**Sequential Decoding for Anytime Communication**

by

Hari Palaiyanur

B.S. (Cornell University) 2004

A thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Engineering - Electrical Engineering and Computer Sciences

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Anant Sahai, Chair
Professor Kannan Ramchandran

Spring 2006

The thesis of Hari Palaiyanur is approved.

Chair Date

Date

University of California, Berkeley

Spring 2006

Sequential Decoding for Anytime Communication



by

Hari Palaiyanur

## Abstract

Sequential Decoding for Anytime Communication

by

Hari Palaiyanur

Master of Science in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Anant Sahai, Chair

This thesis shows the applicability of sequential decoding to the problem of creating practical codes for anytime communication. Anytime information theory is the study of the fundamental limits on reliability with delay in information theory.

In the problem of control under communication constraints, anytime information theory plays a crucial role. For a linear, unstable scalar plant with finite disturbance, the anytime capacity of the channel with feedback gives precisely which moments of the plant can be bounded away from infinity. The anytime capacity of a channel with feedback is not known except in special instances. We propose a scheme that makes use of the anytime capacity of the channel without feedback. A random tree encoder is used to communicate an appropriately quantized version of the state to the decoder. The decoder runs the stack algorithm to get a delay universal estimate of the state at current and past times. Using results of Jelinek, we show that the control problem is served just as well by a sequential decoder with bounded complexity as by a maximum-likelihood decoder. The stack algorithm has a parameter called a 'bias' that allows for a tradeoff between reliability and computation. The tradeoff is investigated with simulations and some analytical results.

The major contributions of the thesis are for the problem of delay-universal lossless source coding, where we use a sequential random binning encoder with a stack algorithm

decoder to show how to achieve the same error exponent with delay as a maximum-a-posteriori decoder. This scheme is not good compared to known schemes for point-to-point source coding, but it easily extends to the case when side information is available only at the decoder. Since the best schemes for lossless source coding have their complexity mostly in the encoder, they do not translate well to source coding with side information. We prove theorems about reliability and computation analogous to Jelinek's [1] for channel coding with a sequential decoder, essentially providing the source coding counterpart. We appeal to a result of Arikan and Merhav [2] to show that the stack decoder is as 'computationally efficient' as any sequential decoder can be. The strategy is finally simulated to verify its practicality.

Professor Anant Sahai
Thesis Committee Chair

To my parents, Ravi and Sumathi

# Contents

# List of Figures

# List of Tables

## Acknowledgements

I thank my advisor Anant Sahai very much for his inspiration, patience and encouragement. The work presented in this thesis was motivated by a desire to describe and simulate a practical anytime decoder for anytime communication problems. I also thank Cheng Chang and Dr. Stark Draper for conversations about different anytime communication problems.

In the past two years, I have had the love and support of friends and family members. I especially thank my parents for their wisdom and guidance. Among my classmates, I have have received plentiful help and friendship from Sameer Vermani, Anand Sarwate, Bobak Nazer and many others. Financially, I am very grateful to have been supported by the Vodafone US Foundation on fellowship for two years, allowing me to focus on research.

# Chapter 1

# Introduction

This thesis is divided into two main parts, where sequential decoding is applied to the problems of anytime channel coding and anytime source coding for discrete memoryless channels and discrete memoryless sources respectively. In the introduction, we describe the main areas of emphasis of this work and detail our contributions to the subject.

## 1.1 Anytime communication

Anytime communication was coined by Anant Sahai [3] and refers to the concept that delay is a fundamental price that can and, in most cases, must be paid to gain reliability in communication problems. Different applications will have differing requirements on latency. For example, voice applications like cellular telephony have a maximum latency around 100 milliseconds, while deep space applications have latency requirements that are many orders of magnitude larger. For point-to-point (P-P) source or channel block codes, these latency requirements put stringent conditions on the length of the code used. In general, the longer the length of the code we are allowed to use, the more reliable we can make it. Alternatively, different applications may have differing reliability requirements, as measured in probability of error. Providing the same reliability for a shorter delay can dramatically

enhance the performance of many applications, particularly interactive ones like gaming and video conferencing.

From the early days of information theory, the idea of using long block lengths to increase reliability has been used and studied in terms of error exponents. The probability of error is generally exponentially decaying in block length or decoding delay, and the rate of decay is referred to as the error exponent. However, there has been one metric of performance for reliable communication over a noisy channel. The capacity of the channel divides achievable and unachievable rates of communication. When the application solely calls for P-P communication without reference to latency, this is a perfectly acceptable performance criterion. Many applications, like control over a communication link, do have latency constraints that make reliability with latency the important performance criterion. Roughly speaking, 'anytime capacity' is the parametrization of this capacity with a desired reliability.

An 'anytime encoder' must allow the decoder to select a target delay without knowledge beforehand. Hence, a block encoder is not an anytime decoder. An 'anytime decoder' is a decoder that gives (exponentially) decaying probability of error with decoding delay, therefore allowing the application to choose how and when to use the decoded information in real time. In Chapter 2, we give precise definitions and state a few results of Sahai and Mitter [4] that make clear the relationship of anytime capacity with the problem of control over a noisy channel.

## 1.2 Control over a noisy channel

One of the two main contributions of this thesis is to the problem of controlling an unstable system over a noisy channel[1]. Figure 1.1 shows the diagram of an unstable plant in a feedback loop with an observer/encoder, noisy channel and controller/decoder. The scalar plant is described in Equation 1.1 and has a state $X(t)$ that evolves in discrete time as a linear system with unstable eigenvalue $\lambda$. At each time, a random but bounded

[1]Much of section 1.2 and Chapter 2 is taken from our publication at Allerton 2005 [5], written jointly with my advisor Anant Sahai.

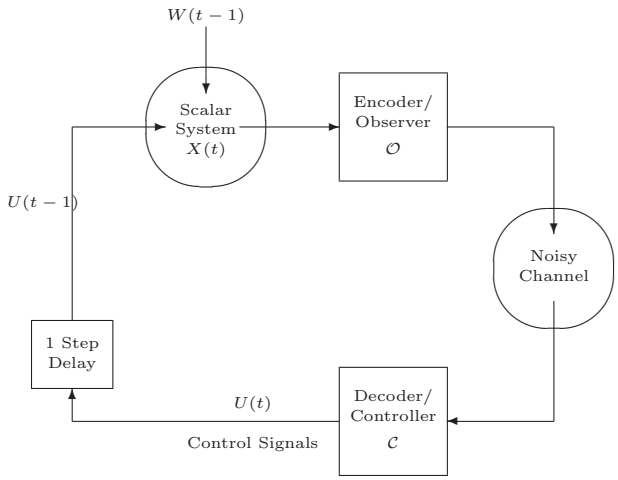


Figure 1.1. Control over a noisy communication channel without any explicit feedback path from controller to observer.

disturbance $W(t)$ is added to the state and the controller is allowed to input a $U(t)$ that will bring the state close to 0 again. The challenge in the problem is to control the system when the observer and controller are separated by a discrete memoryless channel. In this thesis, the goal is to maximize the value of $\eta$ for which the $\eta^{th}$ moment of the state $E[|X(t)|^{\eta}]$ is finite.

$$X(t+1) = \lambda X(t) + W(t) + U(t) \tag{1.1}$$

This is a problem of cooperative distributed control in that there are two different boxes that need to be designed: the observer/encoder that has access to the plant state and generates an input to the noisy channel, and the controller which has access to the channel outputs and applies a control to the plant. The reader is referred to two resources to get the appropriate background. The first is the classic 1978 paper by Ho, Kastner, and Wong [6] in which they summarize the then known relationships among team theory, signaling, and information theory from the perspective of automatic control. The more recent interest in the subject is best understood by reading the September 2004 special issue of Transactions on Automatic Control and the references contained therein.

Here, we focus exclusively on the case shown in Figure 1.1 where the information patterns [7] are non-nested in that the observer does not know everything the controller knows. This means that there is no explicit feedback of the channel outputs to the observer. Sahai's

previous work [8] showed that in the case of finite alphabet channels, the plant itself could be used as a "feedback channel" for communication. The controller can make the plant dance so as to noiselessly communicate the channel output to the encoder/observer. This showed that in principle, the non-nested information pattern did not increase the difficulty of the stabilization problem. [8] further showed that the problem of stabilization over a noisy channel was equivalent to a problem of anytime communication using the noisy channel with perfect feedback.

While this equivalence certainly illuminates the structure of distributed control problems (see [4] for more details), it does not actually allow us to solve any particular problem. This is especially true since the previously known generic constructions for anytime codes are based on infinite random trees with maximum-likelihood decoders. As such, they require growing complexity at both the encoder and decoder. This contrasts in spirit with the idea of stabilization making a system behave in a nice "steady-state" manner that returns to the same few safe states over and over again. If the closed-loop system is behaving nicely, why should the observer and controller have a steadily increasing workload?

In Chapter 2, we begin by introducing the relevant definitions and give some known results for this problem. The rest of the chapter is devoted to describing and analyzing a simple scheme making use of an random, time-varying, infinite constraint length convolutional code and a sequential decoder. The scheme is shown to be practical through simulations.

## 1.3 Convolutional codes and sequential decoding

Convolutional codes, trellis codes and tree codes offer relatively simple encoding schemes for both channel and source coding. Infinite constraint length convolutional codes can claim to yield exponentially decaying probability of error with decoding delay. For block codes on the other hand, probability of error can only decrease exponentially with block length, and not decoding delay. In a traditional view of information theory, this is of little importance since we always allow ourselves block lengths as large as required. In applications like control

over a noisy channel, however, this is crucial because they require that the probability of error go to zero exponentially with delay. Once the output of a block code has no dependance on an input symbol, it is impossible for the output to reduce our uncertainty of the input. Hence, block length is a harsh barrier after which the decoder simply cannot reduce the probability of error.

In channel coding and source coding, maximum likelihood (ML) and maximum a posteriori (MAP) decoding respectively serve as the optimal decoding strategies. For random tree channel coding with ML decoding, the probability of error with delay is bounded by

$$P_d \leq K_1 \exp(-dE_r(R)) \tag{1.2}$$

where $E_r(R)$ is Gallager's random coding error exponent [9] and $K_1$ is a finite constant that doesn't depend on delay.

Unfortunately, ML and MAP decoding suffer from exponential growth in complexity when decoding infinite constraint length convolutional codes. A solution to this issue was proposed in the late 1950's and early 1960's and it falls under the general heading of sequential decoding. One can imagine ML and MAP decoders as stepping through an expanding tree and examining every single node; hence the exponential growth in computation because the tree is growing exponentially. The essential feature of a sequential decoder is that it searches through the code tree, examining nodes sequentially (in an order) and selecting a subset to further examine, in a manner that does depend on nodes further into the tree. The comparison between ML or MAP and sequential decoding is similar to the comparison of breadth-first and depth-first search algorithms on trees. Because of this search nature, it is possible for sequential decoders to have reasonable complexity properties under certain conditions and yet be comparable to ML and MAP in terms of reliability. That is, for any $\epsilon > 0$, with sequential decoding, it is possible to have a probability of error with delay bounded by

$$P_d \leq K_\epsilon \exp(-d(E_r(R) - \epsilon)), \tag{1.3}$$

where $K_\epsilon$ is a finite constant. Also, a sequential decoder serves as a practical anytime decoder because its outputs are not targeted to any particular delay.

The computation, measured in nodes searched, performed by a sequential decoder is a random variable that depends on the code tree and realizations of channel noise or source sequence. Jacobs and Berlekamp [10] showed that the distribution of computation of any sequential decoder used for channel coding can have at best a power-law tail, meaning some of its moments will be infinite. Achievability results by Savage and Jelinek ( [11], [12]) show that a power-law tail for computation is tight and hence the distribution of computation is asymptotically Paretian. The parameter of the Pareto distribution is called the Pareto exponent of computation, and it gives the polynomial rate of decay in the probability of a large number of computations. A particularly nice property of sequential decoders is that they offer flexibility (via the 'bias' parameter) in trading off the finite moments of computation for reliability with delay. We use the results of Jelinek [1] to make several statements about computation or reliability when the bias is not appropriately set. For a thorough review of sequential decoding, see Forney [13].

The main contribution of Chapter 2 of the thesis is a simple scheme for control over a noisy channel (first published in [5]). The solution uses a random coding argument on trellis codes and a sequential decoder. The sequential decoder has a bounded complexity on average and can be considered to be 'stable' if certain properties of the bias are satisfied. To prove the validity of the solution, we make heavy use of results of Jelinek on sequential decoding [1].

## 1.4 Convolutional source coding with a sequential decoder

Lossless source coding is the problem of representing data efficiently in order to store or communicate it and recover with little or no error. The source coding problem is an old one with solutions such as Huffman coding and arithmetic coding. There are more robust solutions such as Lempel-Ziv ( [14], [15], [16]) coding and the Burrows-Wheeler Transform ( [17]) that can efficiently compress sources with memory and more complicated internal structure. The second half of this thesis is concerned with compressing a stream of incoming discrete source symbols that are modelled to be independent and identically

distributed into a bit stream for lossless recovery with exponentially decaying probability of error with decoding delay.

In Chapter 3, we analyze a scheme for lossless P-P source coding that uses a convolutional encoder and a sequential decoder. We prove and verify by simulation the analogous properties of Chapter 2. Just like before, the decoder has a variable bias parameter that can be used to tradeoff performance for complexity. We show the error exponent with delay using a sequential decoder is the same as the error exponent with delay using a MAP decoder. Hence an 'anytime source code' is produced with a simple encoder and a decoder having bounded complexity for all time.

The motivation of this chapter is not to propose a scheme for P-P source coding, because the scheme has in fact been proposed before in various forms. Hellman [18] proposed using convolutional codes for source coding to shift complexity from the encoder to the decoder. Independently, Koshelev [19] suggested using convolutional codes and sequential decoding together for source coding. He gave a computational cutoff rate for source coding with sequential decoding, analogous to the computation cutoff rate of Savage, Jelinek, etc. In the lossy compression setting, Jelinek [20] used a random tree coding argument to show that one could achieve the rate-distortion bound with tree codes. This was followed with work by Mohan and Anderson [21] pairing a stack decoder with a tree encoder for *encoding* of speech.

Our contribution in the area is in extending the results of Jelinek [1] to a source coding setting. We use an 'anytime source encoder' and the stack algorithm 'anytime decoder' to give exponentially decreasing probability of error with delay. The stack algorithm once again has a tunable bias parameter that allows for a tradeoff between reliability and complexity. We prove a theorem showing that a stack algorithm decoder can achieve an error exponent with delay equal to the source coding counterpart of Gallager's random coding error exponent. Next, we show that if the rate $R$ is at least $E_s(\rho)/\rho$, $\rho \in [0,1]$, where $E_s$ is the source counterpart to Gallager's $E_0$ function, then the $\rho^{th}$ moment of computation can be finite. For $\rho = 1$, this gives a sufficient condition on the rate to achieve a finite mean in computation. This is a result parallel to Koshelev's [19]. A theorem of Arikan and

Merhav in [2] is used to prove that the sufficient condition on rate is in fact necessary for any reliable sequential decoder.

The motivation here is to eventually generalize the P-P source tree code and sequential decoder to be used for lossless distributed source coding, the Slepian-Wolf problem shown in Figure 1.2. The encoders are separate and hence each has access to only one source. For two discrete correlated sources, $X$ and $Y$, Slepian and Wolf [22] showed that it is possible to losslessly recover the source pair if the rates of the two distributed encoders satisfy the following conditions, where $H$ denotes the appropriate entropy.

$$R_x + R_y > H(X,Y) \tag{1.4}$$

$$R_x > H(X|Y) \tag{1.5}$$

$$R_y > H(Y|X) \tag{1.6}$$

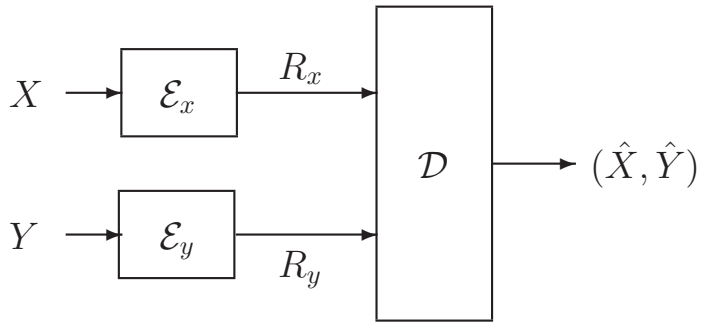


Figure 1.2. Distributed source coding, sometimes called the Slepian-Wolf problem.

If we wish to achieve a good reliability with decoding delay in the Slepian-Wolf problem, variable length codes are in general not a good idea. This is because the distributed encoders do not know when the source is behaving jointly atypically. This is easily seen in the case when $X$ is a random coin toss and $Y$ is $X$ passed through a binary symmetric channel with low crossover probability. Both encoders see random coin tosses, so there is no advantage to assigning longer codewords to any particular source sequences. So the question remains of how to get good reliability with delay in distributed lossless compression.

As a step towards this ultimate goal, we consider the problem of lossless source coding with side information as shown in Figure 1.3. In this problem, the decoder has access to side information $Y$ that is correlated with the source $X$, while the encoder has access to $X$ but not $Y$. It is well known that it is possible to losslessly recover $X$ if the rate $R$ is at least $H(X|Y)$. Again, the encoder does not know when the source is behaving jointly atypically with the side information, so variable-length codes do not provide a general solution. Instead, we consider the random binning scheme used in the point-to-point portion of the chapter. In section 3.6, by a minor modification to the sequential decoder in our P-P source coding scheme, we show that MAP reliability can be achieved by the stack algorithm. This shows that our encoder with stack decoder achieves that same error exponent with delay as the lower bound of Chang, et.al. [23]. In [23], it is shown that this error exponent is tight in the low rate regime for a source with uniform marginals and symmetric connection to the side information. Finally, we again verify the theory by simulating the encoder and decoder.

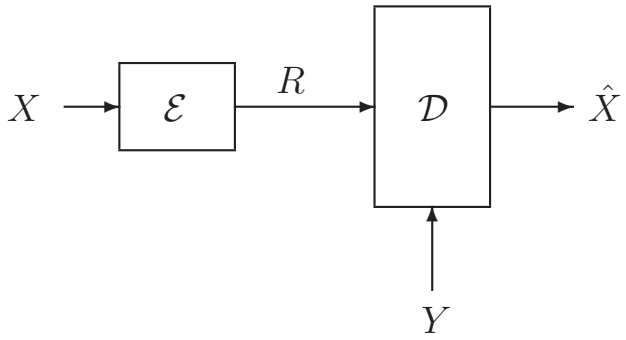


Figure 1.3. Source coding with side information.

# Chapter 2

# Control over a noisy channel

This chapter details our work on the subject of control over a noisy channel, with the observer and controller being separated by a communication channel. We start by defining the problem and goals of the solution. Then, we review some known results linking anytime capacity with control, and set the stage for our proposed solution. The contribution of this chapter is in showing how to use the stack decoder for the problem of distributed control. Then we state and prove the main result of the chapter by giving an achievable scheme using simple encoders and decoders and conditions for the scheme to work. We also define the notion of complexity that is considered and the conditions under which it is kept low. Next, simulation results are shown to verify that the proposed scheme is not impractical. The last section explores the tradeoff between performance and complexity in our scheme and is followed by concluding remarks.

## 2.1 Problem definition

At the heart of the stabilization problem is a scalar plant that is unstable in open loop. Expressed in discrete-time, we have:

$$X(t+1) = \lambda X(t) + U(t) + W(t), \quad t \geq 0 \tag{2.1}$$

where $\{X(t)\}$ is a real-valued state process. $\{U(t)\}$ is the real-valued control process that our controller generates and $\{W(t)\}$ is an external, bounded noise/disturbance process s.t. $|W(t)| \leq \frac{\Omega}{2}$ almost surely. Also, $\lambda \geq 1$ so the system is unstable, while $X(0) = 0$ for convenience.

The distributed nature of the problem comes from having a noisy communication channel in the feedback loop. We assume that the channel is discrete and memoryless with a finite[1] input alphabet $\mathcal{A}$ and finite output alphabet $\mathcal{B}$. The channel is described probabilistically with a transition matrix $P(y|x)$.

We require an observer/encoder system $\mathcal{O}$ to observe $X(t)$ and generate inputs $A(t)$ to the channel. We also require a decoder/controller system $\mathcal{C}$ to observe channel outputs $B(t)$ and generate control signals $U(t)$. In general, we could allow both $\mathcal{O}, C$ to have arbitrary memory and to be nonlinear. In this thesis, we will aim for a nearly memoryless $\mathcal{O}$, which restricts the encoder to be fairly simple.

**Definition 2.1.1** *A closed-loop dynamic system with state $X(t)$ is $\eta$-**stable** if there exists a constant $K$ s.t. $E[|X(t)|^\eta] \leq K$ for all $t \geq 0$.*

Holding the $\eta$-moment within bounds is a way of keeping large deviations rare.[2] The larger $\eta$ is, the more strongly we penalize very large deviations.

Now we define anytime capacity, which is a concept separate of control under communication constraints. Let $M_i$ be the $R$ bit message that a channel encoder takes in at time $i$. Based on all the messages received so far, and any additional information (e.g. past channel output feedback) that it might have, the channel encoder emits the $i$-th channel input. An anytime decoder provides estimates $\hat{M}_i(t)$, the best estimate for message $i$ at time $t$. If we are considering a delay $d$, the probability of error we are interested in is $P(\hat{M}_{t-d}(t) \neq M_{t-d})$.

**Definition 2.1.2** *The $\alpha$-**anytime capacity** $C_{anytime}(\alpha)$ of a channel is the least upper bound of the rates at which the channel can be used to transmit data so that there exists a*

[1] For convenience, we have assumed finite alphabets, but we really only require countable alphabets.

[2] [4] shows how to map this sense of stability to almost-sure stabilization in the undisturbed case when $W(t) = 0$.

*uniform constant $K$ such that for all $d$ and all times $t$ we have*

$$P(\hat{M}_{t-d}(t) \neq M_{t-d}(t)) \leq K2^{-\alpha d}$$

The probability is taken over the channel and any randomness that we deem the encoder and decoder to have access to. Due to the fast convergence of an exponential, it is equivalent to requiring that the probability of error for *all messages sent upto time* $t-d$ is bounded by $K'2^{-\alpha d}$. Below, we define several standard error exponents, which can be found in [9].

**Definition 2.1.3 Gallager's function**, $E_0(\rho)$, for a discrete memoryless channel $P(y|x)$ is

$$E_0(\rho) = \sup_{Q(x)} -\log_2 \sum_y \left( \sum_x Q(x)P(y|x)^{\frac{1}{1+\rho}} \right)^{1+\rho} \tag{2.2}$$

where the maximization is over distributions $Q(x)$ on the input alphabet.

Gallager's **random coding exponent**, $E_r(R)$, at a rate $R$ for the channel without feedback is

$$E_r(R) = \sup_{0 \leq \rho \leq 1} E_0(\rho) - \rho R \tag{2.3}$$

The **sphere packing exponent**, $E_{sp}(R)$, at a rate $R$ for the channel without feedback is

$$E_{sp}(R) = \sup_{\rho \geq 0} E_0(\rho) - \rho R \tag{2.4}$$

Gallager's random coding exponent is achievable with random codes as its name suggests. The sphere packing exponent serves as an upper bound to the error exponent for arbitrary block codes. Hence, for random, time-varying infinite constraint length convolutional codes, the anytime capacity of the channel without feedback satisfies:

$$C_{anytime}(E_r(R)) \geq R \tag{2.5}$$

$$C_{anytime}(E_{sp}(R)) \leq R \tag{2.6}$$

## 2.2 Known results

In the paper of Sahai and Mitter [4], a matching achievability and converse result is given for this problem of distributed control in terms of the anytime capacity of the noisy channel with feedback.

