UNIVERSITY OF CALIFORNIA, SAN DIEGO

Error Characterization, Channel Modeling and Coding for Flash Memories

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering
(Communication Theory and Systems)

by

Veeresh Taranalli

Committee in charge:

2017

DEDICATION

To my parents.

EPIGRAPH

An approximate answer to the right problem
is worth a good deal more than
an exact answer to an approximate problem.
—John Tukey

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

My Ph.D. at UC San Diego would not have been possible without the unwavering support and encouragement of many people whom I would like to thank here.

First, I would like to thank my advisor Prof. Paul H. Siegel for his excellent teaching of courses on coding theory which inspired me to transfer to the Ph.D. program from the M.S. program at UC San Diego. It was Prof. Siegel's encouragement that made this transition possible. During the later years of my Ph.D., it was not only Prof. Siegel's excellent guidance and his attention to detail, but also his kindness and patience in listening to my various ideas that helped me complete my research successfully. I would sincerely like to thank Prof. Young-Han Kim, Prof. Laurence B. Milstein, Prof. Steven Swanson and Prof. Alexander Vardy for their time and effort to serve on my doctoral committee. I would also like to thank all the UC San Diego faculty members for the various intellectually stimulating courses they taught me in extensive topics whose knowledge will serve me for the rest of my life. I would also like to thank my B.Tech advisors at NITK Surathkal, Prof. Sumam David and Prof. U. Sripati for helping me develop a great foundation of technical knowledge on which to build on.

I would like to thank CMRR-STAR group members and alumni for providing an intellectually stimulating environment for research. I would like to thank Aman Bhatia for helping me develop my knowledge of polar codes through our discussions. A special thanks to Hironori Uchikawa for providing me valuable insights into the practical operations of flash memories and applicable coding techniques and also for introducing me to the problem of ECC performance estimation in flash memories that forms the basis of this dissertation. I would like to thank Sarit Buzaglo and Arman Fazeli for our delightful discussions on polar codes over the past year. I would also like to thank Zachary Blair, Brian Butler, Bing Fan, Pengfei Huang, Seyhan Karakulak, Scott Kayser, Yi Liu, Minghai Qin, Osamu Torii, Wei Wu, Xiaojie Zhang for various discussions during my time at UC San Diego that have enriched my knowledge and research.

I would also like to thank Iris Villanueva, Ray Descoteaux and all the other CMRR staff members for providing me all the support and making my stay at CMRR such an enjoyable and memorable experience.

Finally I would like to thank my parents, Uma Taranalli and Prof. Ashok Taranalli, my sister and brother-in-law, Rashmi Taranalli and Praveen Dodagoudar and my fiancée, Manjusha Bolishetty for their love and support. I would also like to thank my friends, Vaibhav Bedia, Krishna Bharadwaj, Vanamali Bhat, Niranjana Bhatta, Sattvik Chakravarthy, Krishna Pavan, Srikanth Prabhu and Sudharsan Seshadri etc. for their friendship which has been a constant source of inspiration for me.

This research was supported in part by the National Science Foundation under Grants CCF-1116739 and CCF-1405119, and by the Toshiba Corporation, Japan, and by the Center for Memory and Recording Research, University of California, San Diego.

Chapter 2 contains material from the papers: Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "Error analysis and inter-cell interference mitigation in multi-level cell flash memories," in Proc. IEEE International Conference on Communications (ICC), London, June 2015, pp. 271–276; Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "Channel models for multi-level cell flash memories based on empirical error analysis," IEEE Transactions on Communications, vol. 64, no. 8, pp. 3169–3181, August 2016. The dissertation author was the primary investigator and author of these papers, and co-authors have approved the use of the material for this dissertation.

Chapter 3 is in part a reprint of the paper: Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "Channel models for multi-level cell flash memories based on empirical error analysis," IEEE Transactions on Communications, vol. 64, no. 8, pp. 3169–3181, August 2016. The dissertation author was the primary investigator and author of these papers, and co-authors have approved the use of the material for this dissertation.

Chapter 4 is in part a reprint of the paper: Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel "On the capacity of the beta-binomial channel model for multi-level cell flash memories," IEEE Journal on Selected Areas in Communications, vol. 34, no. 9, pp. 2312–2324, September 2016. The dissertation author was the primary investigator and author of these papers, and co-authors have approved the use of the material for this dissertation.

Chapter 5 is a reprint of the paper: Veeresh Taranalli, and Paul H. Siegel, "Adaptive linear programming decoding of polar codes," in Proc. IEEE International Symposium on Information Theory (ISIT), Hawaii, June-July 2014, pp. 2982–2986. The dissertation author was the primary investigator and author of these papers, and co-authors have approved the use of the material for this dissertation.

VITA

2009	B. Tech in Electronics and Communication Engineering, National Institute of Technology Karnataka, India
2013	M. S. in Electrical Engineering (Communication Theory and Systems), University of California San Diego, USA
2017	Ph. D. in Electrical Engineering (Communication Theory and Systems), University of California San Diego, USA

PUBLICATIONS

Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "On the capacity of the beta-binomial channel model for multi-level cell flash memories," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 9, pp. 2312–2324, September 2016.

Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "Channel models for multi-level cell flash memories based on empirical error analysis," *IEEE Transactions on Communications*, vol. 64, no. 8, pp. 3169–3181, August 2016.

Sarit Buzaglo, Arman Fazeli, Paul H. Siegel, Veeresh Taranalli, and Alexander Vardy, "On efficient decoding of polar codes with large kernels," submitted to *IEEE Wireless Communications and Networking Conference, Workshop on Polar Coding in Wireless Communications: Theory and Implementation*, 2017.

Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "Error analysis and inter-cell interference mitigation in multi-level cell flash memories," in *Proc. IEEE International Conference on Communications (ICC)*, London, June 2015, pp. 271–276.

Aman Bhatia, Veeresh Taranalli, Paul H. Siegel, Shafa Dahandeh, Anantha Raman Krishnan, Patrick Lee, Dahua Qin, Moni Sharma, and Teik Yeo, "Polar codes for magnetic recording channels," in *Proc. IEEE Information Theory Workshop (ITW)*, Jerusalem, April–May 2015.

Veeresh Taranalli, and Paul H. Siegel, "Adaptive linear programming decoding of polar codes," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Hawaii, June–July 2014, pp. 2982–2986.

ABSTRACT OF THE DISSERTATION

Error Characterization, Channel Modeling and Coding for Flash Memories

by

Veeresh Taranalli

Doctor of Philosophy in Electrical Engineering
(Communication Theory and Systems)

University of California, San Diego, 2017

Professor Paul H. Siegel, Chair

NAND Flash memories have become a widely used non-volatile data storage technology and their application areas are expected to grow in the future with the advent of cloud computing, big data and the internet-of-things. This has led to aggressive scaling down of the NAND flash memory cell feature sizes and also increased adoption of flash memories with multiple cell levels to increase the data storage density. These factors have adversely affected the reliability of flash memories.

In this dissertation, our main goal is to perform detailed characterization of the errors that occur in multi-level cell (MLC) flash memories and develop novel mathematical channel models that better reflect the measured error characteristics than do current models. The channel models thus developed are applied to error correcting code (ECC)

frame error rate (FER) performance estimation in MLC flash memories and to estimating the flash memory channel capacity as represented by the channel models. We also utilize the characterization of inter-cell interference (ICI) errors to evaluate the performance of constrained coding schemes that mitigate ICI and improve the reliability of flash memories.

In Chapter 5, which is self-contained, we propose and study modifications to adaptive linear programming decoding techniques applied to decoding polar codes. We also propose a reduced complexity representation of the polar code sparse factor graph, resulting in time complexity improvements in the adaptive LP decoder.

1 Introduction

1.1 Background

The data storage industry is in a transition phase today, moving from mechanical hard disk drives to semiconductor-based solid state drives as the storage media for majority of the applications. NAND flash memory technology is a fundamental building block of the ubiquitous solid state drives. The demand for NAND flash memories is mainly driven by their ability to execute read/write operations with very small latencies compared to their mechanical counterparts such as hard disk drives. Apart from this, NAND flash memories also provide high density of data storage which has led to their success in portable consumer electronics such as smartphones, wearable devices and more recently, in laptops. The demand for NAND flash memories is also expected to grow for enterprise applications, mainly for cloud computing data centers. However, NAND flash memories also have certain limitations such as small endurance/lifetime and poor reliability. The endurance and reliability of flash memories are adversely affected due to increasing density of flash memory cells and simultaneous reduction of the flash memory cell feature sizes (the current generation is about 15-19nm).

Coding, signal processing and information-theoretic tools can be used to improve the endurance and reliability of NAND flash memories for practical applications. Data storage can be viewed as a form of digital communication wherein messages are transmitted across time (present to future) rather than across space as in conventional digital communication. The communication channel is thus the data storage medium. In his seminal paper in 1948 [1], Claude Shannon established fundamental bounds on the maximum achievable rate of transmission of information through a communication channel and thus introduced the field of information theory. In that same paper [1], Shannon also studied constrained systems that gave birth to the field of constrained coding. Error correction codes such as Hamming codes, BCH and Reed-Solomon (RS) codes have been extensively used in data storage as well as wired and wireless communication systems. More recently, low density parity check (LDPC) codes, spatially coupled LDPC codes and polar codes represent advances in error correction coding. Constrained coding techniques which prevent certain bad data patterns from being written to the storage media have also been successfully used in hard disk drives for inter-symbol interference (ISI) mitigation and timing recovery.

Error correction coding (ECC) and constrained coding techniques can be used to improve the reliability of NAND flash memories. However, to derive maximum benefit, it is essential to study and understand the NAND flash memory channel in terms of the error mechanisms and characteristics of the errors at various levels such as the bit and cell level and the dependence of errors on programmed data patterns. Such error characterization is presented in Chapter 2. For NAND flash memories, ECC frame error rate (FER) performance needs to be evaluated at extremely low levels such as 10^{-12} in practice to accurately quantify the endurance of the flash memory device. This requires simple but accurate mathematical channel models that capture the characteristics of errors and hence are able to provide good ECC FER performance estimates. Such channel models derived from empirical data are presented in Chapter 3. It is also critical to quantify the information-theoretic capacity of NAND flash memory channels to be able to appropriately allocate redundancy in practice. In Chapter 4, we present capacity estimates for NAND flash memories using channel models derived from empirical data.

In this chapter, we first provide a brief introduction to NAND flash memories in Section 1.2. Section 1.3 describes the various error mechanisms in NAND flash memories and Section 1.4 presents a detailed overview of the following chapters in this dissertation.

1.2 Design and Structure of Flash Memories

The fundamental data storing unit in NAND flash memories is a floating-gate transistor commonly referred to as a cell. A cell can be programmed to hold different levels of charge and these charge levels represent the data bits stored in a cell. The programming operation consists of pumping electrons into the floating-gate of a cell using the channel hot electron injection mechanism. To read the data stored in a cell, a voltage referred to as the threshold voltage (V_{th}) is applied to the control gate of a cell while measuring the drain current to determine the logical bit represented by the cell. A cell is erased by using the Fowler-Nordheim electron tunneling mechanism, which consists of applying a strong electric field across the oxide layer to push the electrons out of the floating-gate of the cell. For more details on flash memory cell operations, we refer the reader to [2]. Typically, a programmed cell is represented by a logical '0' and an erased cell is represented by a logical '1' for single-level cell (SLC) flash memories capable of storing 1 bit/cell. Other commonly used cells in today's flash memories are capable of holding up to 4 and 8 distinct charge levels (2, 3 bits/cell respectively) and are referred to as multi-level cell (MLC) and three-level cell (TLC) flash memories respectively. These flash memory cells are organized into a rectangular array interconnected through horizontal wordlines (WL) and vertical bitlines (BL) to form a flash memory block. A collection of such blocks makes up the flash memory chip.

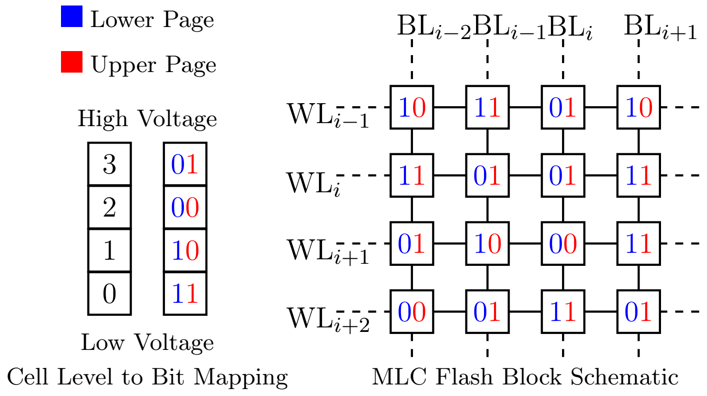


Figure 1.1: Cell level to bit mapping and block schematic in MLC flash memories. In the block schematic, the rectangles depict the MLC flash memory cells connected to horizontal wordlines (WL) and vertical bitlines (BL).

In this dissertation, we focus on MLC flash memories capable of storing 2 bits per cell. A schematic of the block structure of MLC flash memories is shown in Fig. 1.1. The two bits belonging to a MLC flash memory cell are separately mapped to logical units of programming, called pages. A page is also the smallest unit for program and read operations whereas a block is the smallest unit for the erase operation. The most significant bit (MSB) is mapped to the lower page while the least significant bit (LSB) is mapped to the upper page. We represent the four charge levels in MLC flash memory as 0, 1, 2, 3 in the increasing order of charge levels/threshold voltages respectively. The corresponding 2-bit patterns written to the lower (MSB) and upper (LSB) pages are '11', '10', '00' and '01' respectively as shown in Fig. 1.1.

The lower page bit of a cell always precedes the corresponding upper page bit in the programming order as depicted by the 2-step programming process in Fig. 1.2. For example, to program a cell to represent the bits '01', the cell V_{th} is first pushed up to an intermediate state corresponding to lower page bit 0 and then the V_{th} is further pushed up corresponding to the upper page bit 1. This 2-step programming process makes it possible to independently program the lower and upper pages. The page read operation is also evident from the 2-step program process, where a single read at V_B is sufficient to determine the lower page bit and two reads at V_A and V_C can be used to independently determine the upper page bit.

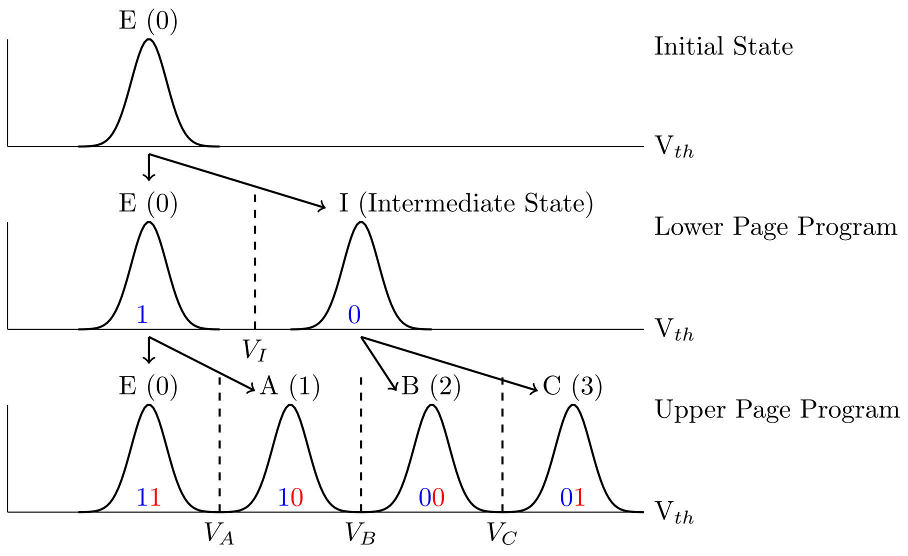


Figure 1.2: 2-step programming of the lower and upper page bits in MLC flash memories.

1.3 Error Mechanisms in Flash Memories

The major error mechanisms in flash memories are program disturb, read disturb, cell wear due to program/erase (P/E) cycling, charge loss over time affecting data retention and inter-cell interference (ICI). We briefly describe these error mechanisms as follows [3].

Program disturb

Program disturb is the mechanism in which errors occur in the stored data when charge collects on the floating-gate of cells not being currently programmed, typically cells on neighboring pages. Due to this, the threshold voltage of these victim cells increases and could possibly result in a bit error. There is no damage to the cells due to program disturb and the disturbed cells can be reset using the block erase operation. The errors due to program disturb can be corrected using ECCs while reading, or substantially mitigated by using the sequential page programming order which imposes the constraint that the lower pages of the wordlines WL_i and WL_{i+1} must be programmed before programming the upper page of the wordline WL_i for MLC flash memories.

Read disturb

Read disturb is the mechanism where charge collects on the floating-gate of cells not being currently read, typically the neighboring pages of the same block that contains the page being read. Due to this, the threshold voltage of these victim cells increases and could possibly result in a bit error. The errors due to read disturb are of concern only when the number of reads between two erase operations is very large, e.g., ∼100,000 reads for MLC flash memories [3]. There is no damage to the cells due to read disturb and the read disturb effect can be reset using the block erase operation. Read disturb errors can be corrected using ECCs or their onset can be delayed by ensuring all pages are read an equal number of times between erase operations.

Cell wear due to P/E cycling

During program/erase operations, electrons that get trapped in the dielectric layer cause a permanent upward shift in the threshold voltage distributions of the flash memory cells. This phenomenon is commonly referred to as cell wear and its effects cannot be reversed. Due to cell wear, bit errors occur and this determines the endurance/lifetime of a flash memory block. The bit errors are usually corrected using an ECC which enhances the endurance. Another technique used in practice to improve endurance of flash memory blocks is wear leveling, which ensures that all available blocks in a flash memory chip are cycled an equal number of times. Flash memory cells programmed to high charge levels are more susceptible to wear and hence data shaping (controlling the frequency of programming high charge levels) can also provide improved endurance.

Charge loss over time

The cells in a programmed flash memory block lose charge over time which results in bit errors and affects the data-retention ability of the flash memory. This effect is reversible with erase operations and does not cause any damage to the cells. In practice, the data retention ability of flash memories is usually of the order of a few months to years. Due to charge loss, there is a downward direction shift in the threshold voltages of the flash memory cells that results in bit errors during read operations. An ECC can correct these data-retention errors, or the data on the flash memory can be periodically refreshed to ensure no loss in stored data.

Inter-Cell Interference (ICI)

Inter-Cell Interference (ICI) is a mechanism that increases the charge on flash memory cells that are programmed to low charge levels when surrounded by neighboring cells programmed to high charge levels, thus resulting in bit errors. The main cause of ICI is the parasitic capacitance coupling among neighboring flash memory cells. The errors due to ICI are data dependent in nature and typically some programmed data patterns are more susceptible to errors than others. ICI errors can be corrected using ECCs or mitigated using constrained codes that forbid ICI-susceptible data patterns from being written onto the flash memory. The ICI effect also depends on the semiconductor feature size and hence is a dominant cause of errors in recent smaller feature size flash memories.

1.4 Dissertation Overview

In this dissertation, we first perform detailed experiments to characterize errors in state-of-the-art MLC flash memories. Using the error characterization results, we evaluate run-length limited (RLL) constrained coding schemes for ICI mitigation, and develop novel channel models for ECC frame error rate (FER) performance prediction and capacity estimation for MLC flash memories. The last chapter in this dissertation is a self-contained chapter on adaptive linear programming decoding of polar codes.

With an aim to quantify, model and understand the types of errors in MLC flash memories, we design and perform a series of program/erase (P/E) cycling and data rentention experiments in Chapter 2. We create a database of errors at various levels of granularity such as bit, cell, page and block, and we record the neighborhood data patterns of cells in error to provide a quantitative understanding of the underlying error mechanisms in MLC flash memories. The error characterization results reported in this dissertation are from 1X-nm and 2Y-nm MLC flash memory chips from different vendors referred to as vendor-A and vendor-B respectively. Using P/E cycling experiments, we identify the overdispersion phenomenon in the empirical statistics of the number of bit errors per frame, which leads us to develop improved channel models for ECC FER performance and capacity estimation as presented in Chapter 3 and Chapter 4 respectively. We also quantify the data dependence of ICI errors in MLC flash memories and identify highly susceptible data patterns across wordlines and bitlines that help us to empirically evaluate the performance of previously proposed (d, k)-RLL constrained coding schemes for ICI mitigation in MLC flash memories.

In Chapter 3, we propose binary discrete parametric channel models for multi-level cell (MLC) flash memories that provide accurate ECC performance estimation by modeling the empirically observed error characteristics under program/erase (P/E) cycling stress. Based on the empirical error characterization results presented in Chapter 2, we observe that a well-studied channel model such as the binary asymmetric channel (BAC) model is unable to provide accurate ECC FER performance estimation, due to the overdispersed statistics of the number of bit errors per frame discussed in Chapter 2. Hence we propose a channel model based on the beta-binomial probability distribution (2-BBM channel model) which is a good fit for the overdispersed empirical error characteristics, and we show through statistical tests and simulation results for BCH, LDPC and polar codes that the 2-BBM channel model provides accurate ECC FER performance estimation in MLC flash memories.

In Chapter 4, we study the capacity of the BBM channel model for MLC flash memories which was discussed in Chapter 3. Using the compound channel approach, we first show that the BBM channel model capacity is zero. However, based on empirical results, this appears to be a very pessimistic estimate of the flash memory channel capacity. Therefore, we propose a refined channel model called the truncated-support beta-binomial (TS-BBM) channel model and derive its capacity. Using empirical error statistics for MLC flash memories presented in Chapter 2, we numerically estimate the TS-BBM channel model capacity as a function of the program/erase (P/E) cycling stress. The capacity of the 2-TS-BBM channel model provides an upper bound on the coding rates for the flash memory chip assuming a single binary error correction code is used.

In Chapter 5, we propose adaptive linear programming (LP) decoding for polar codes. Polar codes are high density parity check codes and hence the sparse factor graph, instead of the parity check matrix, has been used to practically represent an LP polytope for LP decoding. Although LP decoding on this polytope has the ML-certificate property, it performs poorly over a BAWGN channel. We propose modifications to previously proposed adaptive cut generation based LP decoding techniques and apply the modified adaptive LP decoder to short blocklength polar codes over a BAWGN channel. The proposed decoder provides significant FER performance gain compared to the previously proposed LP decoder and its performance approaches that of maximum likelihood (ML) decoding at high SNRs. We also present an algorithm to obtain a smaller factor graph from the original sparse factor graph of a polar code. This reduced factor graph preserves the small check node degrees needed to represent the LP polytope in practice. We show that the fundamental polytope of the reduced factor graph can be obtained from the projection of the polytope represented by the original sparse factor graph and the frozen bit information. Thus, the LP decoding time complexity is decreased without affecting the FER performance by using the reduced factor graph representation.

2 Error Characterization and Inter-Cell Interference Mitigation for Multi-Level Cell Flash Memories

2.1 Introduction

Typically, error correction coding (ECC) schemes have been used to ensure reliability of flash memory operation at the cost of sacrificing a small percentage of the storage capacity. However as reported in recent studies [4, 5, 6], the errors observed in flash memories are asymmetric in nature and hence ECC schemes assuming an underlying symmetric channel model may not be the most efficient. Therefore to aid the design of better ECC schemes it is important to develop an understanding of the dominant types of cell and bit errors and be able to use such error characterization to develop improved flash memory channel models based on empirical data.

In this chapter, we first describe our flash memory error characterization experimental setup and procedures in Section 2.2. We use state-of-the-art flash memory chips of feature sizes 1X-nm from vendor-A, and 2Y-nm from vendor-B, for the error characterization. In Section 2.3.1, we present results on the characterization of bit and cell errors in MLC flash memories and identify and study the evolution of dominant cell error characteristics over the lifetime of the flash memory during P/E cycling. We identify and quantify the overdispersion phenomenon in the statistics of the number of bit errors per ECC frame during P/E cycling in Section 2.3.2. These results will be used to propose improved channel models for flash memories in Chapter 3.

Cell errors due to ICI are dependent on the data patterns written to the flash memory with some data patterns being more susceptible to ICI than others. A characterization of errors due to such susceptible data patterns will be useful in designing coding/signal processing/programming schemes to prevent/correct ICI errors in an efficient manner. In Section 2.3.3, we present results that clearly highlight and quantify the data dependent nature of ICI by studying the correlation of cell errors with their neighborhood data patterns. We also study and quantify the effect of wordline ICI along the horizontal direction and bitline ICI along the vertical direction in isolation. In Section 2.3.4, we present error characterization results obtained during data retention experiments.

Constrained codes can prevent certain ICI-susceptible data patterns from being written to the flash memory. Various techniques for the design and use of constrained codes to mitigate ICI were previously proposed in [7, 8, 9, 10]. In [7], the authors proposed using binary (d,k)-RLL codes to forbid the ICI-susceptible data patterns 1-0-1 and 3-0-3 (along the wordlines) from being written to SLC and MLC flash memories respectively. They also evaluated this constrained coding scheme along with an ECC concatenation for the SLC case using an ICI channel model. In Section 2.4, we extend their constrained coding scheme to forbid the most ICI-susceptible data patterns found in our error characterization. We also experimentally evaluate the effectiveness of the proposed coding schemes on our MLC flash memory chips and present the results.

2.2 Experimental Setup and Procedures

In this section, we describe our experimental setup and procedures for performing the program/erase (P/E) cycling and data retention experiments on multi-level cell flash memory chips to characterize the nature of errors that occur in flash memories.

2.2.1 Experimental Setup

Fig. 2.1 shows a system diagram of our experimental setup. The main component is a Zedboard which has a Xilinx Zynq-7000 All Programmable SoC, an Ethernet interface, and many GPIO pins capable of interfacing with flash memory chips. The flash memory controller operations are implemented in Verilog and programmed onto the Zynq-7000 SoC. The Zynq-7000 SoC also consists of two ARM processor cores capable of running a high level operating system (OS) such as Ubuntu Linux. The communication and data transfer protocol between the OS and the flash memory controller logic uses the Xillybus intellectual property (IP) core. The error characterization experiments are implemented in a high level programming language such as the C language.

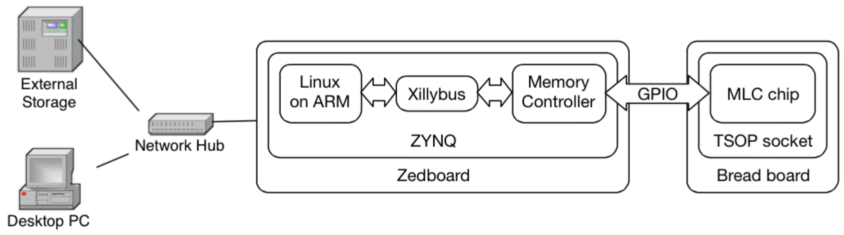


Figure 2.1: A system diagram of our experimental setup.

2.2.2 Procedure for Program/Erase (P/E) Cycling Experiments

The program/erase (P/E) cycling experiment for the MLC flash memory chip under test consists of repeated application of the following steps:

1. Erase MLC flash memory blocks under test.

2. Program MLC flash memory pages (of blocks under test) with pseudo-random (PR) data generated using a Mersenne-Twister pseudo-random number generator. The pseudo-random number generator is initialized with a randomly generated seed for every page in every P/E cycle.

3. Starting with the first cycle, perform a read operation on the MLC flash memory block(s) at intervals of every 100th cycle. Record bit errors and their locations in the block and the programmed values of every victim cell (X) and its 8-neighborhood section (cells a to h) as shown in Fig. 2.2.

e	c	f
a	$\times$	b
g	d	h

Figure 2.2: 3 x 3 neighborhood of a victim cell ($\times$)

We arbitrarily choose sets of contiguous blocks in an MLC flash memory chip for our experiments. The MLC flash memory blocks are P/E cycled up to 10,000 P/E cycles and the experiments are performed at room temperature in a continuous manner. No extra dwell time is added during the P/E cycling as we did not observe any significant difference in the error rates even with $>$ 60 seconds of extra dwell time added between the P/E cycles.

2.2.3 Procedure for Data Retention Experiments

To characterize and study the errors due to the charge loss over time/data retention error mechanism in flash memories, we use a thermal accelerated aging technique by baking the flash memory chips in a convection oven to simulate the effects of long term aging as specified in [11]. The principle used is the Arrhenius equation for reliability [11],

$$A_T = e^{\frac{-E_{aa}}{k}\left(\frac{1}{T_1}-\frac{1}{T_2}\right)} \tag{2.1}$$

where A_T is the acceleration factor for aging, E_{aa} is the apparent activation energy (eV), k is Boltzmann's constant (8.62×10^{-5} eV/K), T_1 and T_2 are the absolute temperatures (in K) of the test and the system respectively. Based on (2.1), the experiment parameters for characterization of data retention errors are as shown,

- Room Temperature, $T_2 = 21°C$
- Baking Temperature, $T_1 = 70°C$
- Activation Energy, $E_{aa} = 1.1eV$
- Baking Duration
  - 90 days aging equivalent - 4.23 hours
  - 180 days aging equivalent - 8.46 hours
  - 270 days aging equivalent - 12.69 hours
  - 360 days aging equivalent - 16.92 hours

To characterize the behavior of data retention errors over the range of P/E cycles, we choose 13 blocks of MLC flash with each block being cycled with pseudo-random data for different number of P/E cycles in the range of 0 - 6000 P/E cycles with intervals of 500 P/E cycles per block. The MLC flash memory chip is then baked for different time durations to simulate accelerated aging and the errors are recorded after the baking. No extra dwell time is added during the P/E cycling as we did not observe any significant difference in the error rates even with $>$ 60 seconds of extra dwell time added between the P/E cycles.

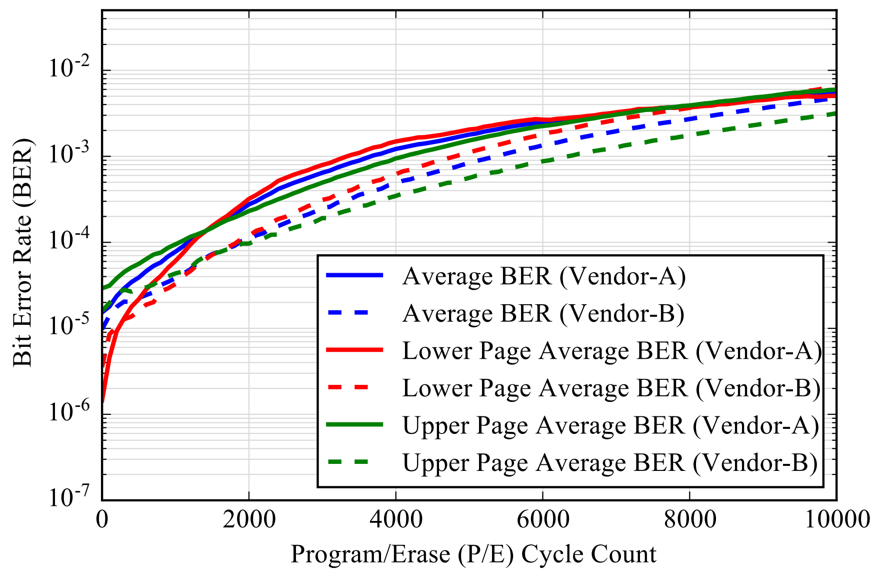


Figure 2.3: Measured average raw bit error rates over 4 blocks of vendor-A and vendor-B chips.

