} \pi_{\text{K}}(k_{a_n+1}) & \text{if } S_{n+1} = 1,\ M_{n+1} = 1,
\end{cases}
\end{aligned}$$

which reduces to (5.49). □

*Proof of Theorem 5.5.1.* We will use a number of independence properties of $P$ in this proof. First, we have that the distribution on $E_{n+1}$ is independent of $X^n$ conditional on $E^n$, because, using Bayes' rule,

$$\begin{aligned}
P(E_{n+1}|E^n, X^n) &= \frac{P(X^n|E^{n+1}) P(E_{n+1}|E^n)}{P(X^n|E^n)} \\
&= \frac{P(X^n|E^n) P(E_{n+1}|E^n)}{P(X^n|E^n)} = \pi(E_{n+1}|E^n),
\end{aligned} \tag{5.52}$$

provided that $P(E^{n+1}, X^n) > 0$. In turn, whether or not $E_{n+1}$ can occur depends only on $M_n$ and $K_n$. For all $n \geq 1$, define the function $N(M_n, K_n)$ as the set of values of the $E_{n+1}$ that have positive probability conditional on $M_n$ and $K_n$, i.e.

$$N(M_n, K_n) := \{(s, m, k) \mid \text{either } s = 1 \wedge M_n = 0 \text{ or } s = 0 \wedge m = M_n \wedge k = K_n\}. \tag{5.53}$$

Thus, the conditional distribution on $E_{n+1}$ given all previous values $E^n$ and all observations $X^n$ is a function of only $E_{n+1}$ itself, $n$, $M_n$ and $K_n$. It remains the same whether or not any of the other variables are included in the conditional. If $S_{n+1} = 1$ then it is not even a function of $K_n$. This interesting property is used three times in the following. Namely,

1. Since $(0, 0, k) \in N(0, k)$, we have $\pi_{\text{T}}(Z > n | Z \geq n) = P(E_{n+1} = (0, 0, k) \,|\, x^n, M_n = 0, K_n = k)$.

2. Since $(1,0,k) \in N(0,k')$ for all $k'$, we have $\pi_{\text{T}}(Z = n|Z \geq n)\pi_{\text{K}}(k)\theta = P(E_{n+1} = (1,0,k) \,|\, x^n, M_n = 0)$.

3. If $k \in \mathcal{K}_1$, then $\pi_{\text{K}}(k)\theta = P(x^0, M_1 = 0, K_1 = k)$.

We first show that the invariants $w_k^a = P(x^{n-1}, M_n = 0, K_n = k)$ and $w_k^b = P(x^{n-1}, M_n = 1, K_n = k)$ hold at the start of each iteration (before line 3). The invariants ensure that $w_k^a + w_k^b = P(x^{n-1}, K_n = k)$ so that the correct probabilities are reported.

Line 1 initialises $w_k^a$ to $\theta\pi_{\text{K}}(k)$ for $k \in \mathcal{K}_1$. By item 3 this equals $P(x^0, M_1 = 0, K_1 = k)$ as required. We omit calculations for $w_k^b$, which run along the same lines as for $w_k^a$. Thus the loop invariant holds at the start of the first iteration.

We proceed to go through the algorithm step by step to show that the invariant holds in subsequent iterations as well. In the loss update in line 4 we update the weights for $k \in \mathcal{K}_n$ to

$$\begin{aligned} w_k^a &= P(x^{n-1}, M_n = 0, K_n = k) \cdot p_k(x_n|x^{n-1}) \\ &= \sum_{\mathbf{s}: M_n = 0, K_n = k} \pi(\mathbf{s}) \left( \prod_{i=1}^{n-1} p_{K_i}(x_i|x^{i-1}) \right) p_{K_n}(x_n|x^{n-1}) \;=\; P(x^n, M_n = 0, K_n = k). \end{aligned}$$

Similarly $w_k^b = P(x^n, M_n = 1, K_n = k)$. Then in line 5, we compute $\texttt{pool} = \pi_{\text{T}}(Z = n|Z \geq n) \sum_{k \in \mathcal{K}_n} P(x^n, M_n = 0, K_n = k) = \pi_{\text{T}}(Z = n|Z \geq n) P(x^n, M_n = 0)$.

Finally, after the loop that starts at line 6 and ends at line 9, we obtain for all $k \in \mathcal{K}_{n+1}$:

$$\begin{aligned} w_k^a &= P(x^n, M_n{=}0, K_n{=}k)\pi_{\text{T}}(Z{>}n|Z{\geq}n) + \pi_{\text{T}}(Z{=}n|Z{\geq}n)P(x^n, M_n{=}0)\pi_{\text{K}}(k)\theta \\ &= P(x^n, M_n = 0, K_n = k)P(S_{n+1} = 0, M_{n+1} = 0|x^n, M_n = 0, K_n = k) \\ &+\; P(x^n, M_n = 0)P(S_{n+1} = 1, M_{n+1} = 0, K_{n+1} = k|x^n, M_n = 0) \\ &= P(x^n, M_n = 0, K_n = k, S_{n+1} = 0, M_{n+1} = 0) \\ &+\; P(x^n, M_n = 0, S_{n+1} = 1, M_{n+1} = 0, K_{n+1} = k) \\ &= P(x^n, S_{n+1}{=}0, M_{n+1}{=}0, K_{n+1}{=}k) + P(x^n, S_{n+1}{=}1, M_{n+1}{=}0, K_{n+1}{=}k) \\ &= P(x^n, M_{n+1} = 0, K_{n+1} = k). \end{aligned}$$

Here we used items 1 and 2 in the second equality. Again, a similar derivation shows that $w_k^b = P(x^n, K_{n+1} = k, M_{n+1} = 1)$. These weights satisfy the invariant at the beginning of the next iteration; after the last iteration the final posterior is also correctly reported based on these weights. □

# Chapter 6

# Individual Sequence Rate Distortion

Rate-distortion theory analyses communication over a channel under a constraint on the number of transmitted bits, the "rate". It currently serves as the theoretical underpinning for many important applications such as lossy compression and denoising, or more generally, applications that require a separation of structure and noise in the input data.

Classical rate-distortion theory evolved from Shannon's theory of communication [79]. It studies the trade-off between the rate and the achievable fidelity of the transmitted representation under some distortion function, where the analysis is carried out *in expectation* under some source distribution. Therefore the theory can only be meaningfully applied if we have some reasonable idea as to the distribution on objects that we want to compress lossily. While lossy compression is ubiquitous, propositions with regard to the underlying distribution tend to be ad-hoc, and necessarily so, because (1) it is a questionable assumption that the objects that we submit to lossy compression are all drawn from the same probability distribution, or indeed that they are drawn from a distribution at all, and (2) even if a true source distribution is known to exist, in most applications the sample space is so large that it is extremely hard to determine what it is like: objects that occur in practice very often exhibit more structure than predicted by the used source model.

For large outcome spaces then, it becomes important to consider structural properties of *individual objects*. For example, if the rate is low, then we may still be able to transmit objects that have a very regular structure without introducing any distortion, but this becomes impossible for objects with high information density. This point of view underlies some recent research in the lossy compression community [77]. At about the same time, a rate-distortion theory which allows analysis of individual objects has been developed within the framework of Kolmogorov complexity [60]. It defines a rate-distortion function not with respect to a source distribution, but with respect to an individual source word. Every source word thus obtains its own associated rate-distortion function.

We will first give a brief introduction to algorithmic rate-distortion theory in Section 6.1. We also describe a novel generalisation of the theory to settings with side information, and we describe two distinct applications of the theory, namely lossy compression and denoising.

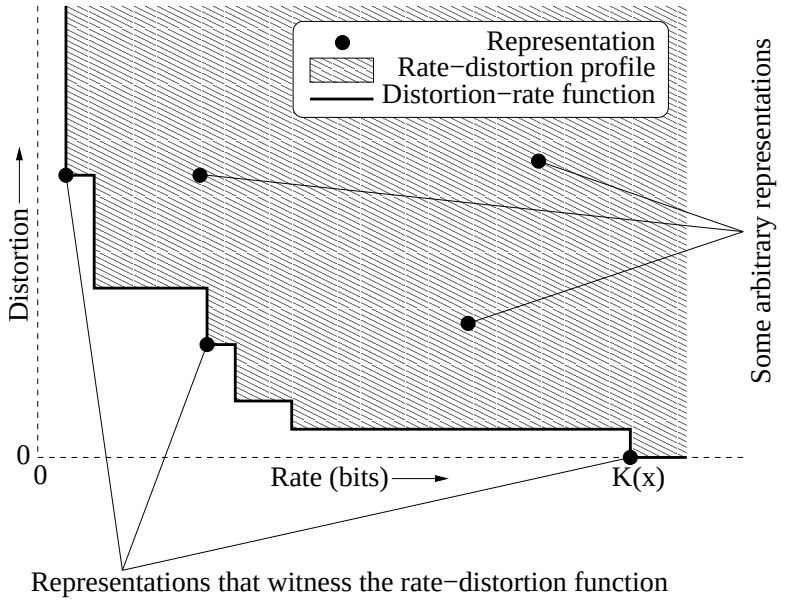
Algorithmic rate-distortion theory is based on Kolmogorov complexity, which is not computable. We nevertheless cross the bridge between theory and practice in Section 6.2, by approximating Kolmogorov complexity by the compressed size of the object by a general purpose data compression algorithm. Even so, approximating the rate-distortion function is a difficult search problem. We motivate and outline the genetic algorithm we used to approximate the rate-distortion function.

In Section 6.3 we describe four experiments in lossy compression and denoising. The results are presented and discussed in Section 6.4. Then, in Section 6.5 we take a step back and discuss to what extent our practical approach yields a faithful approximation of the theoretical algorithmic rate-distortion function, continued by a discussion of how such a practical approach fits within the framework of MDL model selection (Section 6.6). We end with a conclusion in Section 6.7.

## 6.1 Algorithmic Rate-Distortion

Suppose we want to communicate objects $x$ from a set of source words $\mathcal{X}$ using at most $r$ bits per object. We call $r$ the *rate*. We locate a good *representation* of $x$ within a finite set $\mathcal{Y}$, which may be different from $\mathcal{X}$ in general (but we usually have $\mathcal{X} = \mathcal{Y}$ in this text). The lack of fidelity of a representation $y$ is quantified by a distortion function $d : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.

**Figure 6.1** Rate-distortion profile and distortion-rate function



The Kolmogorov complexity of $y$, denoted $K(y)$, is the length of the shortest program that constructs $y$. More precisely, it is the length of the shortest input to a fixed universal binary prefix machine that will output $y$ and then halt; also see the textbook [60]. We can transmit any representation $y$ that has $K(y) \leq r$, the receiver can then run the program to obtain $y$ and is thus able to reconstruct $x$ up to distortion $d(x, y)$. Define the *rate-distortion profile* $P_x$ *of the source word* $x$ as the set of pairs $\langle r, a \rangle$ such that there is a representation $y \in \mathcal{Y}$ with $d(x, y) \leq a$ and $K(y) \leq r$. The possible combinations of $r$ and $a$ can also be characterised by the *rate-distortion function of the source word* $x$, which is defined as $r_x(a) = \min\{r : \langle r, a \rangle \in P_x\}$, or by the *distortion-rate function of the source word* $x$, which is defined as $d_x(r) = \min\{a : \langle r, a \rangle \in P_x\}$. These two functions are somewhat like inverses of each other; although strictly speaking they are not since they are monotonic but not strictly monotonic. A representation $y$ is said to *witness* the rate-distortion function of $x$ if $r_x(d(x, y)) = K(y)$. These definitions are illustrated in Figure 6.1.

Algorithmic rate-distortion theory is developed and treated in much more detail in [92]. It is a generalisation of Kolmogorov's structure function theory, see [93]. We generalise the algorithmic rate-distortion framework, so that it can accommodate side information. Suppose that we want to transmit a source word $x \in \mathcal{X}$ and we have chosen a representation $y \in \mathcal{Y}$ as before. The encoder and decoder often share a lot of information: both might know that grass is green and the sky is blue, they might share a common language, and so on. They would not need to transmit such information. If encoder and decoder share some information $z$, then the programs they transmit to compute the representation $y$ may use this side information $z$. Such programs can be much shorter, and are never much longer, than their counterparts that do not use side information. This can be formalised by switching to the *conditional* Kolmogorov complexity $K(y|z)$, which is the length of the shortest Turing machine program that constructs $y$ on input $z$. We redefine $K(y) = K(y|\epsilon)$, where $\epsilon$ is the empty sequence, so that $K(y|z) \leq K(y) + O(1)$: the length of the shortest program for $y$ can never significantly increase when side information is provided, but it might certainly decrease when $y$ and $z$ share a lot of information [60]. We change the definitions as follows: The *rate-distortion profile of the source word* $x$ *with side information* $z$ is the set of pairs $\langle r, a \rangle$ such that there is a representation $y \in \mathcal{Y}$ with $d(x, y) \leq a$ and $K(y|z) \leq r$. The definitions of the rate-distortion function and the distortion-rate function are similarly changed. Henceforth we will omit mention of the side information $z$ unless it is relevant to the discussion.

While this generalisation seems very natural the authors are not aware of earlier proposals along these lines. In Section 6.4 we will demonstrate one use for this generalised rate-distortion theory: removal of spelling errors in written text, an example where denoising is not practical without use of side information.

### 6.1.1 Distortion Spheres, the Minimal Sufficient Statistic

A representation $y$ that witnesses the rate-distortion function is the best possible rendering of the source object $x$ at the given rate because it minimises the distortion, but if the rate is lower than $K(x)$, then some information is necessarily lost. Since one of our goals is to find the best possible separation between structure and noise in the data, it is important to determine to what extent the discarded information is noise.

Given a representation $y$ and the distortion $a = d(x, y)$, we can find the source object $x$ somewhere on the list of all $x' \in \mathcal{X}$ that satisfy $d(x', y) = a$. The information conveyed about $x$ by $y$ and $a$ is precisely, that $x$ can be found on this list. We call such a list a *distortion sphere*. A distortion sphere of radius $a$, centred around $y$ is defined as follows:

$$S_y(a) := \{x' \in \mathcal{X} : d(x', y) = a\}. \tag{6.1}$$

If $x$ is a completely random element of this list, then the discarded information is pure "white" noise. Moreover, all random elements in the list share all "simply described" (in the sense of having low Kolmogorov complexity) properties that $x$ satisfies. Hence, with respect to the "simply described" properties, every such random element is as good as $x$, see [92] for more details. In such cases a literal specification of the index of any object $x'$ in the list (in particular the original object $x$) is the most efficient code for that $x'$, given only that it is in $S_y(a)$. A fixed-length, literal code requires $\log |S_y(a)|$ bits. (Here and in the following, all logarithms are taken to base 2 unless otherwise indicated.) On the other hand, if the discarded information is structured, then the Kolmogorov complexity of the index of $x$ in $S_y(a)$ will be significantly lower than the logarithm of the size of the sphere. The difference between these two code lengths can be used as an indicator of the amount of structural information that is discarded by the representation $y$. Vereshchagin and Vitányi [92] call this quantity the *randomness deficiency* of the source object $x$ in the set $S_y(a)$, and they show that if $y$ witnesses the rate-distortion function of $x$, then it *minimises* the randomness deficiency at rate $K(y)$; thus the rate-distortion function identifies those representations that account for as much structure as possible at the given rate.

To assess how much structure is being discarded at a given rate, consider a code for the source object $x$ in which we first transmit the shortest possible program that constructs both a representation $y$ and the distortion $d(x, y)$, followed by a literal, fixed-length index of $x$ in the distortion sphere $S_y(a)$. Such a code has length function

$$K(y, d(x, y)) + L_y(x), \text{ where } \quad L_y(x) := \log |S_y(d(x, y))|. \tag{6.2}$$

If the rate is very low then the representation $y$ models only very basic structure and the randomness deficiency in the distortion sphere around $y$ is high. Borrowing terminology from statistics, we may say that $y$ is a representation that "underfits" the data. In such cases we should find that $K(y, d(x, y)) + L_y(x) > K(x)$,

because the fixed-length code for the index of $x$ within the distortion sphere is suboptimal in this case. But suppose that $y$ is complex enough that it satisfies $K(y, d(x, y)) + L_y(x) \approx K(x)$. In [92], such representations are called *(algorithmic) sufficient statistics* for the data $x$. A sufficient statistic has close to zero randomness deficiency, which means that it represents all structure that can be detected in the data. However, sufficient statistics might contain not only structure, but noise as well. Such a representation would be overly complex, an example of overfitting. A *minimal* sufficient statistic balances between underfitting and overfitting. It is defined as the lowest complexity sufficient statistic, in other words the lowest complexity representation $y$ that minimises the total code length. As such it can also be regarded as the "model" that should be selected on the basis of the Minimum Description Length (MDL) principle [4]. For a further discussion of this relationship see Section 6.6. To be able to relate the distortion-rate function to this code length we define the *code length function* $\lambda_x(r) = K(y, d(x, y)) + L_y(x)$ where $y$ is the representation that minimises the distortion at rate $r$.[1]

### 6.1.2 Applications: Denoising and Lossy Compression

Representations that witness the rate-distortion function provide optimal separation between structure that can be expressed at the given rate and residual information that is perceived as noise. Therefore, these representations can be interpreted as denoised versions of the original. In denoising, the goal is of course to discard as much noise as possible, without losing any structure. Therefore the minimal sufficient statistic, which was described in the previous section, is the best candidate for applications of denoising.