**Theorem 2.2.1** ( [4], [8]) For a given noisy channel, bound $\Omega$, and $\eta > 0$, if there exists an observer/encoder $\mathcal{O}$ and controller/decoder $\mathcal{C}$ for the unstable scalar system that achieves $E[|X(t)|^\eta] < K$ for all bounded driving noise $-\frac{\Omega}{2} \leq W(t) \leq \frac{\Omega}{2}$, then $C_{\text{anytime}}(\eta \log_2 \lambda) \geq \log_2 \lambda$ bits per channel use for the noisy channel considered with noiseless feedback.

And for the non-nested information pattern case:

**Theorem 2.2.2** ( [4], [8]) It is possible to control an unstable scalar process driven by a bounded disturbance over a noisy finite output-alphabet channel so that the $\eta$-moment of $X(t)$ stays finite for all time if the channel with feedback has $C_{\text{anytime}}(\alpha) \geq \log_2 \lambda$ for some $\alpha > \eta \log_2 \lambda$ if the observer is allowed to observe the state $X(t)$ exactly.

The encoders and decoders used in the proofs of these theorems were far from simple. Next we define the class of simplified encoders and decoders we wish to restrict ourselves to.

By nearly memoryless observers, we mean $\mathcal{O}$ functions that sample the plant state every $T$ time units (for some $T > 0$), and then apply a channel input that depends only on the last such sample.

**Definition 2.2.3** A **random nearly memoryless observer** is a sequence of maps $\mathcal{O}_t$ such that there exist $\mathcal{O}'_t$ so that:

$$\mathcal{O}_t(X_0^t, Z_0^t) = \mathcal{O}'_t(X(T\lfloor \frac{t}{T} \rfloor), Z_{\lfloor \frac{t}{T} \rfloor})$$

where the $Z_i$ are the iid continuous uniform random variables on $[0, 1]$ that represent the common-randomness available at both the observer/encoder and the controller/decoder.

**Definition 2.2.4** A **sequential decoder** for a trellis code is one that searches paths through the trellis in a sequential and causal manner. A thorough summary on sequential decoders can be found in Forney's survey paper [13].

The following result will guide us in our approach and can be found in [4].

**Theorem 2.2.5** We can $\eta$-stabilize an unstable scalar process driven by a bounded disturbance over a discrete memoryless channel if the channel without feedback has random block-coding error exponent $E_r(R) > \eta \log_2 \lambda$ for some $R > \log_2 \lambda$ and the observer is only allowed access to the plant state where $E_r(R)$ is Gallager's random coding error exponent. Furthermore, this can be achieved with a nearly memoryless random encoder/observer.

The difference between theorems 2.2.5 and 2.2.2 is subtle. The first asks for the random coding exponent *without feedback* to be at least $\eta \log_2 \lambda$, while the other asks for the $\eta \log_2 \lambda$-anytime capacity of the channel *with feedback* to be at least $\log_2(\lambda)$. Recently, there have been results [24] showing that the error exponents for a channel with feedback can be much higher than the random coding exponent and even the sphere packing exponent for the channel without feedback.

The scheme we use in the next section will adapt from the proof of this theorem, using exactly the same encoder. However, the prior proof required ML decoding over a growing trellis at the controller. It does not take long for this scheme to become impractical due to the exponential growth in required computation. This provides the motivation for us to consider using a sequential decoder at the controller.

## 2.3 Main result

### 2.3.1 Theorem and proof

**Theorem 2.3.1** *We can $\eta$-stabilize an unstable scalar process driven by a bounded disturbance over a discrete memoryless channel if the channel without feedback has random block-coding error exponent $E_r(R) > \eta \log_2 \lambda$ for some $R > \log_2 \lambda$. Furthermore, this can be*

*achieved with the same nearly memoryless encoder/observer used in [4] and a stack-based sequential decoder/controller.*

**Proof:** Pick a rate $R > \log_2 \lambda$ for which $E_r(R) > \eta \log_2 \lambda$. Pick $(T, \Delta)$ large enough so that:

$$TR > \log_2(\lambda^T + \frac{\sum_{i=0}^{\infty} \lambda^{T-i}\Omega}{\Delta})$$

This means that if $X(t)$ is known to be within any given box of size $\Delta$, then with no controls applied, the plant state $X(t+T)$ can be in no more than $2^{TR}$ adjacent boxes each of size $\Delta$. It is clear that such a $T, \Delta$ pair always exists as long as $R > \log_2 \lambda$.

Our quantizer $Q$ will look at the plant state $X$ uniformly with a coarseness of $\Delta$. It is clear that we satisfy the following:

a. Knowing that $X(t)$ is in a given bin of width $\Delta$ and assuming that no controls are applied, there are at most $2^{TR}$ possible bins which could contain $X(t+T)$.

b. The descendants of a given bin $dT$ time units later are all in contiguous bins and furthermore, there exists a constant $K$ such that the total length covered by these bins is $\leq K\lambda^{dT}$.

Properties [a] and [b] above easily extend to the case when the control sequence between time $X(t)$ and $X(t+T)$ is known exactly since linearity tells us that the impact of these controls is just a translation of all the bins by a known amount. Thus, we have:

c. Conditioned on actual past controls applied, the set of possible paths that the states $X(0), X(T), X(2T), \ldots$ could have taken through the quantization bins is a subset of a trellis that has a maximum branching factor of $2^{TR}$. Furthermore, the total length covered by the $d$-stage descendants of any particular bin is bounded above by $K\lambda^{dT}$.

Not all such paths through the trellis are necessarily possible, but all possible paths do lie within the trellis. Figure 2.1 illustrates the trellis. The observer/encoder independently assigns symbols from $\mathcal{A}^T$ as time-varying labels to the bins. The probability measure used

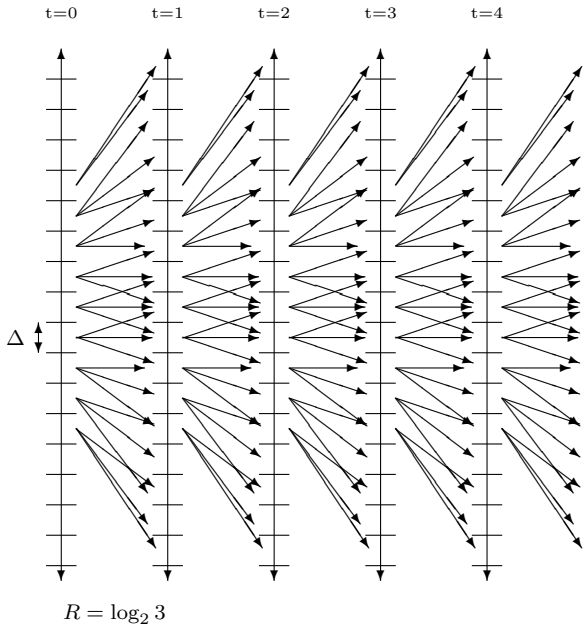


Figure 2.1. A short segment of the randomly labeled regular trellis from the point of view of the controller that knows the actual control signals applied in the past. The example has $R = \log_2 3$ and $\lambda \approx 2.4$ with $\Delta$ large.

to draw the symbols should be the $E_r(R)$ achieving distribution. Thus, the labels on each bin are iid through both time and across bins.

We call two paths through the trellis disjoint with depth $d$ if their last common node was at depth $d$ and the paths are disjoint after that. We immediately observe:

d. If two paths are disjoint in the trellis at a depth of $d$, then the channel inputs corresponding to the past $dT$ channel uses are independent of each other.

In decoding, the controller creates its trellis with the knowledge of past control inputs. Then, it decodes the channel outputs at times which are multiples of $T$ using a version of the Stack or Zigangirov-Jelinek Algorithm that is referred to as 'Algorithm A' in [13]. This version is simple and does not consider merging of paths on the trellis. It is given below for reference.

1. Insert the root of the trellis on the stack, with a metric of 0. Let $\mathbf{x}'$ denote the partial path on the top of the stack.

2. For each of the successors of $\mathbf{x}'$, compute their metrics as the sum of the metric of $\mathbf{x}'$ and the metric of the branch that extends $\mathbf{x}'$. Place each of these successors on the stack, remove $\mathbf{x}'$ and sort the stack by metric so that the partial path with the best metric is at the top.

3. Let $\mathbf{x}'$ be the new partial path on top of the stack. If $\mathbf{x}'$ extends up to the current time, then declare $\mathbf{x}'$ to be the decoded path and the last node of $\mathbf{x}'$ to be the bin in which $X(t)$ resided when the first of the $T$ most recent channel symbols were transmitted. Otherwise, return to step 2.

It has been shown that if $x$ is a channel input symbol and $y$ is a channel output symbol, the optimal metric associated with this symbol pair is of the form $m(x,y) = \log_2 \frac{p(y|x)}{p(y)} - G$, where $G$ is a real parameter of the algorithm. For a branch with $T$ symbols, the metric of the branch would then be $\sum_{i=0}^{T-1} \log_2 \frac{p(y_i|x_i)}{p(y_i)} - G$. The parameter $G$ will be used to tradeoff complexity and performance of the controller. If $G > \max_{x,y} \log_2 \frac{p(y|x)}{p(y)}$, then the stack algorithm reverts to an ML decoder as every extension of a path lowers the path's metric.

Fix a time $t$ and consider an error event at depth $d$. This represents the case that the sequential decoder's declared path last intersected with the true path $dT$ time steps ago. By property [c] above, our control will be based on a state estimate that can be at most $K\lambda^{dT}$ away from the true state. Thus, we have:

e. If an error event at depth $d$ occurs at time $t$, the state $|X(t+T)|$ is smaller than $K'\lambda^{(d+1)T}$ for some constant $K'$ that does not depend on $d$ or $t$.

By property [c], there are no more than $2^{dTR}$ possible disjoint false paths that last intersected the true path $d$ stages ago. By the memorylessness of the channel, the log-likelihood of each path is the sum of the likelihood of the "prefix" of the path leading up to $d$ stages ago and the "suffix" of the path from that point onward. Property [d] tells us that the channel inputs corresponding to the false paths are pairwise independent of the true inputs for the past $dT$ channel uses. For a path that is disjoint from the true path at a depth of $d$ to beat all paths that end up at the true final state, the false path must have a

final log-likelihood that at least beats the metric of the true path at the point of divergence or beyond, until the current time. With this observation and some techniques developed by Jelinek extending the random coding error exponent arguments, we have:

**Theorem 2.3.2** ( [1]) A detailed proof is located in A.1. For a decoder using the Stack Algorithm with bias parameter $G$, let $E_d$ denote the event that some false path that diverged with the true path $d$ branches ago is declared the decoded path. Then, if $G$ satisfies the condition below for $\rho$ such that $R = E_0'(\rho)$ and $E_0(\rho)$ is Gallager's function, $P(E_d) < L\exp_2(-dTE_r(R))$, $L < \infty$.

$$G > \frac{1+\rho}{\rho}\left[E_0(\rho) + f(\rho)\right] \text{ where } f(\rho) \triangleq \log_2 \sum_y p(y)\left[\sum_x \left[\frac{p(y|x)}{p(y)}\right]^{\frac{1}{1+\rho}} p(x)\right]^{\rho}$$

This result is the reason for why a sequential decoder can stabilize the same moments as an ML decoder. Now we finish proving stability of the $\eta$-th moment by applying a union bound over error depths, using the stated theorem, and observation [e].

$$\begin{aligned} E[|X(t+T)|^{\eta}] &= \sum_{d=0}^{\lfloor t/T \rfloor} P(E_d) \cdot E[|X(t+T)|^{\eta}|E_d] \\ &< \sum_{d=0}^{\infty} L2^{-dTE_r(R)} \cdot (K\lambda^{(d+1)T})^{\eta} \\ &= LK^{\eta}\lambda^{T\eta} \sum_{d=0}^{\infty} 2^{-dT(E_r(R) - \eta\log_2\lambda)} \\ &= K' < \infty \end{aligned}$$

The final geometric sum converges because we assume $E_r(R) > \eta \log_2 \lambda$. □

For sequential decoding, the decoder has to perform a random amount of computation at each time step. The random variable of computation is of practical interest in this scenario and we explore it next.

### 2.3.2 The random variable of computation

To explain the classical view of computation for sequential decoding, consider an infinite trellis being partitioned by the true path through the trellis and a sequential decoder running for eternity to decode the true path.

**Definition 2.3.3** *The $i^{th}$* **incorrect subtree** *is defined by the set of paths that have their last common node with the true path at depth $i$ into the trellis. The* **random variable of computation** *$N_i$ is defined to be the number of nodes in the $i^{th}$ incorrect subtree that are ever visited by the sequential decoder.*

**Theorem 2.3.4** ( [13], [1]) If $G = R$ and $\rho$ is defined parametrically by the equation $R = E_0(\rho)/\rho$, then the $\gamma$-th moment of $N_i$ is finite for $\gamma < \rho$. In particular, if $R < E_0(1)$, the expected mean of computation is finite.

This result, combined with the converse of Jacobs and Berlekamp [10] and Arikan [25], shows that the random variable of computation is asymptotically Paretian, with parameter $\rho$. This theorem applies to our sequential decoder, but in the context of stabilization one would like to say something about the number of computations performed at any given time.

**Definition 2.3.5** *The* **random variable $\mathbf{C(k)}$** *is defined to be the number of nodes visited by the sequential decoder at time $k$. The r.v.* **$\mathbf{C_i(k)}$** *is the number of nodes in the $i^{th}$ incorrect subtree visited at time $k$.*

We note the following relation between these two notions of computation. Suppose $E[N_i] = K < \infty$. Then, we have

$$\begin{aligned} \lim_{k\to\infty} E[C(k)] &= \lim_{k\to\infty} E[\sum_{i=1}^{k-1} C_i(k)] \\ &\leq \lim_{k\to\infty} \sum_{n=0}^{k-2} \sup_{l>0} E[C_l(n+l)] \end{aligned}$$

There are more partial paths on the stack as time increases, and the metrics of the incorrect subtree relative to the correct path are the same regardless of $i$, so the expected value of $C_l(n+l)$ should be insensitive to $l$ since the averaging is performed over all possible random

| Parameter | Description |
|---|---|
| $\Delta$ | Size of bins the state is quantized to. |
| $T$ | Time between observations of the state. |
| $\Omega$ | Absolute bound on noise, $\|W(t)\| \in [-\Omega/2, \Omega/2]$. |
| $\lambda$ | Unstable eigenvalue of the plant. |
| $G$ | Per symbol bias value used in stack algorithm metric. |
| $R$ | 'Rate' of code: branch factor is $2^{TR}$. |
| $\epsilon$ | Crossover probability of binary symmetric channel. |

Table 2.1. Parameters for simulation.

codes and channel noise. Thus, we have

$$\begin{aligned} \lim_{k\to\infty} E[C(k)] &\leq \sum_{n=0}^{\infty} E[C_1(n+1)] \\ &= E[\sum_{n=0}^{\infty} C_1(n+1)]] \\ &= E[N_1] < \infty \end{aligned}$$

Having shown that the mean of $C(k)$ stays bounded, we conjecture that if a moment of $N_i$ exists for all $i$, the same moment of $C(k)$ will exist for all $k$.

## 2.4 Simulations

The main advantage of using sequential decoding over ML decoding, as we have claimed, is that the complexity of the decoder does not grow unboundedly as time goes on, except for the requirement of memory that grows roughly linearly with time. By simulating the strategy described in Theorem 2.3.1, we show that complexity is reduced to such an extent as to make the scheme practical. The simulations were performed with a binary symmetric channel with crossover probability $\epsilon$, although they have also been verified with other DMCs. Also, we note that the rate is defined to be $\log_2(B)/T$ where $B$ is the branching factor of the decoding trellis over a block of $T$ symbols.

Since the overall system 'operates' at a resolution of $\Delta$, it is not interesting to look at both the state and the bin of the state, and so we focus on the actual and decoded bins, rather than the state. Figure 2.2 shows a sample path of the actual bin, and a sample

path of the decoded bin along with a sample path of $C(k)$. The sample path shows that large deviations of the state from 0 are rare. Figure 2.4 looks at the sample path during one of these rare deviations. As expected, the decoder completely loses track of the state for several blocks, all the while applying erroneous control inputs, and in one step finally catches up with the errors.

By applying Markov's inequality using a moment $\eta$ that we know can be stabilized, we have

$$P(|X(t)|^\eta \geq x^\eta) \leq K_s x^{-\eta}$$

Similarly, $P(C(k) \geq N) \leq K_c N^{-\gamma}$ if we know the $\gamma$-th moment of computation to be finite. Viewed on a log-log plot, both these quantities should decay linearly with a slope equal to the supremum of finite moments. In Figure 2.5, we see that with a bias that is optimized to achieve the random coding exponent for probability of error, the computation[3] does not have the moment stability guaranteed by Theorem 2.3.4 if we had the bias equal to the rate. Soon, we will give two results on what can be said about the power laws if the bias is optimized for computation or reliability.

In Figure 2.3, a plot of the random coding exponent and sphere packing exponent is given for the example simulation used. It shows that the two exponents, which are respectively lower and upper bounds on the error exponent with delay for coding *without feedback*, agree at rate $R = 0.317$. The power law for the state translates to $E_r(R)/\log_2(\lambda) = 1.2$. In Figure 2.5, we see the power law for the state in this example is 1.7, significantly better than 1.2. This suggests that there is some looseness in the bounds we have derived in the main result. Specifically, there is likely some hidden merging occurring in the trellis due to the noise in the plant.

[3]In all plots, computation refers to $C(k)$. We have made the assumption of ergodicity as well as that the moments of the two computations are equal asymptotically. Although this hasn't been proven, it is plausible because the simulations show that $C(k)$ is not growing unbounded over time.

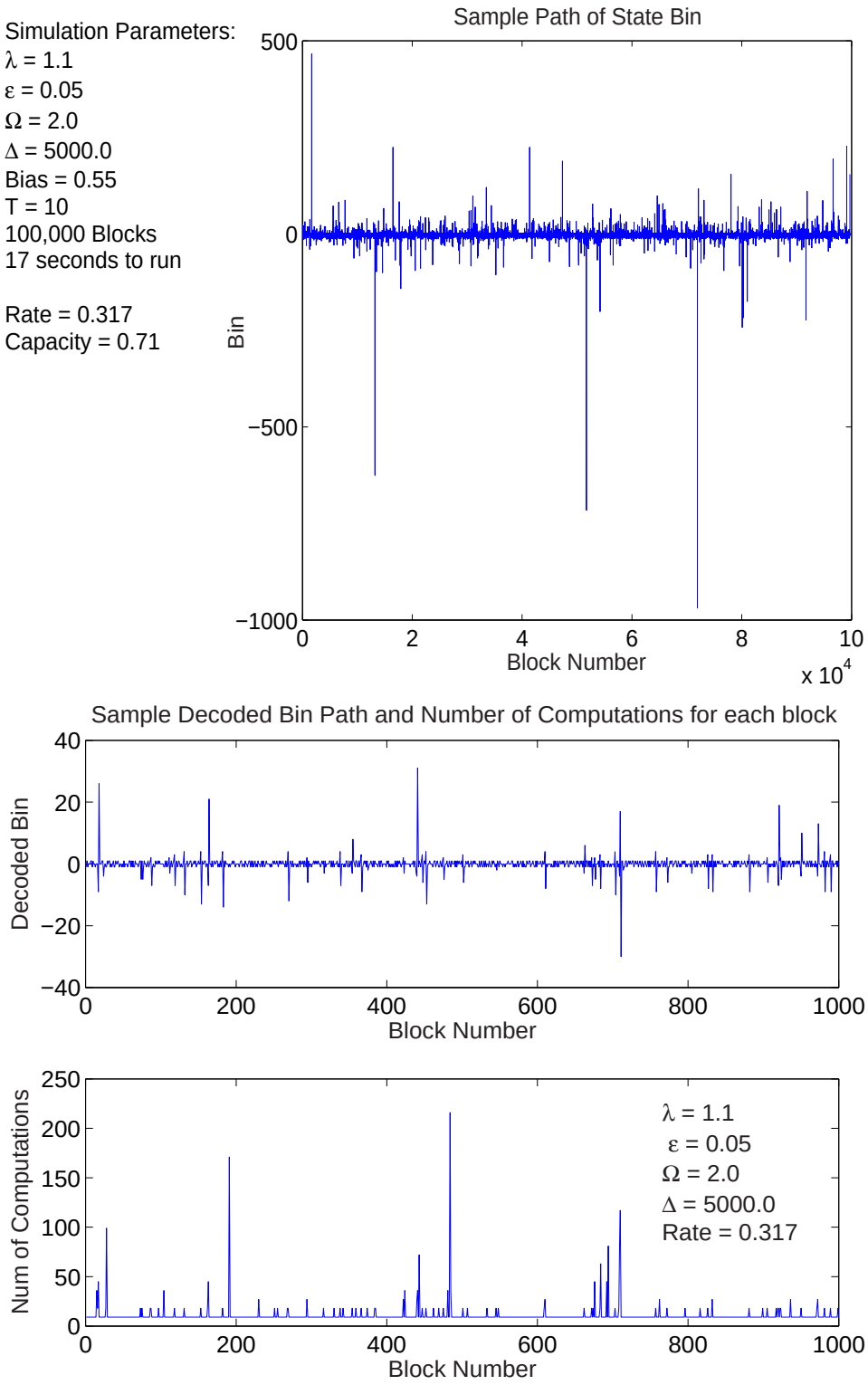


Figure 2.2. Top: A sample path of the actual bin the state lies in over 100,000 branches of length 10. Bottom: A sample path of the decoded bin and C(k). Note that large deviations in the decoded bin from 0 generally occur near high periods of computation.

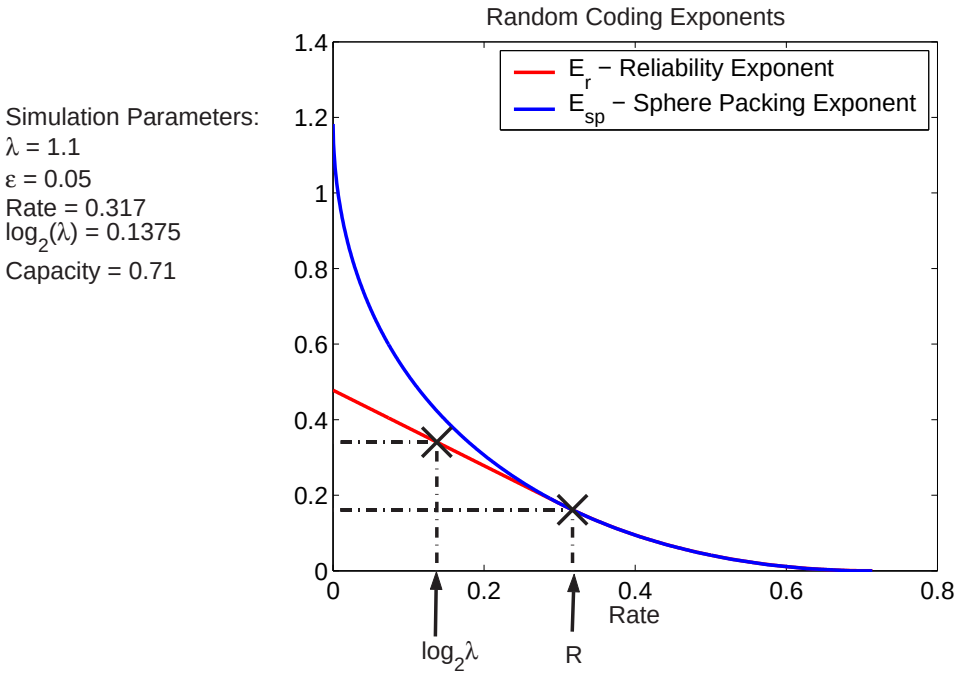


Figure 2.3. An example of running the encoder/decoder at a rate $R$ and the exponent guaranteed by the theorem.

