

Theoretical Modeling and Practical Operation of Channels with Output Memory

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE
UNIVERSITY OF HAWAII AT MĀNOA IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY
IN
ELECTRICAL ENGINEERING

February 2016

By

Meysam Asadi

Dissertation Committee :

Narayana Prasad Santhanam, Chairperson

Aleksandar Kavcic

Anthony Kuh

Anders Høst-Madsen

Bjørn Kjos-Hanssen

©Copyright 2016
by
Meysam Asadi

To my parents

Acknowledgements

I would like to express my sincere gratitude to my advisor Prof. Santhanam for his continuous support during my Ph.D study, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. Prasad offered me lots of interesting problems, and giving me lots of freedom to explore new directions which incited me to widen my research from various perspectives.

My sincere thanks also goes to Prof. Kavcic. Alek initially introduced me with Flash memory channels and has been extremely supportive during my Ph.D journey. Alek not only taught me how to be persistent, visionary and patient in research, he also taught me how to be a nice and understanding person in life.

I wish to express my sincere thanks to Dr. Erich F. Haratsch who provided me an internship opportunity to join his team at LSI Corporation, and who gave me access to precious research data. I would like to thank my internship mentor Dr. Chen and other colleagues Dr. Alhussien, Dr. Cai, Dr. Li and Dr. Sankaranarayanan for their suggestions on my research and career.

I would like to thank Prof. Kuh, Prof. Host-Madsen and Prof. Kjos-Hanssen, for serving on my committee and providing insightful comments and helpful advises on my research despite their tight schedule. I take this opportunity to express gratitude to all of the Department faculty members for their help and support.

Furthermore, I would like to thank my amazing labmates for their cheerful cooperation, encouragement, help and support during the course of my study, Nasir, Maryam, Navid and Elyas. Also, my sincere thanks to good friends Ehsan, Reza, Bahram, Poya, Saeed, Ashkan and Ali have accompanied me through this journey in many ways.

Lastly, and most importantly, I take immense pleasure in thanking my parents, Nasrat and

Mohammad, my older sister Maria and my brothers Mehdi and Maher for their love, support and unceasing encouragement throughout my life. I am also deeply grateful to my beloved girlfriend, Amanda who has been virtually working as hard as me on this thesis. I couldn't imagine how would I finish this thesis if it were not for her constant love and faith in me through all those hard times together.

I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their helping hand in this venture.

Abstract

In this work, we focus on a subclass of channels with memory, called “channels with output memory”, in which the state of the channel is solely characterized by the previous output of the channel. This thesis contains two parts and covers modeling, signal processing and estimation problems in channels with output memory. In the first part, we model the multi-level per cell (MLC) flash channel, which has been widely used as the leading technology in non-volatile solid state drive (SSD) devices during the past decade, as a channel with output memory. We show that the state of an MLC flash channel at any given time depends only on the outputs of the neighboring cells due to the existing capacitance coupling effect. Our results on MLC flash channels mainly stem from the signal processing techniques in which we provide an accurate model for MLC flash memory ; using this model we find a mathematically tractable way to formulate the write process, and finally design the optimal detector for flash memory. The results can be summarized in four different groups : i) Modeling the MLC flash as a channel with output memory. ii) Obtaining a mathematical model to characterize the iterative write process using the renewal theory framework. iii) Obtaining the optimal step size in the iterative programming despite the trade-off between programming latency and accuracy. iv) Designing optimal soft/hard detectors for MLC flash memories.

The second part of our work concentrates on parameter estimation for specific types of channels with output memory, which we call “Markov channels”, in which the channel input/output pairs form a Markov process. For this type of channel, we assume that there is no feedback and the input is a known *i.i.d* process, and we consider that both the set of contexts (channel states), and the transition probabilities (channel parameters) are unknown. We emphasize at the outset that we do *not* exclude slow mixing of the channel evolution. When a process has slow mixing property, a large number of observations is needed to explore the state space properly and the empirical properties of finite sized samples from Markov

processes not necessarily reflect stationary properties. We observe a length- n sample of the input/output pair sequence generated by applying a known *i.i.d* input process and obtaining its corresponding output from an unknown, stationary ergodic Markov channel over a finite alphabet \mathcal{A} . Using this sample, we want (i) a best approximation of the set of transition probabilities (ii) the stationary probabilities of an output string, and (iii) estimate or at least obtain heuristics of the information rate of the process. Two distinct problems that complicate estimation in this setting are long memory ; and slow mixing. Note that any consistent estimator can only converge pointwise over the class of all ergodic Markov models. But can we look at a length- n sample and identify if an estimate is likely to be accurate? Since the memory is unknown a-priori, a natural approach is to estimate a potentially coarser model with memory $k_n = O(\log n)$. As n grows, estimates get refined and this approach is consistent with the above scaling of k_n , which is also known to be essentially optimal. However, the situation is vastly different when we want the best answers possible with a length- n sample. Combining the results of universal compression with the Aldous coupling arguments, we obtain sufficient conditions on the length- n sample (even for slow mixing models) to identify when naive (i) estimates of the model parameters are accurate ; (ii) estimates related to the stationary probabilities are accurate ; we also bound the deviations of the naive estimates from the true values.

Contents

	Page
Acknowledgments	iii
Abstract	v
Contents	
1 Introduction	1
1.1 Overview	1
1.1.1 Multilevel per Cell (MLC) NAND Flash Channel	3
1.1.2 Markov Channel	4
1.2 Literature Review	5
1.2.1 Prior Work on MLC Flash Channels	5
1.2.1.1 Write Process Modeling	5
1.2.1.2 Signal Processing Techniques on MLC Flash	6
1.2.1.3 Coding Techniques for MLC Flash Memory	7
1.2.2 Prior Work on Markov Channels	8
1.2.2.1 Estimation of Markov Processes	8
1.2.2.2 Information rates of channel with memory	8
1.3 Major Contributions	9

1.3.1	Results on MLC flash channels	9
1.3.2	Results on unknown Markov Channels	11
1.4	Thesis Outline	13
2	MLC Flash Channel Model	16
2.1	MLC NAND Flash Memory Basics	17
2.1.1	Fundamental Operations	18
2.1.1.1	Erase Process	18
2.1.1.2	Write Process	18
2.1.1.3	Read Process	19
2.1.2	Degradation Sources	19
2.1.2.1	Program/Erase (P/E) Cycling Effect	20
2.1.2.2	Inter-Cell Interference (ICI)	20
2.1.3	Programming Structure	21
2.1.3.1	Even/Odd Bit-line	21
2.1.3.2	All Bit-line	22
2.2	Channel Model	23
2.2.1	1D Causal Channel with Memory	23
2.2.2	Page-Oriented Memories (2D)	26
3	Write Process Modeling	28
3.1	Notation	28
3.2	ISPP Renewal Process	29
3.3	Distribution of $\tilde{N}(w)$	32
3.3.1	Convergence	32
3.3.2	Gaussian Approximation	36
3.3.3	Distribution of $\tilde{\Gamma}(w)$	42
3.4	Inter-cell Interference (ICI)	44
3.4.1	ICI Effect on the LSB page	44

3.4.2	ICI Effect on the MSB page	46
3.4.3	ICI Analysis	48
3.5	Write Latency	49
3.5.1	LSB page Latency	50
3.5.2	MSB page programming	52
3.5.3	Confidence Interval for η	53
3.6	Step Size Design	55
3.6.1	Latency Distortion Theory	55
3.6.2	Adaptive Regulation of Step Size	57
4	Optimal Detector Design	61
4.1	Viterbi-like 1D Sequence Detection	62
4.1.1	Calculation of the Characteristic Function	64
4.1.2	FFT Implementation	65
4.1.3	Sufficient Statistics	65
4.2	Gaussian Approximation (GA) Detector	68
4.3	Extensions	71
4.3.1	Signal-Dependent Noise	71
4.3.2	Input Intersymbol Interference	72
4.3.3	2D Channels	72
4.4	Simulation Results and Discussion	73
4.5	Conclusion	79
5	Markov Channels	80
5.1	Preliminaries	80
5.1.1	Alphabet and strings	80
5.1.2	Trees	81
5.1.3	Models	81
5.2	Channel Model	83

5.3	Background Topics	84
5.3.1	Context Tree Weighting	84
5.3.2	Coupling for Markov Processes	85
5.4	Long Memory and Slow Mixing	86
5.4.1	Lower Bound on Information Rate	87
6	Estimation On Channels with Output Memory	94
6.1	dependencies dying down	94
6.2	Estimation of State Transition Probabilities	97
6.3	Estimation of State Stationary Probabilities	101
	Appendix A Gaussian Approximation of Finite Random Sum	111
	Appendix B Characteristic function $G_{Z_\ell Y_{k-\ell}}(t)$	113
	Bibliography	123

List of Figures

2.1	The stored voltage distribution of 2-bit MLC flash	17
2.2	2-bit NAND flash block organization and operation.	17
2.3	Page illustration in read and write process.	18
2.4	Thresholds for the read process relative to the conditional PMFs of each 2-bit symbol, (a) LSB page, (b) MSB page.	20
2.5	ICI aggressor cells in an “even/odd bit-line” structure, victim is on even cell.	21
2.6	ICI aggressor cells in an “all bit-line” structure.	22
2.7	NAND Flash Memory Channel Model.	23
2.8	Even/odd bit-line structure used in a NAND flash memory. (a) A cell on the even bit-line is affected by the ICI of 5 neighbors. (b) A cell on the odd bit-line is affected by 3 neighbors.	27
3.1	ISPP for a cell with starting voltage V_0 and threshold w	31
3.2	Histogram of $\tilde{N}(w) + 1$ for threshold voltage $w = 1.5V$, and $\Delta V \sim U[0.1, 0.2]$.	37
3.3	Comparison between Monte Carlo (real) and estimated (analytic) computation of $E[\tilde{N}(w)]$ and $\sigma_{\tilde{N}(w)}$ for all threshold $w \in [0, 5]$	41
3.4	The pdf of $\tilde{\Gamma}(w)$ when $\Delta V_i \sim U[a, b]$	43
3.5	Overshoot distribution after the ISPP process stops, (a) LSB page overshoot distribution, (b) MSB page overshoot distributions.	43

3.6	Programming the LSB page. (a) pdf of V_0 before writing the LSB page, (b) pdf of $\tilde{V}_{\tilde{N}(w_l)+1}(B_0^{(v)})$ after ISPP with threshold w_l and (c) Including the ICI effect on the LSB page ($V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)})$).	44
3.7	Programming the MSB page. (a) Before writing the MSB page ($V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)})$), (b) pdf of ($\tilde{V}_{\tilde{N}(w)+1}(X^{(v)})$) after ISPP for MSB page with threshold voltages w_{m_1} and w_{m_2} and (c) the ICI effect on the MSB page ($V^{(k,\ell)}(X^{(v)}, X^{(a)})$).	46
3.8	Performance of 2-bit MLC flash when $n_{PE} = 1$	49
3.9	Histogram of LSB page voltage after η_l steps (ISPP stopped).	51
3.10	The maximum number of steps (η) when the allowed undershoot error rate is $\alpha = 0.005$	56
3.11	The contour graph of $I(X, Y) \geq 1.9$ (solid line) versus step size parameters a, b and the contour of η (dashed line) versus step size parameters a, b	57
3.12	RBER and Latency versus PEC.	59
3.13	Rate and Latency versus P/E Cycles, adaptive step size $\Delta V_i \sim U[a, b]$ when $a \in [0.1, 0.6]$ and $b - a = 0.1$	60
4.1	A simple MLC flash memory system block diagram.	62
4.2	Branch metric Λ_{MAP} for cases (a) $L = 1$ and (b) $L = 2$	67
4.3	Branch metric computation using the FFT.	68
4.4	The branch metric computation module of the GA detector using FIR filters.	70
4.5	The pdf of each level's voltage for the 4-level flash memory without ICI.	74
4.6	BER comparisons for different detectors when the coupling factor strength is varying and $\sigma = 1$	75
4.7	BER comparisons for different detectors when the SNR is varying and the coupling factor strength is fixed at $s = 0.75$	76
4.8	SIQ comparisons for different detectors when the coupling factor strength is varying and $\sigma = 1$	78
4.9	SIQ comparisons for different detectors when the SNR is varying and the coupling factor strength is fixed at $s = 0.75$	78

5.1	(a) States and parameters of a Markov process in Example 4, (b) Same Markov process reparameterized to be a complete tree of depth 2. We can similarly reparameterize the process on the left with a complete tree of any depth larger than 2.	82
5.2	(a) Markov process in Example 7, (b) Aggregated model at depth 1. From Observation 1, the model on the left can be reparameterized to be a complete tree at any depth ≥ 2 . We can hence ask for its aggregation at any depth. Aggregations of the above model on the left at depths ≥ 2 will hence be the model itself.	89
6.1	(a) The conditional probabilities with which Y'_{j_1} and Y''_{j_1} have to be chosen respectively are $q_{\mathbf{u}}(\cdot)$ and $q_{\mathbf{v}}(\cdot)$. The line on the left determines the choice of Y'_{j_1} and the one on the right the choice of Y''_{j_1} . For example, if U_{j_1} is chosen uniformly in $[0,1]$, the probability of choosing $Y'_{j_1} = a_1$ is $q_{\mathbf{u}}(a_1)$. Instead of choosing Y'_{j_1} and Y''_{j_1} independently, we will reorganize the intervals in the lines so as to encourage $Y'_{j_1} = Y''_{j_1}$. (b) Reorganizing the interval $[0, 1]$ according to the described construction. Here $r(a_1) = \min \{q_{\mathbf{u}}(a_1), q_{\mathbf{v}}(a_1)\}$ and similarly for $r(a_2)$. If U_{j_1} falls in the interval corresponding to $r(a_1)$, then $(Y'_{j_1}, Y''_{j_1}) = (a_1, a_1)$. If $U_{j_1} > C(\mathcal{A})$ in this example, then $(Y'_{j_1}, Y''_{j_1}) = (a_1, a_2)$. When U_{j_1} is chosen uniformly in $[0,1]$, the probability Y'_{j_1} outputs any symbol is the same as in the picture on the left, similarly for Y'_{j_2}	106

List of Tables

4.1	Parameters of the 4-level flash memory	74
4.2	Complexity comparison for different detectors	79

1

Introduction

1.1 Overview

Interest in achieving the best data rate in digital communication channels has highly motivated researchers in information theory and communication communities to study channel capacity and devise coding and signal processing techniques to get as close as possible to the capacity limit. The initial step in any engineering project is to model the problem. In general, there are two ways to model communication channels: In the first approach, the channel is considered as a memory-less one in which the output probability distribution only depends on the current channel input; The second approach is to assume that the channel

has memory, such that each current output symbol depends statistically on the corresponding input and the current state of the channel. The state of channel is typically a function of the previous input and output sequence.

Because the information theoretic analysis and the communication design methods for memory-less channels have been widely studied since Claude Shannon made this concept mathematically precise in his seminal work [1], the communications designer and researcher will prefer, if possible, to either use a physical transmission medium that can be closely modeled with a memory-less channel or ignore the memory and model the channel as a memory-less one. However, in some applications these models are simply not possible due to the inherent complexity of the physical medium that is used to transmit the data. Note that channels with memory have mainly been used to model storage mediums in the past. For example, the partial response channel is widely used in magnetic and optical recording [2] as well as in communications over band-limited channels with inter-symbol interference (ISI). However, with the new advances in signal processing and information theory approaches to compute the capacity for particular types of channels with memory during the past two decades, it is likely to consider them in more communication practices in future.

One of the classes of channels with memory which has been extensively studied is the finite state Markov channel (FSMC). In this class, the set of states is assumed to be finite and the underlying state process is considered to be a Markov process. While FSMC has attracted most of the attention among information theory and communication communities during the past decade, there are several channels in engineering practice that can not be classified as FSMCs. We consider a subclass of channels with memory in which the state of the channel is solely characterized by the previous output of the channel. We call this subclass, “channels with output memory”. Let S_k be the state at time k . Then, the state of this channel is a sequence of previous outputs $S_{k-1} = Y_{k-L}^{k-1}$. Note that if the channel output alphabet $Y_k \in \mathcal{Y}$ be finite ($|\mathcal{Y}| < \infty$) and if the size of previous outputs, which affect the current state, be finite ($L < \infty$), then the channel with output memory becomes a finite state channel. However, in some applications of this channel, \mathcal{Y} is continuous, which makes the channel with output

memory not a FSMC. Another complication arises when the channel with output memory is unknown and its statistical characteristics need to be estimated. This thesis mainly addresses two existing complications of the channel with output memory: on case in which the state space size is either continuous and the other case when the state alphabet is finite but the channel model is unknown.

1.1.1 Multilevel per Cell (MLC) NAND Flash Channel

The MLC NAND flash Channel is the fastest growing type of technology that is widely used in non-volatile storage devices. The NAND flash memory usage is growing because of its low cost and high storage density which results from the continuous improvements in scaling technology that shrinks the sizes of CMOS transistors and MLC technology that stores more than 1 bit per cell. However, as the storage density in flash memories continues to grow, various other factors such as energy consumption, intercell interference (ICI) and program/erase (PE) endurance, continue to degrade the overall system performance.

In information theoretic point of view, a storage medium is typically modeled as a communication channel which contains a program(write) phase and a recovery(read) phase. During the program phase of MLC NAND channel, an n -bit symbol is mapped to one of 2^n non-overlapping voltage partitions, and the voltage of a cell (i.e., a CMOS transistor) is adjusted to fall within a corresponding partition. The stored data is recovered by comparing the voltage of the cell with some predetermined read thresholds during the recovery phase.

Due to the parasitic capacitance coupling effects of the neighboring cells, the change in the threshold voltage of one cell during programming (charging) affects the final voltages of all the other cells (especially those cells which were already programmed) [3]. The existing inter-cell interference (ICI) changes the output voltage of the channel, and makes the output of one cell depend on the output of neighboring cells. Thus, we model this channel as a channel with output memory. Note that the memory of an MLC flash channel depends on the values of channel outputs Y_k . Since Y_k is a continuous random variable $\mathcal{Y} = [0, 5]v$, this

channel does not belong to the well-known class of FSMC.

1.1.2 Markov Channel

The second type of channel with output memory that we consider is one that the pair of its input and output process is a Markov process. Because of this property, we call this type of channel the “Markov channel”. In this type of channel, we assume that both the input and output symbols are from a finite alphabet. The channel state depends on prior outputs alone and there is no feedback. In addition, we assume that the underlying statistical channel model is unknown. Depending on the application at hand, we consider a set \mathcal{P} of all possible channel models which are consistent with what is known about the application—say all memory-less or Markov channel models. In this work, we consider how to estimate the statistical property of this channel by considering a long sequence of inputs and their corresponding outputs. In other words, we try to describe the data (sequence of channel inputs and channel outputs) by estimating a proper channel model from \mathcal{P} .

Note that it is often possible that we can still describe the data, without knowing the underlying model other than that it belongs to \mathcal{P} , using almost as succinct a description as if we knew the underlying statistics. We are motivated by the *backplane channel*, which we will describe shortly. We ask—given n inputs and their corresponding outputs, can we estimate the transition probabilities from the input to output, as well as the stationary probabilities of various states (or set of states)?

The particular application we are motivated by arises in high speed chip-to-chip communications; and is commonly called the backplane channel [4]. Here, residual reflections between inter-chip connects form a significant source of interference. Because of parasitic capacitance, the channel is highly non-linear as well, and consequently the residual signal that determines the channel state is not a linear function of past inputs as in typical interference channels. Therefore, we consider a channel model where the output is not necessarily a linear function of the input; additionally, the channel encountered by any input symbol is determined by

the prior outputs. Such a model also yields to analytical transparency. Therefore, we begin with estimation problems in channels whose states are determined by the output.

We emphasize at the outset that we do *not* exclude slow mixing of the channel evolution. Instead, our philosophy will be: given n samples, what is the best answer we can give, if anything? In this case, we have an estimation problem where any estimator can only converge pointwise to the true values, rather than uniformly over the model class. One way to get around this impasse is to add restrictions on the model space, as was done in most prior work. However, very few such restrictions are justified in our application. Thus, we take a different approach: can we look at some characteristics of our length- n sample and say if any estimates are doing well? Based on these results, we are able to look at a data sample and identify those states of the channel that are amenable to accurate estimation from the sample. They also allow us to sometimes (depending on how the data looks) conclude that certain naive estimators of stationary probabilities or channel transition probabilities happen to be accurate, *even if the channel evolution is slow mixing*.

1.2 Literature Review

1.2.1 Prior Work on MLC Flash Channels

1.2.1.1 Write Process Modeling

A major restriction in MLC flash memories that distinguishes these memories from all other storage channels is “monotonic programming” [5], in a MLC flash memory that distinguishes it from all other storage channels. In this type of memory, the reduction of the stored voltage in a cell is a very costly operation. In fact, it is not possible to individually reduce the voltage of each cell. The reduction process is simultaneously done for a very large group of cells, called a block, and the voltage of all the cells in a block return to an original state, called the “erased state”. To deal with the monotonic programming restriction, an iterative strategy,

called incremental step pulse programming (ISPP), is used to concurrently program the target data in a very long array of cells, known as a "word-line" [6], [7]. ISPP is an iterative process and in each iteration, the voltage of the cell is increased by a small amount during an incremental step. Also, in each iteration, a verification step is executed to check whether the cell has reached the target voltage. Note that although it takes longer to program a single cell using ISPP, the overall write delay is decreased due to the concurrent programming of multiple cells, and the overcharge programming is also enormously reduced because of using small increment steps. Despite the fact that a common programming voltage is applied to all cells during ISPP, the amount of voltage increase in a given step is random due to the injection hardness property of a cell [8]. The charge injection for each cell is intrinsically different, resulting in varied voltage increments during programming of each cell in a word-line.

Several studies have been made on modeling and analyzing the write process in the past. In [9], [10], Jiang et al. primarily focused on obtaining the storage capacity and optimizing the expected programming precision for a single cell. Note that the results obtained are only valid if the individual programming of each cell was possible. If this approach is used in ISPP, it makes the verification step very complicated and time consuming. Moreover, the initial random voltage of erased state was not considered. In [8], ISPP programming of flash memory was studied, and an algorithm was developed for parallel programming of flash memory when the cell hardness information is available. In [11], a similar analysis was made for the case that the increments are exponentially distributed. Although the results are helpful in understanding the storage capacity of parallel programming, they disregard the existing erase state randomness which makes them inapplicable in practice.

1.2.1.2 Signal Processing Techniques on MLC Flash

As discussed, the precise flash memory channel modeling and the knowledge of the exact statistics of the model is required to attain higher information rates and derive proper cod-

ing schemes to achieve those performance limits. Existing literature, such as [12], [13], [14], mainly focused on modeling the MLC flash channel as a memory-less one and derived so-called soft (decision) information by using uniform and/or non-uniform channel output quantization methods. In these works, the memory of the channel, which exists because of the ICI, is disregarded. To compensate for ICI, two similar methods, one called *post-compensation* in [15] and the other called *coupling canceller* in [16], were presented to subtract estimates of ICI from the noisy observation of the channel output (i.e., the sensed voltage of each cell). These two methods can be considered as (hard) detection schemes for the MLC flash memory. To the best of our knowledge, there exists no open literature providing an exact soft-output detector without channel output quantization in MLC NAND flash memories.

1.2.1.3 Coding Techniques for MLC Flash Memory

To guarantee the reliability, on-chip error correcting techniques are widely employed in MLC NAND flash memory products [17]. However, as the market continues to demand higher densities and more levels per cell, simple error-correcting codes (ECCs) (for example, BCH codes with hard decoding) cease to be adequate. Hence, stronger ECCs with soft (decision) decoding, for example, low-density-parity-check (LDPC) codes, are desired in the next-generation flash memories.

Recently, some effort has been exerted on using write-one memory (WOM) code for MLC flash memories. Because of monotonic restriction, flash memories impose similar constraints in which the level of each cell can only increase, and can be decreased only if its entire block is first erased. Thus, a WOM-code can be applied in flash memories to enable additional writes without first having to erase the block. WOM codes in flash memories were investigated both theoretically with respect to the number of block erasures, and practically by simulations; see e.g. [8], [18].

1.2.2 Prior Work on Markov Channels

1.2.2.1 Estimation of Markov Processes

Estimation for Markov processes has been extensively studied and falls into three major categories (i) consistency of estimators *e.g.*, [19–22], (ii) guarantees on estimates that hold eventually almost surely *e.g.*, [23, 24], and (iii) guarantees that hold for all sample sizes but which depend on the model parameters *e.g.*, [25–28]. As mentioned earlier, performance of any estimator can not be bounded uniformly over all Markov models, something reflected in the line (iii) of research and in our work. The crucial distinction is that we can gauge from the observed sample *if* our estimator is doing well, something that does not hold in (iii). The list is not exhaustive, rather work closest to the approaches we take.

In [25, 26, 28], exponential upper bounds on probability of incorrect estimation of (i) conditional and stationary probabilities and (ii) the underlying context tree, are provided for variants of Rissanen’s algorithm *context* and penalized maximum likelihood estimator. However, the introduced deviation bounds depend on the parameters (*e.g.*, depth of the tree, stationary probabilities) of underlying process which are unknown a-priori in practice. In [27], the problem of estimating a stationary ergodic processes by finite memory Markov processes based on an n -length sample of the process is addressed. A measure of distance between the true process and its estimation is introduced and a convergence rate with respect to that measure is provided. However, the deviation bound holds only when the infimum of conditional probabilities of symbols given the pasts are bounded away from zero.

1.2.2.2 Information rates of channel with memory

During the past decade, new analytical bounds and simulation-based methods have been introduced that now permit the calculation of symmetric information rate corresponding to the maximum rate achievable with i.i.d. inputs as well as a sequence of lower bounds on the channel capacity. The idea underlying these methods is to estimate the channel output

entropy by computing the joint probability of a long channel output realization and then invoking the Shannon-McMillan-Breimann theorem. [29]

The computation of the capacity of channels with memory has long been an open problem. The past efforts mainly focus on the class of finite state channels. To summarize, researchers have considered (i) computing information rate, (ii) finding lower and upper bounds for the information rates, and (iii) capacity achieving distributions. A comprehensive review is available in [30–33]. In particular, for ISI channels with an average power constraint and Gilbert-Elliot-type channels, the capacity is already known. In addition, the capacity for output memory channels with an additive noise channel (independent of input) was computed in [34]. Note that, in this line of work, the channel model and its properties was assumed to be known.

1.3 Major Contributions

1.3.1 Results on MLC flash channels

Our results on MLC flash channels mainly belong to modeling and signal processing techniques which can be summarized in four different groups: i) Modeling the MLC flash as a channel with output memory; ii) Obtaining a mathematical model to characterize the iterative write process using the renewal theory framework; iii) Obtaining the optimal step size in ISPP despite the trade-off between programming latency and programming accuracy; iv) Designing optimal soft/hard detectors for MLC flash memory. We first provide the MLC flash channel models, including the one-dimensional (1D) model with *causal output memory* and the two-dimensional (2D) anti-causal model of the MLC flash memory. The obtained models provide the proper insight needed to guide the derivation of novel low-complexity coding scheme and optimal detectors. To attain a precise model for MLC flash channels, we first need to deal with the monotonic restriction in the write process. The monotonicity in the write process, whereby the programmed voltage can not be decreased, is considered

to be a major restriction in MLC flash memory. To solve this issue, an iterative approach known as incremental step pulse programming (ISPP) is used to concurrently program many cells in small steps. We introduce a mathematical model for ISPP using the *renewal theory* framework. A renewal process is a stochastic counting process that models the number of steps required for a random sum to pass a specific threshold [35], [36]. We show that it is possible to model ISPP in flash as a renewal process with a random starting point; We call it “ISPP renewal process”. We show that the ISPP renewal process is a renewal process whose starting point is random. To the best of our knowledge, this is the first attempt to describe the relationship between ISPP in flash and renewal theory. We first derive the connection between the *ISPP renewal process* in the finite threshold regime and the asymptotic classical renewal process. Next, we obtain a close approximation for the probability mass function (PMF) of the number of steps required in the ISPP renewal process, and a linear approximation for its mean and variance. We also bound the maximal error between the true distribution and our approximation which shows they are aligned in realistic scenarios. Finally, we use the results from renewal theory to obtain the distribution of the cell voltages after passing the threshold, called the “overshoot”, and then analyze the resulting ICI using the obtained overshoot distribution.

After having determined the statistics of the write process, our next goal is to use these statistics to describe/optimize trade-offs between speed and accuracy as well as device lifetime. Note that the ISPP renewal process is used in non-volatile memories such as MLC flash and phase-change memory (PCM) to achieve high write speeds. However, the write process is still considered to be the most latent when compared to the other components such as the read process and the ECC. The literature has already acknowledged the high latency of write process and the existing trade-off between write latency and write accuracy. In fact, the read process is an order of magnitude faster than the write process in non-volatile memories [37]. The main reason for write process delay is the large number of iterations required in a typical ISPP process. In [38], the authors propose a procedure, called “approximate storage”, in which they intentionally reduce the number of iterations to provide a large efficiency gain in

terms of latency and energy for the write process. Note that the reduction of ISPP iterations causes a small storage error, called “quality loss”, which can be corrected using the on-chip ECC module. Simulation results show that multi-level phase-change memory cells can be programmed 1.7 times faster when a 10% quality loss is allowed [38]. Using the obtained distribution for the number of steps required in ISPP, we study the relationship between the number of steps allowed in ISPP process and the quality loss, and derive an inequality that needs to be hold in order to guarantee a pre-specified allowed ISPP undershoot error. We also conduct a study that optimizes the write latency provided that the information rate is larger than a pre-specified rate. Finally, using the analytic and semi-analytic tools developed here, we devise an adaptive approach to choose the step size required to improve the overall lifetime of a flash device.

Next, motivated by [15], we focus on designing a detector for MLC flash memories that can potentially improve the hard decision bit-error-rate. In addition, we would like to improve the soft decision quality of the detector if possible. Furthermore, designing the optimal soft and hard detectors provides a benchmark to which all other (sub-optimal) detectors would be compared to. The optimal detector design also helps to derive a closed form expression for the optimal decision making strategy in order to gain insight (such as sufficient statistics) and understand the interplay between channel parameters. Finally, we can use the attained insights in deriving a novel low-complexity suboptimal detector. Our simulation results show that the hard-output bit error rate (BER) performance matches some previously known detectors, but that the soft-output detector outperforms previously known detectors by 0.35 dB.

1.3.2 Results on unknown Markov Channels

In this part, given a realization of a Markov process, we consider a coarser model and provide deviation bounds for sequences which have occurred frequently enough in the sample. In contrast to prior literature, while we make an assumption justified by the physical model—

that dependencies die down in the Markov model class we consider—our bounds can be calculated using only parameters which are well-approximated from data. In particular, we do not assume neither a-prior knowledge on the depth of context tree of the process nor the conditional probabilities given the pasts are bounded uniformly away from zero. Motivated by the operation and estimation in backplane channels, we consider the following abstraction. We have n *i.i.d* input symbols and their corresponding outputs, and the input/output pairs evolve as a Markov process. There is no feedback and the input is a known *i.i.d* process. The channel input/output pairs then form a Markov process $p_{\mathcal{T},q}$ where both the set of contexts \mathcal{T} , and the transition probabilities $q(\mathcal{T})$ (or equivalently the channel parameters $\Theta_{\mathcal{T}}$) are unknown. Using this sample, we want (i) to approximate as best as possible, the parameter set $\Theta_{\mathcal{T}}$ (ii) the stationary probabilities $\mu(\mathbf{s})$ of observing an output string $\mathbf{s} \in \mathcal{T}$, and (iii) estimate or at least obtain heuristics of the information rate of the process.

Two distinct problems complicate estimation of $\Theta_{\mathcal{T}}$ and the stationary probabilities. First is the issue that the memory may be too long to handle—in fact, if the source has long enough memory it may not be possible, with n samples, to distinguish the source even from a memory-less source. Secondly, even if the source has only one bit of memory, it may be arbitrarily slow mixing. When a process has slow mixing property, a large number of observations is needed before exploring the state space of the process properly and the empirical properties of finite sized samples from Markov processes need not reflect stationary properties. In other words, no matter what n is, there will be sources against which our estimates perform very poorly.

A natural way to deal with \mathcal{T} being unknown is to try estimating a potentially coarser approximation with depth k_n to the true model, for some known k_n . With the benefit of hindsight, we take $k_n = O(\log n)$ which reflects the well known conditions for consistency of estimation of Markov processes in [21]. The input/output pairs obtained by the coarser channel model will be a stationary Markov process, and is called the *aggregation* model. The aggregation matches all joint distributions of $p_{\mathcal{T},q}$ involving variables that are less than k_n apart from each other. We show that the information rate corresponding to the aggregation

model is a lower bound on the information rate of the true channel input/output process $p_{\mathcal{T},q}$. While it may not be possible to directly use this result, we develop of the notion of a *partial information rate*, a useful heuristic when the source has not mixed. Moreover, the obtained lower bound also motivates the estimation questions that form our main results.

Given our physical motivation, we assume that the influence of prior outputs dies down as we look further into the past. We formalize this notion as “dependency die down” property. With this assumption, we obtain a set of *good states* or *good* strings of channel outputs from the sample, and combining results in universal compression with Aldous coupling arguments, we obtain sufficient conditions on the length- n sample (even for slow mixing models) to identify when naive (i) estimates of the model parameters and (ii) estimates related to the stationary probabilities are accurate for the set of good states; and also bound the deviations of the naive estimates from true values.

1.4 Thesis Outline

This dissertation consists of an introduction and five self-contained chapters. The chapters cover two important applications of channels with output memory. In chapters 2, 3 and 4 we focus on modeling MLC Flash and find the appropriate mathematical framework needed to accurately analyze iterative programming in the write process. Finally, we design the hard/soft detectors for this channel model. Chapters 5 and 6 are devoted to the Markov channel estimation problem. We introduce the Markov channels and provide the required background information, and then we provide results in chapter 6 for estimating transition probabilities and stationary probabilities for this type of channels.

To be more specific, Chapter 2 consist of two Sections; Section 2.1 elaborates on the major processes that occur in which a symbol is stored in a flash cell, recovered from the same cell, and removed. We call these processes the write process, read process and erase process, respectively. The erase process is needed due to the existing monotonicity restriction in

which the voltage of the cell cannot decrease during the write process. Next we briefly mention the major degradation sources, and two important structures that are used in flash based non-volatile memories to speed up the write and read processes. Then, we model the MLC flash channel as a channel with output memory in Section 2.2. The memory is imposed because of the ICI that changes the voltages of the neighboring cells during write process.