While the minimal sufficient statistic is a denoised representation of the original signal, it is not necessarily given in a directly usable form. For instance, $\mathcal{Y}$ could consist of subsets of $\mathcal{X}$, but a *set* of source-words is not always acceptable as a denoising result. So in general one may need to apply some function $f : \mathcal{Y} \to \mathcal{X}$ to the sufficient statistic to construct a usable object. But if $\mathcal{X} = \mathcal{Y}$ and the distortion function is a metric, as in our case, then the representations are already in an acceptable format, so here we use the identity function for the transformation $f$.

In applications of lossy compression, one may be willing to accept a rate which is lower than the minimal sufficient statistic complexity, thereby losing some structural information. However, for a minimal sufficient statistic $y$, theory does tell us that it is not worthwhile to set the rate to a higher value than the complexity of $y$. The original object $x$ is a random element of $S_y(d(x, y))$, and it cannot be distinguished from any other random $z \in S_y(d(x, y))$ using only

[1] This is superficially similar to the MDL function defined in [93], but it is *not* exactly the same since it involves optimisation of the distortion at a given rate rather than direct optimisation of the code length.

"simply described" properties. So we have no "simply described" test to discredit the hypothesis that $x$ (or any such $z$) is the original object, given $y$ and $d(x,y)$. If we increase the rate and find a model $y'$ with $d(x,y') < d(x,y)$, then commonly the cardinality of $S_{y'}$ is smaller than that of $S_y$, such that some elements of $S_y$ are not included in $S_{y'}$. These excluded elements, however, were perfectly good candidates of being the original object. That is, at rate higher than that of the minimal sufficient statistic, the resulting representation $y'$ models irrelevant features that are specific to $x$, that is, noise and no structure, that exclude viable candidates for the original object: the representation starts to "overfit".

In lossy compression, as in denoising, the representations themselves may be unsuitable for presentation to the user. For example, when decompressing a lossily compressed image, in most applications a *set* of images would not be an acceptable result. So again a transformation from representations to objects of a usable form has to be specified. There are two obvious ways of doing this:

1. If a representation $y$ witnesses the rate-distortion function for a source word $x \in \mathcal{X}$, then this means that $x$ cannot be distinguished from any other object $x' \in S_y(d(x,y))$ at rate $K(y)$. Therefore we should not use a deterministic transformation, but rather report the uniform distribution on $S_y(d(x,y))$ as the lossily compressed version of $x$. This method has the advantage that it is applicable whether or not $\mathcal{X} = \mathcal{Y}$.

2. On the other hand, if $\mathcal{X} = \mathcal{Y}$ and the distortion function is a metric, then it makes sense to use the identity transformation again, although here the motivation is different. Suppose we select some $x' \in S_y(d(x,y))$ instead of $y$. Then the best upper bound we can give on the distortion is $d(x,x') \leq d(x,y) + d(y,x') = 2d(x,y)$ (by the triangle inequality and symmetry). On the other hand if we select $y$, then the distortion is exactly $d(x,y)$, which is only half of the upper bound we obtained for $x'$. Therefore it is more suitable if one adopts a worst-case approach. This method has as an additional advantage that the decoder does not need to *know* the distortion $d(x,y)$ which often cannot be computed from $y$ without knowledge of $x$.

To illustrate the difference one may expect from these approaches, consider the situation where the rate is lower than the rate that would be required to specify a sufficient statistic. Then intuitively, all the noise in the source word $x$ as well as some of the structure are lost by compressing it to a representation $y$. The second method immediately reports $y$, which contains a lot less noise than the source object $x$; thus $x$ and $y$ are qualitatively different, which may be undesirable. On the other hand, the compression result will be qualitatively different from $x$ anyway, because the rate simply is too low to retain all structure. If one would apply the first approach, then a result $x'$ would likely contain *more* noise than the original, because it contains less structure at the same level of distortion (meaning that $K(x') > K(x)$ while $d(x',y) = d(x,y)$).

If the rate is high enough to transmit a sufficient statistic, then the first approach seems preferable. We have nevertheless chosen to always report $y$ directly in our analysis, which has the advantage that this way, all reported results are of the same type.

## 6.2 Computing Individual Object Rate-Distortion

The rate-distortion function for an object $x$ with side information $z$ and a distortion function $d$ is found by simultaneously minimising two objective functions

$$\begin{aligned} g_1(y) &= K(y|z), \\ g_2(y) &= d(x,y), \\ g(y) &= \left\langle g_1(y), g_2(y) \right\rangle. \end{aligned} \tag{6.3}$$

We call the tuple $g(y)$ the *trade-off* of $y$. We impose a partial order on representations:

$$y \preccurlyeq y' \qquad \text{if and only if} \qquad g_1(y) \leq g_1(y') \text{ and } g_2(y) \leq g_2(y'). \tag{6.4}$$

Our goal is to find the set of representations that are minimal under $\preccurlyeq$.

Such an optimisation problem cannot be implemented because of the uncomputability of $K(\cdot)$. To make the idea practical, we need to approximate the conditional Kolmogorov complexity. As observed in [20], it follows directly from symmetry of information for Kolmogorov complexity (see [60, p.233]) that:

$$K(y|z) = K(zy) - K(z) + O(\log n), \tag{6.5}$$

where $n$ is the length of $zy$. Ignoring the logarithmic term, this quantity can be approximated by replacing $K(\cdot)$ by $\tilde{K}(\cdot)$, the length of the compressed representation under a general purpose compression algorithm. The approximate conditional complexity then becomes

$$\begin{aligned} \tilde{K}(y|z) &:= \tilde{K}(zy) - \tilde{K}(z) \\ &\approx K(zy) - K(z) \quad = \quad K(y|z) + O(\log n). \end{aligned} \tag{6.6}$$

This may be a poor approximation: it is an upper bound that may be quite high even for objects that have conditional Kolmogorov complexity close to zero. Our results show evidence that some of the theoretical properties of the distortion-rate function nevertheless carry over to the practical setting; we also explain how some observations that are not predicted by theory are in fact related to the (unavoidable) inefficiencies of the used compressor.

### 6.2.1 Compressor (rate function)

We could have used any general-purpose compressor in (6.6), but we chose to implement our own for three reasons:

- It should be both fast and efficient. We can gain some advantage over other available compressors because there is no need to actually construct a code. It suffices to compute code *lengths*, which is much easier. As a secondary advantage, the code lengths we compute are not necessarily multiples of eight bits: we allow rational idealised code lengths, which may improve precision.
- It should not have any arbitrary restrictions or optimisations. Most general purpose compressors have limited window sizes or optimisations to improve compression of common file types; such features could make the results harder to interpret.

In our experiments we used a block sorting compression algorithm with a move-to-front scheme as described in [17]. In the encoding stage M2 we employ a simple statistical model and omit the actual encoding as it suffices to accumulate code lengths. The source code of our implementation (in C) is available from the authors upon request. The resulting algorithm is very similar to a number of common general purpose compressors, such the freely available bzip2 and zzip (see [82]), but it is simpler and faster for small inputs.

Of course, domain specific compressors might yield better compression for some object types (such as sound wave files), and therefore a better approximation of the Kolmogorov complexity. However, the compressor that we implemented is quite efficient for objects of many of the types that occur in practice; in particular it compressed the objects of our experiments (text and small images) quite well. We have tried to improve compression performance by applying standard image preprocessing algorithms to the images, but this turned out not to improve compression at all. Figure 6.1 lists the compressed size of an image of a mouse under various different compression and filtering regimes. Compared to other compressors, ours is quite efficient; this is probably because other compressors are optimised for larger files and because we avoid all overhead inherent in the encoding process. Most compressors have optimisations for text files which might explain why our compressor compares less favourably on the Oscar Wilde fragment.

### 6.2.2 Code Length Function

In Section 6.1.1 we introduced the code length function $\lambda_x(r)$. Its definition makes use of (6.2), for which we have not yet provided a computable alternative. We use the following approximation:

$$K(y, d(x, y)) \approx \tilde{K}(y) + L_D(d(x, y)|y), \tag{6.7}$$

**Table 6.1** Compressed sizes of three objects that we experiment upon. See Figure 6.4(h) for the mouse, Figure 6.6 for the cross with added noise and Figure 6.10 for the corrupted Oscar Wilde fragment (the middle version). In the latter we give the compressed size *conditional* on a training text, like in the experiments. "A" is our own algorithm, described in Section 6.2.1. For a description of the filters see [75].

| Compression | mouse | cross | Wilde | description |
|---|---|---|---|---|
| A | 7995.11 | 3178.63 | 3234.45 | Our compressor, described in §6.2.1 |
| `zzip` | 8128.00 | 3344.00 | 3184.00 | An efficient block sorting compressor |
| `PPMd` | 8232.00 | 2896.00 | 2744.00 | High end statistical compressor |
| RLE → A | 8341.68 | 3409.22 | – | A with run length encoding filter |
| `bzip2` | 9296.00 | 3912.00 | 3488.00 | Widespread block sorting compressor |
| `gzip` | 9944.00 | 4008.00 | 3016.00 | LZ77 compressor |
| sub → A | 10796.29 | 4024.26 | – | A with Sub filter |
| paeth → A | 13289.34 | 5672.70 | – | A with Paeth filter |
| None | 20480.00 | 4096.00 | 5864.00 | Literal description |

where $L_D$ is yet another code which is necessary to specify the radius of the distortion sphere around $y$ in which $x$ can be found. It is possible that this distortion is uniquely determined by $y$, for example if $\mathcal{Y}$ is the set of all finite subsets of $\mathcal{X}$ and list decoding distortion is used, as described in [93]. If $d(x, y)$ is a function of $y$ then $L_D(d(x, y)|y) = 0$. In other cases, the representations do not hold sufficient information to determine the distortion. This is typically the case when $\mathcal{X} = \mathcal{Y}$ as in the examples in this text. In that case we actually need to encode $d(x, y)$ separately. It turns out that the number of bits that are required to specify the distortion are negligible in proportion to the total three part code length. In the remainder of the chapter we use for $L_D$ a universal code on the integers similar to the one described in [60]; it has code length $L_D(d) = \log(d + 1) + O(\log \log d)$.

We also need to calculate the size of the distortion sphere, which we therefore calculate for each of the distortion functions that we describe below.

### 6.2.3 Distortion Functions

We use three common distortion functions. All distortion functions used in this text are metrics, so they can only be used when $\mathcal{X} = \mathcal{Y}$.

**Hamming distortion** Hamming distortion is perhaps the simplest possible distortion function. It can be defined when $\mathcal{X} = \mathcal{Y} = \Sigma^n$, sequences of $n$ symbols from a finite alphabet $\Sigma$. Let $x$ and $y$ be two objects of equal length $n$. The Hamming distortion $d(x, y)$ is equal to the number of symbols in $x$ that do not match those in the corresponding positions in $y$.

A Hamming-distortion sphere $S_y(a)$ contains all objects of length $n$ that can be constructed by replacing $a$ symbols in $y$ with different symbols from the alphabet $\Sigma$. Thus the size of the sphere is $\binom{n}{a}(|\Sigma|-1)^a$.

**Euclidean distortion** As before, let $x = x_1 \dots x_n$ and $y = y_1 \dots y_n$ be two objects of equal length, but the symbols now have a numerical interpretation. Euclidean distortion is $d(x,y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$: the distance between $x$ and $y$ when they are interpreted as vectors in an $n$-dimensional Euclidean space. Note that this definition of Euclidean distortion differs from the one in [92].

Our variety of Euclidean distortion requires that $\mathcal{X} = \mathcal{Y} = \mathbb{Z}^n$, the set of $n$-dimensional vectors of integers. The size of a Euclidean distortion sphere around some $y \in \mathcal{Y}$ of length $n$ is hard to compute analytically. We use an upper bound that is reasonably tight and can be computed efficiently. One may want to skip this calculation on first reading.

First we define $d(v) := d(v, \mathbf{0}) = \sqrt{\sum_{i=1}^n v_i^2}$ and $S(n,a)$ as the set $\{v : |v| = n, d(v) = a\}$. We have $x \in S_y(a) \Leftrightarrow x - y \in S(n,a)$, so it suffices to bound the size of $S(n,a)$. We define:

$$p(\delta|n,a) := ce^{-\delta^2 n/2a^2} \text{ where } c = 1/\sum_{\delta\in\mathbb{Z}} e^{-\delta^2 n/2a^2}$$

$$P(v) := \prod_{i=1}^n p(v_i|n, d(v));$$

$p(\cdot|n,d(v))$ can be interpreted as a probability mass function on the individual entries of $v$ (which in our application always lie between -255 and 255, so in practice we used a reduced range in the definition of the normalising constant $c$). Therefore $P(v)$ defines a valid probability mass function on outcomes $v$ in $S(n,a)$. Thus,

$$1 > \sum_{v\in S(n,a)} P(v) = \sum_{v\in S(n,a)} c^n e^{-(\sum \delta_i^2)n/2a^2} = \sum_{v\in S(n,a)} c^n e^{-n/2} = c^n e^{-n/2}|S(n,a)|.$$

This yields a bound on the size of $S(n,a)$, which is reasonably tight unless the distortion $a$ is very low. In that case, we can improve the bound by observing that $v$ must have at least $z = n - d(v)^2$ zero entries. Let $v'$ be a vector of length $n - z$ that is obtained by removing $z$ zero entries from $v$. Every $v$ in $S(n,a)$ can be constructed by inserting $z$ zeroes into $v'$, so we have $|S_y(a)| = |S(n,a)| \leq \binom{n}{z}|S(n-z,a)|$. The size of $S(n-z,a)$ can be bounded by using the method described before recursively.

**Edit distortion** The edit distortion of two strings $x$ and $y$, of possibly different lengths, is the minimum number of symbols that have to be deleted from,

inserted into, or changed in $x$ in order to obtain $y$ (or vice versa) [54]. It is also known as *Levenshtein distortion*. It is a well-known measure that is often used in applications that require approximate string matching.

Edit distortion can be defined for spaces $\mathcal{X} = \mathcal{Y} = \Sigma^*$ for a finite alphabet $\Sigma$. We develop an upper bound on the size of the edit distortion sphere, again the calculation may be skipped on first reading.

We can identify any object in $S_y(a)$ by a program $p$ that operates on $y$, and which is defined by a list of instructions to copy, replace or delete the next symbol from $y$, or to insert a new symbol. We interpret a deletion as a replacement with an empty symbol; so the replacement operations henceforth include deletions. Let $d(p)$ denote the number of insertions and replacements in $p$, in other words $d(p)$ is the length of the program minus the number of copies. Clearly for all $x \in S_y(a)$, there must be a $p$ such that $p(y) = x$ and $d(p) = d(x, y) = a$. Therefore the size of $S_y(a)$ can be upper bounded by counting the number of programs with $d(p) = a$. Let $n$ be the length of $y$. Any program that contains $i$ insertions and that processes $y$ completely, must be of length $n + i$. The $i$ insertions, $a - i$ replacements and $n - a + i$ copies can be distributed over the program in $\binom{n+i}{i,a-i,n+a-i}$ different ways. For each insertion and replacement, the number of possibilities is equal to the alphabet size. Therefore,

$$|S_y(a)| \quad \leq \quad |\{p : d(p) = a\}| \quad \leq \quad |\Sigma|^a \sum_{i=\max\{0,a-n\}}^{a} \binom{n+i}{i, n-a+i, a-i}.$$

The sphere can be extremely large, so to facilitate calculation of the log of the sphere size, as is required in our application, it is convenient to relax the bound some more and replace every term in the sum by the largest one. Calculation reveals that the largest term has

$$i = \left\lfloor \frac{1}{4} \left( 2(a-n) + 1 + \sqrt{4(n^2 + n + a^2 + a) + 1} \right) \right\rfloor.$$

### 6.2.4 Searching for the Rate-Distortion Function

The search problem that we propose to address has two properties that make it very hard. Firstly, the search space is enormous: at rate $r$ there are $2^r$ candidate representations to consider, and for the kinds of objects that are typically subjected to lossy compression useful representations are often millions or billions of bits long. Secondly, we want to avoid making too many assumptions about the two objective functions, so that we can later freely change the compression algorithm and the distortion function. Under such circumstances the two most obvious search methods are not practical:

- An exhaustive search is infeasible for search spaces of such large size, unless more specific properties of the objective functions are used in the design

of the algorithm. To investigate how far we could take such an approach, we have implemented an exhaustive algorithm under the requirement that, given a prefix of a representation $y$, we can compute reasonable lower bounds on the values of both objective functions $g_1$ and $g_2$. This allows for relatively efficient enumeration of all representations of which the objective functions do not exceed specific maxima: it is never necessary to consider objects which have a prefix for which the lower bounds exceed the constraints, which allows for significant pruning. In this fashion we were able to find the rate-distortion function under Hamming distortion for objects of which the compressed size is about 25 bits or less within a few hours on a desk-top computer.