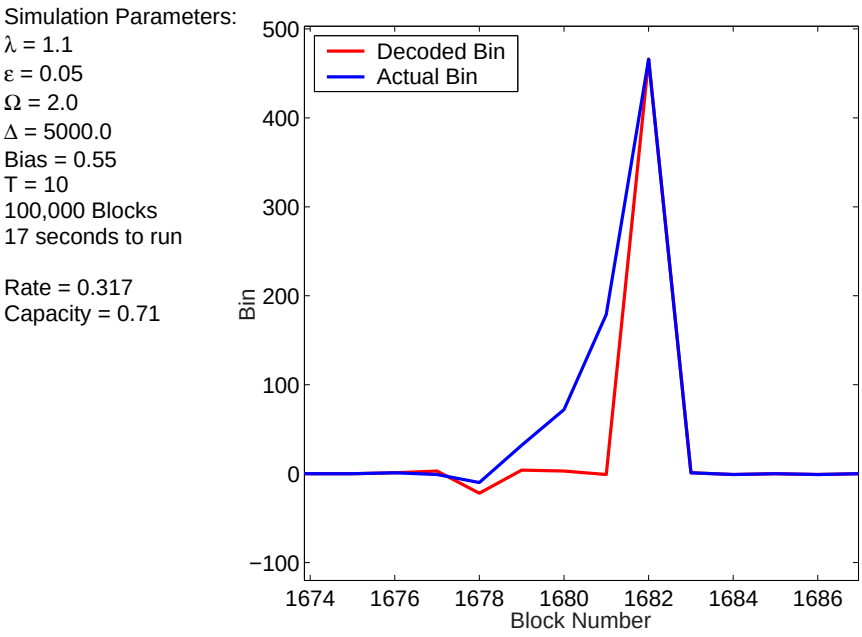


Figure 2.4. A zoomed-in look at the state's deviation from zero for a sample path.

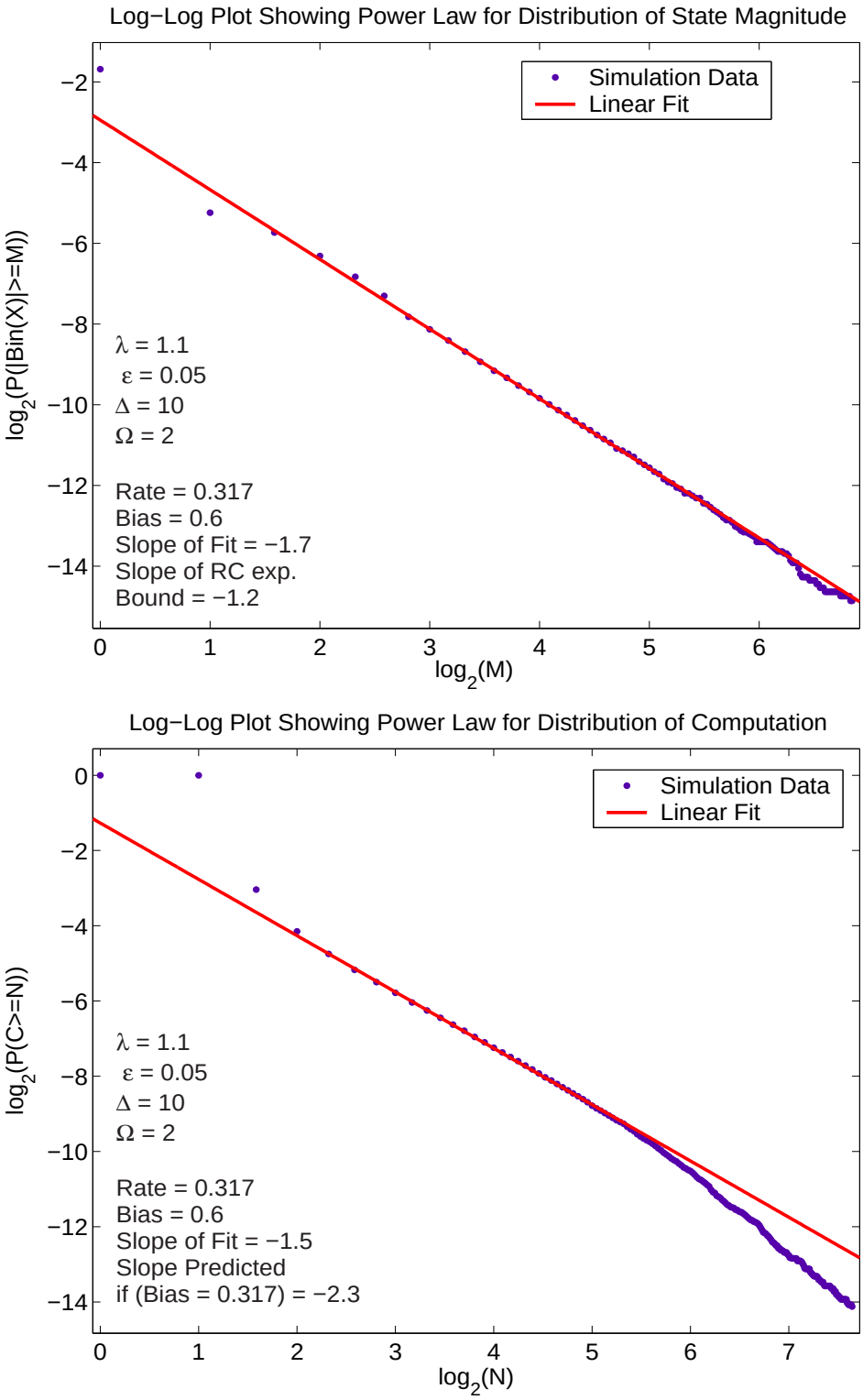


Figure 2.5. Top: Log-log plot of the complementary cdf of the state. Bottom: Log-log plot of the random variable of computation.

## 2.5 Tradeoff between computation and stability

As noted in [1], it is *not possible* in general to optimize both probability of depth $d$ error and computation simultaneously with a single choice of bias. For the BSC, the bias required to achieve the random coding exponent is always greater than the rate of the trellis code.

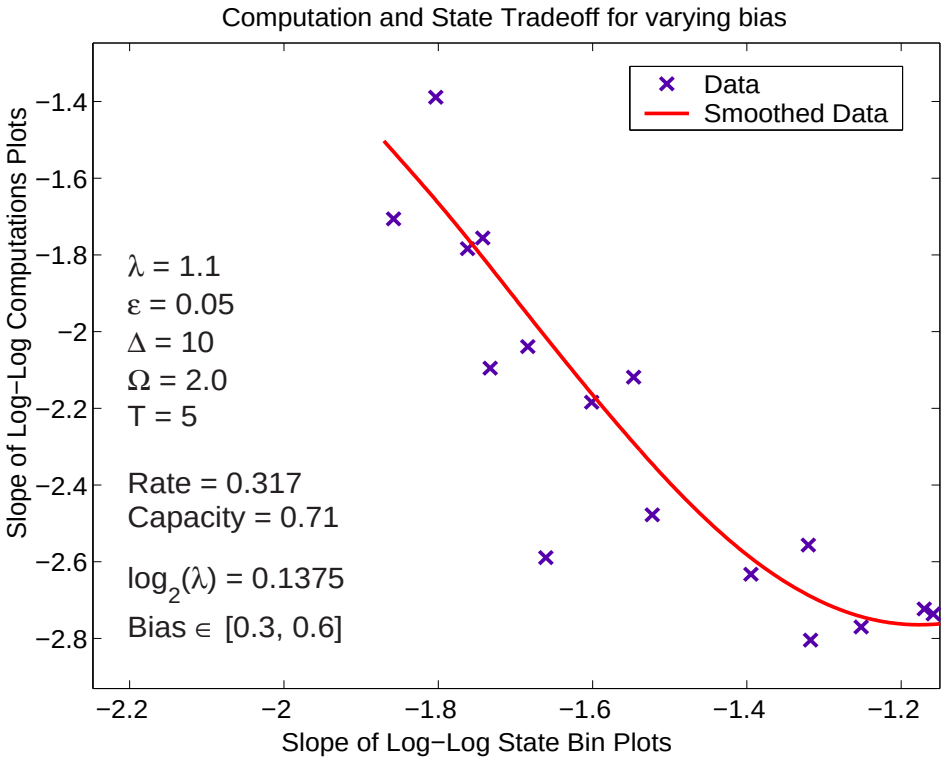


Figure 2.6. An example of the computation and state tradeoff. The value of bias is hidden in this plot, with each mark representing a different value of the bias.

Figure 2.7 shows the tradeoff between these two metrics of performance as a function of the bias parameter. Another way of looking at this tradeoff is shown in figure 2.6. As expected, increasing the bias always improves state stability. However, decreasing the bias does not always improve the moments of computation. By decreasing the bias below the rate, we run the risk of forging ahead into the trellis along paths that are not correct, only to have to return to the nodes we passed over later on. [4]

It should be noted that the only difference between the two plots in Figure 2.7 is $\lambda$.

[4] It is possible to reduce the bias enough so that the decoder never backs up, but this is uninteresting as the state will diverge almost surely.

It is not surprising that a smaller $\lambda$ leads to better state stability, but it is interesting to note that the computation fares better too. Even though we search the same number of descendants of every bin, the true path is restricted to a smaller range of paths through the trellis. This indicates that paths can 'accidentally' remerge with the true path more often when $\lambda$ is small. That is, when the decoder does not realize it is wrong, the noise in the plant can push the false path into the true one. As $\Omega/\Delta$ goes to 0, one expects this phenomenon to dissipate.

### 2.5.1 Setting the bias for reliability

As mentioned before, it is in general not possible to choose the bias so as to simultaneously optimize the probability of error with delay and the Pareto exponent for computation. For the problem of control, we may wish to maximize the moments of the state that are stabilized. This objective translates to choosing the bias to optimize the probability of error with delay. We now see what can be said about the Pareto exponent when the bias is set to optimize reliability.

For a rate $R \in (E_0'(1), C)$, let $\eta$ satisfy the equation $E_0'(\eta) = R$. If $R \in [0, E_0'(1)]$, let $\eta = 1$. Let $G = \frac{E_0(\eta)}{\eta}$. From the conditions of Thoerem 2 in [1], this choice of $G$ guarantees that the probability of error decays exponentially with delay, with exponent equal to Gallager's random coding exponent evaluated at rate $R$.

A separate theorem from Forney [13] tells us that if $\gamma$ and $G$ satisfy $G = E_0(\gamma)/\gamma$, and the rate $R = G$, then the Pareto exponent of computation is $\gamma$. This holds regardless of whether $\gamma > 1$ or not.

Now, returning to the assumption that $G$ is set for reliability, let $G = E_0(\eta)/\eta$ where $E_0'(\eta) = R$. The concave nature of $E_0$ implies that $G > R$, as depicted in Figure 2.5.1. The rate of the code is less than $R' = G = E_0(\eta)/\eta$, so the Pareto exponent must be at least $\eta$, because the Pareto exponent for computation is monotonically decreasing in rate. If $R = E_0(\gamma)/\gamma$, the maximum possible Pareto exponent at rate $R$ is $\gamma$, hence $\eta < \gamma$.

In particular, if $R \in [0, E_0'(1)]$, and $G = E_0(1)$, then the mean of the random variable of

computation is finite. This tells us that if our rate is low enough, we can get the reliability of an ML decoder with a finite mean of computation.

### 2.5.2 Setting the bias for computation

Historically, the usual choice of bias $G$ for the stack algorithm is $G = R$, which always maximizes the Pareto exponent of computation. In general, if $G^*$ is the minimum value of bias to guarantee that the decoder achieves the random coding exponent, then $G^* > R$. So if we set the bias for computation, we are not guaranteed, and generally will not attain the random coding exponent. Let $P(F_d)$ denote the probability of a decoding error by the stack decoder of depth d. From the proof of Theorem 3 of [1], we know that if $G = R < G^*$,

$$\begin{aligned} P(F_d) &\leq K \exp\Big( - d(g(\sigma,\delta) + \delta(\sigma-1)R)\Big) && (2.7)\\ &\quad \delta \in [0,1],\ \sigma \geq 0 && (2.8)\\ g(\sigma,\delta) &\triangleq \ln \sum_y w(y) \Big[ \sum_x Q(x) \Big( \frac{w(y|x)}{w(y)} \Big)^{\sigma} \Big]^{\delta} && (2.9) \end{aligned}$$

We can set $\sigma = 1/(1+\delta)$ and optimize over $\delta$ to get

$$\begin{aligned} P(F_d) &\leq K \exp\Big( - d\Big( \sup_{\delta\in[0,1]} -g(1/(1+\delta),\delta) - \frac{\delta^2}{1+\delta} R\Big)\Big) && (2.10)\\ &\leq K \exp\Big( - dE_c(R)\Big) && (2.11)\\ E_c(R) &\triangleq \sup_{\delta\in[0,1]} -g\Big(\frac{1}{1+\delta},\delta\Big) - \frac{\delta^2}{1+\delta} R && (2.12) \end{aligned}$$

Figure 2.5.2 shows an example of a comparison between $E_c(R)$ and $E_r(R)$. We see that $E_c$ is positive for rates up to capacity, but much less than $E_r(R)$. By setting the bias for computation, we can still get exponentially decaying probability of error and hence there will be some moments of the state that are stabilized.

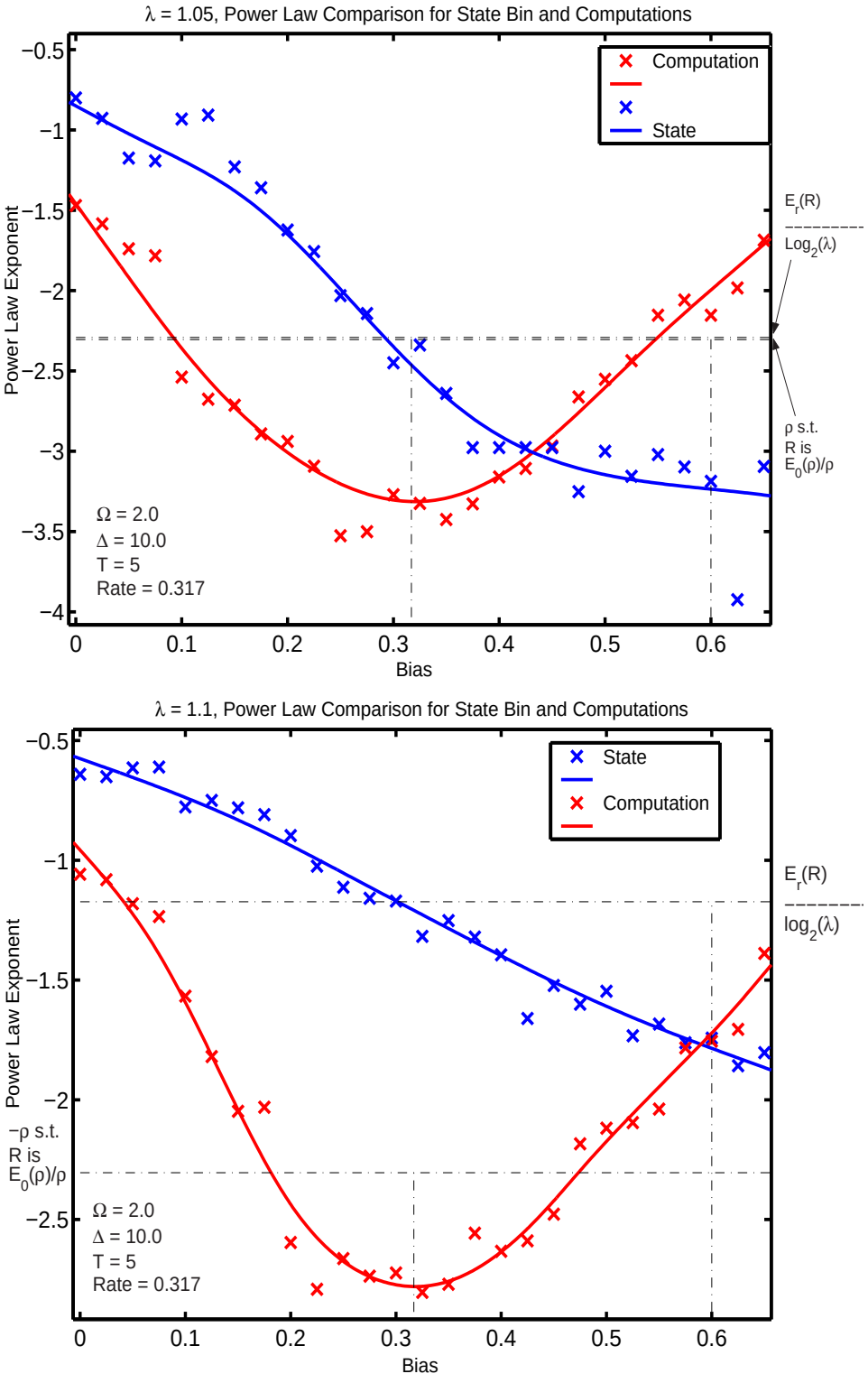


Figure 2.7. Top: Plot of the computation and state stability tradeoff for $\lambda = 1.05$. Bottom: Plot of the computation and state stability tradeoff for $\lambda = 1.1$.

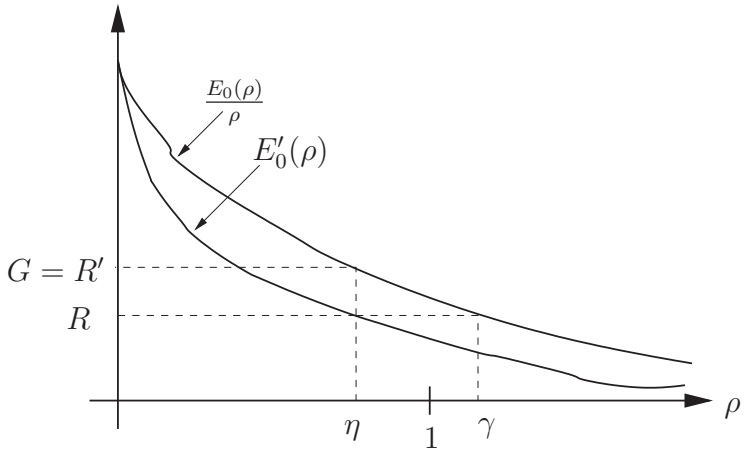


Figure 2.8. Typical plots of $E_0'(\rho)$ and $E_0(\rho)/\rho$.

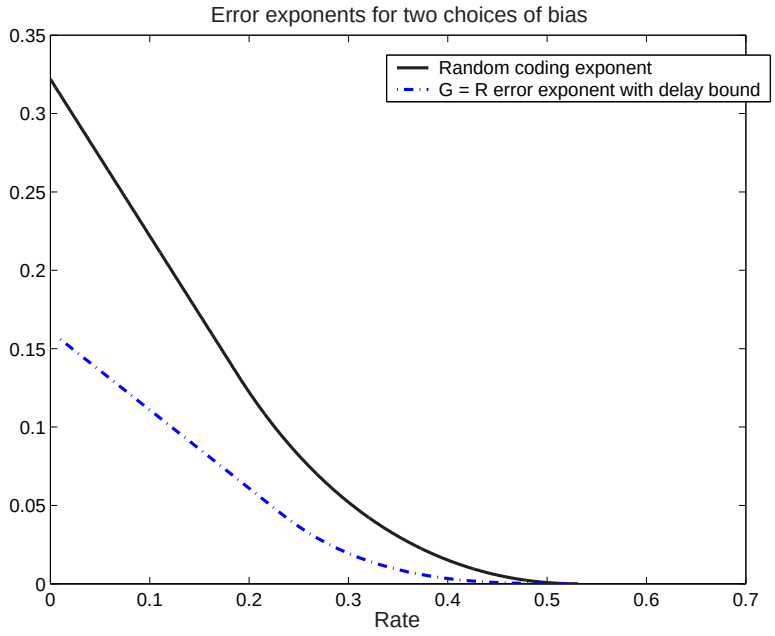


Figure 2.9. Reliability exponent when the bias is set to minimize computation for a binary symmetric channel with crossover probability 0.1.

## 2.6 Comments

An implicit assumption made is that the controller/decoder has unlimited computational speed so that no matter how many computations must be performed at one time, the decoder can finish them and apply the control to the plant before the next time step. Suppose instead that no more than $L-1$ computations may be performed within any $T$ time steps. If there are more than $NL$ computations required at one time, the controller takes no action for $N$ blocks, leaving the state to wander until it can decode. It is known that the distribution of computation can be lower bounded by a Pareto distribution [10] with some parameter $\gamma$. So even in the case that the noise is negligible, assuming $|X(t)| = x$, we have

$$\begin{aligned} E\Big[|X(t+NT)|\Big||X(t)| = x\Big] &\geq E\Big[|X(t+NT)|\Big||X(t)| = x, C(t) > NL\Big] P(C(t) > NL) \\ &\geq xK_c \cdot \lambda^{NT} N^{-\gamma} \end{aligned}$$

Letting $N$ go to infinity, we see that expected value of the state diverges. Clearly, this applies to any moment of the state. So this scheme fails to stabilize the state if the controller has limited computational speed, regardless of how large that limit. However, for the binary erasure channel, one *can* use a finite speed decoder and essentially memoryless encoder to control the state. [26] This leads us to wonder whether there is a way stabilize with finite speed decoding for arbitrary DMCs.

In this chapter, a simple encoding and decoding scheme was given for the problem of control over a noisy channel. The observer was nearly memoryless and the channel encoder was an infinite constraint length convolutional code. The decoder was a stack algorithm sequential decoder that had a bounded complexity keeping in line with the idea that a stable system should not be increasingly difficult to control. We showed simulation results to verify the theory and show practicality of the concept. Finally we attempted to analyze the tradeoff between complexity and reliability of the scheme. Future work in this topic can be focused on using the feedback inherent in the communication channel through the plant *in a simple manner* to boost performance. The room for improvement is large, as the 'focusing bound' of Sahai [24] shows that there is a large gap between the error exponents with delay for channels with and without feedback.

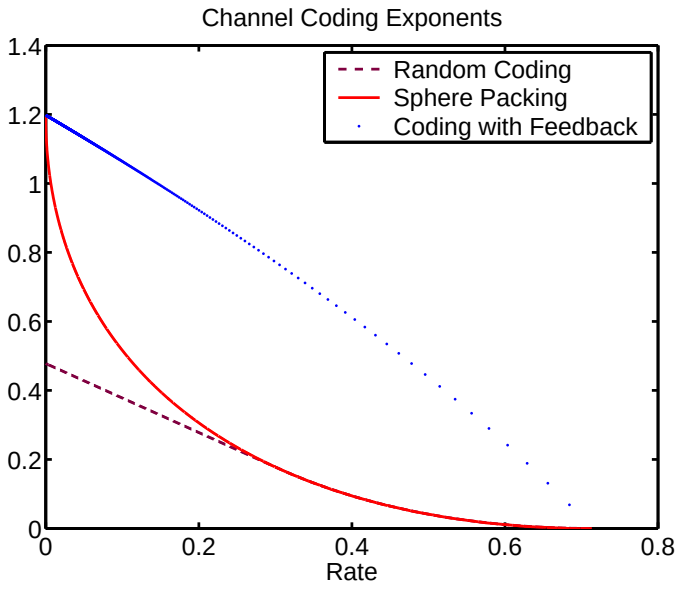


Figure 2.10. Channel coding exponents. There is much room for improvement between the sphere packing bound and the best known upper bound for channel coding with perfect feedback.