2.3 Error Characterization Results

2.3.1 Characterization of Bit and Cell Errors during P/E cycling

The first step in the error characterization of a flash memory chip is to study its raw bit error rate (BER) performance when all the pages in all the blocks under test are programmed with pseudo-random data. This closely resembles the most common use in practice, where random data are stored and retrieved. Fig. 2.3 shows the average raw BER across the P/E cycles when all pages in each block are programmed for both the vendor-A and vendor-B flash memory chips. The raw BER is averaged over 4 blocks tested. Fig. 2.3 also shows the average raw BER separately for the lower and upper pages of the MLC flash memory. Although the lower page is expected to have a smaller BER compared to the upper page [5], we observe that this is only the case up to a certain number of P/E cycles in the beginning and as the P/E cycle count increases, the lower page begins to show a larger number of errors than the upper page. This observation is consistent across both the vendor-A and vendor-B flash memory chips. Using empirical data from 20 blocks of the same flash memory chip, we have also observed consistent measured average raw BER estimates across all the P/E cycles.

We also record the specific cell (symbol) errors corresponding to all the bit errors

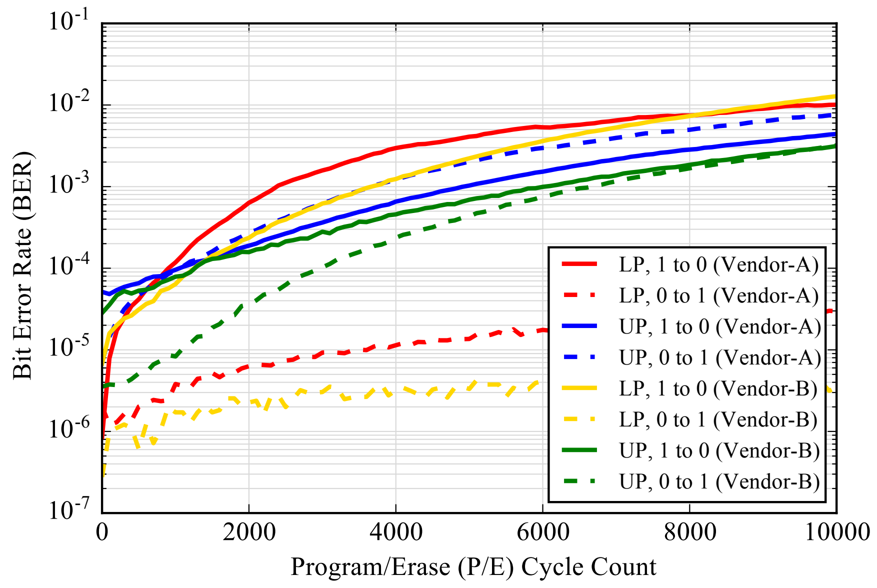


Figure 2.4: Average raw bit error rates corresponding to specific bit errors in the lower pages (LP) and upper pages (UP) over 4 blocks of vendor-A and vendor-B chips.

Table 2.1: Frequency of cell (symbol) errors measured as a percentage of total number of cell errors observed across all P/E cycles when all 4 blocks are programmed with pseudo-random data.

Vendor-A				
Write Cell	**Read Cell Values**			
Values	**11**	**10**	**00**	**01**
11	0.00	17.25	0.08	2.57
10	0.19	0.00	48.19	0.74
00	0.00	0.14	0.00	30.61
01	0.00	0.03	0.20	0.00
Vendor-B				
Write Cell	**Read Cell Values**			
Values	**11**	**10**	**00**	**01**
11	0.00	18.39	0.03	4.01
10	0.07	0.00	62.22	1.84
00	0.00	0.06	0.00	13.39
01	0.00	0.00	0.00	0.00

observed. Table 2.1 shows the frequencies of all possible cell errors as a percentage of the total number of cell errors observed across all the blocks in all the P/E cycles. The corresponding average cell error probabilities across all P/E cycles are $\sim 4.16 \times 10^{-3}$ and

$\sim 2.71 \times 10^{-3}$ for vendor-A and vendor-B chips respectively. We observe that the level 1 to 2 cell error "10 (1) $\rightarrow$ 00 (2)" is the most dominant for both vendor-A and vendor-B chips. This observation explains why the lower page average raw BER is worse than the upper page average raw BER as shown earlier in Fig. 2.3. We also note that the three adjacent level cell errors "10 (1) $\rightarrow$ 00 (2)", "11 (0) $\rightarrow$ 10 (1)" and "00 (2) $\rightarrow$ 01 (3)" are the most frequent and together make up about 96% and 94% of all the cell errors observed for the vendor-A and vendor-B chips respectively. Such knowledge about dominant cell errors can be very useful in utilizing ECC redundancy more effectively. This was demonstrated in [4], where the authors designed two BCH codes with different error correction capabilities for the lower and upper pages of an MLC flash memory and proposed a stagewise combined decoding algorithm for both pages. Their scheme gave better results than using a single BCH code independently for all pages. Fig. 2.4 shows the asymmetry of bit errors in MLC flash memories. We present the average raw BERs corresponding to the specific types of bit errors, i.e., $0 \rightarrow 1$ and $1 \rightarrow 0$ bit errors, in the lower and upper pages of both vendor-A and vendor-B MLC flash memory chips. While there is a high degree of asymmetry in the lower page bit errors throughout the P/E cycle range, the degree of asymmetry in the upper page bit errors is much lower. This agrees well with the observations in Table 2.1, where the dominant cell errors imply a large proportion of $1 \rightarrow 0$ bit errors in the lower page and comparable proportions of $0 \rightarrow 1$ and $1 \rightarrow 0$ bit errors in the upper page. This asymmetry in bit errors in both the lower and upper pages also reflects the dominance of data dependent inter-cell interference (ICI) errors, i.e., the middle cells in the cell level data patterns 303, 313 and 323 across wordlines are highly susceptible to errors as shown by ICI error characterization results in Section 2.3.3.

2.3.2 Characterization of Number of Bit Errors per ECC Frame

As we want to develop parametric channel models for MLC flash memories which provide an accurate representation of the empirically observed bit errors and enable accurate ECC FER performance estimation, we study the distribution of the number of bit errors per frame parameter. This is the key factor in determining the FER performance of an ECC with a specified error correction capability of t number of bit errors per frame.

From the error data collected during P/E cycling experiments, we obtain the sample counts of the number of bit errors per frame for $0 \rightarrow 1$ and $1 \rightarrow 0$ bit errors in both the lower and upper pages by choosing a fixed frame length of $N = 8192$ bits.

This choice of the frame length is representative of the large ECC frame lengths used in practice, while still being small enough to ensure sufficient empirical data can be collected easily. Commonly used ECC frame lengths range from 8192 to 32768 bits and multiple ECC frames are written to a single flash memory page in practice.

The sample mean and variance statistics of the number of bit errors per frame are computed using the sample counts and are shown in Table 2.2 for both vendor-A and vendor-B chips. We also plot two dimensional (2D) maps showing the number of bit errors for every frame in a single block of MLC flash memory at 8,000 P/E cycles in Fig. 2.5. The 2D maps are obtained by stacking horizontally the bit error counts in frames belonging to a page, and then stacking vertically all the pages belonging to a single block. From Table 2.2 and Fig. 2.5, we clearly observe that the variance in the number of bit errors per frame is much larger than the mean, i.e., the experiment data is overdispersed with respect to a binomial distribution, Binomial(n, p), typically used to model count data whose mean and variance are approximately equal when p is small.

Table 2.2: Sample mean and variance of the number of bit errors per frame obtained from empirical data for lower and upper pages across P/E cycles when all 4 blocks are programmed with pseudo-random data. Frame length $N = 8192$.

P/E Cycles	**Vendor-A**				**Vendor-B**			
	Lower Page		**Upper Page**		**Lower Page**		**Upper Page**	
	Mean	**Variance**	**Mean**	**Variance**	**Mean**	**Variance**	**Mean**	**Variance**
2000	2.63	3.08	1.90	2.17	0.98	1.05	0.79	0.86
4000	12.21	18.70	7.76	9.84	5.10	6.97	2.84	3.66
6000	21.90	46.71	18.43	30.06	14.85	29.64	7.18	10.23
8000	30.55	75.89	32.01	66.43	30.03	84.81	14.46	24.37
10000	41.37	111.35	48.88	125.99	52.61	216.95	26.06	51.30

2.3.3 Characterization of Errors due to Inter-Cell Interference

To characterize the ICI effect of neighbor cells on the victim cell, we classify the cell errors observed into different groups identified by the programmed values of the neighbor cells in a 8-neighborhood section as shown in Fig. 2.2. The four types of neighbor groups considered are the neighbors along

- the same wordline as the victim cell (a, b)
- the same bitline as the victim cell (c, d)

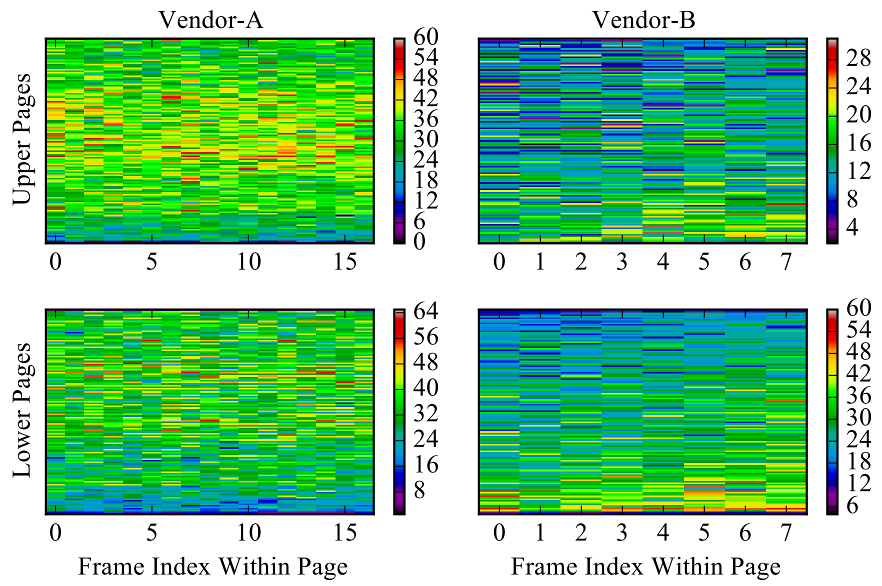


Figure 2.5: Two dimensional maps of bit error counts in frames of lower and upper pages in a single block of MLC flash memory chips from vendor-A and vendor-B at 8,000 P/E cycles.

- the diagonals on the previous wordline (e, f)
- the diagonals on the next wordline (g, h).

Fig. 2.6 shows the percentage of cell errors that were observed in each type of neighbor group for all possible programmed levels of the neighbors. We observe a strong correlation between the programmed levels of the wordline (a, b) and bitline (c, d) neighbors and the cell errors, whereas there is very little correlation of the cell errors with the programmed levels of the neighbor cells along the diagonals (e, f, g, h). For example, we see that the neighbor patterns of (3, 3) are dominant ones for the wordline and bitline neighbors whereas for the diagonal neighbor cells we do not see a dominant neighbor pattern. This clearly suggests that it is sufficient to consider only the wordline and bitline neighbor cells in the design of ICI mitigating/correcting codes. We also observe that the wordline neighbor patterns that have at least one neighbor programmed to the highest level 3 correspond to a significant percentage of the cell errors. It is also interesting to note that wordline patterns such as (2, 3) and (3, 2) correspond to approximately the same percentage of errors indicating that the relative position of the wordline neighbor cell programmed to a 3 does not affect the ICI it causes. This is consistent with the flash memory programming model where all the cells in a wordline are programmed at the

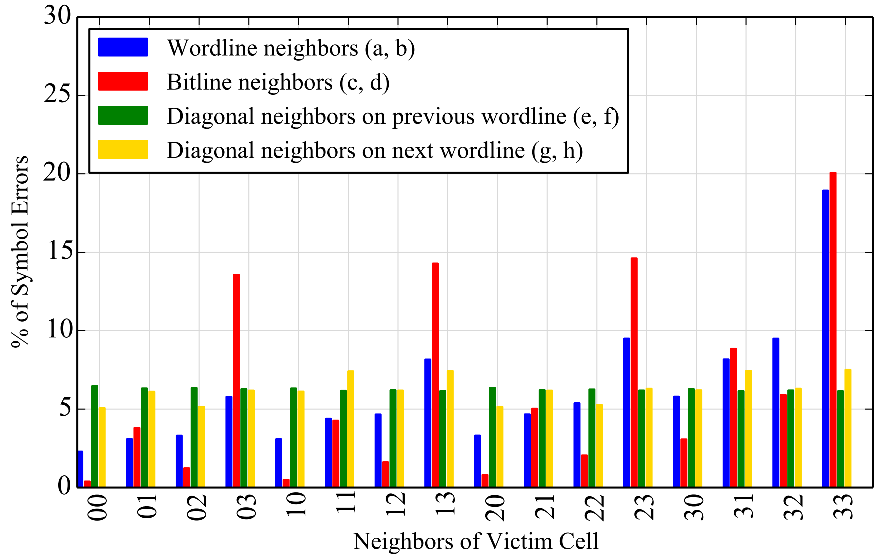


Figure 2.6: Contribution of victim cell neighbors to cell (symbol) errors measured as a percentage of total cell (symbol) errors across all P/E cycles for vendor-A chip.

same time. However, for the bitline neighbors we see that neighbor patterns where the bitline neighbor cell immediately below the victim cell is programmed to the highest level 3 correspond to about 60% of the total cell errors observed. These are the (3, 3), (2, 3), (1, 3) and (0, 3) bitline neighbor patterns. From our results, it is easy to see that the bitline neighbor cells are the most correlated with the cell errors observed, implying that the bitline ICI is stronger than the wordline ICI.

In another P/E cycling experiment, we isolate wordline ICI effects by programming only pages belonging to alternate wordlines (WL_i, WL_{i+2}, ...) using pseudo-random data. The objective is to eliminate the bitline ICI effect in the vertical direction by ensuring that the bitline neighbor cells (c, d) of any programmed cell remain unprogrammed. The raw BER across P/E cycles in this case is shown in Fig. 2.7. We observe a significant BER reduction ($\sim$100X at 4,000 P/E cycles) compared to the case when all wordlines are programmed.

To isolate the bitline ICI, we only programmed cells on alternate bitlines (BL_i, BL_{i+2}, ...) using pseudo-random data. This ensures there is no wordline ICI in the horizontal direction. Fig. 2.7 shows the observed raw BER across P/E cycles in this case. The reduction in the raw BER due to the absence of wordline ICI is only about 5X at 4,000 P/E cycles. Comparing this with the previous experiment where bitline ICI was

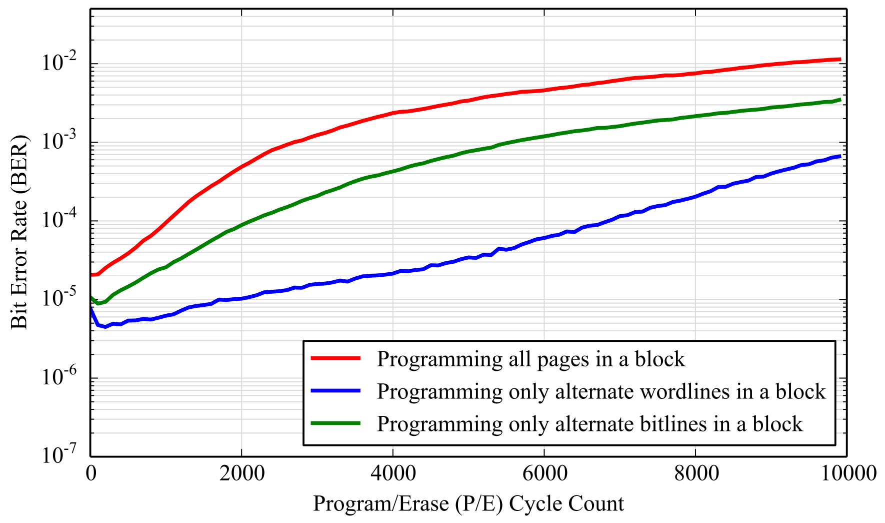


Figure 2.7: Measured average raw bit error rate using 16 blocks of vendor-A chip by programming all pages in a block, only alternate wordlines in a block and only alternate bitlines in a block with pseudo-random data in all P/E cycles.

suppressed, it is clear that the bitline ICI in the vertical direction is the dominant part of the ICI seen in flash memories.

2.3.4 Characterization of Data Retention Errors

Fig. 2.8 shows the average raw bit error rate corresponding to data retention errors with 90, 180, 270 and 360 days of aging compared with the average raw bit error rate without any aging. We observe ∼3X increase in the raw bit error rate due to 90 days of aging after 1000 P/E cycles.

Table 2.3: Frequency of cell (symbol) errors measured as a percentage of total number of cell errors observed across all P/E cycles after 90 days (∼ 3 months) and 180 days (∼ 6 months) of data retention using vendor-A chip.

90 days				
Write Cell	**Read Cell Values**			
Values	**11**	**10**	**00**	**01**
11	0.00	4.40	0.10	0.91
10	17.80	0.00	6.99	0.01
00	0.00	32.58	0.00	1.86
01	0.01	0.02	35.32	0.00

180 days				
Write Cell	**Read Cell Values**			
Values	**11**	**10**	**00**	**01**
11	0.00	2.32	0.08	0.54
10	17.50	0.00	3.45	0.01
00	0.00	35.55	0.00	0.82
01	0.00	0.01	39.72	0.00

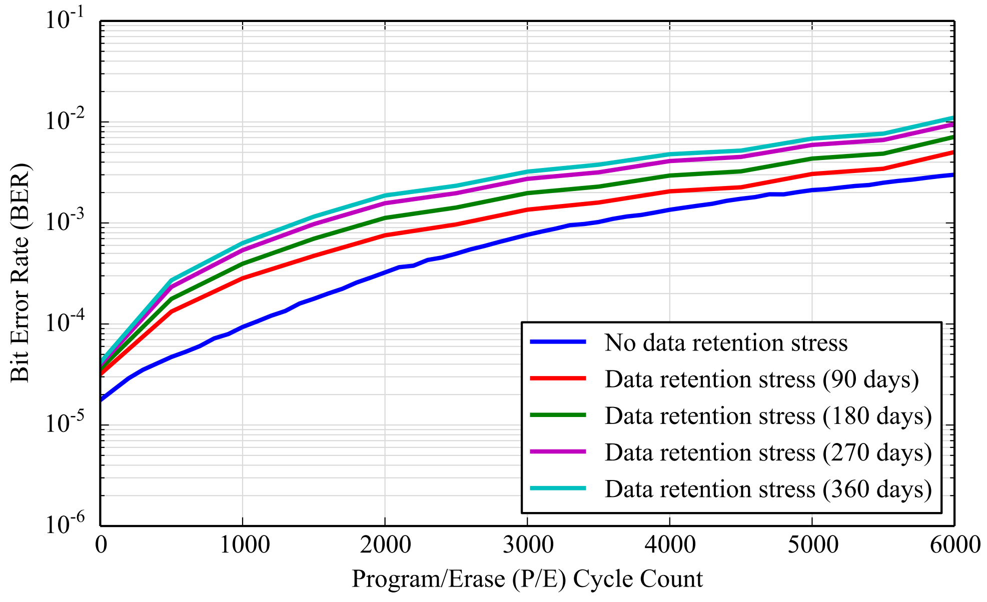


Figure 2.8: Comparison of measured average raw bit error rates for vendor-A chip across 6000 P/E cycles with added data retention stress.

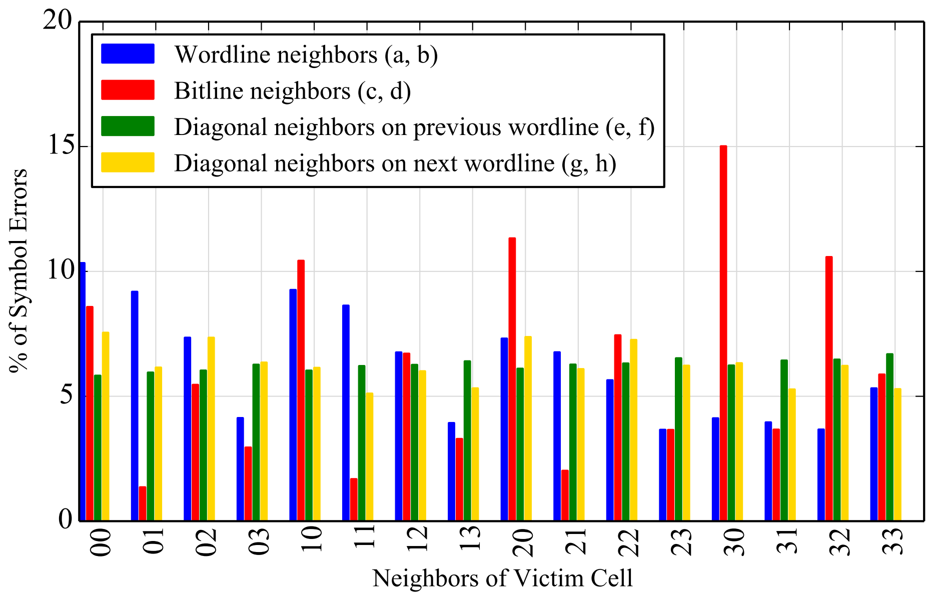


Figure 2.9: Contribution of victim cell neighbors to cell (symbol) errors measured as a percentage of total cell (symbol) errors across all P/E cycles for vendor-A chip. Data retention stress of 90 days.

Table 2.3 shows the distribution of dominant cell errors during data retention in the MLC flash memory blocks after 6000 P/E cycles. We observe that the most

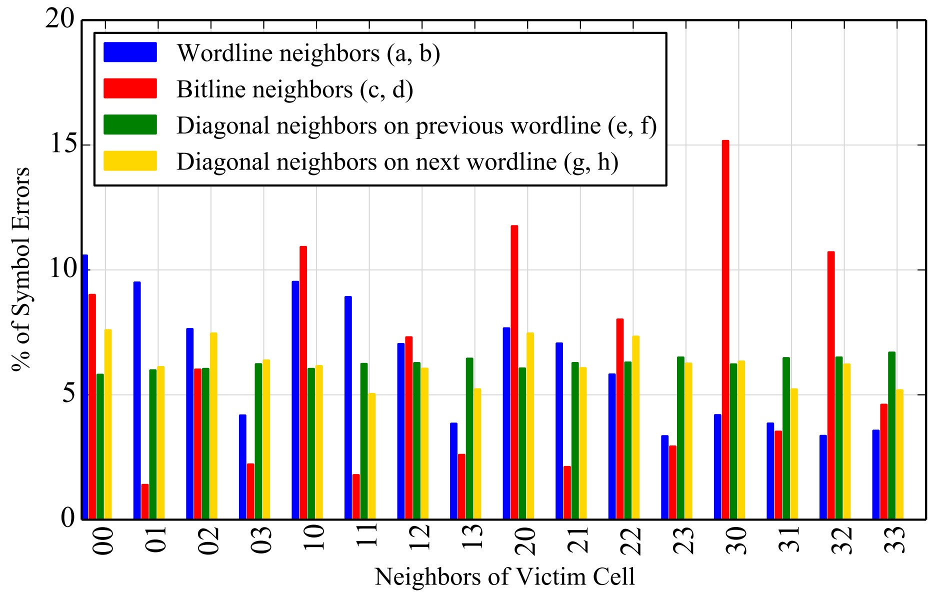


Figure 2.10: Contribution of victim cell neighbors to cell (symbol) errors measured as a percentage of total cell (symbol) errors across all P/E cycles for vendor-A chip. Data retention stress of 180 days.

dominant cell errors are adjacent cell level errors in the downward direction, such as the $3 \rightarrow 2$, $2 \rightarrow 1$ and $1 \rightarrow 0$ errors, with the $3 \rightarrow 2$ and $2 \rightarrow 1$ cell errors being almost equally dominant. We also studied the dependence of the data retention errors on the programmed values of neighbor cells of the victim cell in a 3 x 3 neighborhood as shown in Fig 2.2. Fig. 2.9 and Fig. 2.10 depict this neighborhood data dependence in terms of the correlation of percentage of cell errors observed corresponding to the 2 cell neighbor data patterns along the wordline, bitline and diagonal neighbor directions. We observe that, in general, neighbor cells programmed to the lowest cell level (erase state) 0 correspond to larger number of data retention errors in the wordline and bitline directions. Specifically in the bitline direction, the bitline neighbor cell on the next wordline 'd' being in the erased state 0 has the highest correlation with the data retention cell errors observed.

2.4 ICI Mitigation using Run-length Limited (RLL) Constrained Codes

From the ICI characterization results presented in the previous section, it is clear that the ICI effect on a victim cell is strongly correlated to the programmed levels on its wordline (horizontal) and bitline (vertical) neighbors. It is also observed that the probability of a flash memory cell being in error is the largest if its immediate neighbors along the same wordline and the same bitline are programmed to the highest level '3'. More specifically, our results show that the patterns most susceptible to ICI, considering only immediate wordline and bitline neighbors, are '3-0-3', '3-1-3' and '3-2-3'. Hence the number of cell errors due to ICI can be reduced by ensuring that these cell-level symbol patterns are never written to the flash memory.

In [7], the authors observed that a '3-0-3' pattern in an MLC flash memory consists of a '1-1-1' pattern in the upper page; that is, the binary representation of the '3-0-3' pattern is '01-11-01', where the left bit represents the lower (MSB) page and the right bit represents the upper (LSB) page. Hence to forbid '3-0-3' patterns from being written to the flash memory, it is sufficient to forbid '1-1-1' patterns from being written to the upper page of any wordline. As shown in [7], this can be done efficiently by encoding the data to be stored in the upper page using a suitably chosen binary (d, k)-runlength-limited (RLL) code.

Binary (d, k)-RLL codes are a popular class of constrained codes which have been successfully applied in magnetic recording to mitigate the adverse effects of inter-symbol interference (ISI). The codewords of a (d, k)-RLL code are a subset of binary sequences that satisfy the (d, k)-RLL constraint, which requires that the lengths of consecutive runs of zeros are at least d and at most k. In particular, as noted in [7], the '1-1-1' pattern is forbidden by any (d, k)-RLL code such that $d = 1$.

A (d, k)-RLL constraint can be easily represented using a directed graph with labeled states (nodes) and labeled edges, where the constrained sequences are obtained by reading the edge labels in a sequential manner while traversing a path in the graph. A graph representation of the $(1, 7)$-RLL constraint is shown in Fig. 2.11. The directed graph can be described by an adjacency matrix A, and the capacity of the binary (d, k)-RLL constraint is given by

$$C \ = \ \log_2 \lambda_{max}(A) \tag{2.2}$$

where $\lambda_{max}(A)$ is the largest positive eigenvalue of the matrix A [12, 13]. (The capacity represents the supremum of achievable rates of codes satisfying the constraint.)

The capacity of the $(1, 7)$-RLL constraint computed using (2.2) is ~0.6793. An efficient rate 2/3 $(1, 7)$-RLL encoder based on table lookup was used in [7] to encode the upper pages of an MLC chip, thereby guaranteeing that the symbol pattern '3-0-3' would

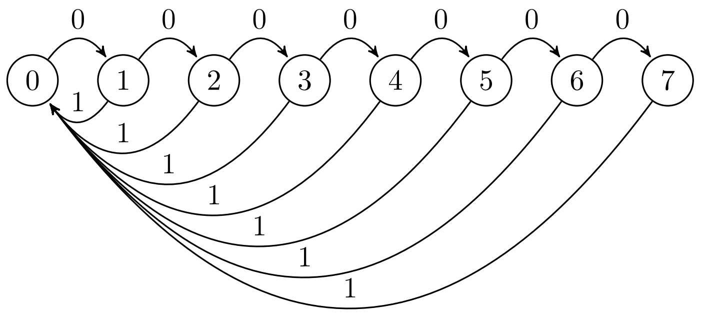


Figure 2.11: Graph representation of the $(1, 7)$-RLL constraint

not be written[1]. Since the lower page is uncoded, corresponding to a rate of 1, the overall rate of this encoding scheme is therefore given by $(1 + 2/3)/2 \approx 0.83$.

Referring to Table 2.1, we see that in our MLC flash memory the cells programmed to level '1' are the most affected by ICI and "10 (1) $\rightarrow$ 00 (2)" is the dominant error. Thus, forbidding only the '3-0-3' pattern to mitigate ICI effects is inadequate, so we extend the approach of [7] and show how to use (d, k)-RLL codes to forbid the '3-1-3' and '3-2-3' patterns in addition to the '3-0-3' pattern.

Note that the bit representations of these two additional patterns are given by '01-10-01' and '01-00-01', respectively, and that both patterns induce a '1-0-1' bit pattern in the upper page. Hence to forbid all three ICI-susceptible patterns it is sufficient to forbid the bit patterns '1-1-1' and '1-0-1' in the upper page of every wordline. This is easily accomplished by using a (d, k)-RLL code satisfying a $d = 2$ constraint, which would ensure at least two zeros between any two ones in the encoded upper page data. A graph representation of the $(2, 7)$ constraint can be obtained from the graph in Fig. 2.11 by eliminating the directed edge from state 1 to state 0. The capacity of the $(2, 7)$-RLL constraint computed using (2.2) is $\sim$0.5174. We can use an efficient 6-state rate-1/2 encoder, proposed in [14], for our $(2, 7)$-RLL constrained code. Since the lower page is again left uncoded, the overall rate of our encoding scheme is given by $(1+1/2)/2 = 0.75$. Although a specific value of k is not required to forbid ICI-susceptible data patterns, we choose $k = 7$ due to the availability of efficient encoders for the $(1, 7)$-RLL and $(2, 7)$-RLL codes [12, 13, 14]. We also note that there exist practical constrained codes with higher rates than the (d, k)-RLL codes examined here that can be used to avoid ICI-susceptible data patterns. For example, maximum transition run (MTR) codes [15],

[1]The authors of [7] also used the (1,7)-RLL code with an NRZI precoder to forbid writing the '0-1-0' pattern into an SLC flash memory, where '1' denotes the erased state. They evaluated the resulting performance improvement using a mathematical model of ICI.

