Noisy Channels with Synchronization Errors: Information Rates and Code Design

JITENDER TOKAS

NATIONAL UNIVERSITY OF SINGAPORE

2006

Noisy Channels with Synchronization Errors: Information Rates and Code Design

JITENDER TOKAS

(*B.Tech. (Hons.), IIT Kharagpur, India*)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2006

Acknowledgements

I wish to thank Prof. Abdullah Al Mamun for being so patient and understanding. I am grateful to him for allowing me to explore and follow my interests.

I am indebted to Dr. Ravi Motwani for giving me the opportunity to work on this interesting and rewarding project. Working with him was a real pleasure. He has always been generous with his time, listening carefully and criticizing fairly.

I am grateful to Prof. Aleksandar Kavčić and Wei Zeng of DEAS, Harvad University for many insightful discussions and useful suggestions.

Lastly, I wish to acknowledge the love and support of my friends and family. This thesis is dedicated to my mom.

Contents

List of Figures

List of Tables

Abbreviations

AEP	asymptotic equipartition property
APP	a-posteriori probability
AWGN	additive white Gaussian noise
BCJR	Bahl-Cocke-Jelinek-Raviv
BER	bit error rate
FSC	finite-state channel
FSM	finte-state model
HMM	hidden Markov model
HMP	hidden Markov process
ISI	intersymbol interference
i.i.d	independent and identically distributed
i.u.d	independent and uniformly distributed
LB	lower bound
LDPC	low-density parity-check code
MAP	maximum a-posteriori probability
MR	magneto regressive
MS	Markov source
PR4	partial-response class-4 polynomial
SNR	Signal to Noise ratio
STP	state-transition probability
UB	upper bound

List of Symbols

X	random variable
x	realization of X
$\mathbf{x}$	row vector
$\mathbb{X}$	alphabet of x
X^n	sequence of random variables, i.e. $X^n = (X_1, X_2, \ldots, X_n)$
$\mathcal{X}, \{X^n\}$	random process
$\mathbb{B}$	antipodal binary set, $\{-1, 1\}$
$P(\cdot)$	probability
$p(\cdot)$	probability density function
$x\|y$	x conditioned on y
$E[\cdot]$	expectation operator
$C(f)$	channel frequency response
$g(t)$	Lorentzian pulse, step response
$h(t)$	impulse response
PW_{50}	width of Lorentzian pulse at 50% amplitude
T	symbol period
T_u	user bit period
K_u	normalized user density
K_c	normalized channel linear density
S_t	time-t state
D	delay operator

$\mathbf{Q}$	state transition matrix
π	state-distribution vector
$H(X)$	entropy
$h(X)$	differential entropy
$I(X;Y)$	mutual information between X and Y
$H(\mathcal{X})$	entropy rate
$I(\mathcal{X};\mathcal{Y})$	information rate
C	capacity
R	code rate
HS	header space
HL	header length
$\mathbf{H}$	parity-check matrix
$\mathbf{G}$	generator matrix

Summary

In this thesis we analyze communication channels which suffer from synchronization errors. Although synchronization errors are omnipresent in practical communication systems, their effect is usually negligible in the signal to noise ratio (SNR) range of interest. However, as the ever increasing potency of error-correcting codes pushes down the SNR limits for reliable communication, timing errors are expected to become the main performance limiting factor. Hence, it is important to study the effect of injecting timing errors in standard channels.

Most of the prior work in timing error channels focuses on insertion/deletion channels. Unfortunately, these channels are poor models of practical communication channels. In this work, we study a more realistic scenario than the insertion/deletion channel. In our model, we assume that timing errors can be quantized fractions of the symbol interval. To keep the problem mathematically tractable, we assume that the timing errors are generated by a discrete Markov chain. We investigate the information rates of baseband linear filter channels plagued by such timing errors and additive white Gaussian noise. The direct computation of the information rate for channels with memory is a difficult problem. Recently, practical simulation-based methods have been proposed to calculate information rates for finite-state intersymbol interference channels. These methods employ the entropy ergodic theorem and exploit the Markov property of the channels. In this report, we extend this strategy to include channels which also suffer from timing errors. Due to the very complex nature of the problem, we could not accurately compute the information rate for such channels. Instead, we propose Monte Carlo methods for computing upper and lower bounds on the mutual information rates. Excluding the high SNR regions, the channel capacity is tightly contained within the obtained upper and lower bounds.

We also investigate the problem of designing codes for channels corrupted by additive white Gaussian noise, intersymbol interference and timing errors. We propose serially concatenated codes for such channels. Marker codes form the inner code, which assists in providing probabilistic re-synchronization. Marker codes are decoded using a modified Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm, which produces soft estimates of timing offsets and input data. We provide simulation results to show the efficacy of marker codes in helping the receiver regain synchronization. However, marker codes are not powerful enough to protect against additive noise. Hence, the need for an outer code. A high-rate regular low-density parity-check (LDPC) code is used as the outer code. The soft-outputs of the marker decoder are fed into the LDPC decoder, which then produces an estimate of the transmitted data. Both the decoders recursively exchange extrinsic information about the data bits to better the estimation process. Simulation results are provided to evaluate the performance of the code.

Chapter 1

Introduction

Since its inception in 1948, information theory has been a subject of extensive research activity. In his seminal paper [1], Shannon provided fundamental limits on information rates for reliable transmission over noisy channel. This limit for a particular channel is termed as the capacity of that channel. Over the years, computing the capacity of communication channels has remained a significant challenge. Closed-form expressions for capacities of even simplistic channel models are still not available.

Recently, Monte Carlo methods were proposed to compute the mutual information rates of intersymbol interference (ISI) channels. In this thesis, we expand upon these techniques to obtain bounds on the capacity of noisy channels which also suffer from synchronization errors. We also design channel codes which are capable of correcting amplitude as well as synchronization errors.

1.1 Motivation

At some point in a digital communication receiver, an analog waveform must be sampled. Sampling at correct time instants is crucial to achieving good overall performance. The process of synchronizing the sampler with the pulses of the received analog waveform is known as timing recovery.

A practical receiver must perform three major tasks - timing recovery, equalization and/or detection and error-control decoding. Thus, in its operations, a receiver contends not only with the uncertainty in the timing of the pulses, but also with additive noise and ISI. An optimal receiver would have to perform these operations jointly by computing the maximum-likelihood estimates of the timing offsets and message bits. However, the complexity of such a receiver would be prohibitively high. Due to this, in conventional receivers these tasks are performed separately and sequentially. The order being timing recovery, followed by equalization and decoding (Fig. 1.1). A natural corollary of this design approach is that the timing recovery schemes ignore any error-correction coding used; instead, assume that the transmitted symbols are mutually independent. Also, the decoder works with the tacit assumption of perfect synchronization.

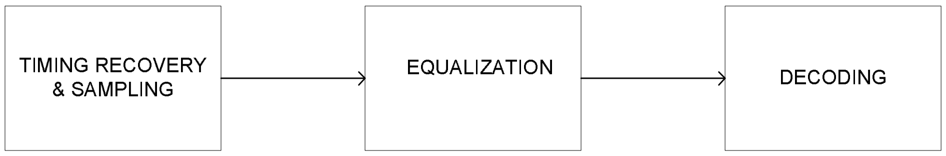


Fig. 1.1: Conventional timing recovery scheme.

However, virtually all timing recovery methods at the receiver produce synchronization errors. Communication systems and data storage systems are some of the real applications which suffer synchronization errors. Such synchronization errors are negligible in most conventional receivers, where the timing recovery units operate at very high signal-to-noise ratios (SNRs). With the advent of more powerful iteratively decodable codes, receivers are capable of operating at unprecedented low SNRs. Also, the future very high density storage systems will exhibit significantly high ISI, and consequently considerably lower SNRs. However, at such low SNRs, the conventional timing recovery schemes fail. This phenomenon can degrade the performance of the decoder, thus potentially offsetting the advantage obtained from using powerful error-correcting codes. For example, in magnetic recording systems, cycle slips in tracking increase steeply with reduction in SNR [2], thus deteriorating the system performance.

This problem can be remedied by modifying the timing recovery schemes in such a way that they are able to harness the power of the error-correcting codes. One method of doing this is performing timing recovery and error-correction decoding iteratively. Several different receiver configurations have been proposed to jointly perform timing-recovery and error-correction decoding using an iterative approach, with complexity comparable to a conventional receiver (see [3] for a good discussion). An obvious improvement to timing recovery schemes which work in conjunction with the decoder would be channel codes which aid in synchronization. Thus, knowing the capacity of channels with timing errors is not just an academic problem. The theoretical limits of transmission rates can serve as benchmark for design of codes which assist in timing recovery.

1.2 Literature Survey

Channels with synchronization errors have been receiving attention for a long time now. However, most of the previous work has concentrated on insertion/deletion channels. In [4], Dobrushin proved Shannon's theorem for memoryless channels with synchronization errors. He stated that the assumption of channel being memoryless can be relaxed; however, the proof for such channels is still unavailable.

In [5], Gallager obtained an analytical lower bound on capacity of memoryless deletion channels. He showed that for binary deletion channels, capacity can be bounded by a simple entropy function of the deletion probability. Much later, Diggavi and Grossglauser [6] extended these results to include non-binary alphabets. They also derived improved lower bounds by using a first order Markov chain for codeword generation. These results were further bettered in [7] by the use of more general processes for generating codewords. Ullman [8]used a combinatorial approach to derive upper and lower bounds on the capacity of insertion/deletion channels . However, the bounds are strong only in the special cases of single or multiple adjacent synchronization errors.

Dobrushin [9] presented a simulation based approach for estimating the capacity of deletion channels in. Recently, Motwani and Kavčić [10] computed lower bounds on the information rates of insertion and deletion channels using Monte-Carlo methods. These are the tightest lower bounds known for such channels. For deletion channels, their lower bound is very close to the upper bound given by Ullmann [8] which suggests that it lies very close to the channel capacity.

A large body of work exists on codes for channels with synchronization errors.

However, most of these coding scheme are applicable only in very restrictive scenarios and provide limited error-correction capability. Golomb et al. [11] developed “comma-free” codes which have the property that no overlap of codewords can be confused as a codeword. If a codeword is corrupted with an insertion or deletion, it is possible to regain re-synchronization after the error. Stiffler [12] and Tavares and Fukada [13] proposed adding a constant vector to binary cyclic codes to create comma-free codes with error-correction power of cyclic codes. However, none of these codes can correct insertion or deletion errors.

Another class of codes is based on the number-theoretic constructions employed by Levenshtein [14]. He defined a quantity *edit distance* (also called *Levenshtein distance*) which is the number of insertions, deletions or substitutions necessary to get one codeword from another. He presented codes capable of correcting single insertion and deletion and also proposed a decoding algorithm. Other codes based on Levenshtein distance were presented in [15], [16]. In [17] and [18], the authors proposed Viterbi decoders based on Levenshtein metric.

Sellers presented “marker codes” in [19]. In this scheme, a synchronizing *marker* sequence is inserted in the bit stream to be transmitted. The decoder looks for the markers and uses any shift in their position to deduce insertion or deletion errors. The codes that Sellers proposed could correct single or multiple adjacent synchronization errors and, in addition, correct a burst of substitution errors surrounding the position of synchronization errors. Recently, Davey and Mackay [20] extended marker codes to a more generalized “watermark code”. Instead of having localized markers, they spread the synchronization information evenly along the data sequence. They also provide a BCJR-like algorithm for the decoding of watermark codes. Watermark codes can be used in concatenation with other codes like LDPC codes to provide protection against additive noise. As the watermark decoder can produce soft outputs, even iterative decoding is possible. These codes are capable of correcting multiple insertion and/or deletion errors. Working in similar lines, Ratzer proposed an optimum decoding algorithm for marker codes in [21].

1.3 Objective of the thesis

In this thesis we analyze baseband linear filter channels which have timing errors injected in them. As can be seen in the previous section, most of the earlier works on channels with synchronization errors are restricted to the framework of insertion/deletion channels. Although these channels have great academic value, they are inadequate to model any practical channel. In this thesis, we look into a more realistic model of timing error channels. We have two main objectives:

- Our first aim is to quantize the loss in information rate that occurs on the introduction of timing errors in standard ISI channels. We are interested in the achievable mutual information rates of such channels.

- Our second aim is to design codes for noisy channels with synchronization errors. An effective code would have to be capable of combatting ISI, additive noise and synchronization errors. As our interest lies in the magnetic storage channels, we concentrate on high rate codes.

The main contribution of this thesis is a fundamental information theoretic result for channels with synchronization errors. We develop a practical method for tightly bounding the capacity of such channels. The application in mind here is magnetic recording, although the presented method is not restricted thereto.

1.4 Organization

This thesis is subdivided into 5 chapters. The first chapter contains the introduction and motivation behind this work.

In the second chapter we review some theorems and algorithms which will be used extensively in this thesis. In the first section we provide a brief introduction to baseband linear channels, with a little detail on magnetic storage channels. Then, we present finite-state models and their properties. In the following section, we provide a synopsis of the recently discovered simulation based method of computing information rates for finite state channels. In the last section we review low-density parity-check (LDPC) codes, their design and decoding.

The third chapter is dedicated to the computation of mutual information rate for timing error channels. We first present a Markov chain model for timing errors. We provide two different strategies for the trellis representation of our channel model. We present the timing error model that we use, along with its various trellis representations. We then describe Monte-Carlo methods which take advantage of the entropy ergodic theorem to upper bound and lower bound the information rate for said channels.

In the fourth chapter we present concatenated codes for timing error channels. The code is comprised of the serial concatenation of marker codes and LDPC codes. Marker codes provide probabilistic re-synchronization and LDPC codes protect against channel noise. The performance of the code is evaluated using simulation results.

The fifth chapter concludes the thesis and suggests some directions for future work.

Chapter 2

Technical Background

2.1 Baseband Linear Filter Channels

Most practical channels have constrained and finite bandwidth. Such channels may be modelled as linear filters having the same passband width as the channel bandwidth W Hz. The finite bandwidth assumption ensures that the frequency response of the channel has an equivalent lowpass representation. Hence, without any loss of generality, we can assume our channel to have a baseband rather than a passband frequency response. We refer to such channes as *baseband linear filter channels.* And they are characterized as a linear filter having a frequency response $C(f)$ that is zero for $|f| > W$, where W is the channel bandwidth.

Within the bandwidth of the channel, we express the frequency response $C(f)$ as

$$C(f) = |C(f)|e^{j\theta(f)}, \tag{2.1}$$

where $|C(f)|$ is the magnitude response characteristic and $\theta(f)$ is the phase response characteristic. Such channels are classified in two categories. A channel is called *ideal* if $|C(f)|$ is constant over its domain of definition and $\theta(f)$ is a linear function of frequency over its domain of definition. For both $|C(f)|$ and $\theta(f)$ the domain is given by $|f| \leq W$.

The channels which do not satisfy the above two conditions are called *distorting* channels. A channel whose $|C(f)|$ doesn't remain constant over $|f| \leq W$ is said to *distort the transmitted signal in amplitude.* And if for some channel $\theta(f)$ can't be expressed as a linear function of frequency, we say that the channel *distorts the transmitted signal in delay.*

A sequence of pulses when transmitted through a distorting channel at rates comparable to the channel bandwidth W get smeared into one another, and they are no longer distinguishable at the receiver. The pulses suffer dispersion in time domain and thus, we have ISI. In this thesis, we study the baseband linear filter channels which causes ISI. We shall also use the term ISI *channels* to refer to such channels. Digital magnetic recording channel is a prominent group in ISI channels and now we shall study them in detail.

2.1.1 Digital Magnetic Recording Channels

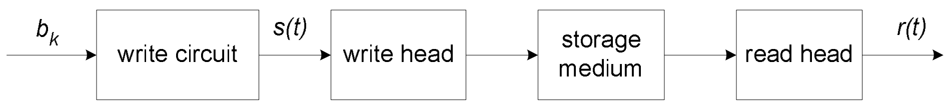


Fig. 2.1: Functional schematic of the magnetic read/write processes

The functional schematic of the read/write process in a conventional magnetic recording system is shown in Fig. 2.1. It consists of write-circuit, write-head/medium/read-head and associated pre-processing circuitry. For saturation magnetic recording, a binary data sequence $b_k = \{-1, 1\}$ is fed into the write-circuit at the rate of $1/T$ (T is the channel bit period). The write circuit is a linear modulator and it converts the bit sequence into a rectangular current waveform $s(t)$, whose amplitude swings between $+1$ and -1 corresponding to the input sequence b_k. This current in the write head induces a magnetic pattern on the storage medium. The direction of magnetization is opposite for $s(t) = +1$ and $s(t) = -1$. Evidently the information about the input bit sequence b_k is stored in the magnetization direction.

In the read-back process, the read head, either an inductive head or a magnetoresistive (MR) head, performs the flux-to-voltage conversion. It is not the medium magnetization, rather the magnetic transitions or the "derivatives" of the medium magnetization that are sensed by the read head. Therefore, an isolated magnetic transition corresponding to the data transition from -1 to 1 results in a waveform $g(t)$ of the read-back signal, while for the inverse transition $-g(t)$ is produced. This read-back voltage pulse is referred to as *isolated transition response.* Assuming that the linearity of channel is maintained in the course of read/write processes, the read-back signal can be reconstructed by the superposition of all transition responses resulting from the stored data pattern.

Formally, the recorded transition at time k is denoted by v_k, where

$$v_k = \begin{cases} 0 & \text{no transition at time } t = kT \\ \pm 1 & \text{otherwise.} \end{cases} \tag{2.2}$$

This notation corresponds directly to the sequence of magnetic transitions, and the sign of an element v_k denotes the direction of the transition (and of the polarization).

Note that $\{v_k\}$ is a correlated sequence, and is related to the sequence $\{b_k\}$ of write current polarities by

$$v_k = \frac{1}{2}(b_k - b_{k-1}), \tag{2.3}$$

with initial condition $b_0 = -1$. With these assumptions, we obtain a linear model for the read-back channel. The noiseless read-back signal can be written as

$$v(t) = \sum_k v_k g(t - kT) \tag{2.4}$$

$$= \sum_k b_k h(t - kT), \tag{2.5}$$

where

$$h(t) = \frac{1}{2}(g(t) - g(t - T)). \tag{2.6}$$

We note that $h(t)$ represents the effective impulse response of the magnetic recording channel as it corresponds to the response of head and medium to a rectangular pulse, i.e. to exactly two subsequent transitions (called dibit). In the literature, $h(t)$ is commonly termed *pulse response* or *dibit response*. Noting that electronics noise is added at the output of the read head, we can write the read-back waveform as

$$r(t) = \sum_k v_k g(t - kT) + \mu(t), \tag{2.7}$$

where $\mu(t)$ represents the electronics noise, which is usually modelled as additive white Guassian noise (AWGN). The linear channel model is shown in Fig. 2.2, where D is the delay operator.

The particular shape of $g(t)$ depends on the read head type. For MR read heads, which are currently standard in products, $g(t)$ can be well approximated by the

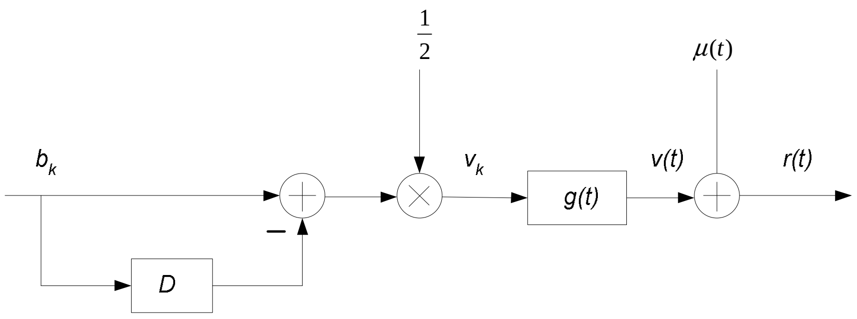


Fig. 2.2: Linear channel model

Lorentzian pulse [22]

$$g(t) = \frac{1}{1 + \left(\frac{2t}{PW_{50}}\right)^2}, \tag{2.8}$$

where PW_{50} is a parameter specifying the pulse width at half of the peak amplitude. PW_{50} is determined by the transition width in the recording media a and head-to-media distance d as follows [22]

$$PW_{50} = 2(a + d). \tag{2.9}$$

The ratio $K_c = PW_{50}/T$, where $\frac{1}{T}$ is the data rate, is a measure of the normalized linear density in a hard-disk system. It is the single most important parameter to characterize the channel in a magnetic recording system. Denoting the duration of user data bit by T_u, the quantity defined as $K_u = PW_{50}/T_u$ is called the normalized user density, which is a measure of the linear density from user's point of view.

Assuming R_c to be the code-rate of the channel encoder, we have $T = R_c T_u$, and consequently $K_c = K_u/R$. Hence, the use of channel code will cause increase in linear density. However, the channel pulse response $g(t)$ as well as the noise variance are functions of the sampling rate $1/T$. Higher recording density implies increased sampling rate and consequently, the noise variance at the input of the read channel detector increases because of bandwidth expansion. Moreover, the energy in the pulse response decreases because the positive transition and negative transition cancel each other more, leading to further decreased SNR. Since it is difficult to achieve coding gain large enough to compensate for the rate loss, only very high rate codes are useful in magnetic recording channels.

We now describe in detail finite-state models (FSM), which are pivotal in the mathematical modelling of magnetic recording channels.

2.2 Finite-State Models

An FSM is a doubly stochastic random process. It has two parts - a non-observable state process $\mathcal{S}$ and an observable output process $\mathcal{Y}$. The state process is of finite size, i.e. its cardinality $L = |\mathbb{S}| < \infty$ and determines the structure of the finite-state model. Whereas, the observable output process can take values from a finite or infinite alphabet set. The output process can be a deterministic or probabilistic function of the underlying state process and inherits its statistical properties.

When the unobservable state process of an FSM is a Markov process, the FSM is referred to as a Hidden Markov Model (HMM) (see [23] for an excellent tutorial introduction). It is worthwhile to note here that HMMs can be extended to infinite state-space [24]. The observable output sequence of such a model is known as a Hidden Markov Process (HMP). The random variables which form the output sequence are conditionally independent, given the underlying Markov process. HMMs form a large and useful class of stochastic process models and find application in a wide range of estimation, signal processing, and information theory problems. We will use the notion of finite-state model for HMM with finite state-space.

2.2.1 Structure

States and state-transitions

The structure of an FSM is determined by its states and the branches connecting the states. The *state-space* $\mathbb{S}$ is a non-empty set of finite cardinality and consists of elements called *states*. The cardinality $L = |\mathbb{S}|$ of the state-set is called the *order* of the FSM. Let $\mathbb{B}$ be a finite set, the elements of which will be termed as *branches* or *state-transitions*. Every branch $c \in \mathbb{B}$ has a well defined *left state* $\text{Lstate}(c) \in \mathbb{S}$ and a well defined *right state* $\text{Rstate}(c) \in \mathbb{S}$.

A path of length n in an FSM is a sequence $c^n = (c_1, c_2, \ldots, c_n)$ of branches $c_k \in \mathbb{B}$, $k = 1, 2, \ldots, n$, such that $\text{Rstate}(c_k)$=$\text{Lstate}(c_{k+1})$. Each branch sequence has a unique state sequence s^n associated with it.

Trellis Representation

An FSM can be represented by a directed graph known as the *state-transition diagram*. Any two states s' and s'' ($s', s'' \in \mathbb{S}$) are connected by a directed edge *iff* $\exists\ c \in \mathbb{B}$ such that $\text{Lstate}(c) = s'$ and $\text{Rstate}(c)$=s''.

Unfolding the state-transition diagram over time results in the *trellis representation* of the FSM. A trellis of length n consists of n concatenated *trellis sections*. A trellis section T_t at time t is characterized by S_t and C_t, which are the time-t state-set and time-t branch-set respectively. Each branch in C_t has a well defined *left state* and a well defined *right state*. More precisely, $\text{Lstate}(C_t)$=S_t and $\text{Rstate}(C_t)$=S_{t+1}. If the state process is time invariant, all trellis sections are identical and the time index t is dropped.

