On the Determination of Optimal Parameterized Prefix Codes for Adaptive Entropy Coding

Amir Said
Media Technologies Laboratory
HP Laboratories Palo Alto

HPL-2006-74
April 28, 2006*

entropy coding, Golomb-Rice codes, data compression

While complex data sources, like images and audio, require sophisticated coding contexts and source modeling, commonly the high cost of estimating conditional probabilities and computing optimal codes can be avoided by storing sets of codewords, and selecting the best code based on source estimates. Golomb-Rice codes are commonly used because they are parameterized with a single integer, are easy to implement, and are optimal for sources with geometric distribution. In this work, we consider the fact that the Golomb-Rice codes are truly optimal only when the source's single parameter is known with certainty, which in practice is never the case. We investigate how these codes perform, depending on how the source is estimated from previous samples. Next, we analyze possible changes, and propose the class of *unary-stem codes*, which increase robustness, while keeping the useful structural properties, by defining sets of codewords parameterized by several integers. While this is somewhat similar to other proposed codes, it is significantly more general, in order to better evaluate how source uncertainty affects the structure of the optimal codes. We show how the new codes can be designed by updating maximum-likelihood or Bayesian estimations, or optimized according to posterior probabilities. We also show how data for modeling the source uncertainty can be accurately computed, and develop algorithms capable of dealing with the infinite number of symbols, to quickly find the optimal codes. Numerical results show that the optimal codes are, as expected, always better than Golomb-Rice codes, and the difference can be quite significant. Furthermore, the analysis of the numerical results shows the advantages of integrating mappings from source-sample data directly to code selection.

* Internal Accession Date Only
A summary of this paper was presented at the 2005 Data Compression Conference, 29-31 March 2005, Snowbird, UT, USA
Approved for External Publication

On the Determination of Optimal Parameterized Prefix Codes for Adaptive Entropy Coding

Amir Said
Hewlett Packard Laboratories
Palo Alto, CA

Abstract

Compression of complex data sources, like video, images and audio, requires sophisticated source modeling and coding methods that extensively use conditional contexts. Fortunately, in many practical cases the high computational cost of estimating a large number of conditional probabilities and then determining optimal codes can be avoided by storing sets of codewords, and selecting the best based on local source estimates. Golomb-Rice codes are commonly used for such purpose because of their convenient features: they are parameterized with a single integer, are easy to implement, and are optimal for sources with geometric distribution, which represent a good approximation of those found in practice.

In this work we consider the fact that the Golomb-Rice codes are truly optimal only when the source's single parameter is known with certainty, which in practice is never the case. We investigate how uncertainty degrades the expected performance of this type of codes, depending on how the source is estimated from previous samples. Next, we analyze how to change the codes to increase their robustness, while keeping the useful structural properties, and propose a new class of codes, called *unary-stem* codes. We maintain the unary+binary structure, but generalize it in a way that the codewords are parameterized by several integers. While this is similar to other proposals for modifications of Golomb-Rice codes, it is quite more general, and allows us to better analyze the changes required for increasing robustness.

The new codes can be designed by constantly updating the maximum-likelihood or Bayesian estimation, while preserving the unary part, or by designing optimal unary-stem codes using a posterior distribution. We show how data for modeling the source uncertainty can be accurately computed, and develop two algorithms, capable of dealing with the infinite number of symbols, to quickly find optimal unary-stem codes. They are based on dynamic programming and on implicit enumeration, using some convenient upper and lower bounds on expected bit rates.

Numerical results show that the codes that do not take into account the source uncertainty in their design can be much less efficient than those that do. In fact, we identify the conditions that are ideal for a popular type of robust code ("Exp-Golomb"), and show that in those conditions Golomb codes produce infinite bit rates. The results also show that the optimal codes may be significantly better than the codes designed using only estimates of the source parameters, but that depends on the amount of information available, and the difference is smaller than that incurred by ignoring the uncertainty. This confirms the necessity of integrating source estimation into the design of parameterized codes, and the usefulness of identifying mappings directly from source samples to code parameters.

Contents

Notation

Partial list of the notation used throughout this document. In brackets we provide a reference to the section where the definition is.

$p(s)$ – general probability distribution [Section 2.1].

$\mathcal{P} = \{p(s)\}_{s=0}^{\infty}$ – sequence of symbol probabilities [Section 2.1].

$F(s)$ – general cumulative distribution [Section 2.1].

$\bar{F}(s)$ – reverse cumulative distribution [Section 2.1].

$\gamma(s, m)$ – number of bits used by symbol s in the binary code with m symbols [Section 2.3].

$\tau(m)$ – function for determining the number of bits in a codeword [Section 2.3].

ρ – single parameter of the geometric distribution [Section 2.2].

α – single parameter of the discrete Cauchy (DC) distribution [Section 4.3].

$p_g(s, \rho)$ – probability of symbol s in a source with geometric distribution [Section 2.2].

$p_c(s, \alpha)$ – probability of symbol s in a source with discrete Cauchy distribution [Section 4.3].

$p_m(s, f)$ – posterior probability of symbol s in a geometric distribution source where parameter ρ has pdf $f(\rho)$ [Section 4.2].

m – single parameter of the Golomb code [Section 2.3].

k – single parameter of the Golomb-Rice [Section 2.4], or Elias-Teuhola codes [Section 2.5].

$\mathcal{M} = \{m_d\}_{d=0}^{\infty}$ – parameters of the unary-stem code [Section 3.1].

$\mathcal{K} = \{k_d\}_{d=0}^{\infty}$ – parameters of the dyadic unary-stem code [Section 3.1].

$L_U(d, \mathcal{M})$ – cumulative sum of the symbols in a unary-stem code [Section 3.1].

$L_D(d, \mathcal{K})$ – cumulative sum of the symbols in a dyadic unary-stem code [Section 3.1].

$L(d)$ – simplified version of $L_U(d, \mathcal{M})$ or $L_D(d, \mathcal{K})$ [Section 3.2].

N_s – number of symbol values (source samples) available for source estimation [Section 4.1].

Σ_s – sum of the N_s of symbol values, used for source estimation [Section 4.1].

Chapter 1

Introduction

1.1 Motivation

The best compression methods for complex data sources, like images and audio, employ sophisticated source modeling. For optimal coding it is essential to have good estimates of the relative frequency of the data symbols, which are used to compute the optimal number of bits to be used per coded symbol. However, the statistical properties of these sources change frequently and in ways that are hard to predict, and a single set of statistics does not yield the best results. Consequently, there is a difficult compromise between reliable estimation of source parameters, the speed in which the coding parameters are changed (adaptation), and the computational complexity.

For instance, Figure 1.1(a) shows a typical natural image, and Figure 1.1(b) shows the distribution of the magnitude of residuals of the MED predictor, which is commonly used for lossless image coding [12, 25]. Figure 1.1(b) is visually similar to Figure 1.1(a) because each object has a different texture, which creates a group of residual samples with similar distribution. Figure 1.1(c), which is the enlargement of a part of Figure 1.1(b), shows that this occurs at various scales.

Trying to fully separate the similarly distributed source samples (full segmentation) is too complex, but it can be replaced with coding contexts—a neighborhood analysis for best code selection—which yield better compression at much lower complexity [8, 10, 18, 25]. Figure 1.2 shows a diagram of this type of coding system. This approach can still be computationally expensive because the source information (probability estimates) has to be translated into the data that is actually used for coding. For example, prefix codes need tables of codewords, and arithmetic codes need cumulative distributions [32, 35]. Depending on the implementation, the computational effort required for these tasks can grow quickly with the alphabet size, and it is impractical to have this type of adaptive coding with large alphabets. In addition, the number of source samples required to obtain good probability estimates also grows considerably with alphabet size, degrading compression. On the other

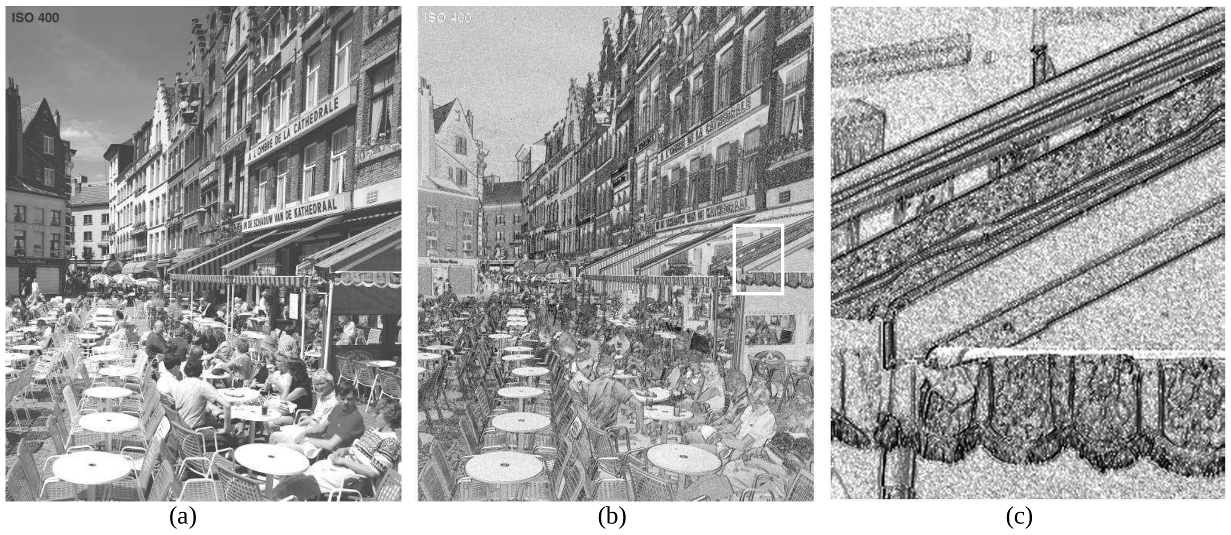


Figure 1.1: (a) a typical natural image; (b) the distribution of prediction residual magnitudes, with gray value equal to $255 - 50 \ln(1 + |p - \hat{p}|)$, making larger errors darker; (c) magnification of the indicated area of (b).

hand, high data throughputs can only be achieved on modern general-purpose processors using large alphabets [34].

When there is *a priori* knowledge about the typical type of probability distributions, the coding complexity can be greatly reduced by combining all the modeling tasks in a much simpler process. Instead of estimating all probabilities and then computing the optimal codes, it is possible to store sets of codewords, and to select the best choice based on estimates of a few source parameters. Figure 1.3 shows a diagram with this simplified coding system. Note that data about code choices can be incorporated as side information, but this is not always necessary, and depends on the type of adaptation. Even when there is only a loose fit to the assumed distribution, the loss in compression can be reasonably small, and can be justified by the significant reduction in complexity.

In order to deal with very large alphabets, it is also convenient to use codes that can be easily generated while encoding and decoding. This way, a reasonably small number of the most commonly used codewords is stored in tables, while the less common codewords are generated only when needed. The class of *Golomb-Rice* codes became quite popular for practical applications [16, 25, 27, 28] because they satisfy these requirements, and their structure allow very fast encoding and decoding.

Discrepancies between observed and the geometric symbol distribution motivated several proposals for modifications of Golomb-Rice codes, which maintain their convenient structure of unary and binary components (cf. Section 2.3), but change the sets of codewords. An early example of such proposal is the set of “Exp-Golomb” codes [7], which also became popular and is now used in the H.264 video coding standard [30].

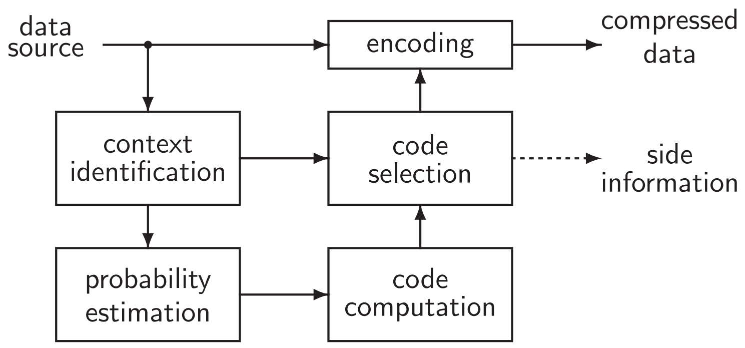


Figure 1.2: System for data compression based on estimation of conditional probabilities.

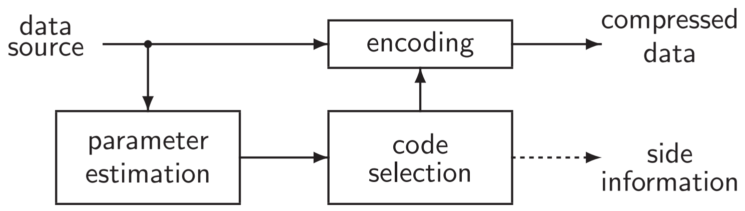


Figure 1.3: System for data compression using parameterized codes.

While some of these code modifications are meant only to create somewhat better codes for particular sources and applications, the *ad hoc* approach is quite unsatisfactory. There are important issues related to the use of these codes that are still unresolved:

- Golomb-Rice codes are truly optimal only if the source parameter is known with certainty, which in practice is never the case. For adaptive coding the source uncertainty must be taken into account when evaluating their performance.
- The Elias-Teuhola ("Exp-Golomb") codes are much more resilient to source uncertainty, but were initially proposed for situations in which no information about the source is known, which is a very special case.
- In the simplified coding system shown in Figure 1.3 we want to estimate codes directly from source samples. This problem is not exactly the same as estimating some predefined distribution parameters, and then selecting the best code corresponding to those parameters.

This last point corresponds to the fact that, given a set of observations, the estimated symbol probabilities are not necessarily equal to the probabilities that are defined by estimated source parameters. Thus, in Figure 1.3 we basically need to find out which code is probably the best (estimate code parameters), and not necessarily estimate source parameters.

However, the estimation of distribution parameters has been very extensively studied, since it is a fundamental problem in statistics, while much less is known about estimating optimal parameterized codes. Thus, we feel that there is a need to study in more depth, and in a combined manner, the problems of source variability and the design of parameterized codes, in order to simultaneously obtain efficient codes and address all the issues mentioned above.

In order to facilitate the analysis, our approach is to consider some scenarios that are simple enough to provide intuitive understanding of the results, and at the same time maintain the essential properties of the practical problem. Thus, we maintain the assumption of a source with geometric distribution, but also assume that its parameter changes constantly, so that the exact distribution is unknown, and can only be estimated from a few past samples.

The advantage of this approach is that it provides better insight into the optimal adaptive coding problem. Since the source is not stationary, it should be modeled assuming a mixture of distributions, and the code design should use this model. It is intuitive to assume that, even if the source distribution is geometric, the uncertainty of the source parameters produces a posterior distribution that is not geometric. However, if we use only experimental data to find out the best codes for a certain type data source with a particular form of contexts, then nothing is learned about the source, and the experiments have to be repeated whenever the estimation (coding contexts) change.

Using this statistical framework we consider the problem of finding optimal codes that maintain the useful properties of Golomb-Rice codes, and perform better with varying sources. Note that the purpose here is not really to define specific "new" and better codes for specific applications, but to propose a new design methodology and to evaluate how the structure of the optimal codes vary with the uncertainty on the source parameters.

1.2 Prior Work

In general terms, any form of coding (not necessarily optimal) that switches sets of codewords according to some context can be considered an implementation of the simplified coding system of Figure 1.3. Thus, we can say that this form of coding had been used even before Shannon introduced the main concepts of information theory that led to the optimized entropy-coding systems of Figure 1.2. This makes it extraordinarily difficult to trace the development of these coding methods, including some important hybrid approaches.

While the idea of using parameterized is certainly not new, technological advances allow coding implementation to use this technique in a much more massive scale. For instance, we currently have

- Embedded processors that allow great flexibility in dealing with different types of coding (e.g., simultaneous use of arithmetic coding and prefix codes in multiple data streams), and real-time processing and generation of codewords;

- Low cost coding systems that use thousands of different contexts, each defining a different code.

The change can be seen in the evolution of the image compression standards. The JPEG standard [14] has a baseline mode with simple Huffman coding that uses up to four set of codewords, and an arithmetic coding mode that uses many more contexts and models. However, due to complexity and implementation costs only the baseline mode had been widely adopted. For lossless image compression, it was later recognized that low complexity and better compression could be achieved, resulting in the creation of the JPEG-LS standard [25].

The JPEG-LS standard is based on a significant amount of work on combined coding and source estimation, and the determination of optimal codes for sources with infinite alphabets done by Merhav, Seroussi, and Weinberger [16, 17, 22, 23, 25]. In fact, they provide solutions to many of the practical problems discussed in this work. One important difference is that we study a wider choice of codes and adaptation scenarios, and use numerical optimization methods more extensively to compute distributions and their optimal codes.

A number of authors studied the application of parameterized codes for quantized values of sources with the Generalized Gaussian (GG) distribution. For instance, Wen and Villasenor [21] studied the performance of Golomb-Rice and Elias-Teuhola codes on such sources, providing bounds on redundancy, and showing that, as expected, the Elias-Teuhola codes are significantly more efficient for distributions with slow decay rate. Xue *et al.* [31] propose a modification of the code structure for improved performance on those sources, which can be interpreted as a code that is a “hybrid” of Golomb-Rice and Elias-Teuhola codes.

It is also interesting to note that some of the ideas related to parameterized prefix codes had been employed in the development of “universal codes” [33], i.e., those that can be used without any prior knowledge of the source. The ability to give good compression in any case is related to the robustness aspect of parameterized codes.

1.3 Organization and Contributions

Since a good amount of mathematical notation has to be defined before we can start presenting the new results and the discussions about adaptive coding with parameterized codes, in Chapter 2 we use the fact that a very natural way to introduce the notation is to start presenting the properties of the simpler Golomb, Golomb-Rice, and Elias-Teuhola (“Exp-Golomb”) codes. Note that we include some very basic material, like tables with examples of codewords, that should help readers that are learning about these codes for the first time, but we also present some facts and properties that are not so well-known, because they are needed later. Thus, we encourage even the readers that are very familiarized with these codes to quickly look at Chapter 2 to learn the notation.

We try to keep the notation consistent throughout the documents. In page 3 we have a list of the functions, variables and parameters that are more important, together with references to their definitions.

The new results begin to be introduced in Chapter 3. Before analyzing the effects of source uncertainty, we propose the class of *unary-stem codes,* which have the unary+binary structure, but are parameterized by an infinite sequence of positive integers. This class contains not only the Golomb, Golomb-Rice and Elias-Teuhola codes, but also many other variations [21, 22]. To the best of our knowledge, no such generalization has been proposed and studied before. In Section 3.1 we define these codes and present some examples, and in Section 3.2 we derive the formulas for computing the average bit rate of any unary-stem code on any data source.

While unary-stem codes are maybe a little too general for practical use,[1] the advantages of such flexibility become apparent when in Chapter 4 we start analyzing the source estimation problem, and strategies for adaptive coding. In Section 4.1 we use a very simple analysis, based on maximum-likelihood estimation, to show that in the presence of source uncertainty there is a clear need to allow more flexibility in the parameterization of the codes, and that the unary-stem class is an attractive solution.

A comparison with Bayesian source estimation is presented in Section 4.2. While a main theoretical difficulty regarding Bayesian methods is in the choice of prior distributions, we show that in our problem the particular choice is not so much a problem, since the context available from past samples can dominate the estimation process. In Section 4.3 we consider one particular type of posterior distribution, which we show that can be considered the "natural match" to Elias-Teuhola codes. At this point, our results show that

- The Golomb and Golomb-Rice codes are optimal only if the prior distribution is an impulse, i.e., the geometric distribution parameter is constant and is known with good accuracy.
- The Elias-Teuhola codes are optimal if the prior distribution is uniform over all the possible values for the source parameter, and no previous source samples are available.

The surprising conclusion is that the two most commonly used types of codes are in fact optimal for two extreme scenarios, not expected to be typically found in practice.

Furthermore, we show that the difference in coding performance, as we move in the direction of one or the other of these two extremes, is definitely not symmetrical. For instance, for sources with geometric distribution the redundancy of the best Elias-Teuhola codes can be about ten times larger than the redundancy of the best Golomb-Rice codes. While this difference is quite significant, the relative redundancy of the Elias-Teuhola codes is typically below 10%. On the other hand, if we use Golomb-Rice codes on the source conditions that are matched to Elias-Teuhola codes we obtain infinite redundancy! While such conditions

[1]Too general in the sense that in practice we do not need to define an infinite sequence of code parameters. Nearly the same results can be obtained with a relatively small number of parameters, or simple rules for generating the parameter sequence [36].

are not very common, it is a very convincing argument for using parameterized codes that are robust.

In Chapter 5 we study how to find the optimal unary-stem codes for a given symbol distribution. The approach here is quite general, since there is no need to constrain the analysis to only the distributions from Chapter 4. We discuss the technical difficulties resulting from the fact that the codewords of the unary-stem codes must have a special structure, and that there is an infinite number of symbols. While is possible to use the Huffman algorithm to find optimal general codes for increasingly larger number of symbols, it cannot be modified to create codewords only in the unary-stem form.

We propose two new algorithms for finding the optimal codes, one based on implicit enumeration, and the other using dynamic programming. In Section 5.2 we modify the formulas derived in Section 3.2 in order to find the bit rates defined by partial choices of unary-stem codes, and that also can be efficiently computed while searching for the optimal codes. Section 5.1 presents some optimality conditions, and Sections 5.3 to 5.5 describe the optimization algorithms.

Numerical results are presented and discussed in Chapter 6. We apply the algorithms proposed in Chapter 5 to find the optimal unary-stem codes for the distributions defined in Chapter 4. We consider a variety of scenarios, changing the prior distributions for the geometric source parameter, and also different number of observed source samples (coding contexts). Since these results define hundreds of codes, with parameters and performance listed in more than eighty tables, we present only the main results in Chapter 6, while the tables showing the parameters of the codes are shown in Appendix C.

The numerical results confirm the hypothesis that the codes that do not take into account the source uncertainty in their design can be much less efficient than those that do. They also show that the optimal codes may be significantly better than the codes designed using only estimates of the source parameters, but that depends on the amount of context information available, and the difference is smaller than that incurred by ignoring the uncertainty. One surprising result is that the difference in performance was actually higher for a prior that represents typical coding (modeled from empiric observations) than for priors that were expected to represent worst-case scenarios.

In Chapter 7 we discuss the main conclusions, and how the experiments confirm the necessity of integrating source estimation into the design of parameterized codes, and the usefulness of identifying mappings directly from source-samples to code parameters.

Appendixes A and B present methods for efficient and accurate numerical computation of some integrals and series, which are needed for determining some of the source parameter estimates, probabilities, and entropy values, defined in Chapter 4.

Chapter 2

Definitions and Notation

2.1 Data Sources

We assume that we have a data source with infinite alphabet $\mathcal{A} = \{0, 1, 2, \ldots\}$, the probability of the data symbols are

$$p(s) = \text{Prob}(S = s), \quad s = 0, 1, 2, \ldots \tag{2.1}$$

and we represent the whole sequence of probabilities by

$$\mathcal{P} = \{p(s)\}_{s=0}^{\infty}. \tag{2.2}$$

We call $\mathcal{P}$ *monotonic* if

$$p(i) \geq p(i+1), \quad i = 0, 1, 2, \ldots. \tag{2.3}$$

and call $\mathcal{P}$ a *decreasing* distribution when we have strict inequalities.

The cumulative probability distribution is represented by

$$F(s) = \sum_{i=0}^{s-1} p(i), \quad s = 0, 1, 2, \ldots \tag{2.4}$$

while the the reverse cumulative distribution is

$$\bar{F}(s) = \sum_{i=s}^{\infty} p(i) = 1 - F(s), \quad s = 0, 1, 2, \ldots \tag{2.5}$$

From these definitions we have the identities

$$\sum_{s=m}^{n-1} p(s) = F(n) - F(m) = \bar{F}(m) - \bar{F}(n), \quad 0 \leq m \leq n. \tag{2.6}$$

While $F(s)$ and $\bar{F}(s)$ are in a sense mathematically equivalent, and $F(s)$ is conventionally used in textbooks, we prefer to use $\bar{F}(s)$, since it is more commonly used in practical computations,[1] and it is convenient to have formulas that are the same as those used for the

[1]It has a more precise floating-point representation, since it is a sum that starts with the smallest numbers.

numerical tests.

The entropy of the source is represented as

$$H(\mathcal{P}) = \sum_{s=0}^{\infty} p(s) \log_2\left(\frac{1}{p(s)}\right). \tag{2.7}$$

and partial entropy sums are defined as

$$\begin{aligned} h(s) &= \sum_{i=0}^{s-1} p(i) \log_2\left(\frac{1}{p(i)}\right), \quad s = 0, 1, 2, \ldots & (2.8) \\ \bar{h}(s) &= \sum_{i=s}^{\infty} p(i) \log_2\left(\frac{1}{p(i)}\right) = H(\mathcal{P}) - h(s), \quad s = 0, 1, 2, \ldots & (2.9) \end{aligned}$$

Particular probability distributions are represented by using subscripts for identification, and extra variables that characterize the distribution, in the form $p_t(s, \cdot, \cdot, \cdots)$.

2.2 Geometric Distribution Sources

Geometric distribution data sources are defined by a single parameter $\rho \in (0, 1)$, and have their symbol probabilities defined by

$$\text{Prob}(S = s) = p_g(s, \rho) = (1 - \rho)\, \rho^s, \quad s = 0, 1, 2, \ldots \tag{2.10}$$

and the cumulative distributions are

$$F_g(s, \rho) = 1 - \rho^s, \quad \bar{F}_g(s, \rho) = \rho^s. \tag{2.11}$$

For this distribution the average value of a data symbol is

$$\text{E}\{S\} = \bar{s} = \sum_{s=0}^{\infty} s\, p_g(s, \rho) = \frac{\rho}{1 - \rho}, \tag{2.12}$$

while the entropy, in bits/symbol, is

$$H_g(\rho) = -\sum_{s=0}^{\infty} p_g(s, \rho) \log_2(p_g(s, \rho)) = -\log_2(1 - \rho) - \frac{\rho}{1 - \rho} \log_2(\rho). \tag{2.13}$$

The entropy can also be computed using $\bar{s}$ as

$$H_g(\bar{s}) = \log_2(\bar{s} + 1) + \bar{s}\, \log_2(1 + 1/\bar{s}). \tag{2.14}$$

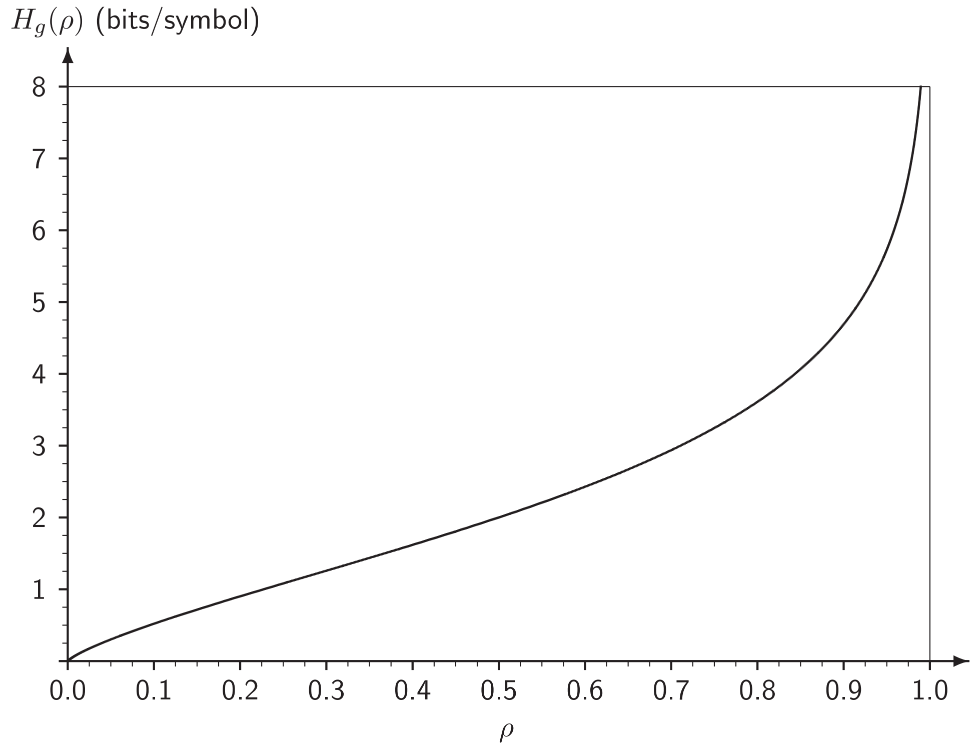


Figure 2.1: Entropy of the geometric distribution as a function of parameter ρ.

Note that the optimal number of bits required for coding data symbols from a geometric distribution source,

$$B_g^*(s,\rho) = \log_2\left(\frac{1}{p_g(s,\rho)}\right) = -s\log_2(\rho) - \log_2(1-\rho), \quad s = 0,1,2,\ldots \tag{2.15}$$

is a linear function of the symbol number s.

Figure 2.1 shows how the source entropy varies with parameter ρ. Since prefix codes are normally used only for sources with entropy larger than 1 bit/symbol, we typically have $\rho > 1/4$. Figure 2.2 shows that in this range we have $H_g(\rho) \approx \log_2(e\,\bar{s})$, where e is the base of the natural logarithms.

2.3 Golomb Codes

Golomb proposed a family of prefix codes for geometric distribution sources [2, 32]. Each *Golomb code* is defined uniquely by a positive integer m, and the process of coding a data symbol s is divided in two steps. First, the quotient value

$$d = \left\lfloor \frac{s}{m} \right\rfloor, \tag{2.16}$$

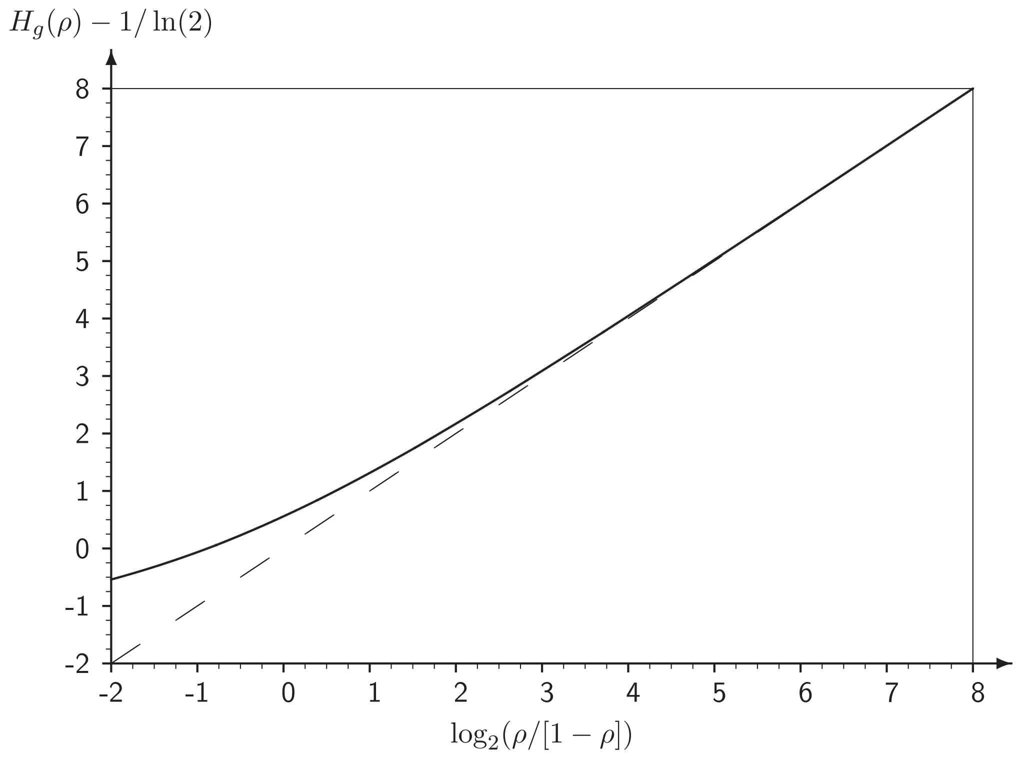


Figure 2.2: Nearly linear relation between $H_g(\rho)$ and $\log_2(\bar{s})$.

is coded using a unary code, i.e., using d b-bits followed by a $\bar{b}$-bit (we adopt the convention of d 1-bits, followed by a 0-bit). Next, the remainder

$$r = s - md, \quad 0 \le r < m \tag{2.17}$$

is coded using an integer number of bits equal to $\lfloor \log_2(m) \rfloor$ or $\lceil \log_2(m) \rceil$.

For each data symbol s, we use $\mathcal{W}_G(s, m)$ to represent the corresponding binary codeword of the Golomb code with parameter m, and $\ell_G(s, m)$ to denote the number of bits in that codeword (codeword length). Table 2.1 shows examples of codewords, and respective number of bits, defined by the first eight Golomb codes (the parameter k is explained in the next section). Note how the codeword lengths grow regularly. This property enables the number of bits in the Golomb codewords to approximate the linear growth of the optimal codeword length, as shown in (2.15).

Figure 2.3 shows the trees representing these prefix codes. Note that we use a different tree graphical representation in order to clearly identify the two parts of the Golomb codes. On the top we have the unary code, and we keep the 1-bit tree branches horizontal to show more clearly the endless repetition of the trees corresponding to the binary representation of the remainders.

Some special notation simplifies the equations to calculate bit rates. First, we define

$$\tau(m) = 2^{\lceil \log_2(m) \rceil} - m, \quad m = 1, 2, \ldots \tag{2.18}$$

Table 2.1: Examples of codewords $\mathcal{W}_G(s,m)$ defined by the first eight Golomb codes, and their corresponding lengths $\ell_G(s,m)$ in bits.

Symbol	Codeword	Bits	Codeword	Bits	Codeword	Bits	Codeword	Bits
s	$m=1,\ k=0$		$m=2,\ k=1$		$m=3$		$m=4,\ k=2$	
0	0	1	00	2	00	2	000	3
1	10	2	01	2	010	3	001	3
2	110	3	100	3	011	3	010	3
3	1110	4	101	3	100	3	011	3
4	11110	5	1100	4	1010	4	1000	4
5	111110	6	1101	4	1011	4	1001	4
6	1111110	7	11100	5	1100	4	1010	4
7	11111110	8	11101	5	11010	5	1011	4
8	111111110	9	111100	6	11011	5	11000	5
9	1111111110	10	111101	6	11100	5	11001	5
10	11111111110	11	1111100	7	111010	6	11010	5
11	111111111110	12	1111101	7	111011	6	11011	5
12	1111111111110	13	11111100	8	111100	6	111000	6
13	11111111111110	14	11111101	8	1111010	7	111001	6
14	111111111111110	15	111111100	9	1111011	7	111010	6
15	1111111111111110	16	111111101	9	1111100	7	111011	6
s	$m=5$		$m=6$		$m=7$		$m=8,\ k=3$	
0	000	3	000	3	000	3	0000	4
1	001	3	001	3	0010	4	0001	4
2	010	3	0100	4	0011	4	0010	4
3	0110	4	0101	4	0100	4	0011	4
4	0111	4	0110	4	0101	4	0100	4
5	1000	4	0111	4	0110	4	0101	4
6	1001	4	1000	4	0111	4	0110	4
7	1010	4	1001	4	1000	4	0111	4
8	10110	5	10100	5	10010	5	10000	5
9	10111	5	10101	5	10011	5	10001	5
10	11000	5	10110	5	10100	5	10010	5
11	11001	5	10111	5	10101	5	10011	5
12	11010	5	11000	5	10110	5	10100	5
13	110110	6	11001	5	10111	5	10101	5
14	110111	6	110100	6	11000	5	10110	5
15	111000	6	110101	6	110010	6	10111	5
16	111001	6	110110	6	110011	6	110000	6

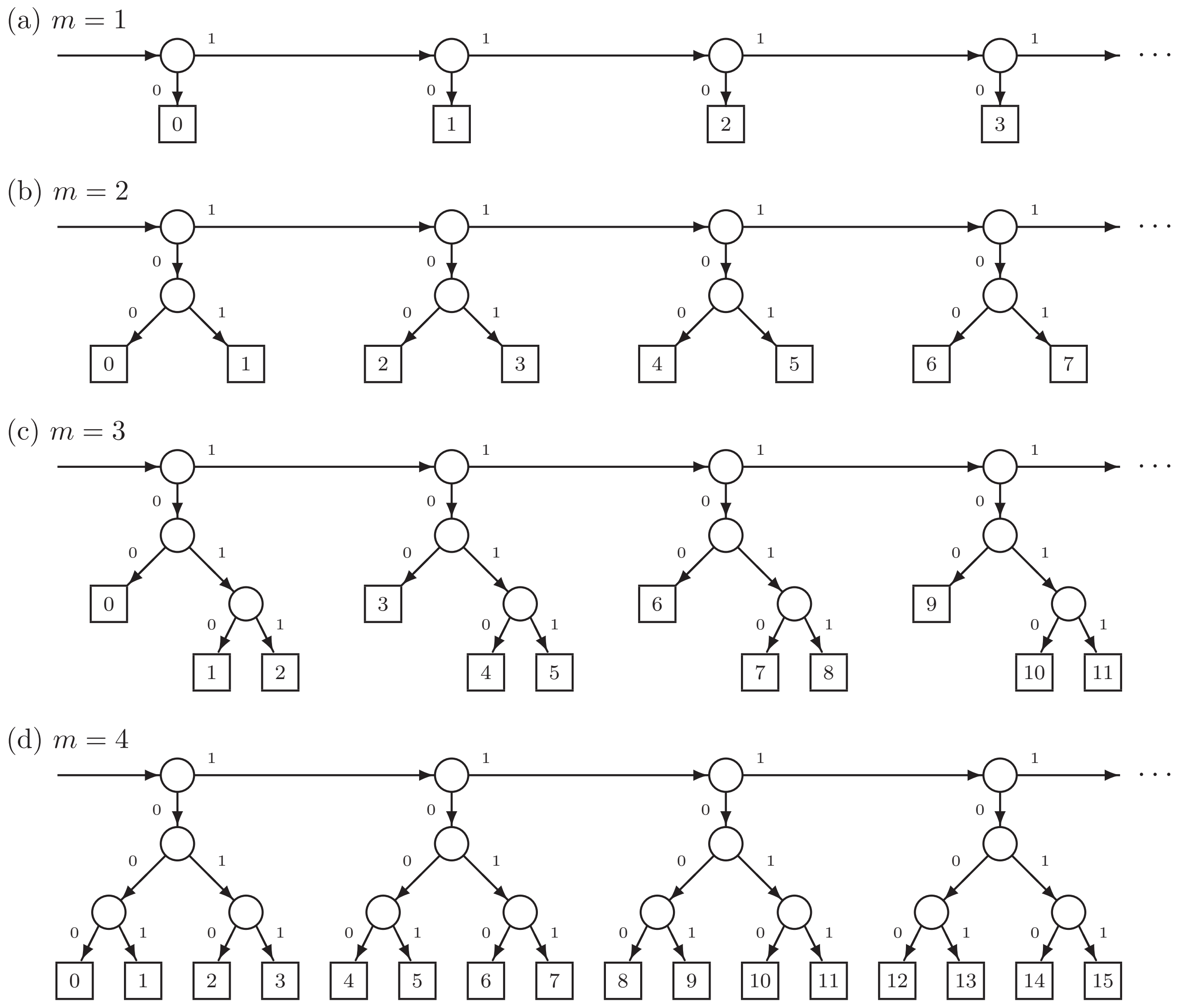


Figure 2.3: Trees corresponding to the first four Golomb codes. The branches in the unary-code part are in the horizontal direction, while the binary part is shown with the usual representation.

and

$$\gamma(s,m) = \begin{cases} \lfloor \log_2(m) \rfloor, & 0 \le s < \tau(m), \\ \lceil \log_2(m) \rceil, & \tau(m) \le s < m. \end{cases} \tag{2.19}$$

Table 2.2 shows some values of $\tau(m)$.

Using this notation, we can compute the number of bits in a codeword using

$$\ell_G(s,m) = 1 + d + \gamma(r,m) = 1 + \left\lfloor \frac{s}{m} \right\rfloor + \gamma\Big(s - m \left\lfloor \frac{s}{m} \right\rfloor, m\Big). \tag{2.20}$$

and compute the average number of bits per coded symbol using

$$\begin{aligned} B_G(m,\rho) &= \sum_{s=0}^{\infty} \ell_G(s,m)(1-\rho)\,\rho^s \\ &= 1 + (1-\rho) \sum_{d=0}^{\infty} \sum_{i=0}^{m-1} [d + \gamma(i,m)]\,\rho^{md+i} \\ &= \frac{1 + \left(1 - \rho^{\tau(m)}\right) \lfloor \log_2(m) \rfloor + \left(\rho^{\tau(m)} - \rho^m\right) \lceil \log_2(m) \rceil}{1 - \rho^m} \\ &= \lceil \log_2(m) \rceil + \frac{1 - \left(1 - \rho^{\tau(m)}\right) \left(\lceil \log_2(m) \rceil - \lfloor \log_2(m) \rfloor\right)}{1 - \rho^m} \end{aligned} \tag{2.21}$$

Gallager and Van Voorhis proved two important results about Golomb codes [5]. First, for any geometric distribution with parameter ρ the optimal Golomb code (with minimum average codeword length) has parameter m satisfying

$$\rho^m + \rho^{m+1} \le 1 \le \rho^m + \rho^{m-1}, \tag{2.22}$$

so the optimal parameter is

$$m^*(\rho) = \operatorname*{argmin}_{m \ge 1} \{B_G(m,\rho)\} = \left\lceil \frac{\log(1+\rho)}{\log(\rho^{-1})} \right\rceil. \tag{2.23}$$

In addition, they proved that these Golomb codes are optimal among all infinite prefix codes. This result is particularly important because the Huffman algorithm [1, 26] cannot be used to find the optimal code when the source alphabet is infinite.

Figure 2.4 shows the relative redundancy obtained with optimal Golomb codes on a data source with geometric distribution. We can see that the redundancy is around 1% or lower for source entropies above 3 bits/symbol, which is quite good, considering the how easy it is to generate and use those codes. However, such good performance occurs only if the source distribution is indeed geometric and if its parameter ρ is known, to enable the computation of the optimal parameter m^*. This problem is analyzed in Section 2.5 and in Chapter 4.

Table 2.2: Some values of the function $\tau(m)$, required for the computation of codeword bits.

m	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$\tau(m)$	0	0	1	0	3	2	1	0	7	6	5	4	3	2	1	0	15	14	13	12

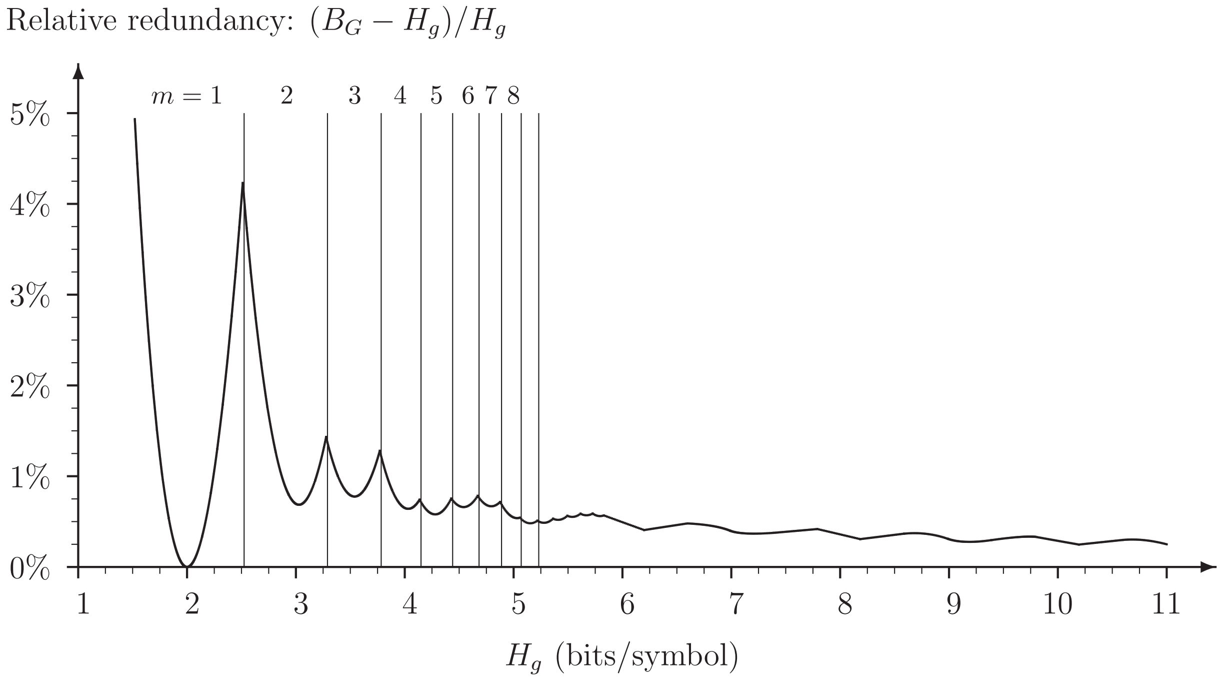


Figure 2.4: Relative coding redundancy of optimal Golomb codes as a function of the source entropy.