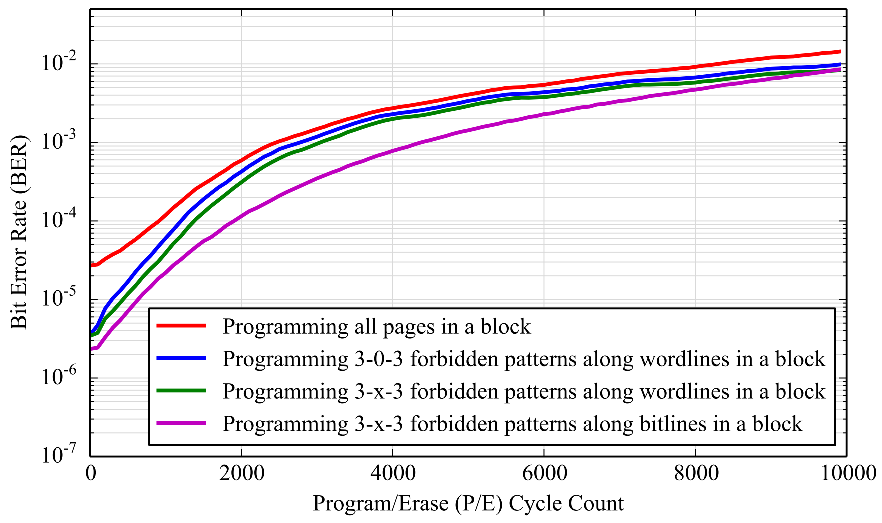


Figure 2.12: Measured average raw bit error rate comparison when all pages are programmed with pseudo-random data and when $(1, 7)$-RLL and $(2, 7)$-RLL coded data are programmed to forbid '3-x-3' patterns along wordlines or bitlines.

originally designed for magnetic recording applications, forbid the bit pattern '1-1-1' and achieve rates close to the capacity, ∼0.8791, of the corresponding constraint.

To evaluate the error rate performance gain due to the (d, k)-RLL coding, we perform P/E cycling experiments as described in Section 2.2 with the (d, k)-RLL coded data being written and read from the flash memory blocks. We separately consider the encoding of data using $(1, 7)$-RLL and $(2, 7)$-RLL codes along the wordlines (horizontal) and the bitlines (vertical) of the flash memory block to measure the effect of forbidding the ICI-susceptible data patterns in each direction. Fig. 2.12 shows the raw BER results obtained from our experiments using the (d, k)-RLL coded data.

Note that forbidding the '3-0-3', '3-1-3' and '3-2-3' data patterns using $(2, 7)$-RLL coding for the upper page also results in forbidding the '3-3-3' pattern. We therefore denote the results corresponding to this case as '3-x-3' forbidden data patterns in the plot legend in Fig. 2.12. We observe that forbidding the ICI-susceptible patterns results in significantly lower raw BER especially in the early life of the flash memory (up to $\sim 1,000$ P/E cycles). However, at later stages in the P/E cycling, forbidding the '3-x-3' patterns across the wordlines does not provide significant performance gain. This is due to the fact that the ICI along the bitlines is dominant and, consequently, coding along the bitlines to prevent '3-x-3' patterns provides the largest performance gain compared to an uncoded system.

2.5 Conclusion

We performed P/E cycling and data retention experiments on MLC NAND flash memories to characterize the error behavior at various levels. At the cell level, our results indicate an asymmetric distribution of cell errors. We also observed and characterized the overdispersion phenomenon in the statistics of the number of bit errors observed per ECC frame during P/E cycling. We studied and characterized the data dependence of ICI along with the wordline and bitline ICI effect and our results clearly show that the bitline ICI in the vertical direction is much more significant than the wordline ICI in the horizontal direction. Using (d,k)-RLL codes to mitigate ICI by forbidding ICI-susceptible patterns, we observed that it is important to consider coding techniques along the bitlines in flash memories for successful mitigation of ICI errors.

Acknowledgements

This chapter is in part a reprint of the material in the papers: Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, “Error analysis and inter-cell interference mitigation in multi-level cell flash memories,” in Proc. IEEE International Conference on Communications (ICC), London, June 2015, pp. 271–276; Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, “Channel models for multi-level cell flash memories based on empirical error analysis,” IEEE Transactions on Communications, vol. 64, no. 8, pp. 3169–3181, August 2016. The dissertation author was the primary investigator and author of these papers, and co-authors have approved the use of the material for this dissertation.

3 Channel Models for Multi-Level Cell Flash Memories

3.1 Introduction

Channel modeling for NAND flash memories is a developing research area with applications to better signal processing and coding techniques. A channel model for a flash memory can be viewed as a simplified representation of the underlying physical mechanisms which induce errors in stored data. As introduced in Chapter 1, the major error mechanisms in NAND flash memories are program disturb and cell wear that occur during program/erase cycling, charge loss that occurs during data retention and inter-cell interference (ICI) [2, 3, 16]. The main applications of a flash memory channel model are improved design, decoding and performance evaluation of error-correcting codes (ECCs) and error-mitigating codes. Other applications include information theoretic studies that provide an analysis of the capacity of flash memories [17], as well as insights for the development of new coding techniques. In this chapter, we focus on the development of parametric channel models for multi-level cell (MLC) flash memories based on empirical error characterization results presented in Chapter 2, that enable accurate ECC frame error rate (FER) performance estimation/prediction.

Efficient evaluation of ECC FER performance is important for storage system design and optimization. One approach to ECC FER performance estimation is to experimentally collect error data for use in Monte-Carlo simulations of the ECC decoder, but this can be impractical because of the large amount of error data required when estimating low frame error rates. Another approach is to analytically predict the performance of a code based upon a measured average raw bit error rate. While this is feasible for algebraic codes with bounded distance decoders, it is difficult for low density parity check (LDPC) codes and polar codes that use probabilistic decoders based upon message passing or successive cancellation. Moreover, the implicit assumption of independent, symmetric bit errors may not be justified.

Previously proposed [18, 19] parametric channel models for MLC flash memories were obtained by using well known probability distributions to model the empirical cell threshold voltage distributions. In [18], a Gaussian distribution, and in [19], a Normal-Laplace mixture model were shown to be a good fit for the experimentally observed cell threshold voltage distributions in MLC flash memories. Such models can be used to reliably predict/estimate the experimentally observed raw bit error rate (RBER) of the flash memory. However in this chapter, we show based on empirical error characterization results presented in Chapter 2, that the RBER is not necessarily a good indicator of the ECC FER performance, and that this is due to the overdispersion phenomenon in the number of bit errors per frame in MLC flash memories. Overdispersion refers to the greater variability in empirical data compared to a statistical model, such as the binomial distribution typically used to model count data. Therefore, a memoryless channel model such as the binary asymmetric channel (BAC) model provides an optimistic estimate of the ECC FER performance when compared to the actual ECC FER performance estimate obtained from empirical data.

In this chapter, first we study the suitability of well known discrete memoryless channel (DMC) models, such as the 4-ary DMC, the BSC and the BAC, to represent the bit errors observed in the MLC flash memory channel. Among the DMC models, a per-page BAC (2-BAC) model appears to align well with our empirical error characterization results. However we show through analysis as well as empirical results that the per-page BAC model is unable to fit the empirical distribution of the number of bit errors per frame and is not a good model for ECC FER performance estimation. This is due to the interdependence of mean and variance statistics of the number of bit errors per frame for a BAC where the number of $0 \to 1$ and $1 \to 0$ errors are modeled as binomial distributions. The binomial distribution is a single parameter (degree of freedom) distribution, hence its mean and variance cannot be chosen independently. Thus the binomial distribution is unable to accurately model the overdispersed empirical error data as described in Chapter 2. A natural next choice is to consider the normal approximation to the binomial distribution which provides two parameters (degrees of freedom) for modeling the observed mean and variance statistics independently. However we observe that the normal approximation based channel model does not accurately fit the shape of the empirical data distribution. Another commonly used probability distribution to model overdispersed data with respect to a binomial distribution is the beta-binomial distribution [20, 21]. Hence we propose a discrete channel model based on the beta-binomial distribution for the lower and upper pages, referred to as the 2-BBM channel model. We show that this model fits the empirical distribution of the number of bit errors per frame and provides accurate ECC FER performance estimation. We also present simple approximations of the 2-BAC model based on the normal and Poisson probability distributions. Although these approximations are able to fit the empirical distribution of the number of bit errors per frame better than the 2-BAC model, they are not as good a fit as the proposed 2-BBM channel model.

Through quantitative evaluation of the proposed channel models using the statistical Kolmogorov-Smirnov (K-S) Two Sample goodness of fit test and using Monte-Carlo simulation results of FER performance for BCH, LDPC and polar codes, we show that the 2-Beta-Binomial channel model is an accurate channel model to represent the overdispersed nature of bit errors in MLC flash memories.

3.2 Definitions and Notation

Let K represent the total number of bit errors in a frame of length N bits. Let K_m be the total number of bit errors in a frame of N bits which consists of m zeros and $N-m$ ones. The relationship between probability distributions of K and K_m is given by

$$\Pr(K=k) = \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N} \Pr(K_m = k) \tag{3.1}$$

where $\frac{\binom{N}{m}}{2^N}$ represents the probability of observing exactly m zeros in a frame of N bits. K_m can be represented as the sum of the number of $0 \to 1$ and $1 \to 0$ bit errors as

$$K_m = K_m^{(0)} + K_{N-m}^{(1)} \tag{3.2}$$

where $K_m^{(0)}$ and $K_{N-m}^{(1)}$ denote the number of $0 \to 1$ and $1 \to 0$ bit errors respectively. K can also be represented as the sum of the total number of $0 \to 1$ and $1 \to 0$ bit errors as

$$K = K^{(0)} + K^{(1)} \quad \text{where,} \tag{3.3}$$

$$\Pr(K^{(u)} = k) = \sum_{m=k}^{N} \frac{\binom{N}{l}}{2^N} \Pr(K_l^{(u)} = k) \tag{3.4}$$

Note that $u \in \{0,1\}$ where $l = m + (N-2m)u$.

We use $\mathrm{E}[X]$ and $\mathrm{Var}[X]$ to denote the expected value (mean) and the variance of a random variable X respectively. We use $X \mid Y$ to denote "X given Y".

3.3 Candidate Discrete Memoryless Channel Models

The primary error mechanism in MLC flash memories is at the cell level and hence the 4-ary DMC model with 4 inputs and 4 outputs can naturally account for all the cell level errors. This 4-ary DMC model requires 16 parameters (only 12 independent parameters) which are the cell level transition probabilities and these parameters can be easily estimated from experiment data such as that shown in Table 2.1. However the 4-ary DMC model is not useful in practice as the logical unit of progam/read operations in current MLC flash memory applications is a binary page. Hence any practically applicable channel model would have to treat the errors in the lower and upper pages of the MLC flash memory independently, even though it is clear that the errors occur at the cell level and hence the lower and upper page bit errors are not independent.

A simpler more commonly used DMC model is the 2-BSC model where two independent BSCs are used to represent the bit errors occuring in the lower and upper pages. The advantage of using the BSC model for each page independently is that it is simple and well studied, with a variety of error correction coding (ECC) techniques available for transmission over the BSC. However, based on our error characterization results in Chapter 2, the bit errors in MLC flash memories during P/E cycling are mostly asymmetric in nature. Therefore, the BSC is clearly not an accurate model to represent the bit errors in MLC flash memories. A numerical comparison of estimated capacities of the 4-ary DMC model and the 2-BSC model was presented in [22], where it was observed that the 4-ary DMC model provides a significant capacity gain compared to the 2-BSC model for MLC flash memories.

3.4 The 2-Binary Asymmetric Channel (2-BAC) Model

Based on the asymmetry of bit errors observed in MLC flash memories (Chapter 2.3), we propose a per page BAC model called the 2-BAC model where two independent BAC models are used to represent the bit errors occuring in the lower and upper pages. The 2-BAC model is a parametric model with 4 parameters which are the prob-

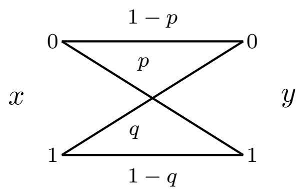


Figure 3.1: Binary asymmetric channel

abilities of $0 \to 1$ and $1 \to 0$ errors in lower and upper page BACs, $p_0^{(l)}$, $p_1^{(l)}$ and $p_0^{(u)}$, $p_1^{(u)}$. For a theoretical evaluation, we mainly compare the mean and variance statistics of the number of bit errors per frame corresponding to a BAC model with the empirically observed sample mean and variances shown in Table 2.2. We consider a BAC as shown in Fig. 3.1, where p is the probability of $0 \to 1$ error and q is the probability of $1 \to 0$ error. Next, we derive closed form expressions for the mean, E$[K]$, and the variance, Var$[K]$, of the number of bit errors per frame corresponding to a BAC model. For the BAC model, $K_m^{(0)}$ and $K_{N-m}^{(1)}$ are distributed according to the binomial probability distribution and are independent i.e.,

$$K_m^{(0)} \sim \text{Binomial}(m, p) \tag{3.5}$$

$$K_{N-m}^{(1)} \sim \text{Binomial}(N-m, q) \tag{3.6}$$

$$K_m^{(0)} \perp\!\!\!\perp K_{N-m}^{(1)} \tag{3.7}$$

The mean and the variance of $K_m^{(0)}$ are given by

$$\text{E}[K_m^{(0)}] = mp \tag{3.8}$$

$$\text{Var}[K_m^{(0)}] = mp(1-p) \tag{3.9}$$

and those of $K_{N-m}^{(1)}$ are given by

$$\text{E}[K_{N-m}^{(1)}] = (N-m)q \tag{3.10}$$

$$\text{Var}[K_{N-m}^{(1)}] = (N-m)q(1-q). \tag{3.11}$$

Proposition 3.4.1. *The mean and the variance of K for a BAC model are given by*

$$\text{E}[K] = \frac{N}{2}(p+q) \tag{3.12}$$

$$\text{Var}[K] = \frac{N}{2}\Big((p+q) - pq - \frac{1}{2}(p^2+q^2)\Big). \tag{3.13}$$

Proof. See Appendix 3.9.1. □

The parameters of the BAC model p and q are estimated as the average $0 \to 1$ and $1 \to 0$ bit error rates obtained from experimental data corresponding to a particular P/E cycle point in the flash memory lifetime. An algorithmic description of the BAC model is presented in Algorithm 1.

Algorithm 1 BAC Model Implementation

Input: Input frame $\mathbf{x}$ of length N, BAC model parameters (p, q).

Output: Data frame with errors $\mathbf{y}$.

```
1: for $x_i \in \mathbf{x}$ do
2:     Generate random sample $u \sim$ Uniform$[0, 1]$.
3:     if $x_i = 0$ then $t = p$ else $t = q$.
4:     if $u \leq t$ then $e_i = 1$ else $e_i = 0$.
5:     $y_i = x_i \oplus e_i$.
```

Using the results of Proposition 3.4.1, we compute E$[K]$ and Var$[K]$ for a BAC model as follows. For example, at 8,000 P/E cycles for the upper page BAC model for vendor-A chip, we have $p = 4.97 \times 10^{-3}$ and $q = 2.84 \times 10^{-3}$ and assuming $N = 8192$, we get E$[K] = 32.01$ and Var$[K] = 32.02$. Comparing E$[K]$ and Var$[K]$ to the sample mean and variance of K recorded using experimental data as shown in Table 2.2, we observe that the BAC model is unable to account for the large observed sample variance. For small values of p and q, from Proposition 3.4.1, we have Var$[K] \approx$ E$[K]$. Therefore, the BAC model is not a good fit for the observed empirical probability distribution of K as shown in Fig. 3.4 and Fig. 3.5 for vendor-A and vendor-B flash memory chips, respectively. As the Var$[K]$ is much less than the observed sample variance, the 2-BAC model for MLC flash memory is expected to provide a more optimistic estimate of the ECC FER performance when compared to the actual performance. We discuss this in more detail in Section 3.7. However, note that the 2-BAC model does provide an accurate estimate of the average raw BER which is given by $\frac{\mathrm{E}[K]}{N}$. This shows that the ability to accurately estimate/predict the average raw BER is not the sole criterion for a good MLC flash memory channel model.

3.5 The 2-Beta-Binomial (2-BBM) Channel Model

As mentioned in Chapter 2, the empirically observed sample mean and variance estimates show that the number of bit errors per frame data is overdispersed with respect to the binomial distribution. This is the major reason for the poor fit of the 2-BAC model discussed in the previous section. To account for the overdispersion, we propose a channel model for MLC flash memories based on the beta-binomial probability distribution called the 2-Beta-Binomial (2-BBM) channel model.

The beta-binomial probability distribution was first proposed in [20] as the probability distribution for counts resulting from a binomial distribution if the probability of success varies according to the beta distribution between sets of trials. Using empirical data, it was also shown in [20] that the beta-binomial probability distribution is a good fit for overdispersed binomial data. Lindsey et al. [21] studied the beta-binomial probability distribution based model in fitting overdispersed human sex ratio in families data and it was found to be a good fit. Stapper et al. [23] developed a yield prediction model for semiconductor memory chips by modeling the overdispersed distribution of number of faults per chip using the gamma-Poisson distribution which is closely related to the beta-binomial distribution.

For the beta-binomial channel model, we model the variables $K_m^{(0)}$ and $K_{N-m}^{(1)}$ as being distributed according to the beta-binomial distribution i.e.,

$$\begin{aligned} p &\sim \text{Beta}(a, b) \\ K_m^{(0)} \mid p &\sim \text{Binomial}(m, p) \\ K_m^{(0)} &\sim \text{Beta-Binomial}(m, a, b) \qquad (3.14) \\ q &\sim \text{Beta}(c, d) \\ K_{N-m}^{(1)} \mid q &\sim \text{Binomial}(N-m, q) \\ K_{N-m}^{(1)} &\sim \text{Beta-Binomial}(N-m, c, d) \qquad (3.15) \\ K_m^{(0)} &\perp\!\!\!\perp K_{N-m}^{(1)} \qquad (3.16) \end{aligned}$$

where (a, b) and (c, d) correspond to the parameters of a beta probability distribution defined as

$$f(\theta; \alpha, \beta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha, \beta)} \quad 0 \leq \theta \leq 1 \qquad (3.17)$$

$$B(\alpha, \beta) = \int_0^1 \theta^{\alpha-1}(1-\theta)^{\beta-1} \mathrm{d}\theta \qquad (3.18)$$

where $B(\alpha, \beta)$ represents the beta function. Thus the Beta-Binomial (BBM) channel model is derived from a BAC model where the bit error probabilities p and q are random variables which vary from frame to frame and are distributed according to the beta distribution. The BBM channel model is a 4-parameter model (compared to the 2-parameter BAC) and hence the 2-BBM channel model for MLC flash memories will be an 8-parameter model. The beta-binomial probability distributions of $K_m^{(0)}$ and $K_{N-m}^{(1)}$ are given by

$$\Pr(K_m^{(0)} = k) = \binom{m}{k} \frac{B(a+k, b+m-k)}{B(a,b)} \tag{3.19}$$

$$\Pr(K_{N-m}^{(1)} = k) = \binom{N-m}{k} \frac{B(c+k, d+N-m-k)}{B(c,d)}. \tag{3.20}$$

The mean and the variance of $K_m^{(0)}$ and $K_{N-m}^{(1)}$ are given by

$$\mathrm{E}[K_m^{(0)}] = \frac{ma}{a+b} \tag{3.21}$$

$$\mathrm{Var}[K_m^{(0)}] = \frac{mab(a+b+m)}{(a+b)^2(a+b+1)} \tag{3.22}$$

$$\mathrm{E}[K_{N-m}^{(1)}] = \frac{(N-m)c}{c+d} \tag{3.23}$$

$$\mathrm{Var}[K_{N-m}^{(1)}] = \frac{(N-m)cd(c+d+N-m)}{(c+d)^2(c+d+1)}. \tag{3.24}$$

Proposition 3.5.1. *The mean and the variance of K for a BBM channel model are given by*

$$\mathrm{E}[K] = \frac{N}{2}\left(\frac{a}{a+b} + \frac{c}{c+d}\right) \tag{3.25}$$

$$\mathrm{Var}[K] = \frac{N}{4}\left(\frac{a(a+b)(a+2b+1)+Nab}{(a+b)^2(a+b+1)}\right) + \frac{N}{4}\left(\frac{c(c+d)(c+2d+1)+Ncd}{(c+d)^2(c+d+1)}\right) - \frac{N}{4}\left(\frac{2ac}{(a+b)(c+d)}\right). \tag{3.26}$$

Proof. See Appendix 3.9.2. □

Proposition 3.5.2. *The mean and the second moment of $K^{(0)}$ and $K^{(1)}$ for a BBM* *channel model are given by*

$$\mathrm{E}[K^{(0)}] = \frac{N}{2}\left(\frac{a}{a+b}\right) \tag{3.27}$$

$$\mathrm{E}[(K^{(0)})^2] = \frac{N}{4}\left(\frac{a(a+2b+1)+Na(a+1)}{(a+b)(a+b+1)}\right) \tag{3.28}$$

$$\mathrm{E}[K^{(1)}] = \frac{N}{2}\left(\frac{c}{c+d}\right) \tag{3.29}$$

$$\mathrm{E}[(K^{(1)})^2] = \frac{N}{4}\left(\frac{c(c+2d+1)+Nc(c+1)}{(c+d)(c+d+1)}\right). \tag{3.30}$$

Proof. See Appendix 3.9.3. □

The parameters a, b, c, d of the BBM channel model are estimated from the sample moments of $K^{(0)}$ and $K^{(1)}$ using the method of moments [20]. From P/E cycling experiment data, we obtain the sample mean and sample second moment estimates of the random variables $K^{(0)}$ and $K^{(1)}$ which represent the total number of $0 \to 1$ and $1 \to 0$ bit errors per frame. Let μ_1, μ_2 represent the first and second moment estimates of $K^{(0)}$ and μ_3, μ_4 represent the first and second moment estimates of $K^{(1)}$. Solving the equations in Proposition 3.5.2 for a, b, c, d, we have the parameter estimates

$$\hat{a} = \frac{\mu_1^2(N+1) - 2\mu_1\mu_2}{N(\mu_2-\mu_1) - \mu_1^2(N-1)} \qquad \hat{b} = \hat{a}\left(\frac{N}{2\mu_1} - 1\right) \tag{3.31}$$

$$\hat{c} = \frac{\mu_3^2(N+1) - 2\mu_3\mu_4}{N(\mu_4-\mu_3) - \mu_3^2(N-1)} \qquad \hat{d} = \hat{c}\left(\frac{N}{2\mu_3} - 1\right). \tag{3.32}$$

Table 3.1 and Table 3.2 show the estimated parameters for the upper and lower pages, respectively, for both vendor-A and vendor-B flash memory chips. An algorithmic description of the BBM channel model is presented in Algorithm 2.

Algorithm 2 BBM Channel Model Implementation

```
Input: Input frame $\mathbf{x}$ of length $N$, BBM channel model parameters $(a, b, c, d)$.
Output: Data frame with errors $\mathbf{y}$.
1: Generate two independent random samples,
   $p \sim \mathrm{Beta}(a, b)$ and $q \sim \mathrm{Beta}(c, d)$.
2: $\mathbf{y} = \mathrm{BAC}(\mathbf{x}, p, q)$ [Use Algorithm 1].
```

Table 3.1: Upper page BBM channel model parameter estimates for vendor-A and vendor-B chips. $N = 8192$.

P/E Cycles	**Vendor-A**				**Vendor-B**			
	a	b	c	d	a	b	c	d
2000	12.72	46368.34	8.05	42569.08	10.82	302596.64	6.86	43747.02
4000	25.95	20940.98	15.46	23556.92	11.39	48028.59	6.00	13142.88
6000	22.67	7596.71	18.16	11890.14	15.58	20535.47	7.16	7193.92
8000	20.72	4143.52	22.28	7821.13	15.28	9068.43	7.58	4092.87
10000	21.36	2819.03	26.12	5890.35	13.36	4142.23	9.28	2938.88

Table 3.2: Lower page BBM channel model parameter estimates for vendor-A and vendor-B chips. $N = 8192$.

P/E Cycles	**Vendor-A**				**Vendor-B**			
	a	b	c	d	a	b	c	d
2000	2.85	446831.46	15.31	24066.27	0.21	94462.08	11.71	49463.01
4000	3.57	315123.27	22.49	7551.62	0.44	147347.64	13.75	11048.61
6000	1.68	95672.63	18.90	3528.74	0.29	80804.58	14.79	4070.45
8000	2.01	86407.03	20.09	2682.08	1.30	319651.96	16.30	2208.23
10000	1.81	61326.54	23.79	2338.70	1.39	337300.38	16.57	1273.77

For evaluation of the BBM channel model, we compute $\mathrm{E}[K]$ and $\mathrm{Var}[K]$ using Proposition 3.5.1. Corresponding to the example used for evaluating the BAC model, the parameter estimates of the upper page BBM channel model for vendor-A are as shown in Table 3.1 and using these parameter estimates, we obtain $\mathrm{E}[K] = 32.01$ and $\mathrm{Var}[K] = 57.88$ for $N = 8192$ at 8,000 P/E cycles. Comparing with the results from Table 2.2, we observe that the $\mathrm{Var}[K]$ obtained using the BBM channel model is still lower than the sample variance; however, it is clear that the BBM channel model is vastly better at modeling the overdispersed number of bit errors per frame empirical data than the BAC model. This will be even more evident based on the ECC FER performance estimation results presented in Section 3.7.

We also observe remarkable consistency in the parameter estimates of the BBM channel model across different blocks of the same MLC flash memory chip. Fig. 3.2 shows the empirical parameter estimates corresponding to the upper page BBM channel models for vendor-A chip using data collected from 3 different sets of 4 contiguous blocks of the MLC flash memory chip. Fig. 3.3 shows the empirical parameter estimates corresponding to the upper page BBM channel models for vendor-A chip obtained using different

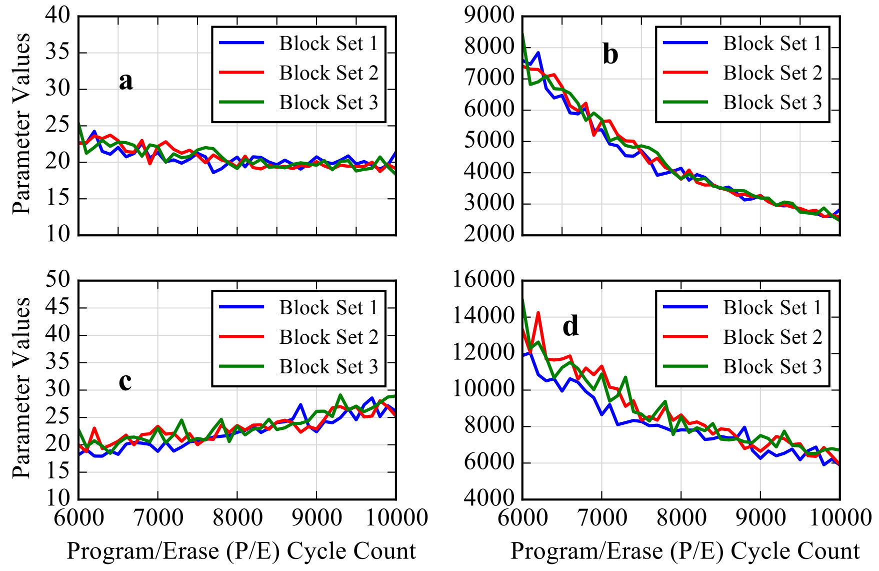


Figure 3.2: Variation of parameter estimates for the upper page BBM channel model ((a, b) for $0 \rightarrow 1$ error, (c, d) for $1 \rightarrow 0$ error) for 3 different 4-block sets for vendor-A chip. $N = 8192$.

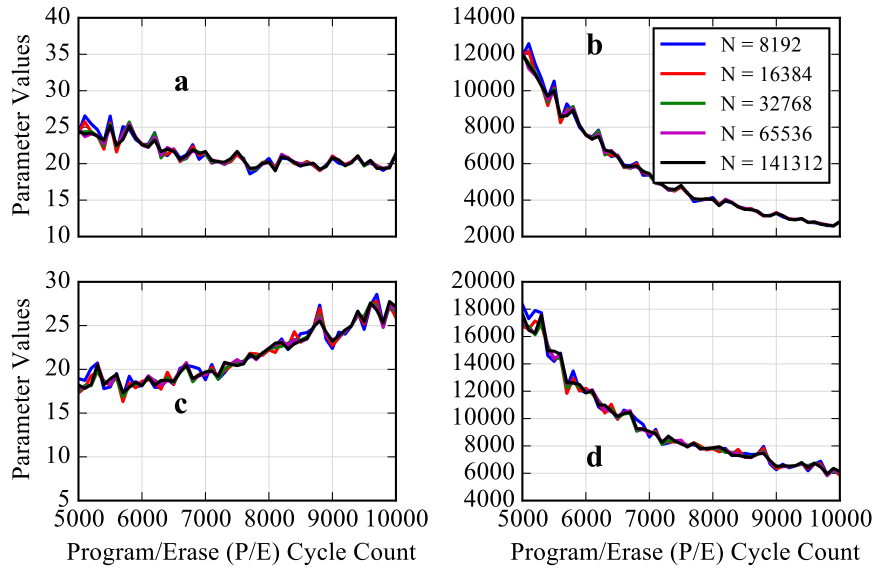


Figure 3.3: Variation of parameter estimates for the upper page BBM channel model ((a, b) for $0 \rightarrow 1$ error, (c, d) for $1 \rightarrow 0$ error) for different frame lengths for vendor-A chip.

frame sizes. Although not shown, we also observe similar consistency in the lower page parameter estimates for both the vendor chips using different sets of blocks on the same chip and different frame sizes. We also note that the estimates for lower page parameters a and b will be noisy because the $0 \to 1$ bit error rate in the lower page is extremely small. This consistency suggests that we may be able to model every flash memory chip with just 8 parameters of the 2-BBM channel model for accurate ECC FER performance estimation.

3.6 Normal and Poisson Approximation Channel Models

To model the overdispersed number of bit errors per frame empirical data, an alternative approach from a statistical viewpoint is to consider approximations to the binomial probability distribution which retain the general shape of the binomial distribution and whose mean and variance can be controlled independently. We propose two such channel models for MLC flash memories based on the normal and Poisson probability distributions called the 2-Normal Approximation to the BAC (2-NA-BAC) model and the 2-Poisson Approximation to the BAC (2-PA-BAC) model respectively. Similar to the 2-BAC and 2-BBM channel models, the 2-NA-BAC (resp., 2-PA-BAC) model consists of two independent NA-BAC (resp., PA-BAC) models for the lower and upper pages of MLC flash memories. The design goal for the NA-BAC and PA-BAC models is to ensure a match between the mean and variance statistics of the data from the model and the observed sample mean and sample variance. Based on this, we define rules for the normal and Poisson approximation as follows.