In Chapter 3, we first introduce the *renewal theory* in Section 3.2 and define a new process, called ISPP renewal process, that models the ISPP process and derive the connection between the “ISPP renewal process” in the finite threshold regime and the asymptotic classical renewal process. After having determined the statistics of the write process in section 3.3, we then use these statistics to analyze the ICI effect in Section 3.4. Using the obtained distribution for the number of steps required in ISPP, we study the relationship between the number of allowed steps in ISPP and the quality loss, and in Section 3.5 derive an inequality that needs to be hold in order to guarantee a pre-specified allowed ISPP undershoot error. Finally, using the analytic and semi-analytic tools developed in this Chapter, we devise an adaptive approach to design the step size required in order to improve the flash life-time, and describe/optimize trade-offs between speed, accuracy and device lifetime in Section 3.6.

In Chapter 4, we first present a mathematically tractable Viterbi-like *maximum a posteriori* (MAP) sequence detector for the 1D causal model with output memory in Section. 4.1. Secondly, we showcase the exact statistics of the channel model necessary for implementing the MAP detector by using the fast Fourier transform (FFT). Thirdly, we introduce a simplified Gaussian approximation (GA) sequence detector which comes at the expense of reduced performance; Both the MAP detector and the GA detector can be employed in a 2D anti-causal flash memory channel. In Section. 4.3, we extend the channel model and detector design to more general scenarios including those with signal-dependent noise, input intersymbol interference, and 2D Markov channel inputs. Finally, in Section. 4.4, we utilize simulation results to show that the MAP detector outperforms the literature’s existing detectors.

In Chapter 5, we first formulate the statistical properties of a Markov channel in Section 5.2. In Section 5.3, we provide the mathematical concepts that is used in the proof of our main result in next Chapter. In Section 5.4, we discuss two distinct difficulties encountered in estimating Markov processes: the long memory Markov process and the Slow mixing property of channels. Then, by introducing a coarser channel model that matches the input and output sequence in Section 5.4.1, we prove that the information rate corresponding to the coarser channel model is a lower bound on the information rate of the true Markov channel.

Chapter 6 consists of three different sections. In Section 6.1, we formalize the notion of dependency die down. Then, combining the results on universal compression and Aldous' coupling arguments, we obtain sufficient conditions (even for slow mixing models) to identify when naive (i) estimates of the channel parameters exist and we can obtain their deviation bounds (Section 6.2) and (ii) estimates related to the stationary probabilities of the channel states are accurate, and we can bound their deviations from their true values (Section 6.3).

2

MLC Flash Channel Model

In section 2.1, We describe some basics about MLC NAND flash memories and explain the major processes and structures available in using these type of memories. Then, in section 2.2 we provide a stochastic model that includes all the important features of a multilevel NAND flash memory. We use a channel model that is mathematically almost identical to the channel model in [12], but we do make certain small changes (as described below) to achieve complete mathematical tractability of the model. These changes only marginally effect the model and the detector performance.

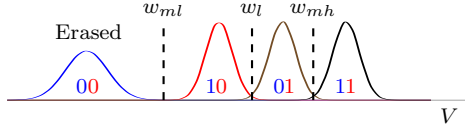


Figure 2.1: The stored voltage distribution of 2-bit MLC flash

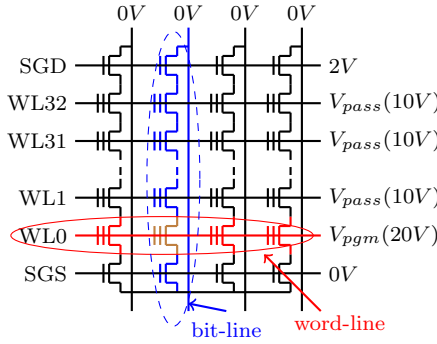


Figure 2.2: 2-bit NAND flash block organization and operation.

2.1 MLC NAND Flash Memory Basics

A NAND flash memory is comprised of arrays of floating gate transistors (cells). Each MLC cell can store multiple bits (usually 2-4 bits). By injecting a specific writing charge on the floating gate of a cell, the voltage of the cell is increased to one of the multiple valid partitions, and the symbol is stored. Figure 2.1 shows the bit mapping to the stored voltage of a 2-bit MLC flash.

In order to increase the speed of programming (writing) and recovering (reading) data in flash memory chips, all of the memory cells are hierarchically organized in blocks and arrays (called word-lines). Each flash memory chip usually contains thousands of blocks, and each block is composed of 32 to 128 word-lines. Each word-line contains 2K to 64K cells which form a very long array. Figure 2.2 shows a 2-bit NAND flash block organization.

Note that the smallest unit that can be simultaneously accessed for programming (writing) or reading is a page. Figure 2.3 illustrates the page definition for a 2-bit NAND flash memory in the write/read process- namely the least significant bit (LSB) page and the most significant bit (MSB) page.

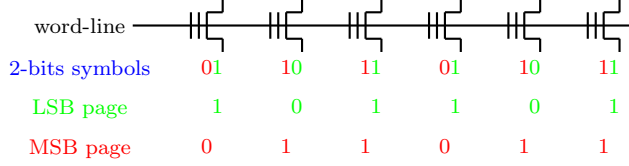


Figure 2.3: Page illustration in read and write process.

2.1.1 Fundamental Operations

MLC flash memories basically support three major processes which we briefly describe as follows:

2.1.1.1 Erase Process

In this process, the charged voltages stored in the floating gates of all the cells in a block are tunneled out through the Fowler-Nordheim mechanism [39]. After the erase process, the voltages of all the cells in a block will fall into the erased state (e.g. state “00” in Figure 2.1). The smallest unit that can be erased is a block. This limitation makes the cell programming a one-way operation because it is not possible to erase a specific cell separately from other cells in a block. It is necessary to erase a memory cell before being able to program it. The distribution of the cell voltage in the erased state, denoted by V_0 , tends to be Gaussian with mean μ_0 and variance σ_0^2 ($V_0 \sim \mathcal{N}(\mu_0, \sigma_0^2)$)[40].

2.1.1.2 Write Process

In each write stage, a page is programmed by applying a high voltage to its corresponding word-line. Due to the monotonic programming restriction, it is very important to accurately program each page such that the voltages of all cells in the page fall in their intended voltage ranges. Thus, the ISPP approach is used to program each page. This approach provides a series of verification steps right after each short increment programming step [5].

ISPP is an iterative technique which executes a verification step after each incremental

voltage-boost step for all the cells in the page. Figure 3.1 shows the voltage changes during the ISPP steps of programming. Let w be the threshold voltage for programming a page. Those cells with the target symbol $B = 1$ (by convention) should pass the threshold voltage w . In the i -th step of programming, the cell voltage \tilde{V}_{i-1} is increased by ΔV_i for each cell whose voltage has not reached the target value w . After the i -th programming pulse, the voltages of all the cells are compared with the target threshold voltage w during the verification phase. In any particular cell, the ISPP is stopped when \tilde{V}_i of that cell is greater than the desired threshold voltage w . The program pulse ΔV_i is modeled as a random variable that tends to be uniformly distributed [5].

2.1.1.3 Read Process

During the read process, the stored symbols of all cells in the same page are read concurrently. The read process is done by comparing the voltage of a cell with some predetermined thresholds $r \in \{r_{m_1}, r_l, r_{m_2}\}$, called “read thresholds”. Figure 2.4a shows reading the LSB page in a 2-bit MLC flash memory and Figure 2.4b shows reading the MSB page. Note that the write thresholds and read thresholds are not necessarily equal. In general, the optimal read thresholds (r) must be obtained by using either a trellis-based Viterbi detector (Maximum Likelihood (ML) detection) or a trellis-based BCJR detector (Maximum a Posteriori (MAP)). In practice, since the input sequence usually is considered to be i.i.d, optimal detection can be executed on a symbol-by-symbol basis without implementing a trellis [40], and simple thresholds suffice.

2.1.2 Degradation Sources

There are two major sources of performance degradation that affect the threshold voltage in each memory cell during the write process:

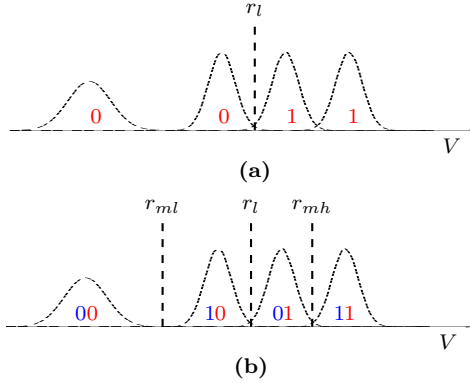


Figure 2.4: Thresholds for the read process relative to the conditional PMFs of each 2-bit symbol, (a) LSB page, (b) MSB page.

2.1.2.1 Program/Erase (P/E) Cycling Effect

The P/E cycling distorts the final threshold voltage of a transistor due to the trapping and detrapping ability of the interface at the transistor gate, which leads to higher fluctuation of the final voltage of the cell. Note that P/E cycling has a higher effect on the voltages of those cells whose target state is the erased state (state “00” in 2-bit case). Moreover, as the flash gets older (i.e., the number of P/E processes in a flash increases), the variance of ΔV_i during ISPP gets larger.

2.1.2.2 Inter-Cell Interference (ICI)

Due to the parasitic capacitance coupling effects of the neighboring cells, the change in the threshold voltage of one cell during programming (charging) affects the final voltages of all the other cells (especially those cells that were already programmed) [3]. ICI is a degradation source that grows with density. As cells are packed closer to each other, the influence of threshold voltages from neighboring cells increases. Let (k, ℓ) denote the location of a memory cell; the cell is located at the k -th word-line and the ℓ -th bit-line. We call the set of aggressors, $\mathcal{A}_{(k, \ell)}$, the set of indices of those neighboring cells that are programmed after the victim cell (k, ℓ) , and their voltage change affects the victim cell’s stored voltage. For a cell in location $(k', \ell') \in \mathcal{A}_{(k, \ell)}$, let $c_{(k'-k, \ell'-\ell)}$ be the capacitance coupling coefficient.

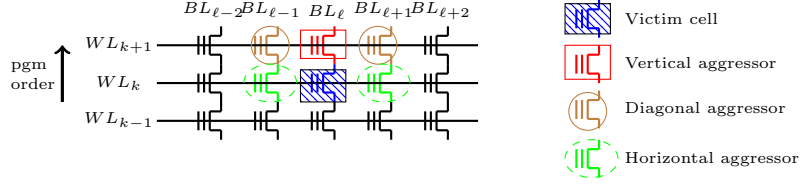


Figure 2.5: ICI aggressor cells in an “even/odd bit-line” structure, victim is on even cell.

Then the ICI effect on (k, ℓ) due to programming (k', ℓ') is obtained as

$$C_{(k'-k, \ell'-\ell)} \cdot \Delta V^{(k', \ell')}.$$

2.1.3 Programming Structure

NAND flash memories use either even/odd bit-line structure or all-bit-line structure.

2.1.3.1 Even/Odd Bit-line

The first structure (called “even/odd bit-line structure”) to program (write) the data is to separate all the cells into those at even bit-lines and those at odd bit-lines. During the process of programming, the cells at even bit-lines along a word-line are written at the same time instant, and then the cells at odd bit-lines along this word-line are written at the next time instant. Each pair of even/odd bit-lines can share peripheral circuits such as sense amplifier and buffer, leading to less silicon cost of peripheral circuits.

For the even/odd bit-line structure, cells in even bit-lines along a given word-line, referred to as even cells, are programmed first at one time instant, and then cells in odd bit-lines, referred to as odd cells, are programmed at a later time instant. Hence, the ICI neighborhoods are also dependent on whether an even cell or an odd cell is programmed in the programming cycle. Let (k, ℓ) denote the location of a memory cell, which means that the cell is located at the k -th word-line and the ℓ -th bit-line. We denote by $\mathcal{O}_{(k, \ell)}$ the indices of the anticausal neighborhood for the *odd* cell and by $\mathcal{E}_{(k, \ell)}$ the indices of the anticausal ICI neighborhood

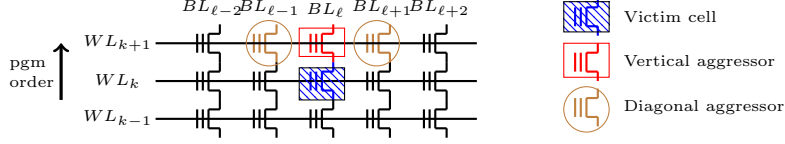


Figure 2.6: ICI aggressor cells in an “all bit-line” structure.

for the *even* cell (illustrated in Figure 2.5). That is,

$$\mathcal{O}_{(k,\ell)} \triangleq \{(k+1, \ell-1), (k+1, \ell), (k+1, \ell+1)\} \quad (2.1)$$

and

$$\mathcal{E}_{(k,\ell)} \triangleq \{(k, \ell-1), (k, \ell+1)\} \cup \mathcal{O}_{(k,\ell)}. \quad (2.2)$$

2.1.3.2 All Bit-line

The other architecture (called “all-bit-line structure”) to program the data is to write all cells along a word-line simultaneously without distinguishing between even and odd cells. The all bit-line structure has the advantage that the Inter-Cell Interference is lower than even/odd bit-line structure.

As shown in Figure 2.6, the set of aggressors $\mathcal{A}_{(k,\ell)}$ in an all-bit-line structure for cell (k, ℓ) is typically the set of nearest neighbors that get programmed after the victim cell:

$$\mathcal{A}_{(k,\ell)} = \{(k+1, \ell-1), (k+1, \ell), (k+1, \ell+1)\}.$$

Since the diagonal aggressors (neighbors) are further apart than the vertical ones, the diagonal ICI effect is negligible compared to the vertical ICI (i.e., $c_{(1,0)} \gg c_{(1,1)} = c_{(1,-1)}$) [40]. Thus, for simplicity we ignore the effect of aggressors in the diagonal positions on the victim cell and only consider the vertical cell $(k+1, \ell)$ as the major aggressor. i.e., $\mathcal{A}_{(k,\ell)} = \{(k+1, \ell)\}$ or equivalently, $c_{(1,1)} = c_{(1,-1)} = 0$.

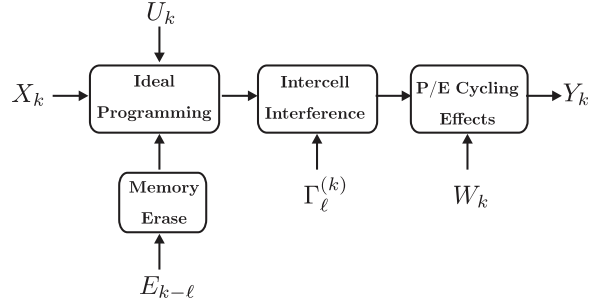


Figure 2.7: NAND Flash Memory Channel Model.

2.2 Channel Model

We start our exposition of the channel model by first presenting a simple *one-dimensional (1D) causal* channel model. Note that an actual flash memory channel is not one-dimensional, but rather two-dimensional (2D) because the channel is a page-oriented channel. Also, note that the actual flash memory channel is not causal, but rather anti-causal, because ICI is an anti-causal effect as only those cells that are programmed *after* the victim cell actually affect the victim cell. Nonetheless, the 1D causal channel model is very useful because it allows us to formulate the optimal detector in the universally accepted manner, namely for a 1D *causal* channel. Extrapolating the detector to cover 2D anti-causal channels is then fairly straightforward.

2.2.1 1D Causal Channel with Memory

Let $k \in \mathbb{Z}$ stand for discrete time. The channel input, denoted by X_k , is the intended stored voltage amount in the k -th cell. The channel output denoted by Y_k is the channel output voltage corresponding to the input value X_k . According to MLC technology, we assume that the channel input random variable X_k takes value from a finite alphabet $\mathcal{X} = \{v_0, v_1, \dots, v_{m-1}\}$ with $|\mathcal{X}| = m < \infty$. We assume that the channel input and channel

output have the relation ¹:

$$Y_k = X_k + \sum_{\ell=1}^L \Gamma_{\ell}^{(k)} (Y_{k-\ell} - E_{k-\ell}) + W_k + U_k \quad (2.3)$$

where ²,

- (a) E_k is the erase-state noise at the k -th cell, modeled as a Gaussian random variable with mean μ_e and variance σ_e^2 , that is, $E_k \sim \mathcal{N}(\mu_e, \sigma_e^2)$.
- (b) $\Gamma_{\ell}^{(k)}$ is a fading-like coefficient that models *causal* ICI from the $(k - \ell)$ -th cell towards the k -th cell (victim cell). We assume $\Gamma_{\ell}^{(k)}$ also to be a Gaussian random variable, $\Gamma_{\ell}^{(k)} \sim \mathcal{N}(\gamma_{\ell}, g_{\ell})$.
- (c) L is the output memory, which implies that the current channel output Y_k is affected by its L neighbors $Y_{k-1}, Y_{k-2}, \dots, Y_{k-L}$.
- (d) U_k denotes the programming noise, resulting from using the ISPP method of programming the k -th cell of a certain word-line. This noise is modeled as a zero mean uniform random variable with width Δ , that is, $U_k \sim \mathcal{U}(-\Delta/2, \Delta/2)$.
- (e) W_k is observation noise due to the PE cycling, and is distributed as a zero mean Gaussian random variable with variance σ_w^2 , that is, $W_k \sim \mathcal{N}(0, \sigma_w^2)$.

We assume that all random variables $\Gamma_{\ell}^{(k)}, E_{k-\ell}, W_k$ and U_k are mutually independent for all k and all ℓ .

Remark We assume that the PE cycling/aging effect is incorporated into the model through the knowledge of σ_w^2 . That is, σ_w^2 may depend on the device age. \square

1. The model in (2.3) is an autoregressive model, but only if the channel input X_k is memoryless (uncorrelated). However, if the channel input has memory (for example, if the channel input is a Markov process of order $M > 0$), this is not an autoregressive model.

2. The channel model in (2.3) does not take the same form as equation (1) in [15], however, they are actually identical. In Dong et al. [15], the authors chose to write the channel model in terms of the “voltage shift”. In our equation (2.3), the “voltage shift” is equal to the difference $Y_{k-\ell} - E_{k-\ell}$, but instead of calling explicit attention to the “voltage shift”, we call explicit attention to the channel outputs $Y_k, Y_{k-1}, \dots, Y_{k-L}$. We think that, for the purpose of detector design, it is much better to explicitly call attention to the channel outputs because the detector specifically operates on the channel outputs.

The major differences in the properties of random variables of this 1D channel model and the proposed channel model in [12, 15] are as follows:

- (a) We assume that $\Gamma_\ell^{(k)}$ are Gaussian, whereas Dong *et al.* [12, 15] assumed the corresponding variables to be distributed as truncated Gaussians. Note that these two distributions almost have the same behavior in the common support range, but the Gaussian distribution is easier to track analytically.
- (b) We assume that the observation noise $U_k + W_k$ is a mixture of uniform and Gaussian noises. The pdf of this mixture is actually very similar (though not identical) to the pdf shown in [15, Fig. 4].

Remark The channel model (2.3) is actually a general 1D causal ICI channel which belongs to the class of channels with memory. Note that the memory of an ICI channel depends on the values of channel outputs Y_k . Although the output process Y_k in this channel (for a stationary i.i.d. input process X_k) is a Markov process, this channel does not belong to well-know class of Markov finite-state channels [41, 42]. \square

Since the ICI channel has memory, it is obvious that a capacity-achieving process X_k may also need to have memory. For this reason, we assume that the process X_k is a Markov process³ of order M . That is,

$$P_{X_k|X^{k-1}}(x_k|x^{k-1}) = P_{X_k|X_{k-M}^{k-1}}(x_k|x_{k-M}^{k-1}). \quad (2.4)$$

The optimal detector is either a trellis-based Viterbi detector [44] (if we are interested in maximum likelihood (ML) or MAP sequence detection) or a trellis-based BCJR detector [45] (if we are interested in ML or MAP symbol detection). In both sequence or symbol detection cases, the trellis state at time k is defined as (X_{k-M+1}^k) . Therefore, the number of trellis states in one trellis section is $|\mathcal{X}|^M = m^M$.

3. This work is not concerned with finding the actual optimal Markov-memory input process X_k ; The interested reader is referred to [43].

Although it seems reasonable to use Markov input process to mitigate the ICI influence in MLC flash memory channels, it is a very difficult task to find the best Markov process. In this difficult optimization problem, one should not only find the optimal number of voltage levels (m) and the best values of these levels $\{v_0, v_1, \dots, v_{m-1}\}$, but also find the Markov distribution that maximizes the channel mutual information rate [46]. To the best of our knowledge, this problem is still open and there exists no literature to address the aforementioned problem. Note that finding the best Markov input process is beyond the scope of this thesis. However, the results in this work are still valid even for the case that the input distribution is Markov.

If X_k is a memoryless (i.i.d) process (i.e., if $M = 0$), even though Y_k has memory, a trellis-based detector is not needed (because $|\mathcal{X}|^M = 1$), and optimal detection (both soft and hard) can be executed on a symbol-by-symbol basis. That is, we can make the optimal decision on the random variable X_k without postulating neighboring channel input realizations x_{k-M}^{k-1} at all, and by considering only the received channel output realizations y_{k-L}^k .

2.2.2 Page-Oriented Memories (2D)

In two-dimensional (2D) page-oriented memories with ICI, a single (victim) cell is only affected by a finite anticausal neighborhood of near-by cells (which are programmed *after* the victim cell). As discussed in section 2.1.3, either the even/odd bit-line or the all-bit-line structure is used in most modern NAND flash memories. Since the amount of interference in the even/odd bit-line structure is higher than that in the all-bit-line structure, we consider only the even/odd bit-line structure using the full-sequence programming strategy. Note that our results can also be applied to the all-bit-line structure.

For the even/odd bit-line structure, cells in even bit-lines along a given word-line, referred to as even cells, are programmed first at one time instant, and then cells in odd bit-lines, referred to as odd cells, are programmed at a later time instant. Hence, the ICI neighborhoods are also dependent on whether an even cell or an odd cell is programmed in the programming

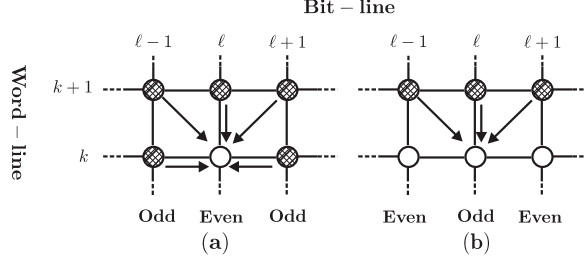


Figure 2.8: Even/odd bit-line structure used in a NAND flash memory. (a) A cell on the even bit-line is affected by the ICI of 5 neighbors. (b) A cell on the odd bit-line is affected by 3 neighbors.

cycle. Let (k, ℓ) denote the location of a memory cell, which means that the cell is located at the k -th word-line and the ℓ -th bit-line. We denote by $\mathcal{O}_{(k, \ell)}$ the indices of the anticausal neighborhood for the *odd* cell and by $\mathcal{E}_{(k, \ell)}$ the indices of the anticausal ICI neighborhood for the *even* cell (illustrated in Figure 2.8). That is,

$$\mathcal{O}_{(k, \ell)} \triangleq \{(k+1, \ell-1), (k+1, \ell), (k+1, \ell+1)\} \quad (2.5)$$

and

$$\mathcal{E}_{(k, \ell)} \triangleq \{(k, \ell-1), (k, \ell+1)\} \cup \mathcal{O}_{(k, \ell)}. \quad (2.6)$$

Therefore, we introduce an appropriate ICI channel model for the (k, ℓ) -th victim cell as

$$Y_{(k, \ell)} = X_{(k, \ell)} + \sum_{\substack{(a, b) \in \\ \mathcal{S}_{(k, \ell)}}} \Gamma_{(a, b)}^{(k, \ell)} (Y_{(a, b)} - E_{(a, b)}) + W_{(k, \ell)} + U_{(k, \ell)} \quad (2.7)$$

where, the set $\mathcal{S}_{(k, \ell)}$ is either $\mathcal{O}_{(k, \ell)}$ or $\mathcal{E}_{(k, \ell)}$ depending on the location of the cell. If $X_{(k, \ell)}$ is a process with 2D memory, an optimal detector is not known (a 2D equivalent of a Viterbi detector is not available), and must be appropriately approximated using adequate (possibly interleaved) one dimensional (1D) detectors [47]. If $X_{(k, \ell)}$ is an i.i.d. process, optimal detection (both soft and hard) can be executed on a symbol-by-symbol basis.

3

Write Process Modeling

3.1 Notation

Our notation is geared towards a 2-bit MLC flash memory with an all-bit-line structure. All random variables are denoted by capital letters. Let $X = B_m B_l$ denote the 2-bit channel input of a cell, where B_l and B_m are binary random variables that represent the LSB and MSB of the input X , respectively. Let Y be the 2-bit channel output that is obtained after the read process. X and Y take value from the set $\mathcal{S} = \{00, 10, 01, 11\}$. We use superscript (k, ℓ) for the binary symbol B and the input X to associate them with cell (k, ℓ) (e.g. $X^{(k, \ell)} = B_m^{(k, \ell)} B_l^{(k, \ell)}$). For notational simplicity, we remove the superscript (k, ℓ)

whenever the location of the cell is unambiguous, and similarly we ignore the subscripts l and m when the significance of the bit is clear. For sake of simplicity, c_v denotes the vertical capacitance coupling coefficient ($c_v = c_{(1,0)}$). All the write threshold voltages are denoted by $w \in \{w_{ml}, w_l, w_{mh}\}$, where w_l denotes the write threshold for LSB, and w_{ml} and w_{mh} denote the low and high write thresholds for MSB, respectively (Figure 2.1). Similarly the read thresholds are denoted by r , where $r \in \{r_{ml}, r_l, r_{mh}\}$. By convention, if $B = 1$ for a cell in a page, the program voltage of the cell needs to pass the threshold w . Due to the equality of the distances between vertical aggressors and the victim cells, the vertical ICI coupling coefficient, denoted by c_v , is assumed to be equal for all cells. Finally, we denote by n_{PE} the P/E cycling number of the flash.

3.2 ISPP Renewal Process

In this section we first model the ISPP page programming using the known concepts in renewal theory[35], and explain the relationship between the ISPP process and the classical renewal process. Note that ISPP only increases the voltage of those cells whose target bit is $B = 1$. The cells with $B = 0$ are already in the final state and their voltages do not need to change during the ISPP renewal process. Let V_j be the sum of voltage increments up to the j -th step. i.e.,

$$V_j = \sum_{i=1}^j \Delta V_i, \quad (3.1)$$

where ΔV_i denotes the increment during the i -th step of ISPP. Let \tilde{V}_j denote the cell voltage after programming step j . Given that the process starts in the erased state V_0 , the cell voltage \tilde{V}_j is

$$\tilde{V}_j = V_0 + V_j. \quad (3.2)$$

Similar to [5], we assume that

$$\Delta V_i \sim U[a, b], \quad (3.3)$$

i.e., ΔV_i is uniformly distributed between a and b ($0 < a < b$). We denote the CDF of increments ΔV_i by $F_{\Delta V}(v)$, and define $\mu_\Delta = E[\Delta V_i]$ and $\sigma_\Delta^2 = \text{Var}(\Delta V_i)$.

Remark Since the write voltage applied to the control gate (20V [39]) is much higher than the stored voltage, it is reasonable to assume that the increment ΔV_i is independent of ΔV_j for each $i, j \in \mathbb{N}$ and $i \neq j$. Note that the voltage V_0 is also assumed to be independent of all ΔV_i 's.

For a write threshold value $w \geq 0$, define the classical counting process as

$$N(w) \triangleq \max\{n : n \in \{0\} \cup \mathbb{N} \text{ and } V_n \leq w\}.$$

$N(w)$ represents the number of voltage increments that occurred prior to passing the threshold w . The counting process $\{N(w), w \geq 0\}$ is called the renewal process.

Definition 1 (ISPP renewal process). *The ISPP renewal process $\tilde{N}(w)$ for a threshold voltage $w \geq 0$ is defined as*

$$\tilde{N}(w) \triangleq \max\{n : n \in \{-1, 0\} \cup \mathbb{N} \text{ and } \tilde{V}_n \leq w\}.$$

Note that if $V_0 = 0$, then $\tilde{N}(w) = N(w)$. For a particular w , the random variable $\tilde{N}(w)$ represents the number of ISPP steps required to stay just below threshold voltage w . Note that there exists another way to formulate the ISPP renewal process as the number of steps required to pass the threshold w . Let

$$\tilde{L}(w) \triangleq \min\{n : n \in \{0\} \cup \mathbb{N} \text{ and } \tilde{V}_n > w\}.$$

Then, clearly

$$\tilde{L}(w) = \tilde{N}(w) + 1.$$

Although formulating ISPP using $\tilde{L}(w)$ seems to be natural and very simple to understand, we will continue to use $\tilde{N}(w)$ because it is consistent with the classical renewal process

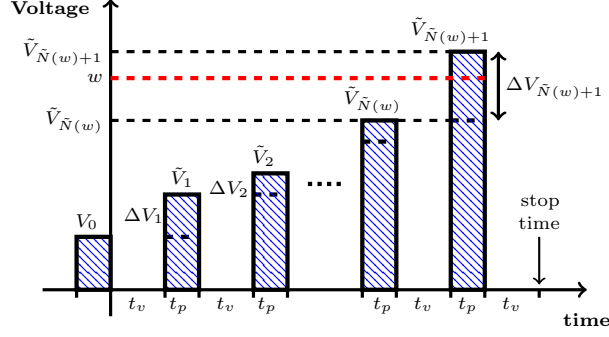


Figure 3.1: ISPP for a cell with starting voltage V_0 and threshold w .

formulation [36], thus simplifying the use of known results.

Remark Note that $\tilde{N}(w)$ is dependent on the target input B . $\tilde{N}(w) = -1$ happens either when $B = 0$, or when $B = 1$ and $V_0 \geq w$. i.e., the cell voltage is already above the threshold w in the erased state and it need not be programmed ($\tilde{L}(w) = 0$).

Remark Note that if $V_0 = 0$, the ISPP renewal process $\tilde{N}(w)$ is reduced to the classical renewal process $N(w)$. The increment ΔV_i represents the inter-occurrence random variable. In practice, the ISPP renewal process $\tilde{N}(w)$ has starting voltage V_0 that tends to be a Gaussian random variable, $V_0 \sim \mathcal{N}(\mu_0, \sigma_0^2)$ [40]. Thus, the ISPP renewal process is a renewal process with a random starting voltage. It belongs to the class of processes known as “delayed renewal processes” [36].

Observation 1. Using the renewal process formulation [36]:

- (a) $\tilde{V}_{\tilde{N}(w)+1}$ is the voltage of the cell immediately after the threshold voltage w is exceeded.
- (b) $\tilde{N}(w) + 1$ is the number of incremental steps required to pass the threshold voltage w .
- (c) The “ISPP overshoot” $\tilde{\Gamma}(w) \triangleq \tilde{V}_{\tilde{N}(w)+1} - w$ is the excess programmed voltage after passing the threshold w .
- (d) $\tilde{N}(w) \geq n$ if and only if $\tilde{V}_n \leq w$.

3.3 Distribution of $\tilde{N}(w)$

While there exist several asymptotic results for the classical renewal process $N(w)$, there are not many results in classical renewal theory when w is finite. To analyze the ISPP process $\tilde{N}(w)$, however, we are interested in obtaining a good approximation of the probability mass function (PMF) of $\tilde{N}(w)$ when w is finite ($0V \leq w \leq 5V$).

In this section, we first derive the connection between the ISPP renewal process $\tilde{N}(w)$ in the finite threshold regime and the classical renewal process $N(w)$. Then, we use some existing asymptotic results to simplify the analysis of the ISPP process in the finite threshold regime. To be precise, we obtain a close approximation for the PMF of $\tilde{N}(w)$, and linear approximations for mean and variance of $\tilde{N}(w)$. We also bound the maximal error between the true distribution and our approximation. Finally, we closely approximate the distribution of the overshoot $\tilde{\Gamma}(w)$.

3.3.1 Convergence

In this section our goal is to characterize $\tilde{N}(w)$. Precisely, we seek to find a relationship between $\tilde{N}(w)$ and $N(w)$. To that end, for any $\ell \in \mathbb{N}$, we define a new counting process $N_\ell(w)$, related to $N(w)$ and two constants μ and μ_Δ defined above, as

$$N_\ell(w) \triangleq \max\{N(w - \mu_0 + \ell\mu_\Delta) - \ell, -1\}, \quad (3.4)$$

Next, we show how $\tilde{N}(w)$ is related to $N_\ell(w)$.

Proposition 1. *Let $\mu_\Delta = E[\Delta V_i]$ and $\sigma_\Delta^2 = \text{Var}(\Delta V_i) = \frac{\sigma_0^2}{\ell}$. For any $w > 0$,*

$$N_\ell(w) \xrightarrow{D} \tilde{N}(w) \quad \text{as } \ell \rightarrow \infty, \quad (3.5)$$

where \xrightarrow{D} denotes convergence in distribution.