Example 2.1 (**DICODE Channel**). *Consider a discrete-time channel with frequency response* $(1-D)/\sqrt{2}$. *The input-output relation for this channel is given by* $Y_t = (Y_t - Y_{t-1})/\sqrt{2}$, *where* Y_t *is the time-t input. We assume that the input signal is bipolar, i.e.* $X_t \in \{+1, -1\}$. *The time-t state is given by the time-t input in the following way:* $S_t = (X_t + 3)/2$. *The state-transition diagram and the corresponding trellis representation are showon in Fig. 2.3, with the associated input and output pair* X_t/Y_t *on each branch.*

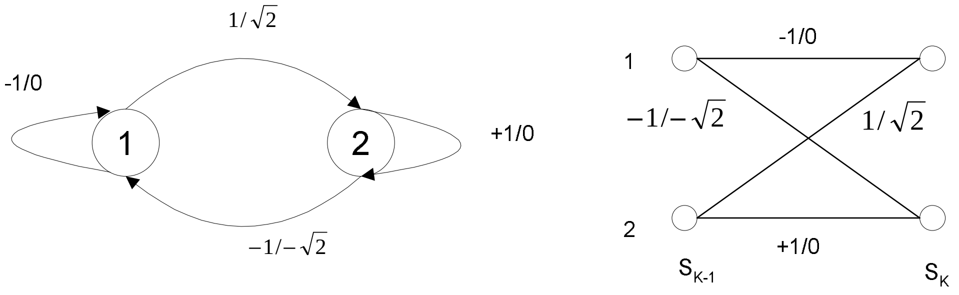


Fig. 2.3: State transition diagram and a trellis section of the DICODE channel.

2.2.2 Markov Property

We assume that the unobservable state process is a *first order Markov process*. This implies that the probability of being in the state j at time t conditioned on all the states up to the state i at time $t-1$, depends only on the state i at time $t-1$. More formally,

$$P(S_t = j | S_{t-1} = i, S_{t-2} = i', \ldots) = P(S_t = j | S_{t-1} = i). \tag{2.10}$$

The above relation is known as the *Markov property.* The probability of going from state $S_{t-1} = i$ to state $S_t = j$ is called the *state-transition probability* (STP). It is convenient to arrange the STPs in a $L \times L$ *state-transition probability matrix* $\mathbf{Q}$, where the entry in row i and column j equals the corresponding STP, i.e.

$$\mathbf{Q}(i,j) \triangleq P(S_t(j)|S_{t-1}(i)). \tag{2.11}$$

In the above equation, $S_t(j)$ denotes that at time t state is j. Clearly, $\mathbf{Q}$ is a matrix whose entries are all nonnegative, and elements in each row add to unity, since

$$\sum_{j \in \mathcal{S}} \mathbf{Q}(i,j) = \sum_{j \in \mathcal{S}} P(S_t(j)|S_{t-1}(i)) = 1 \qquad \forall\, i \in \mathbb{S}. \tag{2.12}$$

In general, the state-transition probabilities of a Markov source may depend on time. Here we discount this possibility and thus, assume that the Markov process is *homogeneous* in time. Such Markov processes are known as *Markov chains* [25].

2.2.3 Classification of States

Any state i is said to be accessible from state j if there is a finite sequence of transitions from j to i with positive probability. If i and j are accessible from each other, they are said to *communicate* with each other. A Markov chain in which any state is accessible from any other state is termed as *irreducible* (*communicating chain*). All states of such a chain belong to a single class and for every pair (s, s') of states, there exists a finite and positive integer n such that

$$P(S_{t+n} = s'|S_t = s) > 0. \tag{2.13}$$

Persistent States

The state i is said to be *persistent* if starting from state i, return to state i is certain, i.e. if

$$P\Big(\bigcup_{t=1}^{\infty}[S_t = i] | [S_0 = i]\Big) = 1. \tag{2.14}$$

Any state that is not persistant is called *transient.* A Markov chain is persistent if all its states are persistent.

Aperiodic States

Consider a Markov chain with a finite state-space. A state i is said to be *periodic* with period T, if return to that state is possible only at instants $T, 2T, 3T, \dots$ (multiples of T), where T is the largest integer with this property. A state with period $T = 1$ is called *aperiodic.* A Markov chain is aperiodic if all its states are aperiodic.

Since in an irreducible Markov chain all states belong to the same class, they are either all transient or all persistent. Similarly, all states are either aperiodic or periodic with the same period. Moreover, if the Markov chain is finite, (2.13) guarantees that the Markov chain is persistent. Thus, for finite Markov chains persistence follows from the property of irreducibility.

2.2.4 Stationary State Distribution

Let the row vector $\pi^{(t)}$ of length L be the *state distribution vector* of a Markov chain at time t. The i^{th} element of $\pi^{(t)}$ is thus the probability of being in state i at time t, i.e.

$$\pi^{(t)}(i) \triangleq P(S_t = i). \tag{2.15}$$

Given the state distribution at time $t-1$, the state distribution at time t can be written as

$$\pi^{(t)} = \pi^{(t-1)}\mathbf{Q}. \tag{2.16}$$

By iteration, we obtain

$$\pi^{(t)} = \pi^{(0)}\mathbf{Q}^{t}, \tag{2.17}$$

where $\pi^{(0)}$ is the initial state distribution vector. A Markov chain is said to be *stationary* if and only if it has a *stationary* state distribution π such that $\pi^{(t)} = \pi \; \forall \, t$ or equivalently,

$$\pi = \pi\mathbf{Q}. \tag{2.18}$$

It is important to note that (2.18) may not always have a unique solution.

Convergence to the Stationary Distribution

For a finite-state irreducible Markov chain, the stationary state distribution is positive, i.e. $\pi(s) > 0 \; \forall s \in \mathbb{S}$ and *unique* [23]. Thus, $\mathcal{S}$ is a stationary process. The next question is whether any initial state distribution converges to the stationary state distribution.

If a finite-state Markov chain is irreducible and *aperiodic*, it holds that all its states are ergodic [25], i.e.

$$\lim_{n\to\infty} [\mathbf{Q}^{n}]_{ij} = \pi(j) \quad \forall \, i, j \in \mathbb{S} \tag{2.19}$$

and the Markov chain is said to be an *ergodic process.* From (2.17), it follows that for $n \to \infty$

$$\pi = \pi^{(0)}\mathbf{Q}^{\infty}, \tag{2.20}$$

i.e. any intial state distribution converges to the stationary distribution which is then called *steady state distribution.* Note that a sufficient, but not necessary, condition for a Markov chain to be ergodic is aperiodicity [25].

2.2.5 Ergodicity Theorem for Markov Chains

We summarize the important properties of finite-state, irreducible, and aperiodic Markov chains in the following theorem.

Theorem 2.1. *Let a finite-state Markov chain with a stochastic state-transition matrix $\boldsymbol{Q}$ be irreducible and aperiodic. All its states are ergodic and the chain form an ergodic process. The chain has a unique stationary distribution, to which it converges from any initial state. This distribution π is called the steady state distribution and satisfies the following properties:*

1. $\lim_{n\to\infty}[\mathbf{Q}^n]_{ij} = \pi(j) \quad \forall\, i, j \in \mathbb{S}$
2. $\pi(j) > 0 \quad \forall\, j \in \mathbb{S}$
3. $\sum_{j\in\mathbb{S}} \pi(j) = 1$
4. $\pi(j) = \sum_{i\in\mathbb{S}} \pi(i)\mathbf{Q}(i,j) \quad \forall\, j \in \mathbb{S}.$

2.2.6 Output Process

The output process $\mathcal{Y}$ of an HMM is observable unlike the state process. Moreover, a realization y_t at time t is not restricted to being discrete. Given the realization $S_0^n = (S_0, S_1, \ldots, S_n)$ of the underlying state process, the output sequence $Y^n = (Y_1, Y_2, \ldots, Y_n)$ is a collection of conditionally independent random variables. The distribution of Y_t is time-invariant and it depends on $\mathcal{S}$ only through S_t.

The n-dimensional density of $(\mathcal{Y}, \mathcal{S}) \equiv (Y^n, S_0^n)$ can thus be written as

$$p(y^n, s_0^n) = p(s_0) \prod_{k=1}^{n} p(y_k, s_k | s_{k-1}). \tag{2.21}$$

We also have the following relation

$$P(y_t, s_t | s_{t-1}) = P(y_t | s_t, s_{t-1}) P(s_t | s_{t-1}) \quad t = 1, 2, \ldots \tag{2.22}$$

Observing that

$$p(y^n) = \sum_{s_0^n} p(y^n, s_0^n) \tag{2.23}$$

and using (2.22), we can write the probability distribution of the output process as

$$\begin{align} p(y^n) &= \sum_{s_0^n} p(s_0) \prod_{k=1}^{n} p(y_k | s_k, s_{k-1}) P(s_k | s_{k-1}) \tag{2.24} \\ &= \sum_{s_0^n} \pi(s_0) \prod_{k=1}^{n} p(y_k | s_k, s_{k-1}) \mathbf{Q}(s_k, s_{k-1}). \tag{2.25} \end{align}$$

If the FSM represents a communication channel, the time-t state S_t is given by some previous channel inputs or a combination of channel inputs and internal channel states.

Theorem 2.2 (**Output Ergodicity**). *The output process $\mathcal{Y}$ of a aperiodic and irreducible finite-space Markov process is stationary and ergodic.*

This theorem follows from the fact that given a realization of the hidden state sequence, the output signals are conditionally independent random variables [26].

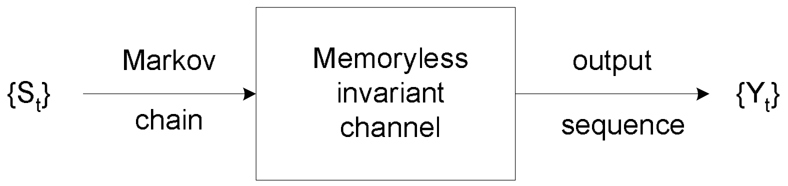


Fig. 2.4: A hidden Markov process

2.3 BCJR Algorithm

From Information theory point of view, an HMP is a discrete-time finite-state homogenous Markov chain observed through a discrete-time memoryless time invariant channel as described in Fig. 2.4. The BCJR algorithm [27] is used to estimate the a-posteriori probabilities (APPs) of the states and transitions of the hidden source, given the observable output sequence. The algorithm can be easily modified to include channels with memory as well. The channel memory can be viewed as a Markov source, which can be combined with the input symbol source to create a *super-source*. The original channel appears to be memoryless to this *super-source* and hence, the BCJR algorithm can be used to obtain the APPs.

The BCJR algorithm is a symbol-by-symbol maximum a-posteriori (MAP) algorithm. We will now briefly describe the BCJR algorithm. We will skip the intermediate steps wherever they directly follow from the arguments presented in [27].

Let us assume that we have a finite-state Markov source transmitting symbols over an AWGN channel of variance σ^2. Let X_1^N be the input data sequence emitted by the Markov source and $S_1^N \in \mathbb{S}^N$ be the state sequence corresponding to the input data. Y_1^N represents the observed output, when the input data sequence X_1^N is sent over the channel. We further assume that the data symbol X_t corresponds to the transition from state S_{t-1} to state S_t. In what follows, the variables s and s' will be used to index the states of the Markov source.

Central to the BCJR algorithm are the following two properties of an HMP:

$$\{S_t^N; Y_t^N\} \perp\!\!\!\perp \{S_1^{t-2}; Y_1^{t-1}\} | S_{t-1} \tag{2.26}$$

and

$$Y_t \perp\!\!\!\perp \{S_\neg, Y_\neg\} | S_{t-1}^t. \tag{2.27}$$

Note that $X \perp\!\!\!\perp Y|Z$ signifies that X is independent of Y given Z. Equation (2.27) states that given an assignment to S_{t-1}^t, the distribution of Y_t is independent of every other variable (both in the past and the future) in the HMP. Equations (2.26) and (2.27) imply an assortment of conditional independence statements, which are used in the derivation of the BCJR algorithm.

Our aim is to compute the following two quantities for each time index:

$$P(S_t = s|y_1^N) = P(S_t = s; y_1^N)|p(y_1^N) \tag{2.28}$$

and

$$P(S_{t-1} = s', S_t = s|y_1^N) = P(S_{t-1} = s', S_t = s; y_1^N)|p(y_1^N). \tag{2.29}$$

However, it is easier to derive the joint probabilities

$$\lambda_t(s) = P(S_t = s; y_1^N) \tag{2.30}$$

and

$$\sigma_t(s', s) = P(S_{t-1} = s'; S_t = s; y_1^N). \tag{2.31}$$

Since $p(y_1^N)$ is a constant for a given y_1^N, we can readily obtain the conditional probabilities of (2.28) and (2.29) once we have $\lambda_t(s)$ and $\sigma_t(s', s)$. The algorithm consists of two independent forward and backward recursions. Before describing the recursive relations, we define a few quantities:

- *forward state-metric*

$$\alpha_t(s) = P(S_t = s; y_1^t) \tag{2.32}$$

- *backward state-metric*

$$\beta_t(s) = p(y_{t+1}^N | S_t = s) \tag{2.33}$$

- *branch metric*

$$\gamma_t(s', s) = p(S_t = s, y_t | S_{t-1} = s') \tag{2.34}$$

The above quantities can be used to calculate $\lambda_t(s)$ and $\sigma_t(s', s)$ by the following equations

$$\sigma_t(s', s) = \alpha_{t-1}(s')\gamma_t(s', s)\beta_t(s), \tag{2.35}$$

$$\lambda_t(s) = \sum_{s'} \sigma_t(s', s). \tag{2.36}$$

The quantity $\gamma_t(s', s)$ can be computed by

$$\begin{aligned}\gamma_t(s', s) &= p(S_t = s, y_t | S_{t-1} = s') \\ &= P(S_t = s | S_{t-1} = s') \cdot p(y_t | S_{t-1} = s', S_t = s) \\ &= P(S_t = s | S_{t-1} = s') \cdot K \cdot e^{-\frac{\|y_t - x_t\|^2}{2\sigma^2}},\end{aligned} \tag{2.37}$$

where K is a scaling factor and x_t is the symbol emitted by the Markov source when a transition from state s' to state s occurs.

The *forward recursion* used to compute α is given by

$$\alpha_t(s) = \sum_{s'} \alpha_{t-1}(s')\gamma_t(s', s). \tag{2.38}$$

The *backward recursion* used to compute β is given by

$$\beta_t(s) = \sum_{s'} \beta_{t+1}(s')\gamma_{t+1}(s', s). \tag{2.39}$$

If we impose the constraints that the Markov source must start and end at the state 0, then we have the following initializations for α and β respectively

$$\alpha_0(s) = \begin{cases} 1 & s = 0 \\ 0 & s \neq 1 \end{cases},$$

$$\beta_N(s) = \begin{cases} 1 & s = 0 \\ 0 & s \neq 1 \end{cases}. \tag{2.40}$$

By equations (2.30) - (2.40) we have completely described the BCJR algorithm. However, before we move ahead, we will make one more observation which will prove to be pivotal in the next section. We can estimate the probability $p(y_1^n)$ using the forward recursion of the algorithm as follows:

$$\begin{aligned} p(y_1^N) &= \sum_s P(S_N = s; y_1^N) \\ &= \sum_s \alpha_N(s) \end{aligned} \tag{2.41}$$

2.4 Information Rates and Capacity

2.4.1 Some Definitions

In this section we briefly review some of the well-known results in information theory which are relevant to this work and follow thereby, the book of Cover and Thomas [28] very closely.

Entropy and Mutual Information

Definition 2.1 (Entropy). *The entropy $H(X)$ of a discrete random variable X with alphabet $\mathbb{X}$ and probability mass function (p.m.f.) $p_X(x) = P\{X = x\}$ (the subscript will be omitted) is defined by*

$$H(X) \triangleq -\sum_{x\in\mathbb{X}} p(x)\log_2 p(x). \tag{2.42}$$

The logarithm is to the base 2 and entropy is expressed in bits. The entropy does not depend on the actual values taken by X, but only on probabilities.

Definition 2.2 (Conditional Entropy). *The entropy of a discrete random variable X conditioned on a discrete random variable Y is given by*

$$H(X|Y) \triangleq -\sum_{y\in\mathbb{Y}} p(y)\sum_{x\in\mathbb{X}} p(x|y)\log_2 p(x|y). \tag{2.43}$$

The differential entropies and conditional differential entropies of continuous valued random variables are defined by replacing the summation with an integration. They are denoted by the lower case "h", i.e. $h(X)$ and $h(X|Y)$.

Definition 2.3 (Mutual Information). *The mutual information between two random variables X and Y with joint p.m.f $p(x, y)$ and marginal p.m.f $p(x)$ and $p(y)$*

respectively, denoted by $I(X;Y)$ is the relative entropy between the joint distribution and the product distribution $p(x)p(y)$, i.e.

$$I(X;Y) \triangleq \sum_{x\in\mathbb{X}, y\in\mathbb{Y}} p(x,y)\log_2 \frac{p(x,y)}{p(x)p(y)}. \tag{2.44}$$

In terms of entropies, we can write the mutual information as

$$I(X;Y) = H(X) - H(X|Y) \tag{2.45}$$

$$= H(Y) - H(Y|X). \tag{2.46}$$

In the case of continuous random variables, differential entropies are used.

Entropy Rate of Stochastic Processes

Definition 2.4 (Entropy Rate)**.** *The entropy rate of a stochastic process $\mathcal{X}$ is defined by*

$$H(\mathcal{X}) \triangleq \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \ldots X_n) \tag{2.47}$$

$$\triangleq \lim_{n\to\infty} H(X_n|X_{n-1}, X_{n-2}, \ldots, X_1) \tag{2.48}$$

when the limit exists. In the first line, the right hand side term is the per-symbol entropy rate. In the second line, the right hand side term is the conditional entropy rate of the last random variable given the past. For stationary stochastic processes both are equal.

The entropy rate is the average description length for a stationary ergodic process. It is well defined for stationary processes.

Example 2.2 (**Entropy Rate of a Markov Chain**).

$$H(\mathcal{X}) = \lim_{n\to\infty} H(X_n|X_{n-1},\dots,X_1) = H(X_n|X_{n-1}),$$

where the second equality follows from the Markov property.

Asymptotic Equipartition Property

The asymptotic equipartition property (AEP) in information theory is the analog of the law of large numbers. The AEP states that the probability of occurrence of a sequence of process $\mathcal{X}$ is close to $2^{-nH(\mathcal{X})}$ with probability 1 for $n \to \infty$. This permits us to divide the set of all sequences in two sets, the *typical set*, where the sample entropy is close to the true entropy and the set of *non-typical* sequences containing all other sequences. We first present theorem for AEP of independent and identically distributed (i.i.d) processes, and later extend it to general processes.

Theorem 2.3 (**Asymptotic Equipartition Property [28]**). *If $X_1, X_2, \dots$ are i.i.d and distributed according to $p(x)$, then*

$$-\frac{1}{n}\log_2 p(X_1,X_2,\dots,X_n) \to H(X) \quad \textit{in probability.} \tag{2.49}$$

Proof: Function of independent random variables are also independent random variables. Since the X_i are i.i.d, we can write

$$\begin{aligned} -\frac{1}{n}\log_2 p(X_1,X_2,\dots,X_n) &= -\frac{1}{n}\sum_i \log_2 p(X_i) && (2.50)\\ &\to -E[\log_2 p(x)] \quad \text{in prob.} && (2.51)\\ &= H(X). && (2.52)\end{aligned}$$

Definition 2.5 (Typical Set [28]). *Let $X_1, X_2, \ldots, X_n$ be i.i.d random variables. The typical set $A_\epsilon^{(n)}$ with respect to $p(x)$ is the set of sequences $(x_1, x_2, \ldots, x_n) \in \mathbb{X}^n$ with the following property:*

$$2^{-n(H(X)+\epsilon)} \leq p(x_1, x_2, \ldots, x_n) \leq 2^{-n(H(X)-\epsilon)}. \tag{2.53}$$

As a consequence of the AEP, the typical set has probability nearly 1 and all elements of the typical set are nearly equiprobable. The elements of the set are called typical sequences and their number is nearly $2^{nH(X)}$.

AEP for Ergodic Processes

The Shannon-McMillan-Brieman theorem [28] is the fundamental theorem for the AEP for stationary ergodic processes with finite alphabets. It states that for long sequences, the entropy rate is proportional to the logarithm of the probability of a typical sequence. Similar results have been extended to differential entropy rate by Barron [29] for processes with infinite alphabet. For the particular case where the process is an ergodic finite-state HMP, i.e. the output process of an FSM, Leroux provided an elegant proof in [26].

To gain insight into the practical implications of the Shannon-McMillan-Brieman theorem, let us consider a discrete stationary and ergodic random process $\mathcal{X}$. We define the sample sequence entropy as follows

$$H(x^n) \triangleq -\log_2 p(x^n). \tag{2.54}$$

The ensemble entropy $H(X^n)$ is the average of sample sequence entropy, i.e.

$$H(X^n) \triangleq \sum_{x^n \in \mathbb{X}^n} p(x^n) H(x^n) \tag{2.55}$$

$$= -\sum_{x^n \in \mathbb{X}^n} p(x^n) \log_2 p(x^n). \tag{2.56}$$

For infinitely long sequences, it converges to the true entropy rate of the process $\mathcal{X}$, i.e.

$$H(\mathcal{X}) = \lim_{n \to \infty} \frac{1}{n} H(X^n). \tag{2.57}$$

For large n, almost all sequences are typical and are equiprobable. Thus, $H(x^n)/n$ converges to $H(X^n)/n$. But in the case of stationary process, $H(X^n)/n$ approaches $H(\mathcal{X})$ for large n. Therefore, we conclude that for large n, average sample sequence entropy converges to true entropy, i.e.

$$-\frac{1}{n} \log_2 p(x^n) \to H(\mathcal{X}) \tag{2.58}$$

with probability one, provided that the process $\mathcal{X}$ is stationary and ergodic. Thus, the entropy of an ergodic random process can be estimated using just a single large realization of the process.

Information Rate and Capacity of Ergodic Processes

Definition 2.6 (Information Rate). *The information rate between two stationary* *processes $\mathcal{X}$ and $\mathcal{Y}$ is defined as*

$$I(\mathcal{X};\mathcal{Y}) \triangleq \lim_{n\to\infty} \frac{1}{n} I(X_1, X_2, \ldots, X_n; Y_1, Y_2, \ldots, Y_n) \tag{2.59}$$

$$= H(\mathcal{X}) - H(\mathcal{X}|\mathcal{Y}) \tag{2.60}$$

$$= h(\mathcal{Y}) - h(\mathcal{Y}|\mathcal{X}) \tag{2.61}$$

provided the limit exits. Note that we assume that $\mathcal{X}$ is finite-valued and $\mathcal{Y}$ is continuous-valued random process.

Referring to $\mathcal{X}$ as input process and $\mathcal{Y}$ as output process of a communication channel, the limit in (2.59) exists if the channel preserves the stationarity and ergodicity of the input process. Such channels are termed as *egrodic channels.*

Definition 2.7 (**Capacity of Ergodic Channels**)**.** *The capacity between a stationary and ergodic input process $\mathcal{X}$ and a stationary and ergodic output process $\mathcal{Y}$ is defined as*

$$C \triangleq \lim_{n\to\infty} \max_{p(x^n)} \frac{1}{n} I(X_1, X_2, \ldots, X_n; Y_1, Y_2, \ldots, Y_n). \tag{2.62}$$

2.4.2 Capacity of Finite-State Channels

A finite-state channel (FSC) is a discrete-time channel where the distribution of the channel output depends on both the channel input, and the underlying channel state. This allows the channel output to depend implicitly on previous inputs and outputs via the channel state.

In practice, there are three types of channel variations which FSCs are typically used to model. Firstly, *flat fading* channel where the channel state is independent of the channel inputs. Secondly, ISI channels where the channel state is a deterministic function of the previous channel inputs. The third type is channels which exhibit both fading and ISI, e.g. *frequency-selective fading* channels. In such channels, the channel state is a stochastic function of the previous inputs. We now give the formal definition of a FSC.

Definition 2.8 (**Finite-State Channels** [30]). *The output y_t at time t of a finite state channel is statistically independent of the state at time t, given the state at time $t-1$ and the channel input at time t, i.e.*

$$P(y_t, s_t|s_{t-1}, x_t) = P(y_t|s_{t-1}, x_t) \cdot P(s_t|s_{t-1}, x_t). \tag{2.63}$$

It is worth noting that the term $P(s_t|s_{t-1}, x_t)$ controls the evolution of states in the channel. An important set of well-behaved FSCs is the class of *indecomposable* FSCs. For an indecomposable FSC, the effect of starting state s_0 dies away with time.

Definition 2.9 (**Indecomposable Finite-State Channels** [30]). *An FSC is indecomposable if for any arbitrarily small $\epsilon > 0$, there exists a n' such that for $n > n'$,*

$$|P(s^n|s_0, x^n) - P(s^n|s_0', x^n)| \leq \epsilon \tag{2.64}$$

for all s^n, x_n, s_0 and s_0'.