Let μ_0 and σ_0^2 denote the sample mean and sample variance of $K^{(0)}$ and μ_1 and σ_1^2 denote the sample mean and sample variance of $K^{(1)}$. Let $\mathcal{N}(\mu, \sigma^2)$ denote a normal distribution with mean μ and variance σ^2 and let $\mathcal{P}(\lambda)$ denote a Poisson distribution with rate parameter λ. Let g_0 and g_1 represent the sampled number of $0 \to 1$ and $1 \to 0$ bit errors per frame.

Definition 3.6.1. *The normal approximation rules for the NA-BAC model are given by*

$$\begin{aligned} g_0 = [\hat{g}_0] \textit{ where } \hat{g}_0 &\sim \mathcal{N}\big(\mu_0, \sigma_0^2\big) \\ g_1 = [\hat{g}_1] \textit{ where } \hat{g}_1 &\sim \mathcal{N}\big(\mu_1, \sigma_1^2\big). \end{aligned} \tag{3.33}$$

where $[\cdot]$ denotes the round to nearest integer operator.

Definition 3.6.2. *The Poisson approximation rules for the PA-BAC model are given by*

$$\begin{aligned} g_0 &= \hat{g}_0 - (\sigma_0^2 - \mu_0) \text{ where } \hat{g}_0 \sim \mathcal{P}(\sigma_0^2) \\ g_1 &= \hat{g}_1 - (\sigma_1^2 - \mu_1) \text{ where } \hat{g}_1 \sim \mathcal{P}(\sigma_1^2). \end{aligned} \tag{3.34}$$

Algorithm 3 NA-BAC and PA-BAC Model Implementation

```
Input: Input frame $\mathbf{x}$ of length $N$, sample $(\mathrm{E}[K^{(0)}], \mathrm{Var}[K^{(0)}])$, and sample
    $(\mathrm{E}[K^{(1)}], \mathrm{Var}[K^{(1)}])$.
Output: Data frame with errors $\mathbf{y}$.
1: Generate integers $g_0$, $g_1$ according to the Normal or Poisson approximation rules.
2: $\mathcal{T}_0 = \{i \mid x_i = 0\}$, $\mathcal{T}_1 = \{i \mid x_i = 1\}$.
3: Pick subsets $\mathcal{E}_0$ of size $g_0$ and $\mathcal{E}_1$ of size $g_1$ uniformly at random from $\mathcal{T}_0$ and $\mathcal{T}_1$,
   respectively.
4: Create a binary error vector $\mathbf{e}$ of length $N$ such that $e_i = 1$ if $i \in \mathcal{E}_0 \cup \mathcal{E}_1$.
5: $\mathbf{y} = \mathbf{x} \oplus \mathbf{e}$.
```

Based on these rules, an algorithmic description of the NA-BAC and PA-BAC models is presented in Algorithm 3. The normal probability distribution is a continuous distribution with infinite support whereas the variables $K^{(0)}$ and $K^{(1)}$ being modeled have finite support and are discrete (integers). Hence we require the round to nearest integer function in Definition 3.6.1. The Poisson probability distribution is a discrete distribution with an infinite support set. Using goodness of fit tests in Section 3.7, we show that the 2-NA-BAC and 2-PA-BAC models are a better fit than the 2-BAC model for the observed empirical data. However, the 2-NA-BAC and the 2-PA-BAC models are not as good a fit as the 2-BBM model to describe the bit errors in MLC flash memories.

3.7 Simulation Results and Evaluation of Channel Models

In this section, we provide a quantitative evaluation of the proposed channel models for MLC flash memories. For this we consider two viewpoints. The first one is a purely statistical viewpoint where we perform the Kolmogorov-Smirnov (K-S) Two Sample test [24] to evaluate the goodness of fit of the proposed channel models when compared with the empirical data. Next, we evaluate the proposed channel models for their application in ECC FER performance estimation. We emphasize the results of this latter evaluation when compared to the former, as accurate ECC FER performance estimation has been the main driving factor in the design of the proposed channel models.

3.7.1 Statistical Goodness of Fit Tests

The Kolmogorov-Smirnov (K-S) Two Sample test is a commonly used statistical test for determining if two sets of data samples are drawn from the same probability distribution. The K-S test is a very general test in that it makes no assumptions about the underlying probability distributions of the input data samples and is a non-parametric test [24]. This makes it suitable for our purpose as we have a varied set of underlying probability distributions of the number of bit errors per frame corresponding to the proposed channel models. The BAC and BBM model distributions do not match any well known probability distributions exactly although, they are close to the binomial distribution, and the NA-BAC and PA-BAC model distributions are approximately normal and Poisson respectively.

We perform K-S Two Sample tests comparing the number of bit errors per frame data samples from the proposed channel models to the empirical data obtained from P/E cycling experiments. The empirical data sample sizes, i.e., number of frames for each page, are 8704 for vendor-A and 4096 for vendor-B, respectively. For the BAC, BBM, NA-BAC and PA-BAC models, we simulate 10000 frames. The beta random variates to simulate the BBM channel model and the K-S Two Sample test statistic values are computed using the SciPy library [25]. The test statistic values are shown in Tables 3.3 and 3.4 for $8,000$ and $4,000$ P/E cycles, respectively. The null hypothesis is that the data samples from a proposed channel model and empirical data belong to the same underlying probability distribution. The test statistic is indicative of the difference in underlying probability distributions of the two input data samples. From Table 3.3, we see that the test statistic values are consistently low for the BBM channel model, thus indicating that it provides the best fit to the empirical data among all the proposed channel models. The p-values recorded (not shown) for all the K-S Two Sample tests in Tables 3.3 and 3.4 are smaller than 0.01 indicating that the test statistic values are estimated with a significant level of confidence. The K-S Two Sample test compares the cumulative distribution functions (CDF) obtained from input data samples to compute the test statistic. Fig. 3.4 and Fig. 3.5 provide a visual comparison of these CDFs corresponding to vendor-A and vendor-B chips.

Table 3.3: Test statistic values from K-S two sample tests comparing the lower and upper page BAC, BBM, NA-BAC, PA-BAC models with empirical data at 8,000 P/E cycles. Frame length $N = 8192$.

K-S Two Sample Tests	**Vendor-A**		**Vendor-B**	
	Lower Page	**Upper Page**	**Lower Page**	**Upper Page**
BAC vs. Experiment	0.0979	0.0744	0.1278	0.0669
BBM vs. Experiment	0.0386	0.0357	0.0190	0.0135
NA-BAC vs. Experiment	0.0430	0.0715	0.0373	0.0659
PA-BAC vs. Experiment	0.0268	0.0777	0.0337	0.1008

Table 3.4: Test statistic values from K-S two sample tests comparing the lower and upper page BAC, BBM, NA-BAC, PA-BAC models with empirical data at 4,000 P/E cycles. Frame length $N = 8192$.

K-S Two Sample Tests	**Vendor-A**		**Vendor-B**	
	Lower Page	**Upper Page**	**Lower Page**	**Upper Page**
BAC vs. Experiment	0.0498	0.0291	0.0436	0.0422
BBM vs. Experiment	0.0268	0.0153	0.0137	0.0053
NA-BAC vs. Experiment	0.0575	0.0973	0.0632	0.1191
PA-BAC vs. Experiment	0.0642	0.0703	0.0223	0.1953

3.7.2 ECC FER Performance Estimation

We evaluate the proposed channel models for their accuracy in ECC FER performance estimation using binary BCH, LDPC, and polar codes. The choice of these ECCs reflects the fact that BCH and LDPC codes are already being used in practical flash memory applications, while polar codes are a promising candidate for the future. The baseline ECC FER performance estimates are obtained from the empirical error data collected from MLC flash memory chips during P/E cycling experiments. As pseudo-random data was written to the flash memory chips during P/E cycling experiments, for ECC decoding we assume an all-zero codeword as the transmitted codeword with the error vector obtained from the empirical error data. This assumption is valid because all the ECCs considered are linear codes. To estimate the ECC FER performance using the proposed channel models, Monte-Carlo simulations are used where pseudo-random codewords of the ECC are generated and transmitted through the appropriate channel model and the received codeword is decoded. At least 400 frame errors are recorded for

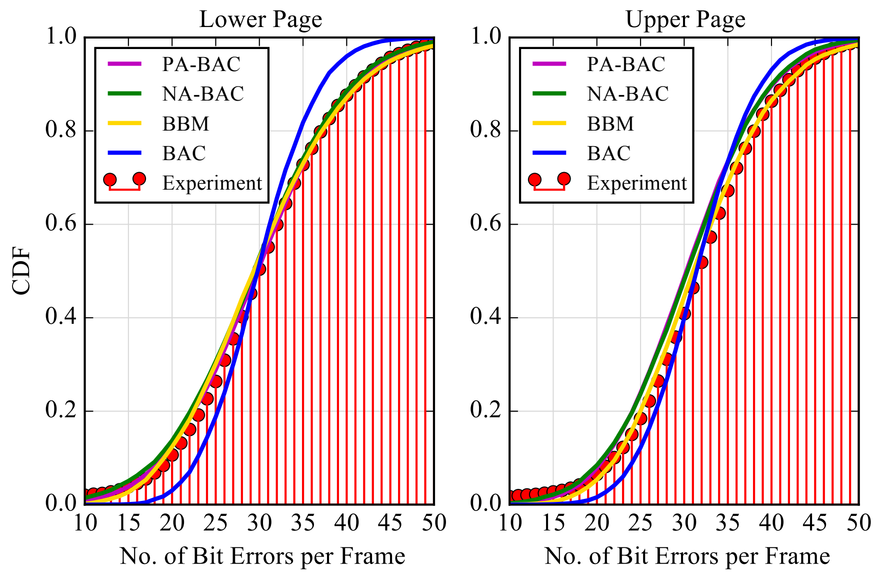


Figure 3.4: Comparison of CDFs for number of bit errors per frame observed from empirical data and from the BAC, BBM, NA-BAC, PA-BAC models at 8,000 P/E cycles for vendor-A chip.

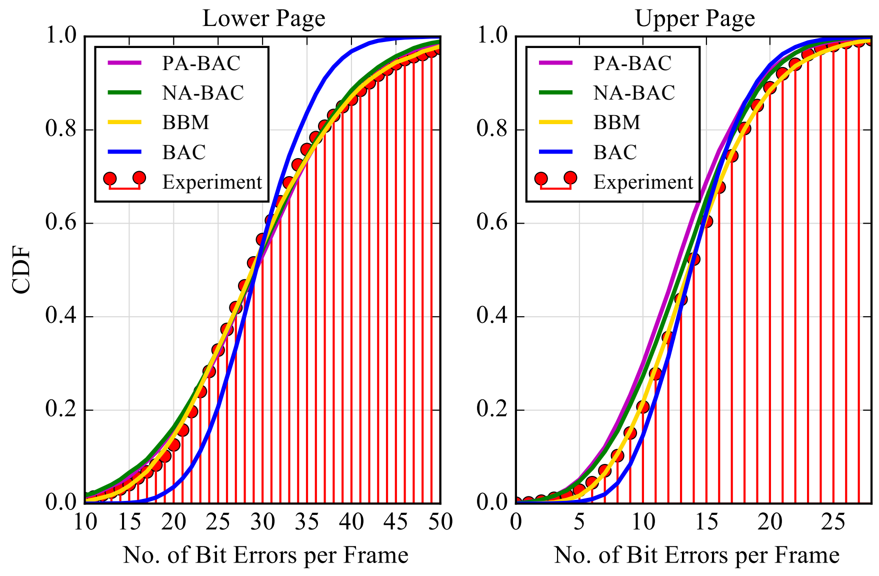


Figure 3.5: Comparison of CDFs for number of bit errors per frame observed from empirical data and from the BAC, BBM, NA-BAC, PA-BAC models at 8,000 P/E cycles for vendor-B chip.

FER estimation.

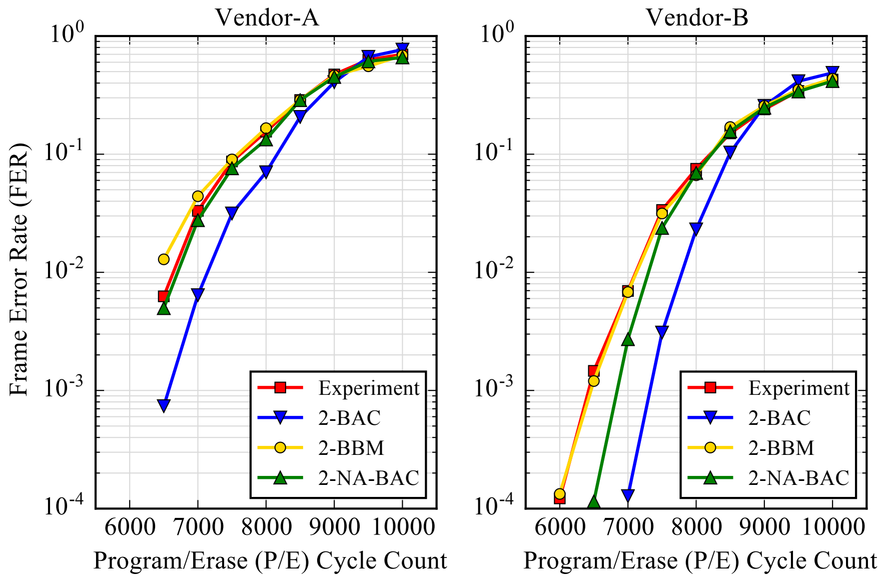


Figure 3.6: Comparison of FER performance of a (N = 8191, k = 7683, t = 39) BCH code using empirical error data and error data from simulation using the 2-BAC, 2-BBM, 2-NA-BAC channel models for vendor-A and vendor-B chips.

The FER performance of a ($N = 8191$, $k = 7683$, $t = 39$) BCH code using empirical data and the proposed channel models is shown in Fig. 3.6. Fig. 3.7 shows the FER performance of a ($N = 8192$, $k = 7683$) regular quasi-cyclic LDPC (QC-LDPC) code with $d_c = 64$ and $d_v = 4$, where d_c and d_v refer to the check node and variable node degrees, respectively, in the parity check matrix. The parity check matrix of the QC-LDPC code is constructed using size 128×128 circulant permutation matrices and the design rate is specified as 0.9375. To ensure the required variable node degree d_v, exactly d_v permutations of the circulant matrix are stacked vertically along the rows of the parity check matrix for every set of columns. Zero matrices of size 128×128 are used to fill up any remaining rows. This is done using the progressive edge growth (PEG) algorithm [26] to avoid short cycles. Note that although the specified design rate corresponds to a code dimension of 7680, we get $k = 7683$ due to three dependent parity checks in the final parity check matrix thus obtained. A sum-product belief propagation decoder with a maximum of 50 iterations and early termination is used to decode the QC-LDPC code. Fig. 3.8 also shows additional results comparing the FER performance of the QC-LDPC code obtained using empirical data and simulation data from the BAC, BBM channel

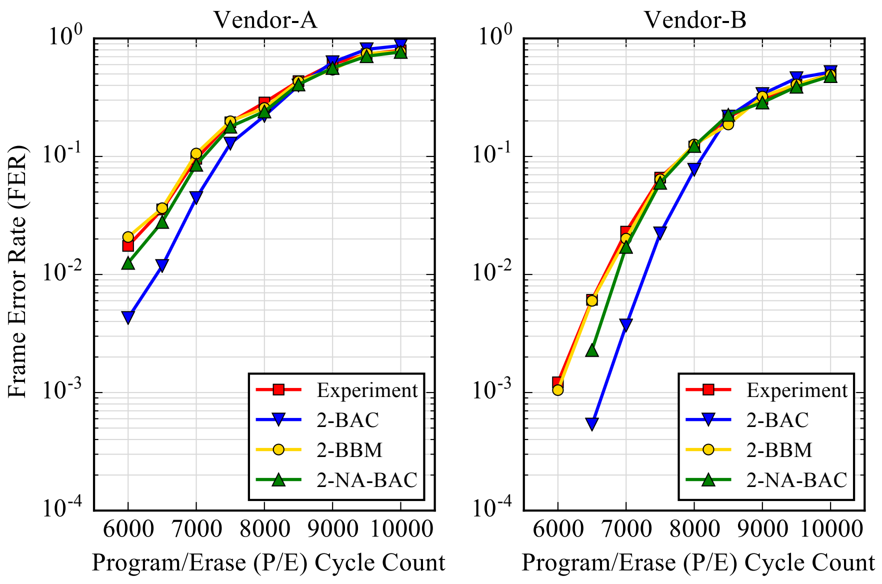


Figure 3.7: Comparison of FER performance of a (N = 8192, k = 7683) regular QC-LDPC code using empirical error data and error data from simulation using the 2-BAC, 2-BBM, 2-NA-BAC channel models for vendor-A and vendor-B chips.

models, separately for the lower and upper pages of vendor-A chip and the lower page of vendor-B chip. The lowest FER performance estimates from empirical data were obtained by P/E cycling 44 and 24 blocks of vendor-A and vendor-B chips, respectively. A total of 6 and 4 frame errors were observed to obtain the lowest FER performance estimates from empirical data for the lower and upper pages of vendor-A chip, respectively. For the lower page of vendor-B chip, 4 frame errors were observed to estimate the lowest FER performance from empirical data. Note that the results for the upper page of vendor-B chip are not shown as we did not observe any frame errors in the empirical data. We also note that a different vendor-B chip was used to obtain the additional results shown in Fig. 3.8 when compared to the rest of the paper. Fig. 3.9 shows the comparison of FER performance of a (N = 8192, k = 7684) polar code using empirical data and the proposed channel models. The polar code is optimized for a binary symmetric channel (BSC) with bit error probability $p = 0.001$ using the construction technique proposed in [27]. The successive cancellation list (SC-List) decoder proposed in [28] is used for decoding the polar code.

For all the ECCs considered and using data from both vendor chips, we observe that the 2-BAC model provides an optimistic estimate of the FER performance when

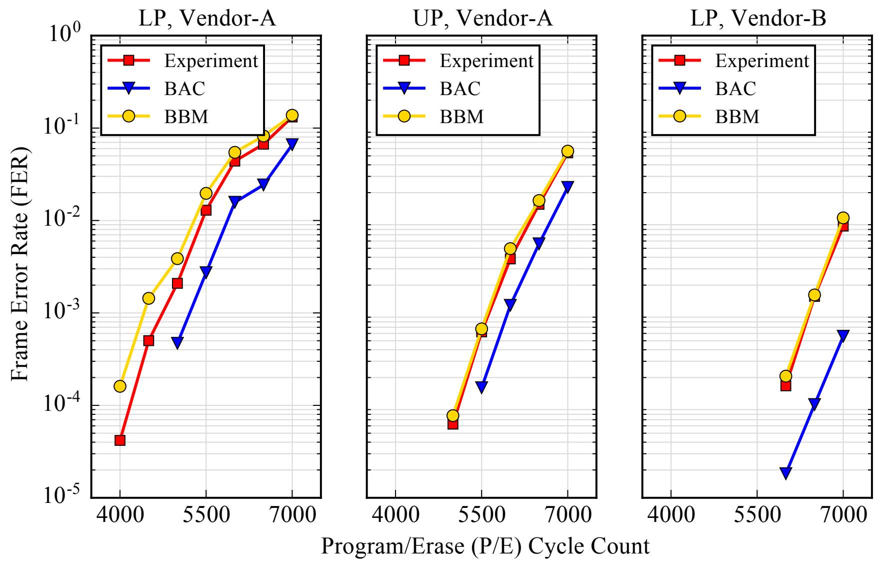


Figure 3.8: Comparison of FER performance of a ($N = 8192$, $k = 7683$) regular QC-LDPC code using empirical error data and error data from simulation using the BAC and BBM channel models for both lower and upper pages of vendor-A chip and the lower page of vendor-B chip.

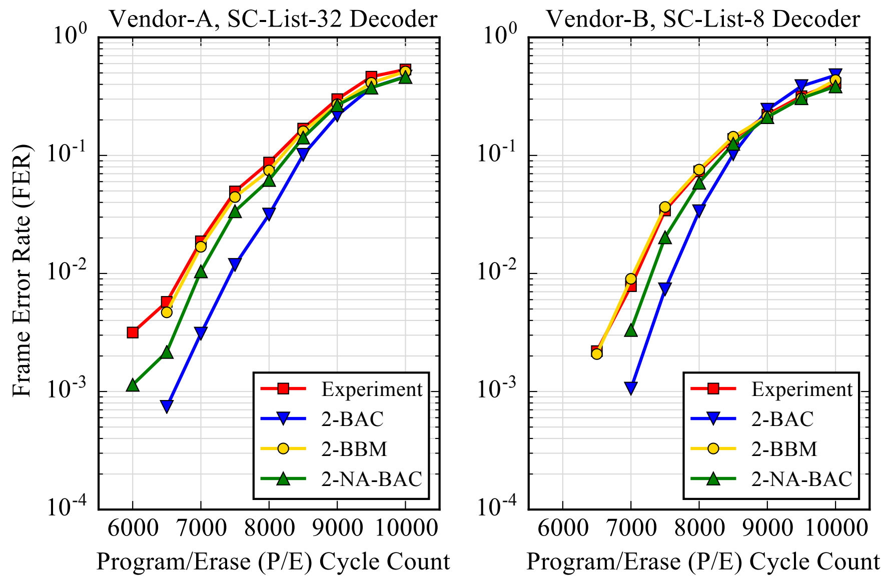


Figure 3.9: Comparison of FER performance of a ($N = 8192$, $k = 7684$) polar code optimized for BSC(0.001) using empirical error data and error data from simulation using the 2-BAC, 2-BBM, 2-NA-BAC channel models for vendor-A and vendor-B chips.

compared to the empirically observed FER performance. This is mainly due to the inability of the 2-BAC model to capture the high variance in the number of bit errors per frame observed empirically. The gap in ECC FER performance estimates using the 2-BAC model and the empirical data is increasing as the FER decreases, and it is about an order of magnitude for vendor-A chip at $6,500$ P/E cycles and greater than an order of magnitude for vendor-B chip at $7,000$ P/E cycles for the BCH code as shown in Fig. 3.6. This gap in ECC FER performance estimates at low FERs is bad for determining the correct endurance (life-time) of a flash memory chip. From the results shown in Fig. 3.8 for the QC-LDPC code, we observe that the BBM channel model estimates the FER performance accurately even at lower FERs around 10^{-4}, for the upper page of vendor-A chip and the lower page of vendor-B chip. The FER performance estimates obtained using the BBM channel model are better than those obtained using the BAC channel model for the lower page of vendor-A chip, however we observe a small mismatch in the BBM channel model FER performance estimates at lower FERs when compared to the empirical FER estimates. This mismatch is due to the inability of the BBM channel model to fit the larger proportions of frames with small number of bit errors observed in the lower tail of the empirical error histograms for the lower page of vendor-A chip. This appears to be a vendor-specific effect, as this kind of effect was not observed in the empirical error histograms corresponding to the lower page of vendor-B chip. Overall, the 2-BBM model is able to match the empirical ECC FER performance estimates accurately, while the estimates obtained using the 2-NA-BAC model lie between those of the 2-BAC and the 2-BBM models. The ECC FER performance estimates using the 2-PA-BAC model are the same as those using the 2-NA-BAC model and are omitted. From these results it is clear that the proposed 2-BBM channel model is able to accurately describe the nature of the number of bit errors per frame in MLC flash memories and hence provides accurate estimates of the ECC FER performance.

3.8 Conclusion

We studied the feasibility of using well known discrete memoryless channel models to model the MLC flash memory channel. Based on empirical error analysis and ECC FER performance estimation for BCH, LDPC, and polar codes, we observe that the 2-BAC model with parameter estimates derived from empirical error data suffices to produce an accurate estimate of the average raw bit error rate, but it provides an incorrect optimistic estimate of the ECC FER performance when compared to the empirically observed ECC FER performance. This is mainly due to the overdispersed nature of the number of bit errors per frame in empirical data which is not modeled well by the 2-BAC model. We proposed the 2-Beta-Binomial (2-BBM) channel model based on the beta-binomial probability distribution and using statistical analysis, goodness of fit tests and ECC FER performance results showed that the 2-BBM channel model accurately describes the nature of the number of bit errors per frame in MLC flash memories. We also note that the BBM channel model can be shown to be equivalent to an urn based channel model [29] and hence has memory associated with it. Although the proposed channel models are for MLC flash memories, the proposed empirical design approach is generic and can easily be extended for three-level cell (TLC) flash memories.

3.9 Appendix

3.9.1 Proof of Proposition 3.4.1

To compute Var$[K]$, we compute its mean E$[K]$ and the second moment E$[K^2]$. Based on (3.1), both these moments of K can be computed from the moments of K_m as

$$\mathrm{E}[K] = \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N}\ \mathrm{E}[K_m] \tag{3.35}$$

$$\mathrm{E}[K^2] = \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N}\ \mathrm{E}[K_m^2]. \tag{3.36}$$

From (3.2) and (3.7), we have

$$\begin{aligned}
\mathrm{E}[K_m] &= \mathrm{E}[K_m^{(0)}] + \mathrm{E}[K_{N-m}^{(1)}] \\
&= mp + (N-m)q && (3.37) \\
\mathrm{Var}[K_m] &= \mathrm{Var}[K_m^{(0)}] + \mathrm{Var}[K_{N-m}^{(1)}] \\
&= mp(1-p) + (N-m)q(1-q). && (3.38)
\end{aligned}$$

Therefore, E$[K_m^2]$ is given by

$$\begin{aligned}
\mathrm{E}[K_m^2] &= \mathrm{Var}[K_m] + (\mathrm{E}[K_m])^2 \\
&= mp + (N-m)q + m(m-1)p^2 + 2m(N-m)pq + \\
&\quad (N-m)(N-m-1)q^2. && (3.39)
\end{aligned}$$

Hence $\mathrm{E}[K]$ and $\mathrm{E}[K^2]$ are given by

$$\begin{aligned}\mathrm{E}[K] &= \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N}\ \mathrm{E}[K_m] \\ &= \frac{N}{2}(p+q) \qquad (3.40)\\ \mathrm{E}[K^2] &= \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N}\ \mathrm{E}[K_m^2] \\ &= \frac{N}{2}(p+q) + \Big(\frac{N^2-N}{2}\Big)pq + \Big(\frac{N^2-N}{4}\Big)(p^2+q^2). \qquad (3.41)\end{aligned}$$

Note that we have used the combinatorial identities

$$\sum_{m=0}^{N} \binom{N}{m}\ m = N2^{N-1} \qquad (3.42)$$

$$\sum_{m=0}^{N} \binom{N}{m}\ m^2 = (N+N^2)2^{N-2}. \qquad (3.43)$$

Therefore we can obtain $\mathrm{Var}[K]$ from (3.40) and (3.41) as

$$\begin{aligned}\mathrm{Var}[K] &= \mathrm{E}[K^2] - (\mathrm{E}[K])^2 \\ &= \frac{N}{2}\Big((p+q) - pq - \frac{1}{2}(p^2+q^2)\Big). \qquad (3.44)\end{aligned}$$

□

3.9.2 Proof of Proposition 3.5.1

We take the same approach as the proof of Proposition 3.4.1. From (3.2) and (3.16), we have

$$\mathrm{E}[K_m] = \left(\frac{ma}{a+b}\right) + \left(\frac{(N-m)c}{c+d}\right) \qquad (3.45)$$

$$\mathrm{Var}[K_m] = \left(\frac{mab(a+b+m)}{(a+b)^2(a+b+1)}\right) + \left(\frac{(N-m)cd(c+d+N-m)}{(c+d)^2(c+d+1)}\right). \qquad (3.46)$$

Therefore, $\mathrm{E}[K_m^2]$ is given by

$$\begin{aligned}\mathrm{E}[K_m^2] &= \mathrm{Var}[K_m] + (\mathrm{E}[K_m])^2 \\ &= \mathrm{Var}[K_m^{(0)}] + (\mathrm{E}[K_m^{(0)}])^2 + \mathrm{Var}[K_{N-m}^{(1)}] + (\mathrm{E}[K_{N-m}^{(1)}])^2 + \\ &\quad 2\,\mathrm{E}[K_m^{(0)}]\,\mathrm{E}[K_{N-m}^{(1)}]. \qquad (3.47)\end{aligned}$$

Substituting using (3.21) - (3.24) and simplifying, we have

$$\mathrm{E}[K_m^2] = \left(\frac{ma(m(a+1)+b)}{(a+b)(a+b+1)}\right) + \left(\frac{(N-m)c((N-m)(c+1)+d)}{(c+d)(c+d+1)}\right) + \left(\frac{2m(N-m)ac}{(a+b)(c+d)}\right). \tag{3.48}$$

Hence $\mathrm{E}[K]$ and $\mathrm{E}[K^2]$ are given by

$$\begin{aligned}\mathrm{E}[K] &= \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N}\ \mathrm{E}[K_m] \\ &= \frac{N}{2}\left(\frac{a}{a+b} + \frac{c}{c+d}\right) \qquad (3.49)\\ \mathrm{E}[K^2] &= \sum_{m=0}^{N} \frac{\binom{N}{m}}{2^N}\ \mathrm{E}[K_m^2] \\ &= \frac{N}{4}\left(\frac{(N+1)a(a+1)+2Nab}{(a+b)(a+b+1)}\right) + \frac{N}{4}\left(\frac{(N+1)c(c+1)+2Ncd}{(c+d)(c+d+1)}\right) \\ &\quad + \frac{N(N-1)}{4}\left(\frac{2ac}{(a+b)(c+d)}\right). \qquad (3.50)\end{aligned}$$

We have used the combinatorial identities (3.42) and (3.43). From (3.49) and (3.50), $\mathrm{Var}[K]$ is easily obtained as

$$\begin{aligned}\mathrm{Var}[K] &= \mathrm{E}[K^2] - (\mathrm{E}[K])^2 \\ &= \frac{N}{4}\left(\frac{a(a+b)(a+2b+1)+Nab}{(a+b)^2(a+b+1)}\right) + \frac{N}{4}\left(\frac{c(c+d)(c+2d+1)+Ncd}{(c+d)^2(c+d+1)}\right) \\ &\quad - \frac{N}{4}\left(\frac{2ac}{(a+b)(c+d)}\right). \qquad (3.51)\end{aligned}$$

□

3.9.3 Proof of Proposition 3.5.2

This proof proceeds along similar lines as the proof of Proposition 3.5.1. From (3.4) and (3.19),

$$\Pr(K^{(0)} = k) = \sum_{m=k}^{N} \frac{\binom{N}{m}}{2^N} \binom{m}{k} \frac{B(a+k, b+m-k)}{B(a,b)}$$

$$\begin{aligned}
\mathrm{E}[K^{(0)}] &= \sum_{k=0}^{N} k \Pr(K^{(0)} = k) \\
&= \frac{1}{2^N} \sum_{k=0}^{N} \sum_{m=k}^{N} k \binom{N}{m} \binom{m}{k} \frac{B(a+k, b+m-k)}{B(a,b)} \\
&= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} \sum_{k=0}^{m} k \binom{m}{k} \frac{B(a+k, b+m-k)}{B(a,b)} \\
&= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} \mathrm{E}[K_m^{(0)}] \\
&= \frac{N}{2} \left(\frac{a}{a+b} \right)
\end{aligned} \tag{3.52}$$

$$\begin{aligned}
\mathrm{E}[(K^{(0)})^2] &= \sum_{k=0}^{N} k^2 \Pr(K^{(0)} = k) \\
&= \frac{1}{2^N} \sum_{k=0}^{N} \sum_{m=k}^{N} k^2 \binom{N}{m} \binom{m}{k} \frac{B(a+k, b+m-k)}{B(a,b)} \\
&= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} \sum_{k=0}^{m} k^2 \binom{m}{k} \frac{B(a+k, b+m-k)}{B(a,b)} \\
&= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} \mathrm{E}[(K_m^{(0)})^2] \\
&= \frac{N}{4} \left(\frac{a(a+2b+1) + Na(a+1)}{(a+b)(a+b+1)} \right)
\end{aligned} \tag{3.53}$$

$$\begin{aligned}
\mathrm{Var}[K^{(0)}] &= \mathrm{E}[(K^{(0)})^2] - (\mathrm{E}[K^{(0)}])^2 \\
&= \frac{N}{4} \left(\frac{a(a+b)(a+2b+1) + Nab}{(a+b)^2(a+b+1)} \right).
\end{aligned} \tag{3.54}$$

We have used the combinatorial identities (3.42) and (3.43) and also the fact that the second moment of a beta-binomial random variable $K_m^{(0)} \sim \text{Beta-Binomial}(m, a, b)$ is given by $\frac{ma(m(a+1)+b)}{(a+b)(a+b+1)}$. The expressions for $\mathrm{E}[K^{(1)}]$ and $\mathrm{Var}[K^{(1)}]$ can be derived similarly. $\square$

Acknowledgements

This chapter is in part a reprint of the material in the paper: Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, "Channel models for multi-level cell flash memories

based on empirical error analysis," IEEE Transactions on Communications, vol. 64, no. 8, pp. 3169–3181, August 2016. The dissertation author was the primary investigator and author of this paper, and co-authors have approved the use of the material for this dissertation.