Proof : It is known from the central limit theorem (CLT) that

$$\sum_{i=0}^{\ell} (\Delta V_i - \mu_{\Delta}) + \mu_0 \xrightarrow{D} V_0 \quad \text{as } \ell \rightarrow \infty. \quad (3.6)$$

Therefore, for any constant d and $\epsilon > 0$, there exists L such that for all $\ell > L$

$$\left| P\left\{ \sum_{i=0}^{\ell} (\Delta V_i - \mu_{\Delta}) + \mu_0 \leq d \right\} - P\{V_0 \leq d\} \right| < \frac{\epsilon}{2}. \quad (3.7)$$

To prove (3.5), we bifurcate the proof into two cases:

i) $n \geq 0$:

$$\begin{aligned}
|\Delta P| &\triangleq \left| P\{N_\ell(w) \geq n\} - P\{\tilde{N}(w) \geq n\} \right| \\
&\stackrel{(a)}{=} \left| P\{N(w - \mu_0 + \ell\mu_\Delta - \ell) \geq n\} - P\{\tilde{N}(w) \geq n\} \right| \\
&\stackrel{(b)}{=} \left| P\left\{ \sum_{i=1}^{\ell} \Delta V_i - \mu_\Delta + \mu_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} - P\{\tilde{N}(w) \geq n\} \right| \\
&= \left| P\left\{ \sum_{i=1}^{\ell} (\Delta V_i - \mu_\Delta) + \mu_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} \right. \\
&\quad \left. - P\left\{ V_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} \right. \\
&\quad \left. + P\left\{ V_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} - P\{\tilde{N}(w) \geq n\} \right| \\
&\stackrel{(c)}{\leq} \left| P\left\{ \sum_{i=1}^{\ell} (\Delta V_i - \mu_\Delta) + \mu_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} \right. \\
&\quad \left. - P\left\{ V_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} \right| \\
&\quad + \left| P\left\{ V_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} - P\{\tilde{N}(w) \geq n\} \right| \\
&\stackrel{(d)}{<} \frac{\epsilon}{2} + \left| P\left\{ V_0 + \sum_{j=1}^n \Delta V_j \leq w \right\} - P\{\tilde{N}(w) \geq n\} \right| \\
&= \frac{\epsilon}{2} + \left| P\{\tilde{V}_n \leq w\} - P\{\tilde{N}(w) \geq n\} \right| \\
&\stackrel{(e)}{=} \frac{\epsilon}{2}, \tag{3.8}
\end{aligned}$$

where (a) holds because when $n \geq 0$, the condition $N_\ell(w) \geq 0$ is equivalent to $N(w - \mu_0 + \ell\mu_\Delta) \geq \ell$, (b) and (e) follow from Observation 1-d, (c) is the triangle inequality and (d) follows from (3.7). Note that

$$P\{\tilde{N}(w) = n\} = P\{\tilde{N}(w) \geq n\} - P\{\tilde{N}(w) \geq n + 1\}. \tag{3.9}$$

Thus, using (3.8), (3.9) and the triangle inequality it is easy to verify that for $\ell > L$,

$$\left| P\{N_\ell(w) = n\} - P\{\tilde{N}(w) = n\} \right| < \epsilon.$$

ii) $n = -1$:

$$\begin{aligned} \Delta P' &\triangleq \left| P\{N_\ell(w) = -1\} - P\{\tilde{N}(w) = -1\} \right| \\ &\stackrel{(a)}{=} \left| P\{N(w - \mu_0 + \ell\mu_\Delta) - \ell < 0\} - P\{\tilde{N}(w) = -1\} \right| \\ &\stackrel{(b)}{=} \left| P\left\{ \sum_{i=1}^{\ell} (\Delta V_i - \mu_\Delta) + \mu_0 > w \right\} - P\{\tilde{N}(w) = -1\} \right| \\ &= \left| P\left\{ \sum_{i=1}^{\ell} (\Delta V_i - \mu_\Delta) + \mu_0 > w \right\} - P\{V_0 > w\} \right. \\ &\quad \left. + P\{V_0 > w\} - P\{\tilde{N}(w) = -1\} \right| \\ &\stackrel{(c)}{<} \frac{\epsilon}{2} + \left| P\{V_0 > w\} - P\{\tilde{N}(w) = -1\} \right| \\ &\stackrel{(d)}{=} \frac{\epsilon}{2} < \epsilon, \end{aligned} \tag{3.10}$$

where (a) holds because when $n = -1$, the condition $N_\ell(w) < 0$ is equivalent to $N(w - \mu_0 + \ell\mu_\Delta) < \ell$, (b) follows from Observation 1-d, (c) is the triangle inequality and (d) was explained in Remark 3.2.

Corollary 1. Let $\zeta = \lceil \frac{\sigma_0^2}{\sigma_\Delta^2} \rceil$. Then,

$$N_\zeta(w) \xrightarrow{D} \tilde{N}(w) \quad \text{as } \zeta \rightarrow \infty. \tag{3.11}$$

Proof : The only issue is to replace ℓ in equation (3.6) with ζ . Let $V(\zeta) \sim \mathcal{N}(\mu_0, \zeta\sigma_\Delta^2)$.

It is known from CLT that

$$\sum_{i=0}^{\zeta} (\Delta V_i - \mu_\Delta) + \mu_0 \xrightarrow{D} V(\zeta) \quad \text{as } \zeta \rightarrow \infty,$$

Hence, to prove (3.11), we only need to show that

$$V(\zeta) \xrightarrow{D} V_0 \quad \text{as } \zeta \rightarrow \infty. \quad (3.12)$$

Towards that end, let $d \in \mathbb{R}$. Then we have

$$\begin{aligned} |P\{V(\zeta) \leq d\} - P\{V_0 \leq d\}| &= \left| \int_{\frac{d-\mu_0}{\sigma_\Delta \sqrt{\zeta}}}^{\frac{d-\mu_0}{\sigma_0}} \frac{e^{-t^2/2}}{\sqrt{2\pi}} dt \right| \\ &< \frac{C}{\sqrt{\zeta+1}}, \end{aligned} \quad (3.13)$$

where C is a constant. Thus, for any $\epsilon > 0$, (3.12) holds if $\zeta > \frac{C^2}{\epsilon^2} - 1$. The rest follows the proof of Proposition 1.

Remark Note that for the ISPP renewal process $\tilde{N}(w)$, it is known that V_0 usually has negative mean ($\mu_0 < 0$) and a large variance $\sigma_0 \gg \sigma_\Delta$. (E.g., if $\sigma_0 = 0.5$ and $\Delta V_i \sim U[0.1, 0.15]$, then $\zeta > 1000$). Thus, the process $\tilde{N}(w)$ can be viewed as a shifted version of the classical renewal process in which the starting point V_0 is a large negative voltage. In this case, the distribution of $\tilde{N}(w)$ is very similar to the asymptotic distribution of the classical renewal process $N(w)$ when $w \rightarrow \infty$.

3.3.2 Gaussian Approximation

Figure 3.2 shows the histogram of the number of required steps for a cell voltage to pass the typical threshold voltage $w = 1.5V$. The histogram was obtained assuming that the number of cells in the page is $m = 10^6$, and the target bit for all the cells is $B = 1$. It is clear from Figure 3.2 that $\tilde{N}(w)$ tends to have a Gaussian-like PMF. Our simulation shows that the PMF for $\tilde{N}(w)$ is very similar for any valid step size $\Delta V \sim U[a, b]$ and for any $w > 0$ as long as $\zeta = \frac{\sigma_0^2}{\sigma_\Delta^2} \gg 1$.

We next embark on quantifying the deviation between PMF of $\tilde{N}(w)$ and its Gaussian

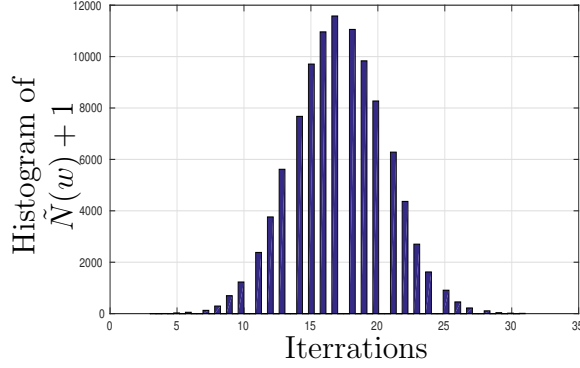


Figure 3.2: Histogram of $\tilde{N}(w) + 1$ for threshold voltage $w = 1.5V$, and $\Delta V \sim U[0.1, 0.2]$.

approximation when w and ζ are finite. For $n = 1, 2, \dots$, let $F_n(w)$ denote the cumulative distribution function (CDF) of the ISPP voltage \tilde{V}_n

$$F_n(w) \triangleq P\{\tilde{V}_n \leq w\}, \quad w \geq 0.$$

Using Observation 1-d, it is easy to verify that

$$P\{\tilde{N}(w) = n\} = F_n(w) - F_{n+1}(w). \quad (3.14)$$

Let us define a new random process $\tilde{N}_\zeta(w)$ whose CDF is Gaussian, i.e.,

$$P\{\tilde{N}_\zeta(w) = n\} \triangleq \begin{cases} 0 & \text{if } n \leq -2 \\ Q\left(\frac{w - \mu_0}{\sigma_0}\right) & \text{if } n = -1, \\ Q\left(\frac{w - \mu_0 - (n+1)\mu_\Delta}{\sigma_\Delta \sqrt{n + \zeta + 1}}\right) - Q\left(\frac{w - \mu_0 - n\mu_\Delta}{\sigma_\Delta \sqrt{n + \zeta}}\right) & \text{if } n \geq 0 \end{cases} \quad (3.15)$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt$.

Corollary 2. $\tilde{N}_\zeta(w) \xrightarrow{D} \tilde{N}(w)$ as $\zeta \rightarrow \infty$.

Proof : Omitted because it follows the proof of Proposition 1.

Proposition 2. Set $\rho \triangleq E(|\Delta V_i - \mu_\Delta|^3)$ and $\zeta \triangleq \lceil \frac{\sigma_0^2}{\sigma_\Delta^2} \rceil$. Then,

$$\tilde{N}_\zeta(w) \xrightarrow{D} \tilde{N}(w) \quad \text{as } \zeta \rightarrow \infty,$$

and for any finite $\zeta > 0$,

$$\left| P\{\tilde{N}_\zeta(w) = n\} - P\{\tilde{N}(w) = n\} \right| \leq \frac{\rho}{\sigma_\Delta^3 \sqrt{n + \zeta}}. \quad (3.16)$$

Proof : We only sketch for $n \geq 0$ ($n = -1$ is clear).

$$\begin{aligned} & P\{N_\zeta(w) \geq n\} \\ & \stackrel{(a)}{=} P\{N(w - \mu_0 + \zeta\mu_\Delta) - \zeta \geq n\} \\ & = P\left\{ \sum_{i=1}^{\zeta+n} \Delta V_i \leq w - \mu_0 + \zeta\mu_\Delta \right\} \\ & = P\left\{ \frac{\sum_{i=1}^{\zeta+n} \Delta V_i - (n+\zeta)\mu_\Delta}{\sqrt{n+\zeta}\sigma_\Delta} \leq \frac{w - \mu_0 + \zeta\mu_\Delta - (n+\zeta)\mu_\Delta}{\sigma_\Delta \sqrt{n+\zeta}} \right\} \\ & \stackrel{(b)}{\approx} 1 - Q\left(\frac{w - \mu_0 - n\mu_\Delta}{\sigma_\Delta \sqrt{n+\zeta}} \right), \end{aligned}$$

where (a) holds because when $n \geq 0$, the condition $N_\zeta(w) \geq n$ is equivalent to $N(w - \mu_0 + \zeta\mu_\Delta) \geq n + \zeta$ using (3.4) and (b) holds because of the CLT. The rest of the proof follows using (9) and the Berry-Esseen Theorem which is given in Appendix A.

Bound (3.16) holds for all possible distributions of the random variable ΔV_i . However, when $\{\Delta V_i\} \sim U[a, b]$, numerical simulations suggest that the bound could be much tighter than $o(\frac{1}{\sqrt{n+\zeta}})$. This observation is proved in the following corollary.

Corollary 3. If $\Delta V_i \sim U[a, b]$, and denoting δ_n as

$$\delta_n = \frac{1}{7.5\pi n} + \frac{1}{\pi} \left(\frac{2}{\pi}\right)^n + \frac{12}{\pi^3 n} \exp(-\pi^2 n/24), \quad (3.17)$$

then,

$$\left| P\{\tilde{N}_\zeta(w) = n\} - P\{\tilde{N}(w) = n\} \right| \leq 2\delta_{n+\zeta}.$$

Proof : Similar to the proof of Proposition 2 and using the Uspensky Theorem, given in the Appendix A.

Remark To obtain a simpler bound when $\Delta V_i \sim U[a, b]$, it is easy to verify that when $n + \zeta \geq 10$, the following bound holds:

$$\left| P\{N_\ell(w) = n\} - P\{\tilde{N}(w) = n\} \right| \leq \frac{1}{10 \cdot (n + \zeta)}.$$

Although it is possible to approximate the mean and variance of $\tilde{N}(w)$ using the approximate distribution given in Proposition 2, such method to approximate $E[\tilde{N}(w)]$ and $\text{Var}(\tilde{N}(w))$ only provides numerical results. To obtain an analytical approximation, we modify an existing linear approximation for mean and variance of $N(w)$ in classical renewal and derive similar formulas for mean and variance of $\tilde{N}(w)$. Note that the linear approximation formulas were derived from a Fourier approximation approach [48].

Proposition 3. Let $\mu_j = E[(\Delta V_i)^j]$ be the j -th moment of the random variable ΔV_i . Let $K_1 = \frac{\mu_2}{2\mu_\Delta^2} - 1$ and $K_2 = \left(\frac{\mu_2}{2\mu_\Delta^2}\right)^2 - \frac{\mu_3}{6\mu_\Delta^3}$. If ΔV_i is a continuous random variable and $\mu_2 < \infty$, then

$$E[\tilde{N}(w)] \approx P\{V_0 \leq w\} \cdot \left(\frac{w - \mu_0}{\mu_\Delta} + K_1 + 1 \right) - 1 \quad (3.18)$$

$$\text{Var}(\tilde{N}(w)) \approx Pr(V_0 \leq w) \cdot \left(\sigma_\Delta^2 \frac{(w - \mu_0 + \zeta \mu_\Delta)}{\mu_\Delta^3} + A \right) + \frac{\sigma_0^2}{\mu_\Delta^2}, \quad (3.19)$$

where $A = K_1 \cdot (1 + K_1) + 4K_2$.

Proof :

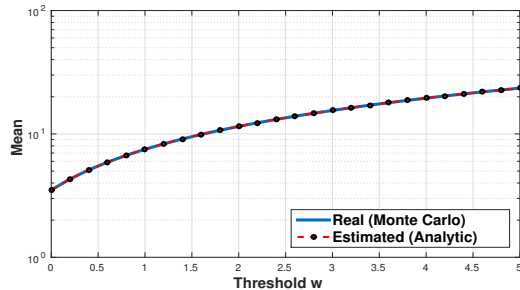
$$\begin{aligned}
\mu_{\tilde{n}} &\triangleq E[\tilde{N}(w)] \\
&= E\left[E[\tilde{N}(w)|V_0]\right] \\
&= Pr(V_0 \leq w) \cdot E\left[\tilde{N}(w)|V_0 \leq w\right] + Pr(V_0 > w) \cdot E\left[\tilde{N}(w)|V_0 > w\right] \\
&\stackrel{(a)}{=} Pr(V_0 \leq w) \cdot E\left[N(w - \mu_0 + \zeta\mu_\Delta) - \zeta\right] - Pr(V_0 > w) \\
&\stackrel{(b)}{\approx} Pr(V_0 \leq w) \cdot \left(\frac{w - \mu_0}{\mu_\Delta} + K_1 + 1\right) - 1, \tag{3.20}
\end{aligned}$$

where (a) holds because $E[\tilde{N}(w)|V_0 \geq w] = -1$, and (b) follows from the linear approximation of $E[N(w)]$ in [48].

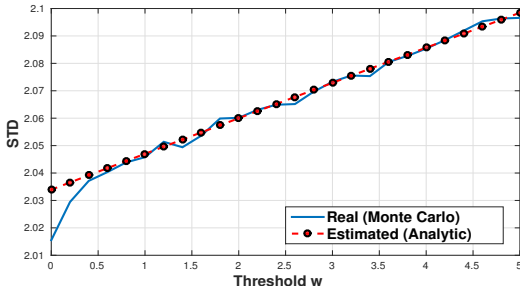
$$\begin{aligned}
\sigma_{\tilde{n}}^2 &\triangleq Var\left(\tilde{N}(w)\right) \\
&= E\left[Var(\tilde{N}(w)|V_0)\right] + Var\left(E[\tilde{N}(w)|V_0]\right) \\
&= Pr(V_0 \leq w) \cdot Var\left(\tilde{N}(w)|V_0 \leq w\right) \\
&\quad + Pr(V_0 > w) \cdot Var\left(\tilde{N}(w)|V_0 > w\right) + Var\left(E[\tilde{N}(w)|V_0]\right) \\
&\stackrel{(a)}{=} Pr(V_0 \leq w) \cdot Var(N(w - \mu_0 + \zeta\mu_\Delta) - \zeta) + \frac{\sigma_0^2}{\mu_\Delta^2} \\
&\stackrel{(b)}{\approx} Pr(V_0 \leq w) \cdot \left(\sigma_\Delta^2 \frac{(w - \mu_0 + \zeta\mu_\Delta)}{\mu_\Delta^3} + A\right) + \frac{\sigma_0^2}{\mu_\Delta^2}, \tag{3.21}
\end{aligned}$$

where (a) holds because $Var\left(\tilde{N}(w)|V_0 > w\right) = 0$, $Var\left(E[\tilde{N}(w)|V_0]\right) = \frac{\sigma_0^2}{\mu_\Delta^2}$ [36], and (b) follows from the Gaussian approximation of $N(w)$ in [36].

Remark In [48], it was shown that for a classical renewal process $N(w)$ in which the increment ΔV_i has a finite ρ -th absolute moment for some $\rho \geq 3$, the linear approximations for $E[N(w)]$ and $Var(N(w))$ hold with error terms $o(w^{1-\rho})$ and $o(w^{2-\rho})$, respectively. Deriving



(a) Mean $E[\tilde{N}(w)]$,



(b) standard deviation $\sigma_{\tilde{N}(w)}$

Figure 3.3: Comparison between Monte Carlo (real) and estimated (analytic) computation of $E[\tilde{N}(w)]$ and $\sigma_{\tilde{N}(w)}$ for all threshold $w \in [0, 5]$.

similar error terms for the ISPP renewal process using Proposition 1 is straightforward.

Remark Since the Gaussian distribution assumption for erased state seems unrealistic (due to the finite support of V_0), we checked the accuracy of our results for the case that the erase state is not Gaussian. Simulation results show that equations (3.18) and (3.19) provide very accurate approximations for mean and variance when the distribution of V_0 is a truncated Gaussian and the distribution is symmetric around μ_0 .

Figure 3.3 compares the mean and standard deviation of $\tilde{N}(w)$ to our proposed analytic mean and standard deviation approximation in equations (3.18) and (3.19) for all possible thresholds $w \in [0, 5]$, where we assume erased state $V_0 \sim \mathcal{N}(-1, 0.25)$ and step size $\Delta V_i \sim U[0.2, 0.3]$. We consider a very long page ($m = 10^6$), and program all the cells to pass the threshold w . Simulation results show that the maximum error for the mean is less than 3mV, which is a very small error from a practical point of view.

3.3.3 Distribution of $\tilde{\Gamma}(w)$

Another key factor in page programming is the voltage overshoot distribution for the cells which pass the target threshold w . Given $B = 1$, let $\tilde{\Gamma}(w)$ denote the amount of overshoot of a cell voltage after passing threshold w . Then,

$$\tilde{\Gamma}(w) \triangleq \tilde{V}_{\tilde{N}(w)+1} - w. \quad (3.22)$$

Remark Note that the programming precision is also directly related to the distribution of increments ΔV_i during the ISPP process. The larger the increment step size is, the wider the pdf of $\tilde{\Gamma}(w)$ becomes, and thereby programming becomes less accurate.

Proposition 4. Let $\zeta = \lceil \frac{\sigma_0^2}{\sigma_\Delta^2} \rceil$. If $V_0 < w$, then

$$\lim_{\zeta \rightarrow \infty} P\{\tilde{\Gamma}(w) \leq \gamma\} = \frac{1}{\mu_\Delta} \int_0^\gamma \{1 - F_{\Delta V}(y)\} dy, \quad \gamma \geq 0. \quad (3.23)$$

Proof : It is straightforward by combining Proposition 1 and the asymptotic residual distribution in classical renewal theory [36].

Remark Since the process $\tilde{N}(w)$ can be viewed as a shifted version of a classical renewal process $N(w)$ in which the process starts at a large negative voltage point v (roughly $v = \mu_0 - \sqrt{\zeta} \cdot \mu_\Delta$), the distribution of the residual $\tilde{\Gamma}(w)$ for any finite threshold w is very similar to the asymptotic distribution of $\Gamma(w)$ in the classical renewal process. To the best of our knowledge, there is no known deviation bound result for the overshoot of the classical renewal process. While simulation results confirm that the distribution of the overshoot $\tilde{\Gamma}(w)$ for finite $w > V_0$ is very similar to equation (3.23), obtaining the deviation bound result is a subject for further research.

Simulation results show that the CDF of the overshoot for finite $\zeta \gg 1$ is very close to the asymptotic result (3.23). Therefore, in this work, for any $w > 0$ and for a finite value ζ , we

approximate the overshoot distribution as

$$P\{\tilde{\Gamma}(w) \leq \gamma\} \approx \frac{1}{\mu_{\Delta}} \int_0^{\gamma} \{1 - F_{\Delta V}(y)\} dy, \quad \gamma \geq 0.$$

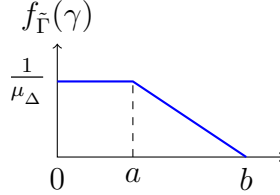


Figure 3.4: The pdf of $\tilde{\Gamma}(w)$ when $\Delta V_i \sim U[a, b]$.

Figure 3.4 shows the pdf for $\tilde{\Gamma}(w)$ when the step size is uniformly distributed ($\Delta V_i \sim U[a, b]$).

Using equation (3.23), it is easy to show that

$$E(\tilde{\Gamma}(w)) = \frac{E[\Delta V_i^2]}{2E[\Delta V_i]} \quad (3.24)$$

$$E(\tilde{\Gamma}(w)^2) = \frac{E[\Delta V_i^3]}{3E[\Delta V_i]}. \quad (3.25)$$

Remark Figure 3.5 shows the overshoot distribution for all states after programming each page given that $\Delta V_i \sim U[a, b]$. Note that this figure shows merely the effect of ISPP programming and excludes all other available degradation sources in MLC flash such as the ICI effect (which we study in the next section).

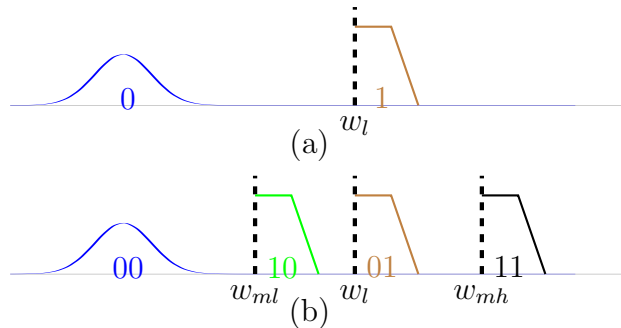


Figure 3.5: Overshoot distribution after the ISPP process stops, (a) LSB page overshoot distribution, (b) MSB page overshoot distributions.

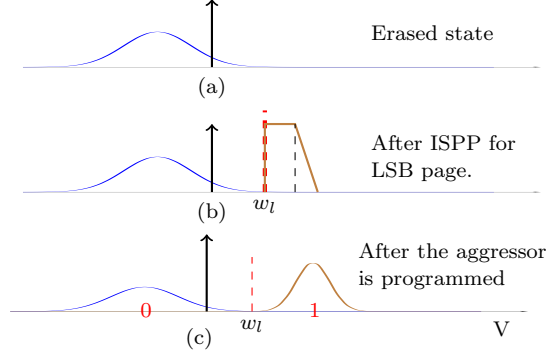


Figure 3.6: Programming the LSB page. (a) pdf of V_0 before writing the LSB page, (b) pdf of $\tilde{V}_{\tilde{N}(w_l)+1}(B_0^{(v)})$ after ISPP with threshold w_l and (c) Including the ICI effect on the LSB page ($V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)})$).

Remark Note that equation (3.23) explains why the overshoot distribution $\tilde{\Gamma}(w)$ is independent of the threshold voltage w and only depends on the statistical property of step size ΔV_i . Also, note that for cells with target input $X = B_m B_l = 11$, the overshoot distribution is obtained from the same equation (3.23), and the proof was shown in Proposition 3.1 in [49].

3.4 Inter-cell Interference (ICI)

Using the obtained distribution for the ISPP overshoot $\tilde{\Gamma}(w)$, it is possible to analyze inter-cell interference (ICI) very accurately. In this section we first analyze the effects of ICI on the LSB page, and then the ICI effect on the MSB page of a 2-bits MLC NAND flash memory.

3.4.1 ICI Effect on the LSB page

In the writing process, there might be a case that only the LSB page is used to store the data. In other words, some of the word-lines might be used to program only a single page rather than multiple pages. Thus, we need to separately analyze the ICI for the LSB page. Also, the ICI effect on the LSB page helps to better estimate the unknown hidden states

which were explained in detail in [50].

Let $B_0^{(v)}$ and $B_0^{(a)}$ denote the LSB target bit corresponding to the victim cell (k, ℓ) and the LSB target bit corresponding to the aggressor cell $(k+1, \ell)$, respectively. Let $V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)})$ denote the voltage of victim cell (k, ℓ) after LSB programming of both pages k and $k+1$. Let $Z(B_0^{(a)})$ be the ICI effect on victim cell (k, ℓ) due to LSB ISPP page programming of the vertical aggressor cell $(k+1, \ell)$. Given $w_l > 0$, the voltage $V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)})$ is obtained as

$$V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)}) = \tilde{V}_{\tilde{N}(w_l)+1}(B_0^{(v)}) + Z(B_0^{(a)}), \quad (3.26)$$

where

$$\tilde{V}_{\tilde{N}(w_l)+1}(B_0^{(v)}) = \begin{cases} V_0 & \text{if } B_0^{(v)} = 0 \\ \tilde{V}_{\tilde{N}(w_l)+1} & \text{if } B_0^{(v)} = 1 \end{cases}, \quad (3.27)$$

and

$$Z(B_0^{(a)}) = \begin{cases} 0 & \text{if } B_0^{(a)} = 0 \\ c_v [w_l + \tilde{\Gamma}(w_l) - V_0] & \text{if } B_0^{(a)} = 1 \end{cases}. \quad (3.28)$$

Using (3.28) we compute the mean and the variance of $Z(B_0^{(a)})$ as

$$\begin{aligned} E[Z(B_0^{(a)})] &= \frac{c_v}{2} \cdot [w_l + E[\tilde{\Gamma}(w_l)] - \mu_0], \\ \text{Var}[Z(B_0^{(a)})] &= \frac{c_v^2}{2} \cdot (\text{Var}(\tilde{\Gamma}(w_l)) + \sigma_0^2) \\ &\quad + \frac{c_v^2}{4} \cdot [w_l + E[\tilde{\Gamma}(w_l)] - \mu_0]^2, \end{aligned}$$

where $E[\tilde{\Gamma}(w_l)]$ and $\text{Var}(\tilde{\Gamma}(w_l))$ are computed using (3.24) and (3.25). Figure 3.6 shows the pdfs of the voltages of the cells in each stage during page programming. Figure 3.6(a) shows the pdf of the cell voltage before LSB page programming (each cell is in the erased state). Figure 3.6(b) shows the pdf of ISPP for LSB page with threshold w_l . Finally, the ICI effect on the cell due to the LSB programming of vertical aggressor is shown in 3.6(c).

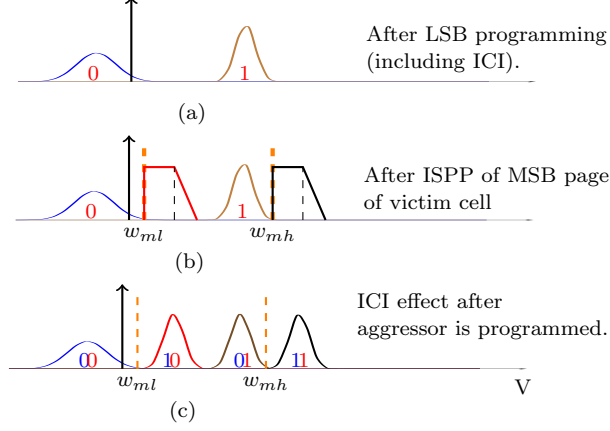


Figure 3.7: Programming the MSB page. (a) Before writing the MSB page ($V^{(k,\ell)}(B_0^{(v)}, B_0^{(a)})$), (b) pdf of ($\tilde{V}_{\tilde{N}(w)+1}(X^{(v)})$) after ISPP for MSB page with threshold voltages w_{m_1} and w_{m_2} and (c) the ICI effect on the MSB page ($V^{(k,\ell)}(X^{(v)}, X^{(a)})$).

3.4.2 ICI Effect on the MSB page

Apart from the cases in which programming terminates after programming the LSB page, it is possible to simplify the overall analysis of ICI by directly computing the ICI effect on the MSB page. Let $X^{(v)} = B_1^{(v)}B_0^{(v)}$ and $X^{(a)} = B_1^{(a)}B_0^{(a)}$ denote the target 2-bit-symbol corresponding to victim cell (k, ℓ) and aggressor cell $(k + 1, \ell)$, respectively. Let $V^{(k,\ell)}(X^{(v)}, X^{(a)})$ denote the voltage of victim cell (k, ℓ) after both pages k and $k + 1$ are MSB-programmed. Let $Z(X^{(a)})$ be the ICI effect on victim cell (k, ℓ) due to MSB page programming of the vertical aggressor cell $(k + 1, \ell)$. Given $w \in \{w_{m_1}, w_l, w_{m_2}\}$, the voltage $V^{(k,\ell)}(X^{(v)}, X^{(a)})$ is obtained as

$$V^{(k,\ell)}(X^{(v)}, X^{(a)}) = \tilde{V}_{\tilde{N}(w)+1}(X^{(v)}) + Z(X^{(v)}, X^{(a)}), \quad (3.29)$$

where

$$\tilde{V}_{\tilde{N}(w)+1}(X^{(v)}) = \begin{cases} V_0 & \text{if } X^{(v)} = 00 \\ \tilde{V}_{\tilde{N}(w_{m_1})+1} & \text{if } X^{(v)} = 10 \\ \tilde{V}_{\tilde{N}(w_l)+1} & \text{if } X^{(v)} = 01 \\ \tilde{V}_{\tilde{N}(w_{m_2})+1} & \text{if } X^{(v)} = 11 \end{cases}, \quad (3.30)$$

Thus, if $X^{(v)} \in \{00, 01\}$, we obtain

$$Z(X^{(v)}, X^{(a)}) = \begin{cases} 0 & \text{if } X^{(a)} = 00 \\ c_v(w_{m_1} + \tilde{\Gamma}(w_{m_1}) - V_0) & \text{if } X^{(a)} = 10 \\ c_v(w_l + \tilde{\Gamma}(w_l) - V_0) & \text{if } X^{(a)} = 01 \\ c_v(w_{m_2} + \tilde{\Gamma}(w_{m_2}) - V_0) & \text{if } X^{(a)} = 11 \end{cases},$$

and if $X^{(v)} \in \{01, 10\}$, we obtain

$$Z(X^{(v)}, X^{(a)}) = \begin{cases} 0 & \text{if } X^{(a)} = 00 \\ c_v \cdot (w_{m_1} + \tilde{\Gamma}(w_{m_1}) - V_0) & \text{if } X^{(a)} = 10 \\ 0 & \text{if } X^{(a)} = 01 \\ c_v \left(w_{m_2} + \tilde{\Gamma}(w_{m_2}) - V^{(k+l)}(B_0^{(v)}) \right) & \text{if } X^{(a)} = 11 \end{cases},$$

where $V^{(k+l)}(B_0^{(v)})$ is computed from (3.26) by setting $B_0^{(a)} = 1$. Figure 3.7 shows the voltage distribution of the victim cell (k, ℓ) during the MSB page programming with threshold voltages w_{m_1} and w_{m_2} .

Remark Note that the ICI effect Z not only depends on the target input of the aggressor cell $X^{(a)}$, but it also depends on the target input state of the victim cell $X^{(v)}$. This can be explained by considering the order of programming LSB and MSB pages in the MLC NAND flash write process [39]. In fact, due to the specific order of programming the pages

of neighboring word-lines, the ICI effect on the LSB is removed during the ISPP verification process of the MSB.

3.4.3 ICI Analysis

In this section, we provide simulation results to show the effects of ICI and analyze the effects of step size variation on the overall performance. The simulation results are obtained from a 4MB, 2-bit MLC flash memory block with the all-bit-line structure. We assume that the target input is an i.i.d process. The write thresholds are set to $W = \{1.0V, 2.0V, 3.0V\}$. The mean and standard deviation of V_0 are assumed to be $\mu_0 = -1.0V$ and $\sigma_0 = 0.5V$, respectively. The vertical capacitance coupling is assumed to be $c_v = 0.06$. The read thresholds r are numerically computed using the designed optimal MAP detector approach in [40]. We fixed the write thresholds and varied the parameters a and b of the random step size $\Delta V_i \sim U[a, b]$ in the ISPP process. We assumed that the minimum step size a varies in the range $[0.05, 1]$ and $b = 1.4 \times a$. Also, we run ISPP until the number of steps reaches a pre-specified maximum number of iterations η , and we assume $\eta = 15$.

Remark The MLC flash channel belongs to the class of channels with memory, and the ICI effect is considered as the major source of channel memory. The information rate for this class of channels can be numerically computed using a forward sum-product recursion of the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [29].

There are two possible ways to compute an information rate measure for MLC flash channels: (i) compute the symbol-by-symbol mutual information, i.e., compute $I_1 = I(X^{(k,\ell)}; Y^{(k,\ell)})$; (ii) obtain the correct information rate for MLC flash using the BCJR algorithm [29]. Let K and L denote the number of word-lines and bit-lines in the block, respectively. Let $X_{(1,1)}^{(K,L)}$ and $Y_{(1,1)}^{(K,L)}$ denote the input sequence and output sequence of the whole block. Then, the MLC flash information rate obtained from [29], denoted by I_2 , is computed as

$$I_2 \triangleq \frac{1}{KL} I(X_{(1,1)}^{(K,L)}; Y_{(1,1)}^{(K,L)}).$$

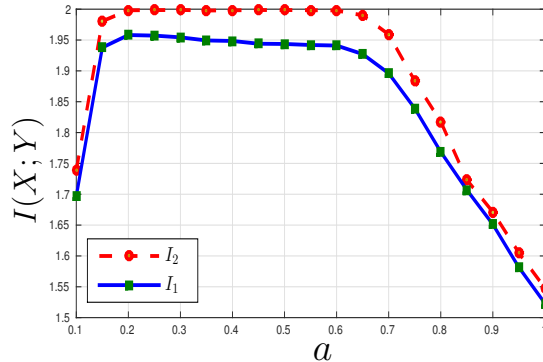


Figure 3.8: Performance of 2-bit MLC flash when $n_{PE} = 1$.

Note that for any i.i.d input process, I_1 is always a lower bound for I_2 , and the difference $I_2 - I_1$ is attributable to the ICI effect of MLC flash. Figure 3.8 shows the consequence of varying the step size a for both information rates I_1 (solid curve) and I_2 (dashed curve). Both I_1 and I_2 were computed using Monte Carlo method. In Figure 3.8, we exclude the aging effect on the performance by setting the program/erase cycling number $n_{PE} = 1$, and we disregard the retention effect by reading the stored data immediately after writing it. Figure 3.8 suggests that the flash performance is optimal when the step size parameter a , is chosen from the range $[0.2, 0.6]$, and $b = 1.4a$.

Remark The performance drop on the right side of both curves in Figure 3.8 is because of the large step size of ISPP programming, which leads to the over-injection problem (large overshoot $\tilde{\Gamma}(w)$). The performance drop on the left side, however, is due to the undershoot problem in ISPP and is caused by the relation between the maximum allowed number of steps η and the increment step size. The undershoot problem will be elaborated in the next section.