It can be easily observed that an indecomposable FSC is equivalent to an irreducible and aperiodic FSM. Hence, the output process of an indecomposable FSC can be viewed as an HMP. We define the capacity of indecomposable FSCs as follows.

Definition 2.10 (**Capacity of Indecomposable Finite-State Channels**). *The capacity of an indecomposable FSC starting in state s_0, driven by a discrete input X_t*

is given by

$$C \triangleq \lim_{n\to\infty} \max_{K^n} \max_{s_0} \frac{1}{n} I(X^n; Y^n | S_0 = s_0), \tag{2.65}$$

where $K^n = \{P_{X^n}(x^n) : x^n \in X^n\}$ *denotes the allowed input p.m.f.s .*

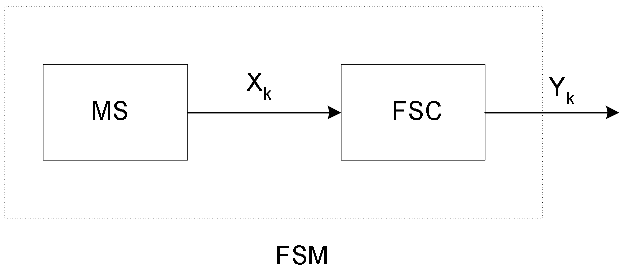


Fig. 2.5: Finite-state model studied in Sec. 2.4.3, comprising of an FSC driven by a Markov source (MS)

2.4.3 A Monte Carlo Method for Computing Information Rates

This method for computing mutual information rates of Markov sources over indecomposable FSCs was presented independently in [31], [32], [33]. It is simply an efficient application of the Shannon-McMillan-Breiman theorem.

Problem Statement: *We have an indecomposable FSC channel, driven by a finite-state Markov source (Fig. 2.5). We have to obtain the mutual information rate between a finite-state input process $\mathcal{X}$ (or equivalently the state process $\mathcal{S}$) and the channel output process $\mathcal{Y}$.*

An FSC can be represented by an FSM. Moreover, any finite-state Markov source (MS), representing the input process to the FSC, can be combined with the channel FSM to form a single *joint source/channel* FSM. It is this FSM that we work with.

As both the input process $\mathcal{X}$ and the output process $\mathcal{Y}$ are assumed to be stationary and ergodic, Shannon-McMillan-Breiman theorem is applicable.

Computing the Entropy Rate $h(\mathcal{Y})$

We begin by noting that

$$h(\mathcal{Y}) = \lim_{n\to\infty} \frac{1}{n} h(Y^n) \tag{2.66}$$

$$= \lim_{n\to\infty} -\frac{1}{n} E_{p(Y^n)}[\log_2 p(Y^n)]. \tag{2.67}$$

In our context, the entropy rate $h(\mathcal{Y})$ is the entropy of our FSM. We generate a very long output sequence y^n by passing a random input sequence through the channel and sampling the corresponding channel output. The probability $p_{Y^n}(y^n)$ can be computed by the forward recursion of the BCJR algorithm as shown in (2.41). Recall from (2.41)

$$p(y_1^n) = \sum_s \alpha_n(s), \tag{2.68}$$

where $\alpha_n(s)$ is the forward state-metric of state s at time t. In practice, the forward and backward state metrics quickly tend to zero and numerical underflow occurs. To avoid this, at each trellis section the states-metrics are normalized. The normalizing factor for the forward state-metric at time t is given by

$$\mu_t = \sum_s \alpha_t(s). \tag{2.69}$$

The forward recursion (2.38) is now

$$\tilde{\alpha}_t(s) = \sum_{s'\in\mathbb{S}} \alpha_{t-1}(s')\gamma_t(s', s), \tag{2.70}$$

$$\alpha_t(s) = \frac{1}{\mu_t}\tilde{\alpha}_t(s). \tag{2.71}$$

It can be easily observed that $p(y_1^n)$ is now given by

$$\begin{aligned} p(y_1^n) &= \prod_{t=1}^{n} \mu_t \underbrace{\sum_s \alpha_t(s)}_{=1} \\ &= \prod_{t=1}^{n} \mu_t. \end{aligned} \tag{2.72}$$

As the output process $\mathcal{Y}$ is ergodic, it follows that an estimate of $h(Y^n) = -E_{p(Y^n)}[\log_2 p(Y^n)]$ can be obtained using a single very long sequence y_1^n. Hence, the computed value of the entropy rate $h(\mathcal{Y})$ for a finite n is then given by

$$\hat{h}(\mathcal{Y}) = -\frac{1}{n}\log_2 p_{Y^n}(y^n) \tag{2.73}$$

$$= -\frac{1}{n}\sum_{t=1}^{n}\log_2 \mu_t. \tag{2.74}$$

The right hand side converges to $h(\mathcal{Y})$ for $n \to \infty$ with probability one owing to the Shannon-McMillan-Breiman theorem. By increasing n, the entropy rate can be obtained to any desired level of accuracy.

As the underlying FSM is indecomposable by assumption, the effect of the starting state fades away with the progression of time. Hence, for long sequences, we can start in any state without affecting the estimate $\hat{h}(\mathcal{Y})$.

Computing the Conditional Entropy Rate $h(\mathcal{Y}|\mathcal{X})$

Given the input sequence x^n and the output sequence y^n, the probability $p_{Y^n|X^n=x^n}(y^n|x^n)$ can be computed by the forward recursion of the BCJR algorithm. Contrary to the computation of $p(y^n)$, the BCJR algorithm now operates on a reduced trellis. As the input sequence is known, there is only one allowed transition from any state in the trellis (the one corresponding to the input value at that instant). The reduced trellis is time-varying as it is induced by the input sequence x^n, which is random. However, since the reduced trellis originates from an irreducible and aperiodic trellis, it is irreducible and aperiodic too. Once we compute $p_{Y^n|X^n=x^n}(y^n|x^n)$, an estimate of $h(\mathcal{Y}|\mathcal{X})$ can be obtained using arguments similar to those presented in the previous paragraph.

For many practical channels $h(\mathcal{Y}|\mathcal{X})$ can be obtained analytically. For example, for ISI channels corrupted with AWGN, the knowledge of noise variance is sufficient to compute $h(\mathcal{Y}|\mathcal{X})$.

2.5 Low-Density Parity-Check Codes

LDPC codes are a class of linear error-correcting codes. They were first introduced in 1962 by Gallager [34], [35]. Despite the fact that LDPC codes would have broken all practical coding records prior to 1993, they were largely forgotten for many years. It is likely that storage requirements of encoding, and computational demands of encoding made their immediate adoption infeasible. However, there has been a renewed interest in LDPC codes since their rediscovery in the last decade [36]. In this section, we review the decoding and structured construction of LDPC codes.

Parity-check codes

A parity-check code is a binary block code which uses a generator matrix $\mathbf{G}$ to map the source words $\mathbf{u}$ to codewords $\mathbf{c} := \mathbf{uG}$ (where $\mathbf{u}$ and $\mathbf{c}$ are row vectors). An N block length code can be equivalently described by a $M \times N$ parity-check matrix

H. The M rows of **H** specify a set of M constraints which all the codewords must satisfy. Thus, the parity-check code is the set of of binary vectors that comply with the constraints imposed by **H**, i.e.

$$\mathrm{C} = \{\mathbf{c} \in 2^N \;:\; \mathbf{c}\mathbf{H}^T = 0\}. \tag{2.75}$$

Each linearly independent constraint cuts the number of valid codewords in half. Thus, if $r = \mathrm{rank}(\mathbf{H}) \leq M$ is the number of linearly independent rows in **H**, then the code rate is $(N - r)/N$.

LDPC Matrix

An LDPC code is described by a parity-check matrix that is sparse [35].

Definition 2.11 (**Regular LDPC Codes** [35])**.** *A regular (N,j,k) LDPC code is a code of block length N defined by a $M \times N$ binary matrix having exactly j ones in each column and exactly k ones in each row. Further, $j < k$ and both are small compared to N.*

By this definition, every parity-check equation of a regular (N,j,k) LDPC code involves k bits, and every bit is involved in j parity check equations. It is instructive to note that N, j and k cannot be chosen independently. The total number of ones in the parity-check matrix **H** is $Mk = Nj$. This implies that Nj/k, which is equal to the number of rows in **H**, must be an integer.

When the restriction of fixed row and column weights in the parity-check matrix is relaxed we get *irregular* LDPC codes. Irregular LDPC codes have been shown to outperform regular LDPC codes for long block lengths [37]. However, in this thesis, we restrict our attention to regular LDPC codes only.

Generator Matrix

The parity-check matrix $\mathbf{H}$ of a rate K/N LDPC code can be expressed as a full rank $(N-K)\times N$ sparse matrix. $\mathbf{H}$ can be written as

$$\mathbf{H} = \begin{bmatrix}\mathbf{H_1} & \mathbf{H_2}\end{bmatrix}, \tag{2.76}$$

where $\mathbf{H_1}$ is a $(N-K)\times K$ matrix and $\mathbf{H_2}$ is a $(N-K)\times(N-K)$ matrix. $\mathbf{H_2}$ is constructed to be invertible. So by row transformation through left multiplication with $\mathbf{H_2}^{-1}$, we obtain a systematic parity-check matrix $\mathbf{H}_{sys}$ that is range equivalent to $\mathbf{H}$. That is,

$$\mathbf{H}_{sys} = \mathbf{H_2}^{-1}\mathbf{H} = \begin{bmatrix}\mathbf{H_2}^{-1}\mathbf{H_1} & \mathbf{I}_{N-K}\end{bmatrix}. \tag{2.77}$$

A systematic generator matrix can be obtained from $\mathbf{H}_{sys}$ as

$$\mathbf{G}_{sys} = \begin{bmatrix}\mathbf{I}_K & (\mathbf{H_2}^{-1}\mathbf{H_1})^T\end{bmatrix}. \tag{2.78}$$

It should be noted that although the original $\mathbf{H}$ matrix is sparse, neither $\mathbf{H}_{sys}$ nor $\mathbf{G}_{sys}$ is sparse in general. $\mathbf{G}_{sys}$ is used for encoding and the original sparse parity matrix $\mathbf{H}$ is used for iterative decoding.

2.5.1 Decoding of LDPC Codes

Tanner Graph

Any parity-check code (including an LDPC code) can be associated with a Tanner graph, which is essentially a visual representation of the parity-check matrix $\mathbf{H}$ [38], [39]. The Tanner graph of a parity-check code consists of N "bit" nodes and M "check" nodes, representing the N bits in the codeword and M parity-check equations respectively. The bit nodes are depicted using circles, while the check nodes are depicted using squares. There is an edge connecting the m-th bit node and n-th check node in the Tanner graph, if there is a 1 in the parity-check matrix $\mathbf{H}$ at the intersection of m-th column and n-th row. Thus, in the case of an (N,j,k) LDPC code, the degrees of bit nodes and check nodes are j and k respectively. As there is no edge connecting two check nodes or two bit nodes, the Tanner graph is a *bipartite* graph.

It is worthwhile to understand here the notion of cycles in an LDPC code. If for any node, there exists a path consisting of consecutive and un-repeated edges leading back to the same node, the LDPC code is said to have cycles. The number of edges in the smallest cycle in a code is termed as the "girth" of the code.

Example 2.3 (Tanner graph). *Consider a short $(8, 2, 4)$ LDPC code whose parity-check matrix $\boldsymbol{H}$ is given in (2.79). The associated Tanner graph is shown in Fig. 2.6. The check nodes are represented by $M = 4$ squares at the top, while the check nodes are represented by $N = 8$ circles at the bottom.*

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}. \tag{2.79}$$

Sum-Product Algorithm

We consider the problem of decoding an LDPC code with parity-check matrix $\mathbf{H}$, given that channel introduces additive memoryless noise. Let $\mathbf{r} = [r_1, r_2, \ldots, r_N]$ be the received noisy vector corresponding to the transmitted binary codeword $\mathbf{c} =$

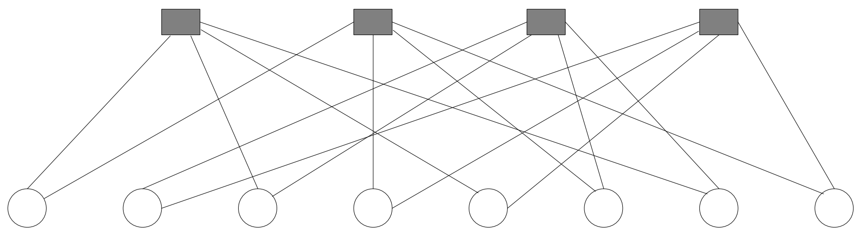


Fig. 2.6: Tanner graph for the LDPC matrix of (2.79).

$[c_1, c_2, \ldots, c_N]$. Thus, we can write $\mathbf{r} := \mathbf{c} + \mathbf{n}$ where $\mathbf{n}$ is the noise. We multiply the received vector with $\mathbf{H}$ to get the syndrome vector

$$\begin{aligned}\mathbf{z} &:= \mathbf{r}\mathbf{H}^T \\ &= \mathbf{c}\mathbf{H}^T + \mathbf{n}\mathbf{H}^T \\ &= \mathbf{n}\mathbf{H}^T.\end{aligned} \tag{2.80}$$

We perform syndrome decoding. The optimal decoder finds the most probable vector $\mathbf{x}$ which explains the observed syndrome vector $\mathbf{z} = \mathbf{x}\mathbf{H}^T$. $\mathbf{x}$ is then our estimate of the noise vector. And the estimated transmitted codeword is $\hat{\mathbf{c}} = \mathbf{r} + \mathbf{x}$.

We use an iterative probabilistic algorithm known variously as sum-product algorithm [39] or belief propagation algorithm [40]. At each step we estimate the posterior probability of the transmitted codeword, given the received vector, and channel properties. The process is best viewed as a message-passing algorithm operating on the Tanner graph associated with $\mathbf{H}$ (Fig. 2.7). We denote the set of bit nodes and check nodes by $\{x_j\}$ and $\{z_i\}$ respectively. The directed edges show causal relationships; the state of check node is determined by the state of the bit nodes to which it is connected. We refer to the neighbours of a bit node x_j as its *children* and to the neighbours of a check node as its *parents*. At each step of the decoding algorithm,

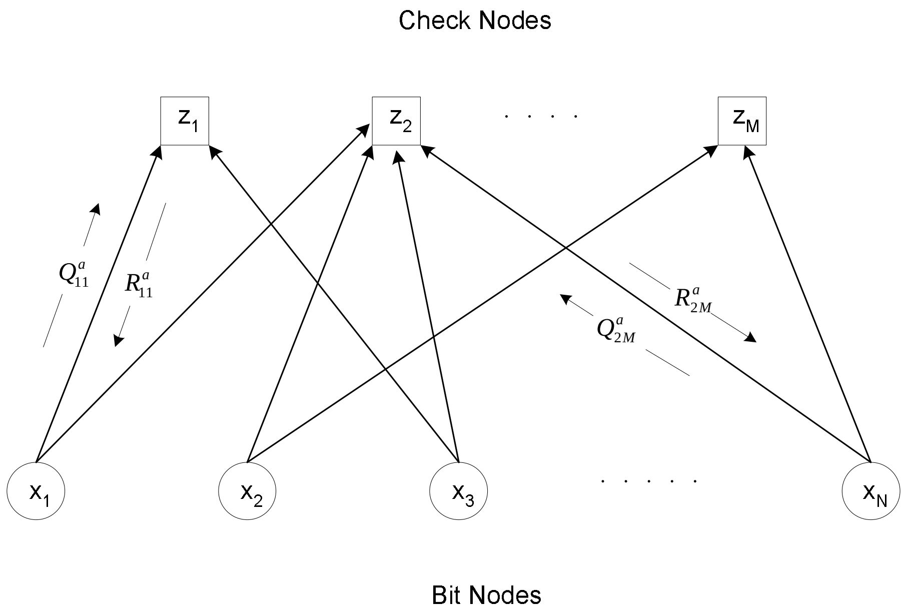


Fig. 2.7: Message passing on the Tanner graph of a LDPC code.

every bit node x_j sends messages Q_{ij}^a to each of its child z_i which are supposed to approximate the node's belief that it is in state $a \in \{0, 1\}$, given messages received from all its other children. Also, each check node z_i sends messages R_{ij}^a to each of its parent x_j approximating the probability of the check node i being satisfied if the parent is assumed to be in state $a \in \{0, 1\}$, taking into account messages received from all its other parents. An iteration of LDPC decoding consists of a round of message passing from each bit node to all adjacent check nodes, followed by another round of message passing from each check node to its adjacent bit nodes. After each iteration we produce a tentative decoding. The algorithm consists of recursively updating these messages until the decoded vector is found to be a codeword or some other stopping criteria is satisfied. We now describe the algorithm in detail.

Initialization

We initialize the algorithm by setting each message Q^a_{ij} to f^a_j, the prior probability that the jth received symbol is a. f^a_j is generated by the channel detector.

Checks to bits

The messages R^a_{ij} that check i sends to parent j should be the probability of check i being satisfied if the parent was in state a. In the sense it is used here, check i is satisfied if it agrees with the corresponding syndrome symbol z_i. More formally,

$$P(z_i|x_j = a) = \sum_{\mathbf{x}:x_j=a} P(z_i|\mathbf{x})P(\mathbf{x}|x_j = a). \tag{2.81}$$

Hence, we sum over all configurations of $\mathbf{x}$ for which the check is satisfied and the parent j is in state a and add up the probability of the configuration. For node z_i, we update the message going to x_j for each value of a as

$$R^a_{ij} = \sum_{\mathbf{x}:x_j=a} P(z_i|\mathbf{x}) \prod_{k\in\mathcal{N}(i)\backslash j} Q^{x_k}_{ik}, \tag{2.82}$$

where $\mathcal{N}(i)$ denote the set of indices of the parents of node z_i and $\mathcal{N}(i)\backslash j$ denoted the indices of all parents except node j. Note that the probability $P(z_i|\mathbf{x})$ of the check node being satisfied is either 0 or 1 for any given configuration.

Bits to Checks

The message that the bit node j sends to check i should be the belief the parent has that it is in state a according to all the other children nodes. Applying Bayes' theorem:

$$P(x_j = a|\{z_i\}_{i\in\mathcal{M}(j)\backslash i}) = \frac{P(x_j = a)P(\{z_i\}_{i\in\mathcal{M}(j)\backslash i}|x_j = a)}{P(\{z_i\}_{i\in\mathcal{M}(j)\backslash i})}. \tag{2.83}$$

Treating the symbols of $\mathbf{z}$ as independent, we take the product of all other children's votes for state a, weighted by the prior. For node x_j, we update the outgoing message to z_i for each value of a as

$$Q_{ij}^a = \alpha_{ij} f_j^a \prod_{k\in\mathcal{M}(ij)\backslash i} R_{kj}^a, \tag{2.84}$$

where $\mathcal{M}(j)$ denotes the set of indices of the children of node x_j and f_j^a is the prior probability that x_j is in state a. The normalizing factor α_{ij} ensures $\sum_a Q_{ij}^a = 1$.

Check stop criterion

At the end of each iteration hard decision is made on each bit's APP as follows:

$$\hat{n_j} = arg \max_a f_j^a \prod_{k\in\mathcal{M}(j)} R_{kj}^a. \tag{2.85}$$

The vector $\hat{n}$ is the tentative estimate of the noise vector. If this satisfies the syndrome equation $\mathbf{z} = \hat{\mathbf{n}}\mathbf{H}^T$, the decoder stops. Otherwise it re-iterates for a prefixed number of times.

It can be shown that for any cycle-free system, the sum-product algorithm converges to the true posterior distribution after a number of iterations [41]. However, there is no such guarantee for the decoding performance when the Tanner graph contains cycles. Therefore, there is no natural termination of the sum-product algorithm in decoding LDPC codes containing cycles, and the decoding only approximates the optimal solution. Owing to the presence of cycles, the successive iterations of the sum-product decoding algorithm tend to get correlated very quickly; this may prevent the iterative sum-product decoding from converging to the optimal solution.

2.5.2 Systematic Construction of LDPC Codes

The methods of constructing LDPC codes can be primarily decomposed into two classes: random constructions and structured constructions. For long block lengths, codes constructed from random matrices give excellent error performance. However for smaller code lengths (not more than several thousand bits) random LDPC codes may have low weight codewords, thus deteriorating the performance. For these lengths, systematic graph-based or algebraic constructions can outperform random ones. Also, the structure in the LDPC matrix can be exploited when implementing the code in hardware. These considerations have motivated the construction of high rate structured LDPC codes for magnetic recording channels.

In this thesis we use a class of structured LDPC codes based on circulant permutation matrices[1]. These codes were initially proposed in [35]. However, the form in which we use them first appeared in [42].

Let $p \geq 5$ be a prime, and let σ be a $p \times p$ matrix obtained from the identity matrix

[1]A permutation matrix is any square matrix with constant row and column weight one; a circulant permutation matrix is a permutation matrix which is cyclic.

by cyclically left shifting by one position, i.e.

$$\sigma = \begin{bmatrix} & & & & 1 \\ 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \end{bmatrix}_{p\times p} . \qquad (2.86)$$

We define a parity-check matrix $\mathbf{H}$ as a $J \times L$ $(L \leq p)$ block matrix of $p \times p$ circulant matrices

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{I} & \mathbf{I} & \mathbf{I} & \cdots & \mathbf{I} \\ \mathbf{I} & \sigma & \sigma^2 & \sigma^3 & \cdots & \sigma^{L-1} \\ \mathbf{I} & \sigma^2 & \sigma^4 & \sigma^6 & \cdots & \sigma^{2(L-1)} \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{I} & \sigma^{J-1} & \sigma^{2(J-1)} & \sigma^{3(J-1)} & \cdots & \sigma^{(J-1)(L-1)} \end{bmatrix} . \qquad (2.87)$$

The code described by matrix $\mathbf{H}$ is a regular (N, J, L) LDPC code, with block length $N = Lp$. The code obtained belongs to the class of self-orthogonal quasi-cyclic codes and therefore, can be encoded in linear time with shift registers [43]. It can be easily observed that the girth g is atleast greater than 4, i.e. $g \geq 6$. It has been recently proved that the girth of such codes can never exceed 12 [44]. Girths of 8, 10 or 12 can be easily achieved by enforcing some additional design constraints [44].

2.6 Summary

In this chapter, we reviewed some of the concepts, theorems and algorithms which are vital for facilitating the understanding of this work. First, we provided a brief description of the magnetic recording channel. That was followed by a detailed analysis of finite-state models and their application in information theory. After that we reviewed LDPC codes.

Chapter 3

Computation of Information Rates

This chapter is dedicated to developing techniques for estimating mutual information rates of noisy channels which also suffer from synchronization errors. We present a general linear filter channel model which is used throughout this thesis. Next, we delineate a quantized Markov process based model for timing errors. The advantage of this model is that it is mathematically tractable yet being fairly accurate. In the following two sections, we analyze two different representations of the the overall channel (inclusive of ISI and timing errors). First we model the channel as an FSM and show that the output process is an ergodic HMP. Then, we give a method to combine the ISI trellis and the timing error trellis to get a trellis representation for the overall channel. In the following section, we present a simulation based approach to upper and lower bound the mutual information rates of such channels. We present the simulation results in the last section. This work was done in collaboration with Harvard University [45].

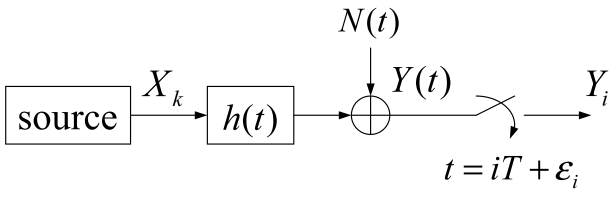


Fig. 3.1: Source and channel model diagram.

3.1 Source and Channel Model

The signal source and ISI channel model considered is shown in Fig. 3.1.

Markov Source

The channel input process $\mathcal{X}$ is a stationary ergodic discrete-time Markov chain of finite order. Its realization x_k at time k ($k \in \mathbb{Z}$) takes on values from the antipodal binary alphabet, $\mathbb{B} = \{-1, +1\}$. If ν is the order of the input Markov chain, then we have

$$P(X_k|X_1^{k-1}) = P(X_k|X_{k-\nu}^{k-1}). \tag{3.1}$$

ISI Channel

We assume that the baseband channel response $h(t)$ is a *finite support function.* We denote the support interval of $h(t)$ by $(-qT, qT)$, where T is the symbol interval. This implies that

$$h(t) = 0 \qquad for\ |t| \geq qT. \tag{3.2}$$

The received waveform $Y(t)$ has the following form

$$Y(t) = \sum_k X_k h(t - kT) + N(t), \tag{3.3}$$

where $N(t)$ is the additive Gaussian noise that is independent of the input. We further assume that the bandwidth of $N(t)$ is limited to $(-\frac{1}{2T}, \frac{1}{2T})$.