The control problem was the path along which this research started, but in retrospect perhaps the more logical starting point would have been channel coding with access to noiseless feedback. There is an intimate relationship between these two problems as discussed in [4]. If we had taken the feedback coding approach initially, we would think of our encoder with access to feedback and stack decoder as achieving an error exponent with delay equal to $E_r(R)$. This is not particularly interesting since this can be achieved without the feedback, but the advantage here is that we can drop the assumption of oracle access to the code at both encoder and decoder. That is, the encoder and decoder both need to know the code being used, and hence must perform some computation to get it. Since the codes have infinite constraint length, this means an increasing amount of computation with time at both encoder and decoder. If we think of the encoder as actually performing a convolution to obtain the code symbols, then encoding the error sequence would be preferable as we expect the probability of a 1 to decrease exponentially in delay. So after a short amount of time, the encoder need not calculate the convolution anymore since all remaining error bits are 0. Thus, an anytime feedback code is produced with low complexity encoding and decoding, although it does not come close to achieving the anytime capacity with feedback.

# Chapter 3

# Source coding with a stack decoder

In this chapter, a P-P source coding scheme using time-varying, random, infinite constraint length convolutional codes is described for which a sequential decoder is used. The sequential decoder is a stack algorithm that has a tunable bias parameter allowing for a tradeoff between reliability and computation, just as in Chapter 2. We then extend these results to the problem of lossless source coding with side information available at the decoder.

## 3.1 Problem definition

The source is modelled as a sequence of identical and independently distributed random variables $X_i$ that take on values from a finite alphabet $\mathcal{X}$. Each $X_i$ is drawn according to a probability mass function $Q(x)$.

Our goal is to code the symbols causally into a fixed rate bit stream so that the symbols can be recovered losslessly by a decoder in the sense that a symbol $X_i$ is recovered with probability 1 in the limit of large decoding delay. This is a weaker requirement than the notion of zero-error lossless coding achieved by Huffman coding, Lempel-Ziv coding, etc.

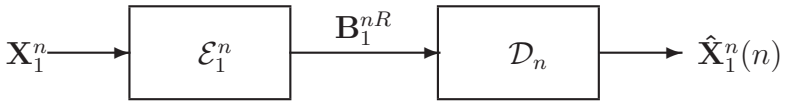


Figure 3.1. Point to point source coding. The encoder is causal if $B_i$ depends only on $X_1^{\lfloor i/R \rfloor}$. The decoder is causal if $\hat{X}_i(n)$ is a function of $B_1^{\lfloor nR \rfloor}$ for $n \geq i$.

Figure 3.1 shows the setup of our 'streaming' source coding problem. At a discrete time instant $n$, the encoder has access to the source realization up through time $n$, which is denoted $X_1^n$. Let the rate of the encoder be $R$ in bits per source symbol. The encoder at time $n$ outputs $\lfloor nR \rfloor - \lfloor (n-1)R \rfloor$ bits that are a function of $X_1^n$. Based on the bits $B_1^{\lfloor nR \rfloor}$, the decoder at time $n$ gives its estimate of the source symbols up through time $n$, denoted as $\hat{X}_1^{nR}$. We will use $X_i^j$ to denote the vector $(X_i, X_{i+1}, \ldots, X_j)$ if $i \leq j$ and the null string if $i > j$.

$$\mathcal{E}_n \quad : \quad \mathcal{X}^n \to \{0,1\}^{\lfloor nR \rfloor - \lfloor (n-1)R \rfloor} \tag{3.1}$$

$$B_{\lfloor (n-1)R \rfloor}^{\lfloor nR \rfloor} \quad = \quad \mathcal{E}_n(X_1^n) \tag{3.2}$$

$$\mathcal{D}_n \quad : \quad \{0,1\}^{nR} \to \mathcal{X}^n \tag{3.3}$$

$$\hat{X}_1^n(n) \quad = \quad \mathcal{D}_n(B_1^{\lfloor nR \rfloor}) \tag{3.4}$$

From now on, we assume that $R$ is an integer so we need not worry about integer effects [1] in the exposition, but the results hold for non-integer rates as well. Also, the only interesting values of $R$ lie in the interval $[H(Q), \log_2(|\mathcal{X}|)]$ since we need a rate of at least the entropy to losslessly encode the source, and if $R > \log_2(|\mathcal{X}|)$, we could just index the source sequences on a per-letter basis and losslessly recover them with no delay.

There are two measures of performance that we will evaluate. First is the probability of error with delay.

**Definition 3.1.1** *The* **probability of error with delay** $d$, $P_e(d)$, *is*

$$P_e(d) = \sup_n P(\hat{X}_1^n(n+d) \neq X_1^n) \tag{3.5}$$

[1] For a non-integer rate, the encoder outputs either $\lfloor R \rfloor$ or $\lceil R \rceil$ bits at every time instant. The integer effect is not important asymptotically, and for convenience we have used the assumption in proofs. In simulations, we have used non-integer rates.

*The probability is a measure over the randomness in the source and any randomness that may be present in the encoder or decoder. The* **error exponent with delay**, *or reliability exponent $E(R)$, at the rate $R$ where the encoder/decoder is operating is*

$$E(R) = \liminf_{d\to\infty} -\frac{1}{d}\log_2 P_e(d) \tag{3.6}$$

The second measure of performance lies in the random variable of computation. The motivation for developing sequential decoders has always been the opportunity to have a nearly optimal decoder without complexity that is exponentially growing in block length or delay. The amount of computation performed by our source decoder will be measured in the number of source sequences that are considered or compared against others.

We do not count the amount of 'internal' computation the decoder must do to determine the codewords of the source sequences. This is the same as the computation the encoder must perform to determine the codewords for the source sequences. That is, we assume an oracle gives the decoder any source codeword it wants. This is somewhat significant in our random tree code construction since the encoder is causal and its output depends on all previous source symbols. This means that as time increases, there is an increasing complexity to determining the bits assigned to a source symbol [2]. Note that this is not a problem in the control problem, because there we have used a quantizer, and all the trellis structure of the code as viewed by the decoder was in fact due to the control problem itself and not specific to the encoder.

**Definition 3.1.2** *If $x_n$ is the true source realization at time $n \geq 1$, the $i^{th}$* **incorrect subtree**, *$\mathcal{C}_i$, is*

$$\mathcal{C}_i = \left\{ y_1^n \in \mathcal{X}^n : n \geq i,\ y_1^{i-1} = x_1^{i-1} \text{ and } y_i \neq x_i \right\} \tag{3.7}$$

*This definition of incorrect subtree is well defined regardless of whether the encoder uses a tree code or not. The $i^{th}$* **random variable of computation**, *$N_i$, is the number of sequences in $\mathcal{C}_i$ that are ever examined by the decoder.*

[2] This effect could perhaps be mitigated with a specific encoder construction, but we have not considered this yet.

The definition of $N_i$ is a bit vague for arbitrary decoders but becomes concrete for sequential decoders. We now discuss the coding strategy used for this chapter and evaluate it.

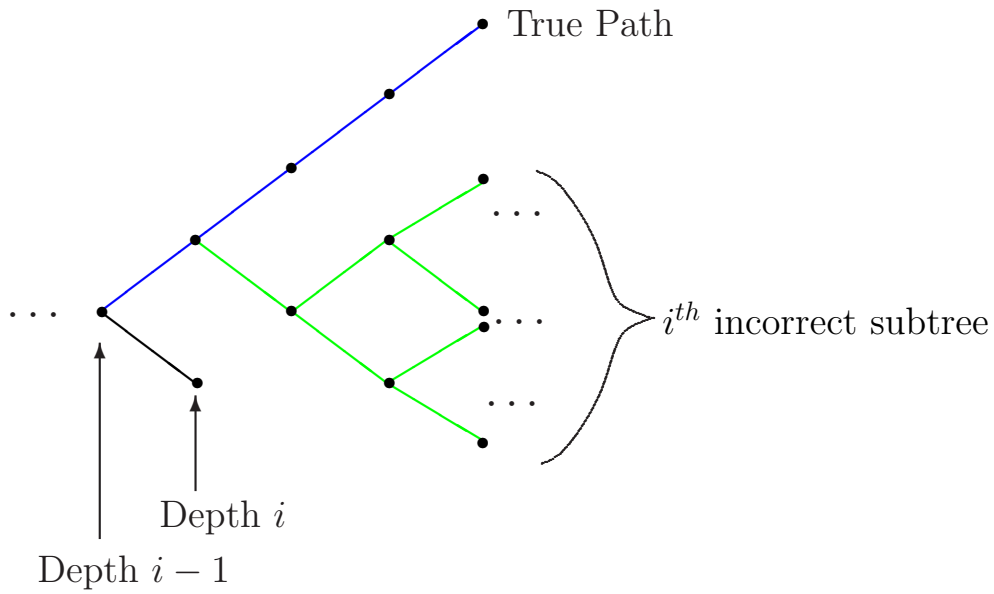


Figure 3.2. The $i^{th}$ incorrect subtree for a binary source.

## 3.2 A random binning scheme with a stack decoder

In this section, the encoder and decoder for the coding strategy of this chapter is introduced. The encoder used is the same as that in the sequential source coding paper of Sahai, Chang and Draper [27]. The bit sequence is arrived at by the use of a random tree code, which can be thought of as a time-varying, infinite constraint length convolutional code. Figure 3.3 shows an example of such a code. The tree begins with a root node from which $|\mathcal{X}|$ branches emanate, each branch corresponding to one of the symbols in the alphabet. The tree grows from each node in the same manner, with $|\mathcal{X}|$ branches going out of every node. On each branch in the tree are the $R$ bits to be sent to the decoder if the source symbol corresponding to the branch occurred at the time which is the depth of the branch in the tree. Equivalently, we can think of every node in the tree being labelled with $R$ bits

and the source sequence tracing a path through the tree and passing to the decoder the bits of the nodes the path goes through. The tree code is naturally a causal encoder.

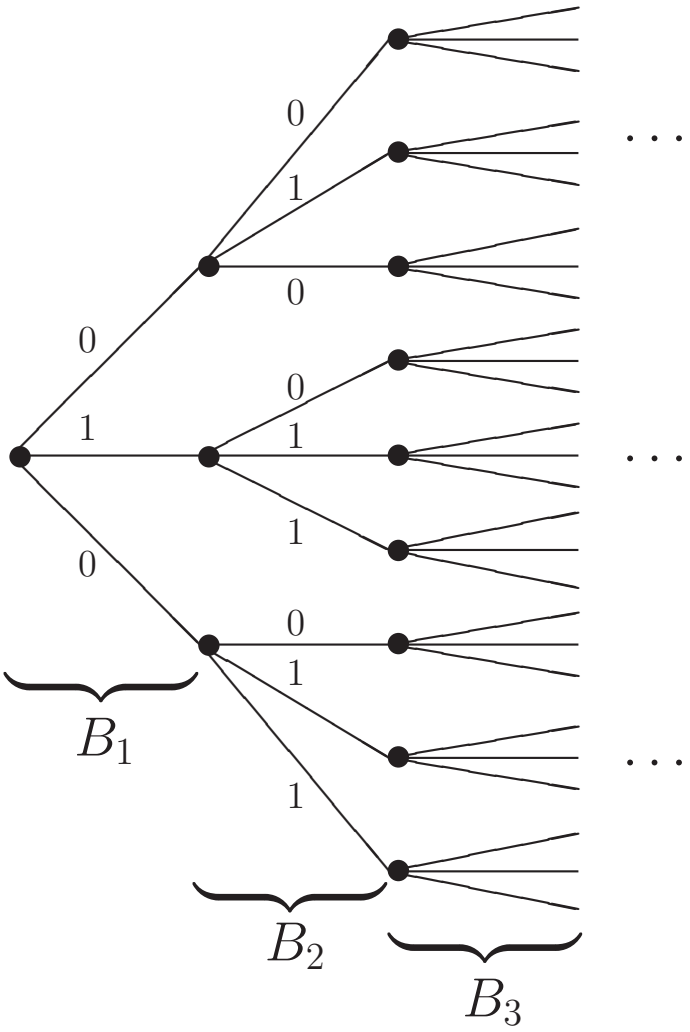


Figure 3.3. An example of a tree code for a source with ternary alphabet. Here the rate $R$ is one bit per source symbol.

The random binning scheme we use is an ensemble of tree codes, with every bit on every branch drawn identically and independently as Bernoulli 1/2 ($\mathcal{B}(1/2)$) random variables. This means that if source sequences $x_1^n$ and $y_1^n$ are the same until time $n-d+1$, i.e. $x_1^{n-d} = y_1^{n-d}$, but $x_{n-d+1} \neq y_{n-d+1}$, the probability that $x_1^n$ and $y_1^n$ are placed in the same 'bin' is $2^{-dR}$. This is because the last $dR$ bits of the codewords for $x_1^n$ and $y_1^n$ are drawn iid $\mathcal{B}(1/2)$. We refer to the bits in the codewords of source sequences as 'parities' because we think of them as coming from a time-varying, infinite constraint length convolutional code.

Decoding will be done by a stack algorithm, and hence is also sequential. We initialize the stack with the root node with a metric of 0. The following is the specific stack algorithm used.

1. Let $x_1^l$ denote the (partial) source sequence at the top of the stack. Remove $x_1^l$ from the stack and consider each of its $|\mathcal{X}|$ extensions by one symbol from $\mathcal{X}$, i.e. $(x_1^l, u)$, $\forall u \in \mathcal{X}$. Let $\tilde{x}_1^{l+1}$ be one of these extensions, and do the following for each of the extensions. If the parities of $\tilde{x}_1^{l+1}$ match the parities received, update the metric of $\tilde{x}_1^{l+1}$ and add it to the stack in a sorted way (highest metric on top). Otherwise discard $\tilde{x}_1^{l+1}$. Note that the parities of $\tilde{x}_1^{l+1}$ will match those received if and only if the label on the branch extending $x_1^l$ to $\tilde{x}_1^{l+1}$ match the $R$ parities received in the last time step.

2. Let $x_1^k$ denote the sequence on top of the stack after all the relevant extensions have been added. If the length of $x_1^k$, $k$, is up to the current time, declare $x$ as the decoded source sequence so far. Otherwise repeat 1.

The metrics are updated in an additive manner, with the metric of $\tilde{x}_1^{l+1}$ being $\Gamma(\tilde{x}_1^{l+1}) = \Gamma(\tilde{x}_1^l) + \Gamma(\tilde{x}_{l+1})$. For $x \in \mathcal{X}$, the metric for the source symbol $x$ is

$$\Gamma(x) = G + \log_2(Q(x))$$

The parameter $G$ is the 'bias' and controls to a large extent the amount of searching through the tree the algorithm performs. The reason for the $\log_2(Q(x))$ of course is that we want to have a near-MAP estimate of the source sequences whose parities agree with the source sequence received so far. The bias is used as a normalizer so that the true path through the tree has a metric that is slowly increasing in time. In contrast with the channel coding situation, false paths have metrics that may or may not be decreasing in time, but false paths are also knocked out of consideration by the parities. For example, if a source is $\mathcal{B}(1/3)$, the most probable sequence (the one with the highest metric) is the all-zero sequence, but it is highly atypical. Therefore, we count on the parities to cut these paths off. A bias equal to 0 is equivalent to MAP decoding because $\log_2(Q(x)) \leq 0$ for all $x \in \mathcal{X}$, hence all paths will have a decreasing metric in time, forcing the stack decoder to search the entire tree.

## 3.3 Bounds on computation and probability of error

In this section, we state two results regarding the computation and probability of error for our scheme. The theorems give conditions on the bias for which we can guarantee 'nearly-optimal' performance. The proofs are very similar to the proofs found in Jelinek's paper [1] on sequential decoding bounds for channel coding.

**Theorem 3.3.1** *Using the encoder and decoder of 3.2, we can achieve asymptotically the same probability of error as with MAP decoding. Define $E_s(\rho)$ as below, for $\rho \geq 0$.*

$$E_s(\rho) \triangleq (1+\rho)\log_2\left(\sum_{x\in\mathcal{X}} Q(x)^{\frac{1}{1+\rho}}\right) \tag{3.8}$$

*Let $\rho^*$ be defined so that rate $R$ in bits per source symbol of the code satisfies $R = \frac{d}{d\rho}E_s(\rho)|_{\rho=\rho^*}$. However, if $\rho^* > 1$, replace it with $\rho^* = 1$. If the bias of the sequential decoder, $G$, satisfies*

$$G < \frac{E_s(\rho^*)}{\rho^*} \tag{3.9}$$

*then for any $\epsilon > 0$, the probability of error with delay, $P_e(d)$, is at most*

$$P_e(d) \leq \tilde{K}_\epsilon 2^{-d(E_r(R)-\epsilon)} \tag{3.10}$$

*where $E_r(R) = \sup_{\rho\in[0,1]} \rho R - E_s(\rho)$, and $\tilde{K}_\epsilon$ is a finite constant independent of $d$. This shows that the error exponent with delay of this scheme is at least $E_r(R)$. The proof is given in A.2.*

Figure 3.4 shows an example of the exponent $E_r(R)$ for a ternary source with probabilities $(.95, .025, .025)$ and entropy .34 bits per symbol. Along with $E_r(R)$, a plot of the block source coding exponent is provided.

$$E_b(R) \triangleq \sup_{\rho\geq 0} \rho R - E_s(\rho) \tag{3.11}$$

As mentioned by Csiszar and Körner [28], the exponent $E_b(R)$, is an upper bound on the error exponent for block codes. The error event for our scheme for a delay $d$ is essentially the same as an error event for a block code with block length $d$. For low rates, $E_r(R) = E_b(R)$,

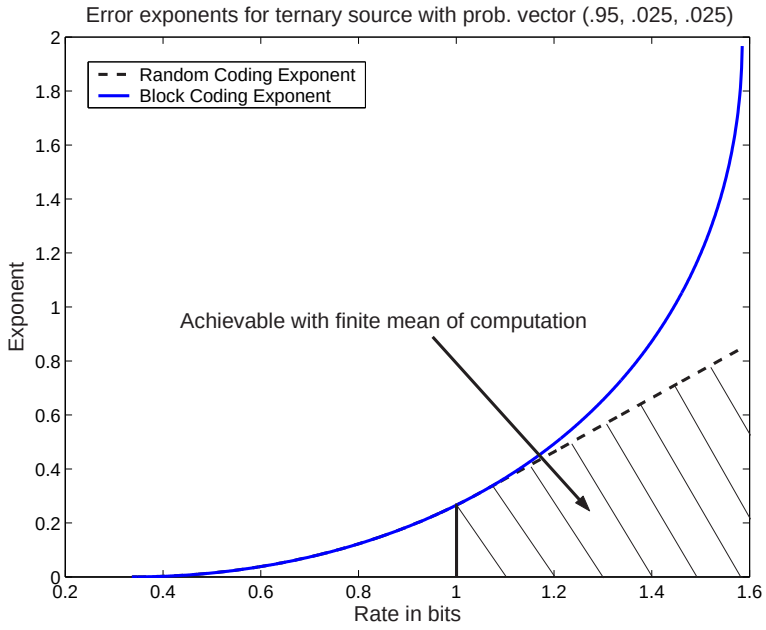


Figure 3.4. The random coding exponent and block coding exponent $E_b(R)$.

but at high rates $E_r(R) < E_b(R)$, with $E_b(R)$ going to infinity as the rate approaches $\log_2 |\mathcal{X}|$. As noted by Chang, et.al. [29], $E_b(R)$ is in general much less than the error exponent with decoding delay achievable with arbitrary codes.

**Theorem 3.3.2** *For the source coding scheme of section 3.2, if $\gamma \in [0,1]$, and the bias $G$ satisfies*

$$\frac{E_s(\gamma)}{\gamma} < G < \frac{1+\gamma}{\gamma}\left[\gamma R - G(\gamma)\right] \tag{3.12}$$

$$G(\gamma) \triangleq \gamma \log_2 \sum_{x \in \mathcal{X}} Q(x)^{1/(1+\gamma)} \tag{3.13}$$

*the $\gamma^{th}$ moment of $N_i$, $E[N_i^\gamma]$, averaged over the source and encoder randomness, is finite for all $i$ if*

$$R > \frac{1}{\gamma} E_s(\gamma) \tag{3.14}$$

*The proof is given in the appendix, section A.3.*

The range of bias values allowed for theorem 3.3.2 is always a nonempty interval if $R > E_s(\gamma)/\gamma$. Figure 3.5 shows the required rate as a function of $\gamma$ for an example source. If

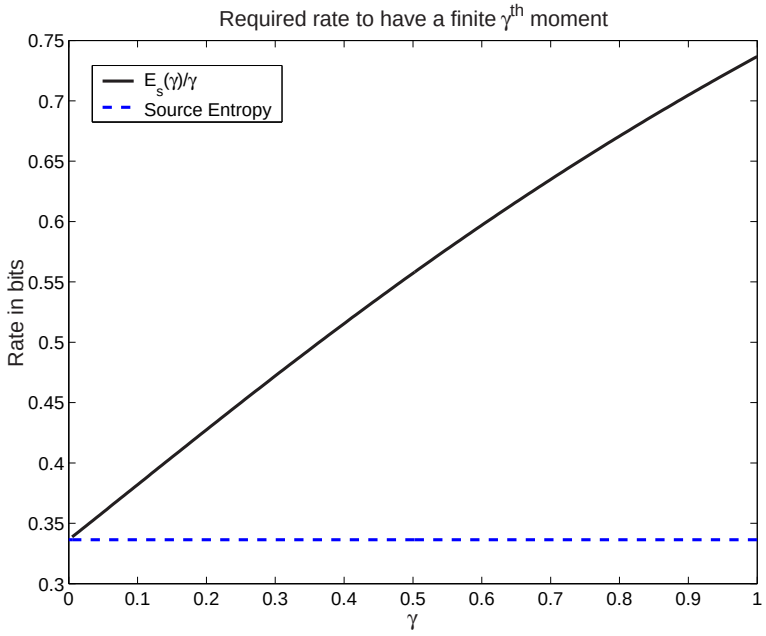


Figure 3.5. Rate required to have finite $\gamma^{th}$ moment of computation. The source is ternary with probabilities $(.95, .025, .025)$.

we want a finite mean for the computation, we would require the rate to be at least $E_s(1)$. Then for a source with pmf $Q$, we have a source coding *computation cutoff rate* of

$$R_{comp} \triangleq E_s(1) = 2\log_2\left(\sum_{x\in\mathcal{X}}\sqrt{Q(x)}\right) \tag{3.15}$$

The next section shows that if the rate is less than $R_{comp}$, then it is impossible for a sequential decoder to decode with exponentially decaying probability of error and still have a finite mean of computation.

## 3.4 Converse for computation in point-to-point sequential source coding

The work of Jacobs and Berlekamp [10] shows that when sequential decoding is used for communication over a noisy channel, the distribution of computation is at best a Pareto distribution. This implies that there will always be moments of the random variable of computation that are infinite. We will use a result of Arkian and Merhav from [2] to deduce

that a similar result holds for point-to-point source coding with sequential decoders. The result applies more generally, and we first state the point-to-point source coding problem as an instance of one in their class of problems.

Consider the problem of guessing the outcome of a source $U$ from a finite alphabet $\mathcal{U}$, which has a pmf $P$. A guessing strategy is an ordering of the elements of $\mathcal{U}$. We guess the value of $U$ by going down our list until we have guessed it correctly, at which point we stop. Let $G(U)$ be the number of guesses required to guess $U$, which is a function of $U$. Clearly, to minimize $E[G(U)^\rho]$, the $\rho^{th}$ moment of $G(U)$, for any $\rho > 0$, we just guess elements of $\mathcal{U}$ in decreasing order of their probability.