2.4 Golomb-Rice Codes

In the special cases when the Golomb codes have parameter $m = 2^k$ we can code all possible values of remainder r using exactly k bits, which is quite advantageous in practical applications. These particular Golomb codes are also called *Golomb-Rice codes*, and we represent their codewords as $\mathcal{W}_R(s,k)$, such that,

$$\mathcal{W}_R(s,k) = \mathcal{W}_G(s, 2^k), \quad k, s = 0, 1, 2, \ldots \tag{2.24}$$

We can see in Table 2.1 examples of codewords of the first four Golomb-Rice codes. In these special cases the number of bits in a codeword is

$$\ell_R(s,k) = 1 + k + d = 1 + k + \left\lfloor \frac{s}{2^k} \right\rfloor, \tag{2.25}$$

and the average number of bits used is

$$\begin{aligned} B_R(k,\rho) &= \sum_{s=0}^{\infty} \ell_R(s,k)(1-\rho)\,\rho^s \\ &= 1 + k + (1-\rho) \sum_{d=0}^{\infty} d \sum_{i=0}^{2^k-1} \rho^{2^k d + i} \\ &= k + \frac{1}{1-\rho^{(2^k)}}. \end{aligned} \tag{2.26}$$

Using (2.26) we conclude that the optimal coding parameter k is

$$k^*(\rho) = \underset{k \geq 0}{\operatorname{argmin}} \{B_R(k,\rho)\} = \max \left\{ 0, \left\lceil \log_2 \left(\frac{\log(\phi)}{\log(\rho^{-1})} \right) \right\rceil \right\}, \tag{2.27}$$

where $\phi = (\sqrt{5}+1)/2$ is the Golden Ratio.

Figure 2.5 shows the relative redundancy of the optimal Golomb-Rice codes, as a function of the source entropy H_g (eq. 2.13). Figure 2.6 shows the same data, but parameterized by the logarithm of the average symbol value $\bar{s}$ (eq. 2.12). In both cases we can see how the graphs are composed of segments with nearly periodic transitions. In fact, if we define

$$\vartheta(\rho) = \log_2 \left(\frac{\log(\phi)}{\log(\rho^{-1})} \right), \tag{2.28}$$

we can compute approximations to $\vartheta(\rho)$ using the source entropy as

$$\vartheta(\rho) \approx H_g(\rho) - 2.498, \tag{2.29}$$

or, using the average symbol value as

$$\vartheta(\rho) \approx \log_2(\bar{s}) - 0.05 + \frac{0.6}{\bar{s}}. \tag{2.30}$$

Figure 2.7 shows that for both approximations the error is quite small. Since the average symbol value is easier to compute, the optimal Golomb-Rice coding parameter k can be computed using

$$k^*(\bar{s}) = \max\left\{0, \left\lceil \log_2(\bar{s}) - 0.05 + \frac{0.6}{\bar{s}} \right\rceil \right\}. \tag{2.31}$$

2.5 Elias-Teuhola ("Exp-Golomb") Codes

While Golomb-Rice codes have great implementation and speed advantages, they are in a sense finely tuned to geometric distribution sources with parameters in a small range. It can be seen in Figures 2.5 and 2.6 that even when the distribution is geometric, the redundancy grows quickly if the source distribution is different from the optimal.

This fact motivated the search for codes that have a similarly convenient structure, but are more robust to changes in the source distribution. For this purpose Teuhola [7] proposed a set of new codes for run-lengths, parameterized by an integer k, similar to the Golomb-Rice parameter. In order to create the codeword corresponding to symbol s, first the value

$$d = \left\lfloor \log_2\left(1 + \frac{s}{2^k}\right) \right\rfloor, \tag{2.32}$$

is coded using a unary code, and it is followed by the $k+d$ bits with the binary representation of

$$r = s - 2^k(2^d - 1), \quad 0 \le r < 2^{k+d}. \tag{2.33}$$

Figure 2.8 shows the tree corresponding to the first of those codes ($k = 0$), and Table 2.3 shows the codewords of the first codes. Note how it is similarly formed by the unary code followed by binary codes, but the number of codewords in the binary part is always a power of two, and doubles after each bit in the unary code. For this reason Teuhola called these codes "Exp-Golomb" codes.

Elias [4] had proposed the same type of code (which he called γ' codes) a few years earlier, but using quite different assumptions, in the search for efficient "universal" integer representations. Since these codes are not truly Golomb codes, we prefer to call them *Elias-Teuhola codes.*

For these codes, the codeword length is

$$\ell_E(s,k) = 1 + k + 2d = 1 + k + 2\left\lfloor \log_2\left(1 + \frac{s}{2^k}\right) \right\rfloor, \tag{2.34}$$

and the average bit rate on a geometric distribution source is

$$B_E(k,\rho) \quad = \quad \sum_{s=0}^{\infty} \ell_E(s,k)(1-\rho)\,\rho^s \tag{2.35}$$

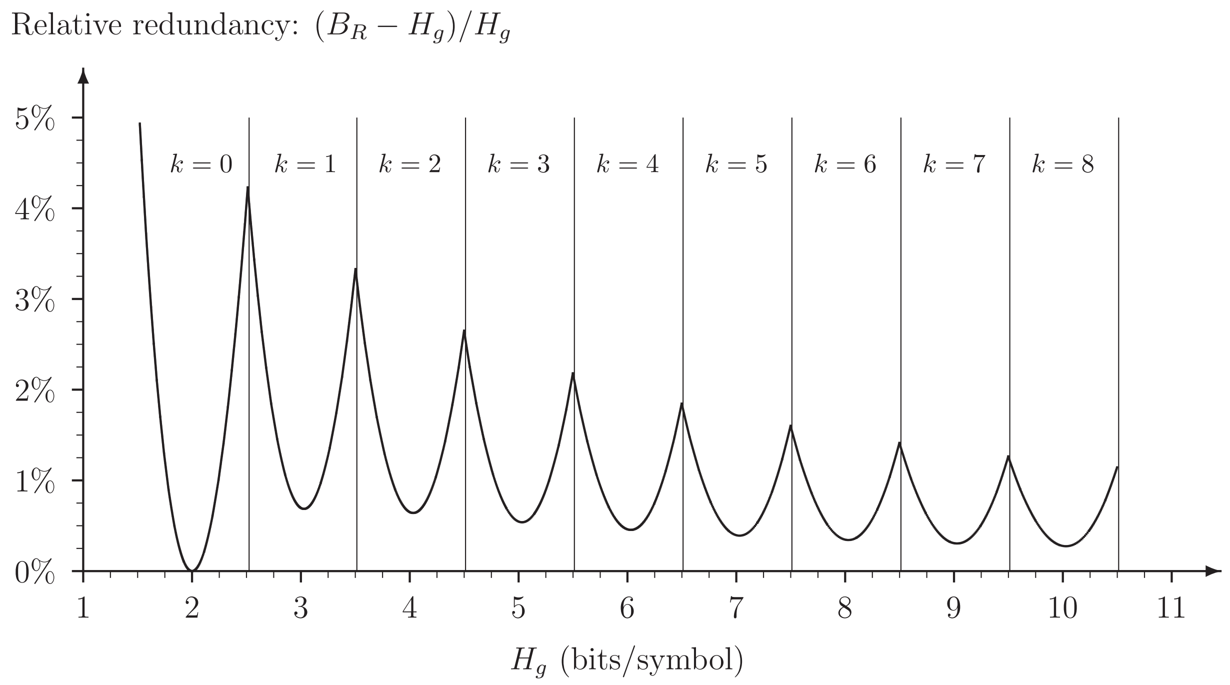


Figure 2.5: Relative coding redundancy of optimal Golomb-Rice codes as a function of the source entropy.

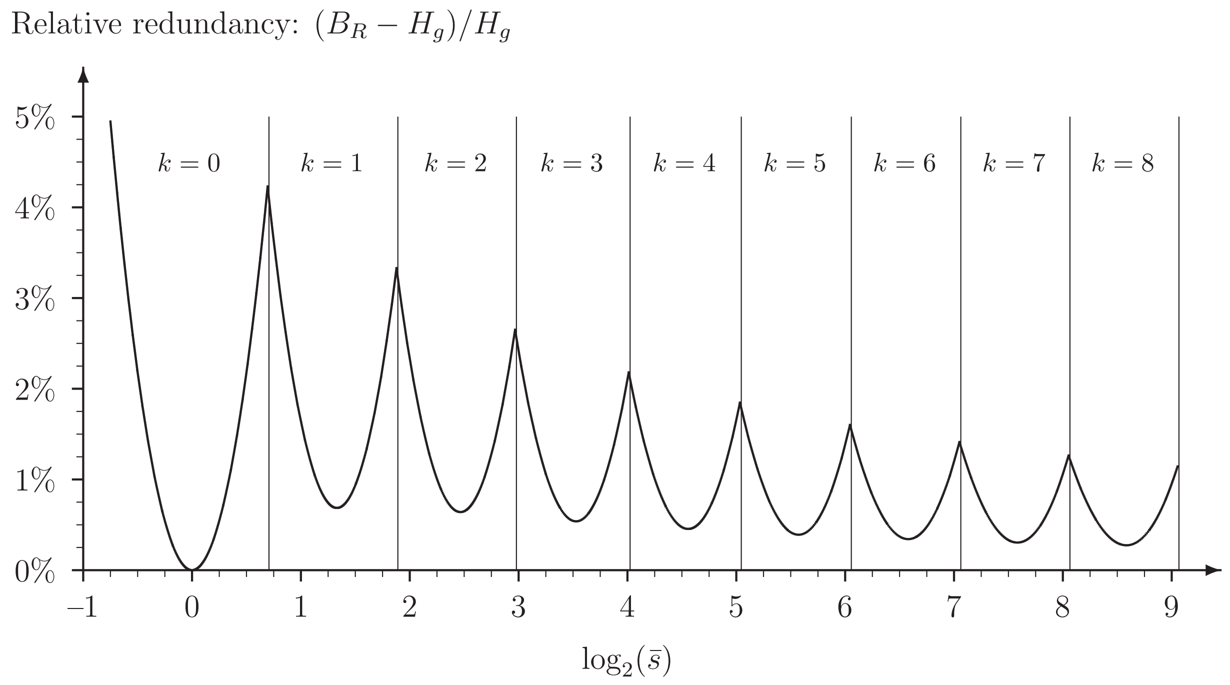


Figure 2.6: Relative coding redundancy of optimal Golomb-Rice codes as a function of the logarithm of the average symbol value.

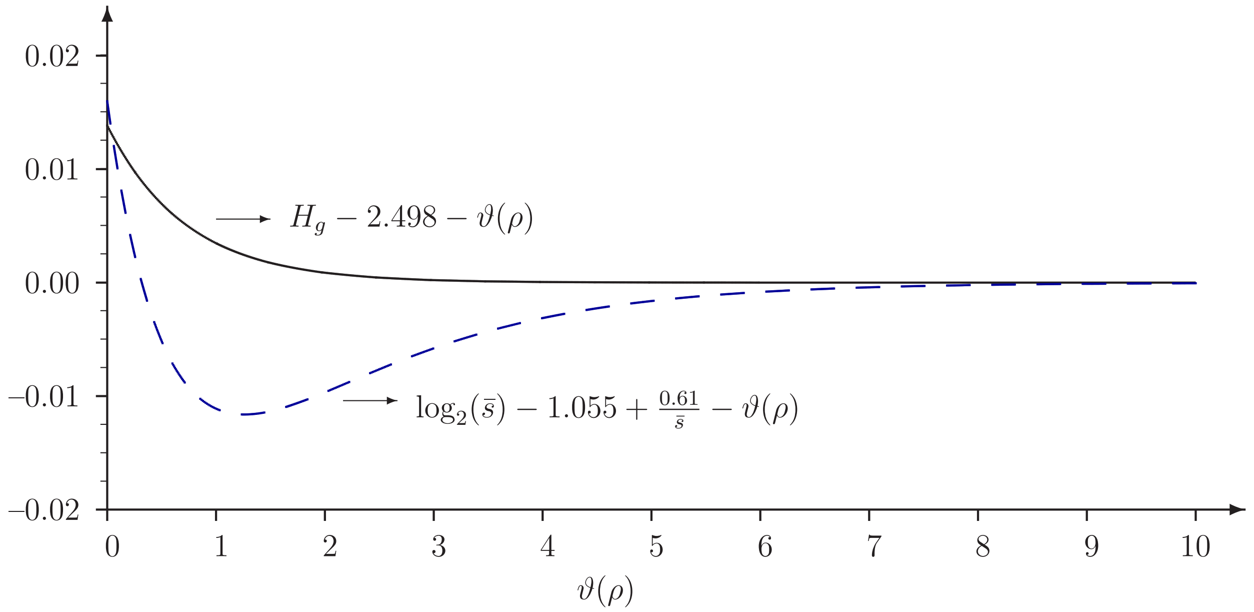


Figure 2.7: Error on approximations for computing optimal parameter k^* of Golomb-Rice codes.

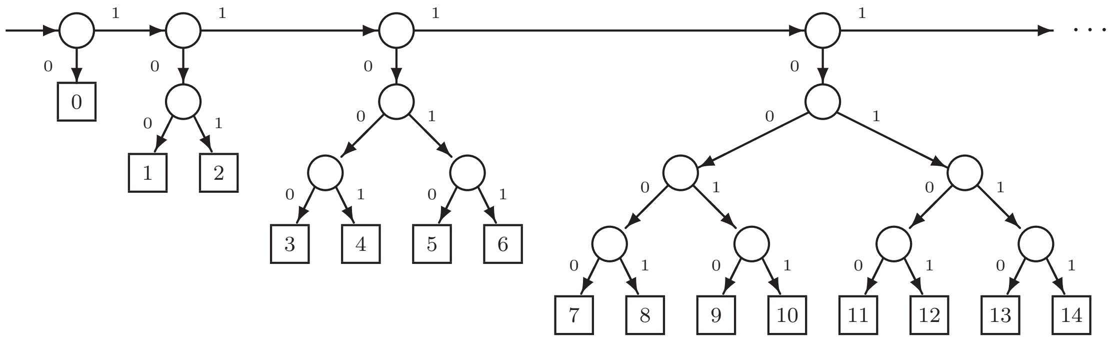


Figure 2.8: Tree corresponding to the first Elias-Teuhola code ($k = 0$).

Table 2.3: Examples of codewords $\mathcal{W}_E(s,k)$ defined by the first four Elias-Teuhola ("Exp-Golomb") codes, and their corresponding lengths $\ell_E(s,k)$ in bits.

Symbol	Codeword	Bits	Codeword	Bits	Codeword	Bits	Codeword	Bits
s	$k=0$		$k=1$		$k=2$		$k=3$	
0	0	1	00	2	000	3	0000	4
1	100	3	01	2	001	3	0001	4
2	101	3	1000	4	010	3	0010	4
3	11000	5	1001	4	011	3	0011	4
4	11001	5	1010	4	10000	5	0100	4
5	11010	5	1011	4	10001	5	0101	4
6	11011	5	110000	6	10010	5	0110	4
7	1110000	7	110001	6	10011	5	0111	4
8	1110001	7	110010	6	10100	5	100000	6
9	1110010	7	110011	6	10101	5	100001	6
10	1110011	7	110100	6	10110	5	100010	6
11	1110100	7	110101	6	10111	5	100011	6
12	1110101	7	110110	6	1100000	7	100100	6
13	1110110	7	110111	6	1100001	7	100101	6
14	1110111	7	11100000	8	1100010	7	100110	6
15	111100000	9	11100001	8	1100011	7	100111	6
16	111100001	9	11100010	8	1100100	7	101000	6
17	111100010	9	11100011	8	1100101	7	101001	6
18	111100011	9	11100100	8	1100110	7	101010	6
19	111100100	9	11100101	8	1100111	7	101011	6
20	111100101	9	11100110	8	1101000	7	101100	6
21	111100110	9	11100111	8	1101001	7	101101	6
22	111100111	9	11101000	8	1101010	7	101110	6
23	111101000	9	11101001	8	1101011	7	101111	6
24	111101001	9	11101010	8	1101100	7	11000000	8

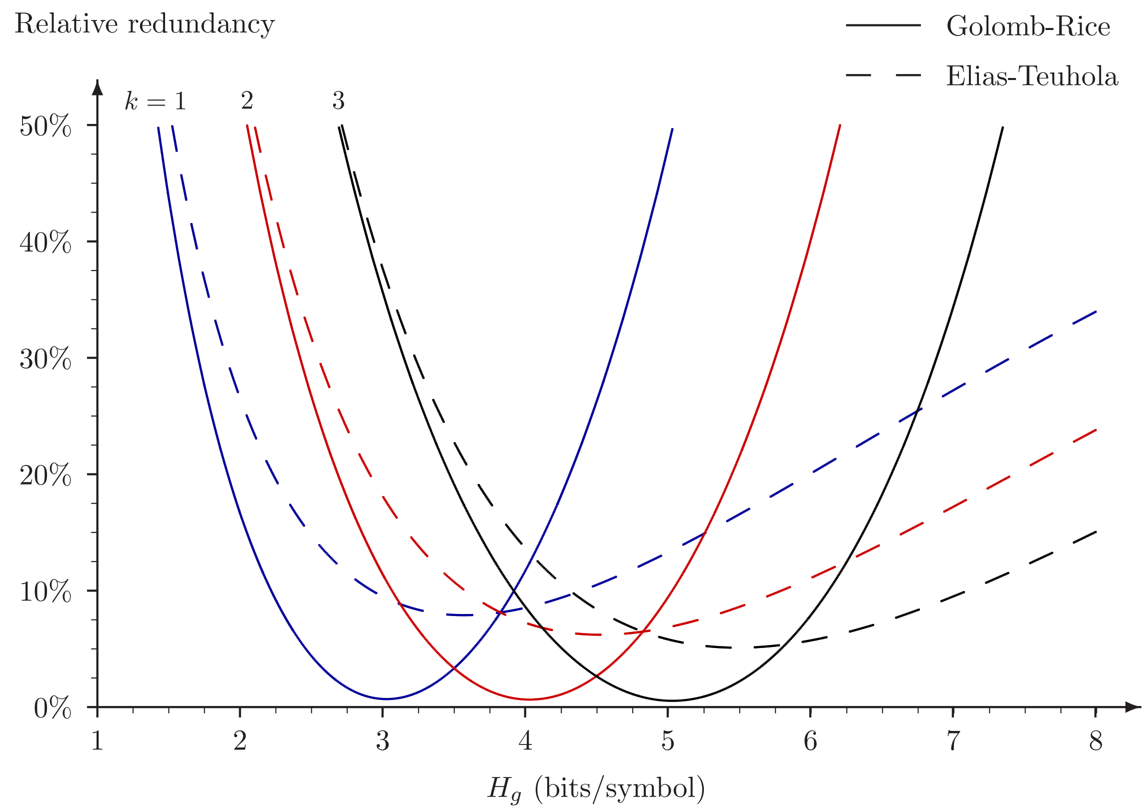


Figure 2.9: Relative coding redundancy of Golomb-Rice and Elias-Teuhola codes as a function of the geometric source entropy.

$$
\begin{aligned}
&= 1 + k + 2(1-\rho)\sum_{d=0}^{\infty} d \sum_{i=0}^{2^{k+d}-1} \rho^{2^k(2^d-1)+i} \\
&= 1 + k + 2\sum_{d=1}^{\infty} \rho^{2^k(2^d-1)} \\
&\approx 1 + k + 2\sqrt{\frac{\rho^{2^{k+1}}}{1-\rho^{3\cdot 2^{k-1}}}}.
\end{aligned}
$$

Figure 2.9 shows the redundancy of Elias-Teuhola codes compared with Golomb-Rice codes, on a source with geometric probability distribution. Note that the redundancy of the Elias-Teuhola codes does not get very near zero, but the flatter curves show that they are much more robust to uncertainty in the source distribution, especially when the code parameter k is underestimated.

2.6 General Representation of Infinite Prefix Codes

Since in practical coding applications we only have finite number of data symbols, the use of infinite alphabets and codes seems to be of theoretical interest only. However, it is very convenient to use codeword-generation rules that are valid for any nonnegative number, instead of having to worry about modifying alphabet sizes.

A useful notation to represent infinite codes was introduced by Golin [29]. An infinite binary tree corresponding to a code can be described by the sequence of ordered pairs $\mathcal{C} = \{(I_l, E_l)\}_{l=0}^{\infty}$, where $I_l \geq 0$ is the integer number of *internal nodes* and $E_l \geq 0$ is the number of *external nodes* or leafs, at depth l of the tree. Binary trees must satisfy the equations

$$\begin{aligned} (I_0, E_0) &= (1, 0), \\ I_l + E_l &= 2\, I_{l-1}, \quad l = 1, 2, \ldots \end{aligned} \tag{2.36}$$

A slightly different representation uses the cumulative sum of leaf nodes

$$T_l = \sum_{i=0}^{l} E_i, \tag{2.37}$$

so that infinite binary codes can be represented by the sequence $\mathcal{C} = \{(I_l, T_l)\}_{l=0}^{\infty}$, satisfying the equations

$$\begin{aligned} (I_0, T_0) &= (1, 0), \\ I_l + T_l &= 2\, I_{l-1} + T_{l-1}, \quad l = 1, 2, \ldots, \end{aligned} \tag{2.38}$$

and the inequalities

$$I_l \geq 1, \quad T_l \geq T_{l-1}, \quad l = 1, 2, \ldots \tag{2.39}$$

Tables 2.4 and 2.5 shows examples of how Golomb and Elias-Teuhola codes can be represented in these forms. The advantage of the second representation is that when symbol probabilities are defined by $\mathcal{P}$ (cf. Section 2.1), the average number of bits used by the code can be computed using the sum

$$B(\mathcal{C}, \mathcal{P}) = \sum_{l=1}^{\infty} \sum_{s=T_{l-1}}^{T_l - 1} l\, p(s). \tag{2.40}$$

Using the reverse cumulative distribution (2.5), the code rate can be computed using an even simpler formula:

$$B(\mathcal{C}, \mathcal{P}) = \sum_{l=0}^{\infty} \bar{F}(T_l). \tag{2.41}$$

Table 2.4: Examples of the compact representation (I_l, E_l) of infinite alphabet codes.

Level	Golomb						Elias-Teuhola		
l	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 6$	$m = 8$	$k = 0$	$k = 1$	$k = 2$
0	(1,0)	(1,0)	(1,0)	(1,0)	(1,0)	(1,0)	(1,0)	(1,0)	(1,0)
1	(1,1)	(2,0)	(2,0)	(2,0)	(2,0)	(2,0)	(1,1)	(2,0)	(2,0)
2	(1,1)	(2,2)	(3,1)	(4,0)	(4,0)	(4,0)	(2,0)	(2,2)	(4,0)
3	(1,1)	(2,2)	(3,3)	(4,4)	(6,2)	(8,0)	(2,2)	(4,0)	(4,4)
4	(1,1)	(2,2)	(3,3)	(4,4)	(6,6)	(8,8)	(4,0)	(4,4)	(8,0)
5	(1,1)	(2,2)	(3,3)	(4,4)	(6,6)	(8,8)	(4,4)	(8,0)	(8,8)
6	(1,1)	(2,2)	(3,3)	(4,4)	(6,6)	(8,8)	(8,0)	(8,8)	(16,0)
7	(1,1)	(2,2)	(3,3)	(4,4)	(6,6)	(8,8)	(8,8)	(16,0)	(16,16)
8	(1,1)	(2,2)	(3,3)	(4,4)	(6,6)	(8,8)	(16,0)	(16,16)	(32,0)

Table 2.5: Examples of the compact representation (I_l, T_l) of infinite alphabet codes.

Level	Golomb						Elias-Teuhola		
l	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 6$	$m = 8$	$k = 0$	$k = 1$	$k = 2$
0	(1,0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)
1	(1,1)	(2, 0)	(2, 0)	(2, 0)	(2, 0)	(2, 0)	(1, 1)	(2, 0)	(2, 0)
2	(1,2)	(2, 2)	(3, 1)	(4, 0)	(4, 0)	(4, 0)	(2, 1)	(2, 2)	(4, 0)
3	(1,3)	(2, 4)	(3, 4)	(4, 4)	(6, 2)	(8, 0)	(2, 3)	(4, 2)	(4, 4)
4	(1,4)	(2, 6)	(3, 7)	(4, 8)	(6, 8)	(8, 8)	(4, 3)	(4, 6)	(8, 4)
5	(1,5)	(2, 8)	(3,10)	(4,12)	(6,14)	(8,16)	(4, 7)	(8, 6)	(8,12)
6	(1,6)	(2,10)	(3,13)	(4,16)	(6,20)	(8,24)	(8, 7)	(8,14)	(16,12)
7	(1,7)	(2,12)	(3,16)	(4,20)	(6,26)	(8,32)	(8,15)	(16,14)	(16,28)
8	(1,8)	(2,14)	(3,19)	(4,24)	(6,32)	(8,40)	(16,15)	(16,30)	(32,28)

Chapter 3

Unary-Stem Codes

3.1 Definition

Observing the structure of the Golomb and Elias-Teuhola codes in Figures 2.3 and 2.8, it is easy to imagine generalized codes that maintain the unary+binary structure, but with different number of codewords in each binary part [36, 38]. Using m_d to represent the number of codewords in the binary code that is defined after d bits in the unary code, this set of codes can be parameterized with an infinite sequence of positive integers $\mathcal{M} = \{m_0, m_1, m_2, \ldots\}$.

Figure 3.1 shows examples of the trees corresponding to these generalized codes. This type of generalization is not new—similar extensions had been proposed for different types of distributions. However, we are not aware of examples in which all the values can be chosen with such degree of freedom.

Using this representation, we can formally characterize these codes by first defining

$$L_U(d, \mathcal{M}) = \sum_{i=0}^{d-1} m_i, \quad d = 0, 1, 2, \ldots \tag{3.1}$$

The codeword corresponding to symbol s is in this case generated by first coding

$$d = \{s : L_U(s, \mathcal{M}) \le s < L_U(s+1, \mathcal{M})\}, \tag{3.2}$$

using the unary code, followed by the variable number of bits required to code

$$r = s - L_U(d, \mathcal{M}), \quad 0 \le r < m_d. \tag{3.3}$$

Let $\mathcal{W}_U(s, \mathcal{M})$ denote the codewords of these generalized codes. Using the notation defined by (2.18) and (2.19), and the value of q defined in (3.2), the codeword length is

$$\ell_U(s, \mathcal{M}) = 1 + d + \gamma(s - L_U(d, \mathcal{M}), m_d). \tag{3.4}$$

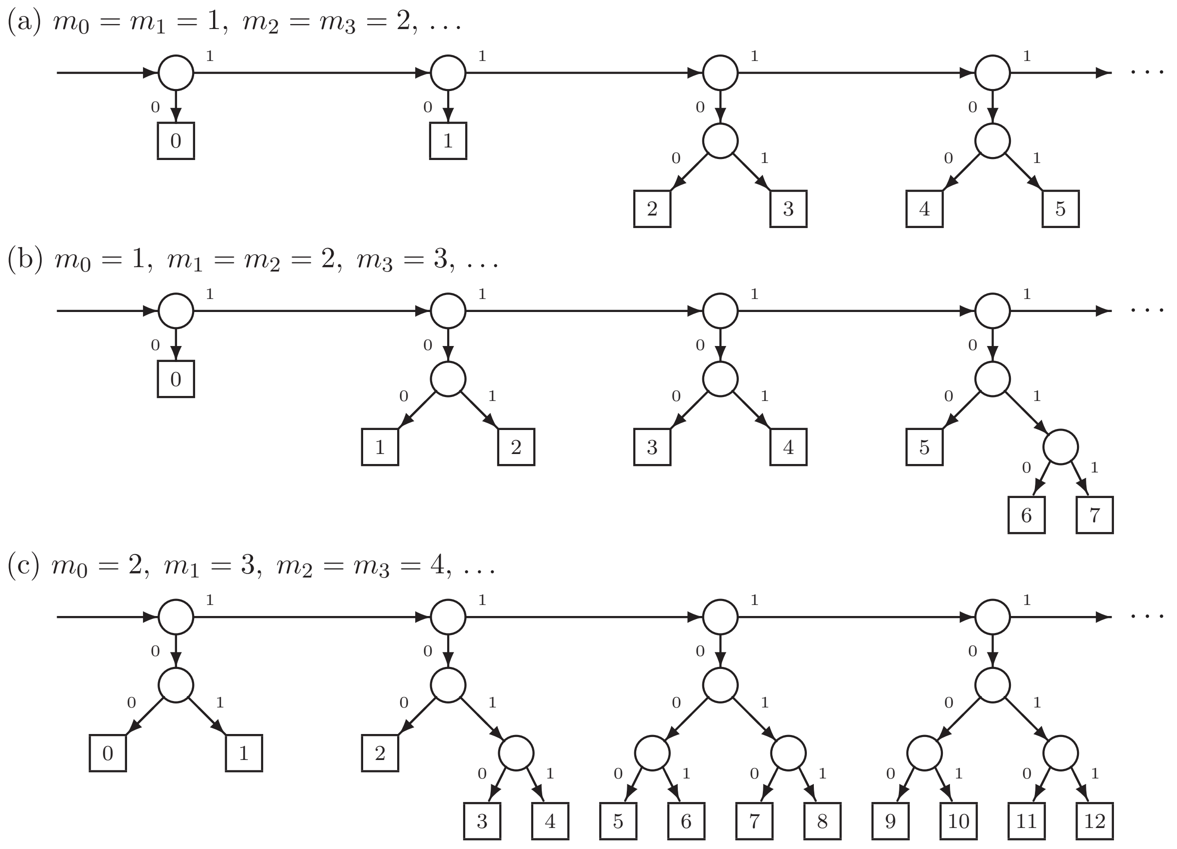

(a) $m_0 = m_1 = 1,\ m_2 = m_3 = 2, \ldots$
(b) $m_0 = 1,\ m_1 = m_2 = 2,\ m_3 = 3, \ldots$
(c) $m_0 = 2,\ m_1 = 3,\ m_2 = m_3 = 4, \ldots$
0
1
...
2
3
4
5
6
7
8
9
10
11
12

Figure 3.1: Trees corresponding to examples of unary-stem codes.

If $m_i = 2^{k_i}$ than we can likewise define generalized Golomb-Rice codes from a sequence $\mathcal{K} = \{k_0, k_1, k_2, \ldots\}$. We call the codes defined by $\mathcal{M}$ the *unary-stem* codes, while those defined by $\mathcal{K}$ are called *dyadic unary-stem* codes. Note that the Elias-Teuhola code with parameter k corresponds to the dyadic unary-stem code with the sequence of parameters $\mathcal{K} = \{k, k+1, k+2, \ldots\}$.

For dyadic unary-stem codes we can define

$$L_D(d, \mathcal{K}) = \sum_{i=0}^{d-1} 2^{k_i}, \quad d = 0, 1, 2, \ldots \tag{3.5}$$

and compute the codeword length as

$$\ell_D(s, \mathcal{K}) = 1 + d + k_d, \tag{3.6}$$

where

$$d = \{s : L_D(s, \mathcal{K}) \le s < L_D(s+1, \mathcal{M})\}, \tag{3.7}$$

The use of a series of parameters allows considerable more freedom in the design of sets of parameterized codes. For example, the solid lines in Figure 3.2 shows the performance of

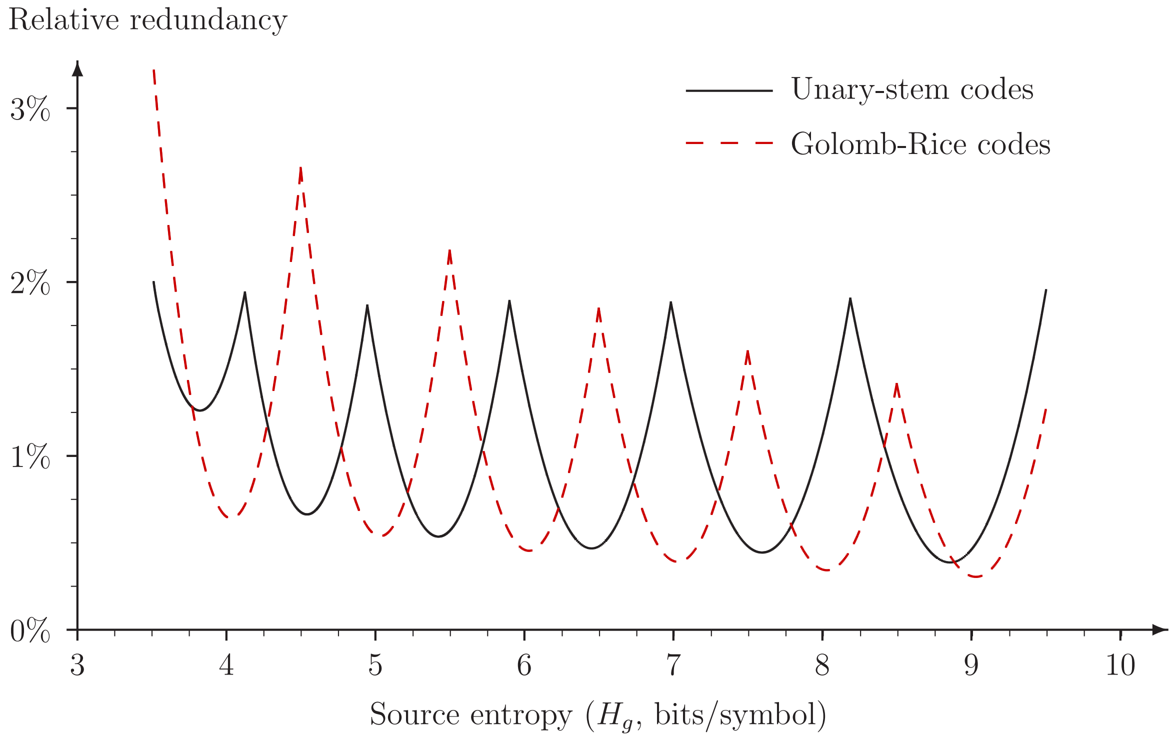


Figure 3.2: Example of tailoring the performance of unary-stem codes to specific requirements: the best unary-stem codes have relative redundancy below 2%.

a set of six unary-stem codes that were designed so that the relative redundancy of the best code is always below 2%, in a certain range of source entropy. The dashed lines show that, in comparison, using six Golomb-Rice codes yields uneven relative redundancies. Cases of more practical interest are shown in Chapter 6.

Unary-stem codes have several practical advantages: they can be implemented with efficiency similar to Golomb and Golomb-Rice codes, and can also have their codewords quickly generated in real-time. They are defined here using an infinite number of parameters only to make the formulations as general as possible. In practical applications, assuming a reasonably good match between the source and the code, parameter m_d should be used with probability roughly proportional to 2^{-d}, i.e., only the first few parameters really need to be more carefully chosen. In addition, we show in the next section that typically these parameters can be computed using simple rules. Thus, similarly to the approach mentioned for storing codewords, the most common sets of code parameters can be stored in tables, while the others can be generated during coding time.

Note that while these code features are an additional motivation for studying and using this type of codes, a main objective of this work is to analyze the difference between Golomb codes and unary-stem codes optimized for adaptive coding on sources that have geometric distribution, but are not stationary. The comparison is much simpler and intuitive because the codes have very similar structures.

3.2 Rate Computation

3.2.1 General Distributions

Note that while in Chapter 2 we consider rates mainly for geometric distributions, here we use the notation defined in Section 2.1 for general distributions.[1] In order to simplify the equations for computing rates we drop the subscripts and second parameter in definitions (3.1) and (3.5), and use simply

$$L(d) = \sum_{i=0}^{d-1} m_i, \quad d = 0, 1, 2, \ldots \tag{3.8}$$

and

$$L(d) = \sum_{i=0}^{d-1} 2^{k_i}, \quad d = 0, 1, 2, \ldots \tag{3.9}$$

In this case there is no danger of confusion, since the two sum have the same meaning, and the type of code can always be inferred from the context.

Proposition 3.2.1 *Given a source distribution $\mathcal{P}$, the average bit rate used by the unary-stem code defined by the sequence of parameters $\mathcal{M}$ is*

$$\begin{aligned} B_U(\mathcal{M}, \mathcal{P}) &= 1 + \lfloor \log_2(m_0) \rfloor + \sum_{d=1}^{\infty} \bar{F}(L(d)) \left\{ 1 + \lfloor \log_2(m_d) \rfloor - \lceil \log_2(m_{d-1}) \rceil \right\} + \\ & \sum_{d=0}^{\infty} \bar{F}(L(d) + \tau(m_d)) \left\{ \lceil \log_2(m_d) \rceil - \lfloor \log_2(m_d) \rfloor \right\}. \end{aligned} \tag{3.10}$$

where $\bar{F}(s)$ is the reverse cumulative distribution defined by (2.5), and function $\tau(m)$ is defined by (2.18).

Proof: We derive the formula in two stages. First, we compute only the bits used by the binary part. Using definition (2.19), this average bit rate is

$$\begin{aligned} B'_U(\mathcal{M}, \mathcal{P}) &= \sum_{d=0}^{\infty} \sum_{s=L(d)}^{L(d+1)-1} p(s) \gamma(s - L(d), m_d) \qquad (3.11) \\ &= \sum_{d=0}^{\infty} \left\{ \lfloor \log_2(m_d) \rfloor \sum_{s=L(d)}^{L(d)+\tau(m_d)-1} p(s) + \lceil \log_2(m_d) \rceil \sum_{s=L(d)+\tau(m_d)}^{L(d+1)-1} p(s) \right\} \\ &= \sum_{d=0}^{\infty} \lfloor \log_2(m_d) \rfloor \left[\bar{F}(L(d)) - \bar{F}(L(d) + \tau(m_d)) \right] + \end{aligned}$$

[1] In Chapter 4 other symbol distributions are defined from Bayesian estimation.

$$\begin{aligned}
& \sum_{d=0}^{\infty} \lceil \log_2(m_d) \rceil \left[\bar{F}(L(d)+\tau(m_d)) - \bar{F}(L(d+1)) \right] \\
= \; & \lfloor \log_2(m_0) \rfloor + \sum_{d=1}^{\infty} \bar{F}(L(d)) \left\{ \lfloor \log_2(m_d) \rfloor - \lceil \log_2(m_{d-1}) \rceil \right\} + \\
& \sum_{d=0}^{\infty} \bar{F}(L(d)+\tau(m_d)) \left\{ \lceil \log_2(m_d) \rceil - \lfloor \log_2(m_d) \rfloor \right\}
\end{aligned}$$

The second component, corresponding to the unary-code, is

$$\begin{aligned}
B''_U(\mathcal{M},\mathcal{P}) &= \sum_{d=0}^{\infty} \sum_{s=L(d)}^{L(d+1)-1} p(s)(d+1) = \sum_{d=0}^{\infty} (d+1) \sum_{s=L(d)}^{L(d+1)-1} p(s) \qquad (3.12) \\
&= \sum_{d=0}^{\infty} (d+1)[\bar{F}(L(d)) - \bar{F}(L(d+1))] \\
&= \sum_{d=0}^{\infty} \bar{F}(L(d)).
\end{aligned}$$

The main result is the sum of these two components. ∎

The equivalent for dyadic unary-stem codes is obtained by simply replacing $\lceil \log_2(m_d) \rceil = \lfloor \log_2(m_d) \rfloor = k_d$.

Corollary 3.2.2 *Given a source distribution $\mathcal{P}$, the average bit rate used by the dyadic unary-stem code defined with the sequence of parameters $\mathcal{K}$ is*

$$B_D(\mathcal{K},\mathcal{P}) = 1 + k_0 + \sum_{d=1}^{\infty} \bar{F}(L(d))(1 + k_d - k_{d-1}). \qquad (3.13)$$

These results show that for unary-stem codes it is more convenient to use the cumulative sum of symbols $L(d)$ (defined for a depth d of the unary code), than to use the general code representation defined in Section 2.6.

An important advantage of using $L(d)$ is that it is a sum of an infinite number of terms which is commonly known in closed form (cf. Chapter 4). This makes it much easier to evaluate contributions of sets with infinite number of symbols to the average bit rate, which is an important part of our code analysis and optimization.

3.2.2 Geometric Distributions

Applying Proposition 3.2.1 to geometric distributions yields the following results.

Proposition 3.2.3 *The average number of bits used by a unary-stem code with parameter sequence $\mathcal{M}$ in a geometric distribution source with parameter ρ is*

$$B_U(\mathcal{M},\rho) = \sum_{d=0}^{\infty} \rho^{L(d)} \left\{1 + \left(1-\rho^{\tau(m_d)}\right)\lfloor \log_2(m_d)\rfloor + \left(\rho^{\tau(m_d)} - \rho^{m_d}\right)\lceil \log_2(m_d)\rceil\right\}, \qquad (3.14)$$

which can be computed recursively using

$$\begin{aligned} B_U^{(d)}(\mathcal{M},\rho) &= 1 + \lfloor \log_2(m_d)\rfloor + \rho^{\tau(m_d)}\left(\lceil \log_2(m_d)\rceil - \lfloor \log_2(m_d)\rfloor\right) + \qquad (3.15)\\ & \quad \rho^{m_d}\left[B_U^{(d+1)}(\mathcal{M},\rho) - \lceil \log_2(m_d)\rceil\right], \end{aligned}$$

with $B_U^{(0)}(\mathcal{M},\rho) = B_U(\mathcal{M},\rho)$.

Proof: Using equation (3.10), we obtain

$$\begin{aligned} B_U(\mathcal{M},\rho) &= \sum_{s=0}^{\infty} \ell_U(s,\mathcal{M})(1-\rho)\,\rho^s \qquad (3.16)\\ &= 1 + (1-\rho)\sum_{d=0}^{\infty}\sum_{s=L(d)}^{L(d+1)-1}\left[d + \gamma(s-L(d),m_d)\right]\rho^s \\ &= 1 + (1-\rho)\sum_{d=0}^{\infty}\rho^{L(d)}\sum_{s=0}^{m_d-1}\left[d+\gamma(s,m_d)\right]\rho^s \\ &= \sum_{d=0}^{\infty}\rho^{L(d)}\left\{1 + \left(1-\rho^{\tau(m_d)}\right)\lfloor \log_2(m_d)\rfloor + \left(\rho^{\tau(m_d)}-\rho^{m_d}\right)\lceil \log_2(m_d)\rceil\right\}. \end{aligned}$$

The recursive equation is derived using the fact that

$$B_U^{(d)}(\mathcal{M},\rho) = \rho^{-L(d)}\sum_{i=d}^{\infty}\rho^{L(i)} r(m_i,\rho), \qquad (3.17)$$

where

$$r(m,\rho) = 1 + \left(1-\rho^{\tau(m)}\right)\lfloor \log_2(m)\rfloor + \left(\rho^{\tau(m)}-\rho^m\right)\lceil \log_2(m)\rceil. \qquad (3.18)$$

Since $L(0) \equiv 0$ it is clear that $B_U^{(0)}(\mathcal{M},\rho) = B_U(\mathcal{M},\rho)$. By induction we also have

$$\begin{aligned} B_U^{(d)}(\mathcal{M},\rho) &= r(m_d,\rho) + \rho^{-L(d)}\sum_{i=d+1}^{\infty}\rho^{L(i)} r(m_i,\rho), \qquad (3.19)\\ &= r(m_d,\rho) + \rho^{m_d}\left[\rho^{-L(d+1)}\sum_{i=d+1}^{\infty}\rho^{L(i)} r(m_i,\rho)\right] \\ &= r(m_d,\rho) + \rho^{m_d} B_U^{(d+1)}(\mathcal{M},\rho). \end{aligned}$$

∎

Corollary 3.2.4 *The average number of bits used by a dyadic unary-stem code with parameter sequence $\mathcal{K}$ in a geometric distribution source with parameter ρ is*

$$B_D(\mathcal{K}, \rho) = 1 + k_0 + \sum_{d=1}^{\infty} \rho^{L(d)}(1 + k_d - k_{d-1}), \tag{3.20}$$

which can be computed recursively using

$$B_D^{(d)}(\mathcal{K}, \rho) = 1 + k_d + \rho^{(2^{k_d})} \left[B_D^{(d+1)}(\mathcal{K}, \rho) - k_d \right], \tag{3.21}$$

with $B_D^{(0)}(\mathcal{K}, \rho) = B_D(\mathcal{K}, \rho)$.

3.2.3 Redundancy of the Unary Code

Unary-stem codes have a very particular structure, with its unary and binary code parts. Thus, it is interesting to analyze the individual contributions of those two components. One useful value is the redundancy of the unary code. The following theorem show how this redundancy can be computed. Again we use the simplified notation, i.e., use $L(d)$ instead of $L_U(d, \mathcal{M})$.

Proposition 3.2.5 *Given a source distribution $\mathcal{P}$, and a unary-stem code defined by $\mathcal{M}$, the redundancy of the unary code part is*

$$\begin{aligned} \delta(\mathcal{M}, \mathcal{P}) &= \sum_{d=0}^{\infty} \left\{ \bar{F}(L(d)) + [\bar{F}(L(d)) - \bar{F}(L(d+1))] \log_2\bigl(\bar{F}(L(d)) - \bar{F}(L(d+1))\bigr) \right\} \\ &= 1 + \log_2\bigl(1 - \bar{F}(L(1))\bigr) + \\ &\quad \sum_{d=1}^{\infty} \bar{F}(L(d)) \left\{ 1 - \log_2\left(\frac{\bar{F}(L(d-1)) - \bar{F}(L(d))}{\bar{F}(L(d)) - \bar{F}(L(d+1))} \right) \right\} \end{aligned} \tag{3.22}$$

Proof: Let us consider a node at depth d in the tree that defines the unary code. The probability that this node is reached while coding a symbol is equal to the sum of the probabilities of symbols $L(d), L(d)+1, \ldots$, which is equal to $\bar{F}(L(d))$. The probability of the branch defined by bit 1 is $\bar{F}(L(d+1))$, since it is connected to the node of the unary code at depth $d+1$. Thus, the conditional probability of that branch is $\bar{F}(L(d+1))/\bar{F}(L(d))$, and of the bit 0 branch is $1 - \bar{F}(L(d+1))/\bar{F}(L(d))$. The redundancy of the unary code is the average of the redundancy of coding a bit in each node, which is

$$\begin{aligned} \delta(\mathcal{M}, \mathcal{P}) &= \sum_{d=0}^{\infty} \bar{F}(L(d)) \left\{ 1 - \frac{\bar{F}(L(d+1))}{\bar{F}(L(d))} \log_2\left(\frac{\bar{F}(L(d))}{\bar{F}(L(d+1))} \right) - \right. \\ &\quad \left. \frac{\bar{F}(L(d)) - \bar{F}(L(d+1))}{\bar{F}(L(d))} \log_2\left(\frac{\bar{F}(L(d)) - \bar{F}(L(d+1))}{\bar{F}(L(d))} \right) \right\} \end{aligned} \tag{3.23}$$

$$= \sum_{d=0}^{\infty} \Big\{ \bar{F}(L(d)) + \bar{F}(L(d+1)) \log_2\!\Big(\frac{\bar{F}(L(d+1))}{\bar{F}(L(d))} \Big) + \\ [\bar{F}(L(d)) - \bar{F}(L(d+1))] \log_2\!\Big(\frac{\bar{F}(L(d)) - \bar{F}(L(d+1))}{\bar{F}(L(d))} \Big) \Big\}$$

Removing the terms that cancel in the sum above yields (3.22). ∎

Corollary 3.2.6 *The redundancy of the unary part of a Golomb code with parameter m on a source with geometric distribution and parameter ρ is*

$$\delta_G(m,\rho) = \frac{1}{1-\rho^m} - H_g(\rho^m). \tag{3.24}$$

where $H_g(\cdot)$ is the entropy of a source with geometric distribution (2.14).