3.5 Write Latency

Although NAND flash-based solid state drives (SSD) are known to be very fast when writing and reading due to their array architecture, reducing their latency is still considered to be

one of the major design goals. The latency in SSDs happens mainly during the write process because of the iterative programming procedure. Note that the read process latency is much shorter than the write latency [37]. Hence, in this section, we focus on the ISPP latency and disregard the read latency. Since the write process in MLC flash is done at the granularity of a page, we compute the latency for a page.

As discussed earlier, ISPP only increases the voltage of those cells with target bit $B = 1$, i.e., the cells with target bit $B = 0$ do not need to pass the threshold. Moreover, the write threshold w is the same for each cell in the word-line. Simulations show that most of the cells in the page pass the threshold much earlier than the lagging few. Thus, it is not efficient to associate the write latency with the last cell that passes the threshold. In practice, the ISPP page programming operation ends in one of the following two possible ways: either all the cells with target bit $B = 1$ in the page reach their target threshold w or the number of steps reaches a predetermined maximum number. For a page, the predetermined maximum allowed number of steps in the ISPP process is called the “maximal delay”, and is denoted by η [5]. Hence, for a 2-bit MLC word-line, we denote the maximal delay corresponding to the LSB and MSB pages by η_l and η_m , respectively.

3.5.1 LSB page Latency

Let m be the length of the word-line. The LSB write latency is related to the number of steps needed for all the cells with target bit $B_0 = 1$ to pass w_l . Let $\tilde{T}_l^{(j)}(w_l)$ denote the ISPP latency corresponding to programming the LSB page of the j -th word-line. According to Definition 1, let $\tilde{N}^{(j,k)}(w_l) + 1$ be the number of steps for cell (j, k) to pass threshold w_l given $B_0^{(j,k)} = 1$. Then,

$$\tilde{T}_l^{(j)}(w_l) = \min\left\{ \max_{k: B_0^{(j,k)}=1} (\tilde{N}^{(j,k)}(w_l) + 1), \eta_l \right\} \quad (3.31)$$

Let k_1 be the index of the cell with largest value $\tilde{N}(w_l) + 1$ in the j -th word-line. In other

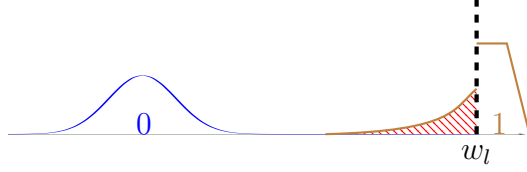


Figure 3.9: Histogram of LSB page voltage after η_l steps (ISPP stopped).

words, cell (j, k_1) is the most latent cell during LSB page programming. If $\eta_l < \tilde{N}^{(j, k_1)}(w_l) + 1$, there exists at least one cell that does not pass the threshold w_l , and causes the undershoot write error when trying to store its corresponding bit $B_0 = 1$. Note that in this case $\tilde{T}_l^{(j)}(w_l) = \eta_l$. Figure 3.9 shows the histogram of cell voltage after LSB page programming. The shaded part on the left side of w_l is because of the undershoot error; the non-shaded part corresponds to the overshoot error.

Remark Note that there exists a trade-off between programming latency and the undershoot programming error. It is possible to reduce η , and thereby decrease the ISPP latency; however, this leads to an increased undershoot error. Further, note that some cells with undershoot errors might pass their target threshold due to the ICI incurred while subsequently programming the neighboring pages. When designing a flash memory, care should be taken so that, the amount of undershoot error is always much smaller than the available ECC capability. In other words, we can allow a small percentage of undershoot errors to obtain some write latency reduction, and handle the rare undershoot errors using the flash ECC module.

Definition 2 (ISPP undershoot error rate). *The allowed undershoot error rate, denoted by α , is the maximum allowed probability that a cell does not reach its target threshold after an ISPP page programming.*

Proposition 5 (Setting η_l). *Let $\zeta = \lceil \frac{\sigma_0^2}{\sigma_\Delta^2} \rceil$. To guarantee a pre-specified allowed ISPP undershoot error rate α , it suffices to choose the number of allowed steps η_l as the smallest integer that satisfies*

$$Q\left(\frac{w_l - \mu_0 - \eta_l \mu_\Delta}{\sigma_\Delta \sqrt{\zeta + \eta_l}}\right) \geq 1 - \alpha + \delta_{\zeta + \eta_l}, \quad (3.32)$$

where $\delta_{\zeta + \eta_l}$ is given by (3.17).

Proof : As discussed earlier, $\tilde{N}(w_l) + 1$ is the number of steps required for a cell to pass the threshold w_l during LSB page programming given that $B = 1$. Let \mathcal{E}_{η_l} represent the event that a cell does not reach the threshold after η_l steps. Clearly, \mathcal{E}_{η_l} happens if and only if $\tilde{N}(w_l) + 1 > \eta_l$. Hence,

$$P\{\mathcal{E}_{\eta_l}\} = P\{\tilde{N}(w_l) + 1 > \eta_l\}$$

Consequently, the allowed undershoot error rate (probability) α must be chosen to satisfy

$$\begin{aligned} \alpha &\geq P\{\mathcal{E}_{\eta_l}\} \\ &= P\{\tilde{N}(w_l) \geq \eta_l\}. \end{aligned} \tag{3.33}$$

Using Corollary 3 it is straightforward to show that

$$P\{\tilde{N}(w_l) \geq \eta_l\} \leq 1 - Q\left(\frac{w_l - \mu_0 - \eta_l \mu_{\Delta}}{\sigma_{\Delta} \sqrt{\zeta + \eta_l}}\right) + \delta_{\zeta + \eta_l}. \tag{3.34}$$

The rest of the proof follows by combining equations (3.32) and (3.34).

3.5.2 MSB page programming

Although the MSB latency analysis is very similar to that of the LSB, there exists a major difference because the MSB is programmed using two threshold voltages $\{w_{m_1}, w_{m_2}\}$. The extra voltage threshold makes the problem more complicated than in the LSB case. Let us assume $w_{m_1} < w_{m_2}$. Then, the ISPP latency corresponding to programming the MSB page of the j -th word-line, denoted by $\tilde{T}_m^{(j)}(w_{m_1}, w_{m_2})$, is defined as

$$\tilde{T}_m^{(j)}(w_{m_1}, w_{m_2}) \triangleq \min\left\{ \max_{\substack{k: B_1^{(j,k)}=1 \\ w \in \{w_{m_1}, w_{m_2}\}}} (\tilde{N}^{(j,k)}(w) + 1), \eta_m \right\}. \tag{3.35}$$

Remark Simulation results show that the maximum number of steps to pass threshold w_{m_1} is much larger than the maximum number to pass threshold w_{m_2} in MSB page programming.

This is mainly because the variance σ_0^2 is much larger than the variance of other states in the flash. This fact helps us to simplify equation (3.35) and use the results of equation (3.31). Thus considering only w_{m_1} according to Proposition 5, to guarantee α , choose η_m to be the smallest integer that satisfies

$$Q\left(\frac{w_{m_1} - \mu_0 - \eta_m \mu_\Delta}{\sigma_\Delta \sqrt{\zeta + \eta_m}}\right) \geq 1 - \alpha + \delta_{\zeta + \eta_m}.$$

3.5.3 Confidence Interval for η

Inequality (3.32) is useful to set the maximum number of ISPP steps (η) when the allowed undershoot error rate α is known. Note that α is the probability that a single cell does not reach its target threshold w after η steps, and in practice α is not part of the device specification; Instead, it is desirable to figure out the relation between η and the *page quality loss* factor (which is typically a design criterion).

Definition 3 (Page quality loss). *Let m denote the size of a page. The 100 q % denotes the page quality loss, where*

$$q \triangleq \frac{\# \text{ of bit errors}}{m}$$

Theorem 4 (Confidence interval for η). *Let η be the maximum number of steps in ISPP page programming. Given the page quality loss 100 q %, an approximate 100(1 - β)% confidence interval for η is the set of integers that satisfy*

$$Q\left(\frac{w - \mu_0 - \eta \mu_\Delta}{\sigma_\Delta \sqrt{\zeta + \eta}}\right) \geq 1 - 2q + \left(2\mathcal{Z}_{\beta/2} \sqrt{\frac{q \cdot (1 - q)}{m}}\right) + \delta_{\zeta + \eta},$$

where $\mathcal{Z}_y = Q^{-1}(y)$ and $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt$.

Proof : Let N_E be the number of undershoot errors after programming a page. Note that N_E is a binomial random variable with parameter m and $p = \alpha$ (i.e., $N_E \sim B(m, \alpha)$). We

assume the undershoot error probability α is unknown. Then, an approximate $100(1 - \beta)\%$ confidence interval for α is the set [51, Theorem 5.3.1].

$$\left[q - \mathcal{Z}_{\beta/2} \sqrt{\frac{q \cdot (1 - q)}{m}} \quad , \quad q + \mathcal{Z}_{\beta/2} \sqrt{\frac{q \cdot (1 - q)}{m}} \right].$$

Recall $\tilde{N}(w) + 1$ is the number of steps needed for a cell to pass the threshold w if the target bit is $B = 1$. Let \mathcal{E}'_η be the event that a cell does not reach its target voltage w after η steps. Then,

$$\begin{aligned} P\{\mathcal{E}'_\eta\} &= P(B = 1, \tilde{N}(w) + 1 > \eta) \\ &\stackrel{(a)}{=} P(B = 1) \cdot P(\tilde{N}(w) \geq \eta), \end{aligned} \quad (3.36)$$

where (a) is because the input distribution is assumed to be i.i.d. and independent of the programming process. In order to guarantee the programming with quality loss under $100q\%$, it is clear that

$$P\{\mathcal{E}'_\eta\} \leq \left(q - \mathcal{Z}_{\beta/2} \sqrt{\frac{q \cdot (1 - q)}{m}} \right) \quad (3.37)$$

Since we assumed that $P(B = 1) = 1/2$, to guarantee (3.37), η must be chosen such that

$$P\{\tilde{N}(w) \geq \eta\} \leq 2 \left(q - \mathcal{Z}_{\beta/2} \sqrt{\frac{q \cdot (1 - q)}{m}} \right). \quad (3.38)$$

As discussed the actual probability $P\{\tilde{N}(w) \geq \eta\}$ is unknown. Using Corollary 3 it is straightforward to show that

$$P\{\tilde{N}(w) \geq \eta\} \leq 1 - Q \left(\frac{w - \mu_0 - n\mu_\Delta}{\sigma_\Delta \sqrt{\zeta + n}} \right) + \delta_{\eta+\zeta}. \quad (3.39)$$

Hence, in order to satisfy (3.37), η must be chosen to satisfy

$$1 - Q \left(\frac{w - \mu_0 - \eta\mu_\Delta}{\sigma_\Delta \sqrt{\zeta + \eta}} \right) + \delta_{\eta+\zeta} \leq 2 \left(q - \mathcal{Z}_{\beta/2} \sqrt{\frac{q \cdot (1 - q)}{m}} \right) \quad (3.40)$$

i.e., η should belong to the set of integers which satisfy (3.40).

3.6 Step Size Design

As Jiang et al. mentioned in [5], finding the trade-off between programming precision and the total programming latency is considered to be one of the fundamental problem of monotonic memory channels. In general, the worst-case write latency is proportional to the maximum number of steps η . Hence, we measure the write latency by η . Note that if the ISPP overshoot were the only source of noise in the write process, the variance of the overshoot $\tilde{\Gamma}(w)$ would be a valid measure for write accuracy. However, due to the presence of other sources of noise (such as the undershoot, ICI and P/E cycling) in MLC flash, we use the mutual information between the target input X and the recovered output Y as the overall accuracy measure.

In this section, we conduct a study that attempts to optimize the write latency measure provided that the information rate is larger than a given rate R . We use an adaptive approach to design the step size such that the flash life-time increases. To increase the life-time, we use larger step size ΔV_i at the early flash age and use smaller step size as the device ages.

3.6.1 Latency Distortion Theory

In this section, we assume that the MLC flash is in its early life. Thus, the P/E cycling number is very small and it is possible to disregard the wear-out noise. We are interested in obtaining the best step size ΔV_i that minimizes the latency while a particular information rate is achieved. Here we assume that there exists a maximum bound b_{\max} for the step size ΔV . It is clear that the step size can not be considered unbounded. In order to set a proper b_{\max} , it is necessary to take into account the number of voltage levels in the cell and the

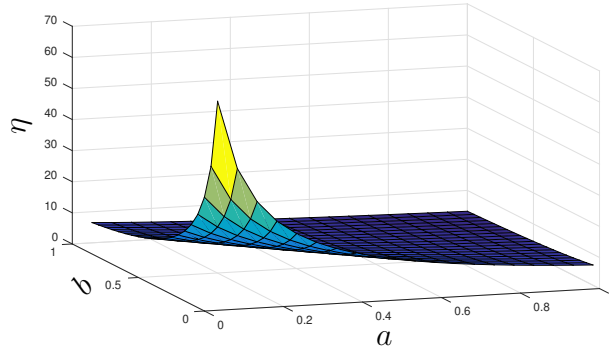


Figure 3.10: The maximum number of steps (η) when the allowed undershoot error rate is $\alpha = 0.005$.

required error margin for other degradation sources such as ICI and wear-out noise. Given that b_{\max} is known, the problem is formulated as follows:

$$\begin{aligned}
 [a^*(R), b^*(R)] &= \arg \min_{a,b} \eta \\
 &\text{such that } I(X;Y) \geq R \\
 &0 < a \leq b \leq b_{\max}
 \end{aligned} \tag{3.41}$$

Note that it is very difficult to analytically solve the above optimization problem because of the enormous number of existing noise sources which make the problem extremely complex. We show a numerical procedure to obtain the best step size $\Delta V \sim U[a^*, b^*]$ in the following example.

Example 1. Given $V_0 \sim \mathcal{N}(-1, 0.3^2)$, $w \in \{1.0, 2.0, 3.0\}$, $b_{\max} = 1V$, Figure 3.10 shows the write latency η for all possible $\Delta V_i \sim U[a, b]$. We compute η from (3.32) for $\alpha = 0.005$. It is clear that η is a decreasing function of a and b . Therefore, the optimal step size belongs to the boundary of the feasible set. Figures 3.11 shows the step size regions for various η given that the information rate $I(X, Y) \geq 1.9$ is attained for both LSB page and MSB page. The solid line shows the feasible set, and the dashed lines show the contour set for all possible $4 \leq \eta \leq 30$. As shown, if η is set to be large, it is possible to use smaller step sizes. To

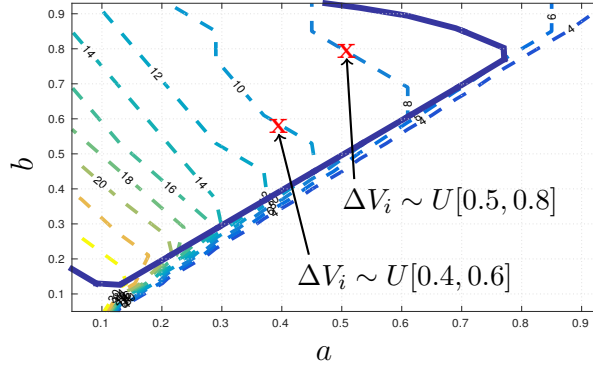


Figure 3.11: The contour graph of $I(X, Y) \geq 1.9$ (solid line) versus step size parameters a, b and the contour of η (dashed line) versus step size parameters a, b .

give concrete examples, as shown in Figure 3.11 both $\Delta V_i \sim [0.5, 0.7]$ and $\Delta V_i \sim [0.4, 0.6]$ are valid step size to attain the desired rate $R \geq 1.9$ with $\eta = 8$ and $\eta = 10$, respectively.

3.6.2 Adaptive Regulation of Step Size

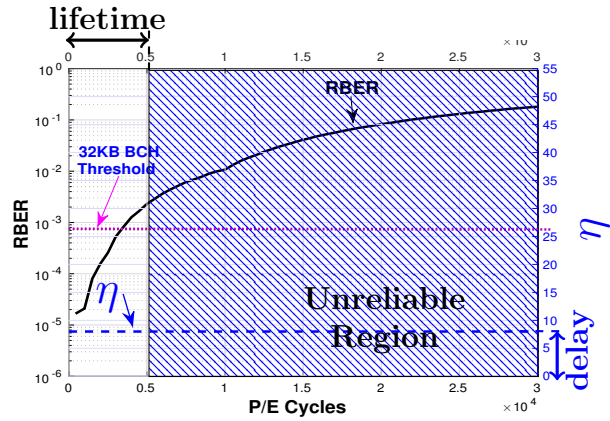
The lifetime of the flash memory for a specific ECC is the minimum P/E cycling number before the ECC fails to guarantee un-correctable bit error rate (UBER) $< 10^{-15}$ (the acceptable reliability in flash storage industry) for a certain retention period. We analyze the effect of step size change on the life time of a flash memory for a fixed storage time. As explained in [52], P/E cycles widen the final voltage distribution and move the mean to the right. Simulation results in [53] show that an exponential random variable can model the characteristics of the P/E cycling effect with 95% accuracy.

The final state distributions in flash memories get worse as the number of program/erase (P/E) cycles increases (aging effect). Thus, in the early life of a flash, more programming errors can be corrected by ECC than at the end of its lifetime. In other words, it is possible to operate faster programming and reduce the flash write latency as long as the writing error can be handled via the flash ECC module.

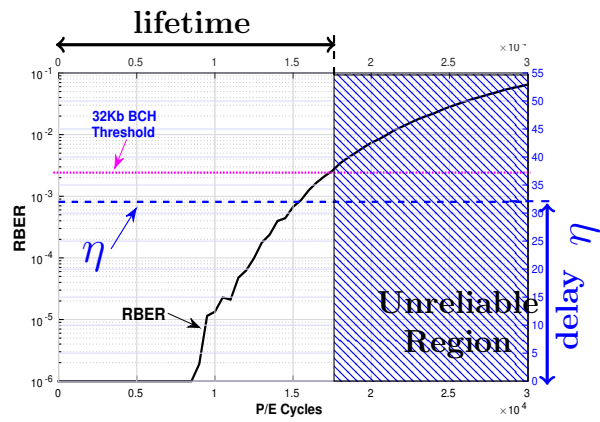
As a result, the channel is time-variant. Namely, a flash memory is programmed faster with

higher acceptable error rates early in the product’s life cycle versus late in the life cycle. In other words, it is possible to use a large step size ΔV_i at the early stage of the flash lifetime and attain the sufficient information rate in fewer programming steps. As the flash ages, however, smaller and more accurate step sizes ΔV_i are needed to still be able to attain the target information rate. As long as the error correction code (ECC) can handle the relaxed distributions (either due to larger step size at the early age of the flash or natural wear-out degradation in the later stages of the life cycle), the effect will be compensated for, and the data is fully recoverable.

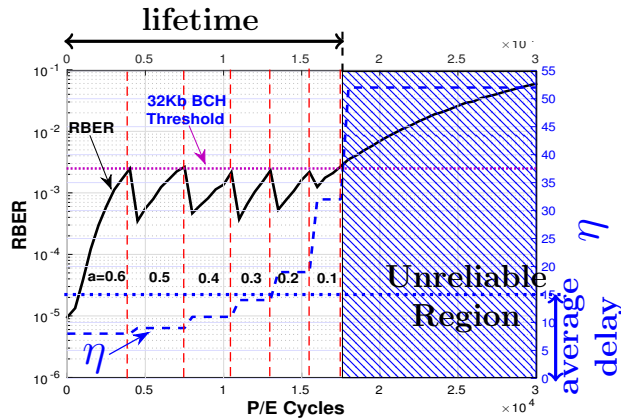
Figure 3.12a shows the raw-bit-error rate (RBER) versus P/E cycles for a 2-bit MLC flash block which was programmed with large step size $\Delta V_i \sim U[0.6, 0.7]$. As shown, the write programming is very fast ($\eta = 8(1.0X)$); however, the device becomes unreliable at small P/E cycle (RBER $\geq 2.6 * 10^{-3}$), i.e., the lifetime is short. Similarly, Figure 3.12b shows RBER versus P/E cycle when the device is programmed with a small step size $\Delta V_i \sim U[0.1, 0.2]$. As shown, the write programming is very slow ($\eta = 32(4.0X)$); however, the device becomes unreliable at large P/E cycle (RBER $\geq 2.6 * 10^{-3}$) i.e., the lifetime is long. Figure 3.12c shows a compromise, namely the RBER for a 2-bit MLC flash block versus P/E cycle for various step sizes that are adaptively tuned to keep the RBER lower than the minimum RBER correctable by ECC (Black curve). Moreover, the blue curve shows the corresponding latency η which gradually increases as flash P/E cycle grows. We consider a simple 32 Kbit BCH code which needs RBER $\leq 2.6 \times 10^{-3}$ (horizontal dashed red line). To simplify the step size design, we assume that $b - a = 0.1V$ ($\Delta V \sim U[a, b]$). Thus, we start with a large step size ($a = 0.6$) and as long as the obtained RBER is under the desired threshold, the flash block gets programed and erased immediately. Note that the step size $a = 0.6V$ is valid to be used in ISPP as long as the P/E Cycle $< 4K$. The corresponding maximum number of steps is $\eta = 8$. As shown in Figure 3.12, when P/E cycle passes $4K$, we reduce the step size to $a = 0.5$, and thereby, the RBER is maintained under the threshold. By tuning the step size properly, the lifetime of the flash can be extended to P/E Cycle= $17K$ (when $a = 0.1$). Note that the adaptive tuning of step size only gradually increases the delay, thereby extending



(a) Large step size $\Delta V_i \sim U[0.6, 0.7]$ resulting in low delay, but short lifetime.



(b) Small step size $\Delta V_i \sim U[0.1, 0.2]$ resulting in large lifetime, but high delay.



(c) Adaptive step size $\Delta V_i \sim U[a, b]$ when $a \in [0.1, 0.6]$ and $b - a = 0.1$, resulting in large lifetime and moderate delays for most of the lifetime.

Figure 3.12: RBER and Latency versus PEC.

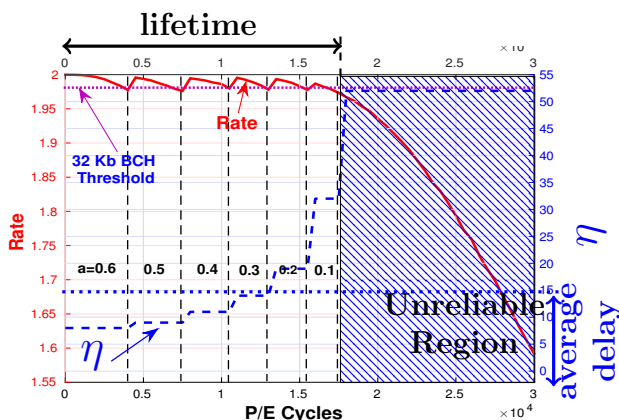


Figure 3.13: Rate and Latency versus P/E Cycles, adaptive step size $\Delta V_i \sim U[a, b]$ when $a \in [0.1, 0.6]$ and $b - a = 0.1$.

the lifetime of the device. As shown, in Figure 3.12, the maximum step size increases from $\eta = 8$ for $a = 0.6V$ to $\eta = 32$ when $a = 0.1$, but the average delay η over the lifetime is relatively small $\bar{\eta} = 15$.

Figure 3.13 illustrates the discussed trade-off between information rate and latency as the flash P/E cycle grows. The solid red curve shows the mutual information $I(X; Y)$ versus P/E cycle, and the blue solid curve shows the corresponding write latency η versus P/E cycle. Note that by carefully tuning the step size, it is possible to maintain the target information rate at the cost of gradually increasing the write latency at higher P/E cycles (i.e., at older age of the device).

4

Optimal Detector Design

The MLC flash memory system can be represented as a concatenation of 4 components: ECC encoder, channel, detector and ECC decoder (as depicted in Fig. 4.1). In this Chapter, we focus on the detector (the shaded block in Fig. 4.1). Motivated by [15], we focus on designing the detector for MLC flash memory in order to improve the hard decision bit-error-rate if possible. In addition, we would like to improve the soft decision quality of the detector if possible. Furthermore, designing the optimal soft and hard detector provides a benchmark which all other (sub-optimal) detector would be compared to. The optimal detector design also helps to derive closed form expression for the optimal decision making strategy in order to gain insight (such as sufficient statistics) and understand the interplay between channel parameters. Finally, we can use the attained insights to guide the derivation

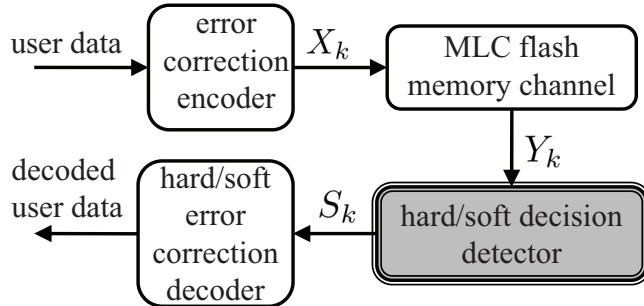


Figure 4.1: A simple MLC flash memory system block diagram.

of novel low-complexity suboptimal detector.

We first provide channel models, including the one-dimensional (1D) model with *causal output memory* and the two-dimensional (2D) anti-causal model of the MLC flash memory, as shown in Sec. 2.2.1. Second, in Sec. 4.1, we present a mathematically tractable Viterbi-like *maximum a posteriori* (MAP) sequence detector for the 1D causal model with output memory. The exact statistics of the channel model necessary for implementing the MAP detector can be obtained by using the fast Fourier transform (FFT). Third, we introduce a simplified Gaussian approximation (GA) sequence detector at the expense of reduced performance, which is shown in Sec. 4.2. Both the MAP detector and the GA detector can be employed in the 2D anti-causal flash memory channel. Fourth, in Sec. 4.3, we extend the channel model and detector design to more general scenarios including those with signal-dependent noise, input intersymbol interference, and 2D Markov channel inputs. Fifth, in Sec. 4.4, we utilize simulation results to show that the MAP detector outperforms the existing detectors in the literature. Finally, we conclude this work in Sec. 4.5.

4.1 Viterbi-like 1D Sequence Detection

We denote the sequence of random variables (X_1, X_2, \dots, X_n) of length n by X_1^n . The realization sequence (x_1, x_2, \dots, x_n) is denoted by x_1^n . The set of all possible realizations of the random sequence X_1^n is denoted by \mathcal{X}^n .

The MAP sequence detector of the state sequence x_1^n is the sequence \hat{x}_1^n that maximizes the joint conditional pdf, i.e.,

$$\hat{x}_1^n = \arg \max_{x_1^n \in \mathcal{X}^n} f(x_1^n, y_1^n | x_{1-M}^0, y_{1-L}^0), \quad (4.1)$$

where M and L are the order of the Markov input process and the output memory, respectively.

As shorthand, denote $f(\underline{x}, \underline{y} | \text{i.c.})$ as the conditional pdf of the right hand side of (4.1), where i.c. stands for the initial condition (x_{1-M}^0, y_{1-L}^0) . We start by factoring the pdf in (4.1) as

$$\begin{aligned} f(\underline{x}, \underline{y} | \text{i.c.}) &= f(x_1^n, y_1^n | x_{1-M}^0, y_{1-L}^0) \\ &= P(x_1^n | x_{1-M}^0, y_{1-L}^0) f(y_1^n | x_{1-M}^n, y_{1-L}^0) \\ &= \prod_{k=1}^n P(x_k | x_{k-M}^{k-1}) f(y_k | x_k, y_{k-L}^{k-1}). \end{aligned}$$

Consequently, the MAP detected sequence is equal to

$$\hat{x}_1^n = \arg \min_{x_1^n \in \mathcal{X}^n} \sum_{k=1}^n \underbrace{[-\ln(P(x_k | x_{k-M}^{k-1}) f(y_k | x_k, y_{k-L}^{k-1}))]}_{\Lambda_{\text{MAP}}(x_{k-M}^k, y_{k-L}^k)}. \quad (4.2)$$

The term inside the summation in (4.2) is called a *branch metric* and is denoted by $\Lambda_{\text{MAP}}(x_{k-M}^k, y_{k-L}^k)$. It is clear that evaluating the branch metric $\Lambda_{\text{MAP}}(\cdot, \cdot)$ requires evaluating the conditional pdf $f(y_k | x_k, y_{k-L}^{k-1})$ or some function there of. Obviously, the branch metric depends on $L+1$ real valued variables y_k, \dots, y_{k-L} . So, it is desired to extract a sufficient statistics from y_{k-L}^k that will allow efficient computations of branch metrics. We next derive the desired sufficient statistics.

4.1.1 Calculation of the Characteristic Function

Computing $f(y_k|x_k, y_{k-L}^{k-1})$ analytically is intractable. Instead, we calculate the conditional characteristic function of Y_k under the assumptions that $X_k = x_k$ and $Y_{k-L}^{k-1} = y_{k-L}^{k-1}$ are given. The conditional pdf $f(y_k|x_{k-M}^k, y_{k-L}^{k-1})$ for each realization y_k is then derived by taking the Fourier transform of the characteristic function.

We rewrite the channel model as

$$Y_k = \underbrace{X_k + W_k}_R + U_k + \sum_{\ell=1}^L \underbrace{\Gamma_\ell^{(k)}(Y_{k-\ell} - E_{k-\ell})}_{Z_\ell}. \quad (4.3)$$

Next, we compute the conditional characteristic function of R and Z_ℓ under the assumptions that $X_k = x_k$ and $Y_{k-L}^{k-1} = y_{k-L}^{k-1}$ are given. Note that if $X_k = x_k$ is given, R is Gaussian $\mathcal{N}(\mu_R, \sigma_R^2)$ where

$$\begin{aligned} \mu_R &= \mathbb{E}[R|X_k = x_k] = x_k \\ \sigma_R^2 &= \text{Var}[R|X_k = x_k] = \sigma_w^2 \end{aligned}$$

Hence, the conditional characteristic function of R is

$$\begin{aligned} G_{R|X_k}(t) &= \mathbb{E}[e^{iRt}|X_k = x_k] \\ &= \exp\left(-\frac{1}{2}\sigma_R^2 t^2 + i\mu_R t\right). \end{aligned} \quad (4.4)$$

Similarly, the conditional characteristic function of Z_ℓ , when $Y_{k-\ell} = y_{k-\ell}$ and $X_k = x_k$ are given, is derived in the Appendix B and denoted by $G_{Z_\ell|Y_{k-\ell}}(t)$. Finally, combining $G_{Z_\ell|Y_{k-\ell}}(t)$ and (4.4), and utilizing the conditional independence of R and Z_ℓ (given $y_{k-\ell}^{k-1}$

and x_k), we get

$$\begin{aligned}
G_{Y_k|X_k, Y_{k-L}^{k-1}}(t) & \tag{4.5} \\
&= G_{R|X_k}(t) G_{U_k}(t) \prod_{\ell=1}^L G_{Z_\ell|Y_{k-\ell}}(t) \\
&= \frac{\text{sinc}(t\Delta/2)}{\sqrt{\prod_{\ell=1}^L (1 + g_\ell \sigma_e^2 t^2)}} \exp\left(-\frac{1}{2}\sigma_R^2 t^2 + i\mu_R t + \Phi(t)\right)
\end{aligned}$$

where

$$\Phi(t) = \sum_{\ell=1}^L \frac{-t^2 [(y_{k-\ell} - \mu_e)^2 g_\ell + \gamma_\ell^2 \sigma_e^2] + 2it (y_{k-\ell} - \mu_e) \gamma_\ell}{2(1 + g_\ell \sigma_e^2 t^2)}. \tag{4.6}$$

4.1.2 FFT Implementation

Since the pdf is the Fourier transform of the characteristic function, the conditional probability $f(y_k|x_k, y_{k-L}^{k-1})$ can be obtained as

$$f(y_k|x_k, y_{k-L}^{k-1}) = \int_{-\infty}^{\infty} G_{Y_k|X_k, Y_{k-L}^{k-1}}(t) e^{-iy_k t} dt. \tag{4.7}$$

Hence, it is possible to numerically compute the branch metric $\Lambda_{\text{MAP}}(x_{k-M}^k, y_{k-L}^k)$ in (4.2) for each branch in the Viterbi trellis using the fast Fourier transform (FFT). For each trellis section, we only need to compute one FFT. In other words, the FFT is the same for all branches of the trellis section, but the actual branch metric values are obtained by sampling the FFT at different points as illustrated in Figure 4.3 below.

4.1.3 Sufficient Statistics

A look at (4.5) and (4.6) reveals that the channel outputs y_{k-L}^k need to be processed (in some way) in order to formulate the branch metrics. The processing complexity depends on the order L .