- A greedy search starts with a poor solution and iteratively makes modifications that constitute strict improvements. We found that this procedure tends to terminate quickly in some local optimum that is very bad globally.

Since the structure of the search landscape is at present poorly understood and we do not want to make any unjustifiable assumptions, we use a genetic search algorithm which performs well enough that interesting results can be obtained.

### 6.2.5 Genetic Algorithm

The used algorithm is an almost completely generic procedure to simultaneously optimise two separate objective functions for objects that are represented as byte sequences. To emphasise this we will consider the abstract objective function $g$ wherever possible, rather than the more concrete rate and distortion functions.

A finite subset of $\mathcal{Y}$ is called a *pool*. The search algorithm initialises a pool $\mathcal{P}_0$ with the representation $y$ that has $d(x, y) = 0$, which means $y = x$ in our setup, and possibly some "blank" representation that has minimal compressed code length but high distortion. The pool is then subjected to a process of selection through survival of the fittest. The *weakness* $w_{\mathcal{P}}(y)$ of an object $y \in \mathcal{P}$ is the number of elements of the pool that are smaller according to $\preccurlyeq$. The *(transitive) reduction* $\mathrm{trd}(\mathcal{P})$ of a pool $\mathcal{P}$ is the subset of all elements with zero weakness. The elements of the reduction of a pool $\mathcal{P}$ are called *models*.

The pool is iteratively updated by replacing elements with high weakness (the fitness function is specified below) by new ones, which are created through either *mutation* (random modifications of elements) or *crossover* ("genetic" recombination of pairs of other candidates). We write $\mathcal{P}_i$ to denote the pool after $i$ iterations. When the algorithm terminates after $n$ iterations it outputs the reduction of $\mathcal{P}_n$.

In the next sections we describe our choices for the important components of the algorithm: the mechanics of crossover and mutation, the fitness function and the selection function which specifies the probability that a candidate is removed from the pool. In the interest of reproducibility we faithfully describe all our

important design choices, even though some of them are somewhat arbitrary. A casual reader may want to skip such details and move on to Section 6.3.

### Crossover

Crossover (also called recombination) is effected by the following algorithm. Given two objects $x$ and $y$ we first split them both in three parts: $x = x_1x_2x_3$ and $y = y_1y_2y_3$, such that the length of $x_1$ is chosen uniformly at random between 0 and the length of $x$ and the length of $x_2$ is chosen from a geometric distribution with mean 5; the lengths of the $y_i$ are proportional to the lengths of the $x_i$. We then construct a new object by concatenating $x_1y_2x_3$.

### Mutation

The introduction of a mutation operation is necessary to ensure that the search space is connected, since the closure of the gene pool under crossover alone might not cover the entire search space. While we could have used any generic mutation function that meets this requirement, for reasons of efficiency we have decided to design a different mutation function for every objective function that we implemented. This is helpful because some distortion functions (here, the edit distortion) can compare objects of different sizes while others cannot: mutation is the means by which introduction of objects of different size to the pool can be brought about when desirable, or avoided when undesirable.

The mutation algorithm we use can make two kinds of change. With probability 1/4 we make a small random modification using an algorithm that depends on the distortion function. Below is a table of the distortion functions and a short description of the associated mutation algorithms:

| Distortion | Mutation algorithm |
|---|---|
| Hamming | Sets a random byte to a uniformly random value |
| Euclidean | Adds an $\mathcal{N}[0; \sigma = 10]$ value to a random byte |
| Edit | A random byte is changed, inserted or deleted |

With probability 3/4 we use the following mutation algorithm instead. It splits the object $x$ into three parts $x = x_1x_2x_3$ where the length of $x_1$ is chosen uniformly at random between 0 and the length of $x$ and the length of $x_2$ is chosen from a geometric distribution with mean 5. The mutation is effected by training a (simplified version of) a third order PPM model [24] on $x_1$ and then replacing $x_2$ with an equally long sequence that is sampled from the model. The advantage of this scheme is that every replacement for $x_2$ gets positive probability, but replacements which have low code length and distortion tend to be much more likely than under a uniform distribution.

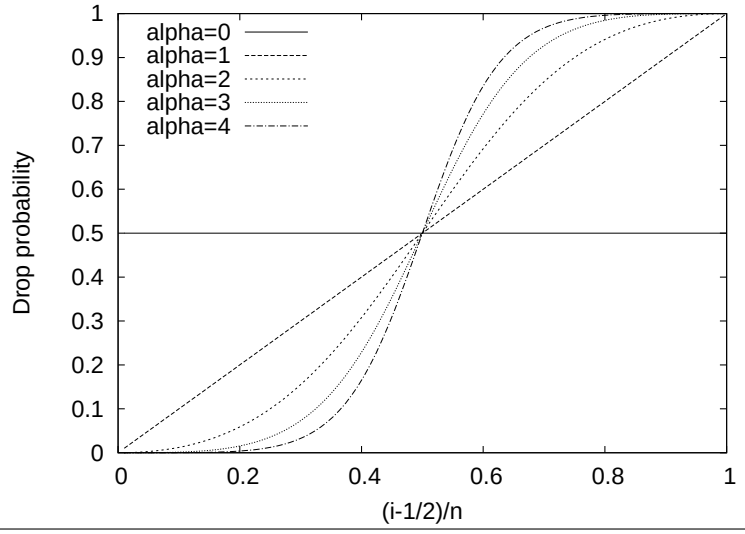
### Fitness Function

In theory, the set of representations that witness the rate-distortion function does not change under monotonic transformation of either objective function $g_1$ or $g_2$. We have tried to maintain this property throughout the search algorithm including the fitness function, by never using either objective function directly but only the ordering relation $\preccurlyeq$. Under such a regime, a very natural definition of fitness is minus the weakness of the objects with respect to pool $\mathcal{P}$.

It has been an interesting mini-puzzle to come up with an efficient algorithm to compute the weakness of all objects in the pool efficiently. Our solution is the following very simple algorithm, which has an $O(n \log n)$ average case running time. It first sorts all elements of the pool by their value under $g_1$ and then inserts them in order into a binary search tree in which the elements are ordered by their value under $g_2$. As an object is inserted into the tree, we can efficiently count how many elements with lower values for $g_2$ the tree already contained. These elements are precisely the objects that have both lower values on $g_1$ (otherwise they would not appear in the tree yet) and on $g_2$; as such their number is the desired weakness.

### Selection Function

**Figure 6.2** Drop probability




A pool $\mathcal{P}$ induces a *tradeoff profile* $p(\mathcal{P}) := \{g(y) : y' \preccurlyeq y \text{ for some } y' \text{ in } \mathcal{P}\}$. It is not hard to see that we have $p(\mathcal{P}) = p(\mathrm{trd}(\mathcal{P}))$ and $p(\mathcal{P}) \subseteq p(\mathcal{P} \cup \mathcal{P}')$ for all $\mathcal{P}' \subseteq \mathcal{Y}$. Therefore monotonic improvement of the pool under modification is ensured as long as candidates with weakness 0 are never dropped.

We drop other candidates with positive probability as follows. Let $y_1, \ldots, y_n$ be the elements of $\mathcal{P}$ with nonzero weakness, ordered such that for $1 \leq i < j \leq n$

we have $w_{\mathcal{P}}(y_i) < w_{\mathcal{P}}(y_j)$ or $g_1(y_i) < g_1(y_j)$ if $y_i$ and $y_j$ have the same weakness. We drop candidate $y_i$ from the pool with probability $1/(1 + (\frac{n}{i-1/2} - 1)^{\alpha})$, which is a modified sigmoid function where $\alpha \in (1, \infty)$ specifies the sharpness of the transition from probability zero to one. This function is plotted for different values of $\alpha$ in Figure 6.2. We used $\alpha = 4$ in our experiments.

## 6.3 Experiments

We have subjected four objects to our program. The following considerations have influenced our choice of objects:

- Objects should not be too complex, allowing our program to find a good approximation of the distortion-rate curve. We found that the running time of the program seems to depend mostly on the complexity of the input object; a compressed size of 20,000 bits seemed to be about the maximum our program could handle within a reasonable amount of time, requiring a running time of the order of weeks on a desk-top computer.

- To check that our method really is general, objects should be quite different from each other: they should come from different object domains, for which different distortion functions are appropriate, and they should contain structure at different levels of complexity.

- Objects should contain primary structure and regularities that are distinguishable and compressible by a block sorting compressor such as the one we use. Otherwise, we may no longer hope that the compressor implements a reasonable approximation of the Kolmogorov complexity. For instance, we would not expect our program to do well on a sequence of digits from the binary expansion of the number $\pi$.

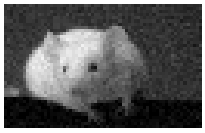
With this in mind, we have selected the objects listed in Figure 6.3.

In each experiment, as time progressed the program found less and less improvements per iteration, but the pool never stabilized completely. Therefore we interrupted each experiment when (a) after at least one night of computation, the pool did not improve a lot, and (b) for all intuitively good models $y \in \mathcal{Y}$ that we could conceive of a priori, the algorithm had found an $y'$ in the pool with $y' \preccurlyeq y$ according to (6.4). For example, in each denoising experiment, this test included the original, noiseless object. In the experiment on the mouse without added noise, we also included the images that can be obtained by reducing the number of grey levels in the original with an image manipulation program. Finally for the greyscale images we included a number of objects that can be obtained by subjecting the original object to JPEG2000 compression at various quality levels.

**Figure 6.3** The four objects that are subjected to rate-distortion analysis.

A picture of a mouse of $64 \times 40$ pixels. The picture is analysed with respect to Euclidean distortion.

A noisy monochrome image of $64 \times 64$ pixels that depicts a cross. 377 pixels have been inverted. Hamming distortion is used.



The same picture of a mouse, but now zero mean Gaussian noise with $\sigma = 8$ has been added to each pixel. Euclidean distortion is used; the distortion to the original mouse is 391.1.

Beauty, real beauty, ends2wheresan intellectual expressoon begins. IntellHct isg in itself a mMde ofSexggeration, an\ destroys theLharmony of n face. [...]

*(See Figure 6.10)*

A corrupted quotation from Chapter 1 of *The Picture of Dorian Gray*, by Oscar Wilde. The 733 byte long fragment was created by performing 68 random insertions, deletions and replacements of characters in the original text. Edit distortion is used. The rest of chapters one and two of the novel are given to the program as side information.

The first experiment illustrates how algorithmic rate-distortion theory may be applied to lossy compression problems, and it illustrates how for a given rate, some features of the image are preserved while others can no longer be retained. We compare the performance of our method to the performance of JPEG and JPEG2000 at various quality levels. Standard JPEG images were encoded using the ImageMagick version 6.2.2; profile information was stripped. JPEG2000 images were encoded to jpc format with three quality levels using NetPBM version 10.33.0; all other options are default. For more information about these software packages refer to [83].

The other three experiments are concerned with denoising. Any model that is output by the program can be interpreted as a denoised version of the input object. We measure the denoising success of a model $y$ as $d(x', y)$, where $x'$ is the original version of the input object $x$, before noise was added. We also compare the denoising results to those of other denoising algorithms:

1. BayesShrink denoising [18]. BayesShrink is a popular wavelet-based denoising method that is considered to work well for images.

2. Blurring (convolution with a Gaussian kernel). Blurring works like a low-pass filter, eliminating high frequency information such as noise. Unfortunately other high frequency features of the image, such as sharp contours, are also discarded.

3. Naive denoising. We applied a naive denoising algorithm to the noisy cross, in which each pixel was inverted if five or more out of the eight neighbouring

pixels were of different colour.

4. Denoising based on JPEG2000. Here we subjected the noisy input image to JPEG2000 compression at different quality levels. We then selected the result for which the distortion to the original image was lowest.

### 6.3.1 Names of Objects

To facilitate description and discussion of the experiments we will adopt the following naming convention. Objects related to the experiments with the mouse, the noisy cross, the noisy mouse and the Wilde fragment, are denoted by the symbols $\mathbb{M}$, $\mathbb{C}$, $\mathbb{N}$ and $\mathbb{W}$ respectively. A number of important objects in each experiment are identified by a subscript as follows. For $\mathbb{O} \in \{\mathbb{M}, \mathbb{C}, \mathbb{N}, \mathbb{W}\}$, the input object, for which the rate-distortion function is approximated by the program, is called $\mathbb{O}_{\text{IN}}$. In the denoising experiments, the input object is always constructed by adding noise to an original object. The original objects and the noise are called $\mathbb{O}_{\text{ORIG}}$ and $\mathbb{O}_{\text{NOISE}}$ respectively. If Hamming distortion is used, addition is carried out modulo 2, so that the input object is in effect a pixelwise exclusive OR of the original and the noise. In particular, $\mathbb{C}_{\text{IN}}$ equals $\mathbb{C}_{\text{ORIG}}$ XOR $\mathbb{C}_{\text{NOISE}}$. The program outputs the reduction of the gene pool, which is the set of considered models. Two important models are also given special names: the model within the gene pool that minimises the distortion to $\mathbb{O}_{\text{ORIG}}$ constitutes the best denoising of the input object and is therefore called $\mathbb{O}_{\text{BEST}}$, and the minimal sufficient statistic as described in Section 6.1.1 is called $\mathbb{O}_{\text{MSS}}$. Finally, in the denoising experiments we also give names to the results of the alternative denoising algorithms. Namely, $\mathbb{C}_{\text{NAIVE}}$ is the result of the naive denoising algorithm applied to the noisy cross, $\mathbb{N}_{\text{BLUR}}$ is the convolution of $\mathbb{N}$ with a Gaussian kernel with $\sigma = 0.458$, $\mathbb{N}_{\text{BS}}$ is the denoising result of the BayesShrink algorithm, and $\mathbb{N}_{\text{JPEG2000}}$ is the image produced by subjecting $\mathbb{N}$ to JPEG2000 compression at the quality level for which the distortion to $\mathbb{N}_{\text{ORIG}}$ is minimised.

## 6.4 Results and Discussion

After running for some time on each input object, our program outputs the reduction of a pool $\mathcal{P}$, which is interpreted as a set of models. For each experiment, we report a number of different properties of these sets. Since we are interested in the rate-distortion properties of the input object $x = \mathbb{O}_{\text{IN}}$, we plot the approximation of the distortion-rate function of each input object: $d_x(r) = \min\{d(x, y) : y \in \mathcal{Y}, K(y) \leq r\} \approx \min\{d(x, y) : y \in \text{trd}(\mathcal{P}), \tilde{K}(y) \leq r\}$. Such approximations of the distortion-rate function are provided for all four experiments. For the greyscale images we also plot the distortion-rate approximation that is achieved by JPEG2000 (and in Figure 6.5 also ordinary JPEG) at different quality levels. Here, the rate is the code length achieved by JPEG(2000), and the

distortion is the Euclidean distortion to $\mathbb{O}_{\text{IN}}$. We also plot the code length function as described in Section 6.1.1. Minimal sufficient statistics can be identified by locating the minimum of this graph.