3.1.1 Quantized Timing Error Model

When there is perfect synchronization between the transmitter and the receiver, the received waveform is sampled at integral multiples of T. However, because of the timing errors, the receiver samples the waveform at times $\mathcal{E}_0, T+\mathcal{E}_1, 2T+\mathcal{E}_2, 3T+\mathcal{E}_3, \ldots, iT+\mathcal{E}_i$, where $\mathcal{E}_i$ is the timing offset for the i-th sample. Using (4.1) the i-th sample Y_i at the receiver can now be written as

$$\begin{aligned} Y_i = Y(iT+\mathcal{E}_i) &= \sum_{k=-\infty}^{+\infty} X_k \cdot h(iT - kT + \mathcal{E}_i) + N_i \\ &= \sum_{k=i-q+\lceil \frac{\mathcal{E}_i}{T} \rceil}^{i+q+\lfloor \frac{\mathcal{E}_i}{T} \rfloor} X_k \cdot h(iT - kT + \mathcal{E}_i) + N_i. \end{aligned} \tag{3.4}$$

For simplicity, we shall assume that $N_i \sim \mathcal{N}(0, \sigma^2)$ are independent and identically distributed (i.i.d.) Gaussian random variables.

The timing error process $\{\mathcal{E}_i\}$ is independent of the input process $\mathcal{X}$ and noise $\mathcal{N}$. Obviously, an accurate model would consider $\mathcal{E}_i$ to be a real valued random variable. However, without much loss in accuracy, we can assume that $\mathcal{E}_i$ can take one of countably many values $\frac{jT}{Q}$, where j is an arbitrary integer and Q is a fixed positive integer, i.e.

$$\mathcal{E}_i \in \mathbb{T} = \left\{ \cdots, \frac{-2T}{Q}, \frac{-T}{Q}, 0, \frac{T}{Q}, \frac{2T}{Q}, \cdots \right\}. \tag{3.5}$$

Clearly, Q is the number of quantization levels in each symbol interval T. Choosing a large value of Q can ensure that we do not loose much in terms of accuracy due to this segmentation. We further assume that the process $\{\mathcal{E}_i\}$ is *slowly* varying with time, and can be represented by the following random walk process

$$\mathcal{E}_{i+1} = \mathcal{E}_i + \Delta_{i+1}, \tag{3.6}$$

$$P(\Delta_i = \xi_i) = \begin{cases} \delta & \text{if } \xi_i = \frac{T}{Q} \\ \delta & \text{if } \xi_i = -\frac{T}{Q} \\ 1-2\delta & \text{if } \xi_i = 0 \end{cases} \tag{3.7}$$

The timing error increments, Δ_i, are assumed to be i.i.d and be independent of all

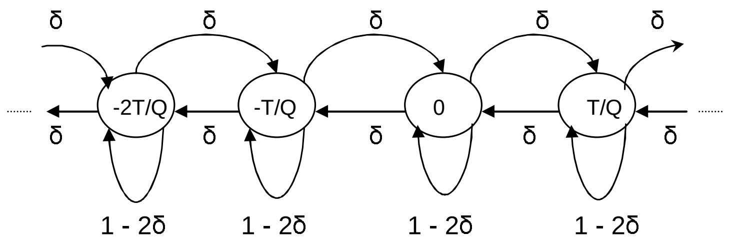


Fig. 3.2: State transition diagram for the timing error Markov chain $\{\mathcal{E}_i\}$

previous samples Y_i and previous timing errors $\mathcal{E}_j$, $j < i$. Thus, we have modelled the timing error process as a first order Markov chain. The choice of a Markov process for modelling timing errors is justified by the fact that any random process can be approximated by a Markov process of sufficiently large memory.

The states of the Markov chain are in the set $\mathbb{T}$ defined in (3.5). Fig. 3.2 depicts the state transition diagram of $\{\mathcal{E}_i\}$. The slowly time-varying assumption is satisfied if $\delta \ll 1$. The initial value of this random process is $\mathcal{E}_0 = 0$. In practical systems, this is generally achieved by using pre-ambles ahead of each block of data symbols. We have chosen this simple model for the ease of exposition. More complicated higher order Markov models can be employed without changing the nature of the problem.

Trellis Representation

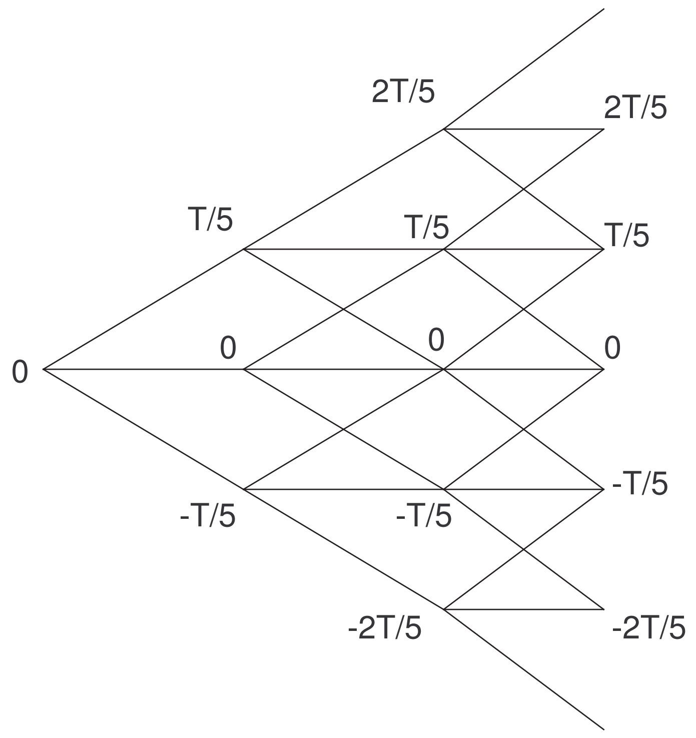


Fig. 3.3: Trellis representation of the timing error process.

The trellis representation of the timing error process $\{\mathcal{E}_i\}$ is depicted in Fig. 3.3. Note that the trellis has been drawn for $Q = 5$ quantization levels and it has been assumed that the initial timing offset is zero. It is evident that the number of states in the trellis keeps increasing as time progresses.

3.2 Finite-State Model for Timing Error Channel

In this section, we present an FSM for the channel described in Sec. 3.1 and show that the output sequence Y_1^L is an HMP. We also prove that Y_1^ℓ, as $\ell \to \infty$, is asymptotically stationary and ergodic.

We assume that the i-th sample is placed in the k-th symbol interval, i.e. $(k-1)T < iT + \mathcal{E}_i \leq kT$ for some $k \in \mathbb{Z}^+$. We also assume that the i-th sampling is done at the M_i-th quantization level of the symbol interval, i.e.

$$iT + \mathcal{E}_i = (k-1)T + \frac{M_i + 1}{Q} T, \tag{3.8}$$

where $M_i \in \{0, 1, \ldots, Q-1\}$. It can be easily observed that the sequence $\{M_i\}$ itself is a first order Markov chain, to be denoted by $\mathcal{M}$. From (3.6) and (3.7) we have

$$P_{M_i|M_{i-1}}(m_i|m_{i-1}) = \begin{cases} \delta & \text{if } m_i = (m_{i-1}+1) \bmod Q \\ \delta & \text{if } m_i = (m_{i-1}-1) \bmod Q, \\ 1-2\delta & \text{if } m_i = m_{i-1}. \end{cases} \tag{3.9}$$

The timing instant $iT + \mathcal{E}_i$ is determined by m_i, which is a realization of M_i as seen in (3.8). As the channel response $h(t)$ is a finite support function, we know that only M_i and $2q$ binary input symbols have effect on the value that Y_i takes (see (3.4)). As the input process is a Markov chain of order ν, the channel state at any instant can be completely defined by M_i and $\kappa = \max(\nu, 2q)$ binary symbols. Without any loss of generality, from now onwards we shall assume that $\nu < 2q$. We denote the channel state by $S \in \mathbb{S} = \{(m, a_1^{2q})\}$, where $m \in \{0, 1, \ldots, Q-1\}$, and a_1^{2q} is a binary vector corresponding to the $2q$ adjacent input symbols that determine a particular sample value at the receiver. It is worthwhile to note that we need not be aware of the indices of the input symbols which correspond to a specific output sample. It is sufficient to know the value of the binary vector a_1^{2q}.

It can be easily shown that the channel state sequence $S_i = (m, a_1^{2q}(i))$, $i \in$

$\{0, 1, \ldots\}$, also forms a first order Markov chain with a finite number of states, i.e.

$$P(S_i|S_{i-1}, S_{i-2}, S_{i-3}, \ldots) = P(S_i|S_{i-1}). \tag{3.10}$$

The total number of channel states is $2^{2q}Q$. Given a channel state $S_i = (m_i, a_1^{2q}(i))$, we know from (3.4) that the i-th sample

$$y_i(S_i) = \sum_{k=1}^{2q} a_k(i)h\big((q - k + \frac{m_i + 1}{Q})T\big) + N_i \tag{3.11}$$

is a Gaussian random variable, i.e. $Y_i \sim \mathcal{N}(\eta_i, \sigma^2)$, where

$$\mu_i = \sum_{k=1}^{2q} a_k(i)h\big((q - k + \frac{m_i + 1}{Q})T\big). \tag{3.12}$$

From the arguments above, we can conclude that $\mathcal{Y}$ is an HMP with the channel state process $\mathcal{S}$ being the embedded Markov chain. Therefore, in order to prove that $\mathcal{Y}$ is asymptotically stationary and ergodic, we need to examine the properties of the hidden state process $\mathcal{S}$. We have the following lemma.

Lemma 1. *The state process $\mathcal{S}$ is an ergodic finite-state Markov chain that has a unique stationary-state distribution to which it converges from any initial state distribution.*

Proof: From Theorem 2.1 we know that to prove that a Markov chain is ergodic, it is sufficient to show that it is irreducible and aperiodic.

To prove irreducibility, we first show that any two states, $s \in \mathbb{S}$ and $s' \in \mathbb{S}$ *communicate.* Let $s = (m, a_1^{2q})$ and $s' = (m', b_1^{2q})$, where $0 \leq m \leq m' \leq Q - 1$, and a_i,b_i are binary symbols. Using (3.9), we can observe that starting from state s, it is possible to reach a state $s'' = (m', c_1^{2q})$ in $m' - m$ transitions for some binary vector c_1^{2q}. Similarly, we can start from state s'' and go to state s' in at most $2q$ transitions by keeping m' unchanged and sequentially sending the binary symbols $b_1, b_2, \ldots, b_{2q}$. Thus, state s' is accessible from state s. On similar lines, we can show that state s is also reachable from state s'. Therefore, we can claim that all the states in $\mathbb{S}$ communicate, i.e. all states belong to the same class. Since $\mathbb{S}$ is a finite set and all the states in $\mathbb{S}$ are in the same class, we can conclude that the Markov chain $\mathcal{S}$ is irreducible.

Next, we show that each state in $\mathbb{S}$ is aperiodic. Let us consider a state $s = (Q-1, a_1^{2q})$, where all the binary symbols $a_i = 1$. Obviously, we can reach state s from state s itself using a single state transition, which implies that state s is an aperiodic state. Since, all states in $\mathbb{S}$ belong to the same class, all states are aperiodic. In other words, the Markov chain $\mathcal{S}$ is aperiodic.

Since the Markov chain $\mathcal{S}$ is both irreducible and aperiodic, we can conclude that it is ergodic. Furthermore, it will converge to a *unique* stationary state distribution, irrespective of the initial state, i.e. the state sequence is *asymptotically stationary.* (see Theorem 2.1)

We know that the statistical properties of an HMP are inherited from similar properties of the underlying state process. We therefore conclude that the sampled sequence $\{Y_i\}$ is ergodic and is asymptotically stationary. In other words, the output process $\mathcal{Y}$ is an ergodic and asymptotically stationary process.

Using similar arguments as for Lemma 1, we can also prove the following:

Lemma 2. *The process $\mathcal{M}$, as defined in* (3.9)*, is an ergodic and asymptotically* *stationary finite-state Markov chain. Furthermore, it has a unique steady-state distribution* $\pi_m(k) = \frac{1}{Q}, k \in \{0, 1, \ldots, Q-1\}$.

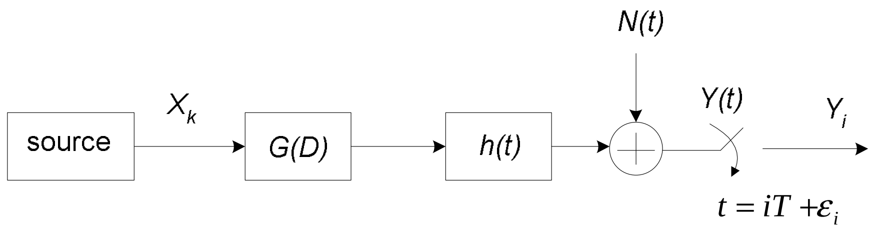


Fig. 3.4: The block diagram for the simulation setup used. $G(D) = 1 - D^2$.

3.3 Joint ISI-Timing Error Trellis

In this section, we give an alternative description of the timing error channel using a joint ISI-timing error trellis. As the name suggests, the joint trellis includes the effects of both ISI as well as timing errors. Although the method to construct the joint trellis that we present is general, it is best explained through an example. Therefore, we first describe our simulation setup.

3.3.1 Simulation Setup

Fig. 3.4 depicts the setup we use for our simulations. The source emits antipodal binary symbols i.e. $X_k \in \mathbb{B}$. These source symbols are first passed through filter $G(D) = 1 - D^2$ and later through the baseband channel. We model the baseband channel response function $h(t)$ as a truncated *sinc* function with the form

$$h(t) = sinc(t)[u(t+T) - u(t-T)], \tag{3.13}$$

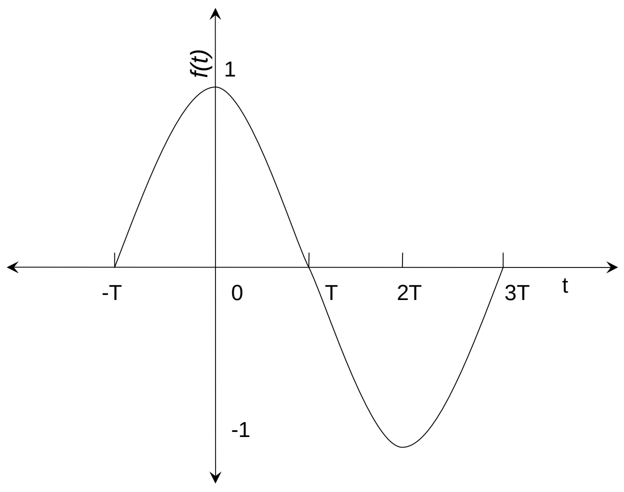


Fig. 3.5: Overall channel response.

where $u(t)$ is the unit step function. For the symbol source, the *overall channel response* $f(t)$ is the combination of the filter $G(D)$ and the channel response function $h(t)$. It is easy to observe that

$$f(t) = h(t) - h(t - 2T). \tag{3.14}$$

The overall channel response is shown in Fig. 3.5.

3.3.2 ISI Trellis

When there is perfect synchronization between the transmitter and the receiver, the channel described in the previous subsection is essentially equivalent to the partial-response class-4 polynomial (PR4) channel, and has a memory length of 2. However, with imperfect timing the channel memory length increases to 3.

3.3.3 Construction of the Joint ISI-Timing Error Trellis

The joint ISI-timing error trellis is formed by the cross-product of the ISI trellis, and the timing error trellis depicted in Fig. 3.3. Any state in the joint trellis has the form $S = (x, x', x'', \psi)$, where $x, x', x'' \in \mathbb{B}^3$ and $\psi \in \mathbb{T}$. It is worthwhile to note some salient points about the joint trellis here:

- Like the timing error trellis, the joint trellis has *countably infinite* states, and it grows without bound as time progresses.

- We model the quantized timing errors as a first order Markov chain, where only three types of transitions are allowed from any state. The state may remain unchanged or there could be a transition to the immediate neighbours. Due to this, if the i-th sample falls at the M_i-th ($M_i \in \{0, 1, \ldots, Q-1\}$) quantization level, the $(i+1)$-th sample is constricted to fall either on the M_i-th quantization level or the levels adjacent to it as described in (3.9). A natural corollary of this restriction is that there can be at most two samples in a symbol interval. Two samples may fall in the same symbol interval at the 0-th and $(Q-1)$-th quantization respectively; this is equivalent to an *insertion*. Or, there may be no samples in a particular symbol interval, which signifies *deletion* of a symbol. Otherwise, there is one sample in a symbol interval at any of the Q quantization levels.

  Now, each section in the joint trellis corresponds to one noiseless channel output. Since our channel model permits variable number of samplings per symbol interval, the ISI state transitions in the joint trellis are dependent on the timing offset transitions.

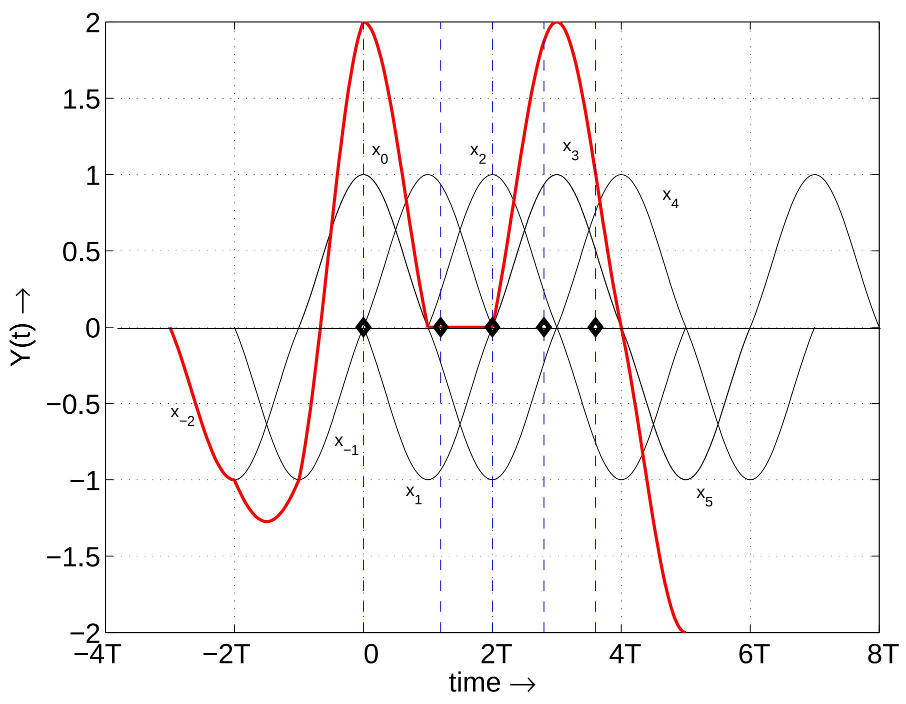


Fig. 3.6: A realization of the sampling process at the receiver. The noiseless received waveform is drawn using thick red line. The sampling instants are marked on the time axis using diamonds.

To understand the construction of the joint trellis, let us consider a realization of the sampling process at the receiver shown in Fig. 3.6. We assume that initially the receiver is perfectly synchronized to the transmitter; thus, the timing offset is zero at $t = 0$. We further assume that the two bits preceding the message block are -1, i.e. $x_{-2} = -1$ and $x_{-1} = -1$. The number of quantization levels is $Q = 5$. The input bits $\{x_0, x_1, x_2, x_3, x_4, x_5\}$ are arbitrarily chosen to be $\{+1, -1, +1, +1, +1, -1\}$, respectively. The sampling instants are marked on the abscissa using diamonds. As can be seen from Fig. 3.6, the timing offsets for the first five samples are $0, \frac{T}{5}, 0, \frac{-T}{5}, \frac{-2T}{5}$, respectively.

We shall now observe how the ISI state transitions occur in the joint trellis for a given realization of the timing error process $\mathcal{E}_k$. The trellis state corresponding to the first sample at $t = 0$ is $(x_{-2}, x_{-1}, x_0, 0)$. The next sampling takes place at $t = 1.2T$. Note that no sample has been taken in the interval $[0, T)$ and thus, a deletion has taken place. To obtain the value of the noiseless sample, we need to know x_{-1}, x_0, x_1 and x_2. Hence, the next ISI state must be given by (x_0, x_1, x_2). The next sample is taken at $t = 2T$. Clearly, this sample is in the same symbol interval as previous one. This is an example of symbol insertion. As we already know all the input bits required to compute the noiseless value, the destination ISI state remains (x_0, x_1, x_2). Only one sample falls in all other symbol intervals, and it can be easily observed that for all such cases the ISI states transition is of the form $(x_{n-2}, x_{n-1}, x_n) \rightarrow (x_{n-1}, x_n, x_{n+1})$, where $n \in \mathbb{Z}$.

It is also important to note that any branch in the joint ISI-timing error trellis "carries" one noiseless channel output and 0,1 or 2 input bits. Absence of input bits on a branch corresponds to symbol deletion, and 2 input bits imply insertion of a symbol. We can generalize the observations made in the previous paragraph

Table 3.1: Rules for finding ISI state transitions give the timing offset state transitions.

$i^{th} \rightarrow (i+1)^{th}$ sample			
Case	Timing transition	ISI transition	Bits on the branch
Insertion	$kT \rightarrow \left((k+1)T + \frac{T}{Q}\right)$	$(x_{n-2}x_{n-1}x_n) \rightarrow (x_n x_{n+1} x_{n+2})$	x_{n+1}, x_{n+2}
Deletion	$(kT + Q) \rightarrow (k+1)T$	$(x_{n-2}x_{n-1}x_n) \rightarrow (x_{n-2}x_{n-1}x_n)$	ϕ
Other	all other transitions	$(x_{n-2}x_{n-1}x_n) \rightarrow (x_{n-1}x_n x_{n+1})$	x_{n+1}

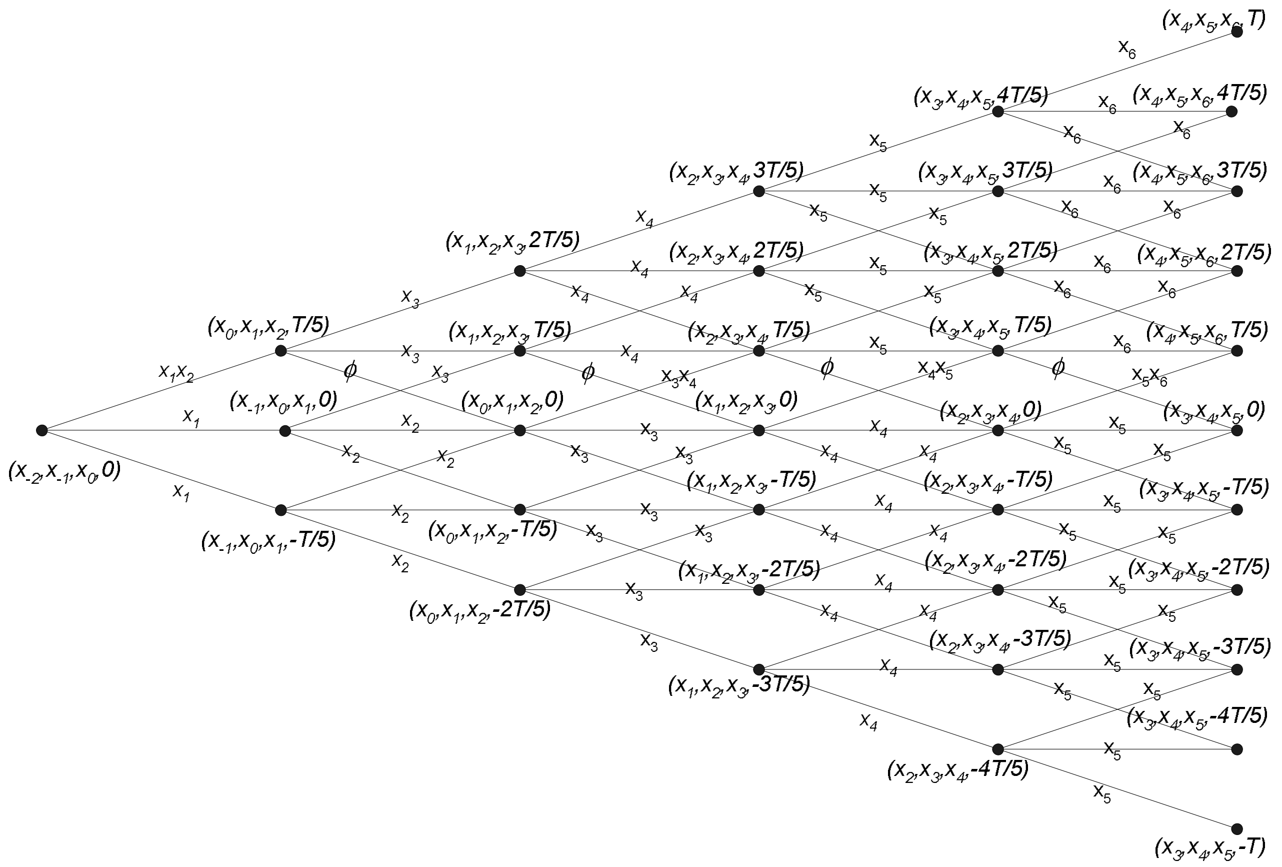


Fig. 3.7: Joint ISI-timing error trellis

into rules for assigning ISI states to the nodes in the joint trellis. These rules are presented in Table 3.1. We represent the number of quantization levels by Q. We assume that the i-th sample falls at $iT + \mathcal{E}_i$, where $\mathcal{E}_i \in \mathbb{T}$ is the timing offset. From (3.8) we know that we can express $iT + \mathcal{E}_i$ in the form $kT + \frac{q}{Q}T$, where $k \in \mathbb{Z}$ and $q \in \{0, 1, \ldots, Q-1\}$. Note that it is not necessary to know the exact relation between i and k here. We further assume that the ISI state at the i-th section in the joint trellis is given by (x_{n-2}, x_{n-1}, x_n). We denote the absence of bits over any branch in the joint trellis by ϕ. Using these rules we construct the joint ISI-timing error trellis presented in a compact form in Fig. 3.7. Observe that all paths merging into any node in the joint trellis carry the same number of input bits.