Proof: This result follows from (3.22), using $L(d) = md$, and $\bar{F}(s) = \rho^s$:

$$\delta_G(m,\rho) = \sum_{d=0}^{\infty} \big\{ \rho^{md} + \rho^{md}(1-\rho^m) \log_2\!\big(\rho^{md}(1-\rho^m)\big) \big\} \tag{3.25}$$

$$= \frac{1}{1-\rho^m} + \log_2(1-\rho^m) + \frac{\rho^m}{1-\rho^m} \log_2(\rho^m)\,. \tag{3.26}$$

∎

Chapter 4

Adaptive Coding and Source Estimation

In Chapter 2 we show that to determine the optimal Golomb code for a data source with geometric distribution we need to know or estimate a single source parameter, ρ. Our assumption here is that we have a memoryless random data source, its symbols have geometric distribution, but the parameter ρ is unknown, and has to be estimated from a small number of samples. We consider that the information provided by a sequence of previous source samples, $\{S_1, S_2, \ldots, S_{N_s}\}$—known by the encoder and decoder—is employed for choosing the code to be used for the next source sample.

We would like to study situations in which the source properties may change rapidly, and consequently we can only use a few samples in the estimation. However, to avoid overly complicating the source model and notation, we use the very simple assumptions described above, keeping in mind that the number of samples N_s can be quite small (e.g., $N_s \leq 4$). We also show that it is easy to extend the model using different weights for each sample, in order to consider diminishing statistical dependencies.

In the following sections we first analyze the maximum-likelihood estimation of ρ, and what happens when we use only the estimation of this parameter in an empiric method for changing the code structure. This approach presents a clear motivation for unary-stem codes. While the maximum-likelihood approach has the advantage of simplicity, it cannot take into account side information about the source, and does not really address the optimal code determination problem. For that purpose we study a Bayesian approach, which conveniently enables the use of pre-defined assumptions about source (e.g., its entropy is constrained to a certain range), and also allows code evaluation and direct optimization.

4.1 Maximum-Likelihood Source Estimation

Under the assumptions presented above, after observing a number N_s of independent source samples, $\{S_1, S_2, \ldots, S_{N_s}\}$, with sum

$$\Sigma_s = \sum_{i=1}^{N_s} S_i, \tag{4.1}$$

the log-likelihood function corresponding to the geometric distribution is

$$\Lambda(\rho) = \sum_{i=1}^{N_s} \ln\left[p_g(S_i, \rho)\right] = \ln(1-\rho)\ N_s + \ln(\rho)\ \Sigma_s. \tag{4.2}$$

Defining $\hat{\rho}$ as the value of ρ that maximizes the likelihood, i.e., the solution of equation $d\Lambda(\rho)/d\rho = 0$, we have

$$\hat{\rho} = \frac{\Sigma_s}{\Sigma_s + N_s}. \tag{4.3}$$

It is interesting to note that

$$\frac{\hat{\rho}}{1-\hat{\rho}} = \frac{\Sigma_s}{N_s} = \hat{s}. \tag{4.4}$$

This value corresponds to the estimate of the mean symbol value (2.12), showing that we can simply use the average of the previous source symbols for computing the maximum-likelihood estimation $\hat{\rho}$, and then use (2.23) to select the code that is optimal on sources with ρ equal to the estimated value.

We can go one step further, and consider that it is possible to improve this estimation with additional information while coding. Let us assume that we use a Golomb code with parameter m to code data symbol s. The unary code is used to represent the conditions $s \geq m, s \geq 2m, \ldots$ Thus, if we know that a number $d \geq 0$ of 1-bits should be used in the unary code, then we know that $s \geq dm$. A property of the geometric probability distribution is that the type of information to be coded with the knowledge that $s \geq dm$ is exactly the same as before since

$$p_g(s|s \geq dm, \rho) = (1-\rho)\rho^{s-dm}, \quad s = dm, dm+1, dm+2, \ldots \tag{4.5}$$

which is why the binary part of the Golomb code never changes (cf. Figure 2.3).

However, using this new information the log-likelihood function can be updated to

$$\begin{aligned}\Lambda_d(\rho) &= \sum_{i=1}^{N_s} \ln\left[p_g(S_i, \rho)\right] + \ln\left[\mathrm{Prob}(s \geq dm)\right] \\ &= \ln(1-\rho)\ N_s + \ln(\rho)\ (\Sigma_s + dm),\end{aligned} \tag{4.6}$$

and the new optimal estimate is

$$\hat{\rho}_d = \frac{dm + \Sigma_s}{dm + \Sigma_s + N_s}, \quad d = 0, 1, 2, \ldots \tag{4.7}$$

The equation above shows that whenever a 1-bit is coded in the unary part, the maximum-likelihood estimate $\hat{\rho}$ increases, since

$$\hat{\rho}_{d+1} > \hat{\rho}_d, \quad d = 0, 1, 2, \ldots \tag{4.8}$$

If the value of d keeps growing, then at some point the value of $\hat{\rho}_d$ will correspond to a larger value of the code parameter m, and consequently we should consider changing the binary part of the code.

For example, suppose we have $\hat{\rho} = 1/2$ and we need to code symbol $s = 100$. If we use (2.23) and assume that $m = 1$ is indeed optimal, then we must use 101 bits to code this symbol. However, in any practical situation it is much more reasonable to assume that $\hat{\rho}$ had been grossly underestimated than to assume that the symbol indeed has probability equal to 2^{-101}, and that using 101 bits is optimal.

Furthermore, using the results above we conclude that, for any Σ_s and N_s, we have

$$\lim_{d \to \infty} \hat{\rho}_d = 1,$$

clearly showing that when do not know ρ with certainty the endless repetition of the binary part of the Golomb codes is not reasonable.

Unary-stem codes (Chapter 3) can address this problem because we can increase the value of m_d according to each new estimate $\hat{\rho}_d$. Under the assumption that the estimate is reasonably accurate, and the probability distribution is not too different from the expected, we can use equation (2.23) to keep updating m_d. The following theorems define how these codes can be constructed using the maximum-likelihood parameter estimation, and some important properties.

Proposition 4.1.1 *Assuming a data source with geometric distribution but unknown parameter ρ, and a set of N_s source samples $\{S_1, S_2, \ldots, S_{N_s}\}$ with sum Σ_s, the unary-stem code defined by the sequence $\mathcal{M}(\Sigma_s, N_s) = \{m_0(\Sigma_s, N_s), m_1(\Sigma_s, N_s), \ldots\}$, such that each $m_d(\Sigma_s, N_s)$ is updated according to the most current maximum-likelihood estimation of ρ, is defined recursively by*

$$m_d(\Sigma_s, N_s) = \left\lceil \frac{\log([2 + r_d(\Sigma_s, N_s)]/[1 + r_d(\Sigma_s, N_s)])}{\log(1 + r_d(\Sigma_s, N_s))} \right\rceil, \quad d = 0, 1, 2, \ldots \tag{4.9}$$

where

$$r_d(\Sigma_s, N_s) = \frac{N_s}{\Sigma_s + \sum_{i=0}^{d-1} m_i(\Sigma_s, N_s)}. \tag{4.10}$$

Proof: The maximum-likelihood function, assuming d 1-bits in the unary part, is

$$\Lambda_d(\rho) = \ln(1 - \rho)\ N_s + \ln(\rho) \left(\Sigma_s + \sum_{i=0}^{d-1} m_i \right) \tag{4.11}$$

(In this proof we represent $m_d(\Sigma_s, N_s)$ simply as m_d.)

The optimal estimate of ρ is

$$\hat{\rho}_d = \frac{L(d) + \Sigma_s}{L(d) + \Sigma_s + N_s}, \quad d = 0, 1, 2, \ldots \tag{4.12}$$

where $L(d)$ is the sum defined by (3.8).[1]

It is convenient to work with the estimated mean symbol value

$$\hat{s}_d = \frac{\hat{\rho}_d}{1 - \hat{\rho}_d} = \frac{L(d) + \Sigma_s}{N_s}, \quad d = 0, 1, 2, \ldots. \tag{4.13}$$

This way we can use (2.23) to compute the optimal value of m, and conclude that the unary-stem code should use the sequence

$$m_d(\Sigma_s, N_s) = \left\lceil \frac{-\log(1 + \hat{\rho}_d)}{\log(\hat{\rho}_d)} \right\rceil = \left\lceil \frac{\log[(2 + 1/\hat{s}_d)/(1 + 1/\hat{s}_d)]}{\log(1 + 1/\hat{s}_d)} \right\rceil, \quad d = 0, 1, 2, \ldots \tag{4.14}$$

which is equivalent to (4.9). ∎

We can also analyze how fast m is expected to grow when formula (4.9) is used for updating m.

Proposition 4.1.2 *The elements of $\mathcal{M}(\Sigma_s, N_s)$ defined by (4.9) have asymptotic growth that is geometric with ratio $1 + \ln(2)/N_s$.*

Proof: Figure 4.1(a) shows a dynamic system representing the set of recursive equations that enable computing m_d, $d = 0, 1, \ldots$, from Σ_s and N_s. Due to its nonlinear component, it is difficult to guess the type of sequences that are generated. However, from the series expansion

$$\frac{\ln[(2 + 1/\hat{s})/(1 + 1/\hat{s})]}{\ln(1 + 1/\hat{s})} = \ln(2)\hat{s} - \frac{1 - \ln(2)}{2} + \frac{3 - 2\ln(2)}{24}\left(\frac{1}{\hat{s}} - \frac{1}{2\hat{s}^2} + \cdots\right), \tag{4.15}$$

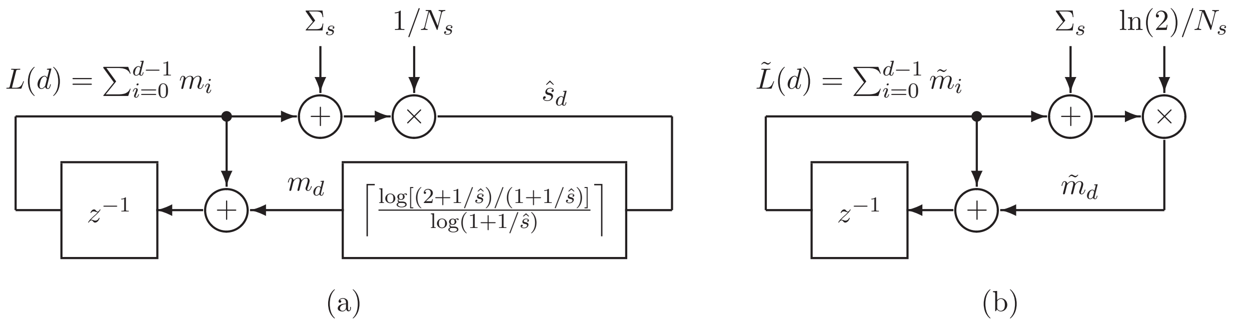
we can derive the approximation

$$\frac{\ln[(2\hat{s} + 1)/(\hat{s} + 1)]}{\ln(1 + 1/\hat{s})} \approx \ln(2)\hat{s}, \quad \text{if} \quad \hat{s} \geq 1, \tag{4.16}$$

for the nonlinear updating part of the system in Figure 4.1(a), replacing it with a single multiplication by $\ln(2)$, and obtain the linear dynamic system shown in Figure 4.1(b). The equations with state update and output for this system are

$$\tilde{L}(d + 1) = \tilde{L}(d)\left[1 + \frac{\ln(2)}{N_s}\right] + \frac{\ln(2)\Sigma_s}{N_s}, \tag{4.17}$$

[1]Note that to simplify the notation the explicit dependence on $\mathcal{M}$ is not shown.



Σ_s
$1/N_s$
$L(d) = \sum_{i=0}^{d-1} m_i$
$\hat{s}_d$
m_d
z^{-1}
$\left\lceil \frac{\log[(2+1/\hat{s})/(1+1/\hat{s})]}{\log(1+1/\hat{s})} \right\rceil$
(a)
Σ_s
$\ln(2)/N_s$
$\tilde{L}(d) = \sum_{i=0}^{d-1} \tilde{m}_i$
$\tilde{m}_d$
z^{-1}
(b)

Figure 4.1: (a) Dynamic system for updating the code parameter m_d; (b) Linear approximation for $\hat{s} > 1$.

and

$$\tilde{m}_d = \tilde{L}(d+1) - \tilde{L}(d). \tag{4.18}$$

The solution, assuming an initial solution $\tilde{L}(1) \geq 1$, is

$$\tilde{L}(d+1) = \left[\tilde{L}(1) + \Sigma_s\right] \left[1 + \frac{\ln(2)}{N_s}\right]^d - \Sigma_s, \quad d = 0, 1, 2, \ldots \tag{4.19}$$

which shows that, whatever the initial conditions, the optimal coding parameters m_d have an asymptotic geometric growth. The growth rate, $1 + \ln(2)/N_s$, depends on the amount of information about the source already gathered, which in this case is determined only by N_s, the number of known source samples. ∎

Some results of the application of this maximum-likelihood technique for defining unary-stem codes, using the exact updating equation (4.9), are shown in Appendix C (pg. 90). Table C.1 contains examples of the parameters $\{m_0, m_1, \ldots\}$ chosen according to the values of N_s and Σ_s, while Table C.2 shows the corresponding values of the estimated $\hat{\rho}$, computed using (4.12). Note that, as stated in Proposition 4.1.2, m_d tends to grow exponentially, but the growth rate decreases quickly with N_s.

It is interesting to note that the Elias-Teuhola codes (Section 2.5) also were defined with m_d growing exponentially, in order to accommodate for the uncertainty of the source, since they correspond to unary-stem codes with parameter sequences $\mathcal{M} = \{m, 2m, 4m, \ldots\}$. We can see in equation (4.19) that with the maximum-likelihood criterion this doubling would occur with the fractional number of samples $N_s = \ln(2)$.

We can extend the maximum-likelihood estimation approach of this section to dyadic unary-stem codes by simply replacing the equation (2.23), for the optimal parameter m, with the equation (2.27) for the optimal parameter k, in the updating formula (4.9). The result is

$$k_d(\Sigma_s, N_s) = \max\left\{0, \left\lceil \log_2\left(\frac{\log(\phi)}{\log(L(d) + \Sigma_s + N_s) - \log(L(d) + \Sigma_s)}\right)\right\rceil\right\}. \tag{4.20}$$

Table C.3 shows the first elements of the sequences $\mathcal{K}(\Sigma_s, N_s)$ obtained with this formula, while Table C.4 lists the $\hat{\rho}_d$ values. We can observe that, even though the conditions for updating the code parameters are different, k_d has a roughly linear growth, which corresponds to the same type of exponential growth of m_d.

In our initial assumptions the parameter N_s can only have positive integer values. We can generalize our model, defining a set of weights $w_i \in [0, 1]$, and the weighted sum and average

$$N_w = \sum_{i=1}^{N_s} w_i, \quad \Sigma_w = \sum_{i=1}^{N_s} w_i S_i, \tag{4.21}$$

and replace N_s and Σ_s with N_w and Σ_w in the estimation equations. This way we can assign larger values of w_i for the samples that yield more reliable estimation, and reduce w_i to model sources that change more quickly.

4.2 Bayesian Estimation

The maximum-likelihood estimation has the advantage of not requiring any prior knowledge about the data source and its parameter ρ. However, while in practice there may be little knowledge about the source's properties, in mathematical terms the uncertainty may not be absolute. For example, due to physical properties, measurement and equipment limitations, and other factors, the entropy of the vast majority of data types (audio, images, temperature, etc.) is within a certain range (e.g., less than 16 bits/sample).

Thus, it is advantageous to include this knowledge of typical sources, and we can do it by assuming that ρ is a random variable with a prior probability density function $f(\rho)$. We can obtain rough estimates of this function by testing typical sources. For instance, Figure 4.2 shows a histogram of estimated values of $\log_2[\rho/(1-\rho)]$ that can be used for guessing the pdf $f(\rho)$ for lossless image compression. The need for guessing this function is not a practical difficulty, since, in our coding technique, the exact shape of $f(\rho)$ is not important. We can be conservative, and choose a function that overestimates the probability of rare cases, but that somehow indicates that these occur less commonly, and also indicate those cases that, for practical reasons, never occur. As we show in the next sections, this is possible because the results should be more strongly defined by the observed data $\{S_1, S_2, \ldots, S_{N_s}\}$. In our simulations we use several distributions for ρ, including the distribution shown in Figure 4.3, which is an approximation of the distribution of Figure 4.2. (The notation used in Figure 4.3 is described in Chapter 6 and Appendix A.)

4.2.1 Source Parameter Estimation and Posterior Probabilities

Given a probability distribution function $f(\rho)$, we again assume that a number of d 1-bits had been used in the unary part of a unary-stem code. Under these conditions we can

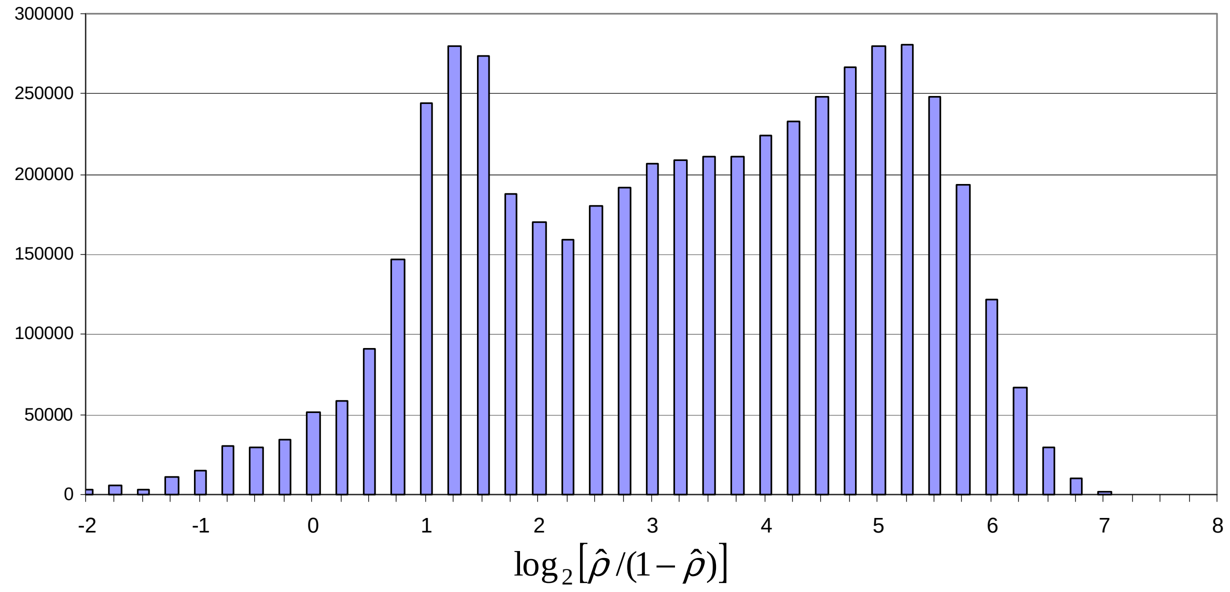


Figure 4.2: Distribution of source parameter estimation on 5×5 blocks of the residual image shown in Figure 1.1.

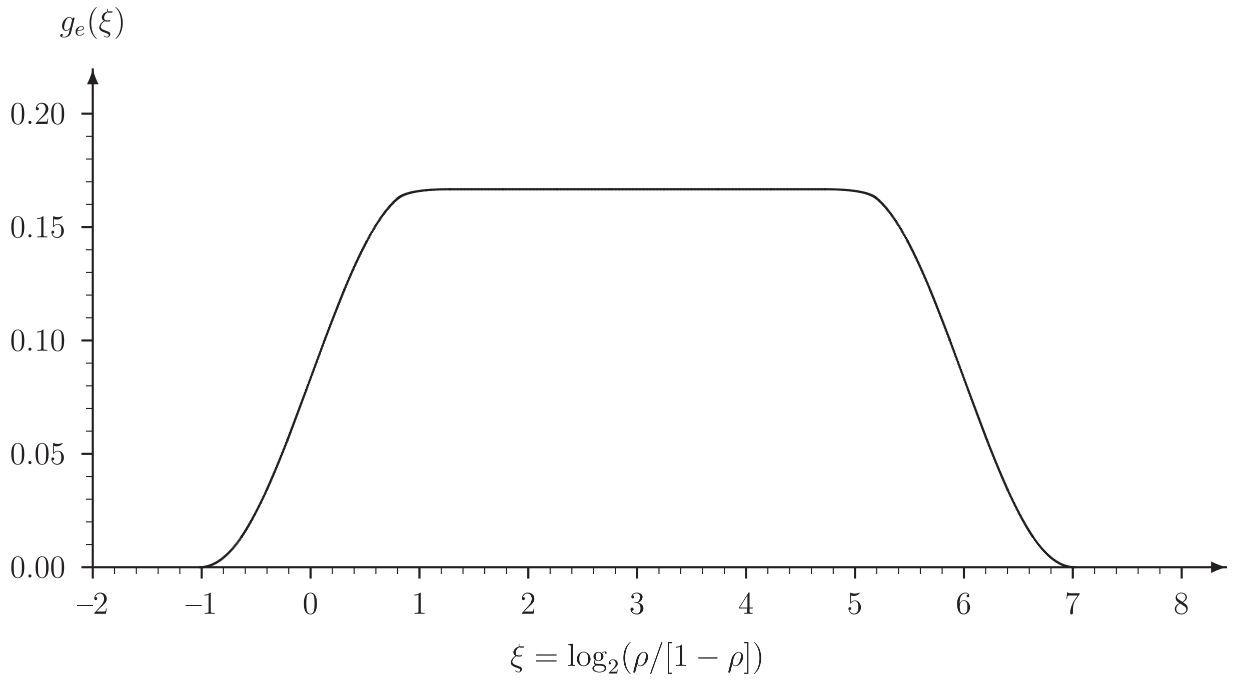


Figure 4.3: Function that roughly approximates the distribution of Figure 4.2, used in the numerical tests.

compute the average value of ρ using Bayes' theorem [9], and obtain

$$\begin{aligned}
\hat{\rho}_d &= \frac{\int_0^1 f(\rho)\rho^{1+L(d)}\left[\prod_{i=1}^{N_s}(1-\rho)p^{S_i}\right]\, d\rho}{\int_0^1 f(\rho)\rho^{L(d)}\left[\prod_{i=1}^{N_s}(1-\rho)p^{S_i}\right]\, d\rho} && (4.22)\\
&= \frac{\int_0^1 f(\rho)(1-\rho)^{N_s}\rho^{1+L(d)+\Sigma_s}\, d\rho}{\int_0^1 f(\rho)(1-\rho)^{N_s}\rho^{L(d)+\Sigma_s}\, d\rho} && (4.23)
\end{aligned}$$

Since all the Bayesian estimation equations use integrals of the type shown above, we can greatly simplify the notation by defining the functional

$$\Phi(m,n,f) = \int_0^1 f(\rho)\rho^{m-1}(1-\rho)^{n-1}\, d\rho. \tag{4.24}$$

and with this notation we have

$$\hat{\rho}_d = \frac{\Phi(\Sigma_s + L(d) + 2, N_s + 1, f)}{\Phi(\Sigma_s + L(d) + 1, N_s + 1, f)}. \tag{4.25}$$

The maximum-likelihood approach only lets us define estimates of optimal codes based on estimates of the source parameter ρ, since the actual probabilities are unknown. The Bayesian approach, on the other hand, also let us compute a set of posterior probabilities that can be readily used for measuring the efficiency of a code, and thus determine which codes are optimal. The conditional probability of symbol s is

$$\begin{aligned}
p_m(s|\{S_1,S_2,\ldots,S_{N_s}\},f) &= \frac{\int_0^1 f(\rho)(1-\rho)\rho^s\left[\prod_{i=1}^{N_s}(1-\rho)\rho^{S_i}\right]\, d\rho}{\int_0^1 f(\rho)\left[\prod_{i=1}^{N_s}(1-\rho)\rho^{S_i}\right]\, d\rho} && (4.26)\\
&= \frac{\int_0^1 f(\rho)\rho^{\Sigma_s+s}(1-\rho)^{N_s+1}\, d\rho}{\int_0^1 f(\rho)\rho^{\Sigma_s}(1-\rho)^{N_s}\, d\rho}.
\end{aligned}$$

We can observe that Σ_s is a sufficient statistic for estimation on this type of data sources. However, its use in the estimation equations always depend on the pre-defined number of samples N_s. For this reason, our notation uses the pair (Σ_s, N_s) to represent the sufficient statistics. Using this notation and the operator defined in (4.24) we have

$$p_m(s|\{S_1,S_2,\ldots,S_{N_s}\},f) = p_m(s,\Sigma_s,N_s,f) = \frac{\Phi(\Sigma_s+s+1, N_s+2, f)}{\Phi(\Sigma_s+1, N_s+1, f)}. \tag{4.27}$$

The average symbol value

$$\bar{s}(\Sigma_s,N_s,f) = \sum_{s=0}^{\infty} s\, p_m(s,\Sigma_s,N_s,f) = \sum_{s=0}^{\infty} \frac{s\Phi(\Sigma_s+s+1,N_s+2,f)}{\Phi(\Sigma_s+1,N_s+1,f)}, \tag{4.28}$$

can be computed using the identity

$$\sum_{s=0}^{\infty} \int_0^1 [s\rho^s] f(\rho)\rho^{\Sigma_s}(1-\rho)^{N_s+1}\, d\rho = \int_0^1 f(\rho)\rho^{\Sigma_s+1}(1-\rho)^{N_s-1}\, d\rho, \tag{4.29}$$

and is defined by

$$\bar{s}(\Sigma_s, N_s, f) = \frac{\Phi(\Sigma_s+2, N_s, f)}{\Phi(\Sigma_s+1, N_s+1, f)}. \tag{4.30}$$

Similarly, the reverse cumulative distribution (2.5) can be computed using

$$\sum_{i=s}^{\infty} \int_0^1 \rho^i f(\rho)\rho^{\Sigma_s}(1-\rho)^{N_s+1}\, d\rho = \int_0^1 f(\rho)\rho^{\Sigma_s+s}(1-\rho)^{N_s}\, d\rho, \tag{4.31}$$

which results in

$$\bar{F}_m(s, \Sigma_s, N_s, f) = \frac{\Phi(\Sigma_s+s+1, N_s+1, f)}{\Phi(\Sigma_s+1, N_s+1, f)}. \tag{4.32}$$

It is interesting to consider the case when we have the information that the current source sample (denoted by the random variable S) is not smaller than a certain value $l \geq 0$. The posterior probabilities in this case define the following distribution

$$\begin{aligned} \text{Prob}(S = l+s \mid \{S \geq l, S_1, \ldots, S_{N_s}\}, f) &= \frac{\int_0^1 f(\rho)\rho^{\Sigma_s+l+s}(1-\rho)^{N_s+1}\, d\rho}{\int_0^1 f(\rho)\rho^{\Sigma_s+l}(1-\rho)^{N_s}\, d\rho} \\ &= \frac{\Phi(\Sigma_s+l+s+1, N_s+2, f)}{\Phi(\Sigma_s+l+1, N_s+1, f)} \\ &= p_m(s, \Sigma_s+l, N_s, f). \end{aligned} \tag{4.33}$$

This means that we have the same type of distribution, but with parameter $\Sigma_s + l$ instead of Σ_s. This is quite convenient, and also yields another interesting result. Under definition (4.1) of Σ_s we always have $\Sigma_s = 0$ if $N_s = 0$, but the result above shows that we can also have $\Sigma_s > 0$ when $N_s = 0$ in any of the computations defined in this section, as long it is understood that the computed quantities do not refer to the original alphabet. Instead, we need to assume that $S \geq \Sigma_s$, and that the symbols are renumbered according to $s = S - \Sigma_s$.

4.2.2 Uniform Prior

The uniform probability density function

$$\Pi_{a,b}(\rho) = \begin{cases} 1/(b-a), & \rho \in (a,b), \\ 0, & \rho \in [0,a] \cup [b,1], \end{cases} \qquad 0 \leq a < b \leq 1, \tag{4.34}$$

is particularly interesting as a prior because we know that

$$\begin{aligned} \Phi(m, n, \Pi_{a,b}) &= \frac{1}{b-a} \int_a^b \rho^{m-1}(1-\rho)^{n-1}\, d\rho \\ &= B(m,n)\, \frac{I_b(m,n) - I_a(m,n)}{b-a} \end{aligned} \tag{4.35}$$

where $I_x(m,n)$ is the incomplete beta function [3, 15],

$$B(m,n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)} \tag{4.36}$$

is the beta function, and

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt \tag{4.37}$$

is the Euler Gamma function.

Since $I_0(m,n) \equiv 0$ and $I_1(m,n) \equiv 1$, we have

$$\Phi(m,n,\Pi_{0,1}) = B(m,n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)}. \tag{4.38}$$

and the average parameter value in this case is

$$\hat{\rho}_d = \frac{L(d) + \Sigma_s + 1}{L(d) + \Sigma_s + N_s + 2}, \quad d = 0, 1, \ldots \tag{4.39}$$

Comparing (4.12) and (4.39), we can see that both forms of estimation have the same general form, and consequently the same asymptotic behavior if used for computing $m_d(\Sigma_s, N_s)$. However, the Bayesian estimation adds a bias to the values of N_s and Σ_s, and hence can be used even when $N_s = \Sigma_s = L(d) = 0$, producing an estimated source parameter $\hat{\rho}_d = 1/2$, i.e., the value in the center of the interval $(0,1)$.

The probability distribution defined by this prior is

$$\begin{aligned} p_m(s,\Sigma_s,N_s,\Pi_{a,b}) &= (N_s+1)\frac{\Gamma(\Sigma_s+s+1)\Gamma(N_s+\Sigma_s+2)}{\Gamma(\Sigma_s+1)\Gamma(N_s+\Sigma_s+s+3)} \times \\ &\quad \frac{I_b(\Sigma_s+s+1,N_s+2) - I_a(\Sigma_s+s+1,N_s+2)}{I_b(\Sigma_s+1,N_s+1) - I_a(\Sigma_s+1,N_s+1)}. \end{aligned} \tag{4.40}$$

The values $a = 0$ and $b = 1$ yield

$$\begin{aligned} p_m(s,\Sigma_s,N_s,\Pi_{0,1}) &= (N_s+1)\frac{\Gamma(\Sigma_s+s+1)\Gamma(N_s+\Sigma_s+2)}{\Gamma(N_s+\Sigma_s+s+3)\Gamma(\Sigma_s+1)} \\ &= \frac{N_s+1}{N_s+\Sigma_s+s+2} \prod_{i=1}^{N_s+1} \frac{\Sigma_s+i}{\Sigma_s+s+i} \end{aligned} \tag{4.41}$$

while the average symbol value is

$$\bar{s}(\Sigma_s,N_s,\Pi_{0,1}) = \frac{\Sigma_s+1}{N_s}. \tag{4.42}$$

(In Section 4.3 we discuss why the average is undefined when $N_s = 0$.)

The reverse cumulative distribution (4.32) in this special case is equal to

$$\begin{aligned}\bar{F}_m(s, \Sigma_s, N_s, \Pi_{0,1}) &= \frac{\Gamma(\Sigma_s + s + 1)\Gamma(N_s + \Sigma_s + 2)}{\Gamma(N_s + \Sigma_s + s + 2)\Gamma(\Sigma_s + 1)} \\ &= \prod_{i=1}^{N_s+1} \frac{\Sigma_s + i}{\Sigma_s + s + i}.\end{aligned} \tag{4.43}$$

4.2.3 Dirichlet Prior

The Dirichlet (1/2) prior distribution [23] yields more conservative estimates of the source, by assigning higher probabilities to the extreme values of ρ. It is defined by

$$D_{1/2}(\rho) = \frac{1}{\pi\sqrt{\rho(1-\rho)}}, \quad \rho \in (0, 1). \tag{4.44}$$

The integrals for this prior can be computed using the fact that they are similar to those defined for the uniform prior:

$$\begin{aligned}\Phi(m, n, D_{1/2}) &= \frac{1}{\pi}\,\Phi(m - 1/2, n - 1/2, \Pi_{0,1}), \\ &= \frac{1}{\pi}\,B(m - 1/2, n - 1/2) = \frac{\Gamma(m - 1/2)\Gamma(n - 1/2)}{\pi\Gamma(m + n - 1)}.\end{aligned} \tag{4.45}$$

The corresponding estimated source parameter is

$$\hat{\rho}_d = \frac{L(d) + \Sigma_s + 1/2}{L(d) + \Sigma_s + N_s + 1}, \quad d = 0, 1, \ldots \tag{4.46}$$

and the posterior probabilities are

$$p_m(s, \Sigma_s, N_s, D_{1/2}) = (N_s + 1/2)\frac{\Gamma(\Sigma_s + s + 1/2)\Gamma(N_s + \Sigma_s + 1)}{\Gamma(N_s + \Sigma_s + s + 2)\Gamma(\Sigma_s + 1/2)}. \tag{4.47}$$

The average symbol value, defined only when $N_s > 1$, is

$$\bar{s}(\Sigma_s, N_s, D_{1/2}) = \frac{2\Sigma_s + 1}{2N_s - 1}, \tag{4.48}$$

and

$$\bar{F}_m(s, \Sigma_s, N_s, D_{1/2}) = \frac{\Gamma(\Sigma_s + s + 1/2)\Gamma(N_s + \Sigma_s + 1)}{\Gamma(N_s + \Sigma_s + s + 1)\Gamma(\Sigma_s + 1/2)}. \tag{4.49}$$

Note that in this case we have

$$p_m(s, \Sigma_s, N_s, D_{1/2}) = \left(\frac{N_s + 1/2}{N_s + \Sigma_s + s + 1}\right)\bar{F}_m(s, \Sigma_s, N_s, D_{1/2}) \tag{4.50}$$

4.3 Uniform Prior with No Context

Assuming that the prior distribution of ρ is uniform in the interval $(0,1)$, i.e., $f(\rho) = \Pi_{0,1}(\rho)$, the posterior distribution, computed without the knowledge of any source sample, is defined by

$$p_m(s, \Pi_{0,1}) = \int_0^1 \rho^s (1-\rho)\, d\rho = \frac{1}{(s+1)(s+2)}, \quad s = 0, 1, 2, \ldots \tag{4.51}$$

Note that these probabilities correspond to those in (4.41), when $\Sigma_s = N_s = 0$.

If we also assume that we know that the source symbol is not larger than a value n, i.e., instead of previous source samples, the current knowledge is partial information about the current sample, we can define $r = s - n$ and compute the distribution

$$p_c(r, n) = \frac{\displaystyle\int_0^1 \rho^{r+n}(1-\rho)\, d\rho}{\displaystyle\int_0^1 \rho^n\, d\rho} = \frac{n+1}{(r+n+1)(r+n+2)}, \quad n, r = 0, 1, 2, \ldots \tag{4.52}$$

We can further generalize this distribution by using as parameter a positive real number α instead of $n+1$, in the form

$$p_c(s, \alpha) = \frac{\alpha}{(\alpha+s)(\alpha+s+1)}, \quad s = 0, 1, 2, \ldots, \quad \alpha > 0. \tag{4.53}$$

We call this distribution *Discrete-Cauchy* (DC) because it has some properties that are similar to the Cauchy continuous distribution.

A convenient property of the DC distribution is that the sum of the probabilities "telescope," simplifying their computation. For instance, the cumulative distribution is

$$\begin{aligned} F_c(s, \alpha) &= \sum_{r=0}^{s-1} p_c(r, \alpha) \qquad (4.54) \\ &= 1 - \frac{\alpha}{\alpha+1} + \frac{\alpha}{\alpha+1} - \frac{\alpha}{\alpha+2} + \cdots + \frac{\alpha}{\alpha+s-1} - \frac{\alpha}{\alpha+s} \\ &= 1 - \frac{\alpha}{\alpha+s} = \frac{s}{\alpha+s}. \end{aligned}$$

The DC distribution has a very "heavy tail," and the probabilities go to zero with factors proportional to $1/s^2$. In fact, this distribution does not have an average value, since

$$\begin{aligned} \sum_{r=0}^{s-1} r\, p_c(r, \alpha) &= \sum_{r=0}^{s-1} \left(\frac{\alpha r}{\alpha+r} - \frac{\alpha r}{\alpha+r+1} \right) = \sum_{r=1}^{s-1} \frac{\alpha r}{\alpha+r} - \sum_{r=1}^{s} \frac{\alpha(r-1)}{\alpha+r} \\ &= \sum_{r=1}^{s-1} \frac{\alpha}{\alpha+r} - \frac{\alpha(s-1)}{\alpha+s}, \end{aligned}$$

and consequently

$$\lim_{s\to\infty} \sum_{r=0}^{s} r\, p_c(r,\alpha) = \infty. \tag{4.55}$$

The series that define the rate for Golomb codes also diverge for this type of distribution.

Proposition 4.3.1 *The average bit rate produced by any Golomb code applied to a source with discrete Cauchy distribution is infinite.*

Proof: For this proof we use formula (3.10) for computing the average bit rate of any unary-stem code, with arbitrary distributions. We need to know the reverse cumulative distribution for the discrete Cauchy distribution, which is readily computed from (4.54)

$$\bar{F}_c(s) = \frac{\alpha}{\alpha + s}. \tag{4.56}$$

Furthermore, we use the fact that for a Golomb code with parameter m we have $L_G(d) = md$.

We divide the proof in two cases. First, assume that the parameter m of the Golomb code is a power of two. In this case the average rate (3.10) is

$$\begin{aligned} B_c(\mathcal{M},\alpha) &= 1 + \lfloor \log_2(m) \rfloor + \sum_{d=1}^{\infty} \bar{F}_c(L_G(d)) \\ &= 1 + \lfloor \log_2(m) \rfloor + \sum_{d=1}^{\infty} \frac{\alpha}{\alpha + md} \\ &= 1 + \lfloor \log_2(m) \rfloor + \frac{\alpha}{m} \sum_{d=1}^{\infty} \frac{1}{d + \alpha/m}. \end{aligned} \tag{4.57}$$

From the inequality

$$\sum_{d=1}^{\infty} \frac{1}{d + \alpha/m} \geq \sum_{d=1+\lceil \alpha/m \rceil}^{\infty} \frac{1}{d}, \tag{4.58}$$

and since we know the harmonic series diverges, we conclude that the average is infinite.

The second case is quite similar. We have the rate defined by

$$\begin{aligned} B_c(\mathcal{M},\alpha) &= 1 + \lfloor \log_2(m) \rfloor + \sum_{d=1}^{\infty} \bar{F}_c(L_G(d) + \tau(m)) \\ &= 1 + \lfloor \log_2(m) \rfloor + \sum_{d=1}^{\infty} \frac{\alpha}{\alpha + md + \tau(m)} \\ &= 1 + \lfloor \log_2(m) \rfloor + \frac{\alpha}{m} \sum_{d=1}^{\infty} \frac{1}{d + [\alpha + \tau(m)]/m}. \end{aligned} \tag{4.59}$$

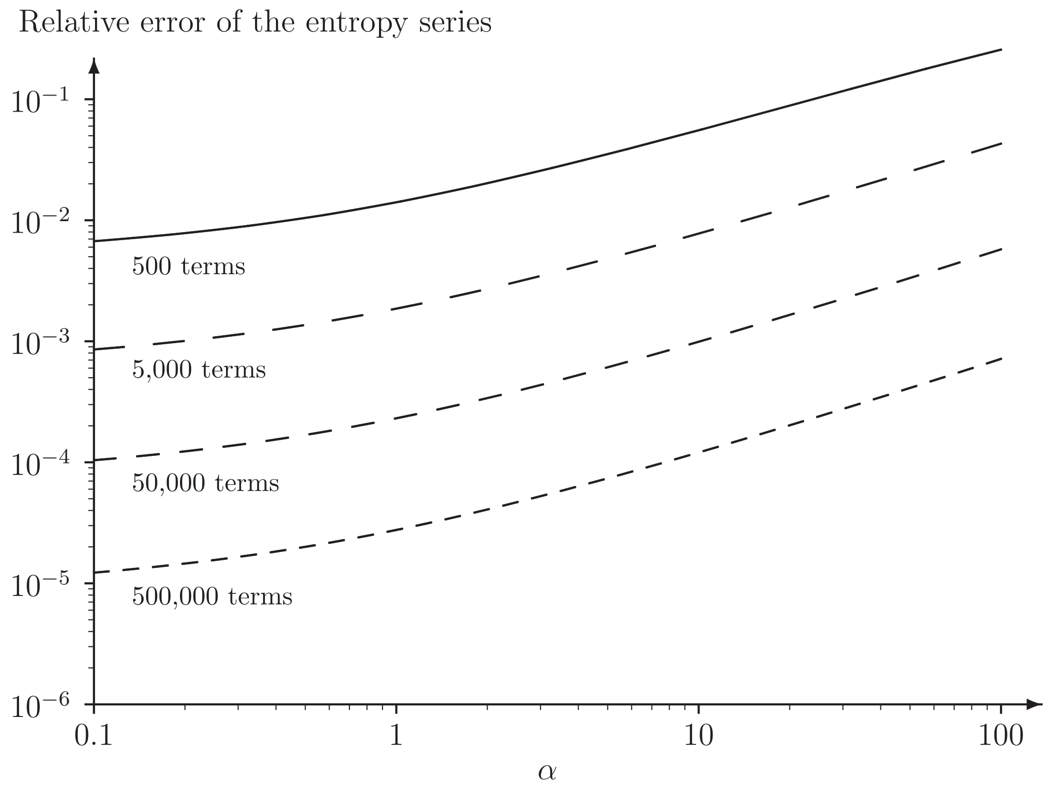


Figure 4.4: Relative error in the direct computation of the entropy of the DC distribution using series (4.60).

which has the same type of divergent sum. ∎

The entropy of the DC distribution is defined by

$$
\begin{aligned}
H_c(\alpha) &= \sum_{s=0}^{\infty} p_c(s,\alpha) \log_2\left(\frac{1}{p_c(s,\alpha)}\right) \\
&= \sum_{s=0}^{\infty} \left(\frac{\alpha}{[\alpha+s][\alpha+s+1]}\right) \log_2\left(\frac{[\alpha+s][\alpha+s+1]}{\alpha}\right). \qquad (4.60)
\end{aligned}
$$

The very slow decay of the probabilities creates problems for the computation of entropy. Figure 4.4 shows how the relative error varies with the parameter α and the number of terms used in series (4.60). Note that a very large number of terms is needed even for relatively low accuracy. This can be a problem during our code analysis, since sufficiently high accuracy is needed for measuring and comparing coding redundancy, when we subtract numbers that are very close to the entropy. In Appendix B we derive series for computing the entropy of this distribution that are much more efficient.

A very good approximation of the entropy is (B.4) ($t = 1$, Figure B.2)

$$H_c(\alpha) \approx \log_2(\alpha+1) + \left(\frac{\alpha}{\alpha+1}\right)\log_2\left(\frac{\alpha+2}{\alpha}\right) + \frac{2\alpha}{\ln(2)(\alpha+3/2)}. \tag{4.61}$$

Using the bounds derived in Appendix B we can also show that

$$\log_2(\alpha+1) \le H_c(\alpha) \le \log_2(\alpha+1) + \frac{2}{\ln(2)}. \tag{4.62}$$

Next we show that—quite differently from the Golomb codes—the Elias-Teuhola codes can efficiently code symbols from the DC sources. First, let us present some results needed for computing the bit rates.

Lemma 4.3.2 *For an Elias-Teuhola code code with parameter k applied to a DC source distribution with parameter α we have a sum of codewords $L_U(d, \mathcal{M})$ (3.1) equal to*

$$L_E(d,k) = 2^k\,(2^d - 1), \tag{4.63}$$

while the samples of the reverse cumulative distribution are

$$\bar{F}_c(L_E(d,k),\alpha) = \frac{\alpha}{\alpha + 2^k(2^d-1)}, \tag{4.64}$$

and when $\alpha = 2^k$ we have

$$\bar{F}_c(L_E(d,k),2^k) = 2^{-d}. \tag{4.65}$$

Proof: We can compute $L_U(d, \mathcal{M})$ using (3.1), noting that Elias-Teuhola codes are unary-stem codes with parameter sequence $\mathcal{M} = \{2^k, 2^{k+1}, 2^{k+2}, \ldots\}$. Thus, the sum is

$$L_E(d,k) = \sum_{i=0}^{d-1} 2^{k+i} = 2^k\,(2^d-1). \tag{4.66}$$

The substitution of these values into

$$\bar{F}_c(s,\alpha) = \frac{\alpha}{\alpha+s} \tag{4.67}$$

and the identity

$$\bar{F}_c(L_E(d,k),2^k) = \frac{2^k}{2^k + 2^k(2^d-1)} = 2^{-d}, \tag{4.68}$$

yield the other results. ∎

Using the results in Lemma 4.3.2 we can compute the average bit rates.