4 On the Capacity of the Beta-Binomial Channel Model for Multi-Level Cell Flash Memories

4.1 Introduction

Channel models for flash memories are important mathematical tools to estimate the capacity of the underlying flash memory channel. In this chapter, we focus on the information-theoretic aspect of flash memory channel models and study the capacity of the 2-Beta-Binomial (2-BBM) channel model for multi-level cell (MLC) flash memories proposed in Chapter 3. As described in Chapter 3, the beta-binomial (BBM) channel model for MLC flash memories accurately models the overdispersed statistics of the number of bit errors per frame observed empirically during program/erase (P/E) cycling. We also showed that the 2-BBM channel model for MLC flash memories provides more accurate ECC frame error rate (FER) performance estimation than a 2-Binary Asymmetric Channel (2-BAC) model. Therefore, the 2-BBM channel model presents itself as a natural candidate for estimating the actual MLC flash memory channel capacity. However, as we will show in this chapter, the capacity of the BBM channel model is zero. Therefore, the main problem presented in this chapter is to derive a non-zero capacity channel model for MLC flash memories which is also representative of the empirically observed overdispersed error statistics and provides accurate ECC FER performance estimation. We note that the BBM channel model can be shown to be equivalent to an urn based channel model, similar to the zero-capacity urn based contagion channel model proposed in [29]. In [29], an alternative finite memory version of the zero-capacity urn based contagion channel model was proposed and this alternative channel model was shown to have a non-zero capacity. Another approach to studying zero-capacity real world channels is provided by the example of slow fading channels in wireless communications. The Shannon capacity of a slow fading wireless communication channel is known to be zero and, hence, in practice, the $\epsilon-$outage capacity is used as an alternative performance measure for slow fading channels [30].

Previously, information-theoretic studies of NAND flash memory channel models have been proposed in [31, 17, 32]. In [31], an approximate channel model that incorporates P/E cycling, inter-cell interference (ICI), and data retention effects for MLC flash memories is proposed, and bounds on its capacity are established. In [17], the problem of estimating flash memory capacity with an underlying m-AM (amplitude modulation) channel model with input dependent Gaussian noise variance is studied. An information-theoretic study of a one-dimensional causal channel model for ICI in flash memories is presented in [32]. It is important to note that all these previous information-theoretic studies used cell-level based channel models. However, in practice, the ECC is applied at the bit (page) level and, hence, we consider a binary-input binary-output channel model such as the BBM channel model for the study of MLC flash memory capacity. The problem of how well the channel models used in [31, 17] represent the empirical error characteristics has also not been addressed. In this context, the BBM channel has been shown to provide accurate bit error rate and ECC FER performance prediction under P/E cycling. This makes the BBM channel model an ideal candidate for an information-theoretic study of the flash memory channel.

In this chapter, we observe that the BBM channel model for MLC flash memories is a compound channel model. Using well known results for compound channel capacity [33, 34], we show that the capacity of the BBM channel model is zero. Using the BBM channel model parameters derived from empirical error characterization results, we observe that the BBM channel model is a very pessimistic channel model for MLC flash memories with respect to the problem of capacity estimation. This is because the probability mass of the beta-distributed $0 \rightarrow 1$ and $1 \rightarrow 0$ bit error probabilities in the BBM channel model is concentrated in a small interval of the support $[0, 1]$. This observation allows us to define a truncated-support beta random variable and, correspondingly, a truncated-support beta-binomial (TS-BBM) channel model. We derive analytically the relationships between the statistics of the number of bit errors per frame resulting from a TS-BBM channel model and those corresponding to the BBM channel model. These results are then used to propose an approximate search algorithm to identify the truncation intervals necessary to obtain a TS-BBM channel model from a BBM channel model. Using Monte-Carlo simulations, we show LDPC and polar code FER performance results using the TS-BBM channel model. Comparing with the results obtained using the BBM channel model in Chapter 3, we observe that the proposed TS-BBM channel model is also able to provide accurate ECC FER performance estimation by modeling the overdispersed statistics of the number of bit errors per frame. Next, we derive the capacity of the proposed TS-BBM channel model using the compound channel approach. We then present non-zero capacity estimates corresponding to the TS-BBM channel models derived from empirical error characterization results under P/E cycling stress. To the best of our knowledge, this is the first study that presents capacity estimates for a binary-input binary-output channel model that has been shown to accurately fit the empirical error characteristics in MLC flash memories. However, note that the BBM and the TS-BBM channel models do not explicitly model the data dependence of cell/bit errors due to ICI [22] and assume independent random bit errors in lower and upper pages of a MLC flash memory. Therefore these constraints of the TS-BBM channel model have to be considered when using the corresponding capacity estimates in practical applications.

4.2 Capacity of the Beta-Binomial Channel Model

Let BAC$(\mathcal{X}, \mathcal{Y}, s)$ denote a binary asymmetric channel with state $s = (p, q)$ where $p = \Pr(y = 1|x = 0)$ and $q = \Pr(y = 0|x = 1)$. Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output alphabets, respectively, both of which are binary. From Fig. 3.3 in Chapter 3, we observe that the parameter values of the beta distributions are independent of the frame length used to estimate them. Next, we define the BBM channel model as a compound channel model and derive its capacity.

Definition 4.2.1. *A compound channel is defined as a set of discrete memoryless channels with state, DMC*$(\mathcal{X}, \mathcal{Y}, s)$*, with input alphabet* $\mathcal{X}$*, output alphabet* $\mathcal{Y}$ *and state* $s \in \mathcal{S}$*, where the channel state is chosen at random and fixed throughout the entire transmission block/frame i.e.,* $\Pr(y^N|x^N, s) = \Pi_{i=1}^{N} \Pr(y_i|x_i, s)$*.*

Definition 4.2.2. *The BBM channel is a compound channel consisting of a set of BACs with state, BAC*$(\mathcal{X}, \mathcal{Y}, s)$*, indexed by the state variable,* $s \in \mathcal{S} = \{(p, q) \mid p \sim Beta(a, b),\ q \sim Beta(c, d)\}$*.*

Proposition 4.2.3. *The capacity of a compound channel assuming no state information is available at the encoder or the decoder is given by [33, 34]*

$$C_{CC} = \sup_{\bar{\pi}} \inf_{s \in \mathcal{S}} I_{\bar{\pi},s}(X;Y) \tag{4.1}$$

where $I_{\bar{\pi},s}(X;Y)$ denotes the mutual information corresponding to a DMC with state s and input probability distribution, $\bar{\pi}$.

Theorem 4.2.4. *The capacity of a BBM channel is* 0.

Proof. From Definition 4.2.2 and Proposition 4.2.3,

$$C_{\text{BBM}} = \sup_{\bar{\pi}} \inf_{s \in \mathcal{S}} I_{\bar{\pi},s}(X;Y) \tag{4.2}$$

where $I_{\bar{\pi},s}(X;Y) = h(\pi_0(1-p) + \pi_1 q) - \pi_0 h(p) - \pi_1 h(q)$ is the mutual information for a BAC with state $s = (p, q)$ and input probability distribution $\bar{\pi} = (\pi_0, \pi_1)$, i.e., $\pi_0 = \Pr(x = 0), \pi_1 = \Pr(x = 1)$. Here $h(\cdot)$ denotes the binary entropy function. Using the max-min inequality [35],

$$C_{\text{BBM}} \leq \inf_{s \in \mathcal{S}} \sup_{\bar{\pi}} I_{\bar{\pi},s}(X;Y) = \inf_{s \in \mathcal{S}} C_s \tag{4.3}$$

where C_s is the capacity of a BAC with state s. As $p, q \in [0, 1]$, $C_s = 0$ when the state $s = (\frac{1}{2}, \frac{1}{2})$. Therefore,

$$C_{\text{BBM}} \leq 0 \implies C_{\text{BBM}} = 0. \tag{4.4}$$

□

4.3 Truncated-Support Beta-Binomial Channel Model

4.3.1 Motivation

In the previous section, we showed that the capacity of a BBM channel model for MLC flash memories is zero. However empirical observations suggest that the BBM channel model is a very pessimistic model in this respect. To elaborate, from Tables 3.1 (c.f., Chapter 3), we observe that the upper page BBM channel model parameters satisfy $a > 2$, $b > 2$, $c > 2$ and $d > 2$. It is known [36] that the beta probability density function (pdf) with parameters α and β is unimodal and bell-shaped with two inflection points in $[0, 1]$ if $\alpha > 2$ and $\beta > 2$. The inflection points about the mode are given by

$$\frac{\alpha - 1}{\alpha + \beta - 2} \pm \frac{\sqrt{\frac{(\alpha-1)(\beta-1)}{\alpha+\beta-3}}}{\alpha + \beta - 2}. \tag{4.5}$$

The difference between the two inflection points, $\zeta_{\alpha,\beta}$, can be used as a measure of spread of the beta density function, where

$$\zeta_{\alpha,\beta} = 2\frac{\sqrt{\frac{(\alpha-1)(\beta-1)}{\alpha+\beta-3}}}{\alpha+\beta-2}. \tag{4.6}$$

Table 4.1: Parameters corresponding to the upper page BBM channel models for vendor-A and vendor-B chips along with the corresponding $\zeta_{a,b}$ and $\zeta_{c,d}$ values. $N = 8192$.

P/E Cycles	**Upper Page**					
	Vendor-A					
	a	b	$\zeta_{a,b}$	c	d	$\zeta_{c,d}$
6000	22.67	7596.71	1.22×10^{-3}	18.16	11890.14	0.69×10^{-3}
8000	20.72	4143.52	2.13×10^{-3}	22.28	7821.13	1.18×10^{-3}
10000	21.36	2819.03	3.17×10^{-3}	26.12	5890.35	1.69×10^{-3}
	Vendor-B					
6000	15.58	20535.47	0.37×10^{-3}	7.16	7193.92	0.69×10^{-3}
8000	15.28	9068.43	0.83×10^{-3}	7.58	4092.87	1.25×10^{-3}
10000	13.36	4142.23	1.69×10^{-3}	9.28	2938.88	1.95×10^{-3}

Table 4.1 shows the values of $\zeta_{a,b}$ and $\zeta_{c,d}$ for the upper page BBM channel model for vendor-A and vendor-B chips along with the parameter estimates. Both $\zeta_{a,b}$ and $\zeta_{c,d}$ are small indicating that the probability mass corresponding to the Beta(a, b) and Beta(c, d) pdf's of the BBM channel model is concentrated in a small interval. For example, from a visual inspection of a typical beta pdf corresponding to the BBM channel model, as shown in Fig. 4.2, it is clear that the pdf value is negligible outside the bit error probability range of [0.002, 0.009]. For the parameter estimates corresponding to the beta distributions of lower page $1 \to 0$ errors, Table 3.2 indicates that we have to consider additional cases. In the case where the beta distribution parameters satisfy $1 < a < 2$ and $b > 2$, the distribution is unimodal with a single inflection point to the right of the mode. When $0 < a < 1$ and $b > 2$, the mode of the beta density occurs at 0, and it is strictly decreasing to the right. These observations motivate us to propose and define a truncated-support beta-binomial (TS-BBM) channel model for MLC flash memories. The primary goal of the TS-BBM channel model is to provide a non-zero capacity estimate which can be reasonably interpreted in the context of MLC flash memories, while modeling the empirically observed distributions of the number of bit errors per frame as accurately as possible. This latter constraint is essential for accurate ECC FER performance estimation as shown in Chapter 3.

4.3.2 Definition and Statistics of the TS-BBM Channel Model

Before defining a TS-BBM channel model, we define truncated-support beta (TS-Beta) and truncated-support beta-binomial (TS-BBM) random variables as follows.

Definition 4.3.1. *A TS-Beta random variable is defined by the probability density function*

$$g(\theta;\alpha,\beta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}. \tag{4.7}$$

Here the support of the beta density function is truncated, i.e., $\theta \in [\theta_l, \theta_u]$, $\theta_l < \theta_u$ and $B_{\theta_l}(\alpha,\beta)$ and $B_{\theta_u}(\alpha,\beta)$ are incomplete beta functions defined as

$$B_\theta(\alpha,\beta) = \int_0^\theta \lambda^{\alpha-1}(1-\lambda)^{\beta-1}\, d\lambda. \tag{4.8}$$

Note that the TS-Beta probability density function is obtained by normalizing the beta probability density function as follows

$$g(\theta;\alpha,\beta) = \begin{cases} 0 & , \ \theta \notin [\theta_l, \theta_u] \\ \dfrac{f(\theta;\alpha,\beta)}{\eta_\theta} & , \ \theta \in [\theta_l, \theta_u] \end{cases} \tag{4.9}$$

where

$$\eta_\theta = \Pr(\theta_l \le \theta \le \theta_u) = \frac{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}{B(\alpha,\beta)}. \tag{4.10}$$

Definition 4.3.2. *A TS-BBM random variable Z is defined by the probability distribution*

$$\Pr(Z = z) = \binom{n}{z}\left(\frac{B_{\theta_u}(\alpha+z,\beta+n-z) - B_{\theta_l}(\alpha+z,\beta+n-z)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}\right). \tag{4.11}$$

Here $z \in \{0, 1, \ldots, n\}$ where n is the number of trials and θ_u, θ_l are the upper and lower limits of the success probability distributed as a TS-Beta random variable.

Before we compute the mean and variance of a TS-BBM random variable, we define the functions δ_θ, ϕ_θ in terms of the beta density function, which will be useful in simplifying the notation and interpretation of the results:

$$\delta_\theta = \rho(f(\theta_u;\alpha+1,\beta+1) - f(\theta_l;\alpha+1,\beta+1)) \tag{4.12}$$

$$\phi_\theta = \rho(\theta_u f(\theta_u;\alpha+1,\beta+1) - \theta_l f(\theta_l;\alpha+1,\beta+1)) \tag{4.13}$$

$$\rho = \frac{\alpha\beta}{(\alpha+\beta)(\alpha+\beta+1)}. \tag{4.14}$$

Proposition 4.3.3. *The mean and variance of a TS-BBM random variable Z are given by*

$$\begin{aligned}
\mathrm{E}[Z] &= n\left(\frac{B_{\theta_u}(\alpha+1,\beta)-B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta)-B_{\theta_l}(\alpha,\beta)}\right) && (4.15)\\
&= \mathrm{E}[\tilde{Z}] - \left(\frac{n}{\alpha+\beta}\right)\frac{\delta_\theta}{\eta_\theta} && (4.16)\\
\mathrm{Var}[Z] &= n\left(\frac{B_{\theta_u}(\alpha+1,\beta)-B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta)-B_{\theta_l}(\alpha,\beta)}\right)\\
&\quad \left(1-n\left(\frac{B_{\theta_u}(\alpha+1,\beta)-B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta)-B_{\theta_l}(\alpha,\beta)}\right)\right)\\
&\quad +n(n-1)\left(\frac{B_{\theta_u}(\alpha+2,\beta)-B_{\theta_l}(\alpha+2,\beta)}{B_{\theta_u}(\alpha,\beta)-B_{\theta_l}(\alpha,\beta)}\right) && (4.17)\\
&= \mathrm{Var}[\tilde{Z}] - \left(\frac{n\beta(\alpha+\beta+n)}{(\alpha+\beta)^2(\alpha+\beta+1)}\right)\frac{\delta_\theta}{\eta_\theta} - \frac{n^2}{(\alpha+\beta)^2}\left(\frac{\delta_\theta}{\eta_\theta}\right)^2\\
&\quad -\left(\frac{n(n-1)}{\alpha+\beta+1}\right)\frac{\phi_\theta}{\eta_\theta} && (4.18)
\end{aligned}$$

where $\tilde{Z}$ is a beta-binomial random variable with parameters (n, α, β).

Proof. See Appendix 4.6.1. □

Next, we define the 2-TS-BBM channel model for MLC flash memories as follows. The $0 \to 1$ and $1 \to 0$ bit error counts for a given input frame containing m zeros are denoted by $K_m^{(0)}$ and $K_{N-m}^{(1)}$, respectively. Similarly, $K^{(0)}$ and $K^{(1)}$ denote the total number of $0 \to 1$ and $1 \to 0$ bit errors, respectively. The variables $K_m^{(0)}$ and $K_{N-m}^{(1)}$ are modeled as being distributed according to the TS-BBM distribution i.e.,

$$\begin{aligned}
p &\sim \text{TS-Beta}(p_l, p_u; a, b)\\
K_m^{(0)} \mid p &\sim \text{Binomial}(m, p)\\
K_m^{(0)} &\sim \text{TS-BBM}(p_l, p_u; m, a, b); && (4.19)\\
q &\sim \text{TS-Beta}(q_l, q_u; c, d)\\
K_{N-m}^{(1)} \mid q &\sim \text{Binomial}(N-m, q)\\
K_{N-m}^{(1)} &\sim \text{TS-BBM}(q_l, q_u; N-m, c, d); && (4.20)\\
K_m^{(0)} &\perp\!\!\!\perp K_{N-m}^{(1)}. && (4.21)
\end{aligned}$$

Before we derive expressions for the mean and variance of the number of bit errors per frame resulting from a TS-BBM channel model, we introduce some simplifying notation as follows. Let

$$U_{p_j}^{(i)} = B_{p_j}(a+i,b) \tag{4.22}$$

$$V_{q_j}^{(i)} = B_{q_j}(c+i,d) \tag{4.23}$$

where $j \in \{l,u\}$ and $i \in \{0,1,2\}$.

Proposition 4.3.4. *The mean and variance of $K^{(0)}$ and $K^{(1)}$ for a TS-BBM channel model are given by*

$$\mathrm{E}[K^{(0)}] = \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right) \tag{4.24}$$

$$\begin{aligned}\mathrm{Var}[K^{(0)}] = {} & \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right)\left(1 - \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right)\right) + \\ & \frac{N(N-1)}{4}\left(\frac{U_{p_u}^{(2)} - U_{p_l}^{(2)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right);\end{aligned} \tag{4.25}$$

$$\mathrm{E}[K^{(1)}] = \frac{N}{2}\left(\frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right) \tag{4.26}$$

$$\begin{aligned}\mathrm{Var}[K^{(1)}] = {} & \frac{N}{2}\left(\frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right)\left(1 - \frac{N}{2}\left(\frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right)\right) + \\ & \frac{N(N-1)}{4}\left(\frac{V_{q_u}^{(2)} - V_{q_l}^{(2)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right).\end{aligned} \tag{4.27}$$

Proof. See Appendix 4.6.2. □

Proposition 4.3.5. *The mean and the variance of K for a TS-BBM channel model are given by*

$$\mathrm{E}[K] = \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} + \frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right) \tag{4.28}$$

$$\begin{aligned}\mathrm{Var}[K] = {} & \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right)\left(1 - \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right)\right) \\ & + \frac{N}{2}\left(\frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right)\left(1 - \frac{N}{2}\left(\frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right)\right) \\ & + \frac{N(N-1)}{4}\left(\frac{U_{p_u}^{(2)} - U_{p_l}^{(2)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} + \frac{V_{q_u}^{(2)} - V_{q_l}^{(2)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right) \\ & - \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}}\right)\left(\frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right).\end{aligned} \tag{4.29}$$

Proof. See Appendix 4.6.3. $\square$

Proposition 4.3.6. *The mean and variance of $K^{(0)}$, $K^{(1)}$, K corresponding to a TS-BBM channel model can be expressed in terms of the mean and variance of $\tilde{K}^{(0)}$, $\tilde{K}^{(1)}$, $\tilde{K}$ corresponding to a BBM channel model with the same parameters, as*

$$\mathrm{E}[K^{(0)}] = \mathrm{E}[\tilde{K}^{(0)}] - \Delta^{(0)}_{mean} \tag{4.30}$$

$$\mathrm{Var}[K^{(0)}] = \mathrm{Var}[\tilde{K}^{(0)}] - \Delta^{(0)}_{var}; \tag{4.31}$$

$$\mathrm{E}[K^{(1)}] = \mathrm{E}[\tilde{K}^{(1)}] - \Delta^{(1)}_{mean} \tag{4.32}$$

$$\mathrm{Var}[K^{(1)}] = \mathrm{Var}[\tilde{K}^{(1)}] - \Delta^{(1)}_{var}; \tag{4.33}$$

$$\mathrm{E}[K] = \mathrm{E}[\tilde{K}] - \Delta_{mean} \tag{4.34}$$

$$\mathrm{Var}[K] = \mathrm{Var}[\tilde{K}] - \Delta_{var} \tag{4.35}$$

where

$$\Delta^{(0)}_{mean} = \frac{N}{2}\frac{1}{(a+b)}\frac{\delta_p}{\eta_p} \tag{4.36}$$

$$\Delta^{(0)}_{var} = \frac{N}{4}\left(w_1\frac{\delta_p}{\eta_p} + w_2\frac{\phi_p}{\eta_p} + w_3\left(\frac{\delta_p}{\eta_p}\right)^2\right); \tag{4.37}$$

$$\Delta^{(1)}_{mean} = \frac{N}{2}\frac{1}{(c+d)}\frac{\delta_q}{\eta_q} \tag{4.38}$$

$$\Delta^{(1)}_{var} = \frac{N}{4}\left(w_4\frac{\delta_q}{\eta_q} + w_5\frac{\phi_q}{\eta_q} + w_6\left(\frac{\delta_q}{\eta_q}\right)^2\right); \tag{4.39}$$

$$\Delta_{mean} = \frac{N}{2}\left(\frac{1}{(a+b)}\frac{\delta_p}{\eta_p} + \frac{1}{(c+d)}\frac{\delta_q}{\eta_q}\right) \tag{4.40}$$

$$\Delta_{var} = \frac{N}{4}\left(w_7\frac{\delta_p}{\eta_p} + w_8\frac{\delta_q}{\eta_q} + w_2\frac{\phi_p}{\eta_p} + w_5\frac{\phi_q}{\eta_q} + w_3\left(\frac{\delta_p}{\eta_p}\right)^2 + w_6\left(\frac{\delta_q}{\eta_q}\right)^2 + w_9\frac{\delta_p}{\eta_p}\frac{\delta_q}{\eta_q}\right), \tag{4.41}$$

$$w_1 = \frac{(a+b)(a+2b+1) + N(b-a(a+b+1))}{(a+b)^2(a+b+1)} \tag{4.42}$$

$$w_2 = \frac{N-1}{a+b+1} \qquad w_3 = \frac{N}{(a+b)^2} \tag{4.43}$$

$$w_4 = \frac{(c+d)(c+2d+1) + N(d-c(c+d+1))}{(c+d)^2(c+d+1)} \tag{4.44}$$

$$w_5 = \frac{N-1}{c+d+1} \qquad w_6 = \frac{N}{(c+d)^2} \tag{4.45}$$

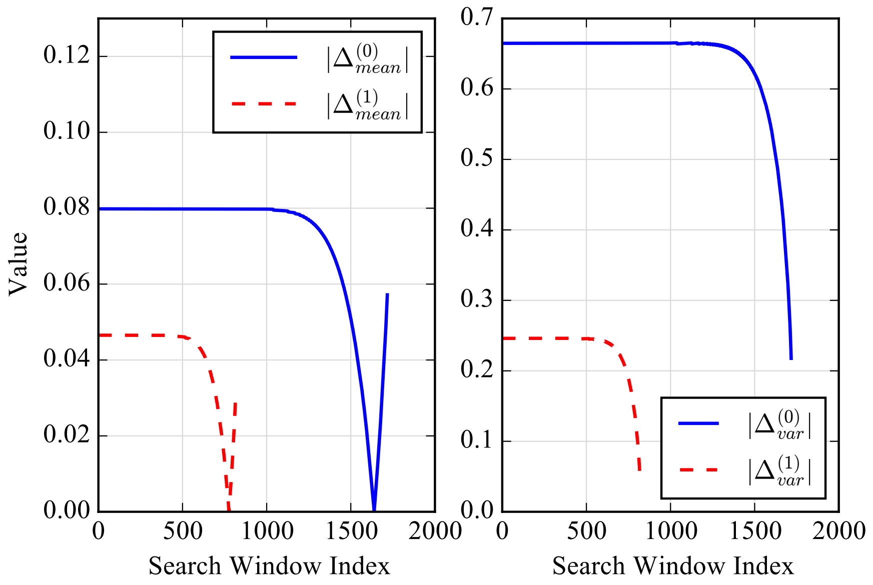


Figure 4.1: Plot showing $|\Delta^{(x)}_{mean}|$ and $|\Delta^{(x)}_{var}|$ obtained using Algorithm 4, corresponding to the beta distributions at 6,000 P/E cycles for vendor-A chip. $N = 8192$, $\mu = 10^{-6}$, $\epsilon = 0.01$.

$$w_7 = \frac{N(b - a(a+b+1))}{(a+b)^2(a+b+1)} + \frac{2d}{(a+b)(c+d)} \tag{4.46}$$

$$w_8 = \frac{N(d - c(c+d+1))}{(c+d)^2(c+d+1)} + \frac{2b}{(a+b)(c+d)} \tag{4.47}$$

$$w_9 = \frac{2}{(a+b)(c+d)}. \tag{4.48}$$

Proof. See Appendix 4.6.4. □

4.3.3 Choosing the Truncation Intervals

As shown in Proposition 4.3.6, the mean and variance statistics of the number of bit errors per frame for the TS-BBM channel model differ from those of the BBM channel model and this difference depends on the functions η_p, η_q, δ_p, δ_q, ϕ_p, ϕ_q of the truncation points (p_l, p_u), (q_l, q_u) of the TS-BBM channel model. Recall that η_p, η_q represent the area under the beta pdf curve (probability) between the chosen truncation points. For choosing the truncation intervals of a TS-BBM channel model, we let η_p, η_q be equal to some large probability value, e.g., $\eta_p = 1 - \epsilon$ and $\eta_q = 1 - \epsilon$ where $\epsilon \ll 1$. Subject to the constraints on η_p, η_q, the choice of the truncation intervals should be such that the respective differences between the mean and the variance of the number of bit errors per frame for the TS-BBM and the BBM channel models are minimized; i.e., $|\Delta^{(0)}_{mean}|$, $|\Delta^{(1)}_{mean}|$ and $|\Delta^{(0)}_{var}|$, $|\Delta^{(1)}_{var}|$ are minimized. Ideally, we would want to solve the following optimization problems to determine the optimal truncation intervals for the TS-BBM channel model:

$$\begin{aligned}\text{(P1) Choose points } p_l^*, p_u^* \text{ s.t. } \eta_p = 1-\epsilon \text{ and}\\ |\Delta^{(0)}_{mean}| \text{ and } |\Delta^{(0)}_{var}| \text{ are minimized.}\end{aligned} \tag{4.49}$$

$$\begin{aligned}\text{(P2) Choose points } q_l^*, q_u^* \text{ s.t. } \eta_q = 1-\epsilon \text{ and}\\ |\Delta^{(1)}_{mean}| \text{ and } |\Delta^{(1)}_{var}| \text{ are minimized.}\end{aligned} \tag{4.50}$$

However there are some major issues we encounter when trying to solve (P1) and (P2) directly. The first issue is that the constraints in the above optimization problems involve incomplete beta functions (expressions for η_p and η_q) which do not have a closed form and the constraints are nonlinear. The second issue is that it is not known if $|\Delta^{(0)}_{mean}|$ and $|\Delta^{(0)}_{var}|$ (and similarly $|\Delta^{(1)}_{mean}|$ and $|\Delta^{(1)}_{var}|$) can be minimized simultaneously subject to the given constraints; i.e., it is not known if feasible solutions exist for problems (P1) and (P2).

Hence we propose a numerical discrete search algorithm to find the truncation intervals that minimize $|\Delta^{(0)}_{mean}|$ (resp., $|\Delta^{(1)}_{mean}|$) and $|\Delta^{(0)}_{var}|$ (resp., $|\Delta^{(1)}_{var}|$) separately, subject to the relaxed constraints $1-\epsilon \leq \eta_p < 1$ and $1-\epsilon \leq \eta_q < 1$. The justification for the relaxed constraints is that, in the context of deriving a TS-BBM channel model from a BBM channel model, the specific values of η_p and η_q are not very important in practice as long as they are close to 1, which ensures that the TS-BBM channel model preserves the variability (randomness) of the original BBM channel model as much as possible. The numerical discrete search algorithm is described by Algorithms 4 and 5. The search for the truncation interval is done by dividing the continuous interval $[0,1]$ into a set of discrete points with resolution μ. Using pairs of these points as endpoints, windows that satisfy the relaxed constraints are identified. The algorithm then computes $|\Delta^{(x)}_{mean}|$ or $|\Delta^{(x)}_{var}|$, and picks the window with a minimum value as the truncation interval. Fig. 4.1 shows a plot of the $|\Delta^{(x)}_{mean}|$ and $|\Delta^{(x)}_{var}|$ obtained using the numerical discrete search algorithm. We note that it appears that an absolute minimum exists for $|\Delta^{(x)}_{mean}|$ functions whereas for $|\Delta^{(x)}_{var}|$ functions, it does not appear to be so. However choosing a small μ ensures that the algorithm provides accurate results for practical purposes, as shown in Table 4.4. The complexity of Algorithm 4 is $O(l \log l)$ where $l = 1/\mu$.