From the discussion above, we conclude that at time t, any state in the joint trellis is given by $S_t = \{x_{k-2}^k, \psi\}$, where $x_{k-2}^k \in \mathbb{B}^3$, $\psi \in \mathbb{T}$, and relation between t and k is contingent upon the realization of timing error process. It can be easily observed that such a state evolution process precludes the possibility of communication between any pair of states in the joint trellis. Due to this, the FSC described by the joint trellis cannot be proved to be ergodic.

3.4 Information Rate Computation

Channels with timing errors can cause random insertions or deletions of symbols. Due to this, for such channels the number of transmitted symbols (m) may be not be equal to the number of received symbols (ℓ). The mutual information rate *per received symbol* for synchronization channels can be expressed as [4]

$$I^{(r)}(\mathcal{X};\mathcal{Y}) = \lim_{\ell\to\infty} \frac{1}{\ell} \lim_{m\to\infty} I(X_1^m; Y_1^\ell) \text{ [bits/received symbol].} \tag{3.15}$$

For short notation, we shall use

$$I(X_1^\infty; Y_1^\ell) = \lim_{m\to\infty} I(X_1^m; Y_1^\ell).$$

We can now expand (3.15) as

$$\begin{aligned} I^{(r)}(\mathcal{X};\mathcal{Y}) &= \lim_{\ell\to\infty} \frac{1}{\ell} I(X_1^\infty; Y_1^\ell) && (3.16)\\ &= \lim_{\ell\to\infty} \frac{1}{\ell} h(Y_1^\ell) - \lim_{\ell\to\infty} \frac{1}{\ell} h(Y_1^\ell; X_1^\infty) && (3.17)\\ &= h(\mathcal{Y}) - h(\mathcal{Y}|\mathcal{X}) \text{ [bits/received symbol]}. && (3.18)\end{aligned}$$

However, the quantity of our interest is $I(\mathcal{X};\mathcal{Y})$, the mutual information rate *per transmitted symbol.* It can be written as

$$\begin{aligned} I(\mathcal{X};\mathcal{Y}) &= \alpha \cdot I^{(r)}(\mathcal{X};\mathcal{Y}) && (3.19)\\ &= \alpha \cdot \lim_{\ell\to\infty} \frac{1}{\ell} I(X_1^\infty; Y_1^\ell), && (3.20)\end{aligned}$$

where α is the expected number of received symbols *per transmitted symbol.* If we denote the number of samples in the first m symbol intervals by L_m, then

$$\alpha = \lim_{m\to\infty} \frac{L_m}{m}. \qquad (3.21)$$

And the channel capacity can be expressed as

$$C = \alpha \cdot \sup_{p(X_1^\infty)} \lim_{\ell\to\infty} \frac{1}{\ell} I(X_1^\infty; Y_1^\ell), \qquad (3.22)$$

where the supremum is taken over all possible stationary and ergodic processes X_1^∞ [4].

We shall now present Monte-Carlo methods to compute upper and lower bounds to $I(\mathcal{X};\mathcal{Y})$, the information rate *per transmitted symbol* for a given stationary, and ergodic input process X_1^∞.

3.4.1 Computation of α

Since L_m is defined as the number of samples in the first m symbol intervals, we have

$$L_m T + \mathcal{E}_{L_m} \le mT \tag{3.23}$$

$$(L_m + 1)T + \mathcal{E}_{L_m+1} > mT. \tag{3.24}$$

From (3.23) and (3.24), we have

$$\frac{m-1}{m} - \frac{\sum_{i=1}^{L_m+1} \Delta_i}{mT} < \frac{L_m}{m} \le 1 - \frac{\sum_{i=1}^{L_m} \Delta_i}{mT}. \tag{3.25}$$

By taking the expected values of all the terms in the inequality (3.25), and letting $m \to \infty$, we have

$$1 \le \lim_{m\to\infty} \frac{E[L_m]}{m} \le 1. \tag{3.26}$$

Therefore, from (3.21) we have $\alpha = 1$.

3.4.2 Computation of $h(\mathcal{Y})$

in Sec. 3.2, we presented a finite-state model for the timing error channel in consideration. We also proved that the output process $\mathcal{Y}$ of the channel, or equivalently the FSM is asymptotically stationary and ergodic. Hence, it follows from the Shannon-

McMillan-Breiman theorem [46] that the sample sequence entropy

$$h(y_1^\ell) \triangleq -\log_2 p(y_1^\ell)$$

upon a scaling factor converges to the true entropy $h(Y)$ with probability one (almost surely) for large ℓ, i.e.

$$\lim_{\ell\to\infty} -\frac{1}{\ell}\log_2 p(y_1^\ell) = h(\mathcal{Y}). \tag{3.27}$$

We now use the Monte-Carlo method described in Sec. 2.4.3 to estimate $h(\mathcal{Y})$. A very long channel output sequence y_1^ℓ is sampled and $p(y_1^\ell)$ is computed using the forward recursion of the BCJR algorithm applied over the channel FSM. For a given realization y_1^ℓ of the HMP $\mathcal{Y}$, we have

$$p(y_1^\ell) = \sum_{s_1^\ell} p(s_1^\ell, y_1^\ell), \tag{3.28}$$

where, $s_1^\ell \in \mathbb{S}^\ell$. We define the *accumulated state-metric* at the k-th sample as $\sigma_k(s_k)$. Using the Markovian property in (3.10) and (3.11), we can write

$$\begin{aligned}\sigma_k(s_k) &= \sum_{s_1^{k-1}} p(y_1^k, s_1^k)\\ &= \sum_{s_{k-1}} \sigma_k(s_{k-1}) p(y_k, s_k|s_{k-1}).\end{aligned} \tag{3.29}$$

$\sigma_k(s_k)$ can now be recursively computed using the BCJR algorithm. Using (3.29) in (3.28) we get

$$p(y_1^\ell) = \sum_{s_\ell\in\mathbb{S}} \sigma_\ell(s_\ell). \tag{3.30}$$

For large k, the accumulated state-metrics $\sigma(s_k)$ calculated according to (3.29) quickly tend to zero and numerical underflow occurs. To circumvent this problem, in practice the recursive relation of (3.29) is changed to

$$\tilde{\sigma}_k(s_k) = \sum_{s_{k-1}\in\mathbb{S}} \sigma(s_{k-1}) p(y_k, s_k | s_{k-1}), \tag{3.31}$$

$$\sigma_k(s_k) = \frac{1}{\mu_k} \tilde{\sigma}_k(s_k) \tag{3.32}$$

where μ_k is the normalizing factor, i.e.

$$\mu_k = \sum_{s_k\in\mathbb{S}} \sigma_k(s_k).$$

Due to the normalization, the sum of the accumulated state-metrics at any time k equals one. Therefore, $p(y_1^\ell)$ is given by the product of all the normalizing factors upto time ℓ, i.e.

$$\begin{aligned} p(y_1^\ell) &= \prod_{k=1}^{\ell} \mu_k \underbrace{\sum_{s_\ell\in\mathbb{S}} \sigma_\ell(s_\ell)}_{=1} \\ &= \prod_{k=1}^{\ell} \mu_k. \end{aligned} \tag{3.33}$$

The estimated entropy rate of the sample sequence is then obtained as

$$\hat{h}(\mathcal{Y}) = -\frac{1}{\ell} \log_2 p(y_1^\ell) = -\frac{1}{\ell} \sum_{k=1}^{\ell} \log_2 \mu_k. \tag{3.34}$$

The entropy rate of the sample sequence is thus the average of the logarithms of the normalizing factors, which *almost surely* converges to $h(\mathcal{Y})$.

3.4.3 Upper Bounding $h(\mathcal{Y}|\mathcal{X})$

We know that

$$h(\mathcal{Y}|\mathcal{X}) = \lim_{\ell\to\infty} \frac{1}{\ell} \lim_{m\to\infty} h(Y_1^\ell|X_1^m) \tag{3.35}$$

$$= \lim_{\ell\to\infty} \frac{1}{\ell} \lim_{m\to\infty} E_{p(X^m)}[h(Y_1^\ell|X_1^m = x_1^m)] \tag{3.36}$$

$$= \lim_{\ell\to\infty} \frac{1}{\ell} \lim_{m\to\infty} E_{p(X^m)}\big[E_{p(Y^\ell|X^m=x_1^m)} \log_2 p(Y_1^\ell|X_1^m = x_1^m)\big], \tag{3.37}$$

where ℓ is the number of symbols received, when m symbols are transmitted. The input sequence x_1^m is passed through the ISI channel, which also adds Guassian noise, to generate the receiver waveform. This waveform is subsequently sampled with timing offsets generated by Markov chain $\{\mathcal{E}_k\}$ to obtain y_1^ℓ. Given the channel input x_1^m and corresponding channel output y_1^ℓ, we can compute the quantity $p(y_1^\ell|x_1^m)$ by the forward recursion of the BCJR algorithm running over the joint ISI-timing error trellis described in Sec. 3.3. As we have to compute $p(y_1^\ell|x_1^m)$, we assume that the input sequence is known to the receiver. However, the uncertainties in the sampling timings are still present. Due to this, the BCJR algorithm now operates on a *reduced trellis* induced by the sequence x_1^m. The reduced trellis is equal to the sub-trellis associated with a given data-path in the joint ISI-timing error trellis. At the k-th sampling instant, the set of states in the reduced trellis is $\bar{\mathbb{S}}'_k \subseteq \mathbb{S}'$, where $\mathbb{S}'$ is the set of states in the joint trellis.

Closely following the method delineated in the previous subsection, we write the accumulated state-metric $\sigma_k(s_k)$ at the k-th trellis section as

$$\sigma_k(s_k) = \sum_{s_1^{k-1}} p(y_1^k, s_1^k | x_1^m) \tag{3.38}$$

$$= \sum_{s_{k-1}} \Big(\sum_{s_1^{k-2}} p(y_1^{k-1}, s_1^{k-1} | x_1^m) \Big) \cdot p(y_k, s_k | x_1^m, s_{k-1}) \tag{3.39}$$

$$= \sum_{s_{k-1}} \sigma_{k-1}(s_{k-1}) \cdot p(y_k, s_k | x_1^t, s_{k-1}). \tag{3.40}$$

Note that t in (3.40) is not equal to k in general. As before, we have

$$p(y_1^\ell | x_1^m) = \sum_{s_\ell \in \bar{\mathbb{S}}'_\ell} \sigma_\ell(s_\ell), \tag{3.41}$$

where $\sigma(s_k)$ is computed recursively using the BCJR algorithm.

Now, we need to make two modifications in the method described above to make it practically implementable. As we know that the joint trellis grows without bound as time progresses, it may not be possible to track all the states. Therefore, we resort to the *reduced-state trellis techniques* presented in [47], [10]. When running the forward recursion of the BCJR algorithm, we keep only a subset $\bar{\bar{\mathbb{S}}}'_k$ of the k-th section states $\bar{\mathbb{S}}'_k$ and discard the rest. This leads to a lower bound (LB) on $p(y_1^\ell | x_1^m)$. Various strategies can be used in selecting the subset $\bar{\bar{\mathbb{S}}}'_k$. Our method is to retain a fixed number of states with the largest accumulated metric and ignoring the rest at each trellis section. Besides this, we also normalize the accumulated state-metrics at each trellis section to avoid numerical underflow. Recursion (3.40) is now modified to

$$\tilde{\sigma}_k(s_k) = \sum_{s_{k-1} \in \bar{\bar{\mathbb{S}}}'} \sigma_{k-1}(s_{k-1}) p(y_k, s_k | s_{k-1}, x_1^t), \tag{3.42}$$

$$\sigma_k(s_k) = \frac{1}{\mu_k}\tilde{\sigma}_k(s_k). \tag{3.43}$$

The normalizing factor μ_k is given by

$$\mu_k = \sum_{s_k \in \bar{\bar{\mathbb{S}}}'_k} \sigma_k(s_k).$$

The lower bound $p(y_1^\ell | x_1^m)_{LB}$ is then given by

$$p(y_1^\ell | x_1^m) \geq p(y_1^\ell | x_1^m)_{LB} = \prod_{k=1}^{\ell} \mu_k. \tag{3.44}$$

A lower bound on $p(y_1^\ell | x_1^m)$ yields an upper bound (UB) on the sample sequence entropy, i.e.

$$\begin{aligned} h(y_1^\ell | x_1^m) &\leq h(y_1^\ell | x_1^m)_{UB} && (3.45) \\ &= -\log_2 p(y_1^\ell | x_1^m)_{LB} && (3.46) \\ &= -\sum_{k=1}^{\ell} \log_2 \mu_k. && (3.47) \end{aligned}$$

As the joint trellis is not ergodic, we can no longer invoke the Shannon-McMillan-Breiman theorem. Instead, we take advantage of the law of large numbers to evaluate $h(\mathcal{Y}|\mathcal{X})$. We first estimate

$$h(Y_1^\ell | X_1^m = x_1^m) = -E[\log_2 p(Y_1^\ell | X_1^m = x_1^m)]$$

using the following method. For a given input sequence x_1^m, generate N different output sequences. For each such sequence, compute $p(y_1^\ell | x_1^m)_{LB}$ as described above. Let $\hat{h}_n$ be equal to $-\log_2 p(y_1^\ell | x_1^m)_{LB}$ of the n-th output sequence y_1^ℓ corresponding to the input x_1^m. Then,

$$\frac{1}{N}\sum_{n=1}^{N}\hat{h}_n = h(Y_1^\ell | X_1^m = x_1^m)_{UB},$$

which is an upper bound to $h(Y_1^\ell | X_1^m = x_1^m)$. The convergence is assured by the law of large numbers for large N. We know

$$h(Y_1^\ell | X_1^m) = \sum_{x_1^m \in \mathbb{B}^m} p(x_1^m) h(Y_1^\ell | X_1^m = x_1^m). \tag{3.48}$$

As the symbol source is ergodic by assumption, we know from AEP that for large m all input sequences are equiprobable. Therefore, a single long input sequence is sufficient to evaluate $h(Y_1^\ell | X_1^m)$. Thus, for large m and N, the method presented above can be used to upper bound $h(\mathcal{Y}|\mathcal{X})$, which leads to a lower bound on the information rate:

$$\begin{aligned} I(\mathcal{X};\mathcal{Y}) &= h(\mathcal{Y}) - h(\mathcal{Y}|\mathcal{X}) \\ &\geq h(\mathcal{Y}) - h(\mathcal{Y}|\mathcal{X})_{UB}. \end{aligned} \tag{3.49}$$

3.4.4 Lower Bounding $h(\mathcal{Y}|\mathcal{X})$

We notice that given the input sequence, the uncertainty in the sample sequence Y_1^ℓ comes not only from the additive Gaussian noise, but also from the timing error sequence $\{\mathcal{E}_i\}$. Therefore, we write $h(Y_1^\ell|X_1^\infty)$ as follows:

$$\begin{aligned} h(Y_1^\ell|X_1^\infty) &= h(Y_1^\ell|X_1^\infty, \mathcal{E}_1^\ell) + I(Y_1^\ell, \mathcal{E}_1^\ell|X_1^\infty) \\ &= h(Y_1^\ell|X_1^\infty, \mathcal{E}_1^\ell) + H(\mathcal{E}_1^\ell|X_1^\infty) - H(\mathcal{E}_1^\ell|X_1^\infty, Y_1^\ell). \end{aligned} \tag{3.50}$$

Now we analyze the three terms in (3.50) separately. Since the additive noise N_i are i.i.d. Gaussian random variables from our previous assumption, we have

$$h(Y_1^\ell|X_1^\infty, \mathcal{E}_1^\ell) = \frac{\ell}{2} \log 2\pi e \sigma^2, \tag{3.51}$$

where σ^2 is the variance of the noise. From the random walk assumption given by (3.6) and (3.7), and the fact that the timing error process is independent of the inputs, we get

$$\begin{aligned} H(\mathcal{E}_1^\ell|X_1^\infty) &= H(\mathcal{E}_1^\ell) \\ &= H(\mathcal{E}_1) + \sum_{k=2}^{\ell} H(\mathcal{E}_k|\mathcal{E}_1^{k-1}) \\ &= H(\mathcal{E}_1) + \sum_{k=2}^{\ell} H(\mathcal{E}_k|\mathcal{E}_{k-1}). \end{aligned} \tag{3.52}$$

Since, $\mathcal{E}_k$ takes only three values given $\mathcal{E}_{k-1}$, we have

$$\begin{aligned}
&H(\mathcal{E}_k|\mathcal{E}_{k-1}) \\
&= \sum_{\varepsilon} P(\mathcal{E}_{k-1}=\varepsilon)H(\mathcal{E}_k|\mathcal{E}_{k-1}=\varepsilon) \\
&= \sum_{\varepsilon} P(\mathcal{E}_{k-1}=\varepsilon)\left[2p\log\frac{1}{p}+(1-2p)\log\frac{1}{1-2p}\right] \\
&= 2p\log\frac{1}{p}+(1-2p)\log\frac{1}{1-2p}. \qquad (3.53)
\end{aligned}$$

If we also assume that the system starts from perfect timing, i.e $\mathcal{E}_0 = 0$ then (3.52) simplifies to

$$H(\mathcal{E}_1^\ell|X_1^\ell) = \ell \cdot \left[2p\log\frac{1}{p}+(1-2p)\log\frac{1}{1-2p}\right]. \qquad (3.54)$$

We do not have a method to compute the term $H(\mathcal{E}_1^\ell|X_1^\infty, Y_1^\ell)$ in (3.50), but we can recursively bound it as follows

$$\begin{aligned}
H(\mathcal{E}_1^\ell|X_1^\infty, Y_1^\ell) &= H(\mathcal{E}_1^{\ell-1}|X_1^\infty, Y_1^\ell) + H(\mathcal{E}_\ell|X_1^\infty, Y_1^\ell, \mathcal{E}_1^{\ell-1}) \\
&\leq H(\mathcal{E}_1^{\ell-1}|X_1^\infty, Y_1^{\ell-1}) + H(\mathcal{E}_\ell|X_1^\infty, Y_1^\ell, \mathcal{E}_1^{\ell-1}). \qquad (3.55)
\end{aligned}$$

The above inequality uses the fact that conditioning reduces entropy. Note that the recursion in (3.55) gives an upper bound on $H(\mathcal{E}_1^\ell|X_1^\infty, Y_1^\ell)$, and thus leads to a lower bound to the conditional entropy rate $h(\mathcal{Y}|\mathcal{X})$.

Next, we use the random-walk assumption in (3.6) and (3.7) as well as the Bayes rule to prove the following relation

$$H(\mathcal{E}_\ell|X_1^\infty, Y_1^\ell, \mathcal{E}_1^{\ell-1}) = H(\mathcal{E}_\ell|X_1^\infty, Y_\ell, \mathcal{E}_{\ell-1}). \qquad (3.56)$$

From the Bayes rule we have

$$\begin{aligned}
&P(\mathcal{E}_n|X_1^\infty, Y_1^n, \mathcal{E}_1^{n-1}) \\
&= \frac{P(X_1^\infty, Y_1^n, \mathcal{E}_1^n)}{P(X_1^\infty, Y_1^n, \mathcal{E}_1^{n-1})} \\
&= \frac{P(X_1^\infty, Y_n, \mathcal{E}_{n-1}^n) \cdot P(Y_1^{n-1}, \mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}^n)}{P(X_1^\infty, Y_n, \mathcal{E}_{n-1}) \cdot P(Y_1^{n-1}, \mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1})} \\
&= P(\mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_{n-1}) \cdot \frac{P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}^n)}{P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1})} \cdot \frac{P(Y_1^{n-1}|X_1^\infty, Y_n, \mathcal{E}_1^n)}{P(Y_1^{n-1}|X_1^\infty, Y_n, \mathcal{E}_1^{n-1})} \\
&= P(\mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_{n-1}) \cdot \frac{P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}^n)}{P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1})}. \qquad (3.57)
\end{aligned}$$

By using the Bayes rule again,

$$\begin{aligned}
P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}^n) &= \frac{P(\mathcal{E}_1^{n-2}, \mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_{n-1})}{P(\mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_{n-1})} \\
&= P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}) \cdot \frac{P(\mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_1^{n-1})}{P(\mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_{n-1})} \\
&= P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}). \qquad (3.58)
\end{aligned}$$

The last equality is based on the fact that $\mathcal{E}_k$ is a first order Markov chain as given by (3.6) and (3.7). Substituting (3.58) into (3.57), we obtain

$$P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}^n) = P(\mathcal{E}_1^{n-2}|X_1^\infty, Y_n, \mathcal{E}_{n-1}) \qquad (3.59)$$

Hence, we can conclude that $H(\mathcal{E}_n|X_1^\infty, Y_1^n, \mathcal{E}_1^{n-1}) = H(\mathcal{E}_n|X_1^\infty, Y_n, \mathcal{E}_{n-1})$.

Using (3.56), we can now write (3.55) as

$$H(\mathcal{E}_1^\ell|X_1^\infty, Y_1^\ell) \leq H(\mathcal{E}_1^{\ell-1}|X_1^\infty, Y_1^{\ell-1}) + H(\mathcal{E}_l|X_1^\infty, Y_\ell, \mathcal{E}_{\ell-1}). \qquad (3.60)$$

The above inequality implies that $H(\mathcal{E}_1^\ell|X_1^\infty, Y_1^\ell)$ can be recursively bounded if we can compute $H(\mathcal{E}_\ell|X_1^\infty, Y_\ell, \mathcal{E}_{\ell-1})$. We next prove the following lemma.

Lemma 3. *Given a stationary input sequence $\{X_k\}$, and the timing error process $\{\mathcal{E}_k\}$ as defined by (3.6) and (3.7), with channel outputs specified by (3.4), the conditional entropy $H(\mathcal{E}_i|X_1^\infty, Y_i, \mathcal{E}_{i-1})$, as $i \to \infty$, does not depend on the index i. Furthermore,*

$$H(\mathcal{E}_i|X_1^\infty, Y_i, \mathcal{E}_{i-1}) = H(M|X_1^{2q}, Y, M'), \tag{3.61}$$

where

$$\begin{aligned} P_{M'}(j) &= \frac{1}{Q}, \qquad j \in \{0, 1, \ldots, Q-1\} \\ P_{M|M'}(m|m') &= \begin{cases} p & \text{if } m = (m' \pm 1) \text{ mod } Q \\ 1-2p & \text{if } m = m' \end{cases}, \end{aligned}$$

$$Y = \sum_{k=1}^{2q} X_k \cdot h(qT - kT + \frac{M+1}{Q}T) + N,$$

and $N \sim \mathcal{N}(0, \sigma^2)$ is additive Gaussian noise.