Proposition 4.3.3 *The average bit rate produced by an Elias-Teuhola code code with parameter k applied to a DC source with parameter α is*

$$B_E(k,\alpha) = 1 + k + 2^{1-k}\alpha \sum_{d=1}^{\infty} \frac{1}{2^d - 1 + 2^{-k}\alpha}, \tag{4.69}$$

and in particular

$$B_E(k, 2^k) = k + 3. \tag{4.70}$$

Furthermore, this rate is bounded by

$$k + 2\log_2(2^{-k}\alpha + 1) \leq B_E(k,\alpha) \leq 1.05 + k + 2\log_2(2^{-k}\alpha + 1), \tag{4.71}$$

Proof: The substitution of $L(d) = L_E(d,k)$ (from Lemma 4.3.2), $k_0 = k$, and $k_d - k_{d-1} = 1$ into the average rate equation (3.13) yields

$$B_E(k,\alpha) = 1 + k + 2\sum_{d=1}^{\infty} \frac{\alpha}{\alpha + 2^k(2^d - 1)}, \tag{4.72}$$

which is equivalent to equation (4.69).

The sums of codewords for $\alpha = 2^k$ are also defined in Lemma 4.3.2, and the substitution yields

$$B_E(k, 2^k) = 1 + k + 2\sum_{d=1}^{\infty} 2^{-d} = k + 3. \tag{4.73}$$

The lower and upper bounds are defined by observing that

$$\lambda(x) = \sum_{d=1}^{\infty} \frac{x}{2^d + x - 1} \approx \log_2(x+1). \tag{4.74}$$

The graph of $\lambda(x) - \log_2(x+1)$ in Figure 4.5 shows that the error of this approximation is inside the interval $[-0.5, 0.025]$, which defines the bounds on $B_E(k,\alpha)$. ∎

Better approximations for this bit rate can be defined by using correction terms that are ratios of polynomials. One solution that minimizes the relative error is

$$\lambda(x) \approx \log_2(x+1) + \frac{x(1-x)(1+0.955x)}{6.1 + x[14.57 + x(14.25 + 2x)]}. \tag{4.75}$$

Figure 4.6 shows that the maximum relative error of this approximation is below 0.2%.

We can find the optimal parameter k for Elias-Teuhola codes on DC sources using the equation for the average rate (4.69).

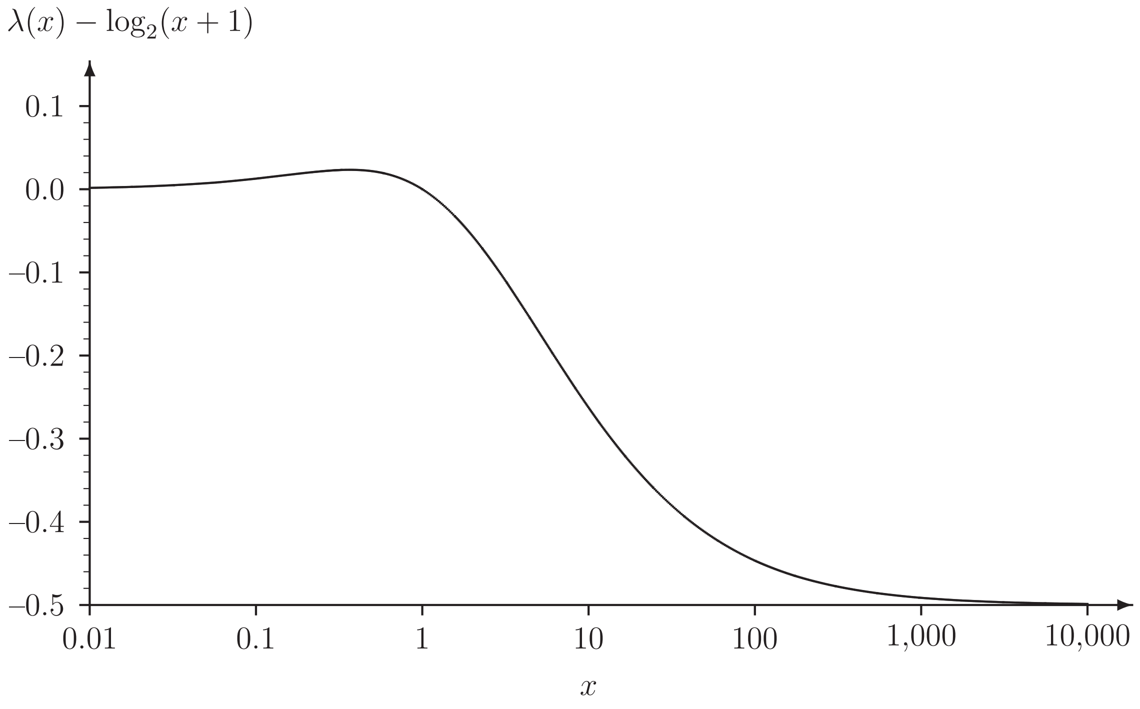


Figure 4.5: Error of approximation (4.74), that defines bounds on the bit rate of Elias-Teuhola codes in DC sources.

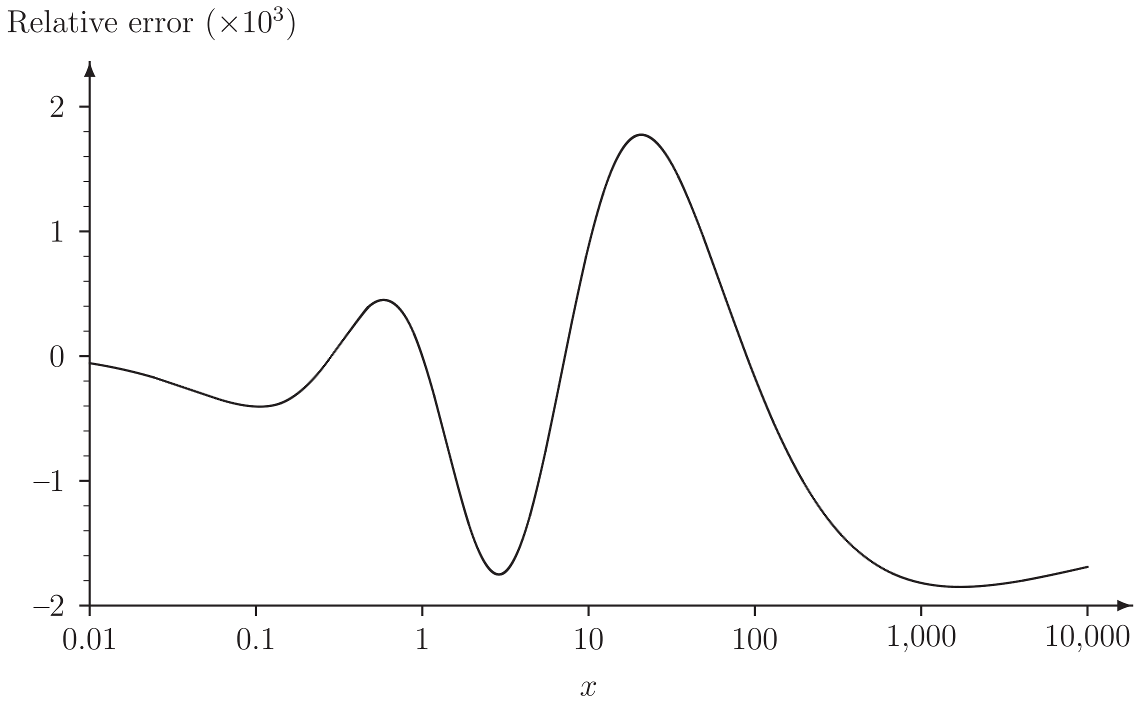


Figure 4.6: Relative error of approximation (4.75), for computing the bit rate of Elias-Teuhola codes in DC sources.

Proposition 4.3.4 *The parameter k of the Elias-Teuhola code that minimizes the average bit rate needed for coding symbols from a source with DC distribution is*

$$k^*(\alpha) = \max\{0, \lfloor \log_2(1.1633\alpha) \rfloor\}. \tag{4.76}$$

Proof: We can use the fact that the bit rate (4.69) can be written as

$$B_E(k, \alpha) = 1 + k + 2\lambda(2^{-k}\alpha), \tag{4.77}$$

where $\lambda(x)$ is defined by (4.74). The optimality conditions are

$$\begin{aligned} B_E(k^* - 1, \alpha) \geq B_E(k^*, \alpha) &\implies \lambda(2^{1-k^*}\alpha) \geq \lambda(2^{-k^*}\alpha) + 1/2, \\ B_E(k^* + 1, \alpha) \geq B_E(k^*, \alpha) &\implies \lambda(2^{-k^*}\alpha) \leq \lambda(2^{-k^*-1}\alpha) + 1/2. \end{aligned} \tag{4.78}$$

Note that these conditions correspond to the same type of equation, for different values of k. Consequently the optimal solution k^* must satisfy the inequalities

$$2^{k^*} \leq b\,\alpha \leq 2^{k^*+1}, \tag{4.79}$$

where $b = 1.1633$ is a constant defined by the unique numerical solution α when we set (4.78) as equality (transition point from $k^* - 1$ to k^*). ∎

Figure 4.7 shows the relative redundancy of Elias-Teuhola codes when applied to sources with the DC distribution, using the optimal parameter k. Note that the relative redundancy is quite low, typically in the range between 1% and 3%. It is also interesting to note that in a way the Elias-Teuhola codes are "matched" to these distributions because the redundancy in the unary code is very small or zero.

Proposition 4.3.5 *The redundancy of the unary part of the Elias-Teuhola code with parameter k is zero when applied to a DC source with parameter $\alpha = 2^k$.*

Proof: Under these conditions we have $\bar{F}(L(d))$ defined by (4.65), and

$$\bar{F}(L(d)) - \bar{F}(L(d+1)) = 2^{-(d+1)}. \tag{4.80}$$

The substitution into the equation for computing the redundancy of the unary code (3.22) results in

$$\delta_E(k, 2^k) = 1 + \log_2\bigl(1 - 2^{-1}\bigr) + \sum_{d=1}^{\infty} 2^{-d} \left\{1 - \log_2\left(\frac{2^{-d}}{2^{-(d+1)}}\right)\right\} = 0. \tag{4.81}$$

∎

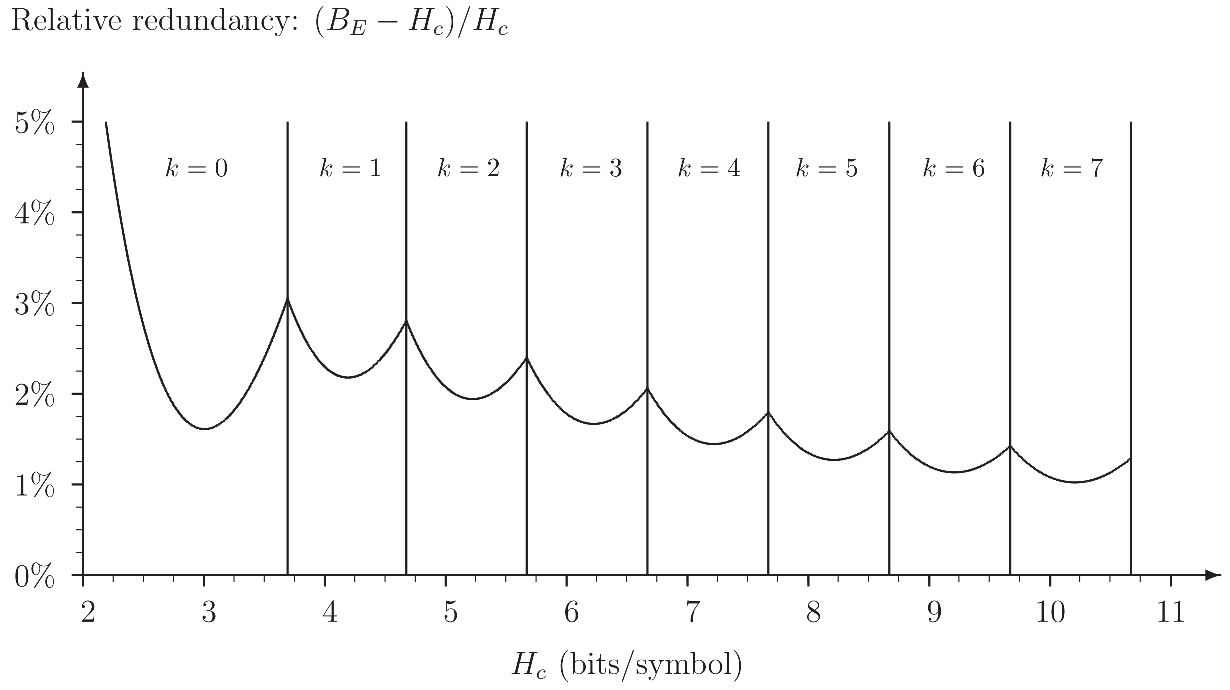


Figure 4.7: Relative coding redundancy of optimal Elias-Teuhola codes as a function of the DC source entropy.

4.4 Entropy Computation

We observed in Section 4.3 that using a uniform prior produces marginal distributions that have a very "heavy tail," and consequently the series for computing the entropy converges very slowly. This problem is even worse with the Dirichlet (1/2) prior, which defines probabilities with asymptotic decay roughly proportional to $s^{-(N_s+3/2)}$. Since imprecise rate and entropy computations can affect the results produced by the code optimization techniques described in Chapter 5, it is interesting to analyze fast methods for entropy computation. Some very effective acceleration techniques are presented in Appendix B, but they are specific to the DC distribution. In this section we consider techniques that are less precise, but much more general. The main objective is to define approximations that yield reasonably good accuracy with small computational effort.

For the computation of the partial entropy sums (2.9) we have, for $s = 0, 1, 2, \ldots,$

$$\begin{aligned} \bar{h}(s) &= \sum_{i=s}^{\infty} p(i) \log_2\left(\frac{1}{p(i)}\right) \qquad (4.82) \\ &= \sum_{i=s}^{\infty} [\bar{F}(i+1) - \bar{F}(i)]\ \log_2(p(i)) \end{aligned}$$

$$
\begin{aligned}
&= \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) + \sum_{i=1}^{\infty} \bar{F}(s+i)\ \log_2\left(\frac{p(s+i-1)}{p(s+i)}\right) \\
&= \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) + \sum_{i=1}^{\infty} \bar{F}(s+i)\ \log_2\left(1 + \frac{p(s+i-1)-p(s+i)}{p(s+i)}\right).
\end{aligned}
$$

The advantage of entropy series in this form is that the logarithm values go to zero when $s+i$ goes to infinity, while in (2.9) the logarithm goes to infinity. This can make the series converge somewhat faster for heavy tailed distributions (e.g., compare Figures 4.6 and B.1), and can also yield fast computation of approximations using $\ln(1+x) \approx x$.

Using equation (4.27) for determination of the posterior probabilities, and definition (4.24), we obtain

$$
\frac{p_m(s-1,\Sigma_s,N_s,f) - p_m(s,\Sigma_s,N_s,f)}{p_m(s,\Sigma_s,N_s,f)} = \frac{\Phi(\Sigma_s+s,N_s+3,f)}{\Phi(\Sigma_s+s+1,N_s+2,f)}. \tag{4.83}
$$

If a continuous reverse cumulative function $\bar{F}(s)$ is defined for real-valued s, and with its two first derivatives denoted by

$$
\bar{F}'(s) = \left.\frac{d\bar{F}(x)}{dx}\right|_{x=s} \qquad \bar{F}''(s) = \left.\frac{d^2\bar{F}(x)}{dx^2}\right|_{x=s}, \tag{4.84}
$$

then we can use the following approximations

$$
\begin{aligned}
\bar{h}(s) &= \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) + \\
&\quad \sum_{i=1}^{\infty} \bar{F}(s+i)\ \log_2\left(1 + \frac{\bar{F}(s+i-1)+\bar{F}(s+i+1)-2\bar{F}(s+i)}{\bar{F}(s+i)-\bar{F}(s+i+1)}\right) \\
&\approx \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) + \sum_{i=1}^{\infty} \bar{F}(s+i)\log_2\left(1 - \frac{\bar{F}''(s+i)}{\bar{F}'(s+i)}\right) \\
&\approx \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) - \frac{1}{\ln(2)}\sum_{i=1}^{\infty} \frac{\bar{F}(s+i)\bar{F}''(s+i)}{\bar{F}'(s+i)}.
\end{aligned} \tag{4.85}
$$

The posterior probabilities defined in this chapter can be approximated, for sufficiently large s, by

$$
\bar{F}(s+i) \approx t\,(\alpha+i)^{-\beta}. \tag{4.86}
$$

For instance, the distributions defined by the uniform prior have $\beta = N_s+1$, while for those defined by the Dirichlet (1/2) prior have $\beta = N_s + 1/2$.

Using (4.85) we obtain the approximation

$$
\begin{aligned}
\bar{h}(s) &\approx \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) + \frac{t(\beta+1)}{\ln(2)}\ \sum_{i=1}^{\infty}(\alpha+i)^{-(\beta+1)} \\
&\approx \bar{F}(s)\log_2\left(\frac{1}{p(s)}\right) + \frac{t(\beta+1)}{\ln(2)}\ \zeta(\beta+1,\alpha+1),
\end{aligned} \tag{4.87}
$$

where

$$\zeta(s,a) = \sum_{i=0}^{\infty}(a+i)^{-s} \tag{4.88}$$

is the Hurwitz zeta function [6, § 12].

The advantage of this series is that there are well-known methods to efficiently compute the value of the Hurwitz zeta function [11]: it is supported by high-level languages (e.g., Mathematica), or freely available implementations (e.g., GNU Scientific Library).

Chapter 5

Determination of Optimal Unary-Stem Codes

The Huffman algorithm [1, 26] is well-known as a simple and very efficient technique for the determination of optimal prefix codes. However, it can only be used with the following assumptions:

1. given a data source with known symbol probabilities, the objective is to find the prefix code that minimizes the average number of bits per symbol;
2. the number of symbols is finite;
3. there are no constraints or special requirements on the symbols (e.g., bits) that form the codewords.

The unary-stem codes described in Section 3.1 violate the last two assumptions, since they have an infinite number symbols, and the codewords need to be defined using the special unary+binary structure.

Depending on the situation, the second assumption may not be important for practical applications. It is always possible to obtain nearly optimal solutions by decomposing the set of data symbols into a first finite set, which contains the most common symbols, and a second infinite set, with the remaining symbols. The Huffman algorithm can be applied to the finite set plus an extra symbol representing the second set, and the symbols in the second set can be coded using some general form (e.g., see [4]). However, there are also limits to this approach: there are problems with the application of the Huffman algorithm to very large alphabets, including computation time and memory for storing code tables.[1]

A main conceptual problem with infinite codes is in the definition of optimality. An infinite solution is useful only if it can be generated with some reasonable efficiency, which

[1]In fact, as explained in Section 1.1, this motivates the use of parameterized codes, like the unary-stem codes.

requires some type of repetitive or regular structure. While there are some impressive theoretical results concerning the optimality of infinite prefix codes [5, 22], they all possess some special property. Since in the general case the codes are not expected to have any type of regularity, there is little incentive to consider the full optimization of an infinite number of parameters.

Constraints on the codewords, on the other hand, are commonly defined by practical requirements, and thus cannot be easily circumvented. Unfortunately, the Huffman algorithm is not very flexible, i.e., there are no known simple modifications that allow the incorporation of constraints in the codeword creation. In order to deal with constraints on codeword construction, or codeword "costs," it is necessary to use very different optimization techniques [24].

In this chapter we consider algorithms for the determination of optimal unary-stem codes that are not based on a simple truncation of the source alphabet to a finite number of symbols, but instead always take into account the full infinite alphabet. This is done by combining two approaches. First, we do not try to determine the exact optimal solutions. Instead, we look for finite sets of code design decisions that we can guarantee to be sufficiently close to the optimal. This means optimizing a finite part of the code, and assuming a remaining infinite part that may be suboptimal, but has little consequence in the expected code performance. Second, we use the fact that the bit rate of unary-stem codes can be conveniently computed using reverse cumulative distributions $\bar{F}(s)$ in (3.10), which can be known in closed-form (e.g., equations (4.32), (4.43), and (4.49)). Thus, even with a finite set of code design decisions, we can still evaluate the resulting contribution of an infinite number of symbols on the average bit rate (cf. Section 5.2).

5.1 Some Necessary Optimality Conditions

If the symbol probabilities $\mathcal{P}$ form a strictly decreasing sequence, and if these probabilities decay fast enough, we can use some simple tests to check optimality of unary-stem codes.

Proposition 5.1.1 *A unary-stem code $\mathcal{M} = \{m_0, m_1, \ldots\}$ is optimal for a decreasing sequence of symbol probabilities $\mathcal{P}$ only if*

$$\lfloor \log_2(m_d) \rfloor \geq \lceil \log_2(m_{d-1}) \rceil - 1, \quad d = 1, 2, 3, \ldots \tag{5.1}$$

Proof: Given $\mathcal{M}$, for any $d > 0$ we have sets of symbol pairs, (s_a, s_b), satisfying

$$L(d-1) \leq s_a < L(d) \leq s_b < L(d+1)$$

and such that the codeword length for symbol s_a is $\lceil \log_2(m_{d-1}) \rceil$ and the codeword length for symbol s_b is $1 + \lfloor \log_2(m_d) \rfloor$. Since we have decreasing probabilities, if inequality (5.1) is not satisfied then the solution cannot be optimal because the length of the codeword of the more probable symbol s_a would be larger than the length of the codeword of the less probable symbol s_b. ∎

Corollary 5.1.2 *A dyadic unary-stem code $\mathcal{K} = \{k_0, k_1, \ldots\}$ is optimal for a decreasing sequence $\mathcal{P}$ only if*

$$k_d \geq k_{d-1} - 1, \quad d = 1, 2, 3, \ldots \tag{5.2}$$

From these results we can conclude that when $\mathcal{P}$ is a decreasing sequence the corresponding optimal unary-stem codes are such that in the series for computation of the code rates, (3.10) or (3.13), all elements are non-negative.

If the probabilities in $\mathcal{P}$ decrease sufficiently fast, we can know in advance if $m_d \geq m_{d-1}$ is a necessary optimality condition using the following results.

Proposition 5.1.3 *A unary-stem code $\mathcal{M} = \{m_0, m_1, \ldots\}$ is optimal for a sequence of symbol probabilities $\mathcal{P}$ only if*

$$r(L(d), m_d, m_{d+1}) \leq r(L(d), m_{d+1}, m_d), \quad d = 0, 1, 2, \ldots \tag{5.3}$$

where

$$\begin{aligned} r(s, m, n) \;=\; & \bar{F}(s + \tau(m))\ \{\lceil \log_2(m) \rceil - \lfloor \log_2(m) \rfloor\} + \\ & \bar{F}(s + m)\ \{1 + \lfloor \log_2(n) \rfloor - \lceil \log_2(m) \rceil\} + \\ & \bar{F}(s + m + \tau(n))\ \{\lceil \log_2(n) \rceil - \lfloor \log_2(n) \rfloor\} + \\ & \bar{F}(s)\ \lfloor \log_2(m) \rfloor - \bar{F}(s + m + n)\ \{1 + \lceil \log_2(n) \rceil\}. \end{aligned} \tag{5.4}$$

Proof: This condition corresponds to comparing the bit rate $B_U(\mathcal{M}, \mathcal{P})$, using (3.10), of parameter sequences $\mathcal{M}$ and $\mathcal{M}'$, that differ only by the interchange of two adjacent parameters, i.e., we have $m_d = m'_{d+1} = a$ and $m_{d+1} = m'_d = b$. Since only two parameters are different, only two values of $L(d)$ are different, and we can easily compare the bit rates, which defines the formulas above. ∎

Corollary 5.1.4 *A dyadic unary-stem code $\mathcal{K} = \{k_0, k_1, \ldots\}$ is optimal for a sequence of symbol probabilities $\mathcal{P}$ only if*

$$r'(L(d), k_d, k_{d+1}) \leq r'(L(d), k_{d+1}, k_d), \quad d = 0, 1, 2, \ldots \tag{5.5}$$

where

$$r'(s, k, l) = \bar{F}(s)\, k + \bar{F}(s + 2^k)\ [1 + l - k] + \bar{F}(s + 2^k + 2^l)\ [1 + l] \tag{5.6}$$

5.2 Rates Defined by Finite Sets of Decisions

The optimization methods described in this chapter evaluate sets of finite numbers of decisions by computing the components of the bit rates that are shared by all the codes defined by those decisions. Assuming that only the first c elements of $\mathcal{M}$ had been chosen, we denote the vectors with these values by

$$\mathbf{m} = [m_0 \; m_1 \; \dots \; m_{c-1}]. \tag{5.7}$$

Similarly, for dyadic unary-stem codes we define

$$\mathbf{k} = [k_0 \; k_1 \; \dots \; k_{c-1}]. \tag{5.8}$$

In order to rigorously define the rate of unary-stem codes we need to define all the elements in the infinite sequence $\mathcal{M}$, even when we know that the contribution of the trailing elements $\{m_c, m_{c+1}, \dots\}$ to the average bit rate is extremely small. A convenient way to deal with this problem is to define a "standard" extension to a partial solution, which has an infinite number of parameters, but that are defined with a simple rule. We used the extensions

$$\mathcal{X}_c(\mathbf{m}) = \{m_0, m_1, m_2, \dots, m_{c-1}, 2^{\lceil \log_2(m_{c-1}) \rceil + 1}, 2^{\lceil \log_2(m_{c-1}) \rceil + 2}, \dots\}, \tag{5.9}$$

and

$$\mathcal{X}_c(\mathbf{k}) = \{k_0, k_1, k_2, \dots, k_{c-1}, k_{c-1}+1, k_{c-1}+2, k_{c-1}+3, \dots\}. \tag{5.10}$$

Let $\mathfrak{U}_c(\mathbf{m})$ represent the set of all infinite sequences $\mathcal{M}$ of non-negative integers that have their first elements equal to the elements of $\mathbf{m}$, i.e.,

$$\mathfrak{U}_c(\mathbf{m}) = \left\{ \{n_0, n_1, \dots\} : n_0 = m_0, \; \dots, n_{c-1} = m_{c-1}, \quad n_i \in \mathbb{Z}_+, \; i = c, c+1, \dots \right\}, \tag{5.11}$$

and similarly define $\mathfrak{U}_c(\mathbf{k})$. We can observe that, for example, $\mathcal{X}_c(\mathbf{m}) \in \mathfrak{U}_c(\mathbf{m})$.

Note that to simplify notation, in this chapter we assume that c, and the different uses of $\mathbf{m}$ and $\mathbf{k}$ are known from the context. We also keep using the simplified notation for the sums $L(d)$

$$L(d) = L_c(d, \mathbf{m}) = \sum_{s=0}^{d-1} m_s, \quad d = 0, 1, \dots, c, \tag{5.12}$$

and

$$L(d) = L_c(d, \mathbf{k}) = \sum_{s=0}^{d-1} 2^{k_s}, \quad d = 0, 1, \dots, c. \tag{5.13}$$

However, the meaning of $L(d)$ does not change.

Using these definitions we can compute a set of average bit rate values related to the decisions defined by $\mathbf{m}$ or $\mathbf{k}$.

5.2.1 Committed Rate

Given a source distribution $\mathcal{P}$, we define the *committed rate* $R_c(\mathbf{m}, \mathcal{P})$ as the part of the average rate that is equal in all the unary-stem codes that belong to $\mathfrak{U}_c(\mathbf{m})$, and that can be computed using only the values defined by $\mathbf{m}$. Since the c first parameters of $\mathcal{M}$ define $L(1), L(2), \ldots, L(c)$, with $L(0) \equiv 0$, using these known values in the series (3.10) gives

$$\begin{aligned} R_c(\mathbf{m}, \mathcal{P}) &= \sum_{d=0}^{c} \bar{F}(L(d)) + \sum_{d=0}^{c-1} \left\{ \bar{F}(L(d)) \lfloor \log_2(m_d) \rfloor - \bar{F}(L(d+1)) \lceil \log_2(m_d) \rceil \right\} + \\ & \quad \sum_{d=0}^{c-1} \bar{F}(L(d) + \tau(m_d)) \left\{ \lceil \log_2(m_d) \rceil - \lfloor \log_2(m_d) \rfloor \right\}. \end{aligned} \tag{5.14}$$

It is important to note that these bit rates include all the data symbols, and not only the symbols that have the codeword length fully defined. The unary part of the code at each stage adds a bit for all the remaining symbols, and the corresponding rate is defined by the term $\bar{F}(L(d))$. Thus, even when no choice is made (i.e., $c = 0$) we can define a nonzero committed rate $R_0 \equiv 1$, because all prefix codes need to use on average at least one bit per symbol.

The committed rate of dyadic unary-stem codes is again obtained with a straightforward substitution of $\lfloor \log_2(m_d) \rfloor = \lceil \log_2(m_d) \rceil = k_d$, which results in

$$R_c(\mathbf{k}, \mathcal{P}) = \sum_{d=0}^{c} \bar{F}(L(d)) + \sum_{d=0}^{c-1} k_d [\bar{F}(L(d)) - \bar{F}(L(d+1))]. \tag{5.15}$$

5.2.2 Upper and Lower Bounds

The minimum average rate obtained by all codes with the same initial parameters is denoted by

$$B_c^*(\mathbf{m}, \mathcal{P}) = \min_{\mathcal{M} \in \mathfrak{U}_c(\mathbf{m})} \{ B_U(\mathcal{M}, \mathcal{P}) \}, \tag{5.16}$$

and the parameters of the best codes are defined by the set of sequences

$$\mathfrak{M}_c^*(\mathbf{m}, \mathcal{P}) = \{ \mathcal{M} : B_U(\mathcal{M}, \mathcal{P}) = B_c^*(\mathbf{m}, \mathcal{P}) \}. \tag{5.17}$$

Using the committed rate we can also define

$$E_c(\mathbf{m}, \mathcal{P}) = R_c(\mathbf{m}, \mathcal{P}) + \bar{h}(L(c)) - \bar{F}(L(c)) \left[1 + \log_2\left(\frac{1}{\bar{F}(L(c))} \right) \right], \tag{5.18}$$

and

$$U_c(\mathbf{m}, \mathcal{P}) = R_c(\mathbf{m}, \mathcal{P}) + \kappa_c \bar{F}(L(c)) + 2 \sum_{d=1}^{\infty} \bar{F}(L(c) + (2^d - 1)\, 2^{\kappa_c}), \tag{5.19}$$

where $\kappa_c = 1 + \lceil \log_2(m_{c-1}) \rceil$.

Proposition 5.2.1 *Given a source distribution $\mathcal{P}$, the minimum rate $B_c^*(\mathbf{m},\mathcal{P})$ is bounded by*

$$\max\{R_c(\mathbf{m},\mathcal{P}), E_c(\mathbf{m},\mathcal{P})\} \leq B_c^*(\mathbf{m},\mathcal{P}) \leq U_c(\mathbf{m},\mathcal{P}). \tag{5.20}$$

Proof: The committed rate $R_c(\mathbf{m},\mathcal{P})$ is a lower bound because it does not include the bits required for coding the symbols $\{L(c), L(c)+1,\ldots\}$. Instead, it adds only the minimum value of one bit, required for any binary prefix code, scaled by the sum of the probabilities of all those symbols, which is $\bar{F}(L(c))$.

$E_c(\mathbf{m},\mathcal{P})$ is also a lower bound because the difference $E_c(\mathbf{m},\mathcal{P}) - R_c(\mathbf{m},\mathcal{P})$ is the scaled conditional entropy of symbols $\{L(c), L(c)+1,\ldots\}$, which is

$$\sum_{s=L(c)}^{\infty} p(s)\log_2\left(\frac{\bar{F}(L(c))}{p(s)}\right) = \bar{h}(L(c)) + \bar{F}(L(c))\log_2(\bar{F}(L(c))). \tag{5.21}$$

minus $\bar{F}(L(c))$, which corresponds to the single bit added to the committed rate, since for this bound we are considering the minimum for any code (entropy).

The upper bound corresponds to the rate of the standard extension, i.e.,

$$U_c(\mathbf{m},\mathcal{P}) = B_U(\mathcal{X}_c(\mathbf{m}),\mathcal{P}),$$

computed using (3.10) and changing (4.63) for the code defined by the standard extension. Since $\mathcal{X}_c(\mathbf{m}) \in \mathfrak{U}_c(\mathbf{m})$, this rate cannot be smaller than the minimum over all codes in the set. ∎

The adaptation of these results for dyadic unary-stem codes is again straightforward. Note that these bounds can be computed very efficiently using the equations above, since there is no need to count contributions of many individual symbols, and the results can be quickly updated if $\bar{F}(s)$ and $\bar{h}(s)$ are pre-computed and stored in tables.

5.3 Optimization via Implicit Enumeration

The first algorithm that we propose for optimizing unary-stem codes is based on the implicit enumeration of all solutions (or branch-and-bound search) [19, 20]. This is a tree-search technique that enumerates sets of solutions (codes), and uses lower bounds on bit rates in order to identify sets that cannot possibly contain the optimal solution. It will then enumerate all viable candidates to find the optimal. For unary-stem codes the true optimal solution has an infinite number of variables, but the search method can be used to find in finite time one solution that is guaranteed to be sufficiently close to the optimal.

The search algorithm is based on the following result.

Corollary 5.3.1 *Given a unary-stem code defined by the sequence of parameters $\mathcal{M}^b$ and a partial solution* $\mathbf{m}$*, if*

$$\max\{R_c(\mathbf{m},\mathcal{P}), L_c(\mathbf{m},\mathcal{P})\} \le B_U(\mathcal{M}^b,\mathcal{P}), \tag{5.22}$$

then there is no unary-code in the set $\mathfrak{U}_c(\mathbf{m})$ *with average bit rate smaller than that obtained by* $\mathcal{M}^b$*.*

Proof: This follows directly from Proposition 5.2.1. If a lower bound on the minimum rate of all codes in $\mathfrak{U}_c(\mathbf{m})$ is larger that the rate $B_U(\mathcal{M}^b,\mathcal{P})$, then it is clear that no code in that set can be better than $\mathcal{M}^b$. ∎

A standard implicit tree enumeration can then be used to find the optimal codes [19, 20]. A tree structure is defined for enumerating partial solutions $\mathbf{m}$ (or $\mathbf{k}$), with depth corresponding to the number c of elements defined in $\mathbf{m}$. The best known solution $\mathcal{M}^b$ is the code with standard extension $\mathcal{X}(\mathbf{m})$ corresponding to the smallest value of $U_c(\mathbf{m},\mathcal{P})$ found up to that stage in the search.

Since the optimal solution requires an infinite number of decisions (i.e., the branching can continue indefinitely), we limited the search by stopping the branching process whenever

$$U_c(\mathbf{m},\mathcal{P}) - B_U(\mathcal{M}^b,\mathcal{P}) \le 10^{-9} B_U(\mathcal{M}^b,\mathcal{P}),$$

i.e., when we know that there is no better solution within a pre-defined precision.

In addition, since we applied the algorithm to strictly decreasing probabilities, the branching at each node starts with the smallest values of m_d (or k_d) that satisfy the necessary optimality condition defined by Proposition 5.1.1.

5.4 Backward Dynamic Programming Optimization

Before presenting this algorithm we need to define some notation related to optimal codes. Given probability distribution $\mathcal{P}$, let $\mathcal{M}^*(\mathcal{P})$ define the parameters of an optimal unary-stem code, i.e.,

$$B_U(\mathcal{M}^*(\mathcal{P}),\mathcal{P}) = \min_{\mathcal{M}}\{B_U(\mathcal{M},\mathcal{P})\}. \tag{5.23}$$

Note that $\mathcal{M}^*(\mathcal{P})$ does not have to be unique. If there are multiple optimal solutions we assume that one of those solutions had been chosen, and we use use $\{\mathbf{m}_c^*(\mathcal{P})\}_{c=0}^{\infty}$ to represent the vectors with the c first coefficients of $\mathcal{M}^*(\mathcal{P})$. This way we have $\mathcal{M}^*(\mathcal{P}) \in \mathfrak{U}_c(\mathbf{m}_c^*(\mathcal{P}))$ for all c.

The committed rate defined by a partial solution $\mathbf{m}$, introduced in Section 5.2.1, is not useful only as a lower bound, as in (5.20). It allows us to decompose the optimization problem in separable subproblems, which can be solved recursively. Thanks to this property we can define optimal solutions using only values of $L(d)$. First, we analyze the minimum committed rate defined by all partial solutions that cover the set with the first i symbols.

Lemma 5.4.1 *Given probability distribution $\mathcal{P}$, if $\{r^*(i)\}_{i=0}^{\infty}$ is the sequence with the minimum of the committed rates*

$$r^*(i) = \min_{\mathbf{m}:L_c(c,\mathbf{m})=i} \{R_c(\mathbf{m},\mathcal{P})\}, \tag{5.24}$$

then its elements satisfy the set of recursive equations

$$\begin{aligned} r^*(i) &= \bar{F}(i) + \min_{m=1,2,\ldots,i} \{r^*(i-m) - \lceil \log_2(m) \rceil \bar{F}(i) + \\ &\quad \lfloor \log_2(m) \rfloor \bar{F}(i-m) + \bar{F}(i+\tau(m)-m) \left(\lceil \log_2(m) \rceil - \lfloor \log_2(m) \rfloor\right)\} \end{aligned} \tag{5.25}$$

Proof: Since for all vectors $\mathbf{m}$ in the minimization we have $L(c) = i$, the committed rate can be written as

$$\begin{aligned} R_c(\mathbf{m},\mathcal{P}) &= \sum_{d=0}^{c} \bar{F}(L(d)) + \sum_{d=0}^{c-1} \sum_{s=0}^{m_d-1} p(s+L(d))\gamma(s,m_d) \\ &= \bar{F}(i) + \sum_{s=0}^{m_{c-1}-1} p(s+i-m_{c-1}))\gamma(s,m_{c-1}) + R_{c-1}(\mathbf{m}',\mathcal{P}) \end{aligned} \tag{5.26}$$

where $\mathbf{m}'$ is the vector obtained by removing the last element of $\mathbf{m}$. From the definition of $r^*(i)$ we have

$$\min_{\mathbf{m}:L_c(c,\mathbf{m}')=i-m_{c-1}} R_{c-1}(\mathbf{m}',\mathcal{P}) = r^*(i-m_{c-1}), \tag{5.27}$$

and consequently

$$\begin{aligned} r^*(i) &= \min_{\mathbf{m}:L_c(c,\mathbf{m})=i} \{R_c(\mathbf{m},\mathcal{P})\} \\ &= \bar{F}(i) + \min_{\mathbf{m}:L_c(c,\mathbf{m})=i} \left\{ R_{c-1}(\mathbf{m}') + \sum_{s=0}^{m_{c-1}-1} p(s+i-m_{c-1})\gamma(s,m_{c-1}) \right\} \\ &= \bar{F}(i) + \min_{m=1,2,\ldots,i} \left\{ r^*(i-m) + \sum_{s=0}^{m-1} p(s+i-m)\gamma(s,m) \right\} \\ &= \bar{F}(i) + \min_{m=1,2,\ldots,i} \{r^*(i-m) + \lfloor \log_2(m) \rfloor \bar{F}(i-m) - \lceil \log_2(m) \rceil \bar{F}(i) + \\ &\quad \bar{F}(i+\tau(m)-m) \left(\lceil \log_2(m) \rceil - \lfloor \log_2(m) \rfloor\right)\} \end{aligned} \tag{5.28}$$

The solutions are indeed optimal because for all values of i we have $r^*(i)$ defined by enumerating all possible solutions, since all values of m_{c-1} are tested, and all the possible combinations of $m_0, m_1, \ldots, m_{c-2}$ are tested to generate $r^*(i-m_{c-1})$.

An alternative to formula (5.25) is derived as

$$\begin{aligned} r^*(i) &= \bar{F}(i) + \min_{j=0,1,\ldots,i-1} \left\{ r^*(j) + \sum_{s=j}^{i-1} p(s)\gamma(s-j,i-j) \right\} \\ &= \bar{F}(i) + \min_{j=0,1,\ldots,i-1} \{r^*(j) + \lfloor \log_2(i-j) \rfloor [\bar{F}(j) - \bar{F}(j+\tau(i-j))] + \\ &\quad \lceil \log_2(i-j) \rceil [\bar{F}(j+\tau(i-j)) - \bar{F}(i)]\} \end{aligned} \tag{5.29}$$

∎

This result can be used to define necessary optimality conditions for $\mathcal{M}^*$.

Proposition 5.4.2 *Given probability distribution $\mathcal{P}$, and an optimal set of unary-stem code parameters $\mathcal{M}^*(\mathcal{P})$, if $\{r^*(i)\}_{i=0}^{\infty}$ is a sequence that satisfies the set of recursive equations (5.25) then, the committed rates for all truncated versions of the optimal sequence of parameters are such that*

$$R_c(\mathbf{m}_c^*(\mathcal{P}), \mathcal{P}) = r^*(L_U(c, \mathcal{M}^*(\mathcal{P}))), \quad c = 0, 1, 2, \ldots \tag{5.30}$$

Proof: This result follows from the fact that for unary-stem codes the average rates depend only of values of $L_U(d, \mathcal{M})$ (or, the simplified form $L(d)$). First, we know that the condition

$$R_c(\mathbf{m}_c^*(\mathcal{P}), \mathcal{P}) < r^*(L_U(c, \mathcal{M}^*(\mathcal{P})))$$

is not possible because it contradicts the definition (5.24) of $r^*(i)$.

If, on the other hand, we had

$$R_c(\mathbf{m}_c^*(\mathcal{P}), \mathcal{P}) > r^*(L_U(c, \mathcal{M}^*(\mathcal{P}))),$$

then, from (5.24) we know that this would imply the existence of a vector $\mathbf{m}^\circ$ with dimension t, and a sequence

$$\mathcal{M}^\circ = \{m_0^\circ, m_1^\circ, \ldots, m_{t-2}^\circ, m_{t-1}^\circ, m_c^*, m_{c+1}^*, m_{c+2}^*, \ldots\}$$

such that

$$R_c(\mathbf{m}_c^*(\mathcal{P}), \mathcal{P}) > R_c(\mathbf{m}^\circ, \mathcal{P}) \quad \Longrightarrow \quad B_U(\mathcal{M}^*, \mathcal{P}) > B_U(\mathcal{M}^\circ, \mathcal{P}).$$

i.e., contradicting the assumption that $\mathcal{M}^*$ is optimal. ∎

We have to note that Proposition 5.4.1 provides only necessary optimality conditions, and not the values of $L_U(c, \mathcal{M}^*(\mathcal{P}))$ which would let us recover the optimal solution $\mathcal{M}^*$. However, we can still obtain solutions very near the optimal by sequentially testing each $r(i)^*$, and the partial solution that it represents. We can use an algorithm with the following steps

1. Compute the values of $r^*(i)$ using dynamic programming equations (5.25), and for each i store the value of m that yields the minimum value, in order to "backtrack" the dynamic programming solution, and recover the corresponding values of $m_0, m_1, \ldots$
2. Compute the rate of full solutions using the standard extension with (5.19), and save the best solution as $\mathcal{M}^b$.
3. Stop when $B_U(\mathcal{M}^b, \mathcal{P}) - r^*(i)$ is sufficiently small (i.e., the contribution of the possibly suboptimal standard extension to the code is insignificant). Output the best solution $\mathcal{M}^b$.

5.5 Forward Dynamic Programming Optimization

Like other forms of dynamic programming, in the optimization of unary-stem codes it is possible to reverse the order of the recursive equations. If we define the shifts of a sequence as

$$\mathcal{S}(i,\mathcal{P}) = \{p(i), p(i+1), p(i+2), \ldots\}, \quad i = 0,1,2,\ldots \tag{5.31}$$

then we can derive the following forward dynamic programming optimization.