Algorithm 4 Get Truncation Interval

```
Input: $N$, $\alpha$, $\beta$, $\epsilon$, $\mu$, optParam
Output: Truncation interval points $\hat{\theta}_l$ and $\hat{\theta}_u$
 1: cIntervals = searchCandidates($\alpha$, $\beta$, $\epsilon$, $\mu$)
 2: Initialize $\bar{\Delta}$ as an empty vector.
 3: Initialize intervalCount = 0.
 4: for $\theta_l, \theta_u \in$ cIntervals do
 5:    Compute $\eta_\theta$ using (4.10).
 6:    Compute $\delta_\theta$ using (4.12) and $\phi_\theta$ using (4.14).
 7:    if optParam == "mean" then
 8:       $\bar{\Delta}$[intervalCount] = $|\Delta^{(x)}_{mean}|$
 9:    else
10:       $\bar{\Delta}$[intervalCount] = $|\Delta^{(x)}_{var}|$
11:    intervalCount = intervalCount + 1
12: minimumIndex = $\arg\min \bar{\Delta}$
13: return $\hat{\theta}_l$, $\hat{\theta}_u$ = cIntervals[minimumIndex]
```

Algorithm 5 searchCandidates(α, β, ϵ, μ)

```
Input: $\alpha$, $\beta$, $\epsilon$, $\mu$
Output: List of candidate truncation interval pairs $[\theta_l, \theta_u]$.
 1: Divide the unit interval $[0, 1]$ into $l$ equal intervals where $l = 1/\mu$ and store the
    points in a list candidatePoints.
 2: Compute cumulative distribution function $F$ of Beta$(\alpha, \beta)$ at every point in candi-
    datePoints.
 3: Initialize candidateList as an empty list.
 4: for startPoint $\in$ candidatePoints do
 5:    Using binary search, find the smallest interval [startPoint, endPoint] such that
       $F$(endPoint) $-$ $F$(startPoint) $\geq 1 - \epsilon$.
 6:    Add [startPoint, endPoint] to candidateList.
 7: return candidateList
```

4.3.4 Results

Table 4.2: Truncation intervals for the TS-BBM channel models obtained using Algorithm 4 using $N = 8192$, $\mu = 10^{-6}$, $\epsilon = 0.01$ for vendor-A chip. All truncation interval points should be multiplied by 10^{-3}.

P/E	**Vendor-A, Upper Page**							
Cycles	min $\lvert\Delta_{mean}^{(x)}\rvert$				min $\lvert\Delta_{var}^{(x)}\rvert$			
	$x = 0$		$x = 1$		$x = 0$		$x = 1$	
	p_l	p_u	q_l	q_u	p_l	p_u	q_l	q_u
6000	1.64	4.89	0.78	2.64	1.71	6.16	0.82	3.36
8000	2.66	8.35	1.56	4.69	2.79	11.02	1.63	6.01
10000	4.06	12.51	2.54	7.03	4.26	16.18	2.66	8.81
	Vendor-A, Lower Page							
6000	0.000	0.064	2.76	9.14	0.000	0.064	2.90	12.34
8000	0.001	0.083	3.94	12.56	0.001	0.083	4.13	16.79
10000	0.001	0.111	5.64	16.34	0.001	0.111	5.91	21.87

Table 4.3: Truncation intervals for the TS-BBM channel models obtained using Algorithm 4 using $N = 8192$, $\mu = 10^{-6}$, $\epsilon = 0.01$ for vendor-B chip. All truncation interval points should be multiplied by 10^{-3}.

P/E	**Vendor-B, Upper Page**							
Cycles	min $\lvert\Delta_{mean}^{(x)}\rvert$				min $\lvert\Delta_{var}^{(x)}\rvert$			
	$x = 0$		$x = 1$		$x = 0$		$x = 1$	
	p_l	p_u	q_l	q_u	p_l	p_u	q_l	q_u
6000	0.36	1.37	0.31	2.29	0.38	1.64	0.34	3.41
8000	0.80	3.05	0.60	4.18	0.84	3.78	0.65	5.56
10000	1.44	6.05	1.17	6.63	1.53	7.83	1.25	10.84
	Vendor-B, Lower Page							
6000	0.000	0.004	1.70	6.63	0.000	0.004	1.80	8.68
8000	0.000	0.004	3.58	13.04	0.000	0.004	3.78	18.90
10000	0.000	0.004	6.33	22.71	0.000	0.004	6.68	30.54

Table 4.2 and Table 4.3 show the truncation intervals for the TS-BBM channel models obtained using Algorithm 4 for vendor-A and vendor-B flash memory chips, respectively. We observe that the truncation intervals corresponding to the min $|\Delta_{var}^{(x)}|$ constraint are wider than those corresponding to the min $|\Delta_{mean}^{(x)}|$ constraint. This is expected because a wider support would lead to a larger variance for the number of bit errors per frame in the TS-BBM channel model. Fig. 4.2 shows the beta pdf and the trun-

Table 4.4: Comparison of mean and variance of the number of bit errors per frame obtained from experiment, BBM and TS-BBM channel models using $N = 8192$, $\epsilon = 0.01$ and $\mu = 10^{-6}$ for vendor-A chip.

P/E Cycles	**Vendor-A, Upper Page**							
	Experiment		**BBM**		**TS-BBM** min $\lvert\Delta^{(x)}_{mean}\rvert$		**TS-BBM** min $\lvert\Delta^{(x)}_{var}\rvert$	
	Mean	**Variance**	**Mean**	**Variance**	**Mean**	**Variance**	**Mean**	**Variance**
6000	18.43	30.06	18.43	27.06	18.43	26.42	18.52	26.79
8000	32.01	66.43	32.01	57.88	32.01	55.96	32.17	56.97
10000	48.88	125.99	48.88	105.10	48.88	100.92	49.11	102.97
	Vendor-A, Lower Page							
6000	21.90	46.71	21.90	46.90	21.90	45.04	22.01	46.01
8000	30.55	75.89	30.55	76.25	30.55	72.86	30.71	74.54
10000	41.37	111.35	41.37	111.92	41.37	106.68	41.56	109.13

cation intervals at 8,000 P/E cycles. We also observe that for the TS-Beta distributions of the lower page $0 \rightarrow 1$ bit errors, the truncation intervals for both the min $|\Delta^{(x)}_{mean}|$ and min $|\Delta^{(x)}_{var}|$ constraints are exactly the same. This is because the $0 \rightarrow 1$ bit error probability in the lower page is extremely small during P/E cycling, as was observed in Chapter 2, which leads to a very narrow-width beta distribution. Because we have $\epsilon = 0.01$, in this case there is only one truncation interval that satisfies the probability mass condition given in step 5 of Algorithm 5. The mean and variance of the total number of bit errors per frame for the TS-BBM channel models in Table 4.2 are shown in Table 4.4 and are compared with the statistics obtained from empirical data and the BBM channel models. We observe that the TS-BBM channel models optimized for the min $|\Delta^{(x)}_{mean}|$ constraint are able to match the mean corresponding to the BBM channel models, which yields a precise estimate of the average raw bit error rate (RBER).

Fig. 4.3 shows the FER performance of a regular QC-LDPC code and a Polar code, respectively, using empirical data, as well as the 2-BAC, 2-BBM and 2-TS-BBM channel models for vendor-B chip. The empirical FER performance estimates are obtained from the error data collected from MLC flash memory chips during P/E cycling experiments. To estimate the FER performance using the proposed channel models, Monte-Carlo simulations are used where pseudo-random codewords of the code are generated and transmitted through the appropriate channel model and the received codeword is decoded. At least 400 frame errors are recorded for FER estimation. The simulation

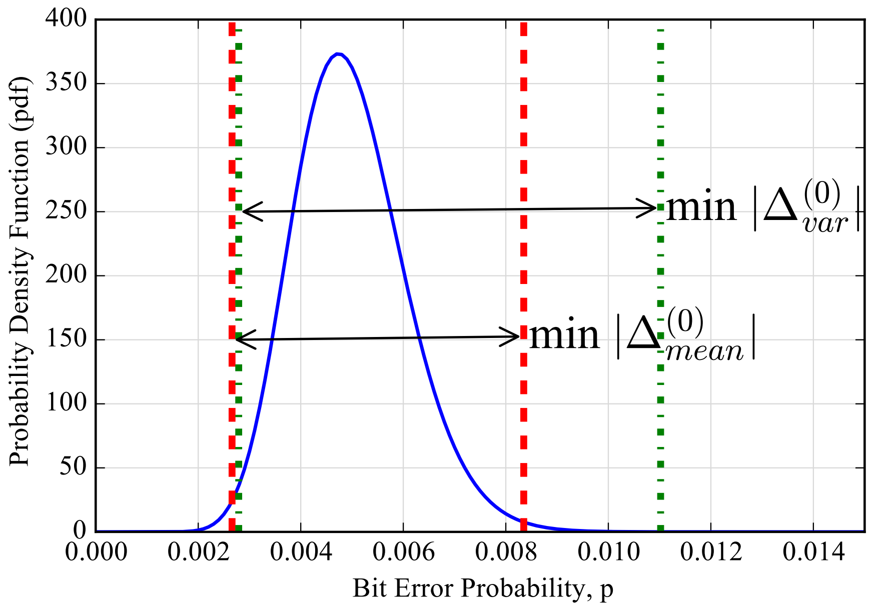


Figure 4.2: Plot showing the beta pdf corresponding to the upper page $0 \rightarrow 1$ errors BBM channel model and the truncation interval points, p_l and p_u, corresponding to TS-BBM channel models in Table 4.2 at 8,000 P/E cycles for vendor-A chip.

of the TS-BBM channel model is implemented using rejection sampling and the beta random variates are generated using the SciPy library [25].

The regular quasi-cyclic LDPC (QC-LDPC) code parameters are $N = 8192$, $k = 7683$, with $d_c = 64$ and $d_v = 4$, where d_c and d_v refer to the check node and variable node degrees, respectively, in the parity check matrix. A sum-product belief propagation decoder with a maximum of 50 iterations and early termination is used to decode the QC-LDPC code. A detailed description of construction of the QC-LDPC code is given in Chapter 3. The polar code parameters are $N = 8192$, $k = 7684$ and it is optimized for a binary symmetric channel (BSC) with bit error probability $p = 0.001$ using the construction technique proposed in [27]. The successive cancellation list (SC-List) decoder proposed in [28] is used for decoding the polar code with list size 8.

We observe that the ECC FER performance estimates obtained using the min $|\Delta^{(x)}_{var}|$ 2-TS-BBM channel model are slightly more accurate (with respect to the empirical results) than those obtained using the min $|\Delta^{(x)}_{mean}|$ 2-TS-BBM channel model. We also observe that the min $|\Delta^{(x)}_{var}|$ 2-TS-BBM channel model is essentially the same as the BBM channel model with respect to ECC FER performance estimation.

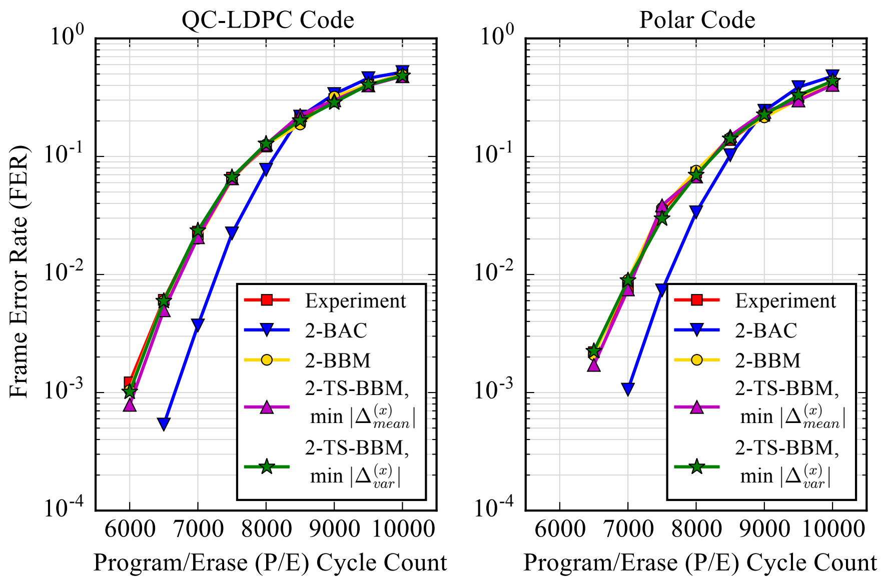


Figure 4.3: Comparison of FER performance of a regular QC-LDPC code and a Polar code using empirical error data and simulated error data from the 2-BAC, 2-BBM and 2-TS-BBM channel models for vendor-B chip.

4.4 Capacity of the TS-BBM Channel Model

The capacity of the TS-BBM channel model can be derived using the compound channel approach presented in Section 4.2, as shown in Lemma 4.4.1 and Theorem 4.4.2 below. Lemma 4.4.1 shows that a pair of BACs can be ordered with respect to their mutual information irrespective of the input probability distribution, when their $0 \to 1$ and $1 \to 0$ bit error probability pairs (p, q), can be ordered componentwise. This result is key to showing, in Theorem 4.4.2, that the capacity of the TS-BBM channel, which is a compound channel consisting of a set of BACs, achieves the upper bound given by the max-min inequality [35] on the compound channel capacity.

Lemma 4.4.1. *Given two BACs, $BAC(\mathcal{X}, \mathcal{Z}, (p_1, q_1))$, $BAC(\mathcal{X}, \mathcal{Y}, (p_2, q_2))$ with $0 \le p_2 \le p_1 < \frac{1}{2}$ and $0 \le q_2 \le q_1 < \frac{1}{2}$, there always exists a degrading channel $BAC(\mathcal{Y}, \mathcal{Z}, (p', q'))$ such that,*

$$\Pr(z|x) = \sum_{y \in \mathcal{Y}} \Pr(y|x) \Pr(z|y) \tag{4.51}$$

and $I(X; Z) \le I(X; Y)$.

Proof. For the first part, it is sufficient to show that p' and q' obtained by solving the set of equations resulting from (4.51) are always positive. From (4.51),

$$(1-p_2)p' + p_2(1-q') = p_1 \tag{4.52}$$

$$q_2(1-p') + (1-q_2)q' = q_1. \tag{4.53}$$

Solving (4.52) and (4.53),

$$q' = \frac{(q_1-q_2)(1-p_2)+(p_1-p_2)q_2}{1-p_2-q_2} \tag{4.54}$$

$$p' = \frac{(p_1-p_2)(1-q_2)+(q_1-q_2)p_2}{1-p_2-q_2}. \tag{4.55}$$

Clearly, $q' \geq 0$ and $p' \geq 0$.
Since $X \rightarrow Y \rightarrow Z$ form a Markov chain, the data processing inequality [37] implies that we have $I(X;Z) \leq I(X;Y)$. □

Assume, without loss of generality,

$$0 \leq p_l < p_u < \frac{1}{2} \qquad 0 \leq q_l < q_u < \frac{1}{2}. \tag{4.56}$$

Theorem 4.4.2. *The capacity of a TS-BBM channel is equal to the capacity of a BAC$(\mathcal{X}, \mathcal{Y}, (p_u, q_u))$ given by*

$$C_{\textit{TS-BBM}} = \left(\frac{p_u}{1-p_u-q_u}\right)h(q_u) - \left(\frac{1-q_u}{1-p_u-q_u}\right)h(p_u) + log_2\left(1 + 2^{\frac{h(p_u)-h(q_u)}{1-p_u-q_u}}\right). \tag{4.57}$$

Proof. The TS-BBM channel is also a compound channel consisting of a set of BACs with varying states $s \in \mathcal{S}$ where $\mathcal{S} = \{(p,q) | p \sim \text{TS-Beta}(p_l, p_u; a, b), q \sim \text{TS-Beta}(q_l, q_u; c, d)\}$. Using the compound channel approach, the capacity of the TS-BBM channel is given by

$$C_{\text{TS-BBM}} = \sup_{\bar{\pi}} \inf_{s \in \mathcal{S}} I_{\bar{\pi},s}(X;Y) \tag{4.58}$$

Using Lemma 1, $\forall s \in \mathcal{S}$ and $\bar{\pi}$, $I_{\bar{\pi},s}(X;Y) \geq I_{\bar{\pi},s'}(X;Y)$, where $s' = (p_u, q_u)$. Therefore,

$$\inf_{s \in \mathcal{S}} I_{\bar{\pi},s}(X;Y) = I_{\bar{\pi},s'}(X;Y) \tag{4.59}$$

so

$$C_{\text{TS-BBM}} = \sup_{\bar{\pi}} I_{\bar{\pi},s'}(X;Y) \tag{4.60}$$

From (4.60), $C_{\text{TS-BBM}}$ is equal to the capacity of BAC$(\mathcal{X}, \mathcal{Y}, (p_u, q_u))$ and is given by (4.57). □

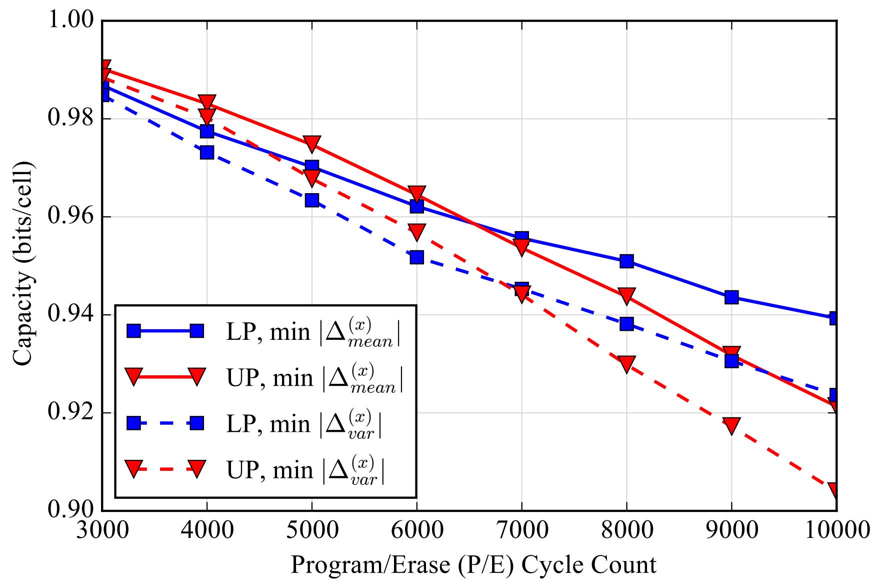


Figure 4.4: Plot showing the evolution of TS-BBM channel model capacities as a function of P/E cycle count for lower and upper pages of vendor-A flash memory chip. $\epsilon = 0.01$, $\mu = 10^{-6}$.

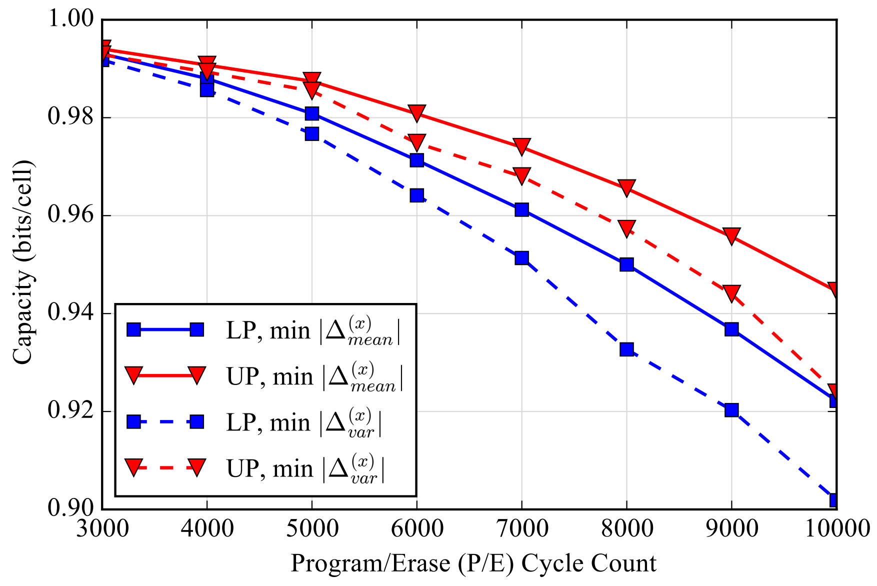


Figure 4.5: Plot showing the evolution of TS-BBM channel model capacities as a function of P/E cycle count for lower and upper pages of vendor-B flash memory chip. $\epsilon = 0.01$, $\mu = 10^{-6}$.

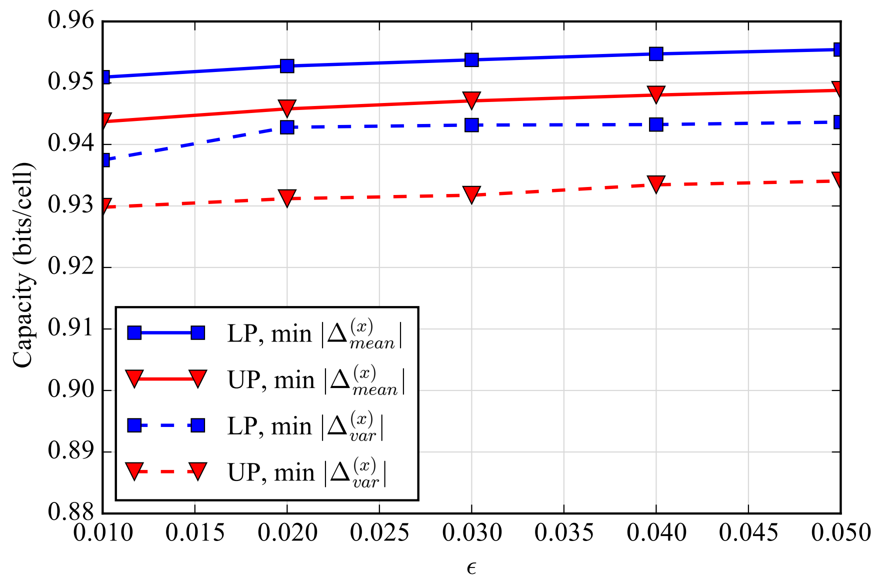


Figure 4.6: Plot showing the TS-BBM channel model capacities corresponding to lower and upper pages of vendor-A flash memory chip at 8,000 P/E cycles for different values of ϵ parameter. $\mu = 10^{-8}$ for LP, min $|\Delta^{(x)}_{var}|$ and $\mu = 10^{-6}$ for others.

Figures 4.4 and 4.5 show the TS-BBM channel model capacity estimates for both the lower and upper pages of vendor-A and vendor-B flash memory chips respectively. The changing TS-BBM channel model capacity estimates across P/E cycles are indicative of the rate of degradation of the underlying flash memory channel, for e.g., the TS-BBM channel model capacity for the upper page of vendor-A chip decreases from ∼0.99 at $3,000$ P/E cycles to ∼0.92 at $10,000$ P/E cycles. We also note that for vendor-B chip, the upper page TS-BBM channel models consistently have larger capacity than the lower page TS-BBM channel models at all P/E cycles, whereas this is not the case for vendor-A chip. Such knowledge of the page capacities as a function of P/E cycle count in flash memories could be useful and advantageous for ECC design. For example, the estimates of the page capacities can be directly utilized for designing rate adaptive ECC schemes and for optimally distributing redundancy between the lower and upper pages at different lifetime stages of the MLC flash memory. We also observe that the TS-BBM channel models optimized for min $|\Delta^{(x)}_{mean}|$ have larger capacities than those optimized for min $|\Delta^{(x)}_{var}|$. Fig. 4.6 shows the variation of the TS-BBM channel model capacities with respect to the ϵ parameter. Choosing a larger ϵ generally leads to a larger capacity TS-BBM channel model; however a larger ϵ also results in a narrower truncation interval of the beta pdf. Hence a TS-BBM channel model corresponding to large ϵ may not be able to accurately model the variance of the number of bit errors per frame resulting in optimistic ECC FER performance estimates.

4.4.1 Coding for the TS-BBM Channel Model

Recall that the TS-BBM channel model is a compound channel consisting of a set of BACs with varying states $s \in \mathcal{S}$ where $\mathcal{S} = \{(p,q)|p \sim \text{TS-Beta}(p_l, p_u; a, b), q \sim \text{TS-Beta}(q_l, q_u; c, d)\}$. From Theorem 4.4.2, the capacity of a TS-BBM channel model is equal to the capacity of the most noisy channel i.e., $\text{BAC}(\mathcal{X}, \mathcal{Y}, (p_u, q_u))$. Although coding techniques based on polar codes and sparse graph codes for achieving the capacity of a single BAC have been proposed in the literature [38, 39, 40], the application of such techniques to a set of BACs as represented by the TS-BBM compound channel appears to be a difficult problem.

Therefore, we look at existing coding techniques that can achieve the symmetric information rate (SIR) of the TS-BBM channel. The SIR of a binary DMC is defined as its mutual information with a uniform input probability distribution. It is known that the difference between the capacity and the SIR of a binary DMC ($\Delta_{C,SIR}$) is at most ~5.8% of its capacity [41]. Hence, we compute and show this difference between the capacity and the SIR of the TS-BBM channel model as a function of the P/E cycle count, in Figures 4.7 and 4.8, for the lower page and upper page models, respectively. We observe that this difference between the capacity and the SIR of the TS-BBM channel model is extremely small for both pages. Therefore for practical applications, the loss (in capacity) is almost negligible if we use linear codes that achieve the SIR of the TS-BBM channel.

Under the assumption that both the encoder and decoder have no knowledge of the channel state, we require a single code that can achieve the SIR of all the component BACs of the TS-BBM channel. Polar codes have been shown to achieve the SIR of any binary DMC [42]. The SIR of a single BAC can be achieved by a polar code constructed as shown in [42], with a suitable choice of values for the frozen bits. However, finding such a set of frozen bit values of a polar code for asymmetric binary DMCs such as the BAC is an open problem. Even though the component BACs in the TS-BBM channel model are ordered by degradation, it is not clear if the frozen bit indices of polar codes corresponding to these component BACs are aligned. We leave these problems open for

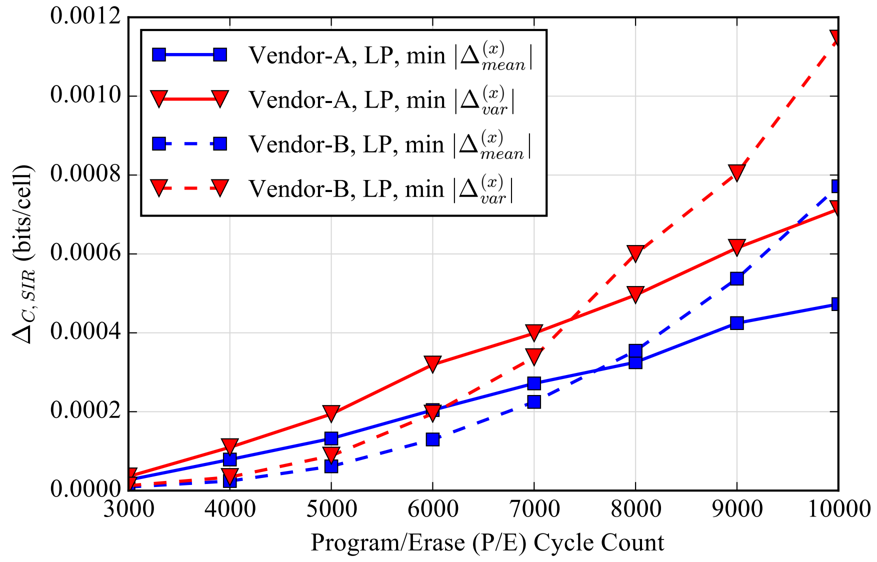


Figure 4.7: Plot showing the difference between the capacity and the symmetric information rate (SIR) for the lower page TS-BBM channel models corresponding to vendor-A and vendor-B flash memory chips. $\epsilon = 0.01$, $\mu = 10^{-6}$.

future work.

4.5 Conclusion

Using the compound channel model approach, we showed that the beta-binomial (BBM) channel model for MLC flash memories proposed in Chapter 3 has zero capacity. Flash memories in practice do have non-zero positive capacities and hence the BBM channel model appears to be a very pessimistic model. As an alternative, based on empirical observations, we proposed the truncated-support beta-binomial (TS-BBM) channel model for MLC flash memories and derived its capacity. We used empirical error characterization data from 1X-nm and 2Y-nm MLC flash memories to obtain the TS-BBM channel model parameters and estimate its capacity. Using FER performance results for a regular QC-LDPC code and a polar code, we also showed that the 2-TS-BBM channel model is almost as good as the BBM channel model for ECC FER performance estimation. When using a single binary ECC for all pages in MLC flash memories, the TS-BBM channel model capacity represents an upper bound on the rate of the ECC. This is because, the ECC must be able to correct all the errors resulting from the most noisy channel in the set of channels represented by the TS-BBM channel model.

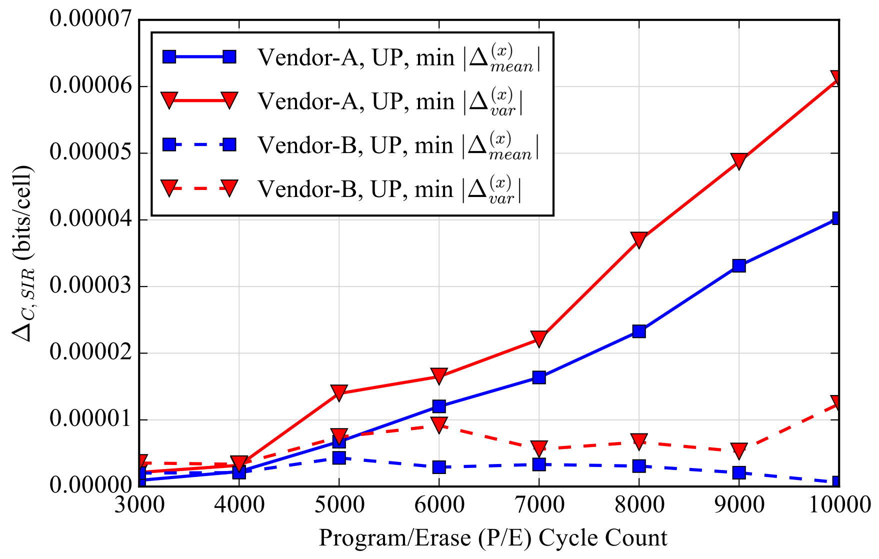


Figure 4.8: Plot showing the difference between the capacity and the symmetric information rate (SIR) for the upper page TS-BBM channel models corresponding to vendor-A and vendor-B flash memory chips. $\epsilon = 0.01$, $\mu = 10^{-6}$.