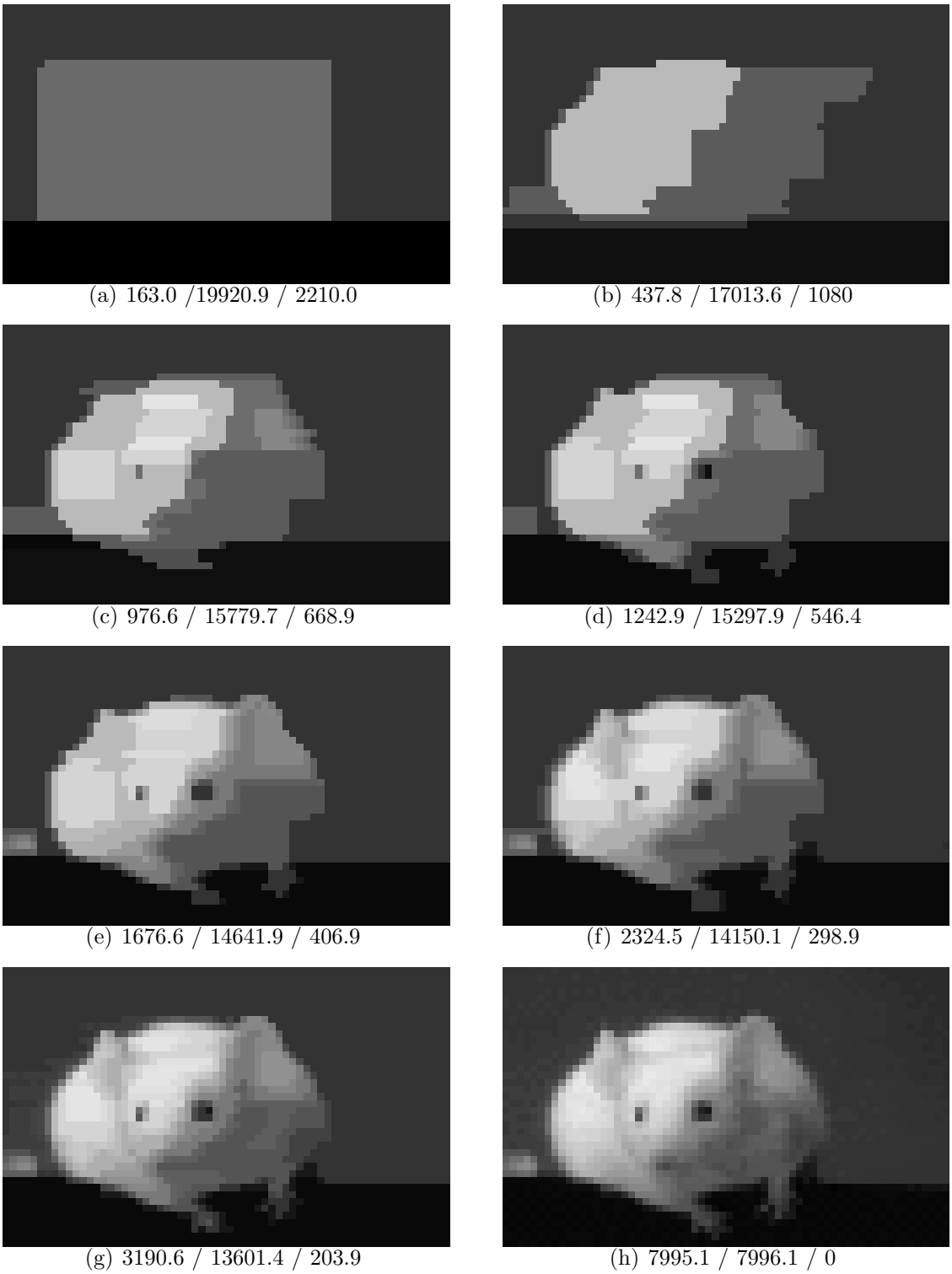
### 6.4.1 Lossy Compression

**Experiment 1: $\mathbb{M}$ouse (Euclidean distortion)**

Our first experiment involved the lossy compression of $\mathbb{M}$, a greyscale image of a mouse. A number of elements of the gene pool are shown in Figure 6.4. The pictures show how at low rates, the models capture the most important global structure of the image; at higher rates more subtle properties of the image can be represented. Image (a) shows a rough rendering of the distribution of bright and dark areas in $\mathbb{M}_{\text{IN}}$. These shapes are rectangular, which is probably an artifact of the compression algorithm we used: it is better able to compress images with rectangular structure than with oval structure. There is no real reason why an oval should be in any way more complex than a rectangle, but most general purpose data compression software is similarly biased. In (b), the rate is high enough that the oval shape of the mouse can be accommodated, and two areas of different overall brightness are identified. After the number of grey shades has been increased a little further in (c), the first hint of the mouse's eyes becomes visible. The eyes are improved and the mouse is given paws in (d). At higher rates, the image becomes more and more refined, but the improvements are subtle and seem of a less qualitative nature.

The code length function (Figure 6.5) shows that the only sufficient statistic in the set of models is $\mathbb{M}_{\text{IN}}$ itself, indicating that the image hardly contains any noise. It also shows the rates that correspond to the models that are shown in Figure 6.4. By comparing these figures it can be clearly seen that the image quality only deteriorates significantly if more than half of the information in $\mathbb{M}_{\text{IN}}$ is discarded. Note that this is not a statement about the compression ratio, where the lossily compressed size is related to the size of the *uncompressed* object rather than its complexity. For example, $\mathbb{M}_{\text{IN}}$ has an uncompressed size of $64 \cdot 40 \cdot 8 = 20480$ bits, and the representation in Figure 6.4(g) has a compressed size of 3190.6 bits. This representation therefore constitutes compression by a factor of $20480/3190.6 = 6.42$, which is substantial for an image of such small size. At the same time the amount of *information* is reduced by a factor of $7995.0/3190.6 = 2.51$.

**Figure 6.4** Lossy image compression results for the mouse (h). The numbers below each image denote its compressed size $\tilde{K}(\cdot)$, total code length $\tilde{K}(\cdot)+L_{(\cdot)}(\mathbb{M}_{\text{IN}})$ and Euclidean distortion $d(\cdot, \mathbb{M}_{\text{IN}})$, respectively.



(a) 163.0 /19920.9 / 2210.0

(b) 437.8 / 17013.6 / 1080

(c) 976.6 / 15779.7 / 668.9

(d) 1242.9 / 15297.9 / 546.4

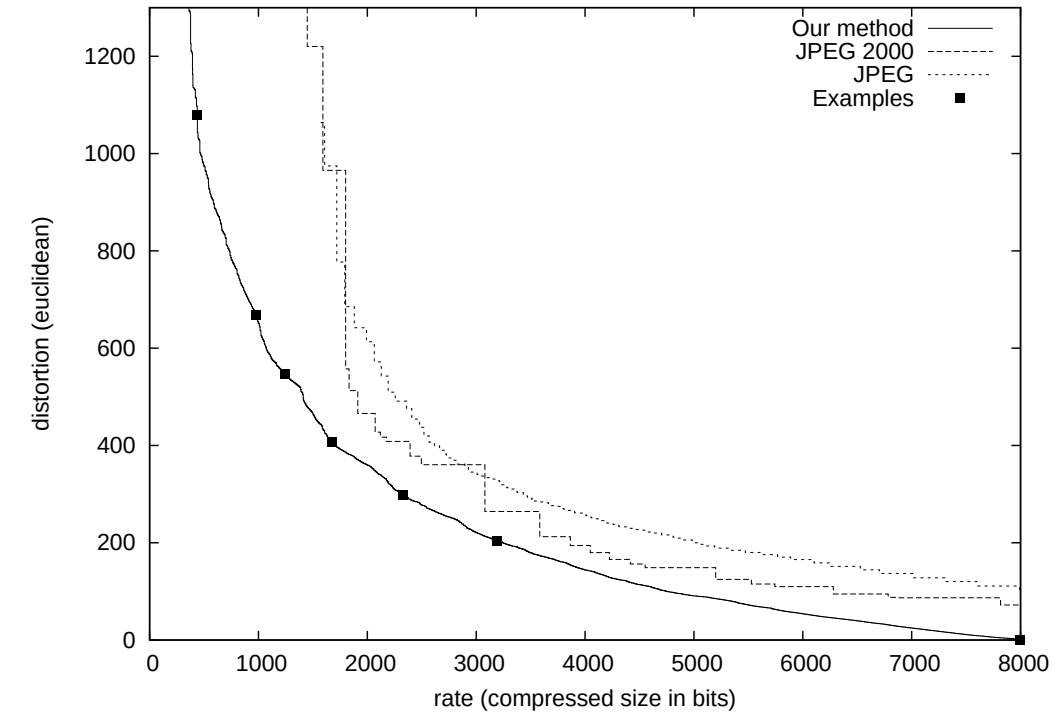
(e) 1676.6 / 14641.9 / 406.9

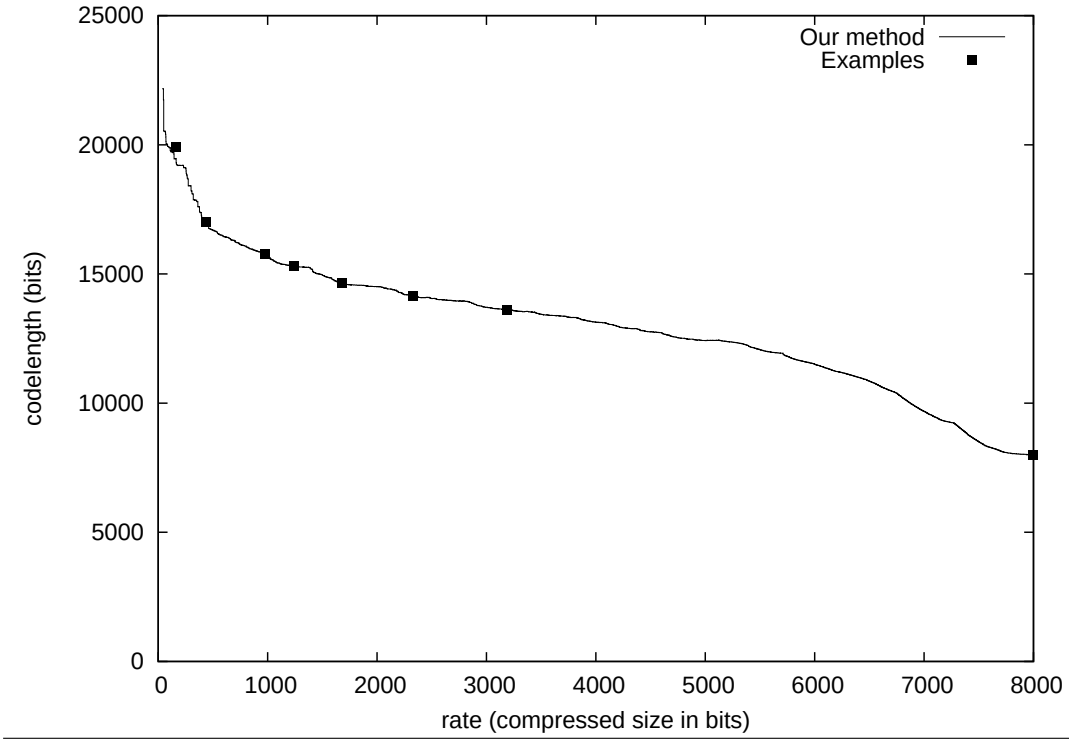
(f) 2324.5 / 14150.1 / 298.9

(g) 3190.6 / 13601.4 / 203.9

(h) 7995.1 / 7996.1 / 0

**Figure 6.5** Approximate distortion-rate and code length functions for the mouse.



### 6.4.2 Denoising

For each denoising experiment, we report a number of important objects, a graph that shows the approximate distortion-rate function and a graph that shows the approximate code length function. In the distortion-rate graph we plot not only the distortion to $\mathbb{O}_{\text{IN}}$ but also the distortion to $\mathbb{O}_{\text{ORIG}}$, to visualise the denoising success at each rate.

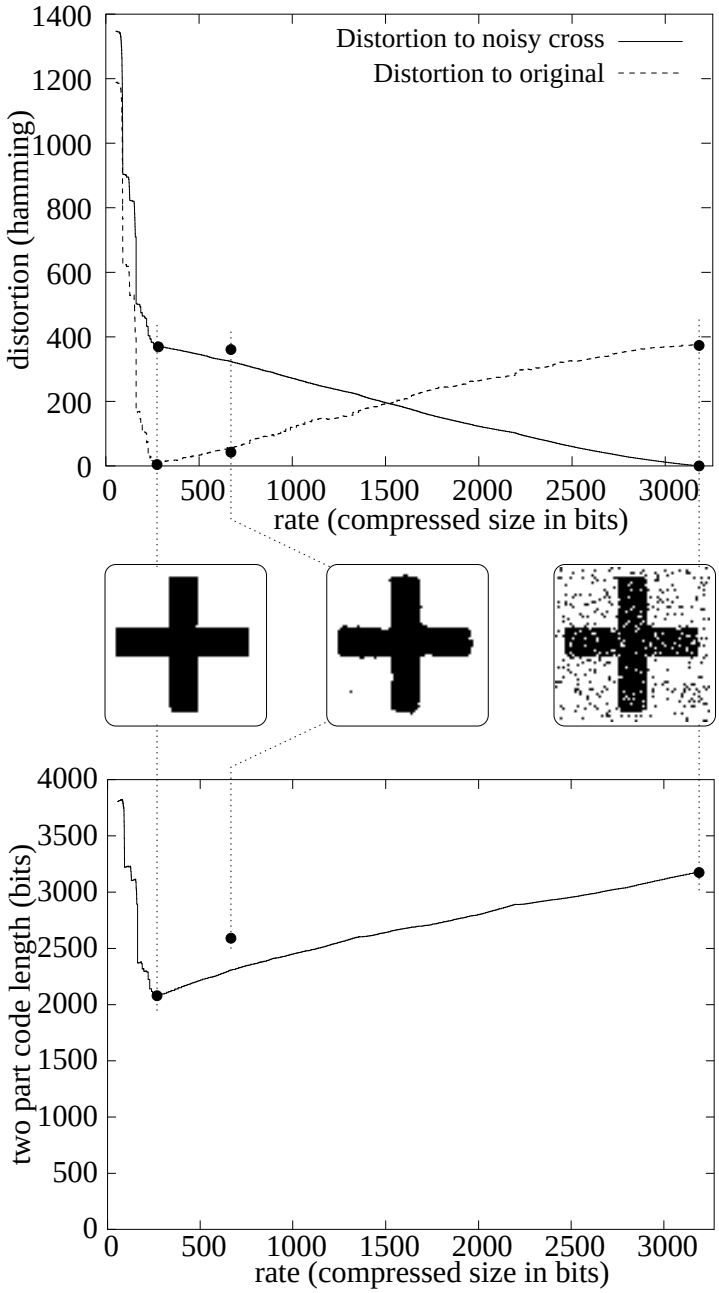
In interpreting these results, it is important to realise that only the reported minimal sufficient statistic and the results of the BayesShrink and naive denoising methods can be obtained without knowledge of the original object – the other objects $\mathbb{O}_{\text{BEST}}$, $\mathbb{O}_{\text{JPEG2000}}$ and $\mathbb{O}_{\text{BLUR}}$ require selecting between a number of alternatives in order to optimise the distortion to $\mathbb{O}_{\text{ORIG}}$, which can only be done in a controlled experiment. Their performance may be better than what can be achieved in practical situations where $\mathbb{O}_{\text{ORIG}}$ is not known.

#### Experiment 2: Noisy $\mathbb{C}$ross (Hamming distortion)

In the first denoising experiment we approximated the distortion-rate function of a monochrome cross $\mathbb{C}_{\text{ORIG}}$ of very low complexity, to which artificial noise was added to obtain $\mathbb{C}_{\text{IN}}$ (the rightmost image in Figure 6.6); the distortion to the noiseless cross is displayed in the same graph. The best denoising $\mathbb{C}_{\text{BEST}}$ (leftmost image) has a distortion of only 3 to the original $\mathbb{C}_{\text{ORIG}}$, which shows that the distortion-rate function indeed separates structure and noise extremely well in this example. The bottom graph shows the code length function for the noisy cross; the minimum on this graph is the minimal sufficient statistic $\mathbb{C}_{\text{MSS}}$. In this low complexity example, we have $\mathbb{C}_{\text{MSS}} = \mathbb{C}_{\text{BEST}}$, so the best denoising is not only very good in this simple example, but it can also be identified.

We did not subject $\mathbb{C}_{\text{IN}}$ to BayesShrink or blurring because those methods are not suitable for monochrome images. Therefore we used the extremely simple, "naive" denoising method that is described in Section 6.3 on this specific image instead. The middle image shows the result $\mathbb{C}_{\text{NAIVE}}$; while it does remove most of the noise, 40 errors remain, a lot more than the number of errors incurred by the minimal sufficient statistic. All errors except one are close to the contours of the cross. This illustrates how the naive algorithm is limited by its property that it takes only the local neighbourhood of each pixel into account, it cannot represent larger structures such as straight lines.

**Figure 6.6** Denoising a noisy cross. Highlighted objects, from left to right: $\mathbb{C}_{\text{BEST}}$, $\mathbb{C}_{\text{NAIVE}}$ and $\mathbb{C}_{\text{IN}}$. Exact values are in the bottom table.




| | $\mathbb{C}_{\text{BEST}}$=$\mathbb{C}_{\text{MSS}}$ | $\mathbb{C}_{\text{NAIVE}}$ | $\mathbb{C}_{\text{IN}}$ |
|---|---|---|---|
| $\tilde{K}(\cdot)$ | 260.4 | 669.2 | 3178.6 |
| $\tilde{K}(\cdot) + L_{(\cdot)}(\mathbb{C}_{\text{IN}})$ | 2081.9 | 2533.3 | 3179.6 |
| $d(\cdot, \mathbb{C}_{\text{IN}})$ | 376 | 389 | 0 |
| $d(\cdot, \mathbb{C}_{\text{ORIG}})$ | 3 | 40 | 377 |

### Experiment 3: $\mathbb{N}$oisy mouse (Euclidean distortion)

The noisy mouse poses a significantly harder denoising problem, where the total complexity of the input $\mathbb{N}_{\text{IN}}$ is more than five times that of the noisy cross. Figure 6.7 shows the denoising results for various denoising methods, and Figure 6.8 shows the rate-distortion curve and, as for the noisy cross, the distortion to the original object $\mathbb{N}_{\text{ORIG}}$.

Figure 6.7(a) shows the input object $\mathbb{N}_{\text{IN}}$; it was constructed by adding noise (centre image) to the original noiseless image $\mathbb{N}_{\text{ORIG}}$ (top-right). We display three different denoising results. Image (h) shows $\mathbb{N}_{\text{BEST}}$, the best denoised object from the gene pool. Visually it appears to resemble $\mathbb{N}_{\text{ORIG}}$ quite well, but there might be structure in $\mathbb{N}_{\text{ORIG}}$ that was lost in the denoising process. Because human perception is perhaps the most sensitive detector of structure in image data, we show the difference between $\mathbb{N}_{\text{BEST}}$ and $\mathbb{N}_{\text{ORIG}}$ in (i). We would expect any significant structure in the original image that is lost in the denoising process, as well as structure that is not present in the original image, but is somehow introduced as an artifact of the denoising procedure, to become visible in this residual. In the case of $\mathbb{N}_{\text{BEST}}$ we cannot make out any particular features.