Now suppose $(U, Y)$ are drawn according to $P_{U,Y}$, we are given the realization $Y$, and we want to guess the realization of $U$ based on our knowledge of $Y$. We let $G(U|Y)$ denote the number of guesses to identify $U$ when $Y$ has occurred. This is a function of both $U$ and $Y$. The strategy to minimize moments of $G(U|Y)$ is of course to guess $U$ in order of decreasing probability $P_{U|Y}$.

Figure 3.6 shows an abstraction of a joint source-channel communication system. A discrete memoryless source outputs symbols from a finite alphabet $\mathcal{U}$ at a rate of one per second, according to a PMF $P$. For a fixed $N$, an encoder maps $N$ source symbols to $NR$ channel input symbols from a finite alphabet $\mathcal{X}$. A discrete memoryless channel with probability law $W(Y|X)$ corrupts the input to produce one output symbol $Y$ from a finite alphabet $\mathcal{Y}$ for every channel input symbol. The decoder is a guessing decoder, in that it makes guesses on the source block $U_1^N$ until it gets it correct. Let $G_N$ be a guessing decoder for the source with $N$ symbols drawn iid according to P. We now define $E_{sc}(\rho)$ to be

$$E_{sc}(\rho) \triangleq \lim_{N\to\infty} \min_{\mathcal{E}_N, G_N} \frac{1}{N} \log_2 E[G_N(U_1^N|Y_1^{NR})^\rho]$$

if the limit exists, where $\mathcal{E}_N$ and $G_N$ are respectively the source-channel encoder and the guessing decoder. We now define some quantities that appear in the bound of $E_{sc}(\rho)$.

**Definition 3.4.1** The **Renyi entropy of order** $\alpha$, $\alpha > 0, \alpha \neq 1$ of a source is

$$H_\alpha(P) = \frac{1}{1-\alpha} \log_2 \sum_{u\in\mathcal{U}} P(u)^\alpha$$

Let $E_0(\rho)$ be the Gallager function of the channel $W$ as defined in Chapter 2, that is

$$E_0(\rho) = \max_Q -\log_2 \sum_y \left[ \sum_x Q(x) W(y|x)^{\frac{1}{1+\rho}} \right]^{1+\rho}$$

Then Theorem 1 of Arikan and Merhav's paper [2] proves that for any $\rho > 0$.

$$E_{sc}(\rho) = [\rho H_{1/(1+\rho)}(P) - RE_0(\rho)]^+ \tag{3.16}$$

where $[x]^+ = \max(0, x)$. We note that if $RE_0(\rho) < \rho H_{1/(1+\rho)}(P)$, then the $\rho^{th}$ moment of

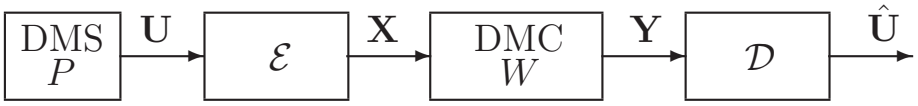


Figure 3.6. The joint source channel coding problem.

guesses for any guessing strategy goes to infinity exponentially with $N$. This will be used to get a condition on the rate of source encoding in our point to point source coding problem that is necessary (but not necessarily sufficient) to have a finite $\rho^{th}$ moment of computation per time.

Now going back to our stack decoding scheme for P-P source coding, we have seen that the probability that the stack decoder makes an error on $U_1^N$ at time $N + d$ goes to 0 exponentially with $d$. Hence, the stack decoder will identify the first $N$ source symbols with probability 1 as time goes on indefinitely. We can thus use the stack decoder to induce a guessing decoder. The guessing decoder is well defined because the stack decoder is a sequential decoder. Hence, its final determination (as $d$ goes to infinity) of $\hat{U}_1^N$ depends only on the portion of the received bits up until time $N$, i.e. $B_1^{NR}$. If we let $B_i = Y_i$ for all $i$, we note that $G(U_1^N|Y_1^{NR})$ gives us the number of nodes in the $N^{th}$ level of the encoding tree that are ever visited by the sequential decoder, which we define to be $D_N$. Therefore, $D_N$ can be taken as a lower bound to the number of nodes visited in order to correctly decode the first $N$ symbols. We want bounds on the moments of $D_N$ as they give approximations to the moments of $N_i$. In particular, we don't want $E[D_N^\rho]$ to increase exponentially as $N$ goes to $\infty$. Of course, this will only hold for certain $\rho$. We can evaluate equation 3.16 for this situation as follows.

Let the channel input alphabet and output alphabet be $\mathcal{X} = \mathcal{Y} = \{0, 1\}$. Let the channel be noiseless, so $W(y|x) = \delta(x - y)$, where $\delta$ is the Kronecker delta function. Hence, we are guaranteed that $X_i = Y_i = B_i$. We can now calculate $H_{1/(1+\rho)}(P)$ and $E_0(\rho)$.

$$\begin{aligned}
H_{1/(1+\rho)}(P) &= \frac{1}{1 - \frac{1}{1+\rho}} \log_2 \sum_{u \in \mathcal{U}} P(u)^{1/(1+\rho)} \\
&= \frac{1+\rho}{\rho} \log_2 \sum_{u} P(u)^{1/(1+\rho)} \\
E_0(\rho) &= \max_{Q} -\log_2 \sum_{y} \left[ \sum_{x} Q(x) W(y|x)^{1/(1+\rho)} \right]^{1+\rho} \\
&= \max_{q} -\log_2 (q^{1+\rho} + (1-q)^{1+\rho}) \\
&= -\log_2 (2 \cdot 2^{-(1+\rho)}) \\
&= \rho
\end{aligned}$$

Now by the theorem, the condition for the $\rho^{th}$ moment of $D_N$ to go to infinity exponentially with $N$ is $\rho H_{1/(1+\rho)}(P) - R E_0(\rho) > 0$. This simplifies to $(1+\rho) \log_2 \sum_u P(u)^{1/(1+\rho)} - \rho R > 0$. Hence, the $\rho^{th}$ moment of $D_N$ goes to infinity exponentially with $N$ if $R < E_s(\rho)/\rho$.

If the $\rho^{th}$ moment of $D_N$ goes to infinity exponentially with $N$, since the decoder eventually visits all these nodes it follows that $E[N_i^\rho]$ is infinite for some $i$ as well. In order for there to be any hope of having the $\rho^{th}$ moment of computation be finite, the rate must be at least $E_s(\rho)/\rho$. This is analogous to the requirement that the rate can be at most $E_0(\rho)/\rho$ in the channel coding case for a sequential decoder to have a finite $\rho^{th}$ moment of computation. The following is a concise statement of the contents of this section.

**Theorem 3.4.2** For any $\gamma > 0$, if a sequential decoder is used to decode a source code and the probability it will correctly determine every source symbol is exponentially decreasing in decoding delay, then the $\gamma^{th}$ moment of $N_i$, is infinite if

$$R < \frac{E_s(\gamma)}{\gamma} \tag{3.17}$$

## 3.5 Simulations

Just as in Chapter 2 we use Monte-Carlo simulation methods to verify the theoretical results of section 3.3, and to get a handle on some of the quantities related to the performance of the scheme.

**Example 3.5.1** Consider a ternary iid source $X_i \in \{0, 1, 2\}$. For this example we choose the probability mass function of the source to be $(.95, .025, .025)$. This source has an entropy of 0.34 bits per symbol.

In the simulation, the rate $R$ is 1 bit per source symbol. Figure 3.7 shows the relevant functions described in the bounds for reliability and computation. The probability of error with delay, $P_e(d)$ is the first quantity that we will look at experimentally. Since probability of error decays exponentially with delay, the *logarithm* of the probability of error decays linearly with delay. That is,

$$\log_2 P_e(d) \sim -E(R)d$$

Hence the slope of the line is exactly the negative of the error exponent achieved by this scheme. This is shown in Figure 3.8. By simulating the system with pseudo-random number generators, we have collected data on the probability of error with delay achieved by the sequential binning encoder with stack decoder. In 3.7, we see that at rate $R = 1$, the bias can be at most 0.6 to achieve the random coding exponent $E_r(1) = 0.27$. We have simulated with a bias equal to 0.5, and the linear fit in 3.8 shows that the experimentally obtained $E(1)$ is roughly 0.3. Since the linear fit is somewhat subjective, the value of 0.3 is close enough to 0.27 to show that they are roughly the same.

Further, if we assume that the moments of computation *at any time* are the same as the moments of computation *in any incorrect subtree*, we can compare the Pareto exponent of the simulation to the theory. In Figure 3.9, the logarithm of probability of the random variable of computation is plotted against the logarithm of the number of computations. Since the rate is 1, and $E_s(1) < 1$ in this case, theorem 3.3.2 tells us that it should be possible to have a Pareto exponent of at least 1. We conjecture that theorem 3.3.2 holds for

| | Theoretical | Experimental |
|---|---|---|
| Error exponent with delay | 0.27 | $\sim 0.3$ |
| Pareto exponent of computation | $> 1$ | $\sim 1.4$ |
| Maximum Pareto exponent | 2.1 | |

Table 3.1. Comparison of theoretical and experimental performance in example 3.5.1.

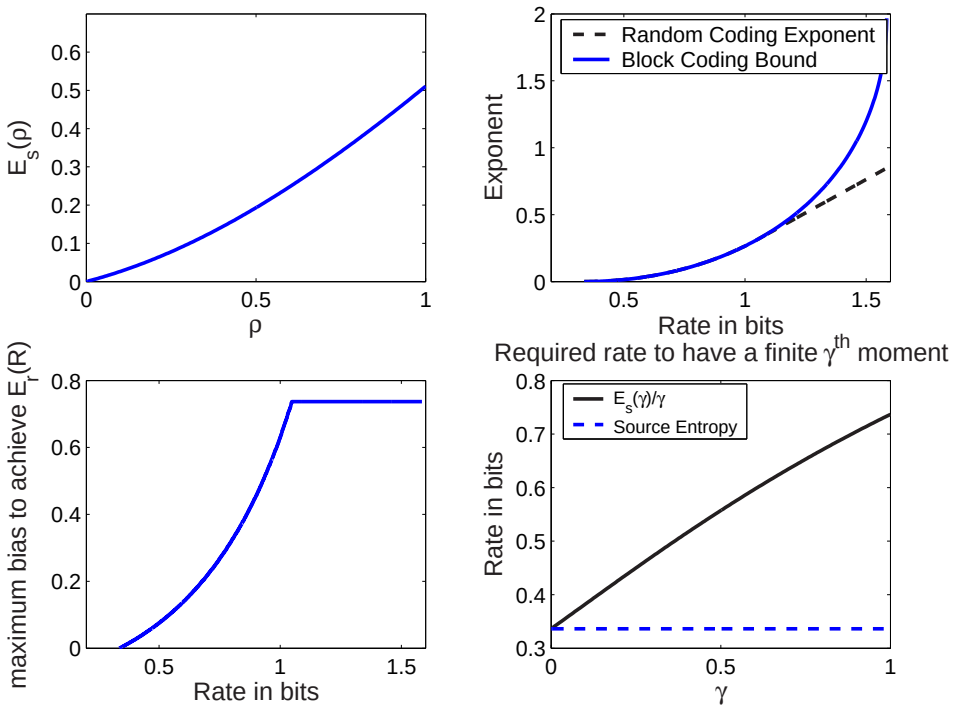


Figure 3.7. The functions of interest associated with a ternary source with probability vector $(.95, .025, .025)$.

moments greater than 1 as well. Since the bias is not set to maximize the Pareto exponent, we do not expect to achieve this conjectured Pareto exponent in this example. In Figure 3.9, the experimentally determined Pareto exponent is 1.2. So in this case, we have achieved the random coding error exponent with a finite mean in computation, as summarized in Table 3.1.

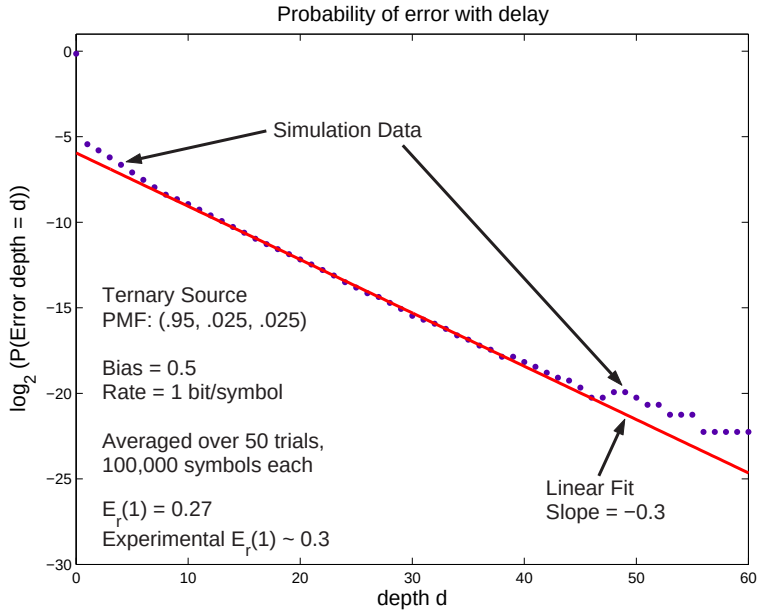


Figure 3.8. Experimentally determining $E_r(R)$, $R = 1$ bit per symbol, for example 3.5.1.

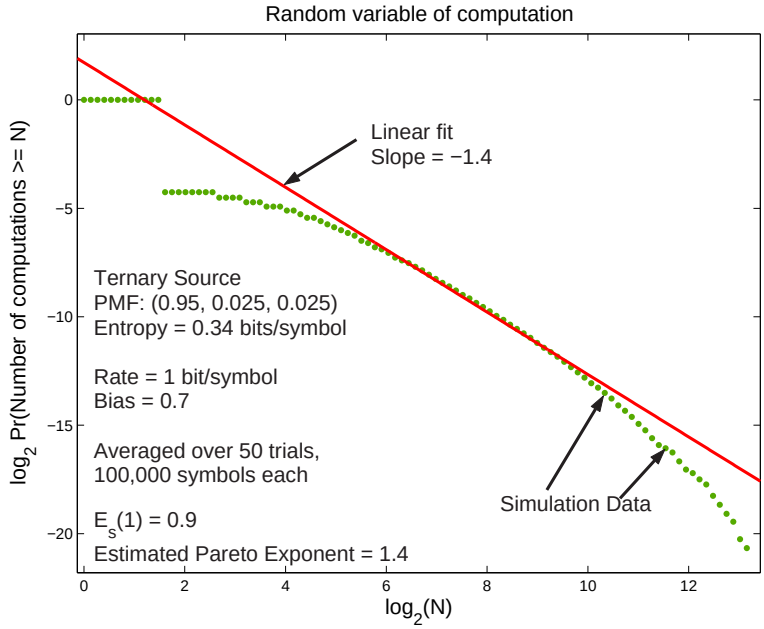


Figure 3.9. Experimentally determining the Pareto exponent for example 3.5.1.

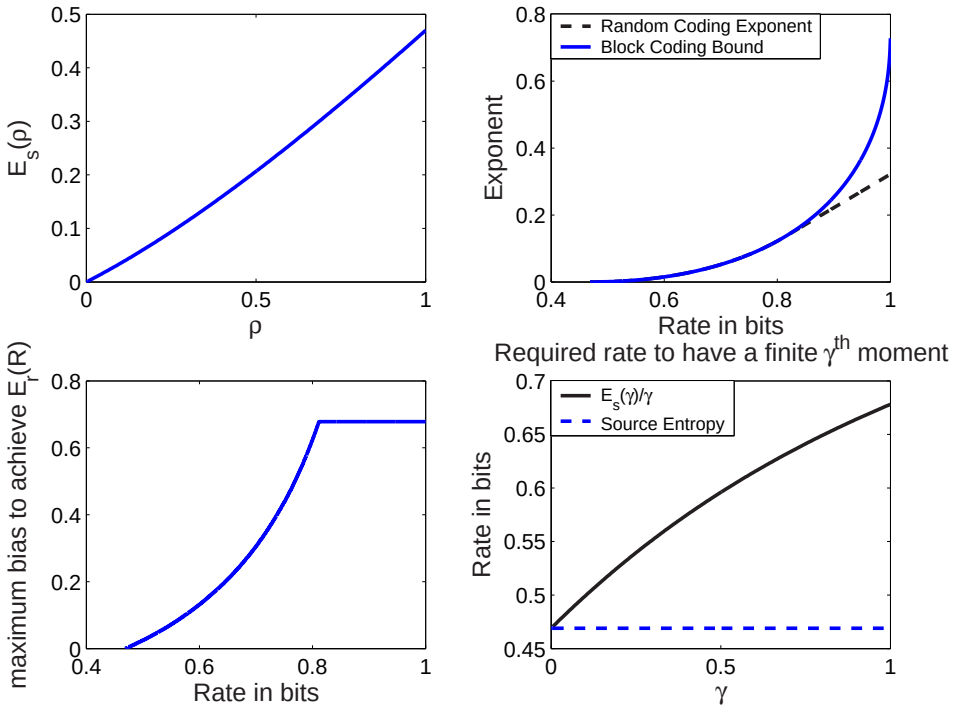


Figure 3.10. Functions of interest associated with a binary source with PMF $(0.9, 0.1)$.

**Example 3.5.2** Now we explore an example that will be comparable to the case when side information is available at the decoder only. The source $X_i$ is a sequence of iid Bernoulli one-half random bits. $Y_i$ are generated by passing $X_i$ through a BSC with crossover probability $\epsilon$. In this example, we consider the case when the side information is available at both the encoder and decoder. The situation is diagrammed in Figure 3.11. It is clear that since $Y$ is available at both the encoder and decoder, compressing $X \oplus Y$ is the same as compressing $X$. Figure 3.10 shows the relevant source coding functions for the error random variable $X \oplus Y$. Since we are just encoding the noise, the rate must be at least $H(X|Y) = H_b(\epsilon)$ where $H_b$ is the binary entropy function.

Again, we experimentally estimate the Pareto exponent of computation and the error exponent with delay. These are shown in Figures 3.13 and 3.12 respectively. Again, we see that we can achieve the random coding error exponent and the Pareto exponent guaranteed by theorem 3.3.2 holds. Since the bias value (0.7) is actually too high to achieve $E_r(R)$ at rate $R = 0.7$, the error exponent in the experiment is somewhat surprising. However,

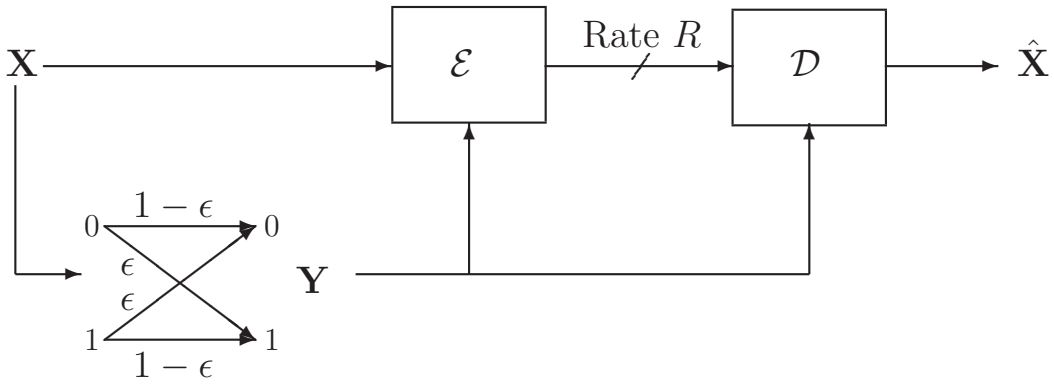


Figure 3.11. An example of point-to-point source coding that can be compared to source coding with side information at the decoder. $X_i$ are Bernoulli (1/2) random bits, $Y$ is $X$ passed through a BSC with crossover probability $\epsilon$. The encoder bins the error sequence $X \oplus Y$.

| | Theoretical | Experimental |
|---|---|---|
| Error exponent with delay | 0.05 | $\sim 0.06$ |
| Pareto exponent of computation | $> 1$ | $\sim 1.2$ |
| Maximum Pareto exponent | 1.2 | |

Table 3.2. Comparison of theoretical and experimental performance in example 3.5.2.

we stress again that the fitting of a line to the curve is somewhat arbitrary and we cannot expect to have precise values of the slope beyond the first digit.

## 3.6 Lossless source coding with side information

Now we consider source coding with side information available at the decoder. That is, two sources $(X_i, Y_i)$ are generated iid according to a joint distribution $P_{XY}$, $\{Y_i\}$ is available at the decoder, $\{X_i\}$ is available *causally* at the encoder and we want to encode $X_i$ into a bit stream to be recovered with the help of $\{Y_i\}$. Figure 3.6 shows the situation. The rate must be at least $H(X|Y)$ to recover $X$ losslessly, even though $Y$ is not available at the encoder.

We will use exactly the same encoder as in section 3.2. The decoder will still be the stack decoder, but the metric assignments now will change. If $Y_i$ is the side information at time $i$, the decoder assigns a symbol $x \in \mathcal{X}$ at time $i$ the metric $\log_2 P(x|Y_i) + G$. There is not

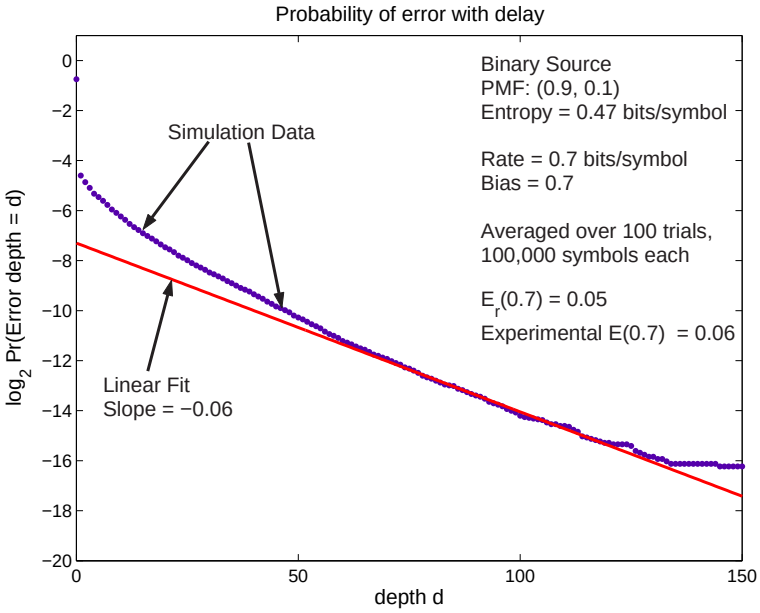


Figure 3.12. Estimating $E(R)$ for example 3.5.2.