Proof: First, by using the finite-support assumption in (3.2) and (3.4), we have

$$H(\mathcal{E}_i|X_1^\infty, Y_i, \mathcal{E}_{i-1}) = H(\mathcal{E}_i|X_{i-q+\lceil \frac{\mathcal{E}_i}{T} \rceil}^{i+q+\lfloor \frac{\mathcal{E}_i}{T} \rfloor}, Y_i, \mathcal{E}_{i-1}). \tag{3.62}$$

Next, we notice that the i-th output is only determined by the $2q$ binary symbols and the value of M_i as shown in (3.11). Thus, we can write

$$H(\mathcal{E}_i|X_1^\infty, Y_i, \mathcal{E}_{i-1}) = H(M_i|X_{i-q+\lceil \frac{\mathcal{E}_i}{T} \rceil}^{i+q+\lfloor \frac{\mathcal{E}_i}{T} \rfloor}, Y_i, M_{i-1}).$$

Finally, since the input is stationary and the process $\mathcal{M}$ is also asymptotically stationary with a unique steady state distribution as shown in Lemma 2, we can ignore the index i when $i \to \infty$. This proves Lemma 3.

We notice that $H(M|X_1^{2q}, Y, M')$ can be computed numerically using the law of large numbers and averaging over large number of simulations. By substituting (3.51), (3.54), (3.60) and (3.61) into (3.50), we have

$$\begin{aligned}\frac{1}{\ell}h(Y_1^{\ell}|X_1^{\infty}) &\geq h_{LB}(\mathcal{Y}|\mathcal{X}) \\ &\triangleq \frac{1}{2}\log 2\pi e\sigma^2 + \left[2p\log\frac{1}{p} + (1-2p)\log\frac{1}{1-2p}\right] - H(M|X_1^{2q}, Y, M').\end{aligned} \tag{3.63}$$

This lower bound on the conditional entropy rate will lead to an upper bound on the mutual information rate.

3.5 Simulation Results

To assess the proposed information rate bounds, we simulate the algorithms in Sec. 3.4 using the simulation setup described in Sec. 3.3.1. The number of quantization levels in a symbol period, Q is set to 10. The lower bounds are obtained by keeping 40 "surviving" states at each section of the joint trellis. The results obtained are depicted in Fig. 3.8 and Fig. 3.9. The upper and lower bounds on the i.u.d (independent and uniformly distributed) information rate are computed for $\delta = 0.008, 0.01, 0.02, 0.03$ and 0.05. δ is the timing error transition probability defined in (3.7). We notice that the upper bounds exceed 1 (bits/channel use) in high SNR regions. This is due to the fact that any sample Y_i is almost "noiseless" at high SNRs and hence contains more information about the previous timing offset $\mathcal{E}_{i-1}$ than compared to samples at low SNRs. In the derivation of (3.55), we drop the conditioning on Y_ℓ, which leads to a looser bound in high SNR regions. Similar arguments can be made to explain the fact that the upper bound for a bigger value of δ surpasses that of smaller value at high SNRs.

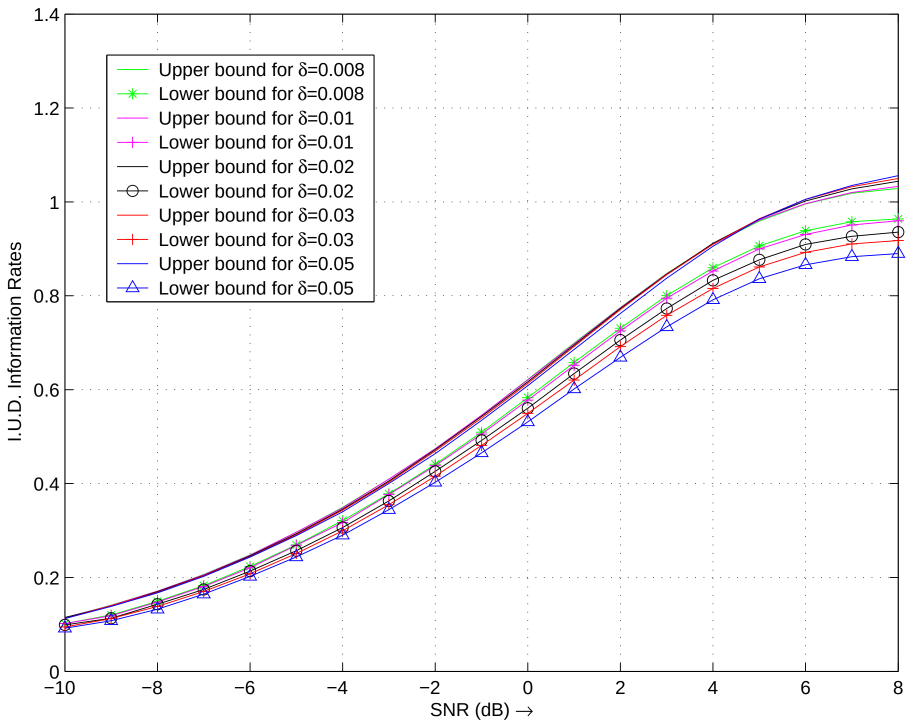
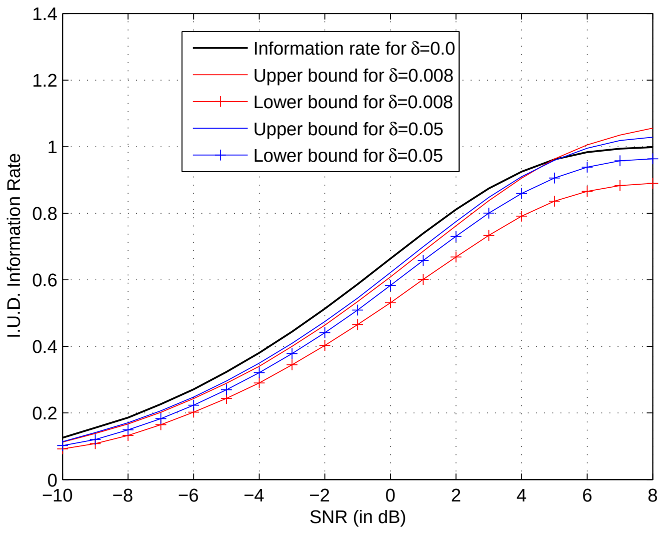
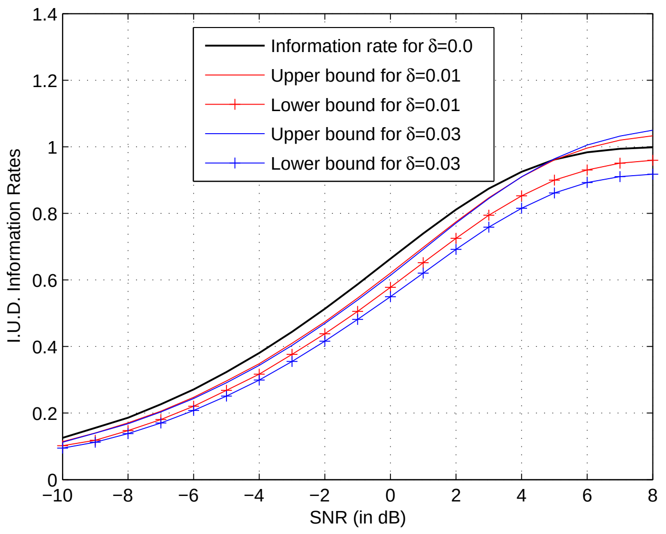


Fig. 3.8: I.U.D. information rate bounds for several values of δ



(a) $\delta = 0.008$ and 0.05



(b) $\delta = 0.01$ and 0.03

Fig. 3.9: The upper and lower bounds on the i.u.d. information rate.

3.6 Summary

In this chapter, we presented a novel mathematical model for channels which are plagued with ISI, additive noise and timing errors. We modelled the quantized timing errors as a first order Markov process. That was followed by development of a finite-state model for the channel. Afterwards, a Monte Carlo method for upper and lower bounding the information rates for the timing error channel was described. Simulation results were presented in the following section.

Chapter 4

Codes for Timing Error Channel

In this chapter, we present error-control codes which help not only in combatting noise, but also aid in regaining synchronization. First off, we give an alternative trellis description of the timing error process presented in the previous chapter. This is followed by the description of a soft-output detection algorithm [48] for linear filter channels, where synchronization errors are quantized and modelled as a Markov process. The algorithm is similar to the BCJR algorithm and generates APPs of the input data symbols and the timing offsets. In the following sections, we delineate our channels codes which consist of a serial concatenation of *Marker codes* and LDPC codes. Marker codes are the inner codes and they assist in re-synchronization at the receiver. LDPC codes are the outer codes; they provide protection against the channel ISI and the additive noise. The code performance is evaluated using simulations. Several plots of the BER performance of the codes are provided along with the code descriptions.

4.1 Alternative Timing Error Trellis

Recall from Sec. 3.1.1 that the Markov model of quantized timing offsets is described by the following two equations:

$$\mathcal{E}_{i+1} = \mathcal{E}_i + \Delta_{i+1}, \tag{4.1}$$

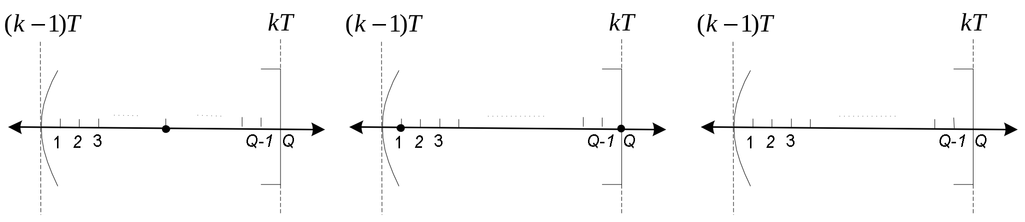
$$P(\Delta_i = \xi_i) = \begin{cases} \delta & \text{if } \xi_i = \frac{T}{Q} \\ \delta & \text{if } \xi_i = -\frac{T}{Q} \\ 1-2\delta & \text{if } \xi_i = 0. \end{cases} \tag{4.2}$$

The process $\{\mathcal{E}_i\}$ take values from a countably infinite set

$$\mathbb{T} = \left\{ \cdots, \frac{-2T}{Q}, \frac{-T}{Q}, 0, \frac{T}{Q}, \frac{2T}{Q}, \cdots \right\}, \tag{4.3}$$

where Q is the number of quantization levels. We define the k-th input symbol interval as the semi-open segment $\big((k-1)T, kT\big]$ on the time axis. A sample in the k-th interval could fall at any of the Q sub-levels. Our particular model for timing errors stipulates that there might be 0, 1 or 2 samples in one symbol interval. Thus, there are three different types of sampling possibilities for each symbol interval as shown in Fig. 4.1.

This gives us an intuition as to an alternative, but equivalent description of the timing error process. We know that due to synchronization mismatch, the number of transmitted symbols may not be equal to the number of received symbols. In the trellis of Fig. 3.3, each section corresponds to one received symbol and variable number of transmitted symbols. This trellis has countably infinite states and thus, is not very useful for practical purposes. However, we notice that the timing error trellis



(a) One Sample. (b) Two samples. (c) No samples.

Fig. 4.1: Three different sampling scenarios for the k-th symbol interval $\big((k-1)T, kT\big]$. The sampling instants are marked by bullets on the time axis.

may also be drawn such that each trellis section corresponds to one input symbol (and variable number of output symbols). This description has a distinct advantage that the number of states in the trellis is finite and small.

We now explain the formation of this new trellis representation. We define a new finite-state timing error process as follows. Let the state associated with the symbol interval $\big((k-1)T, kT\big]$ be ρ_k, which takes values from the set

$$\mathbb{T}' = \big\{0, 1_1, 1_2, \ldots, 1_Q, 2\big\}. \tag{4.4}$$

The cardinality of the set $\mathbb{T}'$ is $Q+2$. The interpretation of the states in $\mathbb{T}'$ is as follows:

- State $\rho_k = 0 \in \mathbb{T}'$ denotes that the k-th interval $\big((k-1)T, kT\big]$ is not sampled at all.
- State $\rho_k = 1_i \in \mathbb{T}'$, for $1 \le i \le Q$ denotes that the k-th symbol interval is sampled only once at the i-th sub-level from the beginning of the interval. An example of this is depicted in Fig. 4.1(a).
- State $\rho_k = 2 \in \mathbb{T}'$ denotes that the k-th symbol interval is sampled twice, at the

1-st and the Q-th sub-level in the interval, as shown in Fig. 4.1(b). Constraints of the Markov process rule out any other way of two samples falling in the same symbol interval.

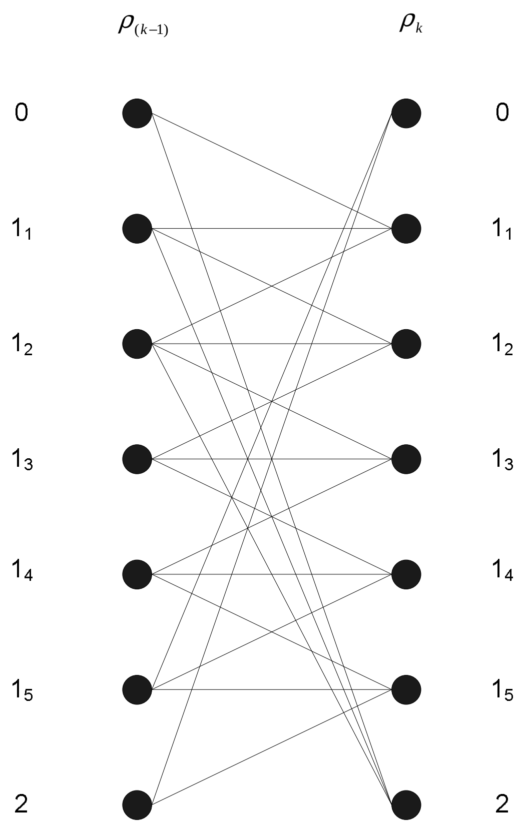
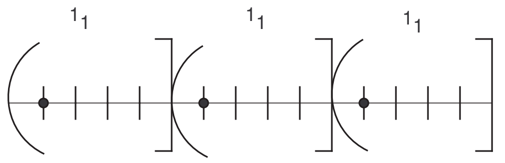
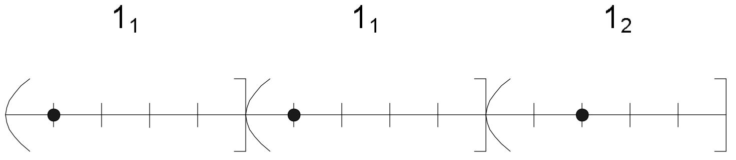


Fig. 4.2: A section of the alternative timing error trellis; drawn for $Q = 5$.

This new timing error process can be represented by a trellis as shown in Fig. 4.2. The trellis is drawn for $Q = 5$ and captures all valid timing state transitions. We associate a transition probability $P(\rho_k|\rho_{k-1})$ with each branch in the trellis. These probabilities are determined by the Markov process of (4.1) and (4.2). For most branches the state transition probabilities are δ or $1 - 2\delta$ and are easily determined. But, for some of the branches, the state transition probabilities are not immediately obvious. We shall now provide one such example. Let us consider the transitions where the starting state is 1_1. We shall assume $Q = 5$.



(a) Possibility 1.



(b) Possibility 2.

Fig. 4.3: Sampling sequences to be considered when computing $P(1_1|1_1)$.

• $\underline{P(1_1|1_1)}$

For computing this state transition probability, we need to consider 3 not 2 symbol intervals as shown in Fig. 4.3. This is required to ensure that the timing offset for the third sample is not $\frac{-T}{5}$, as that would amount to sampling the second interval twice. Using (4.2),

$$\begin{aligned}\text{probability of sampling} &= (1-2\delta)(1-2\delta) + (1-2\delta)\delta \\ &= (1-2\delta)(1-\delta). \qquad (4.5)\end{aligned}$$

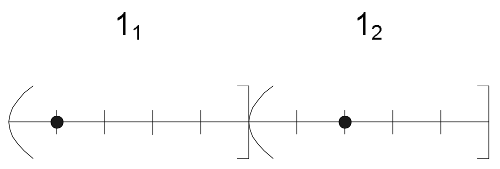


Fig. 4.4: Sampling sequence to be considered for computing $P(1_2|1_1)$.

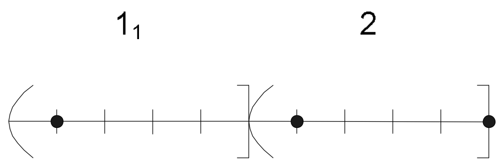


Fig. 4.5: Sampling sequence to be considered for computing $P(2|1_1)$.

• $\underline{P(1_2|1_1)}$

The sampling sequence to be considered for this case is depicted in Fig. 4.4.

$$\text{Probability of sampling} = \delta. \tag{4.6}$$

• $\underline{P(2|1_1)}$

The sampling sequence to be considered for this case is shown in Fig. 4.5.

$$\text{Probability of sampling} = (1-2\delta)\delta. \tag{4.7}$$

Using

$$P(1_1|1_1) + P(1_2|1_1) + P(2|1_1) = 1, \tag{4.8}$$

we can now compute the state transition probabilities as follows:

$$
\begin{aligned}
P(1_1|1_1) &= \frac{(1-2\delta)(1-\delta)}{(1-2\delta)(1-\delta)+\delta+(1-2\delta)\delta} \\
&= 1-2\delta, \qquad (4.9)
\end{aligned}
$$

$$
\begin{aligned}
P(1_2|1_1) &= \frac{\delta}{(1-2\delta)(1-\delta)+\delta+(1-2\delta)\delta} \\
&= \frac{\delta}{1-\delta}, \qquad (4.10)
\end{aligned}
$$

$$
\begin{aligned}
P(2|1_1) &= \frac{(1-2\delta)\delta}{(1-2\delta)(1-\delta)+\delta+(1-2\delta)\delta} \\
&= \frac{(1-2\delta)\delta}{1-\delta}. \qquad (4.11)
\end{aligned}
$$

All other state transition probabilities can be obtained in a similar fashion. A complete list of the timing state transition probabilities is provided in Table 4.1.

Table 4.1: State transition probabilities for the timing error trellis of Fig. 4.2.

$P(0\|1_5) = \delta$	$P(0\|2) = \delta$		
$P(1_1\|0) = 1-\delta$	$P(1_1\|1_1) = 1-2\delta$	$P(1_1\|1_2) = \delta(1-\delta)$	
$P(1_2\|1_1) = \frac{\delta}{1-\delta}$	$P(1_2\|1_1) = 1-2\delta$	$P(1_2\|1_3) = \delta$	
$P(1_3\|1_2) = \delta$	$P(1_3\|1_3) = 1-2\delta$	$P(1_3\|1_4) = \delta$	
$P(1_4\|1_3) = \delta$	$P(1_4\|1_4) = 1-2\delta$	$P(1_4\|1_5) = \delta$	$P(1_4\|2) = \delta$
$P(1_5\|1_4) = \delta$	$P(1_5\|1_5) = 1-2\delta$	$P(1_5\|2) = 1-2\delta$	
$P(2\|0) = \delta$	$P(2\|1_1) = \frac{(1-2\delta)\delta}{1-\delta}$	$P(2\|1_2) = \delta$	

4.1.1 Joint ISI-Timing Error Trellis

For channels which exhibit both ISI and synchronization errors, we can setup a joint trellis, which captures the effect of both ISI and timing errors. We have already seen that for such channels, the nature of the ISI is dependent on the timing offsets. If the channel memory, in presence of timing errors is $\mathcal{P}$, then any state in the ISI trellis is given by $(x_{k-\mathcal{P}+1}, x_{k-\mathcal{P}+2}, \ldots, x_k) \in \mathbb{B}^{\mathcal{P}}$.

We now define the joint trellis by merging the ISI trellis with the timing error trellis of Fig. 4.2. Any state $S_k \in \mathbb{S}'$ at time k of the joint ISI-timing error trellis is determined by a pair of states; the first state is from the ISI trellis $\mathbb{B}^{\mathcal{P}}$ and the second state is from the timing error trellis $\mathbb{T}'$:

$$S_k = (x_{k-\mathcal{P}+1}, x_{k-\mathcal{P}+2}, \ldots, x_k, \rho_k) \in \mathbb{S}' = \mathbb{B}^{\mathcal{P}} \times \mathbb{T}' \tag{4.12}$$

Evidently, a branch in the joint trellis exists if and only if there are corresponding branches in both the ISI and timing error trellises. Since the ISI trellis has $2^{\mathcal{P}}$ states and the timing error trellis has $Q+2$ states, the joint trellis has $2^{\mathcal{P}}(Q+2)$ states. A representative example is provided in Fig 4.6.

We now consider the channel described in Sec. 3.3.1. Its trellis representation, depicted in Fig. 3.8 has countably infinite states. As the channel ISI length is 3 and the number of quantization levels is assumed to be 5, the alternative joint trellis for the same channel will have only 56 states. Thus, now we have two different trellis representations of the same channel, each with its own pros and cons. Trellis of Fig 3.8 has countably infinite states, but standard signal processing algorithms like Viterbi algorithm or BCJR algorithm can be run over it with minimal modifications. On the

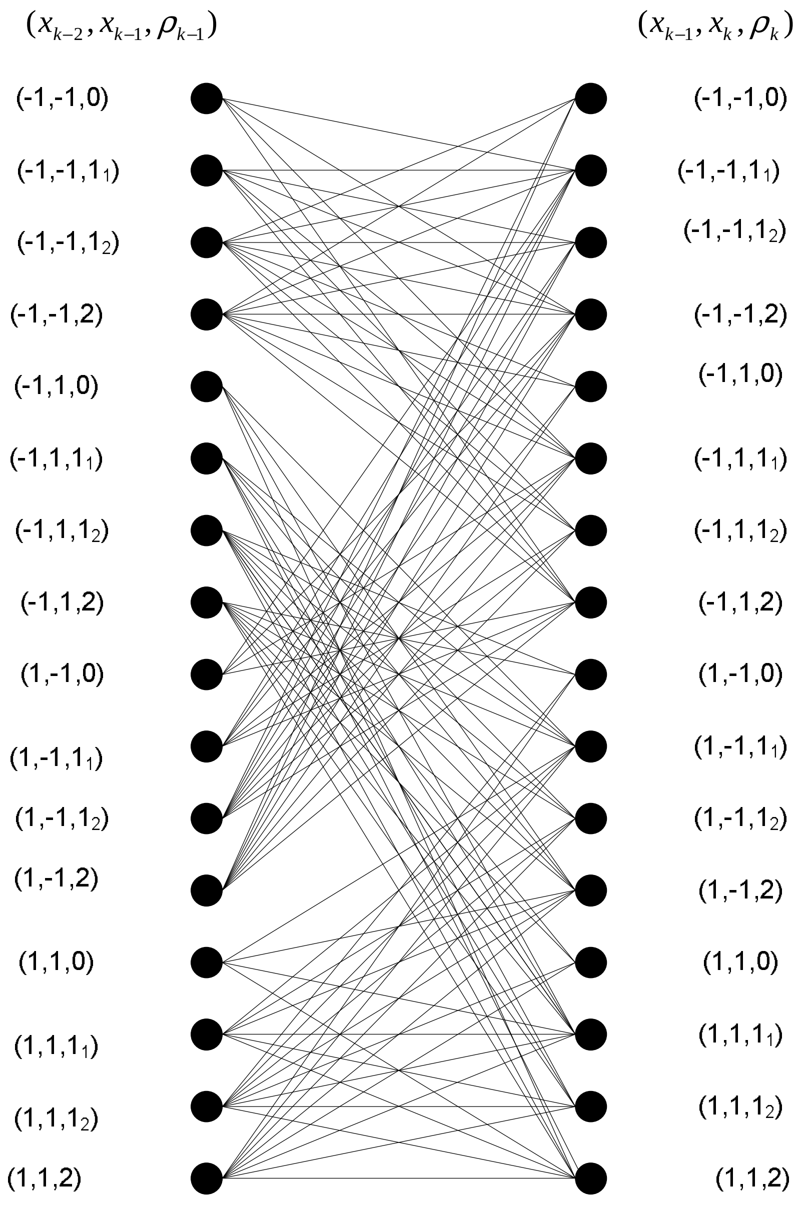


Fig. 4.6: Joint ISI-timing error trellis. We assume that channel ISI length $\mathcal{P} = 2$ and quantization levels $Q = 2$. Any state in the trellis has the form $S_k = (x_{k-1}, x_k, \rho_k)$.

other hand, the joint trellis described above has a finite number of states, but none of the standard signal processing algorithms can be applied on it. We shall now present a MAP algorithm which can run over the joint trellis described in this section.