The minimal sufficient statistic, image (d), also appears to be a reasonably successful denoising, albeit clearly of lower complexity than the best one. In the residual, darker and lighter patches are definitely discernible. Apparently $\mathbb{N}_{\text{IN}}$ does contain some structure beyond what is captured by $\mathbb{N}_{\text{MSS}}$, but this cannot be exploited by the compression algorithm. We think that the fact that the minimal sufficient statistic is of lower complexity than the best possible denoising result should therefore again be attributed to inefficiencies of the compressor.

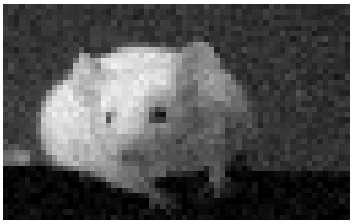
For comparison, we have also denoised $\mathbb{N}_{\text{IN}}$ using the alternative denoising method BayesShrink and the methods based on blurring and JPEG2000 as described in Section 6.3. We found that BayesShrink does not work well for images of such small size: the distortion between $\mathbb{N}_{\text{BS}}$ and $\mathbb{N}_{\text{IN}}$ is only 72.9, which means that the input image is hardly affected at all. Also, $\mathbb{N}_{\text{BS}}$ has a distortion of 383.8 to $\mathbb{N}_{\text{ORIG}}$, which is hardly less than the distortion of 392.1 achieved by $\mathbb{N}_{\text{IN}}$ itself.

Blurring-based denoising yields much better results: $\mathbb{N}_{\text{BLUR}}$ (j) is the result after optimisation of the size of the Gaussian kernel. Its distortion to $\mathbb{N}_{\text{ORIG}}$ lies in-between the distortions achieved by $\mathbb{N}_{\text{MSS}}$ and $\mathbb{N}_{\text{BEST}}$, but it is different from those objects in two important respects. Firstly, $\mathbb{N}_{\text{BLUR}}$ remains much closer to $\mathbb{N}_{\text{IN}}$, at a distortion of 260.4 instead of more than 470, and secondly, $\mathbb{N}_{\text{BLUR}}$ is much less compressible by $\hat{K}$. (To obtain the reported size of 14117 bits we had to switch on the averaging filter, as described in Section 6.2.1.) These observations are at present not well understood. Image (k) shows that the contours of the mouse are somewhat distorted in $\mathbb{N}_{\text{BLUR}}$; this can be explained by the fact that contours contain high frequency information which is discarded by the blurring operation as we remarked in Section 6.3.

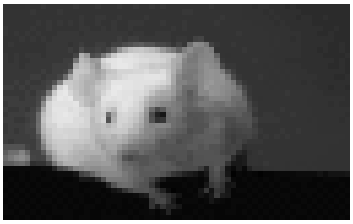
The last denoising method we compared our results to is the one based on the

JPEG2000 algorithm. Its performance is clearly inferior to our method visually as well as in terms of rate and distortion. The result seems to have undergone a smoothing process similar to blurring which introduces similar artifacts in the background noise, as is clearly visible in the residual image. As before, the comparison may be somewhat unfair because JPEG2000 was not designed for the purpose of denoising, might optimise a different distortion measure and is much faster.

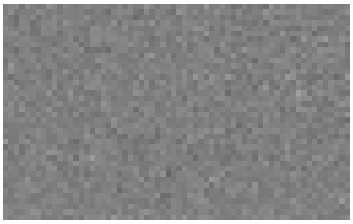
**Figure 6.7** Denoising results for the noisy mouse (a). The numbers below each image denote its compressed size $\tilde{K}(\cdot)$, total code length $\tilde{K}(\cdot) + L_{(\cdot)}(\mathbb{N}_{\text{IN}})$, distortion to $\mathbb{N}_{\text{IN}}$ and distortion to $\mathbb{N}_{\text{ORIG}}$, respectively.



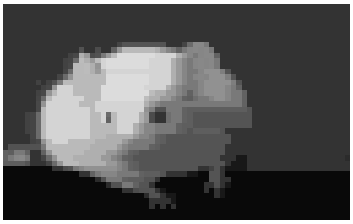
(a) $\mathbb{N}_{\text{IN}}$
16699.7 / 16700.7 / 0 / 392.1



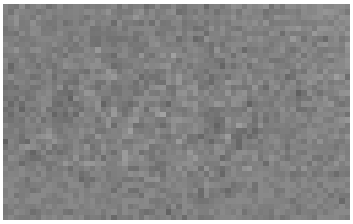
(b) $\mathbb{N}_{\text{ORIG}}(=\mathbb{M}_{\text{IN}})$
7995.1 / 26247.8 / 392.1 / 0



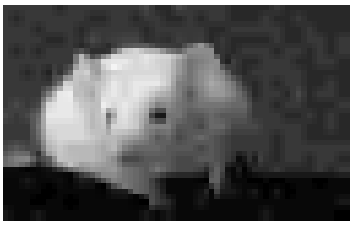
(c) $\mathbb{N}_{\text{NOISE}}$=$\mathbb{N}_{\text{IN}}$-$\mathbb{N}_{\text{ORIG}}$



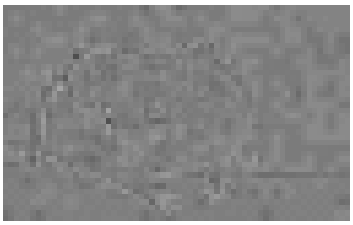
(d) $\mathbb{N}_{\text{MSS}}$
1969.8 / 15504.9 / 474.7 / 337.0



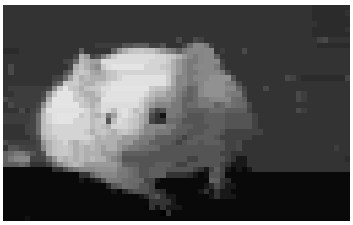
(e) $\mathbb{N}_{\text{MSS}} - \mathbb{N}_{\text{ORIG}}$



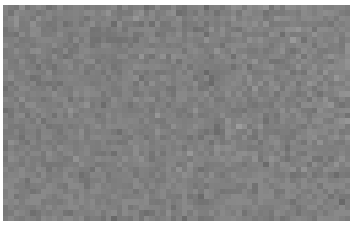
(f) $\mathbb{N}_{\text{JPEG2000}}$
3104 / 16395.4 / 444.4 / 379.8



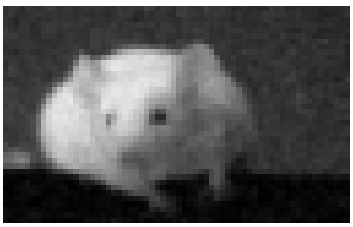
(g) $\mathbb{N}_{\text{JPEG2000}} - \mathbb{N}_{\text{ORIG}}$



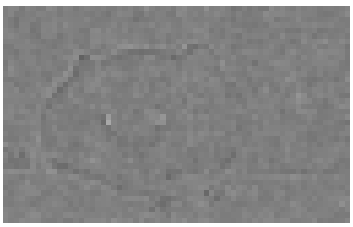
(h) $\mathbb{N}_{\text{BEST}}$
3354.5 / 15952.8 / 368.4 / 272.2



(i) $\mathbb{N}_{\text{BEST}} - \mathbb{N}_{\text{ORIG}}$

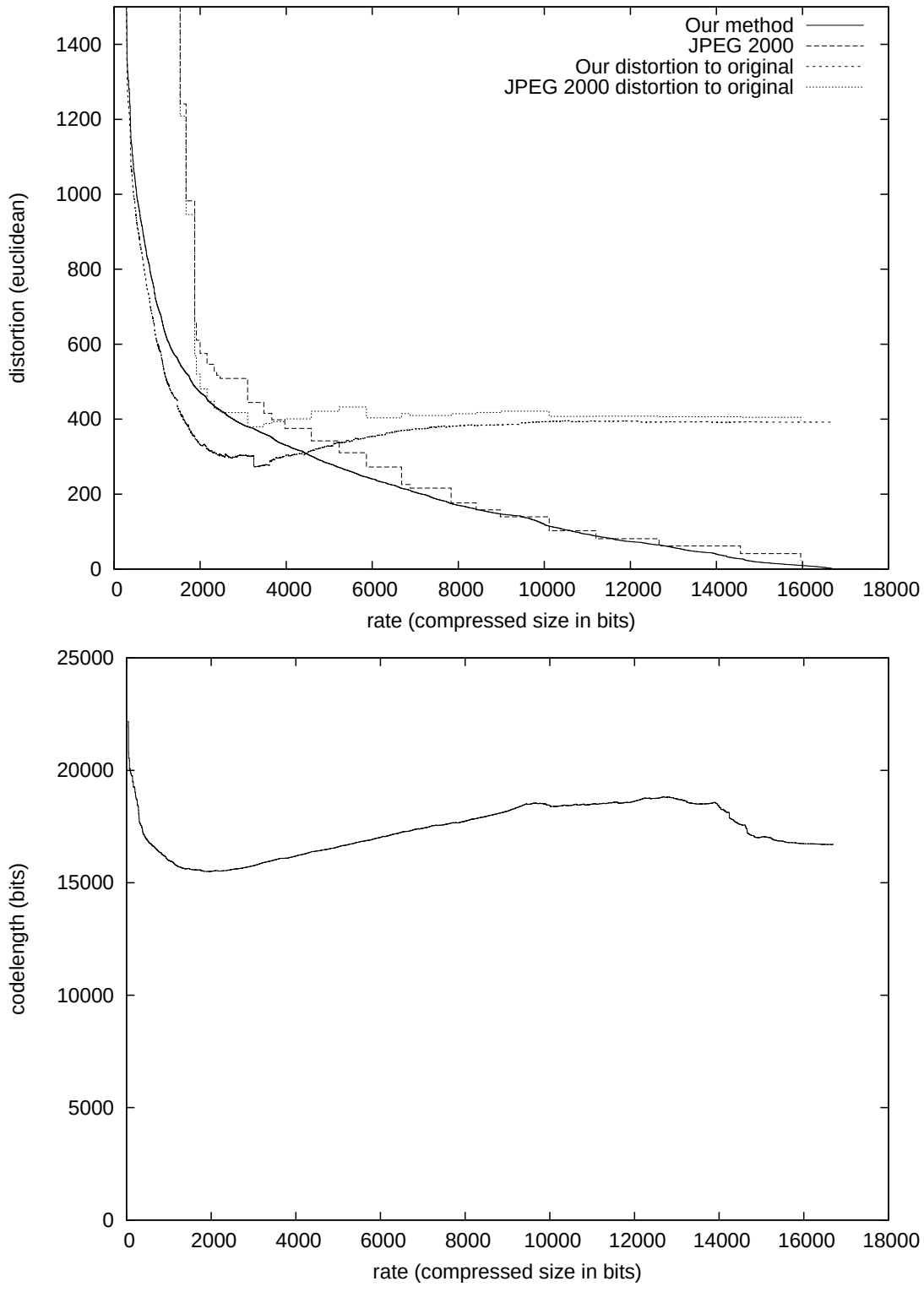


(j) $\mathbb{N}_{\text{BLUR}}$ ($\sigma = 0.458$)
14117.0 / 25732.4 / 260.4 / 291.2



(k) $\mathbb{N}_{\text{BLUR}} - \mathbb{N}_{\text{ORIG}}$

**Figure 6.8** Approximate distortion-rate and code length functions for the noisy mouse.



### Experiment 4: Oscar 𝕎ilde fragment (edit distortion)

The fourth experiment, in which we analyse $\mathbb{W}_{\text{IN}}$, a corrupted quotation from Oscar Wilde, shows that our method is a general approach to denoising that does not require many domain specific assumptions. $\mathbb{W}_{\text{ORIG}}$, $\mathbb{W}_{\text{IN}}$ and $\mathbb{W}_{\text{MSS}}$ are depicted in Figure 6.10, the distortion-rate approximation, the distortion to $\mathbb{W}_{\text{ORIG}}$ and the three part code length function are shown in Figure 6.11. We have trained the compression algorithm by supplying it with the rest of Chapters 1 and 2 of the same novel as side information, to make it more efficient at compressing fragments of English text. We make the following observations regarding the minimal sufficient statistic:

- In this experiment, $\mathbb{W}_{\text{MSS}} = \mathbb{W}_{\text{BEST}}$ so the minimal sufficient statistic separates structure from noise extremely well here.
- The distortion is reduced from 68 errors to only 46 errors. 26 errors are corrected (▲), 4 are introduced (▼), 20 are unchanged (•) and 22 are changed incorrectly (★).
- The errors that are newly introduced (▼) and the incorrect changes (★) typically simplify the fragment a lot, so that the compressed size may be expected to drop significantly. Not surprisingly therefore, many of the symbols marked ▼ or ★ are deletions, or modifications that create a word which is different from the original, but still correct English. The following table lists examples of the last category:

| Line | $\mathbb{W}_{\text{ORIG}}$ | $\mathbb{W}_{\text{IN}}$ | $\mathbb{W}_{\text{MSS}}$ |
|---|---|---|---|
| 3 | `or` | `Nor` | `of` |
| 4 | `the` | `Ghe` | `he` |
| 4 | `any` | `anL` | `an` |
| 4 | `learned` | `JeaFned` | `yearned` |
| 5 | `course` | `corze` | `core` |
| 5 | `then` | `ehen` | `when` |
| 8 | `he` | `fhe` | `the` |

  Since it would be hard for *any* general-purpose mechanical method (that does not incorporate a sophisticated English language model) to determine that these changes are incorrect, we should not be surprised to find a number of errors of this kind.

### Side Information

Figure 6.9 shows that the compression performance is significantly improved if we provide side information to the compression algorithm, and the improvement is typically larger if (1) the amount of side information is larger, or (2) if the

compressed object is more similar to the side information. Thus, by giving side information, correct English prose is recognised as "structure" sooner and a better separation between structure and noise is to be expected. The table also shows that if the compressed object is in some way different from the side information, then adding more side information will at some point become counter-productive, presumably because the compression algorithm will then use the side information to build up false expectations about the object to be compressed, which can be costly.

While denoising performance would probably improve if the amount of side information was increased further, it was infeasible to do so in this implementation. Recall from Section 6.2 that he conditional Kolmogorov complexity $K(y|z)$ is approximated by $\tilde{K}(y|z) = \tilde{K}(zy) - \tilde{K}(z)$. The time required to compute this is dominated by the length of $z$ if the amount of side information is much larger than the size of the object to be compressed. This can be remedied by using a compression algorithm that processes its input sequentially, because the state of such an algorithm can be cached after processing the side information $z$; computing $\tilde{K}(zy)$ would then be a simple matter of recalling the state that was reached after processing $z$ and then processing $y$ starting from that state. Many compression algorithms, among which Lempel-Ziv compressors and most statistical compressors, have this property; our approach could thus be made to work with large quantities of side information by switching to a sequential compressor but we have not done this.

**Figure 6.9** Compressed size of models for different amounts of side information. $\mathbb{W}_{\text{ORIG}}$ is never included in the side information. We do not let $\mathbb{W}_{\text{MSS}}$ vary with side information but keep it fixed at the object reported in Figure 6.10(c).

| Side information $z$ | $\tilde{K}(\mathbb{W}_{\text{ORIG}}\|z)$ | $\tilde{K}(\mathbb{W}_{\text{MSS}}\|z)$ | $\tilde{K}(\mathbb{W}_{\text{IN}}\|z)$ |
|---|---|---|---|
| None | 3344.1 | 3333.7 | 3834.8 |
| Chapters 1,2 (57 kB) | 1745.7 | 1901.9 | 3234.5 |
| Whole novel (421 kB) | 1513.6 | 1876.5 | 3365.9 |

**Figure 6.10** A fragment of *The Picture of Dorian Gray*, by Oscar Wilde.

Beauty, real beauty, ends where an intellectual expression begins. Intellect is in itself a mode of exaggeration, and destroys the harmony of any face. The moment one sits down to think, one becomes all nose, or all forehead, or something horrid. Look at the successful men in any of the learned professions. How perfectly hideous they are! Except, of course, in the Church. But then in the Church they don't think. A bishop keeps on saying at the age of eighty what he was told to say when he was a boy of eighteen, and as a natural consequence he always looks absolutely delightful. Your mysterious young friend, whose name you have never told me, but whose picture really fascinates me, never thinks. I feel quite sure of that.