Example 2. If $L = 1$, then (4.5) and (4.6) reveal that a set of sufficient statistics necessary for the computation of branch metrics is

- (a) y_k
- (b) $\gamma_1(y_{k-1} - \mu_e)$
- (c) $g_1(y_{k-1} - \mu_e)^2$

A way to obtain Λ_{MAP} in this case could be using the lookup table in Figure 4.2 (a). \square

Example 3. If $L = 2$, then (4.6) reveals that the sufficient statistics are obtained by finite impulse response (FIR) filters. The sufficient statistics are:

- (a) y_k , and the following FIR filter outputs
- (b) $\gamma_1(y_{k-1} - \mu_e) + \gamma_2(y_{k-2} - \mu_e)$
- (c) $\gamma_1 g_2(y_{k-1} - \mu_e) + \gamma_2 g_1(y_{k-2} - \mu_e)$
- (d) $g_1(y_{k-1} - \mu_e)^2 + g_2(y_{k-2} - \mu_e)^2$
- (e) $g_1 g_2(y_{k-1} - \mu_e)^2 + g_1 g_2(y_{k-2} - \mu_e)^2$

Consequently, the branch metrics Λ_{MAP} can be computed using a lookup table in Figure 4.2 (b). \square

The complexity of implementing the lookup table grows linearly with L because we need $2L + 1$ sufficient statistics. Quantizing each sufficient statistic to, say, a 7 bit precision requires $7(2L + 1)$ binary inputs to the lookup table. An alternative is to use FFT to compute the pdf $f(y_k | x_k, y_{k-L}^{k-1})$ from the characteristic function $G_{Y_k | X_k, Y_{k-L}^{k-1}}(t)$. Thereby, we can use the sufficient statistics to compute an equivalent form of the characteristic function (4.5) with

$$\Phi(t) = \frac{\sum_{\ell=1}^{2L} C_\ell(y_{k-L}^{k-1}) t^\ell + q(t)}{2 \prod_{\ell=1}^L (1 + g_\ell \sigma_e^2 t^2)}$$

where

$$C_\ell(y_{k-L}^{k-1}) = \begin{cases} \sum_{j=1}^L \alpha_j^{(\ell)} \cdot (y_{k-j} - \mu_e), & \text{if } \ell \text{ is odd} \\ \sum_{j=1}^L \beta_j^{(\ell)} \cdot (y_{k-j} - \mu_e)^2, & \text{if } \ell \text{ is even.} \end{cases} \quad (4.8)$$

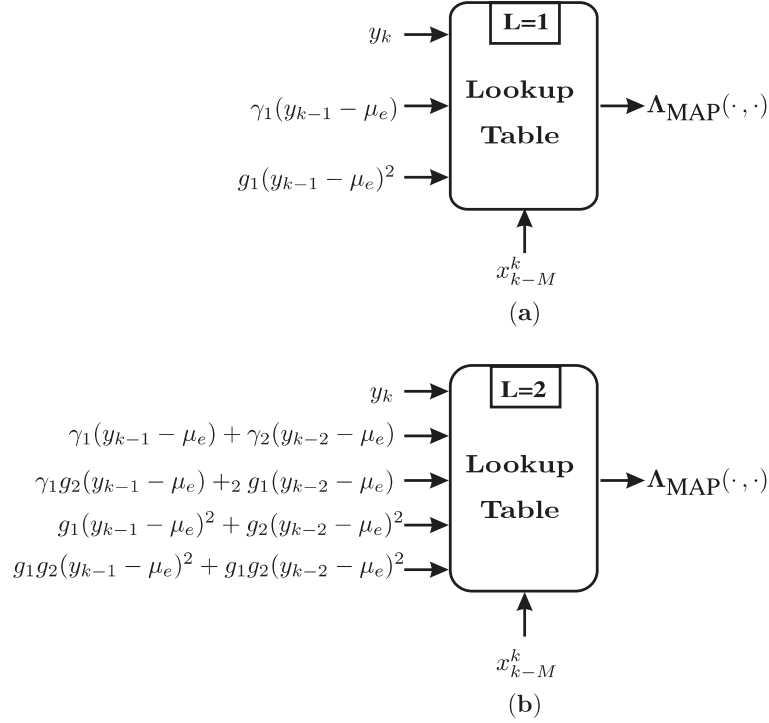


Figure 4.2: Branch metric Λ_{MAP} for cases (a) $L = 1$ and (b) $L = 2$.

and $q(t)$ is a polynomial (whose coefficients are independent of y_{k-j}^{k-1}). It is clear from (4.8) that the sufficient statistics for this computation are outputs of two types of FIR filters where:

- (a) inputs are signals $(y_k - \mu_e)$, if ℓ is odd.
- (b) inputs are signals $(y_k - \mu_e)^2$, if ℓ is even.

This approach is illustrated in Figure 4.3.

Lookup tables may be complicated to implement because of the need to quantize all the sufficient statistics. However, the purpose of this section is not to suggest that a lookup table is always a practical approach. Rather, it is to reveal what the sufficient statistics are. Knowing the sufficient statistics is important because it gives us an analytic insight into what to compute at the receiver end. For example, when $L = 2$, Fig. 4.2 (b) shows that there are 5 sufficient statistics (the 5 inputs to the lookup table from the left). This can actually guide the development of suboptimal detectors. For example, it is easy to verify

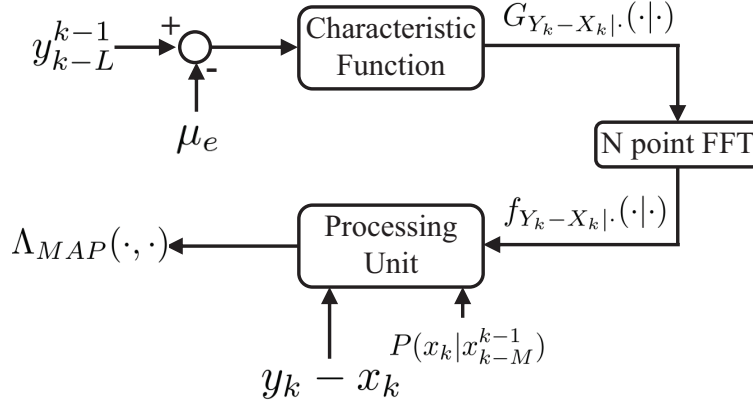


Figure 4.3: Branch metric computation using the FFT.

that if we ignore the 3rd, 4th and 5th inputs to the lookup table, we get the suboptimal post-compensation detector of Dong *et al.* [15]. In the same spirit, we will reveal in Section 4.2 how to derive another suboptimal detector (based on a Gaussian approximation) that ignores the 3rd and 5th inputs in Fig. 4.2 (b), has a closed-form expression for the branch metric and is particularly easy to implement.

4.2 Gaussian Approximation (GA) Detector

At the expense of reduced performance, as an alternative to the optimal procedure given in Sec. 4.1, we give a simplified procedure for computing an approximation of $\Lambda_{MAP}(\cdot, \cdot)$ based on a Gaussian approximation. We rewrite the channel model as

$$\begin{aligned}
 Y_k &= X_k + \sum_{\ell=1}^L \Gamma_{\ell}^{(k)} (Y_{k-\ell} - E_{k-\ell}) + W_k + U_k \\
 &= V_k + U_k.
 \end{aligned} \tag{4.9}$$

According to (2.3), V_k is obtained as the summation of several random variables. Assume that we can approximate $f(v_k | x_k, y_{k-L}^{k-1})$ by a Gaussian pdf as follows:

$$f(v_k | x_k, y_{k-L}^{k-1}) \sim \mathcal{N}(\mu_G(k), \sigma_G^2(k)), \tag{4.10}$$

where

$$\begin{aligned}\mu_G(k) &= \mathbb{E}[V_k | Y_{k-\ell}^{k-1} = y_{k-\ell}^{k-1}, X_k = x_k] \\ &= x_k + \sum_{\ell=1}^L \gamma_\ell (y_{k-\ell} - \mu_e)\end{aligned}\tag{4.11}$$

$$\begin{aligned}\sigma_G^2(k) &= \text{Var}[V_k | Y_{k-\ell}^{k-1} = y_{k-\ell}^{k-1}, X_k = x_k] \\ &= \sum_{\ell=1}^L (g_\ell (\sigma_e^2 + (y_{k-\ell} - \mu_e)^2) + \sigma_e^2 \gamma_\ell^2) + \sigma_w^2.\end{aligned}\tag{4.12}$$

The new approximate conditional distribution $f^{(G)}(y_k | x_k, y_{k-L}^{k-1})$, is obtained by convolving the Gaussian distribution $\mathcal{N}(\mu_G(k), \sigma_G^2(k))$ and the uniform distribution $\mathcal{U}(-\Delta/2, \Delta/2)$. That is,

$$\begin{aligned}f^{(G)}(y_k | x_k, y_{k-L}^{k-1}) &= \int_{y_k - \frac{\Delta}{2}}^{y_k + \frac{\Delta}{2}} \frac{1}{\sqrt{2\pi}\sigma_G\Delta} e^{-\frac{(v_k - \mu_G)^2}{2\sigma_G^2}} dv_k \\ &= \frac{1}{\Delta} \left[Q\left(\frac{y_k - \mu_G - \frac{\Delta}{2}}{\sigma_G}\right) - Q\left(\frac{y_k - \mu_G + \frac{\Delta}{2}}{\sigma_G}\right) \right]\end{aligned}$$

where the standard Q -function is defined as $Q(\zeta) = \frac{1}{\sqrt{2\pi}} \int_{\zeta}^{\infty} \exp(-\frac{\eta^2}{2}) d\eta$.

So, clearly by examining (4.9)-(4.12), we conclude that to compute the branch metrics $\Lambda^{(G)}(\cdot, \cdot)$, we need the following subset of the sufficient statistics

$$\begin{aligned}y_k \\ \theta_k &= \sum_{\ell=1}^L \gamma_\ell (y_{k-\ell} - \mu_e) \\ \nu_k &= \sum_{\ell=1}^L g_\ell (y_{k-\ell} - \mu_e)^2,\end{aligned}$$

which actually coincide with the three sufficient statistics a), b) and d) in Example 3

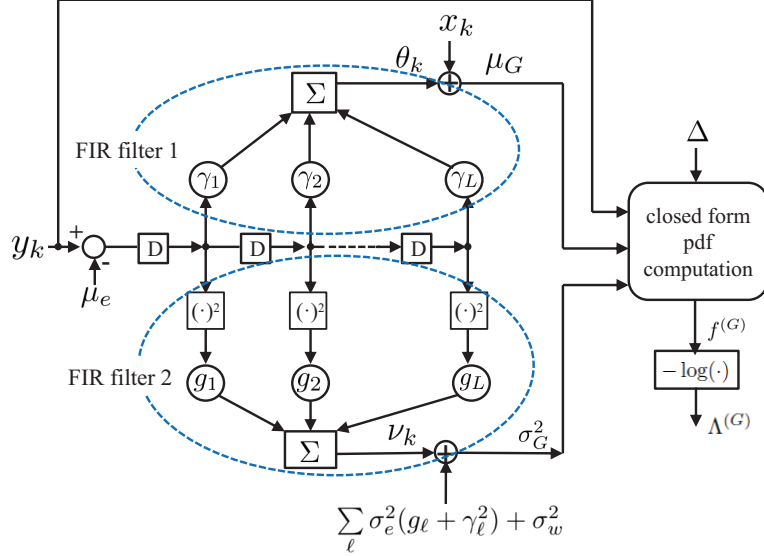


Figure 4.4: The branch metric computation module of the GA detector using FIR filters.

(Fig. 4.2 (b)). Hence, the computation of $\Lambda^{(G)}(x_{k-M}^k, y_{k-L}^k)$ is equivalent to computing $\Lambda^{(G)}(x_{k-M}^k, y_k, \theta_k, \nu_k)$. Note, again, that θ_k and ν_k are obtained by FIR-filtering $(y_{k-\ell} - \mu_e)$ and $(y_{k-\ell} - \mu_e)^2$, respectively. Thereby, the entire set of sufficient statistics can be replaced by a new vector $[y_k, \theta_k, \nu_k]$ of only three components (even if $L > 1$). Furthermore, the actual computation of $\Lambda^{(G)}(x_{k-M}^k, y_k, \theta_k, \nu_k)$ does not require generating lookup tables or FFTs, but can be implemented using sample DSP components such as multipliers and adders. Fig. 4.4 illustrates the branch metric computation module of the suboptimal GA detector.

It is interesting to note that some prior-art detectors, such as the hard-decision detectors in [15, 54], and the soft-decision detectors in [12, 55] can be obtained by further approximations of the Gaussian-approximation detector given in this section. In particular, the hard decision detector (post-compensation detector) in [15] can be obtained as a symbol-by-symbol detector (when X_k is i.i.d.) by computing the decision variables $y_k - \mu_G(k)$ and heuristically determining the decision thresholds to achieve minimum probability of symbol error. The detector in [54] is very similar to [15] in the way that decision variables are determined, but in [54] a heuristically-derived Viterbi detector (obtained by exhaustive target training) is used to refine the detection process. Similarly, the soft-output detector in [12, 55] can be

obtained if instead of $\sigma_G^2(k)$ in (4.12), we use $\sigma_e^2\gamma_1^2$. Consequently, the detectors in [12, 55] are suboptimal versions of the detector proposed here (particularly, if used as precursors to a soft-in-soft-out decoder for LDPC codes) as shown explicitly in Sec. 4.4.

Finally, we note that in [15], a separate predistortion detector and a separate post-compensation detector were proposed, showing that each has some advantages. The detector in this work combines the strengths of both strategies *simultaneously*. This can be achieved by shaping the channel input process X_k into a Markov process (2.4) such as in [56, 57] (which is akin to predistortion in [15]) and subsequently using a Viterbi/BCJR trellis detector as in Secs. 4.1 and 4.2 to detect the Markov input process X_k (akin to the post-compensation detector in [15]).

4.3 Extensions

In this section, we briefly explain how to extend the channel model and detector design when the channel suffers from additional degradations.

4.3.1 Signal-Dependent Noise

If the noise in the channel is signal-dependent, the channel model and the detector must be appropriately altered. Several studies [46, 58] have shown that different levels v_0, v_1, \dots, v_{m-1} give rise to different statistics of the channel noise W_k and U_k (see model in (2.3)). For example, it is possible that the random variables W_k and U_k depend on the realization of the channel input random variable $X_k = v_j$. In that case, we model W_k and U_k to be signal-dependent,

$$\begin{aligned} f_{W_k|X_k}(\cdot|X_k = v_j) &\sim \mathcal{N}(0, \sigma_w^2(j)) \\ f_{U_k|X_k}(\cdot|X_k = v_j) &\sim \mathcal{U}\left(\frac{-\Delta(j)}{2}, \frac{\Delta(j)}{2}\right) \end{aligned}$$

In other words, the parameters σ_w^2 and Δ are appropriately substituted by $\sigma_w^2(j)$ and $\Delta(j)$, depending on the (postulated) realization of the random variable $X_k = v_j$.

Similarly, we may also extend the model of the random fading-like coefficient $\Gamma_\ell^{(k)}$ to be signal-dependent. This means that instead of assuming $\mathbb{E}[\Gamma_\ell^{(k)}] = \gamma_\ell$ and $\text{Var}[\Gamma_\ell^{(k)}] = g_\ell$, we would assume a signal-dependent model $\mathbb{E}[\Gamma_\ell^{(k)} | X_{k-\ell} = v_j] = \gamma_\ell(j)$ and $\text{Var}(\Gamma_\ell^{(k)} | X_{k-\ell} = v_j) = g_\ell(j)$. These changes would appropriately alter the Viterbi/BCJR detectors.

4.3.2 Input Intersymbol Interference

The channel model in (2.3) assumed only output ICI, but no input intersymbol interference (ISI). We can alter the model in (2.3) to account for input ISI as follows:

$$Y_k = \sum_{m=0}^M A_m^{(k)} X_{k-m} + \sum_{\ell=1}^L \Gamma_\ell^{(k)} (Y_{k-\ell} - E_{k-\ell}) + W_k + U_k \quad (4.13)$$

where $A_m^{(k)}$ are either constant coefficients or random variables, say $A_m^{(k)} \sim \mathcal{N}(a(m), \sigma_a^2(m))$. In either case, the optimal detector design is still a Viterbi-like or a BCJR-like detector whose branch metrics $\Lambda_{\text{MAP}}(x_{k-M}^k, y_{k-L}^k)$ can be determined using the FFT of the appropriate characteristic function, or an appropriate Gaussian approximation. The model in (4.13) can also be extended to be signal-dependent and/or 2D. We omit the details.

4.3.3 2D Channels

When the 2D channel model in (2.7) is appropriate, and the channel input is i.i.d, the optimal detector is a simple symbol-by-symbol detector. However, since the channel does have 2D-memory, it is reasonable to expect that the information-theoretically optimal channel input process $X_{(k,\ell)}$ will not be i.i.d. In this case we have two complications for which exact solutions are not known, and we likely need to resort to ad-hoc and/or heuristic approaches:

- (a) Optimizing the input distribution (even under the 2D Markov input assumption) to

maximize the information rate of a 2D channel with memory is not known. To date, only a limited number of computational methods to evaluate information rates of 2D channels with memory are known [59–62], but to the best of our knowledge, no 2D information-rate optimization techniques are available. One approach may be to heuristically adapt 1D techniques to optimize lower-bounds on 2D information rates as in [63], but this is certainly subject the further research.

- (b) Even if an appropriate 2D Markov process $X_{(k,\ell)}$ could be constructed to guarantee a nearly optimal information rate, the optimal detector for a 2D channel is not available (because there exists no equivalents of the Viterbi/BCJR detectors in 2D). A plausible solution is to apply 1D methods in some heuristic fashion (such as, for example, interleaving vertical and horizontal detectors [47, 64–66], or combining 1D horizontal detectors with 1D vertical decision feedback [67–69]), or to design entirely new 2D detectors [70–72] (which is, of course, subject to further research).

From points (a) and (b) above, it is clear that to achieve an information-theoretically optimal transmission/reception strategy further research on 2D capacity computing techniques and 2D detection/interleaving techniques is needed.

4.4 Simulation Results and Discussion

In this section, we give simulation results to show the performances of the detectors when using an even/odd bit-line structure. We use a 4-level flash memory channel, where the channel input X_k is an i.i.d. process with parameters $\Pr(X_k = v_j) = 0.25$ for any of the 4 levels v_0, v_1, v_2 or v_3 . The parameters of the 4-level flash memory (2D channel) with signal-dependent noise are given in Table 4.1. With the parameters as in Table 4.1, and using $\sigma = 1$, Fig. 4.5 depicts the pdf of each level’s voltage when no ICI occurs.

We next assume that the random coupling ratios $\Gamma_{(a,b)}^{(k,\ell)}$ (see (2.7)) have the following Gaussian

Table 4.1: Parameters of the 4-level flash memory

i	0	1	2	3
i th level v_i	1.1	2.7	3.3	3.9
$\Delta(i)$	0	0.3	0.3	0.3
$\sigma_w(i)$	0.35σ	0.03σ	0.03σ	0.03σ

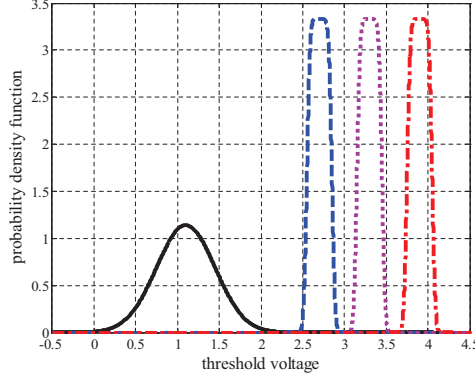


Figure 4.5: The pdf of each level's voltage for the 4-level flash memory without ICI.

distributions

$$\begin{aligned}
 \Gamma_{(k,\ell-1)}^{(k,\ell)} &\sim \mathcal{N}(\gamma_h, g_h), & \Gamma_{(k,\ell+1)}^{(k,\ell)} &\sim \mathcal{N}(\gamma_h, g_h), \\
 \Gamma_{(k+1,\ell-1)}^{(k,\ell)} &\sim \mathcal{N}(\gamma_d, g_d), & \Gamma_{(k+1,\ell+1)}^{(k,\ell)} &\sim \mathcal{N}(\gamma_d, g_d) \\
 \Gamma_{(k+1,\ell)}^{(k,\ell)} &\sim \mathcal{N}(\gamma_v, g_v),
 \end{aligned} \tag{4.14}$$

where the subscripts h , v and d mean horizontal, vertical and diagonal interference, respectively. We also assume that¹ $\gamma_h : \gamma_v : \gamma_d = 0.1 : 0.08 : 0.006$ and $g_i = 0.09\gamma_i^2$ for $i \in \{h, v, d\}$ as introduced in [15] and the references therein. Let s be the *intercell coupling strength factor*. Then $\gamma_h = 0.1s$, $\gamma_v = 0.08s$ and $\gamma_d = 0.006s$.

In the first simulation scenario, we fix $\sigma = 1$ (see Table 4.1) and we let the coupling strength factor s vary from 0 to 2. The bit-error-rate (BER) performances of the MAP detector and the GA detector are shown in Fig. 4.6. In Fig. 4.6, we also show the BER performances of the post-compensation detector [15] and the raw detector [15].

In the second simulation scenario, we fix $s = 0.75$, and vary the parameter σ (see Table 4.1).

1. These notations denote the relative magnitudes of horizontal, vertical and diagonal capacitance couplings.

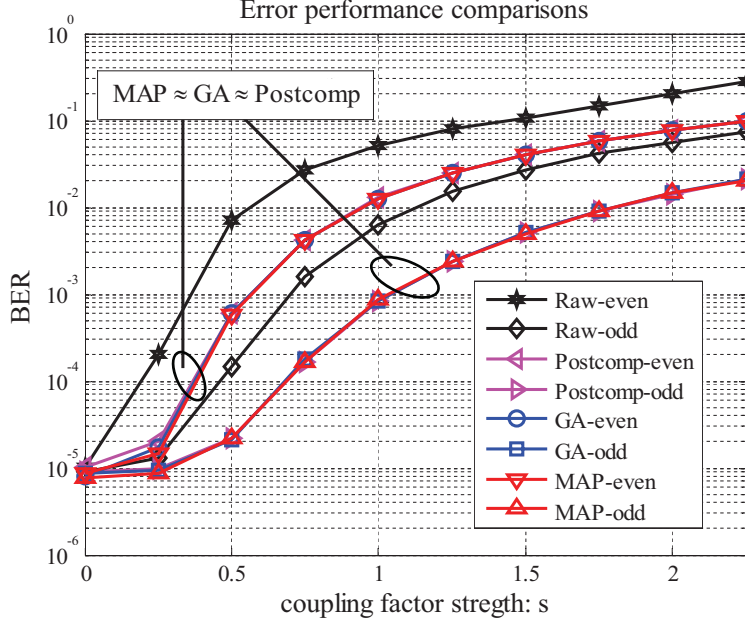


Figure 4.6: BER comparisons for different detectors when the coupling factor strength is varying and $\sigma = 1$.

By varying σ , we effectively vary the signal-to-noise ratio (SNR), defined as

$$\text{SNR} \triangleq \frac{1}{\sum_i \Pr(X_k = v_i) \sigma_w^2(i)}. \quad (4.15)$$

The BER curves for varying SNRs are shown in Fig. 4.7, depicting the performances of the MAP detector, the GA detector, the post-compensation detector [15] and the raw detector [15].

Figs. 4.6 and 4.7 reveal that if the BER is the figure of merit, neither the MAP detector nor the GA detector outperforms the post-compensation detector (originally disclosed in [15]). Hence, to get a better sense of the quality of each detector, we must compare the qualities of their *soft* outputs. Here, we measure the quality of a detector's soft output as follows. Let X_k be the i.i.d. equiprobable channel input, meaning that $P(X_k = v_j) = \frac{1}{m}$. In the case of

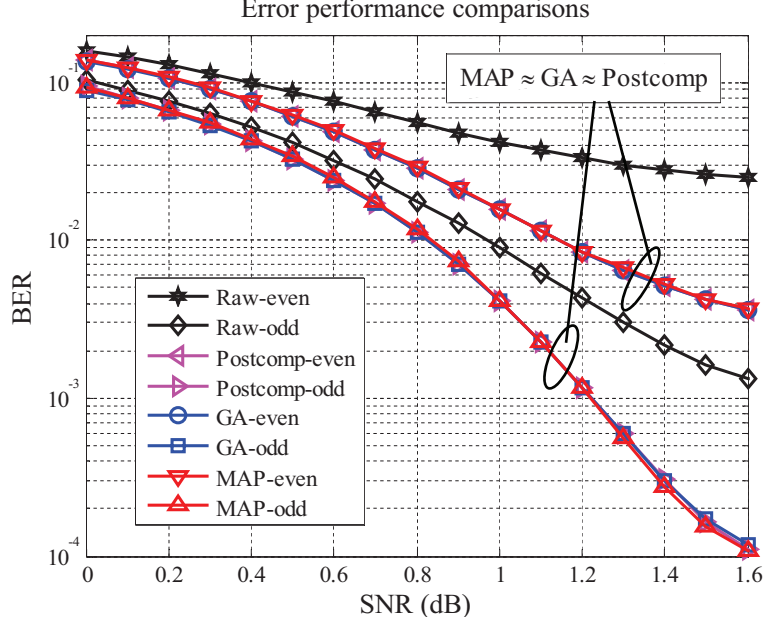


Figure 4.7: BER comparisons for different detectors when the SNR is varying and the coupling factor strength is fixed at $s = 0.75$.

a soft decision detector, the detector output S_k is a vector defined as²

$$S_k = \begin{bmatrix} P(X_k = v_0 | Y_1^n = y_1^n) \\ P(X_k = v_1 | Y_1^n = y_1^n) \\ \vdots \\ P(X_k = v_{m-1} | Y_1^n = y_1^n) \end{bmatrix}$$

and in the case of a hard detector, the detector output S_k is a scalar estimate of the channel input

$$S_k = \hat{X}_k \in \{v_0, v_1, \dots, v_{m-1}\}.$$

We define the *soft information quality* (SIQ) of a detector as

$$q = \frac{1}{2} \left[I(X_k; S_k)|_{\text{even } k} + I(X_k; S_k)|_{\text{odd } k} \right]. \quad (4.16)$$

2. Obviously, in a multilevel flash memory channel, the variable S_k is a collection of m likelihood values - one likelihood value for each level.

As explained in [73], SIQ is the capacity of random linear block codes. Therefore, this quantity is proved to be the highest information rate achievable by a random low-density parity-check (LDPC) error correction code. Furthermore, the SIQ allows us to compare performances of codes without going through the complicated task of simulating the actual codes. For example, if SIQ of detector A is 0.5 dB better than SIQ of detector B , then a random LDPC code using outputs from detector A will outperform the same random LDPC code using outputs from detector B by 0.5 dB. In other words, if we use detector A , we can afford to use a 0.5 dB weaker code and achieve the same overall system performance.

The mutual information terms in (4.16) can be readily computed numerically using Monte-Carlo simulations for any detector (also for a hard-decision detector). For the special case of a MAP detector, the soft-information quality q_{MAP} has an alternative interpretation, i.e., q_{MAP} is equal to the so-called *BCJR-once bound* (see [73] for details). Fig. 4.8 shows the soft information qualities of the MAP detector and the GA detector when the coupling strength factor s varies for fixed SNR, while Fig. 4.9 shows the soft information quality curves when the SNR varies for fixed $s = 0.75$. Also shown in Figs. 4.8 and 4.9 are soft information qualities of the post-compensation detector [15] and the raw detector [15]. Finally, the figures also show an upper bound on the soft information quality of the soft-output detector presented in [12], denoted by q_{Dong}^* . At SIQ = 1.8 bits per cell (which corresponds to a code rate of 0.9 user bits per channel bit), the MAP detector outperforms known detectors by 0.35 dB, as shown in Fig. 4.9.

As we show in Fig. 4.9, the performance of MAP detector is significantly better than current detectors. Table 4.2 compares the computational penalty of each detector. In Table 4.2, the “read process” penalty stands for the required number of reads per written symbol, and the “metric computation” penalty is the computational complexity of computing the branch metrics per written symbol. Note that the variable N in the MAP detector is the number of quantization points in computing the FFT (i.e, N is the support length of the FFT)³.

3. In all our simulations, we used $N = 512$.

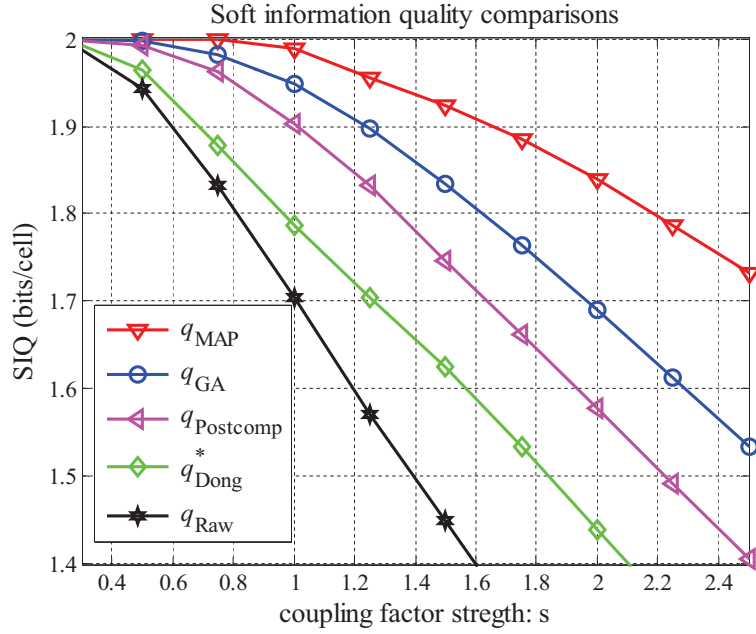


Figure 4.8: SIQ comparisons for different detectors when the coupling factor strength is varying and $\sigma = 1$.

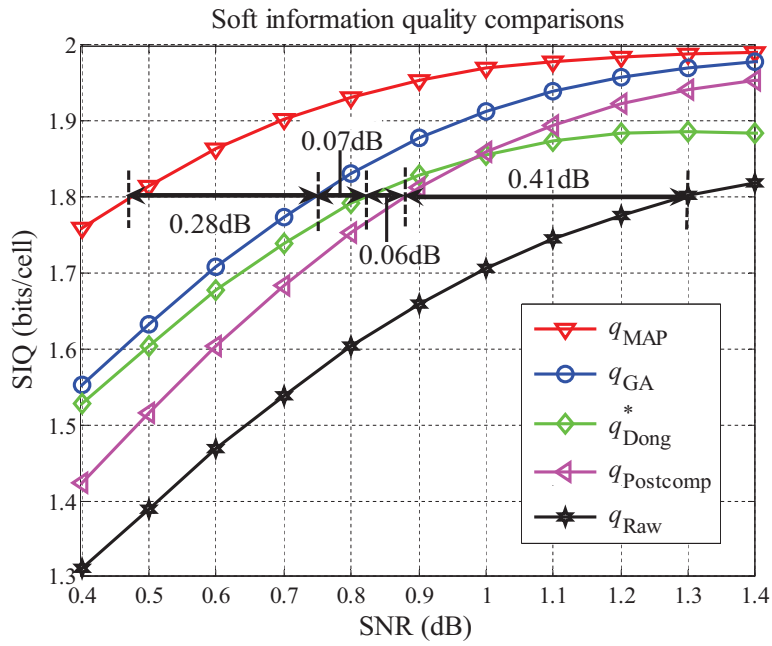


Figure 4.9: SIQ comparisons for different detectors when the SNR is varying and the coupling factor strength is fixed at $s = 0.75$.

Table 4.2: Complexity comparison for different detectors

Detector	Read Process	Metric Computation
No ICI (threshold detector)	$O(1)$	$O(1)$
Post-compensation [15]	$O(L)$	$O(L)$
Gaussian Approximation	$O(L)$	$O(L)$
MAP(FFT-based)	$O(L)$	$O(L)+$ $O(N\log N)$

4.5 Conclusion

We derived the optimal detector structure for multilevel cell (MLC) NAND flash memory channels. The optimal detector is attainable using FFTs of analytically computable characteristic functions. Alternatively, at a small performance loss, a Gaussian-approximation detector is attainable using two FIR filters, i) the first operating on channel outputs, and ii) the second operating on the squares of the channel outputs. We derived the optimal detectors here for both 1D and 2D page-oriented channels and demonstrated their superior performances (particularly if executed as soft-output detectors) through simulations.

5

Markov Channels

5.1 Preliminaries

5.1.1 Alphabet and strings

Most notation here is standard, we include them for completeness. \mathcal{A} is a finite alphabet with cardinality $|\mathcal{A}|$, $\mathcal{A}^* = \bigcup_{k \geq 0} \mathcal{A}^k$ and \mathcal{A}^∞ denotes the set of all semi-infinite strings of symbols in \mathcal{A} .

We denote the length of a string $\mathbf{u} = u_1, \dots, u_l \in \mathcal{A}^l$ by $|\mathbf{u}|$, and use $\mathbf{u}_i^j = (u_i, \dots, u_j)$. The concatenation of strings \mathbf{w} and \mathbf{v} is denoted by \mathbf{wv} . A string \mathbf{v} is a *suffix* of \mathbf{u} , denoted by

$\mathbf{v} \preceq \mathbf{u}$, if there exists a string \mathbf{w} such that $\mathbf{u} = \mathbf{wv}$. A set \mathcal{T} of strings is *suffix-free* if no string of \mathcal{T} is a suffix of any other string in \mathcal{T} .

5.1.2 Trees

As in [74] for example, we use full \mathcal{A} -ary trees to represent the states of a Markov process. We denote full trees \mathcal{T} as a suffix-free set $\mathcal{T} \subset \mathcal{A}^*$ of strings (the *leaves*) whose lengths satisfy Kraft's lemma with equality. The depth of the tree \mathcal{T} is defined as $d(\mathcal{T}) = \max\{|\mathbf{u}| : \mathbf{u} \in \mathcal{T}\}$. A string $\mathbf{v} \in \mathcal{A}^*$ is an *internal node* of \mathcal{T} if either $\mathbf{v} \in \mathcal{T}$ or there exists $\mathbf{u} \in \mathcal{T}$ such that $\mathbf{v} \preceq \mathbf{u}$. The *children* of an internal node \mathbf{v} in \mathcal{T} , are those strings (if any) $a\mathbf{v}, a \in \mathcal{A}$ which are themselves either internal nodes or leaves in \mathcal{T} .

For any internal node \mathbf{w} of a tree \mathcal{T} , let $\mathcal{T}_{\mathbf{w}} = \{\mathbf{u} \in \mathcal{T} : \mathbf{w} \preceq \mathbf{u}\}$ be the subtree rooted at \mathbf{w} . Given two trees \mathcal{T}_1 and \mathcal{T}_2 , we say that \mathcal{T}_1 is included in \mathcal{T}_2 ($\mathcal{T}_1 \preceq \mathcal{T}_2$), if all the leaves in \mathcal{T}_1 are either leaves or internal nodes of \mathcal{T}_2 .

5.1.3 Models

Let $\mathcal{P}^+(\mathcal{A})$ be the set of all probability distributions on \mathcal{A} such that every probability is strictly positive.

Definition 4. A context tree *model* is a finite full tree $\mathcal{T} \subset \mathcal{A}^*$ with a collection of probability distributions $q_{\mathbf{s}} \in \mathcal{P}^+(\mathcal{A})$ assigned to each $\mathbf{s} \in \mathcal{T}$. We will refer to the elements of \mathcal{T} as *states* (or *contexts*), and $q(\mathcal{T}) = \{q(a|\mathbf{s}) : \mathbf{s} \in \mathcal{T}, a \in \mathcal{A}\}$ as the set of *state transition probabilities* or the *process parameters*. \square

Every model $(\mathcal{T}, q(\mathcal{T}))$ allows for an irreducible, aperiodic¹ and ergodic [75]. Such Markov

1. Irreducible since $q_{\mathbf{s}} \in \mathcal{P}^+(\mathcal{A})$, aperiodic since any state $\mathbf{s} \in \mathcal{T}$ can be reached from itself in either $|\mathbf{s}|$ or $|\mathbf{s}| + 1$ steps.