Lemma 5.5.1 *Given probability distribution $\mathcal{P}$, if $\{\bar{r}^*(i)\}_{i=0}^{\infty}$ is the sequence with the minimum rates*

$$\bar{r}^*(i) = \min_{\mathcal{M}} \left\{ B_U(\mathcal{M}, \mathcal{S}(i,\mathcal{P})) \right\}, \tag{5.32}$$

then its elements satisfy the set of recursive equations

$$\begin{aligned} \bar{r}^*(i) \;=\; & \bar{F}(i) + \min_{m=1,2,3,\ldots} \{\bar{r}^*(i+m) + \lfloor \log_2(m) \rfloor [\bar{F}(i) - \bar{F}(i+\tau(m))] + \\ & \lceil \log_2(m) \rceil [\bar{F}(i+\tau(m)) - \bar{F}(i+m)]\}, \quad i = \ldots, 2, 1, 0. \end{aligned} \tag{5.33}$$

Proof: We can prove this result starting with

$$\begin{aligned} \bar{r}^*(0) &= \min_{\mathcal{M}} \{B_U(\mathcal{M},\mathcal{P})\} \\ &= \min_{\mathcal{M}} \left\{ \sum_{d=0}^{\infty} \bar{F}(L(d)) + \sum_{d=0}^{\infty} \sum_{s=0}^{m_d-1} p(s+L(d))\gamma(s,m_d) \right\} \\ &= \bar{F}(0) + \min_{\mathcal{M}} \left\{ \sum_{s=0}^{m_0-1} p(s)\gamma(s,m_0) + B_U(\mathcal{S}(1,\mathcal{M}), \mathcal{S}(m_0,\mathcal{P})) \right\} \\ &= \bar{F}(0) + \min_{m=1,2,3,\ldots} \left\{ \bar{r}^*(m) + \sum_{s=0}^{m-1} p(s)\gamma(s,m) \right\} \end{aligned} \tag{5.34}$$

The other values of $\bar{r}^*(i)$ follow from the fact that if $\mathcal{P}$ is replaced by $\mathcal{S}(i,\mathcal{P})$ then we just have to replace in (5.34) $\bar{r}^*(s)$, $p(s)$ and $\bar{F}(s)$ with $\bar{r}^*(s+i)$, $p(s+i)$ and $\bar{F}(s+i)$, and obtain

$$\begin{aligned} \bar{r}^*(i) &= \bar{F}(i) + \min_{m=1,2,3,\ldots} \left\{ \bar{r}^*(m+i) + \sum_{s=0}^{m-1} p(s+i)\gamma(s,m) \right\} \\ &= \bar{F}(i) + \min_{m=1,2,3,\ldots} \{\bar{r}^*(i+m) + \lfloor \log_2(m) \rfloor [\bar{F}(i) - \bar{F}(i+\tau(m))] + \\ &\qquad \lceil \log_2(m) \rceil [\bar{F}(i+\tau(m)) - \bar{F}(i+m)]\}. \end{aligned} \tag{5.35}$$

∎

This recursive form may not seem very useful since

1. It "starts" from infinite values of i, and requires knowing an infinite number of elements $\bar{r}^*(i+m)$;

2. An infinite number of values m need to be tested to find the minimum for each $\bar{r}^*(i)$.

However, these problems can be solved for obtaining solutions that are optimal within a certain precision. First, we can use the fact that we always have

$$\bar{r}^*(i) \geq \bar{h}(i).$$

In addition, all our numerical results show that the relative redundancy of unary-stem codes are usually small, e.g., below 20%. Thus, for sufficiently large values of i we can use an approximation like

$$\bar{r}^*(i) \approx 1.2\,\bar{h}(i).$$

The values of $\bar{h}(i)$ can be known in closed-form, or approximated with small computational cost using (4.87). When i is sufficiently large the difference should be small enough as not to affect the choice of a solution that is optimal within the desired precision.

The second problem is solved by observing that the sums in the dynamic programming equation are monotonic, i.e., if $m < n$ then

$$\sum_{s=0}^{m-1} p(s+i)\gamma(s,m) \leq \sum_{s=0}^{n-1} p(s+i)\gamma(s,n).$$

Thus, if we have m° and n such that

$$\bar{r}^*(i+m^\circ) + \sum_{s=0}^{m^\circ-1} p(s+i)\gamma(s,m^\circ) < \sum_{s=0}^{n-1} p(s+i)\gamma(s,n) \tag{5.36}$$

then we can disregard all values of m larger than n in the minimization.

In order to obtain the optimal code we need the following result.

Proposition 5.5.2 *Given probability distribution $\mathcal{P}$, and an optimal set of unary-stem code parameters $\mathcal{M}^*(\mathcal{P})$, if $\{\bar{r}^*(i)\}_{i=0}^{\infty}$ is a sequence that satisfies the set of recursive equations (5.33) then*

$$B_U(\mathcal{S}(c,\mathcal{M}^*(\mathcal{P})), \mathcal{S}(c,\mathcal{P})) = r^*(c), \quad \forall\, c = L_U(0,\mathcal{M}^*(\mathcal{P})), L_U(1,\mathcal{M}^*(\mathcal{P})), \ldots \tag{5.37}$$

Proof: This proof is nearly the same as the proof of Proposition 5.4.2. The difference is that the contradiction to the assumption of optimality would occur with a sequence in which the last (instead of the first) elements of $\mathcal{M}^*(\mathcal{P})$ are replaced. ∎

One interesting property of this dynamic programming form is that $\bar{r}^*(0) = B_U(\mathcal{M}^*(\mathcal{P}), \mathcal{P})$, i.e., the rate of the optimal unary-stem code. Thus, it is easier to recover the optimal solution, recovering the optimal dynamic programming decisions beginning from $\bar{r}^*(0)$. In this case there is no need to artificially extend the code, only to guess approximate values of $\bar{r}^*(i)$ for sufficiently large values of i.

For distributions with heavy tails, another approach is to assume an approximation of the reverse distribution $\bar{F}(s)$ that is reasonably precise for large s. While for backward dynamic programming we use an estimate of the current value of m_d to define the standard extension, in this case we do not have such estimate, but with the model we can compute very good approximated upper bounds on $\bar{r}^*(i)$.

For example, using model (4.86), given the values of α, β, and t, estimated from $p(i)$, $\bar{F}(i)$, and $\bar{F}(i)$, we can define

$$\bar{r}^*(i) \approx \min_{k=0,1,2,\ldots} \left\{ t\,\alpha^{-\beta} \left[1 + k + 2 \sum_{d=1}^{\infty} [1 + \alpha^{-\beta} 2^k (2^d - 1)]^{-\beta} \right] \right\}. \tag{5.38}$$

5.6 Optimization of Dyadic and Other Special Codes

The optimization methods presented in this chapter are quite flexible in terms of adding constraints on the parameters m_d of the unary code. For dyadic unary-stem codes the only modification in the implicit enumeration algorithm is the enumeration of non-negative integer values of k_d. The modifications in the dynamic programming methods are also straightforward, in order to enumerate only the allowed values of m_d in the minimizations in the recursive equations. For the backward dynamic programming we need to replace (5.25) with

$$r^*(i) = \bar{F}(i) + \min_{k=0,1,\ldots,\lfloor \log_2(i) \rfloor} \left\{ r^*(i - 2^k) + k[\bar{F}(i - 2^k) - \bar{F}(i)] \right\}, \tag{5.39}$$

and the forward dynamic programming should use

$$\bar{r}^*(i) = \bar{F}(i) + \min_{k=0,1,2,\ldots} \left\{ \bar{r}^*(i + 2^k) + k[\bar{F}(i) - \bar{F}(i + 2^k)] \right\}, \tag{5.40}$$

instead of (5.33).

It is similarly very easy to adapt the algorithms to other special types of codes that had been proposed. For example, another type of codeword formation rule is to allow only [37]

$$m_d = 2^k, \quad \text{or} \quad m_d = 3 \cdot 2^k. \tag{5.41}$$

The codes created with these constraints are between the general and dyadic unary-stem codes, and the implementation is somewhat more complicated than the dyadic codes, but certainly less complex than the general form.

Chapter 6

Numerical Results

Our experiments were designed to analyze how the structure of a optimal unary-stem code changes depending on source uncertainty. In our case, the amount of uncertainty is basically defined, in varying amounts, by the prior distribution $f(\rho)$ and the number of known source samples N_s. The sufficient statistic used for estimating the best code is the sum of sample values Σ_s. Thus, we present the optimal codes found for many combinations of prior distributions, and values of N_s and Σ_s.

Since there is a combination of many different settings, and the optimal codes are defined by a sequence of optimal parameters, we have a very large number of results, in a fairly large number of tables. In order to avoid making the document hard to browse we moved most of the tables with results to Appendix C, and in this chapter present only the main results and conclusions.

6.1 Optimization Algorithms

The code optimization methods presented in Chapter 5 were implemented and tested in a variety of code design problems. The main conclusions about their effectiveness are presented below.

1. **Implicit Enumeration** – while this algorithm is the most flexible, capable of including any type of additional constraints on codeword formation, the experiments have shown that it is not computationally efficient. Its speed depends on the ability to quickly detect non-optimal solutions using tight lower bounds. While in our code design problem the bounds can be considered good, the problem was the enormous number of nearly-optimal solutions. This method can be used to obtain dyadic unary-stem codes [38], when a relatively small number of m_d values need to enumerated, but it is not applicable to heavy tailed distributions. Furthermore, except when the probabilities decay very fast, it is too slow for the general case, when m_d can take any

positive value.

2. **Backward Dynamic Programming** – the complexity of this method is nearly fixed, and proportional to the square of the number of symbols taken into consideration. It is quite intuitive, in the sense that even when we have an infinite number of symbols, at each iteration a finite number of cases is tested, and we just need to stop when the probability values get too small. However, as explained during its derivation, the problem is that the dynamic programming technique does not provide the optimal solution directly, but only the information to recover it by testing solutions using the "standard" code extension. Thus, while this method worked very well for distributions with fast decay, it is not easy to identify the nearly optimal termination points and conditions when applied to heavy-tailed distributions. The consequence is that the algorithm can yield codes that are more biased to early termination, i.e., more optimized to a distribution that is truncated to a finite alphabet, instead of the solution for the infinite alphabet.

3. **Forward Dynamic Programming** – this method is significantly less intuitive than the backward dynamic programming, since it needs some shortcuts and approximations to deal with infinite number of tests. However, it was the method that in practice proved to be the most efficient, precise and reliable. The fact that it has a fixed point, $\bar{r}^*(0)$, from where the optimal solution can be retrieved, proved to be very important. Small changes in the approximations used for values of $\bar{r}^*(i)$ with large i did not affect the final solutions significantly. The early-termination criterion (5.36) was found to be quite efficient, and the average complexity was roughly proportional to the square of the number of $\bar{r}^*(i)$ computed, but with a factor smaller than the backward dynamic programming.

Considering these observations about the optimization methods, it is clear that the forward dynamic programming is the best choice, and there is little need to present details of the comparisons, like time measurements, since they do not provide much insight into our code analysis.

All the codes and results presented in this document were obtained using the forward dynamic programming algorithm of Section 5.5. The values $\bar{r}^*(i)$ were computed for all i such that $\bar{F}(i) > 10^{-9}$, or $i < 200,000$. In addition, the tail of the distributions were extrapolated by using approximation (4.86), choosing the best fit of parameters α, β, and t, and using approximation (5.38).

We optimized three types of codes, those with $m_d \geq 1$, $m_d = 2$ (cf. 5.6

6.2 Prior Distributions and Computations

In our tests we considered three specific prior distributions $f(\rho)$.

1. **Uniform prior** – is a commonly used prior, and as shown in Section 4.3, defines distributions that are in a sense are "matched" to Elias-Teuhola codes. This prior has the advantage simplifying the computations of all numerical values that are needed in the experiments, since they can use efficient methods for computing $\ln(\Gamma(x))$.

2. **Dirichlet (1/2) prior** – this distribution can be considered very conservative, since it assigns higher probabilities to the values of ρ in the extremes of the interval $(0, 1)$. It has some interesting mathematical properties, and may produce posterior probability distributions with very heavy tails, which were useful in evaluating the code optimization algorithms. furthermore, it uses the same tools as the uniform prior for computing numerical values.

3. **Empiric prior** $g_e(\xi)$ – is included in order to evaluate a prior that is derived from practical source coding applications. In this case we use the histogram of Figure 4.2 as a reference, and approximated it with the curve of Figure 4.3. We used Romberg's numerical integration, and the transformation of Appendix A for better numerical accuracy and stability. While the extreme range of values can complicate considerably the computations, for this prior we found that very good accuracy was achievable with reasonable computational effort (we would have much more serious problems if numerical integrations were needed for the Dirichlet and uniform priors).

In Appendix C, Tables C.5 to C.7 show the first values in $\mathcal{M}^*(\mathcal{P})$, i.e., the optimal unary-stem code parameters, for these three priors, and a combination of different values of N_s and Σ_s. Tables C.8 to C.10 show the first values in $\mathcal{K}^*(\mathcal{P})$.

We observe that, with the uniform and the Dirichlet (1/2) priors, similarly to the codes defined by the maximum-likelihood estimation of $\hat{\rho}_d$ (Tables C.1 and C.3)—which we show in Proposition 4.1.2 to have asymptotic exponential growth with rate inversely proportional to N_s—the parameters m_d of the optimal codes also show growth that is nearly exponential. In the codes optimized for the empiric prior, on the other hand, there is an initial fast growth, followed by a decrease in the growth speed, since this prior assigns zero probability to sources that have very large entropy.

In Table C.5, corresponding to the Dirichlet (1/2) prior, we do not provide values for m_{11} and m_{12} when $N_s = \Sigma_s = 0$, because we could not obtain good estimates of their true optimal value. The problem arises from the fact that in this particular case we have $\bar{F}(s) \approx c/\sqrt{s}$ for large s. This extremely slow decay made this the most difficult code to optimize, and in fact was the only case that needed extra attention. Estimates of the optimal code parameters required increasing the number of terms in the dynamic programming optimization to 2 million, which needed several hours of computation. Even with that effort, it was not possible to get reliable values for all parameters. On the other hand, it is important to observe that the code redundancy are very well approximated, i.e., we could not get the exact first values of $\mathcal{M}^*(\mathcal{P})$, but it is not difficult to find codes that have nearly the same redundancy.

6.3 Code Comparison

It interesting to compare not only the relative performance of the different optimal unary-stem codes, but also evaluate how other similar codes perform in the same conditions.

Table 6.1 has a list of the types of codes we evaluated, and the letters used to identify them in the comparison tables. Below we provide a few more details about the codes.

Gol – Golomb code with parameter m defined by (2.23) and (4.3);

Go3 – Golomb code with optimal parameter $m = 2^k$ or $m = 3 \cdot 2^k$ for (4.3);

G-R – Golomb-Rice code with parameter k defined by (2.27) and (4.3);

E-T – Elias-Teuhola code with same k as the Golomb-Rice code;

MxL – Unary-stem code defined by formula (4.9), from maximum-likelihood estimation of ρ;

ML3 – Unary-stem code using $m_d = 2^k$ or $m_d = 3 \cdot 2^k$, from maximum-likelihood estimation of ρ

ML2 – Dyadic unary-stem code defined by (4.20), from maximum-likelihood estimation of ρ;

BEs – Unary-stem code with parameters m_d updated according to Bayesian estimation of ρ (4.25);

BE3 – Unary-stem code with parameters $m_d = 2^k$ or $m_d = 3 \cdot 2^k$ updated according to Bayesian estimation of ρ (4.25);

BE2 – Dyadic unary-stem code with parameters k_d updated according to Bayesian estimation of ρ (4.25);

Opt – Unary-stem code, optimized for posterior probabilities (4.27);

Op3 – Unary-stem code, with $m_d = 2^k$ or $m_d = 3 \cdot 2^k$, optimized for posterior probabilities (4.27);

Op2 – Dyadic unary-stem code optimized for posterior probabilities (4.27);

Huf – Optimal Huffman code for truncated posterior probabilities (4.27).

In tables where several types of code, the unary-stem codes are presented using only the parameters m_d. When the table presents only dyadic unary-stem codes we list values of k_d. In order to provide some characterization of the Huffman code we list some values of E_l, i.e. the number of codewords with length l bits (cf. Section 2.6), starting from the first nonzero element.

The criterion used for evaluating a code's compression performance is the relative redundancy, in percent,

$$\Delta = 100 \, \frac{B_U(\mathcal{M}, \mathcal{P}) - H(\mathcal{P})}{H(\mathcal{P})} \, \%. \tag{6.1}$$

Tables 6.2 to 6.4 show some parameters of the codes and their relative redundancy, when $N_s = \Sigma_s = 0$, i.e., when there is no knowledge of the source, besides the prior distribution. In these tables the Elias-Teuhola code has parameter $k = 0$, and we do not show the Golomb and Golomb-Rice codes because, as shown in Proposition 4.3.1, in those cases they have infinite redundancy. Similarly, the maximum-likelihood estimates are undefined because $N_s = 0$.

Tables 6.5 to 6.7 compare the relative redundancy of the different codes, using different priors and values of N_s and Σ_s. Note that these values are chosen in order to repeat the ratios Σ_s/N_s, which correspond to the estimated average symbol values ($\bar{s}$). The parameters that define these codes, plus the parameters and redundancy of codes constructed using maximum-likelihood estimation, are shown in Appendix C, Tables C.11 to C.46.

We can observe that the redundancy of the Golomb and Golomb-Rice codes depend heavily on N_s, and somewhat less on the type of prior. As expected, these codes do not perform well when there is significant uncertainty about the source's parameter, and the redundancy can be relatively quite large. One interesting observation is that their performance tends to be better on sources with larger entropy (or larger Σ_s/N_s in our tests).

It is important to keep in mind that in our experiments the parameter N_s only provides a simplified way to model the uncertainty about the source parameter. In practical applications what happens is that the source parameter may change too fast, and increasing the number of known samples does not necessarily lead to better compression. Thus, the correct interpretation is that by increasing N_s we are changing some implicit assumption about how fast the source statistics change.

The redundancy values also confirm that the Elias-Teuhola codes are quite versatile, and their redundancy is typically below 5%. However, we can also observe that the relative redundancy can actually increase with N_s. This shows that these codes are indeed quite attractive when there is nearly no *a priori* knowledge of the source, but comparing their performance with the other unary-stem codes, we can see that even broad assumptions about the source's uncertainty enable us to design codes that similarly resilient, but more efficient.

The redundancy of the optimal unary-stem codes is always very close to the redundancy obtained with the Huffman code for the truncated alphabet. This means that for all those distributions, the unary-stem codes are nearly optimal prefix codes. The redundancy compared to the entropy was in all cases below 3%, and is typically below 1%. The codes created with the heuristic of estimating ρ, using a maximum-likelihood or Bayesian approach, and then using the value of m_d or k_d that corresponds to the optimal for that ρ on a stationary source, perform surprising well. We can see that their redundancy is never significantly worse than that of the optimal codes. This shows that even the use of rough, but more realistic models of the source, and even approximations and heuristic code design, are better than codes that are optimal under assumptions that do not correspond to reality.

The results with the empiric prior are particularly interesting. While we can assume that the uniform and Dirichlet priors somehow represent some type of worst-case scenario, with more uncertainty about ρ, in reality the assignment of high probabilities to source with low entropy leads to distributions where the first symbols have large probability, and codes that efficiently represent these symbols may have low overall redundancy, even if the code is badly matched to the remaining symbols. The prior derived from experiments, on the other hand, correctly represent the fact that in practical coding situations we do not have such large symbol probabilities due to noise. Consequently, the results obtained with this prior indeed are a better indication of what to expect from these codes in practical applications.

Table 6.1: Types of codes used.

Code Name	Type
Gol	Golomb code
Go3	Golomb code with parameters $m = 2^k$ or $m = 3 \cdot 2^k$
G-R	Golomb-Rice code
E-T	Elias-Teuhola code
MxL	Unary-stem code from maximum-likelihood estimation (MLE) of ρ
ML3	Unary-stem code from MLE, $m_d = 2^k$ or $m_d = 3 \cdot 2^k$
ML2	Dyadic unary-stem code from MLE
BEs	Unary-stem code from Bayesian estimation (BE) of ρ
BE3	Unary-stem code from BE, $m_d = 2^k$ or $m_d = 3 \cdot 2^k$
BE2	Dyadic unary-stem code from BE
Opt	Optimal unary-stem code for posterior probabilities
Op3	Optimal unary-stem code, $m_d = 2^k$ or $m_d = 3 \cdot 2^k$
Op2	Optimal dyadic unary-stem code
Huf	Optimal Huffman code for truncated posterior probabilities

Table 6.2: Comparisons of unary-stem codes, assuming Dirichlet (1/2) prior, and no known source samples.

$N_s = 0,\ \Sigma_s = 0,\ H = 3.85$ bits/symbol												
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
E-T	4.40	1	2	4	8	16	32	64	128	256	512	1024
BEs	2.58	1	2	5	12	29	69	165	393	938	2239	5342
BE3	2.52	1	2	6	12	32	64	192	384	1024	2048	6144
BE2	4.40	1	2	4	8	16	32	64	128	256	512	1024
Opt	0.79	1	3	12	47	188	753	3012	12039	48006	188453	648860
Op3	0.79	1	3	12	48	192	768	3072	12288	49152	196608	786432
Op2	2.82	1	2	8	32	64	256	1024	2048	8192	32768	131072
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}
Huf	0.28	1	0	1	1	2	3	6	8	13	20	34

Table 6.3: Comparisons of unary-stem codes, assuming uniform ($\Pi_{0,1}(\rho)$) prior, and no known source samples.

$N_s = 0,\ \Sigma_s = 0,\ H = 2.95$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
E-T	1.62	1	2	4	8	16	32	64	128	256	512	1024	2048
BEs	1.26	1	2	3	5	9	15	25	43	72	122	207	350
BE3	1.27	1	2	3	6	8	16	24	48	64	128	192	384
BE2	1.75	1	2	4	8	8	16	32	64	128	128	256	512
Opt	1.01	1	2	3	6	12	23	46	92	183	364	727	1452
Op3	1.01	1	2	3	6	12	24	48	96	192	384	768	1536
Op2	1.62	1	2	4	8	16	32	64	128	256	512	1024	2048
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}
Huf	1.00	1	0	2	1	1	3	4	6	8	12	16	24

Table 6.4: Comparisons of unary-stem codes, assuming the empiric prior $g_e(\xi)$, and no known source samples.

$N_s = 0,\ \Sigma_s = 0,\ H = 6.72$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
E-T	19.87	1	2	4	8	16	32	64	128	256	512	1024	2048
BEs	2.72	5	14	35	68	108	148	186	223	258	290	321	349
BE3	2.69	6	16	32	64	96	128	192	192	256	256	256	384
BE2	3.87	4	16	32	64	128	128	256	256	256	256	256	256
Opt	1.22	12	47	95	157	188	220	281	309	336	362	386	408
Op3	1.25	12	48	96	128	192	192	256	256	384	384	384	384
Op2	2.75	8	32	64	128	128	256	256	256	256	256	256	512
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.40	1	3	4	7	13	23	41	63	92	122	155	186

Table 6.5: Relative redundancy (%) of different codes, assuming Dirichlet (1/2) prior.

N_s	Σ_s	Code										
		Gol	Go3	G-R	E-T	BEs	BE3	BE2	Opt	Op3	Op2	Huf
0	0	—	—	—	4.37*	2.56	2.50	4.37	0.78	0.79	2.81	0.24
1	1	51.01	51.01	51.01	2.75	1.64	1.78	1.87	1.29	1.36	1.82	1.29
1	2	31.01	31.01	31.01	4.42	2.64	2.86	3.01	2.07	2.19	2.93	2.07
1	4	29.69	29.69	21.76	4.94	1.65	2.06	2.35	0.61	0.82	2.29	0.59
1	7	24.67	19.22	34.29	2.31	1.23	1.30	2.28	0.71	1.08	1.86	0.67
1	15	18.07	16.10	28.80	1.93	1.12	1.12	1.95	0.59	0.93	1.56	0.56
1	30	15.87	13.27	23.68	1.67	0.92	0.95	1.64	0.53	0.80	1.35	0.46
1	75	13.04	14.54	10.67	2.58	0.77	1.13	1.27	0.37	0.50	1.24	0.33
1	150	11.30	12.61	9.28	2.31	0.68	0.99	1.13	0.33	0.45	1.10	0.30
2	2	11.56	11.56	11.56	5.21	1.13	1.14	1.30	1.12	1.13	1.30	1.12
2	4	7.15	7.15	7.15	6.05	1.87	1.92	2.34	1.86	1.88	2.34	1.87
2	8	7.20	7.20	5.54	5.92	0.77	0.93	2.32	0.75	0.81	2.32	0.75
2	14	5.99	4.26	10.02	3.24	1.08	1.05	1.79	0.56	0.89	1.76	0.56
2	30	4.23	3.64	8.69	2.67	0.68	0.93	1.57	0.48	0.72	1.46	0.48
2	60	3.82	3.02	7.10	2.28	0.63	0.77	1.29	0.41	0.64	1.25	0.41
2	150	3.24	3.81	2.79	2.87	0.48	0.54	1.25	0.36	0.45	1.24	0.35
2	300	2.86	3.35	2.48	2.57	0.43	0.48	1.11	0.32	0.40	1.10	0.31
5	5	1.96	1.96	1.96	8.78	0.84	0.84	0.86	0.84	0.84	0.86	0.84
5	10	2.04	2.04	2.04	8.49	1.60	1.60	1.73	1.58	1.58	1.73	1.58
5	20	1.88	1.88	2.31	7.61	0.80	0.83	2.18	0.79	0.83	2.18	0.79
5	35	1.53	1.13	3.55	4.64	0.63	1.00	1.72	0.57	0.81	1.72	0.57
5	75	1.04	0.94	3.25	3.79	0.50	0.91	1.50	0.48	0.63	1.50	0.48
5	150	0.95	0.80	2.59	3.22	0.45	0.73	1.24	0.40	0.56	1.24	0.40
5	375	0.86	1.10	1.03	3.54	0.41	0.49	0.95	0.39	0.48	0.95	0.39
5	750	0.76	0.96	0.93	3.15	0.36	0.43	0.86	0.35	0.42	0.86	0.35

Table 6.6: Relative redundancy (%) of different codes, assuming uniform prior.

N_s	Σ_s	Code										
		Gol	Go3	G-R	E-T	BEs	BE3	BE2	Opt	Op3	Op2	Huf
0	0	—	—	—	1.62*	1.26	1.27	1.75	1.01	1.01	1.62	1.00
1	1	20.69	20.69	20.69	4.08	1.27	1.27	1.42	1.12	1.14	1.41	1.12
1	2	11.66	11.66	11.66	6.82	2.13	2.12	2.37	1.88	1.91	2.36	1.88
1	4	9.06	9.06	9.83	8.60	1.09	1.06	2.07	0.96	1.06	2.06	0.96
1	7	7.35	6.55	10.45	3.86	0.96	0.95	2.00	0.69	0.93	1.95	0.69
1	15	5.46	5.39	8.26	3.32	0.79	0.77	1.86	0.58	0.77	1.72	0.58
1	30	4.69	4.62	6.64	3.01	0.72	0.61	1.85	0.47	0.61	1.59	0.47
1	75	3.94	3.97	5.38	5.27	0.62	0.85	1.06	0.47	0.69	1.05	0.43
1	150	3.47	3.48	4.78	4.72	0.56	0.72	0.94	0.41	0.62	0.93	0.38
2	2	7.46	7.46	7.46	6.12	1.20	1.21	1.30	1.16	1.16	1.29	1.16
2	4	4.89	4.89	4.89	7.64	2.46	2.48	2.74	2.36	2.36	2.71	2.36
2	8	3.71	3.71	4.95	8.09	1.21	1.29	2.26	0.82	0.85	2.14	0.82
2	14	3.01	2.78	5.06	4.03	0.84	1.13	1.55	0.82	0.92	1.48	0.82
2	30	2.23	2.29	4.16	3.30	0.81	0.95	1.29	0.61	0.78	1.24	0.60
2	60	1.92	1.99	3.30	2.87	0.62	0.81	1.07	0.54	0.70	1.06	0.50
2	150	1.69	1.73	2.65	4.24	0.50	0.63	1.06	0.35	0.54	1.06	0.35
2	300	1.49	1.52	2.37	3.78	0.43	0.55	0.92	0.31	0.48	0.92	0.31
5	5	1.63	1.63	1.63	9.10	0.80	0.80	0.80	0.80	0.80	0.80	0.80
5	10	2.11	2.11	2.11	9.18	1.83	1.83	1.94	1.83	1.83	1.94	1.83
5	20	1.40	1.40	2.82	8.52	0.79	0.80	2.40	0.78	0.80	2.40	0.78
5	35	1.11	1.21	2.50	4.91	0.59	0.89	1.49	0.59	0.89	1.49	0.59
5	75	0.83	0.95	2.22	3.98	0.54	0.76	1.29	0.47	0.76	1.29	0.47
5	150	0.72	0.85	1.73	3.39	0.44	0.74	1.04	0.41	0.63	1.04	0.41
5	375	0.68	0.74	1.34	4.06	0.39	0.44	1.20	0.37	0.44	1.20	0.37
5	750	0.60	0.65	1.21	3.61	0.35	0.39	1.06	0.33	0.39	1.06	0.33

Table 6.7: Relative redundancy (%) of different codes, assuming $g_e(\xi)$ prior.

N_s	Σ_s	Code										
		Gol	Go3	G-R	E-T	BEs	BE3	BE2	Opt	Op3	Op2	Huf
0	0	—	—	—	19.87*	2.72	2.69	3.87	1.22	1.25	2.75	0.40
1	1	161.58	161.58	161.58	6.28	1.48	1.83	2.33	0.80	0.98	2.25	0.72
1	2	78.21	78.21	78.21	2.65	1.07	1.13	2.67	0.56	0.61	2.55	0.54
1	4	63.10	63.10	41.70	2.07	1.40	1.69	2.11	0.84	1.17	1.94	0.69
1	7	43.59	33.01	60.44	2.58	1.48	1.26	2.54	0.53	0.57	2.32	0.50
1	15	22.83	19.68	38.29	2.32	1.11	0.97	2.09	0.47	0.52	1.95	0.43
1	30	13.03	10.03	21.52	2.03	0.84	1.39	1.79	0.40	0.47	1.59	0.37
1	75	4.30	5.22	3.01	2.32	0.61	0.53	1.53	0.35	0.46	1.17	0.35
1	150	1.36	1.61	1.67	3.10	0.47	0.79	1.00	0.30	0.43	0.98	0.30
2	2	36.99	36.99	36.99	5.76	1.34	1.40	1.85	1.31	1.34	1.84	1.31
2	4	15.45	15.45	15.45	3.75	1.05	1.17	2.01	0.99	1.05	1.99	0.99
2	8	16.14	16.14	9.44	3.67	1.10	1.09	1.79	0.82	1.04	1.74	0.82
2	14	13.11	9.02	20.41	2.90	0.89	0.73	1.98	0.58	0.68	1.95	0.55
2	30	8.52	7.05	16.58	2.56	0.66	0.62	1.75	0.48	0.59	1.73	0.46
2	60	6.25	4.54	11.57	2.29	0.57	0.62	1.43	0.42	0.51	1.43	0.39
2	150	2.84	3.58	1.86	2.50	0.50	0.52	0.93	0.38	0.52	0.93	0.38
2	300	1.08	1.36	1.14	2.94	0.39	0.45	1.10	0.33	0.41	0.99	0.33
5	5	6.83	6.83	6.83	7.85	2.91	2.91	2.94	2.90	2.90	2.94	2.90
5	10	1.95	1.95	1.95	7.15	1.05	1.05	1.33	1.03	1.03	1.31	1.03
5	20	2.76	2.76	1.95	6.65	0.90	0.95	1.66	0.89	0.95	1.66	0.89
5	35	2.33	1.34	5.12	4.44	0.65	0.71	2.02	0.63	0.71	1.90	0.63
5	75	1.58	1.21	4.80	3.68	0.52	0.58	1.59	0.50	0.58	1.58	0.50
5	150	1.50	0.99	3.89	3.13	0.46	0.51	1.35	0.42	0.50	1.34	0.42
5	375	1.14	1.55	0.85	3.17	0.42	0.54	0.74	0.40	0.53	0.74	0.40
5	750	0.68	0.91	0.76	3.14	0.37	0.46	0.73	0.36	0.45	0.73	0.35

Chapter 7

Conclusions

This work is motivated by the practical advantages of using simple and efficient parameterized prefix codes for adaptive entropy coding of some complex data sources. When we consider the very popular Golomb and Golomb-Rice codes, which had been successfully employed in a variety of applications, we know that they are provably optimal for sources in which the symbol probabilities have geometric distribution. On the other hand, practitioners had long known that these code are not very versatile, and can perform poorly if the geometric distribution assumption is not satisfied with a good degree of accuracy. This led to several proposals of modified codes. For example, the similarly popular Elias-Teuhola ("exp-Golomb") codes modify the structure of Golomb-Rice codes—but keeping the most attractive features—in order to obtain greater immunity to changes in the source distribution.

We consider the fact that Golomb codes can be inefficient not only when the symbol probabilities deviate from a geometric distribution, but also due to uncertainty in the parameter of the source. Thus, we consider a new way of looking into the problem and investigate how uncertainty degrades the performance. In order to answer that question we analyze the situation in which the data source has a distribution that is indeed geometrical, but at the same time the parameters of such source may vary frequently due to factors that are hard to isolate and characterize (e.g., local textures in images, percussion events in audio, etc.). We believe that an analysis of such scenario can provide insight into the problem of combining source estimation with the decision of which code should be used for a given source sample.

Using a simple statistical analysis of Golomb codes in such scenario, we identify the type of change in the code structure that keeps the useful practical features and enables better adaptation, and define the family of *unary-stem* prefix codes, which contains many of the code modifications that had been proposed for that purpose. Similar to Golomb codes, unary-stem codes are formed with a combination of a unary and a binary code, but different binary codes can be used after each bit of the unary code. Since these codes are defined for countable infinite number of symbols, in our analysis each unary-stem code is defined by an infinite sequence of parameters. However, for practical applications only a few parameters have to specified.

We analyze the average bit rates used by unary-stem codes for any data source with a countable infinite number of symbols, and show how they can be computed efficiently using cumulative distribution formulas. This feature is later shown to be essential for designing code optimization algorithms, and in their efficient computational implementation. We also show how to identify the contributions of the unary and the binary codes to the redundancy.

Next, we present the statistical framework used for studying these codes. In our simplified model the source is assumed to be stationary, but its parameter is a random variable, and only a few source samples are available for its estimation. This is easier than to assume time-varying sources, which need much more complicated and less intuitive models. Using these assumptions, we present the equations for maximum-likelihood and Bayesian estimation of the source. The Bayesian approach also provides a full characterization of the source, including posterior probabilities, which we used for measuring the expected redundancy of different codes. In the process we propose an heuristic method to design unary-stem codes, changing parameters of the binary codes according to updated estimates of the geometric source parameter.

We study one particular case defined by our model, which corresponds to a uniform prior with no known samples. We show that the resulting posterior distribution happens to be a very good match to the Elias-Teuhola codes (which were proposed without assuming a source distribution). Since these distributions have a very "heavy tail," complicating the numerical computation of entropy and average bit rates, we propose methods and approximation that are much more efficient than direct computation of the series.

Based on the analysis of the average bit rates of unary-stem codes we propose two types of algorithms for finding the optimal code for a given symbol probability distribution. The first uses the well-known and quite general implicit enumeration ("branch-and-bound") approach. Our second approach is to employ dynamic programming to find the solutions of a related optimization problem, and then use those solutions to obtain the optimal code. We discuss two variations, corresponding to backward and forward dynamic programming. Even though the problems have infinite dimension, we show that it is possible to use approximations in order to find solutions that are guaranteed to be optimal within a pre-defined precision.

The numerical tests show that the forward dynamic programming was the best choice for finding the optimal codes, in terms of speed and reliability. The application of that optimization method for many different situations is shown in more than a hundred tables with the parameters that define the optimal codes. The optimization program proved to be very efficient, and in nearly all cases was able to find the optimal codes in very short times. It need larger computational efforts only for probabilities that decay so slowly that averages are undefined, and even the entropy series needs to be computed with tens or hundreds of thousands of terms.

While the theoretical results show that, in some cases when there is no information about a geometric distributed source besides a prior distribution, the redundancy of Golomb codes is infinite, these can be assumed to be extreme cases, not commonly found in practice.

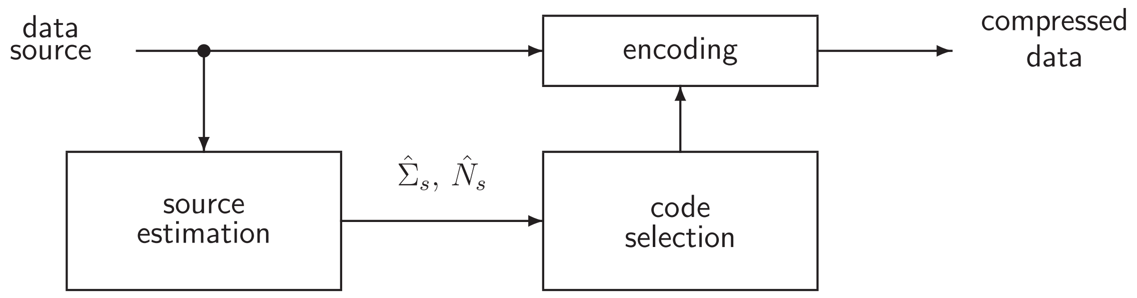


Figure 7.1: Example of a system for data compression using parameterized codes, with source estimation and code selection.

However, the numerical results show that the redundancy of Golomb codes can be significant, even in the sources that are geometric, and if there is some level of uncertainty in the source's parameter.

In some other situations, when there is less uncertainty about the source, and the entropy of the source is sufficiently large, we can observe that redundancy of the Golomb codes is quite small, and the difference between Golomb and unary-stem codes occur only for parameters m_d that are not expected to change the coding efficiency by much. Thus, it is arguable that we need to use unary-stem codes for those situations. However, it is important to note that, if the difference occurs only for symbols that have low probability, then the implementation complexity and performance of the codes are also very similar. On the other hand, with unary-stem codes there is little risk of a gross estimation error leading to single symbols being coded with an unreasonably large number of bits, as can happen with Golomb codes.

The numerical results also show that the class of unary-stem codes can be surprisingly effective in a very large set of data sources. Our test included a wide range of sources with decreasing probabilities–ranging from geometric to those with very slow decay—the redundancy of optimal unary-stem codes was always very close to the minimum achievable by prefix codes, and within a few percent of the entropy. This indicates that these codes may be also useful for coding quantized data from other types of distribution, like the generalized Gaussian.

The algorithm for optimizing unary-stem codes is quite efficient, considering that it is finding approximations to optimal solutions with a countable infinite number of variables, but we do not assume it may be used during coding. Another surprising conclusion from the numerical results is that the heuristic methods of updating the unary-stem code parameters m_d according to the maximum-likelihood or Bayesian estimates $\hat{\rho}_d$, which need much less computations, produced codes that are reasonably good. This indicates that maybe better of nearly-optimal unary-stem codes can be designed with heuristic methods that are somewhat smarter, but also with very low computational complexity.

Returning to the idea that motivated this work, we discussed the advantages of replacing the coding system in Figure 1.3 with the system of Figure 1.2. Now, we can see the advantages of using a system as shown in Figure 7.1, where the source estimation box is used not for estimating source parameters directly, but for determining the values that can enable the selection of the best codes. For instance, in the figure we show an example where two quantities $\hat{\Sigma}_s$ and $\hat{N}_s$ are used for code selection. The first, $\hat{\Sigma}_s$, can be similar to the Σ_s we used, containing the sufficient statistics, and quantized to a few number of values. The second, $\hat{N}_s$, can be used in a manner similar to N_s, providing and estimate of the reliability of the estimate, and thus the type of distribution. This can seem similar to just using some other statistics, like variance, for code selection. However, we have seen that in many of our posterior distributions the variance did not even have a defined value, and thus such estimates would not not be as useful for code selection.

Appendix A

Integral Computations

For Bayesian source estimation (Section 4.2) we need to compute integrals in the form

$$\Phi(m,n) = \int_0^1 f(\rho)\rho^{m-1}(1-\rho)^{n-1}\,d\rho, \tag{A.1}$$

where $f(\rho)$ is the probability density function of ρ.

However, the parameter ρ is not very convenient when applying numerical integration methods, because we may have abrupt changes in the interval $[0,1]$. Note in Figure 4.2 how empiric distributions from real sources tend to be more smoothly distributed when defined in terms of the logarithm of $\rho/(1-\rho)$. Furthermore, we can also see in Figure 2.2 how this value is nearly a linear function of the entropy of the geometric distribution, which has a much more intuitive meaning than ρ.

Thus, one alternative is to define the new variable

$$\xi = \ln\left(\frac{\rho}{1-\rho}\right) \tag{A.2}$$

which results in the integral

$$\begin{aligned}\Phi(m,n) &= \int_{-\infty}^{\infty} f\left(\frac{1}{1+e^{-\xi}}\right)\frac{1}{(1+e^{-\xi})^m\,(1+e^{-\xi})^n}\,d\xi \\ &= \int_{-\infty}^{\infty} \frac{g(\xi)}{(1+e^{-\xi})^{m-1}\,(1+e^{\xi})^{n-1}}\,d\xi\end{aligned} \tag{A.3}$$

where

$$g(\xi) = \frac{1}{4}\operatorname{sech}^2\left(\frac{\xi}{2}\right) f\left(\frac{1}{1+e^{-\xi}}\right), \tag{A.4}$$

is the probability density function of ξ. The inverse conversion is

$$f(\rho) = \frac{1}{\rho(1-\rho)}\, g\left(\ln\left(\frac{\rho}{1-\rho}\right)\right) \tag{A.5}$$

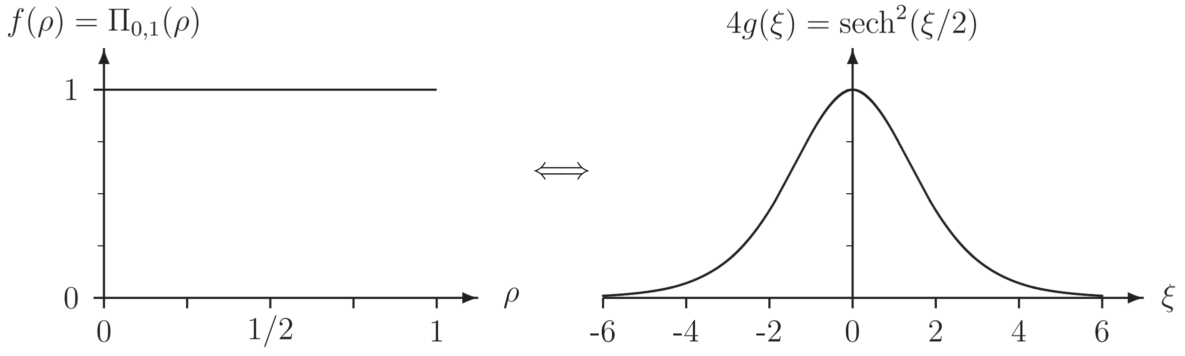


Figure A.1: Change of variables for integration.

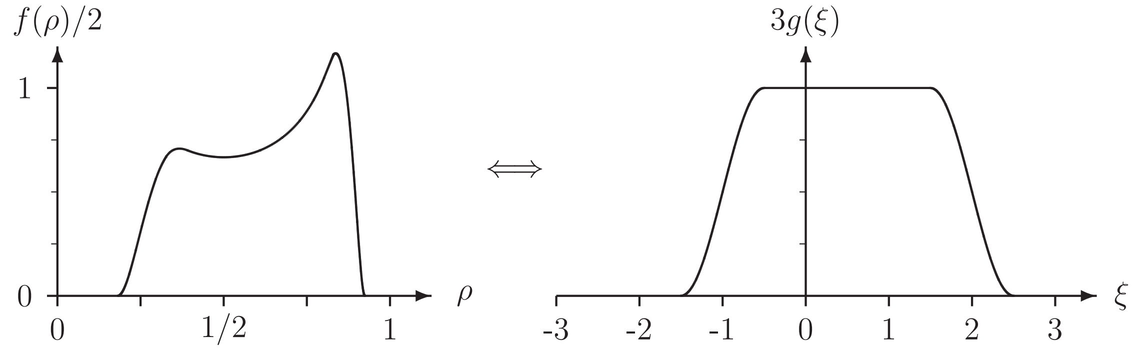


Figure A.2: Change of variables for integration.

Figure A.1 shows $g(\xi)$ when ρ has uniform distribution in the interval $[0, 1]$, and Figure A.2 shows another example, with the inverse transformation.

Appendix B

Discrete-Cauchy Distribution

B.1 Entropy Bounds

Discrete-Cauchy (DC) data sources have their symbol probabilities defined by

$$p_c(s,\alpha) = \frac{\alpha}{\alpha+s} - \frac{\alpha}{\alpha+s+1} = \frac{\alpha}{(\alpha+s)(\alpha+s+1)}, \quad s = 0, 1, 2, \ldots, \quad \alpha > 0. \tag{B.1}$$

In Section 4.3 we show that the entropy of this distribution is in the form (4.60)

$$H_c(\alpha) = \sum_{s=0}^{\infty} \left[\frac{\alpha}{(\alpha+s)(\alpha+s+1)}\right] \log_2\left(\frac{(\alpha+s)(\alpha+s+1)}{\alpha}\right). \tag{B.2}$$

Rearranging the sum terms as in (4.82), we obtain

$$H_c(\alpha) = \log_2(\alpha+1) + \sum_{s=1}^{\infty} \left(\frac{\alpha}{\alpha+s}\right) \log_2\left(\frac{\alpha+s+1}{\alpha+s-1}\right), \tag{B.3}$$

Figure B.1 show that even though (B.3) converges faster than (4.60), the convergence is still quite slow.