4.6 Appendix

4.6.1 Proof of Proposition 4.3.3

The mean of a TS-BBM random variable Z is given by

$$\begin{aligned} \mathrm{E}[Z] &= \sum_{z=0}^{n} z \Pr(Z = z) \\ &= \sum_{z=0}^{n} z \binom{n}{z} \left(\frac{B_{\theta_u}(\alpha + z, \beta + n - z) - B_{\theta_l}(\alpha + z, \beta + n - z)}{B_{\theta_u}(\alpha, \beta) - B_{\theta_l}(\alpha, \beta)} \right). \end{aligned} \tag{4.61}$$

Let

$$\begin{aligned} t_{\theta_u} &= \sum_{z=0}^{n} z \binom{n}{z} B_{\theta_u}(\alpha + z, \beta + n - z) \\ &= \sum_{z=0}^{n} z \binom{n}{z} \int_0^{\theta_u} \lambda^{\alpha+z-1}(1-\lambda)^{\beta+n-z-1} \mathrm{d}\lambda \\ &= \int_0^{\theta_u} \left(\sum_{z=0}^{n} z \binom{n}{z} \lambda^{z-1}(1-\lambda)^{n-z} \right) \lambda^{\alpha-1}(1-\lambda)^{\beta-1} \mathrm{d}\lambda. \end{aligned}$$

The term in parentheses in the above equation is the expected value of a binomial random variable with parameters n and λ, namely $n\lambda$. Substituting, we have

$$t_{\theta_u} = n \int_0^{\theta_u} \lambda^{\alpha+1-1}(1-\lambda)^{\beta-1} \mathrm{d}\lambda$$

$$= nB_{\theta_u}(\alpha+1,\beta). \tag{4.62}$$

Similarly,

$$t_{\theta_l} = nB_{\theta_l}(\alpha+1,\beta). \tag{4.63}$$

Substituting (4.62), (4.63) in (4.61), we find

$$\mathrm{E}[Z] = n\left(\frac{B_{\theta_u}(\alpha+1,\beta) - B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}\right). \tag{4.64}$$

To relate $\mathrm{E}[Z]$ and $\mathrm{E}[\tilde{Z}]$ where $\tilde{Z}$ is a beta-binomial random variable, we use the recurrence relations [36]

$$\frac{B_x(\alpha+1,\beta)}{B(\alpha+1,\beta)} = \frac{B_x(\alpha,\beta)}{B(\alpha,\beta)} - \frac{x^\alpha(1-x)^\beta}{\alpha B(\alpha,\beta)} \tag{4.65}$$

$$B(\alpha+1,\beta) = \left(\frac{\alpha}{\alpha+\beta}\right)B(\alpha,\beta) \tag{4.66}$$

$$B(\alpha,\beta+1) = \left(\frac{\beta}{\alpha+\beta}\right)B(\alpha,\beta). \tag{4.67}$$

Using (4.65), (4.66), (4.67) in (4.64), we get

$$\begin{aligned}\mathrm{E}[Z] &= \frac{n\alpha}{\alpha+\beta} - \left(\frac{n}{\alpha+\beta}\right)\frac{\delta_\theta}{\eta_\theta} \\ &= \mathrm{E}[\tilde{Z}] - \left(\frac{n}{\alpha+\beta}\right)\frac{\delta_\theta}{\eta_\theta}. \end{aligned}\tag{4.68}$$

To compute the variance, we compute the second moment of Z, $\mathrm{E}[Z^2]$, as follows,

$$\begin{aligned}\mathrm{E}[Z^2] &= \sum_{z=0}^{n} z^2 \Pr(Z=z) \\ &= \sum_{z=0}^{n} z^2 \binom{n}{z}\left(\frac{B_{\theta_u}(\alpha+z,\beta+n-z) - B_{\theta_l}(\alpha+z,\beta+n-z)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}\right).\end{aligned}\tag{4.69}$$

Let,

$$\begin{aligned}\bar{t}_{\theta_u} &= \sum_{z=0}^{n} z^2\binom{n}{z} B_{\theta_u}(\alpha+z,\beta+n-z) \\ &= \sum_{z=0}^{n} z^2\binom{n}{z}\int_0^{\theta_u}\lambda^{\alpha+z-1}(1-\lambda)^{\beta+n-z-1}\mathrm{d}\lambda \\ &= \int_0^{\theta_u}\left(\sum_{z=0}^{n} z^2\binom{n}{z}\lambda^{z-1}(1-\lambda)^{n-z}\right)\lambda^{\alpha-1}(1-\lambda)^{\beta-1}\mathrm{d}\lambda.\end{aligned}$$

The term in parentheses in the above equation is the second moment of a binomial random variable with parameters n and λ, namely $n(n-1)\lambda^2 + n\lambda$. Substituting, we have

$$\begin{aligned}\bar{t}_{\theta_u} &= n(n-1)\int_0^{\theta_u} \lambda^{\alpha+2-1}(1-\lambda)^{\beta-1}\mathrm{d}\lambda + n\int_0^{\theta_u} \lambda^{\alpha+1-1}(1-\lambda)^{\beta-1}\mathrm{d}\lambda \\ &= n(n-1)B_{\theta_u}(\alpha+2,\beta) + nB_{\theta_u}(\alpha+1,\beta). \qquad (4.70)\end{aligned}$$

Similarly,

$$\bar{t}_{\theta_l} = n(n-1)B_{\theta_l}(\alpha+2,\beta) + nB_{\theta_l}(\alpha+1,\beta). \qquad (4.71)$$

Substituting (4.70), (4.71) in (4.69), we find

$$\begin{aligned}\mathrm{E}[Z^2] &= n(n-1)\frac{B_{\theta_u}(\alpha+2,\beta) - B_{\theta_l}(\alpha+2,\beta)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)} + n\frac{B_{\theta_u}(\alpha+1,\beta) - B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}; \qquad (4.72)\\ \mathrm{Var}[Z] &= \mathrm{E}[Z^2] - (\mathrm{E}[Z])^2 \\ &= n\left(\frac{B_{\theta_u}(\alpha+1,\beta) - B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}\right)\left(1 - n\left(\frac{B_{\theta_u}(\alpha+1,\beta) - B_{\theta_l}(\alpha+1,\beta)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}\right)\right) + \\ &\quad n(n-1)\left(\frac{B_{\theta_u}(\alpha+2,\beta) - B_{\theta_l}(\alpha+2,\beta)}{B_{\theta_u}(\alpha,\beta) - B_{\theta_l}(\alpha,\beta)}\right). \qquad (4.73)\end{aligned}$$

Using the recurrence relations (4.65), (4.66), (4.67) in (4.73), we get

$$\begin{aligned}\mathrm{Var}[Z] = \mathrm{Var}[\tilde{Z}] &- \left(\frac{n\beta(\alpha+\beta+n)}{(\alpha+\beta)^2(\alpha+\beta+1)}\right)\frac{\delta_\theta}{\eta_\theta} - \frac{n^2}{(\alpha+\beta)^2}\left(\frac{\delta_\theta}{\eta_\theta}\right)^2 - \\ &\left(\frac{n(n-1)}{\alpha+\beta+1}\right)\frac{\phi_\theta}{\eta_\theta} \qquad (4.74)\end{aligned}$$

where $\mathrm{Var}[\tilde{Z}]$ is the variance of a BBM random variable and η_θ, δ_θ, ϕ_θ are defined in equations (4.10), (4.12), (4.14), respectively. $\square$

4.6.2 Proof of Proposition 4.3.4

The probability distribution of $K^{(0)}$ in terms of the probability distribution of $K_m^{(0)}$ is given by

$$\begin{aligned}\Pr(K^{(0)} = k) &= \sum_{m=k}^{N} \frac{\binom{N}{m}}{2^N}\Pr(K_m^{(0)} = k) \\ &= \sum_{m=k}^{N} \frac{\binom{N}{m}}{2^N}\binom{m}{k}\left(\frac{B_{p_u}(a+k,b+m-k)}{B_{p_u}(a,b) - B_{p_l}(a,b)} - \right. \\ &\qquad \left. \frac{B_{p_l}(a+k,b+m-k)}{B_{p_u}(a,b) - B_{p_l}(a,b)}\right). \qquad (4.75)\end{aligned}$$

The mean of $K^{(0)}$ is given by

$$\begin{aligned}\mathrm{E}[K^{(0)}] &= \sum_{k=0}^{N} k \Pr(K^{(0)} = k) \\ &= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} \mathrm{E}[K_m^{(0)}] \\ &= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} m \left(\frac{B_{p_u}(a+1,b) - B_{p_l}(a+1,b)}{B_{p_u}(a,b) - B_{p_l}(a,b)} \right) \\ &= \frac{N}{2} \left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} \right). \end{aligned} \tag{4.76}$$

The second moment of $K^{(0)}$ is given by

$$\begin{aligned}\mathrm{E}[(K^{(0)})^2] &= \sum_{k=0}^{N} k^2 \Pr(K^{(0)} = k) \\ &= \frac{1}{2^N} \sum_{m=0}^{N} \binom{N}{m} \mathrm{E}[(K_m^{(0)})^2]. \end{aligned} \tag{4.77}$$

Using (4.72),

$$\begin{aligned}\mathrm{E}[(K_m^{(0)})^2] = m(m-1) &\left(\frac{B_{p_u}(a+2,b) - B_{p_l}(a+2,b)}{B_{p_u}(a,b) - B_{p_l}(a,b)} \right) \\ &+ m \left(\frac{B_{p_u}(a+1,b) - B_{p_l}(a+1,b)}{B_{p_u}(a,b) - B_{p_l}(a,b)} \right). \end{aligned} \tag{4.78}$$

Substituting (4.78) in (4.77) and simplifying, we get

$$\begin{aligned}\mathrm{E}[(K^{(0)})^2] &= \frac{N(N-1)}{4} \left(\frac{B_{p_u}(a+2,b) - B_{p_l}(a+2,b)}{B_{p_u}(a,b) - B_{p_l}(a,b)} \right) + \\ &\quad \frac{N}{2} \left(\frac{B_{p_u}(a+1,b) - B_{p_l}(a+1,b)}{B_{p_u}(a,b) - B_{p_l}(a,b)} \right) \\ &= \frac{N(N-1)}{4} \left(\frac{U_{p_u}^{(2)} - U_{p_l}^{(2)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} \right) + \frac{N}{2} \left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} \right). \end{aligned} \tag{4.79}$$

Therefore, using (4.76) and (4.79), we get

$$\begin{aligned}\mathrm{Var}[K^{(0)}] &= \mathrm{E}[(K^{(0)})^2] - (\mathrm{E}[K^{(0)}])^2 \\ &= \frac{N}{2} \left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} \right) \left(1 - \frac{N}{2} \left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} \right) \right) + \\ &\quad \frac{N(N-1)}{4} \left(\frac{U_{p_u}^{(2)} - U_{p_l}^{(2)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} \right). \end{aligned} \tag{4.80}$$

Note that we have used the following combinatorial identities to simplify the summations in this proof:

$$\sum_{m=0}^{N} \binom{N}{m} m = N2^{N-1} \tag{4.81}$$

$$\sum_{m=0}^{N} \binom{N}{m} m^2 = (N+N^2)2^{N-2}. \tag{4.82}$$

The expressions for $\mathrm{E}[K^{(1)}]$ and $\mathrm{Var}[K^{(1)}]$ can be derived similarly. □

4.6.3 Proof of Proposition 4.3.5

By definition, we have

$$K = K^{(0)} + K^{(1)} \tag{4.83}$$

$$\mathrm{E}[K] = \mathrm{E}[K^{(0)}] + \mathrm{E}[K^{(1)}]. \tag{4.84}$$

Using the results of Proposition 4.3.4, we have

$$\mathrm{E}[K] = \frac{N}{2}\left(\frac{U_{p_u}^{(1)} - U_{p_l}^{(1)}}{U_{p_u}^{(0)} - U_{p_l}^{(0)}} + \frac{V_{q_u}^{(1)} - V_{q_l}^{(1)}}{V_{q_u}^{(0)} - V_{q_l}^{(0)}}\right). \tag{4.85}$$

To compute $\mathrm{Var}[K]$, we first need $\mathrm{E}[K^2]$, derived as follows:

$$\mathrm{E}[K^2] = \frac{1}{2^N}\sum_{m=0}^{N}\binom{N}{m}\mathrm{E}[K_m^2] \tag{4.86}$$

$$\text{where} \quad \mathrm{E}[K_m^2] = \mathrm{Var}[K_m] + (\mathrm{E}[K_m])^2. \tag{4.87}$$

$$\mathrm{E}[K_m] = \mathrm{E}[K_m^{(0)}] + \mathrm{E}[K_{N-m}^{(1)}] \tag{4.88}$$

$$\mathrm{Var}[K_m] = \mathrm{Var}[K_m^{(0)}] + \mathrm{Var}[K_{N-m}^{(1)}]. \tag{4.89}$$

Using the results of Proposition 4.3.3, we obtain

$$\begin{aligned}\mathrm{E}[K_m] = {} & m\frac{B_{p_u}(a+1,b) - B_{p_l}(a+1,b)}{B_{p_u}(a,b) - B_{p_l}(a,b)} + \\ & (N-m)\frac{B_{q_u}(c+1,d) - B_{q_l}(c+1,d)}{B_{q_u}(c,d) - B_{q_l}(c,d)}\end{aligned} \tag{4.90}$$

$$\begin{aligned}\mathrm{Var}[K_m] = m\left(\frac{B_{p_u}(a+1,b)-B_{p_l}(a+1,b)}{B_{p_u}(a,b)-B_{p_l}(a,b)}\right) - m^2\left(\frac{B_{p_u}(a+1,b)-B_{p_l}(a+1,b)}{B_{p_u}(a,b)-B_{p_l}(a,b)}\right)^2 + \\ m(m-1)\left(\frac{B_{p_u}(a+2,b)-B_{p_l}(a+2,b)}{B_{p_u}(a,b)-B_{p_l}(a,b)}\right) + \\ (N-m)\left(\frac{B_{q_u}(c+1,d)-B_{q_l}(c+1,d)}{B_{q_u}(c,d)-B_{q_l}(c,d)}\right) - \\ (N-m)^2\left(\frac{B_{q_u}(c+1,d)-B_{q_l}(c+1,d)}{B_{q_u}(c,d)-B_{q_l}(c,d)}\right)^2 + \\ (N-m)(N-m-1)\left(\frac{B_{q_u}(c+2,d)-B_{q_l}(c+2,d)}{B_{q_u}(c,d)-B_{q_l}(c,d)}\right). \qquad (4.91)\end{aligned}$$

$\mathrm{E}[K_m^2]$ can be computed using equations (4.87), (4.90) and (4.91). Using $\mathrm{E}[K_m^2]$ in equation (4.86), $\mathrm{E}[K^2]$ can be computed and thus $\mathrm{Var}[K]$ can be written as

$$\begin{aligned}\mathrm{Var}[K] &= \mathrm{E}[K^2] - (\mathrm{E}[K])^2 \\ &= \frac{N}{2}\left(\frac{U_{p_u}^{(1)}-U_{p_l}^{(1)}}{U_{p_u}^{(0)}-U_{p_l}^{(0)}}\right)\left(1-\frac{N}{2}\left(\frac{U_{p_u}^{(1)}-U_{p_l}^{(1)}}{U_{p_u}^{(0)}-U_{p_l}^{(0)}}\right)\right) \\ &\quad+\frac{N}{2}\left(\frac{V_{q_u}^{(1)}-V_{q_l}^{(1)}}{V_{q_u}^{(0)}-V_{q_l}^{(0)}}\right)\left(1-\frac{N}{2}\left(\frac{V_{q_u}^{(1)}-V_{q_l}^{(1)}}{V_{q_u}^{(0)}-V_{q_l}^{(0)}}\right)\right) \\ &\quad+\frac{N(N-1)}{4}\left(\frac{U_{p_u}^{(2)}-U_{p_l}^{(2)}}{U_{p_u}^{(0)}-U_{p_l}^{(0)}}+\frac{V_{q_u}^{(2)}-V_{q_l}^{(2)}}{V_{q_u}^{(0)}-V_{q_l}^{(0)}}\right) \\ &\quad-\frac{N}{2}\left(\frac{U_{p_u}^{(1)}-U_{p_l}^{(1)}}{U_{p_u}^{(0)}-U_{p_l}^{(0)}}\right)\left(\frac{V_{q_u}^{(1)}-V_{q_l}^{(1)}}{V_{q_u}^{(0)}-V_{q_l}^{(0)}}\right). \qquad (4.92)\end{aligned}$$

Note that we have used the following combinatorial identities to simplify the summations in this proof:

$$\sum_{m=0}^{N}\binom{N}{m}\,m = N2^{N-1} \qquad (4.93)$$

$$\sum_{m=0}^{N}\binom{N}{m}\,m^2 = (N+N^2)2^{N-2}. \qquad (4.94)$$

This completes the proof. □

4.6.4 Proof of Proposition 4.3.6

Using (4.10), (4.12)-(4.14) and the recurrence relations in (4.65)-(4.67), we have

$$\frac{U_{p_u}^{(1)}-U_{p_l}^{(1)}}{U_{p_u}^{(0)}-U_{p_l}^{(0)}} = \frac{a}{a+b} - \frac{1}{(a+b)}\frac{\delta_p}{\eta_p} \qquad (4.95)$$

$$\frac{U^{(2)}_{p_u} - U^{(2)}_{p_l}}{U^{(0)}_{p_u} - U^{(0)}_{p_l}} = \frac{a(a+1)}{(a+b)(a+b+1)} - \frac{a+1}{(a+b)(a+b+1)}\frac{\delta_p}{\eta_p} - \frac{1}{(a+b+1)}\frac{\phi_p}{\eta_p} \tag{4.96}$$

$$\frac{V^{(1)}_{q_u} - V^{(1)}_{q_l}}{V^{(0)}_{q_u} - V^{(0)}_{q_l}} = \frac{c}{c+d} - \frac{1}{(c+d)}\frac{\delta_q}{\eta_q} \tag{4.97}$$

$$\frac{V^{(2)}_{q_u} - V^{(2)}_{q_l}}{V^{(0)}_{q_u} - V^{(0)}_{q_l}} = \frac{c(c+1)}{(c+d)(c+d+1)} - \frac{c+1}{(c+d)(c+d+1)}\frac{\delta_q}{\eta_q} - \frac{1}{(c+d+1)}\frac{\phi_q}{\eta_q}. \tag{4.98}$$

Substituting (4.95) and (4.96) in the expressions for $\mathrm{E}[K^{(0)}]$ and $\mathrm{Var}[K^{(0)}]$ given by Proposition 4.3.4 and simplifying, we derive the relationships between $\mathrm{E}[K^{(0)}]$, $\mathrm{E}[\tilde{K}^{(0)}]$ and between $\mathrm{Var}[K^{(0)}]$, $\mathrm{Var}[\tilde{K}^{(0)}]$ as

$$\mathrm{E}[K^{(0)}] = \mathrm{E}[\tilde{K}^{(0)}] - \Delta^{(0)}_{mean} \tag{4.99}$$

$$\mathrm{Var}[K^{(0)}] = \mathrm{Var}[\tilde{K}^{(0)}] - \Delta^{(0)}_{var} \tag{4.100}$$

The relationships among $\mathrm{E}[K^{(1)}]$, $\mathrm{E}[\tilde{K}^{(1)}]$ and $\mathrm{Var}[K^{(1)}]$, $\mathrm{Var}[\tilde{K}^{(1)}]$ can be derived similarly. Substituting (4.95)-(4.98) in the expressions for $\mathrm{E}[K]$ and $\mathrm{Var}[K]$ given by Proposition 4.3.5 and simplifying, we derive the relationships between $\mathrm{E}[K]$, $\mathrm{E}[\tilde{K}]$ and between $\mathrm{Var}[K]$, $\mathrm{Var}[\tilde{K}]$ as

$$\mathrm{E}[K] = \mathrm{E}[\tilde{K}] - \Delta_{mean} \tag{4.101}$$

$$\mathrm{Var}[K] = \mathrm{Var}[\tilde{K}] - \Delta_{var}. \tag{4.102}$$

$\square$

Acknowledgements

This chapter is in part a reprint of the material in the paper: Veeresh Taranalli, Hironori Uchikawa, and Paul H. Siegel, “On the capacity of the beta-binomial channel model for multi-level cell flash memories,” IEEE Journal on Selected Areas in Communications, vol. 34, no. 9, pp. 2312–2324, September 2016. The dissertation author was the primary investigator and author of this paper, and co-authors have approved the use of the material for this dissertation.

5 Adaptive Linear Programming Decoding of Polar Codes

5.1 Introduction

Polar codes, first introduced in [42], were shown to be capacity-achieving for binary input memoryless output symmetric channels. However, their performance on a binary additive white gaussian noise channel (BAWGNC) with successive cancellation (SC) decoding is unimpressive at practical blocklengths. Thus, improving their performance either by improved decoding algorithms or modified constructions of polar codes has been a recent topic in coding theory. The most notable improvement in error rate performance was observed using the successive cancellation list (SC-List) decoding algorithm proposed in [28]. Alternatively, a cyclic redundancy check (CRC) concatenated polar code with SC-List decoding was also shown to improve the performance significantly [28].

Linear Programming (LP) decoding has been a research topic in coding theory and is attractive mainly because of its maximum likelihood (ML)-certificate property [43]. It was introduced for polar codes in [44] where the sparse factor graph was used to represent the LP polytope instead of the high density parity check matrix. For polar codes over a binary erasure channel (BEC), it was shown that LP decoding achieves capacity and also outperforms SC decoding at finite blocklengths [44]. However, for a BAWGNC, the LP decoder in [44] is suboptimal and performs very poorly.

Adaptive LP decoding techniques were proposed in [45, 46] to improve the decoding time complexity as well as the error rate performance. Based on these techniques, we propose modifications to the adaptive cut generation based LP decoder in [46] that significantly improve its error rate performance for polar codes over a BAWGNC. We then present an algorithm to obtain a smaller factor graph representation of a polar code, called the *reduced factor graph*, which decreases the representation complexity of the fundamental polytope and, hence, improves the decoding time complexity of the modified-adaptive LP decoder.

In Section 5.2, we review the LP decoding of polar codes proposed in [44]. In Section 5.3, we review the adaptive LP decoding techniques [45, 46] and describe the proposed modified-adaptive LP decoder for short blocklength polar codes, along with simulation results. In Section 5.4, we present the algorithm for reducing a polar code sparse factor graph.

5.2 LP Decoding of Polar Codes

Consider a binary linear code $\mathcal{C}_l$ of length N and rate $r = \frac{k}{N}$, where $k < N$ is the number of information bits in a codeword. Let $\mathbf{H}$ denote a parity check matrix for $\mathcal{C}_l$. Suppose a codeword $\mathbf{x} \in \mathcal{C}_l$ is transmitted over a binary input memoryless output symmetric channel and $\mathbf{y}$ is the received vector. ML decoding is equivalent to solving the optimization problem [43]:

$$\text{minimize } \boldsymbol{\gamma}^T\mathbf{x} \;\; \text{subject to } \mathbf{x} \in \mathcal{C}_l \tag{5.1}$$

where $x_i \in \{0, 1\}, i \in 1, \ldots, N$ and $\boldsymbol{\gamma}$ is the vector of log-likelihood ratios (LLR) defined as

$$\gamma_i = \log\left(\frac{\Pr(y_i|x_i = 0)}{\Pr(y_i|x_i = 1)}\right). \tag{5.2}$$

In [43], the ML decoding problem (5.1) was relaxed to a linear programming (LP) problem, where the relaxed polytope, also known as the fundamental polytope $\mathcal{Q}$ has both integral and non-integral vertices. The polytope $\mathcal{Q}$ is defined by linear inequalities, also referred to as constraints, generated from each row j of the parity-check matrix $\mathbf{H}$, given by

$$\sum_{i\in V} x_i - \sum_{i\in\mathcal{N}(j)\setminus V} x_i \leq |V| - 1 \quad \forall\, V \subseteq \mathcal{N}(j) \;\text{ s.t. }\; |V| \;\text{ is odd} \tag{5.3}$$

where $\mathcal{N}(j)$ is the support of the row j of $\mathbf{H}$. This polytope $\mathcal{Q}$ has the ML-certificate property which guarantees that an integral solution of the LP problem would be a valid ML codeword. The number of constraints needed to define the polytope $\mathcal{Q}$ is exponential in the maximum parity-check degree of $\mathbf{H}$, i.e., $O(2^{\max_j |\mathcal{N}(j)|})$. Let a polar code $\mathcal{C}$ be constructed using the channel polarization transform of length $N = 2^m$ proposed in [42], which is denoted by a matrix $\mathbf{G_N}$, where $\mathbf{G_N} = \mathbf{B_N}\mathbf{G_2}^{\otimes m}$, $\mathbf{G_2} = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$, the operator $\otimes m$ represents the m-times Kronecker product of $\mathbf{G_2}$, and $\mathbf{B_N}$ is the bit-reversal permutation matrix defined in [42]. Assuming all the $N - k$ frozen (non-information) bits in $\mathcal{C}$ are set to 0, a parity check matrix $\mathbf{H}$ for $\mathcal{C}$ can be constructed by selecting the columns corresponding to the frozen bit indices in $\mathbf{G_N}$ as the parity checks [44]. Thus, $\mathbf{H}$ consists of high density rows with the maximum possible parity-check degree being N. Hence, the number of constraints needed to define the polytope $\mathcal{Q}$ as per (5.3) is $O(2^{N-1})$. This is clearly impractical for all but very short length polar codes. It was also shown in [44] that LP decoding on the fundamental polytope $\mathcal{Q}$ will fail for a BEC(ϵ), binary symmetric channel BSC(p) or a BAWGNC(σ), even if the polytope $\mathcal{Q}$ could be represented in practice.

A sparse factor graph representation with $O(N\log N)$ auxiliary variable nodes was proposed in [42] for the polar code $\mathcal{C}$. An example sparse factor graph for $N = 8$ is shown in Fig. 5.1. It is easy to see that there are only degree-3 or degree-2 check nodes in the sparse factor graph. Let $\mathbf{H}_{\mathcal{P}}$ denote the adjacency matrix of the sparse factor graph where the rows and columns represent the check and variable nodes, respectively. The LP polytope $\mathcal{P}$ is defined as the intersection of local minimal convex polytopes of each row (parity check) in $\mathbf{H}_{\mathcal{P}}$ and the set of cutting planes corresponding to the frozen column indices (variable nodes) in $\mathbf{H}_{\mathcal{P}}$ [44].

Using the polytope $\mathcal{P}$, an LP decoder for the polar code $\mathcal{C}$ as proposed in [44] is given by

$$\text{minimize } \boldsymbol{\gamma}^T\bar{\mathbf{x}} \quad \text{subject to } \mathbf{x} \in \mathcal{P} \subseteq [0,1]^{N(1+\log N)} \tag{5.4}$$

where $\bar{\mathbf{x}}$ is defined as the first N component subset of $\mathbf{x}$ and corresponds to the codeword variable nodes, $\bar{x}_i = x_i \ \forall \ i \in \{1, \ldots, N\}$. Similarly, the projection of polytope $\mathcal{P}$ is defined [44] as

$$\bar{\mathcal{P}} = \{\bar{\mathbf{x}} \in [0,1]^N \mid \exists \ \hat{\mathbf{x}} \ s.t. \ (\bar{\mathbf{x}}, \hat{\mathbf{x}}) \in \mathcal{P}\} \tag{5.5}$$

It was shown that if the projection $\bar{\mathbf{x}}$ (on $\bar{\mathcal{P}}$) of the LP decoder output vector $\mathbf{x}$ in (5.4) is integral then it is guaranteed to be the ML codeword i.e., LP decoding on the polytope $\mathcal{P}$ as defined in (5.4) has the ML-certificate property (Lemma 3 in [44]). It was also shown (Theorem 1 in [44]) that the projection $\bar{\mathcal{P}}$ of polytope $\mathcal{P}$ is tighter than the fundamental polytope $\mathcal{Q}$.

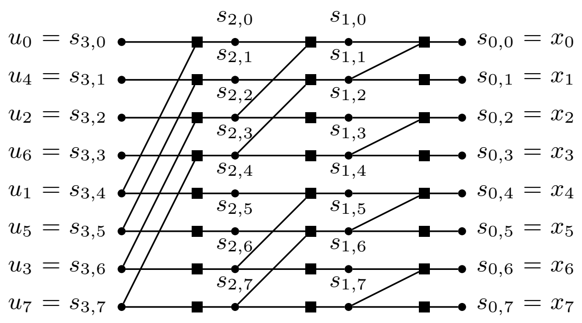


Figure 5.1: Sparse factor graph representation of a length-8 polar code.

5.3 Adaptive LP Decoding of Polar Codes

5.3.1 Adaptive LP Decoding of a Binary Linear Code

An Adaptive LP (ALP) decoder for binary linear codes solves a sequence of LP decoding problems with the addition of intelligently chosen constraints called *cuts* at every iteration [45]. A cut at a point $\mathbf{x} \in [0,1]^N$ is a violated constraint at $\mathbf{x}$ derived from a check node. An ALP decoder starts by solving the initial LP with the constraints

$$x_i \geq 0 \ \text{ if } \ \gamma_i \geq 0; \qquad x_i \leq 1 \ \text{ if } \ \gamma_i < 0 \tag{5.6}$$

The solution of this initial LP coincides with the output of a hard decision decoding of the received LLR values. The ALP decoder then searches constraints from all parity checks to find cuts, adds the cuts to the LP and solves the resulting LP. This procedure is repeated until an integer solution is obtained or no further cuts can be found. Violated constraints or cuts can also be generated using redundant parity checks (RPCs). RPCs are obtained by the modulo-2 addition of parity checks in the parity check matrix of the code. The addition of cuts from RPCs during the LP decoding iterations can only tighten the LP polytope and hence can only improve the error rate performance. In [46], efficient algorithms to perform the cut-search on parity checks and to find cut-inducing RPCs were proposed. Based on these algorithms, an adaptive cut generation based LP (ACG-ALP) decoder (Algorithm 2 in [46]) was proposed. Next, we present modifications to the ACG-ALP decoder which make it suitable for decoding polar codes.