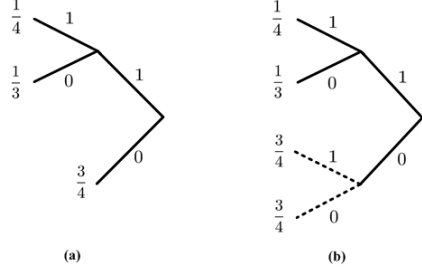


Figure 5.1: (a) States and parameters of a Markov process in Example 4, (b) Same Markov process reparameterized to be a complete tree of depth 2. We can similarly reparameterize the process on the left with a complete tree of any depth larger than 2.

process has a unique stationary distribution μ satisfying

$$\mu Q = \mu, \quad (5.1)$$

where Q is the standard transition probability matrix formed using $q(\mathcal{T})$. Let $p_{\mathcal{T},q}$ be the unique stationary Markov process $\{\dots, Y_0, Y_1, Y_2, \dots\}$ which takes values in \mathcal{A} satisfying

$$p_{\mathcal{T},q}(Y_1|Y_{-\infty}^0) = q(Y_1|\mathbf{s})$$

whenever $\mathbf{s} = \mathbf{c}_{\mathcal{T}}(Y_{-\infty}^0)$, where $\mathbf{c}_{\mathcal{T}} : \mathcal{A}^{\infty} \rightarrow \mathcal{T}$ is the unique suffix $\mathbf{s} \preceq Y_{-\infty}^0$ in \mathcal{T} . As a note, when we write out actual strings in transition probabilities as in $q(0|1000)$, the state 1000 is the sequence of bits as we encounter them when reading the string left to right. If 0 follows the state 1100, the next state is a suffix of 11000, and if 1 follows 1100, the next state is a suffix of 11001.

Observation 2. A useful observation is that any model $(\mathcal{T}, q(\mathcal{T}))$ yields the same Markov process as a model $(\mathcal{T}', q(\mathcal{T}'))$ where $\mathcal{T} \preceq \mathcal{T}'$ and for all $\mathbf{s}' \in \mathcal{T}'$, $q(\cdot|\mathbf{s}') = q(\cdot|\mathbf{c}_{\mathcal{T}}(\mathbf{s}'))$. \square

Example 4. Let $(\mathcal{T}, q(\mathcal{T}))$ be a Model with $\mathcal{T} = \{11, 01, 0\}$ and $q(1|11) = \frac{1}{4}$, $q(1|01) = \frac{1}{3}$, $q(1|0) = \frac{3}{4}$. Fig. 5.1. (b) shows the Markov process as a model $(\mathcal{T}', q(\mathcal{T}'))$ with $\mathcal{T}' = \{11, 01, 10, 00\}$ satisfying conditions in Property 2. \square

5.2 Channel Model

We focus on Markov channels defined as follows. Both input $\{X_n\}_{n=1}^\infty$ and output $\{Y_n\}_{n=1}^\infty$ are finite alphabet processes taking values in \mathcal{A} and the state of channel in each instant depend on sequence of prior outputs of the channel. The input process is drawn from an *i.i.d* process, namely $P(X_n = a) = p_a$ for all $n \in \mathbb{N}$ and $a \in \mathcal{A}$, provided that $\sum_{a \in \mathcal{A}} p_a = 1$. We assume that there is no feedback in this channel setup. The joint probability distribution the channel inputs/outputs factorizes as

$$P(x_1^n, y_1^n | y_{-\infty}^0) = \prod_{k=1}^n P(x_k) P(y_k | y_{-\infty}^{k-1}, x_k). \quad (5.2)$$

The *state* of the channel at time k is therefore determined by $y_{-\infty}^{k-1}$. We consider finite memory channels, and model the possible states of the channels as leaves of a finite full $|\mathcal{A}|$ -ary tree, \mathcal{T} . Recall that $\mathbf{c}_{\mathcal{T}}(y_{-\infty}^{k-1}) \in \mathcal{T}$ is the unique $\mathbf{s} \in \mathcal{T}$ such that $\mathbf{s} \preceq y_{-\infty}^{k-1}$. Therefore, we obtain

$$P(x_1^n, y_1^n | y_{-\infty}^0) = \prod_{k=1}^n P(x_k) P(y_k | \mathbf{c}_{\mathcal{T}}(y_{-\infty}^{k-1}), x_k).$$

Then $\{(X_n, Y_n)\}_{n=1}^\infty$ can be modeled as a Markov process $p_{\mathcal{T}, q}$. Associated with every state $\mathbf{s} \in \mathcal{T}$ is a distribution $q_{\mathbf{s}} \in \mathcal{P}^+(\mathcal{A} \times \mathcal{A})$ which assigns any input/output pair $(a, b) \in \mathcal{A} \times \mathcal{A}$ the probability

$$q_{\mathbf{s}}(a, b) = P(X_k = a, Y_k = b | \mathbf{c}_{\mathcal{T}}(Y_{-\infty}^{k-1}) = \mathbf{s}), \quad \forall k \in \mathbb{N}$$

For convenience, we also denote the input/output transition probabilities encountered upon seeing context $\mathbf{s} \in \mathcal{T}$ by

$$\theta_{\mathbf{s}}(b|a) = P(Y_1 = b | \mathbf{c}_{\mathcal{T}}(Y_{-\infty}^0) = \mathbf{s}, X_1 = a).$$

Therefore, we have

$$q_{\mathbf{s}}(a, b) = p_a \theta_{\mathbf{s}}(b|a).$$

The set $\Theta_{\mathbf{s}} = \{\theta_{\mathbf{s}}(\cdot|a) : a \in \mathcal{A}\}$ is the set of all conditional probabilities associated with state \mathbf{s} . Note that $\theta_{\mathbf{s}}(\cdot|a) \in \mathcal{P}^+(\mathcal{A})$ for all $a \in \mathcal{A}$ and $\mathbf{s} \in \mathcal{T}$. The set

$$\Theta_{\mathcal{T}} = \bigcup_{\mathbf{s} \in \mathcal{T}} \Theta_{\mathbf{s}}$$

is the set of all transition probabilities of channel model and we refer to it as the *channel parameters*. Since the input is a known *i.i.d* process, estimating $q(\mathcal{T})$ and $\Theta_{\mathcal{T}}$ are completely equivalent parameterizations.

As emphasized in the introduction, we do not assume the true channel model is known nor do we assume it is fast mixing. We would like to know if we can estimate the channel parameters and the stationary probabilities of various states of the channel even when we are in the domain where the mixing has not happened.

5.3 Background Topics

5.3.1 Context Tree Weighting

Context tree weighting algorithm is a universal data compression algorithm for Markov sources [74, 76], and the algorithm can be used to capture several insights into how Markov processes behave in the non-asymptotic regime. Let y_1^n be sequence of symbols from an alphabet \mathcal{A} . Let $\hat{\mathcal{T}} = \mathcal{A}^D$ for some positive integer D . For all $\mathbf{s} \in \hat{\mathcal{T}}$ and $a \in \mathcal{A}$, let $n_{\mathbf{s}}^a$ be the empirical counts of string $\mathbf{s}a$ in y_1^n . The depth- D context tree weighting constructs a distribution

$$p_c(y_1^n | y_{-D}^0) \geq 2^{-|\mathcal{A}|^{D+1} \log n} \prod_{\mathbf{s} \in \hat{\mathcal{T}}} \prod_{a \in \mathcal{A}} \left(\frac{n_{\mathbf{s}}^a}{\sum_{a \in \mathcal{A}} n_{\mathbf{s}}^a} \right)^{n_{\mathbf{s}}^a}.$$

Note that no Markov source with memory D could have given a higher probability to y_1^n than

$$\prod_{\mathbf{s} \in \hat{\mathcal{T}}} \prod_{a \in \mathcal{A}} \left(\frac{n_{\mathbf{s}}^a}{\sum_{a \in \mathcal{A}} n_{\mathbf{s}}^a} \right)^{n_{\mathbf{s}}^a}.$$

So, if $|A|^D \log n = o(n)$, then p_c underestimates any memory- D Markov probability by only a subexponential factor. Therefore, $D = \mathcal{O}(\log n)$ is going to be the case of particular interest.

5.3.2 Coupling for Markov Processes

Coupling is an elegant technique that will help us understand how the counts of certain strings in a sample behave. Let $p_{\mathcal{T},q}$ be our Markov source generating sequences from an alphabet $\tilde{\mathcal{A}}$. A coupling ω for $p_{\mathcal{T},q}$ is a joint probability distribution on the sequences $\{Y_m, \bar{Y}_m\}_{m \geq 1}$ where $Y_m \in \tilde{\mathcal{A}}$ and $\bar{Y}_m \in \tilde{\mathcal{A}}$, and ω satisfies the following property: individually taken, the sequences $\{Y_m\}$ and $\{\bar{Y}_m\}$ have to be faithful evolutions of $p_{\mathcal{T},q}$. Specifically, for $m \geq 0$, we want

$$\omega(Y_{m+1}|Y_{-\infty}^m, \bar{Y}_{-\infty}^m) = p_{\mathcal{T},q}(Y_{m+1}|Y_{-\infty}^m) = p_{\mathcal{T},q}(Y_{m+1}|\mathbf{c}_{\mathcal{T}}(Y_{-\infty}^m)), \quad (5.3)$$

and similarly for $\{\bar{Y}_m\}$. In the context of this work, we think of $\{Y_m\}$ and $\{\bar{Y}_m\}$ here as copies of $p_{\mathcal{T},q}$ that were started with two different states $\mathbf{s}, \mathbf{s}' \in \mathcal{T}$ respectively, but the chains evolve jointly as ω instead of independently. For any r and $\mathbf{w} \in \tilde{\mathcal{A}}^r$, $N_n(\mathbf{w})$ (respectively $\bar{N}_n(\mathbf{w})$) is the number of times \mathbf{w} forms the context of a symbol in a length- n time frame, $\{Y_i\}_{i=1}^n$ given $Y_{-\infty}^0$ (respectively $\{\bar{Y}_i\}_{i=1}^n$ given $\bar{Y}_{-\infty}^0$). Then, for any ω ,

$$\begin{aligned} |\mathbb{E}_{p_{\mathcal{T},q}}[N_n(\mathbf{w})|Y_{-\infty}^0] - \mathbb{E}_{p_{\mathcal{T},q}}[\bar{N}_n(\mathbf{w})|\bar{Y}_{-\infty}^0]| &= \left| \sum_{i=1}^n E_{\omega} [\mathbb{1}(\mathbf{c}_{\tilde{\mathcal{A}}^r}(Y_{-\infty}^i) = \mathbf{w}) - \mathbb{1}(\mathbf{c}_{\tilde{\mathcal{A}}^r}(\bar{Y}_{-\infty}^i) = \mathbf{w})] \right| \\ &\leq \sum_{i=1}^n \left| \mathbb{E}_{\omega} [\mathbb{1}(\mathbf{c}_{\tilde{\mathcal{A}}^r}(Y_{-\infty}^i) = \mathbf{w}) - \mathbb{1}(\mathbf{c}_{\tilde{\mathcal{A}}^r}(\bar{Y}_{-\infty}^i) = \mathbf{w})] \right| \\ &\leq \sum_{i=1}^n \omega(\mathbf{c}_{\tilde{\mathcal{A}}^r}(Y_{-\infty}^i) \neq \mathbf{c}_{\tilde{\mathcal{A}}^r}(\bar{Y}_{-\infty}^i)), \end{aligned}$$

where the first equality follows from (5.3).

The art of a coupling argument stems from the fact that ω is completely arbitrary apart from having to satisfy (5.3). If we can find *any* ω such that the chains *coalesce*, namely $\omega(\mathbf{c}_{\tilde{\mathcal{A}}^r}(Y_{-\infty}^i) \neq \mathbf{c}_{\tilde{\mathcal{A}}^r}(\bar{Y}_{-\infty}^i))$ becomes small as i increases, then we know that $\mathbb{E}_{p_{\mathcal{T},q}}[N_n(\mathbf{w})|Y_{-\infty}^0]$

cannot differ too much from $\mathbb{E}_{p_{\mathcal{T},q}}[\bar{N}_n(\mathbf{w})|\bar{Y}_{-\infty}^0]$. For tutorials, see *e.g.*, [77–79].

5.4 Long Memory and Slow Mixing

There are two distinct difficulties in estimating Markov processes as the ones we are interested in. The first is memory that is too long to handle given the size of the sample at hand. The second issue is that even though the underlying process might be ergodic, the transition probabilities are so small such that the process effectively acts like a non-ergodic process given the sample size available. We illustrate these problems in following simple examples.

Example 5. Let $\mathcal{T} = \mathcal{A}^k$ denote a full tree with depth k and $\mathcal{A} = \{0, 1\}$. Assume that $q(1|0^k) = 2\epsilon$ and $q(1|10^{k-1}) = 1 - \epsilon$ with $\epsilon > 0$, and let $q(1|\mathbf{s}) = \frac{1}{2}$ (where 0^k indicates a string with k consecutive zeros) for all other $\mathbf{s} \in \mathcal{T}$. Let $p_{\mathcal{T},q}$ represent the stationary ergodic Markov process associated with this model. Observe that stationary probability of being in state 0^k is $\frac{1}{2^{k+1}-1}$ while all other states have stationary probability $\frac{2}{2^{k+1}-1}$. Let Y_1^n be a realization of this process with initial state $1^k \preceq Y_{-\infty}^0$. Suppose $k \gg \omega(\log n)$.² With high probability we will never find a string of $k - 1$ zeros among n samples, and every bit is generated with probability $1/2$. Thus with this sample size, with high probability, we cannot distinguish $p_{\mathcal{T},q}$ from an *i.i.d* Bernoulli($1/2$) process. \square

We therefore require that dependencies die down by requiring that channel parameters $\theta_{1\mathbf{s}}^a$ and $\theta_{0\mathbf{s}}^a$, corresponding to sibling contexts $1\mathbf{s}$ and $0\mathbf{s}$, satisfy (6.1) in Section 6.1.

Example 6. Let $\mathcal{A} = \{0, 1\}$ and $\mathcal{T} = \{0, 1\}$ with $q(1|1) = 1 - \epsilon$, and $q(1|0) = \epsilon$. For $\epsilon > 0$, this model represents a stationary ergodic Markov processes with stationary distributions $\mu(1) = \frac{1}{2}, \mu(0) = \frac{1}{2}$. Let $\mathcal{T}' = \{0, 1\}$ with $q'(1|1) = 1 - \epsilon, q'(1|0) = 2\epsilon$. Similarly, for $\epsilon > 0$ this model represents a stationary ergodic Markov processes with stationary distributions $\mu'(1) = \frac{2}{3}, \mu'(0) = \frac{1}{3}$. Suppose we have a length- n sample. In this case, we cannot distinguish between these two models if $\epsilon \ll o(1/n)$. \square

2. A function $f_n = \omega(g_n)$ if $\lim_{n \rightarrow \infty} f_n/g_n = \infty$.

5.4.1 Lower Bound on Information Rate

Consider a channel with state tree \mathcal{T} and parameter set $\Theta_{\mathcal{T}}$. Suppose that $d(\mathcal{T}) = D < \infty$. The information rate for an *i.i.d* input process with $P(X_k = a) = p_a$ for this channel is

$$\begin{aligned}
R_{\mathcal{T}} &\stackrel{\text{def}}{=} \lim_{n \rightarrow \infty} \frac{1}{n} I(X^n; Y^n) \\
&= \lim_{n \rightarrow \infty} \frac{1}{n} [H(Y^n) - H(Y^n | X^n)] \\
&\stackrel{a}{=} \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=1}^n \left[H(Y_k | Y^{k-1}) - H(Y_k | Y^{k-1}, X_k) \right] \\
&\stackrel{b}{=} \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=1}^D \sum_{y^{k-1}} P(Y^{k-1} = y^{k-1}) \left[H(Y_k | Y^{k-1} = y^{k-1}) - H(Y_k | Y^{k-1} = y^{k-1}, X_k) \right] \\
&\quad + \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=D+1}^n \sum_{y^{k-1}} P(Y^{k-1} = y^{k-1}) \left[H(Y_k | Y^{k-1} = y^{k-1}) - H(Y_k | Y^{k-1} = y^{k-1}, X_k) \right] \\
&\stackrel{c}{=} \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=D+1}^n \sum_{\mathbf{s} \in \mathcal{T}} \sum_{\substack{y^{k-1}: \\ \mathbf{s} \preceq y^{k-1}}} P(Y^{k-1} = y^{k-1}) \left[H(Y_k | Y^{k-1} = y^{k-1}) - H(Y_k | Y^{k-1} = y^{k-1}, X_k) \right]
\end{aligned}$$

where (a) is by chain rule for entropy and from (5.2) and (b) is straightforward from definition of the conditional entropy. The equality in (c) holds since the first term in (b) vanishes as $n \rightarrow \infty$ and observing that for $k \geq D + 1$,

$$\bigcup_{\mathbf{s} \in \mathcal{T}} \mathcal{A}_{\mathbf{s}}^{k-1} = \bigcup_{\mathbf{s} \in \mathcal{T}} \{y^{k-1} \in \mathcal{A}^{k-1} : \mathbf{s} \preceq y^{k-1}\} = \mathcal{A}^{k-1}.$$

Note that for all $k \in \mathbb{N}$, if $\mathbf{s} \preceq y^{k-1}$, it can easily be shown that

$$\begin{aligned}
H(Y_k | Y^{k-1} = y^{k-1}) - H(Y_k | Y^{k-1} = y^{k-1}, X_k) &= \sum_{a \in \mathcal{A}} p_a \sum_{b \in \mathcal{A}} \theta_{\mathbf{s}}(b|a) \log \frac{\theta_{\mathbf{s}}(b|a)}{\sum_{a' \in \mathcal{A}} p_{a'} \theta_{\mathbf{s}}(b|a')} \\
&\stackrel{\text{def}}{=} R_{\mathbf{s}}(\Theta_{\mathbf{s}}).
\end{aligned}$$

Therefore,

$$\begin{aligned}
R_{\mathcal{T}} &= \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=D+1}^n \sum_{\mathbf{s} \in \mathcal{T}} \sum_{\substack{y^{k-1}; \\ \mathbf{s} \preceq y^{k-1}}} P(Y^{k-1} = y^{k-1}) R_{\mathbf{s}}(\Theta_{\mathbf{s}}) \\
&\stackrel{d}{=} \sum_{\mathbf{s} \in \mathcal{T}} R_{\mathbf{s}}(\Theta_{\mathbf{s}}) \left[\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=D+1}^n \sum_{\substack{y^{k-1}; \\ \mathbf{s} \preceq y^{k-1}}} P(Y^{k-1} = y^{k-1}) \right] \\
&\stackrel{e}{=} \sum_{\mathbf{s} \in \mathcal{T}} R_{\mathbf{s}}(\Theta_{\mathbf{s}}) \left[\lim_{k \rightarrow \infty} \sum_{\substack{y^{k-1}; \\ \mathbf{s} \preceq y^{k-1}}} P(Y^{k-1} = y^{k-1}) \right] \\
&\stackrel{f}{=} \sum_{\mathbf{s} \in \mathcal{T}} \mu(\mathbf{s}) R_{\mathbf{s}}(\Theta_{\mathbf{s}}).
\end{aligned}$$

The equality in (d) is by changing the order of summations and (e) follows by using Cesàro's lemma [80]. Finally, (f) holds by properties of stationary distribution in Markov processes.

As a remark, note that for fixed input distribution, $R_{\mathbf{s}}$ is a function of $\Theta_{\mathbf{s}} = \{\theta_{\mathbf{s}}(\cdot|a) : a \in \mathcal{A}\}$.

Lemma 5. For fixed input distribution, $R_{\mathbf{s}}(\Theta_{\mathbf{s}})$ is convex in $\Theta_{\mathbf{s}}$.

Proof Let $\lambda \in [0, 1]$ and $\bar{\lambda} = 1 - \lambda$. Let $\Theta_{\mathbf{s}} = \{\theta_{\mathbf{s}}(\cdot|a) : a \in \mathcal{A}\}$ and $\Theta'_{\mathbf{s}} = \{\theta'_{\mathbf{s}}(\cdot|a) : a \in \mathcal{A}\}$ be two sets of valid conditional distributions associated to state \mathbf{s} . Then,

$$\begin{aligned}
R_{\mathbf{s}}(\lambda\Theta_{\mathbf{s}} + \bar{\lambda}\Theta'_{\mathbf{s}}) &= \\
&= \sum_{a \in \mathcal{A}} p_a \sum_{b \in \mathcal{A}} \left[\left(\lambda\theta_{\mathbf{s}}(b|a) + \bar{\lambda}\theta'_{\mathbf{s}}(b|a) \right) \log \frac{\lambda\theta_{\mathbf{s}}(b|a) + \bar{\lambda}\theta'_{\mathbf{s}}(b|a)}{\sum_{a' \in \mathcal{A}} p_{a'} \left(\lambda\theta_{\mathbf{s}}(b|a') + \bar{\lambda}\theta'_{\mathbf{s}}(b|a') \right)} \right] \\
&= \sum_{a \in \mathcal{A}} p_a \sum_{b \in \mathcal{A}} \left[\left(\lambda\theta_{\mathbf{s}}(b|a) + \bar{\lambda}\theta'_{\mathbf{s}}(b|a) \right) \log \frac{\lambda\theta_{\mathbf{s}}(b|a) + \bar{\lambda}\theta'_{\mathbf{s}}(b|a)}{\lambda \sum_{a' \in \mathcal{A}} p_{a'} \theta_{\mathbf{s}}(b|a') + \bar{\lambda} \sum_{a' \in \mathcal{A}} p_{a'} \theta'_{\mathbf{s}}(b|a')} \right] \\
&\stackrel{e}{\leq} \sum_{a \in \mathcal{A}} p_a \sum_{b \in \mathcal{A}} \left[\lambda \theta_{\mathbf{s}}(b|a) \log \frac{\lambda \cdot \theta_{\mathbf{s}}(b|a)}{\lambda \cdot \sum_{a' \in \mathcal{A}} p_{a'} \theta_{\mathbf{s}}(b|a')} + \bar{\lambda} \theta'_{\mathbf{s}}(b|a) \log \frac{\bar{\lambda} \cdot \theta'_{\mathbf{s}}(b|a)}{\bar{\lambda} \cdot \sum_{a' \in \mathcal{A}} p_{a'} \theta'_{\mathbf{s}}(b|a')} \right] \\
&= \lambda R_{\mathbf{s}}(\Theta_{\mathbf{s}}) + \bar{\lambda} R_{\mathbf{s}}(\Theta'_{\mathbf{s}})
\end{aligned}$$

where the inequality in (e) follows by using *log-sum* inequality (see *e.g.*, [80]). \square

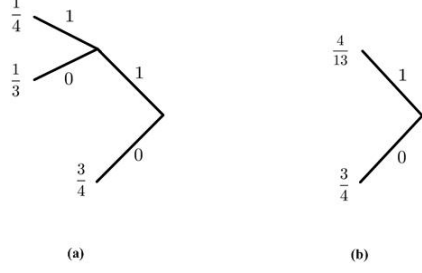


Figure 5.2: (a) Markov process in Example 7, (b) Aggregated model at depth 1. From Observation 1, the model on the left can be reparameterized to be a complete tree at any depth ≥ 2 . We can hence ask for its aggregation at any depth. Aggregations of the above model on the left at depths ≥ 2 will hence be the model itself.

Since the memory is unknown a-priori, a natural approach, known to be consistent, is to use a potentially coarser model with depth k_n . Here, k_n increases logarithmically with the sample size n , and reflects [21] well known results on consistent estimation of Markov processes. We show that coarser models formed by properly aggregating states of the original channel are useful in lower bounding information rates of the true channel.

Definition 5. We say that $p_{\tilde{\mathcal{T}},\tilde{q}}$ aggregates $p_{\mathcal{T},q}$ (or $p_{\mathcal{T},q}$ refines $p_{\tilde{\mathcal{T}},\tilde{q}}$), if $\tilde{\mathcal{T}} \preceq \mathcal{T}$ and $p_{\tilde{\mathcal{T}},\tilde{q}}$ be the stationary Markov process with state transition probabilities given by

$$\tilde{q}(a|\mathbf{w}) = \frac{\sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v})q(a|\mathbf{v})}{\sum_{\mathbf{v}' \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}')}$$

for all $\mathbf{w} \in \tilde{\mathcal{T}}$ and $a \in \mathcal{A}$, where μ is the stationary distribution associated with $p_{\mathcal{T},q}$. Using Observation 2, wolog, no matter what $\tilde{\mathcal{T}}$ is, we will assume $p_{\mathcal{T},q}$ has states \mathcal{T} such that $\tilde{\mathcal{T}} \preceq \mathcal{T}$. \square

Example 7. Let $p_{\mathcal{T},q}$ be a Markov process with $\mathcal{T} = \{11, 01, 0\}$ and $q(1|11) = \frac{1}{4}$, $q(1|01) = \frac{1}{3}$, $q(1|0) = \frac{3}{4}$. For this model, we have $\mu(11) = \frac{4}{25}$, $\mu(01) = \frac{9}{25}$ and $\mu(0) = \frac{12}{25}$. Fig. 5.2. (b) shows an aggregated process $p_{\tilde{\mathcal{T}},\tilde{q}}$ with $\tilde{\mathcal{T}} = \{1, 0\}$. Notice that $\tilde{q}(1|1) = (\frac{4}{25}\frac{1}{4} + \frac{9}{25}\frac{1}{3})/(\frac{4}{25} + \frac{9}{25}) = \frac{4}{13}$ and $\tilde{q}(1|0) = \frac{3}{4}$. \square

Lemma 6. Let $p_{\mathcal{T},q}$ be a stationary Markov process with stationary distribution μ . If $p_{\tilde{\mathcal{T}},\tilde{q}}$

aggregates $p_{\mathcal{T},q}$ then it has a unique stationary distribution $\tilde{\mu}$ and for every $\mathbf{w} \in \tilde{\mathcal{T}}$

$$\tilde{\mu}(\mathbf{w}) = \sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}).$$

Moreover, for all $a_1^k \in \mathcal{A}^k$ such that a_1^k is an internal node of $\tilde{\mathcal{T}}$ we have $\tilde{\mu}(a_1^k) = \mu(a_1^k)$.

Proof Let \tilde{Q} be the transition probability matrix formed by the states of $p_{\tilde{\mathcal{T}},\tilde{q}}$. First notice that by definition, for all $\mathbf{w}, \mathbf{w}' \in \tilde{\mathcal{T}}$

$$\tilde{Q}(\mathbf{w}|\mathbf{w}') = \begin{cases} 0 & \text{if } \nexists a \in \mathcal{A} \text{ s.t. } \mathbf{w} \preceq \mathbf{w}'a \\ \tilde{q}(a|\mathbf{w}') & \text{if } \mathbf{w} \preceq \mathbf{w}'a \text{ for some } a \in \mathcal{A} \end{cases}$$

Since $p_{\mathcal{T},q}$ is irreducible and aperiodic, $p_{\tilde{\mathcal{T}},\tilde{q}}$ will also be irreducible and aperiodic and thus, has a unique stationary distribution $\tilde{\mu}$. Hence, there exists a unique solution for

$$\tilde{\mu}(\mathbf{w}) = \sum_{\mathbf{w}' \in \tilde{\mathcal{T}}} \tilde{\mu}(\mathbf{w}') \tilde{Q}(\mathbf{w}|\mathbf{w}') \quad \forall \mathbf{w} \in \tilde{\mathcal{T}}. \quad (5.4)$$

We will consider a candidate solution of the form

$$\tilde{\mu}(\mathbf{w}) = \sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v})$$

for every $\mathbf{w} \in \tilde{\mathcal{T}}$ and show that this candidate will satisfy (5.4). Then, the claim will follow

by uniqueness of the solution. To show this, note that for $\forall \mathbf{w} \in \tilde{\mathcal{T}}$

$$\begin{aligned}
\sum_{\substack{\mathbf{w}' \in \tilde{\mathcal{T}} \\ \mathbf{w} \preceq \mathbf{w}' a}} \tilde{\mu}(\mathbf{w}') \tilde{q}(a|\mathbf{w}') &= \sum_{\substack{\mathbf{w}' \in \tilde{\mathcal{T}} \\ \mathbf{w} \preceq \mathbf{w}' a}} \left[\sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}'}} \mu(\mathbf{v}) \right] \frac{\sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}'}} \mu(\mathbf{v}) q(a|\mathbf{v})}{\sum_{\mathbf{v}' \in \mathcal{T}_{\mathbf{w}'}} \mu(\mathbf{v}')} \\
&= \sum_{\substack{\mathbf{w}' \in \tilde{\mathcal{T}} \\ \mathbf{w} \preceq \mathbf{w}' a}} \sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}'}} \mu(\mathbf{v}) q(a|\mathbf{v}) \\
&= \sum_{\substack{\mathbf{w}' \in \tilde{\mathcal{T}} \\ \mathbf{w} \preceq \mathbf{w}' a}} \sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}'}} \mu(\mathbf{v} a) \\
&\stackrel{d}{=} \sum_{\mathbf{s} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{s}) \\
&= \tilde{\mu}(\mathbf{w})
\end{aligned}$$

where (d) follows by observing that

$$\mathbf{w} \preceq \{\mathbf{v} a : \exists \mathbf{w}' \in \tilde{\mathcal{T}}, a \in \mathcal{A} \text{ s.t. } \mathbf{w} \preceq \mathbf{w}' a \text{ and } \mathbf{v} \in \mathcal{T}_{\mathbf{w}'}\},$$

and then using properties of the stationary distribution of $p_{\mathcal{T},q}$. Note that the second statement of Lemma automatically follows from the uniqueness of stationary distributions. \square

In a similar manner as Definition 5, given any input output process for a channel we can define an aggregated channel with tree $\tilde{\mathcal{T}}$ and parameter set $\tilde{\Theta}_{\tilde{\mathcal{T}}}$. For all $\mathbf{w} \in \tilde{\mathcal{T}}$, let $\tilde{\Theta}_{\mathbf{w}} = \{\tilde{\theta}_{\mathbf{w}}(\cdot|a) : a \in \mathcal{A}\}$ in which for fixed $a \in \mathcal{A}$, $\tilde{\theta}_{\mathbf{w}}(\cdot|a)$ is given by

$$\tilde{\theta}_{\mathbf{w}}(b|a) = \frac{\sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}) \theta_{\mathbf{v}}(b|a)}{\sum_{\mathbf{v}' \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}')}, \quad \forall b \in \mathcal{A}$$

Theorem 7. Consider a channel with tree \mathcal{T} and parameter set $\Theta_{\mathcal{T}}$. If $\tilde{\mathcal{T}}$ aggregates \mathcal{T} , then $R_{\tilde{\mathcal{T}}} \leq R_{\mathcal{T}}$.

Proof Note that for all $\mathbf{w} \in \tilde{\mathcal{T}}$, $\mathcal{T}_{\mathbf{w}} = \{\mathbf{s} \in \mathcal{T} : \mathbf{w} \preceq \mathbf{s}\}$. Since $\tilde{\mathcal{T}} \preceq \mathcal{T}$, we have

$$\begin{aligned}
R_{\tilde{\mathcal{T}}} &= \sum_{\mathbf{w} \in \tilde{\mathcal{T}}} \tilde{\mu}(\mathbf{w}) R_{\mathbf{w}}(\tilde{\Theta}_{\mathbf{w}}) \\
&\stackrel{a}{\leq} \sum_{\mathbf{w} \in \tilde{\mathcal{T}}} \tilde{\mu}(\mathbf{w}) \sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \left[\frac{\mu(\mathbf{v})}{\sum_{\mathbf{v}' \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}')} R_{\mathbf{v}}(\Theta_{\mathbf{v}}) \right] \\
&\stackrel{b}{=} \sum_{\mathbf{w} \in \tilde{\mathcal{T}}} \sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}) R_{\mathbf{v}}(\Theta_{\mathbf{v}}) \\
&= \sum_{\mathbf{s} \in \mathcal{T}} \mu(\mathbf{s}) R_{\mathbf{s}}(\Theta_{\mathbf{s}}) \\
&= R_{\mathcal{T}}
\end{aligned}$$

where the inequality in (a) holds by Lemma 5 and the fact that $\forall a, b \in \mathcal{A}$

$$\tilde{\theta}_{\mathbf{w}}(b|a) = \frac{\sum_{\mathbf{v} \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}) \theta_{\mathbf{v}}(b|a)}{\sum_{\mathbf{v}' \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}')}.$$

The equality in (b) holds since $\tilde{\mu}(\mathbf{w}) = \sum_{\mathbf{v}' \in \mathcal{T}_{\mathbf{w}}} \mu(\mathbf{v}')$. □

Remark In this Chapter, we are particularly concerned with the slow mixing regime. As our results will show, in general it is not possible to obtain a simple lower bound on the information rate using the data and taking recourse to the Theorem above. Instead, we introduce the *partial* information rate that can be reliably obtained from the data

$$R_{\tilde{G}}^p = \sum_{\mathbf{s} \in \tilde{G}} \frac{\mu(\mathbf{s})}{\mu(\tilde{G})} R_{\mathbf{s}}(\tilde{\Theta}_{\mathbf{s}}),$$

where $\tilde{G} \subseteq \tilde{\mathcal{T}}$ will be a set of *good* states that we show how to identify. The partial information rate is not necessarily a lower bound, but in slow mixing cases it is sometimes the best heuristic possible. □

Notwithstanding the previous remark, we will focus on estimating the aggregated parameters $\Theta_{\mathcal{T}}$, where $\tilde{\mathcal{T}} = \mathcal{A}^{k_n}$ has depth k_n where k_n grows logarithmically as $\alpha_n \log n$ for some

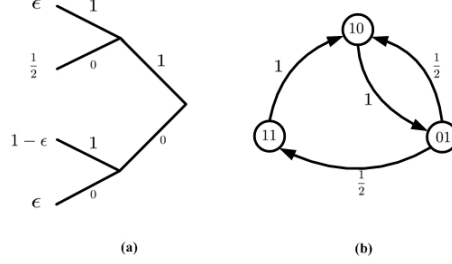


Figure 5.3: (a) Markov in Example 8, (b) Same process when $\epsilon = 0$.

$\alpha_n = \mathcal{O}(1)$. Now $p_{\tilde{\mathcal{T}}, \tilde{q}}$ is unknown and we do not have access to samples from it. And there is, of course, no guarantee that the counts of short strings are any more reliable in a long-memory, slow mixing process.

Example 8. Let $\mathcal{T} = \{11, 01, 10, 00\}$ with $q(1|11) = \epsilon$, $q(1|01) = \frac{1}{2}$, $q(1|10) = 1 - \epsilon$, $q(1|00) = \epsilon$. If $\epsilon > 0$, then $p_{\mathcal{T}, q}$ is a stationary ergodic binary Markov process. Let μ denote the stationary distribution of this process, respectively. A simple computation shows that $\mu(11) = \frac{1}{7-6\epsilon}$, $\mu(01) = \frac{2-2\epsilon}{7-6\epsilon}$, $\mu(10) = \frac{2-2\epsilon}{7-6\epsilon}$ and $\mu(00) = \frac{2-2\epsilon}{7-6\epsilon}$, and $\mu(1) = \frac{1}{7-6\epsilon} + \frac{2-2\epsilon}{7-6\epsilon} = \frac{3-2\epsilon}{7-6\epsilon}$ and $\mu(0) = \frac{2-2\epsilon}{7-6\epsilon} + \frac{2-2\epsilon}{7-6\epsilon} = \frac{4-4\epsilon}{7-6\epsilon}$.