Proposition B.1.1 *The entropy of the DC distribution (B.3) has a lower bound defined by*

$$\begin{aligned} H_c(\alpha) &= \log_2(\alpha+1) + \sum_{s=1}^{t} \left(\frac{\alpha}{\alpha+s}\right) \log_2\left(\frac{\alpha+s+1}{\alpha+s-1}\right) + \\ &+ \frac{2\alpha}{\ln(2)(\alpha+t+1/2)} + \epsilon_a(\alpha), \end{aligned} \tag{B.4}$$

where

$$0 < \epsilon_a(\alpha) < \frac{0.1\alpha}{(\alpha+t)^3}. \tag{B.5}$$

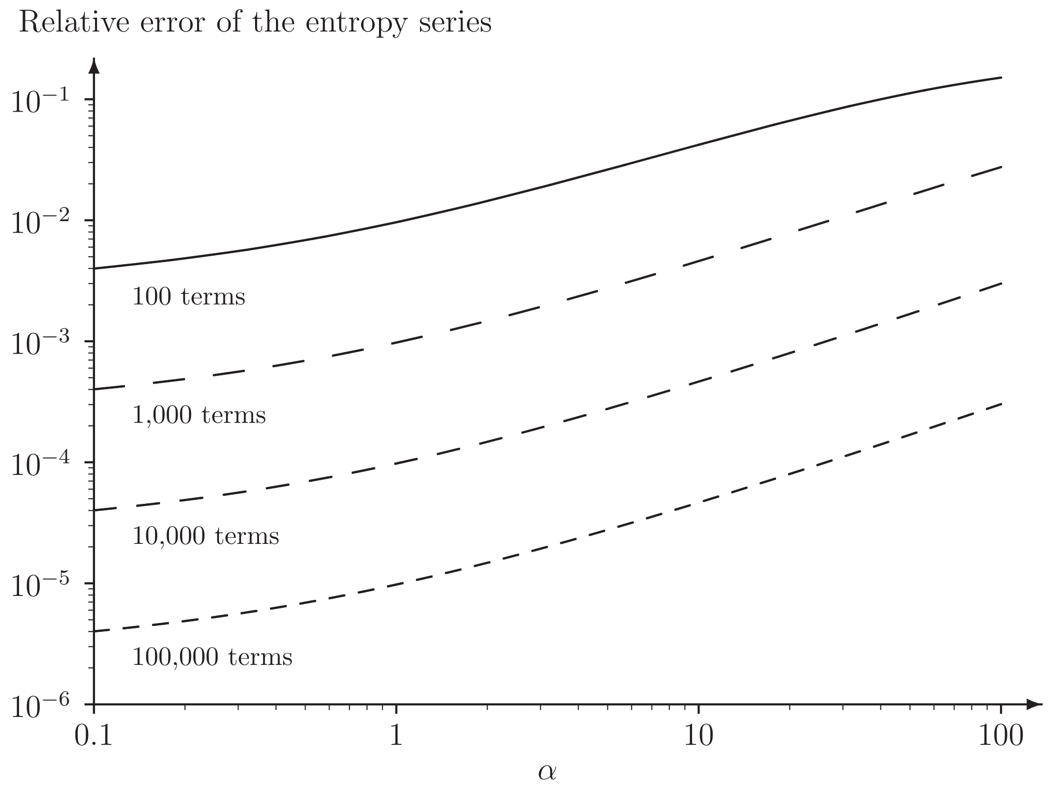


Figure B.1: Relative error in the direct computation of the entropy of the DC distribution using series (B.3).

Proof: We can use the series expansion

$$\left(\frac{\alpha}{\alpha+s}\right)\log_2\left(\frac{\alpha+s+1}{\alpha+s-1}\right) = \frac{2\alpha}{\ln(2)(\alpha+s)}\operatorname{arctanh}\left(\frac{1}{\alpha+s}\right) \tag{B.6}$$

$$= \frac{2\alpha}{\ln(2)}\sum_{n=1}^{\infty}\frac{1}{(2n-1)(\alpha+s)^{2n}} \tag{B.7}$$

to conclude that for α sufficiently large we have

$$\frac{\alpha}{\alpha+s}\log_2\left(\frac{\alpha+s+1}{\alpha+s-1}\right) \approx \frac{2\alpha}{\ln(2)(\alpha+s)^2}. \tag{B.8}$$

Thus, we can use Kummer's technique for acceleration of series [3, § 3.6.26], and approximate the sum of the first terms in the expansion of the logarithms using the following series with known sum

$$\sum_{s=t+1}^{\infty}\left(\frac{\alpha}{\alpha+s-1/2}-\frac{\alpha}{\alpha+s+1/2}\right) = \sum_{s=t+1}^{\infty}\frac{\alpha}{(\alpha+s)^2-1/4} = \frac{\alpha}{\alpha+t+1/2}. \tag{B.9}$$

The difference of the two series defines the error term

$$\epsilon_a(\alpha) = \frac{\alpha}{\ln(2)}\sum_{s=t+1}^{\infty}\left[\frac{1}{\alpha+s}\ln\left(\frac{\alpha+s+1}{\alpha+s-1}\right)-\frac{2}{(\alpha+s)^2-1/4}\right] \tag{B.10}$$

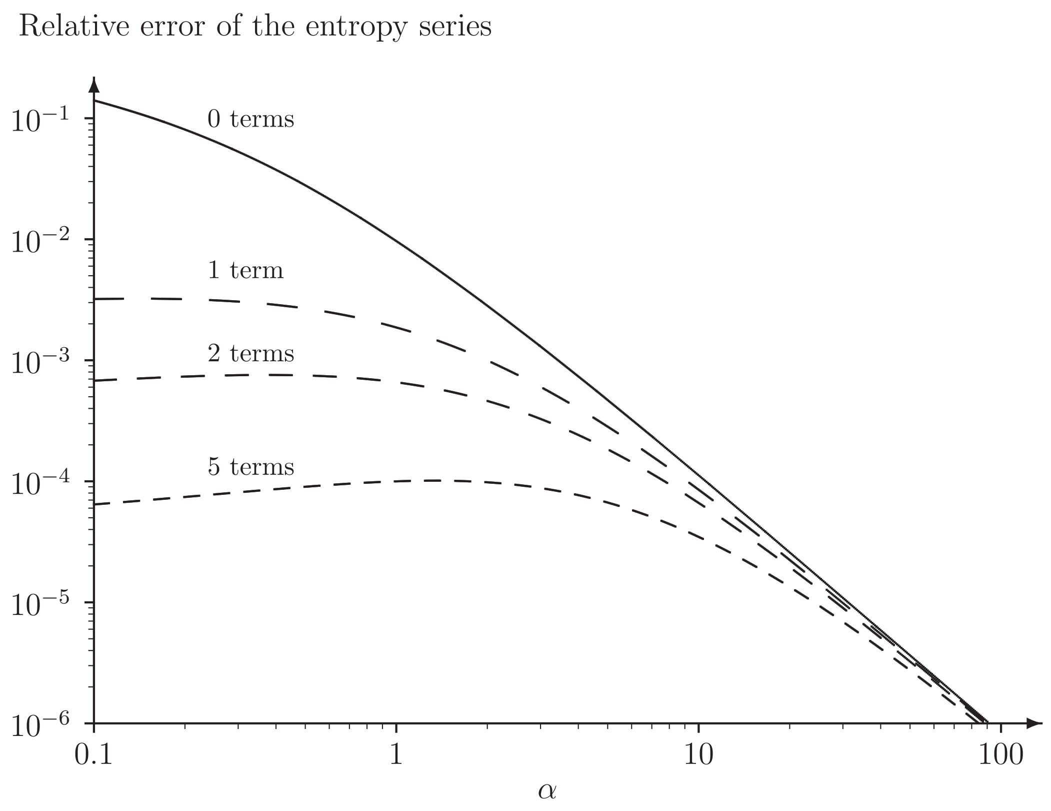


Figure B.2: Relative error in the accelerated computation of the entropy of the DC distribution using (B.4).

$$= \frac{2\alpha}{\ln(2)} \sum_{s=t+1}^{\infty} \sum_{n=2}^{\infty} \left(\frac{1}{2n-1} - \frac{1}{4^{n-1}} \right) \frac{1}{(\alpha+s)^{2n}}. \tag{B.11}$$

Since $\epsilon_a(\alpha)$ is a sum of positive terms we conclude that it is always positive. For large values of α the sum is dominated by the terms $\alpha/(\alpha+s)^4$, and thus the sum is proportional to $\alpha/(\alpha+t)^3$. The upper bound multiplier was computed numerically. ∎

Figure B.2 show the relative errors obtained with the new lower bound series (B.4). Comparing Figure B.2 with Figures 4.4 and B.1, we can observer a much faster convergence. In Section B.2 we extend the ideas used in Theorem B.1.1 to compute series that converge even faster, but before that we present another bound.

Proposition B.1.2 *The entropy of the DC distribution (B.3) has an upper bound defined by*

$$\begin{aligned} H_c(\alpha) &= \log_2(\alpha+1) + \sum_{s=1}^{t} \left(\frac{\alpha}{\alpha+s} \right) \log_2 \left(\frac{\alpha+s+1}{\alpha+s-1} \right) + \\ &+ \frac{\alpha[2(\alpha+t)+1]}{\ln(2)(\alpha+t)(\alpha+t+1)} - \epsilon_b(\alpha), \end{aligned} \tag{B.12}$$

where

$$0 < \epsilon_b(\alpha) < \frac{0.7\alpha}{(\alpha+t)^3}. \tag{B.13}$$

Proof: Similarly to Theorem B.1.1, we use Kummer acceleration, but with the series

$$\sum_{s=t+1}^{\infty}\left(\frac{\alpha}{\alpha+s-1}-\frac{\alpha}{\alpha+s+1}\right)=\sum_{s=t+1}^{\infty}\frac{\alpha}{(\alpha+s)^2-1}=\frac{\alpha}{\alpha+t+1}+\frac{\alpha}{\alpha+t+1}. \tag{B.14}$$

The error term in this case is a sum of negative terms

$$\epsilon_a(\alpha)=\frac{2\alpha}{\ln(2)}\sum_{s=t+1}^{\infty}\sum_{n=2}^{\infty}\left(\frac{1}{2n-1}-1\right)\frac{1}{(\alpha+s)^{2n}}. \tag{B.15}$$

∎

B.2 Fast Entropy Computation

Proposition B.2.1 *The difference between the entropy and the first t terms in its series expansion can be computed with increasing precision by first computing*

$$x=\alpha+t+1/2 \tag{B.16}$$

and using

$$\begin{aligned}
\Lambda(\alpha,t) &= \frac{\ln(2)}{\alpha}\left[H_c(\alpha)-\log_2(\alpha+1)-\sum_{s=1}^{t}\left(\frac{\alpha}{\alpha+s}\right)\log_2\left(\frac{\alpha+s+1}{\alpha+s-1}\right)\right] && \text{(B.17)}\\
&= \frac{2}{x}+O\left([\alpha+t]^{-3}\right) && \text{(B.18)}\\
&= \frac{2}{9}\left(\frac{8}{x}+\frac{x}{x^2-1/4}\right)+O\left([\alpha+t]^{-5}\right) && \text{(B.19)}\\
&= \frac{2}{225}\left[\frac{206}{x}+x\left(\frac{17}{x^2-1/4}+\frac{2}{x^2-1}\right)\right]+O\left([\alpha+t]^{-7}\right) && \text{(B.20)}\\
&= \frac{2}{33075}\left[\frac{31112}{x}+x\left(\frac{1254}{x^2-1/4}+\frac{792}{x^2-1}-\frac{83}{x^2-9/4}\right)\right]+O\left([\alpha+t]^{-9}\right) && \text{(B.21)}
\end{aligned}$$

where $O(\cdot)$ defines the asymptotic decrease of the error.

Proof: We can use the geometric series expansion

$$\frac{\alpha}{\alpha+s-\kappa/2}-\frac{\alpha}{\alpha+s+\kappa/2}=2\alpha\sum_{n=1}^{\infty}\left[\frac{\kappa}{2(\alpha+s)}\right]^{2n} \tag{B.22}$$

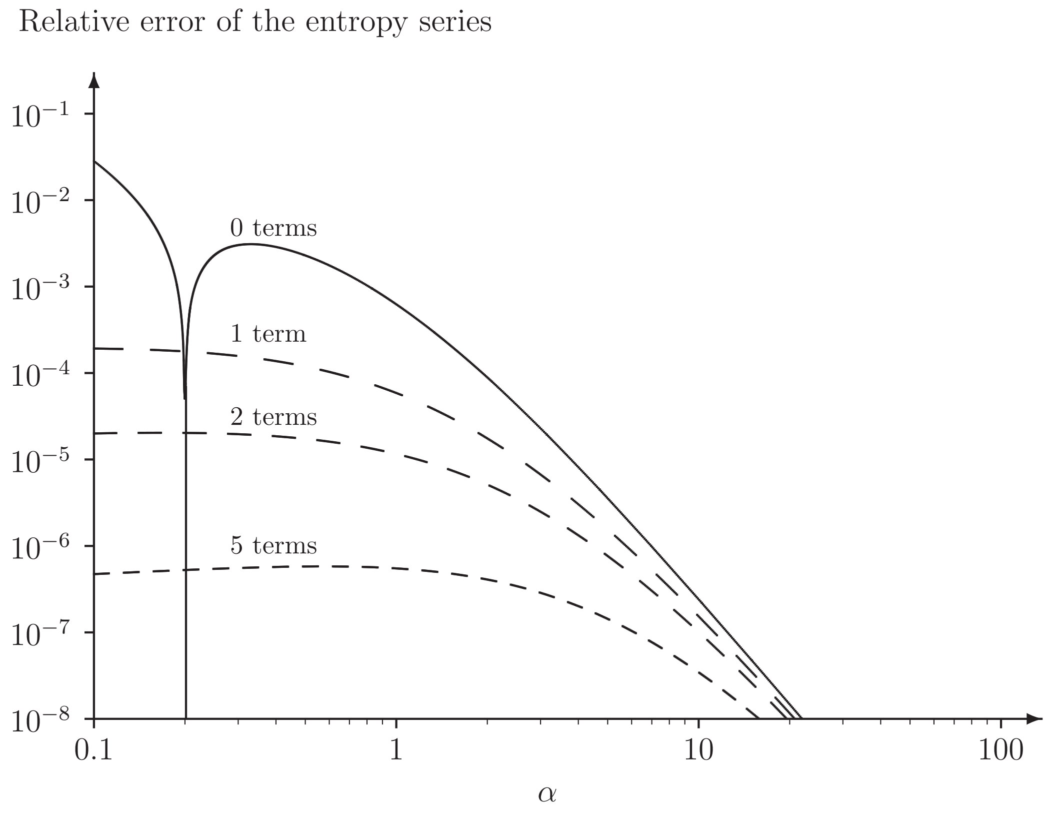


Figure B.3: Relative error in the accelerated computation of the entropy of the DC distribution using series (B.19).

and the fact that, for $\kappa = 1, 2, 3, \ldots$, we have

$$\sum_{s=t+1}^{\infty}\left(\frac{\alpha}{\alpha+s-\kappa/2}-\frac{\alpha}{\alpha+s+\kappa/2}\right) = \sum_{s=t+1}^{\infty} 2\alpha \sum_{n=1}^{\infty}\left[\frac{\kappa}{2(\alpha+s)}\right]^{2n} \tag{B.23}$$

$$= \sum_{k=1}^{\kappa}\frac{\alpha}{\alpha+t+k-\kappa/2}. \tag{B.24}$$

We can use linear combinations of this sums to cancel the corresponding terms in the series

$$\sum_{s=t+1}^{\infty}\left(\frac{\alpha}{\alpha+s}\right)\ln\left(\frac{\alpha+s+1}{\alpha+s-1}\right) = \sum_{s=t+1}^{\infty} 2\alpha \sum_{n=1}^{\infty}\frac{1}{(2n-1)(\alpha+s)^{2n}}, \tag{B.25}$$

and solving the linear systems of equations gives the desired results. ∎

Appendix C

Code Tables

The tables with code parameters and other information are divided in following groups:

- Tables C.1 and C.3 have the code parameters computed using the updated maximum-likelihood estimated values of ρ, for several values of N_s and Σ_s. Table C.1 shows the code parameters $m_d(\Sigma_s, N_s)$ defined by (4.9), and Table C.3 shows $k_d(\Sigma_s, N_s)$ defined by (4.20). Tables C.2 and C.4 show the respective values of $\hat{\rho}_d$, computed according to (4.12).

- Tables C.5 to C.7 present the first parameters that define the optimal unary-stem codes for different priors and values of N_s and Σ_s. Tables C.8 to C.10 have the parameters of the optimal dyadic unary-stem codes. Tables C.5 and C.8 we computed using the Dirichlet (1/2) prior, tables C.6 and C.9 using the uniform distribution prior, and tables C.7 and C.10 using the empiric prior $g_e(\xi)$ (Figure 4.3).

- Tables C.11 to C.46 present comparisons of the different codes, showing both code parameters and redundancy, for different priors and values of N_s and Σ_s. The types of codes are listed in Table 6.1

    1. Tables C.11 to C.22 were obtained assuming a Dirichlet (1/2) prior.
    2. Tables C.23 to C.34 were obtained assuming an uniform prior.
    3. Tables C.35 to C.46 were obtained assuming the empiric prior $g_e(\xi)$.

Table C.1: Examples of unary-stem parameter sequences $m_d(\Sigma_s, N_s)$, defined using the maximum-likelihood source estimation.

N_s	Σ_s	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
1	0	1	1	2	3	5	9	15	25	43	72	122	207	350
1	1	1	2	3	5	9	15	25	43	72	122	207	350	593
1	2	2	3	5	9	15	25	43	72	122	207	350	593	1004
1	4	3	5	9	15	25	43	72	122	207	350	593	1004	1700
1	7	5	9	15	25	43	72	122	207	350	593	1004	1700	2878
1	12	9	15	25	43	72	122	207	350	593	1004	1700	2878	4873
1	20	14	24	41	69	117	198	335	567	960	1626	2753	4661	7892
1	50	35	59	100	169	287	486	822	1392	2357	3991	6757	11441	19371
1	80	56	95	160	271	459	777	1316	2228	3772	6387	10814	18310	31001
2	0	1	1	1	1	2	2	3	4	6	8	10	14	19
2	1	1	1	1	2	2	3	4	6	8	10	14	19	25
2	2	1	1	2	2	3	4	6	8	10	14	19	25	34
2	5	2	3	4	5	7	9	12	17	23	31	41	55	75
2	9	3	5	6	8	11	15	20	27	36	49	66	89	120
2	16	6	8	11	15	20	27	36	49	66	88	119	160	216
2	25	9	12	16	22	29	40	53	72	97	130	175	236	318
2	60	21	28	38	51	69	93	125	168	227	305	411	553	745
2	100	35	47	63	85	115	155	208	280	377	508	684	921	1240
3	1	1	1	1	1	2	2	2	3	4	5	6	7	9
3	2	1	1	1	2	2	2	3	4	5	6	7	9	11
3	4	1	2	2	2	3	4	5	6	7	9	11	13	16
3	6	2	2	3	3	4	5	6	8	9	11	14	17	21
3	10	3	3	4	5	6	8	9	11	14	17	21	26	32
3	20	5	6	8	9	11	14	17	21	26	32	39	48	59
3	30	7	9	11	14	17	21	26	32	39	48	59	73	90
3	80	19	23	29	35	43	53	66	81	99	122	151	185	228
3	150	35	43	53	65	80	99	122	150	184	227	279	344	423
5	2	1	1	1	1	1	1	1	2	2	2	2	3	3
5	3	1	1	1	1	1	1	2	2	2	2	3	3	4
5	5	1	1	1	1	2	2	2	2	3	3	4	4	5
5	8	1	2	2	2	2	3	3	4	4	5	5	6	7
5	16	3	3	3	4	4	5	6	6	7	8	9	11	12
5	30	5	5	6	7	8	9	10	11	13	15	17	19	22
5	60	9	10	11	13	15	17	19	22	25	28	32	37	42
5	100	14	16	18	21	24	27	31	35	40	46	52	59	67
5	200	28	32	36	41	47	54	61	70	79	90	103	117	133
8	4	1	1	1	1	1	1	1	1	1	2	2	2	2
8	6	1	1	1	1	1	1	1	2	2	2	2	2	2
8	8	1	1	1	1	1	2	2	2	2	2	2	3	3
8	12	1	2	2	2	2	2	2	3	3	3	3	4	4
8	20	2	2	2	3	3	3	3	4	4	4	5	5	6
8	50	5	5	6	6	7	7	8	8	9	10	11	12	13
8	80	7	8	9	9	10	11	12	13	14	15	17	18	20
8	150	13	14	16	17	19	20	22	24	26	28	31	33	36
8	300	26	29	31	34	37	40	43	47	51	56	60	66	71

Table C.2: Estimated values of $\hat{\rho}_d$ using the maximum-likelihood criterion, corresponding to the unary-stem codes of Table C.1.

N_s	Σ_s	$\hat{\rho}_0$	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$	$\hat{\rho}_6$	$\hat{\rho}_7$	$\hat{\rho}_8$	$\hat{\rho}_9$
1	0	0.0000	0.5000	0.6667	0.8000	0.8750	0.9231	0.9545	0.9730	0.9839	0.9905
1	1	0.5000	0.6667	0.8000	0.8750	0.9231	0.9545	0.9730	0.9839	0.9905	0.9944
1	2	0.6667	0.8000	0.8750	0.9231	0.9545	0.9730	0.9839	0.9905	0.9944	0.9967
1	4	0.8000	0.8750	0.9231	0.9545	0.9730	0.9839	0.9905	0.9944	0.9967	0.9980
1	7	0.8750	0.9231	0.9545	0.9730	0.9839	0.9905	0.9944	0.9967	0.9980	0.9988
1	12	0.9231	0.9545	0.9730	0.9839	0.9905	0.9944	0.9967	0.9980	0.9988	0.9993
1	20	0.9524	0.9714	0.9831	0.9900	0.9941	0.9965	0.9979	0.9988	0.9993	0.9996
1	50	0.9804	0.9884	0.9931	0.9959	0.9976	0.9986	0.9992	0.9995	0.9997	0.9998
1	80	0.9877	0.9927	0.9957	0.9974	0.9985	0.9991	0.9995	0.9997	0.9998	0.9999
2	0	0.0000	0.3333	0.5000	0.6000	0.6667	0.7500	0.8000	0.8462	0.8824	0.9130
2	1	0.3333	0.5000	0.6000	0.6667	0.7500	0.8000	0.8462	0.8824	0.9130	0.9355
2	2	0.5000	0.6000	0.6667	0.7500	0.8000	0.8462	0.8824	0.9130	0.9355	0.9512
2	5	0.7143	0.7778	0.8333	0.8750	0.9048	0.9286	0.9459	0.9592	0.9697	0.9775
2	9	0.8182	0.8571	0.8947	0.9200	0.9394	0.9545	0.9661	0.9747	0.9811	0.9859
2	16	0.8889	0.9167	0.9375	0.9535	0.9655	0.9744	0.9810	0.9858	0.9895	0.9922
2	25	0.9259	0.9444	0.9583	0.9688	0.9767	0.9826	0.9871	0.9904	0.9929	0.9947
2	60	0.9677	0.9759	0.9820	0.9866	0.9900	0.9926	0.9945	0.9959	0.9969	0.9977
2	100	0.9804	0.9854	0.9891	0.9919	0.9940	0.9955	0.9967	0.9975	0.9982	0.9986
3	1	0.2500	0.4000	0.5000	0.5714	0.6250	0.7000	0.7500	0.7857	0.8235	0.8571
3	2	0.4000	0.5000	0.5714	0.6250	0.7000	0.7500	0.7857	0.8235	0.8571	0.8846
3	4	0.5714	0.6250	0.7000	0.7500	0.7857	0.8235	0.8571	0.8846	0.9063	0.9231
3	6	0.6667	0.7273	0.7692	0.8125	0.8421	0.8696	0.8929	0.9118	0.9286	0.9412
3	10	0.7692	0.8125	0.8421	0.8696	0.8929	0.9118	0.9286	0.9412	0.9516	0.9605
3	20	0.8696	0.8929	0.9118	0.9286	0.9412	0.9516	0.9605	0.9677	0.9737	0.9786
3	30	0.9091	0.9250	0.9388	0.9500	0.9595	0.9670	0.9732	0.9783	0.9824	0.9856
3	80	0.9639	0.9706	0.9760	0.9805	0.9841	0.9871	0.9895	0.9915	0.9931	0.9944
3	150	0.9804	0.9840	0.9870	0.9894	0.9914	0.9930	0.9943	0.9954	0.9962	0.9970
5	2	0.2857	0.3750	0.4444	0.5000	0.5455	0.5833	0.6154	0.6429	0.6875	0.7222
5	3	0.3750	0.4444	0.5000	0.5455	0.5833	0.6154	0.6429	0.6875	0.7222	0.7500
5	5	0.5000	0.5455	0.5833	0.6154	0.6429	0.6875	0.7222	0.7500	0.7727	0.8000
5	8	0.6154	0.6429	0.6875	0.7222	0.7500	0.7727	0.8000	0.8214	0.8438	0.8611
5	16	0.7619	0.7917	0.8148	0.8333	0.8529	0.8684	0.8837	0.8980	0.9091	0.9194
5	30	0.8571	0.8750	0.8889	0.9020	0.9138	0.9242	0.9333	0.9412	0.9479	0.9541
5	60	0.9231	0.9324	0.9405	0.9474	0.9537	0.9593	0.9643	0.9686	0.9724	0.9757
5	100	0.9524	0.9580	0.9630	0.9673	0.9713	0.9747	0.9778	0.9805	0.9828	0.9849
5	200	0.9756	0.9785	0.9811	0.9834	0.9854	0.9871	0.9887	0.9901	0.9913	0.9923
8	4	0.3333	0.3846	0.4286	0.4667	0.5000	0.5294	0.5556	0.5789	0.6000	0.6190
8	6	0.4286	0.4667	0.5000	0.5294	0.5556	0.5789	0.6000	0.6190	0.6522	0.6800
8	8	0.5000	0.5294	0.5556	0.5789	0.6000	0.6190	0.6522	0.6800	0.7037	0.7241
8	12	0.6000	0.6190	0.6522	0.6800	0.7037	0.7241	0.7419	0.7576	0.7778	0.7949
8	20	0.7143	0.7333	0.7500	0.7647	0.7838	0.8000	0.8140	0.8261	0.8400	0.8519
8	50	0.8621	0.8730	0.8824	0.8919	0.9000	0.9080	0.9149	0.9216	0.9273	0.9328
8	80	0.9091	0.9158	0.9223	0.9286	0.9339	0.9389	0.9437	0.9481	0.9521	0.9558
8	150	0.9494	0.9532	0.9568	0.9602	0.9633	0.9662	0.9689	0.9713	0.9736	0.9757
8	300	0.9740	0.9760	0.9780	0.9797	0.9813	0.9828	0.9842	0.9854	0.9866	0.9876

Table C.3: Examples of dyadic unary-stem codes parameter sequences $k_d(\Sigma_s, N_s)$, defined using the maximum-likelihood source estimation.

N_s	Σ_s	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}
1	0	0	0	1	2	3	3	4	5	6	7	7	8	9
1	1	0	1	2	3	3	4	5	6	7	7	8	9	10
1	2	1	2	3	3	4	5	6	7	7	8	9	10	11
1	4	2	3	3	4	5	6	7	7	8	9	10	11	11
1	7	2	3	4	5	6	6	7	8	9	10	10	11	12
1	12	3	4	5	6	6	7	8	9	10	10	11	12	13
1	20	4	5	6	6	7	8	9	10	10	11	12	13	14
1	50	5	6	7	8	8	9	10	11	12	12	13	14	15
1	80	6	7	8	8	9	10	11	12	12	13	14	15	16
2	0	0	0	0	0	1	1	2	2	3	3	3	4	4
2	1	0	0	0	1	1	2	2	3	3	3	4	4	5
2	2	0	0	1	1	2	2	3	3	3	4	4	5	5
2	5	1	1	2	2	3	3	4	4	4	5	5	6	6
2	9	2	2	3	3	4	4	4	5	5	6	6	7	7
2	16	3	3	3	4	4	5	5	6	6	6	7	7	8
2	25	3	4	4	4	5	5	6	6	7	7	7	8	8
2	60	4	5	5	6	6	7	7	7	8	8	9	9	10
2	100	5	5	6	6	7	7	8	8	8	9	9	10	10
3	1	0	0	0	0	1	1	1	1	2	2	2	3	3
3	2	0	0	0	1	1	1	1	2	2	2	3	3	3
3	4	0	1	1	1	1	2	2	2	3	3	3	4	4
3	6	1	1	1	2	2	2	3	3	3	3	4	4	4
3	10	1	2	2	2	3	3	3	3	4	4	4	5	5
3	20	2	3	3	3	3	4	4	4	5	5	5	6	6
3	30	3	3	3	4	4	4	5	5	5	5	6	6	6
3	80	4	4	5	5	5	6	6	6	7	7	7	7	8
3	150	5	5	6	6	6	7	7	7	7	8	8	8	9
5	2	0	0	0	0	0	0	0	1	1	1	1	1	2
5	3	0	0	0	0	0	0	1	1	1	1	1	2	2
5	5	0	0	0	0	1	1	1	1	1	2	2	2	2
5	8	0	1	1	1	1	1	2	2	2	2	2	2	3
5	16	1	1	2	2	2	2	2	3	3	3	3	3	3
5	30	2	2	2	3	3	3	3	3	4	4	4	4	4
5	60	3	3	3	4	4	4	4	4	5	5	5	5	5
5	100	4	4	4	4	5	5	5	5	5	5	6	6	6
5	200	5	5	5	5	5	6	6	6	6	6	7	7	7
8	4	0	0	0	0	0	0	0	0	0	1	1	1	1
8	6	0	0	0	0	0	0	0	1	1	1	1	1	1
8	8	0	0	0	0	0	1	1	1	1	1	1	1	1
8	12	0	1	1	1	1	1	1	1	1	1	2	2	2
8	20	1	1	1	1	1	2	2	2	2	2	2	2	2
8	50	2	2	2	2	3	3	3	3	3	3	3	3	4
8	80	3	3	3	3	3	3	3	4	4	4	4	4	4
8	150	4	4	4	4	4	4	4	4	5	5	5	5	5
8	300	5	5	5	5	5	5	5	5	6	6	6	6	6

Table C.4: Estimated values of $\hat{\rho}_d$ using the maximum-likelihood criterion, corresponding to the dyadic unary-stem codes of Table C.3.

N_s	Σ_s	$\hat{\rho}_0$	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$	$\hat{\rho}_6$	$\hat{\rho}_7$	$\hat{\rho}_8$	$\hat{\rho}_9$
1	0	0.0000	0.5000	0.6667	0.8000	0.8889	0.9412	0.9600	0.9756	0.9863	0.9927
1	1	0.5000	0.6667	0.8000	0.8889	0.9412	0.9600	0.9756	0.9863	0.9927	0.9962
1	2	0.6667	0.8000	0.8889	0.9412	0.9600	0.9756	0.9863	0.9927	0.9962	0.9975
1	4	0.8000	0.8889	0.9412	0.9600	0.9756	0.9863	0.9927	0.9962	0.9975	0.9985
1	7	0.8750	0.9167	0.9500	0.9722	0.9853	0.9924	0.9949	0.9969	0.9983	0.9991
1	12	0.9231	0.9524	0.9730	0.9855	0.9925	0.9949	0.9969	0.9983	0.9991	0.9995
1	20	0.9524	0.9730	0.9855	0.9925	0.9949	0.9969	0.9983	0.9991	0.9995	0.9997
1	50	0.9804	0.9880	0.9932	0.9964	0.9981	0.9987	0.9992	0.9996	0.9998	0.9999
1	80	0.9877	0.9931	0.9963	0.9981	0.9987	0.9992	0.9996	0.9998	0.9999	0.9999
2	0	0.0000	0.3333	0.5000	0.6000	0.6667	0.7500	0.8000	0.8571	0.8889	0.9231
2	1	0.3333	0.5000	0.6000	0.6667	0.7500	0.8000	0.8571	0.8889	0.9231	0.9412
2	2	0.5000	0.6000	0.6667	0.7500	0.8000	0.8571	0.8889	0.9231	0.9412	0.9524
2	5	0.7143	0.7778	0.8182	0.8667	0.8947	0.9259	0.9429	0.9608	0.9701	0.9759
2	9	0.8182	0.8667	0.8947	0.9259	0.9429	0.9608	0.9701	0.9759	0.9826	0.9864
2	16	0.8889	0.9231	0.9412	0.9524	0.9655	0.9730	0.9811	0.9855	0.9901	0.9925
2	25	0.9259	0.9429	0.9608	0.9701	0.9759	0.9826	0.9864	0.9905	0.9927	0.9950
2	60	0.9677	0.9744	0.9818	0.9859	0.9903	0.9926	0.9950	0.9962	0.9969	0.9978
2	100	0.9804	0.9851	0.9880	0.9913	0.9932	0.9953	0.9964	0.9975	0.9981	0.9985
3	1	0.2500	0.4000	0.5000	0.5714	0.6250	0.7000	0.7500	0.7857	0.8125	0.8500
3	2	0.4000	0.5000	0.5714	0.6250	0.7000	0.7500	0.7857	0.8125	0.8500	0.8750
3	4	0.5714	0.6250	0.7000	0.7500	0.7857	0.8125	0.8500	0.8750	0.8929	0.9167
3	6	0.6667	0.7273	0.7692	0.8000	0.8421	0.8696	0.8889	0.9143	0.9302	0.9412
3	10	0.7692	0.8000	0.8421	0.8696	0.8889	0.9143	0.9302	0.9412	0.9492	0.9600
3	20	0.8696	0.8889	0.9143	0.9302	0.9412	0.9492	0.9600	0.9670	0.9720	0.9784
3	30	0.9091	0.9268	0.9388	0.9474	0.9589	0.9663	0.9714	0.9781	0.9822	0.9851
3	80	0.9639	0.9697	0.9739	0.9796	0.9832	0.9858	0.9891	0.9912	0.9926	0.9944
3	150	0.9804	0.9838	0.9862	0.9893	0.9913	0.9927	0.9944	0.9955	0.9962	0.9967
5	2	0.2857	0.3750	0.4444	0.5000	0.5455	0.5833	0.6154	0.6429	0.6875	0.7222
5	3	0.3750	0.4444	0.5000	0.5455	0.5833	0.6154	0.6429	0.6875	0.7222	0.7500
5	5	0.5000	0.5455	0.5833	0.6154	0.6429	0.6875	0.7222	0.7500	0.7727	0.7917
5	8	0.6154	0.6429	0.6875	0.7222	0.7500	0.7727	0.7917	0.8214	0.8438	0.8611
5	16	0.7619	0.7826	0.8000	0.8276	0.8485	0.8649	0.8780	0.8889	0.9057	0.9180
5	30	0.8571	0.8718	0.8837	0.8936	0.9091	0.9206	0.9296	0.9367	0.9425	0.9515
5	60	0.9231	0.9315	0.9383	0.9438	0.9524	0.9587	0.9635	0.9673	0.9704	0.9751
5	100	0.9524	0.9587	0.9635	0.9673	0.9704	0.9751	0.9785	0.9811	0.9832	0.9848
5	200	0.9756	0.9789	0.9814	0.9834	0.9850	0.9863	0.9883	0.9899	0.9910	0.9919
8	4	0.3333	0.3846	0.4286	0.4667	0.5000	0.5294	0.5556	0.5789	0.6000	0.6190
8	6	0.4286	0.4667	0.5000	0.5294	0.5556	0.5789	0.6000	0.6190	0.6522	0.6800
8	8	0.5000	0.5294	0.5556	0.5789	0.6000	0.6190	0.6522	0.6800	0.7037	0.7241
8	12	0.6000	0.6190	0.6522	0.6800	0.7037	0.7241	0.7419	0.7576	0.7714	0.7838
8	20	0.7143	0.7333	0.7500	0.7647	0.7778	0.7895	0.8095	0.8261	0.8400	0.8519
8	50	0.8621	0.8710	0.8788	0.8857	0.8919	0.9024	0.9111	0.9184	0.9245	0.9298
8	80	0.9091	0.9167	0.9231	0.9286	0.9333	0.9375	0.9412	0.9444	0.9500	0.9545
8	150	0.9494	0.9540	0.9579	0.9612	0.9640	0.9664	0.9685	0.9704	0.9720	0.9748
8	300	0.9740	0.9765	0.9785	0.9802	0.9817	0.9829	0.9840	0.9850	0.9858	0.9873

Table C.5: First values of the optimal unary-stem code parameter sequence $\mathcal{M}^*$, for distributions defined by the Dirichlet (1/2) prior.

N_s	Σ_s	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
0	0	1	3	12	47	188	753	3012	12039	48006	188453	648860	—	—
1	0	1	1	2	3	5	7	12	20	27	46	80	106	183
1	1	1	2	3	5	7	12	20	27	46	80	106	183	320
1	2	2	3	5	7	12	20	27	46	80	106	183	320	421
1	4	3	5	7	12	20	27	46	80	106	183	320	421	730
1	7	5	7	12	20	27	46	80	106	183	320	421	730	1277
1	12	7	12	20	27	46	80	106	183	320	421	730	1277	1679
1	20	12	21	27	47	81	107	184	322	423	733	1280	1683	2921
1	50	26	46	80	105	182	319	419	729	1274	1676	2912	5090	6691
1	80	47	81	107	184	322	423	733	1280	1683	2921	5103	6706	11612
2	0	1	1	1	1	2	2	3	4	5	6	9	11	14
2	1	1	1	1	2	2	3	4	5	6	9	11	14	20
2	2	1	1	2	2	3	4	5	6	9	11	14	20	25
2	5	2	3	3	5	6	7	11	13	19	23	28	42	51
2	9	3	5	6	7	10	13	19	23	28	41	50	75	91
2	16	6	7	10	12	18	22	27	40	49	74	90	109	162
2	25	9	11	14	20	25	37	45	55	81	100	149	181	220
2	60	20	25	37	45	55	82	100	149	181	221	327	400	596
2	100	29	43	52	77	94	114	168	205	305	371	451	666	813
3	1	1	1	1	1	1	2	2	3	3	4	5	6	7
3	2	1	1	1	1	2	2	3	3	4	5	6	7	9
3	4	1	1	2	2	3	3	4	5	6	7	9	11	13
3	6	2	2	3	3	4	5	6	7	9	11	13	15	20
3	10	3	3	4	5	6	7	9	11	13	15	20	23	27
3	20	5	6	7	9	11	13	15	20	23	27	37	43	50
3	30	7	9	11	12	14	19	23	27	36	42	49	57	78
3	80	19	22	26	35	41	48	56	76	89	104	142	166	193
3	150	36	42	49	57	77	90	105	144	168	196	228	309	361
5	2	1	1	1	1	1	1	1	2	2	2	2	3	3
5	3	1	1	1	1	1	1	2	2	2	2	3	3	3
5	5	1	1	1	1	2	2	2	2	3	3	3	4	5
5	8	1	2	2	2	2	3	3	3	4	5	5	6	6
5	16	3	3	3	4	4	5	5	6	7	7	9	10	11
5	30	5	5	6	6	7	9	10	11	12	13	15	18	20
5	60	9	10	11	12	14	15	19	21	23	26	29	36	40
5	100	13	15	18	20	23	25	28	35	39	43	48	53	59
5	200	26	29	36	40	45	50	56	69	77	85	95	106	117
8	4	1	1	1	1	1	1	1	1	1	1	2	2	2
8	6	1	1	1	1	1	1	1	1	2	2	2	2	2
8	8	1	1	1	1	1	1	2	2	2	2	2	2	3
8	12	1	1	2	2	2	2	2	2	3	3	3	3	3
8	20	2	2	2	3	3	3	3	3	4	4	5	5	5
8	50	5	5	5	6	6	7	7	8	9	10	10	11	12
8	80	7	8	9	9	10	11	12	13	14	15	17	18	20
8	150	13	14	15	17	19	20	22	23	25	27	29	31	36
8	300	25	27	29	34	37	39	42	46	49	53	57	61	71

Table C.6: First values of the optimal unary-stem code parameter sequence $\mathcal{M}^*$, for distributions defined by the uniform (1/2) prior.

N_s	Σ_s	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
0	0	1	2	3	6	12	23	46	92	183	364	727	1452	2901
1	0	1	1	1	2	3	3	5	7	10	13	21	26	41
1	1	1	1	2	3	3	5	7	10	13	21	26	41	52
1	2	1	2	3	3	5	7	10	13	21	26	41	52	82
1	4	2	3	5	6	10	12	20	25	40	50	80	101	160
1	7	3	5	6	10	13	20	25	40	51	80	101	161	203
1	12	6	7	11	14	22	27	43	54	84	106	167	210	331
1	20	10	12	20	25	40	50	80	101	160	202	321	404	642
1	50	22	27	43	54	84	106	167	210	331	416	658	828	1310
1	80	38	48	77	98	156	197	315	397	633	799	1272	1605	2553
2	0	1	1	1	1	2	2	3	3	5	6	7	9	11
2	1	1	1	1	2	2	3	3	5	6	7	9	11	13
2	2	1	1	2	2	3	3	5	6	7	9	11	13	19
2	5	2	2	3	3	5	6	7	10	12	14	19	23	27
2	9	3	3	5	6	7	10	12	14	19	23	27	39	46
2	16	5	6	7	10	12	14	20	23	28	39	46	55	78
2	25	7	9	11	13	19	22	27	38	45	53	76	90	107
2	60	15	21	24	29	41	48	57	80	95	113	159	189	224
2	100	26	36	43	52	73	87	104	148	176	209	297	353	421
3	1	1	1	1	1	1	2	2	2	3	3	4	5	6
3	2	1	1	1	1	2	2	2	3	3	4	5	6	6
3	4	1	1	2	2	2	3	3	4	5	6	6	7	10
3	6	2	2	2	3	3	4	5	6	6	7	10	11	13
3	10	2	3	3	4	5	6	6	7	10	11	13	15	19
3	20	5	5	6	7	9	10	12	14	18	21	24	28	37
3	30	6	7	9	10	12	14	18	21	24	28	37	42	48
3	80	15	19	22	25	29	38	44	50	58	76	87	100	115
3	150	27	36	42	48	55	73	84	96	110	146	167	192	221
5	2	1	1	1	1	1	1	1	2	2	2	2	3	3
5	3	1	1	1	1	1	1	2	2	2	2	3	3	3
5	5	1	1	1	1	2	2	2	2	3	3	3	3	4
5	8	1	2	2	2	2	3	3	3	3	4	5	5	6
5	16	2	3	3	3	4	4	5	5	6	6	7	9	10
5	30	4	5	5	6	6	7	9	10	11	12	13	14	17
5	60	7	9	10	11	12	13	15	18	20	22	24	27	29
5	100	12	14	15	18	20	22	25	27	30	36	40	44	49
5	200	25	27	30	36	40	44	49	54	60	72	80	88	97
8	4	1	1	1	1	1	1	1	1	1	1	2	2	2
8	6	1	1	1	1	1	1	1	1	2	2	2	2	2
8	8	1	1	1	1	1	1	2	2	2	2	2	2	3
8	12	1	1	2	2	2	2	2	2	3	3	3	3	3
8	20	2	2	2	3	3	3	3	3	4	4	4	5	5
8	50	5	5	5	6	6	6	7	7	8	9	10	10	11
8	80	7	7	8	9	9	10	11	12	12	13	14	15	18
8	150	12	13	14	15	17	19	20	21	23	24	26	28	30
8	300	24	26	28	30	34	37	39	42	45	48	52	56	60

Table C.7: First values of the optimal unary-stem code parameter sequence $\mathcal{M}^*$, for distributions defined by the empiric prior $g_e(\xi)$.

N_s	Σ_s	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
0	0	12	47	95	157	188	220	281	309	336	362	386	408	429
1	0	2	3	6	11	22	40	53	87	108	154	179	205	231
1	1	3	5	7	13	24	43	56	90	111	158	183	209	235
1	2	3	6	11	22	40	53	87	108	154	179	205	231	283
1	4	5	7	13	24	43	56	90	111	158	183	209	235	286
1	7	6	12	23	41	54	88	109	156	181	206	232	284	309
1	12	11	22	40	53	87	108	155	180	205	231	283	308	333
1	20	20	27	48	81	102	148	173	199	225	277	302	327	351
1	50	42	55	89	110	157	182	208	233	285	310	334	358	380
1	80	52	85	106	153	178	203	229	281	306	331	354	377	398
2	0	1	2	2	3	3	5	6	10	13	20	25	39	48
2	1	2	2	3	3	5	6	10	13	20	25	39	48	58
2	2	2	2	3	5	6	10	12	19	24	38	47	57	83
2	5	3	3	5	6	10	13	20	25	39	48	58	84	100
2	9	5	6	10	12	19	24	38	47	57	83	99	116	153
2	16	6	10	13	20	25	39	48	58	84	100	117	155	176
2	25	11	13	21	26	41	50	75	90	107	144	164	185	207
2	60	24	37	46	56	82	98	115	152	173	194	216	239	284
2	100	41	51	76	91	108	145	165	187	208	231	277	299	322
3	1	1	1	2	2	2	3	3	5	6	7	9	11	13
3	2	1	2	2	2	3	3	5	6	7	9	11	13	19
3	4	2	2	3	3	4	5	6	7	10	12	14	20	24
3	6	2	3	3	4	5	6	7	10	12	14	20	24	28
3	10	3	3	5	6	7	10	12	14	19	23	27	38	45
3	20	6	7	9	11	13	19	22	26	37	44	52	71	83
3	30	7	10	12	14	20	24	28	39	46	54	74	86	99
3	80	21	25	36	42	50	58	79	91	105	119	151	168	186
3	150	40	47	56	76	88	101	115	147	164	182	201	220	240
5	2	1	1	1	1	2	2	2	2	3	3	3	4	5
5	3	1	1	1	2	2	2	2	3	3	3	4	5	5
5	5	1	2	2	2	2	3	3	3	4	5	5	6	6
5	8	2	2	2	3	3	3	4	5	5	6	6	7	9
5	16	3	3	3	4	5	5	6	7	9	10	11	12	14
5	30	5	6	6	7	9	10	11	12	14	18	20	22	25
5	60	10	11	12	14	17	19	22	24	27	34	39	43	48
5	100	14	18	20	23	26	29	36	41	46	51	57	71	79
5	200	29	36	40	45	51	57	71	79	88	97	107	118	141
8	4	1	1	1	1	1	1	1	2	2	2	2	2	2
8	6	1	1	1	1	1	2	2	2	2	2	2	2	3
8	8	1	1	1	2	2	2	2	2	2	2	3	3	3
8	12	2	2	2	2	2	2	3	3	3	3	3	4	4
8	20	2	2	3	3	3	3	3	4	4	5	5	5	6
8	50	5	5	6	6	7	7	8	9	10	11	11	12	13
8	80	7	9	9	10	11	12	13	14	15	17	19	20	22
8	150	14	15	17	19	20	22	23	25	27	29	34	37	40
8	300	27	29	34	37	40	43	46	50	54	58	68	73	79

Table C.8: First values of the optimal dyadic unary-stem code parameter sequence $\mathcal{K}^*$, for distributions defined by the Dirichlet (1/2) prior.