4.2 A MAP Algorithm

We now describe a MAP algorithm [48] which can be be applied to the joint ISI-timing error trellis presented in the previous section. The algorithm closely follows the BCJR algorithm, although is appropriately modified to handle timing errors. We assume that number of quantization levels is Q. The states in the joint trellis belong to the set $\mathbb{S}' = \mathbb{B}^{\mathcal{P}} \times \mathbb{T}'$. We further assume that y_1^ℓ is received when the sequence x_1^m is transmitted through the channel, where ℓ need not be equal to m. The algorithm aims to compute the following two probabilities:

$$\lambda_t(s) = P(S_t = s; y_1^\ell). \tag{4.13}$$

$$\sigma_i(\psi) = P(\phi_i = \psi; y_1^\ell), \tag{4.14}$$

where ϕ_i is the sampling offset of the i-th sample relative to iT, with $s \in \mathbb{S}'$ and $\psi \in \mathbb{T}$. Let Z_t represent the vector of output samples corresponding to the input symbol X_t. Thus, Z_t may consist of one sample, two samples or no sample, depending on how many samples fall in the interval $\big((t-1)T, tT\big]$. The sequence Z_t^m represents the output samples corresponding to the input sequence X_t^m. We now define some variables which are required in the forward and backward recursions of the algorithm.

Definitions

- Forward state metric

$$\alpha(t, s, i) = P(S_t = s, Z_1^t = y_1^i) \tag{4.15}$$

is the probability that the time t state is $s \in \mathbb{S}'$ and there are i samples in the first t bit periods.

- Backward state metric

$$\beta(t, s, i) = P(Z_{t+1}^{m} = y_{i+1}^{\ell} | S_t = s) \tag{4.16}$$

is the probability that there are $\ell - i$ samples in the last $m - t$ symbol intervals, given that the time t state equals $s \in \mathbb{S}'$.

- Branch metric

$$\gamma(t, s, s', i) = \begin{cases} P(S_t = s'; Z_t = y_i | S_{t-1} = s) & \text{if } \ s' \in \mathbb{B}^{\mathcal{P}} \times \{1_1, \ldots, 1_Q\} \\ P(S_t = s'; Z_t = y_{i-1}^{i} | S_{t-1} = s) & \text{if } \ s' \in \mathbb{B}^{\mathcal{P}} \times \{2\} \\ P(S_t = s'; Z_t = \emptyset | S_{t-1} = s) & \text{if } \ s' \in \mathbb{B}^{\mathcal{P}} \times \{0\}. \end{cases} \tag{4.17}$$

The branch metrics can be computed easily given the *a priori* statistics of the symbol source and timing errors.

Using these definitions, we may rewrite (4.13) and (4.14) as

$$\begin{aligned} \lambda_t(s) &= \sum_{i=1}^{\ell} P(S_t = s; Z_1^t = y_1^i) P(Z_{t+1}^{m} = y_{i+1}^{\ell} | S_t = s) \\ &= \sum_{i=1}^{\ell} \alpha(t, s, i) \beta(t, s, i), \end{aligned} \tag{4.18}$$

$$
\begin{aligned}
\sigma_i(\psi) = & \sum_{\substack{\left\{t,s\,\in\,\mathbb{B}^{\mathcal{P}}\times\{1_1,\ldots,1_Q\},\right.\\ \left.(t-i)T+qT/Q=\psi\right\}}} \alpha(t,s,i)\beta(t,s,i) \\
& + \sum_{\substack{\left\{t,s\,\in\,\mathbb{B}^{\mathcal{P}}\times\{2\},\right.\\ \left.(t-i)T+T=\psi\right\}}} \alpha(t,s,i)\beta(t,s,i) \\
& + \sum_{\substack{\left\{t,s\,\in\,\mathbb{B}^{\mathcal{P}}\times\{2\},\right.\\ \left.(t-i)T+T/Q=\psi\right\}}} \alpha(t,s,i)\beta(t,s,i).
\end{aligned}
\tag{4.19}
$$

Recursive Relations

Using the Markov properties of the joint ISI-timing error process, we can compute the forward and backward state metrics recursively as follows:

$$
\alpha(t,s,i) = \begin{cases} \sum_{s'\in\mathbb{S}'} \alpha(t-1,s',i-1)\gamma(t,s',s,i) & \text{if } \ s \in \mathbb{B}^{\mathcal{P}} \times \{1_1,\ldots,1_Q\} \\ \sum_{s'\in\mathbb{S}'} \alpha(t-1,s',i)\gamma(t,s',s,i) & \text{if } \ s \in \mathbb{B}^{\mathcal{P}} \times \{0\} \\ \sum_{s'\in\mathbb{S}'} \alpha(t-1,s',i-2)\gamma(t,s',s,i-1) & \text{if } \ s \in \mathbb{B}^{\mathcal{P}} \times \{2\}, \end{cases}
\tag{4.20}
$$

$$
\begin{aligned}
\beta(t,s,i) = & \sum_{s'\in\mathbb{B}^{\mathcal{P}}\times\{1_1,\ldots,1_Q\}} \beta(t+1,s',i+1)\gamma(t+1,s,s',i+1) \\
& + \sum_{s'\in\mathbb{B}^{\mathcal{P}}\times\{0\}} \beta(t+1,s',i)\gamma(t+1,s,s',i) \\
& + \sum_{s'\in\mathbb{B}^{\mathcal{P}}\times\{2\}} \beta(t+1,s',i+2)\gamma(t+1,s,s',i+2).
\end{aligned}
\tag{4.21}
$$

Initialization

The forward and backward recursions need to be initialized with a set of coefficients $\alpha(0, s, i)$ and $\beta(m, s, i)$. These initializing conditions are derived from some prior knowledge about the timing error and input symbols at the beginning and the end of a transmitted block. If we assume that when the transmission starts, the ISI state is $\{-1\}^{\mathcal{P}}$ and timing offset $\mathcal{E}_0 = 0$, we have the following initial condition for the forward recursion:

$$\alpha(0, s, i) = \begin{cases} 1 & \text{if } i = 0 \text{ and } s \in \{-1\}^{\mathcal{P}} \times 1_Q \\ 0 & \text{otherwise.} \end{cases} \tag{4.22}$$

If we assume that the receiver is aware of the block boundaries, in other words it knows the value of ℓ, then the backward recursion can be initialized as follows:

$$\beta(m, s, i) = \begin{cases} 1 & \text{if } i = \ell \\ 0 & \text{otherwise.} \end{cases} \tag{4.23}$$

A Posteriori Probabilities

The a-posteriori estimates of the input symbols and the timing errors may now be obtained as follows:

$$\frac{P(x_t = 1|y_1^\ell)}{P(x_t = -1|y_1^\ell)} = \frac{\sum_{s:x_t=1} \lambda_t(s)}{\sum_{s:x_t=-1} \lambda_t(s)}, \tag{4.24}$$

$$P(\phi_i = \psi|y_1^\ell) = \sigma_i(\psi)/P(y_1^\ell). \tag{4.25}$$

Some Practical Alterations

The modified BCJR algorithm requires a memory size proportional to m^2, where m is the transmitted block length. But, since the timing transition probability δ is small in practical systems, for a particular value of t, $\alpha(t, s, i)$ and $\beta(t, s, i)$ are very likely to be zero as the value of i moves away from t. Thus, in practical systems, it is not essential to compute the α and β values at any instant for all values of i. Instead, we can restrict the search in a window of size "w" about t. This means that at each time t, we compute $\alpha(t, s, i)$ and $\beta(t, s, i)$ for $t - w \leq i \leq t + w$, and assume $\alpha(t, s, i) = \beta(t, s, i) = 0$ for all other values of i. This modification can drastically reduce the storage requirements of the algorithm.

Like before, we need to normalize the forward and backward state metrics at each section of the joint trellis to circumvent numerical underflow. The normalization will not affect the APP estimates of the input symbols, as at any time t all the α (as well as β) values are scaled by the same factor. However, APP estimate of the timing offsets ϕ_i will require the proper accounting of the normalizing factors.

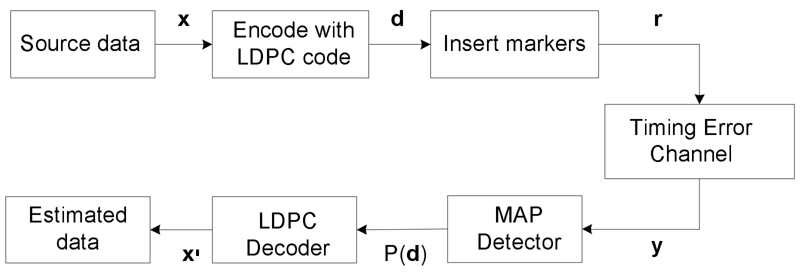


Fig. 4.7: Overview of the encoding-decoding process.

4.3 A Concatenated Error-Control Code

The code is comprised of a serial concatenation of marker code and LDPC code. A block schematic of the coding-decoding process is outlined in Fig 4.7. An input symbol vector $\mathbf{x}$ of length a is encoded using an LDPC code of rate R_{out}. Markers are then inserted in the resulting vector $\mathbf{d}$ of length n, to obtain a binary vector $\mathbf{r}$.

The length m of $\mathbf{r}$ is contingent upon the marker code rate R_{in}. The overall code rate is given by $R = R_{out} \times R_{in}$. The real valued vector received on transmitting $\mathbf{r}$ over the channel is denoted by $\mathbf{y}$. Note that the length ℓ of vector $\mathbf{y}$ may not be equal to m. The MAP decoder generates the APP estimate of the vector $\mathbf{d}$. This information is used to initialize the LDPC decoding algorithm. The LDPC algorithm produces soft-estimates of the input vector $\mathbf{m}$.

For all simulations in this chapter, we will use the setup described in Sec. 3.3.1. The symbol source is assumed to be i.u.d. The block length of data symbols is taken to be 4157. The vector $\mathbf{y}$ is obtained by sampling the received waveform with timing offsets generated by the process $\{\mathcal{E}_i\}$. The number of quantization levels in one symbol interval is $Q = 5$. We define the SNR as

$$SNR = \frac{R \cdot E_b}{N_0}$$

where E_b is the bit-energy and N_0 is the spectral density of the AWGN.

4.3.1 Marker Codes

Marker codes were initially proposed by Sellers in [19] for insertion or deletion channels. The idea is to insert "markers" at regular intervals in the bit stream to be transmitted. We refer to a set of consecutive markers as a *header*. For example, adding the header 011 with a spacing of 5 transforms the bit stream as follows:

$$01001110100110101 \;\Rightarrow\; 01001\mathbf{011}11010\mathbf{011}01101\mathbf{011}01$$

The position and bit values of the markers is known to the receiver. Hence, the decoder searches for inserted headers, and uses any shift in their position to deduce insertions or deletions. The algorithm for decoding of marker codes presented in [19], provides very limited synchronizing capability and fails if the number of insertions (or deletions) exceeds 1. Here we will use marker codes for our channel to help the receiver recover from synchronization loss. However, we will use a different, optimal algorithm for decoding. A marker code is characterized by two parameters, header length (HL), which is the number of markers in a header and, header spacing (HS), which is the number of data bits separating any two headers. The true code rate for marker codes may depend on the length of the input data block. However, it can be approximated as

$$R_{in} = \frac{HS}{HL + HS}. \tag{4.26}$$

Choice of Markers: We may have *fixed markers*, in which case the same header is repeated through out the transmitted sequence. Or, we may have *random headers*, in which the marker bits in each header are drawn from a pseudo random sequence. From our simulations we found that both fixed markers and random markers give roughly the same average performance. However, in high SNR regions, random marker codes outperform fixed marked codes by a very small margin. All the subsequent simulation results have been obtained using random headers. The marker bits are produced using a uniform random bit generator.

Decoding of marker codes

We use the MAP algorithm presented in Sec. 4.2 for the decoding of marker codes. The marker bits serve as a source of extra information to the decoder. The decoder has no a-priori information about the data bits. However, it has complete knowledge of all the marker bits. During the operation of the algorithm, in trellis sections corresponding to marker bits, the number of possible transitions are halved (as we only need consider transitions induced either by a +1 or a -1). Thus, in these sections, the decoder is better equipped to guess the timing offset. As the timing error process has memory, a better estimate of the timing offset for marker bits leads to a better assessment of the timing offsets in the neighbouring samples. Obviously, this translates into more accurate estimates of the data bits.

For our simulations, we assume that the decoder is aware of the block boundaries.

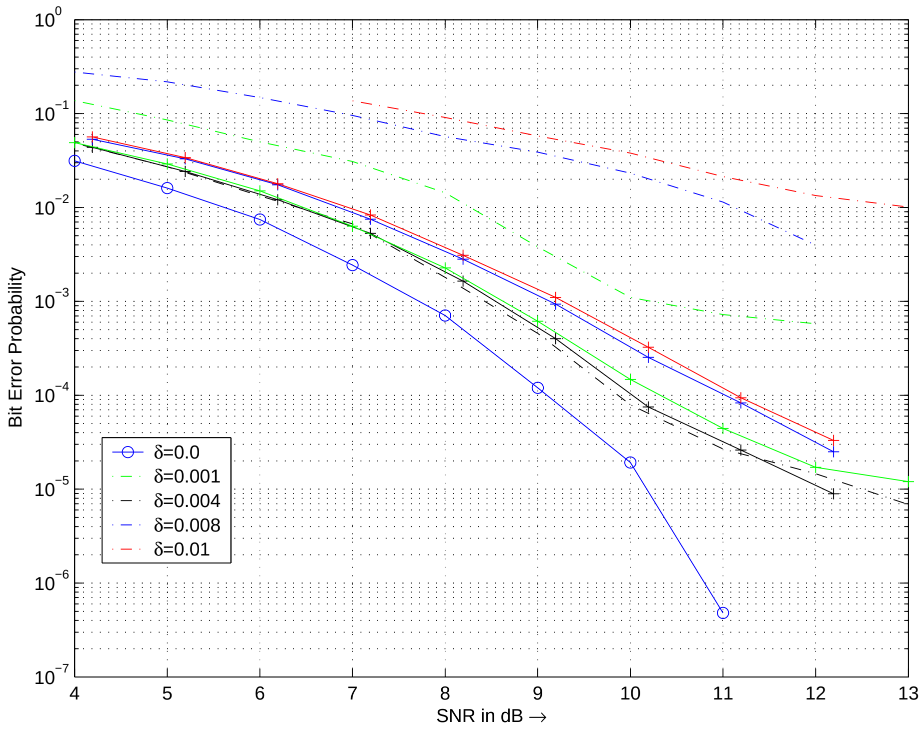


Fig. 4.8: Comparison of bit error probabilities with and without marker codes. For all non-zero values of δ, the broken curve is for uncoded performance. The solid curve (with + signs) with the same colour depicts the corresponding BER when marker codes are employed. The marker codes used in all the simulations have HS = 44 and HL = 2 (R_{in} = 0.9565). No outer code is employed.

For magnetic recording systems this is not an unrealistic assumption. The data bits in each sector of a hard disk are preceded by pre-ambles, which can be used in block boundary identification. The pre-ambles also assist in regaining synchronization at the beginning of each block. So, we further assume that $\mathcal{E}_0 = 0$ for each block. To expedite the simulations we use the windowed version of the algorithm, with $w = 50$. The bit error rate (BER) performance curves for the codes are presented in Fig. 4.8 and Fig. 4.9. From Fig. 4.9, it is evident that marker codes provide only limited

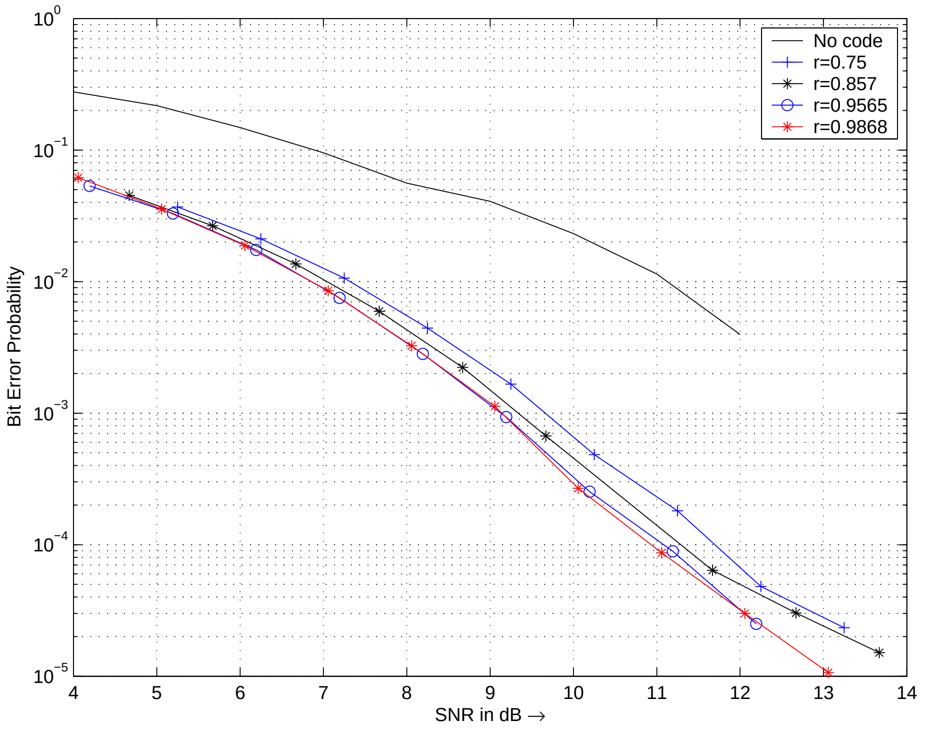
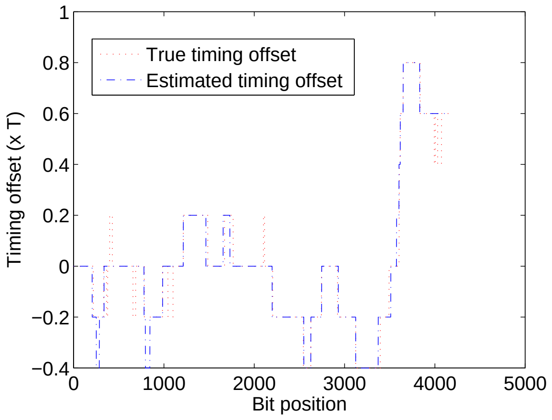


Fig. 4.9: Bit error probabilities for $\delta = 0.008$ for several different marker code rates. HL $= 2$ is all cases, only HS is varied.

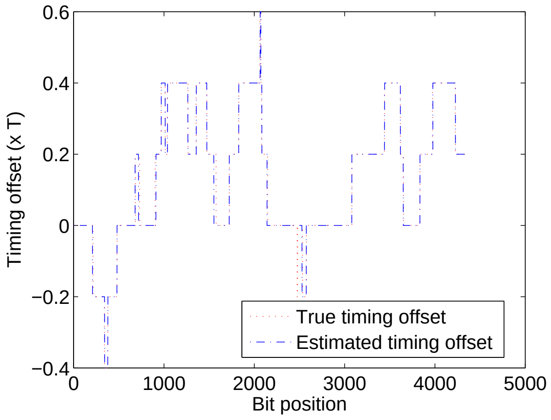
error-control capability as high rate marker codes outperform lower rate ones. This is because at low rates, the coding gain provided by marker codes is not sufficient to make up for the reduction in SNR due to coding.

From Fig. 4.8 notice that the performance improvement brought by marker codes is higher for larger values of δ. Also note that the gap between any two broken curves is much more than the solid curves in the same colours. This implies that marker codes reduce the dependence of decoder performance on δ value.

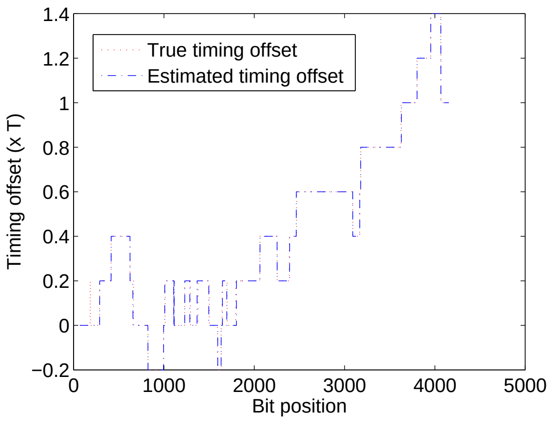
In Fig. 4.10, Fig. 4.11, and Fig. 4.12 we present typical curves to demonstrate the



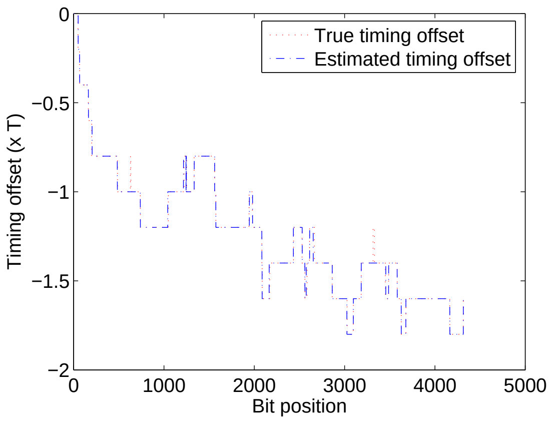
(a) SNR = 5 dB, without marker codes



(b) SNR = 5 dB, marker code rate = 0.9565



(c) SNR = 10 dB, without marker codes



(d) SNR = 10 dB, marker code rate = 0.9565

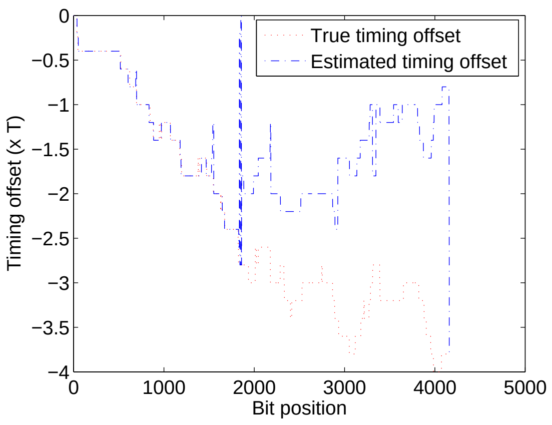
Fig. 4.10: Timing error tracking by the MAP detector when $\delta = 0.004$.

efficacy of the MAP algorithm in estimating the timing offsets. We notice that when either the timing error transition probability δ is low, or SNR is high (or both), the detector performs satisfactorily. Although, the detector is unable to follow some of the narrow peaks in the true timing offsets, there are no cycle slips. For all such cases, the improvement brought by the use of marker codes is not easily discernible. However, we notice that the detector performance deteriorates sharply when we increase the values of δ and noise variance. As can be seen from Fig. 4.11(a) and Fig. 4.12(c), cycle slips are rampant in the middle of the block. Once a cycle slip occurs, the detector is not able to recover from it till the end of the block. Now, if we see the corresponding tracking curves when marker codes are employed, we notice that all the cycle slips have been eliminated and the detector performs a good assessment of the timing offsets.

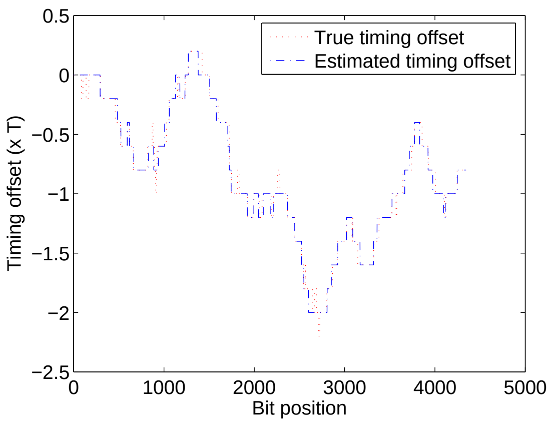
4.3.2 LDPC Code

As is evident from the simulation results presented in the previous section, marker codes are very effective in helping the receiver regain synchronization. Although there is also a fairly good reduction in the BER; but overall, the error performance is not completely satisfactory. This is because marker codes have no error-correcting capability. Therefore, we need an outer code to protect our data against ISI and additive Gaussian noise. We use an LDPC code as the outer code. The reasons for this choice will become apparent soon.