Suppose we have a length n sample. If $\epsilon \ll \frac{1}{n}$, then $\mu(1) \approx \frac{3}{7}$ and $\mu(0) \approx \frac{4}{7}$ respectively. If the initial state belongs to $\{11, 01, 10\}$, the state 00 will not be visited with high probability in n samples, and it can be seen that the counts of 1 or 0 will not be near the stationary probabilities $\mu(1)$ or $\mu(0)$. For this sample size, the process effectively acts like the irreducible, aperiodic Markov chain in Fig. 5.3. (b) which can be shown to be fast mixing. Therefore, the stationary probabilities of the chain in Fig. 5.3. (b), $\frac{\mu(01)}{\mu(1)+\mu(01)}$, $\frac{\mu(10)}{\mu(1)+\mu(01)}$ and $\frac{\mu(11)}{\mu(1)+\mu(01)}$ converge quicker than $\mu(1)$ or $\mu(0)$. \square

6

Estimation On Channels with Output Memory

6.1 dependencies dying down

As noted before in Example 5, if the dependencies could be arbitrary in a channel model, we will not estimate the model accurately no matter how large the sample is. Keeping in mind Observation 2, we formalize dependencies dying down by means of a function $f : \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ with $\sum_{i=1}^{\infty} f(i) < \infty$ and assuming that for all $\mathbf{w} \in \mathcal{A}^*$ and all $c, c' \in \mathcal{A}$

$$\left| \frac{\theta_{c\mathbf{w}}(b|a)}{\theta_{c'\mathbf{w}}(b|a)} - 1 \right| \leq f(|\mathbf{w}|) \tag{6.1}$$

where $a, b \in \mathcal{A}$ and $\theta_{c\mathbf{w}}(b|a) = P(Y_1 = b | \mathbf{c}_{\mathcal{T}}(Y_{-\infty}^0) = c\mathbf{w}, X_1 = a)$. Note that the tree is finite iff there exist a finite D such that $f(i) = 0$ for all $i > D$.

As mentioned in the last section, we will focus on set of the aggregated parameters at depth k_n , $\Theta_{\mathcal{A}^{k_n}}$ where $k_n = \alpha_n \log n$. If k_n is large enough, these aggregated parameters start to reflect the underlying parameters $\Theta_{\mathcal{T}}$. Indeed, by using an elementary argument we will show that both the underlying and aggregated parameters will then be close to the empirically observed values for states that occur frequently enough.

Throughout this Section, we assume that we start with some past $Y_{-\infty}^0$, and we see n samples (X_1^n, Y_1^n) from the channel. All confidence probabilities are conditional probabilities on Y_1^n given X_1^n and $Y_{-\infty}^0$, but we do not write $Y_{-\infty}^0$ out explicitly to avoid cluttering notation. The results hold for all $Y_{-\infty}^0$ (not just with probability 1).

Lemma 8. Let $\{f(i)\}_{i=1}^{\infty}$ be a sequence of real numbers such that there exists some $n_0 \in \mathbb{N}$ for which, $0 \leq f(i) \leq 1$ for all $i \geq n_0$. Then, $\forall j \geq n_0$, we have

$$1 - \sum_{i \geq j} f(i) \leq \prod_{i \geq j} (1 - f(i)) \leq \frac{1}{\prod_{i \geq j} (1 + f(i))}$$

Proof Wolog, let $f(i)$ be decreasing, and consider a distribution q over \mathbb{N} such that

$$\sum_{j \geq i} q(j) = f(i).$$

Let E_i be a sequence of events denoting whether numbers drawn independently as per q is $\geq i$. Thus E_i are independent with $\mathbb{P}(E_i) = f(i)$. Then, $\forall j \in \mathbb{N}$ such that $j \geq n_0$, we have

$$1 - \prod_{i \geq j} (1 - f(i)) = \mathbb{P}\left(\bigcup_{i \geq j} E_i\right) \leq \sum_{i \geq j} \mathbb{P}(E_i) = \sum_{i \geq j} f(i).$$

Hence,

$$1 - \sum_{i \geq j} f(i) \leq \prod_{i \geq j} (1 - f(i)).$$

Since by assumption $0 \leq f(i) \leq 1$ for all $i \geq n_0$, the second inequality can easily be derived by the fact that

$$\prod_{i \geq j} (1 - f^2(i)) \leq 1. \quad \square$$

Lemma 9. Let \mathcal{T} be a model associated with the channel and satisfying condition (6.1). Suppose $\tilde{\mathcal{T}} \preceq \mathcal{T}$ with $d(\tilde{\mathcal{T}}) = k_n$. If $\sum_{i \geq k_n} f(i) \leq 1$, then for all $\mathbf{w} \in \tilde{\mathcal{T}}$ and $a, b \in \mathcal{A}$

$$(1 - \nu_{k_n}) \max_{\mathbf{s} \in \tilde{\mathcal{T}}_{\mathbf{w}}} \theta_{\mathbf{s}}(b|a) \leq \tilde{\theta}_{\mathbf{w}}(b|a) \leq \frac{\min_{\mathbf{s} \in \mathcal{T}_{\mathbf{w}}} \theta_{\mathbf{s}}(b|a)}{(1 - \nu_{k_n})}$$

Proof Let $\mathbf{w} \in \tilde{\mathcal{T}}$ and fix $a, b \in \mathcal{A}$. Note that for $i = k_n$, by assumption we have for all $c, c' \in \mathcal{A}$

$$\left| \frac{\theta_{c\mathbf{w}}(b|a)}{\theta_{c'\mathbf{w}}(b|a)} - 1 \right| \leq f(k_n). \quad (6.2)$$

From Lemma 6, $\tilde{\theta}_{\mathbf{w}}(b|a)$ is a weighted average of $\theta_{c\mathbf{w}}(b|a)$, $c \in \mathcal{A}$. Hence,

$$\min_{d \in \mathcal{A}} \theta_{d\mathbf{w}}(b|a) \leq \tilde{\theta}_{\mathbf{w}}(b|a) \leq \max_{d' \in \mathcal{A}} \theta_{d'\mathbf{w}}(b|a). \quad (6.3)$$

From (6.2) and (6.3), $\forall c \in \mathcal{A}$

$$\tilde{\theta}_{\mathbf{w}}(b|a)(1 - f(k_n)) \leq \theta_{c\mathbf{w}}(b|a) \leq (1 + f(k_n))\tilde{\theta}_{\mathbf{w}}(b|a).$$

Proceeding inductively, for all $\mathbf{s} \in \tilde{\mathcal{T}}_{\mathbf{w}}$ we have

$$\left(\prod_{i \geq k_n} (1 - f(i)) \right) \tilde{\theta}_{\mathbf{w}}(b|a) \leq \theta_{\mathbf{s}}(b|a) \leq \left(\prod_{i \geq k_n} (1 + f(i)) \right) \tilde{\theta}_{\mathbf{w}}(b|a).$$

The lemma follows. □

6.2 Estimation of State Transition Probabilities

Definition 6. For all sequences (X_1^n, Y_1^n) obtained from the channel model $p_{\tau, q}$, let $\hat{\mathcal{T}} = \mathcal{A}^{k_n}$ with $k_n = \alpha_n \log n$ for some function $\alpha_n = \mathcal{O}(1)$. For $\mathbf{s} \in \hat{\mathcal{T}}$, let $\mathbf{Y}_\mathbf{s}^a$ be the sequence of output symbols that follows the output string \mathbf{s} , and correspond to the input $x = a$. Hence, the length of $\mathbf{Y}_\mathbf{s}^a$ is

$$N_n(\mathbf{s}, a) = \sum_{k=1}^n \mathbb{1}\{\mathbf{c}_{\hat{\mathcal{T}}}(Y_{-\infty}^{k-1}) = \mathbf{s}, X_k = a\},$$

the number of occurrences of symbol b in $\mathbf{Y}_\mathbf{s}^a$ is $n_\mathbf{s}(b, a)$, where

$$n_\mathbf{s}(b, a) = \sum_{k=1}^n \mathbb{1}\{\mathbf{c}_{\hat{\mathcal{T}}}(Y_{-\infty}^{k-1}) = \mathbf{s}, Y_k = b, X_k = a\}.$$

We define the naive estimate of $\tilde{\theta}_\mathbf{s}(b|a)$ as

$$\hat{\theta}_\mathbf{s}(b|a) = \frac{n_\mathbf{s}(b, a)}{N_n(\mathbf{s}, a)}$$

Furthermore, let $N_n(\mathbf{s}) = \sum_{a \in \mathcal{A}} N_n(\mathbf{s}, a)$. □

Remark Note that $\mathbf{Y}_\mathbf{s}^a$ is *i.i.d* only if $\mathbf{s} \in \mathcal{T}$, the set of states for the true model. In general, since we do not necessarily know if any of $n_\mathbf{s}(b, a)$ are close to their stationary frequencies, there is no obvious reason why $\hat{\theta}_\mathbf{s}(b|a)$ shall reflect $\tilde{\theta}_\mathbf{s}(b|a)$. □

Let $\nu_j = \sum_{i \geq j} f(i)$. Note that $\nu_j \rightarrow 0$ as $j \rightarrow \infty$ and that $-\nu_j \log \nu_j \rightarrow 0$ as $\nu_j \rightarrow 0$.

Definition 7. Given a sample sequence with size n obtained from the channel model $p_{\tau, q}$, we define the set of good states, denoted by \tilde{G} , as

$$\tilde{G} = \left\{ \mathbf{w} \in \tilde{\mathcal{T}} : \forall a \in \mathcal{A}, N_n(\mathbf{w}, a) \geq \max \left\{ n\nu_{k_n} \log \frac{1}{\nu_{k_n}}, |\mathcal{A}|^{k_n+1} \log^2 n \right\} \right\}.$$

Remark Note that a state is good if the count of the state is $\geq n^{\alpha_n} \log^2 n = 2^{k_n} \log^2 n$. Therefore, if $2^{k_n} \log^2 n \geq n$, or equivalently $k_n \geq \log n - 2 \log \log n$, no state will be good and

the Theorem below becomes vacuously true. This is not a fundamental weakness in this line of argument—it is known that k_n has to scale logarithmically with n for proper estimation to hold. \square

Remark Observe also that because we do not assume the source has mixed, the theorem below does not imply that the parameters are accurate for contexts shorter than k_n . This is perhaps counterintuitive at first glance, but the below result holds not because of mixing, but because of the fall-off of dependencies. \square

Theorem 10. Let $k_n = \alpha_n \log n$. With probability (conditioned on $Y_{-\infty}^0$) $\geq 1 - \frac{1}{2^{|\mathcal{A}|^{k_n+1} \log n}}$, for all $a \in \mathcal{A}$, $Y_{-\infty}^0$ and $\mathbf{s} \in \tilde{G}$ simultaneously

$$\|\tilde{\theta}_{\mathbf{s}}(\cdot|a) - \hat{\theta}_{\mathbf{s}}(\cdot|a)\|_1 \leq 2\sqrt{\frac{\ln 2}{\log n} + \frac{\ln 2}{\log \frac{1}{\nu_{k_n}}}}$$

Proof As before, let $\nu_{k_n} = \sum_{i \geq k_n} f(i)$ and let n be large enough that $\nu_{k_n} \leq \frac{1}{2}$. Note that Lemma 9 implies that for all sequences $(x_1^n, y_1^n) \in \mathcal{A}^n \times \mathcal{A}^n$ (all steps hold for all $Y_{-\infty}^0$)

$$\begin{aligned} p_{\mathcal{T},q}(y_1^n | x_1^n, Y_{-\infty}^0) &\leq \frac{1}{(1 - \nu_{k_n})^n} \prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a,b \in \mathcal{A}} \tilde{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)} \\ &\leq 4^{n\nu_{k_n}} \prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a,b \in \mathcal{A}} \tilde{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)} \end{aligned}$$

where $\tilde{\theta}_{\mathbf{s}}(b|a)$, $n_{\mathbf{s}}(b,a)$ are defined in Definition 6 and the second inequality is because $(\frac{1}{1-t})^n \leq 4^{nt}$ whenever $0 \leq t \leq \frac{1}{2}$. Now, let B_n be the set of all sequences that satisfy

$$4^{n\nu_{k_n}} \prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a,b \in \mathcal{A}} \tilde{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)} \leq \frac{\prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a,b \in \mathcal{A}} \hat{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)}}{2^{2^{|\mathcal{A}|^{k_n+1} \log n}}}.$$

Now using a construction similar to the context tree weighting algorithm [74], we obtain a

distribution p_c satisfying

$$p_c(y_1^n | x_1^n, Y_{-\infty}^0) \geq \frac{\prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a, b \in \mathcal{A}} \hat{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)}}{2^{|\mathcal{A}|^{k_n+1} \log n}}.$$

Hence, for all sequences in B_n , we have

$$\begin{aligned} p_c(y_1^n | x_1^n, Y_{-\infty}^0) &\geq \frac{\prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a, b \in \mathcal{A}} \hat{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)}}{2^{|\mathcal{A}|^{k_n+1} \log n}} \\ &\geq \frac{4^{(n\nu_{k_n} + |\mathcal{A}|^{k_n+1} \log n)} \prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a, b \in \mathcal{A}} \tilde{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)}}{2^{|\mathcal{A}|^{k_n+1} \log n}} \\ &\geq p_{\tau,q}(y_1^n | x_1^n, Y_{-\infty}^0) 2^{|\mathcal{A}|^{k_n+1} \log n}. \end{aligned}$$

Thus, B_n is the set of sequences y_1^n such that p_c assigns a much higher probability than $p_{\tau,q}$. Such a set B_n can not have high probability under $p_{\tau,q}$.

$$\begin{aligned} p_{\tau,q}(B_n) &= p_{\tau,q} \left\{ y_1^n : p_c(y_1^n | x_1^n, Y_{-\infty}^0) \geq p_{\tau,q}(y_1^n | x_1^n, Y_{-\infty}^0) 2^{|\mathcal{A}|^{k_n+1} \log n} \right\} \\ &\leq \sum_{y_1^n \in B_n} p_c(y_1^n | x_1^n, Y_{-\infty}^0) 2^{-|\mathcal{A}|^{k_n+1} \log n} \leq 2^{-|\mathcal{A}|^{k_n+1} \log n}. \end{aligned}$$

Therefore, with probability $\geq 1 - 2^{-|\mathcal{A}|^{k_n+1} \log n}$, we have

$$\prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a, b \in \mathcal{A}} \tilde{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)} \geq \frac{\prod_{\mathbf{s} \in \tilde{\mathcal{T}}} \prod_{a, b \in \mathcal{A}} \hat{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)}}{4^{(|\mathcal{A}|^{k_n+1} \log n + n\nu_{k_n})}}$$

which implies simultaneously for all $\mathbf{s} \in \tilde{\mathcal{T}}$ and for all $a \in \mathcal{A}$

$$\prod_{b \in \mathcal{A}} \tilde{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)} \geq \frac{\prod_{b \in \mathcal{A}} \hat{\theta}_{\mathbf{s}}(b|a)^{n_{\mathbf{s}}(b,a)}}{4^{(|\mathcal{A}|^{k_n+1} \log n + n\nu_{k_n})}}.$$

The above equation implies that $\tilde{\theta}_{\mathbf{s}}(\cdot|a)$ and $\hat{\theta}_{\mathbf{s}}(\cdot|a)$ are close distributions, since we can

rearrange (take logarithm and divide both sides by $N_n(\mathbf{s}, a)$) to obtain

$$\sum_{b \in \mathcal{A}} \frac{n_{\mathbf{s}}(b, a)}{N_n(\mathbf{s}, a)} \log \frac{\hat{\theta}_{\mathbf{s}}(b|a)}{\tilde{\theta}_{\mathbf{s}}(b|a)} = D\left(\hat{\theta}_{\mathbf{s}}(\cdot|a) \parallel \tilde{\theta}_{\mathbf{s}}(\cdot|a)\right) \leq \frac{2(|\mathcal{A}|^{k_n+1} \log n + n\nu_{k_n})}{N_n(\mathbf{s}, a)},$$

where the first equality follows by writing out the value of the naive estimate, $\hat{\theta}_{\mathbf{s}}(b|a) = n_{\mathbf{s}}(b, a)/N_n(\mathbf{s}, a)$. Since (see for example [80])

$$\frac{1}{2 \ln 2} \|\hat{\theta}_{\mathbf{s}}(\cdot|a) - \tilde{\theta}_{\mathbf{s}}(\cdot|a)\|_1^2 \leq D\left(\hat{\theta}_{\mathbf{s}}(\cdot|a) \parallel \tilde{\theta}_{\mathbf{s}}(\cdot|a)\right),$$

for all $\mathbf{s} \in \tilde{\mathcal{T}}$ and $a \in \mathcal{A}$, we now have with confidence bigger than $1 - 2^{-|\mathcal{A}|^{k_n+1} \log n}$ that

$$\|\tilde{\theta}_{\mathbf{s}}(\cdot|a) - \hat{\theta}_{\mathbf{s}}(\cdot|a)\|_1 \leq \sqrt{\frac{(\ln 2)(|\mathcal{A}|^{k_n+1} \log n + n\nu_{k_n})}{N_n(\mathbf{s}, a)}}.$$

The Theorem follows from our Definition 7 of good states. \square

When the dependencies among strings die down exponentially, we can strengthen Theorem 10 to get convergence rate polynomial in n .

Theorem 11. Suppose $f(i) = \gamma^i$ for some $0 < \gamma < 1$. Let ζ be a nonnegative constant such that $\zeta \geq \frac{-\log \gamma}{\log \mathcal{A} - \log \gamma}$. For $k_n = \frac{\log n}{\log \mathcal{A} - \log \gamma}$ and define

$$\tilde{G} \triangleq \{\mathbf{w} \in \tilde{\mathcal{T}} : \forall a \in \mathcal{A}, N_n(\mathbf{w}, a) \geq n^{\zeta + \frac{\log \mathcal{A}}{\log \mathcal{A} - \log \gamma}}\}.$$

Then, for all $\mathbf{s} \in \tilde{G}$ with probability greater than $1 - 2^{-|\mathcal{A}|^{k_n+1} \log n}$ simultaneously

$$\|\tilde{\theta}_{\mathbf{s}}(\cdot|a) - \hat{\theta}_{\mathbf{s}}(\cdot|a)\|_1 \leq 2 \sqrt{\frac{\ln 2 \cdot ((1 - \gamma)|\mathcal{A}| \log n + 1)}{(1 - \gamma)n^\zeta}}. \quad \square$$

Proof The proof is similar to Theorem 10, but involves more careful but elementary algebra specific to the exponential decay case.

Remark According to definition of good states in Theorem 11 and the fact that $d(\tilde{\mathcal{T}}) =$

k_n , we obtain

$$|\tilde{G}| \leq n^{-\frac{\log \gamma}{\log |\mathcal{A}| - \log \gamma}}, |\tilde{\mathcal{T}}| = n^{\frac{\log |\mathcal{A}|}{\log |\mathcal{A}| - \log \gamma}}$$

implying that if $\gamma \leq 1/|\mathcal{A}|$, all states of $\tilde{\mathcal{T}}$ can potentially be good. \square

6.3 Estimation of State Stationary Probabilities

Note that the parameters $\tilde{\theta}$ associated with any good state can be well estimated from the sample while the rest may not be accurate. From Example 6, we know that the stationary probabilities may be a very sensitive function of the parameters associated with states. It is therefore perfectly possible that we estimate the parameters at all states reasonably well, but are unable to gauge what the stationary probabilities of any state may be. How do we tell, therefore, if we can trust our naive counts of states?

To find deviation bounds for stationary distribution of good states, we construct a new process $\{Z_m\}_{m=1}^\infty$, $Z_m \in \mathcal{T}$ from the process $\{Y_n\}_{n=1}^\infty$. If Y_{n_m} is the $(m+1)^{th}$ symbol in the sequence $\{Y_n\}_{n=1}^\infty$ such that $\mathbf{c}_{\tilde{\mathcal{T}}}(Y_{n_m}^\infty) \in \tilde{G}$, then $Z_m = \mathbf{c}_{\mathcal{T}}(Y_{n_m}^\infty)$. The strong Markov property allows us to characterize $\{Z_m\}_{m=1}^\infty$ as a Markov process with transitions that are lower bounded by those transitions of the process $\{Y_n\}_{n=1}^\infty$ that can be well estimated by the Theorems above. More specifically, let $T_0 = \min \{j \geq 0 : \mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^j) \in \tilde{G}\}$ and let $Z_0 = \mathbf{c}_{\mathcal{T}}(Y_{-\infty}^{T_0})$. For all $m \geq 1$, T_m is the $(m+1)^{th}$ occurrence of a good state in the sequence $\{Y_n\}_{n=1}^\infty$, namely

$$T_m = \min \{j \geq T_{m-1} : \mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^j) \in \tilde{G}\},$$

and $Z_m = \mathbf{c}_{\mathcal{T}}(Y_{-\infty}^{T_m})$. Note that T_m is a *stopping time* [78], and therefore $\{Z_m\}_{m=1}^\infty$ is a Markov chain by itself. Let $\tilde{B} = \{\mathbf{s} \in \tilde{\mathcal{T}} : \mathbf{s} \notin \tilde{G}\}$. The transitions between states $\mathbf{s}, \mathbf{s}' \in \tilde{G}$ are then the minimal, non-negative solution of the following set of equations in

$\{Q(\mathbf{s}|\mathbf{s}'') : \mathbf{s}'' \in \mathcal{A}^{k_n}, \mathbf{s} \in \tilde{G}\}$

$$Q(\mathbf{s}|\mathbf{s}') = p_{\mathcal{T},q}(\mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^1) = \mathbf{s} | \mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^0) = \mathbf{s}') + \sum_{\mathbf{s}'' \in \tilde{B}} p_{\mathcal{T},q}(\mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^1) = \mathbf{s}'' | \mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^0) = \mathbf{s}') Q(\mathbf{s}|\mathbf{s}'')$$

An important point to note here is that if \mathbf{s} and \mathbf{s}' are good states,

$$Q(\mathbf{s}|\mathbf{s}') \geq p_{\mathcal{T},q}(\mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^1) = \mathbf{s} | \mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^0) = \mathbf{s}'),$$

and the lower bound above can be well estimated from the sample as shown in Theorem 10.

Property 1. A few properties about $\{Z_m\}$ are in order. $\{Z_m\}$ is constructed from an irreducible process $\{Y_n\}$, thus $\{Z_m\}$ is irreducible as well. Since $\{Y_n\}$ is positive recurrent, so is $\{Z_m\}$. But despite $\{Y_n\}$ being aperiodic, $\{Z_n\}$ could be periodic as in the Example below. But periodicity of $\{Z_m\}$ can be determined by \tilde{G} alone (because \mathcal{T} , while unknown, is a full, finite \mathcal{A} -ary tree). \square

Example 9. Let $\{Y_n\}$ be a process generated by context tree model $p_{\mathcal{T},q}$ with $\mathcal{T} = \{11, 01, 10, 00\}$ and $q(1|11) = \frac{1}{2}$, $q(1|01) = \epsilon$, $q(1|10) = 1 - \epsilon$, $q(1|00) = \frac{1}{2}$. If $\epsilon > 0$, then $p_{\mathcal{T},q}$ represents a stationary aperiodic Markov process. If $\{Z_n\}$ be the restriction of process $\{Y_n\}$ to $\tilde{G} = \{01, 10\}$, the restricted process will be periodic with period 2. \square

Property 2. Suppose $\{Z_m\}$ is aperiodic. Let μ_Y and μ_Z denote the stationary distribution of the processes $\{Y_n\}$ and $\{Z_m\}$, respectively, with n samples of a sequence $\{Y_n\}$ yielding m_n samples of $\{Z_m\}$. Similarly, let $\mu_Z(\mathbf{s})$ denote the stationary probability of the event $\mathbf{s} \preceq Z$. Then for all $\mathbf{s}, \mathbf{s}' \in \tilde{G}$ with $\mu_Y(\mathbf{s}') \neq 0$,

$$\frac{\mu_Y(\mathbf{s})}{\mu_Y(\mathbf{s}')} \stackrel{wp1}{=} \frac{\lim_{n \rightarrow \infty} \sum_{i=1}^n \frac{\mathbb{1}(\mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^i) = \mathbf{s})}{n}}{\lim_{n \rightarrow \infty} \sum_{i=1}^n \frac{\mathbb{1}(\mathbf{c}_{\tilde{\mathcal{T}}}(Y_{-\infty}^i) = \mathbf{s}')}{n}} = \frac{\lim_{m_n \rightarrow \infty} \sum_{i=1}^{m_n} \frac{\mathbb{1}(\mathbf{s} \preceq Z_i)}{m_n}}{\lim_{m_n \rightarrow \infty} \sum_{i=1}^{m_n} \frac{\mathbb{1}(\mathbf{s}' \preceq Z_i)}{m_n}} \stackrel{wp1}{=} \frac{\mu_Z(\mathbf{s})}{\mu_Z(\mathbf{s}')}. \quad \square$$

For any (good) state \mathbf{s} , let $G_{\mathbf{s}} \subset \mathcal{A}$ be the set of letters that take \mathbf{s} to another good state,

$$G_{\mathbf{s}} = \{b \in \mathcal{A} : \mathbf{c}_{\tilde{\mathcal{T}}}(\mathbf{s}b) \in \tilde{G}\}. \quad (6.4)$$

Our confidence in the empirical counts of good states matching their (aggregated) stationary probabilities follows from a coupling argument, and depends on the following parameter

$$\eta_{\tilde{G}} = \min_{\mathbf{u}, \mathbf{v} \in \tilde{G}} \sum_{b \in G_{\mathbf{u}} \cap G_{\mathbf{v}}} \min \{ \tilde{q}_{\mathbf{u}}(b), \tilde{q}_{\mathbf{v}}(b) \}. \quad (6.5)$$

where in a slight abuse of notation,

$$\tilde{q}_{\mathbf{u}}(b) \stackrel{\text{def}}{=} \sum_{a \in \mathcal{A}} \tilde{q}_{\mathbf{u}}(a, b)$$

Note that for any state $\mathbf{s} \in \mathcal{T}$ of the original process $p_{\tau, q}$, if $\mathbf{u} \preceq \mathbf{s}$

$$\tilde{q}_{\mathbf{u}}(b) = \sum_{a \in \mathcal{A}} p_a \tilde{\theta}_{\mathbf{u}}(b|a) \geq \sum_{a \in \mathcal{A}} p_a \theta_{\mathbf{s}}(b|a) (1 - \nu_{|\mathbf{u}|}) = q_{\mathbf{s}}(b) (1 - \nu_{|\mathbf{u}|})$$

Again, Theorems 10 and 11 allow us to estimate $\eta_{\tilde{G}}$ from the sample since it only depends on parameters associated with good states. The counts of various $\mathbf{w} \in \tilde{G}$ now concentrates as shown in the Theorem below, and how good the concentration is can be estimated as a function of $\eta_{\tilde{G}}$ (and ν_{k_n}) and the total count of all states in \tilde{G} as below. Now \tilde{G} as well as $\eta_{\tilde{G}}$ are well estimated from the sample—thus we can look at the data to interpret the empirical counts of various substrings of the data. Let $\Phi_j = \sum_{i \geq j} \nu_i$. For the following theorem, we require ν_i to be summable. Thus, Φ_j is finite for all j and decreases to 0 as j increases. If $f(i) \sim \gamma^i$, then Φ_j also diminishes as γ^j . But $f(i) \sim \frac{1}{i^r}$ diminishes polynomially, then Φ_j diminishes as $1/j^{r-2}$. If $f(i) = 1/i^{2+\eta}$ for any $\eta > 0$, we therefore satisfy the summability of ν_i . However, $f(i)$ can also diminish as $1/(i^2 \text{ poly}(\log i))$ for appropriate polynomials of $\log i$ for the counts of good states to converge.

Theorem 12. If $\{Z_m\}_{m=1}^{\infty}$ is aperiodic, then for any $t > 0$, $Y_{-\infty}^0$ and $\mathbf{w} \in \tilde{G}$ we have

$$p_{\tau, q}(|N_n(\mathbf{w}) - \tilde{n} \frac{\mu(\mathbf{w})}{\mu(\tilde{G})}| \geq t | Y_{-\infty}^0) \leq 2 \exp \left(- \frac{t^2}{2\tilde{n}} \left(\frac{\eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})}{4\ell_n + \eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})} \right)^2 \right)$$

where ℓ_n is the smallest integer such that $\Phi_{\ell_n} \leq \frac{1}{n}$, \tilde{n} is the total count of good states in the

sample and μ is the stationary distribution of $p_{\mathcal{T},q}$.

Proof We define

$$V_i = \mathbb{E}[N_n(\mathbf{w})|Z_0, Z_1, \dots, Z_i],$$

and observe that $\{V_i\}_{i=1}^{\tilde{n}}$ is a Doob Martingale. Note that $V_0 = \mathbb{E}[N_n(\mathbf{w})|Z_0]$ and $V_{\tilde{n}} = N_n(\mathbf{w})$.

Remark To summarize, we first bound the differences $|V_{i+1} - V_i|$ of the martingale using a coupling argument on two copies of the chain $\{Z_n\}$. Since the memory of the process $p_{\mathcal{T},q}$ could be large, in our proof the coupled chains never actually coalesce in the usual sense but enough that the chance they diverge again within n samples is less than $1/n$. This is where the parameter ℓ_n comes in as well. Once we bound the differences in the martingale $\{V_i\}_{i=1}^{\tilde{n}}$, the theorem follows as an easy application of Azuma's inequality. \square

Now since for all $i \geq 0$

$$\begin{aligned} |V_i - V_{i-1}| &= |\mathbb{E}[N_n(\mathbf{w})|Z_0, \dots, Z_i] - \mathbb{E}[N_n(\mathbf{w})|Z_0, \dots, Z_{i-1}]| \\ &\leq \max_{Z'_i, Z''_i} \left| \mathbb{E}[N_n(\mathbf{w})|Z_0, \dots, Z'_i] - \mathbb{E}[N_n(\mathbf{w})|Z_0, \dots, Z''_i] \right|, \end{aligned}$$

we bound the maximum change in $N_n(\mathbf{w})$ if the i 'th good state was changed into another (good) state. To do so, we use a coupling argument as follows. Let \tilde{G} be the set of good states from Definition 7, and suppose good states occur \tilde{n} times in the sequence. Suppose there are sequences $\{Z'_i\}$ (starting from state Z''_i) and $\{Z''_i\}$ (starting from state Z'_i), both faithful copies of $\{Z_i\}$ yet coupled with a joint distribution ω to be described below. From the coupling argument of Section 5.3.2, we have for $\mathbf{w} \in \tilde{G}$ (hence $|\mathbf{w}| = k_n$) for all ω

$$|\mathbb{E}[N_n(\mathbf{w})|Z_0, \dots, Z'_i] - \mathbb{E}[N_n(\mathbf{w})|Z_0, \dots, Z''_i]| \leq \sum_{j=i+1}^{\tilde{n}} \omega(Z'_j \not\approx^{k_n} Z''_j), \quad (6.6)$$

where $Z'_j \approx^{k_n} Z''_j$ if $\mathbf{c}_{\mathcal{A}^{k_n}}(Z'_j) = \mathbf{c}_{\mathcal{A}^{k_n}}(Z''_j)$.

Description of Coupling Suppose we have $\{Z'_i\}_{i=1}^j$ and $\{Z''_i\}_{i=1}^j$. To obtain Z'_{j+1} and Z''_{j+1} , starting from states Z'_j and Z''_j we run copies $\{Y'_{ji}\}_{i \geq 1}$ and $\{Y''_{ji}\}_{i \geq 1}$ of coupled chains individually faithful to $p_{\tau,q}$. Then Z'_j is the state corresponding to the first time $\{Z_j, Y'_{ji}\}_{i \geq 1}$ hits a context in \tilde{G} . Similarly for Z''_j . Specifically, the chains $\{Y'_{ji}\}_{i \geq 1}$ and $\{Y''_{ji}\}_{i \geq 1}$ are coupled as follows. We generate a number U_{j1} uniformly distributed $\in [0, 1]$. Given $(Z'_j$ and $Z''_j)$ with suffixes \mathbf{u} and \mathbf{v} respectively in \tilde{G} , we let $G_{\mathbf{u}} \in \mathcal{A}$ (and $G_{\mathbf{v}}$ similarly) be the set of symbols in \mathcal{A} defined as in (6.4). We split the interval from 0 to 1 as follows: for all $a \in \mathcal{A}$, we assign intervals $r(a)$ of length $\min\{q_{\mathbf{u}}(a), q_{\mathbf{v}}(a)\}$, in the following order: we first stack the above intervals corresponding to $a \in G_{\mathbf{u}} \cap G_{\mathbf{v}}$ (in any order) starting from 0, and then we put in the intervals corresponding to all other symbols. Now let,

$$(Y'_{j1}, Y''_{j1}) = (a, a) \text{ if } U_{j1} \in r(a).$$

Let

$$C(\mathcal{A}) = \sum_{b \in \mathcal{A}} r(b) = \sum_{b \in \mathcal{A}} \min\{q_{Z'_j}(b), q_{Z''_j}(b)\}. \quad (6.7)$$

be the part of the interval is already filled up. Thus if $U_{j1} < C(\mathcal{A})$, equivalently with probability $C(\mathcal{A})$, the two chains output the same symbol. We use the rest of the interval $[C(\mathcal{A}), 1]$ in any valid way to satisfy the fact that Y'_{j1} is distributed as $p_{\tau,q}(\cdot | Z'_j)$ and Y''_{j1} is distributed as $p_{\tau,q}(\cdot | Z''_j)$. For one standard approach, for all a assign

$$r_{\mathbf{u}}(a) = (q_{\mathbf{u}}(a) - q_{\mathbf{v}}(a))^+ = \max\{q_{\mathbf{u}}(a) - q_{\mathbf{v}}(a), 0\}$$

and similarly $r_{\mathbf{v}}(a)$. Note that only one of $r_{\mathbf{u}}(a)$ and $r_{\mathbf{v}}(a)$ can be strictly positive and that for all a , $r(a) + r_{\mathbf{u}}(a) = q_{\mathbf{u}}(a)$ while $r(a) + r_{\mathbf{v}}(a) = q_{\mathbf{v}}(a)$. Therefore,

$$\sum_{a \in \mathcal{A}} r_{\mathbf{u}}(a) = \sum_{a \in \mathcal{A}} r_{\mathbf{v}}(a) = 1 - C(\mathcal{A}).$$

An example of such construction for binary alphabet is illustrated in Fig. 6.1. in which we have assumed $G_{\mathbf{u}} \cap G_{\mathbf{v}} = \{a_1\}$. We will keep two copies of the interval $[C(\mathcal{A}), 1]$, and if

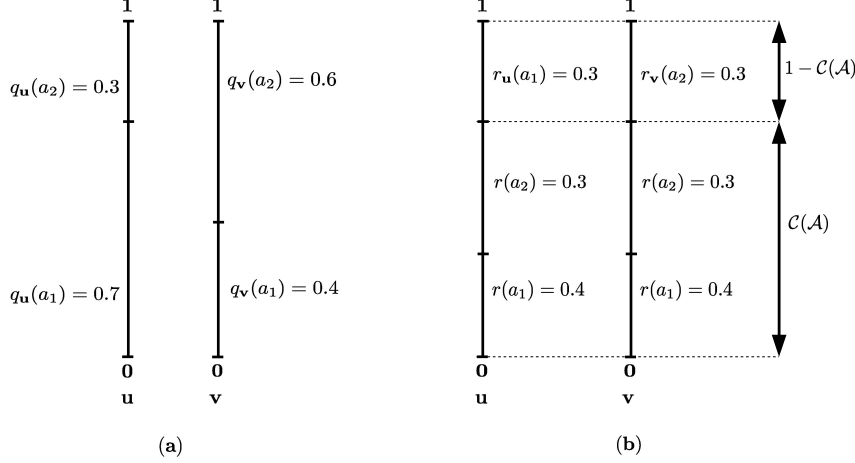


Figure 6.1: (a) The conditional probabilities with which Y'_{j1} and Y''_{j1} have to be chosen respectively are $q_{\mathbf{u}}(\cdot)$ and $q_{\mathbf{v}}(\cdot)$. The line on the left determines the choice of Y'_{j1} and the one on the right the choice of Y''_{j1} . For example, if U_{j1} is chosen uniformly in $[0,1]$, the probability of choosing $Y'_{j1} = a_1$ is $q_{\mathbf{u}}(a_1)$. Instead of choosing Y'_{j1} and Y''_{j1} independently, we will reorganize the intervals in the lines so as to encourage $Y'_{j1} = Y''_{j1}$. (b) Reorganizing the interval $[0, 1]$ according to the described construction. Here $r(a_1) = \min \{q_{\mathbf{u}}(a_1), q_{\mathbf{v}}(a_1)\}$ and similarly for $r(a_2)$. If U_{j1} falls in the interval corresponding to $r(a_1)$, then $(Y'_{j1}, Y''_{j1}) = (a_1, a_1)$. If $U_{j1} > C(\mathcal{A})$ in this example, then $(Y'_{j1}, Y''_{j1}) = (a_1, a_2)$. When U_{j1} is chosen uniformly in $[0,1]$, the probability Y'_{j1} outputs any symbol is the same as in the picture on the left, similarly for Y''_{j1} .