N_s	Σ_s	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}
0	0	0	1	3	5	6	8	10	11	13	15	17	18	19
1	0	0	0	0	1	2	2	3	3	4	5	5	6	6
1	1	0	0	1	2	2	3	3	4	5	5	6	6	7
1	2	0	1	2	2	3	3	4	5	5	6	6	7	8
1	4	1	2	2	3	4	4	5	5	6	7	7	8	8
1	7	2	3	3	4	5	5	6	6	7	8	8	9	9
1	12	3	3	4	5	5	6	6	7	8	8	9	9	10
1	20	3	4	5	5	6	6	7	8	8	9	9	10	11
1	50	5	5	6	7	7	8	8	9	10	10	11	11	12
1	80	5	6	7	7	8	8	9	10	10	11	11	12	13
2	0	0	0	0	0	1	1	1	2	2	3	3	3	4
2	1	0	0	0	1	1	1	2	2	3	3	3	4	4
2	2	0	0	1	1	1	2	2	3	3	3	4	4	4
2	5	1	1	2	2	2	3	3	3	4	4	5	5	5
2	9	2	2	2	3	3	3	4	4	5	5	5	6	6
2	16	2	3	3	3	4	4	5	5	5	6	6	6	7
2	25	3	3	4	4	4	5	5	6	6	6	7	7	7
2	60	4	5	5	5	6	6	6	7	7	8	8	8	9
2	100	5	5	6	6	6	7	7	7	8	8	9	9	9
3	1	0	0	0	0	0	1	1	1	1	2	2	2	3
3	2	0	0	0	0	1	1	1	1	2	2	2	3	3
3	4	0	0	1	1	1	1	2	2	2	3	3	3	3
3	6	1	1	1	1	2	2	2	3	3	3	3	4	4
3	10	1	1	2	2	2	3	3	3	3	4	4	4	4
3	20	2	2	3	3	3	3	4	4	4	4	5	5	5
3	30	3	3	3	4	4	4	4	5	5	5	5	6	6
3	80	4	4	5	5	5	5	6	6	6	6	7	7	7
3	150	5	5	5	6	6	6	6	7	7	7	8	8	8
5	2	0	0	0	0	0	0	0	1	1	1	1	1	1
5	3	0	0	0	0	0	0	1	1	1	1	1	1	2
5	5	0	0	0	0	1	1	1	1	1	1	2	2	2
5	8	0	1	1	1	1	1	1	2	2	2	2	2	2
5	16	1	1	2	2	2	2	2	2	3	3	3	3	3
5	30	2	2	2	3	3	3	3	3	3	4	4	4	4
5	60	3	3	3	3	4	4	4	4	4	4	5	5	5
5	100	4	4	4	4	4	5	5	5	5	5	5	6	6
5	200	5	5	5	5	5	6	6	6	6	6	6	7	7
8	4	0	0	0	0	0	0	0	0	0	0	1	1	1
8	6	0	0	0	0	0	0	0	0	1	1	1	1	1
8	8	0	0	0	0	0	0	1	1	1	1	1	1	1
8	12	0	0	1	1	1	1	1	1	1	1	1	2	2
8	20	1	1	1	1	1	1	2	2	2	2	2	2	2
8	50	2	2	2	2	3	3	3	3	3	3	3	3	3
8	80	3	3	3	3	3	3	3	4	4	4	4	4	4
8	150	4	4	4	4	4	4	4	4	5	5	5	5	5
8	300	5	5	5	5	5	5	5	5	6	6	6	6	6

Table C.9: First values of the optimal dyadic unary-stem code parameter sequence $\mathcal{K}^*$, for distributions defined by the uniform prior.

N_s	Σ_s	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}
0	0	0	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	0	1	1	2	2	3	3	4	4	5	5
1	1	0	0	1	1	2	2	3	3	4	4	5	5	6
1	2	0	1	1	2	2	3	3	4	4	5	5	6	6
1	4	1	1	2	2	3	3	4	4	5	5	6	6	7
1	7	2	2	3	3	4	4	5	5	6	6	7	7	8
1	12	2	3	3	4	4	5	5	6	6	7	7	8	8
1	20	3	3	4	4	5	5	6	6	7	7	8	8	9
1	50	4	5	5	6	6	7	7	8	8	9	9	10	10
1	80	5	5	6	6	7	7	8	8	9	9	10	10	11
2	0	0	0	0	0	1	1	1	2	2	2	3	3	3
2	1	0	0	0	1	1	1	2	2	2	3	3	3	4
2	2	0	0	1	1	1	2	2	2	3	3	3	4	4
2	5	1	1	1	2	2	2	3	3	3	4	4	4	5
2	9	1	2	2	2	3	3	3	4	4	4	5	5	5
2	16	2	2	3	3	3	4	4	4	5	5	5	6	6
2	25	3	3	3	4	4	4	5	5	5	6	6	6	7
2	60	4	4	5	5	5	6	6	6	7	7	7	8	8
2	100	5	5	5	6	6	6	7	7	7	8	8	8	9
3	1	0	0	0	0	0	1	1	1	1	2	2	2	2
3	2	0	0	0	0	1	1	1	1	2	2	2	2	3
3	4	0	0	1	1	1	1	2	2	2	2	3	3	3
3	6	1	1	1	1	2	2	2	2	3	3	3	3	4
3	10	1	1	2	2	2	2	3	3	3	3	4	4	4
3	20	2	2	2	3	3	3	3	4	4	4	4	5	5
3	30	3	3	3	3	4	4	4	4	5	5	5	5	6
3	80	4	4	4	5	5	5	5	6	6	6	6	7	7
3	150	5	5	5	5	6	6	6	6	7	7	7	7	8
5	2	0	0	0	0	0	0	0	1	1	1	1	1	1
5	3	0	0	0	0	0	0	1	1	1	1	1	1	2
5	5	0	0	0	0	1	1	1	1	1	1	2	2	2
5	8	0	1	1	1	1	1	1	2	2	2	2	2	2
5	16	1	1	1	2	2	2	2	2	2	3	3	3	3
5	30	2	2	2	2	3	3	3	3	3	3	4	4	4
5	60	3	3	3	3	3	4	4	4	4	4	4	5	5
5	100	4	4	4	4	4	4	5	5	5	5	5	5	6
5	200	5	5	5	5	5	5	6	6	6	6	6	6	7
8	4	0	0	0	0	0	0	0	0	0	0	1	1	1
8	6	0	0	0	0	0	0	0	0	1	1	1	1	1
8	8	0	0	0	0	0	0	1	1	1	1	1	1	1
8	12	0	0	1	1	1	1	1	1	1	1	1	2	2
8	20	1	1	1	1	1	1	2	2	2	2	2	2	2
8	50	2	2	2	2	2	3	3	3	3	3	3	3	3
8	80	3	3	3	3	3	3	3	3	4	4	4	4	4
8	150	4	4	4	4	4	4	4	4	4	5	5	5	5
8	300	5	5	5	5	5	5	5	5	5	6	6	6	6

Table C.10: First values of the optimal dyadic unary-stem code parameter sequence $\mathcal{K}^*$, for distributions defined by the empiric prior $g_e(\xi)$.

N_s	Σ_s	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}
0	0	3	5	6	7	7	8	8	8	8	8	8	9	9
1	0	1	1	2	3	4	5	5	6	6	7	7	7	8
1	1	1	2	3	4	5	5	6	6	7	7	7	8	8
1	2	1	2	3	4	5	5	6	6	7	7	7	8	8
1	4	2	3	4	5	5	6	6	7	7	7	8	8	8
1	7	3	4	5	5	6	6	7	7	7	8	8	8	8
1	12	3	4	5	6	6	7	7	7	8	8	8	8	8
1	20	4	5	6	6	7	7	7	8	8	8	8	8	8
1	50	5	6	6	7	7	7	8	8	8	8	8	8	8
1	80	6	6	7	7	7	8	8	8	8	8	8	8	9
2	0	0	1	1	1	2	2	3	3	4	4	5	5	5
2	1	1	1	1	2	2	3	3	4	4	5	5	5	6
2	2	1	1	1	2	2	3	3	4	4	5	5	6	6
2	5	1	2	2	3	3	4	4	5	5	5	6	6	6
2	9	2	2	3	3	4	4	5	5	6	6	6	7	7
2	16	3	3	4	4	5	5	6	6	6	7	7	7	7
2	25	3	4	4	5	5	6	6	6	7	7	7	7	8
2	60	4	5	5	6	6	6	7	7	7	7	8	8	8
2	100	5	6	6	6	7	7	7	7	8	8	8	8	8
3	1	0	0	1	1	1	1	2	2	2	3	3	3	4
3	2	0	1	1	1	1	2	2	2	3	3	3	4	4
3	4	1	1	1	1	2	2	2	3	3	3	4	4	4
3	6	1	1	1	2	2	2	3	3	3	4	4	4	5
3	10	1	2	2	2	3	3	3	4	4	4	5	5	5
3	20	2	3	3	3	4	4	4	5	5	5	6	6	6
3	30	3	3	4	4	4	5	5	5	6	6	6	6	7
3	80	4	5	5	5	6	6	6	6	7	7	7	7	7
3	150	5	5	6	6	6	6	7	7	7	7	7	8	8
5	2	0	0	0	0	1	1	1	1	1	1	2	2	2
5	3	0	0	0	1	1	1	1	1	1	2	2	2	2
5	5	0	1	1	1	1	1	1	2	2	2	2	2	2
5	8	1	1	1	1	1	2	2	2	2	2	2	3	3
5	16	1	2	2	2	2	2	2	3	3	3	3	3	4
5	30	2	2	2	3	3	3	3	3	4	4	4	4	4
5	60	3	3	3	4	4	4	4	4	5	5	5	5	5
5	100	4	4	4	4	5	5	5	5	5	6	6	6	6
5	200	5	5	5	5	6	6	6	6	6	6	7	7	7
8	4	0	0	0	0	0	0	0	1	1	1	1	1	1
8	6	0	0	0	0	0	1	1	1	1	1	1	1	1
8	8	0	0	0	1	1	1	1	1	1	1	1	1	2
8	12	1	1	1	1	1	1	1	1	1	2	2	2	2
8	20	1	1	1	1	1	2	2	2	2	2	2	2	2
8	50	2	2	2	3	3	3	3	3	3	3	3	3	4
8	80	3	3	3	3	3	3	4	4	4	4	4	4	4
8	150	4	4	4	4	4	4	4	5	5	5	5	5	5
8	300	5	5	5	5	5	5	5	6	6	6	6	6	6

Table C.11: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and one known source sample.

$N_s = 1$, $\Sigma_s = 1$, $H = 2.64$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	51.01	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	51.01	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	51.01	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	2.75	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	1.36	1	2	3	5	9	15	25	43	72	122	207	350	593
ML3	1.43	1	2	3	6	8	16	24	48	64	128	192	384	512
ML2	2.56	1	2	4	8	8	16	32	64	128	128	256	512	1024
BEs	1.64	1	2	2	3	5	7	10	15	22	32	47	69	100
BE3	1.78	1	2	2	3	4	6	8	12	16	24	32	48	64
BE2	1.87	1	2	2	4	4	8	8	16	16	32	32	64	64
Opt	1.29	1	2	3	5	7	12	20	27	46	80	106	183	320
Op3	1.36	1	2	3	4	6	12	16	24	48	64	96	192	256
Op2	1.82	1	1	2	4	4	8	8	16	32	32	64	64	128
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	1.29	1	0	2	1	2	3	3	5	6	8	11	14	18

$N_s = 1$, $\Sigma_s = 2$, $H = 3.29$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	31.01	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	31.01	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	31.01	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	4.42	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	2.19	2	3	5	9	15	25	43	72	122	207	350	593	1004
ML3	2.31	2	3	6	8	16	24	48	64	128	192	384	512	1024
ML2	4.13	2	4	8	8	16	32	64	128	128	256	512	1024	2048
BEs	2.64	2	2	3	5	7	10	15	22	32	47	69	100	147
BE3	2.86	2	2	3	4	6	8	12	16	24	32	48	64	96
BE2	3.01	2	2	4	4	8	8	16	16	32	32	64	64	128
Opt	2.07	2	3	5	7	12	20	27	46	80	106	183	320	421
Op3	2.19	2	3	4	6	12	16	24	48	64	96	192	256	384
Op2	2.93	1	2	4	4	8	8	16	32	32	64	64	128	256
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	2.07	2	1	2	3	3	5	6	8	11	14	19	24	33

Table C.12: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and one known source sample.

$N_s = 1$, $\Sigma_s = 4$, $H = 4.07$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	29.69	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	29.69	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	21.76	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	4.94	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.82	3	5	9	15	25	43	72	122	207	350	593	1004	1700
ML3	1.04	3	6	8	16	24	48	64	128	192	384	512	1024	1536
ML2	4.40	4	8	8	16	32	64	128	128	256	512	1024	2048	2048
BEs	1.65	2	3	5	7	10	15	22	32	47	69	100	147	215
BE3	2.06	2	3	4	6	8	12	16	24	32	48	64	96	128
BE2	2.35	2	4	4	8	8	16	16	32	32	64	64	128	128
Opt	0.61	3	5	7	12	20	27	46	80	106	183	320	421	730
Op3	0.82	3	4	6	12	16	24	48	64	96	192	256	384	768
Op2	2.29	2	4	4	8	16	16	32	32	64	128	128	256	256
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.59	1	2	3	3	5	6	8	11	14	19	25	33	43

$N_s = 1$, $\Sigma_s = 7$, $H = 4.77$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	24.67	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	19.22	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	34.29	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	2.31	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.07	5	9	15	25	43	72	122	207	350	593	1004	1700	2878
ML3	1.47	6	8	16	24	48	64	128	192	384	512	1024	1536	3072
ML2	2.27	4	8	16	32	64	64	128	256	512	1024	1024	2048	4096
BEs	1.23	4	6	8	12	18	26	38	56	81	119	174	254	372
BE3	1.30	4	6	8	12	16	24	32	48	64	96	128	192	256
BE2	2.28	4	4	8	8	16	16	32	32	64	64	128	128	256
Opt	0.71	5	7	12	20	27	46	80	106	183	320	421	730	1277
Op3	1.08	4	6	12	16	24	48	64	96	192	256	384	768	1024
Op2	1.86	4	8	8	16	32	32	64	64	128	256	256	512	512
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.67	3	3	5	6	8	11	15	18	24	33	42	56	73

Table C.13: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and one known source sample.

$N_s = 1$, $\Sigma_s = 15$, $H = 5.80$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	18.07	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	16.10	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	28.80	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	1.93	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.91	11	18	31	52	88	149	253	428	725	1227	2078	3518	5957
ML3	1.20	12	16	32	48	96	128	256	384	768	1024	2048	3072	6144
ML2	1.89	8	16	32	64	128	128	256	512	1024	2048	2048	4096	8192
BEs	1.12	8	11	16	24	35	51	75	109	160	233	341	499	729
BE3	1.12	8	12	16	24	32	48	64	96	128	192	256	384	512
BE2	1.95	8	8	16	16	32	32	64	64	128	128	256	256	512
Opt	0.59	10	13	23	40	53	91	160	210	365	638	839	1458	2550
Op3	0.93	8	12	24	32	48	96	128	192	384	512	768	1536	2048
Op2	1.56	8	16	16	32	64	64	128	128	256	512	512	1024	1024
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.56	6	7	9	11	16	20	27	36	47	63	83	108	143

$N_s = 1$, $\Sigma_s = 30$, $H = 6.76$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	15.87	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	13.27	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	23.68	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	1.67	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.81	21	36	61	103	174	295	499	845	1431	2423	4102	6946	11760
ML3	1.09	24	32	64	96	192	256	512	768	1536	2048	4096	6144	12288
ML2	1.64	16	32	64	128	256	256	512	1024	2048	4096	4096	8192	16384
BEs	0.92	14	21	31	45	66	96	141	206	301	440	643	940	1375
BE3	0.95	16	24	32	48	64	96	128	192	256	384	512	768	1024
BE2	1.64	16	16	32	32	64	64	128	128	256	256	512	512	1024
Opt	0.53	20	26	46	80	105	182	319	419	729	1274	1676	2912	5090
Op3	0.80	16	24	48	64	96	192	256	384	768	1024	1536	3072	4096
Op2	1.35	16	32	32	64	128	128	256	256	512	1024	1024	2048	2048
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.46	1	10	14	18	23	30	41	54	70	93	124	162	214

Table C.14: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and one known source sample.

$N_s = 1$, $\Sigma_s = 75$, $H = 8.06$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	13.04	52	52	52	52	52	52	52	52	52	52	52	52
Go3	14.54	48	48	48	48	48	48	48	48	48	48	48	48
G-R	10.67	64	64	64	64	64	64	64	64	64	64	64	64
E-T	2.58	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.61	52	88	149	253	428	725	1227	2078	3518	5957	10086	17077
ML3	0.61	48	96	128	256	384	768	1024	2048	3072	6144	8192	16384
ML2	2.47	64	128	256	256	512	1024	2048	4096	4096	8192	16384	32768
BEs	0.77	35	51	75	110	160	234	343	501	733	1071	1566	2290
BE3	1.13	32	48	64	96	128	192	256	384	512	768	1024	1536
BE2	1.27	32	64	64	128	128	256	256	512	512	1024	1024	2048
Opt	0.37	45	79	104	181	317	417	726	1271	1672	2908	5084	6684
Op3	0.50	48	64	96	192	256	384	768	1024	1536	3072	4096	6144
Op2	1.24	32	64	64	128	256	256	512	512	1024	2048	2048	4096
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.33	19	31	40	53	70	93	121	161	212	279	369	486

$N_s = 1$, $\Sigma_s = 150$, $H = 9.06$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	11.30	104	104	104	104	104	104	104	104	104	104	104	104
Go3	12.61	96	96	96	96	96	96	96	96	96	96	96	96
G-R	9.28	128	128	128	128	128	128	128	128	128	128	128	128
E-T	2.31	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.55	104	176	298	505	855	1448	2451	4150	7027	11898	20145	34108
ML3	0.55	96	192	256	512	768	1536	2048	4096	6144	12288	16384	32768
ML2	2.22	128	256	512	512	1024	2048	4096	8192	8192	16384	32768	65536
BEs	0.68	70	102	149	218	319	466	682	997	1458	2131	3116	4556
BE3	0.99	64	96	128	192	256	384	512	768	1024	1536	2048	3072
BE2	1.13	64	128	128	256	256	512	512	1024	1024	2048	2048	4096
Opt	0.33	90	158	208	363	635	835	1453	2543	3346	5818	10180	13394
Op3	0.45	96	128	192	384	512	768	1536	2048	3072	6144	8192	12288
Op2	1.10	64	128	128	256	512	512	1024	1024	2048	4096	4096	8192
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.30	39	60	81	105	139	185	242	320	422	558	736	970

Table C.15: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and two known source samples.

$N_s = 2,\ \Sigma_s = 2,\ H = 2.39$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	11.56	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	11.56	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	11.56	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	5.21	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	1.13	1	1	2	2	3	4	6	8	10	14	19	25	34
ML3	1.14	1	1	2	2	3	4	6	8	12	16	24	32	48
ML2	1.37	1	1	2	2	4	4	8	8	8	16	16	32	32
BEs	1.13	1	1	2	2	3	4	5	6	8	10	13	16	21
BE3	1.14	1	1	2	2	3	4	4	6	8	8	12	16	16
BE2	1.30	1	1	2	2	2	4	4	4	8	8	8	16	16
Opt	1.12	1	1	2	2	3	4	5	6	9	11	14	20	25
Op3	1.13	1	1	2	2	3	4	6	6	8	12	16	24	24
Op2	1.30	1	1	2	2	2	4	4	8	8	8	16	16	16
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	1.12	1	1	0	2	2	1	2	4	3	4	4	6	8

$N_s = 2,\ \Sigma_s = 4,\ H = 3.09$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	7.15	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	7.15	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	7.15	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	6.05	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	1.89	2	2	3	4	6	8	10	14	19	25	34	46	62
ML3	1.90	2	2	3	4	6	8	12	16	24	32	48	64	64
ML2	2.53	2	2	4	4	8	8	8	16	16	32	32	64	64
BEs	1.87	2	2	3	4	5	6	8	10	13	16	21	27	34
BE3	1.92	2	2	3	4	4	6	8	8	12	16	16	24	32
BE2	2.34	2	2	2	4	4	4	8	8	8	16	16	16	32
Opt	1.86	2	2	3	4	5	6	9	11	14	20	25	37	45
Op3	1.88	2	2	3	4	6	6	8	12	16	24	24	32	48
Op2	2.34	2	2	2	4	4	8	8	8	16	16	16	32	32
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	1.87	2	2	1	2	4	3	4	4	7	7	8	11	13

Table C.16: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and two known source samples.

$N_s = 2,\ \Sigma_s = 8,\ H = 3.91$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	7.20	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	7.20	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	5.54	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	5.92	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.84	3	4	6	8	10	14	19	25	34	46	62	83	112
ML3	0.88	3	4	6	8	12	16	24	32	48	64	64	96	128
ML2	2.94	4	4	8	8	8	16	16	32	32	64	64	64	128
BEs	0.77	3	4	5	6	8	10	13	16	21	27	34	43	55
BE3	0.93	3	4	4	6	8	8	12	16	16	24	32	48	48
BE2	2.32	2	4	4	4	8	8	8	16	16	16	32	32	32
Opt	0.75	3	4	5	6	9	11	14	20	25	37	45	55	82
Op3	0.81	3	4	6	6	8	12	16	24	24	32	48	64	96
Op2	2.32	2	4	4	8	8	8	16	16	16	32	32	64	64
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.75	1	2	4	3	4	4	7	7	8	11	13	17	19

$N_s = 2,\ \Sigma_s = 14,\ H = 4.63$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	5.99	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	4.26	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	10.02	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	3.24	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.66	5	7	9	12	17	23	31	41	55	75	101	136	183
ML3	1.26	6	8	12	16	24	32	32	48	64	96	128	192	256
ML2	1.88	4	8	8	16	16	32	32	32	64	64	128	128	256
BEs	1.08	4	5	7	9	11	14	18	23	30	38	48	62	79
BE3	1.05	4	6	8	8	12	16	16	24	32	32	48	64	96
BE2	1.79	4	4	8	8	8	16	16	16	32	32	32	64	64
Opt	0.56	5	6	9	11	14	20	25	37	45	55	81	100	149
Op3	0.89	6	6	8	12	12	16	24	32	48	48	64	96	128
Op2	1.76	4	8	8	8	16	16	16	32	32	64	64	64	128
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.56	3	4	5	5	7	9	10	12	16	18	24	27	35

Table C.17: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and two known source samples.

$N_s = 2$, $\Sigma_s = 30$, $H = 5.66$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	4.23	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	3.64	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	8.69	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	2.67	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.63	11	15	20	27	36	49	66	88	119	160	216	290	391
ML3	0.98	12	16	24	32	48	64	64	96	128	192	256	384	512
ML2	1.55	8	16	16	32	32	64	64	64	128	128	256	256	512
BEs	0.68	9	11	14	18	23	30	38	48	62	79	101	129	165
BE3	0.93	8	12	16	16	24	32	32	48	64	64	96	128	192
BE2	1.57	8	8	16	16	16	32	32	32	64	64	64	128	128
Opt	0.48	10	13	19	23	28	41	50	75	91	111	164	201	299
Op3	0.72	12	12	16	24	32	48	48	64	96	128	192	192	256
Op2	1.46	8	16	16	16	32	32	32	64	64	128	128	128	256
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.48	6	7	11	11	15	18	22	26	33	39	49	59	71

$N_s = 2$, $\Sigma_s = 60$, $H = 6.63$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	3.82	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	3.02	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	7.10	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	2.28	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.47	21	28	38	51	69	93	125	168	227	305	411	553	745
ML3	0.89	24	32	48	64	96	128	128	192	256	384	512	768	1024
ML2	1.33	16	32	32	64	64	128	128	128	256	256	512	512	1024
BEs	0.63	17	22	28	36	46	58	75	95	122	155	198	253	323
BE3	0.77	16	24	32	32	48	64	64	96	128	128	192	256	384
BE2	1.29	16	16	32	32	32	64	64	64	128	128	128	256	256
Opt	0.41	20	25	37	45	55	82	100	149	181	221	327	400	596
Op3	0.64	24	24	32	48	48	64	96	128	192	192	256	384	512
Op2	1.25	16	32	32	32	64	64	64	128	128	256	256	256	512
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.41	12	15	20	23	29	35	43	52	63	77	95	116	140

Table C.18: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and two known source samples.

$N_s = 2$, $\Sigma_s = 150$, $H = 7.94$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	3.24	52	52	52	52	52	52	52	52	52	52	52	52
Go3	3.81	48	48	48	48	48	48	48	48	48	48	48	48
G-R	2.79	64	64	64	64	64	64	64	64	64	64	64	64
E-T	2.87	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.47	52	70	95	128	172	232	312	420	566	762	1026	1381
ML3	0.48	48	64	96	128	192	256	384	512	768	1024	1024	1536
ML2	1.44	64	64	128	128	256	256	256	512	512	1024	1024	2048
BEs	0.48	42	54	69	88	112	143	183	234	299	381	487	622
BE3	0.54	48	48	64	96	128	128	192	256	256	384	512	512
BE2	1.25	32	64	64	64	128	128	128	256	256	256	512	512
Opt	0.36	47	57	84	103	153	186	226	333	407	606	737	896
Op3	0.45	48	64	96	96	128	192	256	384	384	512	768	1024
Op2	1.24	32	64	64	128	128	128	256	256	256	512	512	1024
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.35	17	37	44	55	66	81	99	120	147	179	218	266

$N_s = 2$, $\Sigma_s = 300$, $H = 8.93$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	2.86	104	104	104	104	104	104	104	104	104	104	104	104
Go3	3.35	96	96	96	96	96	96	96	96	96	96	96	96
G-R	2.48	128	128	128	128	128	128	128	128	128	128	128	128
E-T	2.57	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.42	104	140	189	254	342	461	621	836	1126	1516	2041	2749
ML3	0.43	96	128	192	256	384	512	768	1024	1536	2048	2048	3072
ML2	1.29	128	128	256	256	512	512	512	1024	1024	2048	2048	4096
BEs	0.43	84	107	137	175	223	285	364	465	594	759	969	1238
BE3	0.48	96	96	128	192	256	256	384	512	512	768	1024	1024
BE2	1.11	64	128	128	128	256	256	256	512	512	512	1024	1024
Opt	0.32	94	114	168	205	305	371	451	666	813	1210	1473	1791
Op3	0.40	96	128	192	192	256	384	512	768	768	1024	1536	2048
Op2	1.10	64	128	128	256	256	256	512	512	512	1024	1024	2048
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.31	34	73	90	109	134	162	197	242	294	358	436	533

Table C.19: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and five known source samples.

$N_s = 5$, $\Sigma_s = 5$, $H = 2.18$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.96	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	1.96	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	1.96	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	8.78	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	0.84	1	1	1	1	2	2	2	2	3	3	4	4	5
ML3	0.84	1	1	1	1	2	2	2	2	3	3	4	4	4
ML2	0.86	1	1	1	1	2	2	2	2	2	4	4	4	4
BEs	0.84	1	1	1	1	2	2	2	2	3	3	3	4	4
BE3	0.84	1	1	1	1	2	2	2	2	3	3	3	4	4
BE2	0.86	1	1	1	1	2	2	2	2	2	2	4	4	4
Opt	0.84	1	1	1	1	2	2	2	2	3	3	3	4	5
Op3	0.84	1	1	1	1	2	2	2	2	3	3	3	4	4
Op2	0.86	1	1	1	1	2	2	2	2	2	2	4	4	4
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	0.84	1	1	1	1	0	2	2	2	2	1	3	3	2

$N_s = 5$, $\Sigma_s = 10$, $H = 2.91$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	2.04	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	2.04	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	2.04	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	8.49	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	1.58	2	2	2	3	3	3	4	4	5	6	6	7	8
ML3	1.58	2	2	2	3	3	3	4	4	4	6	6	8	8
ML2	1.73	2	2	2	2	2	4	4	4	4	4	8	8	8
BEs	1.60	2	2	2	2	3	3	3	4	4	5	5	6	7
BE3	1.60	2	2	2	2	3	3	3	4	4	4	6	6	6
BE2	1.73	2	2	2	2	2	4	4	4	4	4	4	8	8
Opt	1.58	2	2	2	3	3	3	4	4	5	5	6	7	7
Op3	1.58	2	2	2	3	3	3	4	4	4	6	6	6	8
Op2	1.73	2	2	2	2	2	4	4	4	4	4	4	8	8
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	1.58	2	2	2	1	3	3	2	4	4	3	5	4	5

Table C.20: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and five known source samples.

$N_s = 5$, $\Sigma_s = 20$, $H = 3.75$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.88	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	1.88	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	2.31	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	7.61	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.95	3	4	4	5	5	6	7	8	9	10	12	13	15
ML3	0.99	3	4	4	4	6	6	6	8	8	12	12	12	16
ML2	2.19	4	4	4	4	4	8	8	8	8	8	8	16	16
BEs	0.80	3	3	4	4	5	5	6	7	8	9	10	11	12
BE3	0.83	3	3	4	4	4	6	6	6	8	8	8	12	12
BE2	2.18	4	4	4	4	4	4	8	8	8	8	8	8	16
Opt	0.79	3	3	4	5	5	6	6	7	9	10	11	12	13
Op3	0.83	3	3	4	4	6	6	6	8	8	8	12	12	12
Op2	2.18	4	4	4	4	4	4	8	8	8	8	8	8	16
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.79	1	3	2	4	3	5	4	6	5	6	7	8	9

$N_s = 5$, $\Sigma_s = 35$, $H = 4.48$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.53	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	1.13	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	3.55	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	4.64	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.60	5	6	7	8	9	10	11	13	15	17	19	22	25
ML3	0.81	6	6	6	8	8	12	12	12	16	16	16	24	24
ML2	1.73	4	4	8	8	8	8	8	16	16	16	16	16	16
BEs	0.63	5	5	6	7	8	9	10	11	13	14	16	18	20
BE3	1.00	4	6	6	6	8	8	8	12	12	12	16	16	16
BE2	1.72	4	4	8	8	8	8	8	8	16	16	16	16	16
Opt	0.57	5	6	6	7	9	10	11	12	13	15	18	20	23
Op3	0.81	6	6	6	8	8	8	12	12	12	16	16	24	24
Op2	1.72	4	4	8	8	8	8	8	8	16	16	16	16	16
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.57	3	4	6	5	6	7	8	9	10	11	11	13	15

Table C.21: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and five known source samples.

$N_s = 5$, $\Sigma_s = 75$, $H = 5.53$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.04	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	0.94	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	3.25	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	3.79	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.50	11	12	14	16	18	21	23	27	30	35	39	45	51
ML3	0.69	12	12	16	16	16	24	24	24	32	32	48	48	48
ML2	1.63	8	16	16	16	16	16	16	32	32	32	32	32	64
BEs	0.50	10	11	13	14	16	18	20	23	26	29	33	37	41
BE3	0.91	8	12	12	12	16	16	16	24	24	24	32	32	32
BE2	1.50	8	8	16	16	16	16	16	16	32	32	32	32	32
Opt	0.48	11	12	13	15	18	20	22	25	28	34	38	42	47
Op3	0.63	12	12	12	16	16	24	24	24	32	32	32	48	48
Op2	1.50	8	8	16	16	16	16	16	16	32	32	32	32	32
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.48	5	10	11	11	14	14	16	18	19	22	25	28	30

$N_s = 5$, $\Sigma_s = 150$, $H = 6.50$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	0.95	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	0.80	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	2.59	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	3.22	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.41	21	24	27	31	35	40	46	52	59	68	77	88	100
ML3	0.56	24	24	24	32	32	48	48	48	64	64	64	96	96
ML2	1.40	16	32	32	32	32	32	32	64	64	64	64	64	128
BEs	0.45	19	22	24	28	31	35	39	44	50	56	63	71	80
BE3	0.73	16	24	24	24	32	32	32	48	48	48	64	64	64
BE2	1.24	16	16	32	32	32	32	32	32	64	64	64	64	64
Opt	0.40	21	23	26	29	36	40	44	49	55	61	75	83	93
Op3	0.56	24	24	24	32	32	32	48	48	48	64	64	96	96
Op2	1.24	16	16	32	32	32	32	32	32	64	64	64	64	64
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.40	11	19	20	23	26	28	32	36	39	43	49	54	60

Table C.22: Comparisons of unary-stem codes assuming Dirichlet (1/2) prior and five known source samples.

$N_s = 5$, $\Sigma_s = 375$, $H = 7.81$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	0.86	52	52	52	52	52	52	52	52	52	52	52	52
Go3	1.10	48	48	48	48	48	48	48	48	48	48	48	48
G-R	1.03	64	64	64	64	64	64	64	64	64	64	64	64
E-T	3.54	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.42	52	60	68	77	88	100	114	130	148	168	192	218
ML3	0.51	48	64	64	64	96	96	128	128	128	192	192	192
ML2	0.96	64	64	64	64	64	128	128	128	128	128	256	256
BEs	0.41	48	54	61	68	77	86	97	110	123	139	156	176
BE3	0.49	48	48	64	64	64	96	96	96	128	128	128	192
BE2	0.95	64	64	64	64	64	64	128	128	128	128	128	128
Opt	0.39	50	55	69	76	85	95	105	117	145	161	179	200
Op3	0.48	48	48	64	64	96	96	96	128	128	128	192	192
Op2	0.95	64	64	64	64	64	64	128	128	128	128	128	128
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.39	14	45	48	56	60	67	76	83	93	104	115	128

$N_s = 5$, $\Sigma_s = 750$, $H = 8.80$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	0.76	104	104	104	104	104	104	104	104	104	104	104	104
Go3	0.96	96	96	96	96	96	96	96	96	96	96	96	96
G-R	0.93	128	128	128	128	128	128	128	128	128	128	128	128
E-T	3.15	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.37	104	119	135	154	175	200	227	259	295	336	382	435
ML3	0.45	96	128	128	128	192	192	256	256	256	384	384	384
ML2	0.86	128	128	128	128	128	256	256	256	256	256	512	512
BEs	0.36	95	107	120	136	153	172	194	218	246	277	311	351
BE3	0.43	96	96	128	128	128	192	192	192	256	256	256	384
BE2	0.86	128	128	128	128	128	128	256	256	256	256	256	256
Opt	0.35	99	111	137	153	170	189	210	234	289	322	358	399
Op3	0.42	96	96	128	128	192	192	192	256	256	256	384	384
Op2	0.86	128	128	128	128	128	128	256	256	256	256	256	256
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.35	29	88	97	109	122	135	150	166	186	207	230	256

Table C.23: Comparisons of unary-stem codes assuming uniform prior, and one known source sample.

$N_s = 1$, $\Sigma_s = 1$, $H = 2.49$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	20.69	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	20.69	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	20.69	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	4.08	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	2.21	1	2	3	5	9	15	25	43	72	122	207	350	593
ML3	2.32	1	2	3	6	8	16	24	48	64	128	192	384	512
ML2	3.85	1	2	4	8	8	16	32	64	128	128	256	512	1024
BEs	1.27	1	1	2	2	3	4	6	8	10	14	19	25	34
BE3	1.27	1	1	2	2	3	4	6	8	12	16	24	32	48
BE2	1.42	1	1	2	2	4	4	8	8	8	16	16	32	32
Opt	1.12	1	1	2	3	3	5	7	10	13	21	26	41	52
Op3	1.14	1	1	2	3	3	6	6	12	12	24	24	48	48
Op2	1.41	1	1	2	2	4	4	8	8	16	16	32	32	64
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	1.12	1	1	0	2	1	3	2	3	3	5	7	8	10

$N_s = 1$, $\Sigma_s = 2$, $H = 2.97$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	11.66	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	11.66	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	11.66	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	6.82	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	3.70	2	3	5	9	15	25	43	72	122	207	350	593	1004
ML3	3.89	2	3	6	8	16	24	48	64	128	192	384	512	1024
ML2	6.45	2	4	8	8	16	32	64	128	128	256	512	1024	2048
BEs	2.13	1	2	2	3	4	6	8	10	14	19	25	34	46
BE3	2.12	1	2	2	3	4	6	8	12	16	24	32	48	64
BE2	2.37	1	2	2	4	4	8	8	8	16	16	32	32	64
Opt	1.88	1	2	3	3	5	7	10	13	21	26	41	52	82
Op3	1.91	1	2	3	3	6	6	12	12	24	24	48	48	96
Op2	2.36	1	2	2	4	4	8	8	16	16	32	32	64	64
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	1.88	1	0	2	1	3	2	3	3	5	7	8	10	13

Table C.24: Comparisons of unary-stem codes assuming uniform prior, and one known source sample.

$N_s = 1$, $\Sigma_s = 4$, $H = 3.62$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	9.06	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	9.06	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	9.83	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	8.60	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	2.19	3	5	9	15	25	43	72	122	207	350	593	1004	1700
ML3	2.58	3	6	8	16	24	48	64	128	192	384	512	1024	1536
ML2	7.83	4	8	8	16	32	64	128	128	256	512	1024	2048	2048
BEs	1.09	2	3	4	5	7	9	12	17	23	31	41	55	75
BE3	1.06	2	3	4	6	8	12	16	24	32	32	48	64	96
BE2	2.07	2	2	4	4	8	8	16	16	16	32	32	64	64
Opt	0.96	2	3	5	6	10	12	20	25	40	50	80	101	160
Op3	1.06	2	3	4	6	8	12	16	24	32	48	64	96	128
Op2	2.06	2	2	4	4	8	8	16	16	32	32	64	64	128
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.96	2	1	2	3	4	4	6	8	8	12	15	18	22

$N_s = 1$, $\Sigma_s = 7$, $H = 4.25$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	7.35	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	6.55	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	10.45	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	3.86	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	2.85	5	9	15	25	43	72	122	207	350	593	1004	1700	2878
ML3	3.65	6	8	16	24	48	64	128	192	384	512	1024	1536	3072
ML2	3.83	4	8	16	32	64	64	128	256	512	1024	1024	2048	4096
BEs	0.96	3	4	6	8	10	14	19	25	34	46	62	83	112
BE3	0.95	3	4	6	8	12	16	24	32	48	64	64	96	128
BE2	2.00	4	4	8	8	8	16	16	32	32	64	64	64	128
Opt	0.69	3	5	6	10	13	20	25	40	51	80	101	161	203
Op3	0.93	3	6	6	12	12	24	24	48	48	96	96	192	192
Op2	1.95	4	4	8	8	16	16	32	32	64	64	128	128	256
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.69	1	2	3	4	4	6	7	10	12	14	19	24	30

Table C.25: Comparisons of unary-stem codes assuming uniform prior, and one known source sample.

$N_s = 1$, $\Sigma_s = 15$, $H = 5.21$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	5.46	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	5.39	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	8.26	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	3.32	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	2.75	11	18	31	52	88	149	253	428	725	1227	2078	3518	5957
ML3	3.22	12	16	32	48	96	128	256	384	768	1024	2048	3072	6144
ML2	3.30	8	16	32	64	128	128	256	512	1024	2048	2048	4096	8192
BEs	0.79	6	8	11	15	20	27	36	49	66	88	119	160	216
BE3	0.77	6	8	12	16	24	32	48	64	64	96	128	192	256
BE2	1.86	8	8	8	16	16	32	32	64	64	64	128	128	256
Opt	0.58	6	10	13	20	25	40	51	80	101	161	203	322	406
Op3	0.77	6	8	12	16	24	32	48	64	96	128	192	256	384
Op2	1.72	8	8	16	16	32	32	64	64	128	128	256	256	512
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.58	2	4	6	7	10	12	15	18	24	29	37	46	60

$N_s = 1$, $\Sigma_s = 30$, $H = 6.14$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	4.69	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	4.62	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	6.64	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	3.01	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	2.48	21	36	61	103	174	295	499	845	1431	2423	4102	6946	11760
ML3	3.03	24	32	64	96	192	256	512	768	1536	2048	4096	6144	12288
ML2	2.99	16	32	64	128	256	256	512	1024	2048	4096	4096	8192	16384
BEs	0.72	11	15	20	27	36	49	66	89	120	161	217	292	393
BE3	0.61	12	16	24	32	48	64	96	128	128	192	256	384	512
BE2	1.85	8	16	16	32	32	64	64	64	128	128	256	256	512
Opt	0.47	12	20	25	40	50	80	101	160	202	321	404	642	810
Op3	0.61	12	16	24	32	48	64	96	128	192	256	384	512	768
Op2	1.59	16	16	32	32	64	64	128	128	256	256	512	512	1024
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.47	4	9	11	14	17	23	29	36	46	57	72	90	114

Table C.26: Comparisons of unary-stem codes assuming uniform prior, and one known source sample.

$N_s = 1$, $\Sigma_s = 75$, $H = 7.42$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	3.94	52	52	52	52	52	52	52	52	52	52	52	52
Go3	3.97	48	48	48	48	48	48	48	48	48	48	48	48
G-R	5.38	64	64	64	64	64	64	64	64	64	64	64	64
E-T	5.27	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	2.02	52	88	149	253	428	725	1227	2078	3518	5957	10086	17077
ML3	1.84	48	96	128	256	384	768	1024	2048	3072	6144	8192	16384
ML2	5.18	64	128	256	256	512	1024	2048	4096	4096	8192	16384	32768
BEs	0.62	27	36	49	66	88	119	160	216	290	391	526	709
BE3	0.85	24	32	48	64	96	128	192	256	384	512	512	768
BE2	1.06	32	32	64	64	128	128	128	256	256	512	512	1024
Opt	0.47	28	44	55	86	108	169	212	334	420	663	833	1317
Op3	0.69	32	48	64	96	128	192	256	384	512	768	1024	1536
Op2	1.05	32	32	64	64	128	128	256	256	512	512	1024	1024
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.43	4	20	27	33	43	52	66	84	106	134	167	211

$N_s = 1$, $\Sigma_s = 150$, $H = 8.41$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	3.47	104	104	104	104	104	104	104	104	104	104	104	104
Go3	3.48	96	96	96	96	96	96	96	96	96	96	96	96
G-R	4.78	128	128	128	128	128	128	128	128	128	128	128	128
E-T	4.72	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	1.82	104	176	298	505	855	1448	2451	4150	7027	11898	20145	34108
ML3	1.66	96	192	256	512	768	1536	2048	4096	6144	12288	16384	32768
ML2	4.64	128	256	512	512	1024	2048	4096	8192	8192	16384	32768	65536
BEs	0.56	53	71	96	129	174	234	315	424	571	769	1036	1395
BE3	0.72	48	64	96	128	192	256	384	512	768	1024	1024	1536
BE2	0.94	64	64	128	128	256	256	256	512	512	1024	1024	2048
Opt	0.41	55	86	108	170	213	336	422	665	836	1320	1660	2626
Op3	0.62	48	96	96	192	192	384	384	768	768	1536	1536	3072
Op2	0.93	64	64	128	128	256	256	512	512	1024	1024	2048	2048
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.38	8	41	53	66	83	104	131	166	209	263	331	417

Table C.27: Comparisons of unary-stem codes assuming uniform prior, and two known source samples.

$N_s = 2,\ \Sigma_s = 2,\ H = 2.33$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	7.46	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	7.46	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	7.46	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	6.12	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	1.22	1	1	2	2	3	4	6	8	10	14	19	25	34
ML3	1.22	1	1	2	2	3	4	6	8	12	16	24	32	48
ML2	1.51	1	1	2	2	4	4	8	8	8	16	16	32	32
BEs	1.20	1	1	2	2	2	3	4	5	6	7	9	11	13
BE3	1.21	1	1	2	2	2	3	4	4	6	6	8	12	12
BE2	1.30	1	1	2	2	2	2	4	4	4	8	8	8	16
Opt	1.16	1	1	2	2	3	3	5	6	7	9	11	13	19
Op3	1.16	1	1	2	2	3	3	4	6	6	8	12	12	16
Op2	1.29	1	1	2	2	2	4	4	4	8	8	8	16	16
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	1.16	1	1	0	2	2	1	3	2	3	4	5	6	7

$N_s = 2,\ \Sigma_s = 4,\ H = 2.92$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	4.89	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	4.89	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	4.89	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	7.64	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	2.51	2	2	3	4	6	8	10	14	19	25	34	46	62
ML3	2.53	2	2	3	4	6	8	12	16	24	32	48	64	64
ML2	3.34	2	2	4	4	8	8	8	16	16	32	32	64	64
BEs	2.46	2	2	2	3	4	5	6	7	9	11	13	16	20
BE3	2.48	2	2	2	3	4	4	6	6	8	12	12	16	16
BE2	2.74	2	2	2	2	4	4	4	8	8	8	16	16	16
Opt	2.36	2	2	3	3	5	6	7	9	11	13	19	22	27
Op3	2.36	2	2	3	3	4	6	6	8	12	12	16	24	24
Op2	2.71	2	2	2	4	4	4	8	8	8	16	16	16	32
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	2.36	2	2	1	3	2	3	4	5	6	7	7	9	10

Table C.28: Comparisons of unary-stem codes assuming uniform prior, and two known source samples.