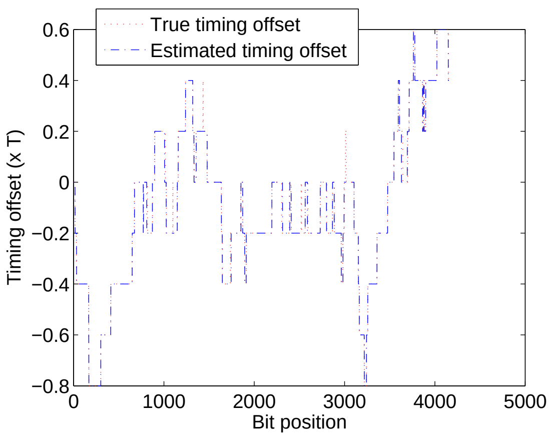
Parity-Check Matrix: In our simulations, we use regular LDPC codes based upon circulant permutation matrices. Recalling (2.85) and (2.86), the parity-check matrix



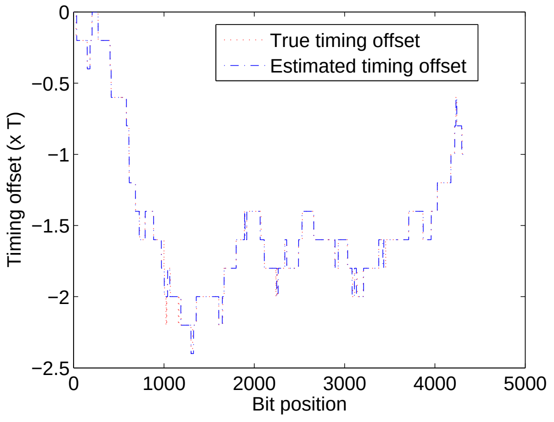
(a) SNR = 5 dB, without marker codes



(b) SNR = 5 dB, marker code rate = 0.9565

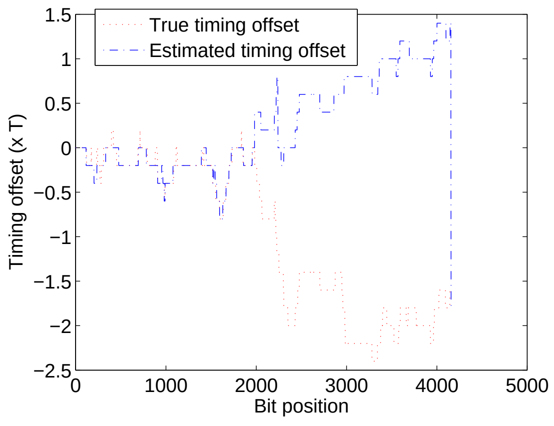


(c) SNR = 10 dB, without marker codes

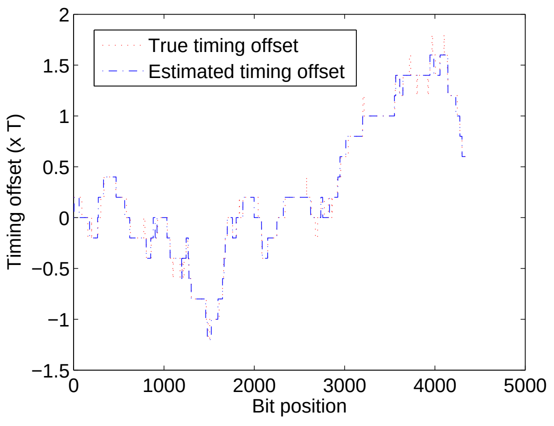


(d) SNR = 10 dB, marker code rate = 0.9565

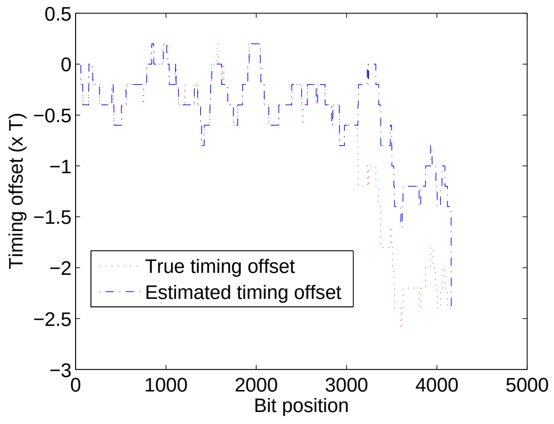
Fig. 4.11: Timing error tracking by the MAP detector when $\delta = 0.008$.



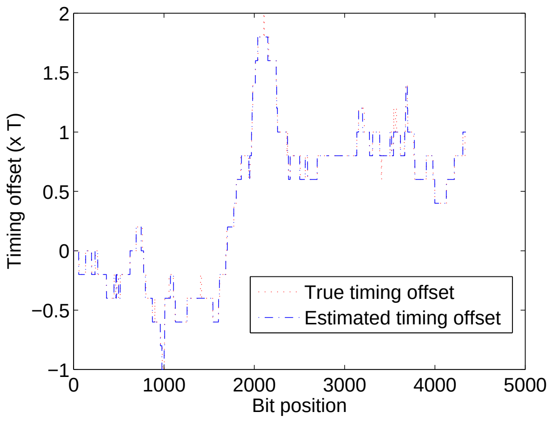
(a) SNR = 5 dB, without marker codes



(b) SNR = 5 dB, marker code rate = 0.9565



(c) SNR = 10 dB, without marker codes



(d) SNR = 10 dB, marker code rate = 0.9565

Fig. 4.12: Timing error tracking by the MAP detector when $\delta = 0.01$.

is given by

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{I} & \mathbf{I} & \mathbf{I} & \cdots & \mathbf{I} \\ \mathbf{I} & \sigma & \sigma^2 & \sigma^3 & \cdots & \sigma^{L-1} \\ \mathbf{I} & \sigma^2 & \sigma^4 & \sigma^6 & \cdots & \sigma^{2(L-1)} \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{I} & \sigma^{J-1} & \sigma^{2(J-1)} & \sigma^{3(J-1)} & \cdots & \sigma^{(J-1)(L-1)} \end{bmatrix}, \qquad (4.27)$$

where σ is a $p \times p$ ($p \geq 5$ is prime)matrix and is given by

$$\sigma = \begin{bmatrix} & & & & 1 \\ 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \end{bmatrix}_{p \times p} . \tag{4.28}$$

Equations (4.27) and (4.28) completely describe a (n, J, L) code, where the block length $n = Lp$.

Decoding: Decoding of the LDPC code is done by running the sum-product algorithm over the Tanner graph of the LDPC code as delineated in Sec. 2.5.1.

Overall decoding of the serially concatenated codes: Fig. 4.13 depicts the

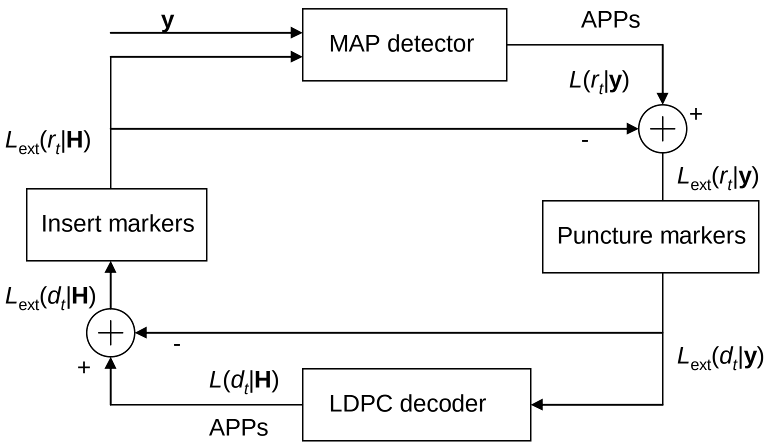

y
MAP detector
APPs
$L(r_t|\mathbf{y})$
+
-
$L_{\text{ext}}(r_t|\mathbf{H})$
$L_{\text{ext}}(r_t|\mathbf{y})$
Insert markers
Puncture markers
$L_{\text{ext}}(d_t|\mathbf{H})$
-
+
$L_{\text{ext}}(d_t|\mathbf{y})$
$L(d_t|\mathbf{H})$
LDPC decoder
APPs

Fig. 4.13: Iterative decoding of the serially concatenated code.

iterative decoding of the serial concatenation of marker codes and LDPC codes. The two decoders exchange *extrinsic information* alternately in the form of likelihood ratios or their logarithms. We define the conditional log-likelihood ratio (LLR) of a binary random variable r_t given $\mathbf{y}$ as

$$L(r_t|\mathbf{y}) = \ln \frac{P(r_t = 1|\mathbf{y})}{P(r_t = 0|\mathbf{y})}. \tag{4.29}$$

The conditional LLR $L(r_t|\mathbf{y})$ splits into two components, the extrinsic LLR $L_{\text{ext}}(r_t|\mathbf{y})$ and intrinsic/a-priori LLR $L(r_t)$, i.e.

$$L(r_t|\mathbf{y}) = L_{\text{ext}}(r_t|\mathbf{y}) + L(r_t). \tag{4.30}$$

The extrinsic information about any bit is obtained from the constraints imposed by the channel or code and the a-priori information about all the other bits in the sequence. Notice that in Fig. 4.13 only the extrinsic information is being fed forward and backward. The LDPC decoder uses the extrinsic information about the bits as a-priori information and vice versa. The sequence of operation is as follows: The MAP detector generates conditional LLRs $L(r_t|\mathbf{y})$ for $\mathbf{r}$ using the received sequence $\mathbf{y}$ and extrinsic information $L_{\text{ext}}(r_t|\mathbf{H})$ provided by the LDPC decoder. Its output is used to obtain $L_{\text{ext}}(d_t|\mathbf{y})$. These LLRs are used by the forward-backward algorithm in the LDPC decoder. The decoder generates the extrinsic information to be fed back to the marker decoder, and also the estimate $\mathbf{x}'$ of the data vector $\mathbf{x}$. The above series of operations constitutes one iteration of the decoding process. Note that in the 0-th iteration, the marker decoder doesn't have any extrinsic information from the LDPC decoder. Also, the markers are known to the receiver and so are their likelihood ratios.

Simulation Results

Fig. 4.14 shows the performance of the serially concatenated code. These simulations were conducted by setting these values to the following parameters: the outer code is a (4422,4,66) LDPC code; the value of $p = 67$. The outer code rate is $R_{out} = 0.9401$. A marker code of rate $R_{in} = 0.987$ ($HS = 149, HL = 2$) is used as the inner code. Thus, the overall code rate is 0.928.

As is seen in Fig. 4.14, the error performance of the receiver is enhanced with each iteration of extrinsic information exchange between the marker decoder and the LDPC decoder. Also notice that the improvement brought by iterative decoding keeps decreasing as the number of iteration grows. This is due to the presence of cycles in the Tanner graph of the code. The results presented in this chapter are indicative of the promise held by marker codes and their concatenation with LDPC codes. The complex nature of the timing error channel makes the theoretical modelling of marker code functioning very difficult. Due to this, we could not perform a more comprehensive analysis of these codes.

4.4 Summary

In this chapter, we presented a novel channel code design methodology for the timing error channel described in Chapter 3. We first showed an alternative trellis representation for the timing error channel. Then, we delineated a MAP algorithm for the timing error channel. That was followed by the description of a serially concatenated code, which is capable of timing recovery as well as error correction. Simulation results were presented in the following section.

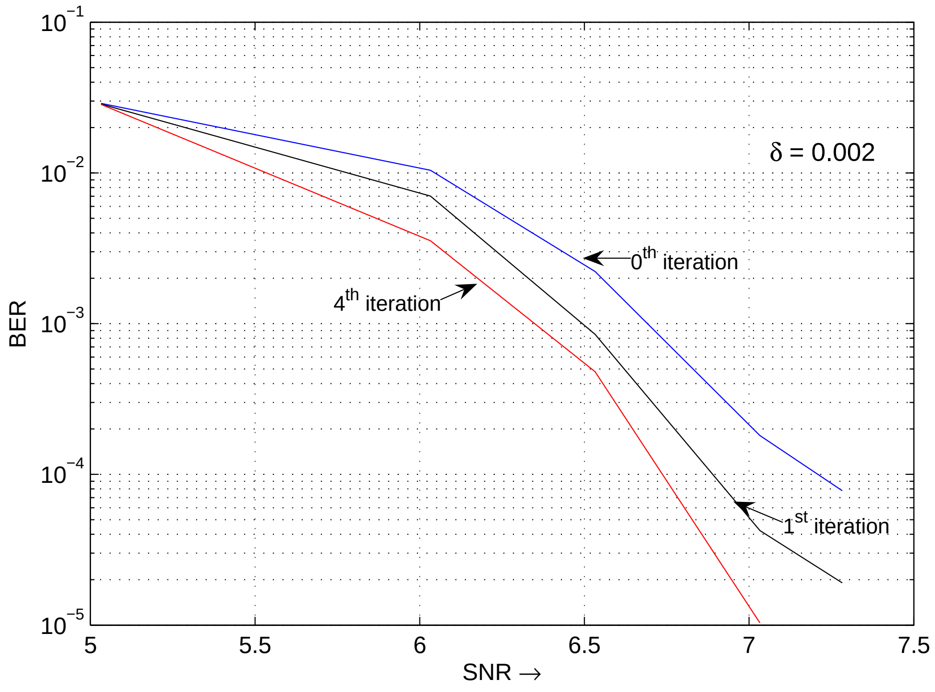


Fig. 4.14: Error performance of the serially concatenated code when $\delta = 0.002$.

Chapter 5

Conclusions and Future Work

In this work, we investigated noisy channels which are also corrupted by timing errors. We studied a more practical and general case than the insertion/deletion channel. In our model, the timing errors can be a quantized fraction of the symbol interval. We employ a very general baseband linear filter channel model and, inject timing errors in it. This is the setup we used for all investigations here.

The two main contributions of this thesis are contained in Chapters 3 and 4. In the former, we have obtained some new fundamental information theoretic results. We present two different ways of representing our timing error channel. The first representation is as an FSM and, in the other we model the channel as a trellis with countably infinite states. We exploit the structure and the Markovian properties of our channel model to compute the mutual information rates. The Monte-Carlo methods that we introduced provide tight upper and lower bounds to information rates for channels with timing errors. This implies that the capacity of such channels is sandwiched between the upper and the lower bound, and is known within a fraction of dB.

In Chapter 4, we presented serially concatenated codes for channels with timing errors. Marker codes are the inner code; they provide probabilistic re-synchronization. Our simulations show that even very high rate marker codes bring significant improvement in the receiver's performance in tracking the timing offsets. A regular LDPC code forms the outer code. The LDPC codes help in controlling errors due to ISI and AWGN. The marker decoder and LDPC decoder exchange extrinsic information alternately to produce better and better estimates of the transmitted data.

Directions For Future Work

We believe that the following problems hold promise and may be very interesting to investigate:

- The mutual information rate for a channel may also be written as

  $$I(\mathcal{X};\mathcal{Y}) = H(\mathcal{X}) - h(\mathcal{X}|\mathcal{Y}). \tag{5.1}$$

  Analytical expressions are available for $H(\mathcal{X})$ for most of the commonly used symbol sources. In [49], a Monte-Carlo method was presented to estimate $h(\mathcal{X}|\mathcal{Y})$ for the case of linear filter channels. In [49], the author also introduced an expectation maximization algorithm to compute the capacity of such channels. One may attempt to extend the algorithm in [49] to include channels with timing errors. The advantage of this approach is that not only we can estimate the capacity, but also obtain the capacity-achieving source.

- There is a need for a theoretical framework for analyzing marker codes. Such a model could be used to design optimum marker codes given the channel parameters. It would also be interesting to compare this model to the performance shown by experimental decoding.

- In [48], it was shown that the symbol error probability is minimum at the block boundaries and reaches it's maximum in the middle of the block. As the error probabilities are different at different positions in a block, it would be beneficial to probe the performance of codes which provide unequal error protection.

- It would be instructive to compare the performance of watermark codes [20] with marker codes in our channel model. Although, the decoding complexity of watermark codes will be considerably higher than that of marker codes, they might outperform marker codes.

Bibliography

[1] C. E. Shannon, “A mathematical theory of communication,” *Bell Syst. Tech. J.*, vol. 27, pp. 623–656, Oct. 1948.

[2] J. Liu, H. Song, and B. V. K. Vijay Kumar, “Dual segmented kalman filters based symbol timing recovery for low-snr partial response data storage channels,” in *Proc. IEEE Intl. Conf. Global Telecommun. (GLOBECOM)*, San Fransisco, USA, Dec. 2003, vol. 7, pp. 4084–4090.

[3] J. R. Barry, A. Kavcic, S. W. McLaughlin, A. Nayak, and W. Zeng, “Iterative timing recovery,” *IEEE Signal Processing Magazine*, vol. 21, pp. 89–102, Jan. 2004.

[4] R. L. Dobrushin, “Shannon’s theorems for channels with synchronization errors,” *Problems of Information Transmission*, vol. 3, no. 4, pp. 18–36, 1967.

[5] R. G. Gallager, “Sequential decoding for binary channels with noise and synchronization errors,” Unpublished Lincoln Lab. report 25 G-2, 1961.

[6] S. N. Diggavi and M. Grossglauser, “On transmission over deletion channels,” *Proc. Allerton Conference 2001, Monticello, Illinois*, Oct. 2001.

[7] E. Drinea and M. Mitzenmacher, “On lower bounds for the capacity of deletion channels,” in *Proc. International Symposium on Information Theory*, Chicago, Illinois, Jun. 2004, p. 227.

[8] J. D. Ullman, “On the capabilities of codes to correct synchronization errors,” *IEEE Trans. Information Theory*, vol. 13, no. 5, pp. 95–105, January 1967.

[9] R. L. Dobrushin, “The computation on a computer of the channel capacity of a line with symbols drop-out,” *Problems of Information Transmission*, vol. 4, no. 3, pp. 92–95, 1968.

[10] A. Kavcic and R. Motwani, “Insertion/deletion channels: reduced-state lower bounds on channel capacities,” in *Proc. IEEE Int. Sym. Information Theory*, Chicago,IL, Jun. 2004.

[11] B. Gordon S. W. Golomb and L. R. Welch, “Comma-free codes,” *Canadian Journal of Mathematics*, vol. 10, no. 2, pp. 202–209, 1958.

[12] J.J. Stiffler, “Comma-free error correcting codes,” *IEEE Trans. Information Theory*, vol. 11, no. 1, pp. 107–111, Jan. 1965.

[13] S. E. Tavares and M. Fukada, “Matrix approach to synchronization recovery for binary cyclic codes,” *IEEE Trans. Information Theory*, vol. 15, no. 1, pp. 93–101, Jan. 1969.

[14] V I. Levenshtein, “Binary codes capable of correcting deletions, insertions and reversals,” *Soviet Physics-Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[15] E. Tanaka and T. Kasai, "Synchronization and substitution error-correcting codes for levenshtein metric," *IEEE Trans. Information Theory*, vol. 22, no. 2, pp. 156–162, Mar. 1976.

[16] G. Tenengolts, "Nonbinary codes, correcting single deletion or insertion," *IEEE Trans. Inforamtion Theory*, vol. 30, no. 5, pp. 766–769, Sept. 1984.

[17] T. Mori and H. Imai, "Viterbi decoding considering insertion/deletion errors," in *Proc. International Symposium Information Theory.* Sept., 1995, p. 145.

[18] M. F. Mansour and A. H. Tewfik, "Convolutional codes for channels with substitutions, insertions and deletions," in *Proc. IEEE Intl. Conf. Global Telecommun. (GLOBECOM)*, Nov. 2002, vol. 2.

[19] F. F. Sellers, "Bit loss and gain correction code," *IRE Trans. Information Theory*, vol. 8, no. 1, pp. 35–38, Jan. 1962.

[20] M. C. Davey and David. J. C. Mackay, "Reliable communication over channels with insertions, deletions and substitutions," *IEEE Trans. Information Theory*, vol. 47, no. 2, pp. 687–698, Feb 2001.

[21] E. A. Ratzer, "Marker codes for channels with insertions and deletions," in *3rd International Symposium on Turbo Codes and Related Topics*, Brest, France, Sept. 2003.

[22] H. N. Bertram, *The Theory of Magnetic Recording*, Cambridge University Press, Apr 1994.

[23] Y. Ephraim and N. Merhav, "Hidden markov processes," *IEEE Trans. Inform Theory*, vol. 48, no. 6, pp. 1518–1569, Jun. 2002.

[24] Z. Ghahramani M. J. Beal and C. E. Rasmussen, "The infinite hidden markov model," *Advances in Neural Information Processing Systems*, vol. 14, pp. 577–585, 2002.

[25] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*, Mc Graw Hill, fourth edition, 2002.

[26] B. G. Leroux, "Maximum-likelihood estimation for hidden markov models," *Stochastic Processes and their Application*, vol. 40, pp. 127–143, 1992.

[27] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Information Theory*, vol. 20, pp. 284–287, Mar. 1974.

[28] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, USA, 1991.

[29] A. R. Barron, "The strong ergodic theorem for densities: Generalized shannon-mcmillan-breiman theorem," *The Annals of Probability*, vol. 13, no. 4, pp. 1292–1303, 1985.

[30] R. G. Gallager, *Information Theory and Reliable Communicaition*, Wiley, New York, 1968.

[31] D. Arnold and H.-A. Loeliger, "On information rate of binary-input channels with memory," in *Proc. IEEE Int. Conf. Communications (ICC)*, Helinski, Finland, Jun. 2001, vol. 9, pp. 2692–2695.

[32] H. D. Pfister, J. B. Soriage, and P. H. Siegel, "On the achievable information rates of finite state ISI channels," in *Proc. IEEE Globecom 2001*, San Antonio, TX, Nov. 2001, pp. 2992–2996.

[33] V. Sharma and S. K. Singh, "Entropy and channel capacity in the regenerative setup with applications to markov channels," in *Proc. IEEE ISIT 2001*, Washington DC, USA, Jun. 2001, p. 283.

[34] R. G. Gallager, "Low-density parity check codes," *IRE Trans. Information Theory*, vol. IT-18, pp. 21–28, Jan. 1962.

[35] R. G. Gallger, *Low-Density Parity Check Codes*, Ph.D. thesis, MIT, Cambridge,MA, 1963.

[36] D. J. C. Mackay and R. M. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding, 5th IMA Conference*, 1995, vol. 1025, pp. 110–111.

[37] A. Shokrollahi T. Richardson and R. Urbanke, "Design of provably good low-density parity-check codes," *IEEE Trans. Information Theory*, vol. 47, pp. 808–821, Feb. 2001.

[38] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Information Theory*, vol. IT 27, no. 5, pp. 533–547, Sept. 1981.

[39] B. Frey F. Kschischang and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Information Theory*, vol. 47, pp. 498–519, Feb. 2001.

[40] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inferenece*, Morgan Kaufmann, San Mateo, 1988.

[41] N. Wiberg, *Codes and decoding on general graphs*, Ph.D. thesis, Linkoping University, Sweden, 1996.

[42] R. Smith, "Easily decoded efficient self-orthogonal block codes," *Electron. Lett.*, vol. 13, no. 7, Mar. 1977.

[43] W. W. Peterson and E. J. Weldon Jr, *Error-Correcting Codes*, MIT Press, 2 edition, 1972.

[44] M. P. C. Fossorier, "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Information Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[45] W. Zeng, J. Tokas, R. Motwani, and A. Kavcic, "Bounds on mutual information rates of nosiy channles with timing errors," in *Proc. IEEE Int. Symposium Information Theory (ISIT)*, Adelaide, Austrailia, Sept. 2005.

[46] L. Breiman, "The individual ergodic theorem of information theory," *The Annals of Math. Statist.*, vol. 28, pp. 809–811, 1957.

[47] D. Arnold, A. Kavcic, H.-A Loeliger, P. Vontobel, and W. Zeng, "Simulation based computation of information rates: upper and lower bounds by reduced-

state techniques," in *Proc. IEEE Int. Sym. on Information Theory*, Yokohama, Japan, Jun. 2003, p. 119.

[48] W. Zeng and A. Kavcic, "Map detection in noisy channel with synchronization errors (including the insertion/deletion channel)," *IEEE Trans. Magnetics*, vol. 39, pp. 2255–2257, Sept. 2003.

[49] A. Kavcic, "On the capacity of markov sources over nosiy channels," in *Proc. IEEE Global Communications Conference*, San Antonio, Texas, Nov. 2001, pp. 2997–3001.