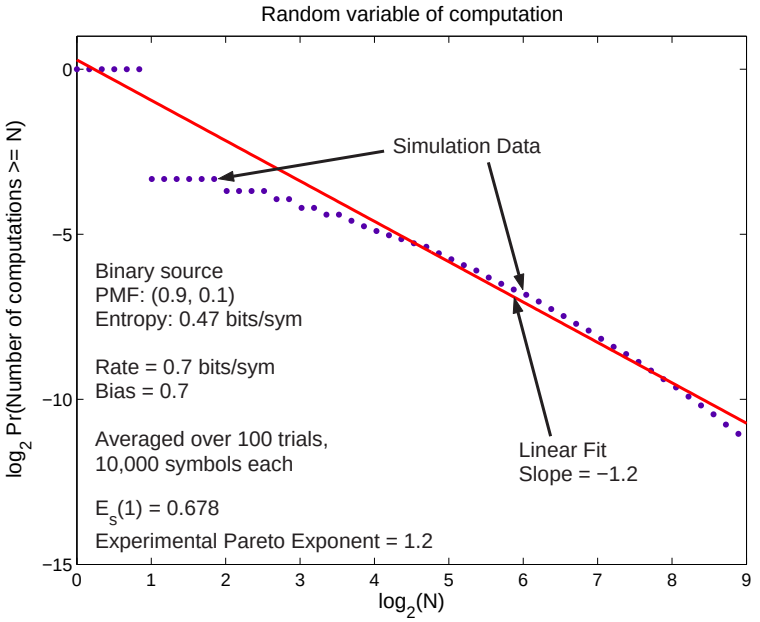


Figure 3.13. Estimating the Pareto exponent for computation for example 3.5.2.

much difference between this setup and point-to-point lossless source coding. Using exactly the same techniques as before, we can prove that the stack decoder with side information achieves the same error exponent as the MAP decoder with side information. This theorem is the side information analog of Theorem 3.3.1.

**Theorem 3.6.1** *Let the source pair $(X_i, Y_i)$ be generated iid according to a distribution $Q(x, y)$ on a finite set $\mathcal{X} \times \mathcal{Y}$. Using the encoder of 3.2 and the modified stack decoder just described, we can achieve asymptotically the same probability of error as with MAP decoding. Define $E_{si}(\rho)$ as below, for $\rho \geq 0$.*

$$E_{si}(\rho) \triangleq \log_2 \sum_{y \in \mathcal{Y}} \left( \sum_{x \in \mathcal{X}} Q(x, y)^{\frac{1}{1+\rho}} \right)^{1+\rho} \tag{3.18}$$

*Let $\rho^*$ be defined so that rate $R$ in bits per source symbol of the code satisfies $R = \frac{d}{d\rho} E_{si}(\rho)|_{\rho=\rho^*}$. However, if $\rho^* > 1$, replace it with $\rho^* = 1$. If the bias of the sequential decoder, $G$, satisfies*

$$G \leq \frac{1+\rho^*}{\rho^*} \left[ E_{si}(\rho^*) - F_{si}(\rho^*) \right] \tag{3.19}$$

$$F_{si}(\rho) \triangleq \log_2 \sum_{y \in \mathcal{Y}} Q(y) \left( \sum_{x \in \mathcal{X}} Q(x|y)^{\frac{1}{1+\rho}} \right)^{\rho} \tag{3.20}$$

*then for any $\epsilon > 0$, the probability of error with delay, $P_e(d)$, is at most*

$$P_e(d) \leq \tilde{K}_\epsilon 2^{-d(E_r(R)-\epsilon)} \tag{3.21}$$

*where $E_r(R) = \sup_{\rho \in [0,1]} \rho R - E_{si}(\rho)$, and $\tilde{K}_\epsilon$ is a finite constant independent of $d$. This shows that the error exponent with delay of this scheme is at least $E_r(R)$. The proof is completely parallel to the proof in A.2, and the last few steps are shown in A.4.*

**Example 3.6.1** We go back to the binary source example, where the side information is generated by passing the source bit through a BSC. The side information this time is only available at the decoder, as is shown in Figure 3.15. In this case, the function $E_{si}(\rho)$

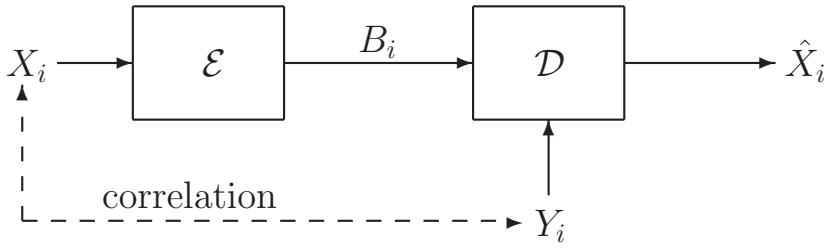


Figure 3.14. Source coding with side information. $X$ and $Y$ are related through some known correlation structure.

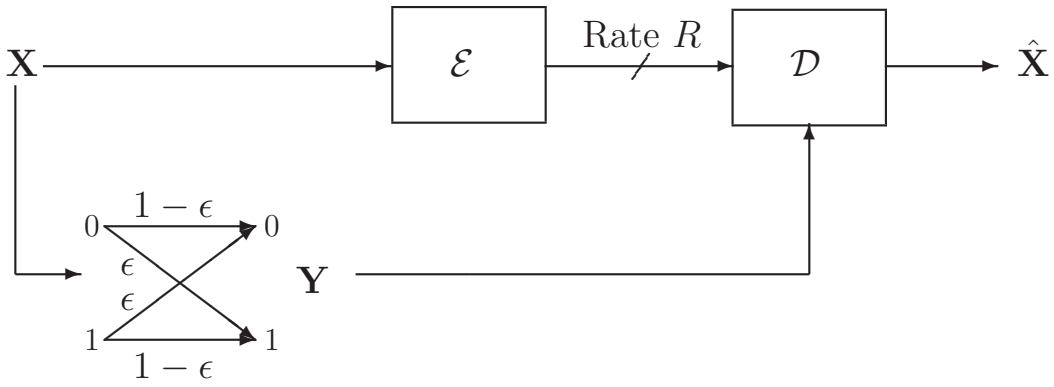


Figure 3.15. An example of lossless source coding with side information. $X_i$ are Bernoulli (1/2) random bits, $Y$ is $X$ passed through a BSC with crossover probability $\epsilon$. The encoder bins its observations of $X$.

| | Theoretical | Experimental |
|---|---|---|
| Error exponent with delay | 0.05 | $\sim 0.08$ |
| Pareto exponent of computation | $> 1$ | $\sim 1.2$ |
| Maximum Pareto exponent | 1.2 | |

Table 3.3. Comparison of theoretical and experimental performance in example 3.6.1.

simplifies as below,

$$\begin{aligned} E_{si}(\rho) &= \log_2 \sum_{y\in\mathcal{Y}} \left( \sum_{x\in\mathcal{X}} p(x,y)^{1/(1+\rho)} \right)^{1+\rho} \right) && (3.22) \\ &= \log_2 \sum_{y\in\mathcal{Y}} p(y) \left( \sum_{x\in\mathcal{X}} p(x|y)^{1/(1+\rho)} \right)^{1+\rho} \right) && (3.23) \\ &= \log_2 \sum_{y=0}^{1} \frac{1}{2} \left( \epsilon^{1/(1+\rho)} + (1-\epsilon)^{1/(1+\rho)} \right)^{1+\rho} && (3.24) \\ &= (1+\rho) \log_2 \left( \epsilon^{1/(1+\rho)} + (1-\epsilon)^{1/(1+\rho)} \right) && (3.25) \end{aligned}$$

This $E_{si}(\rho)$ is the same as the $E_s(\rho)$ function that appears if the side information $Y$ is available at both the encoder and decoder, i.e. point-to-point coding of the error sequence. To compare to the case when $Y$ is available at the encoder as well, we estimate the error exponent with delay and the Pareto exponent for computation through simulation in Figures 3.16 and 3.17 respectively. In this simulation, the rate is once again 0.7 bits per symbol, and the bias is 0.7. Again we see nearly identical values for the error exponent and Pareto exponent, as we should. In theory, they should be exactly identical, but we have simulated with differing randomness in the two simulations.

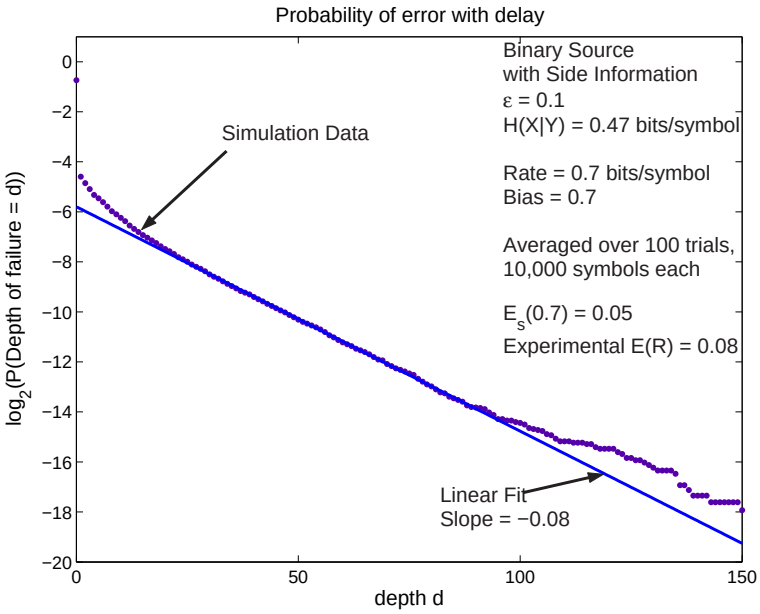


Figure 3.16. Determining $E(R)$ for example 3.6.1.

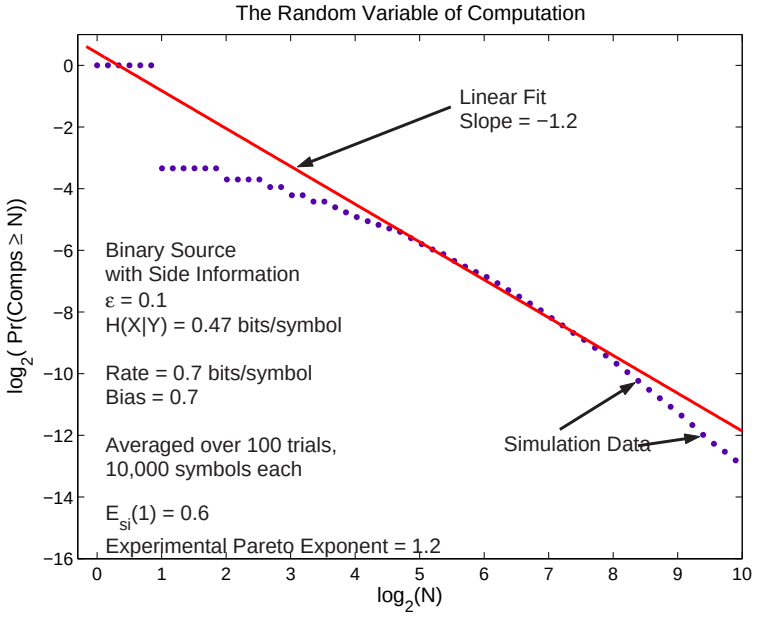


Figure 3.17. Determining the Pareto exponent for computation for example 3.6.1.

# Chapter 4

# Conclusion

Two problems were considered for this thesis. The aim of each was to take the first steps towards practical anytime coding strategies. This was done by using the well-developed theory of convolutional codes and sequential decoding. Sequential decoding offers near ML or MAP reliability for a reasonable computational cost.

The first problem was stabilization over a discrete memoryless channel. We considered a scalar linear system with unstable eigenvalue and bounded disturbances. The nature of the problem fundamentally calls for anytime coding. We chose to restrict the observer to be essentially memoryless, and constructed an anytime code from an infinite constraint length convolutional code, i.e. a tree code. The encoder was left to be random so that the performance could be averaged for analytical tractability, as is generally the case in information theory.

The significant contribution of this half of the thesis was the application of the stack algorithm, a sequential decoder, to the decoder located in the controller. We used results showing that sequential decoding can achieve the same error exponent with delay as ML decoding. The decoder had a parameter, called the bias, that could be used to tradeoff performance with complexity. Results of Jelinek were then used to state a few analytical results about the tradeoff. We showed through simulation that the encoding and decoding strategy was indeed practical and the analytical results were verified by the Monte-Carlo

method. Thus, we have shown a baseline scheme for comparison with strategies proposed in the future. It is clear that we have paid quite a lot for the practicality of our scheme. There is much room for improvement and future work should key in on how to properly use feedback in a simple manner, so as to improve reliability. It would seem that focus should shift to designing better observer/encoders to monitor the state in a non-uniform way, both in time and scale.

The second half of the thesis focused on the anytime analog for source coding. We modelled a source sequence as being made of iid symbols and again used an infinite constraint length convolutional code to encode. The decoder was again a stack algorithm sequential decoder with a variable bias. Several results were then proved about this decoder. First, that it could achieve the same error exponent as MAP decoding, and second, that it could also have computation that was not growing on average with time. Then, we saw that the computational bound was essentially as good as we could do with any sequential decoder that had an exponentially decaying probability of error with delay. This followed as an application of a theorem of Arikan. Next, we showed simulations of this strategy and verified the theory.

The motivation of the point-to-point anytime source coding strategy was to arrive at a practical anytime source code for lossless source coding with side information at the decoder. We used the same sequential binning encoder as in the point-to-point problem, and verified by simulation that the stack decoder gives exponentially decreasing probability of error with delay. We proved a theorem showing that the stack decoder is asymptotically as reliable as a MAP decoder when side information is available at the decoder. If time had permitted, we would have further explored the use of a sequential decoder for fully distributed lossless source coding, sometimes called the Slepian-Wolf problem. It would seem that the approach used in the side information case provides a stepping stone to the Slepian-Wolf problem.

The main contributions of this thesis have been in using the stack algorithm to get delay-universal sequential decoders for the problems of control over a noisy channel and sequential lossless source coding.

# References


[1] F. Jelinek, "Upper bounds on sequential decoding performance parameters," *IEEE Transactions on Information Theory*, vol. 20, pp. 227–239, Mar. 1974.

[2] E. Arikan and N. Merhav, "Joint source-channel coding and guessing with application to sequential decoding," *IEEE Transactions on Information Theory*, vol. 44, Sept. 1998.

[3] A. Sahai, "Anytime information theory," Ph.D. dissertation, Massachussetts Institute of Technology, Cambridge,MA, 2001.

[4] A. Sahai and S. Mitter, "The necessity and sufficiency of anytime capacity for control over a noisy communication link: Part 1," To appear in IEEE Transactions on Information Theory, Aug 2006.

[5] A. Sahai and H. Palaiyanur, "A simple encoding and decoding strategy for stabilization discrete memoryless channels," in *Forty-third Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Sept. 2005.

[6] Y. Ho, M. Kastner, and E. Wong, "Teams, signaling, and information theory," *IEEE Transactions on Automatic Control*, vol. 23, pp. 305–312, Apr. 1978.

[7] H. Witsenhausen, "Separation of estimation and control for discrete timne systems," *Proceedings of the IEEE*, vol. 59, Nov. 1971.

[8] A. Sahai, "The necessity and sufficieny of anytime capacity for control over a noisy communication link," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, Dec. 2004.

[9] R. Gallager, *Information Theory and Reliable Communication*. New York,NY: John Wiley and Sons, 1971.

[10] I. Jacobs and E. Berlekamp, "A lower bound to the distribution of computation for sequential decoding," *IEEE Transactions on Information Theory*, vol. 13, pp. 167–174, Apr. 1967.

[11] J. Savage, "The distribution of sequential decoding computation time," *IEEE Transactions on Information Theory*, vol. 12, Apr. 1966.

[12] F. Jelinek, "An upper bound on moments of sequential decoding effort," *IEEE Transactions on Information Theory*, vol. 15, pp. 140–149, Jan. 1969.

[13] G. Forney, "Convolutional codes 3. sequential decoding." *Information and Control*, vol. 25, pp. 267–297, 1974.

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

[15] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. 23, pp. 337–343, May 1977.

[16] ——, "Compression of individual sequences by variable rate coding," *IEEE Transactions on Information Theory*, vol. 24, pp. 530–536, Sept. 1978.

[17] M. Burrows and D. Wheeler, "A block-sorting lossless data compression algorithm," Digital Equipment Corporation, Tech. Rep. 124, 1994.

[18] M. Hellman, "Convolutional source encoding," *IEEE Transactions on Information Theory*, vol. 21, pp. 651–656, Nov. 1975.

[19] V. Koshelev, "Direct sequential encoding and decoding for discrete sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 340–343, May 1973.

[20] F. Jelinek, "Tree encoding of memoryless discrete-time sources with a fidelity criterion," *IEEE Transactions on Information Theory*, vol. 15, pp. 584–590, Sept. 1969.

[21] S. Mohan and J. Anderson, "Speech encoding by a stack algorithm," *IEEE Transactions on Communications*, vol. 28, pp. 825–830, June 1980.

[22] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, July 1973.

[23] C. Chang and A. Sahai, "Upper bound on error exponents with delay for lossless source coding with side-information," in *Proc. Int. Symp. Inform. Theory*, Seattle, WA, USA, July 2006.

[24] A. Sahai, "Why block length and delay are not the same thing," Submitted to IEEE Transactions on Information Theory. [Online]. Available: www.eecs.berkeley.edu/~sahai/Papers/FocusingBound.pdf

[25] E. Arikan, "An inequality on guessing and its application to sequential decoding," *IEEE Transactions on Information Theory*, vol. 42, pp. 99–105, Jan. 1996.

[26] A. Sahai, "Evaluating channels for control: capacity reconsidered," in *Proceedings of the 2000 American Control Conference*, Chicago, IL, June 2000.

[27] C. Chang, S. Draper, and A. Sahai, "Random sequential binning for distributed source coding," in *Proc. Int. Symp. Inform. Theory*, Adelaide, Australia, Sept. 2005.

[28] I. Csiszar and J. Korner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*, 2nd ed. New York, NY: Academic Press, 1997.

[29] C. Chang and A. Sahai, "The error exponent with delay for lossless source coding," in *IEEE Information Theory Workshop*, Punta del Este, Uruguay, 2006.

# Appendix A

# Appendix

## A.1 The probability of error with delay for channel coding with a sequential decoder

The sequential decoder (using the Stack Algorithm) decides on the bin of $X(kT)$ at time $kT + T - 1$ having received all channel outputs through time $kT + T - 1$. Let the sequential decoder's decoded path for time $kT$ be denoted as $s(k)$. Let the true path taken by the state be denoted as $x$. We will use $u$ to denote any path in the trellis other than the true path. We note that $s(k)$ can change drastically as time evolves. That is, $s(k)$ and $s(k+1)$ may or may not be merged for a large portion of time. This is because the Stack Algorithm is allowed to back up and try different paths in the trellis if for some reason the best available paths have extending branches with very low metrics.

In order to prove the theorem, we will bound the probability that $s(k)$ diverged from the true path $d$ time branches in the past. We will show this probability decays exponentially with an exponent of at least the random block-coding error exponent to within a constant. This result was proved by Jelinek in [1], and we will provide a somewhat detailed proof. First, we define the error event to be analyzed.

$$E_d(k) \triangleq \{s(k) \text{ and } x \text{ have their last common node at time } (k-d)T \}$$

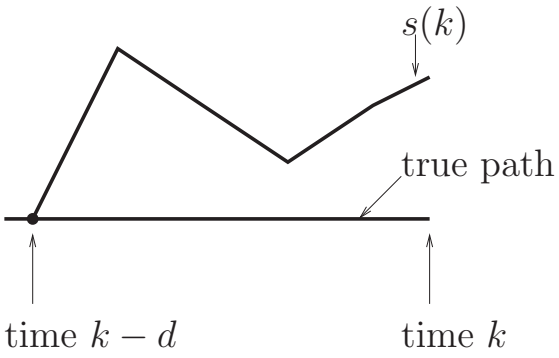


Figure A.1. An example of when $E_d(k)$ occurs.

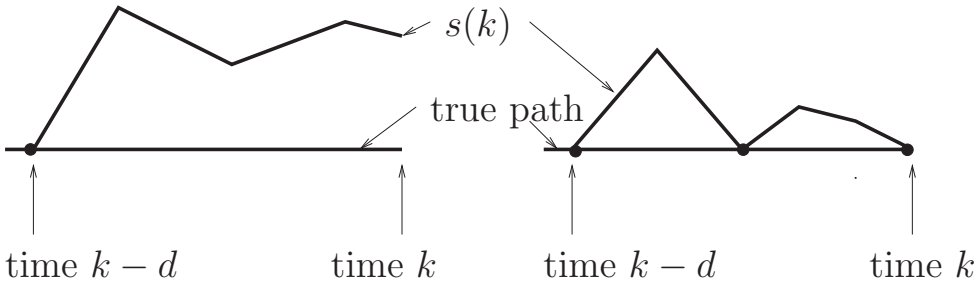


Figure A.2. Examples of when $A_d(k)$ occurs, illustrating the difference with $E_d(k)$.

$$A_d(k) \triangleq \{s(k) \text{ and } x \text{ are merged at time } (k-d)T\}$$

The error events are depicted in Figures A.1 and A.2. We note that the only difference between the two is that an $s(k)$ that causes $A_d(k)$ to occur may merge with the true path after time $k-d$ while one the causes $E_d(k)$ to occur may not. With these definitions, it is clear that $E_d(k) \subset A_d(k)$.

$$\begin{aligned} \Rightarrow P(E_d(k)) &= P(E_d(k)|A_d(k)) \cdot P(A_d(k)) \\ &\leq P(E_d(k)|A_d(k)) \end{aligned}$$

From now on, we only refer to time in multiples of $T$. Now, given that $A_d(k)$ occurs, $s(k)$ and $x$ have the same metric at time $(k-d)$ (meaning at time up til the node at time $(k-d)T$). For $s(k)$ to beat $x$ at time $k$, we must have that the metric of $s(k)_{k-d+1}^{k}$ is better than the minimum of the partial path metrics of the true path starting from $k-d+1$ and

going on until $k-d+j, j=1,2,\ldots,d$. That is, we must have the following event:

$$F_d(k) \triangleq \left\{\exists u_{k-d+1}^k \text{ disjoint with } x_{k-d+1}^k \text{ such that } \Gamma(u_{k-d+1}^k) \geq \min_{1\leq j\leq d} \Gamma(x_{k-d+1}^{k-d+j})\right\}$$

Given $A_d(k)$, $E_d(k) \subset F_d(k)$, so we have

$$\begin{aligned} P(E_d(k)) &\leq P(E_d(k)|A_d(k)) \\ &\leq P(F_d(k)) \end{aligned}$$

Now, let $F_d \triangleq F_d(d)$. Since the code is i.i.d. across time, $P(F_d(k)) = P(F_d)$, if $k \geq d$. Hence we can assume that the last common node between $s(k)$ and the true path is the 'root' of the trellis. We give a few more definitions to further break down the proof into smaller pieces.

$$\begin{aligned} C_d &\triangleq \left\{u_1^d \text{ in trellis: } u_1^d \text{ is disjoint with } x_1^d\right\} \\ F_d &\triangleq \left\{\exists u_1^d \in C_d : \Gamma(u_1^d) \geq \min_{1\leq j\leq d} \Gamma(x_1^j)\right\} \\ F_{d,j} &\triangleq \left\{\exists u_1^d \in C_d : \Gamma(u_1^d) \geq \Gamma(x_1^j)\right\} \\ \Rightarrow F_d &= \bigcup_{j=1}^{d} F_{d,j} \\ P(F_d) &\leq \sum_{j=1}^{d} P(F_{d,j}) \end{aligned}$$

Now, suppose $\Gamma(u_1^d) \geq \Gamma(x_1^j)$, $j \geq 1$. With $G$ as the bias used by the stack decoder, this implies

$$\begin{aligned} \left[\sum_{i=1}^{dT-1} \ln \frac{p(y_i|u_i)}{p(y_i)} - G\right] - \left[\sum_{i=1}^{jT-1} \ln \frac{p(y_i|x_i)}{p(y_i)} - G\right] &\geq 0 \\ \ln \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} + (j-d)TG &\geq 0 \\ \Rightarrow \ln \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \cdot \exp((j-d)TG) &\geq 1 \\ \left[\frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)}\right]^s \cdot \exp(s(j-d)TG) &\geq 1 \\ \text{for } s \geq 0 \end{aligned}$$