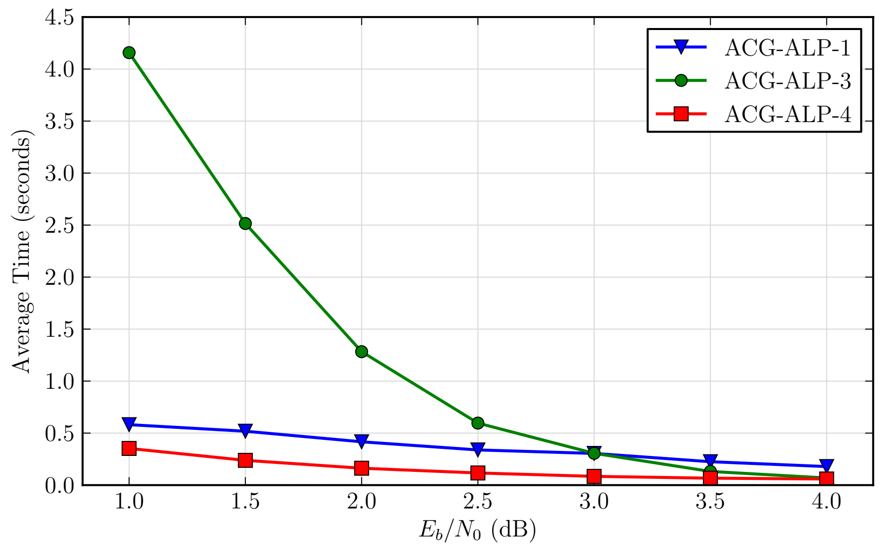


Figure 5.2: Average time for decoding one codeword of a (64, 32) polar code over a BAWGN channel with ACG-ALP decoding.

5.3.2 Modified ACG-ALP Decoder for Polar Codes

A polar code can be defined using the sparse factor graph ($\mathbf{H}_{\mathcal{P}}$) with the frozen bit information or the parity check matrix ($\mathbf{H}$). The availability of these two representations motivates the idea of modifying the ACG-ALP decoder (Algorithm 2 in [46]) to improve its performance when compared to a LP decoder. The ACG-ALP decoder uses the parity check matrix to generate constraints and cuts. RPCs and cuts from these RPCs are also derived from the parity check matrix by the ACG-ALP decoder. Based on these observations, we investigate four ways of using the sparse factor graph and the parity check matrix representations in the ACG-ALP decoder:

1. Use the unmodified ACG-ALP decoder with the parity check matrix $\mathbf{H}$.

2. Use $\mathbf{H}_{\mathcal{P}}$ as the parity check matrix in the ACG-ALP decoder. Add the frozen bit constraints to the inital-LP.

3. Initialize the ACG-ALP decoder with the polytope $\mathcal{P}$ and generate subsequent cut-inducing RPCs from $\mathbf{H}_{\mathcal{P}}$.

4. Initialize the ACG-ALP decoder with the polytope $\mathcal{P}$ and generate subsequent cut-inducing RPCs from $\mathbf{H}$.

Note that the ACG-ALP decoders 1 and 2 do not use constraints from parity checks in defining the initial LP. Hence, these decoders are expected to have a larger average decoding time complexity compared to the ACG-ALP decoders 3 and 4, as shown in Fig. 5.2. Due to its large decoding time complexity, the simulation results corresponding to the ACG-ALP decoder 2 could not be obtained. We have empirically observed that the other three ACG-ALP decoders (1, 3, 4) perform equally well in terms of the frame error rate (FER) performance but the ACG-ALP decoder 4 has the smallest average decoding time complexity. Hence, we select this modified decoder for decoding polar codes (ACG-ALP-Polar) as shown in Algorithm 6.

Algorithm 6 ACG-ALP decoding algorithm for Polar codes

```
Input: $\boldsymbol{\gamma}$, $\mathbf{H}_{\mathcal{P}}$, frozen bit indices, $\mathbf{H}$
Output: Optimal solution of the current LP problem
1: Initialize the LP problem with the constraints obtained from $\mathbf{H}_{\mathcal{P}}$ and frozen bit
   information.
2: Solve the current LP problem to get the solution $\mathbf{x}^*$.
3: if $\mathbf{x}^*$ is nonintegral then
4:     Construct cut-inducing RPC matrix $\tilde{\mathbf{H}}$ from $\mathbf{H}$ [46].
5:     Apply the cut-search algorithm (Algorithm 1 in [46]) to each row of $\tilde{\mathbf{H}}$.
6: if No cut is found then
7:     Terminate.
8: else
9:     Add the cuts found to the LP problem, go to line 2.
```

5.3.3 Simulation Results

Fig. 5.3 and Fig. 5.4 show the FER performance over a BAWGNC of rate-0.5 length-64 and length-128 polar codes, respectively. The performance of the proposed ACG-ALP-Polar decoder is compared with the previously proposed LP decoder for polar codes [44], the SC and the SC-List decoders [28]. We choose a list size = 32 for the SC-List decoder (SC-List-32) as it is known to have performance close to the ML lower bound [28]. The polar codes are constructed using the bit channel degrading merge algorithm presented in [27] optimized for a BAWGN channel with signal-to-noise ratio, SNR (E_s/N_0) = 2.0 dB. The proposed decoder uses the LP solver in the CVXOPT package [47]. A total of 200 frame errors were recorded at each E_b/N_0 point. We also

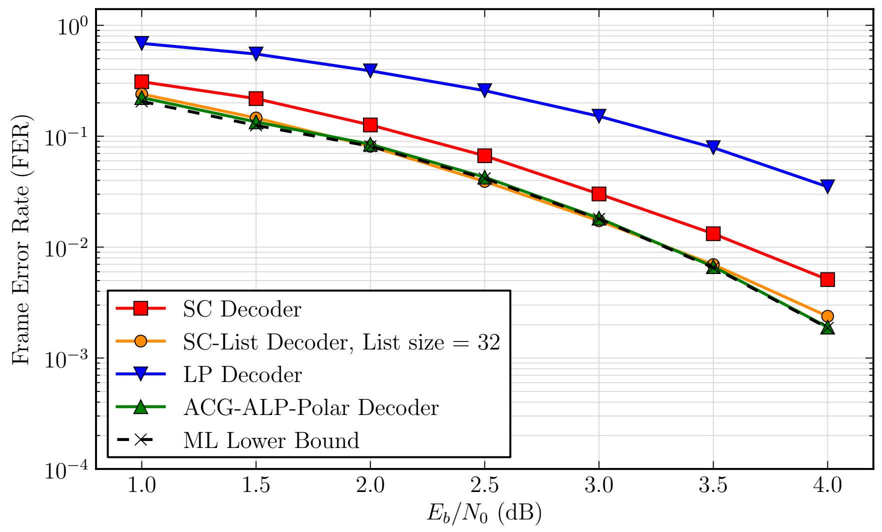


Figure 5.3: FER performance of a (64, 32) polar code over a BAWGN channel.

show an ML lower bound obtained using the ML-certificate property of the proposed decoder. The ACG-ALP-Polar decoder shows a significant improvement in performance compared to the LP decoder for both polar codes. For the length-64 polar code, the ACG-ALP-Polar decoder performance is very close to the ML lower bound. However, for the length-128 polar code, there is a performance gap in the lower SNR region. The ACG-ALP-Polar decoder performs better than the SC decoder for both polar codes studied. Compared to the SC-List-32 decoder, the ACG-ALP-Polar decoder performs equally well for the (64, 32) polar code while its performance is worse in the low SNR region for the (128, 64) polar code. Although the ACG-ALP-Polar decoder shows promise in FER performance, we have empirically observed that its decoding time complexity is larger than that of the SC-List-32 decoder especially in the low SNR region and is prohibitively high for moderate to long blocklength polar codes.

5.4 Polar Code Sparse Factor Graph Reduction

Due to its recursive structure the polar code sparse factor graph has some redundant variable nodes connected to degree-2 check nodes. We use this redundancy and the frozen bit information to propose an algorithm for reducing the number of constraints needed to represent the sparse factor graph based LP polytope $\mathcal{P}$. Simulation results show that the new reduced factor graph representation can be used in the ACG-ALP-Polar

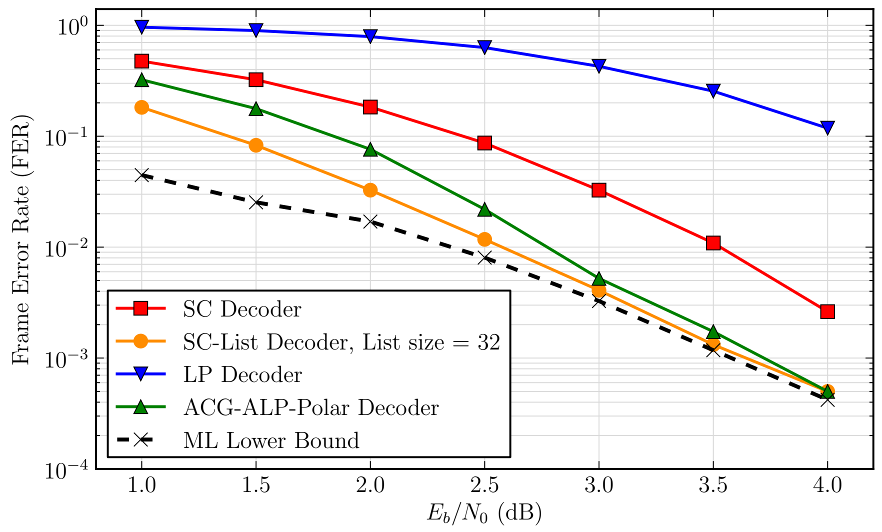
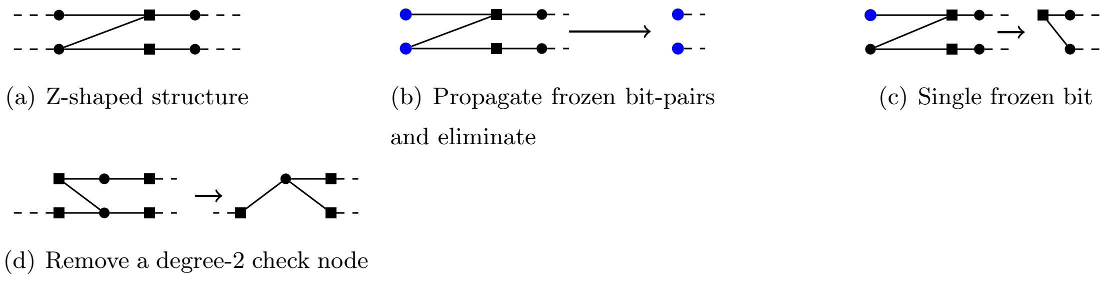


Figure 5.4: FER performance of a (128, 64) polar code over a BAWGN channel.

decoder to achieve an improvement in the time complexity.

5.4.1 Polar Code Sparse Factor Graph Reduction Algorithm

The sparse factor graph, the set of frozen bit indices and their values define the polar code. The set of frozen bit indices is obtained using a polar code construction technique [27]. We assume that all frozen bits are set to 0.



(a) Z-shaped structure
(b) Propagate frozen bit-pairs and eliminate
(c) Single frozen bit
(d) Remove a degree-2 check node

Figure 5.5: Reducing a polar code sparse factor graph.

Every pair of degree-2 and degree-3 check nodes in the sparse factor graph is interconnected through a Z-shaped structure shown in Fig. 5.5(a), referred to as a *Z-structure.* The possible configurations of frozen variable nodes in a Z-structure are:

1. Both the variable nodes on the left are frozen.

2. Only a single variable node is frozen and due to the channel polarization principle, this must be the degree-1 variable node on the left.

In case (1), we propagate the left frozen variable node values (0's in this case) to the right variable nodes of the Z-structure and eliminate the Z-structure as shown in Fig. 5.5(b). We note that this step is similar to the one proposed in the simplified successive cancellation decoder [48] used to reduce the complexity of the SC decoder. We are left with Z-structures in the graph where only a single bit is frozen i.e., the case (2). We reduce such a Z-structure by replacing it with a degree-2 check node as shown in Fig. 5.5(c). Now, there are no more frozen variable nodes in the factor graph and hence starting at the code bit variable nodes on the right, we iteratively reduce degree-2 check nodes as shown in Fig. 5.5(d).

Next, we show that for LP decoding, the polar code sparse factor graph can be reduced further by eliminating degree-1 auxiliary variable nodes and their check node neighbors from the graph.

Lemma 5.4.1. *The constraints from a parity check node which is connected to a degree-1 auxiliary variable node in the polar code sparse factor graph do not affect the LP decoder solution and hence the degree-1 auxiliary variable node and its check node neighbor can be deleted from the graph.*

Proof. From the formulation of LP decoding for polar codes ((5.4) in Section 5.2), we know that the LP decoder objective function is independent of the auxiliary variable nodes (which do not correspond to codeword bits). Hence a degree-1 auxiliary variable node is free to be assigned any feasible value by the LP solver and can be deleted from the graph. □

The steps for reducing a polar code sparse factor graph are described in Algorithm 7. Assuming u_0, u_1, u_2, u_4 (Fig. 5.1) are the frozen bits, the reduced factor graph (RFG) of a (8, 4) polar code sparse factor graph obtained using Algorithm 7 is illustrated in Fig. 5.6.

Let $\mathbf{H}_{\mathcal{R}}$ be a parity-check matrix representation of the polar code reduced factor graph. We show that $\mathbf{H}_{\mathcal{R}}$ has small degree check nodes necessary to represent the LP polytope efficiently.

Lemma 5.4.2. *A polar code reduced factor graph $\mathbf{H}_{\mathcal{R}}$ consists of only degree-3 check nodes.*

Algorithm 7 Reduce Polar Code Sparse Factor Graph

```
Input: Polar code sparse factor graph $\mathbf{H}_{\mathcal{P}}$, frozen bit indices
Output: Reduced factor graph
1: Step 1: Propagate frozen variable node pairs as shown in Fig. 5.5(b) and eliminate the corresponding Z-structures.
2: Step 2: Replace Z-structures containing a single frozen variable node with degree-2 check nodes. (Fig. 5.5(c))
3: Step 3: For each degree-2 check node, delete a variable node neighbor connecting all its neighboring check nodes to the other variable node neighbor. (Fig. 5.5(d))
4: Step 4: Iteratively delete degree-1 auxiliary variable nodes and their check node neighbors until no further degree-1 auxiliary variable nodes exist.
```

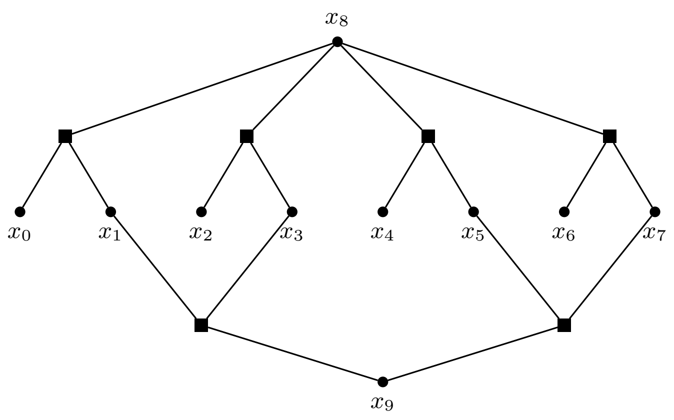


Figure 5.6: Reduced factor graph (RFG) of a (8, 4) polar code.

Proof. We refer to the steps of Algorithm 7 for this proof. A polar code sparse factor graph has only degree-2 and degree-3 check nodes. Step 1 deletes check nodes in Z-structures with two frozen bits (Fig. 5.5(b)) and hence does not change the degree of any other check nodes. Step 2 operates on Z-structures with a single frozen bit (Fig. 5.5(c)) and deletes the degree-2 check node while reducing the degree of the degree-3 check node by 1. Step 3 iteratively deletes degree-2 check nodes and does not affect the degree-3 check nodes. Hence, we are left with only degree-3 check nodes in the factor graph. Step 4 deletes the degree-1 auxiliary variable nodes and their check node neighbors. Therefore, the reduced factor graph $\mathbf{H}_{\mathcal{R}}$ consists of only degree-3 check nodes. □

Theorem 5.4.3. *Let $\mathcal{R}$ be the fundamental polytope of the reduced factor graph $\mathbf{H}_{\mathcal{R}}$ of a polar code $\mathcal{C}$. Then, $\mathcal{R} \subset [0,1]^d$, where $d = f(N,r)$ is the dimension of the vectors in $\mathcal{R}$ and is a function of the polar code blocklength N and the rate r. Let $\mathcal{P}$ be the polytope obtained from the original sparse factor graph and the frozen bit information of $\mathcal{C}$ and let $\tilde{\mathcal{P}}$ be the projected polytope obtained from the projection of vectors in $\mathcal{P}$ onto the d variables in $\mathbf{H}_{\mathcal{R}}$. Then,*

$$\mathcal{R} = \tilde{\mathcal{P}} \tag{5.7}$$

Proof. First we show that $\tilde{\mathcal{P}} \subseteq \mathcal{R}$ i.e., every vector in polytope $\tilde{\mathcal{P}}$ is also in polytope $\mathcal{R}$. Consider a vector $\mathbf{u} \in \mathcal{P}$; its projection of length d, $\tilde{\mathbf{u}} \in \tilde{\mathcal{P}}$, can be constructed by deleting the components of $\mathbf{u}$ corresponding to the variable nodes deleted in Algorithm 7. It is clear that no step in Algorithm 7 requires a change in the value of a variable node which is not deleted and hence $\tilde{\mathbf{u}} \in \mathcal{R}$. Next, we show that $\mathcal{R} \subseteq \tilde{\mathcal{P}}$. Let $\mathbf{v} \in \mathcal{R}$; then $\mathbf{v}$ satisfies all the parity checks in $\mathbf{H}_{\mathcal{R}}$. From the proof of Lemma 2, we know that there are degree-3 parity checks without frozen variable node neighbors in the original sparse factor graph which cannot be reduced. In $\mathbf{H}_{\mathcal{R}}$, even though the variable node indices participating in these checks may be different from those in $\mathbf{H}_{\mathcal{P}}$, the parity checks remain unchanged because there is one representative variable node for a group of deleted variable nodes which were constrained to take on the same values. Hence, the set of parity checks in $\mathbf{H}_{\mathcal{R}}$ is a subset of the parity checks in $\mathbf{H}_{\mathcal{P}}$ and $\mathbf{v} \in \tilde{\mathcal{P}}$. Therefore, $\mathcal{R} = \tilde{\mathcal{P}}$. □

LP decoding on the polytope $\mathcal{P}$ has the ML-certificate property [44] and from Theorem 1 it follows that the polytope $\mathcal{R}$ also has the ML-certificate property. We replace the matrix $\mathbf{H}_{\mathcal{P}}$ with $\mathbf{H}_{\mathcal{R}}$ in the ACG-ALP-Polar decoder (Algorithm 6). The size of the matrix $\mathbf{H}_{\mathcal{R}}$ is strictly smaller than that of $\mathbf{H}_{\mathcal{P}}$ for any polar code of rate < 1.

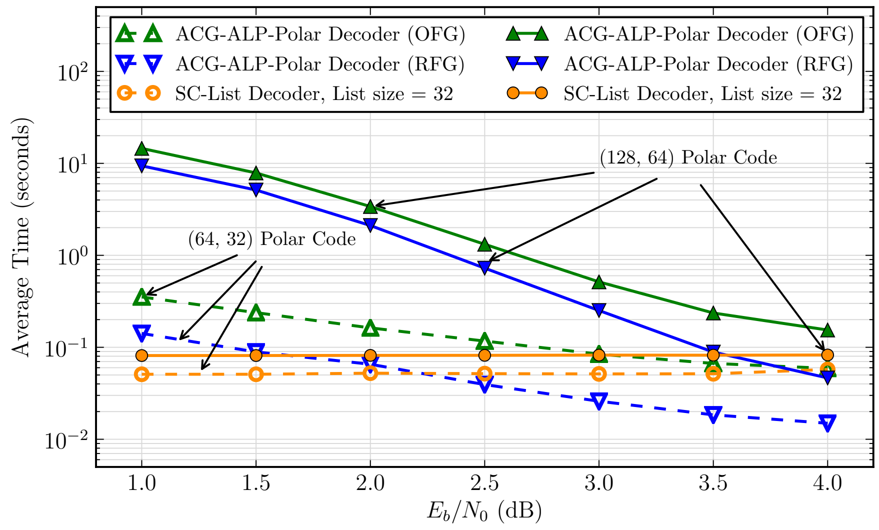


Figure 5.7: Average time for decoding one codeword of a (64, 32) and (128, 64) polar code over a BAWGN channel. OFG – with original factor graph; RFG – with reduced factor graph.

Hence the decoding time complexity of the ACG-ALP-Polar decoder can only decrease by using the reduced factor graph.

5.4.2 Simulation Results

We present simulation results using the reduced factor graph representation in the ACG-ALP-Polar decoder for the two polar codes discussed in Section 5.3.3. The FER performance is unchanged (Fig. 5.3 and Fig. 5.4) and hence is not shown. However, as Fig. 5.7 shows, the decoding time complexity is decreased when using the reduced factor graph representation. Compared to the SC-List-32 decoder, the ACG-ALP-Polar decoder with the reduced factor graph has a lower average decoding time complexity at higher SNRs. The reduction in the representation complexity of polar codes using the reduced factor graph is shown in Fig 5.8.

5.5 Conclusion

We proposed modifications to the ACG-ALP decoder [46] which make it suitable for decoding short to moderate length polar codes with FER performance close to the ML performance. This indicates that with the proper polytope representation, LP decoding works well for polar codes over a BAWGNC. We also presented an algorithm to generate

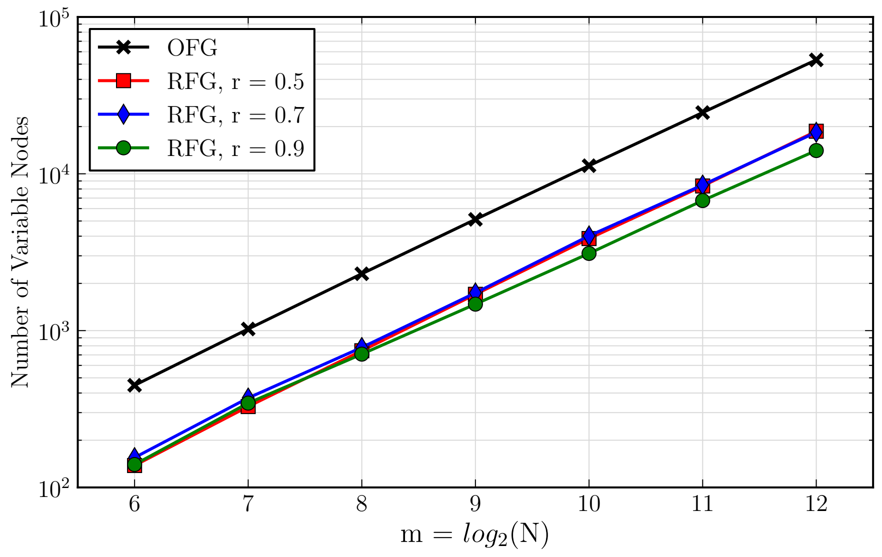


Figure 5.8: Representation complexity ($d = f(N, r)$) for polar codes using the original sparse factor graph (OFG) and the reduced factor graph (RFG).

an efficient reduced factor graph representation of a polar code. This reduced factor graph decreases the decoding time complexity of the ACG-ALP-Polar decoder without degrading its error rate performance.

Acknowledgements

This chapter is a reprint of the material in the paper: Veeresh Taranalli, and Paul H. Siegel, “Adaptive linear programming decoding of polar codes,” in Proc. IEEE International Symposium on Information Theory (ISIT), Hawaii, June-July 2014, pp. 2982–2986. The dissertation author was the primary investigator and author of this paper, and co-authors have approved the use of the material for this dissertation.

Bibliography

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Technical Journal*, vol. 27, pp. 379–423, 1948.

[2] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, "Introduction to flash memory," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, April 2003.

[3] J. Cooke, "The inconvenient truths about NAND flash memory," in *Micron MEMCON 7*, 2007.

[4] E. Yaakobi, J. Ma, L. Grupp, P. H. Siegel, S. Swanson, and J. K. Wolf, "Error characterization and coding schemes for flash memories," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM) Workshops*, December 2010, pp. 1856–1860.

[5] E. Yaakobi, L. Grupp, P. H. Siegel, S. Swanson, and J. K. Wolf, "Characterization and error-correcting codes for TLC flash memories," in *Proc. International Conference on Computing, Networking and Communications (ICNC)*, January 2012, pp. 486–491.

[6] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis," in *Design, Automation and Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 521–526.

[7] Y. Kim, B. Kumar, K. L. Cho, H. Son, J. Kim, J. J. Kong, and J. Lee, "Modulation coding for flash memories," in *Proc. International Conference on Computing, Networking and Communications (ICNC)*, January 2013, pp. 961–967.

[8] A. Berman and Y. Birk, "Error correction scheme for constrained inter-cell interference in flash memory," in *Annual Non-Volatile Memories Workshop (NVMW), 2011*, March 2011.

[9] R. Motwani, "Hierarchical constrained coding for floating-gate to floating-gate coupling mitigation in flash memory," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, December 2011.

[10] M. Qin, E. Yaakobi, and P. H. Siegel, "Constrained codes that mitigate inter-cell interference in read/write cycles for flash memories," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 836–846, May 2014.

[11] J. Solid State Technology Association, "JESD91A Method for developing acceleration models for electronic component failure mechanisms."

[12] P. H. Siegel and J. K. Wolf, "Modulation and coding for information storage," *IEEE Communications Magazine*, vol. 29, no. 12, pp. 68–86, December 1991.

[13] K. A. S. Immink, *Codes for Mass Data Storage Systems.* Shannon Foundation Publishers, 2004.

[14] P. A. Franaszek, "Run-length-limited variable-length coding with error propagation limitation," US Patent 3,689,899 (1972).

[15] J. Moon and B. Brickner, "Maximum transition run codes for data storage systems," *IEEE Transactions on Magnetics*, vol. 32, no. 5, pp. 3992–3994, September 1996.

[16] 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.

[17] X. Huang, A. Kavcic, X. Ma, G. Dong, and T. Zhang, "Multilevel flash memories: Channel modeling, capacities and optimal coding rates," *International Journal on Advances in Systems and Measurements*, vol. 6, no. 3–4, pp. 364–373, 2013.

[18] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *Proc. of the Conference on Design, Automation and Test in Europe (DATE)*, 2013, pp. 1285–1290.

[19] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, "Modelling of the threshold voltage distributions of sub-20 nm NAND flash memory," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Dec 2014, pp. 2351–2356.

[20] J. G. Skellam, "A probability distribution derived from the binomial distribution by regarding the probability of success as variable between the sets of trials," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 10, no. 2, pp. 257–261, 1948.

[21] J. K. Lindsey and P. M. E. Altham, "Analysis of the human sex ratio by using overdispersion models," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 47, no. 1, pp. 149–157, 1998.

[22] V. Taranalli, H. Uchikawa, and P. H. Siegel, "Error analysis and inter-cell interference mitigation in multi-level cell flash memories," in *Proc. IEEE International Conference on Communications (ICC)*, London, UK, June 2015, pp. 271–276.

[23] C. H. Stapper, A. N. McLarent, and M. Dreckmann, "Yield model for productivity optimization of VLSI memory chips with redundancy and partially good product," *IBM Journal of Research and Development*, vol. 24, no. 3, pp. 398–409, May 1980.

[24] F. J. Massey, "The Kolmogorov-Smirnov test for goodness of fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[25] E. Jones, T. Oliphant, P. Peterson *et al.*, “SciPy: Open source scientific tools for Python,” 2001–. [Online]. Available: http://www.scipy.org/

[26] D. M. Arnold, E. Eleftheriou, and X. Y. Hu, “Progressive edge-growth tanner graphs,” in *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 2, 2001, pp. 995–1001.

[27] I. Tal and A. Vardy, “How to construct polar codes,” *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6562–6582, October 2013.

[28] ——, “List decoding of polar codes,” *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[29] F. Alajaji and T. Fuja, “A communication channel modeled on contagion,” *IEEE Transactions on Information Theory*, vol. 40, no. 6, pp. 2035–2041, Nov 1994.

[30] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication.* Cambridge University Press, 2005.

[31] G. Dong, Y. Pan, N. Xie, C. Varanasi, and T. Zhang, “Estimating information-theoretical NAND flash memory storage capacity and its implication to memory system design space exploration,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 9, pp. 1705–1714, Sep. 2012.

[32] Y. Li, A. Kavcic, and G. Han, “On the capacity of multilevel NAND flash memory channels,” *CoRR*, vol. abs/1601.05677, 2016. [Online]. Available: http://arxiv.org/abs/1601.05677

[33] J. Wolfowitz, “Simultaneous channels,” *Archive for Rational Mechanics and Analysis*, vol. 4, no. 1, pp. 371–386, January 1959.

[34] D. Blackwell, L. Breiman, and A. J. Thomasian, “The capacity of a class of channels,” *The Annals of Mathematical Statistics*, vol. 30, no. 4, pp. 1229–1241, December 1959.

[35] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[36] A. K. Gupta and S. Nadarajah, *Handbook of Beta Distribution and Its Applications.* CRC Press, 2004.

[37] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. New York: John Wiley, 2006.

[38] D. Sutter, J. M. Renes, F. Dupuis, and R. Renner, “Achieving the capacity of any DMC using only polar codes,” in *Proc. IEEE Information Theory Workshop (ITW)*, Sep. 2012, pp. 114–118.

[39] J. Honda and H. Yamamoto, “Polar coding without alphabet extension for asymmetric models,” *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 7829–7838, Dec. 2013.

[40] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "How to achieve the capacity of asymmetric channels," *CoRR*, vol. abs/1406.7373, 2014. [Online]. Available: http://arxiv.org/abs/1406.7373

[41] E. E. Majani and H. Rumsey, Jr., "Two results on binary-input discrete memoryless channels," in *Proc. IEEE Symposium on Information Theory (ISIT)*, June 1991, p. 104.

[42] E. Arikan, "Channel Polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[43] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 954–972, March 2005.

[44] N. Goela, S. B. Korada, and M. Gastpar, "On LP decoding of polar codes," in *Proc. IEEE Information Theory Workshop (ITW)*, August 2010.

[45] M. H. Taghavi and P. H. Siegel, "Adaptive methods for linear programming decoding," *IEEE Transactions on Information Theory*, vol. 54, no. 12, pp. 5396–5410, December 2008.

[46] X. Zhang and P. H. Siegel, "Adaptive cut generation algorithm for improved linear programming decoding of binary linear codes," *IEEE Transactions on Information Theory*, vol. 58, no. 10, pp. 6581–6594, October 2012.

[47] M. S. Andersen, J. Dahl, and L. Vandenberghe, "CVXOPT: A python package for convex optimization, version 1.1.6. Available at cvxopt.org," 2013.

[48] A. Alamdar-Yazdi and F. R. Kschischang, "A simplified successive-cancellation decoder for polar codes," *IEEE Communications Letters*, vol. 15, no. 12, pp. 1378–1380, December 2011.