$N_s = 2$, $\Sigma_s = 8$, $H = 3.66$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	3.71	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	3.71	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	4.95	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	8.09	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.41	3	4	6	8	10	14	19	25	34	46	62	83	112
ML3	1.46	3	4	6	8	12	16	24	32	48	64	64	96	128
ML2	4.50	4	4	8	8	8	16	16	32	32	64	64	64	128
BEs	1.21	2	3	4	5	6	7	9	11	13	16	20	25	30
BE3	1.29	2	3	4	4	6	6	8	12	12	16	16	24	32
BE2	2.26	2	2	4	4	4	8	8	8	16	16	16	16	32
Opt	0.82	3	3	5	6	7	9	11	13	19	22	27	38	45
Op3	0.85	3	3	4	6	6	8	12	12	16	24	24	32	48
Op2	2.14	2	4	4	4	8	8	8	16	16	16	32	32	32
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.82	1	3	2	4	3	4	5	5	8	8	9	13	13

$N_s = 2$, $\Sigma_s = 14$, $H = 4.34$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	3.01	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	2.78	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	5.06	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	4.03	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.25	5	7	9	12	17	23	31	41	55	75	101	136	183
ML3	2.46	6	8	12	16	24	32	32	48	64	96	128	192	256
ML2	2.52	4	8	8	16	16	32	32	32	64	64	128	128	256
BEs	0.84	4	5	6	7	9	11	14	17	21	26	32	39	48
BE3	1.13	4	4	6	8	8	12	12	16	24	24	32	32	48
BE2	1.55	4	4	4	8	8	8	16	16	16	32	32	32	32
Opt	0.82	5	6	7	9	11	13	19	22	27	38	45	53	76
Op3	0.92	4	6	6	8	12	12	16	24	24	32	48	48	64
Op2	1.48	4	4	8	8	8	16	16	16	32	32	32	64	64
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.83	4	3	4	5	5	8	8	9	13	13	17	21	23

Table C.29: Comparisons of unary-stem codes assuming uniform prior, and two known source samples.

$N_s = 2$, $\Sigma_s = 30$, $H = 5.34$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	2.23	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	2.29	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	4.16	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	3.30	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	1.38	11	15	20	27	36	49	66	88	119	160	216	290	391
ML3	2.02	12	16	24	32	48	64	64	96	128	192	256	384	512
ML2	2.07	8	16	16	32	32	64	64	64	128	128	256	256	512
BEs	0.81	8	9	11	14	17	21	26	32	39	48	59	73	90
BE3	0.95	8	8	12	16	16	24	24	32	48	48	64	64	96
BE2	1.29	8	8	8	16	16	16	32	32	32	64	64	64	64
Opt	0.61	9	11	13	18	22	26	37	44	52	74	88	105	149
Op3	0.78	8	12	12	16	24	24	32	48	48	64	96	96	128
Op2	1.24	8	8	16	16	16	32	32	32	64	64	64	128	128
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.60	7	7	9	11	12	14	18	21	25	30	36	42	49

$N_s = 2$, $\Sigma_s = 60$, $H = 6.29$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.92	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	1.99	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	3.30	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	2.87	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	1.05	21	28	38	51	69	93	125	168	227	305	411	553	745
ML3	1.88	24	32	48	64	96	128	128	192	256	384	512	768	1024
ML2	1.85	16	32	32	64	64	128	128	128	256	256	512	512	1024
BEs	0.62	14	18	22	27	33	41	50	62	76	94	115	142	175
BE3	0.81	16	16	24	24	32	48	48	64	64	96	128	128	192
BE2	1.07	16	16	16	32	32	32	64	64	64	64	128	128	128
Opt	0.54	15	21	24	29	41	48	57	80	95	113	159	189	224
Op3	0.70	16	24	24	32	48	48	64	96	96	128	192	192	256
Op2	1.06	16	16	32	32	32	64	64	64	128	128	128	256	256
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.50	1	12	15	17	20	25	28	34	40	49	57	68	81

Table C.30: Comparisons of unary-stem codes assuming uniform prior, and two known source samples.

$N_s = 2$, $\Sigma_s = 150$, $H = 7.59$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	1.69	52	52	52	52	52	52	52	52	52	52	52	52
Go3	1.73	48	48	48	48	48	48	48	48	48	48	48	48
G-R	2.65	64	64	64	64	64	64	64	64	64	64	64	64
E-T	4.24	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	1.02	52	70	95	128	172	232	312	420	566	762	1026	1381
ML3	0.88	48	64	96	128	192	256	384	512	768	1024	1024	1536
ML2	2.55	64	64	128	128	256	256	256	512	512	1024	1024	2048
BEs	0.50	35	43	53	66	81	99	122	151	185	228	281	346
BE3	0.63	32	48	48	64	96	96	128	128	192	256	256	384
BE2	1.06	32	32	64	64	64	128	128	128	128	256	256	256
Opt	0.35	41	48	57	80	95	113	159	189	224	316	375	445
Op3	0.54	48	48	64	96	96	128	192	192	256	384	384	512
Op2	1.06	32	32	64	64	64	128	128	128	256	256	256	512
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.35	23	33	40	47	56	66	80	93	112	133	157	188

$N_s = 2$, $\Sigma_s = 300$, $H = 8.58$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	1.49	104	104	104	104	104	104	104	104	104	104	104	104
Go3	1.52	96	96	96	96	96	96	96	96	96	96	96	96
G-R	2.37	128	128	128	128	128	128	128	128	128	128	128	128
E-T	3.78	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.92	104	140	189	254	342	461	621	836	1126	1516	2041	2749
ML3	0.79	96	128	192	256	384	512	768	1024	1536	2048	2048	3072
ML2	2.30	128	128	256	256	512	512	512	1024	1024	2048	2048	4096
BEs	0.43	70	86	106	130	160	197	243	299	368	453	558	687
BE3	0.55	64	96	96	128	192	192	256	256	384	512	512	768
BE2	0.92	64	64	128	128	128	256	256	256	256	512	512	512
Opt	0.31	81	96	114	160	190	226	318	377	448	630	749	889
Op3	0.48	96	96	128	192	192	256	384	384	512	768	768	1024
Op2	0.92	64	64	128	128	128	256	256	256	512	512	512	1024
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.31	47	65	79	93	111	132	156	187	221	264	314	373

Table C.31: Comparisons of unary-stem codes assuming uniform prior, and five known source samples.

$N_s = 5$, $\Sigma_s = 5$, $H = 2.16$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.63	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	1.63	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	1.63	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	9.10	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	0.80	1	1	1	1	2	2	2	2	3	3	4	4	5
ML3	0.80	1	1	1	1	2	2	2	2	3	3	4	4	4
ML2	0.81	1	1	1	1	2	2	2	2	2	4	4	4	4
BEs	0.80	1	1	1	1	2	2	2	2	2	3	3	3	4
BE3	0.80	1	1	1	1	2	2	2	2	2	3	3	3	4
BE2	0.80	1	1	1	1	2	2	2	2	2	2	2	4	4
Opt	0.80	1	1	1	1	2	2	2	2	3	3	3	3	4
Op3	0.80	1	1	1	1	2	2	2	2	3	3	3	3	4
Op2	0.80	1	1	1	1	2	2	2	2	2	2	4	4	4
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	0.80	1	1	1	1	0	2	2	2	2	1	3	3	3

$N_s = 5$, $\Sigma_s = 10$, $H = 2.84$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	2.11	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	2.11	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	2.11	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	9.18	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	1.86	2	2	2	3	3	3	4	4	5	6	6	7	8
ML3	1.86	2	2	2	3	3	3	4	4	4	6	6	8	8
ML2	1.95	2	2	2	2	2	4	4	4	4	4	8	8	8
BEs	1.83	2	2	2	2	3	3	3	4	4	5	5	6	6
BE3	1.83	2	2	2	2	3	3	3	4	4	4	6	6	6
BE2	1.94	2	2	2	2	2	2	4	4	4	4	4	4	8
Opt	1.83	2	2	2	2	3	3	3	4	4	5	5	6	6
Op3	1.83	2	2	2	2	3	3	3	4	4	4	6	6	6
Op2	1.94	2	2	2	2	2	2	4	4	4	4	4	4	8
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	1.83	2	2	2	2	1	3	3	2	4	4	3	5	4

Table C.32: Comparisons of unary-stem codes assuming uniform prior, and five known source samples.

$N_s = 5$, $\Sigma_s = 20$, $H = 3.65$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.40	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	1.40	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	2.82	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	8.52	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.15	3	4	4	5	5	6	7	8	9	10	12	13	15
ML3	1.17	3	4	4	4	6	6	6	8	8	12	12	12	16
ML2	2.79	4	4	4	4	4	8	8	8	8	8	8	16	16
BEs	0.79	3	3	3	4	4	5	5	6	7	7	8	9	10
BE3	0.80	3	3	3	4	4	4	6	6	6	8	8	8	12
BE2	2.40	2	4	4	4	4	4	4	8	8	8	8	8	8
Opt	0.78	3	3	3	4	5	5	6	6	7	8	9	10	11
Op3	0.80	3	3	3	4	4	6	6	6	6	8	8	12	12
Op2	2.40	2	4	4	4	4	4	4	8	8	8	8	8	8
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.78	1	3	3	2	4	3	5	4	6	5	6	8	7

$N_s = 5$, $\Sigma_s = 35$, $H = 4.36$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.11	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	1.21	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	2.50	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	4.91	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.73	5	6	7	8	9	10	11	13	15	17	19	22	25
ML3	1.08	6	6	6	8	8	12	12	12	16	16	16	24	24
ML2	1.61	4	4	8	8	8	8	8	16	16	16	16	16	16
BEs	0.59	5	5	6	6	7	8	9	10	11	12	14	15	17
BE3	0.89	4	6	6	6	8	8	8	8	12	12	12	16	16
BE2	1.49	4	4	4	8	8	8	8	8	8	8	16	16	16
Opt	0.59	5	5	6	6	7	9	10	11	12	13	14	18	19
Op3	0.89	4	6	6	6	8	8	8	12	12	12	16	16	16
Op2	1.49	4	4	4	8	8	8	8	8	8	16	16	16	16
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.59	3	5	4	6	5	6	7	8	9	10	11	12	12

Table C.33: Comparisons of unary-stem codes assuming uniform prior, and five known source samples.

$N_s = 5$, $\Sigma_s = 75$, $H = 5.39$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	0.83	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	0.95	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	2.22	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	3.98	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.65	11	12	14	16	18	21	23	27	30	35	39	45	51
ML3	0.95	12	12	16	16	16	24	24	24	32	32	48	48	48
ML2	1.82	8	16	16	16	16	16	16	32	32	32	32	32	64
BEs	0.54	9	10	11	13	14	16	18	20	22	24	27	30	34
BE3	0.76	8	12	12	12	16	16	16	24	24	24	32	32	32
BE2	1.29	8	8	8	16	16	16	16	16	16	16	32	32	32
Opt	0.47	10	11	12	13	15	18	20	22	24	27	29	36	40
Op3	0.76	8	12	12	12	16	16	16	24	24	24	32	32	32
Op2	1.29	8	8	8	16	16	16	16	16	16	32	32	32	32
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.47	6	9	10	11	11	14	14	16	18	20	21	25	25

$N_s = 5$, $\Sigma_s = 150$, $H = 6.36$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	0.72	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	0.85	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	1.73	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	3.39	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.52	21	24	27	31	35	40	46	52	59	68	77	88	100
ML3	0.77	24	24	24	32	32	48	48	48	64	64	64	96	96
ML2	1.59	16	32	32	32	32	32	32	64	64	64	64	64	128
BEs	0.44	18	20	22	25	28	31	34	38	43	48	53	59	66
BE3	0.74	16	16	24	24	24	32	32	32	48	48	48	64	64
BE2	1.04	16	16	16	32	32	32	32	32	32	32	64	64	64
Opt	0.41	19	21	23	26	29	35	39	43	47	52	57	70	77
Op3	0.63	16	24	24	24	32	32	32	48	48	48	64	64	64
Op2	1.04	16	16	16	32	32	32	32	32	32	64	64	64	64
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.41	13	17	19	20	23	26	29	31	35	39	42	46	52

Table C.34: Comparisons of unary-stem codes assuming uniform prior, and five known source samples.

$N_s = 5$, $\Sigma_s = 375$, $H = 7.66$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	0.68	52	52	52	52	52	52	52	52	52	52	52	52
Go3	0.74	48	48	48	48	48	48	48	48	48	48	48	48
G-R	1.34	64	64	64	64	64	64	64	64	64	64	64	64
E-T	4.06	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.55	52	60	68	77	88	100	114	130	148	168	192	218
ML3	0.60	48	64	64	64	96	96	128	128	128	192	192	192
ML2	1.33	64	64	64	64	64	128	128	128	128	128	256	256
BEs	0.39	44	49	55	61	68	76	85	94	105	117	131	146
BE3	0.44	48	48	48	64	64	64	96	96	96	128	128	128
BE2	1.20	32	64	64	64	64	64	64	64	128	128	128	128
Opt	0.37	46	51	56	69	76	84	93	102	113	138	152	168
Op3	0.44	48	48	64	64	64	96	96	96	128	128	128	192
Op2	1.20	32	64	64	64	64	64	64	128	128	128	128	128
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.37	18	41	46	50	56	60	69	74	82	91	101	112

$N_s = 5$, $\Sigma_s = 750$, $H = 8.66$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	0.60	104	104	104	104	104	104	104	104	104	104	104	104
Go3	0.65	96	96	96	96	96	96	96	96	96	96	96	96
G-R	1.21	128	128	128	128	128	128	128	128	128	128	128	128
E-T	3.61	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.49	104	119	135	154	175	200	227	259	295	336	382	435
ML3	0.54	96	128	128	128	192	192	256	256	256	384	384	384
ML2	1.19	128	128	128	128	128	256	256	256	256	256	512	512
BEs	0.35	87	97	108	121	135	150	168	187	209	233	260	290
BE3	0.39	96	96	96	128	128	128	192	192	192	256	256	256
BE2	1.06	64	128	128	128	128	128	128	128	256	256	256	256
Opt	0.33	92	101	112	137	151	167	184	204	225	275	303	335
Op3	0.39	96	96	96	128	128	192	192	192	192	256	256	384
Op2	1.06	64	128	128	128	128	128	128	256	256	256	256	256
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.33	36	83	90	100	112	121	136	148	165	181	200	222

Table C.35: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and one known source sample.

$N_s = 1$, $\Sigma_s = 1$, $H = 4.06$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	161.58	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	161.58	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	161.58	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	6.28	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	6.04	1	2	3	5	9	15	25	43	72	122	207	350	593
ML3	6.04	1	2	3	6	8	16	24	48	64	128	192	384	512
ML2	6.32	1	2	4	8	8	16	32	64	128	128	256	512	1024
BEs	1.48	2	3	5	8	13	21	32	48	69	94	121	150	179
BE3	1.83	2	3	4	8	12	16	32	48	64	96	128	128	192
BE2	2.33	2	4	4	8	16	16	32	64	64	128	128	128	128
Opt	0.80	3	5	7	13	24	43	56	90	111	158	183	209	235
Op3	0.98	3	6	12	24	48	64	96	128	192	192	192	256	256
Op2	2.25	2	4	8	16	32	32	64	64	128	128	128	256	256
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.72	1	2	3	3	4	6	9	12	17	23	32	45	58

$N_s = 1$, $\Sigma_s = 2$, $H = 4.38$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	78.21	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	78.21	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	78.21	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	2.65	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	2.35	2	3	5	9	15	25	43	72	122	207	350	593	1004
ML3	2.35	2	3	6	8	16	24	48	64	128	192	384	512	1024
ML2	2.70	2	4	8	8	16	32	64	128	128	256	512	1024	2048
BEs	1.07	3	4	7	11	18	28	43	62	86	112	140	169	199
BE3	1.13	3	4	6	12	16	24	48	64	96	128	128	192	192
BE2	2.67	2	4	8	8	16	32	32	64	64	128	128	128	256
Opt	0.56	3	6	11	22	40	53	87	108	154	179	205	231	283
Op3	0.61	3	6	12	24	48	64	96	128	192	192	192	256	256
Op2	2.55	2	4	8	16	32	32	64	64	128	128	128	256	256
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.54	1	2	3	3	4	6	9	12	16	24	33	46	59

Table C.36: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and one known source sample.

$N_s = 1$, $\Sigma_s = 4$, $H = 4.93$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	63.10	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	63.10	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	41.70	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	2.07	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.65	3	5	9	15	25	43	72	122	207	350	593	1004	1700
ML3	1.64	3	6	8	16	24	48	64	128	192	384	512	1024	1536
ML2	2.15	4	8	8	16	32	64	128	128	256	512	1024	2048	2048
BEs	1.40	4	6	10	16	25	39	57	80	106	134	163	192	221
BE3	1.69	4	6	8	16	24	32	48	64	96	128	128	192	192
BE2	2.11	4	8	8	16	32	32	64	64	128	128	128	256	256
Opt	0.84	5	7	13	24	43	56	90	111	158	183	209	235	286
Op3	1.17	6	12	24	48	64	96	128	192	192	192	256	256	256
Op2	1.94	4	8	16	32	32	64	64	128	128	128	256	256	256
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.69	3	3	4	6	9	12	17	23	32	45	59	76	98

$N_s = 1$, $\Sigma_s = 7$, $H = 5.53$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	43.59	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	33.01	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	60.44	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	2.58	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.22	5	9	15	25	43	72	122	207	350	593	1004	1700	2878
ML3	1.22	6	8	16	24	48	64	128	192	384	512	1024	1536	3072
ML2	2.48	4	8	16	32	64	64	128	256	512	1024	1024	2048	4096
BEs	1.48	5	8	13	21	33	49	70	95	122	151	180	209	238
BE3	1.26	6	8	16	24	32	48	64	96	128	128	192	192	256
BE2	2.54	4	8	16	16	32	64	64	128	128	128	128	256	256
Opt	0.53	6	12	23	41	54	88	109	156	181	206	232	284	309
Op3	0.57	6	12	24	48	64	96	128	192	192	192	256	256	256
Op2	2.32	8	16	32	32	64	64	128	128	128	256	256	256	256
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.50	2	3	6	7	11	15	20	28	40	53	69	90	111

Table C.37: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and one known source sample.

$N_s = 1,\ \Sigma_s = 15,\ H = 6.46$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	22.83	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	19.68	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	38.29	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	2.32	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.89	11	18	31	52	88	149	253	428	725	1227	2078	3518	5957
ML3	1.02	12	16	32	48	96	128	256	384	768	1024	2048	3072	6144
ML2	2.20	8	16	32	64	128	128	256	512	1024	2048	2048	4096	8192
BEs	1.11	10	17	26	40	59	82	108	136	165	194	224	253	281
BE3	0.97	12	16	24	48	64	96	128	128	192	192	256	256	256
BE2	2.09	8	16	32	32	64	64	128	128	128	256	256	256	256
Opt	0.47	12	23	42	55	89	110	157	182	208	233	285	310	334
Op3	0.52	12	24	48	64	96	128	192	192	192	256	256	256	384
Op2	1.95	16	32	32	64	64	128	128	128	256	256	256	256	256
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.43	4	7	11	14	21	29	41	53	72	90	114	136	162

$N_s = 1,\ \Sigma_s = 30,\ H = 7.26$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	13.03	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	10.03	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	21.52	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	2.03	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.77	21	36	61	103	174	295	499	845	1431	2423	4102	6946	11760
ML3	0.84	24	32	64	96	192	256	512	768	1536	2048	4096	6144	12288
ML2	1.93	16	32	64	128	256	256	512	1024	2048	4096	4096	8192	16384
BEs	0.84	20	31	47	67	91	118	147	176	205	234	263	291	318
BE3	1.39	16	32	48	64	96	128	128	192	192	256	256	256	384
BE2	1.79	16	32	32	64	64	128	128	128	256	256	256	256	256
Opt	0.40	24	43	57	91	112	159	184	209	235	287	312	336	359
Op3	0.47	24	48	64	96	128	192	192	192	256	256	256	384	384
Op2	1.59	16	32	64	64	128	128	128	256	256	256	256	256	256
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.37	8	15	22	30	41	55	74	92	116	138	164	190	215

Table C.38: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and one known source sample.

$N_s = 1$, $\Sigma_s = 75$, $H = 8.19$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	4.30	52	52	52	52	52	52	52	52	52	52	52	52
Go3	5.22	48	48	48	48	48	48	48	48	48	48	48	48
G-R	3.01	64	64	64	64	64	64	64	64	64	64	64	64
E-T	2.32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.75	52	88	149	253	428	725	1227	2078	3518	5957	10086	17077
ML3	0.83	48	96	128	256	384	768	1024	2048	3072	6144	8192	16384
ML2	2.08	64	128	256	256	512	1024	2048	4096	4096	8192	16384	32768
BEs	0.61	44	63	87	114	142	171	200	230	259	287	314	340
BE3	0.53	48	64	96	128	128	192	192	256	256	256	384	384
BE2	1.53	32	64	64	128	128	128	256	256	256	256	256	256
Opt	0.35	50	84	104	151	176	202	228	280	305	330	353	376
Op3	0.46	48	96	96	128	192	192	256	256	256	384	384	384
Op2	1.17	64	64	128	128	128	256	256	256	256	256	256	256
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.35	13	36	48	63	82	104	127	152	176	201	229	255

$N_s = 1$, $\Sigma_s = 150$, $H = 8.79$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	1.36	104	104	104	104	104	104	104	104	104	104	104	104
Go3	1.61	96	96	96	96	96	96	96	96	96	96	96	96
G-R	1.67	128	128	128	128	128	128	128	128	128	128	128	128
E-T	3.10	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	1.12	104	176	298	505	855	1448	2451	4150	7027	11898	20145	34108
ML3	1.16	96	192	256	512	768	1536	2048	4096	6144	12288	16384	32768
ML2	2.96	128	256	512	512	1024	2048	4096	8192	8192	16384	32768	65536
BEs	0.47	76	101	129	158	187	216	245	274	302	328	353	377
BE3	0.79	64	96	128	128	192	192	256	256	256	384	384	384
BE2	1.00	64	128	128	128	128	256	256	256	256	256	256	256
Opt	0.30	90	111	158	183	208	234	286	311	335	359	381	402
Op3	0.43	96	128	192	192	192	256	256	256	384	384	384	384
Op2	0.98	64	128	128	128	256	256	256	256	256	256	256	512
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.30	38	69	88	109	134	158	184	210	236	263	288	313

Table C.39: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and two known source samples.

$N_s = 2$, $\Sigma_s = 2$, $H = 3.21$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	36.99	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	36.99	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	36.99	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	5.76	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	6.16	1	1	2	2	3	4	6	8	10	14	19	25	34
ML3	6.16	1	1	2	2	3	4	6	8	12	16	24	32	48
ML2	6.31	1	1	2	2	4	4	8	8	8	16	16	32	32
BEs	1.34	2	2	3	4	5	7	9	12	16	22	29	38	50
BE3	1.40	2	2	3	4	4	6	8	12	16	24	32	32	48
BE2	1.85	2	2	2	4	4	8	8	8	16	16	32	32	32
Opt	1.31	2	2	3	5	6	10	12	19	24	38	47	57	83
Op3	1.34	2	2	3	4	6	8	12	16	24	32	48	48	64
Op2	1.84	2	2	2	4	4	8	8	16	16	32	32	64	64
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	1.31	2	2	1	2	3	4	5	5	6	9	12	13	17

$N_s = 2$, $\Sigma_s = 4$, $H = 3.62$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	15.45	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	15.45	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	15.45	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	3.75	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	1.73	2	2	3	4	6	8	10	14	19	25	34	46	62
ML3	1.73	2	2	3	4	6	8	12	16	24	32	48	64	64
ML2	2.01	2	2	4	4	8	8	8	16	16	32	32	64	64
BEs	1.05	2	3	4	5	7	9	12	16	22	29	38	50	64
BE3	1.17	2	3	4	4	6	8	12	16	24	32	32	48	64
BE2	2.01	2	2	4	4	8	8	8	16	16	32	32	32	64
Opt	0.99	2	3	5	6	10	12	19	24	38	47	57	83	99
Op3	1.05	2	3	4	6	8	12	16	24	32	48	48	64	96
Op2	1.99	2	2	4	4	8	8	16	16	32	32	64	64	64
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.99	2	1	2	3	4	5	5	6	9	12	13	17	22

Table C.40: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and two known source samples.

$N_s = 2,\ \Sigma_s = 8,\ H = 4.31$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	16.14	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	16.14	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	9.44	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	3.67	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	1.10	3	4	6	8	10	14	19	25	34	46	62	83	112
ML3	1.08	3	4	6	8	12	16	24	32	48	64	64	96	128
ML2	1.79	4	4	8	8	8	16	16	32	32	64	64	64	128
BEs	1.10	3	4	6	8	10	14	19	25	33	43	56	71	89
BE3	1.09	3	4	6	8	12	16	24	24	32	48	64	64	96
BE2	1.79	4	4	8	8	8	16	16	32	32	32	64	64	64
Opt	0.82	3	5	6	10	13	20	25	39	48	58	84	100	117
Op3	1.04	3	6	6	12	12	24	24	48	48	64	96	96	128
Op2	1.74	4	4	8	8	16	16	32	32	64	64	64	128	128
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.82	1	2	3	4	4	6	7	10	12	15	19	23	30

$N_s = 2,\ \Sigma_s = 14,\ H = 5.02$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	13.11	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	9.02	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	20.41	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	2.90	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.87	5	7	9	12	17	23	31	41	55	75	101	136	183
ML3	0.69	6	8	12	16	24	32	32	48	64	96	128	192	256
ML2	1.96	4	8	8	16	16	32	32	32	64	64	128	128	256
BEs	0.89	5	7	9	12	16	22	29	38	50	64	81	100	120
BE3	0.73	6	8	12	16	16	24	32	48	64	64	96	96	128
BE2	1.98	4	8	8	16	16	16	32	32	64	64	64	128	128
Opt	0.58	6	10	12	19	24	38	47	57	83	99	116	153	174
Op3	0.68	6	8	12	16	24	32	48	48	64	96	96	128	192
Op2	1.95	4	8	8	16	16	32	32	64	64	64	128	128	128
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.55	2	5	5	6	9	12	13	17	22	28	33	43	53

Table C.41: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and two known source samples.

$N_s = 2$, $\Sigma_s = 30$, $H = 6.06$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	8.52	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	7.05	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	16.58	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	2.56	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.62	11	15	20	27	36	49	66	88	119	160	216	290	391
ML3	0.59	12	16	24	32	48	64	64	96	128	192	256	384	512
ML2	1.74	8	16	16	32	32	64	64	64	128	128	256	256	512
BEs	0.66	11	14	19	26	34	44	57	73	91	111	132	155	178
BE3	0.62	12	16	24	32	32	48	64	64	96	128	128	192	192
BE2	1.75	8	16	16	32	32	32	64	64	64	128	128	128	128
Opt	0.48	12	19	24	38	47	57	83	99	116	153	174	195	217
Op3	0.59	12	16	24	32	48	48	64	96	96	128	192	192	192
Op2	1.73	16	16	32	32	32	64	64	64	128	128	128	128	256
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.46	4	9	11	14	18	23	28	36	43	55	67	82	97

$N_s = 2$, $\Sigma_s = 60$, $H = 7.00$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	6.25	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	4.54	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	11.57	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	2.29	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.55	21	28	38	51	69	93	125	168	227	305	411	553	745
ML3	0.52	24	32	48	64	96	128	128	192	256	384	512	768	1024
ML2	1.45	16	32	32	64	64	128	128	128	256	256	512	512	1024
BEs	0.57	21	28	37	48	62	78	97	117	139	162	186	210	235
BE3	0.62	24	32	32	48	64	96	96	128	128	192	192	192	256
BE2	1.43	16	32	32	64	64	64	128	128	128	128	256	256	256
Opt	0.42	24	37	46	56	82	98	115	152	173	194	216	239	284
Op3	0.51	24	32	48	64	96	96	128	128	192	192	192	256	256
Op2	1.43	16	32	32	64	64	64	128	128	128	128	256	256	256
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.39	8	18	23	28	36	43	55	67	82	97	115	134	154

Table C.42: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and two known source samples.

$N_s = 2$, $\Sigma_s = 150$, $H = 8.13$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	2.84	52	52	52	52	52	52	52	52	52	52	52	52
Go3	3.58	48	48	48	48	48	48	48	48	48	48	48	48
G-R	1.86	64	64	64	64	64	64	64	64	64	64	64	64
E-T	2.50	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.47	52	70	95	128	172	232	312	420	566	762	1026	1381
ML3	0.61	48	64	96	128	192	256	384	512	768	1024	1024	1536
ML2	1.07	64	64	128	128	256	256	256	512	512	1024	1024	2048
BEs	0.50	49	63	80	98	119	141	164	188	212	237	262	287
BE3	0.52	48	64	96	96	128	128	192	192	192	256	256	256
BE2	0.93	64	64	64	128	128	128	128	256	256	256	256	256
Opt	0.38	53	78	93	110	148	168	189	211	234	279	302	324
Op3	0.52	48	64	96	96	128	192	192	192	256	256	256	384
Op2	0.93	64	64	64	128	128	128	128	256	256	256	256	256
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.38	11	41	52	63	76	92	108	128	146	168	189	212

$N_s = 2$, $\Sigma_s = 300$, $H = 8.85$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	1.08	104	104	104	104	104	104	104	104	104	104	104	104
Go3	1.36	96	96	96	96	96	96	96	96	96	96	96	96
G-R	1.14	128	128	128	128	128	128	128	128	128	128	128	128
E-T	2.94	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.56	104	140	189	254	342	461	621	836	1126	1516	2041	2749
ML3	0.61	96	128	192	256	384	512	768	1024	1536	2048	2048	3072
ML2	1.38	128	128	256	256	512	512	512	1024	1024	2048	2048	4096
BEs	0.39	89	109	130	152	176	200	225	250	275	299	323	346
BE3	0.45	96	96	128	128	192	192	192	256	256	256	256	384
BE2	1.10	64	128	128	128	128	256	256	256	256	256	256	256
Opt	0.33	97	114	151	172	193	215	238	283	306	328	350	371
Op3	0.41	96	128	128	192	192	192	256	256	256	384	384	384
Op2	0.99	128	128	128	128	256	256	256	256	256	256	256	256
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.33	31	80	95	114	131	152	172	193	216	239	262	284

Table C.43: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and five known source samples.

$N_s = 5$, $\Sigma_s = 5$, $H = 2.59$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	6.83	1	1	1	1	1	1	1	1	1	1	1	1	1
Go3	6.83	1	1	1	1	1	1	1	1	1	1	1	1	1
G-R	6.83	1	1	1	1	1	1	1	1	1	1	1	1	1
E-T	7.85	1	2	4	8	16	32	64	128	256	512	1024	2048	4096
MxL	3.88	1	1	1	1	2	2	2	2	3	3	4	4	5
ML3	3.88	1	1	1	1	2	2	2	2	3	3	4	4	4
ML2	3.91	1	1	1	1	2	2	2	2	2	4	4	4	4
BEs	2.91	1	2	2	2	2	2	3	3	3	4	4	5	6
BE3	2.91	1	2	2	2	2	2	3	3	3	4	4	4	6
BE2	2.94	1	2	2	2	2	2	2	2	4	4	4	4	4
Opt	2.90	1	2	2	2	2	3	3	3	4	5	5	6	6
Op3	2.90	1	2	2	2	2	3	3	3	4	4	6	6	6
Op2	2.94	1	2	2	2	2	2	2	4	4	4	4	4	4
		E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
Huf	2.90	1	0	2	2	2	2	1	3	3	2	4	3	5

$N_s = 5$, $\Sigma_s = 10$, $H = 3.11$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.95	2	2	2	2	2	2	2	2	2	2	2	2	2
Go3	1.95	2	2	2	2	2	2	2	2	2	2	2	2	2
G-R	1.95	2	2	2	2	2	2	2	2	2	2	2	2	2
E-T	7.15	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192
MxL	1.05	2	2	2	3	3	3	4	4	5	6	6	7	8
ML3	1.05	2	2	2	3	3	3	4	4	4	6	6	8	8
ML2	1.33	2	2	2	2	2	4	4	4	4	4	8	8	8
BEs	1.05	2	2	2	3	3	3	4	4	5	6	6	7	8
BE3	1.05	2	2	2	3	3	3	4	4	4	6	6	8	8
BE2	1.33	2	2	2	2	2	4	4	4	4	4	8	8	8
Opt	1.03	2	2	3	3	3	4	5	5	6	6	7	9	10
Op3	1.03	2	2	3	3	3	4	4	6	6	6	8	8	12
Op2	1.31	2	2	2	2	4	4	4	4	4	4	8	8	8
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	1.03	2	2	1	3	3	2	4	3	5	4	6	5	6

Table C.44: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and five known source samples.

$N_s = 5,\ \Sigma_s = 20,\ H = 3.88$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	2.76	3	3	3	3	3	3	3	3	3	3	3	3	3
Go3	2.76	3	3	3	3	3	3	3	3	3	3	3	3	3
G-R	1.95	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	6.65	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.90	3	4	4	5	5	6	7	8	9	10	12	13	15
ML3	0.95	3	4	4	4	6	6	6	8	8	12	12	12	16
ML2	1.66	4	4	4	4	4	8	8	8	8	8	8	16	16
BEs	0.90	3	4	4	5	5	6	7	8	9	10	12	13	15
BE3	0.95	3	4	4	4	6	6	6	8	8	12	12	12	16
BE2	1.66	4	4	4	4	4	8	8	8	8	8	8	16	16
Opt	0.89	3	4	5	5	6	6	7	9	10	11	13	14	18
Op3	0.95	3	4	4	6	6	6	8	8	12	12	12	16	16
Op2	1.66	4	4	4	4	4	8	8	8	8	8	16	16	16
		E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}
Huf	0.89	1	2	4	3	5	4	6	5	6	7	8	9	9

$N_s = 5,\ \Sigma_s = 35,\ H = 4.62$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	2.33	5	5	5	5	5	5	5	5	5	5	5	5	5
Go3	1.34	6	6	6	6	6	6	6	6	6	6	6	6	6
G-R	5.12	4	4	4	4	4	4	4	4	4	4	4	4	4
E-T	4.44	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
MxL	0.64	5	6	7	8	9	10	11	13	15	17	19	22	25
ML3	0.71	6	6	6	8	8	12	12	12	16	16	16	24	24
ML2	2.02	4	4	8	8	8	8	8	16	16	16	16	16	16
BEs	0.65	5	6	7	8	9	10	11	13	15	17	19	22	25
BE3	0.71	6	6	6	8	8	12	12	12	16	16	16	24	24
BE2	2.02	4	4	8	8	8	8	8	16	16	16	16	16	16
Opt	0.63	6	6	7	9	10	11	12	14	18	20	22	25	28
Op3	0.71	6	6	6	8	8	12	12	12	16	16	24	24	24
Op2	1.90	4	8	8	8	8	8	16	16	16	16	16	32	32
		E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}
Huf	0.63	2	6	5	6	7	8	9	10	10	12	14	16	18

Table C.45: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and five known source samples.

$N_s = 5$, $\Sigma_s = 75$, $H = 5.68$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.58	11	11	11	11	11	11	11	11	11	11	11	11	11
Go3	1.21	12	12	12	12	12	12	12	12	12	12	12	12	12
G-R	4.80	8	8	8	8	8	8	8	8	8	8	8	8	8
E-T	3.68	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768
MxL	0.52	11	12	14	16	18	21	23	27	30	35	39	45	51
ML3	0.58	12	12	16	16	16	24	24	24	32	32	48	48	48
ML2	1.59	8	16	16	16	16	16	16	32	32	32	32	32	64
BEs	0.52	11	12	14	16	18	21	23	27	30	35	39	45	51
BE3	0.58	12	12	16	16	16	24	24	24	32	32	48	48	48
BE2	1.59	8	16	16	16	16	16	16	32	32	32	32	32	64
Opt	0.50	11	13	14	18	20	23	25	29	36	40	45	51	57
Op3	0.58	12	12	16	16	24	24	24	32	32	48	48	48	64
Op2	1.58	8	16	16	16	16	16	32	32	32	32	32	64	64
		E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}
Huf	0.50	5	9	12	12	14	16	17	21	21	26	28	32	35

$N_s = 5$, $\Sigma_s = 150$, $H = 6.66$ bits/symbol														
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}
Gol	1.50	21	21	21	21	21	21	21	21	21	21	21	21	21
Go3	0.99	24	24	24	24	24	24	24	24	24	24	24	24	24
G-R	3.89	16	16	16	16	16	16	16	16	16	16	16	16	16
E-T	3.13	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536
MxL	0.46	21	24	27	31	35	40	46	52	59	68	77	88	100
ML3	0.51	24	24	24	32	32	48	48	48	64	64	64	96	96
ML2	1.35	16	32	32	32	32	32	32	64	64	64	64	64	128
BEs	0.46	21	24	27	31	35	40	46	52	59	67	75	84	95
BE3	0.51	24	24	24	32	32	48	48	48	64	64	64	96	96
BE2	1.35	16	32	32	32	32	32	32	64	64	64	64	64	64
Opt	0.42	23	25	28	36	40	45	50	56	70	78	87	96	106
Op3	0.50	24	24	32	32	48	48	48	64	64	64	96	96	96
Op2	1.34	16	32	32	32	32	32	64	64	64	64	64	64	128
		E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.42	9	21	22	24	28	32	35	40	44	49	56	62	69

Table C.46: Comparisons of unary-stem codes assuming the empiric prior $g_e(\xi)$, and five known source samples.

$N_s = 5$, $\Sigma_s = 375$, $H = 7.94$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	1.14	52	52	52	52	52	52	52	52	52	52	52	52
Go3	1.55	48	48	48	48	48	48	48	48	48	48	48	48
G-R	0.85	64	64	64	64	64	64	64	64	64	64	64	64
E-T	3.17	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072
MxL	0.42	52	60	68	77	88	100	114	130	148	168	192	218
ML3	0.54	48	64	64	64	96	96	128	128	128	192	192	192
ML2	0.74	64	64	64	64	64	128	128	128	128	128	256	256
BEs	0.42	52	59	67	75	85	95	106	118	130	144	158	173
BE3	0.54	48	64	64	64	96	96	96	128	128	128	128	192
BE2	0.74	64	64	64	64	64	128	128	128	128	128	128	128
Opt	0.40	54	60	74	83	92	101	112	123	146	159	173	187
Op3	0.53	48	64	64	96	96	96	96	128	128	128	192	192
Op2	0.74	64	64	64	64	64	128	128	128	128	128	128	128
		E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}
Huf	0.40	10	48	53	59	67	74	82	92	101	111	122	135

$N_s = 5$, $\Sigma_s = 750$, $H = 8.83$ bits/symbol													
Code	Δ (%)	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}
Gol	0.68	104	104	104	104	104	104	104	104	104	104	104	104
Go3	0.91	96	96	96	96	96	96	96	96	96	96	96	96
G-R	0.76	128	128	128	128	128	128	128	128	128	128	128	128
E-T	3.14	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144
MxL	0.37	104	119	135	154	175	200	227	259	295	336	382	435
ML3	0.46	96	128	128	128	192	192	256	256	256	384	384	384
ML2	0.75	128	128	128	128	128	256	256	256	256	256	512	512
BEs	0.37	99	111	123	136	149	164	179	195	211	228	249	266
BE3	0.46	96	96	128	128	128	192	192	192	192	256	256	256
BE2	0.73	128	128	128	128	128	128	128	256	256	256	256	256
Opt	0.36	103	113	137	149	162	176	191	206	221	238	271	289
Op3	0.45	96	128	128	128	192	192	192	192	192	256	256	256
Op2	0.73	128	128	128	128	128	128	128	256	256	256	256	256
		E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}
Huf	0.35	25	93	102	112	123	136	148	162	175	189	204	220

Bibliography

[1] D.A. Huffman, "A method for construction of minimum redundancy codes," *Proc. IRE,* vol. 40, pp. 1098–1101, 1952.

[2] S.W. Golomb, "Run-length encodings," *IEEE Trans. Inform. Theory,* vol. 12, pp. 388–401, July 1966.

[3] M. Abramowitz and I.A. Stegun (eds.), *Handbook of Mathematical Functions*, Dover Publications, Inc., New York, 1974.

[4] P. Elias, "Universal codeword sets and representations of integers," *IEEE Trans. Inform. Theory,* vol. 21, pp. 194–203, March 1975.

[5] R.G. Gallager and D.C. Van Voorhis, "Optimal source codes for geometrically distributed integer alphabets," *IEEE Trans. Inform. Theory,* vol. 21, pp. 228–230, March 1975.

[6] Tom M. Apostol, *Introduction to Analytic Number Theory,* Springer-Verlag, New York, 1976.

[7] J. Teuhola, "A compression method for clustered bit-vectors," *Inform. Proc. Letters,* vol. 7, pp. 308–311, Oct. 1978.

[8] G.G. Langdon Jr. and J. Rissanen, "Compression of black-white images with arithmetic coding," *IEEE Trans. on Communications,* vol. 29(6), pp. 858–149, June 1981.

[9] A. Papoulis, *Probability, Random Variables, and Stochastic Processes,* 2nd. ed., McGraw-Hill Pub. Co., New York, 1984.

[10] S. Todd, G.G. Langdon Jr., and J. Rissanen, "Parameter reduction and context selection for compression of gray-scale images," *IBM. J. Res. Develop.,* vol. 29, pp. 188–193, March 1985.

[11] S.L. Moshier, *Methods and Programs for Mathematical Functions,* Prentice-Hall, 1989.

[12] S.A. Martucci, "Reversible compression of HDTV images using median adaptive prediction and arithmetic coding," *Proc. IEEE Int. Symp. on Circuits and Systems*, vol. 2, pp. 1310–1313, May 1990.

[13] T.M. Cover, and J.A. Thomas, *Elements of Information Theory,* John-Wiley & Sons, 1991.

[14] W.B. Pennebaker and J.L. Mitchell, *JPEG: Still Image Data Compression Standard*, Von Nostrand Reinhold, New York, 1992.

[15] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in C, The Art of Scientific Computing,* 2nd. ed., Cambridge University Press, 1992.

[16] N. Merhav, G. Seroussi, and M.J. Weinberger, "Modeling and low complexity adaptive coding for image prediction residuals," *Proc. IEEE Int. Conf. on Image Processing,* vol. 2, pp. 353–356, Sept. 1996.

[17] G. Seroussi and M.J. Weinberger, "Efficient sequential parameter estimation for Golomb-Rice codes, with application to image compression," *Hewlett Packard Laboratories Report,* HPL-1996-15, Palo Alto, CA, Feb. 1996.

[18] X. Wu, "Efficient lossless compression of continuous-tone images via context selection and quantization," *IEEE Trans. Image Processing,* vol. 6, pp. 656–664, May 1997.

[19] C.H. Papadimitriou and K. Steiglitz *Combinatorial Optimization : Algorithms and Complexity,* Dover Publications, 1998.

[20] L.A. Wolsey and G.L. Nemhauser *Integer and Combinatorial Optimization,* John Wiley & Sons, 1999.

[21] J. Wen and J.D. Villasenor, "Structured prefix codes for quantized low-shape-parameter generalized Gaussian sources," *IEEE Trans. Inform. Theory,* vol. 45, pp. 1307–1314, May 1999.

[22] N, Merhav, G. Seroussi, and M.J. Weinberger, "Optimal prefix codes for sources with two-sided geometric distributions," *IEEE Trans. Information Theory,* vol. 46, pp. 12–135, Jan. 2000.

[23] N, Merhav, G. Seroussi, and M.J. Weinberger, "Coding of sources with two-sided geometric distributions and unknown parameters," *IEEE Trans. Information Theory,* vol. 46, pp. 229–236, Jan. 2000.

[24] S. Chan and M.J. Golin, "A dynamic programming algorithm for constructing optimal '1'-ended binary prefix-free codes," *IEEE Trans. Information Theory,* vol. 46, pp. 1637–1644, July 2000.

[25] M.J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS," *IEEE Trans. Image Processing,* vol. 9, pp. 1309–1324, Aug. 2000.

[26] K. Sayood, *Introduction to Data Compression,* 2nd ed., Morgan Kaufmann Pubs., San Francisco, CA, 2000.

[27] C.D. Giurcaneanu and I. Tabus, "Low-complexity transform coding with integer-to-integer transforms," *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing,* vol. 4, pp. 2601–2604, May 2001.

[28] M. Hans and R.W. Schafer, "Lossless compression of digital audio," *IEEE Signal Processing Magazine,* vol. 18, pp. 21–32, July 2001.

[29] M. Golin, "A combinatorial approach to Golomb forests," *Theoretical Computer Science,* vol. 263, pp. 283–304, July 2001.

[30] T. Wiegand, G.J. Sullivan, G. Bjøntegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Trans. Circuits and Systems for Video Technology,* vol. 13, pp. 560–576, July 2003

[31] S. Xue, Y. Xu, and B. Oelman, “Hybrid Golomb codes for a group of quantised GG sources,” *IEE Proc. – Vis. Image Signal Processing,* vol. 150, pp. 256–260, Aug. 2003

[32] K. Sayood, ed., *Lossless Compression Handbook,* Academic Press, San Diego, CA, 2003.

[33] P. Fenwick, “Universal Codes,” Chapter 3 in *Lossless Compression Handbook* (K. Sayood, ed.), Academic Press, San Diego, CA, 2003.

[34] A. Said, “Comparative analysis of arithmetic coding computational complexity,” *Hewlett Packard Laboratories Report,* HPL–2004-75, Palo Alto, CA, April 2004.

[35] A. Said, “Introduction to arithmetic coding theory and practice,” *Hewlett Packard Laboratories Report,* HPL–2004-76, Palo Alto, CA, April 2004.

[36] A. Said, “Resilient tree codes for fast adaptive coding,” *Hewlett Packard Laboratories Report,* HPL–2004–106, Palo Alto, CA, June 2004.

[37] M.H.M. Costa and H.S. Malvar, “Efficient run-length encoding of binary sources with unknown statistics,” *Proc. IEEE Data Compression Conference,* March 2004.

[38] A. Said, “On the inadequacy of Golomb-Rice codes for adaptive coding,” *Proc. IEEE Data Compression Conference,* March 2005.