Now, consider the quantities $\tau_{d,j}$ and $\kappa_{d,j}$ defined below.

$$\begin{aligned}
\tau_{d,j} &\triangleq \Big( \sum_{u_1^d \in C_d} \Big[ \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big]^s \cdot \exp(s(j-d)TG) \Big)^\rho \\
&= \exp(s\rho(j-d)TG) \Big( \sum_{u_1^d \in C_d} \Big[ \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big]^s \Big)^\rho \\
&= \exp(s\rho(j-d)TG)\kappa_{d,j} \\
&\quad \text{with } \rho \in [0,1] \text{ and } s \geq 0. \\
\kappa_{d,j} &\triangleq \Big( \sum_{u_1^d \in C_d} \Big[ \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big]^s \Big)^\rho
\end{aligned}$$

By the construction of $\tau_{d,j}$, if $F_{d,j}$ occurs, $\tau_{d,j} \geq 1$. Since $P(F_{d,j}) = E[1(F_{d,j})]$ and $\tau_{d,j} \geq 1(F_{d,j})$, $P(F_{d,j}) \leq E[\tau_{d,j}]$. Hence, $P(F_d) \leq \sum_{j=1}^d E[\tau_{d,j}]$.

$$\begin{aligned}
E[\tau_{d,j}] &= \sum_{x_1^d} \sum_{y_1^d} p(x_1^d)p(y_1^d|x_1^d)E[\tau_{d,j}|x_1^d, y_1^d] && \text{(A.1)} \\
&= \exp(s\rho(j-d)TG) \sum_{x_1^d} \sum_{y_1^d} p(x_1^{d+1})p(y_1^d|x_1^d)E[\kappa_{d,j}|x_1^d, y_1^d] && \text{(A.2)} \\
E[\kappa_{d,j}|x_1^d, y_1^d] &= E\Big[ \Big( \sum_{u_1^d \in C_d} \Big[ \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big]^s \Big)^\rho \Big| x_1^d, y_1^d \Big] && \text{(A.3)} \\
&\leq |C_d|^\rho E\Big[ \Big( \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big)^s \Big| x_1^d, y_1^d \Big]^\rho, \ u_1^d \in C_d && \text{(A.4)} \\
&\leq \exp(\rho dTR) \cdot E\Big[ \Big( \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big)^s \Big| x_1^d, y_1^d \Big]^\rho && \text{(A.5)}
\end{aligned}$$

Equation A.3 follows from Jensen's inequality because $\rho \in [0,1]$. Equation A.4 is due to the fact that the channel symbols along all the branches of paths in the set $C_d$ are generated iid, so the expectation of the sum is the expectation of one multiplied by the number of items in the sum. Finally, because the trellis branches at a rate $R$, the size of $C_d$ is at most $\exp(dTR)$, which yields Equation A.5. Now we average over the random codeword symbol generation.

$$\begin{aligned}
E\Big[ \Big( \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big)^s \Big| x_1^d, y_1^d \Big] &= \sum_{u_1^d \in \mathcal{X}^{dT}} p(u_1^d) \Big( \frac{p(y_1^d|u_1^d)p(y_1^j)}{p(y_1^j|x_1^j)p(y_1^d)} \Big)^s && \text{(A.6)} \\
&= \sum_{u_1^d} p(u_1^d) \Big( \frac{p(y_1^j|u_1^j)p(y_{j+1}^d|u_{j+1}^d)}{p(y_1^j|x_1^j)p(y_{j+1}^d)} \Big)^s && \text{(A.7)}
\end{aligned}$$

The change in Equation A.7 is that we have cancelled some terms and split $p(y_1^d|u_1^d)$ into two terms.

$$\begin{aligned}
P(F_{d,j}) &\leq \exp(\rho dTR + s\rho(j-d)TG) \\
&\quad \cdot \sum_{x_1^d}\sum_{y_1^d} p(x_1^d)p(y_1^d|x_1^d)\Big[\sum_{u_1^d} p(u_1^d)\Big(\frac{p(y_1^j|u_1^j)p(y_{j+1}^d|u_{j+1}^d)}{p(y_1^j|x_1^j)p(y_{j+1}^d)}\Big)^s\Big]^\rho \\
\text{let } \alpha_{d,j} &\triangleq \exp(\rho dTR + s\rho(j-d)TG) \\
\text{let } \zeta_{d,j} &\triangleq \sum_{x_1^d}\sum_{y_1^d} p(x_1^d)p(y_1^d|x_1^d)\Big[\sum_{u_1^d} p(u_1^d)\Big[\frac{p(y_1^j|u_1^j)p(y_{j+1}^d|u_{j+1}^d)}{p(y_1^j|x_1^j)p(y_{j+1}^d)}\Big]^s\Big]^\rho \\
P(F_{d,j}) &\leq \alpha_{d,j}\cdot\zeta_{d,j}
\end{aligned}$$

Now we simplify $\zeta_{d,j}$ using the iid nature of the encoder and the memorylessness of the channel.

$$\begin{aligned}
\zeta_{d,j} &= \sum_{x_1^d}\sum_{y_1^d} p(x_1^d)p(y_1^d|x_1^d)\Big[\sum_{u_1^d} p(u_1^d)\Big(\frac{p(y_1^j|u_1^j)p(y_{j+1}^d|u_{j+1}^d)}{p(y_1^j|x_1^j)p(y_{j+1}^d)}\Big)^s\Big]^\rho \\
&= \sum_{x_1^d}\sum_{y_1^d} p(x_1^d)p(y_1^d|x_1^d)p(y_1^j|x_1^j)^{-s\rho}p(y_{j+1}^d)^{-s\rho}\Big[\sum_{u_1^d} p(u_1^d)\Big(p(y_1^j|u_1^j)p(y_{j+1}^d|u_{j+1}^d)\Big)^s\Big]^\rho \\
&= \exp(jTf_c(s,\rho) + (d-j)Tg_c(s,\rho)\} \\
&\quad \text{where } f_c(s,\rho) \text{ and } g_c(s,\rho) \text{ are defined below} \\
f_c(s,\rho) &\triangleq \ln\sum_y\Big(\sum_x p(x)p(y|x)^{1-s\rho}\Big)\Big(\sum_x p(x)p(y|x)^s\Big)^\rho \\
&= \ln\sum_y p(y)\Big(\sum_x\Big[\frac{p(y|x)}{p(y)}\Big]^{1-s\rho}p(x)\Big)\Big(\sum_x\Big[\frac{p(y|x)}{p(y)}\Big]^s p(x)\Big)^\rho \\
g_c(s,\rho) &\triangleq \log_2\sum_y p(y)\Big(\sum_x\Big[\frac{p(y|x)}{p(y)}\Big]^s p(x)\Big)^\rho \\
&\quad \rho\in[0,1],\ s\geq 0
\end{aligned}$$

The above equations follow from alternating the order of sums and products in the usual manner of proofs using the Gallager style bound. Combining everything for a bound on $P(F_d)$, we have the following.

$$\begin{aligned}
P(F_d) &\leq \sum_{j=1}^d P(F_{d,j}) \\
&\leq \sum_{j=1}^d \exp(dTR\rho + s\rho(j-d)TG + (d-j)Tg_c(s,\rho) + jTf_c(s,\rho))
\end{aligned}$$

$$= \exp\big(dT(R\rho - s\rho G + g_c(s,\rho))\big) \sum_{j=1}^{d} \exp\big(jT(s\rho G - g_c(s,\rho) + f_c(s,\rho))\big)$$

Now suppose $s\rho G - g(s,\rho) + f(s,\rho) \geq 0$, that is if $G > \frac{1}{s\rho}(g(s,\rho) - f(s,\rho))$. Then,

$$\begin{aligned} \sum_{j=1}^{d} \exp_2(jT(s\rho G - g_c(s,\rho) + f_c(s,\rho))) &\leq d \cdot \exp(dT(s\rho G - g_c(s,\rho) + f_c(s,\rho))) \\ \Rightarrow P(F_d) &\leq d \exp\big(-dT(-\rho R - f(s,\rho))\big) \end{aligned}$$

The value of $s = \frac{1}{1+\rho}$ maximizes $-f(s,\rho)$ for a fixed $\rho$. Now we note that $-f(\frac{1}{1+\rho},\rho) = E_0(\rho)$ where $E_0$ is Gallager's function. Now, by choice of the channel input distribution $P_A$, we have the following.

$$\begin{aligned} P(F_d) &\leq d \exp\big(-dT(-\rho R + E_0(\rho))\big) \\ &\leq \exp\Big(-dTE_r(R) + \ln d\Big) \end{aligned}$$

Asymptotically, $\ln d$ is insignificant compared to $dTE_r(R)$, so we can precisely state

$$\limsup_{d\to\infty} -\ln \frac{1}{dT} P(F_d) \geq E_r(R) \tag{A.8}$$

## A.2 The probability of error with delay for a source coding scheme with sequential decoding

In this section, we prove Theorem 3.3.1 which we state here again for reference. In Chapter 3, we used $log_2$ and rate in bits. For convenience in the proof, we switch to rate in *nats* and use $ln$, the natural logarithm.

**Theorem A.2.1** *Using the stack decoding algorithm with the sequential binning encoder of Chapter 3, we can achieve asymptotically the same probability of error as with MAP decoding. Let $Q(x)$ be the probability of the source on a finite alphabet $\mathcal{X}$. Define $E_s(\rho)$ as below, for $\rho \geq 0$.*

$$E_s(\rho) \triangleq (1+\rho) \ln\Big(\sum_{x\in\mathcal{X}} Q(x)^{\frac{1}{1+\rho}}\Big) \tag{A.9}$$

*Let $\rho^*$ be defined so that rate $R$ in nats per source symbol of the code satisfies $R = E_s'(\rho^*)$. However, if $\rho^* > 1$, replace it with $\rho^* = 1$. If the bias of the sequential decoder, $G$, satisfies*

$$G < \frac{E_s(\rho^*)}{\rho^*} \tag{A.10}$$

*then for any $\epsilon > 0$, the probability of error with delay, $P(\hat{x}_1^n(n+d) \neq x_1^n)$, for $n \geq 1$ is at most*

$$P(\hat{x}_1^n(n+d) \neq x_1^n) \leq \tilde{K}_\epsilon e^{-d(E_r(R)-\epsilon)} \tag{A.11}$$

*where $E_r(R) = \sup_{\rho \in [0,1]} \rho R - E_s(\rho)$, and $\tilde{K}_\epsilon$ is a finite constant independent of $d$.*

**Proof:** First let us set up the notation. Denote the vector of (true) realized source symbols $(x_i, x_{i+1}, \ldots, x_j)$ as $x_i^j$, for $i \leq j$. If $j < i$, $x_i^j$ will refer to the null string. For simplicity, $x_i$ denotes just the $i^{th}$ source symbol. Let $\hat{x}_i^j(n)$ be the decoders $i^{th}$ through $j^{th}$ recovered symbols at time $n$. We will reserve $y_i^j$ to denote source symbols along a path in the encoding tree that are not necessarily the true source symbols. In particular, a path that may cause an error will be denoted with the letter $y$ and the letter $x$ will be reserved for the true source symbols. The letter $B$ will be reserved for the bits received by the decoder, which will be referred to as 'parities'.

The probability measure for the source symbols is $Q$, which takes on non-zero values on a finite alphabet $\mathcal{X}$. The probability measure $P$ will refer to all randomness in the source as well as the randomly generated encoder. When no confusion arises, $Q$ will be applied to multiple symbols like $x_1^k$ with the meaning that $Q(x_1^k) = \prod_{l=1}^k Q(x_l)$. The encoding and decoding schemes used are the same as described in Chapter 3. Let $G$ denote the bias of the sequential decoder so that the metric of a source symbol is $\Gamma(x) = \ln(Q(x)) + G$, $\forall x \in \mathcal{X}$.

The error event of interest is defined in equation A.12 and we will relate it to the theorem at the end of the proof. Figure A.2 shows paths that may lead to an error event of depth 3 occuring, i.e. $F_3$. Just as when the stack algorithm is used for channel decoding, $F_d$ can only occur if there is a false path $y_1^d$, with $y_1 \neq x_1$, that has a metric at least the minimum of the metrics along the true path, $x_1^k$, for $k = 1, \ldots, d$.

$$F_d \triangleq \left\{ \exists\ y_1^d\ ,\ y_1 \neq x_1 \;\middle|\; \Gamma(y_1^d) \geq \min_{1 \leq k \leq d} \Gamma(x_1^k) \text{ and parities of } y_1^d \text{ match } B_1^{dR} \right\} \tag{A.12}$$

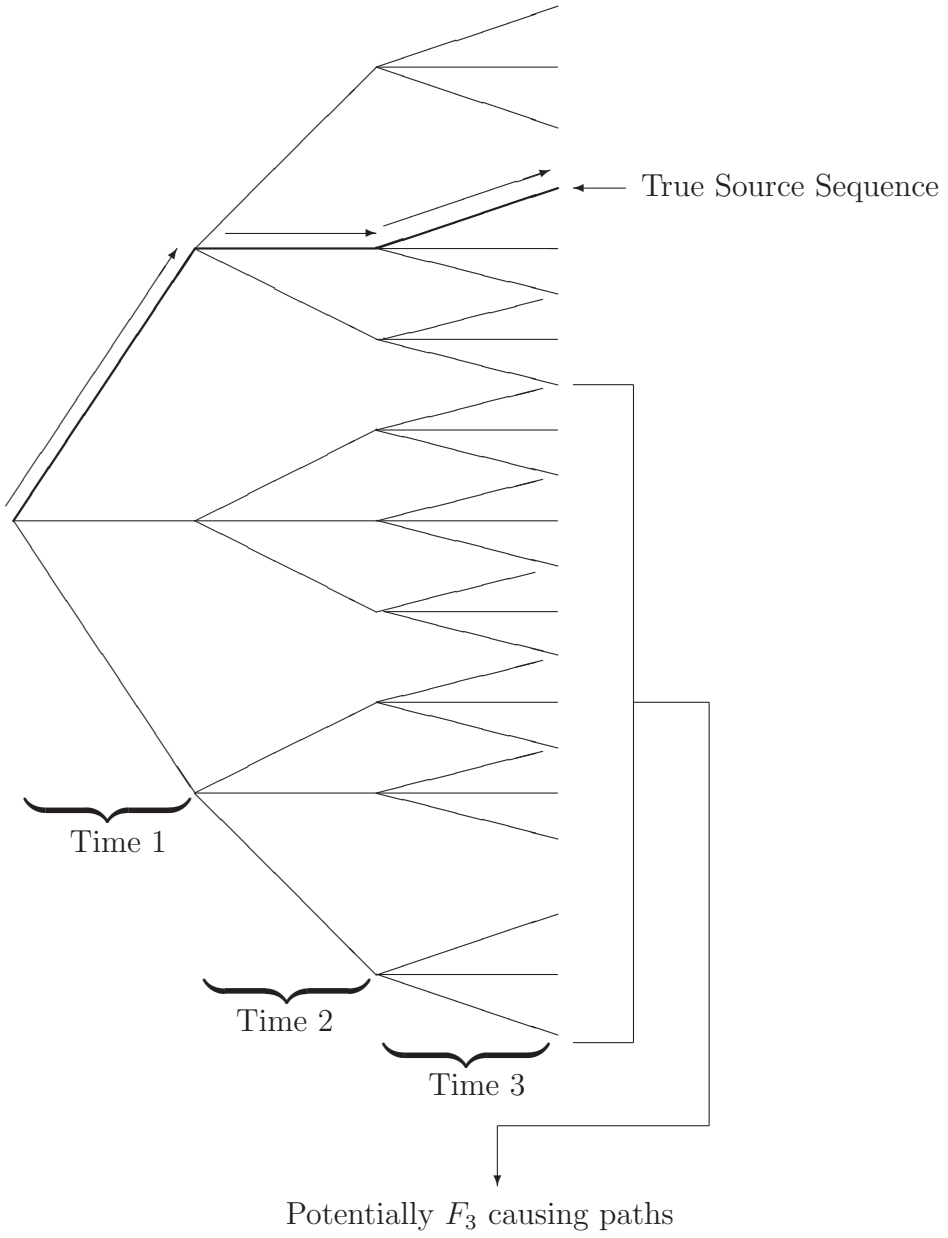


Figure A.3. A ternary tree. The true source sequence is shown above. The condition $y_1 \neq x_1$ selects a portion of the tree containing paths that could potentially cause the error event $F_3$.

The event $F_d$ can be subdivided into events $F_{d,k}$ so that $F_d = \bigcup_{k=1}^{d} F_{d,k}$, where

$$F_{d,k} \triangleq \left\{ \exists\ y_1^d\ , y_1 \neq x_1 \middle|\quad \Gamma(y_1^d) \geq \Gamma(x_1^k) \text{ and parities of } y_1^d \text{ match } B_1^{dR} \right\} \tag{A.13}$$

By conditioning on the source sequence and applying the union bound, we get

$$\begin{aligned} P(F_d) &= \sum_{x_1^d \in \mathcal{X}^d} Q(x_1^d) P(F_d | X_1^d = x_1^d) && \text{(A.14)} \\ &\leq \sum_{x_1^d} Q(x_1^d) \sum_{k=1}^{d} P(F_{d,k} | X_1^d = x_1^d) && \text{(A.15)} \\ &= \sum_{k=1}^{d} \sum_{x_1^d} Q(x_1^d) P(F_{d,k} | X_1^d = x_1^d) && \text{(A.16)} \end{aligned}$$

Suppose $y_1^d$ is a false path that causes $F_{d,k}$ to occur. This means its parities match the received bits and its metric $\Gamma(y_1^d)$ is at least $\Gamma(x_1^k)$. Therefore,

$$\begin{aligned} 0 &\leq \Gamma(y_1^d) - \Gamma(x_1^k) && \text{(A.17)} \\ &= \sum_{l=1}^{d} \Big( \ln(Q(y_l)) + G \Big) - \sum_{l=1}^{k} \Big( \ln(Q(x_l)) + G \Big) && \text{(A.18)} \\ &= \sum_{l=1}^{k} \ln \left( \frac{Q(y_l)}{Q(x_l)} \right) + \sum_{l=k+1}^{d} \ln Q(y_l) + (d-k)G && \text{(A.19)} \end{aligned}$$

Now, denoting $1(\cdot)$ as the indicator function of its argument, and using a Gallager-style union bound, for $\rho \in [0,1]$, we have

$$\begin{aligned} P(F_{d,k} | X_1^d = x_1^d) &\leq E\left[ \Big( \sum_{y_1^d:\ y_1 \neq x_1} 1(y_1^d \text{ causes } F_{d,k} \text{ to occur}) \Big)^{\rho} \middle| X_1^d = x_1^d \right] && \text{(A.20)} \\ &\leq \left( \sum_{y_1^d,\ y_1 \neq x_1} E\left[ 1(y_1^d \text{ causes } F_{d,k} \text{ to occur}) \middle| X_1^d = x_1^d \right] \right)^{\rho} && \text{(A.21)} \\ &\triangleq \left( \sum_{y_1^d,\ y_1 \neq x_1} A_k(y_1^d, x_1^d) \right)^{\rho} && \text{(A.22)} \end{aligned}$$

Here equation A.21 follows from Jensen's inequality because for $\rho \in [0,1]$, the function $a^\rho$ is concave. Continuing with the bounding, we use the fact that the parity generation process

is independent[1] of everything else to get

$$
\begin{aligned}
A_k(y_1^d, x_1^d) &= E\left[1(\text{parities of } y_1^d \text{ match } B_1^{dR}) \cdot 1(\Gamma(y_1^d) \geq \Gamma(x_1^k))\middle| X_1^d = x_1^d\right] && \text{(A.23)}\\
&= E\left[1(\text{parities of } y_1^d \text{ match } B_1^{dR})\right] E\left[1(\Gamma(y_1^d) \geq \Gamma(x_1^k))\middle| X_1^d = x_1^d\right] && \text{(A.24)}\\
&= e^{-dR} E\left[1(\Gamma(y_1^d) \geq \Gamma(x_1^k))\middle| X_1^d = x_1^d\right] && \text{(A.25)}\\
&\leq \exp(-dR) \cdot \exp\left(s(\Gamma(y_1^d) - \Gamma(x_1^k))\right) && \text{(A.26)}\\
&= \exp(-dR) \cdot \exp\left(s\left[\ln \frac{Q(y_1^k)}{Q(x_1^k)} + \ln Q(y_{k+1}^d) + (d-k)G\right]\right) && \text{(A.27)}\\
&\quad \text{for any } s \geq 0 && \text{(A.28)}
\end{aligned}
$$

Substituting for $A_k(y_1^d, x_1^d)$, and removing the restriction that $y_1 \neq x_1$,

$$
\begin{aligned}
P(F_{d,k}|x_1^d) &\leq \left(\sum_{y_1^d} \exp(-dR) \exp\left(s\left[\ln \frac{Q(y_1^k)}{Q(x_1^k)} + \ln Q(y_{k+1}^d) + (d-k)G\right]\right)\right)^\rho && \text{(A.29)}\\
&\leq \exp(-d\rho R + (d-k)s\rho G)\left(\sum_{y_1^d} \left(\frac{Q(y_1^k)}{Q(x_1^k)}\right)^s Q(y_{k+1}^d)^s\right)^\rho && \text{(A.30)}\\
&= \exp(-d\rho R + (d-k)s\rho G)\left(\sum_{y_1^k} \left(\frac{Q(y_1^k)}{Q(x_1^k)}\right)^s\right)^\rho \left(\sum_{y_{k+1}^d} Q(y_{k+1}^d)^s\right)^\rho && \text{(A.31)}
\end{aligned}
$$

Equation A.31 follows from the standard algebra of interchanging sums and products. Finally, we are ready to complete the bound of $P(F_d)$.

$$
\begin{aligned}
P(F_d) &\leq \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} \sum_{x_1^k} \sum_{x_{k+1}^d} && \text{(A.32)}\\
&\qquad Q(x_1^k) Q(x_{k+1}^d) \left(\sum_{y_1^k} \left(\frac{Q(y_1^k)}{Q(x_1^k)}\right)^s\right)^\rho \left(\sum_{y_{k+1}^d} Q(y_{k+1}^d)^s\right)^\rho && \text{(A.33)}\\
&= \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} && \text{(A.34)}\\
&\qquad \sum_{x_1^k} Q(x_1^k)^{1-s\rho} \left(\sum_{y_1^k} Q(y_1^k)^s\right)^\rho \sum_{x_{k+1}^d} Q(x_{k+1}^d) \left(\sum_{y_{k+1}^d} Q(y_{k+1}^d)^s\right)^\rho && \text{(A.35)}\\
&= \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} \left(\sum_{x_1^k} Q(x_1^k)^{1-s\rho}\right) \left(\sum_{y_1^k} Q(y_1^k)^s\right)^\rho \left(\sum_{y_{k+1}^d} Q(y_{k+1}^d)^s\right)^\rho && \text{(A.36)}\\
&= \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} \left(\sum_{x_1^k} Q(x_1^k)^{1-s\rho}\right) \left(\sum_{x_1^k} Q(x_1^k)^s\right)^\rho \left(\sum_{x_{k+1}^d} Q(x_{k+1}^d)^s\right)^\rho && \text{(A.37)}
\end{aligned}
$$

[1]Note that we need only pairwise independence of the parities along two paths.