(a) $\mathbb{W}_{\text{ORIG}}$, the original text

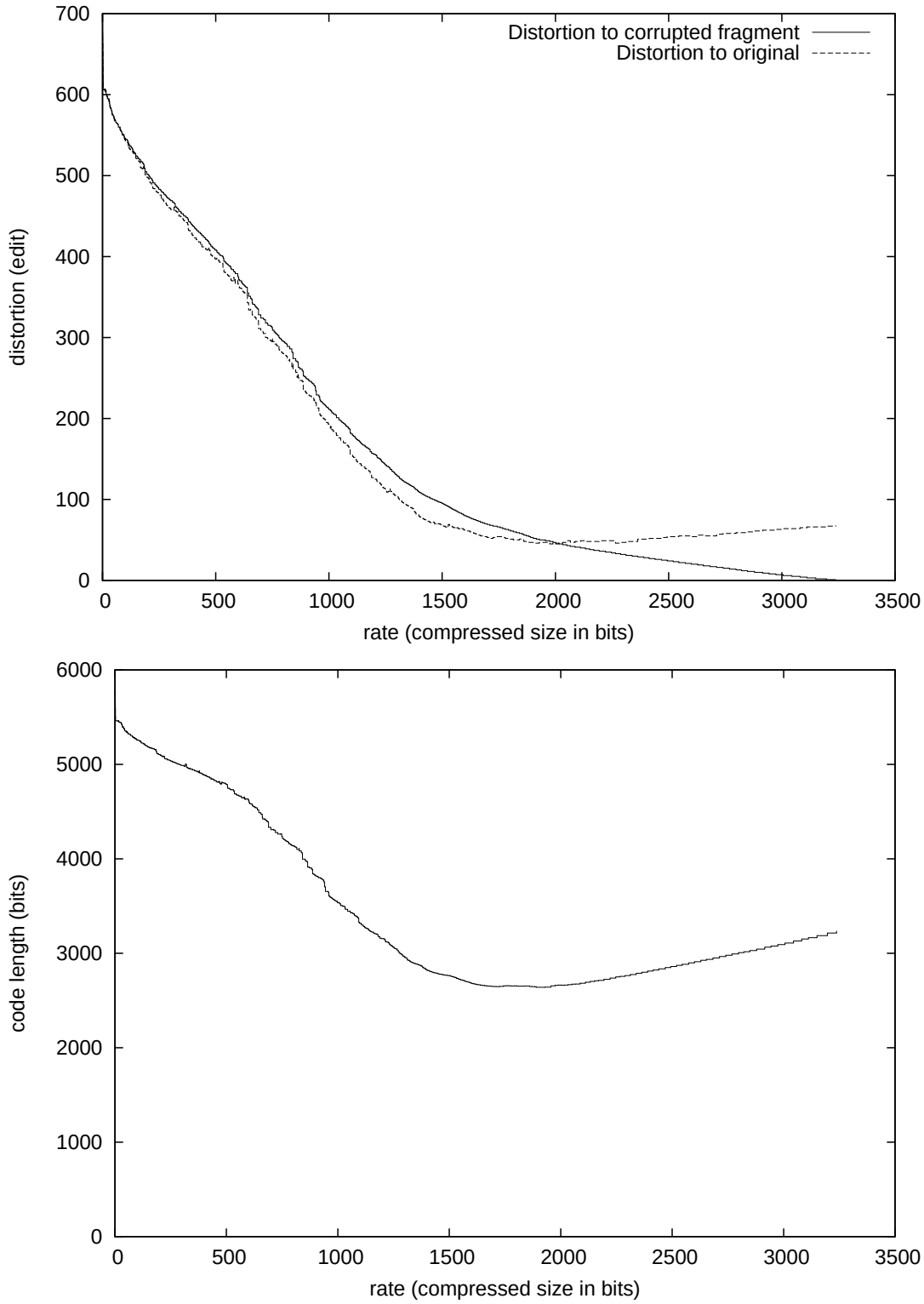
Beauty, real beauty, ends2wheresan intellectual expressoon begins. IntellHct isg in itself a mMde ofSexggeration, an\ destroys theLharmony of n face. :The m1ment one sits down to ahink@ one becomes jll noeˆ Nor all forehbead, or something hNrrid. Look a Ghe successf\l men in anL of te JeaFned professions. How per}ectly tideous 4they re6 Except, of corze, in7 the Ch4rch. BuP ehen in the Church they dol't bthink. =A bishop keeps on saying at the age of eighty what he was told to say wh"n he was aJb4y of eighten, and sja natural cnsequence fhe a(ways looks ab8olstely de[ightfu). Your mysterious youngL friend, wPose name you h\vo never tld me, mut whose picture really fa?scinates Lme,Pnever thinCs. I feel quite surS of that9

(b) $\mathbb{W}_{\text{IN}}$, the corrupted version of the fragment. At 68 randomly selected positions characters have been inserted, deleted or modified. New and replacement characters are drawn uniformly from ASCII symbols 32–126.

Beauty, real beauty, ends-where an intellectual expressoon begins. Intellect is⊡ in itself a mode of ex⊡ggeration, and destroys the harmony of ⊡n⊡ face. ⊡The moment one sits down to think⊡ one becomes ⊡ll no⊡e⊡ ⊡of all forebe⊡d, or something hirrid. Look a⊡ ⊡he successf⊡l men in an⊡ of t⊡e yearned provessions. How perfectly tideous ⊡they ⊡re6 Except, of co⊡r⊡e, in ⊡ the Charch. But when in the Church they dol't ⊡think. ⊡A⊡bishop keeps on saying at the age of eighty what he was told to say wh⊡n he was a⊡bsy of eight⊡en, and ⊡sja natural c⊡nsequence the a⊡ways looks absolutely de⊡ightful. Your mysterious young⊡ friend, whose name you have never t⊡ld me, mut whose picture really fa⊡scinates ⊡me, never thinCs. I feel quite sur⊡ of that9

(c) $\mathbb{W}_{\text{BEST}} = \mathbb{W}_{\text{MSS}}$; it has edit distortion 46 to the original fragment. Marks indicate the error type: ▲=correction; ▼=new error; ●=old error; ★=changed but still wrong. Deletions are represented as ⊡.

**Figure 6.11** Results for a fragment of *The Picture of Dorian Gray* by Oscar Wilde (also see Figure 6.10).



## 6.5 Quality of the Approximation

It is easy to see from its definition that the distortion-rate function must be a non-increasing function of the rate. The implementation guarantees that our approximation is non-increasing as well. In [92] it is assumed that for every $x \in \mathcal{X}$ there exists a representation $y \in \mathcal{Y}$ such that $d(x, y) = 0$; in the context of this chapter this is certainly true because we have $\mathcal{X} = \mathcal{Y}$ and a distortion function which is a metric. The gene pool is initialised with $\mathbb{O}_{\text{IN}}$, which always has zero weakness and must therefore remain in the pool. Therefore at a rate that is high enough to specify $x$, the distortion-rate function reaches zero.

The shape of the code length function for an object $x$ is more complicated. Let $y$ be the representation for which $d(x, y) = 0$. In theory, the code length can never become less than the complexity of $y$, and the minimal sufficient statistic witnesses the code length function at the lowest rate at which the code length is equal to the complexity of $y$. Practically, we found in all denoising experiments that the total code length using the minimal sufficient statistic, $\tilde{K}(\mathbb{O}_{\text{MSS}}) + L_{\mathbb{O}_{\text{MSS}}}(\mathbb{O}_{\text{IN}})$, is *lower* than the code length $\tilde{K}(\mathbb{O}_{\text{IN}})$ that is obtained by compressing the input object directly. This can be observed in Figures 6.6, 6.8 and 6.11. The effect is most pronounced in Figure 6.6, where the separation between structure and noise is most pronounced.

Our hypothesis is that this departure from the theoretical shape of the code length function must be explained by inefficiency of the compression algorithm in dealing with noise. This is evidenced by the fact that it needs 2735.7 bits to encode $\mathbb{C}_{\text{NOISE}}$, while only $\log_2 \binom{4096}{377} \approx 1810$ bits would suffice if the noise were specified with a uniform code on the set of indices of all binary sequences with exactly 377 ones out of $64 \cdot 64$ (see Section 6.2.3). Similarly, $\tilde{K}(\mathbb{N}_{\text{NOISE}}) = 14093$, whereas a literal encoding requires at most 12829 bits (using the bound from Section 6.2.3).

Another strange effect occurs in Figure 6.8, where the code length function displays a strange "bump": as the rate is increased beyond the level required to specify the minimal sufficient statistic, the code length goes up as before, but here at very high rates the code length starts dropping again.

It is theoretically possible that the code length function should exhibit such behaviour to a limited extent. It can be seen in [92] that a temporary increase in the code length function can occur up to a number of bits that depends on the so-called *covering coefficient*. Loosely speaking this is the density of small distortion balls that is required in order to completely cover a larger distortion ball. The covering coefficient in turn depends on the used distortion function and the number of dimensions. It is quite hard to analyse in the case of Euclidean distortion, so we cannot at present say if theory admits such a large increase in the code length function. However, we believe that the explanation is more mundane in this case. Since the noisy mouse is the most complex object of the four we experimented on, we fear that this bump may simply indicate that we

interrupted our search procedure too soon. Quite possibly, after a few years of processing on more expensive hardware, the code length function would have run straight in between $\mathbb{N}_{\text{MSS}}$ and $\mathbb{N}_{\text{IN}}$.

Figure 6.5 shows that our approximation of the distortion-rate function is somewhat better than the approximation provided by either JPEG or JPEG2000, although the difference is not extremely large for higher rates. The probable reason is twofold: on the one hand, we do not know for which distortion function JPEG(2000) is optimised, but it is probably not Euclidean distortion. Therefore our comparison is somewhat unfair, since our method might well perform worse on JPEG(2000)'s own distortion measure. On the other hand, JPEG(2000) is very time-efficient, it took only a matter of seconds to compute models at various different quality levels, while it took our own algorithm days or weeks to compute its distortion-rate approximation. Two conclusions can be drawn from our result. Namely, if the performance of existing image compression software had been better than the performance of our own method in our experiments, this would have been evidence to suggest that our algorithm does not compute a good approximation to the rate-distortion function. The fact that this is not the case is thus reassuring. Vice versa, if we assume that we have computed a good approximation to the algorithmic rate-distortion function, then our results give a measure of how close JPEG(2000) comes to the theoretical optimum; our program can thus be used to provide a basis for the evaluation of the performance of lossy compressors.

## 6.6 An MDL Perspective

So far we have described algorithmic rate-distortion theory in terms of the uncomputable notion of Kolmogorov complexity, and we developed a practical version of the method using a data compression algorithm. In the context of this thesis, it is natural to ask how the practical method we described above fits within established MDL theory. After all, Minimum Description Length was originally developed to obtain a practical version of universal learning based on Kolmogorov complexity, which can even be interpreted as a special case ("ideal MDL", Section 1.1.3). For example, a major theme in the MDL literature is the motivation of the used codes; some codes are considered acceptable, others are not. To what category belongs the data compression algorithm we used?

First we compare the code length function used in MDL to that we used for algorithmic rate-distortion in this chapter. In the considered practical version of algorithmic rate distortion, by (6.2) and (6.7), the total code length of the source object $x \in \mathcal{X}$ using a representation $y \in \mathcal{Y}$ equals

$$\tilde{K}(y) + L_D(d(x,y)|y) + \log |S_y(d(x,y))|. \tag{6.8}$$

In this three part code, the first term counts the number of bits required to

specify the representation, or hypothesis, for the data and the last term counts the number of bits required to specify the noise. Whether the distortion level of the source object given a representation should be interpreted as part of the hypothesis or as noise is debatable; in this chapter we found it convenient to treat the distortion level as part of the hypothesis, as in (6.2). But to match algorithmic rate-distortion to MDL it is better to think of the distortion level as part of the noise and identify $\mathcal{Y}$ with the available hypotheses.

In our description of MDL in the introductory chapter, the total code length of the data $x \in \mathcal{X}$ with the help of a hypothesis $y \in \mathcal{Y}$ is

$$L(y) + L_y(x). \tag{6.9}$$

Algorithmic rate-distortion theory can be understood as a generalisation of MDL as described in the introduction by making (6.8) and (6.9) match. Suppose that we start with an instance of MDL model selection, specified by the code length functions $L(y)$ and $L_y(x)$, and let $P_y(x) = 2^{-L_y(x)}$ be the mass function corresponding to $L_y(x)$. We will show that the model selected by MDL is equal to a sufficient statistic in the rate-distortion problem that is obtained by setting $\tilde{K}(y) = L(y)$, $d(x,y) = L_y(x)$ and $L_D(d|y) = P_y(L_y(X) = d)$. We have

$$\begin{aligned} P_y(x) &= P_y(X = x, L_y(X) = L_y(x)) \\ &= P_y(X = x | L_y(X) = L_y(x)) P_y(L_y(X) = L_y(x)) \\ &= \frac{P_y(L_y(X) = L_y(x))}{|\{x' : L_y(x') = L_y(x)\}|} = \frac{2^{-L_D(d(x,y)|y)}}{|S_y(d(x,y))|}, \end{aligned}$$

so that the code length function that is minimised by sufficient statistics in this rate-distortion problem is exactly the same code length function that is minimised in the original MDL problem. Note that to make this work we only really need the condition that the distortion function has the property that for all $y \in \mathcal{Y}$ and all $x \in \mathcal{X}, x' \in \mathcal{X}$ we have $d(x,y) = d(x',y)$ iff $L_y(x) = L_y(x')$. We chose $d(x,y) = L_y(x)$ because it is a simple function with that property, and because it allows an interpretation of the distortion as the incurred logarithmic loss.

This correspondence actually works both ways: starting with a rate-distortion problem specified by some $\tilde{K}$, code $L_D$ and distortion function $d$, we can also construct an MDL problem that identifies a sufficient statistic by defining $L(y) = \tilde{K}(y)$ and $L_y(x) = L_D(d(x,y)|y) + \log |S_y(d(x,y))|$.

Note that in the introductory chapter we were somewhat unspecific as to *which* hypothesis should be selected if more than one of them minimises the code length; the issue becomes much clearer in a rate-distortion analysis because it allows us to easily express a preference for the *minimal* sufficient statistic as the best hypothesis for the data. If a thorough analysis of the data is required, it also seems sensible to look at the whole rate-distortion function rather than just at the minimal sufficient statistic, since it provides additional information about the structure that is present in the data at each level of complexity.

When we introduced MDL, we stressed that a suitable code $L$ for the specification of hypotheses must have two properties: (a) it has small regret in the worst case over all considered input objects, and (b) it may be a luckiness code that does especially well on some inputs (see Section 1.1). In the experiments in this chapter we used $L = \tilde{K}$, where $\tilde{K}$ is a general purpose data compression algorithm. We need to check that it achieves small regret in the worst case. We will do this for the experiment on the noisy mouse $\mathbb{N}_{\mathrm{IN}}$. In that experiment we only considered finitely many hypotheses, so as in Example 2, the best possible worst-case regret is achieved by the uniform code. This code requires $64 \cdot 40 \cdot 8 = 20480$ bits for all representations in $\mathcal{Y}$. The encoding produced by the data compression algorithm should never be much longer. We did not prove any bounds for our compressor, but we did try to compress ten files of $64 \cdot 40$ random bytes each, to see whether or not the compressed representation would be much larger; the worst result was a compressed size of 20812 bytes. This blowup of 332 bits does seem larger than necessary, but similar or worse overheads are incurred by other compression software such as Lempel-Ziv based compressors and PPM compressors, unless as a built-in feature it checks whether or not it actually manages to compress the input object, reverting to a literal encoding when this fails. All in all, the worst-case performance of the compressor appears to lie within reasonable bounds, although admittedly we did not check this very thoroughly.

The fact that we subject the source object to a rate-distortion analysis in the first place suggests that we believe that reasonable hypotheses can be formulated at different levels of complexity, which means that we should certainly use a luckiness code to differentiate between simple and more complex representations. This is what data compression software is designed to do really well, and what allows for an interesting analysis in the case of our example. Another way to construct a suitable luckiness code $L$, which is more in the spirit of the standard MDL approach, is the following. We start out with a countable set of models of different complexity $\mathcal{M}_1$, $\mathcal{M}_2$, $\ldots$, and take its union $\mathcal{Y} = \cup_n \mathcal{M}_n$ as the set of representations. We can then encode $y$ by first specifying its model index $i$ using some worst-case efficient code, and then specifying $y$ within model $i$ using a second worst-case efficient code. The details of such an approach to rate-distortion have not yet been investigated.

## 6.7 Conclusion

Algorithmic rate-distortion provides a good framework for analysis of large and structured objects. It is based on Kolmogorov complexity, which is not computable. We nevertheless attempted to put this theory into practice by approximating the Kolmogorov complexity of an object by its compressed size. We also generalised the theory in order to enable it to cope with side information, which is interpreted as being available to both the sender and the receiver in a transmission

over a rate restricted channel. We also describe how algorithmic rate-distortion theory may be applied to lossy compression and denoising problems.

Finding the approximate rate-distortion function of an individual object is a difficult search problem. We describe a genetic algorithm that is very slow, but has the important advantage that it requires only few assumptions about the problem at hand. Judging from our experimental results, our algorithm provides a good approximation, as long as its input object is of reasonably low complexity and is compressible by the used data compressor. The shape of the approximate rate-distortion function, and especially that of the associated three part code length function, is reasonably similar to the shape that we would expect on the basis of theory, but there is a striking difference as well: at rates higher than the complexity of the minimal sufficient statistic, the three part code length tends to increase with the rate, where theory suggests it should remain constant. We expect that this effect can be attributed to inefficiencies in the compressor.

We find that the algorithm performs quite well in lossy compression, with apparently somewhat better image quality than that achieved by JPEG2000, although the comparison may not be altogether fair. When applied to denoising, the minimal sufficient statistic tends to be a slight underestimate of the complexity of the best possible denoising (an example of underfitting). This is presumably again due to inefficiencies in the used compression algorithm.