$U_{j1} > C(\mathcal{A})$ we output (Y'_{j1}, Y''_{j1}) based on where U_{j1} falls in both copies. We will stack the first copy of $[C(\mathcal{A}), 1]$ with intervals of length $r_{\mathbf{u}}(a)$ for all a and the second copy of $[C(\mathcal{A}), 1]$ with intervals length $r_{\mathbf{v}}(a)$ for all a . We say $U_{j1} \in (r_{\mathbf{u}}(a), r_{\mathbf{v}}(b))$ if $U_{j1} \in r_{\mathbf{u}}(a)$ in the first copy and $U_{j1} \in r_{\mathbf{v}}(b)$ in the second copy. Furthermore,

$$(Y'_{j1}, Y''_{j1}) = (a, b) \text{ if } U_{j1} \in (r_{\mathbf{u}}(a), r_{\mathbf{v}}(b)).$$

1. If $\mathbf{c}_{\tilde{\tau}}(Z'_j Y'_{j1}) \in \tilde{G}$ and $\mathbf{c}_{\tilde{\tau}}(Z''_j Y''_{j1}) \in \tilde{G}$, we have Z'_{j+1} and Z''_{j+1} .
2. If $Y'_{j1} = Y''_{j1}$ but only one of $\mathbf{c}_{\tilde{\tau}}(Z'_j Y'_{j1}) \in \tilde{G}$ and $\mathbf{c}_{\tilde{\tau}}(Z''_j Y''_{j1}) \in \tilde{G}$, then we have one of Z'_{j+1} and Z''_{j+1} . To get the other, we continue (according to transitions defined by $p_{\tau,q}$) only its corresponding chain till we get a good state.

3. If $Y'_{j1} = Y''_{j1}$, $\mathbf{c}_{\tilde{\mathcal{T}}}(Z'_j Y'_{j1}) \notin \tilde{G}$ and $\mathbf{c}_{\tilde{\mathcal{T}}}(Z''_j Y''_{j1}) \notin \tilde{G}$, we need to continue both chains. We generate Y'_{j2}, Y''_{j2} as we did for the first samples—by generating a new random number U_{j2} uniform in $[0, 1]$, and by coupling as in (6.7) the distributions $q_{Z'_j Y'_{j1}}(\cdot)$ and $q_{Z''_j Y''_{j1}}(\cdot)$ respectively. And continue in this fashion so long as the samples in the two chains remain equal but do not hit good contexts. This will be case that will be most important for us later on.
4. If $Y'_{jl} \neq Y''_{jl}$ at any point and neither chain has seen a good state yet, we just run the chains independently from that point on for how long it takes each to hit a good aggregated state.

Analysis of coupling For any r , let $Z'_r \sim Z''_r$ denote the following event that is a subset of case (1) in the list above,

$$\left\{ Y'_{r1} = Y''_{r1} \text{ and } \mathbf{c}_{\tilde{\mathcal{T}}}(Z'_{r-1} Y'_{r1}) \in \tilde{G} \text{ and } \mathbf{c}_{\tilde{\mathcal{T}}}(Z''_{r-1} Y''_{r1}) \in \tilde{G} \right\}.$$

From (6.5) and using Lemma 9, we can easily show

$$\omega(Z'_r \sim Z''_r | Z'_{r-1}, Z''_{r-1}) \geq \eta_{\tilde{G}}(1 - \nu_{k_n}), \quad (6.8)$$

where the $1 - \nu_{k_n}$ term comes because the parameter $\eta_{\tilde{G}}$ is defined on the aggregated parameters, but Y'_j and Y''_j evolve according to $p_{\mathcal{T}, q}$. Furthermore, if $Z'_i \sim Z''_i$ for the k_n consecutive samples $j - k_n + 1 \leq i \leq j$, then we have $Z'_j \stackrel{k_n}{\approx} Z''_j$. To proceed, once $Z'_j \stackrel{k_n}{\approx} Z''_j$, we would like them to coalesce tighter in every subsequent step, namely we want for all $1 \leq l \leq n$, $Z'_{j+l} \stackrel{k_n+l}{\approx} Z''_{j+l}$. Starting from $Z'_j \stackrel{k_n}{\approx} Z''_j$, one way we can have $Z'_{j+1} \stackrel{k_n+1}{\approx} Z''_{j+1}$ is if $Z'_{j+1} \sim Z''_{j+1}$, or if the chains $\{Y'_{ji}\}_{i \geq 1}$ and $\{Y''_{ji}\}_{i \geq 1}$ evolve through a sequence of $m > 1$ steps before hitting a context in \tilde{G} on the m' th step with $Y'_{jl} = Y''_{jl}$ for each $l \leq m$. This is the situation in case 3 of the list above, but in addition in each step l ,

$$\mathbf{c}_{\mathcal{A}^{k_n}}(\{Z_j, Y'_{ji}\}_{i=1}^l) = \mathbf{c}_{\mathcal{A}^{k_n}}(\{Z_j, Y''_{ji}\}_{i=1}^l),$$

since $Z'_j \stackrel{k_n}{\approx} Z''_j$. Therefore, both chains will hit a common good context in \tilde{G} in m steps. But in addition we will also have

$$Z'_{j+1} \stackrel{k_n+m}{\approx} Z''_{j+1}.$$

Because of the way we have set up our coupling, the probability

$$\begin{aligned} \omega(Y'_{j1} = Y''_{j1} | Z'_j \stackrel{k_n}{\approx} Z''_j) &= \sum_{a \in \mathcal{A}} \min \left\{ q_{Z'_j}(a), q_{Z''_j}(a) \right\} \\ &\geq \sum_{a \in \mathcal{A}} \tilde{q}_{\mathbf{c}_{\tilde{\tau}}(Z'_j)}(a) (1 - \nu_{k_n}) \\ &= 1 - \nu_{k_n}, \end{aligned}$$

where q and \tilde{q} are the model parameters associated with $p_{\tau,q}$ and $p_{\tilde{\tau},\tilde{q}}$ respectively. Similarly

$$\omega\left(Y'_{j(l+1)} = Y''_{j(l+1)} \mid Z'_j \stackrel{k_n}{\approx} Z''_j, \{Y'_{ji}\}_{i=1}^l = \{Y''_{ji}\}_{i=1}^l\right) \geq 1 - \nu_{k_n+l}.$$

Therefore (no matter what m is),

$$\omega(Z'_j \stackrel{k_n+1}{\approx} Z''_j | Z'_{j-1} \stackrel{k_n}{\approx} Z''_{j-1}) \geq \prod_{l=k_n}^{\infty} (1 - \nu_l) \geq 1 - \Phi_{k_n}.$$

Note that we use a very similar argument to obtain for all l

$$\omega(\exists \ell' \leq \ell \text{ s.t. } Z'_{j+\ell'} \stackrel{k_n+\ell}{\approx} Z''_{j+\ell'} | Z'_j \stackrel{k_n}{\approx} Z''_j) \geq \prod_{l=k_n}^{\infty} (1 - \nu_l) \geq 1 - \Phi_{k_n}.$$

because the above event, $\{\exists \ell' \leq \ell \text{ s.t. } Z'_{j+\ell'} \stackrel{k_n+\ell}{\approx} Z''_{j+\ell'}\}$ can happen by going through a sequence tighter and tighter coalesced transitions of $p_{\tau,q}$ (no matter in how many steps we saw contexts in \tilde{G}). And we can easily strengthen the above to say for all l ,

$$\omega(Z'_{j+l} \stackrel{k_n+l}{\approx} Z''_{j+l} | Z'_j \stackrel{k_n}{\approx} Z''_j) \geq 1 - \Phi_{k_n} \tag{6.9}$$

for the same reason. Indeed, we can further strengthen the above statement to note that after we see $Z'_j \stackrel{\ell}{\approx} Z''_j$ for any l , the chance of ever diverging is

$$\omega(\exists l > 0 \text{ s.t. } Z'_{j+l} \not\stackrel{\ell}{\approx} Z''_{j+l} | Z'_j \stackrel{\ell}{\approx} Z''_j) \leq \Phi_\ell. \quad (6.10)$$

Bound on Martingale differences Let ℓ_n be as defined in the statement of this Theorem. In an abuse of notation, we say the chains $\{Z'_i\}$ and $\{Z''_i\}$ have *merged* if for any j , $Z'_j \stackrel{\ell_n}{\approx} Z''_j$, and let τ be the smallest number such that $Z'_\tau \stackrel{\ell_n}{\approx} Z''_\tau$. The probability $\tau > t\ell_n$ can be upper bounded by observing splitting the first $t\ell_n$ samples in blocks of length ℓ_n , and observing that the probability the chains merge in any single block is, using (6.8) k_n times and then (6.9) $\geq \eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})(1 - \nu_{k_n})^{k_n} \geq \eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})/4$. Thus,

$$\omega(\tau > t\ell_n) \leq \left(1 - \eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})/4\right)^t.$$

Furthermore, $\mathbb{E}\tau$ is less than the expected number of blocks before the chains merge in any single block, thus,

$$\mathbb{E}\tau \leq \frac{4\ell_n}{\eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})}.$$

Furthermore, note that for all $j \geq i + 1$,

$$\begin{aligned} \omega(Z'_j \not\stackrel{\ell_n}{\approx} Z''_j) &= \omega(Z'_j \not\stackrel{\ell_n}{\approx} Z''_j \text{ and } \tau < j) + \omega(Z'_j \not\stackrel{\ell_n}{\approx} Z''_j \text{ and } \tau > j) \\ &\stackrel{(a)}{\leq} \Phi_{\ell_n} + \omega(Z'_j \not\stackrel{\ell_n}{\approx} Z''_j \text{ and } \tau > j) \leq \frac{1}{n} + \omega(\tau > j). \end{aligned}$$

where inequality (a) above follows because $Z'_\tau \stackrel{\ell_n}{\approx} Z''_\tau$ by definition and from (6.10). Finally we upper bound (6.6)

$$\sum_{j=i+1}^{\tilde{n}} \omega(Z'_j \not\stackrel{k_n}{\approx} Z''_j) \leq \tilde{n} \cdot \frac{1}{n} + \sum_{j=i+1}^{\tilde{n}} \omega(\tau > j) \leq 1 + \mathbb{E}\tau \leq 1 + \frac{4\ell_n}{\eta_{\tilde{G}}^{k_n} (1 - \Phi_{k_n})}.$$

The final step of the proof comes by bounding the value of $V_0 = \mathbb{E}[N_n(\mathbf{w})|Z_0]$ by a coupling argument very similar to the one above. Suppose $\{Z'_n\}$ and $\{Z''_n\}$ are coupled copies of $\{Z_n\}$, where $\{Z'_n\}$ starts from state Z_0 , while $\{Z''_n\}$ starts from a state chosen randomly according to the stationary distribution of $\{Z_n\}$. The same analysis holds, and from Property 2

$$\left| V_0 - \frac{\mu(\mathbf{w})}{\mu(\tilde{G})} \right| \leq 1 + \frac{4\ell_n}{\eta_{\tilde{G}}^{k_n}(1 - \Phi_{k_n})}$$

as well. (The theorem only has at most constant confidence if the upper bound above $\geq t/2\sqrt{\tilde{n}}$.) The Theorem now follows by a direct application of Azuma's inequality. \square

Remark Note that if the dependencies die down exponentially, namely $f(i) = \gamma^i$ for some $0 < \gamma < 1$, then $\ell_n = \lceil \log(n/(1 - \gamma)^2)/\log 1/\gamma \rceil$. If the dependencies die down polynomially, namely $f(i) = 1/i^r$ for some $r > 2$, then $\ell_n \geq 2 + \left(\frac{n}{(r-1)(r-2)}\right)^{\frac{1}{r-2}}$. Furthermore, for $k_n = \mathcal{O}(\log n)$, if $\eta_{\tilde{G}}^{2k_n} \leq 1/\tilde{n}$, or $\ell_n \geq \sqrt{\tilde{n}}$, the theorem becomes vacuous. \square



Gaussian Approximation of Finite Random Sum

Let $\{X_i\}$ denote a sequence of i.i.d. random variables, and let $S_n = \sum_{i=1}^n X_i$. Set

$$S_n^* = \frac{S_n - n\mu}{\sigma\sqrt{n}},$$

where $\mu = E[X_i]$ and $\sigma^2 = \text{Var}(X_i)$. Then, using central limit theorem

$$P\{S_n^* \leq t\} \rightarrow F_z(t) \text{ as } n \rightarrow \infty,$$

where $F_z(t)$ is the CDF of standard normal distribution.

Theorem 13 (Berry-Esseen Theorem [81]). *Set $\rho = E[|X_i - \mu|^3]$. Then*

$$\delta_n = |P\{S_n^* \leq t\} - F_z(t)| \leq \frac{\rho}{2\sigma^3\sqrt{n}}.$$

When $\{X_i\}$ are uniform, the Berry-Essen bound is

$$\delta_n = \frac{0.65}{\sqrt{n}}.$$

However, numerical simulation shows that δ_n should shrink quicker than $O(1/\sqrt{n})$. The following bound shows that in uniform case, δ_n decreases as $O(1/n)$.

Theorem 14 (Uspensky bound [82]). *When $X_i \in U[a, b]$, and consider*

$$\delta_n = \sup_{t \in \mathbb{R}} |F_n(t) - Fz(t)|.$$

Then,

$$\delta_n = \frac{1}{7.5\pi n} + \frac{1}{\pi} \left(\frac{2}{\pi}\right)^n + \frac{12}{\pi^3 n} \exp(-\pi^2 n/24).$$

B

Characteristic function $G_{Z_\ell|Y_{k-\ell}}(t)$

Under the assumption that $Y_{k-\ell} = y_{k-\ell}$ and $X_k = x_k$ are given, Z_ℓ is the product of two Gaussian random variables, which can be rewritten as

$$Z_\ell = \Gamma \Omega = \Gamma_\ell^{(k)} (Y_{k-\ell} - E_{k-\ell}) \quad (\text{B.1})$$

where $\Gamma \sim \mathcal{N}(\gamma_\ell, g_\ell)$ and $\Omega \sim \mathcal{N}(y_{k-\ell} - \mu_e, \sigma_e^2)$. Then, the characteristic function for the product of two normal random variables $\Gamma \Omega$, denoted by $G_{Z_\ell|Y_{k-\ell}}(t)$, is computed as

$$\begin{aligned}
G_{Z_\ell|Y_{k-\ell}}(t) &= \mathbb{E}[e^{i\Gamma\Omega t} | Y_{k-\ell} = y_{k-\ell}] \\
&= \mathbb{E} \left[\mathbb{E} \left[e^{i\Gamma\Omega t} | \Omega, Y_{k-\ell} = y_{k-\ell} \right] \right] \\
&= \mathbb{E} \left[e^{i\gamma_\ell\Omega t - \frac{1}{2}g_\ell\Omega^2 t^2} | Y_{k-\ell} = y_{k-\ell} \right] \\
&= \frac{1}{\sqrt{2\pi\sigma_e^2}} \int_{-\infty}^{\infty} e^{(i\gamma_\ell\omega t - \frac{1}{2}g_\ell\omega^2 t^2)} e^{-\frac{(\omega + \mu_e - y_{k-\ell})^2}{2\sigma_e^2}} d\omega \\
&= \frac{\exp\left(\frac{-t^2\left((y_{k-\ell} - \mu_e)^2 g_\ell + \gamma_\ell^2 \sigma_e^2\right) + 2it(y_{k-\ell} - \mu_e)\gamma_\ell}{2(1 + g_\ell\sigma_e^2 t^2)}\right)}{\sqrt{1 + g_\ell\sigma_e^2 t^2}}.
\end{aligned}$$

Bibliography

- [1] C. E. Shannon, “A mathematical theory of communication,” *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 623–656, Jul./Oct. 1948.
- [2] B. Vasic and E. M. Kurtas, *Coding and Signal Processing for Magnetic Recording Systems*. Abingdon: CRC Press, 2004. [Online]. Available: <https://cds.cern.ch/record/994193>
- [3] J.-D. Lee, S.-H. Hur, and J.-D. Choi, “Effects of floating-gate interference on NAND flash memory cell operation,” *IEEE Electron Device Letters*, vol. 23, no. 5, pp. 264–266, May 2002.
- [4] D. J. K. Farzan, “Coding schemes for chip-to-chip interconnect applications,” *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 4, pp. 393–406, Apr 2006.
- [5] A. Jiang, H. Li, and J. Bruck, “On the capacity and programming of flash memories,” *IEEE Trans. Inform. Theory*, vol. 58, no. 3, pp. 1549–1564, Mar. 2012.
- [6] H. Lue, T. Hsu, S. Wang, E. Lai, K. Hsieh, R. Liu, and C. Lu, “Study of incremental step pulse programming (ispp) and sti edge effect of be-sonos nand flash,” *Proc. IEEE Int. Symp. on Reliability Physics*, vol. 30, no. 11, pp. 693–694, 2008.
- [7] K. Suh and et al., “A 3.3 v 32 mb nand flash memory with incremental step pulse

- programming,” *IEEE Journal of Solid-State Circuits*, vol. 30, no. 11, pp. 1149–1156, November 1995.
- [8] P. H. S. A. V. E. Yaakobi, A. Jiang and J. K. Wolf, “On the parallel programming of flash memory cells,” in *Proc. IEEE Inform Theory Workshop*, Dublin, Ireland, Aug-Sep. 2010, pp. 1–5.
- [9] A. Jiang and H. Li, “Optimized cell programming for flash memories,” in *Proc. IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, 2009, pp. 914–919.
- [10] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, “Rank modulation for flash memories,” *IEEE Trans. Inform. Theory*, vol. 55, no. 6, pp. 2659–2673, Jun. 2009.
- [11] Y. Cassuto, M. Schwartz, V. Bohossian, and J. Bruck, “Codes for asymmetric limited-magnitude errors with application to multilevel flash memories,” *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1582–1595, April 2010.
- [12] G. Dong, N. Xie, and T. Zhang, “On the use of soft-decision error-correction codes in NAND flash memory,” *IEEE Trans. Circuits Syst.–I: Reg. Papers*, vol. 58, no. 2, pp. 429–439, Feb. 2011.
- [13] J. Kim and W. Sung, “Low-energy error correction of NAND flash memory through soft-decision decoding,” *EURASIP Journal on Advances in Signal Processing 2012*, 2012:195.
- [14] J. Wang, T. Courtade, H. Shankar, and R. D. Wesel, “Soft information for LDPC decoding in flash: mutual-information optimized quantization,” in *Proc. IEEE GLOBECOM 2011*, Houston, Texas, USA, Dec. 2011.
- [15] G. Dong, S. Li, and T. Zhang, “Using data postcompensation and predistortion to tolerate cell-to-cell interference in MLC NAND flash memory,” *IEEE Trans. Circuits Syst.–I: Reg. Papers*, vol. 57, no. 10, pp. 2718–2728, Oct. 2010.

- [16] D. Park and J. Lee, “Floating-gate coupling canceller for multi-level cell NAND flash,” *IEEE Trans. Magn.*, vol. 47, no. 3, pp. 624–628, Mar. 2011.
- [17] S. Gregori, A. Cabrini, O. Khouri, and G. Torelli, “On-chip error correcting techniques for new-generation flash memories,” *Proceedings of the IEEE*, vol. 91, no. 4, pp. 602–616, Apr. 2003.
- [18] E. Yaakobi, S. Kayser, P. H. Siegel, A. Vardy, and J. K. Wolf, “Codes for write-once memories,” *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. 58, no. 9, pp. 5985–5999, SEPTEMBER 2012.
- [19] J. Rissanen, “A universal data compression system,” *IEEE Transactions on Information theory*, vol. 29, no. 5, pp. 656–664, Sep 1983.
- [20] P. Bühlmann and A. Wyner, “Variable length Markov chains,” *Annals of Statistics*, vol. 27, no. 2, pp. 480–583, 1999.
- [21] I. Csiszár and Z. Talata, “Context tree estimation for not necessarily finite memory processes, via bic and mdl,” *IEEE Transactions on Information theory*, vol. 52, no. 3, Mar 2006.
- [22] A. Garivier, “Consistency of the unlimited BIC context tree estimator,” *IEEE Transactions on Information theory*, vol. 52, no. 10, pp. 4630–4635, Sep 2006.
- [23] I. Csiszár, “Large-scale typicality of Markov sample paths and consistency of MDL order estimators,” *IEEE Transactions on Information theory*, vol. 48, no. 6, pp. 1616–1628, Jun 2002.
- [24] I. Csiszár and P. C. Shields, “The consistency of the BIC Markov order estimator,” *Annals of Statistics*, vol. 28, pp. 1601–1619, 2000.
- [25] A. Galves, V. Maume-Deschamps, and B. Schmitt, “Exponential inequalities for VLMC empirical trees,” *ESAIM: Probability and Statistics*, vol. 12, pp. 219–229, Jan 2008.

- [26] A. Garivier and F. Leonardi, “Context tree selection: A unifying view,” *Stochastic Processes and their Applications*, vol. 121, no. 11, pp. 2488–2506, Nov 2011.
- [27] I. Csiszár and Z. Talata, “On rate of convergence of statistical estimation of stationary ergodic processes,” *IEEE Transactions on Information theory*, vol. 56, no. 8, pp. 3637–3641, Aug 2010.
- [28] A. Galves and F. Leonardi, “Exponential inequalities for empirical unbounded context trees,” *In and Out of Equilibrium 2*, vol. 60, pp. 257–269, 2008.
- [29] D. Arnold, H.-A. Loeliger, P. Vontobel, A. avcic, and W. Zeng, “Simulation-based computation of information rates for channels with memory,” *IEEE Transactions on Information Theory*, vol. 52, pp. 498– 3508, 2006.
- [30] P. Vontobel, A. Kavcic, D. Arnold, and H. Loeliger, “A generalization of the blahut arimoto algorithm to finite state channels,” *IEEE Transactions on Information Theory*, vol. 54, no. 5, May 2008.
- [31] P. Sadeghi, P. Vontobel, and R. Shams, “Optimization of information rate upper and lower bounds for channels with memory,” *IEEE Transactions on Information theory*, vol. 55, no. 2, pp. 663–688, Feb 2009.
- [32] D. M. Arnold, H. Loeliger, P. Vontobel, A. Kavčić, and W. Zeng, “Simulation-based computation of information rates for channels with memory,” *IEEE Transactions on Information theory*, vol. 52, no. 8, pp. 3498–3508, Aug 2006.
- [33] J. Chen and P. H. Siegel, “Markov processes asymptotically achieve the capacity of Finite-State Intersymbol Interference channels,” *IEEE Transactions on Information theory*, vol. 54, no. 3, pp. 1295–1303, March 2008.
- [34] F. Alajaji, “New results on the analysis of discrete communication channels with memory,” Ph.D. dissertation, 1994.
- [35] H. C. Tijms, *A First Course in Stochastic Models*. John Wiley & Sons, Ltd, 2003.

- [36] R. G. Gallager, *Stochastic Processes, Theory for Applications*. Cambridge University Press, Cambridge, UK, 2013.
- [37] S. Mittal and J. S. Vetter, “A survey of software techniques for using non-volatile memories for storage and main memory systems,” *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTING SYSTEMS*, June 2015.
- [38] A. Sampson, J. Nelson, K. Strauss, and L. Ceze, “Approximate storage in solid-state memories,” in *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, New York, 2013, pp. 25–36.
- [39] Z. Wang and J. Bruck, “Error patterns in mlc nand flash memory: Measurement, characterization, and analysis,” in *Design, Automation and Test in Europe Conference*, 2012.
- [40] M. Asadi, X. Huang, A. Kavcic, and N. Santhanam, “Optimal detector for multilevel nand flash memory channels with intercell interference,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 825–835, 2014.
- [41] R. G. Gallager, *Information Theory and Reliable Communication*. New York: John Wiley & Sons, Inc, 1968.
- [42] J. Chen and P. H. Siegel, “Markov processes asymptotically achieve the capacity of finite-state intersymbol interference channels,” *IEEE Trans. Inform. Theory*, vol. 54, no. 3, pp. 1295–1303, Mar. 2008.
- [43] P. O. Vontobel, A. Kavčić, D. M. Arnold, and H.-A. Loeliger, “A generalization of the Blahut-Arimoto algorithm to finite-state channels,” *IEEE Trans. Inform. Theory*, vol. 54, no. 5, pp. 1887–1918, May 2008.
- [44] G. D. Forney, Jr., “The Viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.
- [45] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for

- minimizing symbol error rate,” *IEEE Trans. Inform. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.
- [46] X. Huang, A. Kavcic, X. Ma, G. Dong, and T. Zhang, “Optimization of achievable information rates and number of levels in multilevel flash memories,” in *ICN 2013 : The Twelfth International Conference on Networks*, Seville, Spain, Jan. 27-Feb. 1 2013, pp. 125–131.
- [47] M. Marrow, “Detection and modeling of 2-dimensional signals,” Ph.D. thesis, University of California, San Diego, 2004.
- [48] N. R. M. D. J. Daley, “Asymptotic behavior of the variance of renewal processes and random walks,” *Annals of Probability*, vol. 6, no. 3, pp. 516–521, 1978.
- [49] J. Chang, “Inequalities for the overshoot,” *Annals of Applied Probability*, vol. 4, no. 4, pp. 1223–1233, 1994.
- [50] M. Asadi, Z. Chen, and E. Haratsch, “Capacitance coupling parameter estimation in flash memories,” Patent 14/155,687, 2014.
- [51] R. J. Larsen and M. L. Marx, *An introduction to mathematical statistics and its applications; 5th ed.* Boston, MA: Prentice Hall, 2012. [Online]. Available: <https://cds.cern.ch/record/1456977>
- [52] J. Bellorado, “Noise sources in nand flash memory,” Non-Volatile Memory Workshop, March, 2013.
- [53] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, “Threshold Voltage Distribution in MLC NAND Flash Memory: Characterization, Analysis, and Modeling ,” in *DATE*, 2013.
- [54] D. Park and J. Lee, “Coupling canceller maximum-likelihood (CCML) detection for multi-level cell NAND flash memory,” *IEEE Trans. Consumer Electron.*, vol. 57, no. 1, pp. 160–163, Feb. 2011.

- [55] X. Wang, G. Dong, L. Pan, and R. Zhou, “Error correction codes and signal processing in flash memory,” in *Flash Memories*, I. S. Stievano, Ed. InTech, 2011, pp. 57–82.
- [56] A. Kavčić, X. Ma, and N. Varnica, “Matched information rate codes for partial response channels,” *IEEE Trans. Inform. Theory*, vol. 51, no. 3, pp. 973–989, Mar. 2005.
- [57] J. B. Soriaga, H. D. Pfister, and P. H. Siegel, “Determining and approaching achievable rates of binary intersymbol interference channels using multistage decoding,” *IEEE Trans. Inf. Theory*, vol. 53, no. 4, pp. 1416–1429, Apr. 2007.
- [58] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, “Introduction to flash memory,” *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, Apr. 2003.
- [59] J. Chen and P. H. Siegel, “On the symmetric information rate of two-dimensional finite state ISI channels,” *IEEE Trans. Inform. Theory*, vol. 52, no. 1, pp. 227–236, Jan. 2006.
- [60] O. Shental, N. Shental, and S. S. (Shitz), “On the achievable information rates of finite-state input two-dimensional channels with memory,” in *Proc. IEEE. Int. Symp. Inform. Theory*, Adelaide, Australia, Sept. 2005, pp. 2354–2358.
- [61] H.-A. Loeliger and M. Molkaiaie, “Simulation-based estimation of the partition function and the information rate of two-dimensional models,” in *Proc. IEEE Int. Symp. Inform. Theory*, Toronto, Canada, July 6-11 2008, pp. 1113–1117.
- [62] M. Molkaiaie and H.-A. Loeliger, “Monte Carlo algorithms for the partition function and information rates of two-dimensional channels,” *IEEE Trans. Inform. Theory*, vol. 59, no. 1, pp. 495–503, Jan. 2013.
- [63] A. Kavčić, X. Huang, B. Vasic, W. Ryan, and M. F. Erden, “Channel modeling and capacity bounds for two-dimensional magnetic recording,” *IEEE Trans. Magnetics*, vol. 46, no. 3, pp. 812–818, Mar. 2010.
- [64] X. Chen and K. M. Chugg, “Near-optimal data detection for two-dimensional

- ISI/AWGN channels using concatenated modeling and iterative algorithms,” in *Proc. Int. Conf. Commun.*, vol. 2, Atlanta, Georgia, Jun. 1998, pp. 952–956.
- [65] Y. Wu, J. A. O’Sullivan, N. Singla, and R. S. Indeck, “Iterative detection and decoding for separable two-dimensional intersymbol interference,” *IEEE Trans. Magn.*, vol. 39, no. 4, pp. 2115–2120, July 2003.
- [66] M. Marrow and J. K. Wolf, “Iterative detection of 2-dimensional ISI channels,” in *Proc. ITW 2003*, Paris, France, Mar. 31-Apr. 4 2003, pp. 131–134.
- [67] L. Huang, G. Mathew, and T. C. Chong, “Reduced complexity Viterbi detection for two-dimensional optical recording,” *IEEE Trans. Consumer Electron.*, vol. 51, no. 1, pp. 123–129, Feb. 2005.
- [68] S. Nabavi and B. V. K. V. Kumar, “Detection methods for holographic data storage,” in *2006 Optical Data Storage Meeting*, Montreal, Canada, Apr. 2006, pp. 156–158, Paper TuC3.
- [69] M. Keskinöz and B. V. K. V. Kumar, “Efficient modeling and iterative magnitude-squared decision feedback equalization (dfe) for volume holographic storage channel,” in *Proc. Int. Conf. Commun.*, vol. 9, Helsinki, Finland, June 11-14 2001, pp. 2696–2700.
- [70] O. Shental, A. J. Weiss, N. Shental, and Y. Weiss, “Generalized belief propagation receiver for near-optimal detection of two-dimensional channels with memory,” in *Proc. IEEE Inform. Theory Workshop*, San Antonio, Texas, Oct. 2004, pp. 225–229.
- [71] N. Singla, J. A. O’Sullivan, R. S. Indeck, and Y. Wu, “Iterative decoding and equalization for 2-D recording channels,” *IEEE Trans. Magn.*, vol. 38, no. 5, pp. 2328–2330, Sep. 2002.
- [72] A. H. J. Immink and *et al.*, “Signal processing and coding for two-dimensional optical storage,” in *Proc. GLOBECOM 2003*, vol. 7, San Francisco, California, Dec. 2003, pp. 3904–3908.

- [73] A. Kavčić, X. Ma, and M. Mitzenmacher, “Binary intersymbol interference channels: Gallager codes, density evolution, and code performance bounds,” *IEEE Trans. Inform. Theory*, vol. 49, no. 7, pp. 1636–1652, July 2003.
- [74] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, “The context-tree weighting method: basic properties.” *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 653–664, 1995.
- [75] W. Feller, *An Introduction to Probability Theory and Its Applications*. John Wiley and Sons; 2nd Edition, 1957, vol. 1.
- [76] T. J. Tjalkens, Y. Shtarkov, and F. Willems, “Sequential weighting algorithms for multi-alphabet sources,” in *6th Joint Swedish-Russian International Workshop on Information Theory*, August 1993, pp. 230–234.
- [77] P. Ferrari and A. Galves, “Coupling and regeneration for stochastic processes,” Notes for a minicourse presented in XIII Escuela Venezolana de Matemáticas, 2000.
- [78] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*. American Mathematical Society, 2009.
- [79] V. Guruswami, “Rapidly mixing Markov chains: A comparison of techniques,” Writeup at MIT Laboratory for Computer Science, 2000.
- [80] T. Cover and J. Thomas, *Elements of Information Theory*. John Wiley and sons., 1991.
- [81] R. Durrett, *Probability: Theory and Examples*. Pacific Grove, CA: Wadsworth & Brooks/Cole, 1991.
- [82] J. Uspensky, *Introduction to mathematical probability*. New York, McGraw-Hill, 1937.