We get equation A.37 by noting that the $y$'s are just dummy variables and we are free to replace them with $x$'s. Next, we use the iid property of the source along with standard algebra to get to an exponential form. For example, we have

$$\begin{aligned}
\sum_{x_1^k} Q(x_1^k)^{1-s\rho} &= \sum_{x_1}\sum_{x_2}\cdots\sum_{x_k}\prod_{l=1}^{k} Q(x_l)^{1-s\rho} && \text{(A.38)}\\
&= \prod_{l=1}^{k}\sum_{x_l} Q(x_l)^{1-s\rho} && \text{(A.39)}\\
&= \Big(\sum_{x\in\mathcal{X}} Q(x)^{1-s\rho}\Big)^{k} && \text{(A.40)}
\end{aligned}$$

Similarly,

$$\begin{aligned}
\Big(\sum_{x_1^k} Q(x_1^k)^{s}\Big)^{\rho} &= \Big(\sum_{x_1}\sum_{x_2}\cdots\sum_{x_k}\prod_{l=1}^{k} Q(x_l)^{s}\Big)^{\rho} && \text{(A.41)}\\
&= \Big(\prod_{l=1}^{k}\sum_{x_l} Q(x_l)^{s}\Big)^{\rho} && \text{(A.42)}\\
&= \prod_{l=1}^{k}\Big(\sum_{x_l} Q(x_l)^{s}\Big)^{\rho} && \text{(A.43)}\\
&= \Big(\sum_{x\in\mathcal{X}} Q(x)^{s}\Big)^{k\rho} && \text{(A.44)}
\end{aligned}$$

Now define the exponent functions $f(s,\rho)$ and $g(s,\rho)$ as

$$\begin{aligned}
f(s,\rho) &\triangleq \ln\left[\Big(\sum_{x\in\mathcal{X}} Q(x)^{1-s\rho}\Big)\Big(\sum_{x\in\mathcal{X}} Q(x)^{s}\Big)^{\rho}\right] && \text{(A.45)}\\
g(s,\rho) &\triangleq \ln\Big(\sum_{x\in\mathcal{X}} Q(x)^{s}\Big)^{\rho} && \text{(A.46)}
\end{aligned}$$

With these definitions and a bit more algebra, we have

$$\begin{aligned}
P(F_d) &\leq \sum_{k=1}^{d}\exp\Big(-d\rho R + (d-k)s\rho G + kf(s,\rho) + (d-k)g(s,\rho)\Big) && \text{(A.47)}\\
&= \exp\Big(d\big(s\rho G + g(s,\rho) - \rho R\big)\Big)\sum_{k=1}^{d}\exp\Big(k\big(f(s,\rho) - g(s,\rho) - s\rho G\big)\Big) && \text{(A.48)}
\end{aligned}$$

Suppose the following condition holds.

$$f(s,\rho) - g(s,\rho) - s\rho G \geq 0 \qquad \text{(A.49)}$$

Then, we can simplify the bound to

$$\begin{aligned} P(F_d) &\leq \exp\Big(d\big(s\rho G + g(s,\rho) - \rho R\big)\Big) \sum_{k=1}^{d} \exp\Big(k\big(f(s,\rho) - g(s,\rho) - s\rho G\big)\Big) && \text{(A.50)} \\ &\leq \exp\Big(d\big(s\rho G + g(s,\rho) - \rho R\big)\Big) \cdot d \exp\Big(d\big(f(s,\rho) - g(s,\rho) - s\rho G\big)\Big) && \text{(A.51)} \\ &= d \exp\Big(-d(\rho R - f(s,\rho))\Big) && \text{(A.52)} \end{aligned}$$

Now this holds for all $\rho \in [0,1]$ and $s \geq 0$, so we can let $s = 1/(1+\rho)$ and minimize over $\rho$. So, now we have for any $\epsilon > 0$,

$$\begin{aligned} P(F_d) &\leq K_\epsilon \exp\Big(-d(E_r(R) - \epsilon)\Big) && \text{(A.53)} \\ E_r(R) &\triangleq \sup_{\rho \in [0,1]} \rho R - E_s(\rho) && \text{(A.54)} \\ E_s(\rho) &\triangleq (1+\rho) \ln\Big(\sum_{x \in \mathcal{X}} Q(x)^{\frac{1}{1+\rho}}\Big) && \text{(A.55)} \\ K_\epsilon &\triangleq \max\left\{d : \frac{\ln d}{d} \geq \epsilon\right\} < \infty && \text{(A.56)} \end{aligned}$$

Note that $K_\epsilon < \infty$ and is independent of $d$ because $\ln(d)/d$ goes to 0. We note that $E_s(\rho)$ is a differentiable function for all $\rho \geq 0$, with $E_s'(0) = H(Q)$, that is the slope at 0 is the entropy of the source. $E_s(\rho)$ is the source coding counterpart for Gallager's function $E_0(\rho)$. While Gallager's function may be non-differentiable at points because it is the maximization of a function over probability distributions, $E_s(\rho)$ doesn't suffer from this problem. It can be shown that $E_s(\rho)$ is strictly increasing and convex for non-deterministic sources. Hence the optimizing $\rho$ to achieve $E_r(R)$ is parametrically defined as $\rho^* = E_s'(\rho^*)$. Going back to our condition on the bias, the bound is true if the bias $G$ satisfies

$$\begin{aligned} G &\leq \frac{(1+\rho^*)}{\rho^*}\Big[E_s(\rho^*) - g(1/(1+\rho^*), \rho^*)\Big] && \text{(A.57)} \\ &= \frac{E_s(\rho^*)}{\rho^*} && \text{(A.58)} \end{aligned}$$

where $\rho^*$ satisfies $\frac{d}{d\rho} E_s(\rho)\Big|_{\rho=\rho^*} = R$.

Finally, we can prove the statement of the theorem. In order for a depth $d$ or greater error to occur it must be that $\hat{x}_i(n+d) \neq x_i$ for some $1 \leq i \leq n$. Now, assuming the bias

satisfies the required condition, we have

$$
\begin{aligned}
P(\hat{x}_1^n(n+d) \neq x_1^n) &= \sum_{k=0}^{n-1} P(\hat{x}_1^k(n+d) = x_1^k,\ \hat{x}_{k+1} \neq x_{k+1}) && \text{(A.59)}\\
&\leq \sum_{k=0}^{n-1} P(F_{d+n-k}) P(\hat{x}_1^k(n+d) = x_1^k) && \text{(A.60)}\\
&\leq \sum_{k=0}^{\infty} P(F_{d+k}) P(\hat{x}_1^{n-k}(n+d) = x_1^{n-k}) && \text{(A.61)}\\
&\leq \sum_{k=0}^{\infty} P(F_{d+k}) && \text{(A.62)}\\
&\leq \sum_{k=0}^{\infty} K_\epsilon e^{-(d+k)(E_r(R)-\epsilon)} && \text{(A.63)}\\
&= e^{-d(E_r(R)-\epsilon)} \sum_{k=0}^{\infty} K_\epsilon e^{-k(E(R)-\epsilon)} && \text{(A.64)}
\end{aligned}
$$

The critical step is in equation A.60, which says that if the decoded path and true path agree until time $k$, the error event can be thought of as 'rooted' at time $k+1$. Hence, we are reduced to the error event $F_{d+n-k}$. Since we can choose $\epsilon$ arbitrarily small, the geometric series converges and we have

$$
\begin{aligned}
P(\hat{x}_1^n(d+n) \neq x_1^n) &\leq \frac{K_\epsilon}{1 - e^{-(E(R)-\epsilon)}} e^{-d(E_r(R)-\epsilon)} && \text{(A.65)}\\
&= \tilde{K}_\epsilon e^{-d(E_r(R)-\epsilon)} && \text{(A.66)}\\
\tilde{K}_\epsilon &\triangleq \frac{K_\epsilon}{1 - e^{-(E(R)-\epsilon)}} && \text{(A.67)}
\end{aligned}
$$

We have concluded that the probability of error with delay attained by the sequential decoder is asymptotically close to the probability of error with delay attained by a MAP decoder.□

## A.3 Computation bound for source coding with a sequential decoder

We now prove Theorem 3.3.2, stated again for reference. Once again, for convenience, we switch to rate in nats and use the natural logarithm $ln$ instead of $log_2$.

**Theorem A.3.1** *For the source coding scheme of Chapter 3, let $N_i$ denote the number of nodes the sequential decoder visits in the $i^{th}$ incorrect subtree. For a $\gamma \in [0,1]$, if the bias $G$ satisfies*

$$\frac{1+\gamma}{\gamma}H(\gamma) = \frac{E_s(\gamma)}{\gamma} < \quad G \quad < \frac{1+\gamma}{\gamma}\Big[\gamma R - G(\gamma)\Big] \tag{A.68}$$

$$H(\gamma) \quad \triangleq \quad \ln \sum_{x\in\mathcal{X}} Q(x)^{1/(1+\gamma)} \tag{A.69}$$

$$G(\gamma) \quad \triangleq \quad \gamma \ln \sum_{x\in\mathcal{X}} Q(x)^{1/(1+\gamma)} \tag{A.70}$$

*the $\gamma^{th}$ moment of $N_i$, averaged over the source and encoder randomness, is finite for all $i$ if*

$$R \quad > \quad \frac{1}{\gamma}E_s(\gamma) \tag{A.71}$$

**Proof:** Let $N_i$ denote the number of nodes within the $i^{th}$ incorrect subtree that are ever visited by the stack algorithm. A path in the $i^{th}$ incorrect subtree and the true path have the same first $i$ symbols. In bounding a moment of $N_i$, we need only consider the portion of the tree after the $i^{th}$ symbol, so let $N = N_1$. Going through analysis very similar to that found in [1], we start with the definition of the $\gamma^{th}$ moment of $N$ as

$$E\big[N^\gamma\big] \quad = \quad E\Bigg[\bigg(\sum_{l=1}^{\infty} \sum_{y_1^l,\ y_1\neq x_1} 1(\ y_1^l \text{ is visited }\ )\bigg)^\gamma\Bigg] \tag{A.72}$$

If we think of the algorithm running forever, a required condition for a node $y_1^l$ to be visited is

$$\Gamma(y_1^l) \geq \min_{1\leq k<\infty} \Gamma(x_1^k) \tag{A.73}$$

Also, the node $y_1^l$ can be only be visited if its parities match the bits $B_1^{lR}$. So we have,

$$E\big[N^\gamma\big] \quad \leq \quad E\Bigg[\bigg(\sum_{l=1}^{\infty}\sum_{k=1}^{\infty} \sum_{y_1^l,\ y_1\neq x_1} 1(\Gamma(y_1^l) \geq \Gamma(x_1^k) \text{ and parties of } y_1^l \text{ match } B_1^{lR}\ )\bigg)^\gamma\Bigg] \tag{A.74}$$

$$\leq \quad \sum_{l=1}^{\infty}\sum_{k=1}^{\infty} E\Bigg[\bigg(\sum_{y_1^l} 1(\Gamma(y_1^l) \geq \Gamma(x_1^k) \text{ and parities of } y_1^l \text{ match } B_1^{lR})\bigg)^\gamma\Bigg] \tag{A.75}$$

$$= \quad \sum_{k=1}^{\infty}\sum_{l=1}^{\infty}\sum_{x_1^k} Q(x_1^k) E\Bigg[\bigg(\sum_{y_1^l} 1(\Gamma(y_1^l) \geq \Gamma(x_1^k) \text{ and parities match})\bigg)^\gamma \Bigg| x_1^k\Bigg] \tag{A.76}$$

$$\leq \quad \sum_{k=1}^{\infty}\sum_{l=1}^{\infty}\sum_{x_1^k} Q(x_1^k)\bigg(E\Bigg[\sum_{y_1^l} 1(\Gamma(y_1^l) \geq \Gamma(x_1^k) \text{ and parities match})\Bigg| x_1^k\Bigg]\bigg)^\gamma \tag{A.77}$$

Equation A.75 is true because of the inequality $(a+b)^\gamma \leq a^\gamma + b^\gamma$ if $a, b \geq 1$ and $\gamma \in [0,1]$. Equation A.77 follows from Jensen's inequality. Using the independence of the parity generation process, we get

$$\begin{aligned} E\big[N^\gamma\big] &\leq \sum_{k=1}^{\infty}\sum_{l=1}^{\infty}\sum_{x_1^k} Q(x_1^k) e^{-l\gamma R} \sum_{y_1^l} \exp\Big(s(\Gamma(y_1^l) - \Gamma(x_1^k))\Big)^\gamma && \text{(A.78)} \\ &\quad \text{for any } s \geq 0 && \text{(A.79)} \\ &= \sum_{k=1}^{\infty}\sum_{l=1}^{\infty}\sum_{x_1^k} Q(x_1^k) e^{-l\gamma R + (l-k)\gamma G} \Big(\sum_{y_1^l} \Big(\frac{Q(y_1^l)}{Q(x_1^k)}\Big)^s\Big)^\gamma && \text{(A.80)} \\ &= \sum_{k=1}^{\infty}\sum_{l=1}^{\infty} e^{-l\gamma R + (l-k)\gamma G} \sum_{x_1^k} Q(x_1^k) \Big(\sum_{y_1^l} \Big(\frac{Q(y_1^l)}{Q(x_1^k)}\Big)^s\Big)^\gamma && \text{(A.81)} \end{aligned}$$

Now define the quantity $\beta_{l,k}$ as

$$\beta_{l,k} \triangleq \sum_{x_1^k} Q(x_1^k) \Big(\sum_{y_1^l} \Big(\frac{Q(y_1^l)}{Q(x_1^k)}\Big)^s\Big)^\gamma \qquad \text{(A.82)}$$

If $l \geq k$, we know

$$\beta_{l,k} = \exp\Big(k f(s,\gamma) + (l-k) g(s,\gamma)\Big) \qquad \text{(A.83)}$$

where $f$ and $g$ are defined in equations A.45 and A.46 respectively. Now if $l \leq k$, we can simplify to get

$$\begin{aligned} \beta_{l,k} &= \sum_{x_1^l} Q(x_1^l)^{1-s\gamma} \Big(\sum_{y_1^l} Q(y_1^l)^s\Big)^\gamma \sum_{x_{l+1}^k} Q(x_{l+1}^k)^{1-s\gamma} && \text{(A.84)} \\ &= \exp\Big(l f(s,\gamma) + (k-l) h(s,\gamma)\Big) && \text{(A.85)} \\ f(s,\gamma) &\triangleq \ln\Big(\sum_{x\in\mathcal{X}} Q(x)^{1-s\gamma}\Big)\Big(\sum_{x\in\mathcal{X}} Q(x)^s\Big) && \text{(A.86)} \\ h(s,\gamma) &\triangleq \ln \sum_{x\in\mathcal{X}} Q(x)^{1-s\gamma} && \text{(A.87)} \end{aligned}$$

Now we set $s = 1/(1+\gamma)$ and define the functions $G(\gamma)$ and $H(\gamma)$ to equal $g(1/(1+\gamma),\gamma)$ and $h(1/(1+\gamma),\gamma)$ respectively. Of course we already have defined $E_s(\gamma)$ to equal $f(1/(1+$

$\gamma), \gamma)$. Now we can plug all of this into the bound for the $\gamma^{th}$ moment of computation.

$$
\begin{aligned}
E\big[N^\gamma\big] &\leq \sum_{l=1}^{\infty}\sum_{k=l}^{\infty}\exp\Big(-l\gamma R + (l-k)s\gamma G + lE_s(\gamma) + (k-l)H(\gamma)\Big) + \cdots && \text{(A.88)}\\
&\quad \sum_{k=1}^{\infty}\sum_{l=k}^{\infty}\exp\Big(-l\gamma R + (l-k)s\gamma G + kE_s(\gamma) + (l-k)G(\gamma)\Big) && \text{(A.89)}\\
&= \sum_{l=1}^{\infty}\exp\Big(l(E_s(\gamma)-\gamma R)\Big)\sum_{k=1}^{\infty}\exp\Big((k-l)(H(\gamma)-s\gamma G)\Big) + \cdots && \text{(A.90)}\\
&\quad \sum_{k=1}^{\infty}\exp\Big(k(E_s(\gamma)-\gamma R)\Big)\sum_{l=k}^{\infty}\exp\Big((l-k)(s\gamma G + G(\gamma)-\gamma R)\Big) && \text{(A.91)}
\end{aligned}
$$

The two double sums converge if the following conditions hold:

$$
\begin{aligned}
0 &> E_s(\gamma) - \gamma R && \text{(A.92)}\\
0 &> \frac{\gamma}{1+\gamma}G + G(\gamma) - \gamma R && \text{(A.93)}\\
0 &> H(\gamma) - \frac{\gamma}{1+\gamma}G && \text{(A.94)}
\end{aligned}
$$

These conditions can be equivalently stated concisely in terms of $R$ and the bias $G$ as

$$
\begin{aligned}
\frac{1}{\gamma}E_s(\gamma) &< R && \text{(A.95)}\\
\frac{E_s(\gamma)}{\gamma} = \frac{1+\gamma}{\gamma}H(\gamma) &< G < \frac{1+\gamma}{\gamma}\Big[\gamma R - G(\gamma)\Big] = (1+\gamma)R - E_s(\gamma) && \text{(A.96)}
\end{aligned}
$$

Now we note that $\gamma R - G(\gamma) - H(\gamma) = \gamma R - E_s(\gamma) > 0$ if condition A.95 holds. So given that $R > E_s(\gamma)/\gamma$, then there is an interval of positive length consisting of bias values for which $E[N^\gamma] < \infty$. □

## A.4 Probability of error for source coding with side information using a sequential decoder

Again we switch from rate in bits to rate in nats and switch from using $\log_2$ to using ln. The theorem is stated again here for reference.

**Theorem A.4.1** *Let the source pair $(X_i, Y_i)$ be generated iid according to a distribution $Q(x, y)$ on a finite set $\mathcal{X} \times \mathcal{Y}$. Using the encoder of 3.2 and the modified stack decoder of*

*3.6, we can achieve asymptotically the same probability of error as with MAP decoding. Define $E_{si}(\rho)$ as below, for $\rho \geq 0$.*

$$E_{si}(\rho) \triangleq \ln \sum_{y \in \mathcal{Y}} \left( \sum_{x \in \mathcal{X}} Q(x,y)^{\frac{1}{1+\rho}} \right)^{1+\rho} \tag{A.97}$$

*Let $\rho^*$ be defined so that rate $R$ in nats per source symbol of the code satisfies $R = \frac{d}{d\rho} E_{si}(\rho)|_{\rho=\rho^*}$. However, if $\rho^* > 1$, replace it with $\rho^* = 1$. If the bias of the sequential decoder, $G$, satisfies*

$$G \leq \frac{1+\rho^*}{\rho^*} \left[ E_{si}(\rho^*) - F_{si}(\rho^*) \right] \tag{A.98}$$

$$F_{si}(\rho) \triangleq \ln \sum_{y \in \mathcal{Y}} Q(y) \left( \sum_{x \in \mathcal{X}} Q(x|y)^{\frac{1}{1+\rho}} \right)^{\rho} \tag{A.99}$$

*then for any $\epsilon > 0$, the probability of error with delay, $P_e(d)$, is at most*

$$P_e(d) \leq \tilde{K}_\epsilon 2^{-d(E_r(R)-\epsilon)} \tag{A.100}$$

*where $E_r(R) = \sup_{\rho \in [0,1]} \rho R - E_{si}(\rho)$, and $\tilde{K}_\epsilon$ is a finite constant independent of $d$. This shows that the error exponent with delay of this scheme is at least $E_r(R)$.*

**Proof:** The proof is completely parallel to the proof in A.2, the only addition is conditioning on the side information sequence $y_1^d$. Using the same error events, and letting $u$ denote a false sequence in the tree, we get to the following:

$$\begin{aligned}
P(F_d) &\leq \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} \sum_{y_1^d} \sum_{x_1^k} \sum_{x_{k+1}^d} \\
&\qquad Q(y_1^d) Q(x_1^k|y_1^k) Q(x_{k+1}^d|y_{k+1}^d) \left( \sum_{u_1^k} \left( \frac{Q(u_1^k|y_1^k)}{Q(x_1^k|y_1^k)} \right)^s \right)^{\rho} \left( \sum_{u_{k+1}^d} Q(u_{k+1}^d|y_{k+1}^d)^s \right)^{\rho} \\
&= \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} \sum_{y_1^d} \sum_{x_1^k} \sum_{x_{k+1}^d} \\
&\qquad Q(y_1^d) Q(x_1^k|y_1^k)^{1-s\rho} Q(x_{k+1}^d|y_{k+1}^d) \left( \sum_{u_1^k} Q(u_1^k|y_1^k)^s \right)^{\rho} \left( \sum_{u_{k+1}^d} Q(u_{k+1}^d|y_{k+1}^d)^s \right)^{\rho} \\
&= \sum_{k=1}^{d} e^{-d\rho R + (d-k)s\rho G} \sum_{y_1^d} \sum_{x_1^k}
\end{aligned}$$

$$Q(y_1^d)Q(x_1^k|y_1^k)^{1-s\rho}\Big(\sum_{u_1^k}Q(u_1^k|y_1^k)^s\Big)^\rho\Big(\sum_{u_{k+1}^d}Q(u_{k+1}^d|y_{k+1}^d)^s\Big)^\rho$$

$$= \sum_{k=1}^{d} e^{-d\rho R+(d-k)s\rho G}\sum_{y_1^d}Q(y_1^d)\Big(\sum_{x_1^k}Q(x_1^k|y_1^k)^{1-s\rho}\Big)$$
$$\Big(\sum_{x_1^k}Q(x_1^k|y_1^k)^s\Big)^\rho\Big(\sum_{x_{k+1}^d}Q(x_{k+1}^d|y_{k+1}^d)^s\Big)^\rho$$

Now set $s = 1/(1+\rho)$. Using the iid nature of the source and interchanging products with sums, we have

$$\begin{aligned}P(F_d) &< \sum_{k=1}^{d} e^{-d\rho R+(d-k)s\rho G}\Big(\sum_{y\in\mathcal{Y}}Q(y)\Big(\sum_{x\in\mathcal{X}}Q(x|y)^{\frac{1}{1+\rho}}\Big)^{1+\rho}\Big)^k \\ &\qquad \Big(\sum_{y\in\mathcal{Y}}Q(y)\Big(\sum_{x\in\mathcal{X}}Q(x|y)^{\frac{1}{1+\rho}}\Big)^{\rho}\Big)^{d-k} \\ &= \sum_{k=1}^{d} e^{-d\rho R+(d-k)s\rho G}\Big(\sum_{y\in\mathcal{Y}}\Big(\sum_{x\in\mathcal{X}}Q(x,y)^{\frac{1}{1+\rho}}\Big)^{1+\rho}\Big)^k\Big(\sum_{y\in\mathcal{Y}}Q(y)\Big(\sum_{x\in\mathcal{X}}Q(x|y)^{\frac{1}{1+\rho}}\Big)^{\rho}\Big)^{d-k}\end{aligned}$$

Now, having defined $E_{si}(\rho)$ and $F_{si}$ in the theorem statement, we get

$$\begin{aligned}P(F_d) &< \sum_{k=1}^{d}\exp\Big(-d\rho R+(d-k)\frac{\rho}{1+\rho}G+kE_{si}(\rho)+(d-k)F_{si}(\rho)\Big) \\ &= \exp\Big(-d\rho R+d\frac{\rho}{1+\rho}G+dF_{si}(\rho)\Big)\sum_{k=1}^{d}\exp\Big(k\Big[E_{si}(\rho)-F_{si}(\rho)-\frac{\rho}{1+\rho}G\Big]\Big)\end{aligned}$$

If $E_{si}(\rho)-F_{si}(\rho)-F\rho/(1+\rho)\le 0$, we get

$$\begin{aligned}P(F_d) &< d\exp\Big(-d(\rho R-E_{si}(\rho))\Big) \\ &\forall\rho\in[0,1]\end{aligned}$$

The linear term $d$ is dominated by the exponential, so maximizing the exponent with respect to $\rho$ gives the result.