Beside an *approximation* of algorithmic rate-distortion theory, the computable version of the theory can also be interpreted as a *generalisation* of MDL model selection. We concluded this chapter with a discussion of this relationship.

# Bibliography


[1] P. Adriaans and J. van Benthem (editors). *Handbook of the Philosophy of Information*. Elsevier, 2008.

[2] H. Akaike. A new look at statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[3] V. Balasubramanian. Statistical inference, Occam's Razor, and statistical mechanics on the space of probability distributions. *Neural Computation*, 9:349–368, 1997.

[4] A. Barron, J. Rissanen, and B. Yu. The Minimum Description Length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760, 1998.

[5] A.R. Barron. *Logically Smooth Density Estimation*. PhD thesis, Dept. of Electrical Engineering, Stanford University, Stanford, CA, 1985.

[6] A.R. Barron. Information-theoretic characterization of Bayes performance and the choice of priors in parametric and nonparametric problems. In *Bayesian Statistics 6*, pages 27–52. Oxford University Press, 1998.

[7] A.R. Barron. Personal communication, 2008.

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

[9] A.R. Barron and C. Sheu. Approximation of density functions by sequences of exponential families. *Annals of Statistics*, 19(3):1347–1369, 1991.

[10] A.R. Barron, Y. Yang, and B. Yu. Asymptotically optimal function estimation by minimum complexity criteria. In *Proceedings of the 1994 International Symposium on Information Theory*, page 38, Trondheim, Norway, 1994.

[11] J. Berger. Personal communication, 2004.

[12] J.O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Series in Statistics. Springer-Verlag, New York, revised and expanded second edition, 1985.

[13] J.O. Berger and L.R. Pericchi. Objective Bayesian methods for model selection: introduction and comparison. *Institute of Mathematical Statistics Lecture Notes*, (Monograph series) 38:135–207, 1997.

[14] J. Bernardo and A.F.M. Smith. *Bayesian Theory*. Wiley, 1994.

[15] O. Bousquet. A note on parameter tuning for on-line shifting algorithms. Technical report, Max Planck Institute for Biological Cybernetics, 2003.

[16] O. Bousquet and M.K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3:363–396, 2002.

[17] M. Burrows and D.J. Wheeler. A block-sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation, Systems Research Center, May 1994.

[18] G. Chang, B. Yu, and M. Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9:1532–1546, 2000.

[19] N. Chomsky. Three models for the description of language. *IEEE Transactions on Information Theory*, 2(3), September 1956.

[20] R. Cilibrasi and P. Vitányi. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28:49–67, 2004.

[21] B. Clarke. Online forecasting proposal. Technical report, University of Dortmund, 1997. Sonderforschungsbereich 475.

[22] B. Clarke and A. Barron. Jeffreys' prior is asymptotically least favourable under entropy risk. *The Journal of Statistical Planning and Inference*, 41:37–60, 1994.

[23] B.S. Clarke and A.R. Barron. Information-theoretic asymptotics of Bayes methods. *IEEE Transactions on Information Theory*, IT-36(3):453–471, 1990.

[24] J.G. Cleary and I.H. Witten. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, COM-32(4):396–402, April 1984.

[25] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Series in telecommunications. John Wiley, 1991.

[26] A.P. Dawid. 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, 1984.

[27] A.P. Dawid. Statistical theory: The prequential approach. *Journal of the Royal Statistical Society B*, 147, Part 2:278–292, 1984.

[28] A.P. Dawid. Prequential analysis, stochastic complexity and Bayesian inference. In J.M. Bernardo, J.O. Berger, A.P. Dawid, and A.F.M. Smith, editors, *Bayesian Statistics 4*, pages 109–125. Oxford University Press, 1992.

[29] A.P. Dawid. Prequential data analysis. In M. Ghosh and P.K. Pathak, editors, *Current Issues in Statistical Inference: Essays in Honor of D. Basu*, Lecture Notes-Monograph Series, pages 113–126. Institute of Mathematical Statistics, 1992.

[30] X. De Luna and K. Skouras. Choosing a model selection strategy. *Scandinavian Journal of Statistics*, 30:113–128, 2003.

[31] P. Diaconis and D. Freedman. On the consistency of Bayes estimates. *The Annals of Statistics*, 14(1):1–26, 1986.

[32] P. Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 21(2):194–203, 1975.

[33] M.R. Forster. The new science of simplicity. In A. Zellner, H. Keuzenkamp, and M. McAleer, editors, *Simplicity, Inference and Modelling*, pages 83–117. Cambridge University Press, Cambridge, 2001.

[34] D. Foster and E. George. The risk inflation criterion for multiple regression. *Annals of Statistics*, 22:1947–1975, 1994.

[35] L. Gerensce'r. Order estimation of stationary Gaussian ARMA processes using Rissanen's complexity. Technical report, Computer and Automation Institute of the Hungarian Academy of Sciences, 1987.

[36] P. Grünwald and S. de Rooij. Asymptotic log-loss of prequential maximum likelihood codes. In *Proceedings of the Eighteenth Annual Conference on Learning Theory (COLT 2005)*, pages 652–667. Springer, 2005.

[37] P. Grünwald and P. Vitányi. Shannon information and Kolmogorov complexity. Submitted to IEEE Transactions on Information Theory, available at www.cwi.nl/~paulv/publications.html, 2005.

[38] P.D. Grünwald. MDL tutorial. In P.D. Grünwald, I.J. Myung, and M.A. Pitt, editors, *Advances in Minimum Description Length: Theory and Application s*. MIT Press, 2005.

[39] P.D. Grünwald. *The Minimum Description Length Principle*. MIT Press, June 2007.

[40] M. Hansen and B. Yu. Minimum Description Length model selection criteria for generalized linear models. In *Science and Statistics: Festschrift for Terry Speed*, volume 40 of *IMS Lecture Notes – Monograph Series*. Institute for Mathematical Statistics, Hayward, CA, 2002.

[41] M.H. Hansen and B. Yu. Model selection and the principle of Minimum Description Length. *Journal of the American Statistical Association*, 96(454):746–774, 2001.

[42] J.A. Hartigan. *Bayes Theory*. Springer-Verlag, New York, 1983.

[43] D. Helmbold and M. Warmuth. On weak learning. *Journal of Computer and System Sciences*, 50:551–573, 1995.

[44] E.M. Hemerly and M.H.A. Davis. Strong consistency of the PLS criterion for order determination of autoregressive processes. *Ann. Statist.*, 17(2):941–946, 1989.

[45] M. Herbster and M.K. Warmuth. Tracking the best expert. In *Learning Theory: 12th Annual Conference on Learning Theory (COLT 1995)*, pages 286–294, 1995.

[46] M. Herbster and M.K. Warmuth. Tracking the best expert. *Machine Learning*, 32:151–178, 1998.

[47] H. Jeffreys. *Theory of Probability*. Oxford University Press, London, third edition, 1961.

[48] R. Kass and P. Vos. *Geometric Foundations of Asymptotic Inference*. Wiley, 1997.

[49] R.E. Kass and A.E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

[50] P. Kontkanen and P. Myllymäki. A linear-time algorithm for computing the multinomial stochastic complexity. *Information Processing Letters*, 103:227–233, September 2007.

[51] P. Kontkanen, P. Myllymäki, T. Silander, H. Tirri, and P.D. Grünwald. On predictive distributions and Bayesian networks. *Journal of Statistics and Computing*, 10:39–54, 2000.

[52] P. Kontkanen, P. Myllymäki, and H. Tirri. Comparing prequential model selection criteria in supervised learning of mixture models. In T. Jaakkola and T. Richardson, editors, *Proceedings of the Eighth International Conference on Articial Intelligence and Statistics*, pages 233–238. Morgan Kaufman, 2001.

[53] A.D. Lanterman. Hypothesis testing for Poisson versus geometric distributions using stochastic complexity. In Peter D. Grünwald, In Jae Myung, and Mark A. Pitt, editors, *Advances in Minimum Description Length: Theory and Applications*. MIT Press, 2005.

[54] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[55] J.Q. Li and A.R. Barron. Mixture density estimation. In *NIPS*, pages 279–285, 1999.

[56] K.C. Li. Asymptotic optimality of $c_p$ , $c_l$, cross-validation and generalized cross-validation: Discrete index set. *Annals of Statistics*, 15:958–975, 1987.

[57] L. Li and B. Yu. Iterated logarithmic expansions of the pathwise code lengths for exponential families. *IEEE Transactions on Information Theory*, 46(7):2683–2689, 2000.

[58] F. Liang and A. Barron. Exact minimax predictive density estimation and MDL. In Peter D. Grünwald, In Jae Myung, and Mark A. Pitt, editors, *Advances in Minimum Description Length: Theory and Applications*. MIT Press, 2005.

[59] G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley Series in Probability and Statistics, 2000.

[60] M.Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer-Verlag, New York, 2nd edition, 1997.

[61] D.S. Modha and E. Masry. Prequential and cross-validated regression estimation. *Machine Learning*, 33(1), 1998.

[62] A. Moffat. *Compression and Coding Algorithms*. Kluwer Academic Publishers, 2002.

[63] C. Monteleoni and T. Jaakkola. Online learning of non-stationary sequences. In *Advances in Neural Information Processing Systems*, volume 16, Cambridge, MA, 2004. MIT Press.

[64] K.R. Parthasarathy. *Probability Measures on Metric Spaces*. Probability and Mathematical Statistics. Academic Press, 1967.

[65] J. Poland and M. Hutter. Asymptotics of discrete MDL for online prediction. *IEEE Transactions on Information Theory*, 51(11):3780–3795, 2005.

[66] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77-2, pages 257–285, 1989.

[67] J. Rissanen. Modeling by the shortest data description. *Automatica*, 14:465–471, 1978.

[68] J. Rissanen. A universal prior for integers and estimation by Minimum Description Length. *Annals of Statistics*, 11:416–431, 1983.

[69] J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information Theory*, IT-30(4):629–636, 1984.

[70] J. Rissanen. A predictive least squares principle. *IMA Journal of Mathematical Control and Information*, 3:211–222, 1986.

[71] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*, volume 15 of *Series in Computer Science*. World Scientific, 1989.

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

[73] J. Rissanen. MDL denoising. *IEEE Transactions on Information Theory*, 46(7):2537–2543, 2000.

[74] J. Rissanen, T.P. Speed, and B. Yu. Density estimation by stochastic complexity. *IEEE Transactions on Information Theory*, 38(2):315–323, 1992.

[75] G. Roelofs. *PNG – The Definitive Guide*. O'Reilly, 1999. Also available at www.faqs.org/docs/png.

[76] L.J. Savage. *The Foundations of Statistics*. Dover Publications, 1954.

[77] E.D. Scheirer. Structured audio, Kolmogorov complexity, and generalized audio coding. *IEEE Transactions on Speech and Audio Processing*, 9(8), november 2001.

[78] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[79] C.E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423, 623–656, 1948.

[80] R. Shibata. Asymptotic mean efficiency of a selection of regression variables. *Annals of the Institute of Statistical Mathematics*, 35:415–423, 1983.

[81] E. Sober. The contest between parsimony and likelihood. *Systematic Biology*, 4:644–653, 2004.

[82] bzip2 and zzip. www.bzip2.org, debin.net/zzip. Lossless compression software.

[83] ImageMagick and NetPBM. www.imagemagick.org, www.netpbm.sourceforge.net. Open source packages of graphics software.

[84] R.J. Solomonoff. A formal theory of inductive inference, part 1 and part 2. *Information and Control*, 7:1–22, 224–254, 1964.

[85] T. Speed and B. Yu. Model selection and prediction: Normal regression. *Annals of the Institute of Statistical Mathematics*, 45(1):35–54, 1993.

[86] M. Stone. An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion. *Journal of the Royal Statistical Society B*, 39:44–47, 1977.

[87] J. Takeuchi. On minimax regret with respect to families of stationary stochastic processes (in Japanese). In *Proceedings of IBIS 2000*, pages 63–68, 2000.

[88] J. Takeuchi and A. Barron. Asymptotically minimax regret for exponential families. In *Proceedings of SITA 1997*, pages 665–668, 1997.

[89] J. Takeuchi and A.R. Barron. Asymptotically minimax regret by Bayes mixtures. In *Proceedings of the 1998 International Symposium on Information Theory (ISIT 98)*, 1998.

[90] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

[91] J. Tromp. Binary lambda calculus and combinatory logic. www.cwi.nl/ tromp/cl/cl.html, 2007.

[92] N. Vereshchagin and P. Vitányi. Algorithmic rate-distortion theory. arxiv.org/abs/cs.IT/0411014, 2005.

[93] N.K. Vereshchagin and P.M.B. Vitányi. Kolmogorov's structure functions and model selection. *IEEE Transactions on Information Theory*, 50(12):3265–3290, 2004.

[94] P.A.J. Volf and F.M.J. Willems. Switching between two universal source coding algorithms. In *Proceedings of the Data Compression Conference, Snowbird, Utah*, pages 491–500, 1998.

[95] V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35:247–282, 1999.

[96] E.J. Wagenmakers, P.D. Grünwald, and M. Steyvers. Accumulative prediction error and the selection of time series models. *Journal of Mathematical Psychology*, 2006. (this issue).

[97] C.Z. Wei. On predictive least squares principles. *Annals of Statistics*, 20(1):1–42, 1990.

[98] P. Whittle. Bounds for the moments of linear and quadratic forms in independent variables. *Theory of Probability and its Applications*, V(3), 1960.

[99] H. Wong and B. Clarke. Improvement over Bayes prediction in small samples in the presence of model uncertainty. *The Canadian Journal of Statistics*, 32(3):269–283, 2004.

[100] Q. Xie and A. Barron. Asymptotic minimax regret for data compression, gambling and prediction. *IEEE Transactions on Information Theory*, 46(2):431–445, 2000.

[101] Y. Yang. Model selection for nonparametric regression. *Statistica Sinica*, 9:475–499, 1999.

[102] Y. Yang. Mixing strategies for density estimation. *Annals of Statistics*, 28(1):75–87, 2000.

[103] Y. Yang. Can the strengths of AIC and BIC be shared? *Biometrica*, 92(4):937–950, 2005.

[104] Y. Yang. Consistency of cross-validation for comparing regression procedures. Submitted, 2005.

[105] Y. Yang and A.R. Barron. An asymptotic property of model selection criteria. *IEEE Transactions on Information Theory*, 44:117–133, 1998.

[106] Y. Yang and A.R. Barron. Information-theoretic determination of minimax rates of convergence. *Annals of Statistics*, 27:1564–1599, 1999.

[107] B. Yu. Lower bounds on expected redundancy for nonparametric classes. *IEEE Transactions on Information Theory*, 42(1):272–275, 1996.

# Notation

**Numbers**

| | |
|---|---|
| $\lfloor x \rfloor$ | The value of $x$, rounded down to the nearest integer |
| $\lceil x \rceil$ | The value of $x$, rounded up to the nearest integer |
| $\log x$ | The binary logarithm of $x$ |
| $\ln x$ | The natural logarithm of $x$ |
| $\mathbf{1}_A(x)$ | The indicator function, which equals 1 if $x \in A$ and 0 otherwise |

**Sets**

| | | |
|---|---|---|
| $\mathbb{B}$ | Binary numbers 0 and 1 | (Note: in Chapter 6, the blackboard symbols are used for something else) |
| $\mathbb{N}$ | Natural numbers $0, 1, 2, \ldots$ | |
| $\mathbb{Z}$ | Integers $\ldots, -2, -1, 0, 1, 2, \ldots$ | |
| $\mathbb{Z}^+$ | Positive integers $1, 2, \ldots$ | |
| $\lvert\mathcal{X}\rvert$ | The size of a finite set $\mathcal{X}$ | |
| $[n]$ | Abbreviation for $\{1, 2, \ldots, n\}$, with $[0] = \emptyset$ and $[\infty] = \mathbb{Z}^+$ | |

**Sequences**. Let $n \in \mathbb{N}$ and let $\mathcal{X}$ be a countable set.

| | |
|---|---|
| $\mathcal{X}^n$ | The set of all length $n$ sequences of elements from $\mathcal{X}$ |
| $\mathcal{X}^*$ | The set of all finite sequences of elements from $\mathcal{X}$: $\mathcal{X}^* := \cup_{n \in \mathbb{N}} \mathcal{X}^n$ |
| $\mathcal{X}^\infty$ | The set of all infinite sequences of elements from $\mathcal{X}$ |
| $x^n$ | EITHER exponentiation, OR the sequence $x_1, x_2, \ldots, x_n$ for $n \in \mathbb{N}$ |
| $x^0$ / $\epsilon$ | The empty sequence |

**Asymptotics**. Let $a, c, x_0, x \in \mathbb{R}$.

| | | |
|---|---|---|
| $f \to a$ | $\lim_{x \to \infty} f(x) = a$ | |
| $f = O(g)$ | $\exists x_0, c > 0 : \forall x \geq x_0 : \lvert f(x)\rvert \leq c\lvert g(x)\rvert$ | |
| $f = \Omega(g)$ | $\exists x_0, c > 0 : \forall x \geq x_0 : \lvert f(x)\rvert \geq c\lvert g(x)\rvert$ | (*Not* the complement of $o$!) |
| $f = \Theta(g)$ | $f = O(g)$ and $f = \Omega(g)$ | |
| $f \asymp g$ | The same as $f = \Theta(g)$ | |
| $f = o(g)$ | $f(x)/g(x) \to 0$ | |

# Abstract

Model selection is a strange and wonderful topic in learning theory and statistics. At first glance the question seems very clear-cut: how should we decide which set of probability distributions matches the observations at hand best. This question comes up time and again in many different contexts, ranging from testing scientific hypotheses in general (which among these psychological models describes best how people behave?) to more concrete applications (what order polynomial should we use to fit the data in this regression problem? What lossy representation of this image best captures the structural properties of the original?). Thus, model selection is ubiquitous, and the one-size-fits-all criteria based on the Minimum Description Length (MDL) principle and the closely related Bayesian statistics are appreciated by many.

Upon closer inspection, many applications of model selection are not as similar as they may first appear. They can be distinguished by technical properties (are the models nested? Parametric? Countable?), but also by a priori assumptions (is the process generating the data believed to be an element of any of the considered models?), as well as the motivation for performing model selection in the first place (do we want to identify which model contains the data generating process, or do we want to identify which model we may expect to predict future data best?). The best choice of methodology in any situation often depends on such particulars, and is further determined by practical considerations such as whether or not the relevant quantities can be evaluated analytically, and whether efficient algorithms exist for their calculation. MDL/Bayesian model selection has been shown to perform quite well in many different contexts and applications; in this thesis we treat some of the puzzling problems and limitations that have also become apparent over time. We also extend the idea by linking it to other topics in machine learning and statistical inference.

To apply MDL, universal codes or distributions have to be associated with each of the considered models. The preferred code is the Normalised Maximum Likelihood (NML) or Shtarkov code. However, this code yields infinite code word

lengths for many models. This first issue with MDL model selection is investigated in Chapter 2, in which we perform computer experiments to test the performance of some of the available alternatives. One result is that the model selection criterion based on the so-called prequential plug-in code displays inferior performance. This observation seems important because the prequential plug-in code is often thought of as a convenient alternative to other universal codes such as the NML code, as it is much easier to calculate. It was thought to result in code lengths similar to those obtained for other universal codes (such as NML, 2-part codes or Bayesian mixtures), but we discovered that this is only the case if the data generating process is in the model. We show in Chapter 3 that the redundancy of the prequential plug-in code is fundamentally different from the standard set by other universal codes if the data generating process is not an element of the model, so that caution should be exercised when it is applied to model selection.

The third problem treated in this thesis is that MDL/Bayesian model selection normally does not take into account that, even in the ideal case where one of the considered models is "true" (contains the data generating process), and even if the data generating process is stationary ergodic, then still the index of the model whose associated universal code issues the best predictions of future data often changes with the sample size. Roughly put, at small sample sizes simple models often issue better predictions of future data than the more complex "true" model, i.e. the smallest model that contains the data generating distribution. When from a certain sample size onward the true model predicts best, the simpler model has already built up a lot of evidence in its favour, and a lot of additional data have to be gathered before the true model "catches up" and is finally identified by Bayesian/MDL model selection. This phenomenon is described in Chapter 5, in which we also introduce a novel model selection procedure that selects the true model almost as soon as enough data have been gathered for it to be able to issue the best predictions. The criterion is consistent: under mild conditions, the true model is selected with probability one for sufficiently large sample sizes. We also show that a prediction strategy based on this model selection criterion achieves an optimal rate of convergence: its cumulative KL-risk is as low as that of any other model selection criterion. The method is based on the *switch distribution*, which can be evaluated using an efficient expert tracking algorithm. More properties of this switch distribution are treated in Chapter 4, which also contains a survey of this and other expert tracking algorithms and shows how such algorithms can be formulated in terms of Hidden Markov Models.

Finally, in Chapter 6 we evaluate the new theory of algorithmic rate-distortion experimentally. This theory was recently proposed by Vitányi and Vereshchagin as an alternative to classical rate-distortion theory. It allows analysis of the structural properties of individual objects and does not require the specification of a probability distribution on source objects. Instead it is defined in terms of Kolmogorov complexity, which is uncomputable. To be able to test this theory in practice we have approximated the Kolmogorov complexity by the compressed

size of a general purpose data compression algorithm. This practical framework is in fact a generalisation of MDL model selection.

The perspectives offered in this thesis on many aspects of MDL/Bayesian model selection, contribute to a better understanding of the relationships between model selection and such diverse topics as universal learning, prediction with expert advice, rate distortion theory and Kolmogorov complexity.

# Samenvatting

Modelselectie is een ongrijpbaar onderwerp in de leertheorie en statistiek. Op het eerste gezicht lijkt het probleem duidelijk: hoe moeten we beslissen welke verzameling van kansverdelingen het best overeen komt met de beschikbare observaties. Deze vraag duikt telkens weer op in allerlei verschillende contexten, waaronder het toetsen van hypothesen in het algemeen (welke van deze psychologische modellen beschrijft het best hoe mensen zich gedragen?) tot meer concrete toepassingen (een polynoom van welke graad moeten we kiezen om de trend in deze gegevens te beschrijven? Welke "lossy" representatie van dit plaatje beschrijft de structurele eigenschappen van het origineel het best?). Kortom, modelselectie is een sleutelprobleem in vele verschillende toepassingen. De one-size-fits-all-oplossingen die gebaseerd zijn op het Minimum Description Length (MDL) principe en de nauw verwante Bayesiaanse statistiek worden daarom veel gebruikt.

Bij nadere beschouwing blijkt dat de vele toepassingen van modelselectie op essentiële punten verschillen. Ze kunnen worden onderscheiden op basis van technische eigenschappen (zijn de modellen in elkaar bevat? Parametrisch? Telbaar?), maar ook op basis van a priori aannames (nemen we aan dat het proces dat de gegevens genereert een element is van een van onze modellen of niet?), alsmede de oorspronkelijke motivatie voor het doen van modelselectie (willen we het model identificeren dat het proces bevat dat de gegevens genereert, of willen we een model selecteren waarvan we mogen hopen dat het toekomstige uitkomsten goed zal voorspellen?). De meest wenselijke methodologie hangt in de praktijk vaak af van dergelijke kwesties, nog los van praktische afwegingen zoals of de relevante grootheden al dan niet efficiënt kunnen worden uitgerekend.

Op allerlei manieren is aangetoond dat het gebruik van MDL/Bayesiaanse modelselectie leidt tot goede prestaties in vele contexten; in dit proefschrift wordt een aantal van de raadselachtige problemen en beperkingen van de methodologie onderzocht, die niettemin in de loop van de tijd aan het licht zijn gekomen. Ook wordt het toepassingsdomein van MDL/Bayesiaanse modelselectie uitgebreid door het te koppelen aan andere onderwerpen in de machine learning en statistiek.

Om MDL toe te kunnen passen moeten zogenaamde universele codes of uni-

versele kansverdelingen worden toegewezen aan alle modellen die worden overwogen. De code die daarbij volgens sommige literatuur de voorkeur heeft is de Normalised Maximum Likelihood (NML) of Shtarkov code. Het blijkt echter dat deze code voor vele modellen leidt tot oneindige codelengtes, waardoor de prestaties van de verschillende modellen niet meer met elkaar vergeleken kunnen worden. Dit eerste probleem met MDL modelselectie wordt onderzocht in hoofdstuk 2, waarin we computerexperimenten uitvoeren om de prestaties te meten van enkele van de beschikbare alternatieven voor de NML code. Een van de meest interessante resultaten is dat de zogenaamde prequentiële plug-in code leidt tot inferieure modelselectieprestaties. De prequentiële plug-in code wordt vaak gezien als een handig alternatief voor andere codes zoals de NML code, omdat het vaak veel makkelijker is uit te rekenen. Het werd vrij algemeen aangenomen dat de resulterende codelengtes vergelijkbaar waren met die van andere universele codes zoals NML of 2-part codes, maar uit onze experimenten blijkt dus dat dit niet onder alle omstandigheden het geval is. In hoofdstuk 3 wordt aangetoond dat de redundantie van de prequentiële plug-in code fundamenteel verschilt van die van andere universele codes in het geval dat het proces dat de gegevens produceert geen element is van het model. Dit betekent dat prequentiële plug-in codes met beleid moeten worden toegepast in modelselectie.

Het derde probleem dat wordt behandeld in dit proefschrift is dat MDL en Bayesiaanse modelselectie normaal gesproken geen rekening houden met het volgende: zelfs in het ideale geval waarin een van de beschikbare modellen "waar" is (het proces dat de gegevens produceert bevat), en zelfs als het gegevens producerende proces stationair en ergodisch is, dan nog *hangt het af van de hoeveelheid beschikbare gegevens* welk van de beschikbare modellen de beste voorspellingen van toekomstige uitkomsten levert. Ruwweg kan worden gesteld dat, als de hoeveelheid beschikbare gegevens klein is, dat dan eenvoudige modellen vaak betere voorspellingen leveren dan het complexere "ware" model (i.e., het kleinste model dat het gegevens producerende proces bevat). Als vervolgens op een gegeven moment de hoeveelheid beschikbare gegevens zo groot is geworden dat het ware model het beste begint te voorspellen, dan heeft het eenvoudigere model al zo lang zoveel beter gepresteerd dat het soms zeer lang kan duren voordat het door de MDL/Bayesiaanse modelselectieprocedure wordt verworpen. Hoofdstuk 5 beschrijft dit verschijnsel, alsmede een nieuwe modelselectieprocedure die het ware model prefereert vrijwel zodra er voldoende gegevens beschikbaar zijn dat dat model de beste voorspellingen begint te leveren. Deze nieuwe procedure is *consistent*, wat betekent dat (onder milde condities) het ware model wordt geselecteerd met kans 1 mits er voldoende gegevens beschikbaar zijn. We tonen ook aan dat voorspellen op basis van deze modelselectieprocedure leidt tot optimaal snelle convergentie: de cumulatieve KL-risk is bewijsbaar zo laag als die van willekeurig welke andere modelselectieprocedure. De methode is gebaseerd op de *switch-verdeling*, die kan worden uitgerekend met behulp van een efficiënt algoritme voor expert tracking. Meer eigenschappen van deze switch-verdeling

worden behandeld in hoofdstuk 4, waarin we een overzicht geven van deze en andere algoritmes voor expert tracking, en waarin we laten zien hoe zulke algoritmes handig kunnen worden geformuleerd in termen van Hidden Markov Models.

In hoofdstuk 6 tenslotte evalueren we de nieuwe theorie van algoritmische rate-distortion experimenteel. Deze theorie werd recentelijk voorgesteld door Vitányi en Vereshchagin als een alternatief voor klassieke rate-distortiontheorie. Ze maakt analyse mogelijk van de structurele eigenschappen van individuele objecten, en vereist niet dat er een objectbron wordt gespecificeerd in de vorm van een kansverdeling. In plaats daarvan wordt algoritmische rate-distortion gedefinieerd in termen van Kolmogorovcomplexiteit, die niet berekenbaar is. Om deze theorie toch in de praktijk te kunnen toetsen benaderen we de Kolmogorovcomplexiteit met de gecomprimeerde grootte van een algemeen toepasbaar datacompressiealgoritme. De zo verkregen praktische aanpak is in feite een generalisatie van MDL modelselectie.

De perspectieven die in dit proefschrift worden geboden op vele aspecten van MDL/Bayesiaanse modelselectie dragen bij tot een dieper begrip van de verbanden tussen modelselectie en diverse onderwerpen als universal learning, voorspellen met advies van experts, rate-distortiontheorie en Kolmogorovcomplexiteit.

*Titles in the ILLC Dissertation Series:*

ILLC DS-2001-01: **Maria Aloni**
*Quantification under Conceptual Covers*

ILLC DS-2001-02: **Alexander van den Bosch**
*Rationality in Discovery - a study of Logic, Cognition, Computation and Neuropharmacology*

ILLC DS-2001-03: **Erik de Haas**
*Logics For OO Information Systems: a Semantic Study of Object Orientation from a Categorial Substructural Perspective*

ILLC DS-2001-04: **Rosalie Iemhoff**
*Provability Logic and Admissible Rules*

ILLC DS-2001-05: **Eva Hoogland**
*Definability and Interpolation: Model-theoretic investigations*

ILLC DS-2001-06: **Ronald de Wolf**
*Quantum Computing and Communication Complexity*

ILLC DS-2001-07: **Katsumi Sasaki**
*Logics and Provability*

ILLC DS-2001-08: **Allard Tamminga**
*Belief Dynamics. (Epistemo)logical Investigations*

ILLC DS-2001-09: **Gwen Kerdiles**
*Saying It with Pictures: a Logical Landscape of Conceptual Graphs*

ILLC DS-2001-10: **Marc Pauly**
*Logic for Social Software*

ILLC DS-2002-01: **Nikos Massios**
*Decision-Theoretic Robotic Surveillance*

ILLC DS-2002-02: **Marco Aiello**
*Spatial Reasoning: Theory and Practice*

ILLC DS-2002-03: **Yuri Engelhardt**
*The Language of Graphics*

ILLC DS-2002-04: **Willem Klaas van Dam**
*On Quantum Computation Theory*

ILLC DS-2002-05: **Rosella Gennari**
*Mapping Inferences: Constraint Propagation and Diamond Satisfaction*

ILLC DS-2002-06: **Ivar Vermeulen**
*A Logical Approach to Competition in Industries*

ILLC DS-2003-01: **Barteld Kooi**
*Knowledge, chance, and change*

ILLC DS-2003-02: **Elisabeth Catherine Brouwer**
*Imagining Metaphors: Cognitive Representation in Interpretation and Understanding*

ILLC DS-2003-03: **Juan Heguiabehere**
*Building Logic Toolboxes*

ILLC DS-2003-04: **Christof Monz**
*From Document Retrieval to Question Answering*

ILLC DS-2004-01: **Hein Philipp Röhrig**
*Quantum Query Complexity and Distributed Computing*

ILLC DS-2004-02: **Sebastian Brand**
*Rule-based Constraint Propagation: Theory and Applications*

ILLC DS-2004-03: **Boudewijn de Bruin**
*Explaining Games. On the Logic of Game Theoretic Explanations*

ILLC DS-2005-01: **Balder David ten Cate**
*Model theory for extended modal languages*

ILLC DS-2005-02: **Willem-Jan van Hoeve**
*Operations Research Techniques in Constraint Programming*

ILLC DS-2005-03: **Rosja Mastop**
*What can you do? Imperative mood in Semantic Theory*

ILLC DS-2005-04: **Anna Pilatova**
*A User's Guide to Proper names: Their Pragmatics and Semanics*

ILLC DS-2005-05: **Sieuwert van Otterloo**
*A Strategic Analysis of Multi-agent Protocols*

ILLC DS-2006-01: **Troy Lee**
*Kolmogorov complexity and formula size lower bounds*

ILLC DS-2006-02: **Nick Bezhanishvili**
*Lattices of intermediate and cylindric modal logics*

ILLC DS-2006-03: **Clemens Kupke**
*Finitary coalgebraic logics*

ILLC DS-2006-04: **Robert Špalek**
*Quantum Algorithms, Lower Bounds, and Time-Space Tradeoffs*

ILLC DS-2006-05: **Aline Honingh**
*The Origin and Well-Formedness of Tonal Pitch Structures*

ILLC DS-2006-06: **Merlijn Sevenster**
*Branches of imperfect information: logic, games, and computation*

ILLC DS-2006-07: **Marie Nilsenova**
*Rises and Falls. Studies in the Semantics and Pragmatics of Intonation*

ILLC DS-2006-08: **Darko Sarenac**
*Products of Topological Modal Logics*

ILLC DS-2007-01: **Rudi Cilibrasi**
*Statistical Inference Through Data Compression*

ILLC DS-2007-02: **Neta Spiro**
*What contributes to the perception of musical phrases in western classical music?*

ILLC DS-2007-03: **Darrin Hindsill**
*It's a Process and an Event: Perspectives in Event Semantics*

ILLC DS-2007-04: **Katrin Schulz**
*Minimal Models in Semantics and Pragmatics: Free Choice, Exhaustivity, and Conditionals*

ILLC DS-2007-05: **Yoav Seginer**
*Learning Syntactic Structure*

ILLC DS-2008-01: **Stephanie Wehner**
*Cryptography in a Quantum World*

ILLC DS-2008-02: **Fenrong Liu**
*Changing for the Better: Preference Dynamics and Agent Diversity*

ILLC DS-2008-03: **Olivier Roy**
*Thinking before Acting: Intentions, Logic, Rational Choice*

ILLC DS-2008-04: **Patrick Girard**
*Modal Logic for Belief and Preference Change*

ILLC DS-2008-05: **Erik Rietveld**
*Unreflective Action: A Philosophical Contribution to Integrative Neuroscience*

ILLC DS-2008-06: **Falk Unger**
*Noise in Quantum and Classical Computation and Non-locality*

ILLC DS-2008-07: **Steven de Rooij**
*Minimum Description Length Model Selection: Problems and Extensions*